IDOL Eduction

Software Version 12.13

User and Programming Guide

MICRO FOCUS

Document Release Date: October 2022
Software Release Date: October 2022

Legal notices

Documentation updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for updated documentation, visit https://www.microfocus.com/support-and-services/documentation/.

Support

Visit the MySupport portal to access contact information and details about the products, services, and support that Micro Focus offers.

This portal also provides customer self-solve capabilities. It gives you a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the MySupport portal to:

- View information about all services that Support offers
- Submit and track service requests
- Contact customer support
- Search for knowledge documents of interest
- View software vulnerability alerts
- Enter into discussions with other software customers
- Download software patches
- Manage software licenses, downloads, and support contracts

Many areas of the portal require you to sign in. If you need an account, you can create one when prompted to sign in.

About this PDF version of online Help

This document is a PDF version of the online help, and is provided so you can easily print multiple topics or read the online help. Because this content was originally created to be viewed as online help in a web browser, some topics may not be formatted properly. Some interactive topics may not be present in this PDF version. Those topics can be successfully printed from within the online help.

Contents

Part I: Getting Started

This section introduces Eduction and the basic concepts for its use.

- Introduction
- Install Eduction

Chapter 1: Introduction

This section introduces Micro Focus Eduction.

About Eduction

Micro Focus IDOL Eduction identifies and extracts *entities* from text. An entity is a word, phrase, or block of information, such as a person's name, an address, a date, or a telephone number.

Eduction includes a comprehensive set of predefined entities, for many languages and geographical locations, so that you can extract names, credit card numbers, addresses, and so on. You can also extend Eduction by defining your own entities.

You can use Eduction to:

- extract entities from documents and add them to metadata fields before you add the documents to your IDOL index.

  For example, you might extract company names from your document content and tag the documents with these names. Your front-end application can then use these tags to present a list of companies to your users as search filters.

- identify personally identifiable information (PII) in your data, so that you can manage this data and conform to regulation such as the General Data Protection Regulation (GDPR).

  > **TIP:** Micro Focus provides an additional PII grammar package for this purpose. For more information, refer to the *IDOL PII Package Technical Note*.

- perform sentiment analysis. Sentiment analysis identifies positive and negative sentiment in text. For example, you can extract positive and negative comments from product reviews. See Sentiment Analysis, on page 79.

- redact sensitive information in text or IDOL documents, so that you can conform to data protection standards and use your records for multiple purposes.

Micro Focus IDOL allows you to use Eduction in several ways:

- **CFS and NiFi Ingest**. You can use Eduction to enrich documents during the ingestion process, before you add them to the IDOL index. For example, you can extract entities and tag the documents so that is easier to find documents related to a specific person, place, or subject.

To run Eduction as part of the ingestion process, use either Connector Framework Server (CFS) or IDOL NiFi Ingest. For more information, refer to the *Connector Framework Server Administration Guide*, or the *IDOL NiFi Ingest Getting Started Guide*.

- **Eduction Server**. You can use the Eduction Server to extract entities, redact information, and perform sentiment analysis on plain text. See Use Eduction Server, on page 37.
- **Build a custom application using the Eduction SDK**. Micro Focus provides Eduction SDKs for C, .NET, and Java, so that you can include Eduction in your own applications. See Deploy Eduction SDK, on page 25.

For more information about which method to use, see Decide Which Eduction Product to Use, on page 20.

Eduction Concepts

This section introduces some of the Eduction concepts and terminology used throughout this guide.

Grammar Files

A *grammar file* defines one or more entities that you want to extract.

- **Standard grammars**. Eduction includes a collection of grammar files covering common entities such as names, social security numbers, postal addresses, telephone numbers, and so on. For a complete list of standard grammars, see Standard Grammars, on page 113.

  Standard grammar files are licensed by category and by language, so that you can have a license for any combination of category (for example, sentiment, place, or person) and language.

- **User grammars**. You can extend the capabilities of Eduction by writing your own grammar files, either from scratch or by referencing existing entities.

  > **NOTE:** To reference the standard grammars in your own grammar files, you must have an appropriate license.

  You might want to extend or write a grammar if you have specialist entities or values that the standard grammars do not match. These might be new values, or you can create grammars that combine standard entities into more complicated matches.

  > **TIP:** Before you extend a grammar, raise the issue with your Micro Focus support contact. The entity that you want to detect might be supported in an upcoming release, or Micro

> Focus might be able to add support in future. Using an official grammar means you do not have to maintain it.

Grammar files are created in XML format, and can be compiled into the proprietary *ECR* format. Compiling a grammar file into the ECR format makes it much faster to load at runtime.

Most of the standard grammar files are available only in ECR format. However, the Eduction package also includes several XML source grammars to allow you to easily extend the standard grammars (see Standard Grammar – Source, on page 291). You can compile these, and your custom user grammars by using the edktool command-line tool.

> **NOTE:** In Eduction version 12.7 and later, the standard grammars are in compressed ECR format, and edktool compiles grammars to compressed ECR format. You can still use existing uncompressed grammars from previous versions.

> **NOTE:** Eduction can also use XML grammar files directly (that is, without compiling them to ECR files). However, in most cases Micro Focus recommends that you compile your grammars to improve performance.

There are two main ways to define entities:

- Use a dictionary of possible matches, for example to extract names of people or places.
- Use *regular expressions* (regex) to specify what a match looks like without having to list each possibility, for example to extract dates and times, or telephone numbers, which conform to a known pattern.

You can define entities recursively, and rules can refer to entities in other grammar files. This allows you to create more complicated entities that match data such as URLs or postal addresses.

Related Topics

- Extend Grammars, on page 86
- Custom Grammar Guidelines, on page 68
- Compile and Test Grammars, on page 92
- Standard Grammars, on page 113

Sentiment Analysis

Eduction *Sentiment Analysis* allows you to find whether text has positive, negative, or neutral sentiment. For example, you can use it to determine whether users of a particular product or service are satisfied or not, based on an automated analysis of reviews.

The sentiment analysis grammar files contain dictionaries of types of words (such as positive adjectives, negative nouns, and so on), and patterns that describe how to combine these dictionaries into positive and negative phrases.

Eduction has sentiment analysis grammar files available in the following languages:

- Arabic
- Chinese
- Czech
- Dutch
- English
- French
- German
- Italian
- Portuguese
- Russian
- Spanish
- Turkish

For more information, see Sentiment Analysis, on page 79.

Components

Some of the standard grammar files contain *components*, which extract attributes from matched phrases, such as topic, subject, and positive or negative sentiments. These attributes are called components because they are the components of a single match.

For example, if you use sentiment analysis to match the phrase *Their service is fantastic* as conveying positive sentiment, you can then use components to identify *service* as the subject matter, and *fantastic* as the adjective that describes the subject (note that the sentiment is not necessarily an adjective in all cases).

You can also set up components when you write your own custom grammar files.

For more information on how to configure and define components in your grammar files, and when to use them, see Components, on page 67.

Extraction

Eduction *extraction* is the process of matching and retrieving entities from text, according to the rules in your grammars.

To run Eduction, you send a text file, or the raw text, to the Eduction engine (you can do this in several ways, see Eduction Architecture, on the next page). You use the Eduction configuration to specify the grammars and entities that you want to match. Eduction identifies each instance of the requested entity, and returns an XML list of matches.

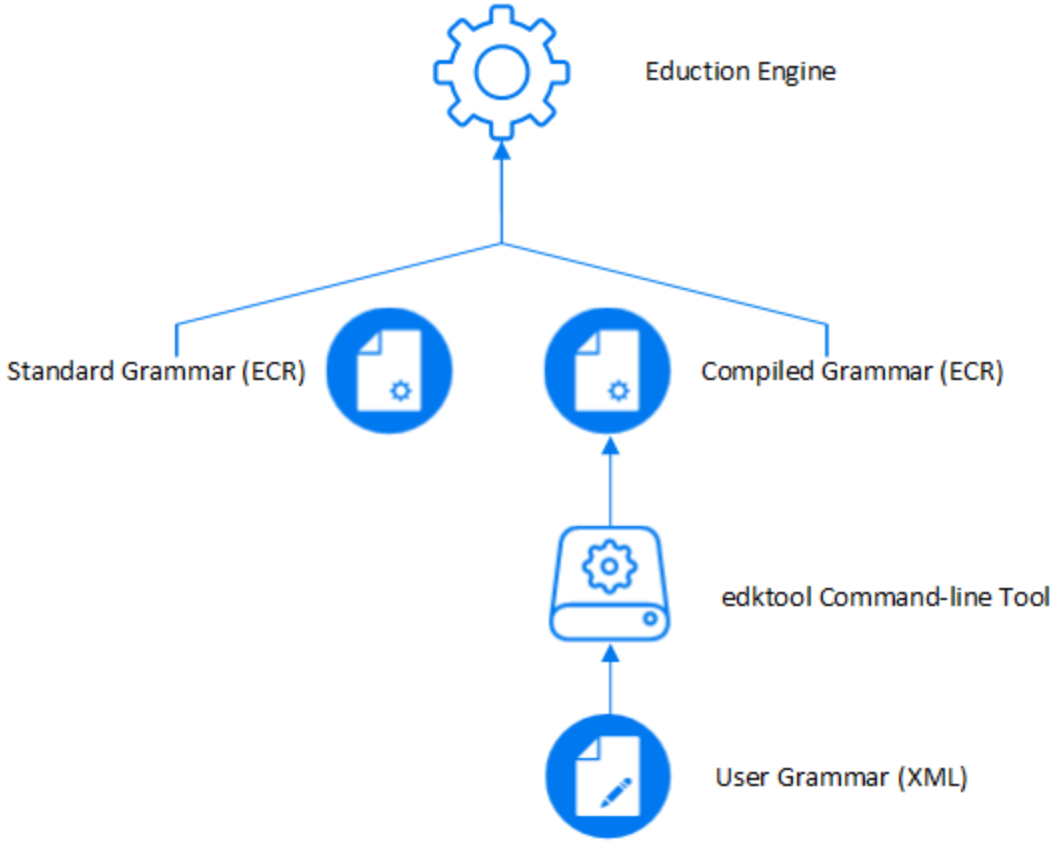
By default, Eduction returns the matched entity, with some additional information about the match, such as a confidence score for the accuracy of the match. It can also identify any configured

components of the entity match, such as the parts of a social security number or phone number (see Components, on the previous page).

Eduction Architecture

You run Eduction by using an *Eduction engine*. This engine uses the Eduction grammars to process text and return the matched entities.

The following diagram shows the use of grammars in an Eduction engine.



The standard grammar files are a wide range of ECR files that Micro Focus provides in your Eduction installation (see Standard Grammar – Compiled, on page 114). You can use these files as they are, or extend them by creating a grammar XML file that includes them. You can also create your own user grammar files from scratch.

You compile XML grammar files into ECR by using the edktool command-line tool. For more information see Compile and Test Grammars, on page 92.

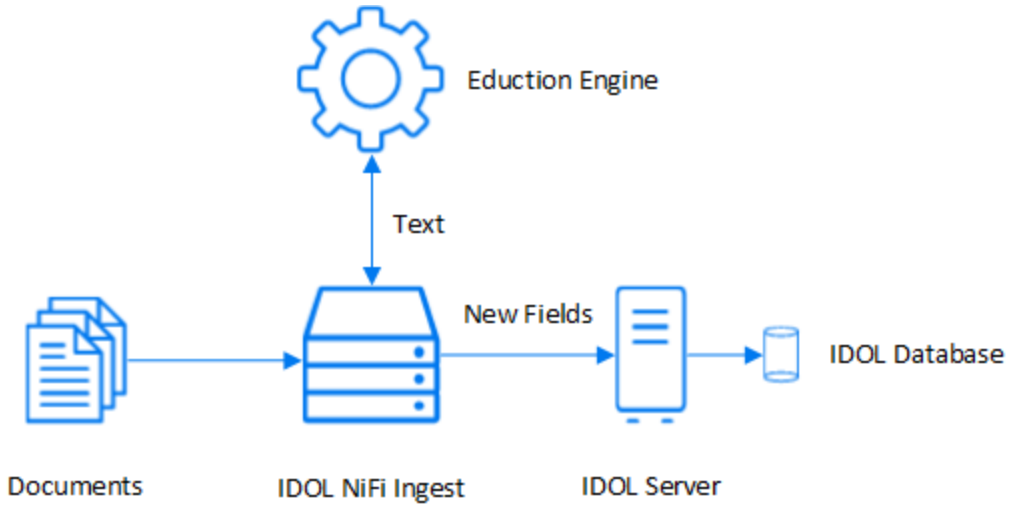
> **NOTE:** Eduction can also use XML grammar files directly (that is, without compiling them to ECR files). However, in most cases Micro Focus recommends that you compile your grammars to improve performance.

How you use the Eduction engine depends on the way you call Eduction. For example, you can:

- Use Eduction as part of an IDOL ingestion process.

  In this case, you use IDOL Connectors to retrieve documents from your repositories and send them to IDOL NiFi Ingest. IDOL NiFi Ingest performs any processing on the documents, including sending text to the Eduction engine. The Eduction engine sends back the entity matches, which IDOL NiFi can add to new fields in your IDOL documents. For more information, refer to the *IDOL NiFi Ingest Getting Started Guide*.

  The following diagram shows this process.



  > **TIP:** This process is similar if you use Connector Framework Server (CFS) to ingest your documents.

- Use Eduction Server to run Eduction.

  In this case, you send your text in ACI actions (from a front-end application or Web browser) to the Eduction Server, which runs the Eduction engine that processes the text and returns the matched entities. See Use Eduction Server, on page 37.

- Call Eduction directly by using the Eduction SDK.

  In this case, your custom application sends text directly to the Eduction engine, which then returns the matched entities. For more information, see Deploy Eduction SDK, on page 25.

You can also run Eduction by using the edktool command-line tool. This method can be useful for testing your grammars or entities when you make modifications (see Compile and Test Grammars, on page 92). However, Micro Focus recommends that you do not use edktool as part of a production system.

> **TIP:** If you do not know which Eduction package is best for your use case, see Decide Which Eduction Product to Use, on the next page.

Decide Which Eduction Product to Use

Micro Focus provides three main ways for you to use Eduction: Eduction in ingest, the Eduction SDK, and Eduction Server. The one you use depends on your use case, and your preference.

Eduction as Part of an Ingestion Process

The IDOL ingestion components, IDOL NiFi Ingest and Connector Framework Server (CFS), allow you to incorporate Eduction as part of a document retrieval process. You can use connectors to retrieve documents from your repositories, and perform Eduction alongside any other document processing.

This method is very useful if you want to automate the process of retrieving and tagging documents. In particular, if you use IDOL NiFi Ingest and CFS to index into IDOL, you can use Eduction to add extra fields to your IDOL documents to make it easier to search for the entity values that you extract.

However, this method is not appropriate if you want to provide text directly to Eduction as part of an application.

Eduction SDK or Eduction Server

The Eduction SDK provides APIs to allow you to run Eduction directly. This option is most suitable for OEM environments, where you want to embed Eduction into an application that you distribute to your users.

The Eduction ACI Server also allows you to use Eduction as part of an application. In this case, you must host the server (and a license server), which makes it less suitable for OEM environments. It might be the most suitable option if you want to use Eduction in a web application, particularly if you want to use other IDOL services.

In other cases, you can use either option, depending on your personal preference. When you choose, you might want to consider the following points:

- The Eduction SDK has a larger initial learning requirement as you start using the SDK.

  Eduction Server accepts HTTP requests and returns XML, so it might be a quicker method to get started with. In particular, if you already use the ACI API in other applications, you do not need to learn how to use additional APIs to run Eduction.

- The Eduction SDK is available only for C, .NET and Java. If you want to create an application in a different language, you might not be able to use the SDK without using additional methods to call out to shared libraries.

  > **NOTE:** To use Eduction Server, you can use any method for making HTTP requests and parsing XML. There are also IDOL SDKs available in C, .NET, and Java.

- To run Eduction, both Eduction Server and the Eduction SDK must have access to the required grammar files.

  In the case of the Eduction SDK, the easiest way to include the grammars is to install them with your applications. You can also embed the grammars in your application. Installing or embedding the grammars increases the size of your application.

  For Eduction Server, you include the grammars with the web server, so this does not add any overhead for your end users.

For more information, see Eduction SDK, on page 24 and Eduction ACI Server, on page 36.

Chapter 2: Install Eduction

This section describes the Eduction packages and components available.

Eduction Packages

Micro Focus provides the following Eduction ZIP packages:

- Eduction SDK. The Eduction SDK, which you can use to create applications that use an Eduction engine directly. This package includes:
  - Libraries and API reference documentation for the C, .NET, and Java APIs.
  - The edktool command-line tool and example configuration file, which you can use to compile and test your custom grammars. See Compile and Test Grammars, on page 92.
- Eduction ACI Server. The Eduction ACI Server, which allows you to run Eduction by using ACI actions from a Web browser or front-end application.
- Eduction Grammars. This package contains Eduction resource files (grammar files and associated post-processing scripts, pre-filter configuration files, and pre-filter dictionaries). The grammar files that you can use depend on your license.
  - PII grammars find Personally Identifiable Information (PII) in your data, so that you can manage this data and conform to regulations such as the General Data Protection Regulation (GDPR). For more information, refer to the *IDOL PII Package Technical Note*.
  - PHI grammars find Protected Healthcare Information (PHI) in your data, to ensure compliance with regulations such as the *Standards of Privacy of Individually Identifiable Health Information* implemented as part of the Health Insurance Portability and Accountability Act (HIPAA). For more information, refer to the *IDOL PHI Package Technical Note*.
  - PCI grammars find Payment Card Industry (PCI) information in your data, to ensure compliance with financial regulations. For more information, refer to the *IDOL PCI Package Technical Note*.
  - GOV grammars find governmental document markings and other information in your data, to help you comply with data management restrictions. For more information, refer to the *IDOL Government Eduction Package Technical Note*.
  - General grammars. Standard Eduction grammar files, including some XML source files for you to use to extend the grammars. See Standard Grammars, on page 113.

Eduction is also included as part of the Micro Focus IDOL ingestion components, Connector Framework Server, and IDOL NiFi Ingest. For more information, refer to the *Connector Framework Server Administration Guide*, and the *IDOL NiFi Ingest Getting Started Guide*.

> **TIP:** If you do not know which Eduction product is best for your use case, see Decide Which Eduction Product to Use, on page 20.

Licenses

To run Eduction, you must have a valid license, which you obtain from Micro Focus Support.

Standard grammar files are licensed by category and by language, so that you can have a license for any combination of category (for example, sentiment, place, or person) and language. The IDOL PII Package also has an additional license to allow you to use the PII grammar files.

The Eduction SDK requires an OEM license key (`licensekey.dat`) file, which you provide to the API.

Eduction Server uses an IDOL license, which you access by using the IDOL License Server. See Use Eduction Server, on page 37.

IDOL NiFi Ingest and CFS also require an IDOL license (and License Server) to run Eduction.

Part II: Eduction SDK

This section describes how to set up and use the Eduction SDK.

- Deploy Eduction SDK
- API Reference

Chapter 3: Deploy Eduction SDK

This chapter describes the files in the Eduction SDK and how to deploy the SDK.

Eduction SDK Package

The Eduction SDK package includes:

- Standard collection of grammar files covering a range of commonly used entities
- edktool command-line tool used for compiling Eduction XML source grammar files into compiled run-time ECR files
- C API
- .NET API
- Java API

The Eduction SDK includes reference documentation for the C, .NET, and Java APIs. To view the documentation, open `c_api/help/index.html`, `dotnet_api/help/index.html`, or `java_api/help/index.html` in a web browser.

Additional Requirements

The Eduction SDK might require additional runtime libraries on your operating system.

Windows Libraries

To run Eduction version 12.13 on the Microsoft Windows operating system, you might need to install Microsoft Visual C++ Redistributable packages. The Eduction SDK and Eduction Server stand-alone zip packages include the required redistributable files for Microsoft Visual C++ 2017. You can also update your packages by using the latest version at: http://support.microsoft.com/kb/2019667.

UNIX Libraries

To run Eduction version 12.13 on UNIX platforms, the server must have the following minimum versions of libraries:

- GLIBC_2.3.2
- GLIBCXX_3.4.21
- GCC_4.8.0

> **NOTE:** The Eduction SDK and Eduction Server stand-alone zip packages provide these libraries in the `libgcc_s` and `libstdc++` shared libraries.
>
> You might need to set the `LD_LIBRARY_PATH` to include the *`InstallDir`*`/bin` directory, to ensure that Eduction can access the installed shared libraries.
>
> You can also copy the shared libraries to the component working directory.

Compilers

You must have an appropriate compiler to compile code against the SDK. For example:

- gcc and make for the C API.
- openjdk and ant for the Java API.

The .NET SDK has the following compiler requirements:

- .NET CORE SDK 1.0+ or .NET Framework Dev Package 4.5+ (requires .NET Standard 1.1+).

  > **DEPRECATED:** Eduction SDK support for .NET Standard 1.1 has been deprecated and might be removed in future. Micro Focus recommends using a .NET implementation that supports .NET Standard 2.0.

C API Component

The C API component of the Eduction SDK includes:

- The Eduction header file (`edk.h`).
- The Eduction SDK library on Windows (`edk.dll`) or shared object on UNIX (`libedk.so`).
- (Windows only) The Eduction SDK library linker file (`edk.lib`).
- Several sample C programs that demonstrate various SDK features.

The C sample programs are provided with a `CMakeLists.txt` file, to allow you to generate build files for different environments (such as Visual Studio 2017 or UNIX Makefile). For more information about how to build the sample programs, refer to the `README.md` file in the samples directory.

> **NOTE:** You might also need additional runtime libraries to run the Eduction SDK. See Eduction SDK Package, on the previous page.

To use the Eduction SDK in C, include the `edk.h` header file from your C source code and link with the SDK library. For details of how to compile and link against the SDK, refer to the sample programs.

On Windows, you must specify the Eduction library in the `PATH` environment variable. On UNIX, the shared object must be in the library search path.

.NET API Component

The .NET API component of the Eduction SDK is available on Microsoft Windows and Linux platforms. It includes:

- The Eduction .NET library (`EductionDotNet.dll`) and the Eduction C SDK library (`edk.dll` on Windows, or `libedk.so` on Linux).
- Several .NET code samples that you can compile and execute by using the Eduction SDK.

The .NET Eduction SDK package includes versions of the `EductionDotNet.dll` that target .NET Standard 1.1 (netstandard 1.1) and .NET Standard 2.0 (netstandard 2.0).

> **DEPRECATED:** Eduction SDK support for .NET Standard 1.1 has been deprecated and might be removed in future. Micro Focus recommends using a .NET implementation that supports .NET Standard 2.0.

For a list of compatible .NET implementations, refer to the Microsoft documentation: https://docs.microsoft.com/en-us/dotnet/standard/net-standard.

You must specify the `EductionDotNet.dll`, and `edk.dll` or `libedk.so` libraries in the `PATH` environment variable.

Java API Component

The Java API component of the Eduction SDK includes:

- The Eduction library on Windows (`edkjni.dll`) or shared object on UNIX (`libedkjni.so`).

> **NOTE:** You might also need additional runtime libraries to run the Eduction SDK. See Eduction SDK Package, on page 25.

- The Java JAR file (`edk.jar`).
- Java code samples that you can compile and execute by using the Eduction SDK.

  The Java sample programs are provided with a Maven `pom.xml` file for building. For information on how to compile and run the sample files, refer to the `README.md` file in the samples directory.

Chapter 4: API Reference

This section describes the C, .NET, and Java APIs for Eduction SDK.

C API Concepts

This section describes concepts required to implement C language applications for Eduction SDK.

Include Files and Naming Conventions

The include file `edk.h` contains the core Eduction API. The types, functions, and macros specific to Eduction SDK are prefixed with the string `Edk`.

Standalone API Usage

The Eduction Software Development Kit (SDK) C API allows C developers to interact directly with the Eduction engine.

The recommended way to create an engine is by using an engine factory. Your application's first call to the Eduction API should be to create a factory with `EdkFactoryCreateWithLicenseKey`, which requires you to supply a valid license key.

There are several ways of creating an engine. You can create an empty engine, by calling `EdkFactoryMakeEngine`. Alternatively you can create a pre-configured engine, by calling `EdkFactoryMakeEngineFromConfigFile` (to use a configuration file on disk) or `EdkFactoryMakeEngineFromConfigBuffer` (to use a configuration that exists in memory).

If you create an engine from a configuration file, no further configuration is necessary, because it will reflect the settings given in that file. Otherwise, you should configure the engine to set its matching behavior. One or more grammar files must be loaded into the engine. At least one entity exposed by the loaded grammars must be specified, to tell the engine what patterns to search for in the input text.

When no more engines are required, the factory can be disposed of using `EdkFactoryDestroy`. This releases the memory used internally by the factory. Engines remain valid even after the factory that created them has been destroyed.

The input data is processed in an Eduction session. Multiple sessions can be created for an Eduction engine. All sessions associated with an engine process data using the configuration for that engine, including the selected grammars and entities. You cannot change the engine settings after creating a

session. Each session maintains its own state, so the sessions can be run concurrently in a multi-threaded application.

Once created, a session can process multiple documents. Data must be UTF-8 encoded. It can either be pulled (streamed) or pushed (added). A function is called to get the next available match. This can be called repeatedly to cycle through all the matches. The text and properties associated with each match can be retrieved using accessor function calls.

A session can continue for as long as necessary. It must, however, be destroyed before the engine it is associated with is destroyed. The call to destroy the engine should be your application's last call to the Eduction API.

This section describes the basic structure of a stand-alone application using the API. For an example of this process, see the source code in the example files (see Example Programs, on page 35).

Typically, your application takes the following actions:

1. Include `edk.h`.

```
#include "edk.h"
```

2. Create an engine factory.

```
// Embed the license key into the application
const char *licensekey = "...";
EdkFactoryHandle factory = NULL;
EdkError error = EdkFactoryCreateWithLicenseKey(&factory, licensekey);

// Check the return value of any API call for success or error
if (EdkSuccess != error)
{
    printf("Error while creating: %d.\n", error);
    return -1;
}
```

3. Instantiate the engine and obtain an engine handle. You can call `EdkFactoryMakeEngineFromConfigFile` to create an engine from an appropriate configuration file.

```
EdkEngineHandle engine = NULL;
error = EdkFactoryMakeEngineFromConfigFile(factory, &engine, "engine.cfg");
```

Alternatively, you can create the engine without a configuration, by calling `EdkFactoryMakeEngine`. In this case, you must configure the engine. For example:

- load the grammar files to use for matching (by calling `EdkLoadResourceFile` one or more times).
- choose the entities to use for matching (by calling `EdkAddTargetEntity`).
- set optional parameters.

```
// Create engine without configuration
EdkEngineHandle engine = NULL;
error = EdkFactoryMakeEngine(factory, &engine);

// Load grammar file
error = EdkLoadResourceFile(engine, "test.ecr");

// Choose entities to use
error = EdkAddTargetEntity(engine, "myentity");

// Set optional parameters
error = EdkSetTokenWithPunctuation(engine, true);
error = EdkSetMaxMatchLength(engine, 12);
```

4. Create a session associated with the engine, and obtain a session handle. You can create and run concurrent sessions in a multi-threaded application. Each session uses the same grammars, but maintains its own state.

```
EdkSessionHandle session = NULL;
error = EdkSessionCreate(engine, &session);
```

5. Send UTF-8 encoded text to the session.

```
struct stat infoText;
stat("test.txt", &infoText);
off_t lenText = infoText.st_size;
FILE* fText = fopen("test.txt", "rb");
char* buffer = (char*)malloc(lenText+1);
size_t sizeText = fread(buffer, 1, lenText, fText);
error = EdkAddInputText(session, buffer, sizeText, true);
```

6. Call `EdkGetNextMatch` to obtain an entity match. You can call this method repeatedly to obtain all matches.

```
while(EdkSuccess == EdkGetNextMatch(session))
{
    // For each match found, do this ...
    const char* szMatch = NULL;
    EdkGetMatchText(session, &szMatch);
    printf("Match found: %s\n", szMatch);
}
```

NOTE: If you create your engine from a configuration file that includes post-processing tasks, the post-processing tasks automatically run as part of `EdkGetNextMatch` and you do not need to run them separately.

7. To process multiple documents, repeat Step 4 to Step 6.
8. Release resources when done. You must destroy all session handles before you destroy the engine handle.

.NET API Concepts

The Eduction SDK provides a .NET API that enables your application to create an extraction engine and perform entity extractions.

This section describes the concepts used to write .NET applications with the Eduction EDK.

The .NET SDK consists of:

- `EductionDotNet.dll`, which contains the Eduction .NET class library.
- `edk.dll` (Windows) or `libedk.so` (Linux), which performs the Eduction functionality.

NOTE: You might also need additional runtime libraries to run the Eduction SDK. See Eduction SDK Package, on page 25.

Concurrency Control

Concurrency in Eduction is handled using *sessions*, represented by an `ITextExtractionSession` object.

You initialize an instance of an `ITextExtractionEngine` object with a configuration file that describes the grammars and settings that you want to use for entity extraction. You can create multiple `ITextExtractionSession` objects from this engine, each of which use the same grammars and settings as the parent engine. Each session maintains its state independent of others.

Character Encoding

The underlying `edk.dll` and grammars assume that all your input is UTF-8 encoded. The Eduction .NET SDK functions that accept `System.string` automatically handle conversion from UTF-16 to UTF-8. However, functions that accept a `System.IO.Stream` (for example `Eduction.ITextExtractionSession.SetInputStream`) require the byte data in the stream to be UTF-8.

Some of the available metadata that the SDK returns represent byte counts or offsets. These values are correct for the UTF-8 representation of the matched texts. Character counts and offsets are independent of the encoding.

Standalone API Usage

This section describes the basic structure of a standalone application using the API. See the source referenced in Example Programs, on page 35. Typically, your application takes the following actions:

1. Create an `EDKFactory` instance and use `GetTextExtractionEngine` to construct an `ITextExtractionEngine`.
2. Use the engine `GetSession()` method to create an `ITextExtractionSession`.
3. Add UTF-8 encoded data to the session, for example by using `SetInputStream` or `AddInputText`.
4. Iterate over the session to obtain `IExtractionMatch` results.

Java API Concepts

Eduction SDK provides a Java API that enables your application to create an extraction engine and perform entity extractions.

This section describes the concepts used to write Java applications with the Eduction SDK.

The Java SDK consists of:

- a JAR file, `edk.jar`, which contains the Eduction Java class library and the interface to the Eduction Java Native Interface (JNI).
- a DLL (Windows) or shared object (UNIX/Linux), `edkjni.dll` or `edkjni.so`, which implements the Eduction JNI library and performs the Eduction functionality.

Java developers can use either the Eduction JNI, the class library, or both. The JNI provides functionality almost identical to that of the Eduction C API. The class library encapsulates related JNI methods, implements exception handling, and provides return values from method calls that simplify application programming.

NOTE: You might also need additional runtime libraries to run the Eduction SDK. See Eduction SDK Package, on page 25.

Naming Conventions

The main JNI class that provides access to native functionality is `EDKJNI`. Support classes for the JNI are prefixed with `EDKJNI`, for example `EDKJNIVersion`.

The Eduction class library classes are prefixed with `EDK`, for example `EDKEngine`.

Concurrency Control

Concurrency in Eduction is handled using *sessions*, represented by an `EDKSession` object.

You initialize an instance of an `EDKEngine` object with corresponding grammars for entity extraction. You can associate each such engine with one or more sessions. All the sessions in the engine share the same grammars. You must configure the engine fully before you create any sessions.

After you create a session, the engine throws an exception if you try to change the engine settings. However, each individual session can process many documents or streams. Each session maintains its state independent of others.

Standalone API Usage

This section describes the basic structure of a standalone application using the API. See the source referenced in Example Programs, on page 35. Typically, your application takes the following actions:

1. Load the EDK library.

```
System.loadLibrary("edkjni");
```

2. Create an `EDKFactory` instance of a `TextExtractionFactory`, supplying a valid license key.

```
try ( TextExtractionFactory<EDKMatch> factory = EDKFactory.fromLicenseKey("...") )
{
    // use try-with-resources to automatically cleanup factory once finished or if
an exception occurs
    // rest of the processing code goes here ...
}
```

3. Use the factory to create an instance of a `TextExtractionEngine`.

```
try ( TextExtractionEngine<EDKMatch> engine = factory.createEngineFromConfigFile
(Paths.get("path", "to", "config.cfg")) ) {
    // use try-with-resources to automatically cleanup engine once finished or if
an exception occurs
    // rest of the processing code goes here ...
}
```

You can specify the options, grammar files, and entities to use in the configuration file. Alternatively the SDK provides functions for setting options programmatically (see the `FromSettings.java` example).

4. Decide whether to push data to the engine when it becomes available, or have the engine read from a stream when it needs more data. The data must be UTF-8 encoded. The following example demonstrates how to read from a stream. The stream is passed to an `EDKJNIInputStream` object which reads the stream when required.

```
try ( FileInputStream in = new FileInputStream("input.txt") ) {
    ReadableInputStream stream = new EDKJNIInputStream(in);
    // ...
}
```

Pass the `EDKJNIInputStream` to a `TextExtractionSession` object, which maintains the state of the matching process. If your application is multi-threaded then each thread should use its own `TextExtractionSession`.

```
try ( TextExtractionSession<EDKMatch> session = engine.createSession() ) {
    session.setInputStream(stream);
    // ...
}
```

5. Begin matching.

```
for (EDKMatch match : session) {
    System.out.println(match.getText());
}
```

> **NOTE:** If you create your Eduction engine from a configuration file that includes post-processing tasks, the post-processing tasks automatically run as part of `EDKMatch` and you do not need to run them separately.

6. Release resources when done.

Example Programs

You can find the sample programs in your Eduction SDK zip package, in the `samples` directory.

The following sample programs are available:

- `compile`. Compiles an XML grammar file into an ECR file.
- `eduction_from_config`. Extracts entities from a plain text input file, using settings read from a configuration file.
- `eduction_from_settings`. (C and Java only). Extracts entities from a plain text input file, using settings created by calling explicit API functions.
- `redaction`. Redacts entities in a plain text input file.
- `repeated_matches`. Demonstrates the use of `EdkGetNextRepeatedMatch`, which you can use to find repeated matches.
- `table_extraction`. Extract entities from a CSV/TSV input file, using column/header settings read from a configuration file.

For more information about these examples, refer to the `README.md` file in the `samples` directory.

Part III: Eduction ACI Server

This section describes how to set up and use Eduction Server.

- Use Eduction Server
- Run Eduction in Eduction Server

Chapter 5: Use Eduction Server

This section describes the Eduction ACI Server and how to configure and send actions.

Introduction to Eduction Server

The Eduction ACI Server is a stand-alone server that uses the IDOL ACI infrastructure. It is one of the available options that you can use to run Eduction.

> **NOTE:** Eduction Server is not included in the Eduction SDK package. You must download the Eduction Server package instead. See Eduction Packages, on page 22.

With an ACI server, you can make Eduction requests from a web browser or ACI client. Browsers can make requests to process small amounts of text by using an HTTP GET request, or larger amounts by using an HTTP POST request.

The Eduction Server processes UTF-8 encoded text, matching upon entities defined in Eduction grammars. It returns results as XML, with tags in the ACI hierarchy.

You use the ACI configuration file (`eductionserver.cfg` by default) to configure the grammars to load, and the entities to match on. Each time you send a request, the server creates a new Eduction engine with the configured grammars and entities.

You can also override the configuration settings for individual actions. See Select Entities at Runtime, on page 54.

Licenses

To use IDOL solutions, you must have a running License Server, and a valid license key file for the products that you want to use. Contact Micro Focus Big Data Support to request a license file for your installation.

License Server controls the IDOL licenses, and assigns them to running components. License Server must run on a machine with a static, known IP address, MAC address, or host name. The license key file is tied to the IP address and ACI port of your License Server and cannot be transferred between

machines. For more information about installing License Server and managing licenses, see the *License Server Administration Guide*.

When you start Eduction, it requests a license from the configured License Server. You must configure the host and port of your License Server in the Eduction configuration file.

You can revoke the license from a product at any time, for example, if you want to change the client IP address or reallocate the license.

CAUTION: Taking any of the following actions causes the licensed module to become inoperable.

You **must not**:

- Change the IP address of the machine on which a licensed module runs (if you use an IP address to lock your license).
- Change the service port of a module without first revoking the license.
- Replace the network card of a client without first revoking the license.
- Remove the contents of the `license` and `uid` directories.

All modules produce a `license.log` and a `service.log` file. If a product fails to start, check the contents of these files for common license errors. See Troubleshoot License Errors, on page 40.

Display License Information

You can verify which modules you have licensed either by using the IDOL Admin interface, or by sending the `LicenseInfo` action from a web browser.

To display license information in IDOL Admin

- In the **Control** menu of the IDOL Admin interface for your License Server, click **Licenses**.

  The **Summary** tab displays summary information for each licensed component, including:

  - The component name.
  - The number of seats that the component is using.
  - The total number of available seats for the component.
  - (Content component only) The number of documents that are currently used across all instances of the component.
  - (Content component only) The maximum number of documents that you can have across all instances of the component.

  The **Seats** tab displays details of individual licensed seats, and allows you to revoke licenses.

To display license information by sending the LicenseInfo action

- Send the following action from a web browser to the running License Server.

`http://LicenseServerHost:Port/action=LicenseInfo`

where:

LicenseServerHost	is the IP address of the machine where License Server resides.
Port	is the ACI port of License Server (specified by the `Port` parameter in the `[Server]` section of the License Server configuration file).

In response, License Server returns the requested license information. This example describes a license to run four instances of IDOL Server.

```
<?xml version="1.0" encoding="UTF-8" ?>
<autnresponse xmlns:autn="http://schemas.autonomy.com/aci/">
  <action>LICENSEINFO</action>
  <response>SUCCESS</response>
  <responsedata>
    <LicenseDiSH>
      <LICENSEINFO>
        <autn:Product>
          <autn:ProductType>IDOLSERVER</autn:ProductType>
          <autn:TotalSeats>4</autn:TotalSeats>
          <autn:SeatsInUse>0</autn:SeatsInUse>
        </autn:Product>
      </LICENSEINFO>
    </LicenseDiSH>
  </responsedata>
</autnresponse>
```

Configure the License Server Host and Port

Eduction is licensed through License Server. In the Eduction configuration file, specify the information required to connect to the License Server.

To specify the license server host and port

1. Open your configuration file in a text editor.
2. In the `[License]` section, modify the following parameters to point to your License Server.

`LicenseServerHost`	The host name or IP address of your License Server.
`LicenseServerACIPort`	The ACI port of your License Server.

   For example:

   ```
   [License]
   LicenseServerHost=licenses
   LicenseServerACIPort=20000
   ```

3. Save and close the configuration file.

Revoke a Client License

After you set up licensing, you can revoke licenses at any time, for example, if you want to change the client configuration or reallocate the license. The following procedure revokes the license from a component.

To revoke a license

1. Stop the IDOL solution that uses the license.
2. At the command prompt, run the following command:

   InstallDir/*ExecutableName*[.exe] -revokelicense -configfile *cfgFilename*

This command returns the license to the License Server.

You can send the `LicenseInfo` action from a web browser to the running License Server to check for free licenses. In this sample output from the action, one IDOL Server license is available for allocation to a client.

```
<autn:Product>
   <autn:ProductType>IDOLSERVER</autn:ProductType>
   <autn:Client>
      <autn:IP>192.123.51.23</autn:IP>
      <autn:ServicePort>1823</autn:ServicePort>
      <autn:IssueDate>1063192283</autn:IssueDate>
      <autn:IssueDateText>10/09/2003 12:11:23</autn:IssueDateText>
   </autn:Client>
      <autn:TotalSeats>2</autn:TotalSeats>
      <autn:SeatsInUse>1</autn:SeatsInUse>
</autn:Product>
```

Troubleshoot License Errors

The table contains explanations for typical licensing-related error messages.

License-related error messages

Error message	Explanation
Error: Failed to update license from the license server. Your license cache details do not match the current service configuration. Shutting the service down.	The configuration of the service has been altered. Verify that the service port and IP address have not changed since the service started.
Error: License for *ProductName* is invalid. Exiting.	The license returned from the License Server is invalid. Ensure that the license has not expired.

License-related error messages, continued

Error message	Explanation
Error: Failed to connect to license server using cached licensed details.	Cannot communicate with the License Server. The product still runs for a limited period; however, you should verify whether your License Server is still available.
Error: Failed to connect to license server. Error code is SERVICE: *ErrorCode*	Failed to retrieve a license from the License Server or from the backup cache. Ensure that your License Server can be contacted.
Error: Failed to decrypt license keys. Please contact Autonomy support. Error code is SERVICE:*ErrorCode*	Provide Micro Focus Big Data Support with the exact error message and your license file.
Error: Failed to update the license from the license server. Shutting down	Failed to retrieve a license from the License Server or from the backup cache. Ensure that your License Server can be contacted.
Error: Your license keys are invalid. Please contact Autonomy support. Error code is SERVICE:*ErrorCode*	Your license keys appear to be out of sync. Provide Micro Focus Big Data Support with the exact error message and your license file.
Failed to revoke license: No license to revoke from server.	The License Server cannot find a license to revoke.
Failed to revoke license from server *LicenseServer Host*:*LicenseServerPort*. Error code is *ErrorCode*	Failed to revoke a license from the License Server. Provide Micro Focus Big Data Support with the exact error message.
Failed to revoke license from server. An instance of this application is already running. Please stop the other instance first.	You cannot revoke a license from a running service. Stop the service and try again.
Failed to revoke license. Error code is SERVICE:*ErrorCode*	Failed to revoke a license from the License Server. Provide Micro Focus Big Data Support with the exact error message.
Your license keys are invalid. Please contact Autonomy Support. Error code is ACISERVER:*ErrorCode*	Failed to retrieve a license from the License Server. Provide Micro Focus Big Data Support with the exact error message and your license file.
Your product ID does not match the generated ID.	Your installation appears to be out of sync. Forcibly revoke the license from the License Server and rename the `license` and `uid` directories.
Your product ID does not match this configuration.	The service port for the module or the IP address for the machine appears to have changed. Check your configuration file.

Start and Stop Eduction Server

This section describes how to start and stop Eduction Server.

Start Eduction

NOTE: Your License Server must be running before you start Eduction.

TIP: You can configure services to start automatically when you start the machine.

To start Eduction

- Start Eduction from the command line using the following command:

  `eductionserver.exe -configfile configname.cfg`

  where the optional `-configfile` argument specifies the path of a configuration file to use.
- On Windows, if you have installed Eduction as a service, start the service from the Windows Services dialog box.
- On Linux, if you have installed Eduction as a service, use one of the following commands:
  - On machines that use systemd:

    `systemctl start eductionserver`
  - On machines that use system V:

    `service eductionserver start`
- On Linux you can use the script `start-eductionserver.sh` which is provided in the installation directory.

Command-Line Options

This section describes additional parameters that you can use when you start Eduction Server from the command-line.

NOTE: Options are case sensitive.

Usage: `eductionserver [options]`

Options	Description
`-version`	Displays the program version. This option must be the only argument. The program ends after the version information is displayed.

Options	Description
`directory`	Sets the working directory. The Eduction ACI server starts from this directory. All other arguments used in the command line are relative to this directory.
`-configfile` *`file`*	Sets the configuration file name. This option overrides the default configuration file name of `eductionserver.cfg`.
`-revokelicense`	Revokes a lock on the Eduction license from the License Server.
`-install`	Installs Eduction as a service (Windows only). Syntax: `-install [-start auto|manual|disabled] [-username username] [-password password]` The `-start` option allows you to specify the startup mode. By default, it is automatic. If you do not supply a user name and password, Eduction runs under a local account.
`-uninstall`	Uninstalls the Eduction service.

NOTE: On Linux, the ACI server requires the C++ library, `libstdc++.so`. To ensure the server can locate the required library, set the Library Path:

```
setenv LD_LIBRARY_PATH bin:$LD_LIBRARY_PATH
```

Stop Eduction

You can stop Eduction by using one of the following procedures.

To stop Eduction

- Send the `Stop` service action to the service port.

  `http://`*`host`*`:`*`ServicePort`*`/action=Stop`

  where:

`host`	is the host name or IP address of the machine where Eduction is installed.
`ServicePort`	is the Eduction service port (specified in the `[Service]` section of the configuration file).

- On Windows, if Eduction is running as a service, stop Eduction from the Windows Services dialog box.
- On Linux, if Eduction is running as a service, use one of the following commands:

- On machines that use systemd:

  ```
  systemctl stop eductionserver
  ```

- On machines that use system V:

  ```
  service eductionserver stop
  ```

- On Linux, if you started Eduction with the script `start-eductionserver.sh`, run the script `stop-eductionserver.sh` which is provided in the installation directory.

Configure Eduction Server

You configure Eduction Server by modifying the Eduction Server configuration file.

The Eduction Server configuration file contains settings required for the server to run, and specifies settings to use when performing Eduction.

The default configuration file is named `eductionserver.cfg`, and is included in your Eduction Server package. You can start Eduction Server using a different configuration file by using the `-configfile` command-line option (see Command-Line Options, on page 42).

This section describes how to modify the Eduction Server configuration file.

Modify Configuration Parameter Values

You modify Eduction configuration parameters by directly editing the parameters in the configuration file. When you set configuration parameter values, you must use UTF-8.

> **CAUTION:** You must stop and restart Eduction for new configuration settings to take effect.

This section describes how to enter parameter values in the configuration file.

Enter Boolean Values

The following settings for Boolean parameters are interchangeable:

```
TRUE = true = ON = on = Y = y = 1
```

```
FALSE = false = OFF = off = N = n = 0
```

Enter String Values

To enter a comma-separated list of strings when one of the strings contains a comma, you can indicate the start and the end of the string with quotation marks, for example:

ParameterName=**cat,dog,bird,"wing,beak",turtle**

Alternatively, you can escape the comma with a backslash:

ParameterName=**cat,dog,bird,wing\,beak,turtle**

If any string in a comma-separated list contains quotation marks, you must put this string into quotation marks and escape each quotation mark in the string by inserting a backslash before it. For example:

ParameterName=**"<font face=\"arial\" size=\"+1\"><b>","<p>"**

Here, quotation marks indicate the beginning and end of the string. All quotation marks that are contained in the string are escaped.

Include an External Configuration File

You can share configuration sections or parameters between ACI server configuration files. The following sections describe different ways to include content from an external configuration file.

You can include a configuration file in its entirety, specified configuration sections, or a single parameter.

When you include content from an external configuration file, the `GetConfig` and `ValidateConfig` actions operate on the combined configuration, after any external content is merged in.

In the procedures in the following sections, you can specify external configuration file locations by using absolute paths, relative paths, and network locations. For example:

```
../sharedconfig.cfg
K:\sharedconfig\sharedsettings.cfg
\\example.com\shared\idol.cfg
file://example.com/shared/idol.cfg
```

Relative paths are relative to the primary configuration file.

> **NOTE:** You can use nested inclusions, for example, you can refer to a shared configuration file that references a third file. However, the external configuration files must not refer back to your original configuration file. These circular references result in an error, and Eduction does not start.
>
> Similarly, you cannot use any of these methods to refer to a different section in your primary configuration file.

Include the Whole External Configuration File

This method allows you to import the whole external configuration file at a specified point in your configuration file.

To include the whole external configuration file

1. Open your configuration file in a text editor.
2. Find the place in the configuration file where you want to add the external configuration file.
3. On a new line, type a left angle bracket (<), followed by the path to and name of the external configuration file, in quotation marks (""). You can use relative paths and network locations. For example:

```
< "K:\sharedconfig\sharedsettings.cfg"
```

4. Save and close the configuration file.

Include Sections of an External Configuration File

This method allows you to import one or more configuration sections (including the section headings) from an external configuration file at a specified point in your configuration file. You can include a whole configuration section in this way, but the configuration section name in the external file must exactly match what you want to use in your file. If you want to use a configuration section from the external file with a different name, see Merge a Section from an External Configuration File, on the next page.

To include sections of an external configuration file

1. Open your configuration file in a text editor.
2. Find the place in the configuration file where you want to add the external configuration file section.
3. On a new line, type a left angle bracket (<), followed by the path of the external configuration file, in quotation marks (""). You can use relative paths and network locations. After the configuration file path, add the configuration section name that you want to include. For example:

   ```
   < "K:\sharedconfig\extrasettings.cfg" [License]
   ```

   NOTE: You cannot include a section that already exists in your configuration file.

4. Save and close the configuration file.

Include Parameters from an External Configuration File

This method allows you to import one or more parameters from an external configuration file at a specified point in your configuration file. You can import a single parameter or use wildcards to specify multiple parameters. The parameter values in the external file must match what you want to use in your file. This method does not import the section heading, such as `[License]` in the following examples.

To include parameters from an external configuration file

1. Open your configuration file in a text editor.
2. Find the place in the configuration file where you want to add the parameters from the external configuration file.
3. On a new line, type a left angle bracket (<), followed by the path of the external configuration file, in quotation marks (""). You can use relative paths and network locations. After the configuration file path, add the name of the section that contains the parameter, followed by the parameter name. For example:

   ```
   < "license.cfg" [License] LicenseServerHost
   ```

To specify a default value for the parameter, in case it does not exist in the external configuration file, specify the configuration section, parameter name, and then an equals sign (=) followed by the default value. For example:

```
< "license.cfg" [License] LicenseServerHost=localhost
```

You can use wildcards to import multiple parameters, but this method does not support default values. The * wildcard matches zero or more characters. The ? wildcard matches any single character. Use the pipe character | as a separator between wildcard strings. For example:

```
< "license.cfg" [License] LicenseServer*
```

4. Save and close the configuration file.

Merge a Section from an External Configuration File

This method allows you to include a configuration section from an external configuration file as part of your Eduction configuration file. For example, you might want to specify a standard SSL configuration section in an external file and share it between several servers. You can use this method if the configuration section that you want to import has a different name to the one you want to use.

To merge a configuration section from an external configuration file

1. Open your configuration file in a text editor.
2. Find or create the configuration section that you want to include from an external file. For example:

   ```
   [SSLOptions1]
   ```

3. After the configuration section name, type a left angle bracket (<), followed by the path to and name of the external configuration file, in quotation marks (""). You can use relative paths and network locations. For example:

   ```
   [SSLOptions1] < "../sharedconfig/ssloptions.cfg"
   ```

   If the configuration section name in the external configuration file does not match the name that you want to use in your configuration file, specify the section to import after the configuration file name. For example:

   ```
   [SSLOptions1] < "../sharedconfig/ssloptions.cfg" [SharedSSLOptions]
   ```

   In this example, Eduction uses the values in the [SharedSSLOptions] section of the external configuration file as the values in the [SSLOptions1] section of the Eduction configuration file.

   > **NOTE:** You can include additional configuration parameters in the section in your file. If these parameters also exist in the imported external configuration file, Eduction uses the values in the local configuration file. For example:
   >
   > ```
   > [SSLOptions1] < "ssloptions.cfg" [SharedSSLOptions]
   > SSLCACertificatesPath=C:\IDOL\HTTPConnector\CACERTS\
   > ```

4. Save and close the configuration file.

Configure Client Authorization

You can configure Eduction to authorize different operations for different connections.

Authorization roles define a set of operations for a set of users. You define the operations by using the `StandardRoles` configuration parameter, or by explicitly defining a list of allowed actions in the `Actions` and `ServiceActions` parameters. You define the authorized users by using a client IP address, SSL identities, and GSS principals, depending on your security and system configuration.

For more information about the available parameters, see the *Eduction Reference*.

IMPORTANT: To ensure that Eduction allows only the options that you configure in `[AuthorizationRoles]`, make sure that you delete any deprecated *`Role`*`Clients` parameters from your configuration (where *`Role`* corresponds to a standard role name, for example `AdminClients`).

To configure authorization roles

1. Open your configuration file in a text editor.
2. Find the `[AuthorizationRoles]` section, or create one if it does not exist.
3. In the `[AuthorizationRoles]` section, list the user authorization roles that you want to create. For example:

   ```
   [AuthorizationRoles]
   0=AdminRole
   1=UserRole
   ```

4. Create a section for each authorization role that you listed. The section name must match the name that you set in the `[AuthorizationRoles]` list. For example:

   ```
   [AdminRole]
   ```

5. In the section for each role, define the operations that you want the role to be able to perform. You can set `StandardRoles` to a list of appropriate values, or specify an explicit list of allowed actions by using `Actions`, and `ServiceActions`. For example:

   ```
   [AdminRole]
   StandardRoles=Admin,ServiceControl,ServiceStatus

   [UserRole]
   Actions=GetVersion
   ServiceActions=GetStatus
   ```

   NOTE: The standard roles do not overlap. If you want a particular role to be able to perform all actions, you must include all the standard roles, or ensure that the clients, SSL identities, and so on, are assigned to all relevant roles.

6. In the section for each role, define the access permissions for the role, by setting `Clients`, `SSLIdentities`, and `GSSPrincipals`, as appropriate. If an incoming connection matches one of the allowed clients, principals, or SSL identities, the user has permission to perform the operations allowed by the role. For example:

   ```
   [AdminRole]
   StandardRoles=Admin,ServiceControl,ServiceStatus
   Clients=localhost
   SSLIdentities=admin.example.com
   ```

7. Save and close the configuration file.
8. Restart Eduction for your changes to take effect.

IMPORTANT: If you do not provide any authorization roles for a standard role, Eduction uses the default client authorization for the role (localhost for `Admin` and `ServiceControl`, all clients for `Query` and `ServiceStatus`). If you define authorization only by actions, Micro Focus recommends that you configure an authorization role that disallows all users for all roles by default. For example:

```
[ForbidAllRoles]
StandardRoles=*
Clients=""
```

This configuration ensures that Eduction uses only your action-based authorizations.

Send Actions to Eduction

You can make requests to Eduction by using either GET or POST HTTP request methods.

- A GET request sends parameter-value pairs in the request URL. GET requests are appropriate for sending actions that retrieve information from Eduction, such as the `GetStatus` action. For more information, see Send Actions by Using a GET Method, below.
- A POST request sends parameter-value pairs in the HTTP message body of the request. POST requests are appropriate for sending data to Eduction. In particular, you must use POST requests to upload and send files to Eduction. For more information, see Send Data by Using a POST Method, on the next page.

NOTE: The `MaxInputString` and `MaxFileUploadSize` configuration parameters set a limit on the maximum size of HTTP strings and the maximum size of files that you can upload. The default values for these parameters allow HTTP strings that contain a maximum of 64,000 characters, and uploaded files with a maximum size of 10,000,000 bytes.

Send Actions by Using a GET Method

You can use GET requests to send actions that retrieve information from Eduction.

When you send an action using a GET method, you use a URL of the form:

`http://host:port/?action=action&parameters`

where:

host	is the IP address or name of the machine where Eduction is installed.
port	is the Eduction ACI port.
action	is the name of the action that you want to run. The `action` (or `a`) parameter must be the first parameter in the URL string, directly after the host and port details.
parameters	are the required and optional parameters for the action. These parameters can follow the `action` parameter in any order.

You must:

- Separate each parameter from its value with an equals symbol (`=`).
- Separate multiple values with a comma (`,`).
- Separate each parameter-value pair with an ampersand (`&`).

For more information about the actions that you can use with Eduction, refer to the *Eduction Reference*.

GET requests can send only limited amounts of data and cannot send files directly. However, you can set a parameter to a file path if the file is on a file system that Eduction can access. Eduction must also be able to read the file.

Send Data by Using a POST Method

You can send files and binary data directly to Eduction using a POST request. One possible way to send a POST request over a socket to Eduction is using the cURL command-line tool.

The data that you send in a POST request must adhere to specific formatting requirements. You can send only the following content types in a POST request to Eduction:

- `application/x-www-form-urlencoded`
- `multipart/form-data`

> **TIP:** Eduction rejects POST requests larger than the size specified by the configuration parameter `MaxFileUploadSize`.

Application/x-www-form-urlencoded

The `application/x-www-form-urlencoded` content type describes form data that is sent in a single block in the HTTP message body. Unlike the query part of the URL in a GET request, the length of the data is unrestricted. However, Eduction rejects requests that exceed the size specified by the configuration parameter `MaxFileUploadSize`.

This content type is inefficient for sending large quantities of binary data or text containing non-ASCII characters, and does not allow you to upload files. For these purposes, Micro Focus recommends sending data as `multipart/form-data` (see Multipart/form-data, below).

In the request:

- Separate each parameter from its value with an equals symbol (`=`).
- Separate multiple values with a comma (`,`).
- Separate each parameter-value pair with an ampersand (`&`).
- Base-64 encode any binary data.
- URL encode all non-alphanumeric characters, including those in base-64 encoded data.

Multipart/form-data

In the `multipart/form-data` content type, the HTTP message body is divided into parts, each containing a discrete section of data.

Each message part requires a header containing information about the data in the part. Each part can contain a different content type; for example, `text/plain`, `image/png`, `image/gif`, or `multipart/mixed`. If a parameter specifies multiple files, you must specify the **`multipart/mixed`** content type in the part header.

Encoding is optional for each message part. The message part header must specify any encoding other than the default (7BIT).

Multipart/form-data is ideal for sending non-ASCII or binary data, and is the only content type that allows you to upload files. For more information about form data, see http://www.w3.org/TR/html401/interact/forms.html.

> **NOTE:** With cURL, you specify each message part using the `-F` (or `--form`) option. To upload a file in a message part, prefix the file name with the `@` symbol. For more information on cURL syntax, see the cURL documentation.

GetStatus

You can use the `GetStatus` service action to verify the Eduction is running. For example:

`http://`*`Host`*`:`*`ServicePort`*`/action=GetStatus`

> **NOTE:** You can send the `GetStatus` action to the ACI port instead of the service port. The `GetStatus` ACI action returns information about the Eduction setup.

GetLicenseInfo

You can send a `GetLicenseInfo` action to Eduction to return information about your license. This action checks whether your license is valid and returns the operations that your license includes.

Send the `GetLicenseInfo` action to the Eduction ACI port. For example:

```
http://Host:ACIport/action=GetLicenseInfo
```

The following result indicates that your license is valid.

```
<autn:license>
       <autn:validlicense>true</autn:validlicense>
</autn:license>
```

As an alternative to submitting the `GetLicenseInfo` action, you can view information about your license, and about licensed and unlicensed actions, on the **License** tab in the Status section of IDOL Admin.

IDOL Admin

IDOL Admin is an administration interface for performing ACI server administration tasks, such as gathering status information, monitoring performance, and controlling the service. IDOL Admin provides an alternative to constructing actions and sending them from your web browser.

Prerequisites

Eduction includes the `admin.dat` file that is required to run IDOL Admin.

IDOL Admin supports the following browsers:

- Edge
- Chrome (latest version)
- Firefox (latest version)

Install IDOL Admin

You must install IDOL Admin on the same host that the ACI server or component is installed on. To set up a component to use IDOL Admin, you must configure the location of the `admin.dat` file and enable Cross Origin Resource Sharing.

To install IDOL Admin

1. Stop the ACI server.
2. Save the `admin.dat` file to any directory on the host.
3. Using a text editor, open the ACI server or component configuration file. For the location of the configuration file, see the ACI server documentation.
4. In the `[Paths]` section of the configuration file, set the `AdminFile` parameter to the location of the `admin.dat` file. If you do not set this parameter, the ACI server attempts to find the `admin.dat` file in its working directory when you call the IDOL Admin interface.

5. Enable Cross Origin Resource Sharing.
6. In the `[Service]` section, add the `Access-Control-Allow-Origin` parameter and set its value to the URLs that you want to use to access the interface.

   Each URL must include:

   - the `http://` or `https://` prefix

   > **NOTE:** URLs can contain the `https://` prefix if the ACI server or component has SSL enabled.

   - The host that IDOL Admin is installed on
   - The ACI port of the component that you are using IDOL Admin for

   Separate multiple URLs with spaces.

   For example, you could specify different URLs for the local host and remote hosts:

   ```
   Access-Control-Allow-Origin=http://localhost:9010
   http://Computer1.Company.com:9010
   ```

   Alternatively, you can set `Access-Control-Allow-Origin=*`, which allows you to access IDOL Admin using any valid URL (for example, `localhost`, direct IP address, or the host name). The wildcard character (`*`) is supported only if no other entries are specified.

   If you do not set the `Access-Control-Allow-Origin` parameter, IDOL Admin can communicate only with the server's ACI port, and not the index or service ports.
7. Start the ACI server.

You can now access IDOL Admin (see Access IDOL Admin, below).

Access IDOL Admin

You access IDOL Admin from a web browser. You can access the interface only through URLs that are set in the `Access-Control-Allow-Origin` parameter in the ACI server or component configuration file. For more information about configuring URL access, see Install IDOL Admin, on the previous page.

To access IDOL Admin

- Type the following URL into the address bar of your web browser:

  `http://`*`host`*`:`*`port`*`/action=admin`

  where:

`host`	is the host name or IP address of the machine where the IDOL component is installed.
`port`	is the ACI port of the IDOL component you want to administer.

Chapter 6: Run Eduction in Eduction Server

This section describes how to run an Eduction matching operation with Eduction Server.

Server Actions

The Eduction ACI server provides the following actions (case insensitive):

`GetStatus`	Returns the status of the Eduction Server, including version information and entities selected for matching. See GetStatus, on page 51.
`EduceFromFile`	Performs Eduction on text (read from a file) and returns the matches.
`EduceFromText`	Performs Eduction on text (provided in the request) and returns the matches.
`RedactFromFile`	Performs Eduction on text (read from a file) and redacts the matches to return the redacted text.
`RedactFromText`	Performs Eduction on text (provided in the request) and redacts the matches to return the redacted text.

For more information about these actions, and example responses, refer to the *Eduction Server Reference*.

For details of how to send an action, see Send Actions to Eduction, on page 49.

Select Entities at Runtime

In Eduction Server, you can specify many configuration settings as query parameters in the ACI request, which allows you to customize the extraction behavior for an individual action.

For example:

```
http://localhost:13000/?action=EduceFromFile&MatchCase=True&Grammars=place_albal.ecr
```

When you set a parameter as part of an action, it overrides the corresponding parameter value in the configuration file.

For information about all the actions and action parameters that are available with Eduction Server, refer to the *Eduction Server Reference*.

Change the Grammars and Entities

You can use the `Grammars` and `Entities` action parameters to specify a subset of your configured grammars or entities.

- `Grammars`. This action parameter corresponds to the `ResourceFiles` configuration parameter. You set `Grammars` to a comma-separated list of grammar files to load, and Eduction uses all the entities in the selected grammar files, including entities not set in your configuration file. The grammar files must already exist in your configuration file.

  `Grammars=`*`GrammarFile`*`[,`*`GrammarFile2`*`]`

- `Entities`. You set `Entities` to a comma-separated list of entities. In this case, Eduction uses only the specified entities. The entities and corresponding grammar files must already exist in your configuration file.

> **NOTE:** If you set both `Entities` and `Grammars`, Eduction ignores the `Grammars` parameter and uses only the specified entities.

You can use wildcard expressions in the `Grammars` or `Entities` parameters. You can use the * wildcard to match any number of characters, or the ? wildcard to match a single character. For example:

```
action=EduceFromText&Text=I thought it was a bad idea. Es ist nicht
gut.&Grammars=sentiment_*.ecr
```

This example uses all the available sentiment grammars for the extraction.

Part IV: Use Eduction

This section describes how to use Eduction matching, and how to improve your results.

- Improve Eduction Matches
- Sentiment Analysis
- Create and Modify Eduction Grammars
- Compile and Test Grammars
- Post-Processing

Chapter 7: Improve Eduction Matches

This section describes some different areas that you might want to adjust to improve your match results and performance.

Case Sensitive Matches

By default, Eduction matches characters case sensitively, which has better performance than case insensitive matching. When you require case insensitive matching, there are several ways to configure it:

- configure `MatchCase`.
- configure individual grammars, entities, and entries with case sensitivity options, in a custom grammar file.
- use case normalization.

Micro Focus recommends that you always create and use Eduction grammars that allow you to do case sensitive matching, because it has better performance. Most of the standard grammars come with entities using common and appropriate case styles. Some also have different entities for different case styles. If your data uses a consistent case, it is unlikely that you need to use case insensitive matching.

Configure MatchCase

The simplest way to turn off case sensitivity is to set the `MatchCase` configuration parameter to **`False`** in the configuration file.

If you run Eduction with `MatchCase=False`, entities are optimized for case-insensitive matching when they are loaded. This can increase the time required to initialize Eduction, so if you regularly use a grammar file with `MatchCase=False` you can optimize the entities for case-insensitive matching at compile time instead. See Optimize Case-Insensitive Matching, on page 94.

`MatchCase` applies to all matches, so it can have a significant performance impact. In general, Micro Focus recommends that you use one of the other options to enable case insensitive matching. For the best performance, write a grammar file using only upper or lower case and then normalize your input (see Case Normalization, below).

Configure Custom Grammars

When you create your own custom XML grammar files, you can configure individual grammars, entities, and entries individually to be case sensitive or insensitive.

When you configure case sensitivity at a lower level, it overrides the higher level settings. Additionally, if you reference the entity in another entity, it maintains its own case sensitivity setting.

Most entities in the standard grammars do not have case sensitivity set explicitly, giving you the flexibility to use case sensitivity as required in your grammars.

> **NOTE:** If you design an entity for case insensitive matching, it is important that entries in the entity have a consistent case style to ensure that all matches are extracted correctly. You should use all lower case, all upper case, or all initial capitals, but not a mixture.
>
> Eduction uses an optimization technique for case insensitive matching that might not extract every possible match if you do not define the entity consistently.

Case Normalization

Case sensitive matching generally has better performance than case insensitive matching. When you require case insensitive matching, you can use case normalization to give the same performance as case-sensitive matching.

When you want to use case normalization:

- Do not set case sensitivity explicitly in grammars and entities.
- Set the `MatchCase` configuration parameter to **`True`**.
- Create all entries in your entities in either all lower case, or all upper case.
- Set `CaseNormalization` to:
  - **`LOWER`** if all your entities are lower case.
  - **`UPPER`** if all your entities are upper case.

Eduction normalizes the input data accordingly before the (case sensitive) matching. This process means that both your input and grammars are all in the same case, so the matching is effectively case insensitive, with the performance benefits of case sensitive matching.

For more information about these configuration parameters, see CaseNormalization, on page 334 and MatchCase, on page 355.

Match Special Characters

By default, Eduction matches whole words; that is, all matches start at the beginning of a word and stop at the end of a word. Any pattern that corresponds to a substring of a word does not match the substring. For example, the pattern *rain* does not match the word *raining*.

Additionally, by default Eduction treats all punctuation as a word boundary, so you cannot match punctuation at the start of a word.

You can change this Eduction matching behavior if the default settings do not allow you to extract the information you want from your data.

Match Part of a Word

In normal use, it usually makes sense to match only whole words with entities. If you do need to allow partial matches, you can set the `MatchWholeWord` configuration parameter to **`False`** (see MatchWholeWord, on page 356).

> **NOTE:** This change can have a significant performance impact.

When you set `MatchWholeWord` to **`False`**, Eduction does not consider the difference between words and word boundaries. This might mean it matches a word boundary at the start of the match. However, if you explicitly want to match punctuation at the start of a match, Micro Focus recommends that you use `TangibleCharacters` instead of `MatchWholeWord` (see Match Punctuation at the Start of a Match, below and TangibleCharacters, on page 371).

> **IMPORTANT:** If you set `MatchWholeWord` to **`False`**, `TangibleCharacters` has no effect.

Match Punctuation at the Start of a Match

By default, Eduction treats all punctuation as a word boundary. When Eduction *parses* and *tokenizes* the input data, it ignores all word boundaries at the start of a word (such as spaces). To match punctuation characters at the start of a match, you can set the punctuation as a tangible character (see TangibleCharacters, on page 371).

For example:

- To match UK phone numbers with area codes, such as `(01223) 448000`, you can set `TangibleCharacters` to **`(`** to include the opening parenthesis as part of the match.
- To match negative numbers, such as `Bob's account shows -455 pounds`, you can set `TangibleCharacters` to **`-`** to specify that `-` is part of the word that you want to extract. By default, Eduction extracts `455` instead of `-455`, even if the `-` character is part of the pattern in your Eduction grammar file.

You can also configure all punctuation marks as tangible characters, by enabling `TokenWithPunctuation` (see TokenWithPunctuation, on page 372).

> **NOTE:** Punctuation in this case refers to punctuation in the ASCII character set.
>
> Punctuation marks regarded as characters from Chinese, Japanese, or Korean (CJK) languages are treated in the same way as other CJK text (see Match CJK Text, on the next page).

When you set tangible characters, Eduction treats those characters as part of the word you want to match, rather than as word boundaries. For this reason, the pattern in your grammar file must explicitly include any characters you have set as tangible.

For example, if your grammar file contains an entity to match the pattern `bob`, by default Eduction returns a match for the phrase `One day "bob" walked in to town`.

However, if you set `TangibleCharacters` to `"`, the same pattern does not retrieve a match for this phrase. In this case, Eduction treats `"` as part of the word that it has to match, so `bob` is not the same as `"bob"`, and it does not match. You must include the `"` character in your grammar pattern to retrieve a match for `"bob"`.

Eduction tries to match whole words inside word boundaries. If you set up a grammar rule that includes punctuation characters that you want to match, but you do not include those characters as tangible characters, the results might not be what you expect. For example:

- Eduction ignores boundary characters if the next set of characters in the data matches a defined pattern.

  For example, if `TangibleCharacters` does not include `!`, the grammar pattern `[!"bo]*` (match `!` or `"` or `b` or `o`, zero or many times) returns a match for `bob` when a document contains the text `!bob`.

- Eduction does not return a match for boundary characters that appear on their own in documents.

- Eduction returns a match for boundary characters if they are included in the pattern and are embedded in another match.

  For example, the grammar pattern `[!"bo]*` returns a match for `b"o!b` if a document contains the text `b"o!b`. If you did not include `"` or `!` in the grammar pattern, Eduction would treat them as boundary characters and would not return a match.

Examples

The following table shows some examples of the difference in Eduction matches when `TangibleCharacters` includes `!`, `"` and `#` and when it does not.

These examples use the grammar rule `[A-Za-z!"#]+@com` (match all uppercase or all lowercase letters or `!` or `"` or `#`, followed by the string `@com`).

Document text	Returns (TangibleCharacters does not include !, " and #)	Returns (TangibleCharacters includes !, " and #)
!Chris@com went to town to	Chris@com	!Chris@com

Document text	Returns (TangibleCharacters does not include !, " and #)	Returns (TangibleCharacters includes !, " and #)
pick up some fruit		
Ch!ris@com went to town to pick up some melons	Ch!ris@com	Ch!ris@com
“Chris@com” went to town to get some tea	Chris@com	“Chris@com”
!@com went to town to get nothing	No match returned	!@com

You must determine your use of `TangibleCharacters` by what you are trying to achieve, and the type of content you are working with. For example, it is likely to be less helpful if you are working with continuous strings of information, where punctuation characters separate possible Eduction matches.

Match CJK Text

All input data and grammars for Eduction must be encoded in UTF-8.

For characters in Chinese, Japanese, and Korean (CJK) languages, the data is tokenized character by character, with a word boundary on both sides of each character. This process ensures that when Eduction processes data that includes CJK characters, it can match the logical word unit in the data without disabling `MatchWholeWord`.

Extract Entities from Tables

Eduction Table mode allows you to extract entities from a table, according to the values in the header of that table. This process allows you to target extraction on likely values in structured data, rather than extracting every possible entity value from a table. It can also improve the confidence that an ambiguous entity value corresponds to a particular type of data.

In standard extraction, Eduction searches text for a value that matches a particular entity. In many cases the entity values are distinctive, and so you can be reasonably confident that matches are relevant. For example, a string that matches an address entity is unlikely to be anything else.

Many other entity values are potentially ambiguous. For example, a number might match several entity types, and a date might be a date of birth or an event date. Without further information, it is difficult to determine whether these values are useful.

For unstructured text, you can use *landmarks* to find relevant information. Landmarks are values that identify a particular entity, without being a part of the entity value. For example, the phrase *Date of Birth* is a landmark. When a document contains the value `Date of Birth: 06/07/80`, it is highly likely that the date is a date of birth, and you can treat the data accordingly.

> **NOTE:** The IDOL PII Package, IDOL PHI Package, and IDOL PCI Package, provide landmark entities in most grammars. To extract entities from tables with the Eduction standard grammar files, you might need to create your own landmark entities.

For structured data, it is less likely that the landmark occurs next to the entity. You might have the value Date of Birth in a table heading, and the actual date values in the rows below. In this case, you can use table extraction to extract the values that correspond to the landmark.

Table Formats

In table mode, Eduction can find header and cell values for CSV (comma-separated values) and TSV (tab-separated values) files. If you use Eduction with Connector Framework Server (CFS) or IDOL NiFi Ingest, you can also use structured XML tables.

Eduction can also process multiple tables from a single text stream when they are separated by the correct table delimiters:

- Start table: "<blank line><tab line>" (`\n\n\t\n`)
- End table: "<tab line><blank line>" (`\n\t\n\n`)

These delimiters reset the header matches, so that Eduction looks for a new header row.

> **NOTE:** The first table does not need a start delimiter. If there is text outside table delimiters, Eduction treats it as a new separate table.

If you use KeyView to extract table data from your files to send to Eduction, you can configure KeyView to give the correct delimiters format when extracting tables. To get the right format from KeyView, you must make the following changes in you KeyView configuration:

- set the target character set to UTF-8.
- enable the Tab Delimited option.
- enable the Output Table Delimiters option.

For more information about the KeyView configuration, refer to the *KeyView Filter SDK Programming Guide*.

Configure Table Extraction

In table extraction, you define an entity or entities that you want to detect in the header row, and entities that you want to detect in the cells under that header. When Eduction matches one of these entities in the header row of a table, it attempts to extract the corresponding cell entities from the cells in that column.

To configure these, you use the HeaderEntityN, on page 352 and CellEntityN, on page 337 configuration parameters.

For example:

```
[Eduction]
HeaderEntity0=pii/date/dob/landmark/all
CellEntity0=pii/date/nocontext/all
```

This example matches date of birth landmark values in the header, and for all subsequent rows in that column, it extracts any date values.

> **NOTE:** You can specify multiple entities, either by providing a comma-separated list, or by using wildcard characters. In this case, if the table header matches any of the configured header entities, Eduction matches the cell content against any of the configured cell entities.
>
> This option might be useful if you want to match a particular entity in multiple languages, or if you want to include a custom entity in addition to a standard one.
>
> You must configure any entities that you want to use for matching in the ResourceFiles, on page 369 parameter. For example, the example configuration above uses the `combined_date.ecr` grammar from the PII grammar set:
>
> `ResourcesFiles=combined_date.ecr`

You can optionally also set:

- MaxSearchHeaderRow, on page 358. The number of rows at the top of the table to search for header entities. This option might be useful if there is irrelevant information in the first few rows of your tables. Eduction searches up to the first *N* non-empty rows, and stops when it finds one of the configured header entities.
- HeaderEntityMatchLimitN, on page 351 and CellEntityMatchLimitN, on page 336. The maximum number of header column and cell matches to allow for the corresponding entities. These options might be useful if you want to find some matches for a particular entity, but would prefer to ignore further matches in favor of reducing the processing time.

To use table extraction with Connector Framework Server (CFS) or IDOL NiFi Ingest, you can also add the EntityFieldN, on page 344 parameter. This parameter specifies the field that CFS or NiFi write the extracted entities to in your documents.

In this case, if you do not set EntityFieldN, on page 344, Eduction uses the value of CellEntityN, on page 337 to create a default field name (the capitalized entity name, with `/ * ?` characters replaced with underscores).

```
[Eduction]
HeaderEntity0=pii/date/dob/landmark/all
CellEntity0=pii/date/nocontext/all
EntityField0=DATE_OF_BIRTH
```

> **NOTE:** You cannot specify EntityFieldN, on page 344 for only some of your CellEntityN, on page 337 values; you must either use the default value for all, or set EntityFieldN, on page 344 for all.

These parameters are the same for extracting entities from CSV or TSV table files, and for structured table data in XML, such as the output from Media Server OCR. For structured XML tables, there is an additional parameter, `TableCellPath`, for CFS and IDOL NiFi Ingest. `TableCellPath` describes the

structure of the XML to allow Eduction to find the cells. For more information, refer to the Connector Framework Server or NiFi Ingest documentation.

For the Eduction SDK, you do not need to configure `TableCellPath`, because you use functions to locate the cells.

> **NOTE:** You cannot extract entities from structured XML data in Eduction Server or edktool. In these cases you must use a CSV or TSV table file.

Run Table Extraction

After you configure table extraction, you can run Eduction as normal, with a CSV or TSV table file as input.

- In the Eduction SDK:
  - C: You provide a table file by using the `AddInputText` or `SetInputStream` functions. You can use the `EdkGetMatchTablePosition` function to retrieve the row and column details of a match.

    For structured XML, call `EdkAddTableCell` to add table cell data to the session. You can optionally also populate an `EdkOffset` struct with offset information, and pass in a pointer to this as part of the `EdkAddTableCell` call. This option allows you to generate matches with offsets that reflect the global position of the cell. By default, the produced matches have offsets relative to the start of the cell.

    When a row is complete, call `EdkEndTableRow`. For the last row of the table, set the `bFinalRow` argument to **`true`**.
  - Java: You provide the table file by using the `addInputText` or `setInputStream` functions. You can use `public EDKMatch.TablePosition getTablePosition()` to return an object with two public members, row and column.

    For structured XML, call `addTableCell` to add table cell data to the session. To generate matches with offsets that reflect the global position of the cell, call the version that accepts `offsetBytes` and `offsetCodepoints` as arguments. The other version produces matches with offsets that are relative to the start of the cell.

    When a row is complete, call `endTableRow`. For the last row of the table, set the `finalRow` argument to **`true`**.
  - .NET: You provide the table file by using `AddInputText` or `SetInputStream` functions. You can use the readonly property `public IExtractionMatchTablePosition TablePosition` to return an object that has the readonly properties Row and Column.

    For structured XML, call `AddTableCell` to add table cell data to the session. To generate matches with offsets that reflect the global position of the cell, call the version that accepts the `TextOffset` parameter, which is a simple struct that contains the offsets in bytes and Unicode characters, of the start of the cell data in the global input stream. The other version produces matches with offsets that are relative to the start of the cell.

    When a row is complete, call `EndTableRow`. For hte last row of the table, set the `final_row` argument to **`true`**.

> **NOTE:** To use Table Extraction with the Eduction SDK, you must create an Eduction engine with a configuration file. See the Standalone API Usage section for your language in API Reference, on page 28.

- In Eduction Server and the edktool command-line tool, you provide the table file as plain input text. Eduction returns the matches in the response.
- In CFS and NiFI, the ingestion process sends the table file to the Eduction engine. CFS and NiFi add the match details to the output documents.

Select Matches

By default, Eduction does not return all possible matches. For example, it does not return matches that overlap a previous match:

- If you have patterns for *fox jumps* and *jumps over*, the text "*The quick brown fox jumps over the lazy dog*" returns only the first match, because the second match overlaps it.

Eduction returns the longest possible match at the same position. For example:

- If you have a pattern for *brown fox*, and one for *brown fox jumps*, only the second match returns.

Eduction has configuration parameters to allow you to modify this and other matching behavior.

Return the Smallest Match

When Eduction finds two matches that start at the same position, it returns only one match unless you enable overlaps. By default, Eduction returns the longer match.

You can configure it to use the shorter match by enabling `NonGreedyMatch` (see NonGreedyMatch, on page 359). However, always consider whether you need to use this option, or whether you could redefine your Eduction grammar to be more precise instead.

Generally, Micro Focus recommends that you make your Eduction grammar definition as precise as possible, which reduces the chance of getting two matches at the same position. A precise Eduction grammar is also more efficient during extraction.

Overlapping and Duplicate Matches

You can return overlapping matches by enabling the `AllowOverlaps` parameter (see AllowOverlaps, on page 332).

When the same string occurs at more than one position in the input data, by default Eduction returns only the first match. You can allow duplicates (for example, if you need to find the positions of all occurrences) by setting the `AllowMultipleResults` parameter (see AllowMultipleResults, on page 329).

If you want to return only unique matches in each document, set `EnableUniqueMatches` to **`True`** (see EnableUniqueMatches, on page 340). Eduction returns only a single occurrence of a particular value (the first match), even if the matches occur for different entities.

Return Multiple Results for a Single Match

In some instances, you might want to get multiple results for a single match. For example, if a word can occur in different contexts, you might want to tag a document according to the occurrence of the word.

```
<entity name="IT_industry">
   <entry headword="software">
      <synonym>CompanyA</synonym>
      <synonym>HP</synonym>
   </entry>
   <entry headword="hardware">
      <synonym>CompanyB</synonym>
      <synonym>HP</synonym>
   </entry>
</entity>
```

With this entity, `CompanyA` returns `software`, while `CompanyB` returns `hardware`. A match of `HP` might return either `software` or `hardware`. If you want to use this entity to return both `software` and `hardware` for `HP`, set the `AllowMultipleResults` configuration parameter to **`True`** (see AllowMultipleResults, on page 329).

Match Validity

Eduction assesses the validity of a match in the following order:

1. If the match is found outside of the required zones, discard it.
2. If the match does not meet the minimum score requirement, discard it.
3. If duplicates are allowed:
   - If the instance of the match is allowable, count this instance and return the match.
   - Otherwise, count this instance and discard it.
4. If duplicates are not allowed for the entity field:
   - If the matched text has been found before, discard it.
   - Otherwise, if the instance of the match is allowable, count this instance and return the matched text.
   - If the instance of the match is not allowable, count this instance and discard it.

Components

Eduction *components* allow you to extract attributes from a single match. The attributes are called components because they are the components of a match.

For example, sentiment analysis can match the phrase *Their service is fantastic* as conveying positive sentiment. Components can then break this phrase down to identify *service* as the subject matter, and *fantastic* as the adjective that describes the subject.

Eduction does not extract components by default.

To use components, you must include components in the grammar that you want to use, and turn on components at run time. You must also configure some additional parameters. See Configure Components, below.

When to Use Components

The English Eduction sentiment analysis grammar defines components for `TOPIC`, `SENTIMENT`, `POSITIVE`, and `NEGATIVE`. You can use these components by configuring Eduction accordingly.

Components are useful when the information that you want to match has an underlying pattern that you want to preserve.

For example, you might use components to extract data from tables and return it in a suitable format. For an example, see EntityComponentFieldN, on page 343.

> **NOTE:** Most of the standard grammars do not define components, because these grammars are mainly dictionaries or basic patterns that you can use to build more complex patterns. You might want to define components when you reference these basic entities in your patterns for custom grammars.

Configure Components

To use the components defined in the Eduction grammar, you must configure Eduction with:

- `OutputSimpleMatchInfo` set to **`False`**. See OutputSimpleMatchInfo, on page 361.
- `EnableComponents` set to **`True`**. See EnableComponents, on page 340.
- the `EntityComponentField` for the entity. See EntityComponentFieldN, on page 343.

Define the Components

In the Eduction grammar, you define components by using the extension operator (`?A=`*`ComponentName`*`:`*`Pattern`*`)` (see Regular Expressions, on page 305). Consider the following example entity:

```
<entity name=test>
   <pattern>(?A=SUBJECT:(?A^noun)) is (?A=SENTIMENT:(?A^adjective))</pattern>
</entity>
```

In this example, an earlier part of the grammar might define the `noun` entity to match nouns such as *service* and *facility*, and the `adjective` entity to match descriptions such as *fantastic* and *appalling*. This `test` entity then matches the phrase *service is fantastic*, and returns the `SUBJECT` component with the text *service*, and the `SENTIMENT` component with the text *fantastic*.

Results Relevance

Eduction returns entities based on the extraction rules from the grammars and dictionaries.

The edktool command-line tool includes a test mode to measure the *precision* and *recall* of your extraction, to determine the result relevance. This mode allows you to check how well your grammar works on your text data.

Precision and recall are statistical measures that compare the results that a human marks and results that the engine returns. The following terms describe result relevance as used in Eduction.

- **True Positives** (TP). Results that are identified by both a human and the engine. That is, the engine returns an entity that is confirmed as true by the person marking the document.
- **False Positives** (FP). Results that are identified by the engine, and are not marked by a human. That is, the engine returns an entity that is not confirmed by the person marking the document.
- **True Negatives** (TN). Results that are not marked by either the person marking the document, or the engine.
- **False Negatives** (FN). Results that are marked by a human, and are not marked by the engine. That is, the engine does not return an entity that has been marked as true by the person marking the document.

From these relevance terms, you can determine precision and recall as follows:

- Recall is the percentage of true relevant entities that are extracted by a rule:

  ```
  TP / (TP + FN) * 100
  ```

- Precision is the percentage of extracted entities that are true entities:

  ```
  TP / (TP + FP) * 100
  ```

For more information about edktool, see Compile and Test Grammars, on page 92.

Custom Grammar Guidelines

This section describes some guidelines that you can use when you create custom grammars.

Eduction is generally very fast at grammar compilation and entity extraction. However, some expressions in the grammar patterns can increase the extraction and compilation times significantly.

The grammar files that are included in the Eduction packages are designed to be as fast as possible. The following section describes some ways to ensure that your user-created Eduction grammars also work quickly.

In general, the more concise a grammar is, the faster it returns matches. For the best performance, use the simplest entity possible that matches what you need to match. Use additional features only if you need them.

> **TIP:** Before you create a custom grammar, check the standard grammars to see whether the entity you want is already supported. If it is not, contact Micro Focus support. The entity you want to detect might be supported in an upcoming release. Alternatively, Micro Focus might be able to add support in future, if other customers want it too.
>
> Using an official grammar means you do not have to maintain it.
>
> It is also generally easier to extend an existing grammar by using a user extension file than to create a completely new grammar.

The following sections describe the most efficient use of specific features, and how you might be able to avoid using slow features in some cases.

For details of the grammar syntax, see Grammar Format Reference, on page 297. In particular, for details of the regular expression syntax used in these examples, see Regular Expressions, on page 305.

> **NOTE:** Some configuration settings affect extraction speed, for example MatchCase, on page 355, MatchWholeWord, on page 356, AllowOverlaps, on page 332, and NonGreedyMatch, on page 359.

For a tutorial that gives an example of how to create a custom grammar, refer to *IDOL Expert*.

Exclusions and Negations

Always try to describe the value that you want to match, rather than a value to exclude.

For example, you can exclude a match by using a score of zero (`score="0"` in the pattern definition). However, this option can significantly increase processing time. Micro Focus recommends that you define your patterns such that your entities do not match the values you want to exclude.

For example, to extract mobile phone numbers that do not end in `25`:

```
<pattern>07[0-9]{7}[013-9][0-9]</pattern>
<pattern>07[0-9]{7}[0-9][0-46-9]</pattern>
```

is faster than

```
<pattern>07[0-9]{9}</pattern>
<pattern score="0">07[0-9]{7}25</pattern>
```

> **TIP:** You can also use a Lua script to filter out the exclusions in post-processing. This option might be preferable when you want to define more general patterns.

Similarly, Micro Focus recommends that you describe the values to match rather than using the negation operator. For example, to avoid matching the digit 0, use `[1-9]` rather than `[^0]`. The negation operator can increase compilation and processing times.

Case Sensitivity

See Also: Case Sensitive Matches, on page 57.

Wherever possible, use case sensitive matching. After exclusions, case insensitive matching is the slowest feature in Eduction. In particular, avoid using the `case="insensitive"` tag in entities or patterns. When case sensitivity is essential, consider using one of the following options:

- Use alternate casing in a regular expression pattern. For example, to match the word *Paul* case insensitively, you might use the pattern `[Pp][Aa][Uu][Ll]`.

  > **NOTE:** Extensive use of this format can greatly increase grammar compilation times, and compiled ECR file size.

- Use text normalization. The Eduction process can normalize text before matching, to convert the input to all lower- or all uppercase. You can define your entities in one case, and normalize the input text the same way.

  This approach might be unsuitable if you need to match capitalized words in certain places.

Reference or Copy Entities

When you create a custom grammar, you can match a previously defined entity and either:

- copy it to the new entity by using the syntax `(?A:`
- reference it in the new entity by using the syntax `(?A^`

For large or complex Eduction grammars, copying entities results in a very large grammar file, which can take an extremely long time to compile. In addition, the resulting file can take longer to load and scan than the equivalent file created by using references.

For example, if an entity matches a static list of several thousand names, always use the reference operator to include it in other patterns. Similarly, reference an entity if it contains patterns that can match a wide variety of expressions.

For very simple grammar files, it might be faster to copy entities, because this method creates an ECR with more efficient instructions for extraction.

In general, Micro Focus recommends that you use references in all cases, unless your grammar file is very simple, and extraction speed is critical.

Merge Entities

When you know what entities you want to use for extraction in advance, you can improve performance by creating a single public entity that includes each of these entities. It is quicker for Eduction to process a single large entity definition than for it to use several smaller definitions. Similarly, extraction is faster because it needs to check only one entity for a match.

For example, the following public entity merges the entities animal, vegetable, and mineral:

```
<entity name="my_entity" type="public">
   <pattern>(?A^animal)</pattern>
   <pattern>(?A^vegetable)</pattern>
   <pattern>(?A^mineral)</pattern>
</entity>
```

In your Eduction configuration, you can use this merged entity rather than the individual ones.

> **TIP:** You can use merged entities to improve performance even if you only use entities from Micro Focus grammar files.

Micro Focus recommends that you merge any entities that you can, unless merging them alters what the grammar can match.

Include Grammars

When you include another grammar in your custom grammar, use the ECR form rather than the XML. When you use an XML inclusion, compiling your grammar requires in-memory compilation of the included grammar file, which might in turn require the same for any further inclusions.

It is always quicker to compile a grammar that includes ECR files.

Use Common Forms of Matches

Where possible, use only the most common match cases for your grammar. Additional forms for acceptable matches can result in increased processing times, particularly for complex forms. Consider your matches carefully and only add additional forms if it is necessary.

Quantifiers

The syntax *expression*`{n,m}` matches at least *n*, but at most *m* consecutive occurrences of the specified *expression*. When *m* is large, it can result in a large ECR file and slow extraction.

In this situation, Micro Focus recommends that you use `{n,}` unless the upper bound *m* is important.

Reduce the Number of Ways to Match

When there are many ways to match a particular entity, Eduction must try many methods to determine whether an input string matches, and to attempt to find the longest match. Where possible, make sure that there are as few ways to match your entity as possible. In particular, avoid using the `{n,m}` operator for entities that can match a wide variety of tokens.

For example, the following entity matches a five letter word that occurs between one and three times. This five letter word must occur between two matches of an entity called *name*.

```
<entity name=”myentity”>
   <pattern>(?A:name) ([A-Za-z]{5}){1,3} (?A:name)</pattern>
</entity>
```

The example name entity might also include five-letter names, such as Chris, James, or Alice, that also match the regex pattern. When Eduction processes text, it tries every combination that might lead to a match. In this case there might be a very large number of options, which could be very slow.

Lua Post-Processing

You can use a Lua Script for more advanced matching, which might improve processing speed in complex cases. You can use Lua scripts in many ways, from checksum validation to checking match proximity (if you use the en masse matching mode).

The match proximity check might be useful for complex entities. That is, if matches from a certain set of entity names are close together, you can consider them to form a single, complex entity. This approach is also useful in cases where the major elements of an entity are separated by filler or unknown content.

Optional Phrases

Try to start your entities with a required entity or phrase. An entity that starts with one or more optional phrases can be very slow during extraction. For example, the following type of pattern might be very slow:

```
<pattern>(?A^animal)?(?A^vegetable)*(?A^mineral)?(?A^name)</pattern>
```

In this case, the entity `name` is required, while `animal`, `vegetable`, and `mineral` are optional. When extracting, Eduction must check each word for matches in the `animal` entity, then check whether it matches `vegetable`, `mineral`, and then `name`. If the word matches `animal`, Eduction must then check whether the following word matches `vegetable`, `mineral`, or `name`, and so on.

This process can be time-consuming, particularly if each of the optional entities occurs regularly in the input text.

This issue does not occur if the pattern starts with the required phrase. For example:

```
<pattern>(?A^name)(?A^animal)?(?A^vegetable)*(?A^mineral)?</pattern>
```

In this case, Eduction must only check each word for matches in `name`, and it checks for the optional phrases only when it finds a match for `name`.

Use Private Entities

Eduction has public and private entities. Public entities are available to match during extraction. Private entities are available to use in other entities, but you cannot extract them directly as matches.

You can use private entities to break up complicated pattern expressions into several simple patterns, which you can use in a single public entity. This process keeps your patterns simple, which makes them easier to maintain and troubleshoot.

Making an entity public unnecessarily in a grammar can result in longer processing times if you use all entities from the grammar. Micro Focus recommends that you mark all entities as private by default. You can then expose as public only the entities that represent the entirety of a match, rather than subelements.

Output Exclusions

You can use output exclusions (the `(?A!` operator) for text that is useful when you identify a match, but that is not part of the match itself. For example, if you have form labels that identify a piece of information (such as *Name*, *Telephone Number*, and so on), you can use these in an entity to match the correct information, and then use an output exclusion so that the final output includes only the important content.

> **NOTE:** Output exclusions can increase compilation times if you exclude complex entities.

Patterns and Headwords

You can extract regular expressions by using patterns (the `<pattern>` element), or by explicitly listing each possible match as a `headword` (in the `<entry>` element).

For example, the following alternatives are equivalent:

```
<pattern>[Ee]xtract(ed|ing|s)?</pattern>
```

and

```
<entry headword="Extract"/>
<entry headword="Extracted"/>
<entry headword="Extracting"/>
<entry headword="Extracts"/>
<entry headword="extract"/>
<entry headword="extracted"/>
<entry headword="extracting"/>
<entry headword="extracts"/>
```

Patterns are often faster to code, easier to maintain, and faster for extraction. However, if there are fewer than about 50 entries represented by one pattern, the compilation time is faster for headwords.

Micro Focus recommends that you use patterns in your entities, unless the pattern becomes too complex. Additionally, if the compilation time becomes too slow, you might want to consider replacing the simplest patterns with headwords.

> **NOTE:** You can also provide a list of words by adding multiple `<pattern>` elements with a word in each, unless the word or phrase contains characters that are also valid regular expression syntax.

Components

See Also: Components, on page 67

In some cases, including components in an Eduction grammar file can increase the extraction time, even if you do not enable the components during the extraction. This occurs because the ECR is less compact than the equivalent file that does not describe components. Do not include components if you do not need them.

When you do use components, Micro Focus recommends that you make the structure of the components as uniform as possible.

For example:

```
   <pattern>(?A=COMPONENT:(?A^entity_A) )(?A^entity_B) (?A^entity_C) (?A^entity_
D)</pattern>
   <pattern>(?A=COMPONENT:(?A^entity_A)) (?A^entity_B)(?A^entity_D) (?A^entity_
C)</pattern>
   <pattern>(?A=COMPONENT:(?A^entity_A) (?A^entity_B))(?A^entity_C) (?A^entity_
E)</pattern>
```

might be slower than:

```
   <pattern>(?A=COMPONENT:(?A^entity_A) (?A^entity_B)) (?A^entity_C) (?A^entity_
D)</pattern>
   <pattern>(?A=COMPONENT:(?A^entity_A) (?A^entity_B)) (?A^entity_D) (?A^entity_
C)</pattern>
   <pattern>(?A=COMPONENT:(?A^entity_A) (?A^entity_B)) (?A^entity_C) (?A^entity_
E)</pattern>
```

Scores

You can add scoring to your entities to indicate the relative confidence for matches. The score for a match is the product of all the scores of the patterns or entities it includes, with a default value of one.

It is up to you how you use scores in your use case. Eduction does not define the meaning of a score.

Whitespace

You can define a space in grammars by using a space character, or by using the `\s` syntax. The `\s` syntax matches all types of whitespace, such as spaces, tabs, and newlines. In most practical

situations, the matches you want from the input text only include spaces and it is slightly faster to use a space, rather than `\s`.

Control Eduction Processing Time

Eduction matching is usually fast, but Micro Focus recommends that you set limits on processing so that your application can deal with a variety of input.

For example, for a large input text with a high density of matches, it can be time-consuming to retrieve all the matches. You must consider carefully whether your application requires all matches (which might number in the millions), or if it is enough to capture the first few hundred for any particular piece of input text.

You can control the number of matches to process by using the MaxMatchesPerDoc configuration parameter, which instructs an Eduction session to stop searching for matches after a certain number of matches have been found. To stop searching for specific entities after a certain number of matches have been found, but continue searching for other entities, set EntityMatchLimitN.

The following configuration parameters also strong affect the number of matches you might obtain:

- AllowMultipleResults, on page 329
- AllowOverlaps, on page 332

To control the amount of time that Eduction can spend processing data, you can set the RequestTimeout, on page 367 configuration parameter.

In the Eduction SDK, you can use the following steps:

1. Set the RequestTimeout, on page 367 configuration parameter. Alternatively:
   - For C, use `EdkSessionSetRequestTimeoutPrecise` on an individual session.
   - For .NET, use the `ITextExtractionSession::SetRequestTimeoutPrecise` method.
   - For Java, use the function `TextExtractionSession::setRequestTimeoutPrecise` to set a timeout for the session.

   In each case, the argument must be in milliseconds.
2. Get the current time in epoch milliseconds, for example by using `time()` in C, `System.DateTimeOffset.Now.ToUnixTimeMilliseconds()` in .NET, or `System.currentTimeMillis()` in Java.
3. In the Eduction SDK, send the time in epoch milliseconds to the session by using one of the following options:
   - For C, use `EdkSessionSetStartTime`, passing in the value you obtained from step 2 as the argument.
   - For .NET, use `ITextExtractionSession::SetStartTimePrecise`, passing in the epoch milliseconds value you obtained from the previous step as the argument.
   - For Java, use `TextExtractionSession::setStartTime`, passing in the epoch milliseconds value you obtained from the previous step as the argument.

4. Obtain matches in the usual fashion, by calling `EdkGetNextMatch` in C, or by looping over the session object in .NET and Java.

   > **DEPRECATED:** Do not use `EdkGetNextMatchTimed` in C, which is deprecated in Eduction SDK version 12.8.0 and later.

5. Check for timeouts in the match loop. You can do this by calling `EdkGetMatchTimedOut` in C, or `TextExtractionSession::getTimedOut` in Java, or `ITextExtractionSession::getTimedOut` in .NET. If a timeout has occurred, you can break out of the loop, as required for your application.

You can find examples of timeout handling in the sample programs provided in the Eduction SDK release package.

> **TIP:** If your application does significant processing before you call Eduction, and you want to use an overall application timeout, obtain the current time in epoch milliseconds at the very start of your application processing rather than waiting until just before you call the Eduction functions.

Find Repeated Matches

A *repeated match* occurs when Eduction finds another match (exactly the same text), at a different location in the input (a different offset). For example, if you are searching some input text for telephone numbers, Eduction could find a match "564123". The same text could occur later in the document and would result in a repeated match. Repeated matches might belong to the same entity, but this is not always the case because multiple entities can match the same text.

> **TIP:** Eduction does not return repeated matches if you configure the engine with EnableUniqueMatches set to `TRUE`.

Eduction normally returns matches in the order that they appear in the input, but you might prefer to process a match, followed by all of its repeated matches, and then return to the next unique match. In extreme cases, where the matched text is repeated many times, this provides a convenient way to stop processing and move on to the next unique match or maybe even the next document.

Each Eduction API provides a way to find the next repeated match. The SDK includes sample programs, in each language, that demonstrate this functionality.

> **NOTE:** This feature is not supported for streaming input or in table mode.

C API

In the C API, instead of calling `EdkGetNextMatch` you can call `EdkGetNextRepeatedMatch`.

If the input contains another match with the same text as the current match, you can then call `EdkGetRepeatedMatchByteOffset` and `EdkGetRepeatedMatchCodepointOffset` to establish the location of the repeated match.

You can call `EdkGetNextRepeatedMatch` repeatedly. If the matched text does not occur again, Eduction returns `EdkNoMatch` and you can proceed to the next unique match by calling `EdkGetNextMatch`.

Any repeated matches that you access using `EdkGetNextRepeatedMatch` are not returned by subsequent calls to `EdkGetNextMatch`. By using `EdkGetNextRepeatedMatch` you are changing the order in which the matches are returned.

The following code sample demonstrates how you might use these functions. For more information about these functions, refer to the API reference documentation.

```
while (EdkGetNextMatch(session) == EdkSuccess)
{
    // call match accessors and do something with the information

    while (EdkGetNextRepeatedMatch(session) == EdkSuccess)
    {
        size_t nRepCodepointOffset = 0;
        size_t nRepByteOffset = 0;
        EdkGetRepeatedMatchCodepointOffset(session, &nRepCodepointOffset);
        EdkGetRepeatedMatchByteOffset(session, &nRepByteOffset);

        // do something with the offsets...
    }
}
```

Java API

The repeated matches functionality in the Java API is similar to the C API. You can iterate over repeated matches as shown in the following code sample. You can call the `getByteOffset()` and `getCodepointOffset()` methods to establish the position of a repeated match.

```
for (EDKMatch match : session)
{
    // call match accessors and do something with the information

    for (EDKRepeatedMatchOffset repeat : match)
    {
        long byteOffset = repeat.getByteOffset();
        long codepointOffset = repeat.getCodepointOffset();

        // do something with the offsets
    }
}
```

.NET API

The repeated matches functionality in the .NET API is similar to the C API. You can iterate over repeated matches as shown in the following code sample. The properties `ByteOffset` and `CodepointOffset` provide the position of a repeated match.

```
foreach (IExtractionMatch match in session)
{
    // call match accessors and do something with the information

    foreach (IExtractionRepeatedMatch repeat in match.RepeatedMatches)
    {
        long byteOffset = repeat.ByteOffset;
        long codepointOffset = repeat.CodepointOffset;

        // do something with the offsets
    }
}
```

Chapter 8: Sentiment Analysis

This section describes the sentiment analysis grammars and some information about how to optimize sentiment analysis.

For more information about the sentiment analysis grammar, see Sentiment Grammars, on page 113.

Eduction Sentiment Grammar Files

Eduction Sentiment Analysis allows you find whether text has positive, negative, or neutral sentiment. For example, you can use it to determine whether users of a particular product or service are satisfied or not, based on an automated analysis of reviews.

The following table lists the languages that support sentiment analysis, and lists the name of the standard sentiment grammar and the user modification file. Each of these languages also support component extraction and user modification.

Language	Sentiment Grammar	User Modification File
Arabic	`sentiment_ara.ecr`	`sentiment_user_ara.xml`
Chinese	`sentiment_chi.ecr`	`sentiment_user_chi.xml`
Czech	`sentiment_cze.ecr`	`sentiment_user_cze.xml`
Dutch	`sentiment_dut.ecr`	`sentiment_user_dutch.xml`
English	`sentiment_eng.ecr`	`sentiment_user_eng.xml`
French	`sentiment_fre.ecr`	`sentiment_user_fre.xml`
German	`sentiment_ger.ecr`	`sentiment_user_ger.xml`
Italian	`sentiment_ita.ecr`	`sentiment_user_ita.xml`
Polish	`sentiment_pol.ecr`	`sentiment_user_pol.xml`
Portuguese	`sentiment_por.ecr`	`sentiment_user_por.xml`
Russian	`sentiment_rus.ecr`	`sentiment_user_rus.xml`
Spanish	`sentiment_spa.ecr`	`sentiment_user_spa.xml`
Turkish	`sentiment_tur.ecr`	`sentiment_user_tur.xml`

The Sentiment Analysis Grammars

Eduction matches input data to patterns defined with regular expressions (grammars).

The sentiment analysis grammar first defines dictionaries with the parts of speech. There are different dictionaries for positive and negative words, and other categories that describe different effects on the sentiment, where appropriate.

These dictionaries combine to form simple phrases that convey positive or negative sentiments. Finally, these phrases are padded, usually with other phrases, to form various patterns for the final entities, which match strings from the text that express positive or negative sentiment.

The grammar files are designed to be used out of the box. You just need to load the appropriate grammar file, and optionally choose the entities (usually positive or negative) to match with.

The sentiment grammar files have *lite* versions. The lite versions are identical to the full versions in most respects, but they do not support components or user modification. They can process data up to twice as fast as the full versions, depending on language.

Micro Focus recommends that you use the lite versions except when you need to use components or modify the built-in dictionaries.

The lite grammars have the same name as the full version, with `_lite` after the language. For example, the file name of the Chinese sentiment grammar file is `sentiment_chi.ecr`, and the file name of the lite version is `sentiment_chi_lite.ecr`.

Extend the Sentiment Analysis Grammar

The grammar files generally contain sufficient information to work with a wide range of data, from formal reports to user reviews and social media feeds. However, the recall (the percentage of matches that are actually returned, out of the total number of matches that should return in theory) can be low for some input data. Also, some examples might convey a different sentiment depending on your viewpoint. For example:

> The phrase *Company A is much better than Company B* might convey a positive or negative sentiment depending on whether you are with Company A or Company B.

In these situations, you can improve the recall or adjust the sentiment analysis by extending the grammar.

You can extend the grammar by adding to the appropriate dictionaries in the sentiment grammar file. For example, if you are on the side of Company A, you can add *Company A* to the positive list.

> **NOTE:** There are slight variations in the grammar files of different languages, so this does not apply to all languages.

For more information, see Extend Grammars, on page 86.

Polarity Scoring

The sentiment analysis grammars support *polarity scoring*. Polarity scoring is a number, usually between 0.50 and 1.50, that represents the strength of the sentiment in the matched phrase. For example:

- a strongly positive or negative phrase might have a score of 1.35
- a typical phrase might have a score of 1.00
- a match where the sentiment is weak or ambiguous might have a score of 0.60

You can edit the user modification files to increase the scores of words in the dictionaries. For example, add the following on a new line in the user modification file to modify the existing entry `flexible` so that it has a score of `1.23`:

```
"      <entry score="1.23" headword="flexible"/>"
```

> **NOTE:** `sentiment_basic_eng.ecr` does not support polarity scoring.

Verb Sentiment Transitivity

The sentiment analysis grammar files support *verb sentiment transitivity*.

This feature enables the TOPIC components of the matches to determine what the sentiment is about with more accuracy, by using advanced contextual understanding of whether that sentiment is being expressed about the subject or object of the sentence.

For example, given two matches, `x likes y` and `x wins at y`, the grammar files can determine that the first match is a positive statement about `y`, whereas the second match is a positive statement about `x`.

Sentiment Analysis

The sentiment analysis grammar files contain:

- dictionaries of types of word (for example, positive adjective, negative noun, neutral adverb, and so on).
- patterns that describe how to combine these dictionaries to form positive and negative phrases.

For example, you could run sentiment extraction using the English sentiment grammar file (`sentiment_eng.ecr`), with the following hotel review as the input file:

The room was nice enough, with a plug in radiator, tv with an English news channel, hot shower, comfy bed. The receptionist we first dealt with was miserable and rude, and just grunted at us and rolled her eyes because we were too early for check in having just got off the morning train from Khabarovsk. Fortunately, a younger receptionist with a nice smile appeared, spoke to us helpfully suggesting a few cafes nearby to pass some time, and we tried to forget about the other

woman.
Breakfast is terrible. Unidentifiable cordials, gloomy porridge, bread rolls filled with things you don't expect for breakfast, like potato, egg and dill. Don't come here for the breakfast, but for the cost of the room in a city like Vladivostok, the hotel is still decent value for money.

The following is a sample of the output that this produces:

```
<?xml version="1.0" encoding="UTF-8"?>
  <MATCHLIST>
    <DOCUMENT Type="IDOL IDX" ID="Unknown">
      <FIELD Name="DRECONTENT">
        <FIELD_INSTANCE Value="1">
          <MATCH EntityName="sentiment/positive/eng" Offset="7" OffsetLength="5"
          Score="1.05" NormalizedTextSize="17" NormalizedTextLength="17"
          OriginalTextSize="17" OriginalTextLength="17">
            <ORIGINAL_TEXT>The room was nice</ORIGINAL_TEXT>
            <NORMALIZED_TEXT>The room was nice</NORMALIZED_TEXT>
            <COMPONENTS>
              <COMPONENT Name="TOPIC" Text="The room" Offset="0"
              OffsetLength="0" TextSize="8" TextLength="8"/>
              <COMPONENT Name="SENTIMENT" Text="nice" Offset="13"
              OffsetLength="13" TextSize="4" TextLength="4"/>
            </COMPONENTS>
          </MATCH>
          <MATCH EntityName="sentiment/negative/eng" Offset="494"
          OffsetLength="492" Score="1.2" NormalizedTextSize="21"
          NormalizedTextLength="21" OriginalTextSize="21"
          OriginalTextLength="21">
            <ORIGINAL_TEXT>Breakfast is terrible</ORIGINAL_TEXT>
            <NORMALIZED_TEXT>Breakfast is terrible</ NORMALIZED_TEXT>
            <COMPONENTS>
              <COMPONENT Name="TOPIC" Text="Breakfast" Offset="0"
              OffsetLength="0" TextSize="9" TextLength="9"/>
              <COMPONENT Name="SENTIMENT" Text="terrible" Offset="13"
              OffsetLength="13" TextSize="8" TextLength="8"/>
            </COMPONENTS>
          </MATCH>
        </FIELD_INSTANCE>
      </FIELD>
    </DOCUMENT>
</MATCHLIST>
```

The following example configuration shows the recommended usage:

```
[Eduction]
ResourceFiles=grammars/sentiment_eng.ecr
// Note: replace sentiment_eng.ecr by sentiment_user_eng.ecr if using user
modification

// standard entities for all sentiment analysis in English:
Entity0=sentiment/positive/eng
Entity1=sentiment/negative/eng
```

```
EntityField0=POSITIVE_VIBE
EntityField1=NEGATIVE_VIBE
EntityComponentField0=TOPIC,SENTIMENT
EntityComponentField1=TOPIC,SENTIMENT

// some invalid matches are given very low scores so that we can filter them out:
MinScore=0.1

// for extraction of Twitter handles, hashtags and emoticons:
TangibleCharacters=@#:;

// for displaying metadata:
OutputScores=True
OutputSimpleMatchInfo=False
EnableComponents=True
```

For more information about the sentiment analysis grammar files, see Sentiment Grammars, on page 113.

Perform Sentiment Analysis on Short Comments

The standard sentiment analysis grammars are designed for high precision. For some sources of short comment data, such as YouTube comments, no positive or negative matches are found in some documents despite sentiment clearly being expressed.

If recall with the full `sentiment_eng.ecr` grammar file is too low, and your documents are generally short comments, use `sentiment_basic_eng.ecr` to extract additional matches. This grammar contains carefully-selected lists of positive and negative terms that help determine the sentiment of a document in which `sentiment_eng.ecr` found no matches.

`sentiment_basic_eng.ecr` contains terms in title case, but research shows that for most data these impair recall, so these are given a lower score. Micro Focus recommends that you set `EntityMinScore`*`N`* to `0.4` to filter out these terms unless you need them.

`sentiment_basic_eng.ecr` does not expose TOPIC or SENTIMENT components, and does not use scores to reflect strength or reliability of polarity. The following additional example configuration shows the recommended usage:

```
[Eduction]
ResourceFiles=grammars/sentiment_eng.ecr,grammars/sentiment_basic_eng.ecr
// optional further layer of analysis for very short documents:
Entity2=sentiment/basic_positive/eng
Entity3=sentiment/basic_negative/eng
EntityField2=BASIC_POSITIVE_VIBE
EntityField3=BASIC_NEGATIVE_VIBE
// remove this setting to include basic matches in titlecase - this is not
recommended because on most data it decreases precision:
EntityMinScore2=0.4
EntityMinScore3=0.4
```

Financial Sentiment Analysis

The financial sentiment analysis grammar, `financial_strength.ecr`, allows you to extract sentiment information from financial news and analysis. You can use this grammar, for example, to monitor news feeds for financial sentiment concerning your company, or a list of companies that you are interested in.

This grammar relies on the `contextualize_matches.lua` post-processing script. This script uses the configurable entities to match sections of your documents, and then recombines these entities, with the surrounding text, to return a different set of entities that represent the financial sentiment.

To use financial sentiment analysis, you must configure both the grammar and the post-processing script. See Post-Processing, on page 105.

To configure financial sentiment analysis, you include one or more of the basic entities in your Eduction configuration. These entities are:

- finance/analyst
- finance/company/tagged/known
- finance/news

Eduction uses these entities to find particular chunks of text that contain financial information, but it does not return the values that it finds for these entities. Instead, the post-processing script finds text around the entity values and combines them with additional sentiment analysis processing to create a new set of entities. The new set describes positive, neutral, and negative sentiment in several contexts.

For details of the entities that the `financial_strength.ecr` and `contextualize_matches.lua` grammar and script combination creates, see the standard grammar reference F, on page 146F, on page 146.

Chapter 9: Create and Modify Eduction Grammars

Eduction uses grammar files to identify and tag entities in documents. They are written in XML in a format specific to Eduction. They are then compiled, using the Eduction command-line tool, into ECR files that Eduction can easily read at runtime.

Eduction includes a collection of standard grammar files that make it easy to identify common entities such as names and phone numbers. These are described in Standard Grammars, on page 113.

Eduction Grammar Structure

An Eduction grammar defines patterns for matching text in a document. A pattern is a combination of characters and operators. An operator is a sequence of special characters that match text by following the rules associated with the operator.

Pattern	Description	Matches
Smith|John	Match either *Smith* or *John*	*Smith* *John*
[0-9]{3}	Match a sequence of three characters in the range 0 through 9	*123* *456*

In this example, the square bracket operators [] are used to match on any of the characters 0 through 9 and the curly braces {} are used to repeat the previous pattern three times.

Grammars are described using XML. The file `edk.dtd` contains the template that defines the XML that Eduction understands. When writing grammars for Eduction, Micro Focus recommends that you reference `edk.dtd` at the start of the XML grammar file using the include statement, and that you use a DTD-compatible XML authoring tool to eliminate syntax errors and save time.

Here is an example of a simple Eduction grammar:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE grammars SYSTEM "edk.dtd">
<grammars>
   <grammar name="mygrammar">
```

```
      <entity name="name" type="public">
         <pattern>Smith|John</pattern>
      </entity>
      <entity name="digits" type="public">
         <pattern>[0-9]{3}</pattern>
      </entity>
   </grammar>
</grammars>
```

This grammar defines two entities: `mygrammar/name` and `mygrammar/digits`.

For full details of the Eduction grammar XML syntax, and the `edk.dtd`, see Grammar Format Reference, on page 297.

For a more extensive set of example Eduction grammar files, see Example Grammar Files, on page 88.

Extend Grammars

The Eduction standard grammars provide good coverage for common pieces of information that you would normally want to extract from your data. They are designed so that you can easily reference them in any custom grammars that you create.

For some data, the coverage provided might not be sufficient. In this case, you can extend the entities provided with new entries to improve the *recall* of the extraction (the percentage of matches that are actually returned, out of the total number of true matches). For more information about recall, see Results Relevance, on page 68.

You cannot edit the standard grammars in place because they are provided in .ECR format. You can, however, add more entries to an existing entity in an .ECR grammar file by extending it in a custom grammar file in XML format.

When to Extend a Grammar

You might consider extending a grammar if the recall of the existing grammar is low. In this case, you can work out what items the existing grammar does not match, and add these as new entries in the appropriate entities in your custom grammar.

You then compile the custom grammar (using edktool) before you use it, to allow Eduction to load it quicker. You can then replace the original grammar file with the new grammar file.

> **TIP:** If you need to detect entities that are not supported by the Eduction grammars available from Micro Focus, raise the issue with your support contact. The entity you want to detect might be supported in an upcoming release. Alternatively, Micro Focus might be able to add support in future, if other customers want it too.
>
> Using an official grammar means that you do not have to maintain it.

Create a Reference to an Existing Entity

You can build custom grammars from scratch. However, the standard grammars provide many basic entities that you can reference in your grammars, which allows you to create new custom grammars quickly.

If you reference other entities in an entity that you create, you can use one of the following reference extensions:

- (`?A^`*`Entity`*) During compilation, create a link to the referenced entity from your entity.
- (`?A:>`*`Entity`*) During compilation, copy the compiled version of the referenced entity to your entity.

For the first option, compilation is quicker, and the resulting grammar file is a lot smaller. The second option can provide a small performance gain during extraction. Micro Focus recommends that you use the first option in most cases, unless the extraction performance is critical (see Reference or Copy Entities, on page 70).

For more information about these options, see Regular Expressions, on page 305. For a tutorial that describes in more detail how to create a new grammar to extend existing entities, refer to *IDOL Expert*.

Add More Entries to an Entity

To add more entries to an entity, create a new XML grammar file. In the new grammar file, include the .ECR file that contains the entity that you want to extend.

Ensure that your grammar file defines the same grammar and entity as the included grammar file. The full entity name, including the grammar prefix, must match for the grammar extension to work. Set the extend mode of the entity in your new grammar to **`Append`**, and add the extra entries in the entity.

Replace the Current Entities

Although most of the time you would add new entries when you extend a grammar, you can sometimes choose to replace it entirely. To do this, set the extend mode of the entity in your new grammar file to **`Replace`**.

Extend the Sentiment Grammars

Grammar extension is particularly useful when you use Eduction for sentiment analysis.

There are two main reasons why you might extend the sentiment grammar file.

- You want Eduction to find some of the matches it misses because the compiled grammar does not include some of the positive or negative adjectives and adverbs in your data. To do this, you

simply extend the appropriate entities with the new entries.

- You want to change the sentiment for some objects. This option is currently available only for the English sentiment grammar.

  For example, the phrase *Company A is much better than Company B* might be positive or negative depending on whether you are with Company A or Company B. If you are with Company A, you can make Eduction return a match from the sentence with a positive sentiment by adding *Company A* to an entity that lists entries that you consider good.

Compile Grammars

After you write a grammar, compile the XML file into an ECR file using the Eduction command-line tool `edktool`. XML files are easy for people to read, but inefficient for computers to process. `edktool` transforms the XML file into an ECR file that is efficient for Eduction to use directly. An example of the `edktool` compile command is:

```
edktool c mygrammar.xml
```

This command produces the output file `mygrammar.ecr`.

For more information about `edktool`, see Compile and Test Grammars, on page 92.

> **NOTE:** Eduction version 12.7 and later compiles grammars to a compressed ECR format.

Example Grammar Files

The following sample grammar files contains the `gram_edk_place.xml` grammar.

grammar.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE grammars SYSTEM "../published/edk.dtd">
<!-- Sample Eduction grammar file showing all elements and attributes in the DTD -->
<grammars debug="true" case="sensitive">
  <include path="grammar_include.xml" type="private">
    <publish name="grammar2/g2e2"/>  <!-- publish previously private entity -->
```

```
  </include>
  <grammar name="grammar1" case="inherited" extend="disallow" debug="inherited">
    <extern name="grammar2"/> <!-- removes the need to refer explicitly to grammar2 -->
    <entity name="entity1" type="public" case="insensitive" extend="disallow"
debug="true">
    <!-- the following entity definitions are not useful but are provided only to
illustrate the options and combinations of elements and attributes available -->
    <pattern score=".1" case="insensitive" replace="replacechars" insert_before="prefix_
" insert_after="_suffix">cat</pattern>
    <pattern score=".2">sat</pattern>
    <entry headword="mat" score=".3" case="inherited" debug="inherited">
      <synonym case="inherited">rug</synonym> <!-- will locate rug but return mat -->
      <!-- will locate rug but return mat -->
      <synonym case="inherited"><![CDATA[carpet]]></synonym> <!-- illustrates allowing
CDATA in this element -->
      </entry>
      <entry headword="dog" score=".6"/>
      <entry>
        <headword score=".8"><![CDATA[rabbit<hi!>&abc&amp;]]></headword>
        <synonym>bunny</synonym>
      </entry>
    </entity>
    <entity name="entity2" type="public">
      <pattern>(?A:g2e1)</pattern>
    </entity>
  </grammar>
</grammars>
```

grammar_include.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE grammars SYSTEM "../published/edk.dtd">
<grammars>
   <grammar name="grammar2">
      <entity name="g2e1">
         <pattern>animal</pattern> <!-- default visibility -->
      </entity>
      <entity name="g2e2" type="private"> <!-- explicitly private -->
         <pattern>mineral</pattern>
      </entity>
      <entity name="g2e3" type="public"> <!-- explicitly public -->
         <pattern>vegetable</pattern>
      </entity>
   </grammar>
</grammars>
```

Example Grammar File to Match Months

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE grammars SYSTEM "../published/edk.dtd">

<grammars version="4.0">
   <include path="winter_names.ecr" type="private"/>
   <grammar name="example">

      <entity name="spring_month" type="private">
         <pattern>[Mm]ar(ch|\.)</pattern>
         <entry headword="April"/>
         <entry headword="april"/>
         <entry headword="Apr"/>
         <entry headword="apr"/>
         <entry headword="Apr."/>
         <entry headword="apr."/>
         <pattern replace="May">[Mm]ay\.?</pattern>
         <entry headword="June">
            <synonym>Jun</synonym>
            <synonym>Jun.</synonym>
            <synonym>june</synonym>
            <synonym>jun</synonym>
            <synonym>jun.</synonym>
         </entry>
      </entity>

      <entity name="summer_month" type="private" case="insensitive">
         <entry headword="June"/>
         <entry headword="July"/>
         <entry headword="August"/>
         <entry headword="September"/>
      </entity>

      <entity name="month" type="public">
         <pattern>(?A^spring_month)</pattern>
         <pattern>(?A:summer_month)</pattern>
         <entry headword="September"/>
         <entry headword="October"/>
         <entry headword="November"/>
         <entry headword="December"/>
         <pattern>(?A^winter_month)</pattern>
         <!-- spelling mistakes -->
         <entry score="0.5" headword="Febuary"/>
      </entity>
```

```
   </grammar>
</grammars>
```

Simplified Grammar File Containing a Dictionary of Place Names

> **NOTE:** The following grammar file is a simplified version provided for example purposes, rather than actual source code.

```
<entity name="city/spain" type="public">
    <entry headword="Barcelona"/>
    <entry headword="Ciudad Real"/>
    <entry headword="Granada"/>
    <entry headword="Madrid"/>
</entity>
<entity name="city/germany" type="headword">
    <entry headword="Berlin"/>
    <entry headword="Frankfurt"/>
    <entry headword="München"/>
    <entry headword="Leipzig"/>
</entity>
```

Simplified Grammar File Containing Patterns to Match Times of Day

> **NOTE:** The following grammar file is a simplified version provided for example purposes, rather than actual source code.

```
<entity name="time_24_hour" type="public">
   <pattern>[01][0-9]:[0-5][0-9]</pattern>
   <pattern>2[0-3]:[0-5][0-9]</pattern>
</entity>

<entity name="time_all" type="public">
   <pattern>(?A:time_24_hour)</pattern>
   <entry headword="Midnight"/>
   <entry headword="midnight"/>
   <pattern>([1-9]|10|11|12) ?[ap]\.?m\.?</pattern>
</entity>
```

Chapter 10: Compile and Test Grammars

This section describes how to use the `edktool` command-line tool to compile and test your custom Eduction grammars. It includes information about the `edktool` configuration file.

About the edktool Command-Line Tool

The `edktool` command-line tool for Eduction allows you to compile and test your grammars. You can use `edktool` to:

- compile grammars.
- list available entities in a grammar file.
- extract entities from a file based on a grammar and select entities from the grammar for extraction.
- test the accuracy of the extraction process.

> **NOTE:** On Linux, `edktool` requires the C++ library, `libstdc++.so`. To ensure the tool can locate the required library, set the Library Path:
>
> ```
> setenv LD_LIBRARY_PATH bin:$LD_LIBRARY_PATH
> ```

For a full list of the command-line options that you can use with edktool, see edktool Command-Line Options, on page 313.

Compile Grammars

You can use edktool to compile an XML file into an ECR file.

The edktool `c` (compile) command compiles a specified XML grammar into an ECR grammar file. For example:

```
edktool c mygrammar.xml
```

This command produces the output file `mygrammar.ecr`.

For more information about the compile command, see Compile, on page 316.

NOTE: Eduction version 12.7 and later compiles grammars to a compressed ECR format.

Use a Compilation Configuration File

When you compile a grammar by using edktool, you can add an optional JSON configuration file to specify additional options for compilation.

Configure Character Expansions

You can configure character expansions, which detect certain characters as if they are a different character. For example, you can detect different varieties of punctuation characters to match a standard form that you use in your grammar files.

To use character expansions, you specify an `expansions` array which contains a list of your expansions. Each array item has a `src` and `dest` element. The source and destination characters should be considered as a single list where any character in the list is expanded to any other. The character chosen as the "src" character is significant only because it is used in normalized matches in place of any "dest" character.

Consider the following example configuration:

```
{
   "expansions": [
      { "src": "a", "dest": ["b", "c"] }
   ]
}
```

If your grammar contains only the following pattern:

```
<pattern>ade</pattern>
```

Eduction expands the pattern to:

```
<pattern>[abc]de</pattern>
```

So if your input contains the following text:

```
ade bde cde dde
```

Eduction matches `ade`, `bde`, and `cde`, and produces the normalized matches `ade`, `ade`, `ade`.

If your grammar contains only the following pattern:

```
<pattern>bde</pattern>
```

Eduction expands the pattern to:

```
<pattern>[abc]de</pattern>
```

...which produces the same matches (`ade`, `bde`, and `cde`) and the same normalized matches (`ade`, `ade`, `ade`) as before.

You could use character expansions if you have written a grammar file where the patterns contain space characters, but you also want to match non-breaking spaces or other Unicode space characters.

Optimize Case-Insensitive Matching

When you have a grammar file that contains case-sensitive entities, but you want to find matches regardless of case, you can run Eduction with the parameter `MatchCase=False`. When Eduction loads a grammar file and `MatchCase=False`, it optimizes the entities for case-insensitive matching to improve run-time performance. However, this can increase the time required to initialize Eduction, so if you regularly use a grammar file with `MatchCase=False` you can optimize the entities for case-insensitive matching at compile-time instead.

To optimize entities for case-insensitive matching, set the option `alternativeCaseArcs` to `true` in your compilation configuration file:

```
{ "alternativeCaseArcs": true }
```

After compiling a grammar file with this option, you can still use the `MatchCase` parameter to choose whether matching is case-sensitive.

To obtain the best case-insensitive performance, you should write a grammar file using only upper or lower case and then normalize the input by setting the `CaseNormalization` parameter. For more information, see Case Normalization, on page 58. Compiling a grammar file with `alternativeCaseArcs` set to `true` is useful if you cannot easily modify your grammar file(s), but only reduces the time required to initialize Eduction (it does not reduce the time required for matching).

To recompile an existing grammar file with `alternativeCaseArcs` set to `true` you could include the existing grammar in a new grammar file as shown in the example below, and then compile the new grammar using `edktool`.

```
<?xml version="1.0" encoding="UTF-8"?>
<grammars version="4.0">
  <include path="published/grammar.ecr" type="public"/>
</grammars>
```

Use the Configuration File

You add a configuration file to your compilation by setting the -c command-line option in the compile command. For more information, see Compile, on page 316.

When you compile a grammar by using the Eduction SDK, you can specify the path to a compilation configuration file by using one of the following options:

- C API: the `EdkLoadResourceFileWithCompileConfig` and `EdkLoadResourceBufferWithCompileConfig` functions.
- Java API: the `loadResourceFile`, `loadResourceFiles`, and `loadResourceBuffer` methods in the `TextExtractionEngine` interface.
- .NET API: the `GetCompiler` method on the `EDKFactory` class.

For more information, refer to the API documentation.

Assess and Measure Eduction Grammars

There might be times when you want to check the effectiveness and performance of your .ECR or .XML grammars.

For example, you might want to check:

- how effective a particular grammar file is at extracting the entities you require from your data.
- how a change to a grammar file affects performance.

The edktool command-line tool has two features, *Assess* and *Measure*, that enable you to find out this kind of information easily (for a full reference for these functions, see Assess, on page 313 and Measure, on page 322).

Assess

This feature takes a list of phrases that you expect to contain matches, and checks whether they do contain a match.

Alternatively, it can take a list of phrases that you do not expect to contain matches, and check that they do not contain a match.

You can use this feature to:

- test the suitability of an Eduction grammar for the task you would like it to do.
- monitor the accuracy of an Eduction grammar while you develop it.
- ensure that further development does not introduce problems to an Eduction grammar that already performs well.

You can set up an assessment by using the following three-stage procedure.

Create the Input Files

Create a *valid* file, which contains an expected match in each line. The expected match can be from either your real data, or from artificial sample data. For example:

```
My mate is Bob Smith
My name is Bob Smith
My mate is Bob Smith
Barbara smith
smith,benjamin
SMYTH, Robert
Dr Bob B. Smith Jr.
Bob SMITH lives here
(etc.)
```

You can also create a list of valid exact matches (examples of text that must be matched in their entirety).

Alternatively, you can create an *invalid* file, where each line must contain no match. For example:

```
Black Smith
Bob up and down
smith, smyth
She is called Barbara. Smith is not her surname.
Benjamins myth
(etc.)
```

You can also create a list of non-valid exact matches (examples of text that might or might not contain a match, but must not be matched in their entirety).

Alternatively, you can set up your input file so that it refers to matches by all available entities, or only by specific entities (for example, `male_name_all`).

> **NOTE:** Eduction ignores blank lines in the input file, and lines that start with `//`.

Update the Configuration File

You can run an extraction from the command line without a configuration file, but in most cases it is easier to use one.

You must add the assessment sections to an Eduction file that would otherwise run a successful extraction. Each assessment must contain either a valid input file, an invalid input file, or both. You must number multiple sections consecutively, starting from 0 (zero).

```
[assessment0]
Valid=my_valid_1.txt

[assessment1]
Valid=my_valid_2.txt
Invalid=my_invalid_1.txt
Exact=True
Entities=my_entity,my_other_entity

[assessment2]
Invalid=my_filename.txt
Exact=False
Entities=my_other_entity
```

To require exact matches, set the `Exact` parameter to **`True`**.

To restrict the extraction to a subset of the available entities, you can:

- set the `Entities` parameter in the assessment section to a comma-separated list of the entities you want to extract.
- set the `ResourceFiles` parameter in the standard CFG configuration file.

Run the Assessment

To run the assessment, open a command prompt in the edktool directory and type:

```
edktool a -l <license> -c <config> [-o <output>] [-a]
```

Alternatively, to run a single assessment section using the command line and no configuration file type:

```
edktool a -l <license> -g <grammars> [-e <entities_for_extraction>] [-x] [-o
<output>] [-a] [-m <entities_for_matching>] [-v <valid_input> | -w <invalid_input>]
```

The following table lists the command-line parameters that you can use, and their functions.

Parameter	Function
`-x`	Equivalent to setting `exact=True` in the configuration file.
`-m`	Equivalent to including the `entities=` parameter in the assessment section of the configuration file.
`-v`	Equivalent to including the `valid=` parameter in the configuration file.
`-w`	Equivalent to including the `invalid=` parameter in the configuration file.
`-a`	Includes all examples in the results, including those which passed the assessment.
`-o`	Sends the results to a specific file, instead of to the console by default.

Use the Assessment Results

The assessment results contain:

- a list of all the examples that did not behave as expected.
- all relevant statistics (some statistics are relevant only if you specified both a valid file and an invalid file).

If all the tests in the assessment pass successfully, the grammar file is working as expected on the data you have given it. In practice, there are usually some failures.

Often there are common themes running through the failures. Perhaps the grammar file only matches text with certain capitalization, or where the data is written in a certain format. This can provide immediate information on how to fix the issues in cases where you have access to the XML source.

When there is a failure that you do not understand, you might find it useful to expand the valid or invalid grammar file with a selection of similar examples, perhaps with different words, formats, capitalization, or punctuation. After you rerun the consistency test with expanded data, and it might become much clearer what the problem is.

Statistics

The results include statistics for recall (if a valid file is present) and true negative rate (if an invalid file is present). If both types of file are present in a single assessment section, it also includes statistics for accuracy, precision, and F1 score (see Results Relevance, on page 68).

The statistics depend on the data provided in the valid or invalid files. In all cases, the statistic falls in the range 0.0000 - 1.0000; a higher score represents a more successful grammar file.

After your initial assessment run, if you make modifications to the grammar file you can use these statistics to compare the performance of the old and new grammars.

> **NOTE:** If you make improvements to a grammar file based on the results of your assessments, these improvements are targeted at the data provided. These statistics are likely to be over-inflated compared to the statistics for generic data.

You can use an assessment that has examples that all pass perfectly as a basis for further development. If the statistics in the assessment drop beneath 1.0000, you can identify that the development has introduced a problem to the grammar file.

Measure

This feature compares the matches found in separate extraction runs on the same input data. You can run this feature from Eduction to view any differences between the two extraction runs. The results are in XML format, and include the metadata.

You can use this feature to:

- monitor the improvement of a grammar file that originally returned too many false matches.
- compare the results of extraction by different versions of a grammar file on the same data to test whether modifications are beneficial.

To set up this feature:

1. Run an extraction using one version of the grammar file. Your input data must be in plain text format.
2. Save the output.
3. (Optional) Remove any false matches from the output by deleting the three XML lines corresponding to the false match. This step is appropriate only if you want to form a benchmark output file with the aim of developing the grammar file to produce output very similar to the benchmark.

   You can also modify existing matches by making them shorter or longer, or you can add entirely new matches that should be found in the input text.

   > **NOTE:** You must ensure that you specify the correct offset (in bytes) when you modify or add matches.

4. Run an extraction on the same plain text input as before, using the newer version of the grammar file. Save the output under a different file name.

5. At the command line, run the Measure command:

   ```
   edktool m -e <expected_file> -a <actual_file> [-o <output_file>]
   ```

Use the Measure Results

The output is an XML document which lists all the differences between the output of `<expected_file>` and the output of `<actual_file>`. You can use the list of differences to monitor how changes to the XML source affect the output on real data. You can then decide which changes are beneficial, and which are not.

Statistics

The Measure results include statistics for precision and recall, although these are relevant only when the expected file is a benchmark, and the aim is to produce a grammar file whose output is as close as possible to this benchmark.

Eduction Configuration File

You define configuration settings for edktool in an IDOL Server format CFG configuration file. For details of the settings that you can use in this file, see Eduction Parameter Reference, on page 327.

The CFG configuration file consists of several sections, which you identify by using a phrase in square brackets. Each section contains parameters (name/value pairs). For example:

```
[Eduction]
ResourceFiles=C:\MyGrammar\gram1.ecr
```

To define Eduction settings in the configuration file

1. Open the CFG configuration file in a text editor.
2. Set the Eduction parameters as required. See Eduction Parameter Reference, on page 327 for more information.
3. Save and close the configuration file.

Modify Configuration Parameter Values

This section describes how to enter parameter values in the configuration file.

Enter Boolean Values

The following settings for Boolean parameters are interchangeable:

```
TRUE = true = ON = on = Y = y = 1
```

```
FALSE = false = OFF = off = N = n = 0
```

Enter String Values

To enter a comma-separated list of strings when one of the strings contains a comma, you can indicate the start and the end of the string with quotation marks, for example:

ParameterName=**cat,dog,bird,"wing,beak",turtle**

Alternatively, you can escape the comma with a backslash:

ParameterName=**cat,dog,bird,wing\,beak,turtle**

If any string in a comma-separated list contains quotation marks, you must put this string into quotation marks and escape each quotation mark in the string by inserting a backslash before it. For example:

ParameterName=**"<font face=\"arial\" size=\"+1\"><b>","<p>"**

Here, quotation marks indicate the beginning and end of the string. All quotation marks that are contained in the string are escaped.

Sample Configuration File

The following shows the configuration for a sample Eduction task:

```
[Eduction]
ResourceFiles=C:\MyGrammar\gram1.ecr,C:\MyGrammar\gram2.ecr
ZoneStart0=<TEXT>
ZoneEnd0=</TEXT>
ZoneStart1=acknowledgements
ZoneEnd1=introduction
Entity0=common/aus_holidays
EntityField0=HOLIDAYS
EntityZone0=0
Entity1=common/us_holidays
EntityField1=HOLIDAYS
EntityZone1=0
Entity2=us/social_security_number
EntityField2=SS_NUMBER
EntityZone2=1
SearchFields=DRECONTENT

[Logging]
LogLevel=Full
```

This sample uses two grammar files. It searches for all Australian and U.S. holidays in the `DRECONTENT` field between the text *<Text>* and *</Text>*, adding the matches as additional fields `HOLIDAYS`. It also searches for a single social security number in `DRECONTENT` between the text *acknowledgements* and *introduction* and adds the results as a new field `SS_NUMBER`.

Chapter 11: Pre-Filter Tasks

This section describes pre-filter tasks.

Introduction

Pre-filtering allows you to narrow down the amount of input text that Eduction processes for a particular set of entities. With pre-filtering, Eduction performs an initial quick matching step that finds sections of text that contain likely matches, rather than running the full match on the whole input.

Pre-filtering text can improve performance for some entities, when there is a broad way to find a potential match without either over-matching too much of the input text, or eliminating potential valid matches.

The quick matching step can either match text by using a regular expression (regex) that you configure, or a dictionary of terms.

For example:

- To match addresses, you can use regex pre-filtering to find numbers in the text (which might correspond to house numbers or postal codes).
- To match names in CJKVT languages (where there is a regular set of surnames, and the Eduction grammars do not attempt to find values that are not already listed as names), you might use a dictionary pre-filter. In this case you perform a quick match to find the surnames, and then the full match finds the full name.

The pre-filtering method is less useful for entities that match a long list of possible words, when there is no simple regular expression or dictionary of terms that matches all your possible entities. For example, for English names the Eduction grammars attempt to match plausible names as well as recognized ones, so there is no way to pre-filter without eliminating potential matches.

Configure a Pre-Filter Task

For each pre-filter task that you want to configure, you set:

- a regular expression that specifies how to find potential matches, or a resource file that provides a dictionary of terms to use for fast matching.
- the amount of text Eduction must use on either side of the potential match to find the more detailed match.

NOTE: Eduction runs all your configured pre-filtering tasks for all input text, so ensure that your pre-filter task applies to all your configured grammars and entities. Use a different configuration for any entities that you do not want to pre-filter.

To configure a pre-filter task

1. In the `[Eduction]` section, add a `PreFilterTaskN` parameter, where *N* is a number starting from 0 for the first task. Set this parameter to the name of a configuration section where you define your pre-filter task.
2. Create the new configuration section.
3. Set one of the following parameters:
   - `Regex` to a regular expression value that finds potential matches in your text.
   - `ResourceFile` to the name of a DPF or JSON file that contains the dictionary of terms to use for pre-matching.
4. Set `WindowCharsBeforeMatch` and `WindowCharsAfterMatch` to the number of characters before and after the potential match segment to use as the match window.
5. Optionally set other parameters to exclude non-valid values or end processing early in certain conditions, such as `Exclusion`, `InvalidRegexAfterMatch`, `InvalidRegexBeforeMatch`, and `PrefilterMaxReturnedBytes`. For more information, see Eduction Parameter Reference, on page 327.
6. Save and close your configuration file.

For example:

```
[Eduction]
PrefilterTask0=AddressPrefilter

[AddressPrefilter]
Regex=\d{1,7}
WindowCharsBeforeMatch=100
WindowCharsAfterMatch=100
```

For more details about these parameters, see Eduction Parameter Reference, on page 327.

TIP: To use pre-filtering tasks through the C and Java Eduction APIs, you must create your Eduction engine from a configuration file. See Standalone API Usage, on page 28 (C) or Standalone API Usage, on page 33 (Java).

Dictionary ResourceFile Format

For dictionary pre-filtering, you can either use a provided dictionary file, or create one yourself. The provided dictionary files are in binary format, and have the extension DPF. You can create custom files in JSON format, which must use the following JSON schema:

```
{
   "id": "eduction#DictionaryPrefilter",
   "$schema": "http://json-schema.org/draft-07/schema",
   "description": "Schema for eduction dictionary prefilter serialization",
   "type": "object",
   "properties": {
      "type": {
         "description": "Type of prefilter",
         "enum": [
            "dictionary"
         ]
      },
      "dictionary_words": {
         "description": "Array of words to use as the dictionary for the prefilter",
         "type": "array",
         "items": {
            "type": "string",
            "minLength": 1
         },
         "minItems": 1,
         "uniqueItems": true
      }
   },
   "additionalProperties": false,
   "required": [
      "type",
      "dictionary_words"
   ]
}
```

For example:

```
{
   "type": "dictionary",
   "dictionary_words": [
      "Smith",
      "Jones"
   ]
}
```

This simplified example pre-filters by finding any instance of the words *Smith* or *Jones* in the text. It then creates a text window around these simple matches, which it uses to perform the full match.

> **NOTE:** Dictionary terms are case-sensitive. If you want to include multiple case options, you must add them all to the dictionary.

Chapter 12: Post-Processing

This section describes post-processing.

Introduction

Post-processing performs additional processing on the matches that Eduction finds.

A common use for post-processing is to validate matches. You can validate some entities, such as credit card numbers, by calculating a checksum. You can discard any matches with an invalid checksum, because even though it matches the correct format, it cannot be genuine. If a match has a valid checksum then you might increase its score, because it is likely to be valid.

Another common use for post-processing is to normalize the output from Eduction. For example, if you extract monetary values, Eduction might find matches that look like "£5.3 million" or "£25". You can use post-processing to normalize these values to "£5,300,000" and "£25", so that IDOL Content or another application can compare and sort the values correctly.

An Eduction post-processing task runs a Lua script that you configure. Eduction passes the matches it finds into a Lua function, either one at a time or all at once (*en masse*). See Write a Lua Script for Post-Processing, on page 107.

Configure Post-Processing in Eduction Server

In Eduction Server, you configure post-processing tasks by setting one or more `PostProcessingTask`*`N`* parameters in the `[Eduction]` section of the configuration file, where *`N`* is a number starting from zero. For example:

```
[Eduction]
PostProcessingTask0=ValidateWithChecksum
PostProcessingTask1=FilterScore
```

Then, create a section for each of the tasks using the names that you defined:

```
[ValidateWithChecksum]
Type=lua
Entities=number/creditcard
```

```
Script=./scripts/checksum.lua
ProcessEnMasse=FALSE
```

The `Entities` parameter specifies the entities to process. You can use wildcards to match multiple entities. The `Script` parameter specifies the path to the Lua script that you want to run. Eduction includes some example scripts, and you can write your own. For information about how to write a post-processing script, see Write a Lua Script for Post-Processing, on the next page.

For more information about Eduction Server configuration parameters, refer to the *Eduction Server Reference*.

Post-Processing with the Eduction API

To perform post-processing in an application built on the Eduction API, add post-processing tasks to the Eduction engine. You can add the tasks:

- when you create the engine, if you construct the engine by supplying a configuration. When you create an Eduction engine from a configuration the configuration automatically adds any post-processing tasks in the configuration to the engine.

  See `EdkEngineCreateFromConfigFile` (C API), the `EdkEngine` constructor (Java API), or `EDKFactory::GetTextExtractionEngine` (.NET API).

- after you create the engine, by calling the appropriate function:
  - `EdkEngineAddPostProcessingTask` in the C API.
  - `addPostProcessingTask` in the Java API.

The following functions allow you to specify a minimum score that a match must have for it to return in the results after all post-processing tasks have completed:

- `EdkEngineSetPostProcessingThreshold` in the C API.
- `setPostProcessingThreshold` in the Java API.

The matches returned by the `EdkGetNextMatch` function in the C API, or by iterating over the matches in the Java API, reflect any modifications made by post-processing. If a post-processing task discards a match or its score does not meet the threshold you have specified, it is not returned at all.

If you configure a post-processing task that processes matches en masse, the API does not return matches until all input has been received. This is necessary because an en masse post-processing task requires all of the matches at the same time.

The Eduction SDK includes reference documentation for the API. For more information about the SDK, see Eduction SDK Package, on page 25.

Write a Lua Script for Post-Processing

An Eduction post-processing task runs a Lua script. Eduction passes the matches into an entry function.

There are two available entry functions to use when you process single matches:

- `processmatch`. This function allows you to modify a match, change the score, or discard a non-valid match. You can use this option for most post-processing, such as checksum validation and normalization.
- `finalizematch`. This function allows you to add new matches into the Eduction session. For example, you might use this option to combine existing matches, and return the combined match as a result. This function can also still perform the same changes as `processmatch`.

Your script must define at least one of these functions.

> **NOTE:** If you define both `processmatch` and `finalizematch`, `processmatch` takes precedence.

There are also two equivalent functions for en masse processing, `processmatches` and `finalizematches`. For more information, see Process Matches En Masse, on the next page.

Use ProcessMatch

The `processmatch` function must accept an `edkMatch` object (the current match) as its first argument. When you run Eduction through the API, the function can also accept a user parameters map as an optional final argument (see Pass Parameters into the Lua Script, on page 109).

Eduction passes matches into the script one at a time. The script must return a Boolean value: `true` to keep the match or `false` to discard it.

The following example changes the score for every match to `0.5`:

```
function processmatch(edkmatch)
    if edkmatch then
        -- change the score for the match
        edkmatch:setScore(0.5)
    end
    return true
end
```

Use FinalizeMatch

The `finalizematch` entry function must accept an `edkMatch` object (the current match) as its first argument, and a session handle as its second argument. When you run Eduction through the API, the function can also accept a user parameters map as an optional final argument (see Pass Parameters into the Lua Script, on page 109).

The following example modifies an entity to append the value `Esq.`, and injects the match back into the session.

```
function finalizematch(edkmatch, session)
   if edkmatch then
      local text = edkmatch:getOutputText()
      m = LuaEdkMatch:new(edkmatch:getEntityName(), text .. " Esq.",
edkmatch:getOffset())
      session:injectMatch(m)
      return true
   end
   return false
end
```

After you inject a match, the session takes ownership of it, so you cannot use the created match in any subsequent functions.

> **NOTE:** You cannot perform additional post-processing on injected matches. Eduction skips these matches at post-processing time, to prevent infinite loops.

Process Matches En Masse

Sometimes, you might prefer to process all the matches together. For example, you might want to increase the scores of matches that appear near other matches. It is easier to do this if you process all the matches at the same time.

To process all the matches at the same time, set the `ProcessEnMasse` parameter to `TRUE` in your Eduction configuration. When `ProcessEnMasse=TRUE`, Eduction passes all the matches it finds into the script together.

Your script for processing matches en masse must define a function either named `processmatches`, or `finalizematches`.

The `processmatches` function must accept a Lua table of `edkEnMasseMatch` objects as its first argument. Each of these objects represents a single match, but you must call the `getMatch` method to obtain an `edkMatch` object. You can then use the `edkMatch` object to manipulate the match. If you want to discard a match, call the method `setOutput` on the relevant `edkEnMasseMatch` object.

The following example demonstrates how to iterate over the elements in the table and discard any match with a score that is less than `0.5`:

```
function processmatches(matches)
    -- example that discards matches with score < 0.5
    for k,v in ipairs (matches) do
        local edkmatch = v:getMatch()
        if edkmatch:getScore() < 0.5 then
            v:setOutput(false)
        end
    end
end
```

The `finalizematches` function must accept a Lua table of `edkEnMasseMatch` objects as its first argument (the same as for `processmatches`), and a session handle as its second argument.

When you run Eduction through the API, the `processmatches` or `finalizematches` functions can also accept a user parameters map as an optional final argument (see Pass Parameters into the Lua Script, below).

For information about the objects and methods that you can use in your Lua post-processing scripts, see Eduction Lua Methods Reference, on page 377.

Reset a Session

You can define an additional function, `resetprocessor`, which Eduction calls whenever the session resets (for example, when it receives a new input stream, or more text after it has already processed a final input).

You can use this function if your script maintains a global state with details of previous matches. When you reuse the session to process another document or input buffer, you can use `resetprocessor` to reset the state. Eduction passes in the current user parameters (see Pass Parameters into the Lua Script, below) to the reset hook, if required.

The following simple example shows a script that replaces the normalized text for each match with a count. It resets the count when you reuse your `EdkSession`.

```
local count = 0

function resetprocessor (params)
    count = tonumber(params['startcount']) or 0
end

function processmatch (edkmatch)
    if edkmatch then
        count = count + 1
        edkmatch:setOutputText(count)
    end
    return true
end
```

Pass Parameters into the Lua Script

You can pass additional parameters into post-processing tasks that you run through the Eduction API. To add an additional parameter (to all post-processing tasks that run during the session), call the appropriate function:

- `EdkSessionSetUserParamValue` in the C API.
- `ITextExtractionSession::SetUserParameter` in the .NET API.
- `setUserParamValue` in the Java API.

Any parameters that you set using these functions are passed into the `processmatch`, `processmatches`, `finalizematch`, or `finalizematches` function of the Lua script as a table of key-value pairs. For example:

```
function processmatch(edkmatch, params)
    for k,v in pairs (params) do
        --print ("Custom parameter ", k, " has value ", v)
    end
    return true
end
```

> **TIP:** Some of the Lua post-processing scripts available in the Eduction packages provide optional user parameters. You can review these scripts for the available parameters, and pass them into the task as required. See Example Scripts, below.

Example Scripts

Eduction includes the following example post-processing scripts.

Checksum Validation

The `checksum_luhn.lua` script verifies the checksum digit of each match using the *Luhn algorithm*, and reduces the score associated with the match if the checksum is wrong. The `checksum_luhn_enmasse.lua` script performs checksum validation as an en masse processing task, discards incorrect matches, and alters the score of correct matches to equal the proportion of matches that have the correct checksum digit.

You can use these scripts with the `number_cc.ecr` and `number_sin_ca.ecr` grammar files to validate most credit card numbers.

Spanish Identity Card Number Validation

You can use the `checksum_dni_es.lua` script with the `number_dni_es.ecr` grammar file to validate Spanish Documento Nacional de Identidad (national identity card) numbers.

Dutch Citizen Service Number Validation

You can use the `checksum_bsn_nl.lua` script with the `number_bsn_nl.ecr` grammar file to validate Dutch Citizen Service Numbers (Burgerservicenummer, or BSNs).

Geographical Coordinate Standardization

You can use the `lat_long.lua` script with the `place_lat_long.ecr` grammar file to convert and standardize the output of geographical coordinates.

Date and Time Standardization

You can use the `datetime.lua` script with the `datetime_advanced_eng.ecr` grammar file to convert and standardize the output of dates and times (and ranges) in English into a standardized format in cases where there are matches on several formats. For example, you can convert both *23/11/13* and *Nov 23 2013* to *2013-11-23*.

The `datetime_advanced_eng.ecr` grammar file can understand English natural language dates, and relative dates such as *last Saturday morning*. You can optionally provide a reference date for `<today>` in the Lua script to customize normalization of relative dates into standard formats, by using the following user parameter:

`refdate`	A date in ISO YYYY-MM-DD format. If you do not set `refdate`, Eduction uses the current date as `<today>`.

For date and time range matches, this script sets the normalized text to `<start>/<end>`, and additionally adds `STARTPOINT` and `ENDPOINT` components that contain the associated dates or times. When there is a multiple date match (for example, *5th and 8th July* matches as *5th July* and *8th July*), the script returns a comma-separated list, with a POINT component for each date.

For contextual date matches (such as *two days after*), the script includes the following optional parameters, which allow you to discard matches where the closest contextual date is too far away from the match:

`contextmaxdistance`	Discard contextual matches that occur more than this many characters from the last date match.
`contextpenaltydistance`	Reduce the score for matches that lie between this distance and the maximum cut-off (`contextmaxdistance`). This optional applies a linear reduction, which scales to zero at the maximum distance.

Filter Matches by Case

You can use the `case_filter.lua` example script to filter out matches by case, for example in personal name grammars.

To use this option, you must set `MatchCase` to **`False`** for the grammar. The script filters out any match that is not one of:

- an exact match as specified in the grammar.
- an upper case match (for example, JANE SMITH).
- a title case match (for example Jane Smith).

> **NOTE:** You might need to update this script to include case mappings for uncommon non-ASCII characters. The script provides sample mappings for common Latin characters with diacritics.

Part V: Reference

This section contains reference materials for Eduction.

- Standard Grammars
- Grammar Format Reference
- edktool Command-Line Options
- Eduction Parameter Reference
- Eduction Lua Methods Reference

Chapter 13: Standard Grammars

This chapter contains specific information concerning the standard grammars that come with Eduction.

File Names

File names consist of up to four parts, separated by underscores:

- **Basic entity type**. For example, *place*, *number,* or *person*.
- **Further detail on the basic type**. For example, *malefirstname* or *ss* for Social Security number. This part is optional.
- **Language**. The three-character ISO 639-2/B code in which the grammar was written. For example, *eng* for English.
- **Country**. The two-character ISO 3166-1 code describing the country for which the grammar was written. For example: *us* for the United States. This part is optional if the grammar does not target a specific country (for example, a credit card number).

NOTE: Entity names follow the same four-part structure, except for the basic type. The further detail and language/country parts are separated by forward slashes. The language code and the optional country code are concatenated.

Sentiment Grammars

Eduction includes standard grammars designed to identify those phrases in a passage of text that indicate positive or negative sentiment. These grammars can also identify which sentiments are expressed for which topics.

The sentiment grammar files have *lite* versions. The lite versions are identical to the full versions in most respects, but they do not support components or user modification. They can process data up to twice as fast as the full versions, depending on language.

Micro Focus recommends that you use the lite versions except when you need to use components or modify the built-in dictionaries.

The lite grammars have the same name as the full version, with `_lite` after the language. For example, the file name of the Chinese sentiment grammar file is `sentiment_chi.ecr`, and the file name of the lite version is `sentiment_chi_lite.ecr`.

All sentiment analysis grammar files except `sentiment_basic_eng.ecr` support components. You can extract the SENTIMENT and TOPIC components in most matches.

Place Name Disambiguation

Ambiguous names in all place grammars have been given a score of 0.98 so that you can filter them out by setting `EntityMinScore`*`N`* to **`0.99`**. For example, if you want to use the `place/state/engau` entity to extract Australian state names using the `place_engau.ecr` grammar file, you can set `EntityMinScore`*`N`* to **`0.99`** to filter out ambiguous names such as *Victoria*.

Standard Grammar – Compiled

The following sections list the compiled grammar files included with Eduction.

> **NOTE:** All the Chinese grammar files support traditional Chinese.

A

address_au.ecr

Entity	Description
address/postcode/au	Australian postal codes. For example, *2600*.
address/state_postcode/au	Australia state or territory, and postal code. For example, *NSW 2060*.
address/city_state_postcode/au	Australian city, state or territory, and postal code. For example, *North Sydney, NSW 2060*.
address/au	Any Australian address. For example: *Shop 17, Winnellie Shopping Centre, 347 Stuart Hwy, Winnellie, NT, 0820*. *P.O.Box 27, Armadale North, Victoria, 3143, AUSTRALIA*. *121 North Seal Way, Cocos Keeling Islands, WA, 6799*. Eduction supports all common delimiters, including newlines.

address_ca.ecr

Entity	Description
address/postcode/ca	Canadian postal codes. For example, *T2P-0B4*, *T2P0B4*, or *T2P 0B4*.
address/region_postcode/ca	Canadian province or territory, and postal code. For example, *Alberta, T2P0B4*.
address/city_region_postcode/ca	Canadian city, province or territory, and postal code. For example, *Calgary, Alberta, T2P 0B4*.
address/ca	Any Canadian address. For example: *240 4th Avenue S.W., Suite 600, Calgary, Alberta T2P 4H4, Canada*. *124 Av de la Peine, Montreal QC, H3Z 2Y7*. *Suite 600, 222-3rd Ave S.W., Calgary Alberta, T2P 0B4*. Eduction supports all common delimiters, including newlines.

address_cn.ecr

Entity	Description
address/pc/chicn	Chinese postal code. For example, *266033*.
address/chicn	Any Chinese address. For example, 中国 , 山东省 , 青岛市 香港东路6号 , 5号楼 , 8号室 李小方 (先生) 收.
address/engcn	A Chinese address in English. For example. *63 Renmin Lu, Qingdao Shi, 266033 Shandong, China*.
address/cn	A Chinese address in Chinese or English.

address_de.ecr

Entity	Description
address/postcode/de	German postal code. For example, *80639*.
address/postcode_city/de	German postal code, and city. For example, *80639, München*.
address/de	Any German address. For example: *Hewlett-Packard-Straße 1, 61352, Bad Homburg vor*

address_de.ecr, continued

Entity	Description
	der Höhe. *Postfach 10 01 65, 32547, Bad Oeynhausen, GERMANY*. *Grüner Weg 6, 61169, Friedberg, GERMANY*. Eduction supports all common delimiters, including newlines.

address_eng.ecr

Entity	Description
address/strnum/eng	Street numbers. For example, *12a* or *14-17B*.
address/pobox/eng	Post office box numbers. For example, *PO Box 26*.
address/pmb/eng	Private mail box number. For example, *Private Mail Box 26*.
address/pmb_or_pobox/eng	Post office box or private mail box number.
address/street_pre/eng	Special street type that prefixes street numbers. For example, *Highway Contract, HC*.
address/street_hwy/eng	Highway. For example, *City Route*.
address/street_grid/eng	Grid address. For example, *400W350N*.
address/street/eng	A street. For example, *Cowley Road* or *5th Street NW*.
address/street_corner/eng	A street corner. For example, *Corner King Street & Queen Street*.
address/street_all/eng	Any street For example, *12a Carlisle Lane*.
address/suite/eng	Suite number. For example, *Suite 1*.
address/floor/eng	Floor or level number. For example, *3rd Floor*, *Second Floor*, *Level 8*.
address/floor_or_suite/eng	A floor or suite number.
address/unitshipmil/eng	A military address analogous to a street address.
address/building/eng	A building. For example, *Spear Tower*.

address_es.ecr

Entity	Description
address/postcode/es	Spanish postal code. For example, *19208*.
address/postcode_city/es	Spanish postal code and city. For example, *19208 Guadalajara*.
address/es	Any Spanish address. For example: *Av. de las Cortes de Cádiz, s/n, C. C. El Corte Inglés, 11011, Cádiz*. *Avda. Alfonso XIII, 6, Santander, España*. *Calle de la Fundición, 3, 33206, Gijón, Spain*. Eduction supports all common delimiters, including newlines.

address_fr.ecr

Entity	Description
address/postcode/fr	French postal codes. For example, *75008*.
address/postcode_city/fr	French postal code, city, and optional CEDEX. For example, *75008, Paris*.
address/fr	Any French address. For example: *3, Avenue Denis Semeria, Saint-Jean-Cap-Ferrat, Provence-Alpes-Côte d'Azur, 06230, France*. *950 route des Colles - BP 27, 06901 Valbonne Sophia Antipolis*. *Bât G1 147 r Oberkampf, 75011 PARIS*. Eduction supports all common delimiters, including newlines.

address_fre.ecr

Entity	Description
address/strnum/fre	A street number. For example, *12a* or *14-17B*.
address/pobox/fre	Post office box number in French. For example, *Boite Postale 26*.
address/park/fre	A business park in French. For example, *Technopark de Marseille*.

address_fre.ecr, continued

Entity	Description
address/building/fre	A building. For example, *Château de Chambord*.
address/delivery_point/fre	A delivery point in French. For example, *BÂTIMENT 15*.
address/street_type/fre	A street type in French. For example, *Rue*.
address/street/fre	A street in French. For example, *Rue Pierre Charron*.
address/street_all/fre	Any street in French.
address/house_type/fre	A house type in French. For example, *Residence*.

address_gb.ecr

Entity	Description
address/postcode/gb	United Kingdom postal codes. For example, *GY9 3UX*.
address/city_county_postcode/gb	UK city, optional county/country name, post code, and optional place name. For example, *Cambridge, CB4 0WZ*.
address/gb	Any United Kingdom address. For example: *Cambridge Business Park, Cowley Road, Cambridge, CB4 0WZ*. *12-14 The Diamond, Londonderry, Northern Ireland, BT48 6HW*. *105 Piccadilly, (First Floor), London, W1J 7NJ*. *Unit D, Acorn Business Park, Ling Road, Tower Park, Poole, Dorset, BH12 4NZ*. *44 Dorset Road, Providenciales, TURKS AND CAICOS ISLANDS*. Eduction supports all common delimiters, including newlines.

address_ger.ecr

Entity	Description
address/strnum/ger	A street number. For example, *12a*.

address_ger.ecr, continued

Entity	Description
address/pobox/ger	A post office box number in German. For example, *Postfach 26*.
address/street/ger	A street in German. For example, *12 Romanstr*.

address_it.ecr

Entity	Description
address/postcode/it	Italian postal code. For example, *12345* or *IT-98765*.
address/postcode_city/it	Italian postal code and city. For example, *52100 Arezzo*.
address/it	Any Italian address. For example: *Strada del Masarone 67, 13900 Biella (MI)*. *Via Balbi 3 e 40 16126 Genova*. *Via Mascarella n° 21/3, 40131 Bologna, Italia*. Eduction supports all common delimiters, including newlines.

address_ita.ecr

Entity	Description
address/strnum/ita	Italian street number. For example, *12a*.
address/pobox/ita	A post office box number in Italian. For example, *Casella postale 26*.
address/street_type/ita	A street type in Italian. For example, *Via* or *Lungomare*.
address/street/ita	An entire street name in Italian. For example, *Via del Fosso de Dragoncello*.

address_jp.ecr

Entity	Description
address/postcode/jp	Japanese postal code. For example, *108-0023*.

address_spa.ecr

Entity	Description
address/strnum/spa	A street number. For example, *12a* or *14-17B.*
address/pobox/spa	A post office box number in Spanish. For example, *Apartado de correos 26*.
address/street_type/spa	A street type in Spanish or in another language spoken in Spain. For example, *Calle* or *Passeig*.
address/street_name/spa	A Spanish name that may refer to a street. For example, *26 de Marzo de 1824* or *Trujillo*. These are often used for street names in South America without a street type such as *Calle*.
address/street/spa	An entire street name in Spanish. For example, *Calle de La Habana*.
address/business_area/spa	A shopping center or business park in Spanish. For example, *Parque Tecnológico de Andalucía*.

address_us.ecr

Entity	Description
address/zipcode/us	U.S. ZIP codes. For example, *94070-1234.*
address/city_state_zipcode/us	U.S. city, state, and ZIP code. For example, *Chicago, IL 80803.*
address/military/us	U.S. military address. For example, *Unit 45013, Box 2666, USAG J, APO AP 96338.*
address/us	Any U.S. address. For example: *30 South Wacker Drive, 22nd Floor, Chicago, IL 60606.* *P.O. Box 29, Sometown, AL 12345.* *5758 West Las Positas Blvd, Suite 100, Pleasanton, CA 94588.* *1 Market Street, Spear Tower, Suite 1900, San Francisco, CA 94105.* Eduction supports all common delimiters, including newlines.

age_eng.ecr

Entity	Description
age/all/eng	An age in English.

age_fre.ecr

Entity	Description
age/all/fre	An age in French.

B

bank.ecr

Entity	Description
bank/engca	Canadian banks. For example, *Canadian Imperial Bank of Commerce*.
bank/engb	UK banks. For example, *HSBC*.
bank/engus	U.S. banks. For example, *Morgan Stanley*.

C

company_chicn.ecr

Entity	Description
company/all/chicn	Any Chinese company.

company_dutnl.ecr

Entity	Description
company/top500/dutnl	Top 500 Dutch companies.
company/designator/dutnl	Dutch company identifiers.

company_engau.ecr

Entity	Description
company/law/engau	Law firms in Australia.

company_engca.ecr

Entity	Description
company/tsx60/engca	A Canadian TSX60 company.
company/TSXVenture50/engca	A Canadian TSX Venture 50 company.
company/all/engca	Any Canadian company. This entity includes all companies matched by the other entities in this section, as well as several hundred other significant companies.

company_enggb.ecr

Entity	Description
company/LSE/enggb	A United Kingdom company listed on the London Stock Exchange.
company/law/enggb	Law firms in the United Kingdom.
company/ftse100/enggb	A FTSE 100 United Kingdom company.
company/all/enggb	Any United Kingdom company. This entity includes all companies matched by the other entities in this section, as well as dozens of other significant companies.

company_engjp.ecr

Entity	Description
company/nikkei225/engjp	A Nikkei225 Japanese company.
company/all/engjp	Any Japanese company. This entity includes all companies matched by the other entities in this section, as well as several hundred other significant companies.

company_engus.ecr

Entity	Description
company/fortune_1000_2008/engus	The 2008 list of Fortune 1000 companies.
company/sp500/engus	U.S. S&P 500 companies.
company/major_company/engus	Major U.S. companies.
company/law/engus	Law firms in the United States.
company/fortune_500/engus	A company that has featured in the Fortune 500 list at any time since 2011.
company/forbes_largest_private_companies2010/engus	The 2010 list of Forbes largest companies.
company/all/engus	Any U.S. company. This entity includes all companies matched by the other entities in this section, as well as several hundred other significant companies.

company_frefr.ecr

Entity	Description
company/CAC_40/frefr	A French CAC 40 company.
company/CAC_40_stocksymbol/frefr	A French CAC 40 company stock symbol.
company/CAC_next_20/frefr	A French CAC Next 20 company.
company/CAC_next_20_stocksymbol/frefr	A French CAC Next 20 company stock symbol.
company/CAC_mid_60/frefr	A French CAC Mid 60 company.
company/CAC_small/frefr	A French CAC Small company.
company/SBF_120/frefr	A French SBF 120 company.
company/all/frefr	Any French company. This entity includes all companies matched by the other entities in this section.

company_generic_eng.ecr

Entity	Description
company/generic/eng	A plausible company name. This entity matches text

company_generic_eng.ecr, continued

Entity	Description
	next to a company designator such as Inc. or Corporation (defined in `company_other_eng.ecr`), optionally with a comma. For example "Eduction Example, Inc.". The entity scores a match higher if it matches words that commonly occur in company names (as defined in `company_other_eng.ecr`), optionally with joiner words and characters, such as *and* or *&*. For example "Eduction Brothers Inc." or "Eduction Bottling Corporation" score higher than "Eduction Example Inc". The entity reduces the score for longer names and names that include a number.

company_gerde.ecr

Entity	Description
company/dax/gerde	A German DAX company.
company/dax_stocksymbol/gerde	A German DAX company stock symbol.
company/cdax/gerde	A German CDAX company.
company/hdax/gerde	A German HDAX company.
company/mdax/gerde	A German MDAX company.
company/sdax/gerde	A German SDAX company.
company/tecdax/gerde	A German TecDAX company.
company/all/gerde	Any German company. This entity includes all companies matched by the other entities in this section.

company_jpnjp.ecr

Entity	Description
company/nikkei225/jpnjp	A Japanese Nikkei 225 company.
company/all/jpnjp	Any Japanese company. This entity includes all companies matched by the other entities in this section, as well as several hundred other significant companies.

company_korkr.ecr

Entity	Description
company/all/korkr	Any Korean company.

company_law_eng.ecr

Entity	Description
company/law_sgl/eng	Law firms with single-word names.
company/law_multi/eng	Law firms with multiple-word names. When names include commas and ampersand characters, the entity includes up to three versions of the name: • full name • with commas removed • with commas and ampersand removed All suffixes are removed for data in these entities.

company_other_eng.ecr

Entity	Description
company/designator/eng	A company designator. For example, *Corp, Inc.*
company/org_legal/eng	Legal practice extensions. For example, *LLC*, *PC*.
company/common_end_word/eng	A common company name end word. For example, *Partners, Bros*.
company/non_name/eng	A non-specific name used in a company name. For example, *American, National*.
company/business/eng	A business term in a company name. For example, *Resorts, Capital, Accountants*.

company_rusru.ecr

Entity	Description
company/all/rusru	Any Russian company.

D

date_chi.ecr

Entity	Description
date/season/chi	The four seasons in Chinese.
date/season_simplified/chi	The four seasons in simplified Chinese.
date/solar_term/chi	The solar terms in Chinese.
date/solar_term_simplified/chi	The solar terms in simplified Chinese.
date/yyyy/chi	The year in Chinese.
date/yyyy_simplified/chi	The year in simplified Chinese and ASCII numbers.
date/mm/chi	The month in Chinese.
date/mm_simplified/chi	The month in simplified Chinese and ASCII numbers.
date/ddd/chi	The day of the week in Chinese.
date/ddd_simplified/chi	The day of the week in simplified Chinese.
date/rel_period/chi	A period relative to the current date in Chinese.
date/rel_period_simplified/chi	A period relative to the current date in simplified Chinese.
date/period/chi	A fixed period of time in Chinese.
date/period_simplified/chi	A fixed period of time in simplified Chinese.
date/rel_day/chi	A day relative to the current date in Chinese.
date/rel_day_simplified/chi	A day relative to the current date in simplified Chinese.
date/ddd_dd/chi	The day of the week and the day of the month in Chinese.
date/ddd_dd_simplified/chi	The day of the week and the day of the month in simplified Chinese and ASCII numbers.
date/ddd_mmdd/chi	The day of the week and the month and day in Chinese.
date/ddd_mmdd_simplified	The day of the week and the month and day in simplified Chinese and ASCII numbers.

date_chi.ecr, continued

Entity	Description
date/mmdd/chi	The month and day in Chinese.
date/mmdd_simplified	The month and day in simplified Chinese and ASCII numbers.
date/mmdd_ddd/chi	The month, day, and day of the week in Chinese.
date/mmdd_ddd_simplified/chi	The month, day, and day of the week in simplified Chinese and ASCII numbers.
date/yyyymmdd/chi	The year, month, and day in Chinese.
date/yyyymmdd_simplified/chi	The year, month, and day in simplified Chinese and ASCII numbers.
date/yyyymmdd_ddd/chi	The year, month, day, and day of the week in Chinese.
date/yyyymmdd_ddd_simplified/chi	The year, month, day, and day of the week in simplified Chinese and ASCII numbers.
date/lunar_mmdd/chi	The month and the day of the lunar calendar in Chinese.
date/lunar_mmdd_simplified/chi	The month and the day of the lunar calendar in simplified Chinese and ASCII numbers.
date/chi	A date in any format in Chinese.
date/simplified/chi	A date in any format in simplified Chinese and ASCII numbers.
date/day_and_time/chi	A time of day on a specific or relative date in Chinese.
date/day_and_time_simplified/chi	A time of day on a specific or relative date in simplified Chinese and ASCII numbers.

date_eng.ecr

Entity	Description
date/season/eng	The four seasons in English. For example, *Winter*, *Spring*.
date/year/eng	A year in English, in any format.
date/mmm/eng	The month in English, written in full or in short form. For example, *September*, *Sept*.

date_eng.ecr, continued

Entity	Description
date/ddd/eng	The day of the week in English. For example, *Monday*, *Tuesday*.
date/rel_period/eng	A period relative to the current date in English.
date/rel_day/eng	A day relative to the current date in English.
date/mmmdd/eng	The month and day in English. For example, *January 5th*, *January 5*, or *January the 5th*.
date/ddmmm/eng	The day and month in English. For example, *5th January*, *5 January*, or *5th of January*.
date/day_date/eng	The date preceded by the day of the week in English. For example, *Sat January 5*, *Saturday the 5th Jan*.
date/month_dd_year/eng	The month, day, and year in English. For example, *January 5th, 2008*.
date/dd_month_year/eng	The day, month, and year in English. For example, *5th January, 2008*.
date/day_date_year/eng	The date and year, preceded by the day of the week, in English. For example, *Saturday, January 5th, 2008*.
date/mmm_year/eng	The month and year in English. For example, *January 2008*.
date/eng	A date in any format in English. Supported formats include: Date and month, with optional day and optional year: • 04 Oct 2008 • 4th October 2008 • 4 Oct • 4th of October 2008 • October 4th 2008 • 4th Oct '08 • 04 OCTOBER '08 • Saturday, October the 4th • Sat 4th of Oct • SATURDAY 4 OCTOBER 2008

date_eng.ecr, continued

Entity	Description
	• SAT OCT 4 • Sat. 4 Oct. 2008 Extra delimiter support for formats where the year is present: • 04_OCT_2008 • 4.10.08 • 04/10/2008 • Saturday 4-10-08 • 04102008 (years 1970-2029 only) • 28-10-2008 • 10/28/08 • OCT 28 2008

date_fre.ecr

Entity	Description
date/season/fre	The seasons in French. For example, *l'Hiver*, *saison des pluies*.
date/ddd/fre	A day of the week, in French. For example, *Lundi*, *Mardi, VEN*.
date/mmm/fre	Month, written in full or in short form, in French. For example, *Septembre*, *Sept*.
date/year/fre	A year in any format.
date/ddmmm/fre	The day and month in French. For example, *5e Janvier*, *5 Janvier*.
date/day_date/fre	The day and month in French, preceded by the day of the week. For example, *Samedi, 5 Janvier*.
date/date_year/fre	The day, month, and year in French. For example, *5 Janvier, 2008*.
date/day_date_year/fre	The day, month, and year in French, preceded by the day of the week. For example, *Samedi, 5 Janvier, 2008*.

date_fre.ecr, continued

Entity	Description
date/mmm_year/fre	The month and year in French. For example, *Janvier, 2008*.
date/fre	A date in any format in French. Supported formats include: Date and month, with optional day and optional year: • 04 OCT. 2008 • 4ième Octobre 2008 • 4 Oct • 4 10 '08 • 04 OCTOBRE '08 • Samedi, 4 Oct • SAMEDI 4 OCTOBRE 2008 • Sam. 4 Oct. 2008 Extra delimiter support for formats where the year is present: • 04_OCT_2008 • 04/10/2008 • Samedi 4-10-08 • 04102008 (years 1970-2029 only)

date_ger.ecr

Entity	Description
date/ddd/ger	A day of the week in German. For example, *Montag*, *Dienstag*.
date/mmm/ger	A month in German. For example, *März*.
date/year/ger	A year in any format.
date/ddmmyyyy_dotspace/ger	dd. mm. yyyy. For example, *5. 1. 2008*.
date/ddmmm/ger	The day and month in German. For example, *5 Januar*.
date/day_date/ger	The day and month in German, preceded by the day

date_ger.ecr, continued

Entity	Description
	of the week. For example*, Samstag, 5. Januar*.
date/date_year/ger	The day, month, and year in German. For example, *5. Januar, 2008*.
date/day_date_year/ger	The day, month, and year in German, preceded by the day of the week. For example*, Samstag, 5. Januar, 2008*.
date/ger	A date in any numeric format in German. Supported formats include: Date and month, with optional day and optional year: • 04 Okt 2008 • 4 OKTOBER 2008 • 4. okt • 4 Oktober '08 • 04 OCT. '08 • 04. 2. 2007 • Samstag, 03.2.2007 • SONNABEND 4 OKTOBER 2008 • SA 04 OKT • Sa. 4. Okt. 2008 Extra delimiter support for formats where the year is present: • 04_OKT_2008 • 04/10/2008 • SA. 04-Okt-2008 • 04102008 (years 1970-2029 only) • 28-10-2008
date/mmm_year/ger	The month and year in German. For example, *Januar 2008*.

date_ita.ecr

Entity	Description
date/season/ita	The seasons in Italian. For example, *la primavera*, *l'inverno*.
date/ddd/ita	A day of the week in Italian. For example, *lunedì*, *MAR*.
date/mmm/ita	A month in Italian. For example, *gen.*, *FEBBRAIO*.
date/year/ita	A year in any format.
date/ddmmm/ita	The day and month in Italian. For example, *5 di gennaio*.
date/day_date/ita	The day and month in Italian, preceded by the day of the week. For example, *sabato 5 di gennaio.*
date/date_year/ita	The day, month, and year in Italian. For example, *5 di gennaio del 2008*.
date/day_date_year/ita	The day, month, and year in Italian, preceded by the day of the week. For example, *sabato 5 di gennaio del 2008*.
date/ita	A date in any format in Italian. Supported formats include: Date and month, with optional day and optional year: • 04 Ott 2008 • 4 OTTOBRE 2008 • 4 ott • 04 di Ottobre 2008 • 4 di Ott del '08 • 4 Ott. '08 • Venerdi', 03 di Ottobre • Sab 4 di Ott • VENERDÌ 03 DI OTTOBRE DEL 2008 • SAB 4 OTT • Sab. 4 Ott. 2008 Extra delimiter support for formats where the year is present:

date_ita.ecr, continued

Entity	Description
	• 04_OTT_2008 • 04/10/2008 • Venerdì 3-10-08 • 04102008 (years 1970-2029 only) • 28-10-2008
date/mmm_year/ita	Month and year in Italian. For example, *gennaio del 2008*.

date_jpn.ecr

Entity	Description
date/season/jpn	The seasons in Japanese.
date/ddd/jpn	A day of the week in Japanese.
date/mmm/jpn	A month in Japanese (Kanji, numerals and fullwidth numerals).
date/year_gregorian/jpn	A year in the Gregorian calendar, in Japanese, in any format, with optional A.D./B.C.
date/year_imperial/jpn	Japanese imperial calendar year from 1868 onwards, in any format.
date/mmmdd/jpn	The month and day in Japanese.
date/day_date/jpn	The day and month in Japanese, preceded by the day of the week.
date/date_year_gregorian/jpn	The year, month, and day in the Gregorian calendar, in Japanese.
date/date_year_imperial/jpn	The year, month, and day in the Japanese imperial calendar, in Japanese.
date/day_date_year_gregorian/jpn	The day, month, and year in the Gregorian calendar, in Japanese, preceded by the day of the week.
date/day_date_year_imperial/jpn	The day, month, and year in the Japanese imperial calendar, in Japanese, preceded by the day of the week.
date/jpn	A date in any numeric format in Japanese.

date_jpn.ecr, continued

Entity	Description
date/mmm_year_gregorian/jpn	The month and year in the Gregorian calendar, in Japanese.
date/mmm_year_imperial/jpn	The month and year in the Japanese imperial calendar, in Japanese.

date_numeric.ecr

Entity	Description
date/dd	A day from 1 to 31.
date/dd_fullwidth	A day from 1 to 31, in fullwidth characters.
date/dd2	A day from 01 to 31.
date/dd2_fullwidth	A day from 01 to 31, in fullwidth characters.
date/mm	A month from 1 to 12.
date/mm_fullwidth	A month from 1 to 12, in fullwidth characters.
date/mm2	A month from 01 to 12.
date/mm2_fullwidth	A month from 01 to 12, in fullwidth characters.
date/yy	The last two digits of the year. For example, *67*, *08*.
date/yy_fullwidth	The last two digits of the year, in fullwidth characters.
date/yyyy	A three- or four-digit year, from 100 to 2099.
date/yyyy_fullwidth	A three- or four-digit year in fullwidth characters, from 100 to 2099.
date/yyyy4	A four-digit year, from 1000 to 2099.
date/yyyy4_fullwidth	A four-digit year in fullwidth characters, from 1000 to 2099.
date/year	A year in any numerical format.
date/year_fullwidth	A year in any numerical format in fullwidth characters.
date/yyyymmddsep	yyyy-mm-dd. For example, *2008-10-28*.
date/yyyymmdd	yyyymmdd. For example, *20081028*.
date/yyyymmdd_safe	yyyymmdd for a date between 19700101 and

date_numeric.ecr, continued

Entity	Description
	20291231. For example, *20081028*.
date/yymmddsep	yy-mm-dd. For example, *08-10-28*.
date/yymmdd	yymmdd. For example, *081028*.
date/ddmmyyyysep	dd-mm-yyyy. For example, *28-10-2008*.
date/ddmmyyyy	ddmmyyyy. For example, *28102008*.
date/ddmmyyyy_safe	ddmmyyyy for a date between 01011970 and 31122029. For example, *28102008*.
date/ddmmyysep	dd-mm-yy. For example, *28-10-08*.
date/ddmmyy	ddmmyy. For example, *281008*.
date/mmddyyyysep	mm-dd-yyyy. For example, *10-28-2008*.
date/mmddyyyy	mmddyyyy. For example, *10282008*.
date/mmddyyyy_safe	mmddyyyy for a date between 01011970 and 12312029. For example, *10282008.*
date/mmddyysep	mm-dd-yy. For example, *10-28-2008*.
date/mmddyy	mmddyy. For example, *102808*.

date_por.ecr

Entity	Description
date/season/por	The seasons in Portuguese. For example, *Verão, Outono.*
date/ddd/por	A day of the week in Portuguese. For example, *Segunda-feira*, *Terça-feira, DOM*.
date/mmm/por	A month in Portuguese. For example, *Setembro*.
date/year/por	A year in any format.
date/ddmmm/por	The day and month in Portuguese. For example, *5 de Janeiro*.
date/day_date/por	The day and month in Portuguese, preceded by the day of the week. For example, *Sábado 5 de Janeiro*.
date/date_year/por	The day, month, and year in Portuguese. For

date_por.ecr, continued

Entity	Description
	example, *5 de maio 2008*.
date/day_date_year/por	The day, month, and year in Portuguese, preceded by the day of the week. For example, *Sábado 5 de janiero de 2008*.
date/por	Any date in Portuguese. Supported formats include: Date and month, with optional day and optional year: • 04 Out. 2008 • 4 OUTUBRO 2008 • 04 de Outubro 2008 • 4 de Out de '08 • SÁB 04 OUT 2008 • Sábado, 04 de Outubro • Terça-feira 14 Out. 1947 • SÁBADO 04 DE OUTUBRO DE 2008 • Quinta-feira, 12 de Setembro de 2013 EC • 4 de Março de 2012 Extra delimiter support for formats where the year is present: • 04_OUT_2008 • 04/10/2008 • Quarta feira 30-12-1953 • 04102008 (years 1970-2029 only) • 28-10-2008
date/mmm_year/por	The month and year in Portuguese. For example, *Junho de 2008.*

date_spa.ecr

Entity	Description
date/season/spa	The seasons in Spanish. For example, *el invierno, la primavera.*

date_spa.ecr, continued

Entity	Description
date/ddd/spa	A day of the week in Spanish. For example, *Lunes, Domingo*.
date/mmm/spa	A month in Spanish. For example, *Septiembre*.
date/year/spa	A year in any format.
date/ddmmm/spa	The day and month in Spanish. For example, *5 de enero*.
date/date_year/spa	The day, month, and year in Spanish. For example, *5 de enero 2008*.
date/day_date/spa	The day and month in Spanish, preceded by the day of the week. For example, *Sábado 5 de enero*.
date/day_date_year/spa	The day, month, and year in Spanish, preceded by the day of the week. For example, *Sábado 5 de enero de 2008*.
date/mmm_year/spa	The month and year in Spanish. For example, *Januar 2008*.
date/spa	Any date in Spanish. Supported formats include: Date and month, with optional day and optional year: • 04 Oct 2008 • 4 OCTUBRE 2008 • 4 OCT • 4 de Octubre 2008 • 4 de Oct de '08 • 04 OCT. '08 • Sábado, 04 de Octubre • Jueves, 12 de Septiembre de 2013 d. J.C. • SÁBADO 04 DE OCTUBRE DE 2008 • SAB 4 OCT • Sab. 4 Oct. 2008 Extra delimiter support for formats where the year is present: • 04_OCT_2008

date_spa.ecr, continued

Entity	Description
	• 04/10/2008 • Sábado 4-10-08 • 04102008 (years 1970-2029 only) • 28-10-2008

datetime_advanced_eng.ecr

Entity[1]	Description
datetime/advanced_hms24/eng	Time in hh:mm:ss.ss ZZZ format (seconds, fractional seconds, and timezone are optional). For example, *04:35*, *18:56:00*, *21:42:56.45 +0100*.
datetime/advanced_hms24_range/eng	Time range in hh:mm:ss.ss ZZZ format (seconds, fractional seconds, and timezone are optional). For example, *04:35-04:36*, *18:56:00-21:00:00*, *21:42:56.45 to 23:59:59.99 +0100*.
datetime/advanced_hm24_dot/eng	Time in hh.mm format. For example, *04.56*.
datetime/advanced_hm24_dot_range/eng	Time range in hh.mm format. For example, *04.56 to 12.34*.
datetime/advanced_hm24tz_nosep/eng	Time in hhmm ZZZ format. For example, *2100 GMT*.
datetime/advanced_hm24tz_nosep_ range/eng	Time range in hhmm ZZZ format. For example, *2100-2330 GMT*.
datetime/advanced_hm24_nosep/eng	Time in hhmm format, with higher scores if the number of minutes is a multiple of 5. For example, *2100*.
datetime/advanced_hm24_nosep_ range/eng	Time range in hhmm format, with higher scores if the number of minutes is a multiple of 5. For example, *2100-2330*.
datetime/advanced_hms12/eng	Time in 12-hour h:mm:ss am/pm ZZZ format (seconds and timezone are optional). For example, *9:30am*, *9:30:00pm GMT*.
datetime/advanced_hms12_range/eng	Time range in 12-hour h:mm:ss am/pm ZZZ format (seconds and timezone are optional). For example, *9:30-10:30am*, *9:30:00am to 9:30:00pm GMT*.
datetime/advanced_hm12_noampm/eng	Time in 12-hour h:mm:ss ZZZ format without am or

datetime_advanced_eng.ecr, continued

Entity[1]	Description
	pm specified (seconds and timezone are optional). For example, *9:30*, *9:30:00 GMT*.
datetime/advanced_hm12_noampm_range/eng	Time range in 12-hour h:mm:ss ZZZ format without am or pm specified (seconds and timezone are optional). For example, *9:30-10:30*, *9:30:00 to 9:30:00 GMT*.
datetime/advanced_hm12_dot/eng	Time in 12-hour h.mm am/pm ZZZ format (am, pm, and timezone are optional, but scores are lower without them, although multiples of 5 minutes are boosted). For example, *6.30am*, *8.45 GMT*, *11.35*.
datetime/advanced_hm12_dot_range/eng	Time range in 12-hour h.mm am/pm ZZZ format (am, pm, and timezone are optional, but scores are lower without them, although multiples of 5 minutes are boosted). For example, *6-7.30am*, *8.45am-6.30pm GMT*, *11-12.35*.
datetime/advanced_hm12_nosep/eng	Time in 12-hour hmm am/pm ZZZ format (am, pm, and timezone are optional, but scores are lower without them, although multiples of 5 minutes are boosted). For example, *630am*, *845 GMT*, *1135*.
datetime/advanced_hm12_nosep_range/eng	Time range in 12-hour hmm am/pm ZZZ format (am, pm, and timezone are optional, but scores are lower without them, although multiples of 5 minutes are boosted). For example, *630-730am*, *845-945 GMT*, *1135 to 345*.
datetime/advanced_namedtime/eng	Times of the day with a specific name in the English language. For example, *noon*, *midnight*.
datetime/advanced_clocktime_loose/eng	Time of the day, or time range, described in English (low confidence, scores reduced). For example, *twelve*.
datetime/advanced_clocktime_loose_range/eng	Time range, described in English (low confidence, scores reduced). For example, *two to three*.
datetime/advanced_clocktime_strict/eng	Time of the day, described in English (high confidence). For example, *twelve o'clock*, *two fifteen*, *ten past one*, *quarter to midnight*.
datetime/advanced_clocktime_strict_range/eng	Time range, described in English (high confidence). For example, *ten to ten forty-five*.
datetime/advanced_clocktime/eng	Time of the day, described in English (high and low

datetime_advanced_eng.ecr, continued

Entity[1]	Description
	confidence, scored appropriately). For example, *twelve o'clock*, *two fifteen in the afternoon*, *ten past one*, *quarter to midnight*, *twelve at night*.
datetime/advanced_clocktime_range/eng	Time range, described in English (high and low confidence, scored appropriately). For example, *ten to ten forty-five*, *two to three*.
datetime/nameddays_strict/eng	Specially named days (confident matched only). For example, *Christmas Day*, *Easter Monday*.
datetime/nameddays_all/eng	Specially named days (high and low confidence matches, scored appropriately). For example, *Christmas*, *Easter*.
datetime/advanced_yyyymmdd/eng	Dates in yyyy-mm-dd, yyyy-Mmm-dd, yyyy.mm.dd, yyyy.Mmm.dd, yyyy/mm/dd, or yyyy/Mmm/dd formats. For example, *2008-10-28*, *2008.Oct.28*.
datetime/advanced_yyyymmdd_ range/eng	Date range in yyyy-mm-dd, yyyy-Mmm-dd, yyyy.mm.dd, yyyy.Mmm.dd, yyyy/mm/dd, or yyyy/Mmm/dd formats. For example, *2008-10-28 to 2008-11-03*, *2008.Oct.28-2008.Nov.03*.
datetime/advanced_yyyymmdd_ nosep/eng	Dates in yyyy-mm-dd, yyyy.mm.dd, or yyyy/mm/dd formats. For example, *20081028*.
datetime/advanced_yyyymmdd_nosep_ range/eng	Date range in yyyy-mm-dd, yyyy.mm.dd, or yyyy/mm/dd formats. For example, *20081028-20081103*.
datetime/advanced_ddmmyyyy/eng	Dates in dd-m-yyyy, dd.m.yyyy, or dd/m/yyyy formats. For example, *28-10-2008*.
datetime/advanced_ddmmyyyy_ range/eng	Date range in dd-m-yyyy, dd.m.yyyy, or dd/m/yyyy formats. For example, *28-10-2008 to 03-11-2008*.
datetime/advanced_ddmmyy/eng	Dates in dd-mm-yy, dd.mm.yy, or dd/mm/yy formats. For example, *28.10.08*.
datetime/advanced_ddmmyy_range/eng	Date range in dd-mm-yy, dd.mm.yy, or dd/mm/yy formats. For example, *28.10.08 -03.11.08*.
datetime/advanced_ddmm/eng	Dates in dd-mm, dd.mm, or dd/mm formats. For example, *28/10*.
datetime/advanced_ddmm_range/eng	Date range in dd-mm, dd.mm, or dd/mm formats. For example, *28/10-03/11*.

datetime_advanced_eng.ecr, continued

Entity[1]	Description
datetime/advanced_ddMmmyyyy/eng	Dates in dd-Mmm-yyyy, dd.Mmm.yyyy, or dd/Mmm/yyyy formats. For example, *28-Oct-2008*.
datetime/advanced_ddMmmyyyy_range/eng	Date range in dd-Mmm-yyyy, dd.Mmm.yyyy, or dd/Mmm/yyyy formats. For example, *28-Oct-2008 to 03-Nov-2008*.
datetime/advanced_ddMmmyy/eng	Dates in dd-Mmm-yy, dd.Mmm.yy, or dd/Mmm/yy formats. For example, *28.Oct.08*.
datetime/advanced_ddMmmyy_range/eng	Date range in dd-Mmm-yy, dd.Mmm.yy, or dd/Mmm/yy formats. For example, *28.Oct.08-03.Nov.08*.
datetime/advanced_ddMmm/eng	Dates in dd-Mmm, dd.Mmm, or dd/Mmm formats. For example, *28/Oct*.
datetime/advanced_ddMmm/eng	Date range in dd-Mmm, dd.Mmm, or dd/Mmm formats. For example, *28/Oct-03/Nov*.
datetime/advanced_mmddyyyy/eng	Dates in m-dd-yyyy, m.dd.yyyy, or m/dd/yyyy formats. For example, *10-28-2008*.
datetime/advanced_mmddyyyy_range/eng	Date range in m-dd-yyyy, m.dd.yyyy, or m/dd/yyyy formats. For example, *10-28-2008 to 11-03-2008*.
datetime/advanced_mmddyy/eng	Dates in mm-dd-yy, mm.dd.yy, or mm/dd/yy formats. For example, *10.28.08*.
datetime/advanced_mmddyy_range/eng	Date range in mm-dd-yy, mm.dd.yy, or mm/dd/yy formats. For example, *10.28.08 to 11.03.08*.
datetime/advanced_mmdd/eng	Dates in mm-dd, mm.dd, mm/dd formats. For example, *10/28*.
datetime/advanced_mmdd_range/eng	Date range in mm-dd, mm.dd, mm/dd formats. For example, *10/28-11/03*.
datetime/advanced_Mmmddyyyy/eng	Dates in Mmm-dd-yyyy, Mmm.dd.yyyy, or Mmm/dd/yyyy formats. For example, *Oct-28-2008*.
datetime/advanced_Mmmddyyyy_range/eng	Date range in Mmm-dd-yyyy, Mmm.dd.yyyy, or Mmm/dd/yyyy formats. For example, *Oct-28-2008 to Nov-03-2008*.
datetime/advanced_Mmmddyy/eng	Dates in Mmm-dd-yy, Mmm.dd.yy, or Mmm/dd/yy formats. For example, *Oct.28.08*.
datetime/advanced_Mmmddyy_	Date range in Mmm-dd-yy, Mmm.dd.yy, or

datetime_advanced_eng.ecr, continued

Entity[1]	Description
range/eng	Mmm/dd/yy formats. For example, *Oct.28.08 to Nov.03.08*.
datetime/advanced_Mmmdd/eng	Dates in Mmm-dd, Mmm.dd, Mmm/dd formats. For example, *Oct/28*.
datetime/advanced_Mmmdd_range/eng	Date range in Mmm-dd, Mmm.dd, Mmm/dd formats. For example, *Oct/28-Nov/03*.
datetime/advanced_Mmmyyyy/eng	Named month/year. For example, *Oct 2008*, *October of 2008*, *October '08*.
datetime/advanced_Mmmyyyy_ range/eng	Named month/year range. For example, *Oct 2008 to Feb 2009*.
datetime/advanced_yyyymm/eng	Dates in yyyy-mm, yyyy.mm, yyyy/mm and yyyy mm formats.
datetime/advanced_yyyymm_range/eng	Date range in yyyy-mm, yyyy.mm, yyyy/mm and yyyy mm formats.
datetime/advanced_textdate_noyear/eng	A date, without a year, described in English. For example, *July 4th*, *July the 4th*, *The morning of Wednesday July fourth*, *4th July,the 4th of July*, *In the morning on Wednesday fourth July*, *Christmas Eve*, *Easter Day*.
atetime/advanced_textdate_noyear_ range/eng	A date range, without a year, described in English. For example, *July 4th-8th*, *July 4th-October 8th*, *July the 4th to the 8th*, *Wednesday 4th to Sunday 8th July*, *4th July-8th October*.
datetime/advanced_textdate_noyear_ multiple/eng	Multiple dates, without a year, described in English. For example, *July 4th and 8th*, *4th, 5th and 6th of July*, *Wednesday 4th and Sunday 8th July*.
datetime/advanced_textdate_ withyear/eng	A date, with a year, described in English. For example, *July 4th 2008*, *July the 4th 2008*, *The morning of Wednesday July fourth 2008*, *4th July 2008*, *the 4th of July 2008*, *In the morning on Wednesday fourth July 2008*, *Christmas Day 2012*, *Easter '02*.
datetime/advanced_textdate_withyear_ range/eng	A date range, with a year, described in English. For example, *July 4th-8th 2008*, *July 4th-October 8th 2008*, *July the 4th to the 8th, 2008*, *Wednesday 4th to Sunday 8th July 2008*, *4th July-8th October 2008*.

datetime_advanced_eng.ecr, continued

Entity[1]	Description
datetime/advanced_textdate_withyear_multiple/eng	Multiple dates, with a year, described in English. For example, *July 4th and 8th, 2008*, *4th, 5th and 6th of July 2008*, *Wednesday 4th and Sunday 8th July 2008*.
datetime/advanced_textdate/eng	A date, described in English (with or without a year).
datetime/advanced_textdate_range/eng	A date range, described in English (with or without a year).
datetime/advanced_textdate_multiple/eng	Multiple dates, described in English (with or without a year).
datetime/advanced_reldate/eng	A day, or part of a day, relative to today, described in English. Less confident matches are scored lower. For example, *This morning*, *tomorrow evening*, *Today* (score 0.9), *yesterday* (score 0.9), *the day after tommorrow* [sic] (score 0.9), *Two weeks ago on Monday*, *This coming Tuesday AM*, *Two weeks on Wednesday afternoon*, *Tuesday week*, *Tomorrow fortnight*, *Two weeks ago last Monday*, *Monday last week*, *Tuesday this wk*, *Wednesday next*, *Not this Tuesday but next* (score 0.9), *Not last Monday but the one before* (score 0.9), *4th* (score 0.5), *the fourth* (score 0.8), *Wednesday the fourth of next month*, *last month on Wednesday 4th*, *The first Sunday of next month*, *Second Monday in July*, *Last Tuesday of April but one*, *Monday morning* (score 0.9), *Tuesday* (score 0.7).
datetime/advanced_reldate_range/eng	A date range, relative to today, described in English. Less confident matches are scored lower. For example, *Wednesday 4th to Friday 6th* (score 1.0), *4th-6th next month* (score 0.9), *4th to 6th* (score 0.8).
datetime/advanced_reldate_multiple/eng	Multiple dates, relative to today, described in English. For example, *4th and 6th of next month*.
datetime/advanced_relmonth/eng	A month, relative to today, described in English. Less confident matches are scored lower. For example, *In January* (score 0.6), *This July* (score 0.6), *September* (score 0.4) *Last September* (score 0.8), *January next year* (score 0.9), *Next month* (score 0.6), *last month* (score 0.6).
datetime/advanced_date_and_time/eng	Any date with time, in any recognized format. Relevant components are extracted. Score indicates

datetime_advanced_eng.ecr, continued

Entity[1]	Description
	the confidence that the matched text is a genuine reference to a date and time.
datetime/advanced_date_only/eng	Any date (without a time), in any recognized format. Relevant components are extracted. Score indicates the confidence that the matched text is a genuine reference to a date.
datetime/advanced_month_only/eng	Any date to month precision, in any recognized format. Relevant components are extracted. Score indicates the confidence that the matched text is a genuine reference to a date.
datetime/advanced_time_only/eng	Any time (without a date), in any recognized format. Relevant components are extracted. Score indicates the confidence that the matched text is a genuine reference to a time.
datetime/advanced/eng	Any date, with optional time, in any recognized format. Relevant components are extracted. Score indicates the confidence that the matched text is a genuine reference to a date and time. You can use the `datetime.lua` script to standardize the output of these entities. To allow precision to the month of the year rather than just day, set `RelaxedPrecision` to **`True`** in your eduction request.
datetime/advanced_date_context/eng	A context-aware date. For example, *two days after*. **NOTE:** To use this entity, you must configure it explicitly; it is not included in the `datetime/advanced/eng` combined entity. You must also configure at least one date or datetime entity for the context entities to use for reference.
datetime/advanced_date_and_time_context/eng	A contextual date with a time. For example *7pm the Tuesday before that*. **NOTE:** To use this entity, you must configure it explicitly; it is not included in the `datetime/advanced/eng` combined entity. You must also configure at least one date or datetime entity for the context entities to use for reference.
datetime/advanced_context/eng	A contextual date with optional time.

datetime_advanced_eng.ecr, continued

Entity[1]	Description
	NOTE: To use this entity, you must configure it explicitly; it is not included in the `datetime/advanced/eng` combined entity. You must also configure at least one date or datetime entity for the context entities to use for reference.

[1]You can use the `datetime.lua` post-processing script to normalize the output of these entities to a standard format.

E

ethnicity_eng.ecr

Entity	Description
ethnicity/nationality/eng	A nationality. For example, *Andorran*, *Welsh*.

ethnicity_engca.ecr

Entity	Description
ethnicity/aboriginal/engca	A Canadian aboriginal group. For example, *Inuit*.
ethnicity/population_group/engca	A Canadian population group. For example, *Arab*, *White*.

ethnicity_enggb.ecr

Entity	Description
ethnicity/enggb	Ethnicity classification in England. For example, *Irish*, *Indian*.
ethnicity/identity_code/enggb	United Kingdom identity code. For example, *IC1*, *IC2*.

ethnicity_engus.ecr

Entity	Description
ethnicity/races/engus	A United States race. For example, *Japanese*, *White*.

ethnicity_engus.ecr, continued

Entity	Description
ethnicity/races_lowercase/engus	A U.S. race in lowercase. For example, *japanese*, *white*.
ethnicity/native_american/engus	A U.S. native. For example, *Cherokee*, *Lambee*.
ethnicity/asian/engus	A U.S. ethnicity of Asian origin. For example, *Pakistani*, *Korean*.
ethnicity/pacific/engus	A U.S. ethnicity of Pacific origin. For example, *Fijian*, *Tongan*.
ethnicity/hispanic/engus	A U.S. ethnicity of Hispanic origin. For example, *Cuban*, *Spanish*.
ethnicity/engus	Any U.S. ethnicity.

ethnicity_fre.ecr

Entity	Description
ethnicity/nationality/fre	Nationality in French. For example, *Andorrane*, *Vietnamien*.
ethnicity/ethnic_groups/fre	Ethnic groups in the French language. For example, *Africain*, *Autres*.

F

The financial strength grammar relies on the `contextualize_matches.lua` post-processing script. This script uses the configurable entities to match sections of your documents, and then recombines these entities to return a different set of entities that represent the financial sentiment.

For more information see Financial Sentiment Analysis, on page 84.

financial_strength.ecr

Entity	Description
finance/analyst	An entity to support finding analyst statements that express financial strength. After post-processing, this entity returns the following entities: • **finance/strong**. A financial analyst statement that shows improving finances. For example,

financial_strength.ecr, continued

Entity	Description
	"Eduction Inc. under weight to over weight". • **finance/neutral**. A neutral financial analyst statement. For example, "Eduction Inc. equal weight". • **finance/weak**. A financial analyst statement that shows degrading finances. For example, "Eduction Inc. over weight to under weight". • **finance/following/strong**. A financial analyst statement indicating that an analyst starts to follow a company with a positive initial outlook. For example, "Analyst Group initiates coverage on Eduction Inc. with a Buy rating". • **finance/following/neutral**. A financial analyst statement indicating that an analyst starts to follow a company with a neutral initial outlook. For example, "Analyst Group initiates coverage on Eduction Inc. with a hold rating" • **finance/following/weak**. A financial analyst statement indicating that an analyst starts to follow a company with a negative initial outlook. For example, "Analyst Group initiates coverage on Eduction Inc. with an Underperform rating."
finance/company/tagged/known finance/news	Entities to support finding news statements that express financial strength. You must configure these entities together. After post-processing, these entities return the following entities: • **news/improve/context**. News about improving finances with an explicit company context. For example, "Eduction Inc. reports rise in profits". • **news/neutral/context**. Neutrally reported news with an explicit company subject. For example, "Eduction Inc. announced $1bn in profits". • **news/degrade/context**. News about degrading finances with an explicit company context. For example, "Eduction Inc reports fall in profits". • **news/improve/nocontext**. News with an implicit company context. For example, "Eduction Inc. had a successful year. Profits

financial_strength.ecr, continued

Entity	Description
	rose 10 percent". • **news/degrade/nocontext**. News of degrading finances with an implicit company context. For example, "Eduction Inc. had a bad year. Profits fell 10 percent". • **news/neutral/reports/nocontext**. A neutral report with an implicit or no company subject. For example, "reported profits of $1bn". • **news/improve/reports/nocontext**. A report of improving finances with an implicit company subject. For example, "It was a good year for Eduction Inc as they announced a 10 percent rise in profits". • **news/degrade/reports/nocontext**. A report of degrading finances with an implicit company subject. For example, "It was a bad year for Eduction Inc as they announced a 10 percent fall in profits". • **news/deal/positive/context**. A reported approved deal with an explicit company subject. For example, "Shareholders approve the Eduction Inc. deal". • **news/deal/negative/context**. A reported blocked deal with an explicit company subject. For example, "Shareholders block the Eduction Inc. deal". • **news/takeover/context**. A reported takeover with explicit company subject. For example, "Eduction Corp. hope to consolidate their market position with a $1bn takeover of Eduction Inc."

G

gender_eng.ecr

Entity	Description
gender/gender_word/eng	A word that describes a family relation or gender in

gender_eng.ecr, continued

Entity	Description
	English. For example, *lady, father*.
gender/gender_context/eng	A gender in the context of English language.
gender/all/eng	A gender in the English language, either in a word or in context.

gender_fre.ecr

Entity	Description
gender/gender_word/fre	A word that describes a family relation or gender in French. For example, *Dame*, *voisines*.
gender/gender_context/fre	A gender in the context of French language.
gender/all/fre	A gender in the French language, either in a word or in context.

gender_ger.ecr

Entity	Description
gender/gender_word/ger	A word that describes a family relation or gender in German. For example, *mensch*, *Frau*.
gender/gender_context/ger	A gender in the context of German language.
gender/all/ger	A gender in the German language, either in a word or in context.

gov_chicn.ecr

Entity	Description
org/gov/chicn	A Chinese government agency.

gov_engca.ecr

Entity	Description
org/gov/engca	A Canadian government agency.

H

holiday_ca.ecr

Entity	Description
holiday/statutory/engca	Statutory Canadian holidays in English. For example, *Good Friday.*
holiday/statutory/freca	Statutory Canadian holidays in French. For example, *Le vendredi saint.*
holiday/statutory/ca	Statutory Canadian holidays, in English or French.
holiday/federal/engca	Federal Canadian holidays in English. For example, *Victoria Day.*
holiday/federal/freca	Federal Canadian holidays in French. For example, *La fête de la Reine.*
holiday/federal/ca	Federal Canadian holidays, in English or French.
holiday/statother/engca	Other statutory Canadian holidays in English. For example, *Family Day.*
holiday/statother/freca	Other statutory Canadian holidays in French. For example, *La fête du Travail.*
holiday/statother/ca	Other statutory Canadian holidays, in English or French.
holiday/alberta/engca holiday/britishcolumbia/engca holiday/manitoba/engca holiday/newbrunswick/engca holiday/newfoundlandlabrador/engca holiday/northwestterritories/engca holiday/novascotia/engca holiday/nunavut/engca holiday/ontario/engca holiday/princeedwardisland/engca holiday/quebec/engca holiday/saskatchewan/engca	Holidays for each Canadian province and territory in English.

holiday_ca.ecr, continued

Entity	Description
holiday/yukon/engca	
holiday/prov_terr/engca	Holidays for Canadian provinces and territories in English.
holiday/other/engca	Other Canadian holidays and observances in English.
holiday/ca	All Canadian holidays.

holiday_enggb.ecr

Entity	Description
holiday/bank_holiday/enggb	British Bank Holiday name.
holiday/holiday/enggb	Traditional days celebrated. For example, *Mother's Day*.

holiday_engus.ecr

Entity	Description
holiday/federal/engus	U.S. federal holidays. For example, *Memorial Day*.
holiday/traditional/engus	Traditional U.S. days celebrated. For example, *Mother's Day*.
holiday/engus	All U.S. holidays.

I

internet.ecr

Entity	Description
internet/host_domain	A host name. For example, *www.myhost.com*.
internet/host_ip/ipv4	An IPv4 IP address. For example, *127.0.0.1*.
internet/host_ip/ipv6	An IPv6 IP address. For example, *1234:5678:90AB:CDEF.*
internet/host_ip/ipv4mapped	An IPv4-mapped IP address. For example,

internet.ecr, continued

Entity	Description
	::FFFF:129.144.52.38.
internet/host_ip	Any IP address.
internet/addr_host	Host address. For example, *www.myhost.com* or *192.231.21.2*.
internet/addr_email	Email address. For example, *jsmith@mailserver.com*.
internet/addr_email_mailto	Email address with *mailto:* prefix. For example, *mailto:jsmith@mailserver.com*.
internet/addr_https	HTTP or HTTPS address.
internet/addr_file	file:// address.
internet/addr_ftp	FTP address.
internet/addr_news	news:// address.
internet/addr_telnet	Telnet address.
internet/addr_gopher	Gopher address.

J

jobtitledicts_eng.ecr

Entity	Description
person/titleprefix_camelcase/eng	Job title prefix in camel case. For example, *Acting*.
person/titleprefix_lowercase/eng	Job title prefix in lowercase. For example, *acting*.
person/titlesuffix_camelcase/eng	Job title suffix in camel case. For example, *Associate*, *Advisor*.
person/titlesuffix_lowercase/eng	Job title suffix in lowercase. For example, *educator*, *trainee*.
person/govdep/engus	U.S. government departments and abbreviations. For example, *National Security Council*, *FBI.*
person/titlegeneric_camelcase/eng	Generic job titles in camel case. For example, *Sales Assistant*.

jobtitledicts_eng.ecr, continued

Entity	Description
person/titlegeneric_lowercase/eng	Generic job titles in lowercase. For example, *sales assistant*.
person/titlefull_camelcase/eng	Full job title in camel case, including prefixes and suffixes. For example, *Head of Customer Communications*.
person/titlefull_lowercase/eng	Full job title in lower case, including prefixes and suffixes. For example, *head of customer communications*.
person/titlecorp/eng	Corporate job titles. For example, *Chief Financial Officer.*
person/titlecorpabb/eng	Abbreviated version of corporate job titles. For example, *CFO.*
person/titlegov/eng	Government and cabinet titles. For example, *President*, *Secretary of Defense.*
person/titleroyal/eng	Royal titles. For example, *King.*
person/titlepolitical/eng	Political titles. For example, *Foreign Minister*, *Governor.*
person/titlereligious/eng	Religious titles. For example, *Pope*, *Father*, *Imam.*

L

languages.ecr

Entity	Description
language/iso_lowercase	Three-letter ISO 639-2/B language code. For example, *fin*, *ger*.
language/all	Language name in a local language, English, or other major language.
language/output_iso	Language name in a local language, English, or other major language (output is normalized to the ISO 639-2/B code)

legal_engus.ecr

Entity	Description
legal/citsupr/engus	Supreme Court Citations. For example, *Roe v. Wade, 410 U.S. 113 (1973)*.
legal/citcofa/engus	Federal Court Reporter Citations. For example, *Universal City Studios, Inc. v. Corley, 273 F.3d 429 (2d Cir. 2001)*.

M

measure_eng.ecr

Entity	Description
measure/len/met/eng	Metric measures of length. For example, *mm.*, *kilometre.*
measure/len/usuk/eng	U.S. and UK measures of length. For example, *foot*, *mile*, *in.*
measure/area/met/eng	Metric measures of area. For example, *sq. m.*, *square kilometres.*
measure/area/usuk/eng	U.S. and UK measures of area. For example, *sq. in.*, *acres.*
measure/vol/met/eng	Metric measures of volume. For example, *microlitres*, *cubic centimetres.*
measure/vol/usuk/eng	U.S. and UK measures of volume. For example, *pinches*, *cups*, *gal.*
measure/mass/met/eng	Metric measures of mass. For example, *gram*, *tonnes.*
measure/mass/usuk/eng	U.S. and UK measures of mass. For example, *pound*, *lb.*

medical_condition.ecr

Entity	Description
medical/disability/social_security/engus	Impairment for the purpose of disability evaluation under social security.
medical/disease_condition	Disease or medical condition. This grammar includes

medical_condition.ecr, continued

Entity	Description
	the clinical and common names for many common medical conditions. For example, *Heart attack* or *myocardial infarction*.
medical/lifestyle	Lifestyle that relates to medical condition.

medical_drug.ecr

Entity	Description
drug/brand	Trade name of medical drugs.
drug/generic	Generic name of medical drugs.
drug/medication	Description of a medication.

medical_healthcare_engus.ecr

Entity	Description
healthcare/provider/AK/engus	U.S. healthcare provider in Alaska.
healthcare/provider/AL/engus	U.S. healthcare provider in Alabama.
healthcare/provider/AR/engus	U.S. healthcare provider in Arkansas.
healthcare/provider/AZ/engus	U.S. healthcare provider in Arizona.
healthcare/provider/CA/engus	U.S. healthcare provider in California.
healthcare/provider/CO/engus	U.S. healthcare provider in Colorado.
healthcare/provider/CT/engus	U.S. healthcare provider in Connecticut.
healthcare/provider/DC/engus	U.S. healthcare provider in Washington, D.C.
healthcare/provider/DE/engus	U.S. healthcare provider in Delaware.
healthcare/provider/FL/engus	U.S. healthcare provider in Florida.
healthcare/provider/GA/engus	U.S. healthcare provider in Georgia.
healthcare/provider/HI/engus	U.S. healthcare provider in Hawaii.
healthcare/provider/IA/engus	U.S. healthcare provider in Iowa.
healthcare/provider/ID/engus	U.S. healthcare provider in Idaho.

medical_healthcare_engus.ecr, continued

Entity	Description
healthcare/provider/IL/engus	U.S. healthcare provider in Illinois.
healthcare/provider/IN/engus	U.S. healthcare provider in Indiana.
healthcare/provider/KS/engus	U.S. healthcare provider in Kansas.
healthcare/provider/KY/engus	U.S. healthcare provider in Kentucky.
healthcare/provider/LA/engus	U.S. healthcare provider in Louisiana.
healthcare/provider/MA/engus	U.S. healthcare provider in Massachusetts.
healthcare/provider/MD/engus	U.S. healthcare provider in Maryland.
healthcare/provider/ME/engus	U.S. healthcare provider in Maine.
healthcare/provider/MI/engus	U.S. healthcare provider in Michigan.
healthcare/provider/MN/engus	U.S. healthcare provider in Minnesota.
healthcare/provider/MO/engus	U.S. healthcare provider in Missouri.
healthcare/provider/MS/engus	U.S. healthcare provider in Mississippi.
healthcare/provider/MT/engus	U.S. healthcare provider in Montana.
healthcare/provider/NC/engus	U.S. healthcare provider in North Carolina.
healthcare/provider/ND/engus	U.S. healthcare provider in North Dakota.
healthcare/provider/NE/engus	U.S. healthcare provider in Nebraska.
healthcare/provider/NH/engus	U.S. healthcare provider in New Hampshire.
healthcare/provider/NJ/engus	U.S. healthcare provider in New Jersey.
healthcare/provider/NM/engus	U.S. healthcare provider in New Mexico.
healthcare/provider/NV/engus	U.S. healthcare provider in Nevada.
healthcare/provider/NY/engus	U.S. healthcare provider in New York.
healthcare/provider/OH/engus	U.S. healthcare provider in Ohio.
healthcare/provider/OK/engus	U.S. healthcare provider in Oklahoma.
healthcare/provider/OR/engus	U.S. healthcare provider in Oregon.
healthcare/provider/PA/engus	U.S. healthcare provider in Pennsylvania.
healthcare/provider/PR/engus	U.S. healthcare provider in Puerto Rico.

medical_healthcare_engus.ecr, continued

Entity	Description
healthcare/provider/RI/engus	U.S. healthcare provider in Rhode Island.
healthcare/provider/SC/engus	U.S. healthcare provider in South Carolina.
healthcare/provider/SD/engus	U.S. healthcare provider in South Dakota.
healthcare/provider/TN/engus	U.S. healthcare provider in Tennessee.
healthcare/provider/TX/engus	U.S. healthcare provider in Texas.
healthcare/provider/UT/engus	U.S. healthcare provider in Utah.
healthcare/provider/VA/engus	U.S. healthcare provider in Virginia.
healthcare/provider/VT/engus	U.S. healthcare provider in Vermont.
healthcare/provider/WA/engus	U.S. healthcare provider in Washington.
healthcare/provider/WI/engus	U.S. healthcare provider in Wisconsin.
healthcare/provider/WV/engus	U.S. healthcare provider in West Virginia.
healthcare/provider/WY/engus	U.S. healthcare provider in Wyoming.
healthcare/provider/all/engus	Any U.S. healthcare provider.

medical_procedure.ecr

Entity[1]	Description
medical/blood_test	A blood test. For example, *Blood Type Test* or *hCG*.
medical/lab_test	A lab test. For example, *CD4 Lymphocyte Count* or *Insulin C-peptide*.
medical/surgical_procedure	A surgical procedure. For example, *Appendectomy* or *Hip Replacement Surgery*.
medical/specialty	A medical specialty. For example, *dermatology* or *psychiatry*.

monetary_value.ecr

Entity	Description
monetary_value/full_value	A monetary value without context. For example, *$5 billion*, *£9.8m*, *EUR50bn*, *£15,275,486*, *25 cents*.

monetary_value.ecr, continued

Entity	Description
	This entity has the following components: • PRECURRENCY, for example *$* from *$100*. • POSTCURRENCY, for example *cents* from *25 cents*. • VALUE, for example *50.0* from *50.0 million* or *1.26* from *£1.26k*. • BIGUNITS, for example *million* from *$50.0 million*. You can use the Lua post-processing script `normalize_money.lua` to normalize the values. The script adds a component named NORMALIZED_VALUE to each match. For example, the input *£9.8m* would have a VALUE of *9.8* but a NORMALIZED_VALUE of *9,800,000*.

money_eng.ecr

Entity	Description
money/fracunits	Fractional units of currency such as *Cent* or *Penny*.
money/iso4217	ISO 4217 currency codes. For example, *AUD* or *USD*.
money/currency	Currency name. For example, *Algerian dinar*.
money/currencyabbrev	Abbreviated currency name. For example, *dinar* or *dollar*.
money/denom_us	U.S. denominations. For example, *penny* or *quarter*.
money/symbol	Currency symbols. For example, *$* or *€*.

[1]This grammar matches the clinical names of many common medical procedures.

N

number_banking_au.ecr

Entity	Description
number/bsb/au	Australian bank state branch number. For example, *34* or *985.*

number_banking_ca.ecr

Entity	Description
number/cpa_transit_micr/ca	Canadian Payments Association MICR transit number, in the format *BBBBB-AAA*, where *BBBB* is a five-digit code that identifies the branch, and *AAA* is a three-digit code that identifies the institution. For example, *25539-001*.
number/cpa_transit_eft/ca	Canadian Payments Association EFT transit number, in the format *0AAABBBBB*, where *AAA* is a three-digit code that identifies the institution, and *BBBB* is a five-digit code that identifies the branch. The first digit is always a leading zero. For example, *000125539*.
number/cpa_transit/ca	Canadian Payments Association transit number.
number/bankaccount/ca	Canadian bank account number. The entity recognizes known account number formats for particular banks, and generic seven- or 12-digit numbers, but assigns higher scores to known formats. This entity does not include the CPA transit numbers.

number_banking_de.ecr

Entity	Description
number/sort_code/de	8-digit German bank sort code. For example, *10019610*.
number/bank_number/de	German bank account number.

number_banking_fr.ecr

Entity	Description
number/bankaccount/fr	French bank account number.

number_banking_gb.ecr

Entity	Description
number/sortcode/gb	United Kingdom bank sort code. *For example, 301007, 30-10-07, or 30 10 07.* This entity recognizes any valid sort code, but assigns higher scores to known formats from several banks.
number/bankaccount/gb	United Kingdom bank account number, including the sort code. The sort code and account number must be separated by white space. The account number can be any eight-digit number.

number_banking_ie.ecr

Entity	Description
number/sortcode/ie	Ireland bank sort code. For example, *906005*, *90-60-05*, or *90 60 05*.
number/bankaccount/ie	Ireland bank account number.

number_banking_us.ecr

Entity	Description
number/aba_micr/us	American Bankers Association MICR transit number, in the format *XXXXYYYYC*, where *XXXX* is the Federal Reserve Routing Symbol, *YYYY* is the ABA Institution Identifier, and *C* is the check digit. For example, *129131673*. The `checksum_aba_rtn.lua` post-processing script can verify the MICR check digit.
number/aba_fraction/us	American Bankers Association fraction transit number, in the format *PP-YYYY/XXXX*, where *PP* is a one-digit or two-digit prefix that represents the bank's check processing center location, *YYYY* is the ABA Institution identifier, and *XXXX* is the Federal Reserve Routing Symbol.

number_banking_us.ecr, continued

Entity	Description
number/aba_routing/us	American Bankers Association transit number. The `checksum_aba_rtn.lua` post-processing script can verify the MICR check digit.
number/bankaccount/us	United States bank account number, including the American Bankers Association routing number, in fraction or MICR format. The routing information and account information must be separated by a single space. The account number can be four to 17 digits, but nine-, ten-, and 12-digit numbers are given higher scores. The `checksum_aba_rtn.lua` post-processing script can verify the MICR check digit.
number/cusip/us	United States Committee on Uniform Securities Identification Procedures (CUSIP) number. The `postprocessing_cusip.lua` post-processing script can validate the checksum and set the score for this entity.
number/cusip_ppn/us	A CUSIP number that contains characters from the insurance industry that denote private placement numbers. This entity includes all the values from number/cusip/us. The `postprocessing_cusip.lua` post-processing script can validate the checksum and set the score for this entity.
number/cusip_internal/us	A CUSIP number from the range reserved for internal use. This entity is distinct from number/cusip/us and number/cusip_ppn/us. The `postprocessing_cusip.lua` post-processing script can set the score for this entity.

number_bsn_nl.ecr

Entity	Description
number/bsn/nl	Dutch Citizen Service Numbers (burgerservicenummer). BSNs always consist of nine digits. The `number_bsn_nl.lua` post-processing script performs validation for this entity.

number_cc.ecr

Entity	Description
number/cc12dn	12-digit credit card numbers with no delimiters.
number/cc12dh	12-digit credit card numbers with hyphen delimiters.
number/cc12ds	12-digit credit card numbers with space delimiters.
number/cc12	All 12-digit credit card numbers.
number/cc13dn	13-digit credit card numbers with no delimiters.
number/cc13dh	13-digit credit card numbers with hyphen delimiters.
number/cc13ds	13-digit credit card numbers with space delimiters.
number/cc13	All 13-digit credit card numbers.
number/cc14dn	14-digit credit card numbers with no delimiters.
number/cc14dh	14-digit credit card numbers with hyphen delimiters.
number/cc14ds	14-digit credit card numbers with space delimiters.
number/cc14	All 14-digit credit card numbers.
number/cc15dn	15-digit credit card numbers with no delimiters.
number/cc15dh	15-digit credit card numbers with hyphen delimiters.
number/cc15ds	15-digit credit card numbers with space delimiters.
number/cc15	All 15-digit credit card numbers.
number/cc16dn	16-digit credit card numbers with no delimiters.
number/cc16dh	16-digit credit card numbers with hyphen delimiters.
number/cc16ds	16-digit credit card numbers with space delimiters.
number/cc16	All 16-digit credit card numbers.
number/cc17dn	17-digit credit card numbers with no delimiters.
number/cc17dh	17-digit credit card numbers with hyphen delimiters.
number/cc17ds	17-digit credit card numbers with space delimiters.
number/cc17	All 17-digit credit card numbers.
number/cc18dn	18-digit credit card numbers with no delimiters.
number/cc18dh	18-digit credit card numbers with hyphen delimiters.

number_cc.ecr, continued

Entity	Description
number/cc18ds	18-digit credit card numbers with space delimiters.
number/cc18	All 18-digit credit card numbers.
number/cc19dn	19-digit credit card numbers with no delimiters.
number/cc19dh	19-digit credit card numbers with hyphen delimiters.
number/cc19ds	19-digit credit card numbers with space delimiters.
number/cc19	All 19-digit credit card numbers.
number/ccdn	All credit card numbers with no delimiters.
number/ccdh	All credit card numbers with hyphen delimiters.
number/ccds	All credit card numbers with space delimiters.
number/cc	Any credit card number. Micro Focus Eduction supports the following credit card formats: • American Express • Bankcard • BORICA • China_T_Union • China Union Pay • Dankort • Diners Club Carte Blanche • Diners Club International • Diners Club enRoute • Discover • Humo • InstaPayment • InterPayment • JCB • LankaPay • Laser • Maestro

number_cc.ecr, continued

Entity	Description
	• Mastercard • Mir • NPS Pridnestrovie • RuPay • Solo • Switch • Troy • UATP • UkrCard • UzCard • Verve • Visa
number/cc_amex	American Express credit card number. American Express credit card account numbers are 15 digits in lengths, and generally start with either *34* or *37*. For example, *378124403602370*.
number/cc_bankcard	Bankcard credit card number (discontinued in 2006).
number/cc_borica	BORICA (Bulgarian national payment system) credit card number. BORICA credit card numbers are 16 digits long and start with 2205. For example, *2205238479827383*.
number/cc_china_T_union	China T-Union credit card number. China T-Union credit card numbers are 19 digits long and start with 31. For example *3138922093929320898*.
number/cc_china_union_pay	China UnionPay credit card number. Most China UnionPay card numbers have prefixes from *620* to *625*, and range in length from 16 to 19 characters.
number/cc_dankort	Dankort credit card number. Dankort credit card numbers are 16 digits long and start with 5019 or 4571. For example, *5019283498292382*.
number/cc_diners_club	Diners Club credit card number. Most Diners Club credit card numbers are 16 or 14 digits long. For example, *30544726571210* (Carte Blanche), *36072371463677* (International), or

number_cc.ecr, continued

Entity	Description
	5484308289255581 (North America).
number/cc_discover	Discover credit card number. Discover credit card numbers start with *6011*, *622126* to *622925*, *644*, *645*, *646*, *647*, *648*, *649*, or *65*, and are 16 digits long. For example, *6011541256841963*.
number/cc_humo	Humo credit card number. Humo credit card numbers are 16 digits long, start with 9860 and do not use Luhn algorithm validation. For example, *9860327840103939*.
number/cc_instapayment	InstaPayment credit card number. InstaPayment credit card numbers start with either *637*, *638*, or *639*, and are 16 digits long. For example, *6393519709142682*.
number/cc_interpayment	Interpayment credit card number. InterPayment credit card numbers are 16-19 digits long and start with *636*. For example, *6363287498237230*.
number/cc_jcb	JCB credit card number. JCB credit card numbers consist of 16 digits. Either the first four digits must be *3088*, *3096*, *3112*, *3158*, or *3337*, or the first eight digits must be in the range *35280000* to *35899999*. For example, *3158745776935953*.
number/cc_lankapay	LankaPay credit card number. LankaPay credit card numbers are 16 digits long and start with *357111*. For example, *3571117236428731*.
number/cc_laser	Laser credit card number (discontinued in 2014). Laser credit card numbers start with *6304*, *6706*, *6771*, or *6709*, and are between 16 to 19 digits long. For example, *6709682431878947*.
number/cc_maestro	Maestro credit card number. Maestro credit card numbers start with *5018*, *5020*, *5038*, *5893*, *6304*, *6759*, *6761*, *6762*, or *6763*, and are between 16 to 19 digits long (although they can have as few as 12 digits). For example, *5018452935461261*.
number/cc_mastercard	Mastercard credit card number. Mastercard credit card numbers start with *51*, *52*, *53*, *54*, or *55*, and are between 16 to 19 digits long.
number/cc_mir	Mir credit card number. Mir credit card numbers are 16-19 digits long and start with numbers in the range

number_cc.ecr, continued

Entity	Description
	2200-2204. For example *2202263487236370*.
number/cc_nps_pridnestrovie	NPS Pridnestrovie credit card number. NPS Pridnestrovie credit card numbers are 16 digits long and start with numbers in the range *6054740–6054744*. For example, *6054740234233891*.
number/cc_rupay	RuPay credit card number. RuPay credit card numbers are 16 digits long and start with *60*, *65*, *81*, *82*, *508*, *353* or *356*. For example, *5081273872817824*.
number/cc_solo	16-digit, 18-digit, or 19-digit Solo credit card number (discontinued in 2011). For example, *6331101999990016*.
number/cc_switch	16-digit, 18-digit, or 19-digit Switch credit card number (rebranded as Maestro). Switch credit card numbers begin with *4903*, *4905*, *4911*, *4936*, *564182*, *633110*, *6333*, or *6759*.
number/cc_troy	Troy credit card number. Troy credit card numbers are 16 digits long and start with 65 or 9792. For example, *9792237482368926*.
number/cc_uatp	UATP credit card number. UATP credit card numbers are 15 digits long and start with 1. For example, *139248729302397*.
number/cc_ukrcard	UkrCard credit card number. UkrCard credit card numbers are 16-19 digits in length and start with numbers in the range *60400100–60420099*. For example, *6040010037281738*.
number/cc_uzcard	UzCard credit card number. UzCard credit card numbers are 16 digits long, start with 8600 and do not use Luhn algorithm validation. For example, *8600234892749881*.
number/cc_verve	Verve credit card number. Verve credit card numbers are 16, 18, or 19 digits long and start with numbers in the ranges *506099–506198*, *650002–650027* or *507865-507964*. For example, *5060993428473286*.
number/cc_visa	Visa credit card number. Most Visa credit card numbers start with *4* and are 16 digits long; however, there are a few that consist of 13 digits. The numbers are always spaced in four groups of four digits each. For example, *4929 8198 5006 5312*.

number_dni_es.ecr

Entity	Description
number/dni/es	Spanish DNI (national identity card). **NOTE:** You must set `EnableComponents` to **`True`** for the provided post-processing Lua script to work.

number_driverlic_ca.ecr

Entity	Description
number/driverlic/AB/ca number/driverlic/BC/ca number/driverlic/MB/ca number/driverlic/NB/ca number/driverlic/NL/ca number/driverlic/NS/ca number/driverlic/NT/ca number/driverlic/NU/ca number/driverlic/ON/ca number/driverlic/PE/ca number/driverlic/QC/ca number/driverlic/SK/ca number/driverlic/YT/ca	Driver's licence number for each Canadian province and territory. The two-letter codes are defined by ISO 3166-2.
number/driverlic/ca	All Canadian driver's licence numbers.

number_driverlic_de.ecr

Entity	Description
number/driverlic/de	German driver's licence number.

number_driverlic_fr.ecr

Entity	Description
number/driverlic/fr	French driver's licence number.

number_driverlic_gb.ecr

Entity	Description
number/driverlic/gb	United Kingdom driving licence number.

number_driverlic_us.ecr

Entity	Description
number/driverlic/AL/us number/driverlic/AK/us number/driverlic/AR/us number/driverlic/AZ/us number/driverlic/CA/us number/driverlic/CO/us number/driverlic/CT/us number/driverlic/DC/us number/driverlic/DE/us number/driverlic/FL/us number/driverlic/GA/us number/driverlic/HI/us number/driverlic/IA/us number/driverlic/ID/us number/driverlic/IL/us number/driverlic/IN/us number/driverlic/KS/us number/driverlic/KY/us number/driverlic/LA/us number/driverlic/MA/us number/driverlic/MD/us number/driverlic/ME/us number/driverlic/MI/us number/driverlic/MN/us number/driverlic/MO/us	Driver’s licence number for each U.S. state.

number_driverlic_us.ecr, continued

Entity	Description
number/driverlic/MS/us number/driverlic/MT/us number/driverlic/NC/us number/driverlic/ND/us number/driverlic/NE/us number/driverlic/NH/us number/driverlic/NJ/us number/driverlic/NM/us number/driverlic/NV/us number/driverlic/NY/us number/driverlic/OH/us number/driverlic/OK/us number/driverlic/OR/us number/driverlic/PA/us number/driverlic/RI/us number/driverlic/SC/us number/driverlic/SD/us number/driverlic/TN/us number/driverlic/TX/us number/driverlic/UT/us number/driverlic/VA/us number/driverlic/VT/us number/driverlic/WA/us number/driverlic/WV/us number/driverlic/WI/us number/driverlic/WY/us	
number/driverlic/us	All U.S. driver’s licence numbers.

number_fuel.ecr

Entity	Description
number/fuel_voyager	A Voyager fleet card number, with various permitted number groupings and delimiters.

number_iban.ecr

Entity	Description
number/ibandn/albania number/ibands/albania number/ibandn/andorra number/ibands/andorra number/ibandn/austria number/ibands/austria number/ibandn/azerbaijan number/ibands/azerbaijan number/ibandn/bahrain number/ibands/bahrain number/ibandn/belarus number/ibands/belarus number/ibandn/belgium number/ibands/belgium number/ibandn/bosniaherzegovina number/ibands/bosniaherzegovina number/ibandn/bulgaria number/ibands/bulgaria number/ibandn/brazil number/ibands/brazil number/ibandn/costarica number/ibands/costarica number/ibandn/croatia number/ibands/croatia	Undelimited (dn) or space-delimited (ds) International Bank Account Number (IBAN) for each country. For more information on IBAN formatting requirements for each country, see https://www.iban.com/structure.html. **NOTE:** North Macedonia is the name for Macedonia since February 2019. The Macedonia entity remains in the grammar for compatibility. It matches the same patterns as North Macedonia.

number_iban.ecr, continued

Entity	Description
number/ibandn/cyprus	
number/ibands/cyprus	
number/ibandn/czechrepublic	
number/ibands/czechrepublic	
number/ibandn/denmark	
number/ibands/denmark	
number/ibandn/dominicanrepublic	
number/ibands/dominicanrepublic	
number/ibandn/egypt	
number/ibands/egypt	
number/ibandn/elsalvador	
number/ibands/elsalvador	
number/ibandn/estonia	
number/ibands/estonia	
number/ibandn/faroeislands	
number/ibands/faroeislands	
number/ibandn/finland	
number/ibands/finland	
number/ibandn/france	
number/ibands/france	
number/ibandn/georgia	
number/ibands/georgia	
number/ibandn/germany	
number/ibands/germany	
number/ibandn/gibraltar	
number/ibands/gibraltar	
number/ibandn/greece	
number/ibands/greece	
number/ibandn/greenland	

number_iban.ecr, continued

Entity	Description
number/ibands/greenland	
number/ibandn/guatemala	
number/ibands/guatemala	
number/ibandn/holysee	
number/ibands/holysee	
number/ibandn/hungary	
number/ibands/hungary	
number/ibandn/iceland	
number/ibands/iceland	
number/ibandn/iraq	
number/ibands/iraq	
number/ibandn/ireland	
number/ibands/ireland	
number/ibandn/israel	
number/ibands/israel	
number/ibandn/italy	
number/ibands/italy	
number/ibandn/jordan	
number/ibands/jordan	
number/ibandn/kazakhstan	
number/ibands/kazakhstan	
number/ibandn/kosovo	
number/ibands/kosovo	
number/ibandn/kuwait	
number/ibands/kuwait	
number/ibandn/latvia	
number/ibands/latvia	
number/ibandn/lebanon	
number/ibands/lebanon	

number_iban.ecr, continued

Entity	Description
number/ibandn/liechtenstein	
number/ibands/liechtenstein	
number/ibandn/lithuania	
number/ibands/lithuania	
number/ibandn/luxembourg	
number/ibands/luxembourg	
number/ibandn/macedonia	
number/ibands/macedonia	
number/ibandn/malta	
number/ibands/malta	
number/ibandn/mauritania	
number/ibands/mauritania	
number/ibandn/mauritius	
number/ibands/mauritius	
number/ibandn/moldova	
number/ibands/moldova	
number/ibandn/monaco	
number/ibands/monaco	
number/ibandn/montenegro	
number/ibands/montenegro	
number/ibandn/netherlands	
number/ibands/netherlands	
number/ibandn/northmacedonia	
number/ibands/northmacedonia	
number/ibandn/norway	
number/ibands/norway	
number/ibandn/pakistan	
number/ibands/pakistan	
number/ibandn/palestine	

number_iban.ecr, continued

Entity	Description
number/ibands/palestine	
number/ibandn/poland	
number/ibands/poland	
number/ibandn/portugal	
number/ibands/portugal	
number/ibandn/qatar	
number/ibands/qatar	
number/ibandn/romania	
number/ibands/romania	
number/ibandn/saintlucia	
number/ibands/saintlucia	
number/ibandn/sanmarino	
number/ibands/sanmarino	
number/ibandn/saotomeprincipe	
number/ibands/saotomeprincipe	
number/ibandn/saudiarabia	
number/ibands/saudiarabia	
number/ibandn/serbia	
number/ibands/serbia	
number/ibandn/seychelles	
number/ibands/seychelles	
number/ibandn/slovakrepublic	
number/ibands/slovakrepublic	
number/ibandn/slovenia	
number/ibands/slovenia	
number/ibandn/spain	
number/ibands/spain	
number/ibandn/sweden	
number/ibands/sweden	

number_iban.ecr, continued

Entity	Description
number/ibandn/switzerland number/ibands/switzerland number/ibandn/timorleste number/ibands/timorleste number/ibandn/tunisia number/ibands/tunisia number/ibandn/turkey number/ibands/turkey number/ibandn/ukraine number/ibands/ukraine number/ibandn/unitedarabemirates number/ibands/unitedarabemirates number/ibandn/unitedkingdom number/ibands/unitedkingdom number/ibandn/virginislandsbritish number/ibands/virginislandsbritish	
number/ibandn	All IBAN numbers without delimiters.
number/ibands	All IBAN numbers with space delimiters.

number_insee_fr.ecr

Entity	Description
number/insee/fr	French INSEE number. INSEE numbers are composed of 13 digits and a two-digit key. `Score="0.2"` is used for examples with unspecified months.

number_licenseplate_ca.ecr

Entity	Description
number/licenseplate/AB/ca	Licence plate numbers for each Canadian province

number_licenseplate_ca.ecr, continued

Entity	Description
number/licenseplate/BC/ca number/licenseplate/MB/ca number/licenseplate/NB/ca number/licenseplate/NL/ca number/licenseplate/NT/ca number/licenseplate/NS/ca number/licenseplate/NU/ca number/licenseplate/ON/ca number/licenseplate/PE/ca number/licenseplate/QC/ca number/licenseplate/SK/ca number/licenseplate/YT/ca	and territory.
number/licenseplate/ca	All Canadian licence plate numbers.

number_licenseplate_de.ecr

Entity	Description
number/licenseplate/de	German vehicle number plate.

number_licenseplate_es.ecr

Entity	Description
number/licenseplate/es	Spanish vehicle number plate.

number_licenseplate_fr.ecr

Entity	Description
number/licenseplate/fr	French vehicle registration number.

number_licenseplate_gb.ecr

Entity	Description
number/licenseplate/gb	United Kingdom vehicle registration number.

number_licenseplate_us.ecr

Entity	Description
number/licenseplate/AL/us number/licenseplate/AK/us number/licenseplate/AR/us number/licenseplate/AZ/us number/licenseplate/CA/us number/licenseplate/CO/us number/licenseplate/CT/us number/licenseplate/DE/us number/licenseplate/DC/us number/licenseplate/FL/us number/licenseplate/GA/us number/licenseplate/HI/us number/licenseplate/IA/us number/licenseplate/ID/us number/licenseplate/IL/us number/licenseplate/IN/us number/licenseplate/KS/us number/licenseplate/KY/us number/licenseplate/LA/us number/licenseplate/MA/us number/licenseplate/MD/us number/licenseplate/ME/us number/licenseplate/MI/us number/licenseplate/MN/us number/licenseplate/MO/us	Licence plate numbers for each U.S. state.

number_licenseplate_us.ecr, continued

Entity	Description
number/licenseplate/MS/us number/licenseplate/MT/us number/licenseplate/NC/us number/licenseplate/ND/us number/licenseplate/NE/us number/licenseplate/NH/us number/licenseplate/NJ/us number/licenseplate/NM/us number/licenseplate/NV/us number/licenseplate/NY/us number/licenseplate/OH/us number/licenseplate/OK/us number/licenseplate/OR/us number/licenseplate/PA/us number/licenseplate/RI/us number/licenseplate/SC/us number/licenseplate/SD/us number/licenseplate/TN/us number/licenseplate/TX/us number/licenseplate/UT/us number/licenseplate/VA/us number/licenseplate/VT/us number/licenseplate/WA/us number/licenseplate/WV/us number/licenseplate/WI/us number/licenseplate/WY/us	
number/licenseplate/us	United States license plate number.

number_mac_address.ecr

Entity	Description
number/EUI48dh	MAC address in EUI-48 format (hyphen-separated). For example, *01-23-45-67-89-Ab*
number/EUI48dc	MAC address in EUI-48 format (colon-separated). For example, *01:23:45:67:89:Ab*
number/EUI48	MAC address in EUI-48 format. For example, *01-23-45-67-89-Ab*
number/EUI64dh	MAC address in EUI-64 format (hyphen-separated). For example, *01-23-45-67-89-ab-CD-eF*
number/EUI64dc	MAC address in EUI-64 format (colon-separated). For example, *01:23:45:67:89:ab:CD:eF*
number/EUI64	MAC address in EUI-64 format. For example, *01-23-45-67-89-ab-CD-eF*

number_ni_gb.ecr

Entity	Description
number/nids/gb	UK National Insurance number with space delimiters.
number/nidn/gb	UK National Insurance number without delimiters.
number/nidh/gb	UK National Insurance number with hyphen delimiters.
number/ni/gb	Any UK National Insurance number. The format of the number is two prefix letters, six digits, and one suffix letter.

number_passport_engca.ecr

Entity	Description
number/passport_number/engca	Canadian passport number (in any context).
number/passport_context/engca	Canadian passport number (when found in English-language context).

number_passport_enggb.ecr

Entity	Description
number/passport_context/enggb	UK passport number (when found in English-language context).

number_passport_engus.ecr

Entity	Description
number/passport_context/engus	U.S. passport number (when found in English-language context).

number_passport_freca.ecr

Entity	Description
number/passport_number/freca	French Canadian passport number (in any context).
number/passport_context/freca	French Canadian passport number (when found in French-language context).

number_passport_frefr.ecr

Entity	Description
number/passport_number/frefr	French passport number (in any context).
number/passport_context/frefr	French passport number (when found in French-language context).

number_passport_gerde.ecr

Entity	Description
number/passport_context/gerde	German passport number (when found in German-language context).

number_phone_au.ecr

Entity	Description
phone/landline/au	A complete landline phone number in Australia. To ensure that this entity performs correctly, set

number_phone_au.ecr, continued

Entity	Description
	`TangibleCharacters` to include '+' and '('.
phone/mobile/au	A complete mobile phone number in Australia. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/other/au	A complete 08- or 09- phone number in Australia. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/all/au	Any complete phone number in Australia. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_be.ecr

Entity	Description
phone/landline/be	A complete landline phone number in Belgium. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/mobile/be	A complete mobile phone number in Belgium. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/other/be	A complete 08- or 09- phone number in Belgium. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/all/be	Any complete phone number in Belgium. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_ca.ecr

Entity	Description
phone/numds/ca	A numeric-only Canadian phone number, delimited by spaces. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/numdh/ca	A numeric-only Canadian phone number, delimited by hyphens. To ensure that this entity performs

number_phone_ca.ecr, continued

Entity	Description
	correctly, set `TangibleCharacters` to include '+' and '(.
phone/numdd/ca	A numeric-only Canadian phone number, delimited by dots. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '(.
phone/numdn/ca	An undelimited, numeric-only Canadian phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '(.
phone/num/ca	Any numeric-only Canadian phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '(.
phone/alphanumds/ca	An alphanumeric Canadian phone number, delimited by spaces.
phone/alphanumdh/ca	An alphanumeric Canadian phone number, delimited by hyphens. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '(.
phone/alphanumdd/ca	An alphanumeric Canadian phone number, delimited by dots.
phone/alphanumdn/ca	An undelimited, alphanumeric Canadian phone number.
phone/alphanum/ca	Any alphanumeric Canadian phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '(.

number_phone_cn.ecr

Entity	Description
phone/landline/cn	A Chinese landline phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '(.
phone/mobile/cn	A Chinese mobile phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '(.
phone/tollfree/cn	A Chinese toll free phone number.

number_phone_cn.ecr, continued

Entity	Description
phone/all/cn	Any Chinese phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_de.ecr

Entity	Description
phone/landline/de	A complete landline phone number in Germany. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/mobile/de	A complete mobile phone number in Germany. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/other/de	A complete freephone or premium phone number in Germany. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/all/de	Any complete German phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_es.ecr

Entity	Description
phone/landline/es	A complete landline phone number in Spain. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/mobile/es	A complete mobile phone number in Spain. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/other/es	A complete freephone or premium phone number in Spain. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/all/es	Any complete phone number in Spain. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_fr.ecr

Entity	Description
phone/landline/fr	A complete landline phone number in France. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/mobile/fr	A complete mobile phone number in France. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/other/fr	A complete 08- or 09- phone number in France. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/all/fr	Any complete phone number in France. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_gb.ecr

Entity	Description
phone/areacode/gb	United Kingdom area code.
phone/landline/gb	A complete landline phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/mobile/gb	A complete mobile phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/freephone/gb	A complete freephone phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/business/gb	A complete 08- or 09- phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/non_geographic/gb	A complete non-geographic phone number in the United Kingdom. For example, *0345 678 579 40.*To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_gb.ecr, continued

Entity	Description
phone/personal/gb	A complete 070- phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/all/gb	Any complete phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_it.ecr

Entity	Description
phone/landline/it	A complete landline phone number in Italy. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/mobile/it	A complete mobile phone number in Italy. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/other/it	A premium rate, freephone, or shared-cost phone number in Italy. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/all/it	Any complete phone number in Italy. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_lu.ecr

Entity	Description
phone/landline/lu	A complete landline phone number in Luxembourg. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/mobile/lu	A complete mobile phone number in Luxembourg. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/all/lu	Any complete phone number in Luxembourg. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_nl.ecr

Entity	Description
phone/landline/nl	A complete landline phone number in the Netherlands. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/mobile/nl	A complete mobile phone number in the Netherlands. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/other/nl	A complete 08- or 09- phone number in the Netherlands. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/all/nl	Any complete phone number in the Netherlands. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_pt.ecr

Entity	Description
phone/landline/pt	A complete landline phone number in Portugal. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/mobile/pt	A complete mobile phone number in Portugal. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/other/pt	Other complete phone number in Portugal. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/all/pt	Any complete phone number in Portugal. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_us.ecr

Entity	Description
phone/numds/us	A numeric-only U.S. phone number, delimited by spaces. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

number_phone_us.ecr, continued

Entity	Description
phone/numdh/us	A numeric-only U.S. phone number, delimited by hyphens. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/numdd/us	A numeric-only U.S. phone number, delimited by dots. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/numdn/us	An undelimited, numeric-only U.S. phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/num/us	Any numeric-only U.S. phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
phone/alphanumds/us	An alphanumeric U.S. phone number, delimited by spaces.
phone/alphanumdh/us	An alphanumeric U.S. phone number, delimited by hyphens.
phone/alphanumdd/us	An alphanumeric U.S. phone number, delimited by dots.
phone/alphanumdn/us	An undelimited, alphanumeric U.S. phone number.
phone/alphanum/us	Any alphanumeric U.S. phone number.

number_sin_ca.ecr

Entity	Description
number/sindh/ca	Canadian social insurance number with dash delimiters.
number/sinds/ca	Canadian social insurance number with space delimiters.
number/sindn/ca	Canadian social insurance number without delimiters.
number/sin/ca	Any Canadian social security number.

number_ss_us.ecr

Entity	Description
number/ssdh/us	Social Security number with dash delimiters.
number/ssdsh/us	Social Security number with soft hyphen delimiters.
number/ssds/us	Social Security number with space delimiters.
number/ssdnbs/us	Social Security number with non-breaking space delimiters.
number/ssdn/us	Social Security number without delimiters.
number/ss/us	Any Social Security number.
number/medicareid/us	Medicare ID.

number_swiftcode.ecr

Entity	Description
number/swiftcode	Swift code.

number_telecoms.ecr

Entity	Description
number/imei	International Mobile Station Equipment Identity number.
number/imeisv	International Mobile Station Equipment Identity Software Version number.
number/meid_hex	Mobile Equipment Identifier (hexadecimal format).
number/iccid	Integrated Circuit Card Identifier number.
number/imsi	International Mobile Subscriber Identity number.
number/plmn	Public Land Mobile Network number.
number/msisdn	Mobile Subscriber Integrated Services Digital Network number.

number_types_chi.ecr

Entity	Description
number/one_to_nine/chi	The numbers one to nine in Chinese.
number/zero_to_nine/chi	The numbers zero to nine in Chinese.
number/zero_to_twelve/chi	The numbers zero to 12 in Chinese.
number/zero_to_twenty_four/chi	The numbers zero to 24 in Chinese.
number/one_to_thirty_one/chi	The numbers one to 31 in Chinese.
number/zero_to_fifty_five/chi	The numbers zero to 55 in steps of five in Chinese.
number/zero_to_fifty_nine/chi	The numbers zero to 59 in Chinese.
number/one_to_ninety_nine/chi	The numbers one to 99 in Chinese.
number/one_to_one_hundred/chi	The numbers one to 100 in Chinese.
number/all/chi	Large numbers in Chinese.
number/num/chi	A simple string of digits that does not start with a zero in Chinese.
number/digits/chi	String of digits in Chinese.
number/fraction/chi	A simple fraction consisting of two strings of digits that do not start with a zero. For example, *-12/13, 1/5*.

number_types_eng.ecr

Entity	Description
number/num/eng	A simple string of digits that does not start with a zero. For example, *123*.
number/ncomma/eng	A number without commas. For example, *123456.*
number/comma/eng	A number with commas. For example, *123,456.*
number/sign/eng	A sign. For example +, -, *plus*, *minus.*
number/natural/eng	A natural number. For example, *123,456.*
number/int/eng	An integer. For example, *-123*,*456*, *minus 2*, *+20.*
number/real/eng	A real number. For example, *123.456*, *-123.456.*
number/fraction/eng	A simple fraction of two unsigned strings of digits that do not start with a zero. For example, *12/13*.

number_types_eng.ecr, continued

Entity	Description
number/fracalpha/eng	An alphabetical fraction. For example, *one half*, *three sixths*, *three and five ninths.*
number/fracnum/eng	Numeric fractions. For example, *-12/13.*
number/fracmixed/eng	Mixed fractions. For example, *1 twelfth*, *8 fourteenths*, *3 and five ninths.*
number/pcnt/eng	Percent. For example, *100%*, *100 percent*, *12.78%.*
number/suff/eng	Numbers with a suffix. For example, *1st*, *3rd.*
number/suffalpha/eng	Fractions with mixed alphabetical and numeric terms. For example, *a 12th*, *three 3rds.*
number/doz/eng	Number based on dozen. For example, *half a dozen*, *2 dozen*, *three and a half dozen*.
number/alpha/eng	Alphabetical numbers less than 100. For example, *one*, *ten*, *thirty-one.*
number/bigalpha/eng	Big alphabetical numbers.
number/bignum/eng	Big numeric abbreviated numbers.
number/big/eng	A big number, alphabetical or numeric abbreviated.
number/sci/eng	A number in scientific notation. For example, *1.23x10^11*, *1.23E+5, 6.1^-3.*
number/fullalpha/eng	A fully written out number up to 999,999,999,999,999. For example, *one thousand two hundred and thirty four*.
number/numord/eng	An ordinal number up to 999. For example, *thirty fourth*.
number/num_plurals/eng	Plural numbers. For example, *dozens, millions*.

number_types_fre.ecr

Entity	Description
number/num/fre	A simple string of digits that does not start with 0. For example, *123*.
number/num_sep/fre	A number with separators. For example, *123.456.789*.

number_types_fre.ecr, continued

Entity	Description
number/digits/fre	A string of digits. For example, *00123*.
number/natural/fre	A natural number. For example, *123.456.789* or *123456789.*
number/fraction/fre	A simple fraction of two unsigned strings of digits that do not start with a zero. For example, *12/13*.
number/int/fre	An integer. For example, *-123.456.789*; *moins 2*; *20*.
number/real/fre	A real number. For example, *123.456*, *-123.456*.
number/numalpha/fre	A fully written out number up to 999. For example, *deux cent trente-quatre*.
number/numord/fre	An ordinal number up to 999. For example, *trente quatrième*.

number_vin.ecr

Entity	Description
number/vin/wmi	The world manufacturer identifier section (3 characters) of a vehicle identification number.
number/vin/vds	The vehicle descriptor section (6 characters) of a vehicle identification number.
number/vin/model_year	The model year character of a vehicle identification number.
number/vin/plant_code	The plant code character of a vehicle identification number.
number/vin/seq_number	The vehicle identifier section sequential number (6 characters) of a vehicle identification number.
number/vin/vis	The vehicle identifier section (8 characters) of a vehicle identification number.
number/vin	A vehicle identification number (17 characters).
number/vin/anonymized	An anonymized vehicle identification number (first 9 or 11 characters).

O

organization.ecr

Entity	Description
org/organization	An organization.

P

person_name_chicn.ecr

Entity	Description
person/femalefirstname_s/chicn	Popular simplified Chinese female first name.
person/malefirstname_s/chicn	Popular simplified Chinese male first name.
person/lastname_s/chicn	Popular simplified Chinese last name.
person/firstname/chicn	Chinese first name.
person/namelastfirst/chicn	Chinese last and first name.

person_name_dutnl.ecr

Entity	Description
person/femalefirstname/dutnl	Popular Dutch female first name.
person/malefirstname/dutnl	Popular Dutch male first name.
person/firstname/dutnl	Dutch first name.
person/surname/dutnl	Dutch surname.
person/namefirstmiddlelast/dutn	Dutch first, optional middle, and last name.

person_name_engcn.ecr

Entity	Description
person/femalefirstname/engcn	Popular Chinese female first name in English.
person/femalefirstname_lowercase/engcn	Popular Chinese female first name in lowercase English.

person_name_engcn.ecr, continued

Entity	Description
person/malefirstname/engcn	Popular Chinese male first name in English.
person/malefirstname_lowercase/engcn	Popular Chinese male first name in lowercase English.
person/lastname/engcn	Popular Chinese last name in English.
person/firstname/engcn	Chinese first name in English.
person/namelastfirst/engcn	Chinese last and first name in English.
person/namefirstlast/engcn	Chinese first and last name in English.

person_name_enggb.ecr

Entity	Description
person/femalefirstname/enggb	Popular UK female first name.
person/malefirstname/enggb	Popular UK male first name.
person/lastname/enggb	Popular UK last name.
person/firstname/enggb	UK first name.
person/namefirstlast/enggb	UK first and last name.
person/namefirstmiddlelast/enggb	UK first, optional middle, and last name.

person_name_enggr.ecr

Entity	Description
person/femalefirstname/enggr	Popular Greek female first name.
person/malefirstname/enggr	Popular Greek male first name.
person/lastname/enggr	Popular Greek last name.
person/firstname/enggr	Greek first name.
person/namefirstlast/enggr	Greek first and last name.

person_name_engin.ecr

Entity	Description
person/femalefirstname/engin	Popular Indian female first name.
person/malefirstname/engin	Popular Indian male first name.
person/lastname/engin	Popular Indian last name.
person/firstname/engin	Indian first name.
person/namefirstlast/engin	Indian first and last name.
person/namefirstmiddlelast/engin	Indian first, optional middle, and last name.

person_name_engjp.ecr

Entity	Description
person/femalefirstname/engjp	Popular Japanese female first name in English.
person/malefirstname/engjp	Popular Japanese male first name in romanji.
person/lastname/engjp	Popular Japanese last name in English.
person/firstname/engjp	Japanese first name in English.
person/namelastfirst/engjp	Japanese last and first name in English.

person_name_engru.ecr

Entity	Description
person/femalefirstname/engru	Popular Russian female first name in English.
person/malefirstname/engru	Popular Russian male first name in English.
person/lastname/engru	Popular Russian last name in English.
person/firstname/engru	Russian first name in English.
person/namefirstlast/engru	Russian first and last name in English.
person/namefirstmiddlelast/engru	Russian first, optional middle, and last name in English.

person_name_engus.ecr

Entity	Description
person/femalefirstname/engus	Popular U.S. female first name.
person/malefirstname/engus	Popular U.S. male first name.
person/lastname/engus	Popular U.S. last name.
person/firstname/engus	U.S. first name.
person/compoundlastname/engus	U.S. last name that might be compound.
person/namefirstlast/engus	U.S. first and last name.
person/namefirstmiddlelast/engus	U.S. first, optional middle, and last name.
person/nameinitial/engus	U.S. initialed name.
person/namelastsuffix/engus	Last name and suffix.
person/namefirstneelast/engus	First name and maiden name.
person/namelastcommafirst/engus	Full name in address book format (*Last Name, First Name*).

person_name_frefr.ecr

Entity	Description
person/femalefirstname/frefr	Popular French female first name.
person/malefirstname/frefr	Popular French male first name.
person/lastname/frefr	Popular French last name.
person/firstname/frefr	French first name.
person/namefirstlast/frefr	French first and last name.
person/namefirstmiddlelast/frefr	French first, optional middle, and last name.

person_name_gerde.ecr

Entity	Description
person/femalefirstname/gerde	Popular German female first name.
person/malefirstname/gerde	Popular German male first name.
person/lastname/gerde	Popular German last name.

person_name_gerde.ecr, continued

Entity	Description
person/firstname/gerde	German first name.
person/namefirstmiddlelast/gerde	German first, optional middle, and last name.

person_name_itait.ecr

Entity	Description
person/femalefirstname/itait	Popular Italian female first name.
person/malefirstname/itait	Popular Italian male first name.
person/lastname/itait	Popular Italian last name.
person/firstname/itait	Italian first name.
person/namefirstlast/itait	Italian first and last name.
person/namefirstmiddlelast/itait	Italian first, optional middle, and last name.

person_name_jpnjp.ecr

Entity	Description
person/femalefirstname/jpnjp	Popular Japanese female first name.
person/malefirstname/jpnjp	Popular Japanese male first name in kanj.
person/lastname/jpnjp	Popular Japanese last name in kanj.
person/firstname/jpnjp	Japanese first name.
person/namelastfirst/jpnjp	Japanese last and first name.

person_name_norno.ecr

Entity	Description
person/femalefirstname/norno	Popular Norwegian female first name.
person/malefirstname/norno	Popular Norwegian male first name.
person/lastname/norno	Popular Norwegian last name.
person/firstname/norno	Norwegian first name.

person_name_norno.ecr, continued

Entity	Description
person/namefirstlast/norno	Norwegian first and last name.
person/namefirstmiddlelast/norno	Norwegian first, optional extra given name, optional middle name, and last name.

person_name_rusru.ecr

Entity	Description
person/femalefirstname_unambiguous	Common Russian female first name in Russian that rarely has any alternative meaning.
person/malefirstname_unambiguous	Common Russian male first name in Russian that rarely has any alternative meaning.
person/femalefirstname/rusru	Russian female first name in Russian.
person/femalelastname/rusru	Russian female last name in Russian.
person/malefirstname/rusru	Russian male first name in Russian.
person/malelastname/rusru	Russian male last name in Russian.
person/firstname/rusru	Russian first name in Russian.
person/lastname/rusru	Russian last name in Russian.
person/fullname/rusru	Russian full name in Russian.

person_name_spaes.ecr

Entity	Description
person/femalefirstname/spaes	Popular Spanish female first name.
person/malefirstname/spaes	Popular Spanish male first name.
person/lastname/spaes	Popular Spanish last name.
person/firstname/spaes	Spanish first name.
person/compoundlastname/spaes	Spanish compound last name.
person/namefirstoptionallast/spaes	Spanish first and optional last name.
person/namefirstlast/spaes	Spanish first and last name.

person_name_spaes.ecr, continued

Entity	Description
person/namelastfirst/spaes	Spanish last name, comma, and first name.
person/fullname	Spanish full name.

person_name_swese.ecr

Entity	Description
person/femalefirstname/swese	Popular Swedish female first name.
person/malefirstname/swese	Popular Swedish male first name.
person/lastname/swese	Popular Swedish last name.
person/firstname/swese	Swedish first name.
person/namefirstlast/swese	Swedish first and last name.

person_politician_engus.ecr

Entity	Description
person/poli_hor/engus	Full names of members of the U.S. House of Representatives. For example, *Robert E. Cramer*, *Robert Cramer.*
person/poli_last_hor/engus	Last name of House of Representatives members. For example, *Cramer.*
person/poli_sen/engus	Full name of U.S. senate members.
person/poli_last_sen/engus	Last name of U.S. senate members.
person/poli_gov/engus	Full name of U.S. governors.
person/poli_last_gov/engus	Last name of U.S. governors.
person/poli_cabinet_gw_bush/engus	Full name of a member of the George W. Bush administration.
person/poli_last_cabinet_gw_bush/engus	Last name of a member of the George W. Bush administration.
person/poli_cabinet_obama/engus	Full name of a member of the Barack Obama administration.
person/poli_last_cabinet_obama/engus	Last name of a member of the Barack Obama

person_politician_engus.ecr, continued

Entity	Description
	administration.
person/poli_cabinet_trump/engus	Full name of a member of the Donald Trump administration.
person/poli_last_cabinet_trump/engus	Last name of a member of the Donald Trump administration.
person/poli_other_2012/engus	Full name of other currently active politician. For example, a Presidential nominee.
person/poli_last_other_2012/engus	Last name of other currently active politician. For example, a Presidential nominee.
person/poli_president/engus	Past and present U.S. Presidents.
person/poli_title_hor/engus	Formal title for legislative members. For example, *Congressman Cramer.*
person/poli_title_sen/engus	Formal title for senate members.
person/poli_title_gov/engus	Formal title for governors.
Entity	**Description**
person/politician/jpnjp	Japanese politician.

person_public_figure_chi.ecr

Entity	Description
person/politician/chicn	Chinese politician.
person/legislativecouncil/chihk	Hong Kong legislative council member.
person/entertainer/chi	Chinese entertainer.
person/npc/chicn	Chinese National People's Congress delegate.

person_public_figure_eng.ecr

Entity	Description
person/public_figure/eng	A list of public figures in English.

person_public_figure_jpn.ecr

Entity	Description
person/public_figure/jpn	A list of public figures in Japanese.

person_salutation_eng.ecr

Entity	Description
person/salutation/common/eng	Common salutation. For example, *Mr*.
person/salutation/military/eng	Military salutation.
person/salutation/political/eng	Political salutation.
person/salutation/religious/eng	Religious salutation.
person/salutation/nobility/eng	Salutation of nobility.
person/salutation/eng	Any salutation in English.

person_salutation_fre.ecr

Entity	Description
person/salutation/fre	French salutations. For example, *Madame*, *Mlle*.

person_suffix_eng.ecr

Entity	Description
person/suffixjr/eng	Name suffixes. For example, *Jr.*
person/suffixrmn/eng	Roman suffixes. For example, *III.*
person/suffixacab/eng	Academic suffix – Bachelor's. For example, *BA.*
person/suffixacam/eng	Academic suffix – Master's. For example, *MA.*
person/suffixacad/eng	Academic suffix – Doctoral. For example, *PhD*.
personal/suffixprof/eng	Professional suffix. For example, *MD.*

place_albal.ecr

Entity	Description
place/city1/albal	Albanian settlement with over 100,000 inhabitants.

place_albal.ecr, continued

Entity	Description
place/city1_uppercase/albal	Albanian settlement with over 100,000 inhabitants, in uppercase.
place/city2/albal	Albanian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/albal	Albanian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/county/albal	Albanian county.
place/county_uppercase/albal	Albanian county in uppercase.
place/district/albal	Albanian district.
place/district_uppercase/albal	Albanian district in uppercase.

place_albxk.ecr

Entity	Description
place/city1/albxk	Kosovan settlement with over 100,000 inhabitants.
place/city1_uppercase/albxk	Kosovan settlement with over 100,000 inhabitants, in uppercase.
place/city2/albxk	Kosovan settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/albxk	Kosovan settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/district/albxk	Kosovan district.
place/district_uppercase/albxk	Kosovan district in uppercase.

place_bosba.ecr

Entity	Description
place/city1/bosba	Settlement of Bosnia and Herzegovina with over 100,000 inhabitants.
place/city1_uppercase/bosba	Settlement of Bosnia and Herzegovina with over 100,000 inhabitants, in uppercase.
place/city2/bosba	Settlement of Bosnia and Herzegovina with between

place_bosba.ecr, continued

Entity	Description
	10,000 and 100,000 inhabitants.
place/city2_uppercase/bosba	Settlement of Bosnia and Herzegovina with between 10,000 and 100,000 inhabitants, in uppercase.

place_chicn.ecr

Entity	Description
place/city/chicn	Chinese city.
place/province/chicn	Chinese province.

place_chihk.ecr

Entity	Description
place/district/chihk	District in Hong Kong.
place/island/chihk	Island in Hong Kong.
place/port/chihk	Port in Hong Kong.
place/hospital/chihk	Hospital in Hong Kong.
place/tunnel/chihk	Tunnel in Hong Kong.
place/bridge/chihk	Bridge in Hong Kong.
place/hotel/chihk	Hotel in Hong Kong.
place/locality/chihk	Place in Hong Kong.

place_chitw.ecr

Entity	Description
place/city/chitw	Major divisions of Taiwan, including six special municipalities, three provincial cities, and 13 counties.
place/district/chitw	Subdivisions of the major divisions of Taiwan. The subdivisions include city districts, county-controlled cities, urban townships, and rural townships.

place_countries.ecr

Entity	Description
country/iso_lowercase	ISO 3166-1 alpha-2 country code.
country/all	Country in a local or major language.
country/output_iso	Country in a local or major language (output is normalized to the ISO 3166-1 alpha-2 code).

place_czecz.ecr

Entity	Description
place/city1/czecz	Czech settlement with over 100,000 inhabitants.
place/city1_uppercase/czecz	Czech settlement with over 100,000 inhabitants, in uppercase.
place/city2/czecz	Czech settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/czecz	Czech settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/czecz	Czech region.
place/region_uppercase/czecz	Czech region in uppercase.

place_dandk.ecr

Entity	Description
place/city1/dandk	Danish settlement with over 100,000 inhabitants.
place/city1_uppercase/dandk	Danish settlement with over 100,000 inhabitants, in uppercase.
place/city2/dandk	Danish settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/dandk	Danish settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/dandk	Danish region.
place/region_uppercase/dandk	Danish region in uppercase.
place/municipality/dandk	Danish municipality.

place_dandk.ecr, continued

Entity	Description
place/municipality_uppercase/dandk	Danish municipality in uppercase.
place/island/dandk	Danish island
place/island_uppercase/dandk	Danish island in uppercase

place_dutnl.ecr

Entity	Description
place/city1/dutnl	Dutch settlement with over 100,000 inhabitants.
place/city1_uppercase/dutnl	Dutch settlement with over 100,000 inhabitants, in uppercase.
place/city2/dutnl	Dutch settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/dutnl	Dutch settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/county/dutnl	Dutch county.
place/county_uppercase/dutnl	Dutch county in uppercase.
place/municipality/dutnl	Dutch municipality.
place/municipality_uppercase/dutnl	Dutch municipality in uppercase.
place/island/dutnl	Dutch island.
place/island_uppercase/dutnl	Dutch island in uppercase.

place_dutsr.ecr

Entity	Description
place/city1/dutsr	Surinamese settlement with over 100,000 inhabitants.
place/city1_uppercase/dutsr	Surinamese settlement with over 100,000 inhabitants, in uppercase.
place/city2/dutsr	Surinamese settlement with under 100,000 inhabitants.
place/city2_uppercase/dutsr	Surinamese settlement with under 100,000

place_dutsr.ecr, continued

Entity	Description
	inhabitants, in uppercase.
place/district/dutsr	Surinamese district.
place/district_uppercase/dutsr	Surinamese district in uppercase.

place_engae.ecr

Entity	Description
place/city1/engae	Settlement of the United Arab Emirates with over 100,000 inhabitants.
place/city1_uppercase/engae	Settlement of the United Arab Emirates with over 100,000 inhabitants, in uppercase.
place/city2/engae	Settlement of the United Arab Emirates with under 100,000 inhabitants.
place/city2_uppercase/engae	Settlement of the United Arab Emirates with under 100,000 inhabitants, in uppercase.
place/emirate/engae	Emirate of the United Arab Emirates.
place/emirate_uppercase/engae	Emirate of the United Arab Emirates in uppercase.

place_engau.ecr

Entity	Description
place/state/engau	Australian state or territory.
place/state_uppercase/engau	Australian state or territory in uppercase.
place/state_abbrev/engau	Australian state or territory abbreviations.
place/state_capital/engau	Australian state or territory capitals.
place/state_capital_uppercase/engau	Australian state or territory capitals in uppercase.
place/city1/engau	Australian city with population greater than 100,000.
place/city1_uppercase/engau	Australian city with population greater than 100,000, in uppercase.
place/city2/engau	Australian city with population between 10,000 and 100,000.

place_engau.ecr, continued

Entity	Description
place/city2_uppercase/engau	Australian city with population between 10,000 and 100,000, in uppercase.
place/city/NSW/engau	Settlement in New South Wales.
place/city_uppercase/NSW/engau	Settlement in New South Wales, in uppercase.
place/city/QLD/engau	Settlement in Queensland, Australia.
place/city_uppercase/QLD/engau	Settlement in Queensland, Australia, in uppercase.
place/city/SA/engau	Settlement in South Australia.
place/city_uppercase/SA/engau	Settlement in South Australia, in uppercase.
place/city/TAS/engau	Settlement in Tasmania.
place/city_uppercase/TAS/engau	Settlement in Tasmania, in uppercase.
place/city/VIC/engau	Settlement in Victoria, Australia.
place/city_uppercase/VIC/engau	Settlement in Victoria, Australia, in uppercase.
place/city/WA/engau	Settlement in Western Australia.
place/city_uppercase/WA/engau	Settlement in Western Australia, in uppercase.
place/city/NT/engau	Settlement in Northern Territory, Australia.
place/city_uppercase/NT/engau	Settlement in Northern Territory, Australia, in uppercase.
place/city/ACT/engau	Settlement in Australian Capital Territory.
place/city_uppercase/ACT/engau	Settlement in Australian Capital Territory, in uppercase.
place/city/engau	Australian cities.
place/city_uppercase/engau	Australian cities in uppercase.

place_engbd.ecr

Entity	Description
place/city1/engbd	Bangladeshi settlement with over 100,000 inhabitants.
place/city1_uppercase/engbd	Bangladeshi settlement with over 100,000

place_engbd.ecr, continued

Entity	Description
	inhabitants, in uppercase.
place/city2/engbd	Bangladeshi settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engbd	Bangladeshi settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/division/engbd	Bangladeshi division.
place/division_uppercase/engbd	Bangladeshi division in uppercase.
place/district/engbd	Bangladeshi district.
place/district_uppercase/engbd	Bangladeshi district in uppercase.

place_engbg.ecr

Entity	Description
place/city1/engbg	Bulgarian settlement with over 100,000 inhabitants.
place/city1_uppercase/engbg	Bulgarian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engbg	Bulgarian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engbg	Bulgarian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/engbg	Bulgarian province.
place/province_uppercase/engbg	Bulgarian province in uppercase.

place_engby.ecr

Entity	Description
place/city1/engby	Belarusian settlement with over 100,000 inhabitants.
place/city1_uppercase/engby	Belarusian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engby	Belarusian settlement with between 10,000 and 100,000 inhabitants.

place_engby.ecr, continued

Entity	Description
place/city2_uppercase/engby	Belarusian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/engby	Belarusian region.
place/region_uppercase/engby	Belarusian region in uppercase.

place_engca.ecr

Entity	Description
place/region/engca	Canadian province or territory.
place/region_uppercase/engca	Canadian province or territory in uppercase.
place/region_abbrev/engca	Canadian province or territory abbreviation.
place/region_all/engca	Canadian province or territory full name or abbreviation.
place/region_all_uppercase/engca	Canadian province or territory full name or abbreviation, in uppercase.
place/region_capitals/engca	Canadian provincial or territorial capital.
place/region_capitals_uppercase/engca	Canadian provincial or territorial capital in uppercase.
place/city1/engca	Canadian settlement with over 100,000 inhabitants.
place/city1_uppercase/engca	Canadian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engca	Canadian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engca	Canadian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/city/AB/engca place/city_uppercase/AB/engca place/city/BC/engca place/city_uppercase/BC/engca place/city/MB/engca place/city_uppercase/MB/engca	Settlements in each Canadian province or territory, in normal or uppercase.

place_engca.ecr, continued

Entity	Description
place/city/NB/engca place/city_uppercase/NB/engca place/city/NL/engca place/city_uppercase/NL/engca place/city/NS/engca place/city_uppercase/NS/engca place/city/ON/engca place/city_uppercase/ON/engca place/city/PE/engca place/city_uppercase/PE/engca place/city/QC/engca place/city_uppercase/QC/engca place/city/SK/engca place/city_uppercase/SK/engca place/city/NT/engca place/city_uppercase/NT/engca place/city/NU/engca place/city_uppercase/NU/engca place/city/YT/engca place/city_uppercase/YT/engca	
place/city/engca	Canadian settlement.
place/city_uppercase/engca	Canadian settlement in uppercase.

place_engcn.ecr

Entity	Description
place/city0/engcn	Chinese settlement with over 1,000,000 inhabitants.
place/city0_uppercase/engcn	Chinese settlement with over 1,000,000 inhabitants, in uppercase.

place_engcn.ecr, continued

Entity	Description
place/city1/engcn	Chinese settlement with over 100,000 inhabitants.
place/city1_uppercase/engcn	Chinese settlement with over 100,000 inhabitants, in uppercase.
place/city2/engcn	Chinese settlement with over 10,000 inhabitants.
place/city2_uppercase/engcn	Chinese settlement with over 10,000 inhabitants, in uppercase.
place/province/engcn	Chinese province.
place/province_uppercase/engcn	Chinese province in uppercase.

place_enggb.ecr

Entity	Description
place/possession/enggb	UK crown dependencies.
place/possession_uppercase/enggb	UK crown dependencies in uppercase.
place/country/enggb	UK countries.
place/country_uppercase/enggb	UK countries in uppercase.
place/country_capital/enggb	UK country capitals.
place/country_capital_uppercase/enggb	UK country capitals in uppercase.
place/county/england/enggb	Counties in England.
place/county_uppercase/england/enggb	Counties in England, in uppercase.
place/county/northern_ireland/enggb	Counties in Northern Ireland.
place/county_uppercase/northern_ireland/enggb	Counties in Northern Ireland, in uppercase.
place/county/scotland/enggb	Counties in Scotland.
place/county_uppercase/scotland/enggb	Counties in Scotland, in uppercase.
place/county/wales/enggb	Counties in Wales.
place/county_uppercase/wales/enggb	Counties in Wales, in uppercase.
place/county/enggb	Counties in UK.

place_enggb.ecr, continued

Entity	Description
place/county_uppercase/enggb	Counties in UK in uppercase.
place/city1/enggb	Settlement in the UK with over 100,000 inhabitants.
place/city1_uppercase/enggb	Settlement in the UK with over 100,000 inhabitants, in uppercase.
place/city/england/enggb	Settlements in each UK country, in normal or uppercase.
place/city_uppercase/england/enggb	
place/city/scotland/enggb	
place/city_uppercase/scotland/enggb	
place/city/wales/enggb	
place/city_uppercase/wales/enggb	
place/city/northern_ireland/enggb	
place/city_uppercase/northern_ireland/enggb	
place/city/enggb	UK settlements.
place/city_uppercase/enggb	UK settlements in uppercase.
place/londonborough/enggb	London borough.
place/island/enggb	Major islands of the United Kingdom.
place/island_uppercase/enggb	Major islands of the United Kingdom in uppercase.

place_enggr.ecr

Entity	Description
place/city1/enggr	Greek settlement with over 100,000 inhabitants.
place/city1_uppercase/enggr	Greek settlement with over 100,000 inhabitants, in uppercase.
place/city2/enggr	Greek settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/enggr	Greek settlement with between 10,000 and 100,000 inhabitants, in uppercase.

place_enggr.ecr, continued

Entity	Description
place/region/enggr	Greek region.
place/region_uppercase/enggr	Greek region in uppercase.
place/prefecture/enggr	Greek prefecture (obsolete after 2010).
place/prefecture_uppercase/enggr	Greek prefecture in uppercase (obsolete after 2010).
place/municipality/enggr	Greek municipality.
place/municipality_uppercase/enggr	Greek municipality in uppercase.
place/island/enggr	Greek island.
place/island_uppercase/enggr	Greek island in uppercase.

place_enggy.ecr

Entity	Description
place/city1/enggy	Guyanan settlement with over 100,000 inhabitants.
place/city1_uppercase/enggy	Guyanan settlement with over 100,000 inhabitants, in uppercase.
place/city2/enggy	Guyanan settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/enggy	Guyanan settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/enggy	Guyanan region.
place/region_uppercase/enggy	Guyanan region in uppercase.

place_enghk.ecr

Entity	Description
place/district/enghk	District in Hong Kong.
place/island/enghk	Island in Hong Kong.
place/enghk	Street in Hong Kong.

place_engid.ecr

Entity	Description
place/city1/engid	Indonesian settlement with over 100,000 inhabitants.
place/city1_uppercase/engid	Indonesian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engid	Indonesian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engid	Indonesian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/engid	Indonesian province.
place/province_uppercase/engid	Indonesian province in uppercase.
place/regency/engid	Indonesian regency.
place/regency_uppercase/engid	Indonesian regency in uppercase.
place/island/engid	Indonesian island.
place/island_uppercase/engid	Indonesian island in uppercase.

place_engie.ecr

Entity	Description
place/city1/engie	Irish settlement with over 100,000 inhabitants.
place/city1_uppercase/engie	Irish settlement with over 100,000 inhabitants, in uppercase.
place/city2/engie	Irish settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engie	Irish settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/county_engie	Irish county.
place/county_uppercase/engie	Irish county in uppercase.
place/island/engie	Irish island.
place/island_uppercase/engie	Irish island in uppercase.

place_engin.ecr

Entity	Description
place/city1/engin	Indian settlement with over 100,000 inhabitants.
place/city1_uppercase/engin	Indian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engin	Indian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engin	Indian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/state/engin	Indian state.
place/state_uppercase/engin	Indian state in uppercase.
place/union_territory/engin	Indian union territory.
place/union_territory_uppercase/engin	Indian union territory in uppercase.
place/district/engin	Indian district.
place/district_uppercase/engin	Indian district in uppercase.
place/island/engin	Indian island.
place/island_uppercase/engin	Indian island in uppercase.

place_engir.ecr

Entity	Description
place/city1/engir	Iranian settlement with over 100,000 inhabitants.
place/city1_uppercase/engir	Iranian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engir	Iranian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engir	Iranian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/engir	Iranian province.
place/province_uppercase/engir	Iranian province in uppercase.
place/county/engir	Iranian county.
place/county_uppercase/engir	Iranian county in uppercase.

place_engjp.ecr

Entity	Description
place_city1/engjp	Japanese settlement with over 100,000 inhabitants, in English.
place_city1_uppercase/engjp	Japanese settlement with over 100,000 inhabitants, in uppercase English.
place/city2/engjp	Japanese settlement with between 10,000 and 100,000 inhabitants, in English.
place/city2_uppercase/engjp	Japanese settlement with between 10,000 and 100,000 inhabitants, in uppercase English.
place/special_ward/engjp	Special ward of Tokyo in English.
place/special_ward_uppercase/engjp	Special ward of Tokyo in uppercase English.
place/island/engjp	Japanese island in English.
place/island_uppercase/engjp	Japanese island in uppercase English.
place/prefecture/engjp	Japanese prefectures in English.
place/prefecture_uppercase/engjp	Japanese prefectures in uppercase English.
place/region/engjp	Japanese regions in English.
place/region_uppercase/engjp	Japanese regions in uppercase English.
place/city/aichi/engjp place/city_uppercase/aichi/engjp place/city/akita/engjp place/city_uppercase/akita/engjp place/city/aomori/engjp place/city_uppercase/aomori/engjp place/city/chiba/engjp place/city_uppercase/chiba/engjp place/city/ehime/engjp place/city_uppercase/ehime/engjp place/city/fukui/engjp place/city_uppercase/fukui/engjp place/city/fukuoka/engjp	Cities in each Japanese prefecture in English, in normal or uppercase.

place_engjp.ecr, continued

Entity	Description
place/city_uppercase/fukuoka/engjp	
place/city/fukushima/engjp	
place/city_uppercase/fukushima/engjp	
place/city/gifu/engjp	
place/city_uppercase/gifu/engjp	
place/city/gunma/engjp	
place/city_uppercase/gunma/engjp	
place/city/hiroshima/engjp	
place/city_uppercase/hiroshima/engjp	
place/city/hokkaido/engjp	
place/city_uppercase/hokkaido/engjp	
place/city/hyogo/engjp	
place/city_uppercase/hyogo/engjp	
place/city/ibaraki/engjp	
place/city_uppercase/ibaraki/engjp	
place/city/ishikawa/engjp	
place/city_uppercase/ishikawa/engjp	
place/city/iwate/engjp	
place/city_uppercase/iwate/engjp	
place/city/kagawa/engjp	
place/city_uppercase/kagawa/engjp	
place/city/kagoshima/engjp	
place/city_uppercase/kagoshima/engjp	
place/city/kanagawa/engjp	
place/city_uppercase/kanagawa/engjp	
place/city/kochi/engjp	
place/city_uppercase/kochi/engjp	
place/city/kumamoto/engjp	
place/city_uppercase/kumamoto/engjp	

place_engjp.ecr, continued

Entity	Description
place/city/kyoto/engjp	
place/city_uppercase/kyoto/engjp	
place/city/mie/engjp	
place/city_uppercase/mie/engjp	
place/city/miyagi/engjp	
place/city_uppercase/miyagi/engjp	
place/city/miyazaki/engjp	
place/city_uppercase/miyazaki/engjp	
place/city/nagano/engjp	
place/city_uppercase/nagano/engjp	
place/city/nagasaki/engjp	
place/city_uppercase/nagasaki/engjp	
place/city/nara/engjp	
place/city_uppercase/nara/engjp	
place/city/niigata/engjp	
place/city_uppercase/niigata/engjp	
place/city/oita/engjp	
place/city_uppercase/oita/engjp	
place/city/okayama/engjp	
place/city_uppercase/okayama/engjp	
place/city/okinawa/engjp	
place/city_uppercase/okinawa/engjp	
place/city/osaka/engjp	
place/city_uppercase/osaka/engjp	
place/city/saga/engjp	
place/city_uppercase/saga/engjp	
place/city/saitama/engjp	
place/city_uppercase/saitama/engjp	
place/city/shiga/engjp	

place_engjp.ecr, continued

Entity	Description
place/city_uppercase/shiga/engjp place/city/shimane/engjp place/city_uppercase/shimane/engjp place/city/shizuoka/engjp place/city_uppercase/shizuoka/engjp place/city/tochigi/engjp place/city_uppercase/tochigi/engjp place/city/tokushima/engjp place/city_uppercase/tokushima/engjp place/city/tokyo/engjp place/city_uppercase/tokyo/engjp place/city/tottori/engjp place/city_uppercase/tottori/engjp place/city/toyama/engjp place/city_uppercase/toyama/engjp place/city/wakayama/engjp place/city_uppercase/wakayama/engjp place/city/yamagata/engjp place/city_uppercase/yamagata/engjp place/city/yamaguchi/engjp place/city_uppercase/yamaguchi/engjp place/city/yamanashi/engjp place/city_uppercase/yamanashi/engjp	
place/city/engjp	Japanese cities in English.

place_engkr.ecr

Entity	Description
place/city1_rr/engkr	South Korean settlement with over 100,000 inhabitants, in revised romanization.

place_engkr.ecr, continued

Entity	Description
place/city1_rr_uppercase/engkr	South Korean settlement with over 100,000 inhabitants, in uppercase revised romanization.
place/city2_rr/engkr	South Korean settlement with between 10,000 and 100,000 inhabitants, in revised romanization.
place/city2_rr_uppercase/engkr	South Korean settlement with between 10,000 and 100,000 inhabitants, in uppercase revised romanization.
place/province_rr/engkr	South Korean province in revised romanization.
place/province_rr_uppercase/engkr	South Korean province in uppercase revised romanization.
place/county_rr/engkr	South Korean county in revised romanization.
place/county_rr_uppercase/engkr	South Korean county in uppercase revised romanization.
place/island_rr/engkr	South Korean island in revised romanization.
place/island_rr_uppercase/engkr	South Korean island in uppercase revised romanization.
place/city1_mcr/engkr	South Korean settlement with over 100,000 inhabitants in McCune-Reischauer romanization.
place/city1_mcr_uppercase/engkr	South Korean settlement with over 100,000 inhabitants, in uppercase McCune-Reischauer romanization.
place/city2_mcr/engkr	South Korean settlement with between 10,000 and 100,000 inhabitants, in McCune-Reischauer romanization.
place/city2_mcr_uppercase/engkr	South Korean settlement with between 10,000 and 100,000 inhabitants, in uppercase McCune-Reischauer romanization.
place/province_mcr/engkr	South Korean province in McCune-Reischauer romanization.
place/province_mcr_uppercase/engkr	South Korean province in uppercase McCune-Reischauer romanization.
place/county_mcr/engkr	South Korean county in McCune-Reischauer romanization.

place_engkr.ecr, continued

Entity	Description
place/county_mcr_uppercase/engkr	South Korean county in uppercase McCune-Reischauer romanization.
place/island_mcr/engkr	South Korean island in McCune-Reischauer romanization.
place/island_mcr_uppercase/engkr	South Korean island in uppercase McCune-Reischauer romanization.

place_englk.ecr

Entity	Description
place/city1/englk	Sri Lankan settlement with over 100,000 inhabitants.
place/city1_uppercase/englk	Sri Lankan settlement with over 100,000 inhabitants, in uppercase.
place/city2/englk	Sri Lankan settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/englk	Sri Lankan settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/englk	Sri Lankan province.
place/province_uppercase/englk	Sri Lankan province in uppercase.
place/district/englk	Sri Lankan district.
place/district_uppercase/englk	Sri Lankan district in uppercase.

place_engmk.ecr

Entity	Description
place/city1/engmk	Macedonian settlement with over 100,000 inhabitants.
place/city1_uppercase/engmk	Macedonian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engmk	Macedonian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engmk	Macedonian settlement with between 10,000 and 100,000 inhabitants, in uppercase.

place_engmk.ecr, continued

Entity	Description
place/municipality/engmk	Macedonian municipality.
place/municipality_uppercase/engmk	Macedonian municipality in uppercase.

place_engmn.ecr

Entity	Description
place/city1/engmn	Mongolian settlement with over 100,000 inhabitants.
place/city1_uppercase/engmn	Mongolian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engmn	Mongolian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engmn	Mongolian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/engmn	Mongolian province.
place/province_uppercase/engmn	Mongolian province in uppercase.

place_engmy.ecr

Entity	Description
place/city1/engmy	Malaysian settlement with over 100,000 inhabitants.
place/city1_uppercase/engmy	Malaysian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engmy	Malaysian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engmy	Malaysian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/state/engmy	Malaysian state.
place/state_uppercase/engmy	Malaysian state in uppercase.
place/district/engmy	Malaysian district.
place/district_uppercase/engmy	Malaysian district in uppercase.

place_engnz.ecr

Entity	Description
place/city1/engnz	New Zealand settlement with over 100,000 inhabitants.
place/city1_uppercase/engnz	New Zealand settlement with over 100,000 inhabitants, in uppercase.
place/city2/engnz	New Zealand settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engnz	New Zealand settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/engnz	New Zealand region.
place/region_uppercase/engnz	New Zealand region in uppercase.
place/terr_auth/engnz	New Zealand territorial authority.
place/terr_auth_uppercase/engnz	New Zealand territorial authority in uppercase.
place/island/engnz	New Zealand island.
place/island_uppercase/engnz	New Zealand island in uppercase.

place_engph.ecr

Entity	Description
place/city1/engph	Philippine settlement with over 100,000 inhabitants.
place/city1_uppercase/engph	Philippine settlement with over 100,000 inhabitants, in uppercase.
place/city2/engph	Philippine settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engph	Philippine settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/engph	Philippine region.
place/region_uppercase/engph	Philippine region in uppercase.
place/province/engph	Philippine province.
place/province_uppercase/engph	Philippine province in uppercase.
place/island/engph	Philippine island.
place/island_uppercase/engph	Philippine island in uppercase.

place_engpk.ecr

Entity	Description
place/city1/engpk	Pakistani settlement with over 100,000 inhabitants.
place/city1_uppercase/engpk	Pakistani settlement with over 100,000 inhabitants, in uppercase.
place/city2/engpk	Pakistani settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engpk	Pakistani settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/engpk	Pakistani province.
place/province_uppercase/engpk	Pakistani province in uppercase.
place/district/engpk	Pakistani district.
place/district_uppercase/engpk	Pakistani district in uppercase.

place_engqa.ecr

Entity	Description
place/city1/engqa	Qatari settlement with over 100,000 inhabitants.
place/city1_uppercase/engqa	Qatari settlement with over 100,000 inhabitants, in uppercase.
place/city2/engqa	Qatari settlement with fewer than 100,000 inhabitants.
place/city2_uppercase/engqa	Qatari settlement with fewer than 100,000 inhabitants, in uppercase.
place/municipality/engqa	Qatari municipality.
place/municipality_uppercase/engqa	Qatari municipality in uppercase.

place_engru.ecr

Entity	Description
place/city1/engru	Russian settlement with over 100,000 inhabitants.
place/city1_uppercase/engru	Russian settlement with over 100,000 inhabitants, in uppercase.

place_engru.ecr, continued

Entity	Description
place/city2/engru	Russian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engru	Russian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/republic/engru	Russian republic (type of region).
place/republic_uppercase/engru	Russian republic (type of region), in uppercase.
place/oblast/engru	Russian oblast (type of region).
place/oblast_uppercase/engru	Russian oblast (type of region), in uppercase.
place/krai/engru	Russian krai (type of region).
place/krai_uppercase/engru	Russian krai (type of region), in uppercase.
place/okrug/engru	Russian okrug (type of region).
place/okrug_uppercase/engru	Russian okrug (type of region), in uppercase.
place/federal_city/engru	Russian federal city (type of region).
place/federal_city_uppercase/engru	Russian federal city (type of region), in uppercase.
place/region/engru	Russian region.
place/region_uppercase/engru	Russian region, in uppercase.
place/island/engru	Russian island.
place/island_uppercase/engru	Russian island, in uppercase.

place_engsa.ecr

Entity	Description
place/city1/engsa	Saudi Arabian settlement with over 100,000 inhabitants.
place/city1_uppercase/engsa	Saudi Arabian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engsa	Saudi Arabian settlement with fewer than 100,000 inhabitants.
place/city2_uppercase/engsa	Saudi Arabian settlement with fewer than 100,000 inhabitants, in uppercase.

place_engsa.ecr, continued

Entity	Description
place/province/engsa	Saudi Arabian province.
place/province_uppercase/engsa	Saudi Arabian province in uppercase.

place_ength.ecr

Entity	Description
place/city1/ength	Thai settlement with over 100,000 inhabitants.
place/city1_uppercase/ength	Thai settlement with over 100,000 inhabitants, in uppercase.
place/city2/ength	Thai settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/ength	Thai settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/ength	Thai province.
place/province_uppercase/ength	Thai province in uppercase.

place_engtw.ecr

Entity	Description
place/city1/engtw	Taiwanese settlement with over 100,000 inhabitants.
place/city1_uppercase/engtw	Taiwanese settlement with over 100,000 inhabitants, in uppercase.
place/city2/engtw	Taiwanese settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engtw	Taiwanese settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/county/engtw	Taiwanese county.
place/county_uppercase/engtw	Taiwanese county in uppercase.

place_engua.ecr

Entity	Description
place/city1/engua	Ukrainian settlement with over 100,000 inhabitants.
place/city1_uppercase/engua	Ukrainian settlement with over 100,000 inhabitants, in uppercase.
place/city2/engua	Ukrainian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engua	Ukrainian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/engua	Ukrainian region.
place/region_uppercase/engua	Ukrainian region in uppercase.

place_engus.ecr

Entity	Description
place/possession/engus	U.S. possessions in long form. For example, *American Samoa*.
place/possession_uppercase/engus	U.S. possessions in long form, in uppercase.
place/possession_abbrev/engus	U.S. possession abbreviations. For example, *GU*.
place/state/engus	U.S. states. For example, *New Hampshire*.
place/state_uppercase/engus	U.S. states, in uppercase.
place/state_abbrev/engus	U.S. states abbreviations. For example, *AL*.
place/poss_state/engus	U.S. possessions and states.
place/poss_state_abbrev/engus	U.S. possession and state abbreviations.
place/statecapital/engus	U.S. state capitals.
place/statecapital_uppercase/engus	U.S. state capitals, in uppercase.
place/city/AL/engus place/city_uppercase/AL/engus place/city/AK/engus place/city_uppercase/AK/engus place/city/AZ/engus place/city_uppercase/AZ/engus	Settlements in each U.S. state, in normal or uppercase.

place_engus.ecr, continued

Entity	Description
place/city/AR/engus	
place/city_uppercase/AR/engus	
place/city/CA/engus	
place/city_uppercase/CA/engus	
place/city/CO/engus	
place/city_uppercase/CO/engus	
place/city/CT/engus	
place/city_uppercase/CT/engus	
place/city/DE/engus	
place/city_uppercase/DE/engus	
place/city/FL/engus	
place/city_uppercase/FL/engus	
place/city/GA/engus	
place/city_uppercase/GA/engus	
place/city/HI/engus	
place/city_uppercase/HI/engus	
place/city/ID/engus	
place/city_uppercase/ID/engus	
place/city/IL/engus	
place/city_uppercase/IL/engus	
place/city/IN/engus	
place/city_uppercase/IN/engus	
place/city/IA/engus	
place/city_uppercase/IA/engus	
place/city/KS/engus	
place/city_uppercase/KS/engus	
place/city/KY/engus	
place/city_uppercase/KY/engus	
place/city/LA/engus	

place_engus.ecr, continued

Entity	Description
place/city_uppercase/LA/engus	
place/city/ME/engus	
place/city_uppercase/ME/engus	
place/city/MD/engus	
place/city_uppercase/MD/engus	
place/city/MA/engus	
place/city_uppercase/MA/engus	
place/city/MI/engus	
place/city_uppercase/MI/engus	
place/city/MN/engus	
place/city_uppercase/MN/engus	
place/city/MS/engus	
place/city_uppercase/MS/engus	
place/city/MO/engus	
place/city_uppercase/MO/engus	
place/city/MT/engus	
place/city_uppercase/MT/engus	
place/city/NE/engus	
place/city_uppercase/NE/engus	
place/city/NV/engus	
place/city_uppercase/NV/engus	
place/city/NH/engus	
place/city_uppercase/NH/engus	
place/city/NJ/engus	
place/city_uppercase/NJ/engus	
place/city/NM/engus	
place/city_uppercase/NM/engus	
place/city/NY/engus	
place/city_uppercase/NY/engus	

place_engus.ecr, continued

Entity	Description
place/city/NC/engus	
place/city_uppercase/NC/engus	
place/city/ND/engus	
place/city_uppercase/ND/engus	
place/city/OH/engus	
place/city_uppercase/OH/engus	
place/city/OK/engus	
place/city_uppercase/OK/engus	
place/city/OR/engus	
place/city_uppercase/OR/engus	
place/city/PA/engus	
place/city_uppercase/PA/engus	
place/city/RI/engus	
place/city_uppercase/RI/engus	
place/city/SC/engus	
place/city_uppercase/SC/engus	
place/city/SD/engus	
place/city_uppercase/SD/engus	
place/city/TN/engus	
place/city_uppercase/TN/engus	
place/city/TX/engus	
place/city_uppercase/TX/engus	
place/city/UT/engus	
place/city_uppercase/UT/engus	
place/city/VA/engus	
place/city_uppercase/VA/engus	
place/city/VT/engus	
place/city_uppercase/VT/engus	
place/city/WA/engus	

place_engus.ecr, continued

Entity	Description
place/city_uppercase/WA/engus place/city/WI/engus place/city_uppercase/WI/engus place/city/WV/engus place/city_uppercase/WV/engus place/city/WY/engus place/city_uppercase/WY/engus	
place/city1/engus	U.S. city with over 100,000 inhabitants.
place/city1_uppercase/engus	U.S. city with over 100,000 inhabitants, in uppercase.
place/city1_trimmed/engus	U.S. city with over 100,000 inhabitants, without a state identifier. For example *Edison*.
place/city1_uppercase_trimmed/engus	U.S. city with over 100,000 inhabitants, without a state identifier, in uppercase. For example *EDISON*.
place/city2/engus	U.S. city with over 10,000 inhabitants.
place/city2_uppercase/engus	U.S. city with over 10,000 inhabitants, in uppercase.
place/county/AL/engus	County in Alabama.
place/county/AK/engus	County in Alaska.
place/county/AZ/engus	County in Arizona.
place/county/AR/engus	County in Arkansas.
place/county/CA/engus	County in California.
place/county/CO/engus	County in Colorado.
place/county/CT/engus	County in Connecticut.
place/county/DE/engus	County in Delaware.
place/county/FL/engus	County in Florida.
place/county/GA/engus	County in Georgia.
place/county/HI/engus	County in Hawaii.
place/county/ID/engus	County in Idaho.
place/county/IL/engus	County in Illinois.
place/county/IN/engus	County in Indiana.

place_engus.ecr, continued

Entity	Description
place/county/IA/engus	County in Iowa.
place/county/KS/engus	County in Kansas.
place/county/KY/engus	County in Kentucky.
place/county/LA/engus	County in Louisiana.
place/county/ME/engus	County in Maine.
place/county/MD/engus	County in Maryland.
place/county/MA/engus	County in Massachusetts.
place/county/MI/engus	County in Michigan.
place/county/MN/engus	County in Minnesota.
place/county/MS/engus	County in Mississippi.
place/county/MO/engus	County in Missouri.
place/county/MT/engus	County in Montana.
place/county/NE/engus	County in Nebraska.
place/county/NV/engus	County in Nevada.
place/county/NH/engus	County in New Hampshire.
place/county/NJ/engus	County in New Jersey.
place/county/NM/engus	County in New Mexico.
place/county/NY/engus	County in New York.
place/county/NC/engus	County in North Carolina.
place/county/ND/engus	County in North Dakota.
place/county/OH/engus	County in Ohio.
place/county/OK/engus	County in Oklahoma.
place/county/OR/engus	County in Oregon.
place/county/PA/engus	County in Pennsylvania.
place/county/RI/engus	County in Rhode Island.
place/county/SC/engus	County in South Carolina.
place/county/SD/engus	County in South Dakota.
place/county/TN/engus	County in Tennessee.
place/county/TX/engus	County in Texas.

place_engus.ecr, continued

Entity	Description
place/county/UT/engus	County in Utah.
place/county/VT/engus	County in Vermont.
place/county/VA/engus	County in Virginia.
place/county/WA/engus	County in Washington.
place/county/WV/engus	County in West Virginia.
place/county/WI/engus	County in Wisconsin.
place/county/WY/engus	County in Wyoming.
place/county_uppercase/AL/engus	County in Alabama in uppercase.
place/county_uppercase/AK/engus	County in Alaska in uppercase.
place/county_uppercase/AZ/engus	County in Arizona in uppercase.
place/county_uppercase/AR/engus	County in Arkansas in uppercase.
place/county_uppercase/CA/engus	County in California in uppercase.
place/county_uppercase/CO/engus	County in Colorado in uppercase.
place/county_uppercase/CT/engus	County in Connecticut in uppercase.
place/county_uppercase/DE/engus	County in Delaware in uppercase.
place/county_uppercase/FL/engus	County in Florida in uppercase.
place/county_uppercase/GA/engus	County in Georgia in uppercase.
place/county_uppercase/HI/engus	County in Hawaii in uppercase.
place/county_uppercase/ID/engus	County in Idaho in uppercase.
place/county_uppercase/IL/engus	County in Illinois in uppercase.
place/county_uppercase/IN/engus	County in Indiana in uppercase.
place/county_uppercase/IA/engus	County in Iowa in uppercase.
place/county_uppercase/KS/engus	County in Kansas in uppercase.
place/county_uppercase/KY/engus	County in Kentucky in uppercase.
place/county_uppercase/LA/engus	County in Louisiana in uppercase.
place/county_uppercase/ME/engus	County in Maine in uppercase.
place/county_uppercase/MD/engus	County in Maryland in uppercase.
place/county_uppercase/MA/engus	County in Massachusetts in uppercase.
place/county_uppercase/MI/engus	County in Michigan in uppercase.

place_engus.ecr, continued

Entity	Description
place/county_uppercase/MN/engus	County in Minnesota in uppercase.
place/county_uppercase/MS/engus	County in Mississippi in uppercase.
place/county_uppercase/MO/engus	County in Missouri in uppercase.
place/county_uppercase/MT/engus	County in Montana in uppercase.
place/county_uppercase/NE/engus	County in Nebraska in uppercase.
place/county_uppercase/NV/engus	County in Nevada in uppercase.
place/county_uppercase/NH/engus	County in New Hampshire in uppercase.
place/county_uppercase/NJ/engus	County in New Jersey in uppercase.
place/county_uppercase/NM/engus	County in New Mexico in uppercase.
place/county_uppercase/NY/engus	County in New York in uppercase.
place/county_uppercase/NC/engus	County in North Carolina in uppercase.
place/county_uppercase/ND/engus	County in North Dakota in uppercase.
place/county_uppercase/OH/engus	County in Ohio in uppercase.
place/county_uppercase/OK/engus	County in Oklahoma in uppercase.
place/county_uppercase/OR/engus	County in Oregon in uppercase.
place/county_uppercase/PA/engus	County in Pennsylvania in uppercase.
place/county_uppercase/RI/engus	County in Rhode Island in uppercase.
place/county_uppercase/SC/engus	County in South Carolina in uppercase.
place/county_uppercase/SD/engus	County in South Dakota in uppercase.
place/county_uppercase/TN/engus	County in Tennessee in uppercase.
place/county_uppercase/TX/engus	County in Texas in uppercase.
place/county_uppercase/UT/engus	County in Utah in uppercase.
place/county_uppercase/VT/engus	County in Vermont in uppercase.
place/county_uppercase/VA/engus	County in Virginia in uppercase.
place/county_uppercase/WA/engus	County in Washington in uppercase.
place/county_uppercase/WV/engus	County in West Virginia in uppercase.
place/county_uppercase/WI/engus	County in Wisconsin in uppercase.
place/county_uppercase/WY/engus	County in Wyoming in uppercase.
place/county/engus	Any U.S. county.
place/county_uppercase/engus	Any U.S. county in uppercase.

place_engvn.ecr

Entity	Description
place/city1/engvn	Vietnamese settlement with over 100,000 inhabitants.
place/city1_uppercase/engvn	Vietnamese settlement with over 100,000 inhabitants, in uppercase.
place/city2/engvn	Vietnamese settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engvn	Vietnamese settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/engvn	Vietnamese province.
place/province_uppercase/engvn	Vietnamese province in uppercase.
place/district/engvn	Vietnamese district.
place/district_uppercase/engvn	Vietnamese district in uppercase.

place_engza.ecr

Entity	Description
place/city1/engza	South African settlement with over 100,000 inhabitants.
place/city1_uppercase/engza	South African settlement with over 100,000 inhabitants, in uppercase.
place/city2/engza	South African settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/engza	South African settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/engza	South African province.
place/province_uppercase/engza	South African province in uppercase.
place/district/engza	South African district.
place/district_uppercase/engza	South African district in uppercase.
place/island/engza	South African island.
place/island_uppercase/engza	South African island in uppercase.

place_estee.ecr

Entity	Description
place/city1/estee	Estonian settlement with over 100,000 inhabitants.
place/city1_uppercase/estee	Estonian settlement with over 100,000 inhabitants, in uppercase.
place/city2/estee	Estonian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/estee	Estonian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/county/estee	Estonian county.
place/county_uppercase/estee	Estonian county in uppercase.

place_finfi.ecr

Entity	Description
place/city1/finfi	Finnish settlement with over 100,000 inhabitants.
place/city1_uppercase/finfi	Finnish settlement with over 100,000 inhabitants, in uppercase.
place/city2/finfi	Finnish settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/finfi	Finnish settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/finfi	Finnish region.
place/region_uppercase/finfi	Finnish region in uppercase.
place/island/finfi	Finnish island.
place/island_uppercase/finfi	Finnish island in uppercase.

place_frefr.ecr

Entity	Description
place/city1/frefr	French settlement with over 100,000 inhabitants.
place/city1_uppercase/frefr	French settlement with over 100,000 inhabitants, in uppercase.

place_frefr.ecr, continued

Entity	Description
place/region_metro/frefr	French metropolitan regions.
place/region_metro_uppercase/frefr	French metropolitan regions in uppercase.
place/department_metro/Alsace/frefr	Departments of each French metropolitan region, in normal or uppercase.
place/department_metro_uppercase/Alsace/frefr	
place/department_metro/Aquitaine/frefr	
place/department_metro_uppercase/Aquitaine/frefr	
place/department_metro/Auvergne/frefr	
place/department_metro_uppercase/Auvergne/frefr	
place/department_metro/BasseNormandie/frefr	
place/department_metro_ uppercase/BasseNormandie/frefr	
place/department_metro/Bourgogne/frefr	
place/department_metro_uppercase/Bourgogne/frefr	
place/department_metro/Brittany/frefr	
place/department_metro_uppercase/Brittany/frefr	
place/department_metro/Centre/frefr	
place/department_metro_uppercase/Centre/frefr	
place/department_metro/ChampagneArdenne/frefr	
place/department_metro_ uppercase/ChampagneArdenne/frefr	
place/department_metro/Corsica/frefr	
place/department_metro_uppercase/Corsica/frefr	
place/department_metro/FrancheComte/frefr	
place/department_metro_ uppercase/FrancheComte/frefr	
place/department_metro/HauteNormandie/frefr	
place/department_metro_	

place_frefr.ecr, continued

Entity	Description
uppercase/HauteNormandie/frefr	
place/department_metro/IleDeFrance/frefr	
place/department_metro_ uppercase/IleDeFrance/frefr	
place/department_metro/LanguedocRoussillon/frefr	
place/department_metro_ uppercase/LanguedocRoussillon/frefr	
place/department_metro/Limousin/frefr	
place/department_metro_uppercase/Limousin/frefr	
place/department_metro/Lorraine/frefr	
place/department_metro_uppercase/Lorraine/frefr	
place/department_metro/MidiPyrenees/frefr	
place/department_metro_ uppercase/MidiPyrenees/frefr	
place/department_metro/NordPasDeCalais/frefr	
place/department_metro_ uppercase/NordPasDeCalais/frefr	
place/department_metro/PaysDeLaLoire/frefr	
place/department_metro_ uppercase/PaysDeLaLoire/frefr	
place/department_metro/Picardie/frefr	
place/department_metro_uppercase/Picardie/frefr	
place/department_metro/PoitouCharentes/frefr	
place/department_metro_ uppercase/PoitouCharentes/frefr	
place/department_ metro/ProvenceAlpesCoteDAzur/frefr	
place/department_metro_ uppercase/ProvenceAlpesCoteDAzur/frefr	

place_frefr.ecr, continued

Entity	Description
place/department_metro/RhoneAlpes/frefr	
place/department_metro_ uppercase/RhoneAlpes/frefr	
place/department_metro/frefr	French metropolitan departments.
place/department_metro_uppercase/frefr	French metropolitan departments in uppercase.
place/departmentcode_metro/frefr	French metropolitan department INSEE codes.
place/departmentcode_overseas/frefr	French overseas department INSEE codes.
place/communecode_metro/frefr	French metropolitan commune INSEE codes.
place/communecode_overseas/frefr	French overseas commune INSEE codes.
place/city/alsace/Bas_Rhin/frefr	Settlements in each French department, in normal or uppercase.
place/city_uppercase/alsace/Bas_Rhin/frefr	
place/city/alsace/Haut_Rhin/frefr	
place/city_uppercase/alsace/Haut_Rhin/frefr	

place_frefr.ecr, continued

Entity	Description
place/city/aquitaine/Dordogne/frefr	
place/city_uppercase/aquitaine/Dordogne/frefr	
place/city/aquitaine/Gironde/frefr	
place/city_uppercase/aquitaine/Gironde/frefr	
place/city/aquitaine/Landes/frefr	
place/city_uppercase/aquitaine/Landes/frefr	
place/city/aquitaine/Lot_et_Garonne/frefr	
place/city_uppercase/aquitaine/Lot_et_Garonne/frefr	
place/city/aquitaine/Pyrenees_Atlantiques/frefr	
place/city_uppercase/aquitaine/Pyrenees_ Atlantiques/frefr	
place/city/auvergne/Allier/frefr	
place/city_uppercase/auvergne/Allier/frefr	
place/city/auvergne/Cantal/frefr	
place/city_uppercase/auvergne/Cantal/frefr	
place/city/auvergne/Haute_Loire/frefr	
place/city_uppercase/auvergne/Haute_Loire/frefr	
place/city/auvergne/Puy_de_Dome/frefr	
place/city_uppercase/auvergne/Puy_de_Dome/frefr	

place_frefr.ecr, continued

Entity	Description
place/city/basseNormandie/Calvados/frefr	
place/city_uppercase/basseNormandie/Calvados/frefr	
place/city/basseNormandie/Manche/frefr	
place/city_uppercase/basseNormandie/Manche/frefr	
place/city/basseNormandie/Orne/frefr	
place/city_uppercase/basseNormandie/Orne/frefr	
place/city/bourgogne/Cote_dOr/frefr	
place/city_uppercase/bourgogne/Cote_dOr/frefr	
place/city/bourgogne/Nievre/frefr	
place/city_uppercase/bourgogne/Nievre/frefr	
place/city/bourgogne/Saone_et_Loire/frefr	
place/city_uppercase/bourgogne/Saone_et_ Loire/frefr	
place/city/bourgogne/Yonne/frefr	
place/city_uppercase/bourgogne/Yonne/frefr	
place/city/brittany/Cotes_dArmor/frefr	
place/city_uppercase/brittany/Cotes_dArmor/frefr	
place/city/brittany/Finistere/frefr	
place/city_uppercase/brittany/Finistere/frefr	

place_frefr.ecr, continued

Entity	Description
place/city/brittany/Ille_et_Vilaine/frefr	
place/city_uppercase/brittany/Ille_et_Vilaine/frefr	
place/city/brittany/Morbihan/frefr	
place/city_uppercase/brittany/Morbihan/frefr	
place/city/centre/Cher/frefr	
place/city_uppercase/centre/Cher/frefr	
place/city/centre/Eure_et_Loir/frefr	
place/city_uppercase/centre/Eure_et_Loir/frefr	
place/city/centre/Indre/frefr	
place/city_uppercase/centre/Indre/frefr	
place/city/centre/Indre_et_Loire/frefr	
place/city_uppercase/centre/Indre_et_Loire/frefr	
place/city/centre/Loir_et_Cher/frefr	
place/city_uppercase/centre/Loir_et_Cher/frefr	
place/city/centre/Loiret/frefr	
place/city_uppercase/centre/Loiret/frefr	
place/city/champagneArdenne/Ardennes/frefr	
place/city_ uppercase/champagneArdenne/Ardennes/frefr	

place_frefr.ecr, continued

Entity	Description
place/city/champagneArdenne/Aube/frefr	
place/city_uppercase/champagneArdenne/Aube/frefr	
place/city/champagneArdenne/Marne/frefr	
place/city_ uppercase/champagneArdenne/Marne/frefr	
place/city/champagneArdenne/Haute_Marne/frefr	
place/city_uppercase/champagneArdenne/Haute_ Marne/frefr	
place/city/corsica/Corse_du_Sud/frefr	
place/city_uppercase/corsica/Corse_du_Sud/frefr	
place/city/corsica/Haute_Corse/frefr	
place/city_uppercase/corsica/Haute_Corse/frefr	
place/city/francheComte/Doubs/frefr	
place/city_uppercase/francheComte/Doubs/frefr	
place/city/francheComte/Jura/frefr	
place/city_uppercase/francheComte/Jura/frefr	
place/city/francheComte/Haute_Saone/frefr	
place/city_uppercase/francheComte/Haute_ Saone/frefr	
place/city/francheComte/Territoire_de_Belfort/frefr	
place/city_uppercase/francheComte/Territoire_de_ Belfort/frefr	

place_frefr.ecr, continued

Entity	Description
place/city/hauteNormandie/Eure/frefr	
place/city_uppercase/hauteNormandie/Eure/frefr	
place/city/hauteNormandie/Seine_Maritime/frefr	
place/city_uppercase/hauteNormandie/Seine_Maritime/frefr	
place/city/ileDeFrance/Seine_et_Marne/frefr	
place/city_uppercase/ileDeFrance/Seine_et_Marne/frefr	
place/city/ileDeFrance/Yvelines/frefr	
place/city_uppercase/ileDeFrance/Yvelines/frefr	
place/city/ileDeFrance/Essonne/frefr	
place/city_uppercase/ileDeFrance/Essonne/frefr	
place/city/ileDeFrance/Hauts_de_Seine/frefr	
place/city_uppercase/ileDeFrance/Hauts_de_Seine/frefr	
place/city/ileDeFrance/Seine_Saint_Denis/frefr	
place/city_uppercase/ileDeFrance/Seine_Saint_Denis/frefr	
place/city/ileDeFrance/Val_de_Marne/frefr	
place/city_uppercase/ileDeFrance/Val_de_Marne/frefr	
place/city/ileDeFrance/Val_dOise/frefr	
place/city_uppercase/ileDeFrance/Val_dOise/frefr	

place_frefr.ecr, continued

Entity	Description
place/city/ileDeFrance/Paris/frefr	
place/city_uppercase/ileDeFrance/Paris/frefr	
place/city/languedocRoussillon/Aude/frefr	
place/city_uppercase/languedocRoussillon/Aude/frefr	
place/city/languedocRoussillon/Gard/frefr	
place/city_uppercase/languedocRoussillon/Gard/frefr	
place/city/languedocRoussillon/Herault/frefr	
place/city_ uppercase/languedocRoussillon/Herault/frefr	
place/city/languedocRoussillon/Lozere/frefr	
place/city_ uppercase/languedocRoussillon/Lozere/frefr	
place/city/languedocRoussillon/Pyrenees_ Orientales/frefr	
place/city_ uppercase/languedocRoussillon/Pyrenees_ Orientales/frefr	
place/city/limousin/Correze/frefr	
place/city_uppercase/limousin/Correze/frefr	
place/city/limousin/Creuse/frefr	
place/city_uppercase/limousin/Creuse/frefr	
place/city/limousin/Haute_Vienne/frefr	
place/city_uppercase/limousin/Haute_Vienne/frefr	

place_frefr.ecr, continued

Entity	Description
place/city/lorraine/Meurthe_et_Moselle/frefr	
place/city_uppercase/lorraine/Meurthe_et_ Moselle/frefr	
place/city/lorraine/Meuse/frefr	
place/city_uppercase/lorraine/Meuse/frefr	
place/city/lorraine/Moselle/frefr	
place/city_uppercase/lorraine/Moselle/frefr	
place/city/lorraine/Vosges/frefr	
place/city_uppercase/lorraine/Vosges/frefr	
place/city/midiPyrenees/Ariege/frefr	
place/city_uppercase/midiPyrenees/Ariege/frefr	
place/city/midiPyrenees/Aveyron/frefr	
place/city_uppercase/midiPyrenees/Aveyron/frefr	
place/city/midiPyrenees/Haute_Garonne/frefr	
place/city_uppercase/midiPyrenees/Haute_ Garonne/frefr	
place/city/midiPyrenees/Gers/frefr	
place/city_uppercase/midiPyrenees/Gers/frefr	
place/city/midiPyrenees/Lot/frefr	
place/city_uppercase/midiPyrenees/Lot/frefr	

place_frefr.ecr, continued

Entity	Description
place/city/midiPyrenees/Hautes_Pyrenees/frefr	
place/city_uppercase/midiPyrenees/Hautes_Pyrenees/frefr	
place/city/midiPyrenees/Tarn/frefr	
place/city_uppercase/midiPyrenees/Tarn/frefr	
place/city/midiPyrenees/Tarn_et_Garonne/frefr	
place/city_uppercase/midiPyrenees/Tarn_et_Garonne/frefr	
place/city/nordPasDeCalais/Nord/frefr	
place/city_uppercase/nordPasDeCalais/Nord/frefr	
place/city/nordPasDeCalais/Pas_de_Calais/frefr	
place/city_uppercase/nordPasDeCalais/Pas_de_Calais/frefr	
place/city/paysDeLaLoire/Loire_Atlantique/frefr	
place/city_uppercase/paysDeLaLoire/Loire_Atlantique/frefr	
place/city/paysDeLaLoire/Maine_et_Loire/frefr	
place/city_uppercase/paysDeLaLoire/Maine_et_Loire/frefr	
place/city/paysDeLaLoire/Mayenne/frefr	
place/city_uppercase/paysDeLaLoire/Mayenne/frefr	
place/city/paysDeLaLoire/Sarthe/frefr	
place/city_uppercase/paysDeLaLoire/Sarthe/frefr	

place_frefr.ecr, continued

Entity	Description
place/city/paysDeLaLoire/Vendee/frefr	
place/city_uppercase/paysDeLaLoire/Vendee/frefr	
place/city/picardie/Aisne/frefr	
place/city_uppercase/picardie/Aisne/frefr	
place/city/picardie/Oise/frefr	
place/city_uppercase/picardie/Oise/frefr	
place/city_uppercase/picardie/Somme/frefr	
place/city/poitouCharentes/Charente/frefr	
place/city_uppercase/poitouCharentes/Charente/frefr	
place/city/poitouCharentes/Charente_Maritime/frefr	
place/city_uppercase/poitouCharentes/Charente_ Maritime/frefr	
place/city/poitouCharentes/Deux_Sevres/frefr	
place/city_uppercase/poitouCharentes/Deux_ Sevres/frefr	
place/city/poitouCharentes/Vienne/frefr	
place/city_uppercase/poitouCharentes/Vienne/frefr	
place/city/provenceAlpesCoteDAzur/Alpes_de_ Haute_Provence/frefr	
place/city_ uppercase/provenceAlpesCoteDAzur/Alpes_de_ Haute_Provence/frefr	
place/city/provenceAlpesCoteDAzur/Hautes_ Alpes/frefr	
place/city_ uppercase/provenceAlpesCoteDAzur/Hautes_ Alpes/frefr	
place/city/provenceAlpesCoteDAzur/Alpes_ Maritimes/frefr	
place/city_	

place_frefr.ecr, continued

Entity	Description
uppercase/provenceAlpesCoteDAzur/Alpes_ Maritimes/frefr	
place/city/provenceAlpesCoteDAzur/Bouches_du_ Rhone/frefr	
place/city_ uppercase/provenceAlpesCoteDAzur/Bouches_du_ Rhone/frefr	
place/city/provenceAlpesCoteDAzur/Var/frefr	
place/city_ uppercase/provenceAlpesCoteDAzur/Var/frefr	
place/city/provenceAlpesCoteDAzur/Vaucluse/frefr	
place/city_ uppercase/provenceAlpesCoteDAzur/Vaucluse/frefr	
place/city/rhoneAlpes/Ain/frefr	
place/city_uppercase/rhoneAlpes/Ain/frefr	
place/city/rhoneAlpes/Ardeche/frefr	
place/city_uppercase/rhoneAlpes/Ardeche/frefr	
place/city/rhoneAlpes/Drome/frefr	
place/city_uppercase/rhoneAlpes/Drome/frefr	
place/city/rhoneAlpes/Isere/frefr	
place/city_uppercase/rhoneAlpes/Isere/frefr	
place/city/rhoneAlpes/Loire/frefr	
place/city_uppercase/rhoneAlpes/Loire/frefr	
place/city/rhoneAlpes/Rhone/frefr	
place/city_uppercase/rhoneAlpes/Rhone/frefr	
place/city/rhoneAlpes/Savoie/frefr	
place/city_uppercase/rhoneAlpes/Savoie/frefr	
place/city/rhoneAlpes/Haute_Savoie/frefr	
place/city_uppercase/rhoneAlpes/Haute_Savoie/frefr	

place_frefr.ecr, continued

Entity	Description
place/city/frefr	French cities.
place/city_uppercase/frefr	French cities in uppercase.

place_fregf.ecr

Entity	Description
place/city2/fregf	French Guianan settlement with over 10,000 inhabitants.
place/city2_uppercase/fregf	French Guianan settlement with over 10,000 inhabitants, in uppercase.
place/canton/fregf	French Guianan canton.
place/canton_uppercase/fregf	French Guianan canton in uppercase.

place_geo_dut.ecr

Entity	Description
place/country/dut	Country in Dutch.
place/country_capital/dut	Country capital in Dutch.

place_geo_eng.ecr

Entity	Description
place/region/eng	Regions. For example, *Asia-Pacific*.
place/region_uppercase/eng	Regions in uppercase.
place/continent/eng	Continents. For example, *Africa*.
place/continent_uppercase/eng	Continents in uppercase.
place/ocean/eng	Oceans. For example, *Pacific*.
place/ocean_uppercase/eng	Oceans in uppercase.
place/country/eng	Countries. For example, *Australia*.
place/country_uppercase/eng	Countries in uppercase.

place_geo_eng.ecr, continued

Entity	Description
place/country_capital/eng	Country capitals. For example, *Canberra*.
place/country_capital_uppercase/eng	Country capitals in uppercase.
place/direction/eng	Directions. For example, *Southwest*.
place/direction_uppercase/eng	Directions in uppercase.
place/direction_abb/eng	Direction abbreviations. For example, *SW.*
place/direction_mod/eng	Direction modifiers. For example, *Southwestern, Central, Downtown*.
place/direction_mod_uppercase/eng	Direction modifiers in uppercase.
place/area/eng	Areas. For example, *Cape, Canyon, Grassland, Peninsula.*
place/area_uppercase/eng	Areas in uppercase.
place/street_type/eng	Street types. For example, *Ave*, *Street*, *Place.*
place/street_type_uppercase/eng	Street types in uppercase.

place_geo_fre.ecr

Entity	Description
place/street_type/fre	Street types in French. For example, *Chauss*, *Cloitre*.
place/street_type_lowercase/fre	Street types in lowercase French. For example, *chauss*, *cloitre*.
place/street_type_uppercase/fre	Street types in uppercase French. For example, *CHAUSS*, *CLOITRE*.
place/house_type/fre	House types in French. For example, *Residence*, *Batiment*.
place/house_type_uppercase/fre	House types in uppercase French.
place/direction/fre	Directions in French. For example, *Sudouest*.
place/direction_uppercase/fre	Directions in uppercase French.
place/direction_abb/fre	Direction abbreviations in French. For example, *NO*.

place_gerat.ecr

Entity	Description
place/city1/gerat	Austrian settlement with over 100,000 inhabitants.
place/city1_uppercase/gerat	Austrian settlement with over 100,000 inhabitants, in uppercase.
place/city2/gerat	Austrian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/gerat	Austrian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/state/gerat	Austrian state.
place/state_uppercase/gerat	Austrian state in uppercase.

place_gerde.ecr

Entity	Description
place/state/gerde	German states.
place/state_uppercase/gerde	German states in uppercase.
place/state_abbrev/gerde	German state abbreviations.
place/city/state_capital/gerde	German state capitals.
place/city_uppercase/state_capital/gerde	German state capitals in uppercase.
place/city/bw/gerde place/city_uppercase/bw/gerde place/city/by/gerde place/city_uppercase/by/gerde place/city/be/gerde place/city_uppercase/be/gerde place/city/bb/gerde place/city_uppercase/bb/gerde place/city/hb/gerde place/city_uppercase/hb/gerde place/city/hh/gerde place/city_uppercase/hh/gerde	Settlements in each German state, in normal or uppercase.

place_gerde.ecr, continued

Entity	Description
place/city/he/gerde place/city_uppercase/he/gerde place/city/mv/gerde place/city_uppercase/mv/gerde place/city/ni/gerde place/city_uppercase/ni/gerde place/city/nw/gerde place/city_uppercase/nw/gerde place/city/rp/gerde place/city_uppercase/rp/gerde place/city/sl/gerde place/city_uppercase/sl/gerde place/city/sn/gerde place/city_uppercase/sn/gerde place/city/st/gerde place/city_uppercase/st/gerde place/city/sh/gerde place/city_uppercase/sh/gerde place/city/th/gerde place/city_uppercase/th/gerde	
place/city1/gerde	German settlement with more than 100,000 inhabitants.
place/city1_uppercase/gerde	German settlement with more than 100,000 inhabitants, in uppercase.
place/city/gerde	German cities.
place/city_uppercase/gerde	German cities in uppercase.

place_hrvhr.ecr

Entity	Description
place/city1/hrvhr	Croatian settlement with over 100,000 inhabitants.
place/city1_uppercase/hrvhr	Croatian settlement with over 100,000 inhabitants, in uppercase.
place/city2/hrvhr	Croatian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/hrvhr	Croatian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/county/hrvhr	Croatian county.
place/county_uppercase/hrvhr	Croatian county in uppercase.

place_hunhu.ecr

Entity	Description
place/city1/hunhu	Hungarian settlement with over 100,000 inhabitants.
place/city1_uppercase/hunhu	Hungarian settlement with over 100,000 inhabitants, in uppercase.
place/city2/hunhu	Hungarian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/hunhu	Hungarian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/county/hunhu	Hungarian county.
place/county_uppercase/hunhu	Hungarian county in uppercase.

place_itait.ecr

Entity	Description
place/city1/itait	Italian settlement with over 100,000 inhabitants.
place/city1_uppercase/itait	Italian settlement with over 100,000 inhabitants, in uppercase.
place/city2/itait	Italian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/itait	Italian settlement with between 10,000 and 100,000

place_itait.ecr, continued

Entity	Description
	inhabitants, in uppercase.
place/region_abbreviation/itait	2-letter abbreviation for an Italian region. For example, RM (includes SCV and RSM).
place/region/itait	Italian region.
place/region_uppercase/itait	Italian region in uppercase.
place/municipality/itait	Italian municipality.
place/municipality_uppercase/itait	Italian municipality in uppercase.
place/island/itait	Italian island.
place/island_uppercase/itait	Italian island in uppercase.
place/locality/itait	Italian place.
place/locality_uppercase/itait	Italian place in uppercase.

place_jpnjp.ecr

Entity	Description
place/prefecture/jpnjp	Japanese prefectures.
place/region/jpnjp	Japanese regions.

place_jpnjp.ecr, continued

Entity	Description
place/city/aichi/jpnjp	Settlements in each Japanese prefecture.
place/city/akita/jpnjp	
place/city/aomori/jpnjp	
place/city/chiba/jpnjp	
place/city/ehime/jpnjp	
place/city/fukui/jpnjp	
place/city/fukuoka/jpnjp	
place/city/fukushima/jpnjp	
place/city/gifu/jpnjp	
place/city/gunma/jpnjp	
place/city/hiroshima/jpnjp	
place/city/hokkaido/jpnjp	
place/city/hyogo/jpnjp	
place/city/ibaraki/jpnjp	

place_jpnjp.ecr, continued

Entity	Description
place/city/ishikawa/jpnjp	
place/city/iwate/jpnjp	
place/city/kagawa/jpnjp	
place/city/kagoshima/jpnjp	
place/city/kanagawa/jpnjp	
place/city/kochi/jpnjp	
place/city/kumamoto/jpnjp	
place/city/kyoto/jpnjp	
place/city/mie/jpnjp	
place/city/miyagi/jpnjp	
place/city/miyazaki/jpnjp	
place/city/nagano/jpnjp	
place/city/nagasaki/jpnjp	
place/city/nara/jpnjp	
place/city/niigata/jpnjp	
place/city/oita/jpnjp	
place/city/okayama/jpnjp	
place/city/okinawa/jpnjp	

place_jpnjp.ecr, continued

Entity	Description
place/city/osaka/jpnjp	
place/city/saga/jpnjp	
place/city/saitama/jpnjp	
place/city/shiga/jpnjp	
place/city/shimane/jpnjp	
place/city/shizuoka/jpnjp	
place/city/tochigi/jpnjp	
place/city/tokushima/jpnjp	
place/city/tokyo/jpnjp	
place/city/tottori/jpnjp	
place/city/toyama/jpnjp	
place/city/wakayama/jpnjp	
place/city/yamagata/jpnjp	
place/city/yamaguchi/jpnjp	
place/city/yamanashi/jpnjp	
place/city/jpnjp	Japanese settlements.
place/misc/jpnjp	Japanese places.

place_kokr.ecr

Entity	Description
place/province/korkr	Province of South Korea, in Korean language.
place/province_DPRK/korkr	Province of North Korea (DPRK) as claimed by South Korea (Republic of Korea), in Korean language.
place/district/korkr	District of South Korea, in Korean language.
place/city1/korkr	Settlement in South Korea with over 100,000 inhabitants, in Korean language.
place/city_DPRK/korkr	Settlement in North Korea (DPRK) as claimed by South Korea (Republic of Korea), in Korean

place_kokr.ecr, continued

Entity	Description
	language.
place/city2/korkr	Settlement in South Korea with between 10,000 and 100,000 inhabitants, in Korean language.

place_lat_long.ecr

Entity	Description
place/lat_long	Geographical co-ordinate in any format (minimum precision is 1/10 degree or one minute of a degree). Supports the components NS, EW, LAT_DEGREES, LAT_DECIMAL, LAT_MINUTES, LAT_SECONDS, LONG_DEGREES, LONG_DECIMAL, LONG_MINUTES, and LONG_SECONDS You can use the `lat_long.lua` post-processing script to process this entity.
place/utm	Geographical co-ordinate written using the Universal Transverse Mercator convention. Supports no components.

place_lavlv.ecr

Entity	Description
place/city1/lavlv	Latvian settlement with over 100,000 inhabitants.
place/city1_uppercase/lavlv	Latvian settlement with over 100,000 inhabitants, in uppercase.
place/city2/lavlv	Latvian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/lavlv	Latvian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/municipality/lavlv	Latvian municipality.
place/municipality_uppercase/lavlv	Latvian municipality in uppercase.

place_litlt.ecr

Entity	Description
place/city1/litlt	Lithuanian settlement with over 100,000 inhabitants.
place/city1_uppercase/litlt	Lithuanian settlement with over 100,000 inhabitants, in uppercase.
place/city2/litlt	Lithuanian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/litlt	Lithuanian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/county/litlt	Lithuanian county.
place/county_uppercase/litlt	Lithuanian county in uppercase.

place_mil_engus.ecr

Entity	Description
place/mil/engus	U.S. military places.
place/mil_uppercase/engus	U.S. military places in uppercase.

place_mulbe.ecr

Entity	Description
place/city1/mulbe	Belgian settlement with over 100,000 inhabitants.
place/city1_uppercase/mulbe	Belgian settlement with over 100,000 inhabitants, in uppercase.
place/city2/mulbe	Belgian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/mulbe	Belgian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/mulbe	Belgian province.
place/province_uppercase/mulbe	Belgian province in uppercase.
place/region/mulbe	Belgian region.
place/region_uppercase/mulbe	Belgian region in uppercase.

place_mulch.ecr

Entity	Description
place/city1/mulch	Swiss settlement with over 100,000 inhabitants.
place/city1_uppercase/mulch	Swiss settlement with over 100,000 inhabitants, in uppercase.
place/city2/mulch	Swiss settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/mulch	Swiss settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/canton/mulch	Swiss canton.
place/canton_uppercase/mulch	Swiss canton in uppercase.
place/canton_abbr/mulch	Two-letter abbreviation for a Swiss canton (always uppercase).

place_mullu.ecr

Entity	Description
place/city2/mullu	Luxembourgish city.
place/city2_uppercase/mullu	Luxembourgish city in uppercase.
place/district/mullu	Luxembourgish district.
place/district_uppercase/mullu	Luxembourgish district in uppercase.
place/canton/mullu	Luxembourgish canton.
place/canton_uppercase/mullu	Luxembourgish canton in uppercase.

place_norno.ecr

Entity	Description
place/city1/norno	Norwegian settlement with over 100,000 inhabitants.
place/city1_uppercase/norno	Norwegian settlement with over 100,000 inhabitants, in uppercase.
place/city2/norno	Norwegian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/norno	Norwegian settlement with between 10,000 and

place_norno.ecr, continued

Entity	Description
	100,000 inhabitants, in uppercase.
place/county/norno	Norwegian county.
place/county_uppercase/norno	Norwegian county in uppercase.
place/island/norno	Norwegian island.
place/island_uppercase/norno	Norwegian island in uppercase.

place_polpl.ecr

Entity	Description
place/city1/polpl	Polish settlement with over 100,000 inhabitants.
place/city1_uppercase/polpl	Polish settlement with over 100,000 inhabitants, in uppercase.
place/city2/polpl	Polish settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/polpl	Polish settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/polpl	Polish province.
place/province_uppercase/polpl	Polish province in uppercase.
place/county/polpl	Polish county.
place/county_uppercase/polpl	Polish county in uppercase.
place/province/polpl	Polish province (in English).
place/province_uppercase/polpl	Polish province in uppercase (in English).
place/county/polpl	Polish county (in English).
place/county_uppercase/polpl	Polish county in uppercase (in English).

place_porbr.ecr

Entity	Description
place/city1/porbr	Brazilian settlement with over 100,000 inhabitants.
place/city1_uppercase/porbr	Brazilian settlement with over 100,000 inhabitants, in

place_porbr.ecr, continued

Entity	Description
	uppercase.
place/city2/porbr	Brazilian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/porbr	Brazilian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/state/porbr	Brazilian state.
place/state_uppercase/porbr	Brazilian state in uppercase.
place/island/porbr	Brazilian island.
place/island_uppercase/porbr	Brazilian island in uppercase.

place_porpt.ecr

Entity	Description
place/city1/porpt	Portuguese settlement with over 100,000 inhabitants.
place/city1_uppercase/porpt	Portuguese settlement with over 100,000 inhabitants, in uppercase.
place/city2/porpt	Portuguese settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/porpt	Portuguese settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/district/porpt	Portuguese district.
place/district_uppercase/porpt	Portuguese district in uppercase.
place/island/porpt	Portuguese island.
place/island_uppercase/porpt	Portuguese island in uppercase.

place_rummd.ecr

Entity	Description
place/city1/rummd	Moldovan settlement with over 100,000 inhabitants.
place/city1_uppercase/rummd	Moldovan settlement with over 100,000 inhabitants, in uppercase.

place_rummd.ecr, continued

Entity	Description
place/city2/rummd	Moldovan settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/rummd	Moldovan settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/district/rummd	Moldovan district.
place/district_uppercase/rummd	Moldovan district in uppercase.

place_rumro.ecr

Entity	Description
place/city1/rumro	Romanian settlement with over 100,000 inhabitants.
place/city1_uppercase/rumro	Romanian settlement with over 100,000 inhabitants, in uppercase.
place/city2/rumro	Romanian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/rumro	Romanian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/county/rumro	Romanian county.
place/county_uppercase/rumro	Romanian county in uppercase.

place_slksk.ecr

Entity	Description
place/city1/slksk	Slovakian settlement with over 100,000 inhabitants.
place/city1_uppercase/slksk	Slovakian settlement with over 100,000 inhabitants, in uppercase.
place/city2/slksk	Slovakian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/slksk	Slovakian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/slksk	Slovakian region.
place/region_uppercase/slksk	Slovakian region in uppercase.

place_slvsi.ecr

Entity	Description
place/city1/slvsi	Slovenian settlement with over 100,000 inhabitants.
place/city1_uppercase/slvsi	Slovenian settlement with over 100,000 inhabitants, in uppercase.
place/city2/slvsi	Slovenian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/slvsi	Slovenian settlement with between 10,000 and 100,000 inhabitants, in uppercase.

place_spaar.ecr

Entity	Description
place/city1/spaar	Argentinian settlement with over 100,000 inhabitants.
place/city1_uppercase/spaar	Argentinian settlement with over 100,000 inhabitants, in uppercase.
place/city2/spaar	Argentinian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spaar	Argentinian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/spaar	Argentinian province.
place/province_uppercase/spaar	Argentinian province in uppercase.
place/island/spaar	Argentinian island.
place/island_uppercase/spaar	Argentinian island in uppercase.

place_spabo.ecr

Entity	Description
place/city1/spabo	Bolivian settlement with over 100,000 inhabitants.
place/city1_uppercase/spabo	Bolivian settlement with over 100,000 inhabitants, in uppercase.
place/city2/spabo	Bolivian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spabo	Bolivian settlement with between 10,000 and 100,000

place_spabo.ecr, continued

Entity	Description
	inhabitants, in uppercase.
place/department/spabo	Bolivian department.
place/department_uppercase/spabo	Bolivian department in uppercase.
place/province/spabo	Bolivian province.
place/province_uppercase/spabo	Bolivian province in uppercase.

place_spacl.ecr

Entity	Description
place/city1/spacl	Chilean settlement with over 100,000 inhabitants.
place/city1_uppercase/spacl	Chilean settlement with over 100,000 inhabitants, in uppercase.
place/city2/spacl	Chilean settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spacl	Chilean settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/spacl	Chilean region.
place/region_uppercase/spacl	Chilean region in uppercase.
place/commune/spacl	Chilean commune.
place/commune_uppercase/spacl	Chilean commune in uppercase.

place_spaco.ecr

Entity	Description
place/city1/spaco	Colombian settlement with over 100,000 inhabitants.
place/city1_uppercase/spaco	Colombian settlement with over 100,000 inhabitants, in uppercase.
place/city2/spaco	Colombian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spaco	Colombian settlement with between 10,000 and 100,000 inhabitants, in uppercase.

place_spaco.ecr, continued

Entity	Description
place/department/spaco	Colombian department.
place/department_uppercase/spaco	Colombian department in uppercase.

place_spaec.ecr

Entity	Description
place/city1/spaec	Ecuadorian settlement with over 100,000 inhabitants.
place/city1_uppercase/spaec	Ecuadorian settlement with over 100,000 inhabitants, in uppercase.
place/city2/spaec	Ecuadorian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spaec	Ecuadorian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/province/spaec	Ecuadorian province.
place/province_uppercase/spaec	Ecuadorian province in uppercase.
place/island/spaec	Ecuadorian island.
place/island_uppercase/spaec	Ecuadorian island in uppercase.

place_spaes.ecr

Entity	Description
place/city1/spaes	Spanish settlements with over 100,000 inhabitants.
place/city1_uppercase/spaes	Spanish settlements with over 100,000 inhabitants, in uppercase.
place/city2/spaes	Spanish settlements with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spaes	Spanish settlements with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/spaes	Region in Spain.
place/region_uppercase/spaes	Region in Spain in uppercase.
place/province/spaes	Province in Spain.

place_spaes.ecr, continued

Entity	Description
place/province_uppercase/spaes	Province in Spain in uppercase.
place/island/spaes	Balearic and Canary Islands.
place/island_uppercase/spaes	Balearic and Canary Islands in uppercase.

place_spamx.ecr

Entity	Description
place/city1/spamx	Mexican settlements with over 100,000 inhabitants.
place/city1_uppercase/spamx	Mexican settlements with over 100,000 inhabitants, in uppercase.
place/city2/spamx	Mexican settlements with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spamx	Mexican settlements with between 10,000 and 100,000 inhabitants, in uppercase.
place/state/spamx	States in Mexico.
place/state_uppercase/spamx	States in Mexico in uppercase.
place/islands/spamx	Mexican islands.
place/islands_uppercase/spamx	Mexican islands in uppercase.

place_spape.ecr

Entity	Description
place/city1/spape	Peruvian settlement with over 100,000 inhabitants.
place/city1_uppercase/spape	Peruvian settlement with over 100,000 inhabitants, in uppercase.
place/city2/spape	Peruvian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spape	Peruvian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/spape	Peruvian region.
place/region_uppercase/spape	Peruvian region in uppercase.

place_spapy.ecr

Entity	Description
place/city1/spapy	Paraguayan settlement with over 100,000 inhabitants.
place/city1_uppercase/spapy	Paraguayan settlement with over 100,000 inhabitants, in uppercase.
place/city2/spapy	Paraguayan settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spapy	Paraguayan settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/spapy	Paraguayan region.
place/region_uppercase/spapy	Paraguayan region in uppercase.
place/commune/spapy	Paraguayan commune.
place/commune_uppercase/spapy	Paraguayan commune in uppercase.

place_spauy.ecr

Entity	Description
place/city1/spauy	Uruguayan settlement with over 100,000 inhabitants.
place/city1_uppercase/spauy	Uruguayan settlement with over 100,000 inhabitants, in uppercase.
place/city2/spauy	Uruguayan settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spauy	Uruguayan settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/department/spauy	Uruguayan department.
place/department_uppercase/spauy	Uruguayan department in uppercase.

place_spave.ecr

Entity	Description
place/city1/spave	Venezuelan settlement with over 100,000 inhabitants.
place/city1_uppercase/spave	Venezuelan settlement with over 100,000

place_spave.ecr, continued

Entity	Description
	inhabitants, in uppercase.
place/city2/spave	Venezuelan settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/spave	Venezuelan settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/spave	Venezuelan region.
place/region_uppercase/spave	Venezuelan region in uppercase.
place/state/spave	Venezuelan state.
place/state_uppercase/spave	Venezuelan state in uppercase.
place/island/spave	Venezuelan island.
place/island_uppercase/spave	Venezuelan island in uppercase.

place_srpme.ecr

Entity	Description
place/city1/srpme	Montenegrin settlement with over 100,000 inhabitants.
place/city1_uppercase/srpme	Montenegrin settlement with over 100,000 inhabitants, in uppercase.
place/city2/srpme	Montenegrin settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/srpme	Montenegrin settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/municipality/srpme	Montenegrin municipality.
place/municipality_uppercase/srpme	Montenegrin municipality in uppercase.

place_srprs.ecr

Entity	Description
place/city1/srprs	Serbian settlement with over 100,000 inhabitants.
place/city1_uppercase/srprs	Serbian settlement with over 100,000 inhabitants, in

place_srprs.ecr, continued

Entity	Description
	uppercase.
place/city2/srprs	Serbian settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/srprs	Serbian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/district/srprs	Serbian district.
place/district_uppercase/srprs	Serbian district in uppercase.

place_swese.ecr

Entity	Description
place/city1/swese	Swedish settlement with over 100,000 inhabitants.
place/city1_uppercase/swese	Swedish settlement with over 100,000 inhabitants, in uppercase.
place/city2/swese	Swedish settlement with between 10,000 and 100,000 inhabitants.
place/city2_uppercase/swese	Swedish settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/county/swese	Swedish county.
place/county_uppercase/swese	Swedish county in uppercase.
place/island/swese	Swedish island.
place/island_uppercase/swese	Swedish island in uppercase.

place_turtr.ecr

Entity	Description
place/city1/turtr	Turkish settlement with over 100,000 inhabitants.
place/city1_uppercase/turtr	Turkish settlement with over 100,000 inhabitants, in uppercase.
place/city2/turtr	Turkish settlement with between 10,000 and 100,000 inhabitants.

place_turtr.ecr, continued

Entity	Description
place/city2_uppercase/turtr	Turkish settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place/region/turtr	Turkish region.
place/region_uppercase/turtr	Turkish region in uppercase.
place/province/turtr	Turkish province.
place/province_uppercase/turtr	Turkish province in uppercase.
place/district/turtr	Turkish district.
place/district_uppercase/turtr	Turkish district in uppercase.

profanity_chi.ecr

Entity	Description
profanity/biological/chi	Potentially offensive term in Chinese pertaining to biological processes (including obscured representations).
profanity/sexual/chi	Potentially offensive term in Chinese pertaining to sex (including obscured representations).
profanity/personal/chi	Directly insulting term in Chinese (including obscured representations).
profanity/exclaim/chi	Potentially offensive term in Chinese pertaining to exclamation (including obscured representations).
profanity/chi	Any potentially offensive Chinese term (including obscured representations). Eduction gives higher scores to matches with a greater tendency to offend. The following `MinScore` parameter values are provided as a guide: • `MinScore=0.7` removes many weakly offensive terms and phrases • `MinScore=1.1` returns moderately-offensive terms and phrases • `MinScore=1.3` returns only strongly offensive terms and phrases

profanity_chi.ecr, continued

Entity	Description
	• `MinScore=2.5` returns no matches at all
profanity/phrase/chi	Any potentially offensive Chinese phrase (including obscured representations). The following `MinScore` parameter values are provided as a guide: • `MinScore=0.7` removes many weakly offensive terms and phrases • `MinScore=1.1` returns moderately-offensive terms and phrases • `MinScore=1.3` returns only strongly offensive terms and phrases • `MinScore=3.0` returns no matches at all

profanity_eng.ecr

Entity	Description
profanity/blasphemous/eng	Religious term often used for blasphemy (including obscured representations).
profanity/homophobic/eng	Homophobic term (including obscured representations).
profanity/racial/eng	Racial derogatory term (including obscured representations).
profanity/personal/eng	Personally insulting term. Contains all racial and homophobic offensive terms (including obscured representations).
profanity/sexual/eng	Potentially offensive term pertaining to sex (including obscured representations).
profanity/biological/eng	Potentially offensive term pertaining to biological processes (including obscured representations).
profanity/censored/eng	Word that appears in the text in a fully-censored format.
profanity/eng	Any potentially-offensive English term (including obscured representations) Eduction gives higher scores to matches with a

profanity_eng.ecr, continued

Entity	Description
	greater tendency to offend. The following `MinScore` parameter values are provided as a guide: • `MinScore=0.1` removes false matches, for example from URL shorteners • `MinScore=0.7` removes many weakly offensive terms and phrases • `MinScore=1.1` returns moderately-offensive terms and phrases • `MinScore=1.3` returns only strongly offensive terms and phrases • `MinScore=2.5` returns no matches at all

psi_api_credentials.ecr

Entity[1]	Description
psi/api_credentials/client_id/aws	An Amazon Web Services client ID.
psi/api_credentials/client_id/bitly	A Bitly client ID.
psi/api_credentials/client_id/facebook	A Facebook ID.
psi/api_credentials/client_id/flickr	A Flickr ID.
psi/api_credentials/client_id/foursquare	A Foursquare client ID.
psi/api_credentials/client_id/linkedin	A LinkedIn client ID.
psi/api_credentials/client_id/twitter	A Twitter client ID.
psi/api_credentials/secret_key/aws	An Amazon Web Services secret key.
psi/api_credentials/secret_key/bitly	A Bitly secret key.
psi/api_credentials/secret_key/facebook	A Facebook secret key.
psi/api_credentials/secret_key/flickr	A Flickr secret key.
psi/api_credentials/secret_key/foursquare	A Foursquare secret key.
psi/api_credentials/secret_key/linkedin	A LinkedIn secret key.
psi/api_credentials/secret_key/twitter	A Twitter secret key.

psi_private_key.ecr

Entity	Description
psi/private_key/pem/begin	Textual encoding pre-encapsulation boundary for Public Key Cryptography Standards (RFC7468). To ensure that this entity performs correctly, set `TangibleCharacters` to include '-'.
psi/private_key/pem/end	Textual encoding post-encapsulation boundary for Public Key Cryptography Standards (RFC7468). To ensure that this entity performs correctly, set `TangibleCharacters` to include '-'.
psi/private_key/pem	PEM format private key. To ensure that this entity performs correctly, set `TangibleCharacters` to include '-'. For this entity, you might need to increase the value of `MaxEntityLength` to at least 1024.
psi/private_key/xkms/rsakeypair/begin	XML Key Management Specification 2.0 RSAKeyPair start element. To ensure that this entity performs correctly, set `TangibleCharacters` to include '<'.
psi/private_key/xkms/rsakeypair/end	XML Key Management Specification 2.0 RSAKeyPair end element. To ensure that this entity performs correctly, set `TangibleCharacters` to include '<'.
psi/private_key/xkms/rsakeypair	XML Key Management Specification 2.0 RSAKeyPair element. To ensure that this entity performs correctly, set `TangibleCharacters` to include '<'. For this entity, you might need to increase the value of `MaxEntityLength` to at least 1536.
psi/private_key/xkms/rsakeyvalue/begin	XML Signature RSAKeyValue start element. To ensure that this entity performs correctly, set `TangibleCharacters` to include '<'.
psi/private_key/xkms/rsakeyvalue/end	XML Signature RSAKeyValue end element. To ensure that this entity performs correctly, set `TangibleCharacters` to include '<'.
psi/private_key/xkms/rsakeyvalue	XML Signature RSAKeyValue element. To ensure that this entity performs correctly, set `TangibleCharacters` to include '<'. For this entity, you might need to increase the value of `MaxEntityLength` to at least 1536.
psi/private_key/xkms/dsakeyvalue/begin	XML Signature DSAKeyValue start element. To ensure that this entity performs correctly, set

psi_private_key.ecr, continued

Entity	Description
	`TangibleCharacters` to include '<'.
psi/private_key/xkms/dsakeyvalue/end	XML Signature DSAKeyValue end element. To ensure that this entity performs correctly, set `TangibleCharacters` to include '<'.
psi/private_key/xkms/dsakeyvalue	XML Signature DSAKeyValue element. To ensure that this entity performs correctly, set `TangibleCharacters` to include '<'. For this entity, you might need to increase the value of `MaxEntityLength` to at least 1536.

[1]You must use the `psi_api_credentials_postprocessing.lua` post-processing script for this grammar. The post-processing script adjusts the scores for likely and unlikely matches, and normalizes the scores for these entities in the range 0-1.

S

The sentiment grammar files have *lite* versions. The lite versions are identical to the full versions in most respects, but they do not support components or user modification. They can process data up to twice as fast as the full versions, depending on language.

Micro Focus recommends that you use the lite versions except when you need to use components or modify the built-in dictionaries.

The lite grammars have the same name as the full version, with `_lite` after the language. For example, the file name of the Chinese sentiment grammar file is `sentiment_chi.ecr`, and the file name of the lite version is `sentiment_chi_lite.ecr`.

sentiment_ara.ecr and sentiment_ara_lite.ecr

Entity	Description
sentiment/positive/ara	An Arabic phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/ara	An Arabic phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/ara	A positive or negative phrase in Arabic. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase.

sentiment_ara.ecr and sentiment_ara_lite.ecr, continued

Entity	Description
	Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.

sentiment_chi.ecr and sentiment_chi_lite.ecr

Entity	Description
sentiment/positive/chi	A Chinese phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/chi	A Chinese phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/chi	A positive or negative phrase in Chinese. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.

sentiment_cze.ecr and sentiment_cze_lite.ecr

Entity	Description
sentiment/positive/cze	A Czech phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/cze	A Czech phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/cze	A positive or negative phrase in Czech. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.

sentiment_dut.ecr and sentiment_dut_lite.ecr

Entity	Description
sentiment/positive/dut	A Dutch phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/dut	A Dutch phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to **`:;@#`**.

sentiment_eng.ecr and sentiment_eng_lite.ecr

Entity	Description
sentiment/positive/eng	An English phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/eng	An English phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/eng	A positive or negative phrase in English. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to **`:;@#`**.

sentiment_basic_eng.ecr

Entity	Description
sentiment/positive/eng sentiment/negative/eng sentiment/eng	If recall with `sentiment_eng.ecr` is too low, and your documents are generally short comments, use `sentiment_basic_eng.ecr` to extract additional matches. This grammar contains carefully-selected lists of positive and negative terms that help determine the sentiment of a document in which `sentiment_eng.ecr` found no matches. TOPIC and SENTIMENT components are **not**

sentiment_basic_eng.ecr, continued

Entity	Description
	supported. `sentiment_basic_eng.ecr` contains terms in title case, but research shows that for most data these impair recall, so these are given a lower score. Micro Focus recommends that you set `EntityMinScore`*`N`* to **`0.4`** to filter out these terms unless you need them.

sentiment_fre.ecr and sentiment_fre_lite.ecr

Entity	Description
sentiment/positive/fre	A French phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/fre	A French phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/fre	A positive or negative phrase in French. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to **`:;@#`**.

sentiment_ger.ecr and sentiment_get_lite.ecr

Entity	Description
sentiment/positive/ger	A German phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/ger	A German phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/ger	A positive or negative phrase in German. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase.

sentiment_ger.ecr and sentiment_get_lite.ecr, continued

Entity	Description
	Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.

sentiment_ita.ecr and sentiment_ita_lite.ecr

Entity	Description
sentiment/positive/ita	An Italian phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/ita	An Italian phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/ita	A positive or negative phrase in Italian. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.

sentiment_pol.ecr and sentiment_pol_lite.ecr

Entity	Description
sentiment/positive/pol	A Polish phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/pol	A Polish phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/pol	A positive or negative phrase in Polish. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.

sentiment_por.ecr and sentiment_por_lite.ecr

Entity	Description
sentiment/positive/por	A Portuguese phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/por	A Portuguese phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/por	A positive or negative phrase in Portuguese. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.

sentiment_rus.ecr and sentiment_rus_lite.ecr

Entity	Description
sentiment/positive/rus	A Russian phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/rus	A Russian phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/rus	A positive or negative phrase in Russian. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.

sentiment_spa.ecr and sentiment_spa_lite.ecr

Entity	Description
sentiment/positive/spa	A Spanish phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/spa	A Spanish phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/spa	A positive or negative phrase in Spanish. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to **`:;@#`**.

sentiment_tur.ecr and sentiment_tur_lite.ecr

Entity	Description
sentiment/positive/tur	A Turkish phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
sentiment/negative/tur	A Turkish phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
sentiment/tur	A positive or negative phrase in Turkish. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. Micro Focus recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to **`:;@#`**.

T

team_american_football.ecr

Entity	Description
org/football/us	American Football team in the U.S.
org/football/ca	Canadian Football team in Canada. All synonyms for team names produce the same normalized text (for example, *The Bears* normalizes to *Chicago Bears*) to identify variant team names.

team_baseball.ecr

Entity	Description
org/baseball/mlb	Major League baseball team in the U.S. and Canada. All synonyms for team names produce the same normalized text (for example, *LA Dodgers* normalizes to *Los Angeles Dodgers*) to identify variant team names.

team_basketball.ecr

Entity	Description
org/basketball/nba	Basketball team in the NBA. All synonyms for team names produce the same normalized text (for example, *Sixers* normalizes to *Philadelphia 76ers*) to identify variant team names.

team_hockey.ecr

Entity	Description
org/hockey/nhl	Hockey team in the NHL. All synonyms for team names produce the same normalized text (for example, *NJ Devils* normalizes to *New Jersey Devils*) to identify variant team names.

team_soccer.ecr

Entity	Description
org/soccer/us	Soccer team in U.S. and Canada (Major League Soccer).
org/soccer/gb	Football (soccer) team in the United Kingdom. Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Celtic*.
org/soccer/de	Football (soccer) team in Germany (current Bundesliga teams). Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Wolfsburg*.
org/soccer/fr	Football (soccer) team in France. Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Nice*.
org/soccer/nl	Football (soccer) team in the Netherlands. Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Ajax*.
org/soccer/es	Football (soccer) team in Spain (current Primera & Segunda Divisiónes teams). Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Barcelona*.
org/soccer/it	Football (soccer) team in Italy (current teams in Serie A and Serie B). Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Inter*. All synonyms for team names produce the same normalized text (for example, *Man United* normalizes to *Manchester United*) to identify variant team names.

time_chi.ecr

Entity	Description
time/time_of_day/chi	A descriptive time of day in Chinese.
time/time_of_day_simplified/chi	A descriptive time of day in simplified Chinese.
time/period/chi	An amount of time in Chinese.
time/period_simplified/chi	An amount of time in simplified Chinese.
time/alpha_time/chi	Time of the day in Chinese words.
time/alpha_time_simplified/chi	Time of the day in simplified Chinese words and ASCII numbers.

time_chi.ecr, continued

Entity	Description
time/hms/chi	Time in hours and minutes with optional seconds and fractions.
time/hms_simplified/chi	Time in hours and minutes with optional seconds and fractions, in simplified Chinese and ASCII numbers.
time/chi	Any time of day in Chinese, in a variety of formats.
time/simplified/chi	Any time of day in simplified Chinese and ASCII numbers, in a variety of formats.

time_eng.ecr

Entity	Description
time/time_of_day/eng	A descriptive time of day in English. For example, *dawn*, *morning*, *Mid-afternoon.*
time/period/eng	An amount of time. For example, *day*, *quarter*, *month*, *decades.*
time/alpha_time/eng	Time of day in English words, for example, *4 o'clock*, *ten past five*.
time/hms/eng	Time in hours and minutes with optional seconds and fractions.
time/eng	Any time in English or numeric format. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 04:54 a.m. • 02:20 at night • quarter past midnight • 20 to midnight • ten past six • One o'clock • 6.10pm

time_eng.ecr, continued

Entity	Description
	• 1.49 in the afternoon • noon • 5:00 UTC+1 • 19:15 Hawaii-Aleutian Time

time_fre.ecr

Entity	Description
time/time_of_day/fre	A descriptive time of day in French. For example, *l'aube*, *Matin*.
time/period/fre	An amount of time. For example, *une décennie*, *un siècle*.
time/alpha_time/fre	Time of day in French words. For example, *sept heures du matin*, *trois heures de l'après-midi*.
time/hms/fre	Time in hours and minutes with optional seconds and fractions.
time/fre	Any time in French or numeric format. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 04:54 du matin • 4.54 de la nuit • minuit et 15 • midi moins vingt • 6 heures 20 du soir • 1 heure 49 de l'apres midi • une heure trente cinq • six heures et dix • midi

time_fre.ecr, continued

Entity	Description
	• 5:00 UTC+1 • 19:15 PDT

time_ger.ecr

Entity	Description
time/time_of_day/ger	A descriptive time of day. For example, *Nachmittag*.
time/period/ger	An amount of time in German (all declensions). For example, *Jahrzehnt*.
time/alpha_time/ger	Time of day in German words. For example, *fünf nach zehn*.
time/hms/ger	Time in hours and minutes with optional seconds and fractions.
time/ger	Any time in German or numeric format. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 04:54 morgens • 4.54 nachts • viertel nach mitternacht • 6.20 nachmittags • 1 Uhr 49 nachmittags • Fünf Uhr • Sechs Uhr Zehn • mittag • 5:00 UTC+1 • 19:15 PDT

time_ita.ecr

Entity	Description
time/time_of_day/ita	A descriptive time of day in Italian. For example, *pomeriggio*.
time/period/ita	An amount of time in Italian. For example, *giorno*, *Mesi*, *secolo*.
time/alpha_time/ita	Time of day in Italian words. For example, *Sono le 4*, *5 y 10 delpomeriggio*, *mezzanotte meno cinque.*
time/hms/ita	Time in hours and minutes with optional seconds and fractions.
time/ita	Any time in Italian or numeric format. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 4.54 del mattino • Tre e tre quarti di notte • un quarto alle sette • dieci all'una • sono le due meno cinque • 6 e 20 • 1 e 49 del pomeriggio • 13:35 • sei e dieci • Mezzo giorno • 5:00 UTC+1 • 19:15 PDT

time_numeric.ecr

Entity	Description
time/hms12	12-hour time in hours and minutes, with optional seconds and fractions.

time_numeric.ecr, continued

Entity	Description
time/hms24	24-hour time in hours and minutes, with optional seconds and fractions.
time/tz_abbr	Standard timezone abbreviations.
time/tz_abbr_plus	Standard timezone abbreviations with optional +/- hh:mm modifier.

time_por.ecr

Entity	Description
time/time_of_day/por	A descriptive time of day in Portuguese. For example, *manhã*, *pôr do dol.*
time/period/por	An amount of time in Portuguese. For example, *dia*, *Mês*, *séculos.*
time/alpha_time/por	Time of day in Portuguese words. For example, *São dez*, *doze e um quarto da noite*, *meia-noite menos 15*.
time/hms/por	Time in hours and minutes with optional seconds and fractions.
time/por	Any time in Portugese. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 4.54 da manhã • doze e quarto da noite (Brazilian Portuguese) • São vinte e cinco para as cinco da manhã • cinco e vinte da manhã • 1 e 49 da tarde • 13:35 • seis e dez • Meio-dia • 5:00 UTC+1

time_por.ecr, continued

Entity	Description
	• 19:15 PDT • 7 em ponto

time_spa.ecr

Entity	Description
time/time_of_day/spa	A descriptive time of day in Spanish. For example, *a la medianoche*, *al amanecer*.
time/period/spa	An amount of time in Spanish. For example, *década*.
time/alpha_time/spa	Time of day in Spanish words. For example, *a media mañana*.
time/hms/spa	Time in hours and minutes with optional seconds and fractions.
time/spa	Any time in Spanish or numeric format. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 04:54 de la mañana • 4.54 por la noche • doce y cuarto de la noche • Son las cinco menos veinticinco de la mañana • cinco y veinte de la mañana • 1 y 49 de la tarde • 13:35 • seis y diez • mediodía • 5:00 UTC+1 • 19:15 PDT • 7 en punto

transport_airport.ecr

Entity	Description
airport/icao	Airport ICAO code.
airport/iata	Airport IATA code.

transport_car.ecr

Entity	Description
car/make_model	Make and model of car.

U

university.ecr

Entity	Description
org/university	A university.

Standard Grammar – Source

Eduction includes standard grammar files in source form (XML) and their compiled equivalents (ECR). The source files import compiled Eduction standard grammar files and illustrate sample usage. You can modify these XML source files and recompile them to customize a grammar for the needs of your Eduction application.

The following table lists public entities defined in the XML source files. It excludes the public entities that are republished from the imported Eduction ECR grammar files.

File	Entity	Description
measure.xml	measure/all/eng	An editable collection of patterns that match length, area, volume, and mass.
money.xml[1]	money/all	All currency amounts. **NOTE:** This grammar file supports some English alphabetic numbers, for example, *seven cents*, *$12 million*, *one hundred dollars*, *£5m*.
pci_dss.xml	pci_dss/person_name/engus pci_dss/date/engus pci_dss/credit_card/engus pci_dss/bank_names/engus	Person names. Dates. Credit and debit card numbers. Bank names.
pii.xml	pii/person_name/engus pii/phone_number/engus	Personal names. Phone numbers.

[1]When matching symbols in the money entities, the Eduction option `MatchWholeWord` must be set to **`0`** (false). Otherwise, when encountering a string such as $10.70, Eduction will not recognize that *$* is the start of a token. Instead, it looks only for matches starting on the *1* and on the *7,* and will not return *$10.70*.

File	Entity	Description
	pii/email_address/engus	Email addresses.
	pii/ip_address/engus	IP addresses.
	pii/social_security/engus	Social Security numbers.
	pii/car_numberplate/engus	Car license plate numbers.
	pii/driver_license/engus	Driver's license numbers.
	pii/credit_card/engus	Credit and debit card numbers.
	pii/date/engus	Dates.
	pii/country	Countries.
	pii/state/engus	U.S. states or possessions.
	pii/county/engus	U.S. counties.
	pii/city/engus	U.S. cities.
	pii/address/engus	Geographical addresses.
	pii/zipcode/engus	U.S. zipcodes.
	pii/age/engus	Age.
	pii/gender/engus	Gender.
	pii/race/engus	Race.
	pii/job_title/engus	Job title.
	pii/disease_and_condition/engus	Disease or medical condition.
	pii/account_number/engus	Generic account number with 6-8 digits in a predictable context.
	pii/license_number/engus	Generic license number with specific alphanumeric format.
	pii/facebook_url/engus	Example URL for a personal Web page (Facebook).

File	Entity	Description
place_europe.xml	place/country/europe	European country in English (and some local languages).
	place/country_uppercase/europe	European country in English and local languages (uppercase).
	place/city1/europe	European settlement with over 100,000 inhabitants, in local language.
	place/city1_uppercase/europe	European settlement with over 100,000 inhabitants, in local language (uppercase).
	place/city2/europe	European settlement with between 10,000 and 100,000 inhabitants, in local language.
	place/city2_uppercase/europe	European settlement with between 10,000 and 100,000 inhabitants, in local language (uppercase).
	place/region/Europe	High-level administrative division, in local language.
	place/region_uppercase/Europe	High-level administrative division, in local language (uppercase).

File	Entity	Description
place_south_america.xml	place/country/south_america	South American country in English, Spanish, or Portuguese.
	place/country_uppercase/south_america	South American country in English, Spanish, or Portuguese (uppercase).
	place/city1/south_america	South American settlement with over 100,000 inhabitants, in local language.
	place/city1_uppercase/south_america	South American settlement with over 100,000 inhabitants, in local language (uppercase).
	place/city2/south_america	South American settlement with between 10,000 and 100,000 inhabitants, in local language.
	place/city2_uppercase/south_america	South American settlement with between 10,000 and 100,000 inhabitants, in local language (uppercase).
	place/island/south_america	South American island, in local language.
	place/island_uppercase/south_america	South American island, in local language (uppercase).
	place/region/south_america	High-level administrative division, in local language.
	place/region_uppercase/south_america	High-level administrative division, in local language (uppercase).
retention.xml	retention/admission_date retention/discharge_date retention/birth_date retention/age/eng	Admission date. Discharge date. Birth date. Age.
sample.xml	sample/solar_system	A simple entity for planets of the solar system.
sentiment_user_chi.xml	sentiment/user_client_name sentiment/user_client_brand sentiment/user_client_rv1_name	You can use these files to modify the sentiment analysis grammar files for the relevant languages to give access to extra domain-specific vocabulary.

File	Entity	Description
	sentiment/user_client_rv1_brand sentiment/user_third_party_company_name sentiment/user_third_party_company_brand sentiment/user_positive_adjective sentiment/user_negative_adjective sentiment/user_positive_noun sentiment/user_negative_noun sentiment/user_neutral_noun sentiment/user_positive_verb sentiment/user_negative_verb sentiment/user_neutral_verb sentiment/user_positive_idiom sentiment/user_negative_idiom	
sentiment_user_ara.xml sentiment_user_cze.xml sentiment_user_dutch.xml sentiment_user_eng.xml	sentiment/user_positive_adjective sentiment/user_negative_adjective sentiment/user_neutral_adjective sentiment/user_positive_adverb sentiment/user_negative_adverb sentiment/user_neutral_adverb sentiment/user_positive_noun	

File	Entity	Description
sentiment_user_fre.xml sentiment_user_ger.xml sentiment_user_ita.xml sentiment_user_pol.xml sentiment_user_por.xml sentiment_user_rus.xml sentiment_user_spa.xml sentiment_user_tur.xml	sentiment/user_negative_noun sentiment/user_neutral_noun sentiment/user_positive_verb sentiment/user_negative_verb sentiment/user_neutral_verb sentiment/user_positive_match sentiment/user_negative_match sentiment/user_good_noun (English only)	

The entities in this table incorporate the compiled Eduction entities in combination with Eduction XML grammar to create additional entities. The XML illustrates how to use the compiled Eduction entities. You can modify these XML files and compile them into Eduction ECR files that you can then use for specific applications.

The Eduction grammar files have three advantages:

- Allows for fined-grained access to basic entities that include more complex entities. You can then customize the complex entities to increase the precision and recall of the matching process.
- Provides both the compiled ECR grammar files as well as source-form XML grammar files that reference them.
- Separate ECR files reduce the memory footprint and file size.

Chapter 14: Grammar Format Reference

This section provides a reference for the syntax and regular expressions that you can use in grammar files.

For details of how to modify the grammars, see Create and Modify Eduction Grammars, on page 85. For details of the grammar files provided with Eduction, see Standard Grammars, on page 113.

Eduction Grammar Syntax

The tables in this section describe the Eduction grammar syntax defined in the `edk.dtd` (see Eduction Grammar DTD, on page 310).

In these tables, the terms in angled brackets <> describe the value that you must insert. The XML elements, attributes, and values are defined in lower case.

Although the Eduction compiler accepts uppercase element and attribute names, this functionality is deprecated, retained for backward compatibility.

The `edk.dtd` file represents the current definition for Eduction grammar files. All Eduction grammars must follow this DTD.

<grammars>

Element: grammars

Child Elements: <include>, <grammar>

Description: This is the top-level element in an Eduction grammar.

Example: `<grammars version="1.0" debug="true" case="sensitive">`

Attribute	Value	Default	Description
`version`	<version string>	none	An optional character string that provides version information for the grammar.
`case`	sensitive insensitive inherited	inherited	Determines whether a match is case sensitive. The value `inherited` takes the value from the application level, which in the case of Eduction applications is usually `sensitive`.
`debug`	true false	false	Displays verbose information for the grammars element while `edktool` compiles the grammar.

<include>

Element: include

Child Elements: <publish>

Description: References another Eduction grammar file for inclusion.

Example: `<include path="winter_names.ecr" type="private"/>`

Attribute	Value	Default	Description
`path`	<path to the grammar file>		The path to the grammar file to include. A value is required.
`type`	public private	public	The default setting of `public` retains the private/public visibility of entities in an included XML grammar (included ECR grammars, by definition of a compiled grammar, only contain public entities). Set `type` to **`private`** to hide the included public entities in the file that includes the grammar.

<publish>

Element: publish

Child Elements: none

Description: Makes a private entity public. The entity can be anywhere in an included XML file chain.

> **NOTE:** You cannot access private entities in a compiled ECR file, so that even if you know the name of the private entity, `publish` cannot make it public.

Example: `<publish name="grammar2/g2e2"/>`

Attribute	Value	Default	Description
`name`	<entity name>		The name of the private entity from an included XML file that you want to make public. A value is required.

<grammar>

Element: grammar

Child Elements: <extern>, <entity>

Description: Defines a grammar, which is a collection of entities. You use entities for matching.

Example: `<grammar name="grammar1" case="inherited" extend="disallow" debug="inherited">`

Attribute	Value	Default	Description
`name`	<grammar name>		The name of the grammar. A value is required.
`case`	sensitive insensitive inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the <grammars> parent.
`extend`	append replace disallow	disallow	The mode to use for updating a grammar. Set this option to **append** to add the new entities to the existing grammar definition. Set it to **replace** to ignore the existing definition and use the new one. Set `extend` to **disallow** to prevent the new definition adding to or replacing values if there is already a grammar with the same name.
`debug`	true false inherited	inherited	Displays verbose information for the dictionary element during compilation. The value `inherited` accepts the debug mode of the `grammars` parent.

<extern>

Element: extern

Child Elements: none

Description: Identifies an external grammar by name so that you do not have to explicitly name the grammar when you refer to the entities that it contains. For example, if the external grammar is called `grammar1`, and it has an entity `entity1`, then in your current grammar, you can refer to the entity as `entity1` rather than `grammar1/entity1`.

Example: `<extern name="grammar2"/>`

Attribute	Value	Default	Description
`name`	<grammar name>		The name of the external grammar. A value is required.

<entity>

Element: entity

Child Elements: <entry>, <pattern>

Description: Defines an entity that you can use for matching.

Example: `<entity name="entity1" type="public" case="insensitive" extend="disallow" debug="true">`

Attribute	Value	Default	Description
`name`	<grammar name>		The name of the grammar. A value is required.
`type`	public private	private	Defines the entity as public or private.
`case`	sensitive insensitive inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the <grammars> parent.
`extend`	append replace disallow	disallow	Extends or replaces an existing entity definition.
`debug`	true false inherited	inherited	Displays verbose information for the entity element during compilation. The value `inherited` accepts the debug mode of the `grammar` parent.

<entry>

Element: entry

Child Elements: <headword>, <synonym>

Description: An entry represents an individual value that an entity matches. The entry has one or more attributes such as the actual phrase that returns (the *headword*), the case, and so on.

Example:`<entry headword="mat" score=".3" case="inherited" debug="inherited">`

Attribute	Value	Default	Description
`headword`	#CDATA		The dictionary entry. You can specify headword as either an attribute or a subelement, but you must not use both.
`score`	>= 0	1	A score to use to assign any weightings to the matches. • A score of `1` is the default score. • A score of `0` always excludes the matching tag from the results. You can use this value to specify exceptions to grammar rules. You can use these weightings for a variety of purposes: • They can represent the confidence you have in the accuracy of the match (where a value of `1` represents certainty, and lower values represent lesser confidence). • They can represent the importance of a match. For example, in the sentiment grammars the score represents the strength of the sentiment. When there are multiple scores, Eduction multiplies them together. For example, if a match on an entity has a score of `1.5`, and you use that entity in another entity that has a score of `0.4`, the resulting score is `0.6`. If you specify a minimum score during extraction, Eduction extracts only those matches with a sufficiently high score. You can also display the exact scores of any match during extraction. **NOTE:** Micro Focus recommends that you set entity and pattern scores to be no lower than `0.01` and no higher than `100`.
`case`	sensitive insensitive inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the <entity> parent.
`debug`	true false inherited	inherited	Displays verbose information for the entry element during compilation. The value `inherited` accepts the debug mode of the <entity> parent.

<headword>

Element: headword

Child Elements: none

Description: A headword is the exact sequence of characters that produce an entity match.

Example: See example in <entry>, on page 300.

Attribute	Value	Default	Description
`<element contents>`	<the headword>		The headword value. **NOTE:** If the `entry` element contains a headword attribute, it cannot have a headword subelement.
`case`	sensitive insensitive inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the <entry>, on page 300 parent.
`score`	>= 0	1	A score to use to assign any weightings to the matches. • A score of `1` is the default score. • A score of `0` always excludes the matching tag from the results. You can use this value to specify exceptions to grammar rules. You can use these weightings for a variety of purposes: • They can represent the confidence you have in the accuracy of the match (where a value of `1` represents certainty, and lower values represent lesser confidence). • They can represent the importance of a match. For example, in the sentiment grammars the score represents the strength of the sentiment. When there are multiple scores, Eduction multiplies them together. For example, if a match on an entity has a score of `1.5`, and you use that entity in another entity that has a score of `0.4`, the resulting score is `0.6`. If you specify a minimum score during extraction, Eduction extracts only those matches with a sufficiently high score. You can also display the exact

Attribute	Value	Default	Description
			scores of any match during extraction. **NOTE:** Micro Focus recommends that you set entity and pattern scores to be no lower than `0.01` and no higher than `100`.

<synonym>

Element: synonym

Child Elements: none

Description: A synonym is an alternative sequence of characters to a headword. Synonym matching produces an entity match, but returns the headword in place of the matching synonym. For example, if you search for *dog* with the synonym *canine* enabled, matches for *canine* return as if they matched *dog*.

Example:

```
<entry headword="Vatican City">
   <synonym score="1.2">The Vatican</synonym>
   <synonym score="1.1">Holy See</synonym>
   <synonym>Città del Vaticano</synonym>
   <synonym>Citta del Vaticano</synonym>
</entry>
```

Attribute	Value	Default	Description
`<element contents>`	<the synonym>		The synonym value.
`case`	sensitive insensitive inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the <entry>, on page 300 parent.
`score`	>= 0	the same as for the parent `<entry>`	The score value for the synonym. By default, all synonyms are assigned the same score as the parent entry. If you change the score, the synonym match returns the parent entry as usual, but with the adjusted score.

<pattern>

Element: pattern

Child Elements: none

Description: Defines a pattern used for matching.

Example: `<pattern score=".1" case="insensitive" replace="replacechars" insert_before="prefix_" insert_after="_suffix">cat</pattern>`

Attribute	Value	Default	Description
`pattern`	<actual pattern>		A value is required.
`score`	>= 0	1	A score to use to assign any weightings to the matches. • A score of `1` is the default score. • A score of `0` always excludes the matching tag from the results. You can use this value to specify exceptions to grammar rules. You can use these weightings for a variety of purposes: • They can represent the confidence you have in the accuracy of the match (where a value of `1` represents certainty, and lower values represent lesser confidence). • They can represent the importance of a match. For example, in the sentiment grammars the score represents the strength of the sentiment. When there are multiple scores, Eduction multiplies them together. For example, if a match on an entity has a score of `1.5`, and you use that entity in another entity that has a score of `0.4`, the resulting score is `0.6`. If you specify a minimum score during extraction, Eduction extracts only those matches with a sufficiently high score. You can also display the exact scores of any match during extraction. **NOTE:** Micro Focus recommends that you set entity and pattern scores to be no lower than `0.01` and no higher than `100`.
`case`	sensitive insensitive inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the `grammars` parent.
`replace`	<text to replace the match>	<no default>	Eduction replaces the text of the match with the specified text.
`insert_before`	<text to insert before	<no default>	Eduction inserts the specified text before the text of the match.

Attribute	Value	Default	Description
	the match>		
`insert_after`	<text to insert after the match>	<no default>	Eduction inserts the specified text after the text of the match.
`debug`	true false inherited	inherited	Displays verbose information for the pattern element during compilation. The value `inherited` accepts the debug mode of the <entity> parent.

Regular Expressions

This section describes the regular expressions syntax that Eduction supports.

The Eduction engine parser interprets regular expression syntax nearly identically to the UNIX regular expression syntax. The regular expression syntax also includes some extensions for matching substrings.

Operators

The following table describes the base regular expression operators available in the Eduction engine, and the pattern the operator matches.

Operator	Matched Pattern
`\`	Quote the next metacharacter.
`^`	Match the beginning of a line.
`$`	Match the end of a line.
`.`	Match any character (except newline).
`|`	Alternation.
`()`	Used for grouping to force operator precedence.
`[xy]`	The character `x` or `y`.
`[x-z]`	The range of characters between `x` and `z`.
`[^z]`	Any character except `z`. **NOTE:** For performance reasons, Micro Focus recommends that you explicitly list all the characters that you want to match, rather than using this

Operator	Matched Pattern
	operator. **NOTE:** To use negated character classes in case-insensitive entities, you must include letters in both cases, for example `[^Zz]` rather than `[^z]`.

Quantifiers

Operator	Matched Pattern
`*`	Match 0 or more times.
`+`	Match 1 or more times.
`?`	Match 0 or 1 times.
`{n}`	Match exactly *n* times.
`{n,}`	Match at least *n* times.
`{n,m}`	Match at least *n* times, but no more than *m* times.

Metacharacters

Operator	Matched Pattern
`\t`	Match tab.
`\n`	Match newline.
`\r`	Match return.
`\f`	Match formfeed.
`\a`	Match alarm (bell, beep, and so on).
`\e`	Match escape.
`\v`	Match vertical tab.
`\021`	Match octal character (in this example, 21 octal).
`\xF0`	Match hex character (in this example, F0 hex).
`\x{263a}`	Match wide hex character (Unicode).
`\w`	Match word character: `[A-Za-z0-9_]`.
`\W`	Match non-word character: `[^A-Za-z0-9_]`.

Operator	Matched Pattern
`\s`	Match whitespace character. This metacharacter also includes `\n` and `\r`: `[ \t\n\r]`.
`\S`	Match non-whitespace character: `[^ \t\n\r]`.
`\d`	Match digit character: `[0-9]`.
`\D`	Match non-digit character: `[^0-9]`.
`\b`	Match word boundary.
`\B`	Match non-word boundary.
`\A`	Match start of string (never match at line breaks).
`\Z`	Match end of string. Never match at line breaks; only match at the end of the final buffer of text submitted for matching.
`\p{`*`class`*`}`	Match any character that belongs to the specified Unicode character class. For example, `\p{Sc}` matches any currency symbol. You can omit the braces for single-character class names: `\p{C}` and `\pC` are equivalent. For a list of supported character classes, see Supported Unicode Character Classes, on page 309.
`\P{`*`class`*`}`	Match any character that does not belong to the specified Unicode character class. For example `\P{Sc}` matches any character that is not a currency symbol. You can omit the braces for single-character class names: `\P{C}` and `\PC` are equivalent. For a list of supported character classes, see Supported Unicode Character Classes, on page 309. **NOTE:** For performance reasons, Micro Focus recommends that you avoid using negated character classes where possible.

Extensions

Operator	Matched Pattern

`(?A:`*`entity`*`)`	Match a previously defined entity, and copy it into the definition of the new entity.
	
	For example:
	
	`<include path="number_types_eng.ecr"/>`
	`  <entity name="fracpos" type="private">`
	`  <pattern>(?A:number/fracalpha/eng)</pattern>`
	`  </entity>`
	
	Copying an entity improves pattern execution speed, but increases compilation time and memory usage. Micro Focus recommends that you use reference `(?A^` in all cases, unless your grammar file is very simple, and extraction speed is critical.

Operator	Matched Pattern
(?A^ *entity*)	Match a previously defined entity, and reference it in the definition of the new entity. Referencing an entity minimizes the size and memory usage of the grammar, but can decrease performance. The performance impact depends on the size and structure of the grammar.

(?A!*expr*)	Match the expression *expr* but exclude its output. This option designates an expression that helps identify an entity, but is not part of it.
	
	For example:
	
	`<entity name="age_landmark" type="private">`
	`  <pattern>Age:{0,1}\s*</pattern>`
	`  <pattern>Años:{0,1}\s*</pattern>`
	`</entity>`
	`<entity name="age" type="public">`
	`  <pattern>(?A!(?A^age_landmark))[1-9][0-9]?</pattern>`
	
	When you use this grammar to search the following text:
	
	`Name: Simon. Age: 32. Address. 12 Fifth Street, Las Vegas.`
	
	The grammar returns the text `32` but ignores `12`, because it does not have the prefix "`Age:`", which is matched upon but excluded from the output.

(?A= *component* :*expr*)	Define a component in an entity definition. A component is a named part of an entity.
	
	For example, the following grammar defines `areacode` and `main` as components:
	
	`<grammars>`
	`  <grammar name="number">`
	`  <entity name="phone" type="public">`
	`  <pattern>(?A=areacode:[0-9]{3})-(?A=main:[0-9]{3}-[0-9]{4})</pattern>`
	`  </entity>`
	`  </grammar>`
	`</grammars>`
	
	If the data contains the following phrase:
	
	`The phone number is 408-555-1342.`
	
	and the following configuration options are set:
	
	`OutputSimpleMatchInfo=false`
	`EnableComponents=true`
	
	then the output displays the `areacode` value `408` and the main value `555-1342` separately.

Supported Unicode Character Classes

This section lists the Unicode character classes that are supported by the regular expression engine. Use the one- or two-letter class name in your patterns. For example to match a currency symbol:

```
\p{Sc}
```

The supported classes are:

- `C` - Other
- `Cc` - Control
- `Cf` - Format
- `Co` - PrivateUse
- `Cs` - Surrogate
- `L` - Letter
- `Ll` - LowercaseLetter
- `Lm` - ModifierLetter
- `Lo` - OtherLetter
- `Lt` - TitlecaseLetter
- `Lu` - UppercaseLetter
- `M` - Mark
- `Mc` - SpacingMark
- `Me` - EnclosingMark
- `Mn` - NonSpacingMark
- `N` - Number
- `Nd` - DecimalNumber
- `Nl` - LetterNumber
- `No` - OtherNumber
- `P` - Punctuation
- `Pc` - ConnectorPunctuation
- `Pd` - DashPunctuation
- `Pe` - ClosePunctuation
- `Pf` - FinalPunctuation
- `Pi` - InitialPunctuation
- `Po` - OtherPunctuation

- `Ps` - OpenPunctuation
- `S` - Symbol
- `Sc` - CurrencySymbol
- `Sk` - ModifierSymbol
- `Sm` - MathSymbol
- `So` - OtherSymbol
- `Z` - Separator
- `Zl` - LineSeparator
- `Zp` - ParagraphSeparator
- `Zs` - SpaceSeparator

Character class assignments are as provided by unicode.org - see http://www.unicode.org/Public/UNIDATA/UnicodeData.txt.

Eduction Grammar DTD

The XML DTD describing the Eduction grammar (that is, `edk.dtd`) is as follows:

```
<!ELEMENT grammars (include*, grammar*)>
<!ATTLIST grammars
version CDATA #IMPLIED
case (sensitive|insensitive|inherited) "inherited"
debug (true|false) "false"
>
<!ELEMENT include (publish*)>
<!ATTLIST include
path CDATA #REQUIRED
type (private|public) "public"
>
<!ELEMENT publish EMPTY>
<!ATTLIST publish
name CDATA #IMPLIED
>
<!ELEMENT grammar (extern*,entity+)>
<!ATTLIST grammar
name CDATA #REQUIRED
case (sensitive|insensitive|inherited) "inherited"
extend (append|replace|disallow) "disallow"
debug (true|false|inherited) "inherited"
>
<!ELEMENT extern EMPTY>
<!ATTLIST extern
name CDATA #REQUIRED
```

```
>
<!ELEMENT entity (entry*,pattern*)+>
<!ATTLIST entity
name CDATA #REQUIRED
type (private|public) "private"
case (sensitive|insensitive|inherited) "inherited"
extend (append|replace|disallow) "disallow"
debug (true|false|inherited) "inherited"
>
<!ELEMENT entry (headword?,synonym*)>
<!ATTLIST entry
headword CDATA #IMPLIED
score CDATA "1"
case (sensitive|insensitive|inherited) "inherited"
debug (true|false|inherited) "inherited"
>
<!ELEMENT headword (#PCDATA)>
<!ATTLIST headword
score CDATA "1"
case (sensitive|insensitive|inherited) "inherited"
>
<!ELEMENT synonym (#PCDATA)>
<!ATTLIST synonym
case (sensitive|insensitive|inherited) "inherited"
>
<!ELEMENT pattern (#PCDATA)>
<!ATTLIST pattern
score CDATA "1"
case (sensitive|insensitive|inherited) "inherited"
replace CDATA #IMPLIED
insert_before CDATA #IMPLIED
insert_after CDATA #IMPLIED
>
```

Chapter 15: edktool Command-Line Options

This section provides a reference for the command-line options that you can use with the edktool command-line tool.

edktool Options

This section describes how to use `edktool` to:

- list the entities that exist in a grammar file.
- test and benchmark extraction.
- compile grammar files.

To view a summary of the options you can use with `edktool`, run the following command:

```
edktool help
```

The tool also provides more detailed help for some features. For example, to view more information about the `compile` feature, run the following command:

```
edktool help compile
```

Assess

This command assesses the performance and accuracy of an Eduction grammar against a set of pre-tagged examples.

You must supply a text file with one phrase on each line; the `Assess` feature checks whether each line contains a match.

You must specify at least one input file, using the `-v` parameter or the `-w` parameter. If required, you can specify both of these parameters.

The following table describes the parameters for this command.

`-l <`*`licensefile`*`>`	The file containing a valid license key for Eduction. If you do not specify a license key, edktool attempts to load the license `licensekey.dat` in its current working directory. You must specify the license parameter if your license is in a different location.

`-c <configfile>`	A configuration file to control the assessment. See Eduction Configuration File, on page 99. You can specify one or more grammar files and one or more entities in place of a configuration file. Specifying a configuration file overrides the grammar (`-g`) or entity (`-e`) parameters.
`-g <grammarfile>`	A grammar file to use. Edktool ignores this option if you set a configuration file with `-c`. If you provide a grammar file but you do not specify any entities with `-e`, Eduction extracts all entities in the grammar file. You can use wildcard expressions in this parameter. See Wildcard Expressions in edktool, on page 325.
`-e <entity>`	The entities to extract. Separate multiple entities with a comma. Edktool ignores this option if you set a configuration file with `-c`. You can use wildcard expressions in this parameter. See Wildcard Expressions in edktool, on page 325.
`-x`	(Optional) Modifies the behavior so that `Assess` checks for exact matches.
`-m <matched entities>`	(Optional) This parameter does not change the extraction behavior, but enables you to check which entities are producing the matches.
`-v <valid_input>`	A file of phrases where a match would be valid.
`-w <invalid_input>`	A file of phrases where a match would be invalid.
`-a`	(Optional) Display additional output, including the results for every phrase in your input files. By default, the output includes explanations of each failure, and statistics such as recall, precision, and F1 (depending on the type of input file you provide).
`-o <outputfile>`	(Optional) Send the output to a file. By default, Eduction sends output to the console. The output is a list of all phrases that failed. For valid input this would be a phrase that contained no match; for invalid input this would be a phrase that contained a match.
`-q`	(Optional) Run in quiet mode. In this case, edktool removes all descriptive messages from the output and shows only a list of examples that failed, in the form `"FAIL: "text" is matched by "entity""` or similar, depending on the test specifications. If you also set the `-a` parameter, examples that pass are also included in the output.

For more information on how to use the Assess feature to check the effectiveness and performance of your grammar files, see Assess and Measure Eduction Grammars, on page 95.

Example

```
edktool a -l <license> -c <configuration_file> [-a] [-o <output_file>]
```

Run several assessments from a single Eduction configuration file.

The configuration file must contain a numbered `[assessmentN]` section for each assessment you want to run. You must specify the input files, the entities to match, and whether to require exact matches. For example:

```
[assessment0]
valid=data.txt

[assessment1]
entities=entity1,entity2
valid=match.txt
invalid=should_not_match.txt
exact=true
```

You can specify multiple entities by separating them with commas, or by using wildcard expressions (see Wildcard Expressions in edktool, on page 325).

Benchmark

This command runs edktool in benchmarking mode. This mode runs multiple concurrent extraction sessions, several times to test the performance of a grammar. Edktool reads the input document once, and feeds it into each session. It produces timing information after all runs are complete.

The following table describes the parameters for this command.

`-l <licensefile>`	The file containing a valid license key for Eduction. If you do not specify a license key, edktool attempts to load the license `licensekey.dat` in its current working directory. You must specify the license parameter if your license is in a different location.
`-i <inputfile>`	The file to perform entity extraction on. The input file must be plain text.
`-c <configfile>`	A configuration file controlling the extraction. See Eduction Configuration File, on page 99. You can specify one or more grammar files and one or more entities in place of a configuration file. Specifying a configuration file overrides the grammar or entity parameters.
`-g <grammarfile>`	A grammar file to use. Edktool ignores this option if you set a configuration file with `-c`. If you provide a grammar file but do not specify any entities with `-e`, Eduction extracts all entities in the grammar file.

	NOTE: You can use the `MinScore` parameter only if you use `-c`. Without a configuration file, you can specify a grammar that supports scoring, but edktool does not filter out matches based on those scores.
	You can use wildcard expressions in this parameter. See Wildcard Expressions in edktool, on page 325.
`-e <entity>`	The entities to extract. Separate multiple entities with a comma. Edktool ignores this option if you set a configuration file with `-c`.
	You can use wildcard expressions in this parameter. See Wildcard Expressions in edktool, on page 325.
`-d`	(Optional) Return details of the matching strings and their locations in the input file.
`-s <sessions>`	The number of sessions to run concurrently during each iteration of the benchmarking test.
`-n <number>`	The number of iterations of the benchmarking test to run.
`-b`	Set this parameter to read the input file in binary mode, rather than text mode. If you create a grammar file that matches entities with only Windows (CR LF) line endings and you run edktool on Windows, edktool must read the input file in binary mode for it to find any matches. Micro Focus recommends that you create grammar files capable of handling both Windows and Unix line endings.

The benchmarking command runs the specified number of concurrent sessions and iterations and then displays the timing for each run, with a summary showing:

- the total number of observations.
- maximum and minimum times.
- the standard deviation.

Compile

This command creates a compiled Eduction grammar file.

`edktool c <grammarfile>`

The following table lists the optional parameters for this command.

`-l <licensefile>`	The file containing a valid license key for Eduction.
	If you do not specify a license key, edktool attempts to load the license `licensekey.dat` in its current working directory. You must specify the license parameter if your license is in a different location.
`-i <inputfile>`	The uncompiled (source) XML Eduction grammar file to process.

`-e <entity>`	A comma-separated list of entities to include in the output file. When you include entities in the command line, Eduction includes only those entities in the output file. Otherwise, Eduction includes all entities from the input file in the output file. You can use wildcard expressions in the entity list. See Wildcard Expressions in edktool, on page 325.
`-o <outputfile>`	The output file name. If you do not specify the output file name, Eduction creates an output file using the XML grammar file name with `.ecr` appended.
`-c <configfile>`	The compilation configuration file to use. See Use a Compilation Configuration File, on page 93.

When you compile a grammar, the XML file must follow the Eduction syntax rules for grammar files. The ECR file is a proprietary format that is optimized for fast loading into the Eduction engine at run time. While the engine can load XML grammar files, as well as compiled ECR files, compiling a grammar file makes loading quicker.

Compiled grammar files are binary files, which you cannot read. You can use the `List` option to view the public entities in a compiled grammar file.

Examples

To compile `mygrammar.xml` into `mygrammar.ecr`:

```
edktool c mygrammar.xml
```

To compile all the entities in the `common` entity type in `mygrammar.xml` into `compiledgrammar.ecr`:

```
edktool c -i mygrammar.xml -e common/* -o compiledgrammar.ecr
```

Extract

This command extracts entities from a document. It can print the output to a file, or to the console. You can use this option to test your grammars.

The following table describes the available parameters for this command.

`-l <licensefile>`	The file containing a valid license key for Eduction. If you do not specify a license key, edktool attempts to load the license `licensekey.dat` in its current working directory. You must specify the license parameter if your license is in a different location.
`-i <inputfile>`	The file to perform entity extraction on. The input file must be a UTF-8 encoded plain text file.
`-c <configfile>`	A configuration file controlling the extraction. See Eduction Configuration File, on page 99.

	You can specify one or more grammar files and one or more entities in place of a configuration file. Specifying a configuration file overrides the grammar or entity parameters.
`-g <grammarfile>`	A grammar file to use. Edktool ignores this option if you set a configuration file with `-c`. If you provide a grammar file but do not specify any entities with `-e`, Eduction extracts all entities in the grammar file. You can use wildcard expressions in this parameter. See Wildcard Expressions in edktool, on page 325.
`-e <entity>`	The entities to extract. Separate multiple entities with a comma. Edktool ignores this option if you set a configuration file with `-c`. You can use wildcard expressions in this parameter. See Wildcard Expressions in edktool, on page 325.
`-o <outputfile>`	The file to write the results of the extraction to. The output file is an XML file that contains the matched entities.
`-q`	(Optional) Run in quiet mode. In this case, `edktool` removes all descriptive messages from the output and shows the XML matchlist only (that is, an XML document with all the matches and any configured metadata).
`-r <redaction_file>`	(Optional) The name of a copy of the input file to produce, with all matches redacted. For example: `The driver ########## was questioned.` See Redact Extraction Results, on the next page.
`-b`	Read the input file in binary mode, rather than text mode. If you create a grammar file that matches entities with only Windows (CR LF) line endings and you run edktool on Windows, edktool must read the input file in binary mode for it to find any matches. Micro Focus recommends that you create grammar files capable of handling both Windows and Unix line endings.

The extract option requires an input file (in plain text format) and either a configuration file or a grammar file. If you do not provide a configuration file, `edktool` searches the file for any specified entities in the specified grammar (or all entities, if none are specified). For example, in the simplest command line:

```
C:\>edktool e -i myData.txt -g grammar1.ecr,grammar2.ecr
```

This command runs edktool without a configuration file. It processes the data file `myData.txt` with the grammar files `grammar1.ecr` and `grammar2.ecr`. Eduction identifies all the entities in the two grammar files, and matches on these. The output is sent to the console in XML format, identifying matches in the data file and using the entity names to generate field names for the matches that contain the matched data. It matches the entire body of the plain text input file.

Redact Extraction Results

You can enable redaction on extracted matches in edktool either by setting `RedactedOutput` to **`True`** in the edktool configuration file, or by specifying a redaction file using the `-r` parameter at the command line.

The entities identified as matches by `edktool` are redacted from the input text to form the redacted output. For example:

Input:

```
The driver Joe Bloggs was questioned.
```

Output:

```
The driver ########## was questioned.
```

Eduction sends redacted output to the file specified in the `-r` parameter. If you do not specify this argument but you have enabled redaction in the configuration file, Eduction displays redacted output in the console after the list of matches, unless you have set the `-q` parameter at the command line to enable quiet mode. In quiet mode, edktool does not display redacted output in the console.

Examples

```
edktool e -i myPlainTextFile.txt -g myGrammar.ecr
```

Extracts all entities in `myGrammar.ecr` from `myPlainTextFile.txt`, sending the output to the console in XML format, with the field names for the matching text automatically generated from the entity names found in `myGrammar.ecr`.

XML Output Format

This section describes the XML output tags and attributes that edktool returns when you run the extract command with a plain text file.

<MATCH>

The details of a match.

Attribute	Description
`EntityName`	The name of the matched entity.
`Offset`	The match start position in the buffer, in bytes.
`OffsetLength`	The match start position in the buffer, in characters. This value returns only when `OutputSimpleMatchInfo=FALSE`. See OutputSimpleMatchInfo, on page 361.
`Score`	The final score for this match.
`NormalizedTextSize`	The length of the normalized text, in bytes.

Attribute	Description
	This value returns only when `OutputSimpleMatchInfo=FALSE`. See OutputSimpleMatchInfo, on page 361.
`NormalizedTextLength`	The length of the normalized text, in characters. This value returns only when `OutputSimpleMatchInfo=FALSE`. See OutputSimpleMatchInfo, on page 361.
`OriginalTextSize`	The length of the matched text, in bytes. This value returns only when `OutputSimpleMatchInfo=FALSE`. See OutputSimpleMatchInfo, on page 361.
`OriginalTextLength`	The length of the matched text, in characters. This value returns only when `OutputSimpleMatchInfo=FALSE`. See OutputSimpleMatchInfo, on page 361.

The following table lists the child elements that `<MATCH>` contains.

Child Elements	Description
`<ORIGINAL_TEXT>`	The text from the input that was matched. This value returns only when `OutputSimpleMatchInfo=FALSE`. See OutputSimpleMatchInfo, on page 361.
`<NORMALIZED_TEXT>`	The normalized match text. Normalized text might differ from the original text if, for example, a synonym was matched (in which case the original text is replaced with the headword), or a pattern used the replace attribute or the `(?A!...)` syntax. The text might also be adjusted by post-processing scripts, if configured.
`<COMPONENTS>`	The match components, if present for this entity and when `EnableComponents=TRUE`.

<COMPONENT>

The details of a component.

Attribute	Description
`Name`	The component name.
`Text`	The component text (normalized, if applicable).
`Offset`	The component start position in the matched text, in bytes.

Attribute	Description
`OffsetLength`	The component start position in the matched text, in characters.
`TextSize`	The component text length, in bytes.
`TextLength`	The component text length, in characters.

Generate

This command generates an uncompiled XML source file from a plain text file.

The plain text file must contain a list of headwords that you want to use to create a single entity. You cannot include patterns, synonyms, scoring, and so on in the plain text file.

The generate command creates a headword from each line in the plain text file. It ignores whitespace and blank lines. You can include comment lines (a line beginning with `//`). Edktool skips these comment lines when it generates the XML grammar.

`edktool g -i <inputfile>`

The following table describes the parameters for this command.

`-i <inputfile>`	The plaintext grammar file to process. This file must contain one potential match on each line.
`-e <entity>`	The name of the entity to create (the resulting XML grammar file contains a single entity). If you do not specify a name, the entity is given a default name based on the name of the input file. You can use wildcard expressions in this parameter. See Wildcard Expressions in edktool, on page 325.
`-o <outputfile>`	The output file name.

Help

This command lists the valid `edktool` commands, with brief descriptions for each.

List

This command lists the entities in an uncompiled (source) XML Eduction grammar file or a compiled ECR grammar file. Listing the contents of an XML file lists all entities in the file, both private and public. Listing the contents of a compiled ECR file lists all public entities.

> **NOTE:** The compiled ECR file does not include any private entities that are not referenced by the public entities.

The entities, and components if you use the `-a` option, are listed in alphabetical order.

To enable this feature, type `edktool l <grammarfile>` at the command line.

The following table describes the optional parameters for this command.

`-a`	List the license requirements for a particular compiled grammar file, as well as the components that the entity can return. For example, the following output: `category: place languages: English or French` indicates that the user must be licensed for either English or French in the place category. If multiple lines appear, then the license must satisfy the conditions in every line.
`-q`	Run in quiet mode. In this case, `edktool` removes all descriptive messages from the output and shows the entity list only. The output includes components if you also set the `-a` parameter.

Example

To list all public entities in the compiled grammar file `mygrammar.ecr`:

```
edktool list mygrammar.ecr
```

Measure

This command measures the precision and recall between extraction runs by comparing the *expected results* of entity extraction with the *actual results*.

You create expected results once and keep them as a base reference for ongoing tests. You then generate actual results as required each time you modify a grammar. Edktool compares the two results to generate precision and recall information.

To generate expected results, run `edktool -extract`, and then revise the generated output file so that it contains the correct matches. From then on, you use `edktool -extract` to create the actual results, and the `measure` command compares the two files to generate precision and recall information on an ongoing basis.

The following table describes the parameters for this command.

`-e <expectedfile>`	The expected results file.
`-a <actualfile>`	The actual results file from subsequent extraction runs with modified grammar files.
`-o <resultsfile>`	The output file, including the results for precision, recall, and differences.
`-q`	(Optional) Run in quiet mode. In this case, `edktool` removes all descriptive messages from the output and shows only the differences between the expected and actual output.

For more information on how to use `measure` to check the effectiveness and performance of your grammar files, see Assess and Measure Eduction Grammars, on page 95.

Example

The following example compares `expected.xml` with `actual.xml` and puts the difference in `difference.xml`, including precision and recall. Quiet mode is enabled, so all descriptive messages are removed from the output.

```
edktool m -e expected.xml -a actual.xml -o difference.xml -q
```

Permissions

This command reads any specified directory and returns a list of all compiled grammar files inside it that you can access using the specified license.

```
edktool p -d <directory> -l <licencefile>
```

The following table describes the optional parameters for this command.

`-l <licensefile>`	The file containing a valid license key for Eduction. If you do not specify a license key, edktool attempts to load the license `licensekey.dat` in its current working directory. You must specify the license parameter if your license is in a different location.
`-a`	Set this parameter to return a list of all compiled grammar files inside the directory that are **not** accessible under the specified license.
`-q`	Run in quiet mode. In this case, `edktool` removes all descriptive messages from the output and shows a list of file names only, in the format `Valid: filename.ecr` or, if you also included the `-a` parameter, `Invalid: filename.ecr`.

Unify

This command generates a grammar file that contains one or more combined entities.

A *combined entity* is a single entity that combines the patterns for a predetermined set of entities. For example, in the PII grammar package there is an entity named `pii/address/all` that matches a postal address from any supported country. This entity was created by combining existing entities such as `pii/address/gb`, `pii/address/fr`, `pii/address/de`, and so on.

Eduction can find matches for a combined entity faster than it can find matches for the equivalent list of source entities, because in the combined entity the list of patterns is optimized as a single unit.

Eduction automatically returns matches using the source entity names, so you do not lose any information about which entity produced a match. For example, when you run Eduction and find matches for `pii/address/all`, Eduction reports matches for `pii/address/gb` or `pii/address/fr` rather than for the combined entity.

The following table describes the parameters for this command.

`-l <licensefile>`	The file containing a valid license key for Eduction. If you do not specify a license key, edktool attempts to load the license `licensekey.dat` in its current working directory. You must specify the license parameter if your license is in a different location.
`-g <grammarfile>`	A comma-separated list of grammar files that contain the source entities to combine.
`-e <entity>`	The definition for a combined entity, in the form: `combined-name=source-name1,source-name2,...` If you want to create more than one combined entity you can use the `-e` parameter multiple times.
`-o <outputfile>`	The output grammar file.

Examples

The following example creates a grammar file named `custom.ecr` that contains an entity named `pii/address/custom`, by combining the address entities for France and Germany.

```
edktool unify -o custom.ecr -g address.ecr -e
pii/address/custom=pii/address/fr,pii/address/de
```

You can create more than one combined entity by passing the `-e` parameter multiple times.

```
edktool unify -o custom.ecr -g address.ecr,telephone.ecr -e
pii/address/custom=pii/address/fr,pii/address/de -e
pii/telephone/context/custom=pii/telephone/context/fr,pii/telephone/context/de
```

Validate

This command validates an Eduction configuration.

`edktool v`

The following table lists the parameters for this command.

`-c <configfile>`	The configuration file to validate.
`-l <licensefile>`	The file containing a valid license key for Eduction. If you do not specify a license key, edktool attempts to load the license `licensekey.dat` in its current working directory. You must specify the license parameter if your license is in a different location.
`-o <outputfile>`	(Optional) The output file name to use to output any errors.
`-i`	(Optional) Initialize an extraction session to verify that entities can be loaded from the configuration.

The following example shows the output from a validate command on an invalid configuration file.

```
Config file invalid
[Eduction]
//Referenced parameter configured with no value
TANGIBLECHARACTERS=
//Key deprecated
OUTPUTSCORES=True
//Key deprecated
OUTPUTSIMPLEMATCHINFO=False

[pii_postprocessing]
//Key not used
TYPE=lua
```

Wildcard Expressions in edktool

The `-e` and `-g` parameters in the Generate, Compile, Assess, Extract and Benchmark options in edktool support wildcard expressions. For example, if you want to use all of the available sentiment analysis files in the `grammars` directory, you can type `-e "grammars/sentiment_*.ecr"` instead of typing a lengthy comma-separated list of multiple files.

You can use the `*` wildcard to match any number of characters, or the `?` wildcard to match a single character.

> **NOTE:** In some cases (for example, on Linux operating systems), the command shell automatically expands wildcard expressions, which can produce unexpected results in Eduction. To avoid this, enclose your wildcard expression in quotation marks.

Chapter 16: Eduction Parameter Reference

This section lists the parameters that you can use in your Eduction configuration file.

You can use these parameters in the Eduction SDK either by creating an edk engine with an appropriate configuration file, or by using the API to set equivalent parameters after you create the engine. You can also use these parameters in your edktool configuration file.

> **NOTE:** For a list of parameters available in Eduction Server, refer to the *Eduction Server Reference*.

AllowDuplicates

DEPRECATED: The `AllowDuplicates` parameter is deprecated in Eduction version 12.5.0 and later.

This parameter is still available for existing implementations, but it might be incompatible with new functionality. The parameter might be deleted in future.

A list of document fields in which Eduction can write multiple results from a single entity. If you allow multiple results from a single entity by setting AllowMultipleResults, below to `TRUE`, and the input text contains more than one match to an entity, Eduction writes the results to multiple fields with the same name.

This parameter is used only when generating output in IDOL IDX format. It has no effect on XML.

You can specify multiple fields by separating them with commas.

Type:	String
Default:	
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`AllowDuplicates=ANIONIC_SURFACTANTS,PERSON`
See Also:	AllowMultipleResults, below EntityN, on page 346 EntityFieldN, on page 344

AllowMultipleResults

This parameter specifies how many results to return, when Eduction finds multiple matches at the same offset (starting position) in the input text. Eduction returns only one result by default, but you can choose to return all of the matches or up to one per entity.

Set this parameter to one of the following values:

- **`All`** or **`True`**. Eduction returns all results.
- **`OnePerEntity`**. Eduction returns up to one result per entity at each offset.
- **`No`** or **`False`**. Eduction does not return multiple results at the same offset.

This parameter can be useful when the same text has multiple interpretations. For example, if the input text contains the word `Georgia`, it might refer to a person's name, the U.S. state, or the country. By default, Eduction returns only one match. This behavior is appropriate if it is not important to you that `Georgia` has multiple interpretations. Set `AllowMultipleResults=All` to return all three matches. Set `AllowMultipleResults=OnePerEntity` to return one match from each entity.

Example

The following table shows how the results from Eduction change when you set the parameters `AllowMultipleResults` and `AllowOverlaps`.

In this example, the input is "The President of the United States of America is in London today to meet the British Prime Minister", and three entities have been defined:

- entity1 matches political offices, for example "President of the United States".
- entity2 matches corporate titles including "President".
- entity3 matches places including "United States" and "United States of America".

Parameters	AllowOverlaps=False	AllowOverlaps=True
AllowMultipleResults =False	Eduction returns the match "President of the United States" (entity1). The match "President" (entity2) is ignored because it shares the same starting point as "President of the United States" and `AllowMultipleResults=FALSE`. The matches "United States" and "United States of America" (entity3) are ignored because they overlap with "President of the United States" and `AllowOverlaps=FALSE`.	Eduction returns the match "President of the United States" (entity1). The match "President" (entity2) is ignored because it shares the same starting point as "President of the United States" and `AllowMultipleResults=FALSE`. Overlapping matches are allowed, so Eduction returns a match "United States of America" (entity3). The match "United States" (entity3) is ignored because it shares the same starting point as "United States of America" and `AllowMultipleResults=FALSE`.
AllowMultipleResults =OnePerEntity	Eduction returns the match "President of the United States" (entity1). Eduction returns the match "President" (entity2). Although it shares the same starting point as "President of the United States" it is matched by a different entity and	Eduction returns the match "President of the United States" (entity1). Eduction returns the match "President" (entity2). Although it shares the same starting point as "President of the United States" it is matched by a different entity and

	`AllowMultipleResults` is set to `OnePerEntity`. The matches "United States" and "United States of America" (entity3) are ignored because they overlap with "President of the United States" and `AllowOverlaps=FALSE`.	`AllowMultipleResults` is set to `OnePerEntity`. Overlapping matches are allowed, so Eduction returns a match "United States of America" (entity3). The match "United States" (entity3) is ignored because it shares the same starting point as "United States of America" and `AllowMultipleResults` is set to `OnePerEntity`.
AllowMultipleResults =True	Eduction returns the match "President of the United States" (entity1). Eduction returns the match "President" (entity2) because `AllowMultipleResults=True`. The matches "United States" and "United States of America" (entity3) are ignored because they overlap with "President of the United States" and `AllowOverlaps=FALSE`.	Eduction returns all of the matches. These are "President of the United States" (entity1), "President" (entity2), "United States" (entity3), and "United States of America" (entity3).

Type:	String
Default:	False
Required:	No
Configuration Section:	**Eduction**
Example:	`AllowMultipleResults=All`
See Also:	AllowOverlaps, on the next page EntityN, on page 346 EntityFieldN, on page 344 NonGreedyMatch, on page 359

AllowOverlaps

A Boolean value that specifies whether to return more than one match, when Eduction finds overlapping matches that start at different characters (offsets). To return overlapping matches set this parameter to **`True`**.

> **NOTE:** To specify whether to return overlapping matches that have the same offset, use the configuration parameter AllowMultipleResults, on page 329.

Example

The following table shows how the results from Eduction change when you set the parameters `AllowMultipleResults` and `AllowOverlaps`.

In this example, the input is "The President of the United States of America is in London today to meet the British Prime Minister", and three entities have been defined:

- entity1 matches political offices, for example "President of the United States".
- entity2 matches corporate titles including "President".
- entity3 matches places including "United States" and "United States of America".

Parameters	AllowOverlaps=False	AllowOverlaps=True
AllowMultipleResults =False	Eduction returns the match "President of the United States" (entity1). The match "President" (entity2) is ignored because it shares the same starting point as "President of the United States" and `AllowMultipleResults=FALSE`. The matches "United States" and "United States of America" (entity3) are ignored because they overlap with "President of the United States" and `AllowOverlaps=FALSE`.	Eduction returns the match "President of the United States" (entity1). The match "President" (entity2) is ignored because it shares the same starting point as "President of the United States" and `AllowMultipleResults=FALSE`. Overlapping matches are allowed, so Eduction returns a match "United States of America" (entity3). The match "United States" (entity3) is ignored because it shares the same starting point as "United States of America" and `AllowMultipleResults=FALSE`.
AllowMultipleResults =OnePerEntity	Eduction returns the match "President of the United States" (entity1).	Eduction returns the match "President of the United States" (entity1).

	Eduction returns the match "President" (entity2). Although it shares the same starting point as "President of the United States" it is matched by a different entity and `AllowMultipleResults` is set to `OnePerEntity`. The matches "United States" and "United States of America" (entity3) are ignored because they overlap with "President of the United States" and `AllowOverlaps=FALSE`.	Eduction returns the match "President" (entity2). Although it shares the same starting point as "President of the United States" it is matched by a different entity and `AllowMultipleResults` is set to `OnePerEntity`. Overlapping matches are allowed, so Eduction returns a match "United States of America" (entity3). The match "United States" (entity3) is ignored because it shares the same starting point as "United States of America" and `AllowMultipleResults` is set to `OnePerEntity`.
AllowMultipleResults =True	Eduction returns the match "President of the United States" (entity1). Eduction returns the match "President" (entity2) because `AllowMultipleResults=True`. The matches "United States" and "United States of America" (entity3) are ignored because they overlap with "President of the United States" and `AllowOverlaps=FALSE`.	Eduction returns all of the matches. These are "President of the United States" (entity1), "President" (entity2), "United States" (entity3), and "United States of America" (entity3).

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	**Eduction**
Example:	`AllowOverlaps=True`
See Also:	AllowMultipleResults, on page 329 NonGreedyMatch, on page 359

CantHaveFieldCSVs

> **DEPRECATED:** The `CantHaveFieldCSVs` parameter is deprecated in Eduction version 12.5.0 and later.
>
> This parameter is still available for existing implementations, but it might be incompatible with new functionality. The parameter might be deleted in future.

The names of fields that Eduction ignores when reading an XML file. This option allows you to specify fields in documents that you want to discard before the documents are stored.

To specify multiple fields, separate them with commas (there must be no space before or after a comma). You can use wildcards.

Type:	String
Default:	
Required:	No
Configuration Section:	**Server**
Example:	`CantHaveFieldCSVs=*/STANDARD_HEADER` In this example, any `STANDARD_HEADER` fields that a document contains are discarded before the document is stored in IDOL server.
See Also:	

CaseNormalization

The case conversion to use for all incoming text. To improve performance, use this parameter to convert all text to lowercase or uppercase before attempting to match text.

This parameter takes one of the following values:

- **None**. No case conversion.
- **Lower**. All incoming text is converted to lowercase.
- **Upper**. All incoming text is converted to uppercase.

If your grammar file consists of only lowercase or only uppercase characters but your text is mixed case, you can improve performance by setting `CaseNormalization` to **`Lower`** or **`Upper`** respectively. This provides a greater performance improvement than setting MatchCase, on page 355 to **`False`**.

If you set this parameter to `Lower` or `Upper`, set MatchCase, on page 355 to **`True`**.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**
Example:	`CaseNormalization=lower`
See Also:	CaseNormalizationBehavior, below CaseSensitiveFieldName, below MatchCase, on page 355

CaseNormalizationBehavior

The algorithm to use for case normalization. This parameter accepts one of the following values:

- **Default**. The default behavior.
- **Turkic**. Use this option with Turkic languages to ensure that case normalization performs correctly with the dotted and dotless "i" characters.

Type:	String
Default:	Default
Required:	No
Configuration Section:	**Eduction**
Example:	`CaseNormalizationBehavior=Turkic`
See Also:	CaseNormalization, on the previous page

CaseSensitiveFieldName

Set `CaseSensitiveFieldName` to **`True`** to preserve the case of configured field names. In this case, the field names are case sensitive.

Set `CaseSensitiveFieldName` to **`False`** to convert all field names to uppercase when Eduction produces matches.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`CaseSensitiveFieldName=True`
See Also:	

CellEntityMatchLimit*N*

The maximum number of matches to return for the entities specified by the corresponding CellEntityN, on the next page parameter. Eduction searches for cell entity matches in the columns that have a header that match the HeaderEntityN, on page 352 entities (up to HeaderEntityMatchLimitN, on page 351). After Eduction finds this number of matches in those columns, it stops searching for further matches, so Eduction does not return any further matches or spend time looking for them.

You might set this parameter if you want to see some matches for a particular entity, but would prefer to ignore further matches in favor of reducing the processing time.

If the CellEntityN, on the next page parameter specifies multiple entities, using a wildcard or a comma-separated list, the limit applies separately to each entity. For example, the following configuration would permit up to five matches for English dates, and up to five for French dates, and so on:

```
CellEntity0=pii/date/nocontext/eng,pii/date/nocontext/fre,pii/date/nocontext/spa
CellEntityMatchLimit0=5
```

Eduction applies the limit after post-processing, so any matches that are discarded by post-processing do not count towards the limit.

The limit applies only to matches in a single table. Eduction resets the limit when it encounters a table delimiter.

Type:	Integer
Default:	No limit
Required:	No
Configuration Section:	**Eduction**
Example:	With the following configuration Eduction returns a maximum of five matches for the `pii/date/nocontext/all` entity.

	CellEntity0=pii/date/nocontext/all CellEntityMatchLimit0=5
See Also:	CellEntityN, below HeaderEntityMatchLimitN, on page 351

CellEntity*N*

The entities to extract for the cell rows of input tables. This parameter allows you to extract entities from structured data.

When matching CSV or TSV input, Eduction matches the first one or more non-empty row of input against the configured header entities (see HeaderEntityN, on page 352 and MaxSearchHeaderRow, on page 358). In subsequent rows, Eduction matches individual cells against the cell entity corresponding to the matched header entity, if any.

For example:

```
HeaderEntity0=pii/date/dob/landmark/all
CellEntity0=pii/date/nocontext/all
```

This example matches date of birth landmark values in the header, and for all subsequent rows in that column, it extracts any date values.

You can specify multiple entities in a comma-separated list. If the table header matches any of the configured header entities, Eduction matches the cell content against any of the configured cell entities. This option might be useful if you want to match a particular entity in multiple languages, or if you want to include a custom entity.

You can also use wildcard expressions in the entity names. The * wildcard matches any number of characters, and the ? wildcard matches a single character.

For more information about table extraction, see Extract Entities from Tables, on page 61.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**
Example:	HeaderEntity0=pii/date/dob/landmark/all CellEntity0=pii/date/nocontext/all
See Also:	HeaderEntityN, on page 352 CellEntityMatchLimitN, on the previous page

CJKNormalization

This parameter allows you to specify how to normalize Chinese, Japanese, and Korean data before extraction.

You can set the following values:

- `Kana`. Normalize half width kana to full width kana.
- `OldNew`. Normalize old kanji to new kanji.
- `Number`. Normalize Chinese or kanji number characters to ASCII number characters.
- `HWNum`. Normalize full width number characters to ASCII number characters.
- `HWAlpha`. Normalize full width alphabet characters to ASCII alphabet characters.
- `SimpChi`. Normalize traditional Chinese to simplified Chinese.
- `FWJamo`. Normalize half width jamo to full width jamo.

Separate multiple options with a comma.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**
Example:	`CJKNormalization=SimpChi,Kana`
See Also:	

Databases

> **DEPRECATED:** The `Databases` parameter is deprecated in Eduction version 12.5.0 and later.
>
> This parameter is still available for existing implementations, but it might be incompatible with new functionality. The parameter might be deleted in future.

The names of the databases to which a document belongs. Eduction runs only on documents that belong to the comma-separated list of databases. If you do not list databases, Eduction runs on documents from all databases.

> **NOTE:** If you restrict `Databases`, and an IDX does not have a `DREDBNAME` entry for a document, Eduction does not match on that document. However, if you select all databases, Eduction does

match the document.

Type:	String
Default:	
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`Databases=DB1,DB2,DB3`
See Also:	EntityN, on page 346 EntityFieldN, on page 344

DocumentDelimiterCSVs

DEPRECATED: The `DocumentDelimiterCSVs` parameter is deprecated in Eduction version 12.5.0 and later.

This parameter is still available for existing implementations, but it might be incompatible with new functionality. The parameter might be deleted in future.

The fields in an XML file that mark the start and end of an IDOL document. You must have only one document level for each XML schema.

When identifying fields use the formats:

- `FieldName` to match root-level fields.
- `*/FieldName` to match all fields except root-level.
- `Path/FieldName` to match fields that the specified path points to.

Type:	String
Default:	*/DOCUMENT
Required:	No
Configuration Section:	**Server**
Example:	`DocumentDelimiterCSVs=*/DOCUMENT,*/SPEECH` In this example, the beginning and end of individual documents in a file is marked by opening and closing `DOCUMENT` and `SPEECH` tags.
See Also:	

EnableComponents

Set this parameter to **`False`** to return only the entity. Set it to **`True`** to return the entity and all the components of the entity.

This parameter requires `OutputSimpleMatchInfo` to be set to **`False`**.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	**Eduction**
Example:	`EnableComponents=True`
See Also:	OutputSimpleMatchInfo, on page 361

EnableUniqueMatches

Set `EnableUniqueMatches` to **`True`** to return only a single occurrence of a particular value. In this case, two `Entity`*`N`* definitions cannot return the same value, even if they use different patterns. If the same value occurs more than once, Eduction returns only the first instance, even if the matches occur for different entities.

By default, Eduction displays duplicates unless you set `EnableUniqueMatches` to **`True`** to explicitly remove them.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	**Eduction**
Example:	`EnableUniqueMatches=True`
See Also:	

Entities

A list of entities that you want to modify using the post-processing script. If you do not set this parameter, you can use the script to modify the matches for every entity.

You can separate multiple entities with a comma, and you can use wildcard expressions. The asterisk (*) wildcard matches any number of characters, and the question mark (?) wildcard matches a single character.

For example, set `Entities` to `phone/*` to apply the script to the `phone/landline/gb`, `phone/mobile/gb` entities and so on.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for an Eduction post-processing task

Example:	`PostProcessingTask0=EductionLuaPostProcessing`
	
	`[EductionLuaPostProcessing]`
	`Script=scripts/eduction_post_process.lua`
	`Entities=phone/landline/gb,phone/mobile/gb`

See Also:	Script, on page 369 ProcessEnMasse, on page 364 PostProcessingTaskN, on page 361

EntityAdvancedField*N*

A comma-separated list of advanced fields to return, associated with the entities specified by the EntityN, on page 346 parameter.

To use this option you must:

- set `OutputSimpleMatchInfo` to **`False`** for edktool.
- set `EnableComponents` to **`True`** for edktool.
- define components in the entity definition.

You configure `EntityAdvancedField`*`N`* in a similar way to EntityFieldN. You set `EntityAdvancedField`*`N`* to a comma-separated list of advanced fields that you want to return. The

value of the advanced field is the output of simple operations (`min`, `max`, `sum`, and `ave`) on the values of entity components.

For example, for the following configuration:

```
Entity0=testgrammar/testentity
EntityField0=FIELD0
EntityAdvancedField0=OfferPrice:max(price1 price2),BidPrice:min(price1 price2)
```

And the following data:

```
share price1 price2
Com1  165    167
Com2  1890   1880
```

An entity with the following pattern:

```
<grammar name="testgrammar">
<entity name="testentity" type="public">
<pattern>(?A=price1:\d+)\s+(?A=price2:\d+)</pattern>
</entity>
</grammar>
```

Returns the following results as fields:

```
#DREFIELD FIELD0="165 167"
#DREFIELD OfferPrice="167"
#DREFIELD BidPrice="165"
#DREFIELD FIELD0="1890 1880"
#DREFIELD OfferPrice="1890"
#DREFIELD BidPrice="1880"
```

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings

Example:	`Entity0=testgrammar/testentity`
	`EntityField0=FIELD0`
	`EntityAdvancedField0=OfferPrice:max(price1 price2),BidPrice:min(price1 price2)`

See Also:	EntityN, on page 346 EntityZoneN, on page 349

EntityComponentField*N*

A comma-separated list of entity components that you want to return as fields, associated with the entities specified by the EntityN, on page 346 parameter.

To use this option you must:

- set `OutputSimpleMatchInfo` to **`False`** for edktool.
- set `EnableComponents` to **`True`** for edktool.
- define components in the entity definition.

You configure `EntityComponentField*N*` in a similar way to EntityFieldN. Set `EntityComponentField*N*` to a comma-separated list of entity components to return as fields.

For example, for the following configuration:

```
Entity0=testgrammar/testentity
EntityField0=FIELD0
EntityComponentField0=Name,Age
```

And the following data:

```
name    age
geoff   45
jane    54
```

An entity with the following pattern:

```
<grammar name="testgrammar">
<entity name="testentity" type="public">
<pattern>name\s+age(\n(?A=Name:\w+)\s+(?A=Age:\d+)){1,}</pattern>
</entity>
</grammar>
```

Returns the following values as fields:

```
#DREFIELD Name="geoff"
#DREFIELD Age="45"
#DREFIELD Name="jane"
#DREFIELD Age="54"
```

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**
Example:	`Entity0=testgrammar/testentity`

	`EntityField0=FIELD0` `EntityComponentField0=Name,Age`
See Also:	EntityN, on page 346 EntityZoneN, on page 349

EntityField*N*

A comma-separated list of document fields to associate with the entities specified by the EntityN, on page 346 or CellEntityN, on page 337 parameter. When a document contains particular entities, Eduction saves the associated text in the fields that you specify in this parameter. The entity field number *N* must match the corresponding EntityN, on page 346 or CellEntityN, on page 337 number.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**

Example:	`Entity0=edk_common_entities/ss_number`
	`EntityField0=SOCIAL_SECURITY_NUMBER`
	
	`Entity1=edk_common_entities/postal_address`
	`EntityField1=SHIPPING_ADDRESS`

See Also:	EntityN, on page 346 CellEntityN, on page 337 EntitySearchFieldsN, on page 347 EntityZoneN, on page 349

EntityMatchLimit*N*

The maximum number of matches to return for the entities specified by the corresponding EntityN parameter. When Eduction reaches the limit it stops searching for those entities, so no further matches are returned and Eduction does not spend time looking for them.

If the `EntityN` parameter specifies multiple entities, using a wildcard or a comma-separated list, the limit applies separately to each entity. For example, the following configuration would permit up to three matches for British names, and up to three for French names, and so on:

```
Entity0=pii/name/*
EntityMatchLimit0=3
```

The limit is applied after post-processing, so any matches that are discarded by post-processing do not count towards the limit.

You might set this parameter if you want to see some matches for a given entity but would prefer to ignore further matches in favor of reducing the processing time.

Type:	Integer
Default:	No limit
Required:	No
Configuration Section:	**Eduction**

Example:	With the following configuration Eduction returns a maximum of three matches for the `pii/name/gb` entity.
	`Entity0=pii/name/gb`
	`EntityMatchLimit0=3`

See Also:	EntityN, on the next page MaxMatchesPerDoc, on page 357

EntityMatchRange*N*

A range of matching instances of the entity that you want to return. The entity match range number *N* must match the corresponding EntityN number.

Specify the matches to return as a comma separated list, for example `1,2,3,5,7`. You can also specify a range with a hyphen, for example `1-3,5,7`.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**

Example:	`Entity0=edk_common_entities/ss_number`
	`EntityMatchRange0=1-3,6,9-`
	This example specifies the first through third match for the `ss_number` entity, as well as the sixth match and all matches starting with the ninth.

See Also:	EntityN, on the next page

EntityMinScore*N*

The minimum score that a match for the corresponding EntityN, below must have for it to return. The lowest possible score is 0. The upper limit varies depending on the entity.

Set a higher minimum score to indicate that matches must meet a higher confidence level to return.

> **NOTE:** Eduction applies the minimum score threshold before it runs any post-processing tasks (seePost-Processing, on page 105). If your post-processing task reduces the score for a match to below the `EntityMinScore`*N* threshold, Eduction does not automatically discard the match.
>
> To filter matches after all post-processing tasks have completed, set PostProcessThreshold, on page 362

The *score* for an entity is defined by the author of the grammar and defaults to **1**. See the Eduction Grammar Syntax for a description of the score attribute.

The entity number (*N*) in `EntityMinScore`*N* must match the corresponding entity number in the `Entity`*N* entry.

Type:	Long
Default:	0 (returns all matches)
Required:	No
Configuration Section:	**Eduction**

Example:	To specify a minimum score of `0.5` for `Entity0`:
	`Entity0=edk_common_entities/ss_number`
	`EntityMinScore0=0.5`

See Also:	EntityN, below MinScore, on page 358 PostProcessThreshold, on page 362

Entity*N*

A comma-separated list of entities to extract. Entities are defined in the grammar file that you set in the ResourceFiles parameter. Replace *N* with the zero-based rank of the entity.

You must associate each entity with a field by setting EntityFieldN.

You cannot use the entity name `entities/ZoneStart`*N* or `entities/ZoneEnd`*N* (where *N* is a numeric value). These entity names are reserved for use by Eduction.

If you do not define an `EntityN` parameter, Eduction looks for all entities in all loaded grammar files. In this case, Eduction automatically generates the `EntityFieldN` settings from the entities found in grammar files, by converting the entity names to uppercase and replacing slashes with an underscore. For example, for the entity `edk_common_entities/place`, Eduction generates the entity field: `EDK_COMMON_ENTITIES_PLACE`.

If you want to use several entities, you can use wildcard expressions instead of typing a lengthy comma-separated list. For example:

```
Entity0=place/city1/*,place/city2/*
EntityField0=CITY
Entity1=place/*/spabo
EntityField1=BOLIVIAN_PLACE
```

You can use the * wildcard to match any number of characters, or the ? wildcard to match a single character.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**

Example:	`Entity0=edk_common_entities/ss_number`
	`EntityField0=SOCIAL_SECURITY_NUMBER`
	`EntityZone0=0`
	
	`Entity1=edk_common_entities/postal_address`
	`EntityField1=SHIPPING_ADDRESS`
	`EntityZone1=1`
	
	`ZoneStart0=Social Security:`
	`ZoneEnd0=Shipping Address`
	
	`ZoneStart1=Shipping Address:`
	`ZoneEnd1=Billing Address`

See Also:	EntityFieldN, on page 344 EntityMinScoreN, on the previous page EntitySearchFieldsN, below EntityZoneN, on page 349

EntitySearchFields*N*

The document fields to search for the corresponding EntityN, on the previous page entity.

Use this parameter if you want to search a different set of fields for this entity than the fields you set in SearchFields, on page 370. If you do not set `EntitySearchFields`*N*, Eduction searches the fields specified by SearchFields, on page 370.

Type:	String
Default:	The value of SearchFields, on page 370
Required:	No
Configuration Section:	**Eduction**

Example:	In the following example, Eduction returns matches for `Entity0` (`airport/icao`) only if they occur in the `STARTAIRPORT` or `DESTAIRPORT` fields.
	
	`EntitySearchFields`*N* is not set for `Entity4` (`place/state/engus`), so Eduction returns matches if they are present in the specified SearchFields, on page 370.
	
	`[Eduction]`
	`SearchFields=DRECONTENT`
	
	`Entity0=airport/icao`
	`EntityField0=AIRPORTCODE`
	`EntitySearchFields0=STARTAIRPORT,DESTAIRPORT`
	
	`Entity1=person/femalefirstname/engus`
	`EntityField1=FIRSTNAME`
	`EntitySearchFields1=PASSENGER_FIRSTNAME`
	
	`Entity2=person/malefirstname/engus`
	`EntityField2=FIRSTNAME`
	`EntitySearchFields2=PASSENGER_FIRSTNAME`
	
	`Entity3=person/lastname/engus`
	`EntityField3=SURNAME`
	`EntitySearchFields3=PASSENGER_SURNAME`
	
	`Entity4=place/state/engus`
	`EntityField4=STATE`

See Also:	EntityN, on page 346 EntityFieldN, on page 344 SearchFields, on page 370

EntityZone*N*

Associates an EntityN entity with one or more zones that you define by using ZoneStartN and ZoneEndN. Set `EntityZone`*`N`* the number of the ZoneStartN and ZoneEndN parameters to associate with the EntityN. Eduction searches for the entity in the specified zones. The entity zone number *N* must match the corresponding EntityN number.

Type:	Long
Default:	None
Required:	No
Configuration Section:	**Eduction**

Example:	`Entity0=edk_common_entities/ss_number`
	`EntityField0=SOCIAL_SECURITY_NUMBER`
	`EntityZone0=0`
	
	`Entity1=edk_common_entities/postal_address`
	`EntityField1=SHIPPING_ADDRESS`
	`EntityZone1=1`
	
	`ZoneStart0=Social Security:`
	`ZoneEnd0=Shipping Address`
	
	`ZoneStart1=Shipping Address:`
	`ZoneEnd1=Billing Address`

See Also:	ZoneEndN, on page 374 ZoneStartN, on page 375

Exclusion

Set `Exclusion` to **`True`** to specify that you want to discard any previous pre-filter windows that match the associated pre-filter Regex, on page 367 or ResourceFile, on page 368 parameter.

Eduction runs your pre-filter tasks in the configured order. You can use an `Exclusion` to exclude match windows that were found by a previous pre-filter. Eduction compares any candidate windows that have been found by an earlier task to the regex or dictionary for the exclusion task. It discards any windows that match the exclusion filter.

For example, if you create a pre-filter that finds numbers to find potential address matches, you can add a subsequent exclusion to remove numbers that are part of dates:

```
[Eduction]
PrefilterTask0=NumberWordPrefilter
PrefilterTask1=ExcludeDatesPrefilter
PrefilterTask2=StreetMarkersPrefilter

[NumberWordPrefilter]
Regex=\p{N}+,?\p{Z}+\p{Lu}\p{L}
WindowCharsBeforeMatch=100
WindowCharsAfterMatch=100

[ExcludeDatesPrefilter]
Regex=\b([1-9]|[12]\d|3[01]) (Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec)
Exclusion=True

[StreetMarkersPrefilter]
ResourceFile=prefilter/address_street_markers.dpf
WindowCharsBeforeMatch=100
WindowCharsAfterMatch=100
```

The `NumberWordPrefilter` task might find potential matches for dates such as *15 March*. The subsequent `ExcludeDatesPrefilter` removes windows that include these dates, so that they are not checked against the Eduction entities.

The third task attempts to find windows based on a dictionary of street markers. Because this task is configured after the exclusion filter, it finds input text such as *15 March Street* that had been removed by the exclusion.

> **NOTE:** The parameters that control the window size (WindowCharsAfterMatch, on page 373 and WindowCharsBeforeMatch, on page 374) are not relevant to pre-filters that have `Exclusion` set to **`True`**.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	***MyPreFilterTask***
Example:	`Exclusion=True`
See Also:	Regex, on page 367 ResourceFile, on page 368

HeaderEntityMatchLimit*N*

The maximum number of columns to match with the entities specified by the corresponding HeaderEntityN, on the next page parameter. After Eduction finds this number of columns with headers that match the header entities, it stops searching subsequent columns, so Eduction does not spend time looking for further matches.

You might set this parameter if you want to see some matches for a particular entity but would prefer to ignore further matches in favor of reducing the processing time.

If the HeaderEntityN, on the next page parameter specifies multiple entities, using a wildcard or a comma-separated list, the limit applies separately to each entity. For example, the following configuration would permit up to three matches for English date landmarks, and up to three for French date landmarks, and so on:

```
HeaderEntity0=pii/date/dob/landmark/eng,pii/date/dob/landmark/fre,pii/date/dob/landmark/spa
HeaderEntityMatchLimit0=3
```

Eduction applies the limit after post-processing, so any matches that are discarded by post-processing do not count towards the limit.

The limit applies only to matches in a single table. Eduction resets the limit when it encounters a table delimiter.

Type:	Integer
Default:	No limit
Required:	No
Configuration Section:	**Eduction**

Example:	With the following configuration Eduction finds a maximum of three columns that match the `pii/date/dob/landmark/all` entity.
	`HeaderEntity0=pii/date/dob/landmark/all`
	`HeaderEntityMatchLimit0=3`

See Also:	HeaderEntityN, on the next page CellEntityMatchLimitN, on page 336 MaxSearchHeaderRow, on page 358

HeaderEntity*N*

The entities to extract for the header rows of input tables. This parameter allows you to extract entities from structured data.

When matching CSV or TSV input, Eduction matches the first non-empty row of input against these configured header entities. You can use this to extract *landmark* values that describe something that you want to find in a table column. You specify the entity to extract from the other cells by setting CellEntityN, on page 337

> **TIP:** You can optionally configure Eduction to search additional rows for the header entities by setting MaxSearchHeaderRow, on page 358.

For example:

```
HeaderEntity0=pii/date/dob/landmark/all
CellEntity0=pii/date/nocontext/all
```

This example matches date of birth landmark values in the header, and for all subsequent rows in that column, it extracts any date values.

> **NOTE:** The IDOL PII Package, IDOL PHI Package, and IDOL PCI Package, provide landmark entities in most grammars. To extract entities from tables with the Eduction standard grammar files, you might need to create your own landmark entities.

You can specify multiple entities in a comma-separated list. If the table header matches any of the configured header entities, Eduction matches the cell content against any of the configured cell entities. This option might be useful if you want to match a particular entity in multiple languages, or if you want to include a custom entity in addition to a standard one.

You can also use wildcard expressions in the entity names. The * wildcard matches any number of characters, and the ? wildcard matches a single character.

For more information about table extraction, see Extract Entities from Tables, on page 61.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**
Example:	`HeaderEntity0=pii/date/dob/landmark/all` `CellEntity0=pii/date/nocontext/all`
See Also:	HeaderEntityMatchLimitN, on the previous page CellEntityN, on page 337

	MaxSearchHeaderRow, on page 358

InvalidRegexAfterMatch

A regular expression that cannot form part of the valid match for a pre-filter.

When Eduction runs a pre-filter task, it finds a potential match and creates a match window by selecting characters before and after the potential match (you configure the size of the match window by using WindowCharsAfterMatch, on page 373 and WindowCharsBeforeMatch, on page 374). You can use `InvalidRegexAfterMatch` to specify a pattern that definitely cannot form part of a valid match. When Eduction finds something that matches this non-valid pattern in the match window after the potential match, it truncates the match window to exclude the non-valid pattern and any text after it.

For example, if you know that your pattern can only include space characters, letters, and numbers, you can use the following setting to truncate the match window if Eduction finds anything else:

```
InvalidRegexAfterMatch=[^ \p{L}\p{N}]
```

Type:	String
Default:	
Required:	No
Configuration Section:	***MyPreFilterTask***
Example:	`InvalidRegexAfterMatch=[^ \p{L}\p{N}]`
See Also:	Regex, on page 367 ResourceFile, on page 368 WindowCharsAfterMatch, on page 373 WindowCharsBeforeMatch, on page 374

InvalidRegexBeforeMatch

A regular expression that cannot form part of the valid match for a pre-filter.

When Eduction runs a pre-filter task, it finds a potential match and creates a match window by selecting characters before and after the potential match (you configure the size of the match window by using WindowCharsAfterMatch, on page 373 and WindowCharsBeforeMatch, on page 374). You can use `InvalidRegexBeforeMatch` to specify a pattern that definitely cannot form part of a valid match. When Eduction finds something that matches this non-valid pattern in the match window

before the potential match, it truncates the match window to exclude the non-valid pattern and any text before it.

For example, if you know that your pattern can only include space characters, letters, and numbers, you can use the following setting to truncate the match window if Eduction finds anything else:

```
InvalidRegexBeforeMatch=[^ \p{L}\p{N}]
```

Type:	String
Default:	
Required:	No
Configuration Section:	***MyPreFilterTask***
Example:	`InvalidRegexBeforeMatch=[^ \p{L}\p{N}]`
See Also:	Regex, on page 367 ResourceFile, on page 368 WindowCharsAfterMatch, on page 373 WindowCharsBeforeMatch, on page 374

LanguageDirectory

The path of an IDOL Server language directory that contains the relevant sentence breaking libraries and associated data files.

This parameter enables tokenization of Chinese, Japanese, Korean, and Thai languages.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**
Example:	`C:\Program Files\IDOLServer\IDOL\langfiles`
See Also:	Locale, below

Locale

The locale to use to tokenize the input text.

By default, Eduction uses an English tokenizer. You can set `Locale` to one of the following values to enable tokenization for Chinese, Japanese, Korean, and Thai:

- **CHI**
- **JPN**
- **KOR**
- **THA**

NOTE: The Eduction standard grammar files are developed without this setting. Micro Focus recommends that you use this parameter only when you are using custom grammar files that have been developed with a specific tokenization.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**
Example:	`Locale=THA`
See Also:	LanguageDirectory, on the previous page

MatchCase

Set this parameter to **`False`** to ignore case when matching characters.

By default, Eduction is case sensitive when matching characters, which improves the performance of matching. You can use `MatchCase` to turn off all case sensitivity in matching.

You can also use case attributes in your grammars to turn off case sensitivity for particular entities. See Case Sensitive Matches, on page 57.

Type:	Boolean
Default:	True
Required:	No
Configuration Section:	**Eduction**
Example:	`MatchCase=False`
See Also:	

MatchTimeout

The maximum time to spend searching for matches (to all chosen entities) at a particular offset. You can specify the timeout in seconds and fractional seconds (for example 5.5 means five and a half seconds).

After it reaches the timeout, Eduction returns the best match it has found (if any) and continues looking for matches later in the text.

Eduction usually finds matches very quickly, so in most cases you do not need to set this timeout.

> **NOTE:** When edktool reports settings, it returns values for the timeout in milliseconds. For example, if you set `MatchTimeout` to 4.5, it returns the timeout value as 4500.

Type:	Integer
Default:	60
Required:	No
Configuration Section:	**Eduction**
Example:	`MatchTimeout=30`
See Also:	RequestTimeout, on page 367

MatchWholeWord

Set `MatchWholeWord` to **`True`** to match only terms in the text that begin and end on a whole word boundary.

Set `MatchWholeWord` to **`False`** to match terms that start and end anywhere, including in the middle of a word in the text.

For example, when `MatchWholeWord=True`, a search for the term `80` does not find a match in the text string `80mph`. When `MatchWholeWord=False`, a search for the term `par` finds a match in the text string `separated`.

For more information on modifying the matching behavior by using `MatchWholeWord`, see Match Special Characters, on page 59.

Type:	Boolean
Default:	True
Required:	No

Configuration Section:	**Eduction**
Example:	`MatchWholeWord=False`
See Also:	TangibleCharacters, on page 371 TokenWithPunctuation, on page 372

MaxEntityLength

The maximum number of characters in a returned entry.

Reducing this number can assist performance by preventing Eduction from scanning a long string of text for an entity that is expected to be small.

Type:	Integer
Default:	256
Required:	No
Configuration Section:	**Eduction**
Example:	`MaxEntityLength=100`
See Also:	EntityN, on page 346 ZoneEndN, on page 374 ZoneStartN, on page 375

MaxMatchesPerDoc

The maximum number of matches to allow in each document.

Type:	Integer
Default:	Unlimited
Required:	No
Configuration Section:	**Eduction**
Example:	`MaxMatchesPerDoc=15`
See Also:	

MaxSearchHeaderRow

In table mode, `MaxSearchHeaderRow` is the maximum number of rows to search a table for the HeaderEntityN, on page 352 entities.

By default, Eduction searches only the first row of a table for the configured header entities. Set `MaxSearchHeaderRow` to a higher number of rows to allow Eduction to search additional rows. It then searches for up to the `MaxSearchHeaderRow` number of rows, until it finds a header match, and then stops. For example, if you set `MaxSearchHeaderRow` to 5, and Eduction finds a header match on the third row, it stops searching.

This method might be useful if your tables sometimes contain irrelevant information in the first few rows.

> **NOTE:** Only rows with content count towards the row count; Eduction does not count any empty rows.

You can also set `MaxSearchHeaderRow` to **-1** (unlimited) to search the whole table to find a header match.

For more information about table extraction, see Extract Entities from Tables, on page 61.

Type:	Integer
Default:	1
Required:	No
Configuration Section:	**Eduction**
Example:	`MaxSearchHeaderRow=5`
See Also:	CellEntityN, on page 337 HeaderEntityN, on page 352

MinScore

The minimum score that a match must have for Eduction to return it. The lowest possible score is 0. The upper limit varies depending on the entity.

Set a higher minimum score to indicate that matches must meet a higher confidence level to return.

This parameter applies to all entities. You can also set EntityMinScoreN, on page 346, which applies to the entities specified by the corresponding EntityN, on page 346 parameter. If you set both parameters, a match must exceed both thresholds.

NOTE: Eduction applies this threshold before it runs any post-processing tasks (see Post-Processing, on page 105). To filter matches after all post-processing tasks have completed, set PostProcessThreshold, on page 362

NOTE: You can override the value of this parameter by using the associated Eduction SDK functions.

Type:	Long
Default:	0 (all matches are returned)
Required:	No
Configuration Section:	**Eduction**
Example:	`MinScore=0.5`
See Also:	EntityMinScoreN, on page 346

NonGreedyMatch

Set `NonGreedyMatch` to **`True`** to configure Eduction to return the shortest match that it finds. In this case, when Eduction finds two matches that start at the same word, from two different entities, Eduction returns only the shortest match.

Setting `NonGreedyMatch` to **`True`** implicitly turns off AllowOverlaps, on page 332 and AllowMultipleResults, on page 329. If you have set these parameters, `NonGreedyMatch` takes precedence.

For more information on how to configure the Eduction matching behavior using `NonGreedyMatch`, see Select Matches, on page 65.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	**Eduction**
Example:	`NonGreedyMatch=True`
See Also:	AllowMultipleResults, on page 329 AllowOverlaps, on page 332

NumTasks

DEPRECATED: The `NumTasks` parameter is deprecated for Eduction version 12.5.0 and later. Use the PostProcessingTaskN, on the next page parameter to define your post-processing tasks. In this case, Eduction automatically calculates the number of tasks.

The `NumTasks` parameter is still available for existing implementations, but it might be incompatible with new functionality. The parameter might be deleted in future.

The number of post-processing tasks that you want to configure.

See Post-Processing, on page 105 for more information.

Type:	Integer
Default:	None
Required:	No
Configuration Section:	**[PostProcessingTasks]**
Example:	`NumTasks=1`
See Also:	TaskN, on page 371

OutputScores

DEPRECATED: The `OutputScores` parameter is deprecated in Eduction version 12.5.0 and later.

This parameter is still available for existing implementations, but it might be incompatible with new functionality. The parameter might be deleted in future.

Set `OutputScores` to **`True`** to include the score associated with a match in the output from an extraction task. If the output is in IDX format, Eduction adds the score as a new `DREFIELD`, with the field name `SCORE`. If the output is in XML format, it adds the score as an attribute with the name `"score"`.

NOTE: This parameter applies to `edktool` only.

Type:	Boolean
Default:	True

Required:	No
Configuration Section:	**Eduction**
Example:	`OutputScores=True`
See Also:	

OutputSimpleMatchInfo

> **DEPRECATED:** The `OutputSimpleMatchInfo` parameter is deprecated in Eduction version 12.5.0 and later.
>
> This parameter is still available for existing implementations, but it might be incompatible with new functionality. The parameter might be deleted in future.

Set `OutputSimpleMatchInfo` to **`True`** to generate basic match information only, such as document, entity, position, and original text. To use this option, you must use `edktool` with both the extract option and the option to generate a list of matches.

If `OutputSimpleMatchInfo=True`, EnableComponents, on page 340 has no effect and reverts to **`False`**.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	**Eduction**
Example:	`OutputSimpleMatchInfo=False`
See Also:	EnableComponents, on page 340

PostProcessingTask*N*

The name of an Eduction post-processing task to run.

This parameter specifies the name of a section in the Eduction configuration file that contains the parameters required to run the task. To run multiple tasks, use numbered parameters (`PostProcessingTask0`,`PostProcessingTask1`, and so on).

You can use a post-processing task to modify the output from Eduction, or format the output to meet your requirements.

For more information, see Post-Processing, on page 105.

Type:	String
Default:	
Required:	No
Configuration Section:	**Eduction**

Example:	`PostProcessingTask0=EductionLuaPostProcessing`
	
	`[EductionLuaPostProcessing]`
	`Script=scripts/eduction_post_process.lua`

See Also:	Script, on page 369 Entities, on page 341 ProcessEnMasse, on page 364

PostProcessThreshold

DEPRECATED: The `[PostProcessingTasks]` configuration section is deprecated for Eduction version 12.5 and later. Set `PostProcessThreshold` in the `[Eduction]` section.

The `[PostProcessingTasks]` section is still available for existing implementations, but it might be incompatible with new functionality. The parameter might be deleted in future.

The minimum score that a match must have, after all post-processing tasks have completed, for Eduction to return that match.

The threshold applies to all entities. Specify a higher threshold to indicate that matches must meet a higher confidence level to return.

This parameter is similar to MinScore, on page 358. Use `PostProcessThreshold` when you want to filter the matches after all post-processing tasks have completed, rather than before post-processing begins.

NOTE: You can override the value of this parameter by using the associated Eduction SDK functions.

Type:	Number
Default:	All matches are returned
Required:	No
Configuration	**Eduction**

Section:	
Example:	`PostProcessThreshold=0.4`
See Also:	MinScore, on page 358

PreFilterMaxReturnedBytes

The maximum number of bytes that a pre-filter task can produce without a resulting Eduction match.

The pre-filter process generates windows with potential matches, based on your pre-filter criteria. Eduction runs matching on these windows as soon as the pre-filter generates them. `PreFilterMaxReturnedBytes` allows you to specify a threshold for the amount of data that the pre-filter can output without a subsequent match. This option allows you to more quickly reject documents that match the pre-filter but are unlikely to contain real matches.

When a match is found in a pre-filter window the counter is reset and another match must occur before the specified limit is reached, otherwise Eduction stops processing.

To turn off this pre-filter threshold, set `PreFilterMaxReturnedBytes` to **0** (unlimited).

> **NOTE:** If you use Eduction SDK functions to set a pre-filter byte threshold, the SDK functions override the value of the `PreFilterMaxReturnedBytes` configuration parameter.

Type:	Integer
Default:	0 (unlimited)
Required:	No
Configuration Section:	**Eduction**
Example:	`PreFilterMaxReturnedBytes=1000000`
See Also:	PreFilterTaskN, below

PreFilterTask*N*

The name of an Eduction pre-filtering task to run. This parameter specifies the name of a section in the Eduction configuration file that contains the parameters required to run the task. Replace the value *N* with the number of the task, starting from zero (`PreFilterTask0`, `PreFilterTask1`, and so on)

Pre-filtering tasks use a regular expression to perform an initial check to find strings that might match an entity in your input text. It then restricts Eduction matching to windows of several characters

around these potential matches. This process can improve performance for some grammars, by restricting the amount of the input text that Eduction checks in detail for matches.

In the pre-filter configuration sections, you set Regex, on page 367 to a regular expression that finds potential matches. You use WindowCharsAfterMatch, on page 373 and WindowCharsBeforeMatch, on page 374 to specify the size of the matching window to use.

Eduction runs all your configured pre-filtering tasks for all input text, so ensure that your pre-filter task applies to all your configured grammars and entities. Use a different configuration for any entities that you do not want to pre-filter.

When you configure multiple pre-filtering tasks, Eduction runs each of them on your full input text. It then merges any windows that overlap from different tasks, and uses all the resulting windows as input text for the full matching process.

For more information, see Pre-Filter Tasks, on page 101.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**

Example:	`[Eduction]`
	`PreFilterTask0=AddressPreFilter`
	
	`[AddressPreFilter]`
	`Regex=\d{1,7}`
	`WindowCharsAfterMatch=100`
	`WindowCharsBeforeMatch=100`

See Also:	Regex, on page 367 ResourceFile, on page 368 WindowCharsAfterMatch, on page 373 WindowCharsBeforeMatch, on page 374

ProcessEnMasse

Set `ProcessEnMasse` to **`True`** to use the entire set of educed matches as input for your post-processing script, rather than a single match. This option enables your script to look at all the matches at once and modify them accordingly.

For example, to increase the score of a match if it is found near other matches, you must consider all of the matches together.

By default, Eduction sends an individual match to the post-processing script.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for an Eduction post-processing task.

Example:	`PostProcessingTask0=EductionLuaPostProcessing`
	
	`[EductionLuaPostProcessing]`
	`Script=scripts/eduction_post_process.lua`
	`ProcessEnMasse=True`

See Also:	Entities, on page 341 Script, on page 369 PostProcessingTaskN, on page 361

RedactedOutput

Set `RedactedOutput` to **`True`** to redact sensitive information in the output text.

You can also set **one** of RedactionOutputString, below or RedactionReplacementCharacter, on the next page. If you do not set either parameter, the default behavior is to replace redacted text with `[redacted]` in the output. If you configure both, RedactionReplacementCharacter, on the next page takes precedence.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	**Eduction**
Example:	`RedactedOutput=False`
See Also:	RedactionOutputString, below RedactionReplacementCharacter, on the next page

RedactionOutputString

A string that replaces redacted information in the output text.

Type:	String
Default:	[redacted]
Required:	No
Configuration Section:	**Eduction**
Example:	`RedactionOutputString=[censored]`
See Also:	RedactionReplacementCharacter, below

RedactionReplacementCharacter

A single character that replaces each character in redacted text.

Type:	String
Default:	Use [redacted] for the whole redacted string
Required:	No
Configuration Section:	**Eduction**
Example:	`RedactionReplacementCharacter=*`
See Also:	RedactionOutputString, on the previous page

RedactionType

The method to use for replacing text when you use redaction. You can use one of the following options:

- **`Normalize`**. Replace a match with the normalized form of the text. For example, you could replace a match *ten to nine twenty third of June* with the normalized version *8:50 23rd of June*.
- **`Replace`**. Replace a match with a censored string or replacement character.
- **`Entity`**. Replace the match with the name of the matching entity.

NOTE: If you set AllowOverlaps, on page 332 to **`True`**, you cannot set `RedactionType` to **`Normalize`**, because Eduction Server cannot normalize overlapping phrases.

Type:	String

Default:	`Replace`
Required:	No
Configuration Section:	**Eduction**
Example:	`RedactionType=Normalize`
See Also:	RedactionReplacementCharacter, on the previous page RedactionOutputString, on page 365

Regex

A regular expression to use to find potential matches in your input text.

Eduction uses the configured `Regex` to run an initial check to find potential matches in your input text. It then creates a match window by selecting characters before and after the potential match, and uses your grammar to find actual matches in these windows.

Set this parameter to a regular expression that broadly matches everything that might be a match for your intended entity. For example, to match telephone numbers or addresses, you might find any string of digits. This narrows down the amount of text that Eduction must match against your entity, while ensuring that it does not miss potential matches.

You can use WindowCharsAfterMatch, on page 373 and WindowCharsBeforeMatch, on page 374 to determine the size of the matching window to use.

Type:	String
Default:	
Required:	No
Configuration Section:	***MyPreFilterTask***
Example:	`Regex=\d{1,7}`
See Also:	WindowCharsAfterMatch, on page 373 WindowCharsBeforeMatch, on page 374

RequestTimeout

The maximum time to spend processing a single input file or document. You can specify the timeout in seconds and fractional seconds (for example 5.5 means five and a half seconds).

After it reaches the timeout, Eduction stops processing and returns any results that were found. In most cases Eduction does not reach the default timeout, but it can prevent Eduction running for a long time with abnormal documents.

NOTE: When edktool reports settings, it returns values for the timeout in milliseconds. For example, if you set `RequestTimeout` to 4.5, it returns the timeout value as 4500.

Type:	Integer
Default:	300
Required:	No
Configuration Section:	**Eduction**
Example:	`RequestTimeout=120`
See Also:	MatchTimeout, on page 356

ResourceFile

The full path to a JSON file that contains a dictionary of terms to use to find potential matches in your input text.

Eduction uses the terms in the configured file to run an initial check to find potential matches in your input text. It then creates a match window by selecting characters before and after the potential match, and uses your grammar to find actual matches in these windows.

You can use a provided dictionary file, which is in a binary format with the file extension DPF, or you can create a custom JSON file with your own dictionary. For details of the format of the JSON file, see Dictionary ResourceFile Format, on page 102.

You can use WindowCharsAfterMatch, on page 373 and WindowCharsBeforeMatch, on page 374 to determine the size of the matching window to use.

Type:	String
Default:	
Required:	No
Configuration Section:	***MyPreFilterTask***
Example:	`ResourceFile=custom_dictionary.json`
See Also:	WindowCharsAfterMatch, on page 373 WindowCharsBeforeMatch, on page 374

ResourceFiles

The full path to a compiled ECR file that contains Eduction grammar entries. You must specify at least one resource file.

You can specify multiple resource files by separating them with commas, or by using wildcard expressions. You can use the * wildcard to match any number of characters, or the ? wildcard to match a single character. For example, set `ResourceFiles` to `<grammar_files_directory>/sentiment_*.ecr` to use all available sentiment grammars.

Type:	String
Default:	None
Required:	Yes
Configuration Section:	**Eduction**
Example:	`ResourceFiles=C:\MyGrammar\gram1.ecr,C:\MyGrammar\gram2.ecr`
See Also:	

Script

The path to the Lua post-processing script to run to process the data that the Eduction engine returns. See Post-Processing, on page 105 for more information.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for an Eduction post-processing task.
Example:	`Script=./scripts/checksum.lua`
See Also:	Entities, on page 341 ProcessEnMasse, on page 364 PostProcessingTaskN, on page 361

SearchFields

> **DEPRECATED:** The `SearchFields` parameter is deprecated in Eduction version 12.6.0 and later.
>
> This parameter is still available for existing implementations, but it might be incompatible with new functionality. The parameter might be deleted in future.

A comma-separated list of fields to search for entities, for example `DRECONTENT` or `DRETITLE`.

To search for a specific entity only in specific fields, you can set EntitySearchFieldsN, on page 347, which overrides the value of this parameter for specific entities.

When you select multiple fields in a document for parsing, Eduction returns matches in the following field order:

- `DREREFERENCE`
- `DRETITLE`
- `DRECONTENT`
- Any remaining fields in the order in which they are specified.

Type:	String
Default:	`DRETITLE,SUMMARY,DRECONTENT`
Required:	No
Configuration Section:	**Eduction**
Example:	`SearchFields=DRECONTENT,DRETITLE`
See Also:	EntityN, on page 346 EntitySearchFieldsN, on page 347

SuppressMatchLogging

Set `SuppressMatchLogging` to **`True`** to suppress log entries for every entity and zone pattern found in a document.

When you have set logging to **`Full`** in the Eduction configuration file, Eduction records a log entry for every entity and zone pattern that it finds in a document. You can set `SuppressMatchLogging` to **`True`** to suppress these log entries.

This option is useful when you want to log the performance timing information, but do not want the verbose match entries.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Eduction
Example:	`SuppressMatchLogging=True`
See Also:	

TangibleCharacters

A list of punctuation characters to treat as part of the word, rather than as word boundaries. By default, Eduction treats almost all punctuation characters as word boundaries.

> **NOTE:** You cannot specify spaces, returns, and tabs as `TangibleCharacters`.

This parameter has no effect when MatchWholeWord is set to **`False`**.

For more information on using `TangibleCharacters` to specify punctuation characters to match, or to match punctuation at the start of a match, see Match Special Characters, on page 59.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**
Example:	`TangibleCharacters=-/\@`
See Also:	MatchWholeWord, on page 356 TokenWithPunctuation, on the next page

Task*N*

> **DEPRECATED:** The `Task`*`N`* parameter is deprecated for Eduction version 12.5.0 and later. Use the PostProcessingTaskN, on page 361 parameter in the `[Eduction]` section.
>
> The `Task`*`N`* parameter is still available for existing implementations, but it might be incompatible with new functionality. The parameter might be deleted in future.

The name of an Eduction post-processing task to run. This parameter specifies the name of a section in the Eduction configuration file that contains the parameters required to run the task. To run multiple tasks, use numbered parameters (`Task0`,`Task1`, and so on).

You can use a post-processing task to modify the output from the Eduction module, or format the output to meet your requirements. For more information, see Post-Processing, on page 105.

Type:	String
Default:	None
Required:	No
Configuration Section:	**PostProcessingTasks**

Example:	`Task0=EductionLuaPostProcessing`
	
	`[EductionLuaPostProcessing]`
	`Script=scripts/eduction_post_process.lua`

See Also:	Script, on page 369 Entities, on page 341 ProcessEnMasse, on page 364 NumTasks, on page 360

TokenWithPunctuation

Set `TokenWithPunctuation` to **`True`** to treat all punctuation characters as part of a word token, rather than treating them as word boundaries. This option is equivalent to setting TangibleCharacters, on the previous page to all punctuation characters.

This parameter has no effect when MatchWholeWord, on page 356 is set to **`False`**.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	**Eduction**
Example:	`TokenWithPunctuation=True`
See Also:	MatchWholeWord, on page 356 TangibleCharacters, on the previous page

For more information on using `TokenWithPunctuation` to configure all punctuation marks as tangible characters, see Match Special Characters, on page 59.

ValidatePostProcessingTasks

Set `ValidatePostProcessingTasks` to **`False`** to skip post-processing entity validation when an Eduction engine starts.

By default, when Eduction starts a new engine, it checks that the configured entities are present in the loaded grammar files. You might want to skip this step when you have a stable configuration, and speed is critical.

Type:	Boolean
Default:	True
Required:	No
Configuration Section:	**Eduction**
Example:	`ValidatePostProcessingTasks=False`
See Also:	PostProcessingTaskN, on page 361

WindowCharsAfterMatch

The minimum number of UTF-8 characters to include in the matching window when Eduction finds a potential match.

Eduction uses the configured Regex, on page 367 to run an initial check to find potential matches in your input text. It then creates a match window by selecting characters before and after the potential match, and then uses your grammar to find actual matches in these windows.

You configure the window size by using WindowCharsBeforeMatch, on the next page and `WindowCharsAfterMatch`. Eduction sets the end of a match window at the closest word boundary after the configured number of characters.

Type:	Long
Default:	None
Required:	No
Configuration Section:	***MyPreFilterTask***
Example:	`WindowCharsAfterMatch=100`
See Also:	TokenWithPunctuation, on the previous page

WindowCharsBeforeMatch

The number of UTF-8 characters to include in the matching window when Eduction finds a potential match.

Eduction uses the configured Regex, on page 367 to run an initial check to find potential matches in your input text. It then creates a match window by selecting characters before and after the potential match, and then uses your grammar to find actual matches in these windows.

You configure the window size by using WindowCharsAfterMatch, on the previous page and `WindowCharsBeforeMatch`. Eduction sets the start of a match window at the closest word boundary before the configured number of characters.

Type:	Long
Default:	None
Required:	No
Configuration Section:	***MyPreFilterTask***
Example:	`WindowCharsBeforeMatch=100`
See Also:	TokenWithPunctuation, on page 372

ZoneEnd*N*

A regular expression that defines the end point of a zone.

A zone is a section of a field defined by a start and end pattern. Zones locate entities in parts of a field. If you do not add zone entries, Eduction searches the entire field. If you do not set the end pattern, the search begins at a match for the start pattern and continues until the end of the field.

Use EntityZoneN to associate an entity identified in an EntityN parameter with one or more zones that you define by usingZoneStartN and `ZoneEnd`*`N`*.

NOTE: Your start and end patterns must not match the same text in a field.

Type:	String
Default:	None
Required:	No
Configuration	**Eduction**

Section:	

Example:	`ZoneStart0=Social Security:`
	`ZoneEnd0=Shipping Address`
	
	`ZoneStart1=Shipping Address:`
	`ZoneEnd1=Billing Address`

See Also:	EntityN, on page 346 EntityZoneN, on page 349 ZoneStartN, below

ZoneStart*N*

A regular expression that defines the start point of a zone.

A zone is a section of a field defined by a start and end pattern. Zones locate entities in parts of a field. If you do not add zone entries, Eduction searches the entire field. If you do not set the start pattern, the search begins at the start of the field and continues until a match for the end pattern.

Use EntityZoneN to associate an entity identified in an EntityN parameter with one or more zones that you define by using `ZoneStart`*`N`* and ZoneEndN.

NOTE: Your start and end patterns must not match the same text in a field.

Type:	String
Default:	None
Required:	No
Configuration Section:	**Eduction**

Example:	`ZoneStart0=Social Security:`
	`ZoneEnd0=Shipping Address`
	
	`ZoneStart1=Shipping Address:`
	`ZoneEnd1=Billing Address`

See Also:	EntityN, on page 346 EntityZoneN, on page 349 ZoneEndN, on the previous page

Chapter 17: Eduction Lua Methods Reference

This section describes the methods you can use in your Lua post-processing scripts.

> **TIP:** The Eduction SDK allows you to use all standard IDOL Lua functions and methods. Most of these functions and methods are not relevant to Eduction, but in some cases you might find the general functions useful.
>
> For details of the available methods, refer to the *IDOL NiFi Ingest Getting Started Guide*.

edkComponent Methods

The following methods are available on `edkComponent` objects.

You can obtain an `edkComponent` object using the getComponent method of an edkmatch object.

Method	Description
getMatchedOffset	Returns the offset of a component in the matched text, in bytes.
getMatchedOffsetLength	Returns the offset of a component in the matched text, in Unicode characters.
getMatchedText	Returns the text that a component originally matched.
getMatchLength	Returns the size of the component matched text, in Unicode characters
getMatchSize	Returns the size of the component matched text, in bytes.
getName	Returns the name of a component.
getOffset	Returns the offset of a component in the normalized text, in bytes.
getOffsetLength	Returns the offset of a component in the normalized text, in Unicode characters.

Method	Description
getText	Returns the text that a component matches.
setMatchLength	Modifies the size of the component matched text, in Unicode characters
setMatchSize	Modifies the size of the component matched text, in bytes.
setName	Modifies the name of a component.
setText	Modifies the text that a component matches.

getMatchedOffset

Provides information on where in a document a particular component is found.

Syntax

```
edkcomponent:getMatchedOffset()
```

Returns

The position of the component, in bytes from the beginning of the original match text buffer.

getMatchedOffsetLength

Provides information on where in a document a particular component is found.

Syntax

```
edkcomponent:getMatchedOffsetLength()
```

Returns

The position of the component, in characters from the beginning of the original match text buffer.

getMatchedText

Returns the input text for a particular component, that is, the text **before** any normalization or modification that occurs as part of the extraction process.

Syntax

```
edkcomponent:getMatchedText()
```

Returns

The input text for the component match.

getMatchLength

Provides information on the size of a particular component match.

Syntax

```
edkcomponent:getMatchLength()
```

Returns

The size of the component matched text, in characters.

getMatchSize

Provides information on the size of a particular component match.

Syntax

```
edkcomponent:getMatchSize()
```

Returns

The size of the component matched text, in bytes.

getName

Retrieves the name of a component.

Syntax

```
edkcomponent:getName()
```

Returns

The component name. You can use setName to edit the component name.

getOffset

Provides information on where in a document a particular component is found.

Syntax

```
edkcomponent:getOffset()
```

Returns

The position of the component, in bytes from the beginning of the normalized match text buffer.

getOffsetLength

Provides information on where in a document a particular component is found.

Syntax

```
edkcomponent:getOffsetLength()
```

Returns

The position of the component, in characters from the beginning of the normalized match text buffer.

getText

Returns the normalized output text matched by a particular component.

Syntax

```
edkcomponent:getText()
```

Returns

The matched text for a specified component. You can use setText to edit the text.

setMatchLength

Modifies the size of a component match.

Syntax

```
edkcomponent:getMatchLength(new_length)
```

Arguments

Argument	Description
`new_length`	The new length of the match, in characters.

Returns

The new length of the component matched text, in characters.

Related Topics

- getMatchLength, on page 379

setMatchSize

Modifies the size of a component match.

Syntax

```
edkcomponent:setMatchLength(new_size)
```

Arguments

Argument	Description
`new_size`	The new size of the match, in bytes.

Returns

The new size of the component matched text, in bytes.

Related Topics

- getMatchSize, on page 379

setName

Edits the name of the component that you retrieved with getName.

Syntax

```
edkcomponent:setName(new_name)
```

Arguments

Argument	Description
`new_name`	The new name for the component.

Returns

The new component name.

setText

Modifies the normalized text matched by a particular component that you retrieved with getText.

Syntax

```
edkcomponent:setText(new_text)
```

Arguments

Argument	Description
`new_text`	The new matched text for the component.

Returns

The new matched text for the specified component.

edkEnMasseMatch Methods

An `edkEnMasseMatch` object represents a match that is being processed in an en-masse post-processing task. You cannot manipulate the match directly; instead you call the getMatch method to obtain an edkmatch object.

The following methods are available on `edkEnMasseMatch` objects.

Method	Description
getMatch	Returns an edkmatch object that represents the match.
getOutput	Returns whether the match is included in the results.
setOutput	Sets whether to include the match in the results.

getMatch

Returns an edkmatch object that you can use to manipulate the match.

Syntax

```
edkEnMasseMatch:getMatch()
```

Returns

An edkmatch object.

Example

The following example Lua script uses the `getMatch` method to obtain an `edkmatch` object:

```
function processmatches(matches)
    -- example that discards matches with score < 0.5
    for k,v in ipairs (matches) do
        local edkmatch = v:getMatch()
        if edkmatch:getScore() < 0.5 then
            v:setOutput(false)
        end
    end
end
```

getOutput

Returns whether the match is included in the results.

Syntax

```
edkEnMasseMatch:getOutput()
```

Returns

Boolean. `True` if the match is going to be included in the results, or `false` if it is going to be discarded.

setOutput

Sets whether to include the match in the results.

Syntax

```
edkEnMasseMatch:setOutput(output)
```

Arguments

Argument	Description
`output`	A Boolean value that specifies whether to include the match in the results (true) or discard it (false).

Example

The following script uses the `setOutput` method to discard matches with a score less than `0.5`:

```
function processmatches(matches)
    -- example that discards matches with score < 0.5
    for k,v in ipairs (matches) do
        local edkmatch = v:getMatch()
        if edkmatch:getScore() < 0.5 then
            v:setOutput(false)
        end
    end
end
```

edkMatch Methods

The following methods are available on `edkMatch` objects.

Constructor	Description
LuaEdkMatch:new, on the next page	Creates an `edkMatch` object.

Method	Description
addComponent	Adds a new component to the match.
getComponent	Returns a specific component.
getComponentCount	Returns the total number of components in a match.
getContribution	Returns a specific scoring contribution.
getContributionsCount	Returns the number of scoring contributions, that contributed to the score for the match.
getEntityName	Returns the name of an entity in a match.
getMatchedText	Returns the input text for a match.

Method	Description
getOffset	Returns the position of a match (in bytes).
getOffsetLength	Returns the position of a match (in characters).
getOutputText	Returns the output text for a match.
getScore	Returns the score of a match.
setEntityName	Edits an entity name in a match.
setMatchedText	Edits the input text for a match.
setOffset	Edits the position of a match (in bytes).
setOffsetLength	Edits the position of a match (in bytes).
setOutputText	Assigns a new value to the output text for a match.
setScore	Edits the score of a match.

LuaEdkMatch:new

The constructor for a `LuaEdkMatch` object (creates a new `LuaEdkMatch` object).

Syntax

```
LuaEdkMatch:new()
```

or

```
LuaEdkMatch:new(entity_name, matched_text, offset)
```

Arguments

> **NOTE:** You can use this constructor without arguments to create a match without values set. However, to reduce the amount of repeated code, Micro Focus recommends that you use the alternative version that allows you to create a match with some important information populated.

Argument	Description
`entity_name`	The name of the entity that the match belongs to.
`matched_text`	The matching text.
`offset`	The position of the match (in bytes).

Returns

(`edkMatch`). The new `edkMatch` object.

addComponent

Adds a new component to the match. For example, if your Eduction task returns an email address as a match, you can use `addComponent` to extract the text after the @ symbol and add it as a DOMAIN component for the match.

You can also use `addComponent` to add metadata from other sources. For example, if you have extracted a place name, you can add components called “LATITUDE” and “LONGITUDE”, and populate them with data from a different source, regardless of the fact that they were not components of the original text.

Syntax

```
edkmatch:addComponent(name, offset, offsetLength)
```

or

```
edkmatch:addComponent(name, offset, offsetLength, originalOffset,
originalOffsetLength)
```

Arguments

Argument	Description
`name`	The name of the new component (for example, TOPIC, or SENTIMENT)
`offset`	The position of the text to use as the new component, in bytes from the start of the normalized text of the match.
`offsetLength`	The position of the text to use as the new component, in characters from the start of the normalized text of the match.
`originalOffset`	(optional) The position of the text to use as the new component, in bytes from the start of the original text of the match. If you do not set `originalOffset`, the value is set to the same as offset.
`originalOffsetLength`	(optional) The position of the text to use as the new component, in characters from the start of the original text of the match. If you do not set `originalOffsetLength`, the value is set to the same as offset.

NOTE: If you are unsure of the correct `offset` or `offsetLength`, or the component value comes from an external source, you can set `offset` or `offsetLength` to **0**.

Returns

The new empty component object.

Related Topics

- getName, on page 379
- setName, on page 381
- getText, on page 380
- setText, on page 382

getComponent

The `getComponent` method returns a specified component object. The components are zero-indexed. For example, if you have six components, you can get the last component by using `edkmatch:getComponent(5)`.

Syntax

```
edkmatch:getComponent(index)
```

Arguments

Argument	Description
`index`	The number of the component to get (where the first component has the index zero).

Returns

The component object at the specified index position in the match.

Related Topics

- getName, on page 379
- setName, on page 381
- getText, on page 380
- setText, on page 382

getComponentCount

Returns the total number of components in a match.

Syntax

```
edkmatch:getComponentCount()
```

Returns

The number of components.

getContribution

Returns a specified scoring contribution.

The final score for a match (as retrieved through the getScore method) can be the product of multiple scoring contributions. For some entities the score is then normalized (for example so that it is always a value between 0 and 1).

Syntax

```
edkmatch:getContribution(index)
```

Arguments

Argument	Description
index	The number of the contribution to get (where the first contribution has the index zero).

Returns

The scoring contribution.

Example

The following example demonstrates how to obtain scoring contributions:

```
function processmatch(edkmatch)
    local contributionsCount = edkmatch:getContributionsCount()
    print ("Contributions count: ", contributionsCount)
    if contributionsCount >= 1 then
        for i=0, contributionsCount-1, 1 do
            local contribution = edkmatch:getContribution(i)
            print ("Contribution " .. i .. ": " , contribution)
        end
    end
    print ("Score: " , edkmatch:getScore())
    return true
end
```

This script produces output similar to:

```
Contributions count:    4.0
Contribution 0:         1.0
Contribution 1:         0.75
Contribution 2:         1.0
Contribution 3:         0.6
Score:  0.45
```

getContributionsCount

Returns the number of scoring contributions, that contributed to the score for the match.

Syntax

```
edkmatch:getContributionsCount()
```

Returns

The number of scoring contributions.

Example

For an example that demonstrates how to obtain scoring contributions, see the example for the getContribution method.

getEntityName

Gets an entity name from a match.

Syntax

```
edkmatch:getEntityName()
```

Returns

The name of the entity in a match. You can use setEntityName to edit the name.

getMatchedText

Returns the input text for a particular match, that is, the text **before** any normalization or modification that occurs as part of the extraction process.

Syntax

```
edkmatch:getMatchedText()
```

Returns

The input text for a match. You can use setMatchedText to edit the text.

Related Topics

- getOutputText, on the next page

getOffset

Provides information on where in a document a particular match is found.

Syntax

```
edkmatch:getOffset()
```

Returns

The position of the match, in bytes from the beginning of the text buffer. You can use setOffset to edit this information.

getOffsetLength

Provides information on where in a document a particular match is found.

Syntax

```
edkmatch:getOffsetLength()
```

Returns

The position of the match, in characters from the beginning of the text buffer. You can use setOffsetLength to edit this information.

getOutputText

Returns the output text for a match; that is, the text after any normalization or modification that takes place as part of the extraction process.

Syntax

```
edkmatch:getOutputText()
```

Returns

The output text of a match. You can use setOutputText to edit the text.

Related Topics

- getMatchedText, on the previous page

getScore

Retrieves the score for a match.

Syntax

```
edkmatch:getScore()
```

Returns

The score for the match. You can use setScore to edit the score.

setEntityName

Modifies the name of the entity that you retrieved by using getEntityName.

Syntax

```
edkmatch:setEntityName(new_name)
```

Arguments

Argument	Description
new_name	The new name for the entity in the match.

Returns

The new entity name.

setMatchedText

Modifies the input text that you retrieved by using getMatchedText.

The input text is the text before any normalization or modification that takes place as part of the extraction process. By contrast, setOutputText enables you to modify the output text after any changes.

Syntax

```
edkmatch:setMatchedText(new_text)
```

Arguments

Argument	Description
`new_text`	The new value that you want to assign to the input text.

Returns

The new input text.

setOffset

Modifies the position of a match in a document.

Syntax

```
edkmatch:setOffset(new_offset)
```

Arguments

Argument	Description
`new_offset`	The new position of the match, in bytes from the start of the text buffer.

Returns

The new position of the match (in bytes).

Related Topics

- getOffset, on page 390

setOffsetLength

Modifies the position of a match in a document.

Syntax

```
edkmatch:setOffsetLength(new_length)
```

Arguments

Argument	Description
`new_length`	The new position of the match, in characters from the start of the text buffer.

Returns

The new position of the match (in characters).

setOutputText

Modifies the output text that you retrieved by using getOutputText.

The output text is the text after any normalization or modification that takes place as part of the extraction process. By contrast, setMatchedText enables you to modify the input text before any changes are made.

Syntax

```
edkmatch:setOutputText(new_text)
```

Arguments

Argument	Description
new_text	The new value that you want to assign to the output text for a match.

Returns

The new output text.

setScore

Modifies the match score that you retrieved with getScore.

Syntax

```
edkmatch:setScore(new_score)
```

Arguments

Argument	Description
new_score	The new score for the match.

Returns

The new score for the match.

session Methods

The following methods are available on `session` objects.

Constructor	Description
injectMatch	Injects a match into the current matching session.

injectMatch

Injects a match into the current matching session.

This method accepts an `edkMatch` object. This `edkMatch` must be one that you create in the Lua script; nothing happens if you call `injectMatch` on the match from Eduction that you first pass into the `finalizematch` function.

After you call this function, the session takes ownership of the match, so you cannot use the created match in any subsequent functions.

> **NOTE:** You cannot perform additional post-processing on injected matches. Eduction skips these matches at post-processing time, to prevent infinite loops.

Syntax

```
session:injectMatch(edkMatch)
```

Arguments

Argument	Description
`edkMatch`	An `edkMatch` object.

Standard IDOL Lua Functions

The Eduction SDK allows you to use all standard IDOL Lua functions and methods. Most of these functions and methods are not relevant to Eduction, but in some cases you might find the general functions useful.

For details of the available methods, refer to the *IDOL NiFi Ingest Getting Started Guide*.

Glossary

A

ACI (Autonomy Content Infrastructure)
A technology layer that automates operations on unstructured information for cross-enterprise applications. ACI enables an automated and compatible business-to-business, peer-to-peer infrastructure. The ACI allows enterprise applications to understand and process content that exists in unstructured formats, such as email, Web pages, Microsoft Office documents, and IBM Notes.

ACI Server
A server component that runs on the Autonomy Content Infrastructure (ACI).

ACL (access control list)
An ACL is metadata associated with a document that defines which users and groups are permitted to access the document.

action
A request sent to an ACI server.

active directory
A domain controller for the Microsoft Windows operating system, which uses LDAP to authenticate users and computers on a network.

C

Category component
The IDOL Server component that manages categorization and clustering.

Community component
The IDOL Server component that manages users and communities.

compiled grammar
A grammar file that has been compiled from XML into ECR file format using the Eduction command-line tool edktool, so that Eduction can use it directly. See also: XML, ECR file, grammar, standard grammar, user grammar.

components
An attribute of a matched entity (a component of the single match), for example a topic or sentiment.

connector
An IDOL component (for example File System Connector) that retrieves information from a local or remote repository (for example, a file system, database, or Web site).

Connector Framework Server (CFS)
Connector Framework Server processes the information that is retrieved by connectors. Connector Framework Server uses KeyView to extract document content and metadata from over 1,000 different file types. When the information has been processed, it is sent to an IDOL Server or Distributed Index Handler (DIH).

Content component
The IDOL Server component that manages the data index and performs most of the search and retrieval operations from the index.

D

DAH (Distributed Action Handler)
DAH distributes actions to multiple copies of IDOL Server or a component. It allows you to use failover, load balancing, or distributed content.

database
An IDOL server data pool that stores indexed information. The administrator can set up one or more databases, and specifies how data is fed to the databases. By default IDOL server contains the databases Profile, Agent, Activated, Deactivated, News and Archive.

DIH (Distributed Index Handler)
DIH allows you to efficiently split and index extremely large quantities of data into multiple copies of IDOL Server or the Content component. DIH allows you to create a scalable solution that delivers high performance and high availability. It provides a flexible way to batch, route, and categorize the indexing of internal and external content into IDOL Server.

E

ECR file
ECR is a proprietary format for grammar files that Eduction can easily read at runtime. You can write grammar files in XML, then use the Eduction command-line tool edktool to compile them into ECR format. See also: XML, compiled grammar.

edktool
A command-line tool for compiling and testing Eduction grammars.

Eduction
The process of extracting entities (patterns of text) from documents.

Eduction engine
The part of any Eduction component that processes text and performs extraction and redaction operations. You can access the engine by using the Eduction SDK, Eduction Server, or an IDOL ingestion component (CFS or IDOL NiFi Ingest).

entity
In Eduction, an entity is a word, phrase, or block of information that the Eduction component can match and extract from documents. An entity can be a specific text string, such as a name, or it can be a pattern of text such as an address or phone number. You define the pattern in a grammar, which Eduction uses to find the entities in documents.

extraction
Eduction extracts entities from documents based on the rules you have created in your dictionaries and grammars, and returns an XML list of matches, or adds the matches to the source document as new fields. See also: XML, grammar, dictionary.

F

field
Fields define different parts of content in IDOL documents, such as the title, content, and metadata information.

G

grammar
In Eduction, a grammar is a pattern that defines an entity.

H

headword
A word or short phrase that Eduction matches in an entity (for example, the name of a person or place).

I

IDOL
The Intelligent Data Operating Layer (IDOL) Server, which integrates

unstructured, semi-structured and structured information from multiple repositories through an understanding of the content. It delivers a real-time environment in which operations across applications and content are automated.

IDOL Proxy component
An IDOL Server component that accepts incoming actions and distributes them to the appropriate subcomponent. IDOL Proxy also performs some maintenance operations to make sure that the subcomponents are running, and to start and stop them when necessary.

IDX
A structured file format that can be indexed into IDOL server. You can use a connector to import files into this format or you can manually create IDX files.

importing
After a document has been downloaded from the repository in which it is stored, it is imported to an IDX or XML file format. This process is called “importing”.

index
The IDOL server data index contains document content and field information for analysis and retrieval.

indexing
The process of storing data in IDOL server. IDOL server stores data in different field types (such as, index, numeric and ordinary fields). It is important to store data in appropriate field types to ensure optimized performance.

Intellectual Asset Protection System (IAS)
An integrated security solution to protect your data. At the front end, authentication checks that users are allowed to access the system that contains the result data. At the back end, entitlement checking and authentication combine to ensure that query results contain only documents that the user is allowed to see, from repositories that the user has permission to access. For more information, refer to the IDOL Document Security Administration Guide.

K

KeyView
The IDOL component that extracts data, including text, metadata, and subfiles from over 1,000 different file types. KeyView can also convert documents to HTML format for viewing in a Web browser.

L

landmark
A value that identifies a particular entity, without being a part of the entity value. For example, the phrase "Date of Birth" is a landmark for an entity that extracts dates of birth.

LDAP
Lightweight Directory Access Protocol. Applications can use LDAP to retrieve information from a server. LDAP is used for directory services (such as corporate email and telephone directories) and user authentication. See also: active directory, primary domain controller.

License Server
License Server enables you to license and run multiple IDOL solutions. You must have a License Server on a machine with a known, static IP address.

linguistic sentiment analysis (LSA)
See sentiment analysis.

Lua
An embedded scripting language that you can use to write custom scripts to expand certain IDOL functionality.

Luhn algorithm
A formula used to validate identification numbers, such as credit card numbers and social security numbers. The formula checks for errors by performing mathematical operations in the number to calculate a number that must agree with the final digit of the number.

M

metadata
Data that describes and gives information about other data. For example, the metadata for a text document might include information about the author of the document, the date it was written, or a short summary.

O

OmniGroupServer (OGS)
A server that manages access permissions for your users. It communicates with your repositories and IDOL Server to apply access permissions to documents.

P

parsing
The process of analyzing text according to the rules of a formal grammar.

pattern
A description of the entity you want to extract, which enables Eduction to produce a list of matches based on that pattern. Usually, a pattern specifies in general terms what a match looks like (for example, phone numbers), by using regular expressions. You can also use it to specify an exact list, but in this case you usually use headwords. See also: entity, extraction, grammar, headword, regular expressions.

polarity scoring
A number, usually between 0.50 and 1.50, that represents the strength of the sentiment in the matched phrase.

post-processing script
A script that performs additional processing on matched entities. This script can validate matches (for example to calculate a checksum for an ID number), and discard matches if they do not meet the script requirements.

precision
Precision is the percentage of extracted entities that are true entities. See also: recall.

primary domain controller
A server computer in a Microsoft Windows domain that controls various computer resources. See also: active directory, LDAP.

R

recall
The recall of an extraction is the percentage of matches that are actually returned, out of the total number of matches that should return in theory. See also: precision.

regular expressions
A string that allows you to define a particular string pattern in a concise format. Matching in Eduction uses regular expressions to define what you want to match.

relevance
The similarity that a particular query result has to the initial query. IDOL Server assigns results a percentage relevance score according to how closely it matches the query criteria.

S

sentiment analysis
A form of Eduction that identifies positive and negative sentiment in text.

standard grammar
Eduction includes a set of standard grammars that allow you to extract the most common entities, such as person, place, or company names, legal terms, addresses, dates, and times. See also: entity, compiled grammar, grammar, user grammar.

T

tagging
The process of adding extra information to documents. The tag might be a category, or entities returned from Eduction. Tagging usually adds a field to a document, which you can use to search by the name of a tag.

tokenization
The process of analyzing text to split it into tokens. See Also: tokens

tokens
IDOL Server stores document text as a series of tokens. Generally, a token is a word, but it can also include other strings of characters (such as a phone number or e-mail address).

U

user grammar
XML files created by the user that describe entities that can locate patterns in text using the Eduction grammar language.

V

View
An IDOL component that converts files in a repository to HTML formats for viewing in a Web browser.

W

Wildcard
A character that stands in for any character or group of characters in a query.

X

XML
Extensible Markup Language. XML is a language that defines the different attributes of document content in a format that can be read by humans and machines. In IDOL Server, you can index documents in XML format. IDOL Server also returns action responses in XML format.

Send documentation feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Micro Focus IDOL Eduction 12.13 User and Programming Guide

Add your feedback to the email and click **Send**.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to swpdl.idoldocsfeedback@microfocus.com.

We appreciate your feedback!