



Host Access for the Cloud Web Client

3.0.1

Table of contents

Welcome to Host Access for the Cloud Web Client	4
Connection Settings	5
Connection Settings	5
Common Connection Settings	5
3270 and 5250 connection settings	11
VT connection settings	12
UTS connection settings	14
T27 connection settings	16
ALC connection settings	18
Working with Sessions	20
Using Quick Keys	20
Editing the Screen	21
Logging Out	22
Macros	23
Creating Macros	23
Macro API Objects	28
Sample Macros	61
Run Macro on Event	75
Display Settings	76
Color Mapping	76
Configure Hotspots	77
Configure screen dimensions for VT, UTS and T27 hosts	78
Set Cursor Options	79
Set Font Options	79
Set VT Scrollback Buffer Options	80
Set Keyboard Options	81
Terminal Settings	83
Set Other Display Options	84
Map Keys	86
Map Keys	86

Host Keyboard Mapping	88
Transfer Files	103
IND\$FILE	103
AS/400	110
FTP	112
Batch transfers	117
Specify Edit Options	119
Printing	122
Capture a screen	122
Print a screen	122
Host printing	123
Customizing Host Sessions	128
Managing User Preferences	129
Legal Notice	130

Welcome to Host Access for the Cloud Web Client

The Host Access for the Cloud web client provides browser-based HTML5 access to 3270, 5250, VT, UTS, ALC, and T27 host applications. The Host Access for the Cloud product eliminates the need to touch the desktop; no software to deploy, patches to apply, or configurations to make. You can provide platform-independent user access to all your host applications.

Connection Settings

Connection Settings

There are common connection settings that are applicable to all host types.

Common Connection Settings

There are additional settings which are specific to your type of host.

3270 and 5250 settings

T27

UTS

VT

ALC

Common Connection Settings

These options are common to all supported host types.

- **Connect at startup**

By default, sessions are configured to connect to the host automatically when you create or open a session. However, you can set up a session so that it doesn't automatically connect to the host. Choose NO to manually connect to the host.

- **Reconnect when host terminates connection**

When set to Yes, Host Access for the Cloud attempts to reconnect as soon as the host connection terminates.

- **Protocol**

From the drop down list, select the protocol you want to use to communicate with the host. To establish a host connection, both the web client and the host computer must use the same network protocol. The available values are dependent on the host to which you are connecting. They are:


Protocol	Description
----------	-------------

TN3270	TN3270 is a form of the Telnet protocol, which is a set of specifications for general communication between desktop and host systems. It uses TCP/IP as the transport between desktop computers and IBM mainframes.
TN3270E	TN3270E or Telnet Extended is for users of TCP/IP who connect to their IBM mainframe through a Telnet gateway that implements RFC 1647. The TN3270E protocol allows you to specify the connection device name (also known as LU name), and provides support for the ATTN key, the SYSREQ key, and SNA response handling. If you try to use Telnet Extended to connect to a gateway that doesn't support this protocol, standard TN3270 will be used instead.
TN5250	TN5250 is a form of the Telnet protocol, which is a set of specifications for general communication between desktop and host systems. It uses TCP/IP as the transport between desktop computers and AS/400 computers.
Secure Shell (VT)	<p>You can configure SSH connections when you need secure, encrypted communications between a trusted VT host and your computer over an insecure network. SSH connections ensure that both the client user and the host computer are authenticated; and that all data is encrypted. Two authentication options are available:</p> <p>Keyboard Interactive - You can use this authentication method to implement different types of authentication mechanisms. Any currently supported authentication method that requires only the user's input can be performed with Keyboard Interactive.</p> <p>Password - This option prompts the client for a password to the host after a host connection is made. The password is sent to the host through the encrypted channel.</p>
Telnet (VT)	Telnet is a protocol in the TCP/IP suite of open protocols. As a character stream protocol, Telnet transmits user input from character mode applications over the network to the host one character at a time, where it is processed and echoed back over the network.
INT1 (UTS)	Provides access to Unisys 1100/1200 hosts using the TCP/IP network protocol.

TCPA (T27)	Use this protocol to connect to Unisys ClearPath NX/LX series or A Series hosts. TCPA Authentication is the process of verifying user login information. When properly configured, you can request a security credential from your application's credential server and send the credential back to the server. If the credential is valid, your application will be logged in; you do not have to enter a user ID or password. If the credential is not valid however, you will be prompted for a user ID and a password.
MATIP (ALC)	Mapping of Airline Traffic Over Internet Protocol (MATIP) uses TCP/IP for airline reservation, ticketing, and messaging traffic.

• **TLS Security**

TLS protocols allow a client and server to establish a secure, encrypted connection over a public network. When you connect using TLS, Host Access for the Cloud authenticates the server before opening a session, and all data passed between and the host is encrypted using the selected encryption level.

 **tip**

When TLS security is set to TLS 1.3 or TLS 1.2, you have the option to verify the host name against the name on the server certificate. It is highly recommended that you enable host name verification for all sessions.

The following options are available:

Security Options	Description
None	No secure connection is required.
TLS 1.3	Connect using TLS 1.3. When Verify server identity is set to Yes , the client checks the server or host name against the name on the server certificate. It is highly recommended that you enable host name verification for all sessions.
TLS 1.2	Connect using TLS 1.2. When Verify server identity is set to Yes , the client checks the server or host name against the name on the server certificate. It is highly recommended that you enable host name verification for all sessions.

• **Enable emulation tracing**

You can choose to generate host traces for a session. No is the default. Select Yes to create a new emulation host trace each time the session is launched.

Using Terminal ID Manager

MSS To use Terminal ID Manager, you must have a Terminal ID Manager server configured. See [Terminal ID Manager Guide](#).

Terminal ID Manager provides IDs to client applications at runtime and manages pooled IDs for different host types. An ID is connection data that is unique for an individual host session.

If you decide to use Terminal ID Manager and have configured the Terminal ID Manager server, then you can select from the options below to configure the criteria for acquiring an ID. All criteria must be met in order for an ID to be returned.

note

Keep in mind that by specifying a criterion, you are indicating that the ID should be allocated only when an ID with that specific value is found. The set of criteria selected here must be an exact match of the set of criteria specified on at least one Pool of IDs in Terminal ID Manager before the ID request can succeed.

Terminal ID Manager Criteria

Criterion	Description
Pool name	Include this attribute and enter the name of the pool to limit the ID search to a specified pool.
Client IP address	The IP address of the client machine will be included as part of the request for an ID.
Host address	The address of the host configured for this session will be included as part of the request for an ID.
Host port	The port for the host configured for this session will be included as part of the request for an ID.
Session name	When selected, requires that the ID is configured to be used by this session exclusively.
Session type	The session type (for example, IBM 3270, IBM 5250, UTS, ALC or T27) is always included as part of any request for an ID.

User name	<p>Use this criterion to ensure that only IDs created for exclusive use by specific users will be allocated. The current user's name, which must be found on an ID before it can be allocated, is the name of the user that the session is allocated to at runtime.</p> <p>To configure a session based on user names, a default place holder user name is available: <code>tidm-setup</code>.</p> <p>For the administrator to configure sessions using <code>tidm-setup</code>, the Terminal ID Manager must have IDs provisioned for <code>tidm-setup</code>. You can override the default name with one of your own by modifying the <code><install-dir>/sessionserver/conf/container.properties</code> file as follows:</p> <pre>id.manager.user.name=custom-username</pre> <p>where <code>custom-username</code> is replaced by the name you want to use.</p>
Application name (UTS)	<p>The name of the host application will be used as part of the request for an ID.</p>

To determine the connection attempt behavior if Terminal ID Manager does not successfully allocate an ID to this session, use **If ID is not allocated**:

- **Fail connection attempt** - When selected, the session does not attempt to connect when an ID is not allocated.
- **Allow connection attempt** - When selected, the session attempts to connect when an ID is not allocated. The attempt may be rejected by the host. There are some host types that permit a user to connect without an ID.

To confirm that Terminal ID Manager can provide an ID using the criterion and value selections you have made, click [Test Terminal ID Manager Criteria](#).

- **Send keep alive packets** - Use this setting to provide a constant check between your session and the host so that you become aware of connection problems as they occur. Choose from the following types of keep alive packets:

This option	Does this ...
None	The default. No packets are sent.
System	The TCP/IP stack keeps track of the host connection and sends keep alive packets infrequently. This option uses fewer system resources than the Send NOP Packets or Send Timing Mark Packets options.
Send NOP packets	Periodically, a No Operation (NOP) command is sent to the host. The host is not required to respond to these commands, but the TCP/IP stack can detect if there is a problem delivering the packet.

This option	Does this ...
Send timing mark packets	Periodically, a Timing Mark Command is sent to the host to determine if the connection is still active. The host should respond to these commands. If a response is not received or there is an error sending the packet, the connection shuts down.

- **Keep alive timeout (seconds)** - If you choose to use either the Send NOP packets or the Send timing mark packets option, select the interval between the keep alive requests set. The values range from 1 to 36000 seconds (1 hour); the default is 600 seconds.

Test Terminal ID Manager Criteria

The Terminal ID Manager provides IDs to client applications at runtime. To confirm that Terminal ID Manager can provide an ID using the criteria and value selections you selected use this test option.

Criteria for the current session are specified on the Connection panel after selecting **Use Terminal ID Manager** from either the Device Name (3270, 5250 host types), the Terminal ID (UTS) field, or the Station ID (T27) field. By default, the selected criteria for the current session are displayed.

Click **Test** to confirm that Terminal ID Manager can provide an ID matching the configured criterion and value selections. The test returns the name of an available ID that satisfies the selected attribute values.

Testing for other criteria and values

You can also use this panel to test criteria different from those associated with the current session.

1. Select any of the session types from the Session type list, and select the criteria you want to test. You can test alternate values that you want to use in a sample Terminal ID Manager request.
2. Click **Test** to confirm that Terminal ID Manager can provide an ID matching the criterion and value selections. The test returns the name of an available ID that satisfies the selected values.

3270 and 5250 connection settings

In addition to the [Common connection settings](#), 3270 and 5250 host types require these specific settings.

- **Terminal model**

Specify the terminal model (also known as a display station) you want Host Access for the Cloud to emulate. There are different terminal models available depending on the host type.

If you choose **Custom Model**, you can set the number of columns and rows to customize the terminal model.

- **Use Kerberos automatic sign-on (5250 only)** 

When set to **Yes** the user does not have to enter sign-on credentials. Kerberos automatic sign-on is configured in the MSS Administrative Console > Host Access for the Cloud. In configuring HACloud to use the Kerberos authentication protocol there are terms that you should understand and prerequisites to adhere to in advance of configuring this option. These options are explained in detail in the MSS Administrative Console > Host Access for the Cloud panel documentation, available from the Help button. See the Deployment Guide for more information.

- **Terminal ID (3270 only)**

When Host Access for the Cloud connects to a Telnet host, the Telnet protocol and the host negotiate a terminal ID to use during the initial Telnet connection. In general, this negotiation will result in the use of the correct terminal ID, and so you should leave this box empty.

- **TLS Security**

TLS protocols allow a client and server to establish a secure, encrypted connection over a public network. When you connect using TLS, Host Access for the Cloud authenticates the server before opening a session, and all data passed between and the host is encrypted using the selected encryption level. See [Common Connection Settings](#) for detailed information on this common setting.

- **Device name**

If you selected TN3270, TN3270E, or TN5250 as the protocol, specify the device name to use when the session connects to the host. The device name is also known as the host LU or pool. You can also choose to:

- **Generate a unique device name** - Automatically generates a unique device name.

- **Use Terminal ID Manager** - Displays additional settings to complete. See [Using Terminal ID Manager](#)

- **Always Prompt the User for ID** - The end user is prompted for the device ID each time a connection is attempted.

- **Prompt the User if ID not Specified** - The end user is prompted the first time a connection is attempted after which the value is saved. The saved value will continue to be used without additional prompting.

If you do not specify a device name for the session, the host dynamically assigns one to the session. A device name that is set within a macro will override this setting.

VT connection settings

In addition to the [Common connection settings](#), VT hosts require additional settings. These settings vary depending on the protocol you are using; Telnet or SSH. The settings are applicable to both protocols unless noted.

VT session configuration options

VT Settings	Description
Terminal ID	This setting determines the response that Host Access for the Cloud sends to the host after a primary device attributes (DA) request. This response lets the host know what terminal functions it can perform. The Host Access for the Cloud response for each Terminal ID is exactly the same as the VT terminal's response; some applications may require a specific DA response. This terminal ID setting is independent of the Terminal type setting. The options are: VT220, VT420, VT100, DEC-VT100, and VT52.
Allow unknown hosts (SSH)	<p>This setting provides the administrator with the ability to decide whether the web client will allow unknown hosts. Options are:</p> <p>Yes - Unknown hosts and all SSH connections are permitted. Web client users are not prompted about whether hosts should be trusted.</p> <p>Ask - The web client user is prompted whether the host should be trusted when they connect to an unknown host they haven't encountered before. If they choose to trust the host, then its public key is stored in their user preferences and subsequent connections will not elicit a prompt unless the host key changes.</p> <p>No - No unknown hosts are permitted. Only those hosts the administrator chooses to trust when configuring the session are permitted. End users are never prompted and the session will either connect or not connect depending on the administrator's choices.</p>

Suppress banner messages (SSH)	When enabled, the SSH banner is not displayed. This option is useful when recording SSH login macros.
Local Echo (Telnet)	Automatic (default). How Host Access for the Cloud responds to remote echo from a Telnet host: Automatic attempts to negotiate remote echo, but does what the host commands. Yes means Host Access for the Cloud negotiates local echo with the host, but always echoes, while No means Host Access for the Cloud negotiates remote echo with the host, but does not echo.
Renegotiate Echo (Telnet)	No (default). When set to Yes, passwords are not displayed on the local screen, but all other typed text is visible. Host Access for the Cloud supports the Telnet Suppress Local Echo (SLE) option when connected to a host in half-duplex mode. This means that Host Access for the Cloud will suppress character echo to the host computer, and with SLE support Host Access for the Cloud can be instructed to suppress echo locally.
Set Host Window Size	Yes (default). This setting sends the number of rows and columns to the Telnet host whenever they change. This enables the Telnet host to properly control the cursor if the window size is changed.
Request Binary (Telnet)	No (default). Telnet defines a 7-bit data path between the host and the terminal. This type of data path is not compatible with certain national character sets. Fortunately, many hosts allow for 8-bit data without zeroing the 8th bit, which resolves this problem. In some cases, however, it may be necessary to force the host to use an 8-bit data path by selecting this check box.
Send LF after CR (Telnet)	No (default). A "true" Telnet host expects to see a CrNu (carriage return/ null) character sequence to indicate the end of a line sent from a terminal. There are some hosts on the Internet that are not true Telnet hosts, and they expect to see a Lf (linefeed) character following the Cr at the end of a line. If you're connecting to this type of Telnet host, select Yes.
Ctrl-break sends (Telnet)	Choose what sequence Ctrl-break sends to the host when pressed. Options are: Telnet break sequence (the default), Interrupt process, or Nothing.

Host Character Set	The default value for the Host character set depends on the type of terminal you are emulating. This setting reflects the current terminal state of VT Host Character Set, which can be changed by the host. The associated default setting, saved with the model is DEC Supplemental.
Auto Answerback	No (default). This setting specifies whether the answerback message (set with the Answerback property) is automatically sent to the host after a communications line connection.
Answerback String	<p>This setting allows you to enter an answerback message if the host expects an answer in response to an ENQ character.</p> <p>The answerback string supports characters with codes less than or equal to 0xFFFF via Unicode escape sequences. The escape sequence begins with \u followed by exactly four hexadecimal digits. You can embed Unicode escape sequences in any string. For example, this embedded \u0045 will be interpreted as this embedded E, since 45 is the hexadecimal code for the character E.</p> <p>To pass Unicode escape sequences to the host, escape the sequence with a leading backslash. For example, to send the string literal \u001C to the host, map a key to \\u001C. Host Access for the Cloud will convert this to the string \u001C when that key is pressed and send the 6 characters of the resulting string to the host.</p>

More information

[TLS descriptions](#)

UTS connection settings

In addition to the [Common connection settings](#), UTS hosts require these additional settings:

UTS INT1 session configuration options

UTS INT1 options	Description
------------------	-------------

Application	<p>The name of the host application or host operating mode to be accessed. This is the word or phrase that the local machine sends to the host when you first establish communication with the host. If you were using a host terminal, this would be the \$\$OPEN name of the application. The application name is typically the same as the environment name. However, they can be different. For example, the environment name might be MAPPER, and the application might be UDSSRC. During a terminal emulation session, you would type \$\$OPEN MAPPER at the prompt, and INT1 would send UDSSRC to the host once the connection is established.</p>
TSAP	<p>The desired Transport Service Access Point (TSAP), up to 32 characters (such as TIPCSU for TIP connections, RSDCSU for Demand connections). A TSAP is required only if you are connecting to a Host LAN Controller (HLC) or to a Distributed Communications Processor (DCP) in IP router mode. If you're not sure which value to use, contact your host administrator.</p>
Initial transaction	<p>The character, word, or phrase that the local machine will send to the host when communication with the host is first established (up to 15 characters). This parameter is optional and is primarily used with TIP. For example, you might type ^ to run MAPPER. This parameter can also be used to transmit passwords.</p>
Start transaction	<p>When you configure an initial transaction, by default, the data is sent as soon as the session connection is established. You can decide when to send an initial transaction by using a particular string to trigger the initial transaction.</p> <p>For example, to wait for a successful login before sending initial transaction data, type in a string to be used to identify a successful login. You can use this setting in combination with Send initial transaction.</p>
Send initial transaction	<p>You can determine when the initial transaction is sent:</p> <p>Immediately - Default.</p> <p>When start of entry (SOE) character is received - Useful when multi-line transactions must be completed before sending the string.</p> <p>After specified milliseconds</p>

Terminal ID

Choose options to specify a terminal ID or to use the Terminal ID Manager. To specify a terminal ID, type it in the **Specify Terminal ID** field.

Specify Terminal ID

The Terminal ID, a terminal identifier (typically up to 8 alphanumeric characters) to use for the communication session associated with this path. Also known as a TID or PID, each terminal ID should be unique to the host.

Prompt the User if ID not Specified

The end user will be prompted the first time a connection is attempted after which the value is saved. The saved value will continue to be used without additional prompting.

Always prompt the user for ID

When you select this option the end user will be prompted for the terminal ID each time a connection is attempted.

Use Terminal ID Manager

If you choose **Use Terminal ID Manager**, you are prompted to select the Terminal ID attributes you want to use to obtain an ID. See Terminal ID Manager Attributes.

To test the attributes, click **Test**.

More information

[Terminal ID Manager Attributes](#)

[TLS Descriptions](#)

T27 connection settings

Along with the [Common connection settings](#), you can configure these additional T27 connection options:

T27 Connection Settings

T27 options	Description
Terminal type	Select the type of terminal to emulate during the session. T27 emulation supports Unisys TD830, TD830 ASCII, TD830 INTL, and TD830 NDL terminal types.
Request binary	<p>You must enable the Request binary option when you require pass through printing. The default is No.</p> <p>TCPA defines a 7-bit data path between the host and the terminal emulator. This type of data path is not compatible with certain national character sets. However, many hosts allow for 8-bit data without zeroing the 8th bit, which resolves this problem. However, it may be necessary to force the host to use an 8-bit data path; you can do so by selecting this option.</p>
Line width	Select the number of characters the host will send to the client. The default is 80 characters.
TLS security	See TLS Descriptions for a description of the various options.
Station ID	<p>Choose an option to specify a station ID or use the Terminal ID Manager. To specify a station ID, choose Specify Station ID and type the name in the Station ID field.</p> <p>Each station id should be unique to the host and typically consists of up to eight alphanumeric characters.</p> <p>Prompt the user if ID not specified The end user will be prompted the first time a connection is attempted after which the value is saved. The saved value will continue to be used without additional prompting.</p> <p>Always prompt the user for ID When you select this option the end user will be prompted for the station ID each time a connection is attempted.</p> <p>Use Terminal ID Manager If you select Use Terminal ID Manager, you will see a number of Terminal ID criteria to configure. See Terminal ID Manager Criteria for descriptions of the various options.</p> <p>If you do not specify a station id for the session, the host dynamically assigns one to the session.</p>

More information

- [TLS Descriptions](#)
- [Terminal ID Manager Criteria](#)

ALC connection settings

In addition to the [Common connection settings](#), ALC hosts require these additional settings:

ALC Connection Settings

ALC options	Descriptions
TLS security	See TLS Descriptions for a description of the various options.
Character encoding	Choose ASCII, EBCDIC, or IPARS (default) as the code set.
Configuration file	Enter the configuration (CNF) file that associates configuration information appropriate for a specific host type.
Terminal address	<p>Select whether you want to specify the terminal address or use the Terminal ID Manager.</p> <p>Terminal address - Specify whether to use 2-byte or 4-byte addressing mode.</p> <p>Although a unique 5-byte address is required when you specify the terminal ID instead of using ID Manager, this option specifies how many bytes of the 5-byte terminal ID address are sent with each message for the purposes of multiplexing. If you specify 2-byte addressing mode, only the last 2 bytes of the ASCU (Agent Set Control Unit) cluster address (A1, A2) are sent. If you specify 4-byte addressing mode, the full ASCU cluster address (H1, H2, A1, A2) is sent.</p> <p>Specify the unique 5-byte terminal address for this session. The terminal address is made up of five 2-hex-digit values in this order: H1, H2, A1, A2, and TA (terminal address). This unique address is usually assigned by the network administrator.</p> <p>Terminal ID Manager- Provides IDs to client applications at runtime. If you choose this option, there are additional configuration options to complete. See Terminal ID Manager Criteria for descriptions of those options.</p>

More information

- [TLS Descriptions](#)
- [Terminal ID Manager Criteria](#)

Working with Sessions

All the sessions you have access to are available in the Available Sessions list. Sessions are initially created and configured by your system administrator and accessed through a distributed URL (for example, `https://<clusterdns>/webclient`).

To open a session

1. Select the session and click to open.
2. Interact with your host application using the open session.
3. You can create multiple instances of a configured session.

You can have multiple sessions open at a time and easily switch between them using the tabs arranged across the top of the screen. The current session is always the left-most tab and is indicated by a white background and bold text. Each session remains active for 30 minutes.

Use the toolbar to access the various options available to you as you interact with the session. You can disconnect from a session, close the session, turn on Quick Keys, and access other settings. Some options may be only available once your administrator has granted permission.

Using Quick Keys

The Quick Key terminal keyboard provides a graphical representation of the keys on a host keyboard and gives you quick access to terminal keys.

Click a terminal key on the Quick Key keyboard to send the key to the host. Tool tips, which are available by hovering over a key, provide a description of the mapping.

Quick keys are available for each supported host type and are accessed by clicking the tool bar

icon .

Editing the Screen

note

Each browser handles copy, paste and cut functions differently and in some cases will not support the use of the toolbar buttons or the right-click context menu. It is highly recommended that you use keyboard commands for those functions. Although keyboard commands vary depending on your operating system, in Windows they are: CTRL+C to copy, CTRL+V to paste, and CTRL+X to cut.

It is far more common to encounter problems with the paste function rather than either cut or copy. If the Paste toolbar button is not visible, it is likely that browser security is preventing read access to the system clipboard. Different browsers behave differently when it comes to providing access to the clipboard. However, pasting is almost always available using the keyboard commands, (Control + V on Windows and Command + V on Macs). This assumes you have not remapped those keys. You can also use the browser's built-in paste menu item or button.

To copy from the terminal

1. Highlight the area on the terminal screen that you want to copy.
2. Click **Copy** from the toolbar or select **Copy** from the right-click context menu available within the terminal screen. You can alternatively use the keyboard command, **CTRL+C**.

To paste into the terminal screen

1. Position the cursor where you want to paste content.
2. If the browser supports the paste function, click **Paste** from the toolbar or select **Paste** from the right-click context menu available within the terminal screen. If your browser does not support this functionality, these options will not be available and you should use the keyboard command, **CTRL+V**.

To cut areas from the terminal screen

note

This function is available for all supported terminal types except for VT hosts.

1. Highlight the area on the terminal screen that you want to cut.
2. Click **Cut** from the toolbar or select **Cut** from the right-click context menu available within the terminal screen. You can alternatively use the keyboard command, **CTRL+X**.

More information

[Specify Edit Options](#)

Logging Out

In the upper right corner of the screen, open the drop-down list associated with your user name and select **Logout** to stop working with the host application.

Macros

Creating Macros

A macro is a series of keyboard actions that you record and then run. You can use these JavaScript macro programs to automate user interactions with the terminal. You can access and run macros from all supported devices.

Host Access for the Cloud records and saves advanced macros as JavaScript, making it easy to edit and enhance your recorded macros. You can record macros to playback later, run macros at startup or when the session connects or disconnects from the host. You can also write macros from scratch to perform complex tasks that the recorder cannot capture.

Macros are made available to users in two ways; created by an administrator or recorded by users for their own private use. All advanced macros are associated with a session and they all accomplish the same goal, automating host interaction. The only difference between the two flavors is simply who can access them and who manages their creation and availability:

- **Macros created by administrators**

Administrators create macros when they create the session. They are specific to a session and are available to all users who have access to the session from the Macro icon on the toolbar. Administrators can designate macros to run at startup or when the session connects or disconnects from the host.

- **Macros created by users**

End-user macros are created by individuals for sessions they are authorized to access. The administrator grants permission to create macros by setting a User Preference Rule. Users can access the session under their own credentials or in a Guest role. Macros that Guest users create are available to all Guest users. Users who are logged in using their own credentials can only see macros that they have created.

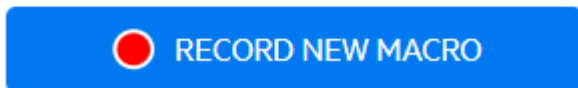
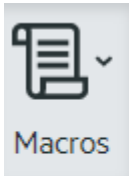
Advanced macros are listed in alphabetical order in the drop down list available from the toolbar. Macros created by the end-user are listed first and followed by an indicator of three vertical grey dots, which when selected, displays the Edit and Delete options. Macros created by the administrator are listed without the indicator as those macros cannot be modified by the end-user.


Working with Macros

Follow these steps to record, edit, and run macros.

Record

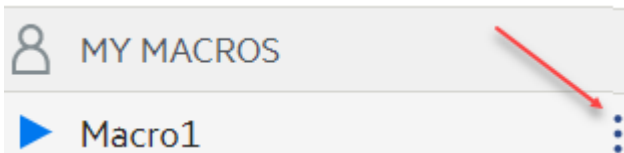
1. Click the Macro icon on the toolbar, and then click Record New Macro.




2. Navigate the host application to record the series of steps you want included in the macro.
3. Click  on the toolbar to stop recording. The red dot pulses to indicate the recording is in process.
4. When prompted, type a name for the macro.


Edit

1. From the Macro drop-down list, select the macro you want to edit.




2. Click the three vertical dots to expand the field.
3. Click  to open the Macro Editor (in the left panel).
4. Use JavaScript to make whatever changes are necessary. You can run and save the modified macro using the toolbar icons in the upper panel of the editor.

Run

To run a macro, choose the macro from the drop-down list and click .

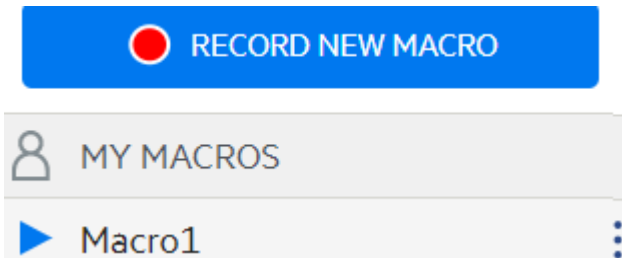
You can also map keys that will automatically trigger an already recorded macro. In the Key Map settings dialog box, choose **Run Macro** from the **Action** drop down list. Choose a macro to associate with the key mapping from the **Value** list.

Stop

You can stop a macro before it completes from either the Macro Editor or the toolbar. Click  to stop the macro. To rerun the macro, navigate back to the macro starting screen.

Delete

1. From the Macro drop down list, select the macro you want to delete.
2. Expand the field, by clicking the three vertical dot icon.



3. Click **Delete**.

View

The Macro drop down list is available from the toolbar to all users who have permission to record macros or are accessing a session where macros have been pre-recorded by the administrator for use with that session.

Macros are listed under either **MY MACROS** or **MACROS** depending on how they were recorded.

All users, whether they are logged in using their credentials or as Guest, can see the macros associated with the session. Macros listed under the **MY MACROS** heading are listed in alphabetical order by name and are visible to those users that recorded them. Macros recorded by the administrator and attached to a session are listed alphabetically under **MACROS**.

Debugging Macros

Since macros are written in JavaScript and executed in the browser, the best way to debug and troubleshoot them is by using your web browser's built-in tools. Modern browsers come with a very capable set of tools for debugging JavaScript code. You can place breakpoints, step through code, and output debug information.

hint

JavaScript is case sensitive. Keep that in mind when editing JavaScript code.

To debug a macro:

1. Open the macro for editing. See [Working with macros](#) for instructions.

2. Open your browser's development tools.

Browser	Open debugger
Mozilla Firefox	From the toolbar, open the Menu, and choose Developer. From the Web Developer menu, choose Debugger. The debugger opens in a lower panel.
Google Chrome	From the toolbar, open the Menu, and choose More tools. Choose Developer Tools to open the Debugger.

3. Use one of the these tools in your macro code, and run the code.

• debugger

The most thorough approach to debugging is to use the `debugger;` statement. When you insert these statements into your macro code then run it, with the browser's development tools open, the execution will stop on those lines. You can step through your macro, view the value of local variables and whatever else you need to check.

You are encouraged to place multiple `debugger;` statements in your code to help get to the correct line. The asynchronous nature of JavaScript can make stepping through code challenging. This can be offset by using multiple, carefully placed `debugger;` statements.

Example 1: `debugger`

```
var hostCommand = menuSelection + '[enter]';
debugger; // <- Browser's debugger will stop here
ps.sendKeys(hostCommand);
```

• `console.log()`, `alert()`

These two functions are commonly used for debugging JavaScript. While not as flexible as the debugger statement they provide a quick way to output debug information. These functions output the information to the JavaScript "Console" tab in the browser's developer tools.

Example 2: `console.log()`, `alert()`

```
var hostCommand = menuSelection + '[enter]';
console.log('Command:' + hostCommand); // <- Will output the string to "Console" tab
alert('Command:' + hostCommand); // Will pop up a small window containing the data
ps.sendKeys(hostCommand);
```

• `ui.message()`

The Host Access for the Cloud Macro API provides an `ui.message()` function that is very similar to JavaScript's `alert()` function. You can also use `ui.message()` to output debug information.

Example 3: `ui.message()`

```
var hostCommand = menuSelection + '[enter]';
ui.message('Command:' + hostCommand); // <- Will pop up a message window
ps.sendKeys(hostCommand);
```

Keep in mind:

- Stepping and “yields”

While the yield statements in macros make them easier to understand, they can make the code more challenging to step through with the debugger. Consider either using multiple debugger statements or carefully placed debugger statements of `console.log()` calls to output the right debug information.

Using the Macro API

In Host Access for the Cloud, macros are recorded and written using JavaScript.

The Macro API consists of a set of objects which you can use to interact with the host, wait for screen states, and interact with the user.

About promises and yields

Because JavaScript is single-threaded and uses 'callback functions' and 'promises' to help manage the flow of execution through code, often code can be difficult to follow. Host Access for the Cloud combines the concept of 'promises' with the 'yield' keyword so macro code can be organized in a more linear fashion.

- **Promises**

Promises are patterns to help simplify functions that return their result asynchronously, at some point in the future. All 'wait' and 'ui' functions in the Macro API return promise objects.

- **Yield**

Macros use the yield keyword to block the execution of the macro until a promise is resolved, or done. So putting yield in front of any 'wait' or 'ui' function makes the macro execution pause until that function has finished executing. You can place the yield keyword in front of any function that returns a promise, even your own custom functions.

note

The ability to make macro execution block by combining yield with promises is enabled by the `createMacro()` function.

Errors

Errors are handled in macros using a try / catch statement. Some API functions may throw errors if, for example, conditions can't be met or a timeout occurs. The thrown error is 'caught' in the catch statement. You can wrap smaller blocks of code in a try / catch statement to handle errors at a more granular level.

Macro developers can also throw errors by using `throw new Error('Helpful error message');`

More information

[Macro API Objects](#)

[Sample Macros](#)

Macro API Objects

You can create macros using the Macro API. By default for use in macros, there are four primary objects available:

- **Session** - the main entry point for access to the host. You use the Session object to connect, disconnect and provide access to the PresentationSpace object.
- **PresentationSpace** - represents the screen and provides many common capabilities such as getting and setting the cursor location, sending data to the host and reading from the screen. It is obtained by calling `session.getPresentationSpace()`.
- **Wait** - provides a simple way to wait for various host states to occur before continuing to send more data or read from the screen. For example, you can wait for the cursor to be located at a certain position, text to be present in a position on the screen or simply wait for a fixed amount of time. All 'Wait' function calls require the yield keyword, which is explained below.
- **User Interface** - automatically available in your macro as the "ui" variable. It provides basic user interface capabilities. You can use this object to display data to the user or prompt them for information. All 'UI' function calls require the yield keyword.

All available objects

See the list of available objects in the right navigation, "On this page." (You may need to expand your browser.)

Attribute

Use the Attribute, along with the AttributeSet, to decode the formatting information present on the data cell.

Attribute	Indicates...
PROTECTED	a protected data cell
MODIFIED	a modified data cell
NUMERIC_ONLY	the beginning of a numeric only data cell
ALPHA_NUMERIC	an alpha numeric data cell.
HIGH_INTENSITY	whether the data cell contains high intensity text
HIDDEN	whether the data cell contains hidden text
PEN_DETECTABLE	whether the data cell is pen detectable
ALPHA_ONLY	an alpha only data cell
NUMERIC_SHIFT	the beginning of a numeric shift field
NUMERIC_SPECIAL	the data cell marks the beginning of a numeric special field
KATAKANA_SHIFT	a section of Katakana text
MAGNETIC_STRIPE	the data cell marks the beginning of a magnetic strip field
SIGNED_NUMERIC_ONLY	the data cell is a signed numeric field
TRANSMIT_ONLY	the data cell is a transmit only field
FIELD_END_MARKER	the data cell marks the end of a modified field
FIELD_START_MARKER	the data cell marks the start of a modified field
SPECIAL_EMPHASIS_PROTECTED	a special emphasis protected field
TAB_STOP	that the data cell contains a tab stop
REVERSE	the data cell displays in reverse video mode
BLINKING	the data cell contains blinking text

Attribute	Indicates...
RIGHT_JUSTIFIED	the data cell marks the beginning of a right justified field
LEFT_JUSTIFIED	the data cell marks the beginning of a left justified field
LOW_INTENSITY	the data cell contains low intensity text
UNDERLINE	the data cell contains underlined text
DOUBLE_BYTE	the data cell contains double byte text
COLUMN_SEPARATOR	the data cell contains a column separator
BOLD	the data cell contains bold text
DOUBLE_WIDTH	the data cell marks a double width field
DOUBLE_HEIGHT_TOP	a double height top data cell
DOUBLE_HEIGHT_BOTTOM	a double height bottom data cell
CONTROL_PAGE_DATA	the data cell contains control page data
RIGHT_COLUMN_SEPARATOR	the data cell contains a right column separator
LEFT_COLUMN_SEPARATOR	a data cell containing a left column separator
UPPERSCORE	the data cell contains an upperscore
STRIKE_THROUGH	the data cell contains strike through text

AttributeSet

The AttributeSet object allows the user to decode the attributes that are present on the data cell. The AttributeSet object returns values defined in the [Attribute](#) object and when used together, you can get formatting information from the data cell.

Method	Description
<code>contains(attribute)</code>	<p>Determines if the set contains the specified Attribute.</p> <p>Parameters</p> <p><code>{Number}</code> attribute to check</p> <p>Returns</p> <p><code>{Boolean}</code> True if the attribute is in the set</p>

<code>isEmpty()</code>	<p>Determines if the attribute set is empty.</p> <p>Returns</p> <p>{Boolean} True if the set is empty.</p>
<code>size()</code>	<p>Indicates the number of attributes in a set.</p> <p>Returns</p> <p>{Number} The attribute count.</p>
<code>toArray()</code>	<p>Converts the internal attribute set to an array.</p> <p>Returns</p> <p>{Number[]} Array of values of attributes in the set.</p>
<code>toString()</code>	<p>Converts the internal attribute set to a string.</p> <p>Returns</p> <p>{String} Space-delimited names of attributes in the set.</p>
<code>forEach(callback, thisArg)</code>	<p>Function to iterate over each element in the attribute set.</p> <p>Parameters</p> <p>{forEachCallback} Callback to perform the specific operation. Called with the name of each attribute in the set.</p> <p>{Object} thisArg optional pointer to a context object.</p>

```
forEachCallback(string,  
object)
```

A user provided callback function where you provide the behavior, to be used as the callback parameter to forEach.

Parameters

```
{String} String name of an attribute in the attribute
```

```
set.
```

```
{Object} thisArg optional pointer to a context object.
```

AutoSignOn

Method	Description
<code>getPassTicket()</code>	<p>Obtains a pass ticket to be used for signing onto a mainframe application. Multiple pass tickets may be requested using different application IDs.</p> <p>Parameters</p> <p>{String} application ID tells the host which application the sign on is for</p> <p>Returns</p> <p>{Promise} fulfilled with the pass ticket key or rejected if the operation fails. The pass ticket obtained from DCAS only works with the current host session and is valid for ten minutes.</p>
<code>sendUserName()</code>	<p>Applies the user name contained in the pass ticket to the field at the current cursor location on the current host screen. The user name must be sent before the password. Sending the password first will invalidate the pass ticket, and you will need to get another one.</p> <p>Parameters</p> <p>{String} passTicketKey opbtained from getPassTicket</p> <p>Returns</p> <p>{Promise} fulfilled if the user name is successfully sent. Rejected if the operation fails.</p>

`sendPassword()`

Applies the password contained in the pass ticket to the field at the current cursor location on the current host screen. The user name must be sent before the password. Sending the password first will invalidate the pass ticket, and you will need to get another one.

Parameters

{String} passTicketKey obtained from getPassTicket

Returns

{Promise} fulfilled if the password is successfully sent. Rejected if the operation fails.

Color

Color constants to use for the DataCell object foreground and background colors.

Color	Description	Numeric Value
BLANK_UNSPECIFIED	No color specified	0
BLUE	Blue	1
GREEN	Green	2
CYAN	Cyan	3
RED	Red	4
MAGENTA	Magenta	5
YELLOW	Yellow	6
WHITE_NORMAL_INTENSITY	Normal intensity white	7
GRAY	Gray	8
LIGHT_BLUE	Light blue	9
LIGHT_GREEN	Light green	10
LIGHT_CYAN	Light cyan	11
LIGHT_RED	Light red	12
LIGHT_MAGENTA	Light magenta	13
BLACK	Black	14
WHITE_HIGH_INTENSITY	High intensity white	15

Color	Description	Numeric Value
BROWN	Brown	16
PINK	Pink	17
TURQUOISE	Turquoise	18

ControlKey

The ControlKey object defines constants for sending cursor control keys and host commands using the sendKeys method. Constants are available for these host types:

IBM 3270

IBM 5250

VT

UTS

IBM 3270

Key word	Description
ALTVIEW	Alternate view
ATTN	Attention
BACKSPACE	Back space
BACKTAB	Back tab
CLEAR	Clear or clear display
CURSOR_SELECT	Cursor select
DELETE_CHAR	Delete, delete character
DELETE_WORD	Delete word
DEST_BACK	Destructive backspace
DEV_CANCEL	Device cancel
DOWN	Cursor down
DSPSOSI	Display SO/SI
DUP	Duplicate field

Key word	Description
END_FILE	End of field
ENTER	Enter
ERASE_EOF	Erase end of field
ERASE_FIELD	Erase field
ERASE_INPUT	Erase input
FIELD_MARK	Field mark
HOME	Cursor home
IDENT	Ident
INSERT	Insert
LEFT_ARROW	Cursor left
LEFT2	Left two positions
NEW_LINE	New line
PA1 - PA3	PA1 - PA3
PF1 - PF24	PF1 - PF24
PAGE_DOWN	Page down
PAGE_UP	Page up
RESET	Reset, reset terminal
RIGHT2	Right 2 positions
RIGHT_ARROW	Cursor right, right
SYSTEM_REQUEST	System request
TAB	Tab key
UP	Cursor up

IBM 5250

Key word	Description
ALTVIEW	Alternate view

Key word	Description
ATTN	Attention
AU1 - AU16	AU1 - AU16
BACKSPACE	Back space
BACKTAB	Back tab
BEGIN_FIELD	Begin field
CLEAR	Clear or clear display
DELETE_CHAR	Delete, delete character
DEST_BACK	Destructive backspace
DOWN	Cursor down
DSPSOSI	Display SO/SI
DUP	Duplicate field
END_FILE	End of field
ENTER	Enter
ERASE_EOF	Erase end of field
ERASE_FIELD	Erase field
ERASE_INPUT	Erase input
FIELD_EXT	Field exit
FIELD_MINUS	Field minus
FIELD_PLUS	Field plus
FIELD_MARK	Field mark
HELP	Help request
HEXMODE	Hex mode
HOME	Cursor home
INSERT	Insert
LEFT_ARROW	Cursor left
NEW_LINE	New line

Key word	Description
PA1 - PA3	PA1 - PA3
PF1 - PF24	PF1 - PF24
PAGE_DOWN	Page down
PAGE_UP	Page up
[print]	Print
RESET	Reset, reset terminal
RIGHT_ARROW	Cursor right, right
SYSTEM_REQUEST	System request
TAB	Tab key
UP	Cursor up

VT

Key word	Description
BACKSPACE	Back space
BREAK	Break
CLEAR	Clear or clear display
CURSOR_SELECT	Cursor select
DELETE_CHAR	Delete, delete character
DOWN	Cursor down
EK_FIND	Edit keypad find
EK_INSERT	Edit keypad insert
EK_NEXT	Edit keypad next
EK_PREV	Edit keypad previous
EK_REMOVE	Edit keypad remove
EK_SELECT	Edit keypad select
END_FILE	End of field

Key word	Description
ENTER	Enter
F1 - F24	F1 - F24
HOLD	Hold
HOME	Home
INSERT	Insert
KEYPAD_COMMA	Keypad comma
KEYPAD_DOT	Keypad decimal
KEYPAD_ENTER	Keypad enter
KEYPAD_MINUS	Keypad minus
KEYPAD0 - KEYPAD9	Keypad 0 - Keypad 9
LEFT_ARROW	Cursor left
PF1 - PF20	PF1 - PF20
PAGE_DOWN	Page down
PAGE_UP	Page up
RESET	Reset, reset terminal
RETURN	Return, carriage return
RIGHT_ARROW	Cursor right, right
TAB	Tab key
UDK16 - UDK20	User defined key 6 - User defined key 20
UP	Cursor up

UTS

Key word	Description
BACKSPACE	Back space
BACKTAB	Back tab
CHAR_ERASE	Erases character at the cursor and advances the cursor.

Key word	Description
CLEAR_DISPLAY	Clear display
CLEAR_EOD	Clear to end of display
CLEAR_EOF	Clear to end of field
CLEAR_EOL	Clear to end of line
CLEAR_FCC	Clear Field Control Character
CLEAR_HOME	Clear display and cursor home
CONTROL_PAGE	Toggles the control page
DELETE_LINE	Deletes the line containing the cursor and shifts remaining lines up one row
DELIN_LINE	Deletes character under cursor and shifts remaining characters on line to the left.
DELIN_PAGE	Deletes character under cursor and shifts remaining characters on page to the left.
DOWN	Moves the cursor down one line. Wraps at bottom.
DUP_LINE	Creates a copy of the current line and overwrites the next line with the duplicate.
END_FIELD	Moves the cursor to the end of the current field.
END_PAGE	Moves the cursor to the end of the current page.
EURO	Inserts the Euro character
F1 - F22	Function keys F1-F22
HOME	Moves the cursor to beginning of current page (row 1, col 1)
INSERT	Toggles insert/overwrite mode.
INSERT_IN_LINE	Inserts space at cursor position and shifts the remaining characters on the line to the right. The character in the far right column on the line is discarded.
INSERT_IN_PAGE	Inserts space at cursor position and shifts the remaining characters on the page to the right. The character in the far right column on each line is discarded.

Key word	Description
INSERT_LINE	Inserts a new line at the cursor row and shifts the remaining lines down. The last row on the page is discarded.
LEFT_ARROW	Moves the cursor one position to the left wrapping if necessary.
LOCATE_FCC	Finds the next field control character on the screen.
MSG_WAIT	Retrieves messages queued to the terminal.
RETURN	Carriage return
RIGHT_ARROW	Moves the cursor one position to the right, wrapping if necessary.
SOE	Inserts the Start of Entry character
START_OF_FIELD	Moves the cursor to the beginning of the field.
START_OF_LINE	Moves the cursor to column 1 of current line.
TAB	Moves the cursor to the next tab position of the screen.
TOGGLE_COL_SEP	Toggles the column separator attribute.
TOGGLE_STRIKE_THRU	Toggles the strike-through attribute on the current data cell.
TOGGLE_UNDERLINE	Toggles the underline attribute on the current data cell.
TRANSMIT	Transmits changed field data to the host.
UNLOCK	Sends the UNLOCK key to the host.
UP	Moves the cursor up one row, wrapping if necessary.

DataCell

The DataCell object provides information about a particular position on a terminal screen.

Method	Description
<code>getPosition()</code>	Returns the position of this data cell on the screen. Returns <code>{Position}</code> the position of the data cell on the screen

<code>getChar()</code>	Obtains the character associated with the cell. Returns {String} The character associated with the cell.
<code>getAttributes()</code>	Returns the set of attributes specified for this data cell instance. See AttributeSet . Returns {AttributeSet} The set of attributes for this data cell instance.
<code>getForegroundColor()</code>	Returns the foreground color, as defined in the Color object, for this data cell. Returns {Number} Foreground color for this data cell. The color is defined in the Color object.
<code>getBackgroundColor()</code>	Returns the background color, as defined in the Color object, for this data cell. Returns {Number} Background color for this data cell. The color is defined in the Color object.
<code>toString</code>	Converts the internal data cell to a string. Returns {String} The string representation of a data cell.
<code>isFieldDelimiter()</code>	Tests if this cell represents a field delimiter. Returns {Boolean} True if this cell is a field delimiter, false if otherwise.

Dimension

Represents the size of the screen or screen area.

Method	Description

`Dimension(rows,cols)`

Creates a new `Dimension` instance.

Parameters

`{Number} rows` screen rows dimension

`{Number} cols` screen columns dimension

Field

Use the `Field` object, along with [FieldList](#), to obtain the information present in a field on the screen.

Method	Description
<code>getAttributes()</code>	Returns the set of attributes specified for this field instance. See AttributeSet . Returns <code>{AttributeSet}</code> The set of attributes for this field.
<code>getForegroundColor()</code>	Returns the foreground color for the field. Returns <code>{Number}</code> Foreground color for this field. These values are defined in the Color object.
<code>getBackgroundColor()</code>	Returns the background color of the field. Returns <code>{Number}</code> Background color for this field. These values are defined in the Color object.
<code>getStart()</code>	Returns the starting position of the field. The starting position is the position of the first character of the field. Some host types use a character position to store field level attributes. In this case, the attribute position is not considered the start position. Returns <code>{Position}</code> Starting position of the field. Throws <code>{RangeError}</code> For zero length fields.
<code>getEnd()</code>	Returns the ending position of the field. The ending position is the position in the presentation space containing the last character of the field. Returns <code>{Position}</code> Ending position of the field. Throws <code>{RangeError}</code> For zero length fields.

<code>getLength()</code>	<p>Returns the length of the field. For host types that use a character position to store the field attributes, the field length does not include the field attribute position</p> <p>Returns</p> <p>{Number} Length of the field.</p>
<code>getDataCells()</code>	<p>Obtains the data cells that comprise this field. See DataCell.</p> <p>Returns</p> <p>{DataCell[] } Data cells that comprise this field.</p>
<code>getText()</code>	<p>Obtains the text from the field.</p> <p>Returns</p> <p>{String} field text.</p>
<code>setText()</code>	<p>Sets the field text. For certain host types, like VT, the text is transmitted to the host right away, but in other host types, the text is not transmitted to the host until an Aid key is invoked. If the text is shorter than the field, the text is placed in the host field, and the remainder of the field is cleared. If the text is longer than the host field, then as much text as will fit is placed in the field.</p> <p>Parameter</p> <p>{String} Text to set on the field.</p> <p>Throws</p> <p>{Error} If the field is protected.</p>
<code>clearField()</code>	<p>Clears the current field in an emulation-specific manner.</p> <p>Throws</p> <p>{Error} If the field is protected or clear is not supported.</p>
<code>getPresentationSpace()</code>	<p>Obtains the PresentationSpace that created this field.</p> <p>Returns</p> <p>{PresentationSpace} Parent of this field instance.</p>

toString()

Creates a user-friendly description of the field.

Returns

{String} A user readable rendition of the field.

FieldList

Use the FieldList object, along with Field object, to obtain field list information.

Method	Description
getPresentationSpace()	<p>Obtains the PresentationSpace that created this field.</p> <p>Returns</p> <p>{PresentationSpace} Parent of this field instance.</p>
findField(position, text, direction)	<p>Returns the field containing the specified text. The search starts from the specified position and proceeds either forward or backward. If the string spans multiple fields, the field containing the starting position is returned. When searching forward the search will not wrap to the top of the screen. When searching backward the search will not wrap to the bottom of the screen.</p> <p>Parameters</p> <p>{Position} Position from which to start the search. See Position object.</p> <p>{String} The text to search for (optional). If not provided, returns the next field to the right of or below the specified position.</p> <p>{Number} direction of the search (optional). Use PresentationSpace. SearchDirection constants for this parameter. For example, PresentationSpace.SearchDirection.FORWARD or PresentationSpace.SearchDirection.BACKWARD. If not provided, searches forward.</p> <p>Returns</p> <p>{Field} containing the string or null if a field meeting the given criteria is not found.</p> <p>Throws</p> <p>{RangeError} If the position is out of range.</p>

<code>get(index)</code>	Obtains the field at the given index. Parameters <code>{Number}</code> index into the field list. Returns <code>{Field}</code> located at the specified index. Throws <code>{RangeError}</code> If the index is out of range.
<code>isEmpty()</code>	Determines if the field list is empty. Returns <code>{Boolean}</code> True if the list is empty.
<code>size()</code>	Indicates the number of fields in the list. Returns <code>{Number}</code> The field count.
<code>toString()</code>	Creates a user-friendly description of the field list. Returns <code>{String}</code> A user readable rendition of the field list.

FileTransfer

Use the FileTransfer object to list and transfer files between the host system and the client.

The Host Access for the Cloud file transfer API abstracts the file path conventions used by different host file implementations. Follow URL or Linux file system path formats when formatting file paths used by the API. For example, `/root/directory/file`.

It is important to observe any rules specific to host systems, such as allowable characters or name lengths.

note

Browsers place significant security restrictions around the ability of Javascript to interact with client file systems.

Method	Description
--------	-------------

```
getHostFileListing(remotePath)  
( )
```

Request a listing of host files. If `RemotePath` is omitted, a file listing for the current remote working directory is shown.

Parameters

`{String}` (optional) If specified will get file listing for specified remote path. If not specified, will get file listing for current remote working directory.

Returns

`{Promise}` Resolves to an array of `HostFile` objects contained at `remoteName`. Rejected if the remote path cannot be read.

```
sendFile(localFile,  
remoteName)
```

Sends specified file to the host.

Parameters

`{File}` Javascript file object pointing to local file to send.

`{String}` (optional) Fully-qualified remote file name as allowed by remote system (Unix, Windows, MVS, VAX).

Returns

`{Promise}` fulfilled with a `HostFile` object representing the sent file on success. Rejected if an error occurred sending the file.

```
getDownloadURL(remoteName)
```

Constructs a link to download a file from a host system.

Parameters

`{String}` Fully-qualified remote file name as allowed by remote system (Unix, Windows, MVS, VAX).

Returns

`{URL}` that can be used to retrieve the file from the Host Access for the Cloud session server.

`setTransferOptions(options)`

Set transfer options for current FileTransfer session. The transfer options are applied to all future transfers until the session is either exited or overridden by another call to

`setTransferOptions`.

Parameters

{JSON} see FileTransferOptions for allowed names and values.

Returns

{Promise} fulfilled when the call completes. Rejected if an error occurred setting the options.

`cancel()`

Cancels the current transfer in progress.

Parameters

{String} Fully-qualified remote file name as allowed by remote system (Unix, Windows, MVS, VAX).

Returns

{Promise} fulfilled when the call completes. Rejected if an error occurred canceling the transfer.

FileTransferFactory

A fileTransferFactory object is available to all macros. If file transfers are configured for the session, you can use it to get a reference to a FileTransfer object.

Method	Description
--------	-------------

`getIND$File()` Returns a FileTransfer object for interacting with the configured Ind\$File type for the session.

Returns
`{FileTransfer}`

Throws
`{Error}` If the session hasn't been configured to allow IND\$File transfers.

FileTransferOptions

File transfer option object specification. Example:

```
`fileTransfer.setTransferOptions({ transferMethod : 'ascii' });``
```

Method	Description
<code>transferMethod</code>	<code>{String}</code> Allowed values: 'ascii' 'binary'

HostFile

A HostFile object represents a file on the host file system.

Method	Description
<code>getName()</code>	Gets the file name. Returns <code>{String}</code> the file name.
<code>getParent()</code>	Gets the parent of this host file. Returns <code>{String}</code> the parent of this host file. This means different things on different host types. For example on TSO this is the name of the catalog in which the file resides.
<code>getSize()</code>	The byte size of the file. Returns <code>{Number}</code> the size of the file in bytes.

<code>getType()</code>	The type of file represented. Returns
------------------------	---

HostFileType

The HostFileType object defines constants for determining the type of a HostFile object.

Value	Description
FILE	Represents a file on the host system.
DIR	Represents a directory on the host system.
UNKNOWN	Represents a host file of unknown origin.

OIA

Operator Information Area (OIA) interface. The OIA object returns values that are defined in the OIAStatus object.

Method	Description
<code>getStatus ()</code>	Returns the set of enabled status flags. See StatusSet . Returns {StatusSet} Containing the status information.
<code>getCommErrorCode()</code>	Returns the current communication error code. Returns {Number} the current communication error code. If one doesn't exist, it will be 0.

`getProgErrorCode()`

Returns the current program error code.

Returns

{Number} the current program error code. If one doesn't exist, it will be 0.

OIAStatus

OIAStatus	Description
CONTROLLER_READY	Controller ready
A_ONLINE	Online with a non-SNA connection
MY_JOB	Connected to a host application
OP_SYS	Connected to a SSCP (SNA)
UNOWNED	Not connected
TIME	Keyboard inhibited
SYS_LOCK	System lock following AID key
COMM_CHECK	Communication check
PROG_CHECK	Program check
ELSEWHERE	Keystroke invalid at cursor location
FN_MINUS	Function not available
WHAT_KEY	Keystroke invalid
MORE_THAN	Too many characters entered in the field
SYM_MINUS	Symbol entered not available
INPUT_ERROR	Operator input error (5250 only)
DO_NOT_ENTER	Do not enter
INSERT	Cursor in insert mode
GR_CURSOR	Cursor in graphics mode
COMM_ERR_REM	Communications error reminder
MSG_WAITING	Message waiting indicator

OIAStatus	Description
ENCRYPT	Session is encrypted
NUM_FIELD	Invalid character in numeric only field

Position

Method	Description
<code>Position(row, col)</code>	<p>Creates a new Position instance.</p> <p>Parameters</p> <ul style="list-style-type: none"> <code>{Number}</code> row screen row coordinate <code>{Number}</code> col screen column coordinate

PresentationSpace

Use the PresentationSpace object to interact with the terminal screen. Setting and getting the cursor position, sending keys, and reading text are some of the interactions available.

Method	Description
<code>getCursorPosition()</code>	<p>Returns a Position instance representing the current cursor position. An unconnected session has a cursor position of 0,0.</p> <p>Returns</p> <ul style="list-style-type: none"> <code>{Position}</code> current cursor location
<code>setCursorPosition(position)</code>	<p>Moves the host cursor to the specified row and column position. For some hosts, such as VT, the host may constrain the movements of the cursor.</p> <p>Parameters</p> <ul style="list-style-type: none"> <code>{Position}</code> Position new cursor position. <p>Returns</p> <p>None</p> <p>Throws</p> <ul style="list-style-type: none"> <code>{RangeError}</code> If the position is not valid on the current screen.

<code>isCursorVisible()</code>	<p>Tests that the cursor is currently visible in the presentation space. The cursor is considered not visible if the session is not connected.</p>
	<p>Returns</p> <p><code>{Boolean}</code> True if the cursor is visible. False if the cursor is not visible.</p>
<code>sendKeys(keys)</code>	<p>Transmits a text string or <code>ControlKey</code> to the host at the current cursor position in the presentation space. If the cursor is not in the desired position, then use <code>setCursorPosition</code> function first.</p> <p>The text string can contain any number of characters and <code>ControlKey</code> objects.</p> <p>For example: <code>"myname" + ControlKey.TAB + "mypass" + ControlKey.ENTER</code> will transmit a user ID, tab to the next field, transmit a password, and then transmit the Enter key.</p> <p>If you need to transmit a square bracket, double the brackets (<code>[[</code> or <code>]]</code>).</p>
	<p>Parameters</p> <p><code>{String}</code> keys text and/or control keys to transmit</p>
<code>getText(start, length)</code>	<p>Returns a string representing a linear area of the presentation space. No new line characters are inserted if row boundaries are encountered.</p>
	<p>Parameters</p> <p><code>{Position}</code> start position from which to retrieve text</p> <p><code>{Number}</code> length the maximum number of characters to return. If the length parameter causes the last position of the presentation space to be exceeded then only those characters up to the last position will be returned.</p>
	<p>Returns</p> <p><code>{String}</code> representing a linear area of the presentation space which may be empty if the session is not connected.</p>
	<p>Throws</p> <p><code>{RangeError}</code> If the position or length are not valid on the current screen.</p>

`getSize()`

Gets the dimensions of the screen as a Dimension object.

Returns

{Dimension} Containing the number of rows and columns. The screen size is [row:0, col:0] if the session is not connected.

`getDataCells(start, length)`

Returns [DataCell](#) instances where the first member will be for the position specified by the start parameter. The maximum number of DataCell instances in the list is specified by the length parameter.

Parameters

{Position} start the first position on the host screen in which to retrieve DataCell instances. See [Position](#).

{Number} length of the maximum number of DataCell instance to be retrieved. If not specified, returns DataCells from the start position to the end of the screen.

Returns

{DataCell[]} Instances which may be empty if the session is not connected. If position is not specified, returns all DataCells. If length is not specified, returns DataCells from the start position to the end of the screen.

Throws

{RangeError} if start or length are out of range.

`getFields()`

Returns a list of the fields in the presentation space. If the host type does not support fields or the current screen is not formatted then the return value will always be an empty list. See [FieldList](#).

Returns

`{FieldList}` of host defined fields in the presentation space.

Session

The session object is the main entry point for interacting with the host. It contains functions for connecting, disconnecting, and obtaining the `PresentationSpace` object.

Method	Description
<code>connect()</code>	Connects to the configured host. If needed, use <code>wait.forConnect()</code> to block macro execution until the session is connected. Returns <code>None</code>
<code>disconnect()</code>	Disconnects from the configured host. If needed, use <code>wait.forDisconnect()</code> to block macro execution until the session is connected. Returns <code>None</code>
<code>isConnected()</code>	Determines whether the connection to the host is connected. Returns <code>{Boolean}</code> true if host connection is established; false if not.
<code>getPresentationSpace()</code>	Provides access to the <code>PresentationSpace</code> instance for this session. Returns <code>{PresentationSpace}</code> instance associated with this session.
<code>getDeviceName()</code>	Returns the device name for a connected session or an empty string if the session is disconnected or doesn't have device name. Returns <code>{String}</code> The connected device name.

<code>getType()</code>	Returns the type of host session. See SessionType . Returns {String} The type of host session.
<code>setDeviceName()</code>	Provides a means to modify the device name on a session instance. Parameters {String} name Device name to use when connecting to a host. Throws {Error} If an attempt is made to set the device name while the session is connected.
<code>getOIA()</code>	Provides access to the OIA instance for this session. Returns {OIA} Associated with this session

SessionType

Constants used to identify the type of host to which the connection is being made. See [Session](#) object.

Available host types:

IBM_3270

IBM_5250

VT

ALC

UTS

T27

StatusSet

You can use the StatusSet object to decode the OIA status. The StatusSet object returns values defined in the [OIAStatus](#) object and when used together, you can get status information from the OIA.

Method	Description
--------	-------------

<code>contains(statusFlag)</code>	<p>Determines if the set contains the specified status flag from OIAStatus constants.</p> <p>Parameters</p> <p>{Number} statusFlag status to check</p> <p>Returns</p> <p>{Boolean} True if the status flag is present in the set.</p>
<code>isEmpty()</code>	<p>Determines if the status set is empty.</p> <p>Returns</p> <p>{Boolean} True if the set is empty.</p>
<code>size()</code>	<p>Indicates the number of status flags in the set.</p> <p>Returns</p> <p>{Number} The status count</p>
<code>toArray()</code>	<p>Converts the internal status set to an array.</p> <p>Returns</p> <p>{Object []} Array of status flags in the set.</p>
<code>toString()</code>	<p>Converts the internal status set to a string.</p> <p>Returns</p> <p>{String} Space delimited names of status flags in the set.</p>
<code>forEach(callback, thisArg)</code>	<p>Function to iterate over each element in the status set.</p> <p>Parameters</p> <p>{forEachCallback} Callback to perform the specific operation. Called with the name of each status in the set.</p>


```
forEachCallback(string,  
thisArg)
```

A user-provided callback function where you provide the behavior, to be used as the callback parameter to `forEach`.

Parameters

`{String} String` The name of a status in the status set.

`{Object} thisArg` Optional pointer to a context object.

User Interface

The user interface object provides functions for interacting with the user, prompting for and displaying basic information. The UI object is made automatically available in your macro as the “ui” variable”.

note

Important: All UI functions require the ‘yield’ keyword in front of them. This allows the macro to block execution until the conditions of the UI function have been met.

`[parameter]` denotes an optional parameter.

Method	Description
<pre>prompt(message, [defaultAnswer], [mask])</pre>	<p>Prompts the user for information in the user interface.</p> <p>Parameters</p> <p><code>{String} message</code> title to display to the user. Default: blank String.</p> <p><code>{String} defaultAnswer</code> to use if user leaves it blank. Default: blank String</p> <p><code>{Boolean} mask</code> indicates whether to hide the prompt (as with a password)</p> <p>Returns</p> <p><code>{Promise}</code> Fulfilled when the user closes the dialog window. Returns the users input on “OK” or null on “Cancel”.</p>

```
message([message])
```

Display a message in the user interface.

Parameters

{String} message to display to the user.

Default: blank String.

Returns

{Promise} Fulfilled when the user closes the message window.

Wait

Use the wait object to wait for a particular session or screen state. For example, you can wait until the cursor is found at a particular location or text is present at a certain location before continuing with the macro execution.

Wait functions are often used in conjunction with asynchronous functions such as `connect()` and `sendKeys()`.

note

All functions take timeouts as an optional parameter and have a default time out value of 10 seconds (10000ms).

Important: All wait functions require the 'yield' keyword in front of them. This allows the macro to block execution until the conditions of the wait function are met.

[parameter] denotes an optional parameter.

Method	Description
<code>setDefaultTimeout(timeout)</code>	<p>Sets the default timeout value for all functions.</p> <p>Parameters</p> <p>{Number} default timeout to use for all wait functions in milliseconds.</p> <p>Returns</p> <p>{None}</p> <p>Throws</p> <p>{RangeError} If the specified timeout is less than zero.</p>

<pre>forConnect([timeout])</pre>	<p>Waits for a connect request to complete.</p> <p>Parameters</p> <p>{Number} in milliseconds.</p> <p>Returns</p> <p>{Promise} Fulfilled if the session is already connected or when connection occurs. Rejected if the wait times out.</p>
<pre>forDisconnect([timeout])</pre>	<p>Waits for a disconnect request to complete.</p> <p>Parameters</p> <p>{Number} timeout in milliseconds.</p> <p>Returns</p> <p>{Promise} Fulfilled if the session is already disconnected or when it finally disconnects. Rejected if the wait times out.</p>
<pre>forFixedTime([timeout])</pre>	<p>Waits unconditionally for fixed time. Time is in milliseconds (ms).</p> <p>Parameters</p> <p>{Number} timeout in milliseconds.</p> <p>Returns</p> <p>{Promise} Fulfilled after time elapses.</p>
<pre>forScreenChange([timeout])</pre>	<p>Waits for the host screen to change. This function returns when a screen update is detected. It makes no guarantees about the number of subsequent updates that may arrive before the screen is complete. Waiting repeatedly until the screen contents match some known stopping criteria is advisable.)</p> <p>Parameters</p> <p>{Number} timeout in milliseconds.</p> <p>Returns</p> <p>{Promise} Resolved if the screen change. Rejected if the wait times out.</p>
<pre>forCursor(position, [timeout])</pre>	<p>Waits for the cursor to arrive at the specified position.</p> <p>Parameters</p> <p>{Position} The position specifying the row and column</p> <p>Returns</p> <p>{Promise} Fulfilled if the cursor is already located or when it is finally located. Rejected if the wait times out.</p>

```
forText(text, position,  
[timeout])
```

Wait for text located at a specific position on the screen

Parameters

{String} text to expect
{Position} position specifying the row and column
{Number} timeout in milliseconds

Returns

{Promise} Fulfilled if the text is already at the specified position or whenever it is located. Rejected if the wait times out.

Throws

{RangeError} if the position is not valid.

```
forHostPrompt(text, column,  
[timeout])
```

Waits for a command prompt located at a particular column on the screen.

Parameters

{String} text prompt to expect
{Number} column where cursor is expected
{Number} timeout in milliseconds.

Returns

{Promise} Fulfilled if the conditions are already met or when the conditions are finally met. Rejected if the wait times out.

Throws

{RangeError} if the column is out of range.

```
forHostSettle([settleTime],  
[timeout])
```

NOTE: `wait.forHostSettle` should only be used when other more targeted wait functions are insufficient.

Monitors incoming screen data and resolves `settleTime` ms after the last update **and** the keyboard is unlocked. This function is useful when data arrives in multiple packets and you want to be sure the whole screen has been received before carrying on.

Parameters

{Number} time to wait after the last update to make sure more data doesn't arrive unexpectedly. The default is 200 milliseconds.

{Number} timeout in milliseconds.

Returns

{Promise} Fulfilled when the settle time has elapsed after receipt of the last screen update and the keyboard is unlocked.

Sample Macros

To help you create successful macros that take advantage of all the capabilities of the Macro Editor, these samples are available as a starting point.

Basic Host Interaction

This sample illustrates basic host interaction, including:

Sending data to the host

Waiting for screens to display

Using the `yield` keyword to wait for asynchronous functions

Reading text from the screen

Displaying basic information to the user

Handling error basics

All macros have the following objects available by default:

1. **session** - main entry point for access to the host. Can connect, disconnect and provides access to the `PresentationSpace`.

The PresentationSpace object obtained from the session represents the screen and provides many common capabilities, such as getting and setting the cursor location, sending data to the host, and reading from the screen.

2. **wait** - provides a simple way to wait for various host states before continuing to send more data or read from the screen.
3. **UI** - provides basic user interface capabilities. Display data to the user or prompt them for information.

```

// Create a new macro function
var macro = createMacro(function*(){
'use strict';

// All macros have the following objects available by default:
// 1. session - Main entry point for access to the host. Can connect, disconnect and provides access to
the PresentationSpace.
// The PresentationSpace object obtained from the session represents the screen and provides many
common capabilities such as getting and setting the
// cursor location, sending data to the host and reading from the screen.
// 2. wait - Provides a simple way to wait for various host states before continuing to send more data
or read from the screen.
// 3. ui - Provides basic User Interaction capabilities. Display data to the user or prompt them for
information.

// Declare a variable for reading and displaying some screen data.
// As a best practice all variables should be declared near the top of a function.
var numberOfAccounts = 0;

// Start by obtaining the PresentationSpace object, which provides many common screen operations.
var ps = session.getPresentationSpace();

try {
// Can set and get the cursor location
ps.setCursorPosition(new Position(24, 2));

// Use the sendKeys function to send characters to the host
ps.sendKeys('cics');

// SendKeys is also used to send host keys such as PA and PF keys.
// See "Control Keys" in the documentation for all available options
ps.sendKeys(ControlKey.ENTER);

// Wait for the cursor to be at the correct position.
// The wait object provides various functions for waiting for certain states to occur
// so that you can proceed to either send more keys or read data from the screen.
yield wait.forCursor(new Position(24, 2));

// You can mix characters and control keys in one sendKeys call.
ps.sendKeys('data' + ControlKey.TAB + ControlKey.TAB + 'more data' + ControlKey.ENTER);

// The "yield" keyword must be used in front of all "wait" and "ui" function calls.
// It tells the browser to pause execution of the macro until the
// (asynchronous) wait function returns. Consult the documentation for which functions
// require the yield keyword.
yield wait.forCursor(new Position(10, 26));
ps.sendKeys('accounts' + ControlKey.ENTER);

// Can also wait for text to appear at certain areas on the screen
yield wait.forText('ACCOUNTS', new Position(3, 36)) ;
ps.sendKeys('1' + ControlKey.ENTER);

// All wait functions will timeout if the criteria is not met within a time limit.
// Can increase timeouts with an optional parameter in the wait functions (in milliseconds)
// All timeouts are specified in milliseconds and the default value is 10 seconds (10000ms).
yield wait.forCursor(new Position(1, 1), 15000);
ps.sendKeys('A' + ControlKey.ENTER);

// PS provides the getText function for reading text from the screen
numberOfAccounts = ps.getText(new Position(12, 3), 5);

// Use the ui object to display some data from the screen
ui.message('Number of active accounts: ' + numberOfAccounts);

// The try / catch allows all errors to be caught and reported in a central location
} catch (error) {
// Again we use the ui object to display a message that an error occurred
yield ui.message('Error: ' + error.message);
}
//End Generated Macro
});

```

```
// Run the macro and return the results to the Macro Runner
// The return statement is required as the application leverages
// this to know if the macro succeeded and when it is finished
return macro();
```

User Interaction

This sample illustrates how to use the provided API methods to prompt the user for input or alert them with a message.


```

var macro = createMacro(function*(){
  'use strict';

  // The "ui" object provides functions for prompting the user for information and displaying information

  // Declare variables for later use
  var username;
  var password;
  var flavor;
  var scoops;

  //Begin Generated Macro
  var ps = session.getPresentationSpace();

  try {
    // Prompt the user to enter their name and store it in a variable.
    // Note that 'yield' keyword is needed to block execution while waiting for the user input.
    username = yield ui.prompt('Please enter your username');

    // Prompt the user to enter a value with a default provided to them.
    flavor = yield ui.prompt('What is your favorite flavor of ice cream?', 'Chocolate');

    // Prompt the user to enter private information by using the 'mask' option and the input field will be masked as they type.
    // If a parameter is not used, 'null' can be used to specify that it isn't to be used.
    // Here we illustrate that by specifying that we don't need to show a default value .
    password = yield ui.prompt('Please enter your password', null, true);

    // The prompt function returns null if the user clicks the 'Cancel' button instead of the 'OK' button.
    // One way to handle that case is to wrap the call in a try/catch block.
    scoops = yield ui.prompt('How many scoops would you like?');
    if (scoops === null) {
      // This will exit the macro.
      return;
      // Alternatively could throw an Error and have it be caught in the "catch" below
    }
    // Use the collected values to order our ice cream
    ps.sendKeys(username + ControlKey.TAB + password + ControlKey.ENTER);
    yield wait.forCursor(new Position(5, 1));
    ps.sendKeys(flavor + ControlKey.TAB + scoops + ControlKey.ENTER);

    // Display a message to the user. Using the 'yield' keyword in front of the call will block
    // further execution of the macro until the user clicks the 'OK' button.
    yield ui.message('Order successful. Enjoy your ' + scoops + ' scoops of ' + flavor + ' ice cream ' + username + '!');
  } catch (error) {
    // Here we use the ui object to display a message that an error occurred
    yield ui.message(error.message);
  }
  //End Generated Macro
});

return macro();

```

Paging Through Data

This sample illustrates how to page through a variable number of screens and process the data on each screen.

```

// Create a new macro function.
var macro = createMacro(function*(){
  'use strict';

  // Create variable(s) for later use
  var password;
  var accountNumber;
  var transactionCount = 0;
  var row = 0;

  // Obtain a reference to the PresentationSpace object.
  var ps = session.getPresentationSpace();

  try {
    // Enter Username and Password to log on to the application.
    yield wait.forCursor(new Position(19, 48));
    ps.sendKeys('bjones' + ControlKey.TAB);

    yield wait.forCursor(new Position(20, 48));
    password = yield ui.prompt('Password:', null, true);
    ps.sendKeys(password);
    ps.sendKeys(ControlKey.ENTER);

    // Enter an application command.
    yield wait.forCursor(new Position(20, 38));
    ps.sendKeys('4');
    ps.sendKeys(ControlKey.ENTER);

    // Going to list transactions for an account.
    yield wait.forCursor(new Position(13, 25));
    ps.sendKeys('2');
    // Input an account number. Hard coded here for simplicity.
    yield wait.forCursor(new Position(15, 25));
    accountNumber = yield ui.prompt('Account Number:', '167439459');
    ps.sendKeys(accountNumber);
    ps.sendKeys(ControlKey.ENTER);

    // Wait until on account profile screen
    yield wait.forText('ACCOUNT PROFILE', new Position(3, 33));

    // Search for text that indicates the last page of record has been reached
    while (ps.getText(new Position(22, 12), 9) !== 'LAST PAGE') {

      // While the last page of record has not been reached, go to the next page of records.
      ps.sendKeys(ControlKey.PF2);
      yield wait.forCursor(new Position(1, 1));

      // If the cursor position does not change between record screens, and there is no text
      // on the screen you can check to confirm a screen is updated, you may wait for a
      // fixed time period after an aid key is sent for the screen to settle.
      // For example:
      // yield wait.forFixedTime(1000);

      // For each of the rows, increment the count variable if it contains data.
      for (row = 5; row <= 21; row++) {

        // There are 2 columns on the screen. Check data on column 1.
        // In this example we know that if there is a space at a particular
        // position then there is a transaction.
        if (ps.getText(new Position(row, 8), 1) !== ' ') {
          transactionCount++;
        }
        // Check data on column 2.
        if (ps.getText(new Position(row, 49), 1) !== ' ') {
          transactionCount++;
        }
      }
    }

    // After going through all record pages, display the number of records in a message box.
    yield ui.message('There are ' + transactionCount + ' records found for account ' + accountNumber + '.');

    // Log out of the application
    ps.sendKeys(ControlKey.PF13);
    ps.sendKeys(ControlKey.PF12);

    // The try / catch allows all errors to be caught and reported in a central location
  } catch (error) {
    // Here we use the ui object to display a message that an error occurred
    yield ui.message(error.message);
  }
});

// Here we run the macro and return the results to the Macro Runner
// The return statement is required as the application leverages
// this to know if the macro succeeded
return macro();

```

Invoking a Web Service

This sample illustrates how to make an AJAX / REST call directly from a macro to a web service. You can integrate data from your host application into the web service call or from the web service into your host application.

In this example, we are calling the Verastream Host Integrator (VHI) CICSAcctsDemo REST service. However, you can easily adapt the code to call any web service. You are not limited to VHI.

In the example, the call goes through a proxy configured in the session server (shown below) to avoid a "Same Origin Policy" complication. If you are using a web service that supports [Cross-origin Resource Sharing \(CORS\)](#) and are using a modern browser, the proxy is unnecessary.

Since the jQuery library is available in macros, you may use the `$.post()` function directly to invoke REST services.

This example also demonstrates how to wrap a jQuery REST call in a new Promise. The promise returned from the custom function below allows "yield" to be used in the main macro code. This allows the main macro execution to wait until the service call is complete before continuing.

```

var macro = createMacro(function*() {
  'use strict';

  // Create a few variables for later user
  var username;
  var password;
  var accountNumber;
  var accountDetails;

  // Create a function that will make an AJAX / REST call to a VHI Web Service.
  // Could be adjusted to call any web service, not just VHI.
  // If not using CORS, the request will likely need to pass through a
  // proxy on the session server. See sample notes for more information.
  /**
   * Hand-coded helper function to encapsulate AJAX / REST parameters, invoke the
   * REST service and return the results inside a Promise.
   * @param {Number} acctNum to send to the REST query.
   * @param {String} username to access the REST service.
   * @param {String} password to access the REST service.
   * @return {Promise} containing $.post() results that are compatible with yield.
   */
  var getAccountDetails = function (acctNum, username, password) {
    var url = "proxy1/model/CICSAcctsDemo/GetAccountDetail";
    var args = {"filters": {"AcctNum": acctNum}, "envVars": {"Username": username, "Password": password}};

    // Wrap a jQuery AJAX / HTTP POST call in a new Promise.
    // The promise being returned here allows the macro to yield / wait
    // for its completion.
    return Promise.resolve($.post(url, JSON.stringify(args)))
      .catch(function (error) {
        // Map errors that happen in the jQuery call to our Promise.
        throw new Error('REST API Error: ' + error.statusText);
      });
  };

  // Begin Generated Macro
  var ps = session.getPresentationSpace();
  try {
    // Could interact with the host here, log into a host app, etc...
    // Gather username and password
    username = yield ui.prompt('Username:');
    password = yield ui.prompt('Password:', null, true);
    accountNumber = yield ui.prompt('Account Number:');
    if (!username || !password || !accountNumber) {
      throw new Error('Username or password not specified');
    }

    // Invoke external REST service, and yields / waits for the call to complete.
    accountDetails = yield getAccountDetails(accountNumber, username, password);

    // We now have the data from our external service.
    // Can integrate the data into our local host app or simply display it to the user.
    // For this sample we simply display the resulting account details.
    if (accountDetails.result && accountDetails.result.length > 0) {
      yield ui.message(accountDetails.result[0].FirstName + ' $' + accountDetails.result[0].AcctBalance);
    } else {
      yield ui.message('No records found for account: ' + accountNumber);
    }
  } catch (error) {
    // If an error occurred during the AJAX / REST call
    // or username / password gathering we will end up here.
    yield ui.message(error.message);
  }
});

// Run our macro
return macro();

```

Cross Origin Scripting Proxy Support

If you have web services that do not support CORS, any AJAX/REST calls will fail if they attempt to access a server other than the one where the Host Access for the Cloud application originated. This is a browser security feature.

The Host Access for the Cloud server provides a way to explicitly proxy to trusted remote servers.

- Open `..\install_dir\sessionserver\microservice\sessionserver\service.yml` for editing.
- In the `env` section add:

```
name: zfe.proxy.mappings
value: proxy-path=proxy-to-address
```

Where `proxy-path` refers to the desired url-mapping and `proxy-to-address` refers to the URL where the call will be proxied.

- In this example:

```
name: zfe.proxy.mappings
value: proxy1=http://remote-vhi-server:9680/vhi-rs/
```

Calls made to `<server:port>/proxy1` will be proxied to `http://remote-vhi-server:9680/vhi-rs/`

- Multiple proxy mappings can be specified using a comma to separate the individual proxy mappings
- Keep in mind that even when a REST server supports CORS headers, some older browsers may not, so this example may still be relevant.

hint

Your `service.yml` file may be replaced whenever you redeploy Host Access for the Cloud. Always back up your files.

Working with Data Cells and Attributes

This macro illustrates how to use `DataCells` and `AttributeSet` to inspect a given row/column on the screen for text and attributes. In this sample you can see:

- How to get a collection of `DataCells` for a given position and length.
- How to iterate through `DataCells` to build up a text string
- How, for comparison, you can also do a similar thing using `getText()`.
- And finally, how to work with attributes, get a string listing, or determine whether specific ones are set at a given screen location.

```

var macro = createMacro(function*() {
  'use strict';

  // Obtain the PresentationSpace for interacting with the host
  var ps = session.getPresentationSpace();

  // Declare variables for later use
  var cells;
  var text;
  var attrs;

  // Set the default timeout for "wait" functions
  wait.setDefaultTimeout(10000);

  // Sample macro for working with DataCells and Attributes
  try {
    yield wait.forCursor(new Position(24, 2));

    // Get DataCells from the presentation space.
    // Row 19, col 3 is the prompt, 35 characters long
    // "Choose from the following commands:"
    cells = ps.getDataCells({row:19, col:3}, 35);
    text = '';

    // You can display text using getText
    yield ui.message("Screen text: " + ps.getText({row:19, col:3}, 35));

    // Or you can assemble the text from the DataCells at each position
    for(var index = 0; index < cells.length; index++) {
      text = text.concat(cells[index].getChar());
    }
    // And display the text
    yield ui.message("Cells text: " + text);

    // Get the attributes for the first DataCell (cell[0])
    attrs = cells[0].getAttributes();

    // Display whether we have any attributes on the data cell
    yield ui.message("Attribute set is empty: " + attrs.isEmpty());

    // Display how many attributes are set
    yield ui.message("Number of attributes: " + attrs.size());

    // Display which attributes are set
    yield ui.message("Attributes: " + attrs.toString());

    // Now display whether the high intensity attribute is set
    yield ui.message("Is high intensity: " +
      attrs.contains(Attribute.HIGH_INTENSITY));

    // Now display whether the underline attribute is set
    yield ui.message("Is underline: " +
      attrs.contains(Attribute.UNDERLINE));

    // Now display whether alphanumeric, intensified and pen-detectable attributes are set
    yield ui.message("Is alphanumeric, intensified and pen-detectable: " +
      attrs.containsAll([Attribute.ALPHA_NUMERIC, Attribute.HIGH_INTENSITY, Attribute.PEN_DETECTABLE]));

    // Now display whether underline, intensified and pen-detectable attributes are set
    yield ui.message("Is underline, intensified and pen-detectable: " +
      attrs.containsAll([Attribute.UNDERLINE, Attribute.HIGH_INTENSITY, Attribute.PEN_DETECTABLE]));
  } catch (error) {
    yield ui.message(error);
  }
  //End Generated Macro
});

// Run the macro
return macro();

```

Using Fields and Field Lists

This macro sample illustrates how to use common functions to interact with the fields in the Macro API. For example, how to get field text, view field information, and use `field.setText` as an alternative to `sendKeys` to interact with the host.

note

Due to browser considerations, `ui.message` collapses strings of spaces down to a single space. The spaces are preserved in the actual JavaScript.

```

var macro = createMacro(function*() {
  'use strict';

  // Obtain the PresentationSpace for interacting with the host
  var ps = session.getPresentationSpace();

  // Declare variables for later use
  var fields;
  var field;
  var searchString = 'z/VM';

  // Set the default timeout for "wait" functions
  wait.setDefaultTimeout(10000);

  // Sample macro for working with FieldList and Fields
  try {
    yield wait.forCursor(new Position(24, 2));

    // Get the field list.
    fields = ps.getFields();

    // Run through the entire list of fields and display the field info.
    for(var index = 0; index < fields.size(); index++) {
      field = fields.get(index);

      yield ui.message("Field " + index + " info: " + field.toString());
    }

    yield ui.message("Now, find a field containing the text '" + searchString + "'");
    field = fields.findField(new Position(1, 1), searchString);

    if(field !== null) {
      yield ui.message("Found field info: " + field.toString());
      yield ui.message("Found field foreground is green? " + (Color.GREEN === field.getForegroundColor()));
      yield ui.message("Found field background is default? " + (Color.BLANK_UNSPECIFIED === field.getBackgroundColor()));
    }

    // Now, find command field and modify it.
    field = fields.findField(new Position(23, 80));
    if(field !== null) {
      field.setText("cics");
    }

    yield ui.message("Click to send 'cics' to host.");
    ps.sendKeys(ControlKey.ENTER);

    // Wait for new screen; get new fields.
    yield wait.forCursor(new Position(10, 26));
    fields = ps.getFields();

    // Find user field and set it.
    field = fields.findField(new Position(10, 24));
    if(field !== null) {
      field.setText("myusername");
    }

    // Find password field and set it.
    field = fields.findField(new Position(11, 24));
    if(field !== null) {
      field.setText("mypassword");
    }

    yield ui.message("Click to send login to host.");
    ps.sendKeys(ControlKey.ENTER);

    // Wait for new screen; get new fields.
    yield wait.forCursor(new Position(1, 1));
    fields = ps.getFields();

    // Find command field and set logoff command.
    field = fields.findField(new Position(24, 45));
    if(field !== null) {
      field.setText("cesf logoff");
    }

    yield ui.message("Click to send logoff to host.");
    ps.sendKeys(ControlKey.ENTER);

  } catch (error) {
    yield ui.message(error);
  }
  //End Generated Macro
});

// Run the macro
return macro();

```

Automatic Sign-On Macro for Mainframes

In this example the Autosignon object is used to create a macro that uses the credentials associated with a user to obtain a pass ticket from the Digital Certificate Access Server (DCAS).

```
var macro = createMacro(function*() {
  'use strict';

  // Obtain the PresentationSpace for interacting with the host
  var ps = session.getPresentationSpace();

  // Variable for login pass ticket
  var passTicket;

  // Login application ID
  var appId = 'CICSV41A';

  // Set the default timeout for "wait" functions
  wait.setDefaultTimeout(10000);

  // Begin Generated Macro
  try {
    yield wait.forCursor(new Position(24, 2));

    // Obtain a pass ticket from DCAS.
    passTicket = yield autoSignon.getPassTicket(appId);

    ps.sendKeys('cics');
    ps.sendKeys(ControlKey.ENTER);

    yield wait.forCursor(new Position(10, 26));

    // Replace generated username with sendUserName(passTicket) ...
    yield autoSignon.sendUserName(passTicket);

    // ps.sendKeys('bvtst01' + ControlKey.TAB + ControlKey.TAB);
    ps.sendKeys(ControlKey.TAB + ControlKey.TAB);

    yield wait.forCursor(new Position(11, 26));

    // Replace generated password with sendPassword(passTicket) ...
    yield autoSignon.sendPassword(passTicket);

    // var userInput3 = yield ui.prompt('Password:', '', true);
    // if (userInput3 === null) {
    //   // throw new Error('Password not provided');
    // }
    // ps.sendKeys(userInput3);
    ps.sendKeys(ControlKey.ENTER);

    yield wait.forCursor(new Position(1, 1));
    yield ui.message('Logged in. Log me off.');
```

Using File Transfer (IND\$File)

This series of sample macros demonstrate how to use the File Transfer API to retrieve a list of files, download a file, and upload a file to a 3270 host.

note

You must be logged in and at a ready prompt before running these macros.

[List files](#)

[Download file](#)

List files

This macro demonstrates how to use the File Transfer API to retrieve a list of files on a 3270 host using IND\$File transfer. The IND\$File transfer object is retrieved from the file transfer factory and then used to obtain an array of HostFile objects from either TSO or CMS.

```
var macro = createMacro(function*() {
  'use strict';

  try {
    var fileTransfer = fileTransferFactory.getInd$File();
    var hostFiles = yield fileTransfer.getHostFileListing();

    yield ui.message('Found ' + hostFiles.length + ' files');
    if (hostFiles.length > 0) {
      var firstFile = hostFiles[0];
      var msg1 = 'The catalog name is ' + firstFile.getParent() + '. ';
      var msg2 = 'The first file is ' + firstFile.getName();
      yield ui.message(msg1 + msg2);
    }
  } catch (error) {
    yield ui.message(error);
  }
});

// Run the macro
return macro();
```

Download file

This macro shows how to use the File Transfer API to download a file from a 3270 host using IND\$File transfer. The IND\$File transfer object is retrieved from the file transfer factory. In this example, the transfer method is set to ASCII to demonstrate use of the setTransferOptions function.

The sample macro downloads the first file returned from a call to getHostFileListing by creating a download URI with a call to the getDownloadUrl function. The macro can be used in either a CMS or TSO environment but the choice must be specified on the first line or the code modified slightly for the intended system.

```

var hostEnvironment = 'CMS'; // 'TSO'
// Construct file path, ie catalog/file.name or catalog/partition/file
function getPath (fileNode) {
  var prefix = fileNode.getParent() ? fileNode.getParent() + '/' : '';
  return prefix + fileNode.getName();
}

var macro = createMacro(function*() {
  'use strict';

  try {
    var fileTransfer = fileTransferFactory.getInd$File();

    // The transferMethod options are 'binary' and 'ascii'
    fileTransfer.setTransferOptions({transferMethod: 'ascii'});

    // This demo retrieves the first file returned in the listing
    var hostFiles = yield fileTransfer.getHostFileListing();
    var firstHostFile = hostFiles[0];

    if (hostEnvironment === 'CMS') {
      yield wait.forText('Ready', new Position(1,1), 5000);
    }

    // Download
    // If you already know the path of the file you want, just pass that to getDownloadURL()
    var downloadUrl = fileTransfer.getDownloadURL(getPath(firstHostFile));

    // This changes the browser location. You may experience different results on different browsers
    window.location = downloadUrl;

    // If you want to read the file contents into a variable instead of downloading
    // it, you can use jQuery
    // var fileContents = yield $.get(downloadUrl);

  } catch (error) {
    yield ui.message(error);
  }
});

// Run the macro
return macro();

```

Upload file

This macro illustrates how to use the File Transfer API to upload a file to a 3270 host using IND\$File transfer. The sample macro prompts the user to choose a file from the local file system by triggering the browser's file selection dialog. It then retrieves the current catalog on TSO or drive identifier on CMS by calling `getHostFileListing`. Finally, the `sendFile` function is called to deliver the selected local file to the host.

The macro can be used in either a CMS or TSO environment but the choice should be specified on the first line. In this example, the transfer method is set to **ascii**; you may want to change this to **binary**.

```

var hostEnvironment = 'CMS'; // 'TSO'
// Open the browser's file chooser dialog programmatically
function promptForFileToUpload () {
  return new Promise(function (resolve, reject) {
    // We are not notified if the user cancels the file chooser dialog so reject after 30 seconds
    var timerId = setTimeout(reject.bind(null, 'Timed out waiting for file selection'), 30000);
    var fileSelector = document.createElement('input');
    fileSelector.setAttribute('type', 'file');
    fileSelector.onchange = function (evt) {
      var file = evt.target.files[0];
      clearTimeout(timerId);
      resolve(file);
    };
    fileSelector.click();
  });
}

var macro = createMacro(function*() {
  'use strict';

  try {
    var fileTransfer = fileTransferFactory.getInd$File();

    // The transferMethod options are 'binary' and 'ascii'
    fileTransfer.setTransferOptions({transferMethod: 'ascii'});

    var localFile = yield promptForFileToUpload();

    // Retrieve the current catalog name and append the selected file name to it
    var hostFiles = yield fileTransfer.getHostFileListing();
    var destination = hostFiles[0].getParent() + '/' + localFile.name;

    if (hostEnvironment === 'CMS') {
      yield wait.forText('Ready', new Position(1,1), 5000);
    }

    var result = yield fileTransfer.sendFile(localFile, destination);

  } catch (error) {
    yield ui.message(error);
  }
});

// Run the macro
return macro();

```

Run Macro on Event

Use the Macro panel to select which macros to run and when to run them.

- Run macro on startup - Choose a macro to run automatically when the session is opened.
- Run macro on connect - Choose a macro to run automatically when the session connects to the host.
- Run macro on disconnect - Choose a macro to run automatically when the session disconnects from the host.

More information

[Creating Macros](#)

[Using the Macro API](#)

[Sample Macros](#)

Display Settings

Display settings vary depending on the host type and are specific to the session you are configuring.

Color Mapping

You can customize the color of your screen and the appearance of different host attributes in the terminal window. For each item, you can select a color for the foreground and the background colors for all supported host connections. Colors are specified using the color grid or by entering the Hex code format.

There are many web sites that list available Hex colors, for an example see [w3schools.com HTML Color Picker](https://www.w3schools.com/html/html_color_picker.asp).

You may see different options depending on the type of host connection.

Options specific to UTS hosts

- **Use color information from the host** - To use the colors specified here rather than any colors specified by the host, clear this option.

- **Enable blink** - To disable blinking, clear this option.

- **Select attribute to edit** - In UTS emulation, colors are set directly by the host. You can specify colors for text associated with specific screen display options. Including the following and available combinations:

Plain, Underline(UND), Strikethru (STK), Left Column Separator (LCS), Control Page, and Status Line (OIA).

- **Video intensity** - The video intensities, Blink, Dim, Protected, and Reverse are combined with the attributes to create additional combinations. For example, you could map foreground or background colors to all cells with Dim + Blink + Underline or Reverse + Protected + Strikethru + Underline.

When you select a video intensity (or combination of intensities), those intensities are combined with the value of the attribute drop down list to form a single color mapping.

Options specific to VT and T27 hosts

- **Enable blink** - To disable blinking, clear this option.
- **Enable bold** - Displays text set with bold attributes as bold text in the terminal window. To display bold characters as plain text, clear this option.
- **Enable underline** - Displays text with underline.
- **Inverse video (VT-only)** - This option reverses the foreground and background colors when the VT host sends an inverse video escape sequence. If this option is not enabled, the inverse video sequences sent from the host are ignored.

To customize colors for all host types

1. From the left navigation panel, click Display.
2. Under Color Mappings, click the background color field to open the color grid. From the color grid, select the color you want to use as the host background color. Alternatively, type the Hex color number for the color you want to use.
3. From the drop down list, select the default host color you want to change. For example, if you select host pink from the drop down list and then change the foreground color to red, whenever you encounter pink text it will appear as red.
4. Open the color grid for the Foreground to choose a color to map the new color for the text or type the Hex code you want to use. Select Background to map the new color to the background field.
5. Click Save to close the Display panel and resume configuring your host connection.

Restore defaults clears any changes you made and resets the colors to the default host settings.

Configure Hotspots

Hotspots are buttons that appear over common host commands in terminal sessions. When you use hotspots you can control the terminal session using a mouse or a finger-tap instead of the keyboard. The hotspot transmits a terminal key or command to the host. By default, hotspots are configured for the most common 3270, 5250, and VT commands.

Hotspots are enabled and visible by default, however you can disable hotspots for a particular session or choose to hide them.

- Enable hotspots
Choose No to disable hotspots for the session you are connecting to.
- Show hotspots

Choose No to hide hotspots on the screen. Hotspots remain functional.

Hotspots for 3270 hosts

Hotspot	Description
PF1...PF24	Transmits a PF1...PF24 to the host
PA1, PA2, or PA3	Transmits a PA1, PA2, or PA3 to the host
enter	Transmits an Enter key to the host
more	Transmits a Clear key to the host

Hotspots for 5250 hosts

Hotspot	Description
enter	Transmits an Enter key to the host
more	Transmits a Roll Up key to the host (scrolls down one page)
PF1...PF24	Transmits a PF1...PF24 to the host

Hotspots for VT hosts

Hotspot	Description
F1...F20	Transmits a F1...F20 to the host

Configure screen dimensions for VT, UTS and T27 hosts

As an administrator you can select the number of columns and rows for VT, UTS and T27 sessions.

1. Open the Display panel.
2. Under Dimensions, specify the number of columns and rows you want each screen to possess. The default values are 80 columns by 24 rows.
There are some host-specific settings available:
 - Pages - If you are connecting to a T27 host screen, you can set the number of pages to display. The default is 2.

- Clear on host change - If you are connecting to a VT host screen, select this option to clear the terminal window and move the contents to the scrollbar buffer when the column size changes.

3. Click Save.

Set Cursor Options

Use the cursor options to configure the appearance and behavior of the cursor and ruler.

This option	Does this...
Cursor type	Underline displays the text cursor as an underline. Vertical bar displays the cursor as a vertical line. Block displays the text cursor as an inverse video block.
Ruler type	Vertical displays a vertical ruler at the cursor position. Horizontal displays a horizontal ruler at the cursor position. Crosshair displays both a horizontal and vertical ruler at the cursor position.
Cursor color	Click the color field to open the color grid. From the color grid, select the color you want to use as the color of both the cursor and ruler. Alternatively, type the Hex color number for the color you want to use.
Cursor blinks	By default, the cursor (whether in block or underline mode) blinks. Clear this option to display a visible non-blinking cursor.

Set Font Options

Use these font options to make sure that your terminal characters display with your preferred font size and style.

This option	Does this...
Font size	Auto (default) - The font scales automatically according to the size of the window. With this option selected, you can choose to Preserve the aspect ratio which means that the font size will be adjusted dynamically but the terminal display will not be stretched or scaled to fill the available space. Fixed Specify the size, in pixels, for the terminal window display.

Zero character	To differentiate the default zero character from the letter O, select one of the following options: Default Zero with a slash Zero with a dot
----------------	--

Set VT Scrollback Buffer Options

The VT scrollback buffer contains the data that has scrolled off the display and is no longer accessible by the host computer. When a scrollback buffer exists, you can view it by using the vertical scroll bar.

The scrollback buffer is enabled by default. When enabled, the session maintains a buffer of lines that have scrolled off the terminal screen. This option is available to all users when they are granted permission to modify Terminal Display Settings by the administrator.

This option	Does this...
Scrollback row limit	Limits the number of rows held in the scrollback buffer. The default setting is 500 rows.
Save display before clearing	When selected (the default), the data on the terminal display moves into the scrollback buffer when you, or the host, clear the terminal display. If you prefer not to have the terminal display saved to the scrollback buffer, clear this option; when the terminal display is cleared, the data is discarded
Save from scrolling regions	When top and bottom screen margins are set (for example, by a text editor such as EDT or TPU, or with the DECSTBM function) the area within the margins is called the scrolling region. When this option is cleared, scrolling text within this region isn't saved to the scrollback buffer. Select this option to save information within scrolling regions to the scrollback buffer. Note: This can cause display memory to fill quickly.
Save before clearing from any row	This setting specifies whether data that has been cleared from a portion of the terminal window is saved in display memory.

Compress
blank rows

Select this option to save room in display memory by compressing multiple blank rows into a single blank row.

Set Keyboard Options

You can set the following keyboard options:

[3270 options](#)

[5250 options](#)

[VT options](#)

[T27 options](#)

3270 options

- Typeahead

When this option is selected, Host Access for the Cloud buffers the characters that you type in the terminal window. Typeahead allows you to keep typing after you send data to the host. Without typeahead, characters you type are ignored until the host is ready for more data.

- Word wrap

When this option is selected, word wrap functionality is enabled within a multi-line, unprotected field. In word wrap mode, some of the blank spaces between words are replaced by line breaks so that each line is visible in the terminal window and can be read without horizontal scrolling.

- Attention key sends

Specifies what is sent when the ATTN key is pressed. The options are Telnet break, Abort output, and Interrupt process.

5250 options

- Typeahead

When this option is selected, Host Access for the Cloud buffers the characters that you type in the terminal window. Typeahead allows you to keep typing after you send data to the host. Without typeahead, characters you type are ignored until the host is ready for more data.

- Error auto reset

When selected, the next key pressed after a keyboard error clears the error, restores the previous error line data, and attempts to execute the keystroke as follows:

If the cursor is in a valid input field and the key is a data key, the data is entered there if it is valid data for that field (for example, a numeric character in an input field that only accepts numbers).

- If the cursor is in a valid input field and the key is a function key, the key operation is executed.
- If the current cursor position is not in a valid input field and the key is a data key, the cursor is moved to the next valid input field and the data is entered there if it is valid data for that field.
- If the current cursor position is not in a valid input field and the key is a function key, the cursor is moved to the next valid input field and the key is ignored.
- If the current screen contains no valid input fields, you'll see an error message with each keystroke you press, and no keystrokes are executed

When cleared, you must press Reset to clear the error message from the error line before you can resume data entry.

By default, this option is not selected.

- Waive field checks for PF key

Select this option to allow PF keys to be sent to the host from restricted fields. This option is cleared by default.

VT options

- Backspace sends

Configures the function that the Backspace key sends. On the VT terminal keyboard the back arrow key (<x) is configurable: it can send either a delete (ASCII 127) or a backspace (ASCII 8) character.

- Local echo (VT)

This option causes each character typed at the keyboard to be displayed on the screen. This option is cleared by default, because most hosts echo back received characters.

- Cursor keys

Controls the characters that the four arrow keys (on both the numeric and editing keypad) transmit. This setting is typically set by the host. In general, you should keep this set to Normal.

If the arrow keys aren't working properly, it may mean that this option remained incorrectly set to Application when a host program terminated abnormally. Changing this setting back to Normal should fix the problem with the arrow keys.

- Keypad

Controls the characters that the numeric keypad keys transmit. This setting is typically set by the host. In general, you should keep this set to Numeric.

If the number or PF keys aren't working properly, it may mean that this option was incorrectly left set to Application when a host program terminated abnormally. Changing this setting back to Numeric should fix the numeric keypad.

T27 options

- Enable lower case (T27)

Enables lower case, as well as upper case letters to be displayed on the screen. Default. If this option is disabled only upper case letters will display.

Terminal Settings

Terminal settings vary depending on your host type.

3270 and 5250 terminal settings

- Host character set

Select the 3270 or 5250 host character set you want to use. This setting chooses a conversion table to convert host characters (EBCDIC) into PC characters (ANSI). This setting should match the national character set used by your host system. If it doesn't match, then some characters, such as accents, may not display correctly. See your host documentation for definitions of the characters in each set. The default value is US English (037).

- Country extended graphics code (3270 only)

With this option selected (the default), additional characters are available in the configured National character set. See your host documentation for details.

VT terminal settings

- Terminal type (VT)

Specifies which terminal should be emulated. These choices affect the codes generated by the numeric keypad, the interpretation of control functions, and the response to terminal identification requests.

- Terminal ID (VT)

Specifies the response that Host Access for the Cloud sends to the host after a primary device attributes (DA) request. This response lets the host know what terminal functions it can

perform. This setting is independent of the terminal type setting. When set to the default value of Reflection, Host Access for the Cloud responds to a primary DA request with the set of features it supports. If your host requires a more specific terminal ID, select another value from the list.

- New line (VT)

Select this option to send both a carriage return and linefeed when you press Enter. When Host Access for the Cloud receives a linefeed, form feed, or vertical tab, it moves the cursor to the first column of the next line. When this option is cleared (default), the Enter key sends only a carriage return. A linefeed, form feed, or vertical tab received from the host moves the cursor down one line in the current column. If lines on the display keep getting overwritten (that is, the host is not sending a linefeed along with a carriage return), select this option. If the New line option is selected but the host does not expect to receive a linefeed with each carriage return, lines will be double-spaced on the display.

T27 terminal settings

- Host character set (T27)

Using this option you can specify host to screen translation. Select the language used to convert characters received from the host before they are displayed on the local machine. The default is No translation.

Set Other Display Options

This option	Does this...
Column separator style (5250)	Use this option to specify which character (if any) should be used to render column separators in 5250 terminal sessions. The options are: Dots- Dots are used to separate columns. The default. Vertical bars - Vertical lines are used to separate columns. None - No characters are used to separate columns
Input field underlining (3270, 5250)	You can determine how the underlining of host input fields is handled: Host controls underlining (Default) Always underline input fields Never underline input fields

Status line (VT)	<p>To enable a status line at the bottom of the display. None to disable the status line. (Default)</p> <p>Indicator to display the page, cursor position, and printer status.</p> <p>Host Writable to have the host application display information in the status line.</p>
Preserve aspect ratio	<p>Select this option to maintain the host screen aspect ratio regardless of the size of the browser window. Aspect ratio describes the proportional relationship between the width of an image and its height.</p>
Display OIA (3270, 5250)	<p>Select this option to display the operation and status messages in the Operator Information Area (OIA) at the bottom of the terminal window . By default, OIA display is enabled.</p>
Display status line (ALC)	<p>Turns on a status line at the bottom of the display.</p>
Ignore mouse click on window activation	<p>When a mouse click activates the terminal window, this option specifies whether actions such as updating the terminal cursor position, clearing a selection, or executing a hotspot are also performed. By default, these actions are not performed.</p>
Auto wrap (VT)	<p>When selected, characters automatically wrap at the right margin and continue on the next line. When cleared, characters do not wrap when they reach the right margin of the display. New characters overwrite the character at the right margin until a carriage return is entered.</p>

Map Keys

Map Keys

You can create keyboard shortcuts that perform any assignable action during a session. The Key Mappings settings page provides a view of the default keyboard map for each host type and the mapped custom keys, indicated in boldface type, for that session.

See [Host Keyboard Mapping](#) for the different host keyboard mappings.



Mapping keys as an administrator and as an end-user

There are a few differences in behavior between the administrator and the end-user when mapping keys.

- End users can only add or modify key mappings if they are granted permission by the administrator using the User Preference Rules panel.
- Any changes the administrator makes show up to the end user as indistinguishable from default host key mappings. Once granted permission, the person can modify, add or delete any mappings regardless of any administrator changes. However restoring key mappings only restores them back to the modified state created by the administrator for the current session.

Adding or modifying mapped keys

1. From the toolbar, click Settings.
2. From the left navigation pane, open the Key Mappings panel. The mapped keys for the host type you are connecting to are visible.
3. To add a new key mapping:

- Click . You can choose to type the key sequence you want to use or use the keyboard by toggling  between the two options.
- From the Action drop down list, select the action you want to associate with the key selection. If you select Send text, enter the string you want sent to the host in the Value field. Likewise, if you select Run Macro, choose the macro you want triggered by the keyboard shortcut. You must create the macro before you can map it to the Run Macro action.

The Send text action supports mapping characters with codes less than or equal to `0xFFFF` via Unicode escape sequences. The escape sequence begins with `\u` followed by exactly four hexadecimal digits. You can embed Unicode escape sequences in any string. For example, this embedded `\u0045` will be interpreted as this embedded E, since 45 is the hexadecimal code for the character E.

To pass Unicode escape sequences to the host, escape the sequence with a leading backslash. For example, to send the string literal `\u001C` to the host, map a key to `\\u001C`. Host Access for the Cloud will convert this to the string `\u001C` when that key is pressed and send the 6 characters of the resulting string to the host.



The Disable action renders the key inoperable. When pressed the key will not initiate any action. This differs from the Unmap action which removes the key mapping but preserves a browser short-cut if it is defined.

- Click the blue check mark to accept the mapping and add the key map to the session.

4. To modify an existing mapping:

Select the row containing the key you want to modify.



Follow the steps for adding a new key mapping, clicking  to save the new mapping. Alternatively, you can click away from the modified row and the change will be saved. All new and modified key maps are indicated in boldface type. You can restore the original key mapping at any time by clicking .

Filtering the list

The Filter field makes it easy to see just those mappings you are interested in. The filter is based on keywords and affects each column of the table. For example, if you enter Send text in the Filter field, only keys mapped to the Send text action are displayed.

Using the Show only modified mappings option lets you see only those mappings that have been previously modified.

Some things to remember:

- Mapping right and left modifier keys to individual actions

You can map the right and left modifier keys to individual actions. However when they are combined with other keys, there is no differentiation between the right and left keys. For example, Left-Alt can be mapped to Action-A while Right-Alt is mapped to Action-B, but Left-Alt + H will be stored as Alt+H and both Left-Alt+H and Right-Alt+H will be associated with a single mapped action.

- Key stroke combinations and copy/paste operations

Different key stroke combinations are also used for copy/paste operations. For example, on a VT host screen, Ctrl+ Shift + A initiates a Select All action. See Editing the Screen for a list of copy/paste key actions.

- Keyboard shortcuts and browsers

Browsers use keyboard shortcuts to save both time and mouse clicks. When mapping keystrokes it is important to keep this in mind. Handy Keyboard Shortcuts gives a brief

overview of the keyboard shortcuts used by different browsers. In most cases Host Access for the Cloud key mappings take precedence over browser key shortcuts. Occasionally, where this is not the behavior you want for a particular key combination, you can choose Unmap in the action list to unmap the short-cut. This lets the key event to pass through to the browser.

Host Keyboard Mapping

The following tables provide the default keys, key name, and key description for the different host keyboard mappings.

[IBM 3270 Keyboard Mapping](#)

[IBM 5250 Keyboard Mapping](#)

[VT Keyboard Mapping](#)

[UTS Keyboard Mapping](#)

[T27 Keyboard Mapping](#)

[ALC Keyboard Mapping](#)

IBM 3270 keyboard mapping

Key	Maps to	Description
Ctrl + F1	Attention	Sends the ATTENTION key to the host
Shift + Tab	Backtab	Moves the cursor to the previous unprotected field
Ctrl + F2	Clear	Clears the screen and sends the CLEAR key to the host
Alt + ArrowLeft	Cursor left double	Moves the cursor two positions to the left
Alt + ArrowRight	Cursor right double	Moves the cursor two positions to the right
Ctrl + F3	Cursor select	Simulates a lightpen select in the current field
Alt + Delete	Delete word	Deletes three characters from the current field
Ctrl + 5	Duplicate	Inserts the DUP character at the cursor location
Enter	Enter	Sends the ENTER key to the host

Key	Maps to	Description
End	Erase end of field	Erases all data from the cursor location to the end of the current field
Alt + F5	Erase input	Erases all data in all unprotected fields of the current screen
Ctrl + Alt + F	Field delimiter	Toggles whether field delimiters are displayed on screen
Ctrl + 6	Field mark	Inserts the Field Mark character at the cursor location
Home	Home	Moves the cursor to the first unprotected field on the screen
Insert	Insert	Toggles Insert mode
Shift + Enter	New line	Moves to the next unprotected field
Ctrl + 1	PA1	Sends the PA1 key to the host
Pageup	PA1	Sends the PA1 key to the host
Ctrl + 2	PA2	Sends the PA2 key to the host
Pagedown	PA2	Sends the PA2 key to the host
Ctrl + 3	PA3	Sends the PA3 key to the host
F1 - F10	PF1 - PF10	Sends the PF1, PF2...PF10 key to the host
Alt + 1 or F11	PF11	Sends the PF11 key to the host
Alt + 2 or F12	PF12	Sends the PF12 key to the host
Shift + F1	PF13	Sends the PF13 key to the host
Shift + F2	PF14	Sends the PF14 key to the host
Shift + F3	PF15	Sends the PF15 key to the host
Shift + F4	PF16	Sends the PF16 key to the host
Shift + F5	PF17	Sends the PF17 key to the host
Shift + F6	PF18	Sends the PF18 key to the host
Shift + F7	PF19	Sends the PF19 key to the host
Shift + F8	PF20	Sends the PF20 key to the host

Key	Maps to	Description
Shift + F9	PF21	Sends the PF21 key to the host
Shift + F10	PF22	Sends the PF22 key to the host
Alt3	PF23	Sends the PF23 key to the host
Shift + F11	PF23	Sends the PF23 key to the host
Alt4	PF24	Sends the PF24 key to the host
Shift + F12	PF24	Sends the PF24 key to the host
Ctrl + P	Print	Prints the contents of the screen to the printer
Escape	Reset	Resets keyboard error conditions
Ctrl + S	System request	Sends the SYSTEM REQUEST key to the host

IBM 5250 keyboard mapping

Key	Maps to	Description
Escape	Attention	Sends the ATTENTION key to the host
Ctrl + F2	Clear	Clears the screen and send the CLEAR key to the host
Ctrl + F3	Cursor select	Simulates a lightpen select in the current field
Ctrl + Backspace	Destructive backspace	Moves the cursor one position to the left
Ctrl + 5	Duplicate	Inserts the DUP character at the cursor location
Ctrl + End	End of field	Moves the cursor to the end of the field
End	Erase end of field	Erases all data from the cursor location to the end of the current field
Alt + End	Erase input	Erases all data in all unprotected fields of the current screen
Alt + F5	Erase input	Erases all data in all unprotected fields of the current screen
Ctrl + Enter	Field exit	Moves the cursor out of an input field

Key	Maps to	Description
KP + Subtract	Field exit minus	Moves the cursor out of a signed-numeric or numeric-only field, inserting a minus sign in the last position of a signed-numeric field, or changing the last position in a numeric-only field to an alphabetic character that tells the system that this field has a negative value.
Ctrl + Subtract	Field exit minus	Moves the cursor out of a signed-numeric or numeric-only field, inserting a minus sign in the last position of a signed-numeric field, or changing the last position in a numeric-only field to an alphabetic character that tells the system that this field has a negative value.
KP + Add	Field exit plus	In a signed-numeric field, this function moves the cursor to the next field, removing a minus sign if there is one in the last position. In a numeric-only field, this function moves the cursor to the next field, changing the last position to an alphabetic character that tells the system that this field has a positive value.
Ctrl + Add	Field exit plus	In a signed-numeric field, this function moves the cursor to the next field, removing a minus sign if there is one in the last position. In a numeric-only field, this function moves the cursor to the next field, changing the last position to an alphabetic character that tells the system that this field has a positive value.
Ctrl + 6	Field mark	Inserts the field mark character at the cursor location
Ctrl + H	Help	Sends the Help key to the host
Alt + F7	Hex mode	Places the terminal in Hex mode
Home	Home	Moves the cursor to the first unprotected field on the screen
Insert	Insert	Toggles Insert mode
Shift + Enter	New line	Moves to the next unprotected field
Ctrl + 1	PA1	Sends the PA1 key to the host
Ctrl + 2	PA2	Sends the PA2 key to the host
Ctrl + 3	PA3	Sends the PA3 key to the host
F1 - F11	PF1 - PF11	Sends the PF1, PF2....PF11 key to the host

Key	Maps to	Description
Alt + 1	PF11	Sends the PF11 key to the host
Alt + 2	PF12	Sends the PF12 key to the host
F12	PF12	Sends the PF12 key to the host
Shift + 1	PF13	Sends the PF13 key to the host
Shift + F2...F10	PF14...PF22	Sends the PF14....PF22 key to the host
Alt + 3	PF23	Sends the PF23 key to the host
Shift + F11	PF23	Sends the PF23 key to the host
Alt + 4	PF24	Sends the PF24 key to the host
Shift + F12	PF24	Sends the PF24 key to the host
Ctrl + P	Print	Prints the contents of the screen to the printer
Control	Reset	Resets the keyboard error conditions
Pageup	RollDown	Sends the RollDown key to the host
Pagedown	RollUp	Sends the RollUp key to the host
Ctrl + Home	Start of field	Moves the cursor to the start of the field
Ctrl + S	System request	Sends the SYSTEM REQUEST key to the host

VT keyboard mapping

Key	Maps to	Description
Ctrl + Cancel	Break	Sends the Break key to the host
Ctrl + Enter	Enter	Sends the Enter key to the host
Alt + F1	F1	Sends the F1 key to the host
Ctrl + F1...F10	F11...F20	Sends the F11...F20 key to the host
Home	Find	Sends the Find key to the host
F1	Hold	Sends the Hold Screen to the host

Key	Maps to	Description
Pause	Hold	Sends the Hold Screen to the host
Insert	Insert	Sends the Insert key to the host
Ctrl + Insert	Keypad 0	Sends the numeric keypad 0 key to the host
Ctrl + End	Keypad 1	Sends the numeric keypad 1 key to the host
Ctrl + ArrowDown	Keypad 2	Sends the numeric keypad 2 key to the host
Ctrl + Pagedown	Keypad 3	Sends the numeric keypad 3 key to the host
Ctrl + ArrowLeft	Keypad 4	Sends the numeric keypad 4 key to the host
Ctrl + Clear	Keypad 5	Sends the numeric keypad 5 key to the host
Ctrl + ArrowRight	Keypad 6	Sends the numeric keypad 6 key to the host
Ctrl + Home	Keypad 7	Sends the numeric keypad 7 key to the host
Ctrl + ArrowUp	Keypad 8	Sends the numeric keypad 8 key to the host
Ctrl + Pageup	Keypad 9	Sends the numeric keypad 9 key to the host
Ctrl + Alt-add	Keypad comma	Sends the numeric keypad Comma key to the host
Ctrl + add	Keypad minus	Sends the numeric keypad Minus key to the host
Ctrl + decimal	Keypad period	Sends the numeric keypad Period key to the host
Ctrl + Delete	Keypad period	Sends the numeric keypad Period key to the host
Ctrl + Alt + ArrowUp	Row up	In the scrollbar moves up a row
Ctrl + Alt + ArrowDown	Row down	In the scrollbar moves down a row
Pagedown	Next	Sends the Next Screen key to the host
Ctrl + Pause	PF1	Sends the PF1 key to the host
Ctrl + Divide	PF2	Sends the PF2 key to the host
Ctrl + Multiply	PF3	Sends the PF3 key to the host
Ctrl + Subtract	PF4	Sends the PF4 key to the host

Key	Maps to	Description
Pageup	Previous	Sends the Prev Screen key to the host
Delete	Remove	Sends the Remove key to the host
End	Select	Sends the Select key to the host
Shift + F6...F10	UDK 6...10	Sends the User Defined Key 6...10 to the host
Shift + Ctrl + F1...F10	UDK11...20	Sends the User Defined Key 11...20 to the host

UTS keyboard mapping

Key	Maps to	Description
F4	Clear Change Bit	Sends the CLEARCHANGE BIT key to the host
Keypad+Enter	Carriage Return	Sends a carriage return to the host
Ctrl + PageDown	Clear End of Display	Clears text from the cursor location to the end of the display
Ctrl+PageUp	Clear End of Display FCC	Clears all data (including FCC information) from the cursor to the end of the display
Ctrl+End	Clear End of Field	Clears text from the cursor location to the end of the field
Ctrl+Shift+end	Clear End of Line	Clears text from the cursor location to the end of the row
F7	Clear FCC	Clears the field control character
Ctrl+Home	Clear Home	Sends the CLEAR_HOME key to the host
Ctrl+H	Column Separator Right	Sends the COLUMN_SEP_RIGHT key to the host
Ctrl+F1	Control Page	Sends the CONTROL_PAGE key to the host
Keypad+2	Cursor Down	Moves the cursor one row down
Keypad+4	Cursor Left	Moves the cursor one column to the left
Keypad+6	Cursor Right	Moves the cursor one column to the right

Key	Maps to	Description
Keypad+8	Cursor Up	Moves the cursor one row up
Delete	Delete in Line	Sends the DELETE_IN_LINE key to the host
Ctrl+Delete	Delete in Page	Sends the DELETE_IN_PAGE key to the host
Ctrl+Shift+Delete	Delete Line	Deletes the row at the cursor location
Ctrl+ArrowDown	Duplicate Line	Duplicates the row at the cursor location
F8	Enable FCC	Enables the field control character
Keypad+-	End of Display and Transmit	Sends the EOD_AND_TRANSMIT key to the host
Shift+End	End of Field	Moves the cursor to the end of the field
End	End of Line	Moves the cursor to the end of the row
Ctrl+ArrowRight	End of Page	Moves the cursor to the end of the page
Shift+Space	Erase Character	Erases the character at the cursor location
Ctrl+Shift+E	Euro Character	Sends the Euro character to the host
Ctrl+1...Ctrl+9	F1...F9	Sends the F1...F9 key to the host
Ctrl+0	F10	Sends the F10 key to the host
Ctrl+-	F11	Sends the F11 key to the host
Ctrl+=	F12	Sends the F12 key to the host
Ctrl+Q	F13	Sends the F13 key to the host
Ctrl+W	F14	Sends the F14 key to the host
Ctrl+E	F15	Sends the F15 key to the host
Ctrl+R	F16	Sends the F16 key to the host
Ctrl+T	F17	Sends the F17 key to the host
Ctrl+Y	F18	Sends the F18 key to the host
Ctrl+U	F19	Sends the F19 key to the host
Ctrl+I	F20	Sends the F20 key to the host
Ctrl+O	F21	Sends the F21 key to the host

Key	Maps to	Description
Ctrl+P	F22	Sends the F22 key to the host
Shift+F3	FF	Sends a formfeed to the host
F9	Generate FCC	Generates a field control character
Home	Home	Moves the cursor to the first field in the display
Ctrl+Shift+Space	Insert in Line	Sends the INSERT_IN_LINE key to the host
I Ctrl+Space	Insert in Page	Sends the INSERT_IN_PAGE key to the host
Ctrl+Shift+Insert	Insert Line	Inserts a new row into display memory
Insert	Insert Mode	Toggles insert character mode
F5	Locate FCC	Disables the field control characters and moves to the first character of the next field to the right of the cursor
F3	Message Wait	Sends the MESSAGE_WAIT key to the host
Shift+F2	New Line	Moves the cursor to a new row
Keypad+Shift+2	Next Field	Moves the cursor to the next field
Keypad+Shift+4	Next Field	Moves the cursor to the next field
PageDown	Page Down	Sends the Page Down key to the host
PageUp	Page Up	Sends the Page Up key to the host
Keypad+Shift+6	Previous Field	Moves the cursor to the previous field
Keypad+Shift+8	Previous Field	Moves the cursor to the previous field
Clear	SOE Character	Sends the SOE character to the host
F12	SOE Character	Sends the SOE character to the host
Ctrl+Clear	Set Tab	Sends the SET_TAB key to the host
Ctrl+Tab	Set Tab	Sends the SET_TAB key to the host
Shift+Home	Start of Field	Moves the cursor to the start of the field
Ctrl+ArrowLeft	Start of Line	Moves the cursor to the start of the row
Ctrl+[System Mode	Sends the SYSTEM_MODE key to the host

Key	Maps to	Description
Ctrl+J	Toggle Column Separator	Toggles the column separator
Ctrl+F12	Toggle Message Wait Beep	Sends the TOGGLEMSGWAITBEEP key to the host
Ctrl+L	Toggle Strike Thru	Toggles strike thru mode
Ctrl+K	Toggle Underline	Toggles underline mode
Ctrl+Enter	Transmit	Transmits the contents of the display to the host
ScrollLock	Transmit	Transmits the contents of the display to the host
Keypad++	Transmit	Transmits the contents of the display to the host
Keypad+Ctrl+	Transmit	Transmits the contents of the display to the host
Escape	Unlock	Sends the UNLOCK key to the host
Ctrl+]	Workstation Mode	Sends the WORKSTATION_MODE key to the host

T27 keyboard mapping

Key	Maps to	Description
Backspace	Backspace	Moves the cursor one column to the left
Shift+tab	back tab	Moves the cursor to the previous field
Ctrl+Delete	Clear End of Line	Clears text from the cursor location to the end of the row
Shift+Home	Clear Page Home	Clears the page and homes the cursor
Left Ctrl	Control Page	Puts the session in control mode
Down arrow	Cursor Down	Moves the cursor one row down

Key	Maps to	Description
Left arrow	Cursor Left	Moves the cursor one column to the left
Right arrow	Cursor Right	Moves the cursor one column to the right
Up arrow	Cursor Up	Moves the cursor one row up
Ctrl+left arrow	Cursor Word Left	Moves the cursor to the previous word
Ctrl+right arrow	Cursor Word Right	Moves the cursor to the next word
Ctrl+D	Delete Line	Deletes the row at the cursor location
Ctrl+End	End of Line	Moves the cursor to the end of the row
End	End of Page	Moves the cursor to the last field on the page
Shift+Ctrl+E	Euro Character	Sends a Euro character to the host
Home	Home	Moves the cursor to the first field in the display
Insert	Insert Mode	Puts the session in insert mode
Ctrl+I	Insert Line	Inserts a new row into display memory
Ctrl+1	PF1	Sends the PF1 key to the host
Ctrl+10	PF10	Sends the PF10 key to the host
Ctrl+2	PF2	Sends the PF2 key to the host
Ctrl+3	PF3	Sends the PF3 key to the host
Ctrl+4	PF4	Sends the PF4 key to the host
Ctrl+5	PF5	Sends the PF5 key to the host
Ctrl+6	PF6	Sends the PF6 key to the host
Ctrl+7	PF7	Sends the PF7 key to the host
Ctrl+8	PF8	Sends the PF8 key to the host
Ctrl+9	PF9	Sends the PF9 key to the host
PageDown	Page Down	Displays the next page
PageUp	Page Up	Displays the previous page

Key	Maps to	Description
Ctrl+E	Put ETX	Inserts an end-of-text character and homes the cursor
Keypad /	Put Local	Puts the session in local mode
Keypad *	Put Receive	Puts the session into receive mode
Enter	Return	Sends the return key to the host
Keypad Enter	Return	Sends the return key to the host
Ctrl+A	Select All	Selects all text
Shift+down arrow	Select Down	Selects text down
Shift+left arrow	Select Left	Selects text left
Shift+right arrow	Select Right Selects text right	Selects text right
Shift+up arrow	Select Up	Selects text up
Shift+Ctrl+1	Shift F1	Sends the Shift F1 key to the host
Shift+Ctrl+0	Shift F10	Sends the Shift F10 key to the host
Shift+Ctrl+2	Shift F2	Sends the Shift F2 key to the host
Shift+Ctrl+3	Shift F3	Sends the Shift F3 key to the host
Shift+Ctrl+4	Shift F4	Sends the Shift F4 key to the host
Shift+Ctrl+5	Shift F5	Sends the Shift F5 key to the host
Shift+Ctrl+6	Shift F6	Sends the Shift F6 key to the host
Shift+Ctrl+7	Shift F7	Sends the Shift F7 key to the host
Shift+Ctrl+8	Shift F8	Sends the Shift F8 key to the host
Shift+Ctrl+9	Shift F9	Sends the Shift F9 key to the host
F5	Specify	Transmits the cursor location to the host
Tab	Tab	Moves the cursor to the next field
F2	Transmit	Transmits the page to the host
Keypad +	Transmit	Transmits the page to the host

Key	Maps to	Description
Ctrl+F2	Transmit Line	Transmits the current row to the host
Keypad -	Transmit Line	Transmits the current row to the host

ALC keyboard mapping

Key	Maps to	Description
Ctrl+M	Auto Move Down	Toggles the session ability to receive multiple pages
Backspace	Backspace	Moves the cursor one column to the left
Shift+tab	back tab	Moves the cursor to the previous field
Ctrl+Home	Clear	Clears the screen and sends the CLEAR key to the host
Ctrl+B	Clear Broadcast	Clears the SITA broadcast message
:	Colon	Inserts a colon character at the cursor position
Ctrl+L	Cross of Lorraine	Inserts the Cross of Lorraine character at the cursor position
↓	Cursor Down	Moves the cursor down a row
Keypad ↓	Cursor Down	Moves the cursor down a row
←	Cursor Left	Moves the cursor to the previous word
Keypad ←	Cursor Left	Moves the cursor to the previous word
→	Cursor Right	Moves the cursor to the next word
Keypad →	Cursor Right	Moves the cursor to the next word
↑	Cursor Up	Moves the cursor up a row
Keypad ↑	Cursor Up	Moves the cursor up a row
Delete	Delete character	Deletes the character at the cursor location
Ctrl+Delete	Delete Line	Deletes the line at the cursor position
=	Display	Inserts the display character at the cursor position

Key	Maps to	Description
Ctrl+N	Display New Line	Inserts the display character at a new line
]	Dollar	Inserts the U.S. dollar sign character at the cursor position
.	End Item	Inserts the end item character at the cursor position
End	End of Line	Moves the cursor to the end of the line
Ctrl+T	End Transaction	Closes the PNR
Ctrl+E	Erase End of Display	Erases all data from the cursor position to the end of display
Ctrl+End	Erase End of Line	Erases all data from the cursor position to the end of line
Home	Home	Moves the cursor to the first unprotected field on the screen
Ctrl+I	Ignore	Cancel any changes made to the current PNR
Ctrl+Insert	Insert Line	Inserts a new line into display memory
Insert	Insert Space	Inserts a space into display memory
\	New Line	Inserts the newline character at the cursor position
[Pillow	Inserts the pillow character at the cursor position
Ctrl+G	Pound	Inserts a British pound mark at the cursor position
Ctrl+Enter	Print Enter	Sends the response to the printer
Ctrl+P	Protected Reset	Moves the cursor to the first unprotected field
Ctrl+↑	Recall Next Input	Recalls the next input or entry
Ctrl+↓	Recall Previous Input	Recalls the previous input or entry
Ctrl+Z	Reenter	Resends the previously sent message to the host

Key	Maps to	Description
Ctrl+R	Repeat	Redisplays the last message sent by the host
Escape	Reset	Resets keyboard error conditions
Shift+Ctrl+↓	Scroll Line Down	Scrolls the display down one line
Shift+Ctrl+↑	Scroll Line Up	Scrolls the display up one line
PageDown	Scroll Page Down	Scrolls the display down one page
PageUp	Scroll Page Up	Scrolls the display up one page
Ctrl+A	Select All	Selects all text
Shift+↓	Select Down	Selects all text down
Shift+↑	Select Up	Selects all text up
Shift+←	Select Left	Selects all text left
Shift+→	Select Right	Selects all text right
'	Start of Message	Inserts a start-of-message character at the cursor position
F12	Statistics	Displays communication statistics
Tab	Tab	Moves the cursor to the next unprotected field
Ctrl+F	Toggle CODACOM	Toggles CODACOM mode
Enter	Transmit	Transmits page to the host
Keypad Enter	Transmit	Transmits page to the host
Shift+Enter	Transmit	Transmits page to the host
Shift+Escape	Unlock Keyboard	Unlocks the keyboard
Ctrl+U	Unsolicited Message	Retrieves an unsolicited message from the host

Transfer Files

Host Access for the Cloud supports three different file transfer protocols:

IND\$FILE for 3270 host transfers

AS/400 for 5250 host transfers

File Transfer Protocol (FTP), which allows a local computer to act as an FTP client.

Once connected, you can view files on the server and transfer files between your local computer (or any networked drive) and the host.

Batch file transfer is available for FTP transfers. Using this option you can download and upload multiple files in one operation.

Before you can transfer or send files, the administrator must enable the transfer, send options for the current session, and make the necessary configurations. This is done on the File Transfer settings panel.

Depending on the host file system and transfer method you want to use, you will see different configuration options. Once configured, the file transfer dialog box is available from the tool bar.

[IND\\$FILE](#)

[AS/400](#)

[FTP](#)

[Batch transfers](#)

IND\$FILE

IND\$FILE is a file transfer program from IBM which you can use to transfer information between your computer and a 3270 host computer.

From the Host file system drop down list, select which IBM 3270 operating environment the host is running. Host Access for the Cloud supports TSO (Time Sharing Option), CMS (Conversational Monitor System) and CICS. The default selection is None.

There is support for ASCII or binary transfers and, if you connected to a TSO host, you can navigate directly to a particular TSO dataset.

General options for CICS, CMS, and TSO host file types

Automatically show host files - By default, the host file list contains all the host files that are available to transfer. To retrieve host files only when you request them, disable this option. On the Transfer dialog box, click Show host files to retrieve the host files.

Transfer options for CICS, CMS, and TSO host file types

Option	Description
Transfer method	<p>Binary - Use for program files and other types of files that should not be translated, such as files that have already been formatted for a particular type of printer or files with application-specific formatting. Binary files contain non-printable characters; using this method, a file is not converted or translated during the transfer.</p> <p>ASCII - Use to transfer text files with no special formatting. ASCII files on the PC are translated to the EBCDIC character set on the host and host text files are converted from EBCDIC to ASCII when they are downloaded.</p>
CR/LF processing	<p>If this option is selected, carriage return - line feed pairs will be stripped from files sent to the host and added to the end of each line on files received from the host.</p>
Startup command	<p>Specifies the host program used to initiate the file transfer. IND\$File, the default, is appropriate for CMS and TSO hosts. For CICS hosts, IND\$File may be appropriate, or you may need to specify your site's CICS transaction (for example, CFTR).</p>
Startup parameters	<p>Use this field for any parameters specific to the IND\$File program on your host system. The contents of this field are appended to the end of the transfer command generated by Host Access for the Cloud. Host Access for the Cloud does not validate the parameters.</p>
Max field size	<p>Select a field size to use with the Write Structured Field protocol. The default value is 4 kilobytes. Typically, the larger the buffer size, the faster the transfer. Most systems support 8K; if you choose a value that is too large for your host, it will disconnect your session when you first attempt to send a file big enough to fill the buffer. The person who installs the host communication software usually supplies this value. For example, IBM's host TCP/IP product gets this value from the DATABUFFERPOOLSIZ parameter, which defaults to 8K buffers. See your system administrator if you don't know what to enter here.</p>


Lead key	You can specify certain actions before transferring or listing files. Your choices are None, Auto Sense, and Clear. If set to None, LISTCAT is issued automatically. If set to Auto Sense, the current screen contents are examined to determine if a LISTCAT or TSO LISTCAT should be sent. If set to Clear, the Clear key is sent before issuing command. For TSO, Clear also means "TSO" will not be prepended to the request files command.
PC code page	The character set to use when reading or writing local files during a file transfer. The value Default uses the code page corresponding to your operating systems locale. If you need a different character set to specify the PC code page, select it from the list.
Host code page	The character set to use when translating EBCDIC characters while transferring files to or from the host. The default, Use NCS setting , uses the national character set specified on the Display panel under Terminal. If you need a different character set to specify the host code page, select it from the list.
Response timeout (seconds)	Specifies how many seconds Host Access for the Cloud should wait for a host response before timing out and returning an error. The default value is 60 seconds.
Startup timeout (seconds)	Specifies the number of seconds Host Access for the Cloud should wait for a host response when attempting to connect to a host. If the specified amount of time elapses with no response from the host, Host Access for the Cloud times out and returns an error. The default value is 25 seconds.

Send options for CICS, CMS, and TSO host file types

Option	Description	Applies to this host type
--------	-------------	---------------------------

Record format	<p>Use this option to specify the record format for files sent to the host.</p> <p>Default - The host determines the record format. This is the default option.</p> <p>Fixed - Forces the host to create fixed-length records.</p> <p>Undefined - Forces the host to create files without a specific record format (this value is only relevant for TSO systems).</p> <p>Variable - Forces the host to create variable-length records and preserves the format of a binary file.</p>	TSO, CMS
Allocation units	<p>Specifies the disk subdivisions for your primary and secondary space allocations. If you select Default (default), the unit is determined by the host. You can also select Cylinder, Track, or Block. If you select Block, use the Average block box to specify the size for an average block (in bytes).</p>	TSO
Logical record length	<p>The record size (in bytes) for the file being created on the host. If you leave this box blank, the record size is determined by the host. You can set any value between 0 and 32767 to accommodate any range accepted by your host. This option is not available on CICS hosts. For ASCII files, set this value to accommodate the longest line in your file. When you leave this box blank, the host usually accepts lines of up to 80 characters.</p>	TSO, CMS
If host file exists	<p>Specifies how the transfer should operate if a file with the same name already exists.</p> <p>Append - Append the contents of the local file to the existing host file.</p> <p>Overwrite - Overwrite the contents of the host file. With CICS systems there is no way to tell if a host file already exists, so Overwrite is the only available option for sending files to a CICS system.</p>	TSO, CMS


Block size (bytes)	On TSO hosts, specifies the block size for the file being created on the host. For files with fixed-length records, this value must be a multiple of the Logical record length (because blocks are divided into logical records). You can set any value between 0 and 32767, to accommodate any range accepted by your host.	TSO
Average block (bytes)	The size for an average block. This value is only relevant if you are using blocks as your allocation unit.	TSO
Primary allocation (allocation units)	The size of the primary allocation for the host file being created.	TSO
Secondary allocation (allocation units)	The size of any additional allocations in the event that the primary allocation is not sufficient. Multiple secondary allocations (known as "extents") are allowed, up to a host-specified limit (generally 15).	TSO

 **note**

When using CICS as the host system you must enter the names of the files you are transferring manually. A list of files to choose from is not available.

Transferring files (IND\$FILE)

You must be connected and logged into the host to transfer files for the current 3270 session.

1. Verify that the host is in a 'ready' state to accept the IND\$FILE command.
2.  From the tool bar, click the IND\$File icon.
3. The File Transfer dialog box displays a list of host files and directories that are available to transfer. Directories and files are indicated by an icon when you select the file. For CICS hosts, type in the names of the files you want to transfer.
4. Select the transfer method. The options are:
 - Binary

Use for program files and other types of files that should not be translated, such as files that have already been formatted for a particular type of printer or files with application-specific formatting.

Binary files contain non-printable characters; using this method, a file is not converted or translated during the transfer.

- ASCII

Use to transfer text files with no special formatting. ASCII files on the PC are translated to the EBCDIC character set on the host and host text files are converted from EBCDIC to ASCII when they are downloaded.

5. If you are connected to a TSO host, click Level to type in the new dataset you want to view. Host Access for the Cloud updates the remote file list using the dataset level you specify.



note

When specifying files using *Upload As* or *Download*, a fully qualified data set name needs to be enclosed in single quotes. Data set names not enclosed in single quotes will, by default, be prefixed with a high level qualifier specified in the TSO PROFILE.

You can refresh the file list at any time by clicking the Refresh icon in the upper left corner of the File Transfer dialog box.

[Downloading files \(IND\\$FILE\)](#)

[Uploading files \(IND\\$FILE\)](#)

[Troubleshooting your file transfers](#)

Downloading files (IND\$FILE)

1. From the list, click the name of the file to initiate the transfer.

or

Click **Download** and enter the name of the host file you want to transfer. You can download from both TSO and CMS host types. However, TSO and CMS represent host files differently; this means the format of the file name you enter into the message prompt will vary.

TSO - Wrap the name of the host path in single quotes to specify the complete dataset name. For example, 'BVTST03.DATA.TXT'. To specify a file location relative to the dataset level you set above, omit the single quotes. For example, DATA.TXT, which identifies the same dataset but relative to BVTST03.

CMS - A typical CMS input would be BVTSTT01 DATA A1. Single quotes are not needed.

2. If necessary, you can cancel the transfer from the transfer progress panel.

Uploading files (IND\$FILE)

note

IBM mainframe computer systems impose certain naming conventions for files. For detailed information on naming requirements, see the [IBM documentation](#).

Choose either method for uploading files:

- From the **File Transfer** dialog box, click **Upload**.

You can specify a different name for the uploaded file. Click **Upload as**, browse to the file you want to upload, and when prompted type the name you want to use. Remember that when connected to a TSO host, a fully qualified data set name needs to be enclosed in single quotes. See Step 5 under Transferring files.

or

- Drag the file you want to upload from its location to the **File Transfer** dialog box. Click **Refresh** to verify the file was successfully uploaded.

If you cancel the upload process before a file has been completely transferred, a partial file will be left behind on the host.

Troubleshooting your file transfers

Occasionally you might encounter errors when attempting a file transfer. These errors may be mainframe issues or may be caused by browser security settings.

- If a transfer completes but the file doesn't contain the data expected, verify that the transfer method is properly set to either Binary or ASCII.
- There is a 50MB file size limit on file transfer upload operations. You can modify this value. See the Deployment Guide for instructions.
- For host-specific errors, see [IBM File Transfer Error Messages](#)

AS/400

Using AS/400 file transfer you can transfer data between your computer and an iSeries host.

Generally, AS/400 file transfers are straightforward and not complex. However, since the host data is managed as a DB2 database, you can, using the SQL Editor, create fairly complex queries.

To configure AS/400 file transfer

1. Create an HACloud 5250 terminal session, enter a host name or address and name the session.
2. On the Settings panel, choose File Transfer.
3. Select Enable AS/400 and proceed with the configuration.

• Host

The host address that you provided for the terminal session is pre-populated in the host field. You can, if needed, use a different host. To specify a different port, append the port number to the host address. For example, host.mycompany.com:23.

• TLS Security

From the drop down list, select the TLS security option you want to use.

To use this option: The AS/400 database server's certificate must be added to the list of trusted certificates in MSS. If the certificate has not already been added, then see [Trusted Certificates](#) in the MSS documentation for instructions.

• Default transfer method

Set your preferred default transfer method; fixed width Text, or Comma Separated Values (CSV). The transfer method can be changed when you perform a transfer.

• Include column headers by default


Select this option to include column headers by default for all downloaded data. You can modify this setting for each download on the file transfer dialog box.

Column headers do not originate on the host file, but are added when a file is downloaded. They are automatically removed when a file is uploaded.

4. Click **Save** and connect to the session.

Transferring files (AS/400)



After the session has been configured to use AS/400 file transfer functionality, click  on the toolbar to open the **File Transfer** dialog box. This dialog contains a list of host files that are available to transfer. If prompted, enter your AS/400 login credentials.


[Downloading files \(AS/400\)](#)

[Downloading using SQL](#)


[Uploading files \(AS/400\)](#)

[Adding a library](#)

Downloading files (AS/400)

The AS/400 file system consists of Libraries, Files and Members. Libraries are identified by this icon: . While you cannot download libraries, you can click on the library to see the files and members contained within.

Select **Include column headers** to display column headings for downloaded data.

1. Open the library containing the files ().
2. Expand the file that contains the member you want to download.
3. Click on a member to download it.
4. Open the browser's download folder to confirm the file is there. Open the file in a text editor.

Downloading using SQL

You can create SQL queries to obtain only the data you need from a file member on the host. This lets you select specific fields and ignore others.

1. Open the library and file you want to download.
2. Open the options menu and click **SQL**.



3. The SQL Editor opens and contains the SELECT statement that is used to download the whole member. The file member is referenced as LIBRARYNAME/FILENAME(MEMBERNAME).
4. Click **Run** to download the whole member. or edit the SQL and click Run to retrieve a subset of the data.

Uploading files (AS400)

You can only upload data into files as either new or replacement members. The AS/400 file contains a specification describing the data in members and each member in a given file has the same structure. Typically, you cannot (or should not) download a member from one file and upload it to another unless both files have the same data specification.

Since data can only be uploaded as members, you need to open a file and display its members in the file list dialog before the Upload button is enabled.

1. Open the file that you want to upload to. The Upload button is now available.

2. Either:

Click the **Upload** button and select a file from your local file system to upload.

or

Click the down arrow on the Upload button and select **Upload as...** Then select the file, give it a new name, and click OK.

Adding a Library

Typically, as an AS/400 user, you will have access to a certain set of libraries that has been assigned by a system administrator. These libraries appear as the top level entries in the file transfer dialog. If you need to access a library that is not on your list, your system administrator can update your configuration so that the new library is added to your list. Occasionally you may need to work with a library on a temporary basis; you do not need it added permanently to your library list.

To add a library

On the AS/400 file transfer dialog box, click **Add Library**. This button is available from the library list panel. This addition is not permanent and you will have to add the library again if you close and reopen the file transfer dialog box.

FTP

With Host Access for the Cloud your local computer can act as an FTP client. Using the FTP client, you can connect to an FTP server running on another machine. Once connected, you can view files on the server and use FTP to transfer files between your local computer (or any networked drive) and the FTP server. Using FTP, a client can upload, download, delete, rename, move and copy files on a server, either singly or as a batch transfer, where you can build lists of files to be transferred as one operation.



If you plan on using a batch transfer, first select and configure the Enable FTP option.

To configure FTP

Select Enable FTP and proceed with the configuration:

- **Protocol**

Use FTP to start a standard FTP session. Use SFTP to start an SFTP session.

You can set up an FTP client to use the SFTP protocol and perform all operations over an encrypted secure shell transport. Host Access for the Cloud uses user name and password to authenticate.

- **Host**

Specify the host name or IP address of the FTP server to which you want to connect.

- **Port**

The port of the FTP server specified.

- **Anonymous user**

Select this option to log onto the specified FTP server as a guest, with the user name "Anonymous". If the host you are connecting to does not support anonymous users, it may be necessary to supply your credentials.

- **Account**

In the Account field, specify the name of the account to log into on the FTP server. A few servers require the account name as part of the login. For case-sensitive servers, be sure to use the appropriate case when entering the account name.

Select **Prompt user for account** to give the user the option to override the account value you set here. If the checkbox is not checked, then the set value is sent to the host upon connect/login without prompting the user.

- **Session timeout (seconds)**

This value tells the FTP client the maximum number of seconds to wait for data packets being transferred to or from the host. If nothing is received within the period specified, a timeout error displays and the transfer terminates; in this case, try the operation again. If you receive repeated timeout errors, increase the timeout value. Entering 0 (zero) in this box prevents the FTP client from ever timing out when waiting for a response. For SFTP sessions, the default is 0 (zero).

- **Keep Alive time (seconds)**

Select this option and enter a time in seconds if you want to continue your connection to the server beyond the server's automatic timeout value for inactivity. Most servers have an idle

time value that specifies how long a user's FTP session can last when no activity is detected. When the user exceeds the time limit, the server connection is closed.

This setting allows you to direct the FTP client to send a NOOP command to the server at timed intervals to prevent the server from closing the connection due to inactivity. Be aware that by continuing your session you may prevent another user from making a connection to the FTP server.

- **Automatically show host files**

By default, the host file list contains all the files that are available in the current working directory. To retrieve the list of host files only when you request them, disable this option. Then on the Transfer dialog box, click Show host files to retrieve the list of host files. This feature is useful when the host file system contains large numbers of files.

- **Startup QUOTE command**

The string you specify here is sent to the FTP server as the session starts. The Startup QUOTE command allows you to set FTP server options when starting the session.

- **Initial remote directory**

Specify the path to a home or default directory for the FTP site. When a connection to the FTP site is opened, the server working directory is set automatically to the specified home path. The files and folders in the server home directory appear in the FTP session window. If the initial remote directory is not found, a warning is reported and the connection continues.

- **If remote files exists when uploading file**

Specify how you want to handle the transfer if a file with the same name already exists.

Select this option	to do this
Append	Append the file being sent to the existing fileB1
Ask user (default)	Prompt for a decision on how to handle the duplicate file name
Cancel	Cancel the file transfer
Fail	Cancel the file transfer and receive a notification of failure
Overwrite	Overwrite the existing file on the remote machine
Skip	When multiple files are in a request, skip the file matching an existing file name, but proceed with the transfer for other files.
Unique	Create a new file with a unique file name


- **Host encoding**

Specifies the character set used by the host to display the names of files that are transferred. By default Host Access for the Cloud uses UTF-8 (Unicode). If you transfer files with the default

setting and the file names are unrecognizable, change the Host encoding option to the character set used by the host. (This option does not affect the encoding for the contents of the files that are transferred; it applies to the file names only.)

Transferring files (FTP)



After the administrator configures a session to include FTP functionality, click  on the toolbar to open the FTP File Transfer window containing a list of host files that are available to transfer. Directories and files are indicated by an icon when you select the file.

1. Select the transfer method. The options are:

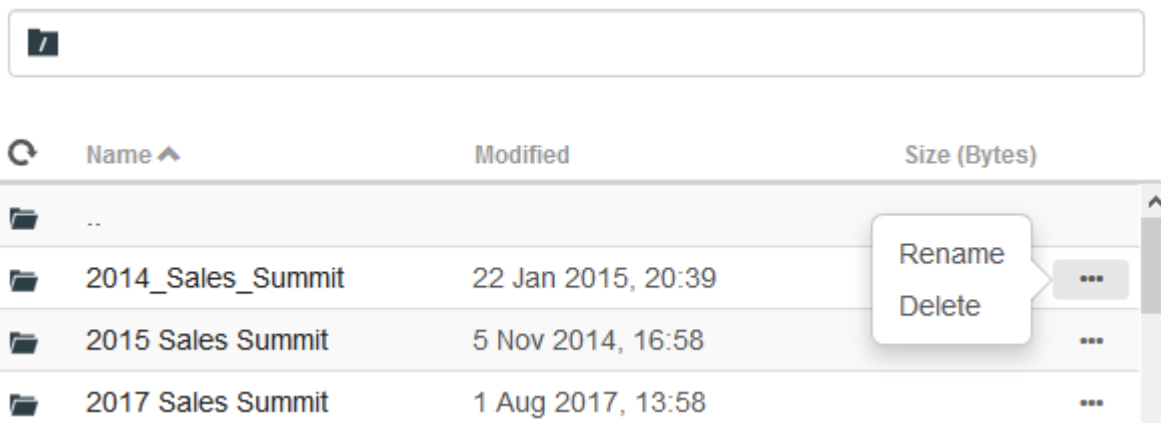
- Binary

Use for program files and other types of files that should not be translated, such as files that have already been formatted for a particular type of printer or files with application-specific formatting. Binary files contain non-printable characters; using this method, a file is not converted or translated during the transfer.

- ASCII

Use to transfer text files with no special formatting. Text files on the PC are translated to the appropriate character set on the host and host text files are similarly converted to the local character set when they are downloaded.

2. You can rename, delete, or download a file from the list of files.



3. Refresh the file list at any time by clicking the Refresh icon in the upper left corner of the File Transfer dialog box.

Downloading files (FTP)

1. From the list, select the file to initiate the transfer.
2. Optionally select the **Download** button and specify a fully qualified path to a file for download. The format of the remote file identifier depends on the type of file system on the host system but here are a few examples:

Unix example: /home/user/subdirectory/myfile.txt

TSO example: 'ACCT.INVOICES(INTL)'

AS/400 example: ACCT/INVOICES.INTL

3. If necessary, you can cancel the transfer from the transfer progress panel.

Uploading files (FTP)

Choose from the following methods when uploading files:

1. From the **File Transfer** dialog box, click **Upload**.
Choose the file you want to upload from the Browse window.
2. Click the down arrow on the Upload button and select **Upload as...** Then select the file, give it a new name, and click OK.
3. Drag the file you want to upload from its location to the **File Transfer** dialog box.

TSO

You can provide a fully qualified data set name as the receiver of the transferred file. The name must be surrounded by single quotes e.g. 'ACCT.INVOICES(INTL)'

Click **Refresh** to verify the file was successfully uploaded.

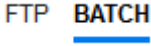


Click **New Directory** to create a new directory on the remote server. You are prompted to enter the new directory name.

Batch transfers

note

You must first enable FTP on the File Transfer settings panel FTP tab before you can configure batch transfers.

To transfer multiple files in one operation, use the **Batch** option.

1. From the Settings > File Transfer > FTP panel, check **Enable FTP**.
2. Click  to open the Batch file transfer panel.
3. Select Cancel batch when single failure occurs to stop the transfer if a file fails to transfer.
4. Click  to create the list of files you want to transfer.
 - a. Name the list. To aid in building similar lists, you can copy an existing list, rename it, and then add or delete files as needed using the options available when the original list is highlighted.
 - b. From the right panel, click  to open the Add transfer request dialog box.
5. On the Add transfer request panel, begin building the list:

Option	Description
Transfer	Choose whether to upload or download the file.
Local file name	Identify the file you want to transfer. You can enter the name of the file or browse to it.
Remote file path	Provide a location to name and store the file after transfer. You can: Keep original file name and use the initial remote directory - leave the field blank Use a new file name - enter newfilename.txt. Puts the file in the initial remote directory using the given name. Keep original file name but use a new directory path - /folder/. Uses the original file name with the new path. Use new directory and a new file name - /folder/newfilename.txt.
Transfer method	You can choose from Binary or ASCII transfer methods.

If remote file exists	Decide how to handle file transfer if a remote file already exists. The options are: Overwrite (default) - Overwrite the existing file on the remote machine Append - Append the file being sent to the existing file Ask user - Prompt for a decision on how to handle the duplicate file name Cancel - Cancel the file transfer Fail - Cancel the file transfer and send notification of the failure Skip - The file matching an existing file name is skipped, but the transfer proceeds for the other files in the batch Unique - Create a new file with a unique file name
-----------------------	--


6. Click **Save**.

Transferring files (Batch)







hint

Administrators grant permission to transfer files using the **User Preference Rules** option from the Settings panel.



Click  on the tool bar to open the list that contains the files you want to transfer.

1. Due to browser requirements, you need to specify the location of all files that you want to upload. Locate files as needed using the Search icon. Those files are easily identified with a yellow icon as such:

	Local file name	Transfer	Remote file path
<input checked="" type="checkbox"/>	 Locate "aed.jpg"	  Upload	aed.jpg
<input checked="" type="checkbox"/>	 Locate "ascii.txt"	  Upload	ascii.txt

2. Files in the batch list are selected by default. To edit the file prior to transfer, you can eliminate files from the transfer operation by clearing their respective check boxes, or by selecting **All** from the drop down menu. You can also filter the list of transferable files based on their download or upload status.
3. Click **Start** to initiate the transfer.

Specify Edit Options

Edit options to use for different copy, paste and cut operations.

Copy options

Select text by left-clicking and dragging over it with the mouse or by pressing and holding the shift key while modifying the selection with the arrow keys. By default, different terminal types use different selection modes when copying text. VT terminals use a linear selection mode while all others use block mode selection. To toggle between block and linear selection modes on any terminal type, press and hold down the **Alt** key, then select the text.

- **Copy input fields only** - Select this option to only copy data from input fields. Data from protected fields is replaced with spaces when placed on the clipboard.
- **Use entire display when there is no selection** - This option applies the Copy command to the entire terminal display when nothing is selected.

Paste options


Choose Paste to write the contents of the clipboard at the cursor location.

- **Skip protected fields** - Specifies how pasted text is mapped onto the screen:
 - If unselected (the default), the text is interpreted as a linear stream that can contain new lines and delimiters and is pasted accordingly.
 - If selected, the text is interpreted as a host screen data and overlaid onto the current screen starting at the current cursor position. Where the current screen contains an unprotected field, the source text is pasted; where the current screen contains a protected field, the source text is skipped.
- **Wrap to next field on current line** - Select this option to have data pasted from the clipboard fill as much of the current field as possible. Any remaining data will be pasted to the next field on the same row, until the end of the row or the data has been exhausted. If **Wrap text to next line** is also selected, additional data will be pasted to subsequent fields on the next row, and the data is aligned vertically with the starting cursor position.
- **Wrap to next line** - If selected, the Paste command fills the first field with as much Clipboard data as the field will hold. Any remaining text is pasted to the line immediately below, assuming it is writable (for example, an input field). Otherwise, the remaining text is truncated. Subsequent lines of data are pasted to align vertically with the starting cursor position.

By default, this option is not selected and text that overflows the field is truncated.

- **Restore starting cursor position after paste** - By default, the host cursor is positioned at the end of the data following a paste operation. Select this option to restore the host cursor to its starting position after the paste operation is complete.

Cut options

The Cut operation is available for all supported terminals except for VT host types. Select the area you want to cut, then click the  button on the toolbar. You can use either the context menu or key combination to cut the data from the screen and save it to the clipboard. Data in protected fields is copied to the clipboard but is not removed from the screen.

Key combinations


Commonly used key combinations for editing functions are supported in HACloud. These keys pass through to the browser, which generates the appropriate editing functions.

Key combination	Host type	Action
Ctrl + C	3270, 5250, UTS, T27, and ALC	Copy
Ctrl + V	3270, 5250, UTS, T27, and ALC	Paste
Ctrl + X	3270, 5250, UTS, T27, and ALC	Cut

These key combinations are mapped in HACloud to various screen edit actions:

Key combination	Host type	Action
Ctrl + A	3270, 5250, UTS, T27, ALC	Select all text on the screen
Shift + Arrow key	All	Change the extent of the current selection
Ctrl + Shift + A	VT	Select all
Ctrl + Shift + C	VT	Copy

Key combination	Host type	Action
Ctrl + Shift + V	VT	Paste

 **Note**

In HACloud you can use the key mapper to map editing actions to key combinations. You can access Edit actions using a context menu on the terminal by right-clicking on the screen. Editing actions may be restricted by browser permissions. When an action is unavailable to the user, the associated toolbar buttons and context menu items will not appear.

Printing

There are various printing options available for 3270, 5250, and UTS hosts. You can take screen captures, print a selected screen, and enable and configure host printing capabilities:

The settings available to you regarding page setup and orientation are dependent on your browser options.


Capture a screen

Use the screen capture feature to capture multiple screens then save them as a file for printing or sharing. This option is available to all users once the administrator selects it using **User Preferences**.

1. Navigate to the screen you want to capture.

2.



Click  to capture the screen. The counter displays the number of captures you've taken. Each capture will print to a separate page.

3. Click **Save** to browse to the location where you want to save the capture. Your browser determines how the save option functions. For example, in Chrome, depending on your browser settings, the file will be saved in the download file or you will see a **Save As** dialog to select a location to save the captured file.

4. To append your newly saved screens to an existing screen capture file, click **Append and save**. When you print the appended file, each screen capture is printed to a separate page.

5. You can clear the captures whenever you want by clicking **Clear**.

Print a screen

The print screen option prints the contents of the terminal screen. It does not print the toolbar or other display information.

1. Navigate to the screen you want to print.

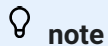
2. Click **Print Screen** on the tool bar.

3. Use your browser's print dialog to select the printer and page setup options.

Host printing

This feature is available to 3270, 5250, and UTS host sessions. You can create one or more printer sessions and associate them with the current terminal session. Each printer session is bound to a Device ID on the host system and any subsequent print jobs sent to that Device ID will be directed to the Host Access for the Cloud web client.

The host session builds either a PDF (the default) or [text file](#) and sends it to the web client. After receiving the file, the web client downloads the file following your browser's configured download options. Different browsers provide different options for handling downloaded files. When the file is received, you can direct it to any printer that you have access to.



An administrator can provide end users with the ability to print by setting the **Host Printing User Preferences** option.

To configure host printing

1. From a host session, click **Settings** on the tool bar to open the left navigation panel.
2. In the left panel, click **Print**.
3. Click **Add** to open the configuration dialog box. Customize your printer session by using the settings on each tab: Connection settings, Page Setup settings, and Advanced settings.
4. Click **Save** to return to your session. The settings take effect when the session is reopened.

Host Printing topics

[Connection settings: 3270, 5250, UTS](#)

[Page Setup settings](#)

[Advanced settings](#)

[To print your host printer session](#)

Connection settings

By default, printer sessions are available from the printer icon on the terminal session tool bar. If you do not want end users to have access to this printer session, clear **Enable this printer session** on the Connection tab.

These settings differ depending on your host type.

[3270 connection settings](#)

[5250 connection settings](#)

[UTS connection settings](#)

3270 connection settings

Setting	Description
Name	Provide an easily identifiable name for your printer session. Required.
Protocol	Select the protocol to use. The options are: TN3270E - TN3270E of Telnet Extended, is for users of TCP/IP software who connect to their IBM mainframe through a Telnet gateway that implements RFC 1647. TN3287 - TN3287 is for users of TCP/IP software who connect to their IBM mainframe through a Telnet gateway that implements RFC 1646.
Device ID	Specify whether you want to use a Device ID, prompt for Device Name or, if you select TN3270E, a TN Association, to link the terminal session with the print session. Required. Select one: Specify Device ID - Specify the Device ID to use when the printer session connects to the host. Use TN Association - (TN3270E) If you choose to use a TN association, Host Access for the Cloud uses the device name specified in the connection settings to link the 3270 and 3287 session together. TN Association is only available if you select TN3270E as the protocol. Prompt the user - When the printer session connects, the user is prompted to supply the Device ID for the printer session.

5250 connection settings

Setting	Description
Name	Provide an easily identifiable name for your printer session. Required.

Device ID	Specify whether you want to use a Device ID or prompt for Device ID: Specify Device ID - Specify the Device ID to use when the printer session connects to the host. Prompt the user - When the printer session connects, the user is prompted to supply the Device ID for the printer session.
-----------	---

UTS connection settings

Setting	Description
Name	Provide an easily identifiable name for your printer session. Required.
Protocol	<p>The choice of DEMAND or MAPPER protocols depends on the type of UTS session you create. UTS session types are determined by the values you supply for the TSAP and Application options on the Connection panel. For example, if you enter values that create a UTS MAPPER or DEMAND session, you should select MAPPER or DEMAND as the protocol. Specify which protocol to use:</p> <p>MAPPER - You can choose to specify the Device ID to use when the printer session connects to the host or prompt the user to supply the Device ID for the printer session, then continue configuring the session.</p> <p>DEMAND - After providing a name for the session, you can continue configuring the session using Page Setup and Advanced tabs.</p>

Page Setup settings

The Page Setup tab contains setting options for paper size and orientation, along with dimensions, margins, and scaling values.

Setting	Description
Paper size	Select the size of paper used by the printer.
Orientation	Choose from three modes: Portrait (vertical), Landscape (horizontal) or Auto , which is the default. With Auto selected, the printer evaluates the print job and uses the most appropriate format.
Units of measurement	Select the unit of measurement you want to use for page margins and page sizes. The values are inches or millimeters.
Dimensions	Enter the number of rows and columns to display per printed page. 60 is the default row value and the column value defaults to 80.
Margins	Sets the left, right, top, and bottom page margins.

Setting	Description
Scaling	Sets the horizontal and vertical scaling for printed output. Increase the percentage to increase the horizontal or vertical space used by the printout.

Advanced settings

Choose the file output format and the download options.

Output format

- **PDF** - (default) prints the file in PDF format.
- **Text** - prints the file in plain text format.

Download printer output

- **Automatically** - (default) The file is downloaded automatically when the print job is complete. When this option is selected, the Inactivity timeout setting is not available.
- **Manually** - Once a print job commences, you can initiate a download anytime by locating the print job in the download list available from the Print icon on the tool bar and clicking Flush. The print job is aggregated into a single file and downloaded.
- **After inactivity timeout** - Using this option you can print multiple print jobs, have them aggregated into a single file, and then automatically downloaded when you specify.

If you decide on a value greater than 0 (for example, 5 seconds) any print jobs assigned to a printer that arrive within 5 seconds of each other will be appended to the same file. After 5 seconds and no remaining print jobs, the file is downloaded.

If you specify 0 for the inactivity timeout, each print job is downloaded immediately upon completion. You can always interrupt a print job by clicking **Flush**.

To print your host printer session

When the terminal session opens, you can now:

1. Select the printer session you want to use. All print sessions associated with the opened terminal

session are available to you. Click  on the tool bar to see a list.

2. The host session receives the print data from the host and builds a file to print. A link to this file is sent to the web client indicating it is available for download.

You can monitor the various print jobs using the tool bar page counter or the counter associated with separate printers in the print drop down list.

The page counter on the tool bar reflects the total number of pages either being actively printed or complete but waiting for the file to download from the server. You can trigger a download by selecting Flush from the printer list.

The page counter attached to printers in the printer drop down list displays the same value but on a per printer basis. The sum of these separate print jobs is reflected in the tool bar count. The count is cleared once the print jobs are downloaded.

3. After the file becomes available, the file either begins downloading or waits for you to trigger a download using the Flush option, depending on the options you configured.

If necessary, due to an overlong running print job or some other issue, you can flush your current print job. The **Flush** option is available from the list of printer sessions accessed from the printers icon on the tool bar. When you flush a print job, whatever has been accumulated so far is printed and processing of print data continues.

Customizing Host Sessions

Administrators can use these features to customize sessions for end users:

- **Plus** - Enable custom controls to provide a more efficient work flow and a more modern and friendly interface. See Use Plus in the Deployment Guide.

Using this option, administrators can add tool tips to fields, replace old-style numbered lists with more modern drop-down lists, add buttons to the host interface and program them to start macros or perform other actions, and replace manual date entry with a graphical calendar date-picker.

- **Server-side Events** - Supply procedural Java code that extends and improves the presentation of host data.

Using server side events, you can define specific events and suspend the host application, replacing or interrupting it with code that you have supplied to the session, as well as extend error handling options. For example, you can add an event that recognizes when an error occurs and then implements the code to intercept the error, take control, and correct the error. See Server Side Events in the Deployment Guide.

- **Advanced** - Only use as directed by Technical Support.

These options are configured on the Customization panel.

1. Click Settings on the toolbar to open the left navigation panel.
2. Click Customization.

Managing User Preferences

As an administrator you can choose what options users can configure for their sessions. These options are set on a per session basis and all users who have access to a particular session can configure their own session instance.

1. From the left navigation panel, choose **User Preference Rules**.
2. Select which options you want to allow your users to configure.
3. Click Save.

Each user's configurations are specific to their instance of the session and will not conflict with those of other users.

There is a **Restore Defaults** option available on the various settings and display panels. As an administrator, this option restores the web client back to its default settings. For end users this option will restore the values set by the administrator when the session was created.

Warning

When the authentication method is set to None, be aware that all users share the same settings. During session configuration, it is best to not allow users to modify their session settings (User Preference Rules), because they can overwrite each other's choices.

More information

[Display Settings](#)

[Specify Edit Options](#)

[Transfer Files](#)

[Creating Macros](#)

Legal Notice

© 2024 Open Text

The only warranties for products and services of Open Text and its affiliates and licensors (“Open Text”) are as may be set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Open Text shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.