MICRO FOCUS

Reflection ZFE
Users Guide

August 2018

Contents

About Reflection ZFE

The Reflection ZFE web client provides browser-based HTML5 access to 3270, 5250, VT, UTS, and T27 host applications. The Reflection ZFE 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.

The web client operates with complete session protection using SSL/TLS to secure communication with your mainframe systems.

1 Release Notes

Reflection ZFE version 2.3.1 released August 2018. These release notes list the features and known issues in this release and information on how to obtain the product. Reflection ZFE provides terminal emulation for 3270, 5250, VT, ALC, UTS, and T27 host types, while requiring only an HTML 5-capable browser.

NOTE: The End User License Agreement (EULA) is available in English, Spanish, French, Italian and German in the `<install location>\licenses` directory.

What's New

This is the first interim release of Reflection ZFE since the 2.3 release in July 2018. The list of new features and fixes includes everything included in this release as well as the 2.3 release with their release versions noted. All releases are cumulative and this 2.3.1 release contains everything released in all prior releases. You can see release notes for previous releases here.

Host Access Management and Security Server version 12.5.1 released with Reflection ZFE 2.3.1. MSS release notes are available here.

NOTE: As of the 2.3 release, Reflection ZFE uses activation files to enable host access and product functionality. After installing, open the MSS Administrative Console (Configure Settings > Product Activation) to activate. An activation file follows this format: `activation.<module_name>.jaw`.

- Features and fixes include:
  - You can transfer multiple files in one operation using the new FTP batch file transfer option. (2.3.1)
  - Reflection ZFE Airlines Edition adds support for ALC host types. (2.3)
  - Easier and more user friendly certificate and key store management (2.3)
  - Reflection ZFE now includes OpenJDK: Azul Zulu as a replacement for the Oracle JDK (2.3)
  - New font options, including resizing, zero customization, and scaling functionality (2.3)
- Multiple bug fixes

Known Issues

Micro Focus Technical Support is always available to help you with any issues you may encounter in Reflection ZFE.

- Reflection ZFE and replication

  Management and Security Server (MSS) provides the ability to replicate Reflection ZFE settings, including user customizations, macros, and session servers. In order for the list of Reflection ZFE session servers in the MSS Administrative Console to reflect changes made on other MSS servers, you must restart MSS.

Unresolved issues from previous releases are listed in Technical References under Known Issues.

Installing the Product

Read Installing Reflection ZFE for specific system and installation requirements and helpful tips.

Contacting Micro Focus

For specific product issues, contact Micro Focus Support (https://www.microfocus.com/support-and-services/).

Additional technical information or advice is available from several sources:

- Product documentation, Knowledge Base articles and videos - see Support for Reflection ZFE.
- The Micro Focus Community pages – see Micro Focus Communities.

Legal Notice

For information about legal notices, trademarks, disclaimers, warranties, export and other use restrictions, U.S. Government rights, patent policy, and FIPS compliance, see https://www.microfocus.com/about/legal/.

The only warranties for this product and any associated updates or services are those that may be described in express warranty statements accompanying the product or in an applicable license agreement you have entered into. Nothing in this document should be construed as creating any warranty for a product, updates, or services. The information contained in this document is subject to change without notice and is provided “AS IS” without any express or implied warranties or conditions. Micro Focus shall not be liable for any technical or other errors or omissions in this document. Please see the product’s applicable end user license agreement for details regarding the license terms and conditions, warranties, and limitations of liability.

Any links to third-party websites take you outside Micro Focus websites, and Micro Focus has no control over and is not responsible for information on third party sites.

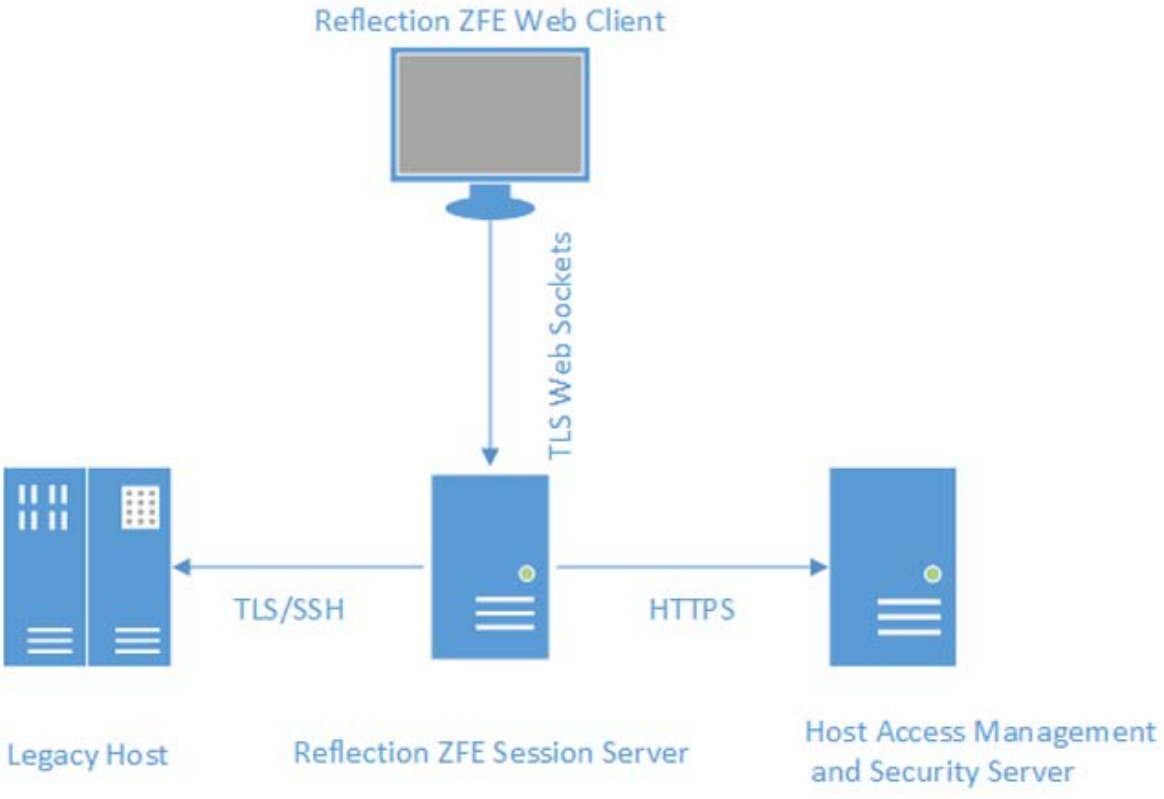
2 Getting Started

Reflection ZFE provides zero-footprint terminal emulation that delivers browser-based HTML5 access to 3270, 5250, VT, UTS, ALC and T27 host applications without the need to touch the desktop or install and manage Java runtime environments. A centralized administrative location reduces IT costs and desktop management time while efficiently providing and delivering host access to end users. Communication is protected using HTTPS, SSL/TLS, and SSH security.

Next steps

- Learn how ZFE works
- Evaluate ZFE
- Walk through ZFE
- Install ZFE

How does it work?



It is a simple solution. Reflection ZFE drives down IT costs. You have eliminated the need to touch the desktop.

Reflection ZFE components

- Host Access Management and Security Server

  The Host Access Management and Security Server (MSS) provides an Administrative Console, a web-based centralized location where you can add, edit, and delete terminal sessions. MSS is part of the broader Micro Focus story and is compatible with other Micro Focus products.

- Session Server

  The session server is an NT service or UNIX daemon that provides the engine that runs host sessions. Multiple session servers can serve up tens of thousands of sessions and provide efficient and rapid access to your host data.

- Web Client

  The web client is the web-based terminal emulator where your users can easily access authorized sessions from any platform and from any location.

  The Web client provides macros, keyboard and color mapping, on-screen keyboard, copy/paste functionality, host-initiated screen updates, and file transfer capabilities.

Browser and operating system support

Reflection ZFE is a 64-bit product and supports Google Chrome, Mozilla Firefox, and Microsoft Internet Explorer and Edge browsers. A complete list of supported platforms and other installation requirements is available in the Installation Guide.

Security considerations

When you open up your legacy hosts to users outside the corporate firewall - business partners, remote users, mobile sales personnel, and others - you need to shield your information from known security threats. With Reflection ZFE, you can provide secure web-to-host access to all your users, whether they're around the corner or around the world. Reflection ZFE, along with the MSS, provides HTTPS connections and a variety of authorization and authentication options.

Reflection ZFE supports the TLS and SSH protocols to protect mission-critical data. To secure your passwords and other sensitive data, use the HTTPS protocol, which provides TLS encryption. Supported cipher suites include AES128, 168-bit Triple DES, and other strong ciphers, ensuring confidentiality and integrity of data over the Internet and other insecure networks.

Reflection ZFE can be connected securely to the browser, the host, and the management server. See Making Secure Connections for information on securing those connections.

Evaluating Reflection ZFE

Reflection ZFE is a zero-footprint terminal emulator that lets you:

- Forget about desktop software management and dependencies

- Centralize control of all host access to mainframe applications for both user provisioning and security requirements
- Gain insight into end-user host access using centralized metering and reporting to optimize computing resources

If you don't have our software yet, visit https://www.microfocus.com/products/reflection/zfe/trial/ and fill out an evaluation request form. You'll be sent an e-mail message with instructions to download and install an evaluation copy of Reflection ZFE good for 120 days. Using this evaluation copy you can open and close host sessions and maintain 5 active host connections at a time. The trial site has all the information you need to take the next step.

If you have questions about using the download site, see Using the Micro Focus Downloads Web Site (FAQ).

The installation wizard walks you through the installation process.

When you install Reflection ZFE, make sure that you are pointing to the Management and Security Server you want to use. MSS uses activation files to provide product-specific functionality. This file is included in the evaluation download and once Reflection ZFE is installed should be in place and ready to be activated. This is done in the MSS Administrative Console, under **Configure Settings - Product Activation**. Activation file names use this format: `activation.<module_name>.jaw`

Learn more about activation files.

Additional resources

To read more about Reflection ZFE and Micro Focus there are a number of resources available.

- Overview of Reflection ZFE
- Reflection ZFE Data Sheet
- Overview of Host Access Management and Security Server

Walking through Reflection ZFE

This walk-through assumes you are the administrator. You have installed Reflection ZFE and pointed it to the appropriate Management and Security Server (MSS), activated the product, and now you want to start assigning, authenticating, and providing host access for your users.

The steps

- ✓ Open the MSS Administrative Console.
- ✓ Create and launch a new session. This opens a new browser window and the web client **Connection** panel displays.
- ✓ Configure settings, including key and color mapping, enabling hotspots and macros, and other connection and user preference options.
- ✓ Assign users to sessions.

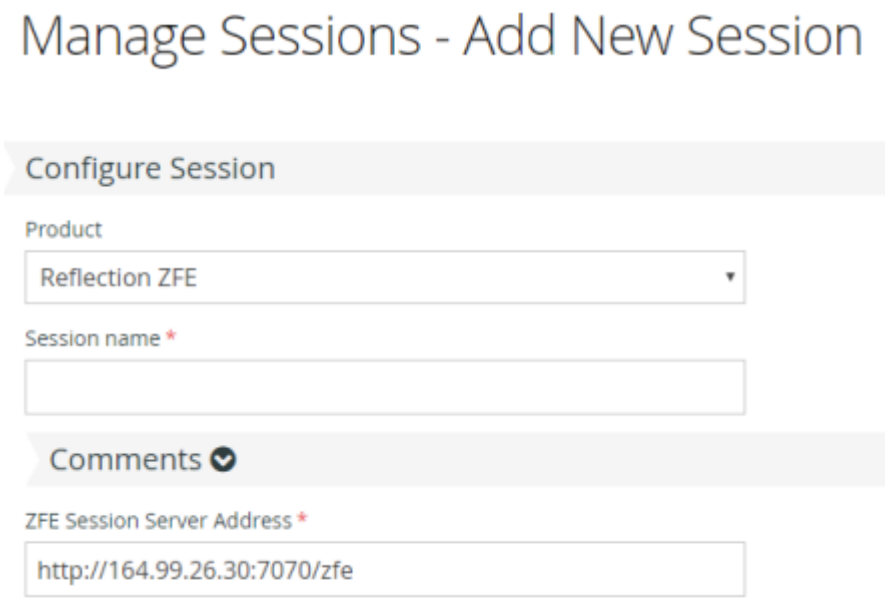
Open the Administrative Console

1. In a Windows environment, from the Start menu, under Micro Focus Reflection ZFE, click Administrative Console or open the URL for the administrator login page in your web browser. The URL uses this format: `https://myserver.mycompany.com:443/adminconsole`.
2. If you connect using HTTPS and your server has a self-signed certificate, your browser will warn you about the certificate you created. This is expected behavior; you can accept the self-signed certificate or choose to proceed and the administrator login page will open. After you purchase a CA-signed certificate or import the self-signed certificate into your certificate store, these warnings will stop.
3. Log on as an administrator by entering the password that you specified when you installed MSS. The default user name is ***admin.***

Create a new session

You add, edit, and manage sessions from the Manage Session panel of the Administrative Console. When you add a session it becomes available in the session list of this panel.

1. From the Manage Session panel, click **Add** to create a new session

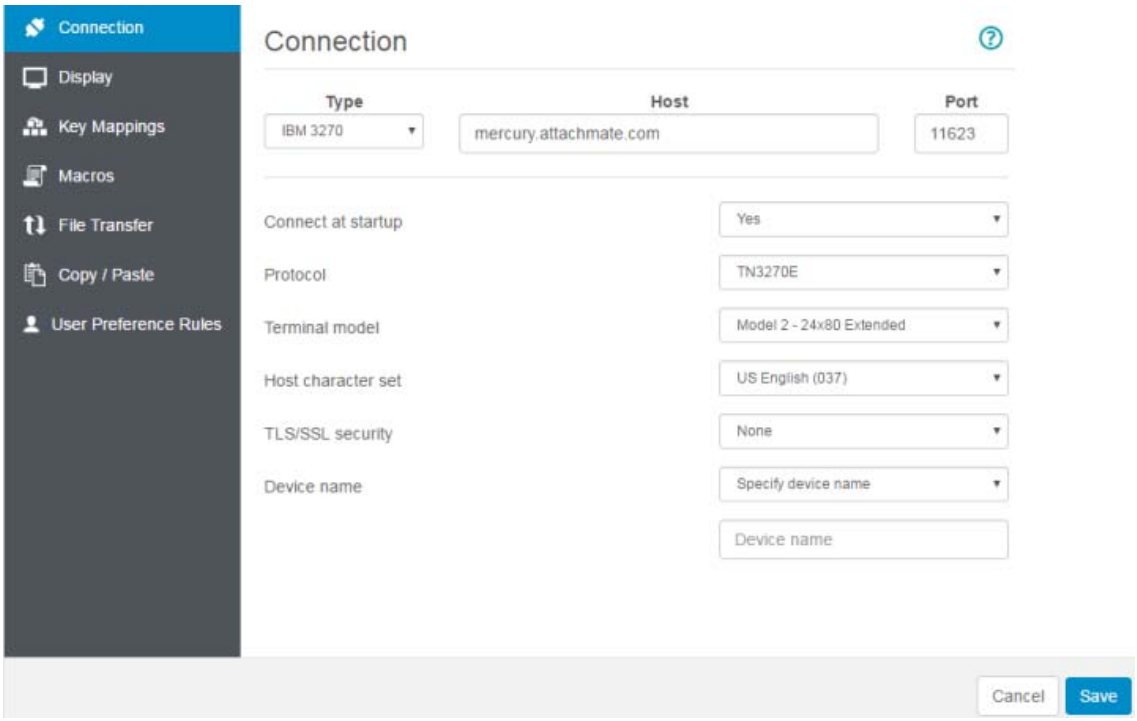


2. If it is not already selected, select Reflection ZFE, enter a session name, and any comments you want to capture and click **Launch** to open a new browser window and start configuring the session for the server listed at the ZFE server address.

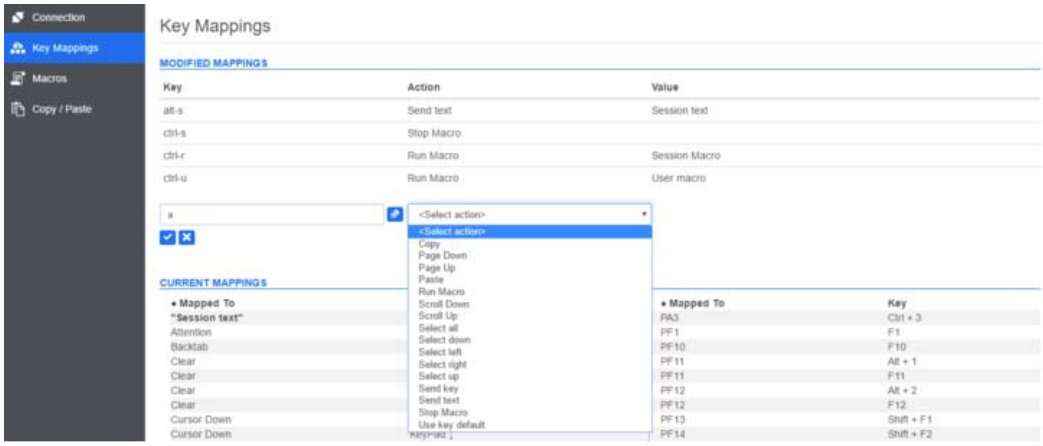
Configure settings and connect

You configure different settings and options for the session, as well as connect to the host, in the web client browser window.

1. From the left panel, click **Connection.** On the **Connection** panel, for the session you are creating, choose the host type, and enter the name and port number of the host.



2. Connection settings vary depending on the type of host connection. For detailed descriptions of the setting options for each host type, see the web client help. Setting options include mapping keystrokes to selected keys, mapping host colors to match your preferences, and recording session macros. For this walk-through we will map a few keys and record a macro.
3. To map keys to selected keys, open **Key Mappings**.
4. Press the key or key combination you want to use to trigger the selected action.



5. From the **Action** drop down list select the action you want mapped to the selected keystroke. Click  to complete the key mapping. You can continue adding and mapping keys.
6. Click **Save** to complete mapping keys.
7. From the left navigation panel, you can map host colors, set font and keyboard options, and enable hotspots by opening the **Display** panel. Color choices are specific to each session.

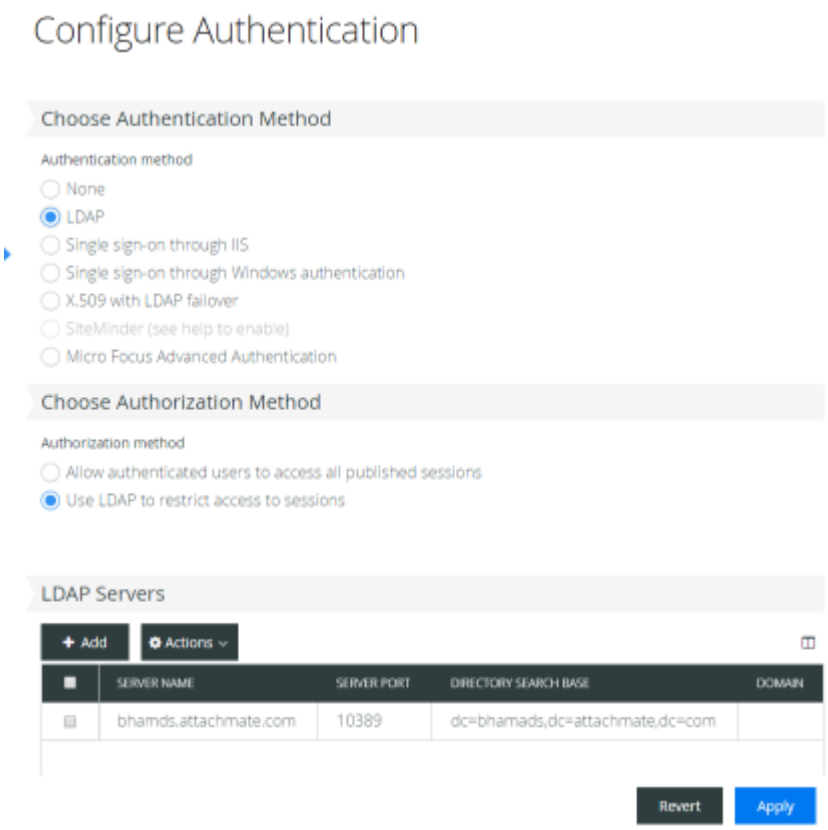
8. Under **Macros** set various macro options. See Creating Macros for instructions on how to record, create, and edit macros.
9. To set file transfer settings before you connect to the host, open **File Transfer**.
10. Open **User Preference Rules** to extend configuration options to your end users.
11. Click **Exit** to return to the Administrative Console browser window and assign users to the session you have created.

Assign users to sessions

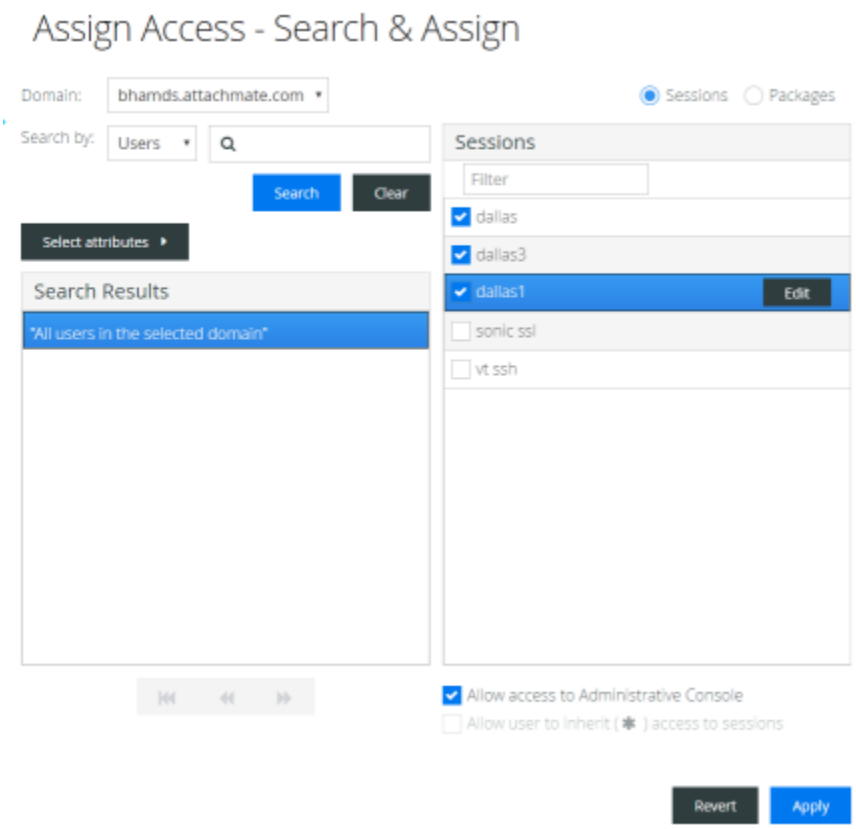
Now that sessions are created, you need to grant users access to those sessions. Using the URL you provide, each user has access to the sessions you assign to him. A user can be assigned to multiple sessions.

Users are assigned to sessions in the access and authentication panels of the MSS Administrative Console.

1. Authentication and authorization validates the identity of a user and the method you want to use to map sessions to individual users or groups of users. From the left navigation panel, select **Configure Authentication**.
2. Choose an authentication method. Your options change depending on your selection.



3. There are descriptions of the various options in the MSS documentation. Click ?.
4. Click **Apply** to complete the process.
5. Open **Assign Access** to map sessions to individual users or groups of users.



6. Map the sessions to the users you want to access the sessions and click **Apply.** You can also choose to allow users to inherit access to sessions and to the Administrative Console.

The next steps

Your users have been assigned to sessions you created, You have authentication and authorization in place, now it's time to make your legacy data available through the web browser. The Reflection ZFE Web client provides your users with just that ability.

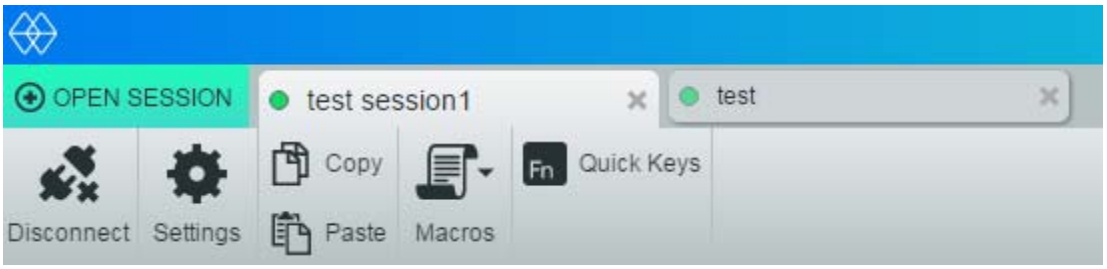
How do users interact with the session?

It really is as simple as clicking a link. The connection URL to the Reflection ZFE web client usually looks something like this:

`https://myserver.mycompany.com:port/zfe`

As an administrator you can share the primary Reflection ZFE login URL with your users. This address opens the web client and provides access to the Reflection ZFE sessions assigned to them. Users may have to login if configured as such.

Disconnecting, closing and opening new sessions, and other functions, such as recording macros, are available from the toolbar.



For more information on Reflection ZFE

For more information about Reflection ZFE, review the product Help. For further assistance regarding evaluation software and product updates, visit our Technical Support site.

Installing Reflection ZFE

You can install Reflection ZFE from the Micro Focus Download site.

In this section

See Setting Post-Installation Options for information on:

- How to Adjust Session Timeout Values
- How to Set Up the Terminal ID Manager
- How to Set Up Metering
- How to Start and Stop Services Automatically
- How to Change Ports
- How to Set Up Automated Single Sign-On for Mainframe

Before you install

Keep these things in mind when installing Reflection ZFE.

- **Host Access Management and Security Server**

  ✓ Host Access Management and Security Server (MSS) is used for session management: MSS is installed with Reflection ZFE in a typical installation, however, you can use an existing MSS installation if that works better for you. The install program will install MSS, the ZFE session server, and documentation to a single machine. Different components can reside on different machines.

- ✓ You will be asked for the user name and password for the MSS machine used by Reflection ZFE. It is a good idea to have those credentials in hand before you start installation.
- ✓ MSS uses activation files (activation.jaw) to enable product functionality. The Reflection ZFE install program contains the needed activation file to enable communication between Reflection ZFE and MSS and is typically activated as part of the install process. You can, if necessary, activate the product in Administrative Console >Configure Settings >Product Activation. Also, you will need to provide an activation file if you intend to use an already installed or remote MSS server that has not been activated for use with Reflection ZFE. It is important that you install compatible versions of both products. You can read all about MSS activation files and learn the activation process in the Host Access Management and Security Server Installation Guide.

- **Reflection ZFE and Activation Files**

  To work in a production environment, activation is required. Activation files are downloaded from the Micro Focus download site along with the install package and are specific for the different editions of Reflection ZFE.

  Remember that after you install, if activation was not part of the installation, you need to open the Administrative Console (Configure Settings > Product Activation) and complete the product activation process. See Upgrading from previous versions for information on handling activation files when you upgrade.

- **Downloading Reflection ZFE**

  The Micro Focus download site contains the compressed files necessary to install all supported platforms, including the Windows connector. Different activation files will enable different editions/platforms of Reflection ZFE.

- **Reflection ZFE and Java**

  The Reflection ZFE session server requires a Java JDK version 8 or higher and MSS requires a Java JRE version 8 or higher. This Java requirement is met during installation, except for systems, such as Linux on System Z that require an IBM JDK. See Installing on z/Linux (SUSE E11.x and RHEL 6.x) for information on using the *nojdk* option.

  Both Reflection ZFE and MSS require that the Java installation support unlimited strength encryption. More information is available on the Java web site.

  If necessary, you can use the environment variables specified in the *nojdk* option and `INSTALL4J_JAVA_HOME_OVERRIDE` to specify a specific Java installation.

- If you plan on using the IIS Reverse Proxy with Reflection ZFE, read Accessing Reflection ZFE using the IIS Reverse Proxy for prerequisites and configuration instructions.

System Requirements

NOTE: All requirements listed are the **minimum** required to successfully install Reflection ZFE.

Supported web browsers

The only thing needed to access Reflection ZFE terminal emulation is a supported web browser. The following web browsers are currently supported:

- Google Chrome 33+
- Mozilla Firefox 27+
- Microsoft Internet Explorer 11

  See Browser issues for information on performance issues when using Internet Explorer.

- Microsoft Edge
- Apple iOS Safari 7+

MSS is platform independent and supports any web browser that supports JavaScript and Cascading Style Sheets (CSS).

Session server operating systems

The Reflection ZFE session server supports the following 64-bit platforms:

- Windows 2008 Server
- Red Hat Enterprise Linux (RHEL) 6.x
- SUSE Enterprise Linux 11.x

Installing on z/Linux (SUSE E11.x and RHEL 6.x)

For systems, such as Linux on System Z, that require an IBM JDK, you can use the "*nojdk*" installer media, which does not include a bundled JDK.

— The installation must be able to locate a Java executable to start. If a Java executable cannot be found by the installer, then you can set the `INSTALL4J_JAVA_HOME` environment variable to refer to a Java installation's `bin` directory.

— When started, the installation program will automatically search for version-compatible JDKs on the system. If more than one JDK is found, a list is displayed from which you can choose. If only a JRE is found on the system, you can continue with the installation, but the Reflection ZFE server will not run correctly until you have updated the `wrapper.java.command` property located in `sessionserver/container.conf` to refer to a JDK installation.

If necessary, you can use the environment variables named above and `INSTALL4J_JAVA_HOME_OVERRIDE` to specify a specific Java installation.

Installing on UNIX platforms

- You must either install as "root" or use a user account with root privileges to complete successfully. When the installation has successfully completed, the installed application can be started and managed by "root" or someone running as 'root".
- If you are running on Linux platforms, follow these steps to set the session server to start automatically when your system first boots up.
- Elevated privileges are needed to open any application ports lower than 1024. Reflection ZFE will not start using a lower port number unless you have system privileges to open low numbered ports.
- You can use the `chmod` command to assign application privileges to users other than root.
- If you are installing on a headless Linux system and there are no fonts installed on the system, you may encounter this font-related error: `java.lang.Error: Probable fatal error: No fonts found`. Ensure that `fontconfig` or at least one font is installed on the system in order to proceed with the installation.

Preparing to install

Reflection ZFE supports TLS and SSH protocols to protect mission-critical data. To secure your passwords and other sensitive data, you should require browsers to use the HTTPS protocol.

To configure a Reflection ZFE session to use TLS, you must first establish a “trust” for the public certificate chain of the host to which you’re connecting. MSS centrally manages the trust store that Reflection ZFE uses. By default, the Reflection ZFE session server fetches this trust store every time it attempts a connection.

For a successful installation you must have a valid certificate signed by a trusted Certificate Authority (CA) and install it on the session server. To head off any installation issues, read Making Secure Connections. In a typical Reflection ZFE installation there are three main connection points that you need to consider in regard to security, the Making Secure Connections topic deals with all three; web browser to Reflection ZFE session server, Reflection ZFE session server to MSS, and Reflection ZFE session server to the host legacy system.

Simple install

1. From the Micro Focus download site, download your product install package. The package includes support for all supported platforms.
2. Download the activation file for the associated Reflection ZFE Edition.
3. Following the install program prompts, install Reflection ZFE and if needed, Management and Security Server (MSS).
4. Open the MSS Administrative Console and add the downloaded activation file.

Ports used by Reflection ZFE

Configure your firewall to allow connections on the following TCP listening ports:

Component	Default Port Numbers
Reflection ZFE session server	7070 - HTTP
	7443 - HTTPS
MSS	80 - HTTP
	443 - HTTPS

Both the Reflection ZFE and the MSS Administrative Server ports can be changed depending on your network needs. To modify the Reflection ZFE session server ports, see How to Change Ports.

Upgrading from previous versions

It’s best to back up any previous work before you upgrade.

1. From the Micro Focus download site, download the install package and activation files for the version you are upgrading to.
2. Remove any activation files from MSS associated with prior versions of Reflection ZFE. Leaving obsolete activation files in place may result in limited access to sessions.

3. Install Reflection ZFE.
4. If not handled during the installation process, install the new activation file or files into MSS using Administrative Console > Configure Settings > Product Activation.

Troubleshooting the Installation

To complete a successful installation, make sure that you have taken care of these common issues:

- ***Are the activation files installed and activated in the Administrative Console?***

  MSS uses activation files to enable product functionality. With your installation you received an activation file associated with the type of host you are connecting to. For example, if you are licensed for the Unisys Edition, if not handled as part of the install process, you will need to open the Administrative Console, go to Configure Settings > Product Activation and verify that both the Reflection ZFE Unisys activation file is in place.

- ***Is MSS configured for HTTPS?***

  Connect to the system where the Administrative Server is installed and log in to the Administrative Server. In the Administrative Console, open the Security Setup section and note the protocol selection.

- ***Verify that both MSS and Reflection ZFE are using trusted certificates.***

  MSS imports certificates and private keys to `C:\ProgramData\Micro Focus\MSS\MSSData\certificates`.

  If you are not using trusted certificates, have you configured Reflection ZFE to run using HTTP?

- ***Are your connection properties configured properly?***

  In the unlikely event that you have to verify connection information, the `container.properties` file for both the management component and the Reflection ZFE session server contains the connection properties needed to make the Reflection ZFE to MSS connection as well as the browser to Reflection ZFE connection.

  You can find the file in the Reflection ZFE installation at `<install-dir>/sessionserver/conf/container.properties`.

Connecting using HTTP

If you do not have a trusted certificate in place, you can configure Reflection ZFE to use HTTP. This configuration is not secure and should be used only when no other option is available.

Connecting to...	Do this...
An existing remote MSS Administrative Server	1. During the Reflection ZFE installation, after you accept the license agreement and choose a destination directory, select Use remotely hosted MSS. Click Next. 2. Enter either the host name, DNS name, or IP address. 3. Change the port from 443 to 80. 4. Select HTTP and complete the installation process.
The MSS Administrative Server that is installed with Reflection ZFE	1. Select Install MSS and follow the installation instructions. 2. Clear the **Perform this action** option and click Finish. If this option is not disabled, you can open `<install-directory>\Micro Focus\ReflectionZFE\sessionserver\conf\container.properties` in a text editor and change 443 to 80 in the following line: `management.server.url=http://yourmachine:80/mss` If this option is not cleared, an internal error is generated and you will be asked to contact your system administrator. 3. Restart the Reflection ZFE Session Server service.

Other known issues

This section documents miscellaneous known issues and work around tips for Reflection ZFE.

- SSL/TLS error message issues
- Install does not complete on UNIX or Linux platforms

SSL/TLS error message issues

- **(ECL1011) Error connecting to host: Connection to host failed.**

  This error can display in a number of situations that are not simply due to a connection failure.

  - You may see this error if an SSL/TLS connection failed due to the lack of a trusted certificate in the MSS trust store.
  - This error displays when a SSL/TLS handshake failure occurs when you use TLS to connect to or from a plain text host.

Install does not complete on UNIX or Linux platforms

The Reflection ZFE install program may stall on UNIX or Linux systems, particularly headless ones. This stall is caused by an insufficient amount of entropy in the system, typically due to a lack of interaction with the operating system's UI (or lack of UI).

To remedy the issue:

1. Stop the installation process.
2. On the installer's command line, prepend –J to the Java System property: `./reflectionzfe-xxxx-linux-x64.sh -J-Djava.security.egd=file:///dev/urandom`
3. Run the installation program containing the added argument.

Related Topics

Setting Post-Installation Options
Making Secure Connections

3 Managing ZFE

Creating and configuring sessions, setting advanced post-install options, and making sure everything runs smoothly and securely means that your users will be successful. The following should help you administer and manage your Reflection ZFE sessions and host connections.

Setting Post-Installation Options

There are a number of post-installation configurations that you can make to ensure that Reflection ZFE runs successfully.

How to Adjust Session Timeout Values

The default timeout value for an inactive Reflection ZFE session is 30 minutes. This means that a session that was not logged out and has had no activity will close after 30 minutes. You can configure this setting on the server.

1. Open `<install_location>Micro Focus\ReflectionZFE\sessionserver\webapps\zfe|WEB-INF\web.xml.`
2. Adjust the session timeout value:

```
<session-config>
   <session-timeout>30</session-timeout> <!--In minutes-Minimum values of 5-->
   <cookie-config>
         <max-age>604800</max-age> <!--1 week in seconds-->
   </cookie-config>
</session-config>
```

3. Restart the server.

How to Set Up the Terminal ID Manager

The Management and Security Server provides a Terminal ID Manager to pool terminal IDs, track ID usage, and manage inactivity timeout values for specific users, thus conserving terminal ID resources and significantly reducing operating expenses.

The Terminal ID Manager Add-On requires a separate license.

Before you configure the Terminal ID Manager for Reflection ZFE, verify that you have this option enabled for MSS. There are complete instructions in the MSS Installation Guide.

TIP: If MSS and Reflection ZFE are installed on the same machine and using port 80, no additional configuration is needed.

Configuring Terminal ID Manager for Reflection ZFE

To configure the Terminal ID Manager for Reflection ZFE, you must provide the correct address to the Terminal ID Manager.

1. Open the `sessionserver/conf/container.properties` file.
2. Update `id.manager.server.url=http://localhost:80/tidm` to reflect the address of the Terminal ID Manager configured in Management and Security Server.
3. Restart the Reflection ZFE Session Server.

How to Set Up Metering

The Management and Security Server provides metering capabilities to monitor Reflection ZFE host sessions.

Before you configure metering for Reflection ZFE, verify that you have metering enabled for MSS. There are complete instructions in the MSS Installation Guide.

In Reflection ZFE metering is set globally for all emulation sessions created by the Reflection ZFE session server. Settings are configured in the `sessionserver/conf/container.properties` file.

Table 3-1 *Metering options*

Property	Description
`metering.enabled`	Turns metering on or off, with a value of "true" or "false". Any value other than "true" turns metering off.
`metering.host.required`	Determines whether the session can connect to the host even if the metering server cannot be contacted. "True" means that session connections will fail if the metering host is unavailable. "False" means that session connections will still work even if the metering host is unavailable.
`metering.server.url`	Specifies the name or address of the metering server, the port, the protocol, and the webapp context. The syntax is "`host:port|protocol|context`". This syntax is the same as that used by the MSS server in the `MssData/serverconfig.props` file to register metering servers. The host:port section of the URL must escape the ":" character. For example, `test990.attachmate.com\:8080`.

```
#Example additions to sessionserver/conf/container.properties
metering.enabled=true
metering.host.required=false
metering.server.url=10.10.11.55\:80|http|meter
```

NOTE: In the event that all licenses are in use and you attempt to make a connection, the session will be disconnected. To determine whether the host has disconnected or the metering service has stopped the connection, see the `Reflection ZFE/sessionserver/logs/server.log` file.

How to Start and Stop Services Automatically

All server components are installed as services and can be configured to start during installation.

If you are running on Linux platforms, follow these steps to set the session server to start automatically when your system first boots up.

Create a file called `zfe` containing the following and using your installation directory:

```
#!/bin/sh
#
#This script manages the service needed to run the session server
#chkconfig:235 19 08
#description:Manage the Reflection ZFE session server

###BEGIN INIT INFO
# Provides:          zfe
# Required-Start:    $all
# Required-Stop:     $all
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Description:       Start the Reflection ZFE Session Server
### END INIT INFO

INSTALL_DIR=<enter installation directory>
BIN_DIR=$INSTALL_DIR/sessionserver/bin
case "$1" in
start)
echo "Starting Reflection ZFE Session Server"
$BIN_DIR/server start

RETVAL=0
;;
stop)
echo "Stopping Reflection ZFE Session Server"
$BIN_DIR/server stop

RETVAL=0
;;
status) echo "Current Reflection ZFE Session Server status"
$BIN_DIR/server status
```

```
RETVAL=0
;;
restart) echo "Restart Reflection ZFE Session Server"
$BIN_DIR/server restart

RETVAL=0
;;
*)
echo "Usage: $0 (start|stop|status|restart)"

RETVAL=1
;;
esac
exit $RETVAL
```

Then complete the relevant steps.

Platform	Follow these steps
Linux	1. Copy the file to the `/etc/init.d` directory. 2. Set the file permission. Run `chmod` using the value 755. For example, `chmod 755` zfe 3. Run `chkconfig` to add the initialization script. For example, `/sbin/chkconfig --add zfe`

How to Change Ports

Both the Reflection ZFE session server and MSS ports can be modified depending on your network needs. The default ports used by Reflection ZFE are:

Table 3-2 *Reflection ZFE and MSS Default Ports*

Session server	HTTP - 7070 HTTPS - 7443
Management and Security Server	HTTP - 80 HTTPS - 443

To change the default ports:

Table 3-3 *Changing default ports*

Component	Instructions

Reflection ZFE session server	The Reflection ZFE session server ports are set, and can be modified, in `sessionserver/conf/container.properties`.
	
	`servletengine.port=7070`
	`servletengine.ssl.port=7443`
	
	To turn the port off, set the port value to 0. You can disable your non-secure SSL port by changing the value from 7070 to 0.

Component	Instructions
Management and Security Server	The SSL port MSS uses to make an HTTPS connection is set to 443 by default. If you need to change the port number, start the Management Server. This creates the default PropertyDS.xml file. Then, open PropertyDS.xml in the MssData directory. Change the value from 443 to the appropriate port number in the section below, and then restart the Management Server. `<CORE_PROPERTY NAME="sslport"> <STRING>443</STRING>`

How to Set Up Automated Single Sign-On for Mainframe

Automated Sign-On for Mainframe is an add-on product to Management and Security Server that enables an end user to authenticate to a terminal emulation client and be automatically logged on to a host application on the z/OS mainframe.

The Management and Security Server Administrator Guide for Automated Sign-On for Mainframe has complete information on configuring this option.

1. Install and configure the Automated Sign-On for Mainframe add-on for Management and Security Server. You can find complete instructions here.
2. After the Management and Security Server setup is complete, open the Administrative Console to add sessions and map users to those sessions. During that process, you can complete the additional configuration needed to implement automated sign-on.
3. A Reflection ZFE macro sends the user's mainframe username and pass ticket to the host application. The user is then automatically logged in. To help create the macro:
   - The Macro API contains the AutoSignon object that provides the methods needed to create a Reflection ZFE login to use with the Automated Sign-On for Mainframe feature.
   - You can also reference the sample macro Automatic Sign-On Macro for Mainframes that uses the AutoSignon object to create a macro that uses the credentials associated with a user to obtain a pass ticket from the Digital Certificate Access Server (DCAS).

Connecting to the Host

Reflection ZFE supports IBM 3270, 5250 and VT hosts as well as UTS, T27 and ALC host types.

To connect to the host:

1. From the **Type** drop down list, select the type of host you are connecting to.
2. Identify the host to which you want to connect. You can use a full host name or its IP address.
3. Type the number of the port you want to use.
4. Complete the information needed for the host connection.
5. Save your connection settings.

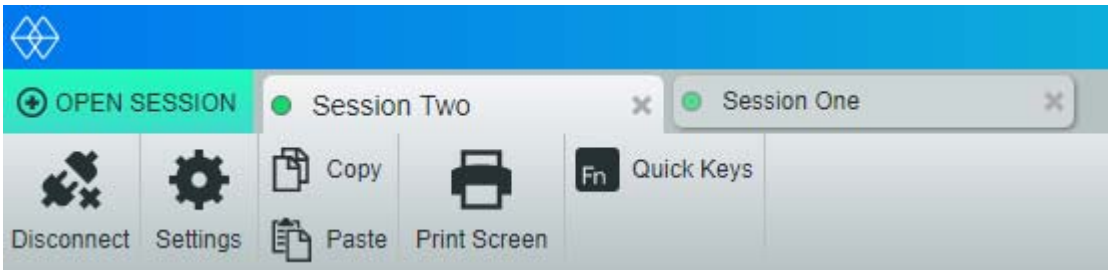
Your users gain access to the host through sessions that you create and configure. Sessions are created by an administrator in the MSS Administrative Console. When you launch a session from the Administrative Console, the web client Connection panel opens in a separate browser window. You configure connection options from this panel. Options vary depending on your host type.

- Providing access to the session
- Common connection settings

- 3270 and 5250 connection settings
- How to test Terminal ID Manager criteria
- VT connection settings
- UTS connection settings
- T27 connection settings
- ALC connection settings

Providing access to the session

Your users have access to their assigned sessions through a URL you provide (for example, `https://<sessionserver>:7443/zfe`). From this URL users select which session to open from the list of available sessions you have configured for them.

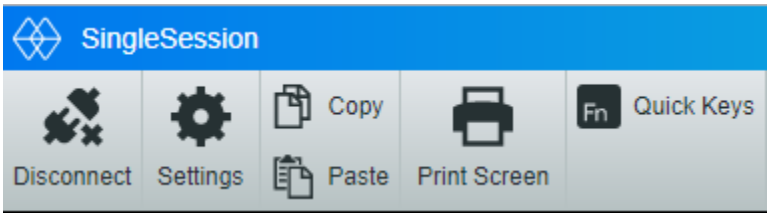


Your users can switch between sessions, open additional sessions and close sessions with which they are no longer working.

NOTE: A new session will not be launched if the specified session already exists when the user opens the link.

Alternatively, you can use **single session mode** and provide URLs to particular sessions that are launched using name and session parameters, (for example a direct link on a company portal page). To enable the launch of a single session use the query parameter `singleSession`. You can use this parameter on its own to just launch the Reflection ZFE web client in single session mode, for example, `http://<sessionserver>:7443/zfe/?singleSession`, or it can be used in conjunction with a named session parameter to launch a particular named session in single session mode: `http://<sessionserver>:7443/zfe/?singleSession&name=HumanResources`. The order of the parameters does not matter.

When your users access a single session, they cannot switch between open sessions and cannot open new sessions.



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, Reflection ZFE 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 Reflection ZFE 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:

Table 3-4 *Protocol Descriptions*

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)	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 There are two authentication options available: ◆ **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. ◆ **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.
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.

Protocol	Description
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.

- **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. The trace file is stored in `<install directory>/sessionserver/logs/hosttraces/<date(yyyymmdd>/<trace-file>`. Host trace files are created each time a session is launched.

3270 and 5250 connection settings

In addition to the common configuration settings, 3270 and 5250 host types require these specific settings.

- **Terminal model**

  Specify the terminal model (also known as a display station) you want Reflection ZFE 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.

- **Terminal ID** (3270 only)

  When Reflection ZFE 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/SSL Security**

  SSL and TLS protocols allow a client and server to establish a secure, encrypted connection over a public network. When you connect using SSL/TLS, ZFE authenticates the server before opening a session, and all data passed between and the host is encrypted using the selected encryption level. The following options are available:

Table 3-5 *TLS/SSL Descriptions*

Security options	Description
None	No secure connection is required.
TLS 1.2 - 1.0	Allow connection through TLS 1.2, TLS 1.1, TLS 1.0 depending on the capabilities of the host or server to which you are connecting.
TLS 1.2	Select this value to connect using TLS. As part of the TLS protocol, the client checks the server or host name against the name on the server certificate. Therefore, TLS connections require the common name on the server certificate to match the host or proxy server name.

NOTE: See the section on Making Secure Connections for information on adding trusted certificates, key stores, using SSH, and other advanced security information.

- **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**. An unique device name will be automatically generated.
  - **Use Terminal ID Manager** which displays additional settings to complete.
  - **Prompt User**. When you select this option the end user will be prompted for the device ID each time a connection is attempted.

  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.

  If you selected **Terminal ID Manager** you can use it to provide IDs to client applications at runtime. You can use the Terminal ID Manager to manage pooled IDs for different host types. An ID is connection data that is unique for an individual host session. To use Terminal ID Manager, you must have a Terminal ID Manager server configured. See Terminal ID Manager in the Management and Security Server Installation Guide.

  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 that has 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.

Table 3-6 *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.

Criterion	Description
User name	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. To configure a session based on user names, a default place holder user name is available: **tidm-setup**. For the administrator to configure sessions using **tidm-setup**, the Terminal ID Manager needs to have IDs provisioned for tidm-setup. You can override the default name with one of your own by modifying the `Micro Focus/ReflectionZFE/sessionserver/conf/container.properties` file as follows: `id.manager.user.name=custom-username` Where custom-username is replaced by the name you want to use.
Application name (UTS)	The name of the host application will be used as part of the request for an ID.

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** -If selected, the session will not attempt to connect when an ID is not allocated.
- **Allow connection attempt** -If selected, the session will attempt 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.

- **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.
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.

How to 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.

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.

Table 3-7 *VT session configuration options*

VT Settings	Description
Terminal ID	This setting determines the response that Reflection ZFE sends to the host after a primary device attributes (DA) request. This response lets the host know what terminal functions it can perform. The Reflection ZFE 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.
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 Reflection ZFE responds to remote echo from a Telnet host: Automatic attempts to negotiate remote echo, but does what the host commands. Yes means Reflection ZFE negotiates local echo with the host, but always echoes, while No means Reflection ZFE 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. Reflection ZFE supports the Telnet Suppress Local Echo (SLE) option when connected to a host in half-duplex mode. This means that Reflection ZFE will suppress character echo to the host computer, and with SLE support Reflection ZFE can be instructed to suppress echo locally.

VT Settings	Description
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	This setting allows you to enter an answerback message if the host expects an answer in response to an ENQ character. 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. 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. Reflection ZFE will convert this to the string \u001C when that key is pressed and send the 6 characters of the resulting string to the host.

UTS connection settings

In addition to the common connection settings, UTS hosts require these additional settings:

Table 3-8 *UTS INT1 session configuration options*

UTS INT1 options	Description
Application	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.
TSAP	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.
Initial transaction	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.
Start transaction	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. 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.**
Send initial transaction	You can determine when the initial transaction is sent: ◆ **Immediately** - Default. ◆ **When start of entry (SOE) character is received** - Useful when multi-line transactions must be completed before sending the string. ◆ **After specified milliseconds**
Terminal ID	Choose whether to specify a terminal ID or use the Terminal ID Manager. To specify a terminal ID, type it in the **Specify Terminal ID** field. 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**.
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.

T27 connection settings

Along with the common connection settings, you can configure these additional T27 connection options:

Table 3-9 *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	You must enable the Request binary option when you require pass through printing. The default in No. 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.
Line width	Select the number of characters the host will send to the client. The default is 80 characters.
TLS/SSL security	See Table 3-4 TLS/SSL Descriptions for a description of the various options.
Station ID	Choose whether 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. Each station id should be unique to the host and typically consists of up to eight alphanumeric characters. If you do not specify a station id for the session, the host dynamically assigns one to the session. If you select Use Terminal ID Manager, you will see a number of Terminal ID criteria to configure. See Terminal Manager ID Criteria for descriptions of the various options.

ALC connection settings

In addition to the common connection settings, ALC hosts require these additional settings:

Table 3-10 *ALC Connection Settings*

ALC options	Description
TLS/SSL security	See Table 3-4 TLS/SSL 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.

ALC options	Description
Terminal address	Select whether you want to specify the terminal address or use the Terminal ID Manager. ◆ Terminal address - Specify whether to use 2-byte or 4-byte addressing mode. 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. 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. ◆ **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.

Making Secure Connections

Reflection ZFE and Management and Security Server use HTTPS connections and certificates to ensure secure communication between clients, web browsers, and servers.

Certificates are electronic credentials that are used to verify the identities of individuals, computers, or networks. Certificates are stored in key stores along with private keys that are used to complete secure transactions. Public keys encrypt data then the private key decrypts the data. Together, the public and private keys are known as a key pair. All key store entries are identified using a unique identifier, known as an alias.

Reflection ZFE and MSS use Bouncy Castle as the provider for their key store operations and use the .bcfks (Bouncy Castle FIPS key store) extension for cryptographic files. They are:

- `servletcontainer.bcfks` - used to securely connect the Reflection ZFE web application to the session server
- `system.bcfks` - used for X.509 authentication
- `trustcerts.bcfks` - The Reflection ZFE trust store where trusted certificates for connections between the session server and MSS are stored.

Values you must know

- **changeit** is the default password
- **servlet-engine** is the alias of the key pair entry of servletcontainer.bcfks
- **bcfks** is the keystore format

About the tools

- **KeyStore Explorer** - You can take advantage of the KeyStore Explorer utility to provide a simple user interface to create signing requests (CSR) and import CA-signed certificates into Reflection ZFE.
  - To launch Keystore Explorer on Windows - run `\ReflectionZFE\utilities\keystore-explorer.bat` as an administrator or with administrative rights.
  - To launch Keystore Explorer on UNIX - run `ReflectionZFE\utilities\keystore-explorer.sh` as an administrator or with administrative rights..

  The utility has an online Help system available to walk you through the user interface.
- **Java Keytool** - The Java Key and Certificate Management Tool manages a key store of cryptographic keys, X.509 certificate chains, and trusted certificates. It uses a command line interface. The Java Key and Certificate Management Tool documentation is available for both Unix and Windows platforms:
  - Windows (http://docs.oracle.com/javase/8/docs/technotes/tools/windows/keytool.html)
  - Unix (http://docs.oracle.com/javase/8/docs/technotes/tools/unix/keytool.html)

NOTE: Due to a keytool issue, Reflection ZFE starts with a java keystore (.jks) format and then converts the keystore to a .bcfks format after the CA Reply is received.

Making connections

In a typical Reflection ZFE installation there are three main connection points that you need to consider in regard to security:

- Securing the Web Browser to the Session Server
- Securing the Session Server to MSS
- Securing the Session Server to the Host

Securing the Web Browser to the Session Server

NOTE: A connection between the web browser and the session server is made during installation using a self-signed certificate. There is no further configuration necessary unless you want to replace the self-signed certificate provided with a different certificate.

To replace the certificate:

- Create a Certificate Signing Request (CSR) for Reflection ZFE and send it to the Certificate Authority of your choice. When you receive the signed certificate from the CA, then you:
- Import the CA-signed certificate/chain into Reflection ZFE for HTTPS

You can accomplish these tasks using either KeyStore Explorer or the Java Keytool command line instructions.

NOTE: The key store commands noted here are for a default installation and start at the `sessionserver/etc` directory. If you have installed Reflection ZFE to another location, you must modify the path appropriately.

To configure a non-default key store:

You can use a key store other than the default, `servletcontainer.bcfks`, to store your CA signed certificates.

- Specify the following properties in `container.properties` found here:`../ReflectionZFE/sessionserver/conf`:

  ```
  management.server.client.ssl.keyStoreFileName
  management.server.client.ssl.keyStorePassword
  servletengine.ssl.keystore
  servletengine.ssl.keystorepassword
  ```

  Where the key store path is set to the non-default key store file name and the key store password is set to the obfuscated value generated by the following command from the `sessionserver` directory:

  ```
  ../java/jre/bin/java -cp ./services/servletengine/lib/jetty-util-9.4.7.v20170914.jar org.eclipse.jetty.util.security.Password admin passwordToObfuscate
  ```

  For example:

  ```
  management.server.client.ssl.keyStoreFileName=../etc/custom.bcfks
  management.server.client.ssl.keyStorePassword=OBF:1vn21ugu1saj1v9i1v941sar1ugw1vo0
  servletengine.ssl.keystore=../etc/custom.bcfks
  servletengine.ssl.keystorepassword=OBF:1vn21ugu1saj1v9i1v941sar1ugw1vo0
  ```

- Finally, since you are using a custom key store, there is no longer a reason to generate the default key store when the service starts. Edit `service-ctx.xml`, located in `../sessionserver/services/servletengine/META-INF`, to remove references to `servletEngineKeystoreGenerator`. Comment out the highlighted section, and remove reference to `depends-on="servletEngineKeystoreGenerator"`.

  As follows:

```
<bean id="servletEngineKeystoreGenerator" class="com.attachmate.integration.container.ssl.KeyManagerFactorySC" init-method="initialize">
    <property name="alias" value="servlet-engine"/>
    <property name="country" value="US"/>
    <property name="keyAlgorithm" value="RSA"/>
    <property name="keySize" value="2048"/>
    <property name="keystoreName" value="../etc/servletcontainer.bcfks"/>
    <property name="keystorePassword">
        <bean id="deobfuscator" class="org.eclipse.jetty.util.security.Password" factory-method="deobfuscate">
            <constructor-arg value="${servletengine.ssl.keystorepassword:OBF:1vn21ugu1saj1v9i1v941sar1ugw1vo0}"/>
        </bean>
    </property>
    <property name="organization" value="Micro Focus"/>
    <property name="organizationalUnit" value="Host Connectivity"/>
    <property name="serverCert" value="../etc/servletcontainer.cer"/>
    <property name="signatureAlgorithm" value="1.2.840.113549.1.1.11"/>
    <property name="validity" value="3650"/>
</bean>

<bean id="servletEngineSslContext" class="com.microfocus.servletengine.jetty.ServletEngineSslContextFactory" depends-on="servletEngineKeystoreGenerator">
    <property name="keyStorePath"     value="${servletengine.ssl.keystore:../etc/servletcontainer.bcfks}"/>
    <property name="keyStoreType"     value="${servletengine.ssl.keystoretype:BCFKS}"/>
    <property name="keyStorePassword" value="${servletengine.ssl.keystorepassword:OBF:1vn21ugu1saj1v9i1v941sar1ugw1vo0}"/>
    <property name="trustStoreType"   value="${servletengine.ssl.keystoretype:BCFKS}"/>
    <property name="includeProtocols" value="TLSv1,TLSv1.1,TLSv1.2"/>
    <property name="excludeCipherSuites" value="^.*_(MD5)$"/>
</bean>
```

How to create a CSR using KeyStore Explorer

TIP: You should run the following commands, using administrative privileges, from the `sessionserver\etc` directory.

Open `sessionserver/etc/servletcontainer.bcfks` in KeyStore Explorer. Use the password **changeit**.

To create a CSR you will create a key pair and then generate a certificate request. If you do not need to update certificate information, you can skip creating the key pair and proceed to generating the certificate request.

TIP: The KeyStore Explorer utility has documentation available to aid you in these tasks.

Create a new key pair

1. Select the `servlet-engine` key pair.
2. From the right-click menu, click **Delete** and then click **Yes** to confirm.
3. From the Tools menu, select **Generate Key Pair**.
4. On the Generate Key Pair dialog box, enter the appropriate algorithm information and certificate details. Click OK.
5. Enter `servlet-engine` as the alias. Click OK.
6. Enter `changeit` as the password and click OK.

Generate a certificate request

1. Select the `servlet-engine` key pair.
2. From the right-click menu, select **Generate CSR.**
3. Browse to the file location where you want to generate the CSR and enter the file name. Click OK.

How to create a CSR using Java Keytool

Windows

Create Key Pair (replace the `dname` parameter with your own) :

```
..\..\java\bin\keytool.exe -genkeypair -dname "CN=zfe-1.microfocus.com, O=Micro
Focus, C=US" -alias servlet-engine -keyalg RSA -keysize 2048 -keystore
servletcontainer.jks -validity 1095 -storetype jks -storepass changeit -keypass
changeit
```

Generate Certificate Request:

```
..\..\java\bin\keytool.exe -certreq -alias servlet-engine -keystore
servletcontainer.jks -file cert_request.csr -ext ExtendedkeyUsage=serverAuth -
storetype jks -storepass changeit
```

UNIX

Create Key Pair (replace the `dname` parameter with your own) :

```
../../java/bin/keytool -genkeypair -dname "CN=zfe-1.microfocus.com, O=Micro Focus,
C=US" -alias servlet-engine -keyalg RSA -keysize 2048 -keystore
servletcontainer.jks -validity 1095 -storetype jks -storepass changeit -keypass
changeit
```

Generate Certificate Request:

```
../../java/bin/keytool -certreq -alias servlet-engine -keystore
servletcontainer.jks -file cert_request.csr -ext ExtendedkeyUsage=serverAuth -
storetype jks -storepass changeit
```

After you receive the certificate from the CA, you will import the certificate into Reflection ZFE.

How to import a CA-signed certificate into Reflection ZFE

For both KeyStore Explorer and the Java Keytool, if the CA Reply contains separate root and intermediate certificate files, import the root certificate into the keystore first, followed by the intermediate certificate.

Using KeyStore Explorer

1. Open `servletcontainer.bcfks` in KeyStore Explorer. Use the password **changeit**.
2. If separate root and intermediate certificate files are available, from the tool bar, select **Import Trusted Certificate** to import certificates.
3. Select the `servlet-engine` key pair. Right-click and select **Import CA Reply** to import the file into the key pair.
4. If prompted, enter the password, **changeit**.
5. Browse to the location where the CA Reply file is stored, select the file, and click Import.

Using Java Keytool

These examples use keytool commands at the `sessionserver/etc` directory.

Windows

Import Root CA and intermediate certificates

```
..\..\java\bin\keytool.exe -importcert -alias rootca -trustcacerts -file
<RootCA.cer> -keystore servletcontainer.bcfks –storetype bcfks -storepass changeit
```

```
..\..\java\bin\keytool.exe -importcert -alias intermediateca -trustcacerts -file
<IntermediateCA.cer> -keystore servletcontainer.jks –storetype jks -storepass
```

Import CA Reply

```
..\..\java\bin\keytool.exe -importcert -alias servlet-engine -trustcacerts -file
<CertChainFromCA.p7b> -keystore servletcontainer.jks –storetype jks -storepass
changeit
```

Convert keystore to BCFKS format

Before running the following command, rename existing `servletcontainer.bckfs` to `servletcontainer.bckfs_prev`:

```
..\..\java\bin\keytool.exe -importkeystore -srckeystore servletcontainer.jks -
srcstorepass changeit -destkeystore servletcontainer.bckfs -deststoretype bcfks -
deststorepass changeit -providername BCFIPS -providerpath ..\lib\bc-fips-1.0.1.jar
-providerclass org.bouncycastle.jcajce.provider.BouncyCastleFipsProvider
```

UNIX

Import Root CA and intermediate certificates

```
../../java/bin/keytool -importcert -alias rootca -trustcacerts -file <RootCA.cer> -
keystore servletcontainer.bcfks –storetype bcfks -storepass changeit
```

```
../../java/bin/keytool -importcert -alias intermediateca -trustcacerts -file
<IntermediateCA.cer> -keystore servletcontainer.jks –storetype jks -storepass
```

Import CA Reply

```
../../java/bin/keytool -importcert -alias servlet-engine -trustcacerts -file
<CertChainFromCA.p7b> -keystore servletcontainer.jks –storetype jks -storepass
changeit
```

Convert keystore to BCFKS format

Before running the following command, rename existing `servletcontainer.bckfs` to `servletcontainer.bckfs_prev`:

```
../../java/bin/keytool -importkeystore -srckeystore servletcontainer.jks -
srcstoretype  jks -srcstorepass changeit -destkeystore servletcontainer.bckfs -
deststoretype bcfks -deststorepass changeit -providername BCFIPS -providerpath ../
lib/bc-fips-1.0.1.jar -providerclass
org.bouncycastle.jcajce.provider.BouncyCastleFipsProvider
```

Securing the Session Server to MSS

To secure the session server to MSS, using the Administrative Console, you register the session server with its associated MSS. When you follow this process, secure connections are handled automatically.

To register the session server:

1. Open the MSS Administrative Console.
2. From the upper left panel, click the arrow and choose Reflection ZFE

3. On the right panel, click **Add** to register the session server. There is help available for each panel.

Securing the Session Server to the Host

Follow these steps to configure a TLS connection between the Reflection ZFE session server and a host that supports TLS:

1. Configure the keystore using the MSS Administrative Console.
2. Configure the Reflection ZFE terminal session.

How to configure the keystore in MSS

For a Reflection ZFE session to trust the TLS host it connects to, the public certificate of the host must be added to a trusted keystore using the Management and Security Server (MSS). The Reflection ZFE session retrieves this certificate the first time a session connects.

Open the MSS Administrative Console > Configure Settings > Trusted Certificates and choose **Terminal Emulator Clients**. You can access the documentation for the Administrative Console by clicking the Help icon in the upper right of the page.

When the certificate is successfully added to the MSS server's trusted keystore, you are returned to the list of certificates and you should see the new host.

How to configure a Reflection ZFE terminal session

Depending on your host type, you can configure a terminal session using different security protocols.

To configure a terminal session using TLS

To connect to the new trusted host using TLS, configure a Reflection ZFE terminal session as usual, and in the Settings dialog box, specify TLS as the security protocol. Make sure to specify the correct TLS port for the connection.

To configure a VT terminal session using Secure Shell (SSH)

Secure shell provides encrypted communications between the client and a VT host.

MSS has a known hosts list that contains the public keys of hosts that you can connect to using SSH. SSH connections can be made only to hosts already trusted by an administrator.

The first time an SSH connection is made from a Reflection ZFE session to a host, the known hosts file is downloaded from MSS to the Reflection ZFE session server.

When you attempt to create or edit a session using SSH in the session management panel, you will be notified if the key is not recognized as trusted and asked if you want to trust the key and continue.

- If you enter yes, the host will be trusted and added to the known host list, and you will be prompted for the SSH host password.
- If you do not answer yes, then the host will remain untrusted and the session will be disconnected.

You can also configure the SSH known hosts file manually by establishing an SSH connection from a Reflection ZFE session to the host, and adding the remote host’s key fingerprint to the known hosts list in MSS.

Configure known hosts file for SSH connections using MSS

To configure the known hosts file for SSH connections in MSS:

1. Connect to the system where MSS is installed and navigate to the server's certificates folder: `C:\ProgramData\Micro Focus\Mss\MssData\certificates` (Windows) or `/var/opt/microfocus/mss/Mssdata/certificates` (UNIX).
2. Copy the public certificate file of the new SSH host into the `MssData/certificates` (Windows) or `/etc/ssh/ssh_host_rsa_key.pub` (UNIX) folder described above. Only `ssh-rsa` and `ssh-dss` are valid as public key types for MSS known_hosts entries.

   The host's public key format can be OpenSSH, Base64-encode,.DER, or.PFX. The file should follow this format: hostname, IP-address key-type key. For example, a public key entry might look like this: `alpsuse132, 10.117.16.232 ssh-rsa AAAAB3NzaC1yc2EAAAADAQABA.....`
3. Log in to MSS (for example, `http://mycompany.com/adminconsole`).
4. Open the **Administrative Console**.
5. Click Configure Settings > Secure Shell.
6. Follow the directions in MSS to import a known host.

After the public key is imported into the known hosts file, you will return to the Secure Shell Known Hosts page and the new host will appear in the list.

Configuring X.509 Authentication

X.509 client authentication allows clients to authenticate to servers with certificates rather than with a user name and password by leveraging the X.509 public key infrastructure (PKI) standard.

Typical installations handles this process for you, if needed the manual instructions are as follows:

Prerequisite

- Using the procedure described for a manual configuration in Securing the Session Server to MSS verify that a trusted certificate has been installed in the certificate store. The procedure may vary depending on your operating system and browser.

Basic steps:

1. Install the signing authorities certificate into MSS and Reflection ZFE.
2. Restart the servers.
3. Configure X.509 in the Management and Security Server Administrative Console.

Step 1. Install the signing authorities certificate into MSS and Reflection ZFE

MSS's trusted store may already contain your signing authority certificate. This is often the case with well-known certificate signing authorities, and if so, then you can skip this step.

To check:

Open the Administrative Console, click Configure Settings, and open the Trusted Certificates tab. Open **Trusted Root Certificate Authorities** to see a list of available certificates.

If your certificate is not listed you need to install your signing root CA into MSS and into the Reflection ZFE session server following the prompts and documentation in the Administrative Console.

To install the certificate into the Reflection ZFE session server:

In <RZFE_install_directory>\sessionserver\etc import the certificate: `keytool -importcert -file <cert-file> -alias <alias-to-store-cert-under> -keystore servletcontainer.bcfks -storetype bcfks -providername BCFIPS -providerclass org.bouncycastle.jcajce.provider.BouncyCastleFipsProvider -providerpath ../lib/bc-fips-1.0.1.jar -storepass changeit`

Step 2. Restart all the servers

For the configuration to take effect, you must restart all servers.

Step 3. Configuring X.509 with LDAP fail over in the MSS Administrative Console

Once the certificates are in place, you can enable X.509 with LDAP fail over in **Management and Security Server Administrative Console | Assign Access**. See the Administrative Console online help for descriptions of the configuration options.

Configuring Single Sign-on through IIS

This option uses Microsoft IIS web server. This option requires no additional setup as long as you used the Management and Security Server automated installer and chose to integrate with IIS during the installation process. You can find more information on install configurations in the Management and Security Server documentation.

Enabling Reflection ZFE for use with single sign-on through IIS

To enable Reflection ZFE to work with this authentication method, add the following property in the `<install dir>/sessionserver/conf/container.properties` file:

```
management.server.iis.url= <url>
```

The value of this property is the IIS web server address and port along with the / MSS path. For example: `http://server/mss`. If authentication fails, you may need to remove the domain name in order for the domain credentials to be passed to `IIS: http://server/mss`.

Using the IIS Reverse Proxy with Reflection ZFE

NOTE: To comply with Common Criteria security requirements it may be necessary to place the Reflection ZFE server behind a proxy by following the instructions in Accessing Reflection ZFE using the IIS Reverse Proxy.

To proxy Reflection ZFE through IIS, when using IIS single sign-on, you need to set an additional property in the same `container.properties` file:

```
servletengine.iis.url=<url>
```

The value takes the same form as the URL above, but uses the Reflection ZFE address. For example: `http://server/zfe`. It is not necessary to use the short host name form in this URL.

After you have completed this configuration, you choose this authentication option in **Management and Security Server Administrative Console | Assign Access**. See the Administrative Console online help for descriptions of the configuration options.

Enabling FIPS Level Security

The Federal Information Processing Standards (FIPS) 140-2 validated cryptographic modules are used by the US federal government as a security regulation standard. Reflection ZFE supports this standard and you can easily enable FIPS mode by changing a property setting in the session server and in the management component.

To enable FIPS mode:

There is a `container.conf` file located in the Reflection ZFE session server:

`<install_directory>\sessionserver\conf\container.conf`

1. Open `container.conf`.
2. Set the value of the following property from False to **True**: `wrapper.java.additional.x =-Dcom.attachmate.integration.container.FIPS.enabled=true`.
3. Restart the server.

Related Topics

- Technical Note 2400, Attachmate Products with FIPS 140-2 Validated Crypto Modules
- Technical Note 2783, Security Updates and Reflection ZFE

Logging

Reflection ZFE uses Log4J 2.9.1 to implement logging. Log4J has its own configuration file and documentation. The configuration file, located in `ReflectionZFE/sessionserver/conf/log4j2.xml`, has a number of logging levels configured for output, and contains comments about the type of information that you can gather by changing logging levels.

For more information, see the Log4J documentation (https://logging.apache.org/log4j/2.x/).

The default logging (log4j) configurations are:

- Log file output is saved to `logs/server.log`
- In addition to logging to the `server.log` file, all console output is captured by the Reflection ZFE session server and stored in a file on disk.
- The configuration for how the console output is stored on file in `ReflectionZFE/sessionserver/conf/container.conf`.

  The file storage configuration properties include, but are not limited to the following (there are comments in `container.conf` that provide more information):

  - `wrapper.logfile` - the location of the captured log file (default is `.../logs/server.log`)
  - `wrapper.logfile.rollmode` - the mechanism in which the existing log file is stored as a backup and a new file is created (default is rolling over when the log file reaches a certain size and storing the rolled over log file with a roll number modifier)
  - `wrapper.logfile.maxsize` - the maximum size the log file can reach before it is rolled over (default is 10MB)
  - `wrapper.logfile.maxfiles` - the maximum number of rolled log files to keep on disk (default is 10)

- There are various types of logging levels you can use to produce different types of information. Log4j supports the following levels (these definitions are taken from the Log4j documentation where you can find more detailed information:
  - Trace - this level designates finer-grained informational events than Debug
  - Debug - this level designates fine-grained informational events that are most useful to debug an application.
  - Info - This level designates informational messages that highlight the progress of the application at coarse-grained level.
  - Warn - This level designates potentially harmful situations.
  - Error - This level designates error events that might still allow the application to continue running.
  - Fatal - This level designates very severe error events that will presumably lead the application to terminate.

4 Using Reflection ZFE

There are multiple session and display options available so you can personalize your session and make sure you are working efficiently.

- Display Settings
- Map Keys
- Configure User Macros
- Transfer Files
- Specify Copy and Paste Options
- Working with Sessions
- Printing
- Customize Sessions
- Set User Preferences

Display Settings

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

- Color mapping
- Configure hotspots
- Configure screen dimensions for VT, UTS and T27 hosts
- Set cursor options
- Set font options
- Set VT scrollback buffer options
- Set keyboard options
- Terminal Settings
- Set other display options

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

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.
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.

Table 4-1 *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

Table 4-2 *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

Table 4-3 *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 scrollback 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 keyboard options

- **Type ahead**

  When this option is selected, Reflection ZFE 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 keyboard options

- **Type ahead**

  When this option is selected, Reflection ZFE 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 keyboard 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 keyboard options

- **Enable lower case** (T27)

  Enables lower case, as well as upper case letters to be displayed on the screen. Default. If this is 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 Reflection ZFE 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, Reflection ZFE 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 Reflection ZFE 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

Some display options are host-specific as noted below. When the host type is not indicated, the options apply to all supported host types.

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)	To enable a status line at the bottom of the display. Choose: ◆ **None** to disable the status line. (Default) ◆ **Indicator** to display the page, cursor position, and printer status. ◆ **Host Writable** to have the host application display information in the status line.
Preserve aspect ratio	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.
Display OIA (3270, 5250)	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.
Display status line (ALC)	Turns on a status line at the bottom of the display.
Ignore mouse click on window activation	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.
Auto wrap (VT)	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.

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 for that session.

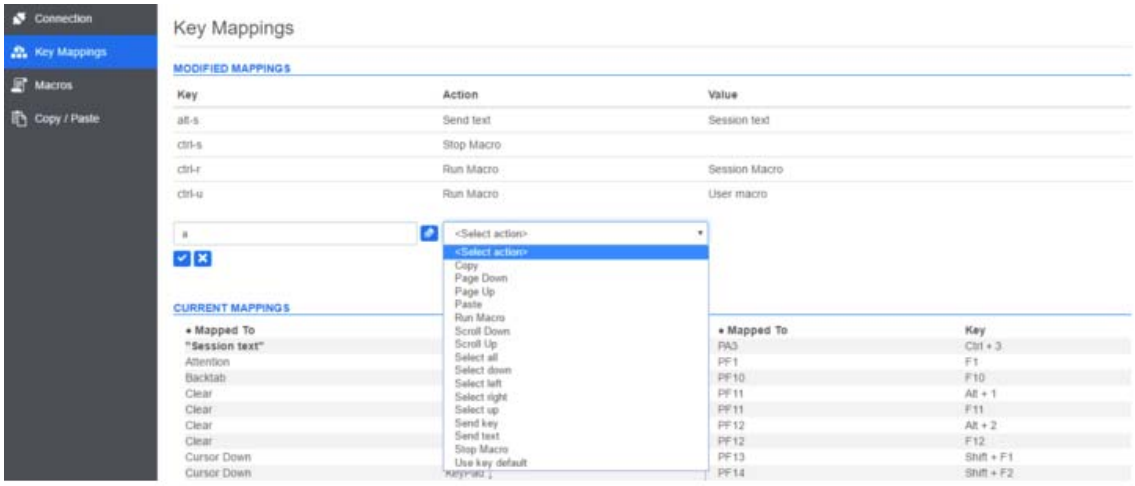
When you create a keyboard shortcut to perform actions, like **Run Macro** or **Send Text**, you can specify the necessary parameters in the Value field.

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.

TIP: Browsers use keyboard shortcuts to save both time and mouse clicks. When mapping keystrokes it is important to keep this in mind. For example, Ctrl+F1 opens Internet Explorer help as well as the UTS control page. Handy Keyboard Shortcuts gives a brief overview of the keyboard shortcuts used by different browsers.

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 Copying and Pasting for a list of copy/paste key actions.

1. From the toolbar, click **Settings**.
2. From the left navigation pane, open the **Key Mappings** panel.
3. Under **Modified Mappings**, click **Add** to enter a new key map entry.
4. Press the key or key combination you want to trigger the assigned action in the key field.
5. 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.



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*. Reflection ZFE will convert this to the string *\u001C* when that key is pressed and send the 6 characters of the resulting string to the host.

TIP: If you are using escape sequences on a VT host, the VT escape sequences begin with either the Unicode value of the escape character, \u001B or the CSI character \u009B. For example, to map the F1 key to `Send<ESC>[M` you'd enter `\u001B[[M`. The two brackets are necessary.

---

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

To remove existing entries from the **Modified Mappings** table, hover over the line containing the entry you want to delete and click the blue X that displays on the right side.

7 Click **Save** to close the Key Mapping panel and resume configuring your host connection.

After you complete mapping keys for the session, the updated Current Mappings table displays the custom key mappings in boldface type. Click the column header to sort the list by Key or Mapped to action.

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

Table 4-4 *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
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.

Key	Maps to	Description
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	PF11	Sends the PF11 key to the host
F11	PF11	Sends the PF11 key to the host
Al + 2	PF12	Sends the PF12 key to the host
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
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

Table 4-5 *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 the 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
KP + Subtract	Field exit minus	Moves the cursor out of a signed-numeric or numeric-only field
Ctrl + Subtract	Field exit minus	Moves the cursor out of a signed-numeric or numeric-only field
KP + Add	Field exit plus	Moves the cursor out of a signed-numeric or numeric-only field
Ctrl + Add	Field exit plus	Moves the cursor out of a signed-numeric or numeric-only field
Ctrl + 6	Field mark	Inserts the field mark character at the cursor location
Ctrl + H	Help	Sends the Help key to the host
Ctrl + X	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
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

Key	Maps to	Description
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
Shift + F9	PF21	Sends the PF21 key to the host
Shift + F10	PF22	Sends the 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

Table 4-6 *VT Keyboard Mapping*

Key	Maps to	Description
Ctrl + Cancel	Break	Sends the Break key to the host
Ctrl + Enter	Enter	Send the Enter key to the host
Alt + F1	F1	Sends the F1 key to the host
Ctrl + F1	F11	Sends the F11 key to the host
Ctrl + F2	F12	Sends the F12 key to the host
Ctrl + F3	F13	Sends the F13 key to the host
Ctrl + F4	F14	Sends the F14 key to the host
Ctrl + F5	F15	Sends the F15 key to the host
Ctrl + F6	F16	Sends the F16 key to the host
Ctrl + F7	F17	Sends the F17 key to the host
Ctrl + F8	F18	Sends the F18 key to the host

Key	Maps to	Description
Ctrl + F9	F19	Sends the F19 key to the host
Ctrl + F10	F20	Sends the F20 key to the host
Home	Find	Sends the Find key to the host
F1	Hold	Sends the Hold Screen to the host
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 scrollback buffer moves up a row
Ctrl + Alt + ArrowDown	Row down	In the scrollback buffer 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
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	UDK6	Sends the User Defined Key 6 to the host
Shift + F7	UDK7	Sends the User Defined Key 7 to the host

Key	Maps to	Description
Shift + F8	UDK8	Sends the User Defined Key 8 to the host
Shift + F9	UDK9	Sends the User Defined Key 9 to the host
Shift + F10	UDK10	Sends the User Defined Key 10 to the host
Shift + Ctrl + F1	UDK11	Sends the User Defined Key 11 to the host
Shift + Ctrl + F2	UDK12	Sends the User Defined Key 12 to the host
Shift + Ctrl + F3	UDK13	Sends the User Defined Key 13 to the host
Shift + Ctrl + F4	UDK14	Sends the User Defined Key 14 to the host
Shift + Ctrl + F5	UDK15	Sends the User Defined Key 15 to the host
Shift + Ctrl + F6	UDK16	Sends the User Defined Key 16 to the host
Shift + Ctrl + F7	UDK17	Sends the User Defined Key 17 to the host
Shift + Ctrl + F8	UDK18	Sends the User Defined Key 18 to the host
Shift + Ctrl + F9	UDK19	Sends the User Defined Key 19 to the host
Shift + Ctrl + F10	UDK20	Sends the User Defined Key 20 to the host

Table 4-7 *UTS Keyboard Mapping*

Key	Maps to	Description
F4	Clear Change Bit	Sends the CLEARCHANGEBIT 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.
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.

Key	Maps to	Description
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.
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.
ICtrl+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.

Key	Maps to	Description
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.
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.

Table 4-8 *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.
Left arrow	Cursor Left	Moves the cursor one column to the left.

Key	Maps to	Description
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 PF9key to the host.
PageDown	Page Down	Displays the next page.
PageUp	Page Up	Displays the previous page.
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.
Shift+up arrow	Select Up	Selects text up.

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

Table 4-9 *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

Key	Maps to	Description
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.
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	Cancels any changes made to the current PNR
Ctrl+Instert	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
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

Key	Maps to	Description
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	Statstics	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

Configure User Macros

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.

Related Topics

Creating Macros
Using the Macro API
Sample Macros

Transfer Files

Reflection ZFE supports two different file transfer protocols; IND$FILE for 3270 host transfers and 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 use the File Transfer Protocol (FTP) to transfer files between your local computer (or any networked drive) and the FTP server.

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 and 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
- 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. Reflection ZFE 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	• 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.
CR/LF processing	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.
Startup command	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 many need to specify your site's CICS transaction (for example, CFTR).
Startup parameters	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 Reflection ZFE. Reflection ZFE does not validate the parameters.

Option	Description
Max field size	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 TICP/IP product gets this value from the DATABUFFERPOOLSIZE parameter, which defaults to 8K buffers. See your system administrator if you don't know what to enter here.
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 Reflection ZFE 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 Reflection ZFE 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, Reflection ZFE 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	Use this option to specify the record format for files sent to the host. ◆ Default - The host determines the record format. This is the default option. ◆ Fixed - Forces the host to create fixed-length records. ◆ Undefined - Forces the host to create files without a specific record format (this value is only relevant for TSO systems). ◆ Variable -Forces the host to create variable-length records and preserves the format of a binary file.	TSO, CMS
Allocation units	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).	TSO

Option	Description	Applies to this host type
Logical record length	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.	TSO, CMS
If host file exists	Specifies how the transfer should operate if a file with the same name already exists. ◆ **Append** - Append the contents of the local file to the existing host file. ◆ **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.	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

- Downloading files
- Uploading files
- Troubleshooting your file transfers

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, containing 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. Reflection ZFE updates the remote file list using the dataset level you specify.

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

1. From the list, select the file or directory to initiate the transfer. You can choose to save or open the files in the format you selected in step 3.
2. If necessary, you can cancel the transfer from the transfer progress panel.

Uploading files

NOTE: IBM mainframe computer systems impose certain naming conventions for files. For detailed information on naming requirements, see the IBM documentation.

There are two methods for uploading files:

1. From the File Transfer dialog box, click **Upload**.
2. 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.

Or:

1. Drag the file you want to upload from its location to the File Transfer dialog box.
2. 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.

For host-specific errors, see IBM File Transfer Error Messages.

FTP

With Reflection ZFE 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.

TIP: If you plan on using a batch transfer, **Enable FTP** must first be selected and configured.

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. Reflection ZFE 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.

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

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

This option	Does this...
Append	Append the file being sent to the existing file
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

- **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.

- **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.

- **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.

- **Host encoding**

  Specifies the character set used by the host to display the names of files that are transferred. By default Reflection ZFE 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.)

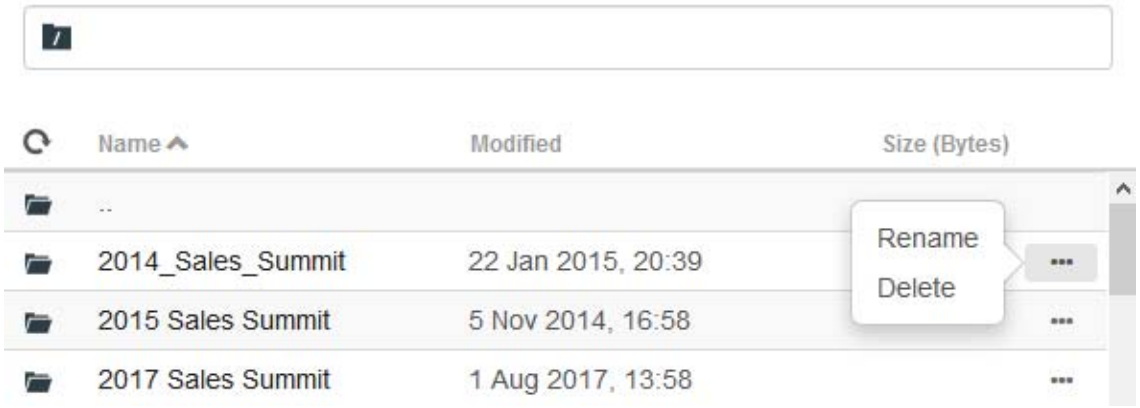
Transferring files

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. 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.
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

1 From the list, select the file to initiate the transfer.

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

Uploading files

There are two methods for uploading files:

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

2 Choose the file you want to upload from the Browse window.

Or:

1 Drag the file you want to upload from its location to the File Transfer dialog box.

2 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 FTP **BATCH** 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.
   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 name for the file to use after transfer.
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

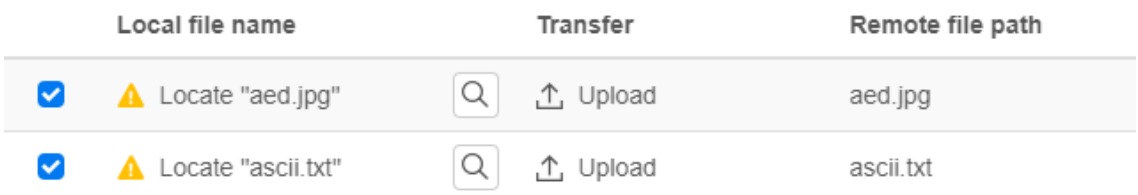
6. Click **Save**.

Transferring files

TIP: 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:



2. Files in the batch list are selected by default. You can clear the check box to eliminate the file from the transfer operation.
3. Click **Start** to initiate the transfer.

Specify Copy and Paste Options

You can specify different options to use for copy and paste operations.

Copy options

Select text by dragging over it with the mouse. By default, different host types use different selection modes when copying text; IBM 3270, 5250 and UTS hosts use a block selection mode, while VT hosts use a linear selection mode. To toggle between block and linear selection modes, 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

Click Paste to paste the contents of the clipboard at the cursor location.

- **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.
- **Mask 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.

Key combinations

There are certain key combinations that map to different copy/paste actions.

Key Combination	Host type	Action
Ctrl + A	UTS, 3270, 5250	Select all
Shift + Arrow key	UTS, 3270, 5250, VT	Change the extent of the current selection
Ctrl + C	UTS, 3270, 5250	Copy
Ctrl + V	UTS, 3270, 5250	Paste
Ctrl + Shift + A	VT	Select all
Ctrl + Shift + C	VT	Copy
Ctrl + Shift + V	VT	Paste

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://<sessionserver>:7443/zfe).

- Using Quick Keys
- Copying and Pasting
- Creating Macros
- Logging Out

To open a session

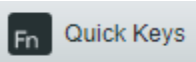
1. Select the session and click to open.
2. Interact with your host application using the open session.
3. You can create multiple instances of a configured session.

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 

.

Copying and Pasting

NOTE: Each browser handles copy and paste functions differently and in some cases will not support the use of copy and paste buttons. 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 and **CTRL+V** to paste.

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 use the keyboard command, **CTRL+C**.

To paste into the terminal screen

1. Position the cursor where you want to paste content.
2. Use the keyboard command, **CTRL+V**, or click **Paste** from the toolbar. If your browser does not support pasting from the toolbar, you will be prompted to use the keyboard command.

Related Topics

Specify Copy and Paste Options

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.

Reflection ZFE 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**

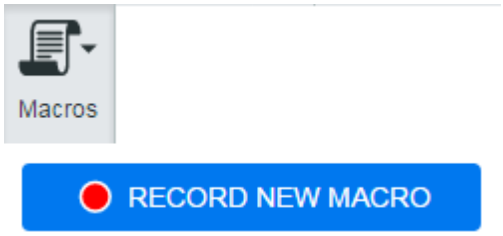
  End-user macros are created by individuals for sessions they are authorized to access. The administrator grants permission to create macros by setting a User Preference Rule. Users can access the session under their own credentials or in a **Guest** role. Macros that Guest users create are available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

- To record a macro
- To edit a macro
- To run a macro
- To stop a macro
- To delete a macro
- To view macros
- To debug a macro
- Using the Macro API

To record a macro

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.

To edit a macro

You can edit macros that you have recorded. These macros are listed under **My Macros**.

To edit an existing macro:

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 Edit to open the Macro Editor.

   The Macro Editor opens 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.

To run a macro

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.

To stop a macro

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.

To delete a macro

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**.

To view macros

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.

To debug a macro

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.

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

To debug a macro:

1. Open the macro for editing. See To edit a macro for instructions.
2. Open your browser's development tools.

Table 4-10 *Browser debugging support*

Browser	Open debugger
Mozilla Firefox 40.0.3	◆ 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 45.0	◆ From the toolbar, open the Menu, and choose More tools. ◆ Choose Developer Tools to open the Debugger.
Microsoft Internet Explorer 11	◆ From the toolbar, open Settings, and choose F12 Developer Tools. ◆ Open the Debugger tab.

These instructions are for supported browsers and are dependent on the versions used.

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 4-1 *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 4-2 *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 Reflection ZFE 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 4-3 *ui.message()*

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

Notes to keep in mind when debugging macros

- 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.

- Internet Explorer

  Debugging in Internet Explorer involves transformed code and may be more difficult than on other browsers.

Using the Macro API

In Reflection ZFE macros are recorded and written using JavaScript. JavaScript is a popular and prevalent programming language. There are a wide variety of learning resources and tools available to you.

The Reflection ZFE 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. Reflection ZFE 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 Reflection ZFE Macro API return promise objects.

- **Yield**

  Reflection ZFE 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 with `'throw new Error('Helpful error message');`

Related Topics

- 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

  Session is 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

  The PresentationSpace object 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

  The UI object is made 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.

Other available objects

- Attribute
- AttributeSet
- AutoSignon
- Color
- ControlKey
- DataCell
- Dimension
- Field
- FieldList
- FileTransferFactory
- FileTransfer
- File Transfer Options
- HostFile
- Host File Type
- OIA
- OIAStatus
- Position
- SessionType
- StatusSet

Attribute

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

Table 4-11 *Attributes*

Attribute	Description
PROTECTED	Indicates a protected data cell.
MODIFIED	Indicates a modified data cell.
NUMERIC_ONLY	Indicates the beginning of a numeric only data cell.
ALPHA_NUMERIC	Indicates an alpha numeric data cell.
HIGH_INTENSITY	Indicates whether the data cell contains high intensity text.
HIDDEN	Indicates whether the data cell contains hidden text
PEN_DETECTABLE	Indicates whether the data cell is pen detectable
ALPHA_ONLY	Indicates an alpha only data cell.
NUMERIC_SHIFT	Indicates the beginning of a numeric shift. field
NUMERIC_SPECIAL	Indicates the data cell marks the beginning of a numeric special field
KATAKANA_SHIFT	Indicates a section of Katakana text.
MAGNETIC_STRIPE	Indicates the data cell marks the beginning of a magnetic strip field.
SIGNED_NUMERIC_ONLY	Indicates the data cell is a signed numeric field.
TRANSMIT_ONLY	Indicates the data cell is a transmit only field
FIELD_END_MARKER	Indicates the data cell marks the end of a modified field.
FIELD_START_MARKER	Indicates the data cell marks the start of a modified field.
SPECIAL_EMPHASIS_PROTECTED	Indicates a special emphasis protected field.
TAB_STOP	Indicates that the data cell contains a tab stop.
REVERSE	Indicates the data cell displays in reverse video mode.
BLINKING	Indicates the data cell contains blinking text
RIGHT_JUSTIFIED	Indicates the data cell marks the beginning of a right justified field.
LEFT_JUSTIFIED	Indicates the data cell marks the beginning of a left justified field.
LOW_INTENSITY	Indicates the data cell contains low intensity text
UNDERLINE	Indicates the data cell contains underlined text.
DOUBLE_BYTE	Indicates the data cell contains double byte text.
COLUMN_SEPARATOR	Indicates the data cell contains a column separator.
BOLD	Indicates the data cell contains bold text.
DOUBLE_WIDTH	Indicates the data cell marks a double width field.
DOUBLE_HEIGHT_TOP	Indicates a double height top data cell.

Attribute	Description
DOUBLE_HEIGHT_BOTTOM	Indicates a double height bottom data cell.
CONTROL_PAGE_DATA	Indicates the data cell contains control page data.
RIGHT_COLUMN_SEPARATOR	Indicates the data cell contains a right column separator.
LEFT_COLUMN_SEPARATOR	Indicates a data cell containing a left column separator.
UPPERSCORE	Indicates the data cell contains an upperscore.
STRIKE_THROUGH	Indicates 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.

Table 4-12 *AttributeSet*

METHODS

`contains(attribute)`	Determines if the set contains the specifiedAttribute. **Parameters** `{Number}` attribute to check **Returns** `{Boolean}` True if the attribute is in the set.
`isEmpty()`	Determines if the attribute set is empty. **Returns** `{Boolean}`True if the set is empty.
`size()`	Indicates the number of attributes in a set. **Returns** `{Number}` The attribute count.
`toArray()`	Converts the internal attribute set to an array. **Returns** `{Number[]}`Array of values of attributes in the set.
`toString()`	Converts the internal attribute set to a string. **Returns** `{String}` Space-delimited names of attributes in the set.

METHODS

`forEach(callback, thisArg)`	Function to iterate over each element in the attribute set. **Parameters** `{forEachCallback}` Callback to perform the specific operation. Called with the name of each attribute in the set. `{Object}` this Arg optional pointer to a context object.
`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.

Color

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

Table 4-13 *Color constants*

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
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

Table 4-14 *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
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

Key word	Description
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

Table 4-15 *IBM 5250*

Key word	Description
ALTVIEW	Alternate view
ATTN	Attention
AU1 - AU16	AU1 - AU16
BACKSPACE	Back space
BACKTAB	Back tab
BEGIN_FIELD	Begin field
CLEAR	Clear
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

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

VT

Table 4-16 *VT*

Keywords	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

Keywords	Description
EK_REMOVE	Edit keypad remove
EK_SELECT	Edit keypad select
ENTER	Enter
END_FILE	End of field
F1 - F24	F1 - F24
HOLD	Hold
HOME	Home
INSERT	Insert
KEYPAD_COMMA	Keypad comma
KEYPAD_DOT	Keypad decimal
KEYPAD_MINUS	Keypad minus
KEYPAD_ENTER	Keypad enter
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

Table 4-17 *UTS*

Key word	Description
BACKSPACE	Moves the cursor to the previous tab position on the screen.
BACKTAB	Back tab <Shift> <Tab>
CHAR_ERASE	Erases character at the cursor and advances the cursor.
CLEAR_DISPLAY	Clear display
CLEAR_EOD	Clear to end of display
CLEAR_EOF	Clear to end of field

Key word	Description
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
DOWN	Moves the cursor down one line. Wraps at bottom.
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.
DUP_LINE	Creates a copy of the current line and overwrites the next line with the duplicate.
EURO	Inserts the Euro character
END_FIELD	Moves the cursor to the end of the current field.
END_PAGE	Moves the cursor to the end of the current page.
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.
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.

Key word	Description
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.

Table 4-18 *DataCell*

METHODS	
`getPosition()`	Returns the position of this data cell on the screen. **Returns** `{Position}` the position of the data cell on the screen
`getChar()`	Obtains the character associated with the cell. **Returns** `{String}` The character associated with the cell.
`getAttributes()`	Returns the set of attributes specified for this data cell instance. See AttributeSet. **Returns** `{AttributeSet}`Of attributes for this data cell instance.
`getForegroundColor()`	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.
`getBackgroundColor()`	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.
`toString`	Converts the internal data cell to a string. **Returns** `{String}` The string representation of a data cell.

METHODS

`isFieldDelimiter()`	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.

Table 4-19 *Dimension*

Method	
`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.

Table 4-20 *Field*

Method	
`getAttributes()`	Returns the set of attributes specified for this field instance. See AttributeSet. **Returns** `{AttributeSet}` The set of attributes for this field
`getForegroundColor()`	Returns the foreground color of the field. **Returns** `{Number}` the foreground color for this field. These values are defined in the Color object.
`getBackgroundColor()`	Returns the background color of the field. **Returns** `{Number}` the background color for this field. These values are defined in the Color object.

Method	
`getStart()`	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** `{Position}` Starting position of the field. **Throws** `{RangeError}` For zero length fields.
`getEnd()`	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** `{Position}` Ending position of the field. **Throws** `{RangeError}` For zero length fields.
`getLength()`	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. **Returns** `{Number}` Length of the field.
`getDataCells()`	Obtains the data cells that comprise this field. SeeDataCell . **Returns** `{DataCell[]}`Data cells that comprise this field.
`getText()`	Obtains the text from the field. **Returns** `{String}` field text.
`setText()`	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. **Parameters** `{String}`Text to set on the field. **Throws** `{Error}` If the field is protected.

Method	
`clearField()`	Clears the current field in an emulation-specific manner. **Throws** `{Error}` If the field is protected or clear is not supported.
`getPresentationSpace()`	Obtains the PresentationSpace which created this field. **Returns** `{PresentationSpace}` Parent of this field instance.
`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.

Table 4-21 *FieldList*

Method	
`getPresentationSpace()`	Obtains the PresentationSpace which created this field list. **Returns** `{PresentationSpace}`Parent of this field list instance.
`findField(position, text, direction)`	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. **Parameters** `{Position}` Position from which to start the search. See Position object. `{String}` The text to search for (optional). If not provided, returns the next field to the right of or below the specified position. `{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. **Returns** `{Field}` containing the string or null if a field meeting the given criteria is not found. **Throws** `{RangeError}` If the position is out of range.

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

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.

Table 4-22 *fileTransferFactory*

Method	
`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.

FileTransfer

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

The Reflection ZFE 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.

Table 4-23 *FileTransfer*

Method	
`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 can not be read.
`sendFile(localFile, remoteName)`	Sends specified file to the host. **Parameters** `{File}` Javascript file object pointing to local file to send. `{String}` 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 Reflection ZFE 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.

Method	
`cancel()`	Cancels the current transfer in progress. **Returns** `{Promise}` fulfilled when the call completes. Rejected if an error occurred canceling the transfer.

HostFile

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

Table 4-24 *HostFile*

Method	
`getName()`	Gets the file name **Returns** `{String}` the file name.
`getParent()`	Gets the parent of this host file **Returns** `{String}` 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.
`getSize()`	The byte size of the file **Returns** `{Number}` the size of the file in bytes.
`getType()`	The type of file represented **Returns**

Host File Type

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

Table 4-25 *HostFileType*

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.

File Transfer Options

File transfer option object specification.

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

Table 4-26 FileTransferOptions

Method	
`transferMethod`	`{String}`Allowed values: ◆ 'ascii' ◆ 'binary'

OIA

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

Table 4-27 OIA

Method	
`getStatus` ()	Returns the set of enabled status flags. See StatusSet. **Parameters** **Returns** `{StatusSet}` Containing the status information.
`getCommErrorCode()`	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

Table 4-28 OIAStatus

OIAStatus	Description
CONTROLLER_READY	Controller ready
A_LINE	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

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

AutoSignon

Some mainframe hosts have a Digital Certificate Access Server (DCAS). You can request a temporary, one-time pass ticket from DCAS for logging into a host application. Using this object, you can write and configure a macro to run when the session starts and to automatically log you in using the credentials of the currently logged in Reflection ZFE user.

Table 4-29 *AutoSignon*

Method	
`getPassTicket()`	Obtains a pass ticket to be used for signing onto a mainframe application. Multiple pass tickets may be requested using different application IDs. **Parameters** `{String}` application ID tells the host which application the sign on is for **Returns** `{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.

Method	
`sendUserName()`	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. **Parameters** `{String}` passTicketKey opbtained from getPassTicket **Returns** `{Promise}` fulfilled if the user name is successfully sent. Rejected if the operation fails.
`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.

Position

Represents a row and column on the screen.

Table 4-30 *Position*

Method	
`Position(row,col)`	Creates a new Position instance. **Parameters** `{Number} row` screen row coordinate `{Number} 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.

Table 4-31 *PresentationSpace*

METHODS

`getCursorPosition()`	Returns a Position instance representing the current cursor position. An unconnected session has a cursor position of 0,0. **Returns** `{Position} current cursor location`
`setCursorPosition(position)`	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. **Parameters** `{Position}` Position new cursor position. **Returns** None **Throws** `{RangeError}` If the position is not valid on the current screen.
`isCursorVisible()`	Tests that the cursor is currently visible in the presentation space. The cursor is considered not visible if the session is not connected. **Returns** `{Boolean}` True if the cursor is visible. False if the cursor is not visible.
`sendKeys(keys)`	Transmits a text string or ControlKey to the host at the current cursor position in the presentation space. If the cursor is not in the desired position, then use `setCursorPosition` function first. The text string can contain any number of characters and ControlKey objects. For example: `"myname" + ControlKey.TAB + "mypass" + ControlKey.ENTER` will transmit a user ID, tab to the next field, transmit a password, and then transmit the Enter key. If you need to transmit a square bracket, double the brackets ([[or]]). **Parameters** `{String} keys text and/or control keys to transmit`

METHODS

`getText(position,length)`	Returns a string representing a linear area of the presentation space. No new line characters are inserted if row boundaries are encountered.
	
	Parameters
	
	`{Position} start position from which to retrieve text`
	
	`{Number} 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.`
	
	Returns
	
	`{String) representing a linear area of the presentation space which may be empty if the session is not connected.`
	
	Throws
	
	`{RangeError}` If the position or length are not valid on the current screen.

`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.

Table 4-32 *Session object functions*

METHODS

`connect()`	Connects to the configured host. If needed, use `wait.forConnect()` to block macro execution until the session is connected. **Returns** `None`
`disconnect()`	Disconnects from the configured host. If needed, use `wait.forDisconnect()` to block macro execution until the session is connected. **Returns** `None`
`isConnected()`	Determines whether the connection to the host is connected. **Returns** `{Boolean} true if host connection is established; false if not`
`getPresentationSpace()`	Provides access to the PresentationSpace instance for this session. **Returns** `{PresentationSpace}` instance associated with this session.
`getDeviceName()`	Returns the connected available device name, the configured device name, or null if no device name is configured. The connected device name is the name agreed upon during the connection negotiation process between the host and the terminal. It may be what is specified, or it could possibly be different, if for example a device name pool was specified. **Returns** `{String}`The connected device name, the configured device name, or null.
`getType()`	Returns the type of host session. See SessionType. **Returns** `{String}` The type of host session.
`setDeviceName()`	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.
`getOIA()`	Provides access to the OIA instance for this session. **Returns** `{OIA}` Associated with this session

SessionType

Constants used to identity the type of host to which the connection is being made. See Session object.

Table 4-33 *SessionType*

Host Type	Description
IBM_3270	Indicates an IBM 3270 terminal session.
IBM_5250	Indicates an IBM 5250 terminal session.
VT	Indicates a VT session.

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.

Table 4-34 *StatusSet*

Method	
`contains(statusFlag)`	Determines if the set contains the specified status flag from OIAStatus constants. **Parameters** `{Number} statusFlag` status to check **Returns** `{Boolean}` True if the status flag is present in the set.
`isEmpty()`	Determines if the status set is empty. **Returns** `{Boolean}` True if the set is empty.
`size()`	Indicates the number of status flags in the set. **Returns** `{Number}` The status count
`toArray()`	Converts the internal status set to an array. **Returns** `{Object []}` Array of status flags in the set.
`toString()`	Converts the internal status set to a string. **Returns** `{String}` Space delimited names of status flags in the set.

Method	
`forEach(callback, thisArg)`	Function to iterate over each element in the status set. **Parameters** `{forEachCallback}` Callback to perform the specific operation. Called with the name of each status in the set. `{Object}` thisArg optional pointer to a context object.
`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.

Table 4-35 *User Interaction*

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

Table 4-36 *Waiting for the host*

METHODS

`setDefaultTimeout(timeout)`	Sets the default timeout value for all functions. **Parameters** `{Number} default timeout to use for all wait functions in milliseconds.` **Returns** None **Throws** `{RangeError}` If the specified timeout is less than zero.
`forConnect([timeout])`	Waits for a connect request to complete. **Parameters** `{Number}` in milliseconds. **Returns** `{Promise}` Fulfilled if the session is already connected or when connection occurs. Rejected if the wait times out.
`forDisconnect([timeout])`	Waits for a disconnect request to complete. **Parameters** `{Number}` timeout in milliseconds. **Returns** `{Promise}`Fulfilled if the session is already disconnected or when it finally disconnects. Rejected if the wait times out.

METHODS

`forFixedTime([timeout])`	Waits unconditionally for fixed time. Time is in milliseconds (ms) **Parameters** `{Number}` timeout in milliseconds. **Returns** `{Promise}`Fulfilled after time elapses

`forCursor(position, [timeout])`	Waits for the cursor to arrive at the specified position.
	Parameters
	`{Position} The position specifying the row and column,`
	`{Number} timeout in milliseconds`
	Returns
	`{Promise}`Fulfilled if the cursor is already located or when it is finally located. Rejected if the wait times out.

`forText(string, 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(string, 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.

Sample Macros

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

- Basic Host Interaction
- User Interaction
- Paging Through Data
- Invoking a Web Service
- Working with DataCells and Attributes
- Using Fields and Field Lists
- Automatic Sign-On Macro for Mainframes
- Using File Transfer (IND$File)

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 ZFE 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 eaxmple 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++;
        }
      }
    }
 // 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 ZFE 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, so 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, AJAX/REST calls will fail if they attempt to access a different server than the one where the ZFE application originated. This is a browser security feature.

The Reflection ZFE server provides a way explicitly to proxy to trusted remote servers.

- Open `..\ReflectionZFE\sessionserver\webapps\zfe\WEB-INF\web.xml` in your ZFE deployment.
- Modify the file as shown:

```
<!--
   Example of how to proxy 3rd party services that do not support CORS.
   The following configuration would allow you to invoke remote VHI REST
   services via local URLs (e.g. an HTTP POST to
   "proxy1/model/CICSAcctsDemo/GetAccountDetail" would be routed to
   "http://remote-vhi-server:9680/vhi-rs/model/CICSAcctsDemo/GetAccountDetail").
   -->
   <servlet>
       <servlet-name>vhi-rs-proxy1</servlet-name>
       <servlet-class>org.eclipse.jetty.proxy.ProxyServlet$Transparent</servlet-class>
       <init-param>
           <param-name>proxyTo</param-name>
           <param-value>http://remote-vhi-server:9680/vhi-rs/</param-value>
       </init-param>
       <init-param>
           <param-name>prefix</param-name>
           <param-value>/proxy1</param-value>
       </init-param>
       <async-supported>true</async-supported>
   </servlet>
   <servlet-mapping>
       <servlet-name>vhi-rs-proxy1</servlet-name>
       <url-pattern>/proxy1/*</url-pattern>
   </servlet-mapping>
```

- Uncomment the servlet and servlet-mappings sections.
- Change `http://remote-vhi-server:9680/vhi-rs` to the actual URL of your target REST server. You can also rename the url-pattern.
- Keep in mind that even when a REST server supports CORS headers, some older browsers may not, so this example may still be relevant.

TIP: Your customized `web.xml` file may be replaced whenever you redeploy Reflection ZFE. Always back up your files.

Working with DataCells 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 how to 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.');
        ps.sendKeys('cesf logoff');
        ps.sendKeys(ControlKey.ENTER);
    } catch (error) {
        yield ui.message(error);
    }
    //End Generated Macro
});

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

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

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();
```

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.

Printing

There are various printing options available:

- Capture a screen
- Print a screen
- 3270 Host Printing

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.

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 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.

3270 Host Printing

This feature is available to 3270 host sessions. You can create one or more 3287 printer sessions and associate them with the current 3270 terminal session. Each printer session is bound to a Logical Unit (LU) on the host system and any subsequent print jobs sent to that LU will be directed to the Reflection ZFE web client.

The 3287 session builds a PDF file that contains the file to print and sends it to the Reflection ZFE web client. After receiving the file, the Reflection ZFE web client downloads the file following your browser's configured download options. Different browsers provide different options for handling downloaded files. When the PDF file is received, you can direct it to any printer that you have access to.

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

Related Topics

Connection settings
Page Setup settings
To print your 3270 printer session

To configure 3270 host printing

1 From a 3270 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. There are three tabs; Connection settings, Page Setup settings, and Advanced settings. Each tab has different settings available to customize your printer session.

4 Click **Save** to return to your session. The settings take effect when the session is reopened.

Related Topics

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.

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.
Host LU Name	Specify whether you want to use a Host LU Name, prompt for LU Name or, if you select TN3270E, a TN Association, to link the terminal session with the print session. Required. Select one: ◆ **Specify Host LU Name** - Specify the name of the host LU (logical unit) to use when the session connects to the host. The host LU is also known as the device name. ◆ **Use TN Association** - (TN3270E) If you choose to use a TN association, Reflection ZFE 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 session connects, the user is prompted to supply the LU Name for the printer session.

Related Topics

Page Setup settings
Advanced settings
To print your 3270 printer session

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	There are three modes you can choose; **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.
Scaling	Sets the horizontal and vertical scaling for printed output. Increase the percentage to increase the horizontal or vertical space used by the printout.

Related Topics

Connection settings
Advanced settings
To print your 3270 printer session

Advanced settings

There are three options available to you when deciding when to download the PDF file.

- **Automatically** - (default) The PDF 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 PDF file and downloaded.
- **After inactivity timeout** - Using this option you can print multiple print jobs, have them aggregated into a single PDF, 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 PDF. After 5 seconds and no remaining print jobs, the PDF 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**.

Related Topics

Connection settings
Page Setup settings
To print your 3270 printer session

To print your 3270 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 3287 session receives the print data from the host and builds a PDF file to print. A link to this file is sent to the Reflection ZFE 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 PDF 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.

Customize Sessions

There are two features you can use to customize sessions for your end users:

- **Plus** - Enable custom controls to provide a more efficient work flow and a more modern and friendly interface. See Use Plus to customize screens.

  Using this option, you 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 Use server side events.

Both of these options are configured on the Customization panel.

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

Related Topics

Use Plus to customize screens

Use Plus to customize screens

NOTE: The Plus feature requires customization files (.rdar) produced by Micro Focus Screen Designer version 9.5 or higher. The Screen Designer is available in Micro Focus Rumba Desktop 9.5. Reflection Desktop 16.1 includes a limited version of the Screen Designer and Micro Focus Plus. To get access to more controls and full use of Plus and the Screen Designer, you can purchase and install the Micro Focus Reflection Desktop Plus add-on.

1. On the **Customization** panel, click **Enable Plus**.
2. Select the customization file you want to use from the drop down list or upload a file from a different location. Customization files are identified by a `rdar` file extension.

   Customization (.rdar) files are archive files and the output of a Screen Designer project and are used to provide the custom control criteria. For more information on the Micro Focus Screen Designer there are instructional videos available here: Screen Designer videos.

   If you are updating the customization (.rdar) file associated with your Plus enabled session, you must first delete the folder containing the old .rdar file from the session server. After you delete the folder, you can open your Plus enabled session and the new rdar file will be downloaded to the session server.
3. Verify the number of milliseconds for the host settle delay time is accurate. This is the time that the server waits for a synchronous connection before deciding that the host has finished sending data.
4. When you return to your session, Plus is available. Click  on the toolbar to turn off the custom controls.

When you enable Plus for a session, all end users of that session see the Plus icon on the toolbar and any controls made available through the Screen Designer customization file.

Related Topics

Customize Sessions

Use server side events

Using server side events, you can supply procedural Java code that can extend and improve the presentation of host data.

1. Make the Reflection ZFE SDK available to your development environment. The SDK is available at `install-dir\Micro Focus\ReflectionZFE\sessionserver\sdk.`
2. Write the Java code necessary to accomplish the task and compile the code into a Java class within a JAR (Java Archive) file.
3. Copy the JAR file to `install-dir\Micro Focus\ReflectionZFE\sessionserver\webapps\zfe\WEB-INF\lib` and restart the session server.

   If you have more than one session server on which you want the event to run, you must copy the JAR file to this location on each server.

4 Add the session you want to associate with the event in the Administrative Console.

5 As you configure the session in the Reflection ZFE web client, open the **Customization** panel.

6 Under **Server Side Events** type the fully qualified class name to the event.

7 Launch the session and test the event.

To help you create server side events, Reflection ZFE has an SDK and samples that are available to provide you with a starting point.

NOTE: The samples have logging statements in the code. If you run the samples you will not see this logging output on the session server unless you modify `<install-dir>/sessionserver/conf/logging.properties` as follows:

```
# Logging
handlers = java.util.logging.FileHandler
.level = ALL
# File Logging
java.util.logging.FileHandler.pattern = ../logs/serversideevents.log
java.util.logging.FileHandler.formatter = java.util.logging.SimpleFormatter
java.util.logging.FileHandler.level = INFO
```

Restart the session server.

Access API documentation and event samples

Related Topics

Customize Sessions
Using the Reflection ZFE SDK
Developing with Reflection ZFE

Set 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.

Related Topics

Display Settings
Specify Copy and Paste Options
Configure User Macros

5 Developing with Reflection ZFE

Reflection ZFE has a collection of APIs and libraries that help you develop efficient client/server and Web applications that integrate host data into various development environments.

- Using the Reflection ZFE SDK you can use the provided Java API to enhance the presentation of host data using server side events.
- Using the Reflection ZFE Connector for Windows you can interact with host sessions; creating new sessions and authenticating and connecting to existing Reflection ZFE sessions, using the API and samples provided.

Related Topics

Customize Sessions
Reflection ZFE Connector for Windows API documentation
Reflection ZFE SDK

Using the Reflection ZFE SDK

Use server side events to supply procedural Java code that can extend and improve the presentation of host data. To help you create server side events, Reflection ZFE has an SDK and samples that are available to provide you with a starting point.

You can view the Reflection ZFE SDK.

Examples and documentation

NOTE: The samples have logging statements in the code. If you run the samples you will not see this logging output on the session server unless you modify `<install-dir>/sessionserver/conf/logging.properties` as follows:

```
# Logging
handlers = java.util.logging.FileHandler
.level = ALL
# File Logging
java.util.logging.FileHandler.pattern = ../logs/serversideevents.log
java.util.logging.FileHandler.formatter = java.util.logging.SimpleFormatter
java.util.logging.FileHandler.level = INFO
```

Restart the session server.

To access the SDK for direct viewing and to import to your IDE:

1. Navigate to `<install-dir>\Micro Focus\ReflectionZFE\sessionserver\sdk\java`.
2. In the SDK directory, access:
   - `\javadoc`. This directory contains JavaDoc files for direct viewing.
   - `\samples` - This directory contains Java sources for direct viewing.

- `\zfe-sdk.jar` - The JAR file contains the Java classes to import into your IDE.
- `\zfe-sdk-javadoc.jar` - The JAR file contains JavaDoc files to import into your IDE.

Using the Reflection ZFE Connector for Windows

Reflection ZFE Connector for Windows is a separate installation which you can find on the Micro Focus download site. Using the Reflection ZFE Connector for Windows, you can interact with ZFE host sessions in your .NET application or within Visual Basic for Applications. Here are a few things to keep in mind as you prepare to install:

- Two install platforms are available: a 32-bit version and a 64-bit version. Depending on which one you install, the default base install path will be `C:\Program Files (x86)\Micro Focus\ReflectionZFE\Connector for Windows` or `C:\Program Files\Micro Focus\ReflectionZFE\Connector for Windows`.
- The installation platform you choose also determines the solution platform in which you can develop. For example: If you have installed the 32-bit version of Microsoft Office® and want to use Visual Basic for Applications with the connector, then you must install the 32-bit version of the Reflection ZFE Connector for Windows.
- .NET 4.5.2 is required.

Examples and connector documentation

Reflection ZFE Connector for Windows API documentation

Documentation is available to reference from your IDE. There are also samples to help you take advantage of the connector. Both are located here:

1. Navigate to the install directory. In a default install, either `C:\Program Files (x86)\Micro Focus\ReflectionZFE\Connector for Windows` or `C:\Program Files\Micro Focus\ReflectionZFE\Connector for Windows` depending on your platform.
2. In the `Connector for Windows` directory you will find:
   - `MicroFocus.ZFE.Connector.dll` - a .NET Framework assembly to reference in your C# or .NET project.
   - `MicroFocus.ZFE.Connector.tlb` - a Type Library to use in your COM or Visual Basic for Applications project.
   - `\help` - this directory contains information which will aid in using the connector.
   - `\samples` - this directory contains the code samples that provide a starting point for developing your own applications.

Using the connector with Microsoft Visual Studio

If you are using Microsoft Visual Studio to develop applications, keep these things in mind:

- When using Microsoft Visual Studio with Reflection ZFE Connector for Windows, make sure your solution platform is set to either x86 or x64, depending on your installation. Because of the native components used within the Reflection ZFE Connector for Windows SDK, the **Any CPU** platform is not supported. Use the Configuration Manager for your Visual Studio Solution to create a platform for x86 or x64.
- When adding a reference to the Reflection ZFE Connector for Windows library, Visual Studio may set the **Copy Local** reference property to **True**. This should be set to **False** so that the library and its dependencies are executed from the SDK install directory.

6 Technical References

In this section you can find information on specific issues that you may encounter. In the Micro Focus Technical Support Handbook you will find information about how to get technical support for your product, access to our online resources, and how to contact and work with our worldwide technical support organization.

Using Default Java Cryptography

Reflection ZFE uses Bouncy Castle, which is a Java implementation of cryptographic algorithms, to ensure secure connections between components. Occasionally it may become advantageous to replace the Bouncy Castle implementation with standard Java cryptography.

This is a two step process; first replacing Bouncy Castle with the Java implementation and second, importing certificates to enable communication between MSS and Reflection ZFE.

Replacing Bouncy Castle with the Java cryptographic implementation:

1. Open `sessionserver\conf\container.conf` in a text editor.
2. Set `-Dcom.attachmate.integration.container.CRYPTO.enabled` to **false**. For example:

   ```
   wrapper.java.additional.10=-
   Dcom.attachmate.integration.container.CRYPTO.enabled=false
   ```

3. Update these trust store settings to use the default JKS format.

   ```
   wrapper.java.additional.6=-Djavax.net.ssl.trustStore=../etc/
     servletcontainer.jks
   wrapper.java.additional.8=-Djavax.net.ssl.trustStoreType=jks
   wrapper.java.additional.12=-
     Dmanagement.server.client.ssl.trustStoreFileName=../etc/servletcontainer.jks
   wrapper.java.additional.13=-Dmanagement.server.client.ssl.trustStoreType=jks
   ```

   Save the file.
4. Open `\sessionserver\services\servletengine\META-INF\service-ctx.xml`
5. Modify the settings as follows:

   Update `keystoreName` and `keystorePath` from `bfcks` to `jks`
   Update `keystoreType` and `trustStoreType` from `BCFKS` to `JKS`

Insert `<property name="keyStoreType" value ="JKS"/` to the `mutualAuthKeystoreGenerator` bean.

6 Restart the session server.

Configuring communication between MSS and Reflection ZFE by enabling their respective keystores:

1 To import the MSS certficate to the Reflection ZFE keystore, from the `sessionserver/etc` directory, run the following commands:

```
keytool -importcert -file <path-to-the-MSS-certificate> -alias mgmt-server -
  keystore servletcontainer.jks -storetype jks - storepass changeit
keytool -importcert -file <path-to-the-MSS-certificate> -alias mgmt-server -
  keystore system.jks -storetype jks -storepass changeit
```

2 To import the Reflection ZFE certificate to the MSS trusted subsystem keystore, from the `MSS/server/etc` directory, run the following command:

```
keytool -importcert -file <path-to-the-ZFE-certificate> -alias zfe-server -
  keystore system.bcfks -storetype bcfks - storepass changeit -providername
  BCFIPS -providerclass
  org.bouncycastle.jcajce.provider.BouncyCastleFipsProvider -providerpath ../
  lib/bc-fips-1.0.1.jar
```

3 Restart both MSS and the Reflection ZFE session server.

Copying Sessions between Management and Security Servers

You can copy and convert Reflection for the Web sessions and make them available to another Management and Security Server (MSS) and Reflection ZFE.

NOTE: In the following procedure the Management and Security Server you are copying sessions from is the **source**, and the Management and Security Server you are copying to is the **destination**.

To copy sessions from the source server to the destination server follow these steps:

1 Stop the destination MSS server, if necessary.

2 On both source and destination MSS servers, open *SessionDS.xml*, located:

- On Windows: `C:\ProgramData\Micro Focus\MSS\MSSData`
- On Linux: `/var/opt/microfocus/mss/mssdata`

3 In the source XML file, locate the OBJECT_ARRAY element.

4 Still in the source XML file, under OBJECT_ARRAY, locate and copy the Reflection for the Web child *Session* elements.

5 Open the destination XML file and paste them under the destination file's OBJECT_ARRAY element.

6 Still in the destination file, locate the OBJECT_ARRAY size attribute that corresponds to the number of sessions. Increase that value by the number of session elements you added. For example, if you pasted six *Session* elements in the destination file and the existing OBJECT_ARRAY size attribute value is 4; increase the value by six. The size attribute should now be ten. And you should now have 10 *Session* elements listed under the OBJECT_ARRAY element.

7 Session names must be unique. Check the destination file for duplicate session names.You can find session names in the *Session* child element, *SessionName*.

8 Copy the configuration files for every session added to *SessionDS.xml* from the source to the destination server. The names of the configuration files are located under the *Session* element in the child element, *configuration*. The files themselves are located:

- On Windows: `C:\ProgramData\Micro Focus\MSS\MSSData\deploy\dyncfgs`
- On Linux: `/var/opt/microfocus/mss/mssdata/deploy/dyncfgs`

9 If you stopped the destination MSS server, restart it. Open the Administrative Console. You should see all your copied Reflection for the Web sessions in the **Manage Sessions** list.

10 The next step is to save the Reflection for the Web session as a Reflection ZFE session. In Manage Sessions, right-click the session you want to export. Session types are identified by an icon in the Type column.

11 See Export a Reflection for the Web session for information on saving a Reflection for the Web session to a Reflection ZFE session in the Administrative Console.

Macro Replication across Servers

Macros that are created by the end-user are not replicated between Management and Security Servers (MSS). Although Management and Security Servers can be deployed in a clustered environment where data is replicated across all servers in the deployment; macros created by the end-user for a session cannot be replicated.

To ensure that every Reflection ZFE session server has access to all end-user macros, it is important that every session server in a multi-server deployment environment points to a single Management and Security Server. To do this, specify the same Management and Security Server during the installation of the separate Reflection ZFE session servers.

You can back up the Reflection ZFE user macros by copying the `<install-location>/Micro Focus/MSS/server/services/zfemgmt/conf/prefs` directory to another location.

Configuring User Names when Using Anonymous Access Control

Users need access to their macros, user configurations, and other personalized settings whether they are authenticated anonymously through Management and Security Server or not. Reflection ZFE uses user names to store user-specific information; but what happens when users are anonymously authenticated through Management and Security Server's access control interface?

In a default environment, Reflection ZFE uses the session id of the HTTP session as the value for the user name. While this user name is unique for each browser session, it changes over time and when MSS is configured in anonymous mode, in order to consistently retrieve user settings, all users of that Reflection ZFE session necessarily share the same settings.

However, Reflection ZFE supports a number of ways that, as an administrator, you can configure a unique identifier for each user so their customized settings can be stored and retrieved.

NOTE: These configuration modifications do not alter the security considerations of using Management and Security Server in anonymous mode.

Related Topics

Configuration options
Troubleshooting the configuration

Configuration options

There are four different configuration options you can choose from when configuring user name identifiers. You must restart the Reflection ZFE session server before any changes take effect.

- **To use an HTTP request cookie value as the user name**

  Add the following lines to `<session-server>/conf/container.properties`:

  ```
  zfe.principal.name.provider=com.microfocus.rzfe.webclient.security.rsg.CookieKeyAnonymousPrincipalNameProvider
  zfe.principal.name.identifier=<the-cookie-key-to-be-used>
  ```

- **To use an HTTP request header value as the user name**

  Add the following lines to: `<session-server>/conf/container.properties`:

  ```
  zfe.principal.name.provider=com.microfocus.rzfe.webclient.security.rsg.HeaderKeyAnonymousPrincipalNameProvider
  zfe.principal.name.identifier=<the-header-key-to-be-used>
  ```

- **To use an HTTP request URL parameter as the user name**

  Add the following lines to: `<session-server>/conf/container.properties`

  ```
  zfe.principal.name.provider=com.microfocus.rzfe.webclient.security.rsg.UrlParameterAnonymousPrincipalNameProvider
  zfe.principal.name.identifier=<the-url-parameter-key-to-be-used>
  ```

- **To use the client IP address as the user name**

  Add the following line to: `<session-server>/conf/container.properties`

  ```
  zfe.principal.name.provider=com.microfocus.rzfe.webclient.security.rsg.RemoteAddrAnonymousPrincipalNameProvider
  ```

Troubleshooting the configuration

If any of your users experience problems when connecting to a Reflection ZFE web application after you have made the configuration changes, check the following:

- Users experience a **503 Service Unavailable** message when connecting to a Reflection ZFE web application. First check the log file (`<session-server>/logs/server.log`), then:

  -- If the log file contains this message: **"Unable to create AnonymousPrincipalNameProvider instance for class..."**, then the `zfe.principal.name.provider` property is probably mis-typed. Check the spelling and letter case to remedy this issue.

  --If the log file contains this message:**"zfe.principal.name.identifier is not defined"**, then the property is missing. Ensure the property is defined to remedy this issue.

- Users are unable to properly authenticate.

  Users should receive an error message indicating the initial HTTP request to the Reflection ZFE web application did not contain the required information.

Accessing Reflection ZFE using the IIS Reverse Proxy

This note describes how to use the IIS Reverse Proxy with Reflection ZFE. In order to comply with Common Criteria security requirements, it is necessary to place the Reflection ZFE server behind a proxy in this manner.

Prerequisites

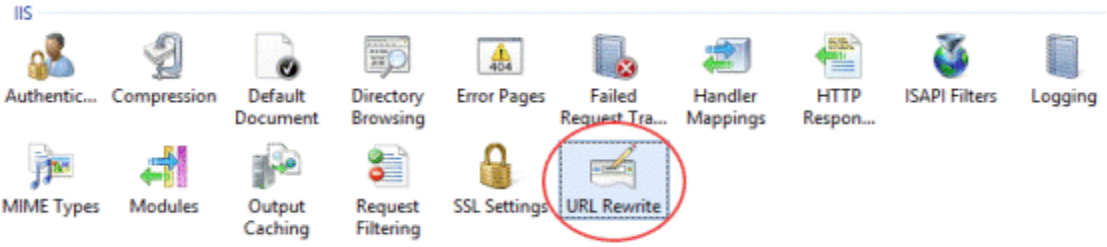
- Internet Information Services (IIS) 8.0 or later is required.
- The IIS **WebSockets protocol** must be enabled. See IIS 8.0 WebSocket Protocol Support for information on how to enable this protocol.
- IIS **Application Request Routing (ARR) 3.0** or later is required.
- The IIS **URL Rewrite** module must be installed.

Configure the IIS Reverse Proxy for Reflection ZFE

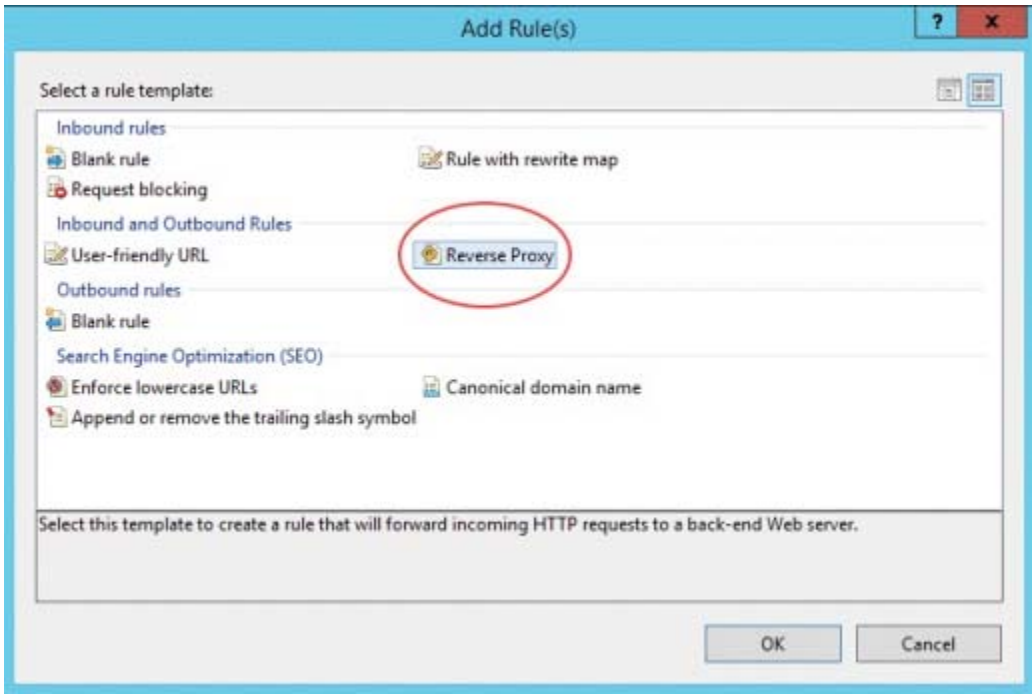
This example illustrates configuring an IIS server with the IP address of 192.168.1.1 to proxy connections to the Reflection ZFE Session Server at http://10.10.10.1:7070.

Configuring IIS

1 Launch the Internet Information Services (IIS) Manager, navigate to the web site you want to use, and open the **URL Rewrite** feature.



2 Choose the **Add Rule(s)** action and add a Reverse Proxy rule.



3 For the inbound rule, enter the Reflection ZFE server's IP address or host name and port. For example, if the Reflection ZFE session server is on the same machine as IIS and is using its default port, enter `localhost:7070`.

   After you create the rule, edit it and add a condition: {REQUEST_URI} matching the pattern (`.*/zfe.*`). This rule is necessary to ensure that native IIS content is accessible and not redirected to the specified host and port.

4 Check the outbound rule **Rewrite the domain names...** and enter the host name or IP address of the IIS server in the To: box

5 Click OK to create the new Reverse Proxy Rule.

Configuring Reflection ZFE

In order to proxy connections, the IIS **URL Rewrite** module must inspect and rewrite the web pages and WebSocket connections that pass through the proxy. For rewriting to succeed, these items must be sent in an uncompressed form. Note that, if configured, compression will still occur from the IIS server to the client's browser. The Reflection ZFE session server must also be configured to allow WebSocket connections to originate from the proxy.

1 Rename `jetty-web.xml`. For example, from `jetty-web.xml` to `jetty-web-disabled.xml`. The default location for this file is: `<install dir>/sessionserver/webapps/zfe/WEB-INF`.

2 Open `container.properties` in a text editor. The default location for this file is: `<install dir>/sessionserver/conf`.

3 Add the following lines to container.properties:

`websocket.compression.enable=false`

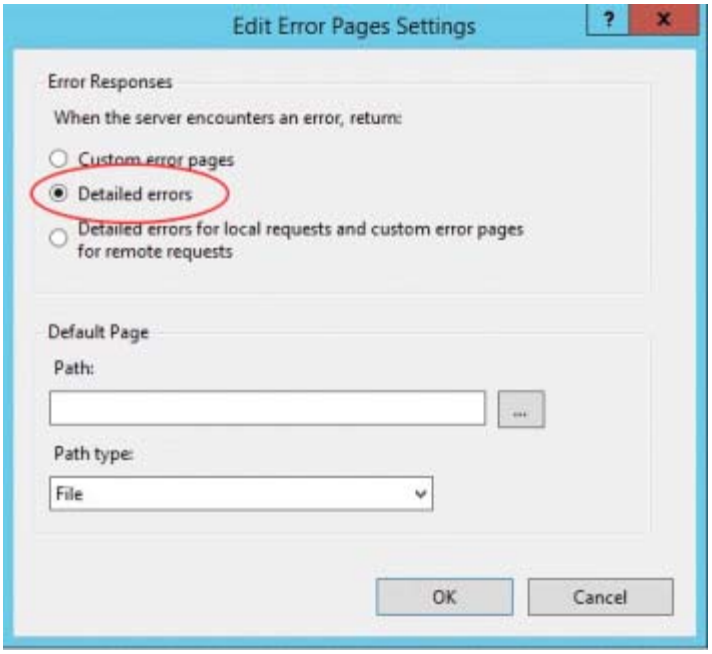
`websocket.allowed.origins=http://<IIS server name or IP address>`. For example: `192.168.1.1`.

Save changes to the file. The **Allowed Origins** property is a comma-delimited list of URLs. If web clients will be connecting to your website using an HTTPS connection, adjust the URL accordingly. If both secure and non-secure connections will be used, use both URLs as the value: `websocket.allowed.origins=http://192.168.1.1,https://192.168.1.1`. To avoid errors, make sure that all possible address formats are included in the Allowed Origins list.

4 Restart the web site and restart the Reflection ZFE session server and test the proxy by connecting to: `http(s)://192.168.1.1/zfe`.

Troubleshooting

If you receive web server errors, enabling detailed errors may help diagnose the problem. In the IIS Manager, open the **Error Pages** feature and check **Detailed errors**:



Typically errors in the 5XX range are caused by issues with compression being enabled or mistakes in the **Allowed Origins** value.

If the IIS proxy will be connecting to the Reflection ZFE session server with HTTPS, then the certificate used with the session server must be trusted by the IIS server. If the Reflection ZFE session server is using a self-signed certificate, this certificate must be added to the Windows trust store. If the Reflection ZFE session server is using a signed certificate, then the signer must be a trusted CA.

Improving Connection Times on Non-Windows Platforms

To improve connection times on non-Windows platforms, follow these steps after installing the Reflection ZFE session server, particularly when the system is virtualized or otherwise headless:

1. Stop the Reflection ZFE session server service.
2. Open the `<Reflection ZFE installation folder>/sessionserver/conf/container.conf` file in a text editor.
3. Locate this line, and edit as follows:

   `#wrapper.java.additional.x=-Djava.security.egd=file:///dev/urandom`

   - Remove the # to uncomment the line.
   - Replace x with <n+1>, where <n> is the highest number noted in the other `wrapper.java.additional.<n>` lines.
   - Save the file.
4. Restart the Reflection ZFE session server service.

Known Issues

These issues have been identified in previous releases and are known issues.

Browser issues

The following notes are specific to different web browsers.

Recommended browsers

It is highly recommended that you use Google Chrome or Mozilla Firefox. While Reflection ZFE supports Microsoft Internet Explorer (IE) 11, there are known performance issues with Internet Explorer's JavaScript engine that may negatively affect the end user experience with Reflection ZFE.

These issues have been identified and have remedies, however the easiest solution is to use a different browser.

Internet Explorer Unable to Play Recorded Reflection ZFE Macros

When using certain older versions of Microsoft Internet Explorer (IE) web browser with Reflection ZFE, attempts to playback macros may fail with an error. The error message reads: *Macro Error: Error transpiling macro code: TypeError: unknown: Circular reference in value argument not supported.*

This is a problem with this version of Internet Explorer and JavaScript. It may be possible to avoid this error if you delete the createMacro() function and replace it using JavaScript Promises (for example, then()).

Because this issue is specific to early versions of Internet Explorer, the easiest solution to this problem is to use a different browser (Chrome or Firefox) or a more recent version of Internet Explorer. You can successfully play back macros using Internet Explorer version 11.0.9600.18161, update version 11.0.27. Run Windows Update to update Internet Explorer.

HTTPS connections between Apple iOS mobile devices and the Reflection ZFE session server

Reflection ZFE users cannot connect to a session server over HTTPS from their Apple iPad when using a self-signed certificate. If feasible, the quickest solution is to use HTTP instead of HTTPS.

If HTTPS is needed, you have the following options:

- Obtain a valid certificate signed by a trusted CA and install it on the session server.
- Find an alternate browser that will accept the self-signed certificate. See System Requirements for a list of supported browsers.
- Leverage a custom certificate authority:
  1. Create a custom CA, CA root certificate, and a server certificate signed by that CA's root certificate.
  2. Install the server certificate on the session server.
  3. Install the custom CA root certificate on the iPad by means of a profile. The iPad should now accept the server certificate as it was signed by a "trusted CA".

  For a list of CAs trusted by Apple iOS, see Lists of available trusted root certificates in iOS (https://support.apple.com/en-us/HT204132).

Internet Explorer Displays Blank Screens

When using the Microsoft Internet Explorer (IE) web browser with Reflection ZFE (RZFE) or Host Access Management and Security Server (MSS), you may see a blank screen instead of the expected session.

When using Microsoft Internet Explorer to access Reflection ZFE sessions or Host Access Management & Security Server, you may experience issues such as the following:

- Reflection ZFE renders properly for some URLs and not others (a blank screen is displayed). The behavior varies depending on whether the ZFE session is using an IP address, short hostname, or fully-qualified name.
- In MSS, you can't create or open a ZFE session unless that session is on the same server as MSS. A blank screen displays where you expect to see the session.

Explanation

This issue is specific to the way Internet Explorer toggles various settings depending on its interpretation of website security. The settings in question are Compatibility View and Third-party Cookies. Depending on what "zone" Internet Explorer determines your web site to be in, these settings need to be either enabled or disabled. Internet Explorer bases its determination on the site URL. For example, if the server name in the URL does not contain dots (for example, http://mycorporateserver/mss/AdminStart.html), Internet Explorer assumes the address belongs in the Local Intranet zone. If it does, the site is assigned to the Internet zone.

Zone	Internet Explorer Default Settings
Local Internet Zone	Compatibility View enabled (not desired)
	Third-party Cookies enabled (desired)
Internet Zone	Compatibility View disabled (desired)
	Third-party Cookies disabled (not desired)

While it is possible for a website to override Compatibility View by specifying Document Mode with an X-UA-Compatible meta HTML tag, and Reflection ZFE does use that particular mode, MSS does not. Thus, if a Reflection ZFE server and a Management and Security Server are both in the Local Intranet zone (with default Compatibility View enabled), it is likely that Reflection ZFE would still perform correctly, but MSS would not. See http://blogs.msdn.com/b/ieinternals/archive/2012/06/05/the-local-intranet-security-zone.aspx for more information.

Solution

To use Internet Explorer 10 or 11 with ZFE and MSS servers, you need:

- Compatibility View disabled
- Third-party Cookies enabled

You need to determine what zone your web site is in and then make the necessary adjustments to the Internet Explorer settings. Because Internet Explorer can be configured in so many different ways depending on your situation, it is hard to provide one solution for successfully using Internet Explorer with Reflection ZFE and MSS. These are some possible configurations to follow:

- If both RZFE and MSS are in the Internet zone, manually add the RZFE server to the Local Intranet or Trusted Sites zone (Internet Options > Security > Local intranet > Sites). Use fully qualified host names or IP addresses.
- If both servers are in the Internet zone, change the default behavior for that zone and enable Third-party Cookies (Internet Options > Privacy > Advanced > Override automatic cookie handling).
- If both servers are in the Local Intranet zone, change the default behavior for that zone and disable Compatibility View (Tools > Compatibility View settings).

Key mapping issues with different browsers

Certain keys on a numeric keypad and some browser-specific keys cannot be mapped. For example, in Chrome, Ctrl+n and Ctrl+w cannot be mapped.

Host specific issues

The following are issues that are specific to different host types.

Displaying the Euro character

If the EURO character does not display correctly on the terminal screen, talk to your system administrator to make sure the host character set for the session is setup correctly. By default, Reflection ZFE uses a character set which does not support the Euro character (€). To display the Euro character, change the character set to one that supports the Euro character.

Issues encountered with VT hosts

Type	Description
Performance issues	◆ Heavy text output, such as form “Is-IR” may cause slow performance ◆ Scrolling regions may appear slow or choppy ◆ Cursor movememnt may be slow or choppy ◆ Internet Explorer is particularly slow, and performance degrades further when used for rows and columns.
Character sets	◆ Graphical characters and some character sets are not supported. ◆ Some non-English characters may cause the terminal display to freeze.
Other VT issues	◆ Insert/delete column (DECIC, DECDC) may fail. ◆ VT400 will not recognize DECSCL.

Field outlines in 3270 sessions

The 3270 attributes for field outlines are not fully supported. Reflection ZFE currently supports underline and overline; however, left vertical line, right vertical line, and combinations of the four line types are not yet supported.