Zowe Version 2.7.x Documentation

Table of contents:

Version: v2.7.x LTS

Zowe overview

Zowe™ is an open source software framework that allows mainframe development and operation teams to securely manage, control, script, and develop on the mainframe. It was created to host technologies that benefit the IBM Z platform for all members of the Z community, including Integrated Software Vendors (ISVs), System Integrators, and z/OS consumers. Like Mac or Windows, Zowe comes with a set of APIs and OS capabilities that applications build on and also includes some applications out of the box. Zowe offers modern interfaces to interact with z/OS and allows you to work with z/OS in a way that is similar to what you experience on cloud platforms today. You can use these interfaces as delivered or through plug-ins and extensions that are created by clients or third-party vendors. Zowe is a project within the Open Mainframe Project.

Zowe demo video

Watch this video to see a quick demo of Zowe.

Download the deck for this video | Download the script

Component overview

Zowe consists of the following components:

- Zowe Application Framework
- API Mediation Layer
- Zowe CLI
- Zowe Explorer
- Zowe Client Software Development Kits SDKs
- Zowe Launcher
- ZEBRA (Zowe Embedded Browser for RMF/SMF and APIs) - Incubator

Zowe Application Framework

A web user interface (UI) that provides a virtual desktop containing a number of apps allowing access to z/OS function. Base Zowe includes apps for traditional access such as a 3270 terminal and a VT Terminal, as well as an editor and explorers for working with JES, MVS Data Sets and Unix System Services.

▸ Learn more

API Mediation Layer

Provides a gateway that acts as a reverse proxy for z/OS services, together with a catalog of REST APIs and a dynamic discovery capability. Base Zowe provides core services for working with MVS Data Sets, JES, as well as working with z/OSMF REST APIs. The API Mediation Layer also provides a framework for Single Sign On (SSO).

▸ Learn more

To learn more about the architecture of Zowe, see Zowe architecture.

Zowe CLI

Zowe CLI is a command-line interface that lets you interact with the mainframe in a familiar, off-platform format. Zowe CLI helps to increase overall productivity, reduce the learning curve for developing mainframe applications, and exploit the ease-of-use of off-platform tools. Zowe CLI lets you use common tools such as Integrated Development Environments (IDEs), shell commands, bash scripts, and build tools for mainframe development. Though its ecosystem of plug-ins, you can automate actions on systems such

as IBM Db2, IBM CICS, and more. It provides a set of utilities and services for users that want to become efficient in supporting and building z/OS applications quickly.

<details>
<summary>Learn more</summary>
</details>

Zowe Explorer

Zowe Explorer is a Visual Studio Code extension that modernizes the way developers and system administrators interact with z/OS mainframes. Zowe Explorer lets you interact with data sets, USS files, and jobs that are stored on z/OS. The extension complements your Zowe CLI experience and lets you use authentication services like API Mediation Layer. The extension provides the following benefits:

- Enabling you to create, modify, rename, copy, and upload data sets directly to a z/OS mainframe.
- Enabling you to create, modify, rename, and upload USS files directly to a z/OS mainframe.
- Providing a more streamlined way to access data sets, uss files, and jobs.
- Letting you create, edit, and delete Zowe CLI `zosmf` compatible profiles.
- Letting you use the Secure Credential Store plug-in to store your credentials securely in the settings.
- Letting you leverage the API Mediation Layer token-based authentication to access z/OSMF.

For more information, see Information roadmap for Zowe Explorer.

Zowe Client Software Development Kits (SDKs)

The Zowe Client SDKs consist of programmatic APIs that you can use to build client applications or scripts that interact with z/OS. The following SDKs are available:

- Zowe Node.js Client SDK
- Zowe Python Client SDK

For more information, see Using the Zowe SDKs.

Zowe Launcher

Provides an advanced launcher for Zowe z/OS server components in a high availability configuration. It performs the following operations:

- Stopping the Zowe server components using the `STOP` (or `P`) operator command

- Stopping and starting specific server components without restarting the entire Zowe instance using `MODIFY` (or `F`) operator command

Zowe Chat (Technical Preview)

Zowe Chat is a chatbot that aims to enable a ChatOps collaboration model including z/OS resources and tools. Zowe Chat enables you to interact with the mainframe from chat clients such as Slack, Microsoft Teams, and Mattermost. Zowe Chat helps to increase your productivity by eliminating or minimizing the context switching between different tools and user interfaces.

<details>
<summary>Learn more</summary>
</details>

For more information, see Installing Zowe Chat and Using Zowe Chat.

ZEBRA (Zowe Embedded Browser for RMF/SMF and APIs) - Incubator

Provides re-usable and industry compliant JSON formatted RMF/SMF data records, so that many other ISV SW and users can exploit them using open-source SW for many ways.

For more information, see the ZEBRA documentation or visit the ZEBRA test/trial site.

Zowe IntelliJ Plug-in

Zowe IntelliJ plug-in for Intellij-based IDEs is a smart and interactive mainframe code editing tool that allows you to browse, edit, and create data on z/OS via z/OSMF REST API.

Zowe IntelliJ plug-in helps you to:

- Start working with z/OS easily with no complex configurations.
- Organize datasets on z/OS, files on USS into working sets.
- Allocate datasets, create members, files and directories with different permissions.
- Perform operations like renaming, copying and moving data in a modern way.
- Edit datasets, files and members. Smart auto-save will keep your content both in the editor and on the mainframe in-sync.
- Create multiple connections to different z/OS systems.
- Perform all available operations with jobs.

- Highlight all IntelliJ supported languages automatically and recognize them once opened from the mainframe.

For more information, see Using Zowe IntelliJ plug-in.

Zowe Third-Party Software Requirements and Bill of Materials

- Third-Party Software Requirements (TPSR)
- Bill of Materials (BOM)

Version: v2.7.x LTS

Overview

The installation of Zowe™ consists of the following processes:

- Installation of the Zowe server-side components.

  You can install the components either on z/OS only or you can install the components both on z/OS and on Docker.

- Installation of Zowe client-side components.

  You can install Zowe CLI or Zowe Explorer, a Visual Studio Code extension powered by Zowe CLI.

The Zowe server components provide a web desktop that runs a number of applications such as API Mediation Layer that includes the Single Sign-on (SSO) capability, organization of the multiple Zowe servers under a single website, and other useful features for z/OS developers.

Because Zowe is a set of components, before installing Zowe, use this guide to determine which components you want to install and where you want to install them.

Consider the following scenarios:

- If you plan to use Zowe CLI on PC only, you may not need to install the Zowe server components.

  Note: Some CLI plug-ins require the installation of components on z/OS. If you plan to use core Zowe CLI groups from your PC, the z/OS you connect to does not require any components of Zowe to be installed on z/OS, unless you want to take advantage of advanced authentication methods such as single sign-on or multi-factor authentication.

- If you use the Docker technical preview to run the Linux parts of Zowe in a container, you only need to configure the Zowe z/OS component to start the ZSS server.

Version: v2.7.x LTS

Getting started

Learn how to start exploring the Zowe components, applications and plug-ins.

Zowe server-side components

- Using Zowe Desktop
- Using Zowe API Mediation Layer

Zowe client-side components

- Using Zowe CLI
- Using Zowe Explorer
- Using Zowe SDKs

Explore available plug-ins

- Zowe CLI plug-ins
- Zowe Explorer extensions
- Using Zowe IntelliJ Plug-in

Incubator components

- Using Zowe Chat (incubator)

Version: v2.7.x LTS

Extending Zowe

Zowe is designed as an extensible tools platform. One of the Zowe architecture goals is to provide consistent interoperability between all Zowe components including extensions. The Zowe Conformance Program defines the criteria to help accomplish the aforementioned goal. By satisfying the Zowe Conformance Program criteria, extension providers are assured that their software remains functional throughout the Zowe release cycle. For more information, see the Zowe Conformance Program.

Zowe can be extended in the following ways:

On the server side:

- Extend Zowe API Mediation Layer
  - Dynamic API registration
  - Static API registration

On the client side:

- Extend Zowe CLI
- Add a plug-in to the Zowe Desktop
- Extend Zowe Explorer

To help Zowe extenders better understand how extensions are developed and deployed, we provide a set of sample extensions. These sample extensions contain the necessary boilerplate project setup, application code, and installation scripts to jumpstart the extension development and deployment to Zowe.

Note: For more information on the architecture of Zowe, see Zowe Architecture.

Extend Zowe CLI

Zowe CLI extenders can build plug-ins that provide new commands. Zowe CLI is built using Node.js and is typically run on a machine other than z/OS, such as a PC, where the CLI can be driven through a terminal or command prompt, or on an automation machine such as a DevOps pipeline orchestrator.

For more information about extending the Zowe CLI, see Developing a new plug-in. This article includes a sample plug-in that is provided with the tutorial; see Installing the sample plug-in.

Extend Zowe API Mediation Layer

Zowe API Mediation Layer extenders can build and onboard additional API services to the API ML microservices ecosystem. REST APIs can register with the API Mediation Layer, which makes them available in the API Catalog and for routing through the API Gateway.

To register a z/OS service with the API Mediation Layer, there are two approaches:

- Dynamic API registration
- Static API registration

For information about how to onboard REST APIs, see the Onboarding Overview.

To streamline the process of onboarding new REST API services to the API Mediation Layer, see Onboarding a REST API service with the YAML Wizard.

Dynamic API registration

Registration of a REST API service to the API ML is performed through a call to the Discovery Service by sending registration data and metadata for the service being registered. Registration requires that the z/OS service must know the web address of the API ML Discovery Service. When Dynamic registration is performed, the service that performs the registration must periodically send heartbeat requests to the Discovery Service for each registered service instance. These heartbeat requests serve to renew the corresponding service instance registration with API ML. These requests enable the Discovery Service to monitor the availability of registered service instances. Services that are registered dynamically display the status of the service in the API Catalog after initial service registration.

For more information about how to build a service which is able to register, see the Onboarding Overview.

Static API registration

For services that cannot be modified to be dynamically discoverable, it is possible onboard them to the API ML by providing the API ML a static definition file with API service details. This registration method does not require modifications to the existing API service code. For more information, see Onboard a REST API without code changes required. Unlike services that use Dynamic API registration, the status of services onboarded through Static API registration is not displayed in the API Catalog.

Add a plug-in to the Zowe Desktop

The Zowe Desktop allows a user to interact with z/OS applications through a web browser. The Desktop is served by the Zowe Application Framework Server on z/OS, also known as Z Lightweight User Experience (ZLUX). The Zowe desktop comes with a set of default applications. You can extend it to add new applications. For more information, see Developing for Zowe Application Framework.

The Zowe Desktop is an angular application that allows native plug-ins to be built that provide for a high level of interoperability with other desktop components. The React JavaScript toolkit is also supported. Additionally, you can include an existing web application in the Zowe Desktop using an iframe.

Notes: For more information, see the following samples:

- Sample iframe App.
- Sample Angular App.
- Sample React App.

Extend Zowe Explorer

Zowe Explorer provides extension APIs that assist third party extenders to create extensions that access Zowe Explorer resource entities to enrich the user experience. There are many ways Zowe Explorer can be extended to support many different use cases.

For the kinds of extensions that are supported and how to get started with extending Zowe Explorer, see Extensions for Zowe Explorer.

Sample extensions

Sample Zowe API and API Catalog extension

The repository https://github.com/zowe/sample-node-api contains a sample Zowe extension with a node server providing sample APIs for looking at cars in a dealership. For more information, see sample-node-api.

Sample Zowe Desktop extension

The repository https://github.com/zowe/sample-trial-app contains a sample Zowe extension with a node server providing a web page that gives a user interface to the APIs included with the API sample above.

Version: v2.7.x LTS

Troubleshooting overview

To isolate and resolve Zowe™ problems, you can use the troubleshooting and support information.

How to start troubleshooting

When you run into some issues and are looking for troubleshooting tips, the following steps may help you.

1. Search the error message or error code in your error log by using the Search bar in the Zowe Docs site. If there is an existing solution, follow the instructions to troubleshoot.

2. If no solution is available or the existing solutions cannot apply to your problem, you could search the keywords, error messages, or error code in the Zowe GitHub repository. If you find a closed issue or pull request, try troubleshooting by using the information shared in the item's Conversation section. If the issue is still open, post your question or comment to prompt a discussion on your problem.

2. If your problem is not solved, try the following options:

- Create an issue in the Zowe GitHub repository with a detailed description of the problem you have encountered.
- Bring up your questions to the corresponding channels as shown below:
    - Zowe CLI Slack channel
    - Zowe API ML Slack channel
    - Zowe Chat Slack channel
    - Zowe Documentation Slack channel
- Reach out to your available Zowe support team for assistance.

Known problems and solutions

Some common problems with Zowe are documented, along with their solutions or workarounds. If you have a problem with Zowe installation and components, review the problem-solution topics to determine whether a solution is available to the problem that you are experiencing.

You can also find error messages and codes, must-gathers, and information about how to get community support in these topics.

Troubleshooting Zowe server-side components

- Troubleshooting Zowe Launcher
- Troubleshooting Zowe z/OS component startup
- Troubleshooting API Mediation Layer
- Troubleshooting Zowe Application Framework

Troubleshooting Zowe client-side components

- Troubleshooting Zowe CLI
- Troubleshooting Zowe Explorer
- Troubleshooting Zowe Chat
- Troubleshooting Zowe IntelliJ plug-in

Verifying a Zowe release's integrity

Following a successful install of a Zowe release, the Zowe runtime directory should contain the code needed to launch and run Zowe. If the contents of the Zowe runtime directory have been modified then this may result in unpredictable behavior. To assist with this Zowe provides the ability to validate the integrity of a Zowe runtime directory, see Verify Zowe runtime directory

Understanding the Zowe release

Knowing which version of Zowe you are running might help you isolate the problem. Also, the Zowe community who helps you will need to know this information. For more information, see Understanding the Zowe release.

Version: v2.7.x LTS

Contribute to Zowe

You are welcome to contribute to Zowe in many forms and help make this project better! We want to make it as easy as possible for you to become a Zowe contributor. This topic outlines the different ways that you can get involved and provides some of the resources that are available to help you get started. All feedback is welcome.

Report bugs and enhancements

- Report bugs: Download and try one of the latest Zowe builds. Report any bugs you find by creating a Zowe bug report in GitHub.
- Report enhancements: Got an idea for a feature? Or something you're already using could be improved? Post an enhancement request in GitHub!
- Upvote enhancements and bugs: You can show us that an issue matters to you by applying the thumbs-up emoji for a specific issue. See this link to view the list of issues sorted by the most upvotes. This information is taken into account when planning the upcoming PI.

If you have an issue that is specific to a sub-project or community team, feel free to submit an issue against a specific repo.

Fix issues

- There are many issues and bugs with the label `Good first issue` in the Zowe GitHub repositories to help you get familiar with the contribution process. Check out the following list of GitHub repos to make your contribution!

    - Zowe sub-projects repositories
    - Zowe operations squads repositories

  When you decide to work on an issue, check the comments on that issue to ensure that it's not taken by anyone. If nobody is working on it, comment on that issue to let others know that you want to work on it to avoid duplicate work. The squad can assign that issue to you and provide guidance as well.

- You can also reach out to the Zowe squads on Slack to check with the squads if there is any good starter issue that you can work on.

Send a Pull Request

All code in Zowe aligns with the established licensing and copyright notice guidelines.

Before submitting a Pull Request, review the general Zowe Pull Request Guidelines and make sure that you provide the information that is required in the Pull Request template in that specific repo.

All Zowe commits need to be signed by using the Developer's Certificate of Origin 1.1 (DCO), which is the same mechanism that the Linux® Kernel and many other communities use to manage code contributions. You need to add a `Signed-off-by` line as a part of the commit message. Here is an example `Signed-off-by` line, which indicates that the submitter accepts the DCO:

```
Signed-off-by: John Doe <john.doe@hisdomain.com>
```

You can find more information about DCO signoff in the zac repo.

Report security issues

Please direct all security issues to `zowe-security@lists.openmainframeproject.org`. A member of the security team will reply to acknowledge receipt of the vulnerability and coordinate remediation with the affected project.

Contribution guidelines

Check out the contribution guidelines for different components and squads to learn how to participate.

- Zowe CLI
- Zowe API Mediation Layer
- Zowe Application Framework
- Zowe Explorer
- Zowe Client SDKs
- Zowe IntelliJ plug-in
- Zowe Docs

Promote Zowe

- Contribute a blog about Zowe. Read the Zowe blog guidelines to get started.
- Present Zowe on conferences and social channels

Helpful resources

- General code guidelines
- UI guidelines
- Zowe learning resources

Version: v2.7.x LTS

Zowe CLI command reference guide

View detailed documentation on commands, actions, and options in Zowe CLI. You can read an interactive online version, download a PDF document, or download a ZIP file containing the HTML for the online version.

Currently, this reference documentation only contains the web help for the Zowe CLI core component and CLI plug-ins maintained by Zowe. As third-party plug-ins are approved under the Zowe V2 LTS Conformance Program and contribute their web help to Zowe, we will update the documentation accordingly. To view the web help for V1 conformant plug-ins, click the version drop-menu on the top right corner of this page and click the link to any previous v1.xx.x version of this page.

- Browse online
- Download CLI reference in PDF format
- Download CLI reference in ZIP format