Zowe Conformance Test Evaluation Guide

The Zowe Conformance Test Evaluation Guide is a set of self-certify and self-service tests to help the developer community integrate and extend specific technology into the Zowe framework.

Below are the requirements for the available conformance programs. Items marked **(required)** are required for achieving conformance in a given program.

Zowe API Mediation layer - 2019

1) **Application Service**
   a) An application provides at least one service or UI **(required)**
2) **Register with discovery services**
   a) A service must be registered using one of the following methods: **(required)**
      Dynamic registration - **preferred** (best practice)
      Static definition - **minimum requirement**
   b) The service must provide a default service ID that is prefixed by the provider name (for example: `acme`, `xyzcorp`, `bar`). **(required)**
   c) The service ID must be configurable externally after deployment. **(required)**
   d) The service ID must written in lower case, contain no symbols, and is limited to 64 characters. **(required)**
   e) The API ID must follow the same rules as for Java packages. The example of the API ID: `org.zowe.apiml.apicatalog.` **(required)**
   f) The published service URL must follow the gateway URL conventions. **(required)**
   g) For versioned APIs, service URLs must contain a service version before the service ID in the following formats:
      - api/v1/{serviceId} reserved for REST APIs
      - ui/v1/{serviceId} reserved for UIs
      - ws/v1/{serviceId} reserved for WebSockets

      For non-versioned APIs or APIs versioned differently (e.g. z/OSMF), use the following formats:
      - api/{serviceId} reserved for REST APIs
      - ui/{serviceId} reserved for UIs
      - ws/{serviceId} reserved for WebSockets

      (required)
3) **API Documentation**
   a) Documentation is Swagger/Open API 2.0 compliant. **(required)**
   b) Every public resource is documented with a description of each resource. **(required)**
   c) Every method of each REST endpoint is documented. **(required)**
   d) Every method of each REST endpoint is demonstrated by an example. **(required)**
   e) Every parameter (headers, query parameters, payload, cookies, etc.) is documented with definitions of all possible values and their associated meanings. **(required)**
   f) Every error code, including errors returned in the payload is documented. **(required)**
4) **API naming and addressing**
   a) Encoded slash is not used. **(required)**

b) The service interprets values independent of their URL encoding. **(required)**
c) lowerCamelCase is used for names of resources, parameters, and JSON properties.

5) **Service requests and responses**
   a) API - Request and response payloads are in JSON or binary data format.
   b) API - In JSON format, links are relative, and must not contain the schema, hostname, and port. **(required)**
   c) WebSocket - Service URIs contained in WebSocket messages payload are addressed through the API ML Gateway. **(required)**
   d) UI - UI uses relative links and does not contain the schema, hostname, and port. **(required)**
6) **Authentication and Authorization**
   a) Resources are protected by mainframe credentials. **(required)**
   b) Services accept basic authentication (minimum requirement). **(required)**
   c) Services accept Zowe JWT token in the cookie**.**
7) **Versioning and Support**
   a) Service implementation follows the semantic versioning model.
   b) Last two major versions are supported by API services. **(required)**
8) **UI**
   a) UI uses only relative URLs. **(required)**
9) **WebSocket Services**
   a) WebSocket connection creation, all subsequent communication between WebSocket client, and server is routed through the API ML Gateway. **(required)**
   b) WebSocket connections are closed by the initiator through API ML Gateway. **(required)**

Zowe CLI - 2019

1) **Infrastructure**
   a) Plug-in is constructed on the Imperative CLI Framework. **(required)**
   b) Plug-in is NOT run as a standalone CLI. **(required)**
   c) Plug-in commands write to stdout or stderr via Imperative Framework response.console APIs. **(required)**
2) **Installation**
   a) Plug-in is installable with the zowe plugins install command. **(required)**
   b) Plug-in is installable into the @lts-incremental version of the core Zowe CLI and follows semantic versioning. **(required)**
   c) Plug-in is uninstallable via the zowe plugins uninstall command. **(required)**
3) **Naming**
   a) If the plug-in introduces a command group name, it does not conflict with existing conformant plug-in group names. **(required)**
4) **Profiles**
   a) If the plug-in has unique connection details, it introduces a profile that lets users store these details for repeated use. **(required)**
   b) Plug-in users are able to override all profile settings via the command line and/or environment variables.

Zowe App Framework – 2019

1) **Packaging**
   a) Every plugin must have a unique ID. The ID format follows java package naming conventions. The Zowe project reserves org.zowe. **(required)**
   b) Every plugin and each of its services must have a version. **(required)**
   c) Directory layout adheres to the ZLUX App filesystem structure. **(required)**
   d) Source code layout is recommended adheres to the ZLUX App filesystem structure for tooling consistency.
2) **Web UIs ALL**
   a) All Apps must contain an icon image file to represent it, located at web/assets/icon.png within the App's package. **(required)**
3) **Web UI IFrame**
   a) IFrame Apps (apps with framework type "iframe") which embed a top-level iframe within (example: https://github.com/zowe/api-layer/blob/master/zlux-api-catalog/web/index.html) must use the ID "zluxIframe" for that element. This is required for the app to be a recipient of app to app communication. **(required)**
4) **Web UI Non-IFrame**
   a) DOM elements originating from your App should always be a child of the Zowe viewport DOM element, "com-rs-mvd-viewport". **(required)**
   b) Network requests to the Zowe App Server must never be done without the use of the URI Broker. **(required)**
   c) Access to resources outside the App Server should also be made through the URI Broker whenever possible.
   d) Apps must not pollute the global namespace with regards Javascript, HTML, and CSS. **(required)**
   e) When using a library present in the Zowe App Framework core, you must depend on the same version. **(required)**
   f) Web apps should extend the framework's default build scripts for webpack and typescript.
5) **UI Design**
   a) Apps should follow the UI Design guidelines at https://github.com/zowe/zlc/blob/master/process/UI_GUIDELINES.md
6) **Localization and Internationalization (I10n and I18n)**
   a) The active language to be used for string selection must be retrieved using ZoweZLUX.globalization.getLanguage(), which determines language by multiple factors. **(required)**
   b) No strings visible in a UI should be hard-coded, rather resource strings must be used in accordance with one of the existing internationalization support mechanisms.
7) **App Server**
   a) Data services should be written such that all synchronous and asynchronous errors are caught. Utilize try-catch and check the existence of error objects from asynchronous calls. Uncaught exceptions effect server responsiveness and disrupt clients. **(required)**
8) **Documentation**
   a) Every HTTP API must be documented in swagger 2.0. The swagger document must be stored in doc/swagger. **(required)**
   b) In addition, it is recommended to have documentation about the format of any Websocket APIs, to be placed within doc.

9) **Logging**
   a) An Apps non-IFrame web components, or App Framework dataservices (eg Javascript and Typescript) must log only through the "zlux" logger. **(required)**
   b) ZSS services log only through the Zowe ZSS Logger. **(required)**
   c) Passwords must never be logged. **(required)**
   d) Error reporting should follow the standard tooling.
10) **Storage**
   a) User preferences, if applicable to a plugin, must be stored through the configuration data service. **(required)**