Ozone Quick Start Guide

DOD GOSS

Version 8.0.0.0-GA, 2019-12-31

Table of Contents

1. Introduction

1.1. Objectives

The purpose of this guide is to explain how to use the Ozone Widget Framework (OWF). This is including, but not limited to, the use of application components, full applications and their configuration settings. This guide provides an introduction to the Ozone Widget Framework (OWF). OWF consists of an environment and a set of tools used for discovering, organizing and displaying Web applications in a single browser window. The guide explains how to use OWF, set up an OWF environment on a user's local machine, and navigate OWF security.

1.2. Document Scope

This guide is not an exhaustive reference. It is intended for users, developers and administrators seeking a quick introduction on how to deploy, start, and use OWF. For information about specific areas, see the relevant documentation included with the OWF bundle.

In this document, the terms Store and Marketplace are used interchangeably.

OWF runs on the Django framework and requires Python 3.7 or higher. By default, OWF will run using Waitress, a WSGI server. Waitress does not support TLS and a reverse proxy will be required, if a secure connection is desired.

1.3. Related Documents

Table 1. Related Documents

Document	Purpose
Quick Start Guide	Walkthrough of basic OWF functions such as using widgets; unpacking the OWF bundle; setting up a local instance of OWF; installing security certificates; truststore and keystore configuration.
User's Guide	Understanding the OWF user interface; adding, deleting, modifying widgets and using intents; accessing and using the Store;using dashboards; creating, deleting, adding, switching, modifying dashboard pages; defining accessibility features such as high-contrast themes.
Administrator's Guide	Understanding administrative tools: adding, deleting, and editing users, groups, widgets, and dashboards; creating default content for users, groups and group dashboards.
Configuration Guide	Overview of basic architecture and security; OWF installation instructions; instructions for modifying default settings; database set up and logging guidance; framework and theme customization instructions; OWF upgrade instructions; .env file glossary and related information; directions for adding and deleting help content.

2. Components

This is an overview explaining how to start using the Ozone Widget Framework. Find detailed information in the OWF User's Guide.
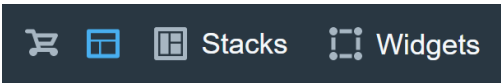
2.1. Toolbar

Figure 1. Toolbar (left)
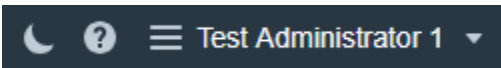
Figure 2. Toolbar (right)

Table 2. Toolbar buttons

	Marketplace view Select and switch the view to a Marketplace, to discover and add stacks and widgets.
	Dashboard view Switch the view to the active dashboard.
	Stacks menu Used to find, start, and manage stacks and dashboards.
	Widgets menu Used to list available widgets and add them to the dashboard.
	Theme toggle Toggle between Light and Dark themes.
	Help Online repository of Ozone guides and tutorials.

2.2. User menu

Figure 3. User drop-down menu

Table 3. User menu options

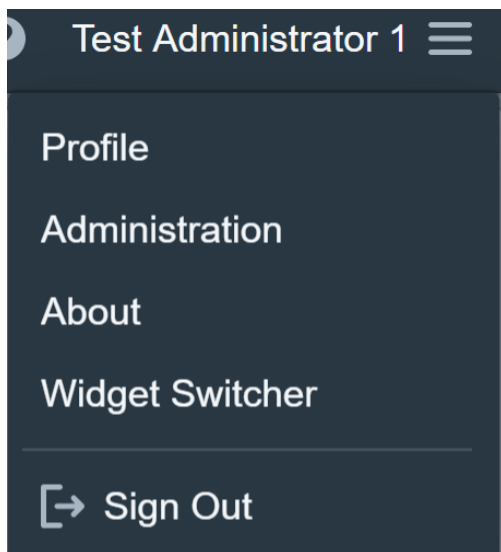| Profile	View user information for the currently logged in user.
Administration	Manage stacks, widgets, users, groups and system configuration settings. *(Note: This option will only appear to users with Administrator privileges.)*
About	Information box containing a description of the current version of the application.
Widget Switcher	Manage widgets that are running in the background.
Sign Out	Log out of the application.

2.3. Stacks

A stack is a collection of dashboards. It can be used to group similar dasboards into a single category that can then be shared with other users.

2.4. Dashboards

In simple terms, a dashboard is a screen where a user can dictate which widgets to load, which layouts to use, and the arrangement of the widgets within the specified layouts.

Each time a saved dashboard loads, the layout maintains the same look and feel as the last time the dashboard was accessed by the user.

Users can receive dashboards by the following methods:

- Create their own.
- Add from the Marketplace.
- Stack assigned to individual user.

- Stack assigned to a group that the user is a member of.

Group-assigned dashboards provide identical layouts and widgets for each member of a group. Each group member can customize their instance of a pre-configured dashboard. Dashboards that were shared to a user can be restored to their default states.

2.4.1. Stacks menu

The stacks menu lists all of the user's stacks and the stacks can be expanded to list all of the dashboards in each stack. The stacks included here are Ozone dashboards, either created in OWF or obtained from the Marketplace.

From the stacks menu, users can create, edit, share, restore, or delete a stack. Users can also create, restore, edit, or delete dashboards.

To open the stacks menu, click the stacks menu button [⊞ Stacks] in the OWF toolbar.
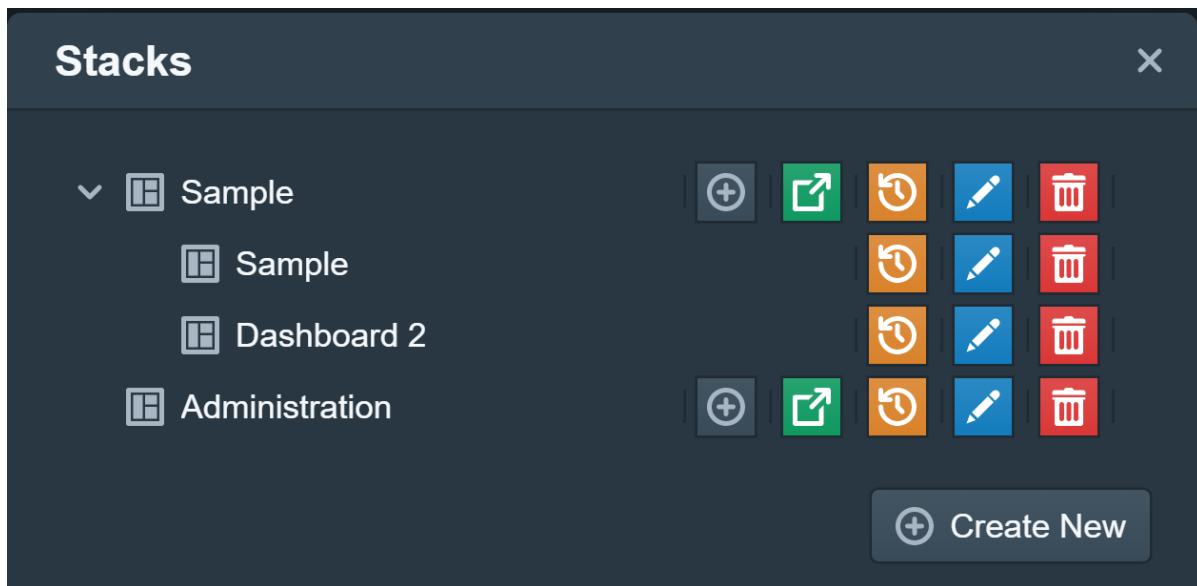

Figure 4. Stacks menu

Opening a dashboard

Stacks can contain one or more layouts called Dashboards. If a stack has only one dashboard, then clicking its icon in the stacks menu will start it.

To open a dashboard:

1. Click the stacks menu button in the toolbar to open the stacks menu.

2. Click a stack.

   a. If it has one dashboard, it will automatically open.

b. If it has more than one dashboard, then clicking the stack's icon will expand the list, displaying all of the stack's dashboards.

2.5. Widgets

A widget is a lightweight, single-purpose application that offers a summary or limited view of a larger application or dashboard. In OWF, a widget is a global description for a piece of Web content that can be configured by the user and displayed within an dashboard.

2.5.1. Widget toolbar

Users can access their widgets from the widget toolbar by clicking the widgets button ⬚ Widgets in the toolbar. Once open, the Widget Toolbar appears on the left side of the screen.
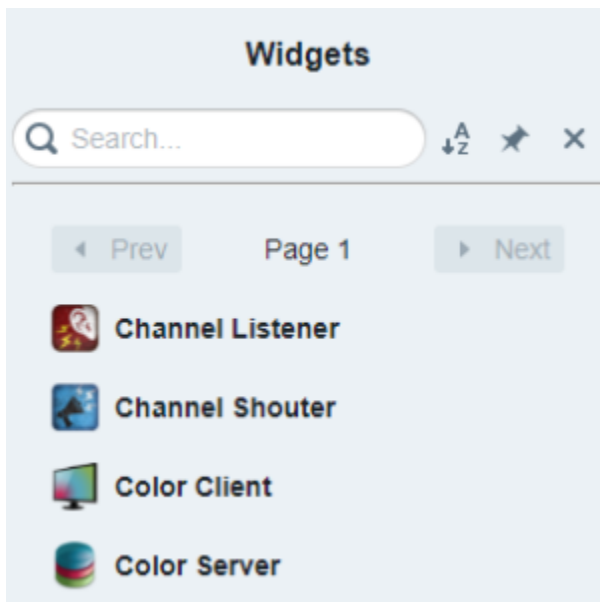
Figure 5. Widget toolbar

Adding a widget to a dashboard

To start a widget in a dashboard:

1.
   Open the widget toolbar by clicking the widgets button ⬚ Widgets in the toolbar.
2. Start a widget using one of the following methods:
   - Click the widget to immediately add it to the active dashboard.
   - Drag the widget from the widget Toolbar into the active dashboard.
     i. The widget toolbar disappears revealing the active dashboard.
     ii. Drop the widget in the desired location of the dashboard.

      iii. The widget toolbar reappears after the widget has been added.

3. Repeat this action to open another widget.

4. When finished, close the widget toolbar by clicking the ✕ in the upper-right corner.

2.6. Intents

Intents are the instructions for carrying out a widget's intentions. One widget requests an action (think of actions as verbs like view, share, edit, etc.), then another widget receives that request and performs the action. Intents build on OWF's publish / subscribe feature by allowing users to choose the widget(s) that will use data. This binding capability enables two widgets to share data in a way that improves their function.

For example, the NYSE widget charts data about the stock exchange. Some users may want to view that data as a Web page. This is possible if the NYSE widget has an Intent that tells it to send data to widgets that display data in a Web format.

> ℹ️ Widgets may have multiple intents associated with them. Users cannot create widget intents. Administrators and developers (logged in as administrators) add widget intents through the OWF administrator interface. Developers also add the intents through widget descriptor URLs.

2.6.1. How to Use Intents

When a widget sends an intent request, a pop-up window appears displaying all of the open widgets that can receive the requested intent action and data for an intended purpose (graphing, displaying, etc.).
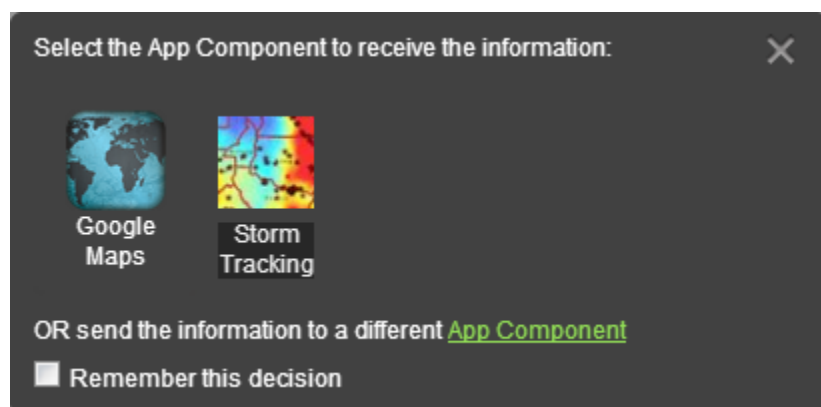

Figure 6. Widget selection dialog for intents

Select a widget to accept the requested Intent:

- Click one of the widgets displayed on the window OR

- Click the widget link to send the information to a widget that is not open on the screen:
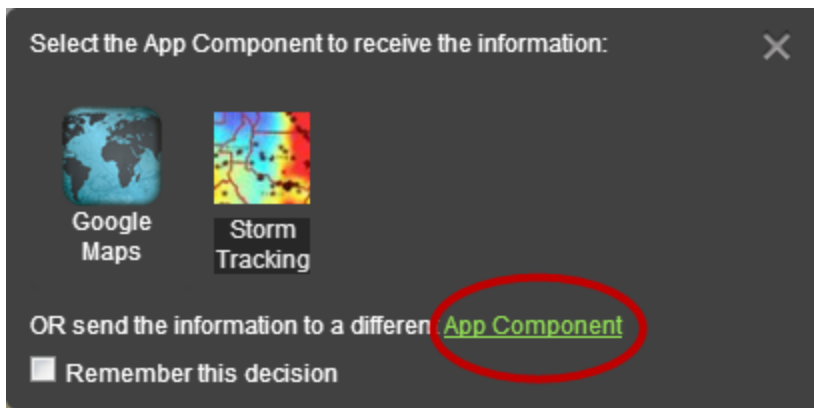
Figure 7. Select an alternate widget for the intent

> ℹ️ Checking the "Remember" box will allow the selected widget to automatically open the requesting widget's data. This function will continue until the user breaks the connection by closing either the sending or receiving widget.

After a user selects a receiving widget, the intent data is automatically sent to and processed by the receiving widget. To place the widget on the dashboard, click or drag it from the menu into the dashboard. Once the receiving widget is placed and open in the dashboard, it will receive the sending widget intent request.

2.7. Themes

OWF currently provides a light and a dark theme.

To toggle the selected theme:

1.
   Click the Theme button 🌙 in the toolbar.

2.8. The Marketplace

The AML Marketplace, similar to a commercial app store, operates as a thin-client registry of applications and services. The Marketplace provides search and discovery functionality that enables OWF users to find, add and share useful tools including (but not limited to) widgets, dashboards, and web applications.

If OWF has been configured to be connected to one or more Marketplaces, the Marketplace button allows the user to select and launch the Marketplace within the OWF user interface.

When a single Marketplace is connected, clicking the Marketplace button immediately launch the default Marketplace.

If multiple Marketplaces are connected, clicking the Marketplace button opens the Marketplace Switcher, allowing the user to select the desired Marketplace to launch.
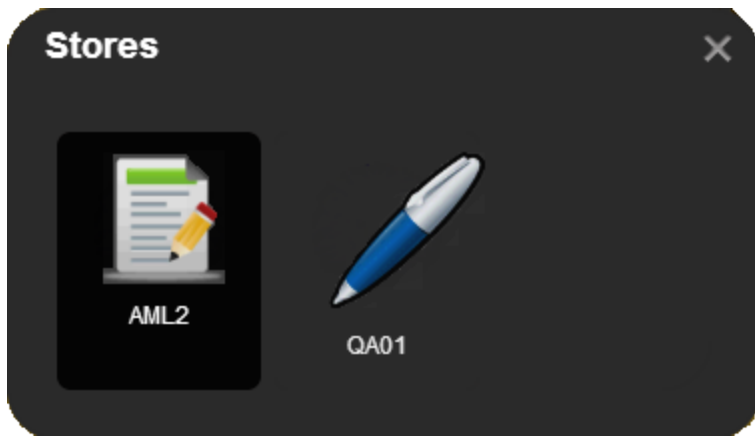
Figure 8. Marketplace selection dialog

3. The Ozone Bundle

OWF is normally distributed as a zipped bundle. This bundle contains the main component needed to deploy and run the framework.

3.1. Prerequisites

3.1.1. Python

Running the OWF bundle requires that Python be installed on the host machine.

OWF has been developed and fully tested using Python 3.7.4. Versions prior to or later than Python 3.7.4 are not currently supported.

3.1.2. Docker (optional)

The bundle will include a Dockerfile that can be used to launch OWF in a container.

3.2. Instructions

3.2.1. Overview

> The following is a summary. Please refer to the sections below for extended details.

1. Unzip the zipped OWF bundle.

2. Activate python virtual environment (optional)

3. From a command-line, start the bundle
   `OWF-8.X.X.X/start.sh`

4. In a supported browser, navigate to: http://localhost:8000/

5. Authenticate access to OWF by entering username `admin` and password `password`

3.2.2. Starting OWF

Scripts to start the server are included in the bundle in the root directory.

The initial data is inserted into the database by the start script.

By default OWF runs using in-memory database. If an external database is desired, follow the OWF Configuration Guide for instructions on configuring.

3.2.3. Accessing OWF

After the application has finished the initialization process, OWF can be accessed by opening a web browser and navigating to the following URL:

- http://localhost:8000/

3.3. Authenticating to OWF

In the default installation, the user may authenticate by using the default login form using a username and password.

3.3.1. Using the login form

To login as the default administrator, use the following credentials at the login form:

- username: `admin`
- password: `password`

To login as a default unprivileged user, use the following credentials at the login form:

- username: `user`
- password: `password`

3.3.2. Certificate authentication

If the system is configured for certificate-based (X509/PKI) authentication, sample user certificates are provided and must be installed in the user's browser. These certificates can be found under the config/ssl_auth/samples directory where the OWF bundle was deployed. See screenshots on the pages that follow for general instructions on importing certificates.

> The password for the user certificate is `password`.

3.3.2.1. Installing a user certificate using Internet Explorer (IE)

1. Navigate to Tools → Internet Options → Content → Certificates → Personal.

2. Click the Import button and navigate to the /tomcat/certs directory where the OWF bundle was deployed.

3. Select the testUser1 certificate and click OK.

4. Click Next and enter `password` as the password when prompted.

5. Select a folder to house the certificate.

6. Click Finish.

7. A dialog box should display, stating that the import was successful.

> ℹ️ #1 — In certain versions of Internet Explorer, certificate/connection failures are shown, despite a successful import of the testUser1 certificate. If this is the case, follow the directions above and select the X.509 from the drop-down and import the certificate titled ca.crt.
>
> #2 — The default dialog filter for a particular browser may be set for CER or CRT files. If this is the case, the drop-down for file type must be set for .p12. Once .p12, is selected, the certificate will show up as being available for importing.
>
> #3 — Some Intranets may require additional customization.
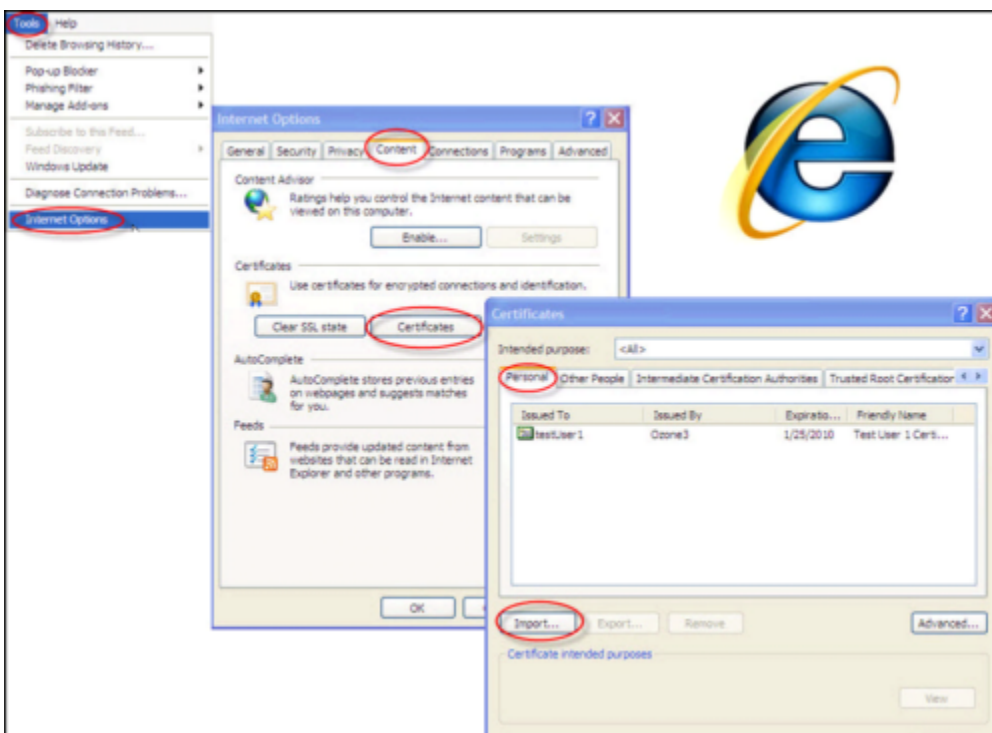
Figure 9. Internet Explorer user certificate dialog

3.3.2.2. Installing a user certificate using Firefox

1. Navigate to Tools → Options → Advanced → Encryption → View Certificate → Your Certificates.
   (Newer versions: → Options → Advanced → Certificates → View Certificates → Your Certificates.)

2. Click the import button and navigate to the /tomcat/certs directory where the OWF bundle was deployed.

3. Select the testUser1 certificate, click OK.

4. Enter `password` as the password when prompted.

5. Click Finish.

6. A dialog box should display, stating that the import was successful.

Figure 10. Firefox user certificate dialog

4. Development Build Process

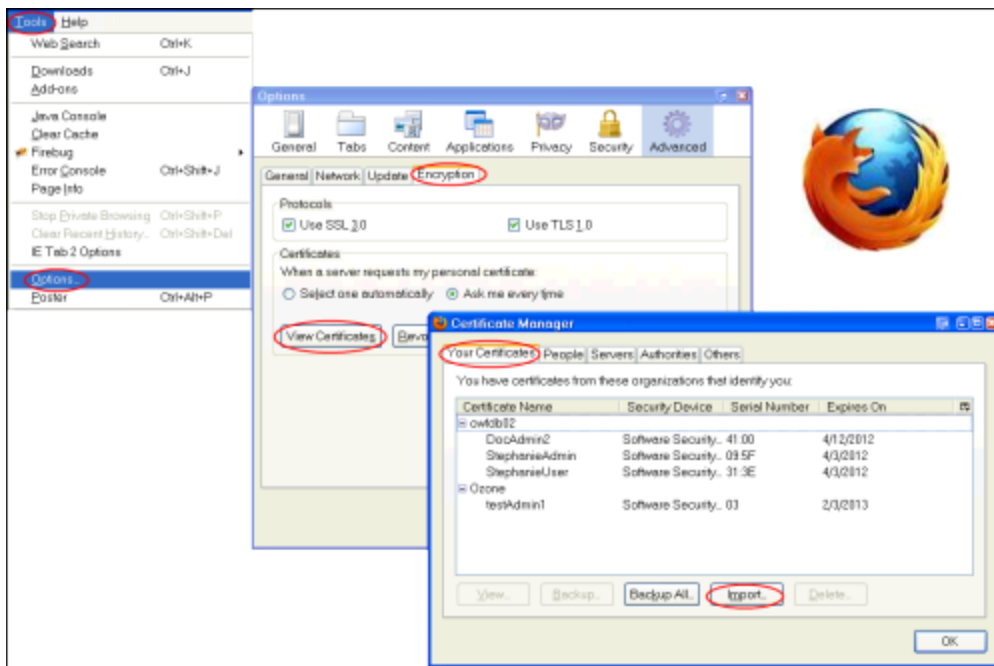OWF v8.0.0.0 introduced server-side rendering for the pages requested by the client(browser). This is to create a more configurable environment for the OWF application. Due to the nature of server-side rendering, both the server and the client projects will need to be built.

In order to integrate the client bundler(webpack) with the server-side rendering of the OWF pages, a bundle metadata file, `/ozone-framework-client/packages/application/webpack-stats.json`, is now generated by the client project and consumed by the OWF backend. This metadata file keeps track of the files generated by the webpack bundler, because webpack generates a different hash in the filename every build.

When the OWF server is started using the base settings, it will automatically look in the `/ozone-framework-client/packages/application/` directory for the metadata file. However, when running the server with the production settings, the metadata file will need to exist in `/ozone-framework-python-server/config/` directory.

4.1. Additional requirements

- NodeJs
- Python
- Cloned ozone GitHub repository
- Docker (optional)

4.2. Frontend Setup

Start using Node.js and npm

```
# install dependencies
ozone/ozone-framework-client/packages/application$ npm install

# build the client project bundle once
ozone/ozone-framework-client/packages/application$ npm run build
# or if the webpack hot-reload feature is desired, to see runtime changes to
the frontend view the server side `index.html` page (default
`localhost:8000`).
ozone/ozone-framework-client/packages/application$ npm run start

# copy the static files(icon images and etc.) to the build directory
ozone/ozone-framework-client/packages/application$ npm run copy-required-
public
```

4.3. Example Widgets Setup

Start using Node.js and npm

```
# from the /ozone-example-widgets directory


# install dependencies
ozone/ozone-example-widgets$ npm install


# build the client project bundle
ozone/ozone-example-widgets$ npm run build
# or if the OWF application expects the widgets to be hosted on a standalone
server running on localhost:4000
ozone/ozone-example-widgets$ npm run start


# copy the pre-bundled client widget api to the build directory
ozone/ozone-example-widgets$ npm run copy-owf-js
```

4.4. Backend Setup

Start using start script

```
# from the /ozone-framework-python-server directory

# it is recommended that you create a virtual python environment to avoid
poluting the global packages environment
# this can be achieved using the pipenv package
ozone/ozone-framework-python-server$ pip install pipenv
# create virtual env
ozone/ozone-framework-python-server$ pipenv shell

# install dependencies
ozone/ozone-framework-python-server$ pip install -r requirements.txt

# run migrations to create the database schema, if needed
ozone/ozone-framework-python-server$ python manage.py makemigrations

# run the start script, which will assure that the database schema is up-to-
date and load the default data.
ozone/ozone-framework-python-server$ ./start-dev.sh
```

4.5. Starting OWF via Docker

Start single app bundle

```
// From the project's root directory
ozone$ docker-compose up
```

4.5.1. Accessing OWF

Once all of the projects finish the initialization process, OWF can be accessed by opening a web browser and navigating to the following URL:

- http://localhost:8000/

5. Allowing Remote Access to OWF

To run OWF remotely, and NOT from a localhost environment, execute the following steps:

1. Identify a server host name.

2. Generate a server certificate.

3. Install the server certificate.

4. Modify configuration files.

5.1. Identifying a Server Name

The server host name can be chosen arbitrarily and entered into the users' HOST files, or it can be obtained from DNS. This Quick Start Guide will refer to the selected server host name as `servername` and to OWF as `https://servername:port/`.

5.2. Modify the Client's Environment Properties

In order to access OWF from remote computers, the proper setting must be set for the frontend to communicate with the backend.

The recommended approach is to set the environment variable for SERVER_URL via the .env file(explanation of this is provided in the Configuration Guide). The default is set to SERVER_URL=http://localhost:8000, but this should reflect the servername that is desired for the remote machine.

Glossary

Accordion (layout)

Display widgets in equal, horizontal panes that do not scroll (each individual widget may scroll using its own scroll bar).

Affiliated Store

A store that another organization uses for their system. When a local store is connected to an affiliated store, users in the local store can search for and add listings from the affiliated store (assuming the user has proper authentication for the affiliated store).

App

Deprecated term for a Stack.

App Component

Deprecated term for a widget.

Dashboard

An organized collection of widgets with a customizable layout.

Filters

A feature used to reduce the number of search results by type or category.

Fit (layout)

Allows a user to place a single widget on the screen.

Help

Repository of instructional guides and video tutorials.

Intent

Instructions for carrying out a widget's intentions.

Listing

Any software dashboard or widget that a user enters into the Store is called a "Listing." Listings can be a various types of Web content.

Marketplace

A searchable catalog of shared listings of widgets and stacks (also referred to as the Store).

OWF

Abbreviation for Ozone Widget Framework.

Pages

Deprecated term for a dashboard.

Portal (layout)

A column-oriented layout that organizes widgets of varying heights. Each new widget loads above the first one on the screen. The user drags a dividing bar to specify widget's height. The widgets and the Ozone window scroll.

Required Listings

An association between Listings. *Example: if Listing A needs Listing B to function, Listing B is a Required Listing.*

Stack

A collection of Dashboards (pages). Allows administrators and users to group Dashboards into folder-like collections that allow for easy transition from one to another.

Store

Commonly used term for the Ozone Marketplace.

Tabbed (layout)

Display one widget per screen, with tabs the top of the screen to switch from one widget to another.

Toolbar

The navigation bar at the top of the application. It links to a user's stacks, widgets, the Store, online Help and options from the drop-down User Menu.

User

A person signed into the Ozone application, usually referring to a person without administrative privileges.

Widget

A light-weight, single-purpose Web application that offers a summary or limited view of a larger Web application and may be configured by the user and displayed within a Dashboard.

Widget Menu

The Widgets Menu displays all available widgets. Use this feature to start or add widgets to a dashboard.