RADS

Introduction to Plugin Architecture

https://github.com/radservice/rad-community/fork

Last Revised on Oct 2021

Prerequisites

1. Understand Java web enterprise application and able to code.
2. Understand basic RADS components and its usages.

Content

1. Introduction to Plugins
2. Plugin Types

Chapter 1

Introduction to Plugins

What Is Plugin?

- Allows the functionality of the system to be extended dynamically.
- To achieve extensibility and adaptability of product features
- Any kind of integration that is not yet available in RADS as standard feature can be accomplished by developing a plugin, without breaking the fundamental core of the product.
- RADS plugin architecture supported 2 type of plugin structure
  - Standard Java Plugin
  - Dynamic OSGi Plugin

Standard Java Plugin

- Build as a standard Java JAR
- Plugin classes should be placed in a package name starting with net.purwana.rads. (OEM release package are allowed to specify other package names)
- Make JAR available in the Java classpath (e.g., place it under WEB-INF/lib or application server lib folder).
- Requires restarting the JVM for deployment or changes.
- May cause library version conflicts with base libraries or other plugins.
- Easier to develop and test using normal Java classes and libraries.

Dynamic OSGi Plugin

- Build as an OSGi (Open Services Gateway initiative framework) JAR bundle.
- Deploy JAR using the Manage Plugins in the Web Console.
- Supports dynamic loading/unloading/reloading without restarting.
- Runs in isolated mode, preventing library version conflict with base libraries or other plugins.
- More difficult to develop and test due to OSGi configuration and isolation.
- Technology powering this structure: Apache Felix.

Chapter Review

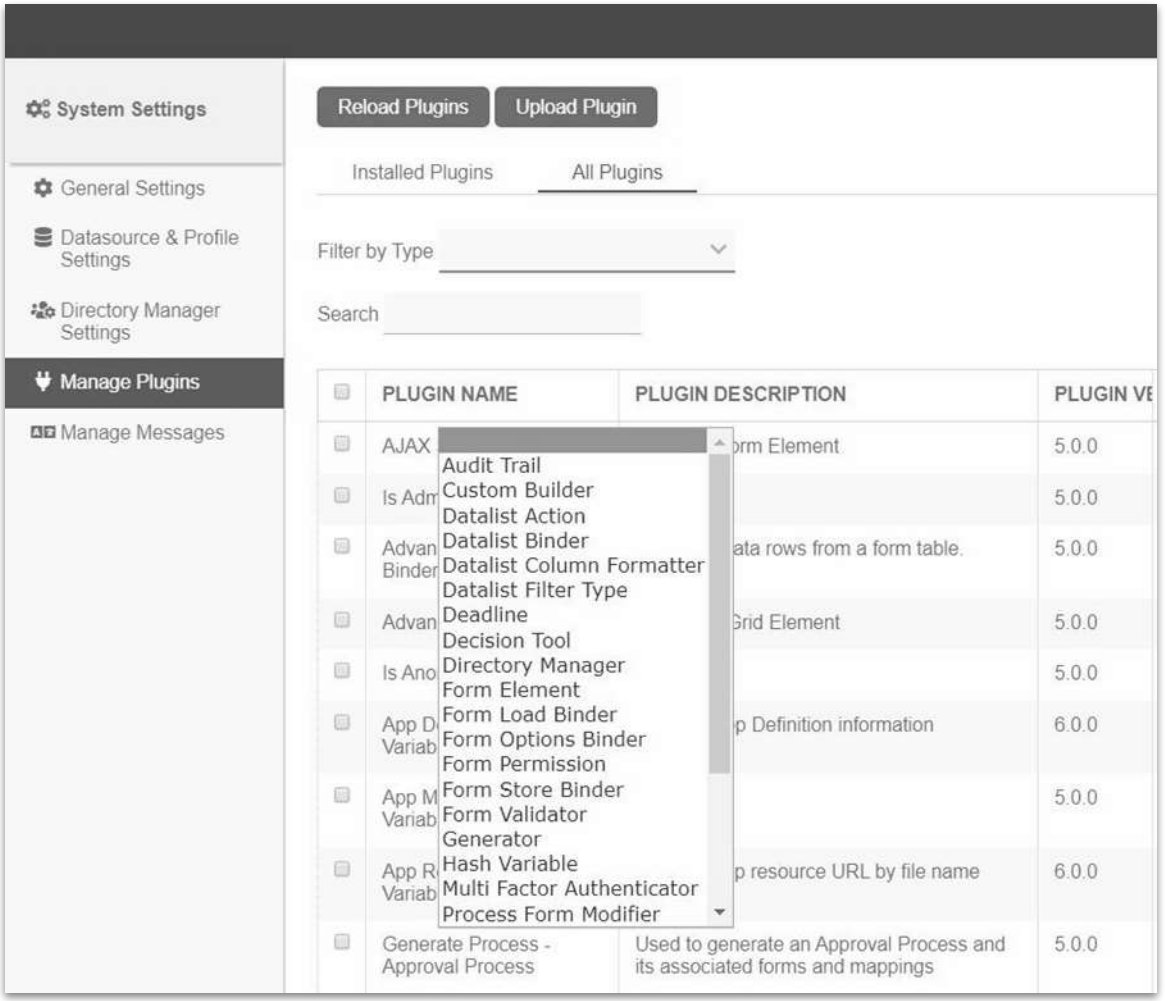
- Able to differentiate between the 2 types of plugin.

Chapter 2

Plugin Types

Locate Available Plugins

- Settings -> Manage Plugins



Workflow Engine Plugin Types

- **Deadline Plugins** provide the ability to recalculate deadline limit and SLA limit based on programming logic.
- **Process Participant Plugins** are used to provide custom selection of users to workflow participants.
- **Process Tool / Post Form Submission Processing Plugins** to integrate with external systems and allow extra processing in the middle of a process.

Workflow Engine Plugin Types

- **Rules Decision Plugins** (New in RADS) are used in a route in a process flow to decide the next transition to flow to after the route.
- **Process Form Modifier Plugin** (New in RADS) is used to modify the form design mapped to a process flow activity.

Datalist Builder Plugin Types

- **Datalist Action Plugins** to extend methods of executing an action on list item. (e.g. Delete a record)
- **Datalist Binder Plugins** to extend methods of loading data for a list.
- **Datalist Column Formatter Plugins** to extend ways of formatting column data.
- **Datalist Filter Type Plugins** provides more means of filtering data.

Userview Builder Plugin Types

- **Userview Menu Plugins** to extend types of pages available in Userview Builder.
- **Userview Permission Plugins** to handle permissions and access rights in a userview.
- **Userview Theme Plugins** to change the UI design of userview.

Form Builder Plugin Types

- **Form Field Element Plugins** to extend types of fields available in Form Builder.
- **Form Permission Plugins** to handle permissions and access rights in a form.
- **Form Load Binder Plugins** to extend methods of loading data in a form from any data source.
- **Form Options Binder Plugins** to extends method to loading data for a form field options from any data source.

Form Builder Plugin Types

- **Form Store Binder Plugins** to extend methods of storing data in a form to any data source.
- **Form Validator Plugins** to extend ways to validate form data.

App Level Plugin Types

- **Audit Trail Plugins** is triggered after process related event to provide extra processing capabilities. (e.g. Capture reporting data or user notification.)
- **Hash Variable Plugins** to extend support of processing Hash Variable.
- **Web Service Plugins** to provide additional HTML page or Web Service for AJAX call for the system.
- **Generator Plugins** to generate/scaffolding work for app based on current working form in the Form Builder.

System Level Plugin Types

- **Directory Manager Plugins** to integrate users from external system. E.g. Active Directory or LDAP.
- **Multi Factor Authenticator Plugin** is used by Security Enhanced Directory Manager to provide multi factor authentication feature.
- **Web Service Plugins** to provide additional HTML page or Web Service for AJAX call for the system.
- **Custom Builder** (New in RADS) is used to extend the builder feature in addition to the default form, list, userview & process builder.

Chapter Review

- Understand all types of plugins and the purpose that each of the plugin type serves.

Module Review

1. Introduction to Plugins.
2. Plugin Types.

Recommended Further Learning

- Building RADS from source.
- Learn to create a Plugin.

Stay Connected With RADS

- rads.purwana.net
- https://github.com/radservice/rad-community