Customization With Overlay Project

Introduction

There are many ways how to customize midPoint behavior. MidPoint is designed with practicality in mind and therefore the most frequent customizations can be done by configuration. But it is not possible to make everything configurable. Some customizations needs extension or even modification of midPoint code.

This page describes how to set up and maintain an *overlay project* that contains midPoint customizations.

How Overlay Works?

Overlay projects takes midPoint binary distribution (`midpoint.war`), extracts it, adds your customization to it and then repackages it again. The customizations may be configuration files, web resources (HTML, CSS, images) and even Java code. The overlay can also override stock midPoint files and classes and replace them with custom versions.

The overlay project is build using Apache Maven. As midPoint itself is built using Maven and midPoint development binaries are also distributed in Maven repositories this is a natural choice.

The overlay project contains only the customized files. There is no need to copy stock midPoint sources and even binaries. The Maven will download everything that it needs directly from Evolveum repositories. As the overlay project only contains customizations it can easily be maintained in a version control system.

Example Overlay Projects

GUI Customization Overlay Example

The example overlay project is located in the Evolveum/midpoint-overlay-example github repository:

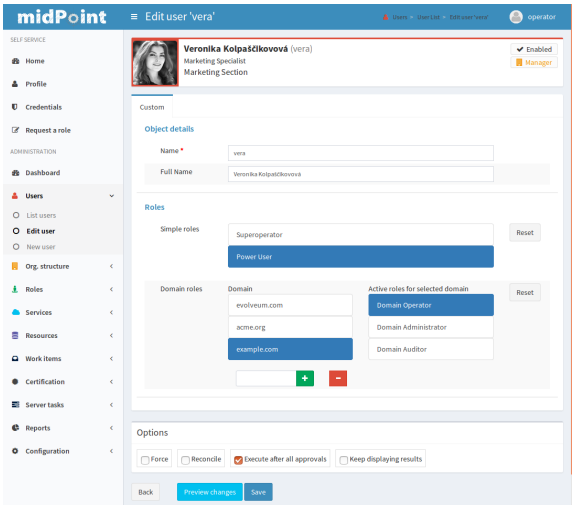
Version	
master (development)	https://github.com/Evolveum/midpoint-overlay-example
3.9 (latest stable)	https://github.com/Evolveum/midpoint-overlay-example/tree/v3.9
3.8	https://github.com/Evolveum/midpoint-overlay-example/tree/v3.8
3.7	https://github.com/Evolveum/midpoint-overlay-example/tree/v3.7
3.6	https://github.com/Evolveum/midpoint-overlay-example/tree/v3.6
3.5	https://github.com/Evolveum/midpoint-overlay-example/tree/v3.5
3.4.1	https://github.com/Evolveum/midpoint-overlay-example/tree/v3.4.1
3.4	https://github.com/Evolveum/midpoint-overlay-example/tree/v3.4

Download the project and build it using the following maven command:

```
mvn package
```

The resulting file is `target/midpoint.war`. This is the file that contains stock midPoint together with the customizations introduced by the overlay project. Simple deploy the WAR file in the usual way. Customized version of midPoint will start.

Log in as user operator (password: 5ecr3t) and try to edit any user. You will see completely customized form:



The code of this form is in the overlay project in the `src/main/java/com/example/midpoint/gui/forms` directory.

You might notice, that the midPoint deployment also added new objects to the repository and new schema file. These files are also located in the overlay in the `src/main/resources` directory.

Custom Web Service Overlay Example

The example overlay project is located in the Evolveum/midpoint-custom-service github repository:

Version	
master (development)	https://github.com/Evolveum/midpoint-custom-service
3.9 (latest stable)	https://github.com/Evolveum/midpoint-custom-service/tree/v3.9
3.8	https://github.com/Evolveum/midpoint-custom-service/tree/v3.8
3.7	https://github.com/Evolveum/midpoint-custom-service/tree/v3.7
3.6	https://github.com/Evolveum/midpoint-custom-service/tree/v3.6
3.5	https://github.com/Evolveum/midpoint-custom-service/tree/v3.5
3.4.1	https://github.com/Evolveum/midpoint-custom-service/tree/v3.4.1
3.4	https://github.com/Evolveum/midpoint-custom-service/tree/v3.4

Initial Objects

Stock midPoint installation has a set of objects that are deployed to midPoint repository when a fresh midPoint installation starts. This feature can also be used in the overlay projects. Just put your custom objects into the `src/main/resources/initial-objects` directory. Place each object in a separate XML file. The files have to be named following the 9*xx*-*nnnnnnnn*.`xml` convention, where *xx* is any number and *nnnnnnnn* is object name (see the sample project for an example). The files will be imported in the order given by the *xx* numbers. The prefix 9 is a convention to avoid collisions with stock initial objects.

Initial midPoint Home

The overlay project may also contain files that are copied to MidPoint Home Directory when midPoint starts. Just place these files in `src/main/resources/initial-midpoint-home` directory in the overlay project. This is a good tool how to bundle schema files that contain custom schema extensions.

Web Services

The overlay project can quite easily implement a custom web service. Although midPoint provides a comprehensive IDM Model Web Service Interface, custom web service is sometimes required for integration purposes. There is a sample overlay project that implements a simple web service: https://github.com/Evolveum/midpoint-custom-service

Developer corner

Are you using the latest snapshot as your base for the overlay project? The snapshots are usually build once per day in the night. Sometimes this is not fresh enough and you may even need to build your overlay project on the very latest commit that just popped up on the GitHub. Here is example how you do it easily with maven.

```
// first go to the midPoint project you just pulled from the GitHub and do the usual
mvn clean install -DskipTests=true -T4

// after build is complete install binaries to your LOCAL maven repo:
mvn install:install-file -Dfile=.\gui\admin-gui\target\midpoint.war -DgroupId=com.evolveum.midpoint.gui -
DartifactId=admin-gui -Dversion=3.6-SNAPSHOT -Dpackaging=war
mvn install:install-file -Dfile=.\gui\admin-gui\target\midpoint-classes.jar -DgroupId=com.evolveum.midpoint.gui
-DartifactId=admin-gui -Dversion=3.6-SNAPSHOT -Dclassifier=classes -Dpackaging=jar

// now go to your overlay project and use the -o option to make maven build in the offline mode
mvn package -o
```

TODO Describe

- use in eclipse
- customizing look and feel
- customization and upgradeablity
- bundling connectors

See Also

- GUI Development Guide
- Look & Feel Customization HOWTO