# OpenL Tablets BRMS WebStudio User Guide

Release 5.23

**Document number:** TP_OpenL_WS_UG_3.5_LSh

Revised: 11-23-2020

# Table of Contents


1 Preface ... 6
1.1 Audience ... 6
1.2 Related Information ... 6
1.3 Typographic Conventions ... 6
2 Introducing OpenL Tablets WebStudio ... 8
2.1 What Is OpenL Tablets WebStudio? ... 8
2.2 Working with Projects in OpenL Tablets WebStudio ... 8
2.3 OpenL Tablets WebStudio Components ... 10
2.4 Security Overview ... 11
3 Getting Started ... 12
3.1 Signing In to OpenL Tablets WebStudio ... 12
3.2 Modifying User Profile ... 12
Modifying User Details ... 13
Modifying User Settings ... 14
3.3 Displaying the OpenL Tablets WebStudio Help ... 15
3.4 Signing Out of OpenL Tablets WebStudio ... 15
3.5 Introducing Rules Editor ... 15
Rules Editor Overview ... 15
View Modes ... 17
3.6 Introducing Repository Editor ... 20
4 Using Rules Editor ... 22
4.1 Filtering Projects ... 22
4.2 Viewing a Project ... 23
4.3 Viewing a Module ... 23
4.4 Managing Projects and Modules ... 24
Editing and Saving a Project ... 25
Saving a Project for Backward Compatibility ... 25
Updating and Exporting a Project ... 26
Copying a Project ... 26
Exporting, Updating, and Editing a Module ... 27
Comparing and Reverting Module Changes ... 28
Copying a Module ... 31
4.5 Defining Project Dependencies ... 33
4.6 Viewing Tables ... 34
4.7 Modifying Tables ... 35
4.8 Referring to Tables ... 37
4.9 Managing Range Data Types ... 38
4.10 Creating Tables by Copying ... 41
Copying the Existing Table ... 42
Creating a New Version of the Table ... 42
Creating a Table as a New Business Dimension Version ... 42
4.11 Performing a Search ... 43
Performing a Simple Search ... 43

Performing an Advanced Search ..... 44
4.12 Creating Tables ..... 46
Creating a Datatype Table ..... 46
Creating a Data Table ..... 48
Creating a Test Table and Defining the ID Column for Test Cases ..... 50
Creating a Simple Rules Table ..... 53
4.13 Comparing Excel Files ..... 55

**5 Editing and Testing Functionality ..... 58**

5.1 Editing Tables ..... 58
Editing a Comma Separated Array of Values ..... 58
Editing Default Table Properties ..... 58
Editing Inherited Table Properties ..... 59
5.2 Using Table Versioning ..... 63
5.3 Performing Unit Tests ..... 64
Adding Navigation to a Table ..... 65
Running Unit Tests ..... 65
Creating a Test ..... 71
5.4 Tracing Rules ..... 71
5.5 Using Benchmarking Tools ..... 74

**6 Using Repository Editor ..... 77**

6.1 Browsing Design Repository ..... 77
6.2 Filtering the Project Tree ..... 78
6.3 Creating Projects in Design Repository ..... 78
Creating a Project from Template ..... 79
Creating a Project from Excel Files ..... 81
Creating a Project from ZIP Archive ..... 82
Importing a Project from Workspace ..... 83
6.4 Opening a Project ..... 84
Opening a Project Revision Using the Open Revision Button ..... 84
Opening a Project Revision Using the Revisions Tab ..... 85
6.5 Closing a Project ..... 86
6.6 Saving a Project ..... 86
6.7 Viewing a Project Properties ..... 86
6.8 Modifying Project Contents ..... 87
Creating a Folder ..... 87
Uploading a File ..... 87
Updating a File ..... 88
Deleting a Folder or a File ..... 88
Copying a File ..... 89
6.9 Copying a Project ..... 90
6.10 Removing a Project ..... 91
Deleting a Project ..... 91
Erasing a Project ..... 91
6.11 Deploying Projects ..... 91
Creating a Deploy Configuration ..... 92
Defining Projects to Deploy ..... 92
Deploying a Deploy Configuration ..... 93
Opening Deployed Configurations ..... 93
Redeploying Projects ..... 93

Configuring Additional Rules Deploy Configuration Settings ..... 94
Defining Rule Service Version ..... 96
6.12 Comparing Project Revisions ..... 98
6.13 Exporting a Project or a File ..... 99
6.14 Unlocking a Project ..... 100
6.15 Browsing the Deployment Repository ..... 100
**7 Working with Project Branches ..... 102**
7.1 Creating a Branch ..... 102
7.2 Working with Branches ..... 102
7.3 Resolving Conflicts ..... 105
**8 Using Administration Tools ..... 106**
8.1 Managing Common Settings ..... 107
Managing User Workspace Settings ..... 107
Managing History Settings ..... 107
Managing Other OpenL Tablets WebStudio Settings ..... 107
8.2 Managing Git Repository Settings ..... 107
Setting Up a Connection to a Git Repository ..... 108
Setting Up Git Repository Functionality ..... 108
8.3 Managing Repository Settings ..... 110
8.4 Managing System Settings ..... 112
8.5 Managing User Information ..... 113
Managing Groups ..... 113
Managing Users ..... 116
**9 Working with Design Repository via REST ..... 121**
9.1 Commands for the Design Repository REST Service ..... 121
9.2 Authentication and Authorization ..... 122
**Appendix A: ZIP Project Structure ..... 123**
Single Project Structure ..... 123
Single Project Structure #2 ..... 123

# 1 Preface

This preface is an introduction to the *OpenL Tablets WebStudio User Guide*. The following topics are included in this preface:

- Audience
- Related Information
- Typographic Conventions

## 1.1 Audience

This guide is intended for the following users:

| Audience | | |
|---|---|---|
| **User type** | **Purpose** | **Required knowledge** |
| Business users | View and modify company business rules stored in tables. | Knowledge of decision tables is required. |
| Developers | • Manage technical details of rule tables.<br>• Organize and deploy rule projects. | Knowledge of OpenL Tablets technology is required. |

## 1.2 Related Information

OpenL Tablets WebStudio is a tool of the OpenL Tablets product. For information on OpenL Tablets Rules, see ***[OpenL Tablets Reference Guide]***.

## 1.3 Typographic Conventions

The following styles and conventions are used in this guide:

| Typographic styles and conventions | |
|---|---|
| **Convention** | **Description** |
| **Bold** | • Represents user interface items such as check boxes, command buttons, dialog boxes, drop-down list values, field names, menu commands, menus, option buttons, perspectives, tabs, tooltip labels, tree elements, views, and windows.<br>• Represents keys, such as **F9** or **CTRL+A**.<br>• Represents a term the first time it is defined. |
| `Courier` | Represents file and directory names, code, system messages, and command-line commands. |
| **`Courier Bold`** | Represents emphasized text in code. |
| Select **File > Save As** | Represents a command to perform, such as opening the **File** menu and selecting **Save As**. |
| *Italic* | • Represents any information to be entered in a field.<br>• Represents documentation titles. |
| < > | Represents placeholder values to be substituted with user specific values. |
| Hyperlink | Represents a hyperlink. Clicking a hyperlink displays the information topic or external source. |

| Typographic styles and conventions | |
|---|---|
| **Convention** | **Description** |
| ***[name of guide]*** | Reference to another guide that contains additional information on a specific feature. |

# 2 Introducing OpenL Tablets WebStudio

This chapter introduces main OpenL Tablets WebStudio concepts. The following topics are included in this chapter:

- What Is OpenL Tablets WebStudio?
- Working with Projects in OpenL Tablets WebStudio
- OpenL Tablets WebStudio Components
- Security Overview

## 2.1 What Is OpenL Tablets WebStudio?

**OpenL Tablets WebStudio** is a web application employed by business users and developers to view, edit, and manage business rules and rule projects created using OpenL Tablets technology. For more information on OpenL Tablets, see ***[OpenL Tablets Reference Guide]***.

By using OpenL Tablets WebStudio, users can modify rules directly in a web browser without installing additional tools. OpenL Tablets WebStudio provides an advanced functionality for creating and modifying rules, viewing errors, and executing tests.

## 2.2 Working with Projects in OpenL Tablets WebStudio

OpenL Tablets WebStudio is intended for a multi-user environment. It provides a centralized storage of rule projects called **Design repository**. Design repository is stored on the OpenL Tablets WebStudio server and can be accessed by any user. However, users cannot modify projects directly in Design repository. Instead, to make modifications to a project, users must execute the following procedure:

| **Procedure for modifying a project** | | |
|---|---|---|
| **Step** | **Action** | **Description** |
| 1 | Open a project. | When a project is opened, its status is set to **No Changes**, and a copy of it is created in the user's workspace, a specific location on the OpenL Tablets WebStudio server. Work copies of projects made editable by a particular user are stored there. Users can only access their personal workspaces. |
| 2 | Modify a project. | After any modification of a project, its status is set to **In Editing**.<br>A project in the **In Editing** status is locked in Design repository to avoid loss of information. Other users cannot edit it until the project is saved. Other users can only open the project in read-only mode, with the **No Changes** status.<br>Modifications to a project in the **In Editing** status are performed on the working copy stored in the user's workspace. Modifications do not become immediately visible to other users. |
| 3 | Save a project. | Saving a project copies the modified copy of the project from the user's workspace to Design repository. A new revision of the project is created in Design repository. A project can be restored to any of its previous revisions.<br>From this moment, changes are visible to other users and the project is available for editing. |

Closing a project deletes it from the user's workspace without saving changes and does not affect the revision in Design repository. Closed projects can be browsed in repository editor but are not available in Rules Editor.

The following diagram illustrates general rules project lifecycle. This is a simplified schema of rules development workflow where activities as opening, opening for editing, closing, deleting, and erasing the current project or deploying configuration are omitted.

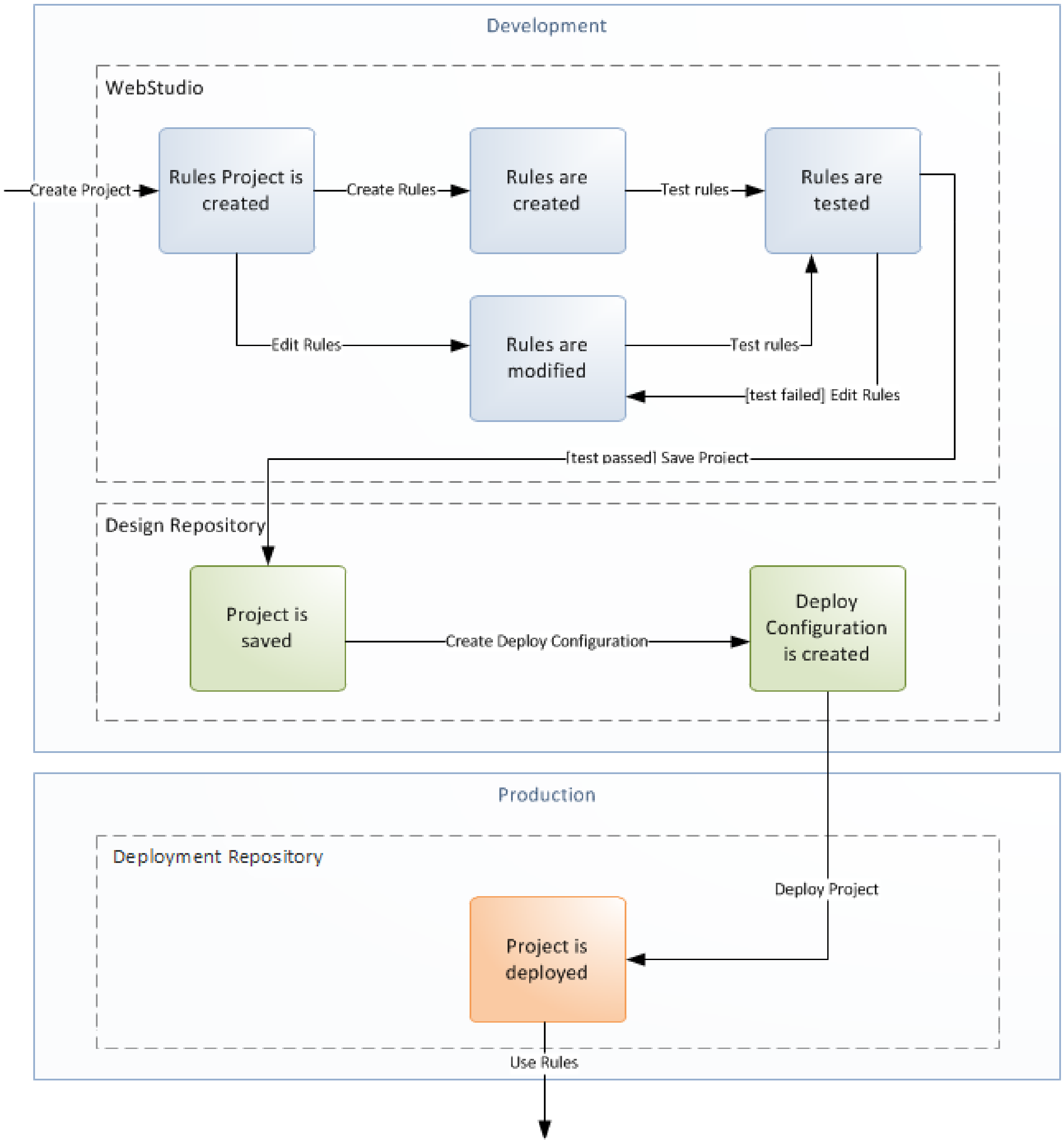



*Figure 1: Rules project development workflow*

Development of rules starts with creating a new project that will contain the rules. If the project already exists, it must be opened for editing. Then rules are created or updated and properly tested. After rules are completed and all tests are passed, a user saves the project. At this point, the updated revision of the project is saved to

Design repository and applied changes of the project become available for viewing and editing by other users. If no more changes to the project are planned in the nearest future, a user can close the project.

Saved project revision is used to create deploy configuration. Several projects can be included in the same deploy configuration. Deploy configurations are used to deploy updates to the production environment.

If the project is not required any more, it can be deleted.

# 2.3 OpenL Tablets WebStudio Components

OpenL Tablets WebStudio consists of the following main components:

| **OpenL Tablets WebStudio components** | |
|---|---|
| **Component** | **Description** |
| Rules Editor | Graphic user interface running in a web browser allowing users to browse rule modules, modify table data, and run tests. Rule project configurations are browsed and updated there as well.<br>Rules Editor is the default user interface displayed when a user opens OpenL Tablets WebStudio.<br>Rules Editor does not display all rule module files but provides a logical view of rules stored in a module. This view is convenient for users who modify business rules.<br>Rules Editor displays only modules available in projects stored in the user's workspace.<br>To retrieve a project to the user's workspace, open the project as described in Working with Projects in OpenL Tablets WebStudio.<br>For more information on using Rules Editor, see Using Rules Editor. |
| Repository editor | Graphic user interface running in a web browser allowing users to browse and manage projects in Design repository.<br>Unlike Rules Editor, repository editor displays physical contents of rule projects.<br>Users can easily switch between Rules Editor and repository editor in user interface.<br>Repository editor provides the following main functions:<br>• uploading projects from the file system to Design repository<br>• editing, saving, opening, and closing projects<br>• modifying project structure and properties<br>• managing project revisions<br>• copying and deleting projects in Design repository<br>• managing and tracing deploy configurations<br>For more information on using repository editor, see Using Repository Editor. |
| Design repository | Centralized storage of rule projects accessible by all OpenL Tablets WebStudio users. Projects uploaded to Design repository are visible to other users.<br>Design repository creates a separate project revision each time a project is saved. Any project revision can be opened. |
| Deploy configurations repository | Centralized storage of final rule projects to be delivered to the production environment where solution applications use them.<br>Projects can be deployed to deployment repository from Design repository using deploy configurations. **Deploy configuration** is a specific OpenL Tablets WebStudio project type. It identifies rule projects and project revisions to be deployed to deployment repository. Deploy configurations are saved and versioned so that developers can identify which specific rule project revisions are deployed. |
| Deployment repositories | Production storages of deployed rule projects where solution applications use them. |

| **OpenL Tablets WebStudio components** | |
|---|---|
| **Component** | **Description** |
| User workspace | Project storage on the server containing projects edited by users. Each user has a personal workspace unavailable to other users. |

## 2.4 Security Overview

OpenL Tablets WebStudio supports the following user modes:

| **OpenL Tablets WebStudio user modes** | |
|---|---|
| **Mode** | **Description** |
| Demo mode | This is a multi user mode with the list of users predefined in the database. All changes in the database will be lost after the application restart. The User's projects will be located in the user-workspace\<user name> folder. |
| Single user mode | In this mode, only one user who is currently logged in on the computer can work in OpenL Tablets WebStudio.<br>This mode is selected when OpenL Tablets WebStudio is installed on the local machine. All user projects are located in the root of the user-workspace directory. Single user mode is set by default and does not require additional settings, including logon to the system. Moreover, the system works faster in this mode but neither user management nor access control is provided. |
| Multi user mode | This mode enables multiple users to work in OpenL Tablets WebStudio and supports a security mechanism restricting access to certain product functions based on user access rights.<br>Each OpenL Tablets WebStudio user is identified by a unique name. When a user opens OpenL Tablets WebStudio in a web browser, he or she must log into the system.<br>Users can have varied levels of access in OpenL Tablets WebStudio. For example, system administrators usually have full access to all OpenL Tablets WebStudio functions, whereas other users may only have access rights to view or modify business rules.<br>WebStudio is used to authenticate and manage user credentials/permissions.<br>In this mode, user's projects are located in the user-workspace\<user name> directory. |
| Active Directory | In this mode, multiple users can run OpenL Tablets WebStudio using their unique user names.<br>The User's projects will be located in the user-workspace\<user name> directory.<br>Active Directory will be used to authenticate and manage user credentials.<br>A place where user permissions will be managed can be WebStudio or Active Directory |
| SSO: CAS | In this mode, multiple users can run OpenL Tablets WebStudio using their unique user names.<br>The User's projects will be located in the user-workspace\<user name> directory.<br>CAS (Central Authentication Service) server will be used to authenticate and manage user credentials.<br>A place where user permissions will be managed can be WebStudio or SSO Identity provider |
| SSO: SAML | In this mode, multiple users can run OpenL Tablets WebStudio using their unique user names.<br>The User's projects will be located in the user-workspace\<user name> directory.<br>SAML (Security Assertion Markup Language) supporting Identity Provider server will be used to authenticate and manage user credentials.<br>A place where user permissions will be managed can be WebStudio or SSO Identity provider |

# 3 Getting Started

This chapter explains logging into OpenL Tablets WebStudio and briefly introduces the user interface. The following topics are included in this chapter:

- Signing In to OpenL Tablets WebStudio
- Modifying User Profile
- Displaying the OpenL Tablets WebStudio Help
- Signing Out of OpenL Tablets WebStudio
- Introducing Rules Editor
- Introducing Repository Editor

## 3.1 Signing In to OpenL Tablets WebStudio

To sign in to OpenL Tablets WebStudio, proceed as follows:

1. In the web browser address bar, enter the OpenL Tablets WebStudio URL provided by the system administrator.

   The OpenL Tablets WebStudio URL has the following pattern:

   ```
   http://<server>:<port>/webstudio
   ```

   In the single user mode, users are automatically signed in using the DEFAULT account. In the multi-user mode, the following form appears.

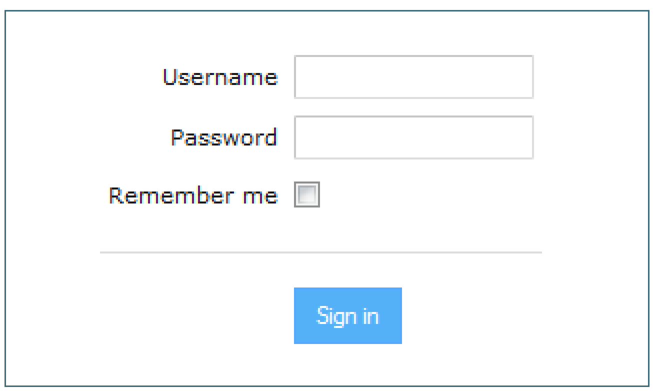



*Figure 2: Login window*

2. Enter the user name and password provided by the system administrator and click **Sign in**.

For more information on OpenL Tablets WebStudio UI, see Introducing Rules Editor and Introducing Repository Editor. For more information on the single and multi-user modes, see Security Overview.

## 3.2 Modifying User Profile

OpenL Tablets WebStudio provides a drop-down dialog located in the top-right corner of the application, under the user name, for updating user profile information, changing the password, and editing user settings. All data is stored in the user profile and includes **User details** and **User settings** sections.


*Figure 3: Opening the user profile window*

This section describes how to modify user profile information and includes the following topics:

- Modifying User Details
- Modifying User Settings

## Modifying User Details

To manage user details, proceed as follows:

1. In OpenL Tablets WebStudio, in the top-right corner of the window, click an arrow icon next to the username.
2. In the actions list, click **User Details**.

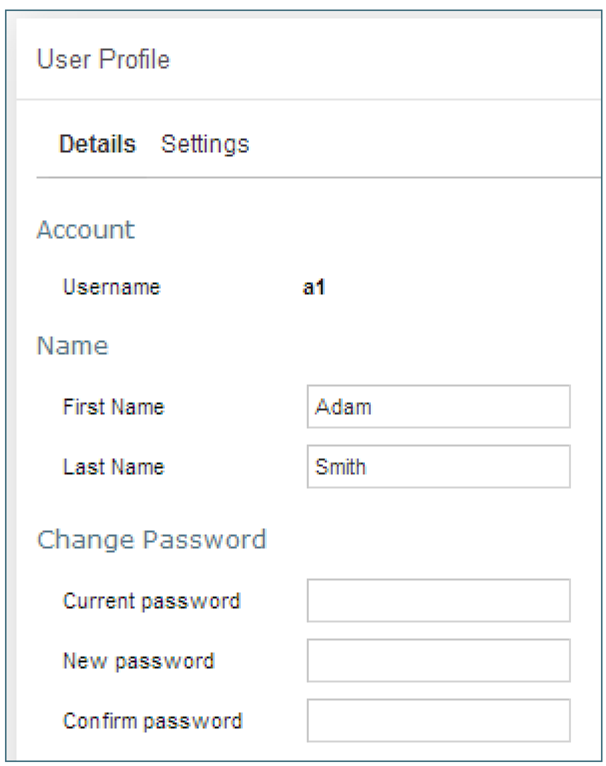



*Figure 4: Viewing user details*

3. To update user's first or last name, in the **Name** section, modify values as required.

4. To update the password, in the **Change Password** section, enter the current and new password values.

## Modifying User Settings

To manage user settings, proceed as follows:

1. In OpenL Tablets WebStudio, in the top-right corner of the window, click an arrow icon next to the username.
2. In the actions list, click **User settings**.

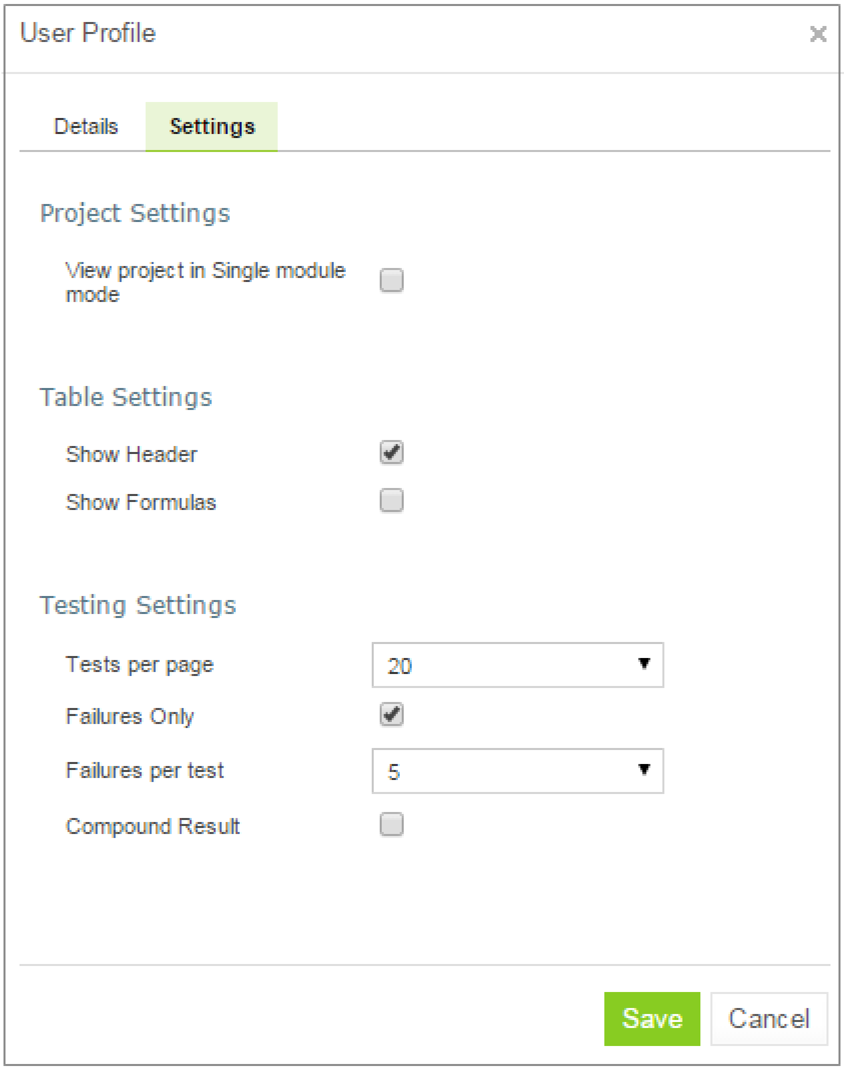



*Figure 5: Viewing user settings*

3. To enable opening the project in the single module mode, in the **Project Settings** section, select the **Open project…** check box.

   For more information on module opening modes, see Viewing a Module.
4. In the **Table Settings** section, identify whether table header and MS Excel formulas must be displayed.
5. In the **Testing settings**, select values for displaying rule test results.

   By default, all test results are displayed with five test tables, or unit tests, and compound result is not displayed. For more information on testing settings, see Running Unit Tests.

# 3.3 Displaying the OpenL Tablets WebStudio Help

To display the OpenL Tablets WebStudio help topics, in OpenL Tablets WebStudio, in the top-right corner of the window, click an arrow icon next to the username and select **Help**.

# 3.4 Signing Out of OpenL Tablets WebStudio

To sign out of OpenL Tablets WebStudio, proceed as follows:

1. In OpenL Tablets WebStudio, in the top-right corner of the window, click an arrow icon next to the username.
2. In the actions list, click **Sign out**.

# 3.5 Introducing Rules Editor

This section briefly introduces Rules Editor and includes the following topics:

- Rules Editor Overview
- View Modes

For more information on tasks that can be performed in Rules Editor, see Using Rules Editor.

## Rules Editor Overview

**Rules Editor** enables users to browse rule modules and modify table data. A default editor is displayed when a user opens a table in a module.

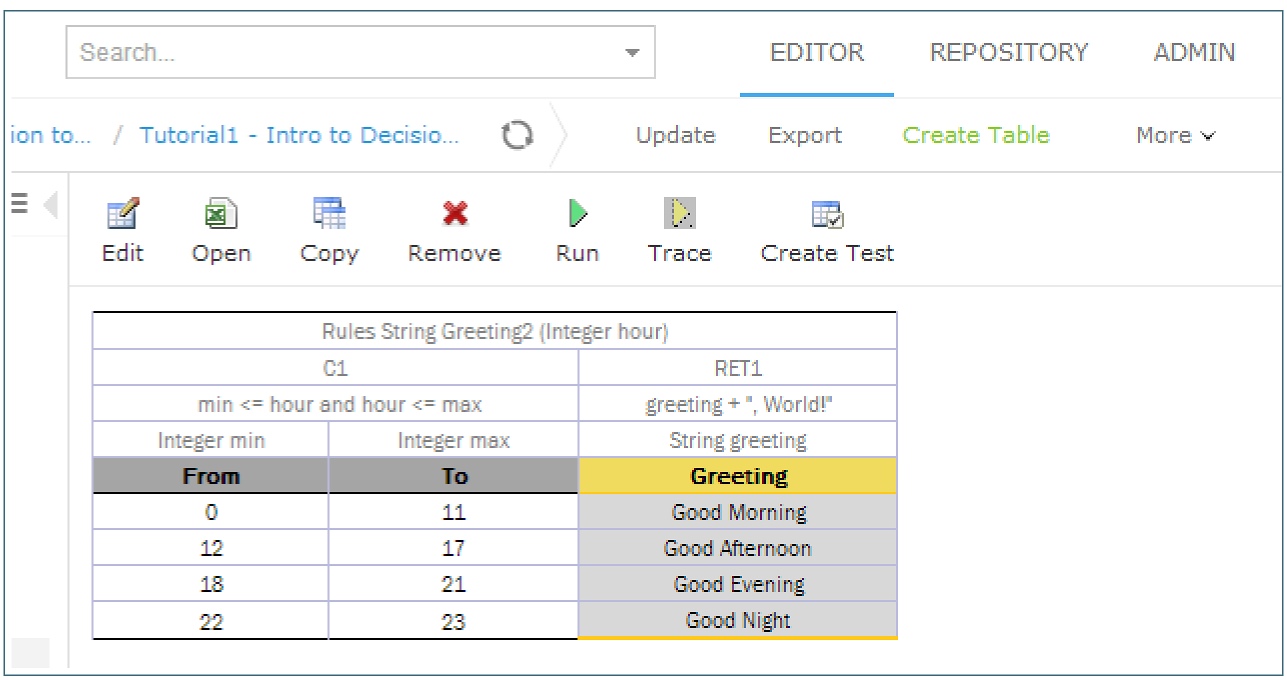



*Figure 6: OpenL Tablets WebStudio Rules Editor*

Rules Editor displays one module at a time. To switch between modules, select a module in the **Projects** tree or use breadcrumb navigation for quick switching between projects or modules of the current project.

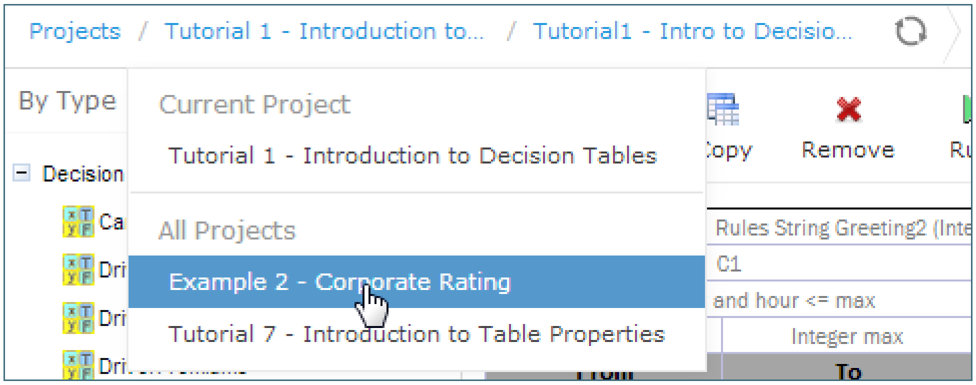


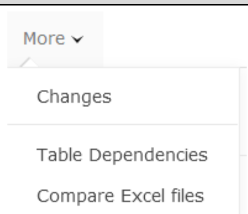
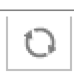

*Figure 7: Rules Editor breadcrumb navigation*

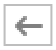
One rule project can contain several modules.

The following table describes Rules Editor organization:

| **Rules Editor organization** | |
|---|---|
| **Pane** | **Description** |
| Left pane | Displays the module tree providing a list of elements in the currently displayed rule module. |
| Middle pane | Displays contents of the table selected in the left pane and provides controls for modifying table data, running tests, and checking test results. |
| Right pane | Displays properties of the currently displayed table. |
| Upper part of the window | Contains toolbars with controls as described further in this section. |

The following table describes the Rules Editor toolbar controls:

<table>
<tr><th colspan="2">Rules Editor toolbar controls</th></tr>
<tr><th>Control</th><th>Description</th></tr>
<tr><td></td><td>The following table describes the available options:
<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>Changes</td><td>Opens a page for reverting module changes.</td></tr>
<tr><td>Table Dependencies</td><td>Opens a graph displaying dependencies among tables of the module.</td></tr>
<tr><td>Compare Excel files</td><td>Initiates a dialog for comparing Excel files.</td></tr>
</table></td></tr>
<tr><td></td><td>Runs a simple search.<br>For more information on performing searches, see Performing a Search.</td></tr>
<tr><td></td><td>Switches the opening mode for a current module.<br>For more information on module opening modes, see Viewing a Module.</td></tr>
<tr><td></td><td>Refreshes OpenL Tablets WebStudio with the latest changes in Excel files.</td></tr>
<tr><td></td><td>Initiates the table creation wizard.</td></tr>
<tr><td></td><td>Displays recently viewed tables instead of the module tree.</td></tr>
</table>

| Rules Editor toolbar controls | |
|---|---|
| **Control** | **Description** |
|  | Returns to the module tree view. |
|  | Hides comment tables and dispatcher tables generated automatically when a rule table is overloaded by business dimension property. |
| Save | Sets the project status to **No Changes**. |
| Update Export | Updates the current module or project with uploaded file or zip file. Exports the current version of the module or project. |
| REPOSITORY | Switches user interface to repository editor.<br>For more information on repository editor, see Introducing Repository Editor. |
| EDITOR | Switches user interface to Rules Editor.<br>For more information on Rules Editor, see Using Rules Editor. |
| ADMIN | Switches user interface to the **Administration** mode.<br>For more information on administrative functions, see Using Administration Tools. |

# View Modes

OpenL Tablets WebStudio provides different modes for displaying rule elements. In this guide, modes are contingently divided into a **simple view** and **extended view**.

When a table is opened in a simple view, OpenL Tablets WebStudio hides various technical table details, such as table header and MS Excel formulas. An example of a table opened in a simple view is as follows.

| Vehicle Age | Premium Increase |
|---|---|
| <1 | $400 |
| 1-4 | $300 |
| 5-10 | $250 |
| | $0 |

*Figure 8: A rule table in a simple view*

In the extended view, all table structure is displayed. An example of a table opened in an extended view is as follows.

| SimpleRules DoubleValue AgeSurcharge (Integer vehicleAge) | |
|---|---|
| **Vehicle Age** | **Premium Increase** |
| <1 | $400 |
| 1-4 | $300 |
| 5-10 | $250 |
| | $0 |

*Figure 9: A rule table in an extended view*

To switch between views, use the **Show Header** and **Show Formula** options in **User settings**.

Rule tables can be organized, or sorted, and displayed in the module tree in different way depending on the selected value.

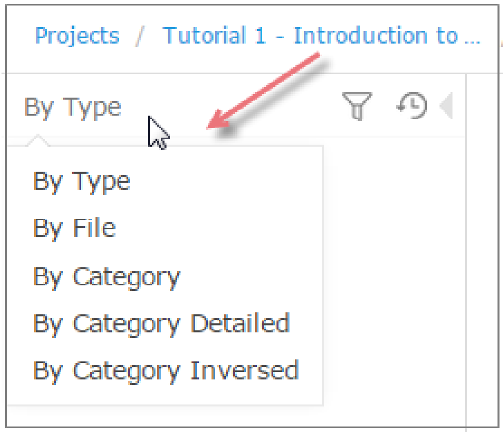
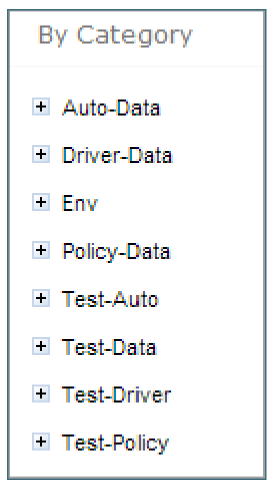



*Figure 10: Modes for sorting tables in the module tree*

By default, tables are sorted by type.

| Modes for organizing tables in Rules Editor | |
|---|---|
| **Mode** | **Description** |
| **By Category** | The tree structure is rather logical than physical. Rule tables are organized into categories based on the **Category** table property or, if the property is not defined, based on the Excel table sheet names. This view is **simple.**<br>An example of a module tree sorted by the category parameter is as follows:<br><br><br>*Figure 11: Module tree sorted by category* |

<table>
<tr><th colspan="2">Modes for organizing tables in Rules Editor</th></tr>
<tr><th>Mode</th><th>Description</th></tr>
<tr><td>By Category Detailed</td><td>The By Category Detailed view displays modules sorted by the first value of the Category property. In the following example, the same module tree is sorted by Category Detailed and, for example, Test > Auto category is displayed in the Test node and Auto sub-node:

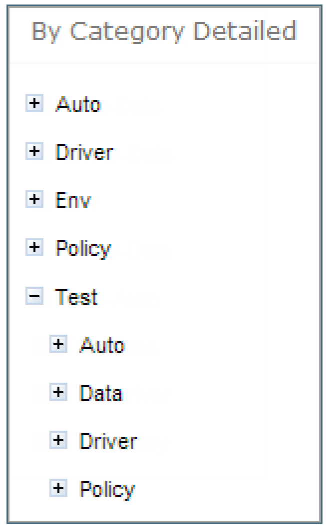



Figure 12: Module tree sorted by Category Detailed</td></tr>
<tr><td>By Category Inversed</td><td>The following example provides the module tree sorted by Category Inversed where the modules are sorted by the second value of the Category property:

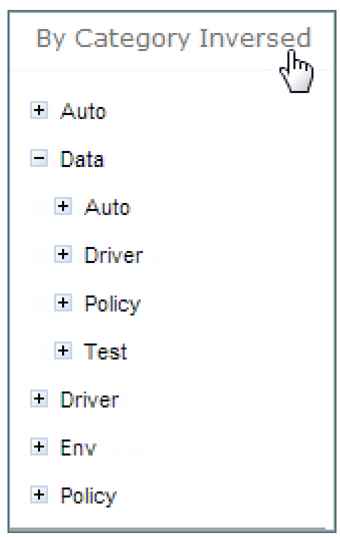
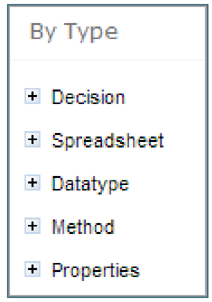
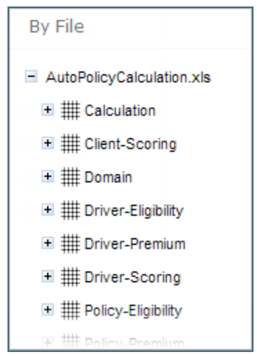



Figure 13: Module tree sorted by Category Inversed</td></tr>
</table>

**Note:** If the scope in a **Properties** table is defined as **Module**, in the **By Category** view, this table is displayed in the **Module Properties** sub-node as in the last example. If the scope is defined as **Category**, the table is displayed in the Category **Properties** sub-node.

The two following modes display a project in a way convenient to experienced users, with module tree elements organized by physical structure rather than logically, in an **extended** view.

| Modes for organizing tables in Rules Editor in extended view | |
|---|---|
| **Mode** | **Description** |
| **By Type** | An example of a module tree displayed in extended view and sorted by type is as follows:<br><br><br>*Figure 14: Module tree sorted by type* |
| **By File** | The following module tree is sorted by file and by spreadsheets within the file:<br><br><br>*Figure 15: Module tree sorted by file* |

# 3.6 Introducing Repository Editor

**Repository editor** provides controls for browsing and managing Design repository. A user can switch to repository editor by clicking the **Repository** control. Repository editor resembles the following:

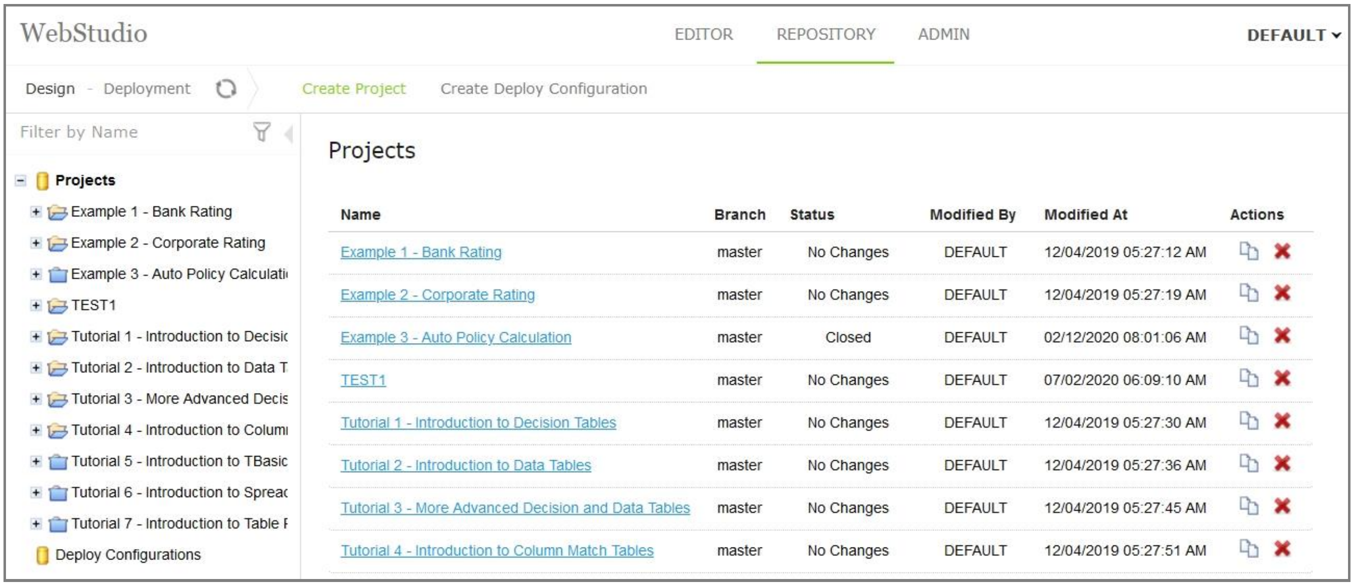



*Figure 16: OpenL Tablets WebStudio repository editor*

The following table describes repository editor organization:

| **Repository editor organization** | |
|---|---|
| **Pane** | **Description** |
| Left pane | Contains a tree of projects stored in Design repository and user's workspace. Unlike Rules Editor, repository editor displays physical project contents in terms of files and folders. |
| Middle pane | Displays content for the element selected in the tree. |

A user can switch to Rules Editor by clicking the **Rules Editor** control.

For more information on tasks that can be performed in repository editor, see Using Repository Editor.

# 4 Using Rules Editor

This chapter describes basic tasks that can be performed in Rules Editor. For more information on Rules Editor, see Introducing Rules Editor.

The following topics are included in this chapter:

- Filtering Projects
- Viewing a Project
- Viewing a Module
- Managing Projects and Modules
- Defining Project Dependencies
- Viewing Tables
- Modifying Tables
- Referring to Tables
- Managing Range Data Types
- Creating Tables by Copying
- Performing a Search
- Creating Tables
- Comparing Excel Files

## 4.1 Filtering Projects

To limit a list of projects displayed in the **Projects** list, start typing a project name in the field located above the list of projects.

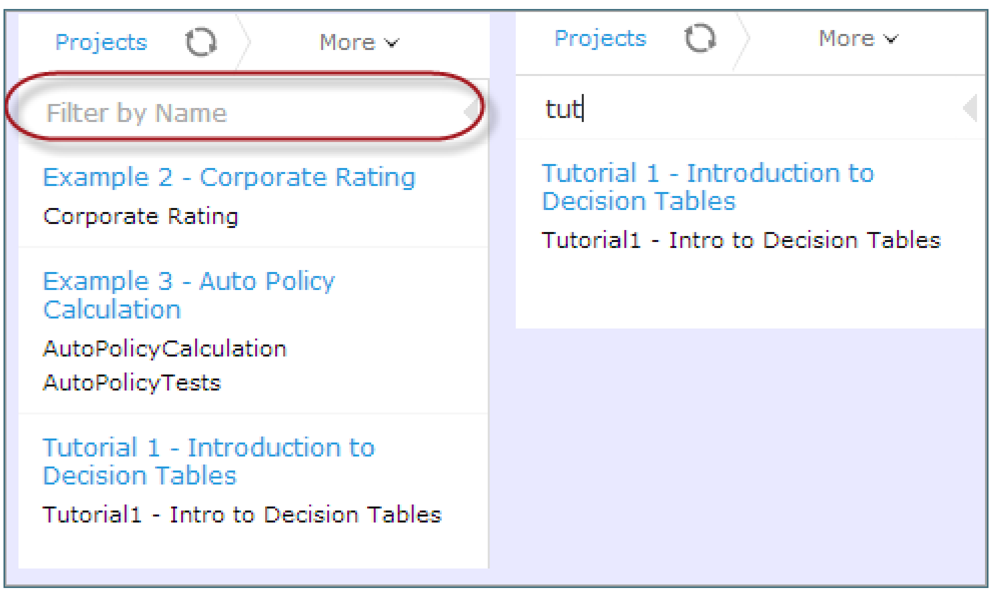



*Figure 17: Filtering projects by Name*

To get a full list of projects, delete filter text value in the field.

## 4.2 Viewing a Project

Rules Editor allows a user to work with one project at a time. To select a project, in the **Projects** tree, select the blue hyperlink of the required project name. The project page with general information about the project and configuration details appears in the middle pane of the editor.


*Figure 18: A project page in Rules Editor*

If a particular project is not available, it must be opened as described in Opening a Project.

## 4.3 Viewing a Module

Rules Editor allows a user to work with one module at a time. To select a module, in the **Projects** tree, select the black hyperlink of the module name. The following module information is displayed:

- tree in the left pane displaying module tables
- general module information displayed in the middle pane, including project and module names, associated Excel file, number of tables, and module dependencies

If a particular module is not available, the project in which it is defined must be opened as described in Opening a Project.

To support convenient work with dependencies, the following modes for opening and viewing a module are available in OpenL Tablets WebStudio:

| **OpenL Tablets WebStudio mode opening and viewing modes** | |
|---|---|
| **Mode** | **Description** |
| Single-module mode | Displays the module considering only the module dependencies defined in the **Environment** table of the module and skipping any other modules of the current project and project dependencies. That is, the **All Modules** option is ignored. |
| Multi-module mode | Displays all modules of the current project with all their dependencies, that is, modules of projects defined as the project dependencies. In other words, the whole project with its infrastructure is opened.<br>Note that the module tree in the left pane displays the tables of a current module only, but, actually, tables of other project modules and project dependency modules can be accessed from any rule or test of the current module as well. |

For more information on project and module dependencies, see the *Project and Module dependencies* section in ***[OpenL Tablets Reference Guide]***.

By default, modules of a project are opened in the multi-module mode. This is a common production mode.

To open a single module without complete project infrastructure, such as other project modules and project dependencies, to simplify or speed up rules development, for instance, change the project opening settings for each user individually in the user profile by selecting the **Open project in Single module mode** check box as described in Introducing Rules Editor.

To change the mode for a currently viewed module without updating user settings, in the top line menu, in the module mode drop-down list, select the required mode.

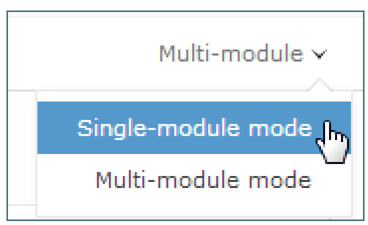



*Figure 19: Modes of opening and viewing a module in OpenL Tablets WebStudio*

# 4.4 Managing Projects and Modules

This section explains the following tasks that can be performed on projects in Rules Editor:

- Editing and Saving a Project
- Saving a Project for Backward Compatibility
- Updating and Exporting a Project
- Copying a Project
- Exporting, Updating, and Editing a Module
- Comparing and Reverting Module Changes
- Copying a Module

# Editing and Saving a Project

A project can be opened for editing and saved directly in Rules Editor. To save the edited project, click **Save** Save.

**Note:** If a project is in the **Local** status, this option is not available in Rules Editor.

1. To modify the project in the **Project** page, modify the values as described in the following table:

| **Editable project settings** | |
|---|---|
| **Project details** | **Available actions** |
| General project information and configuration, such as OpenL version compatibility, description, and custom file name processor | Put the mouse cursor over the project name and click **Edit** .<br>For more information on OpenL version compatibility, see Saving a Project for Backward Compatibility.<br>For more information on properties pattern for the file name, see the **Properties from File Name** section of ***[OpenL Tablets Reference Guide]***. |
| Project sources | Put the mouse cursor over the **Sources** label and click **Manage Sources** . |
| Modules configuration | Put the mouse cursor over the **Modules** label or a particular module name and click **Add Module** or **Edit Module** or **Remove Module** . |
| Project dependencies | Manage dependencies as described in Defining Project Dependencies. |

All changes are saved in the project `rules.xml` file. For more information on this XML file, see the ***[OpenL Tablets Developer's Guide]***.

# Saving a Project for Backward Compatibility

For backward compatibility, a project can be saved in earlier OpenL versions, for example, 5.11.0 or 5.12.0.

It is important that the structure of `rules.xml` and `rules-deploy.xml` is changed after saving a project in a previous OpenL version, and may result, for example, in disappeared UI fragments.

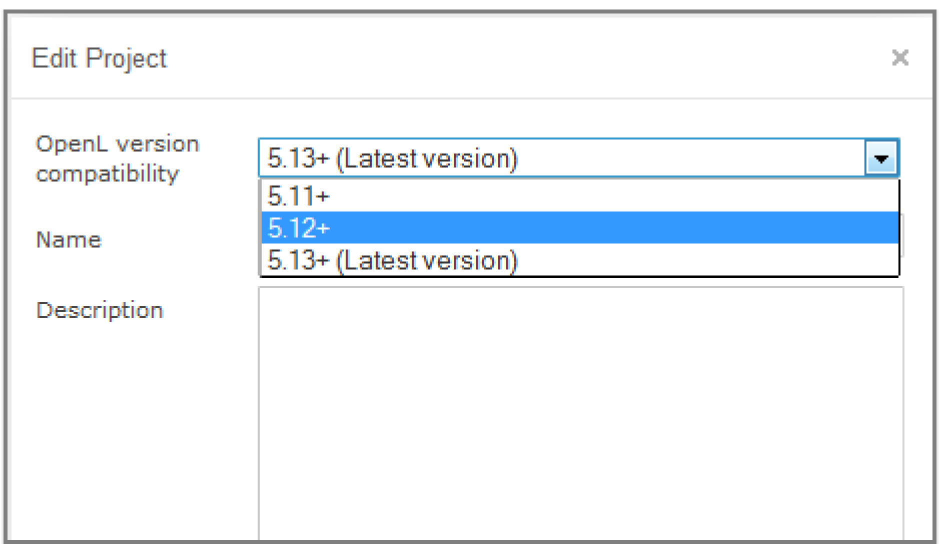



*Figure 20: Selecting an OpenL version for creating a backward compatible project version*

# Updating and Exporting a Project

To update or export a project, proceed as follows:

1. To update a project directly in Rules Editor, in the top line menu, click **Update** and make the necessary changes.
   The **Update** button is available for projects in the **In Editing** status.
2. To export the project to the user's local machine, for a project, in the top line menu, click **Export**.
   Exported project is downloaded as a `.zip` archive.

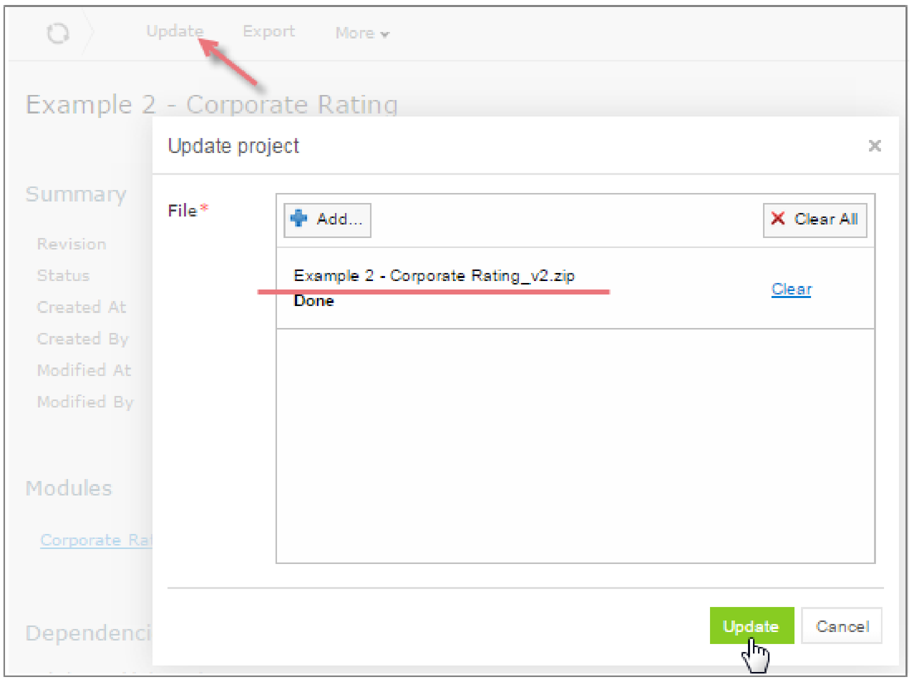



*Figure 21: Importing and updating the project from a .zip file*

# Copying a Project

To create a copy of a project, proceed as follows:

1. In the top line menu, click **Copy**.
2. In the window that appears, enter the new project name.
3. Modify **Comment** if needed.
4. When you need to copy a project with revisions, select the **Copy old revisions** check box and provide the desired number of revisions in the **Revisions to copy** field.
5. Click **Copy.**

   The new project appears in the project list.

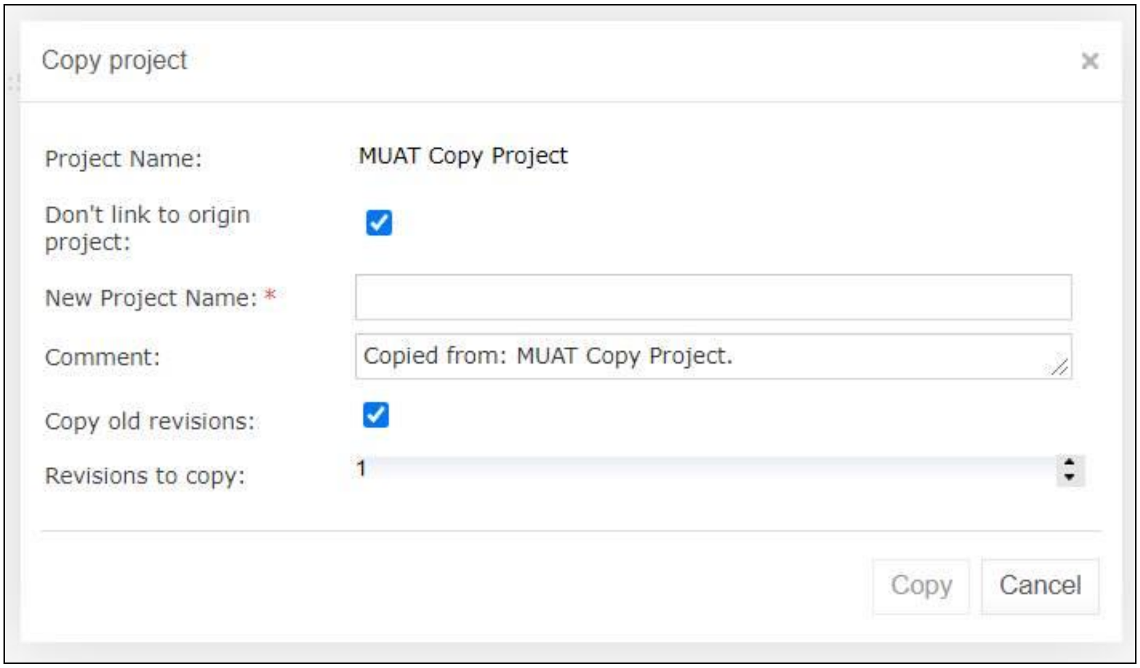



*Figure 22: Copying project*

## Exporting, Updating, and Editing a Module

A user can export, update, or edit a module directly in Rules Editor. Proceed as follows:

1. To upload a changed module file, for a module, in the top line menu, click **Upload**.
2. To export the module to the user's local machine, for a module, in the top line menu, click **Export**.
3. To modify module configuration, such as module name, path, and included or excluded methods, in the **Module** page place the mouse cursor over the module name and click **Edit** .

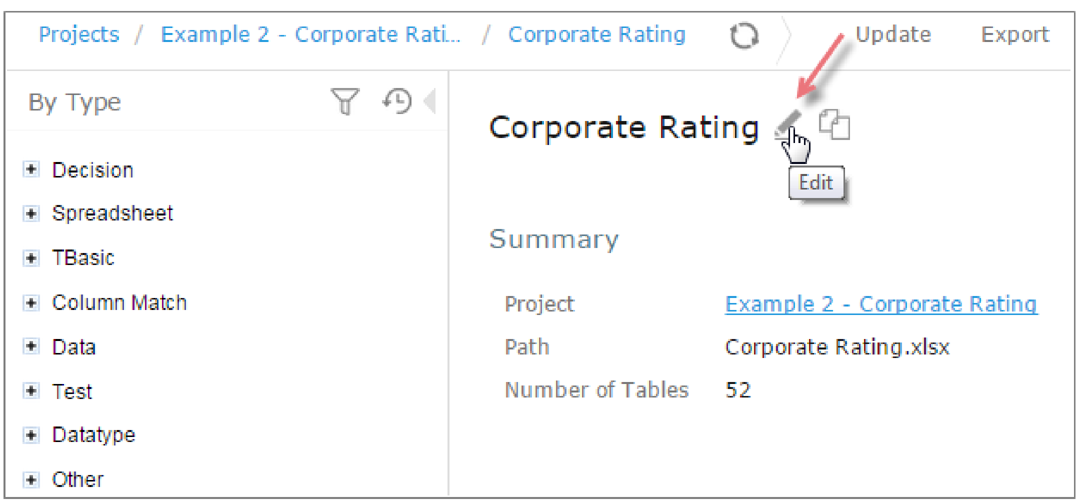



*Figure 23: Initiating module editing*

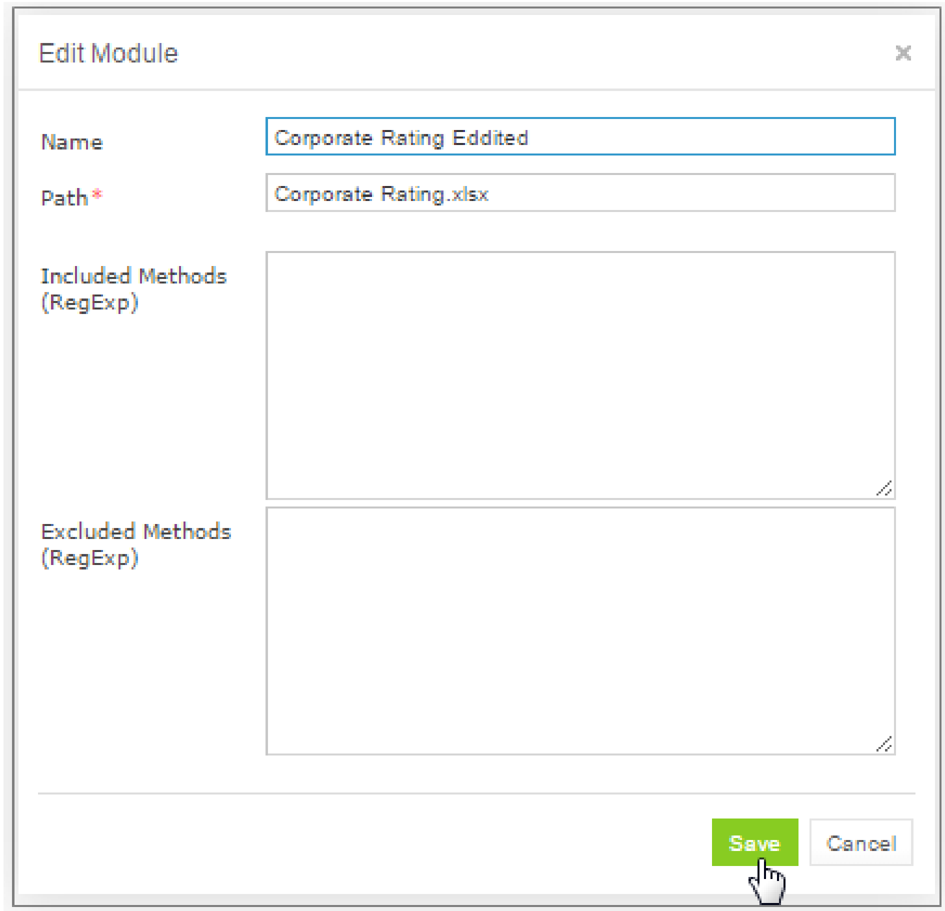



*Figure 24: Editing module information*

4. To save the changes, click **Save** 

.

## Comparing and Reverting Module Changes

OpenL Tablets WebStudio allows comparing module versions and rolling back module changes against the specific date.

To compare module versions, proceed as follows:

1. In the **Projects** tree, select the module.
2. In the top line menu, select **More > Changes**.

   The **Changes** page appears displaying all module versions, with the latest versions on the top.

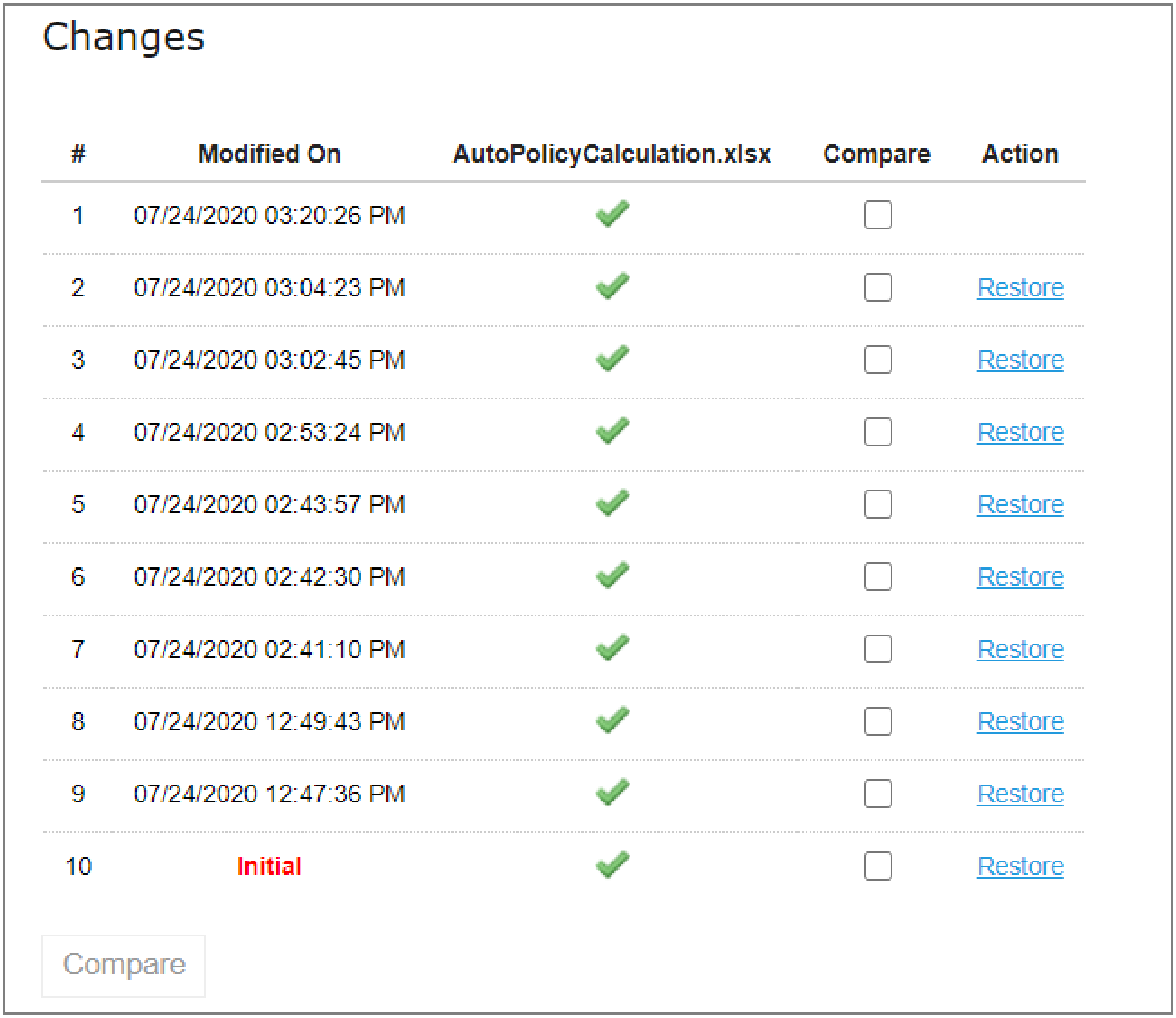
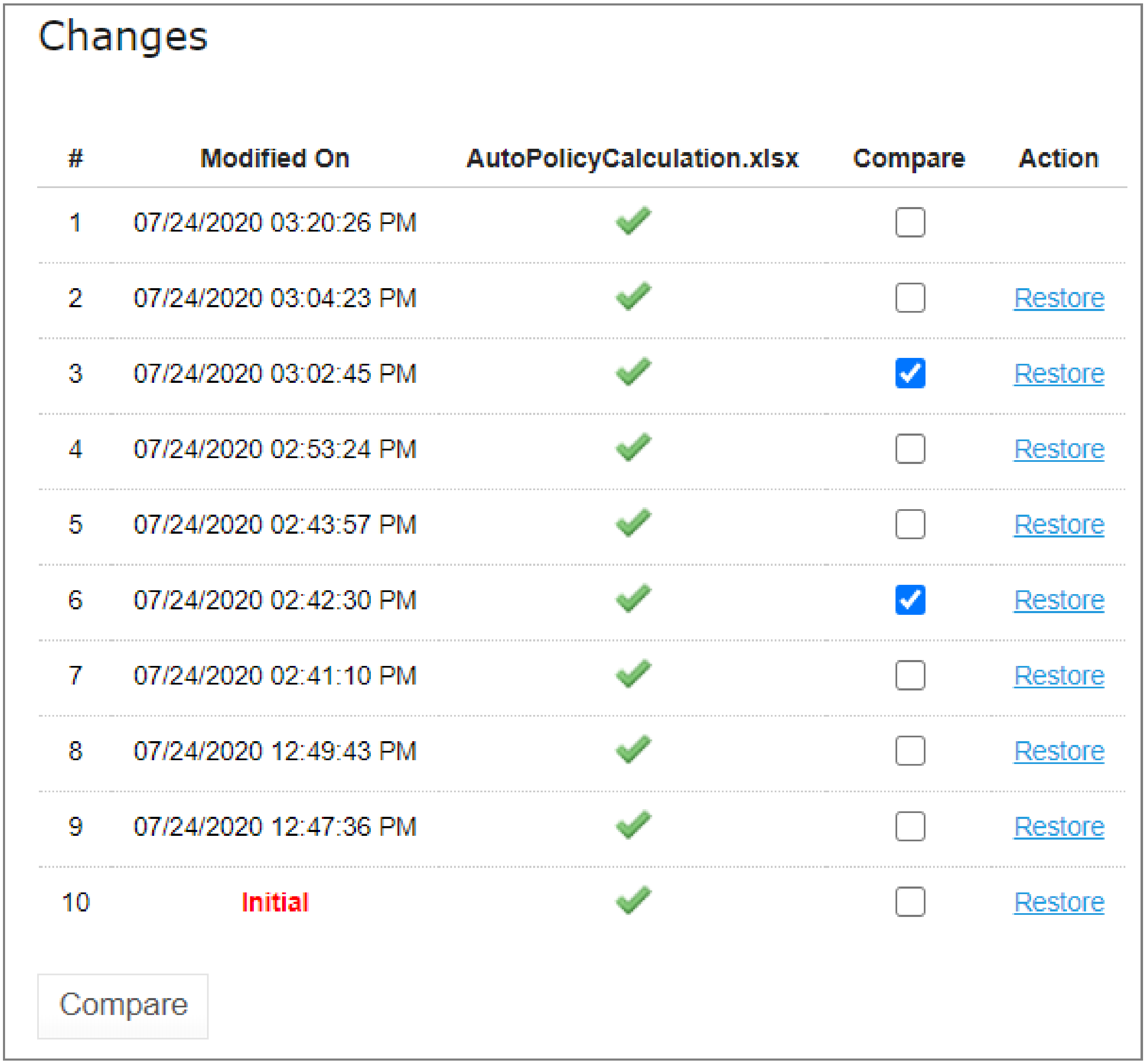



*Figure 25: Displaying the Changes window*

3. To compare the changes, select check boxes for the required versions and click **Compare**.



Changes

| # | Modified On | AutoPolicyCalculation.xlsx | Compare | Action |
|---|---|---|---|---|
| 1 | 07/24/2020 03:20:26 PM | ✔ | ☐ | |
| 2 | 07/24/2020 03:04:23 PM | ✔ | ☐ | Restore |
| 3 | 07/24/2020 03:02:45 PM | ✔ | ☑ | Restore |
| 4 | 07/24/2020 02:53:24 PM | ✔ | ☐ | Restore |
| 5 | 07/24/2020 02:43:57 PM | ✔ | ☐ | Restore |
| 6 | 07/24/2020 02:42:30 PM | ✔ | ☑ | Restore |
| 7 | 07/24/2020 02:41:10 PM | ✔ | ☐ | Restore |
| 8 | 07/24/2020 12:49:43 PM | ✔ | ☐ | Restore |
| 9 | 07/24/2020 12:47:36 PM | ✔ | ☐ | Restore |
| 10 | Initial | ✔ | ☐ | Restore |

Compare

*Figure 26: Comparing module versions*

The system displays the module in a separate browser window where changed tables are marked as displayed in the following example.


*Figure 27: Tables with changes*

4. To view the changes, click the required table.

The result of the comparison is displayed in the bottom of the window.

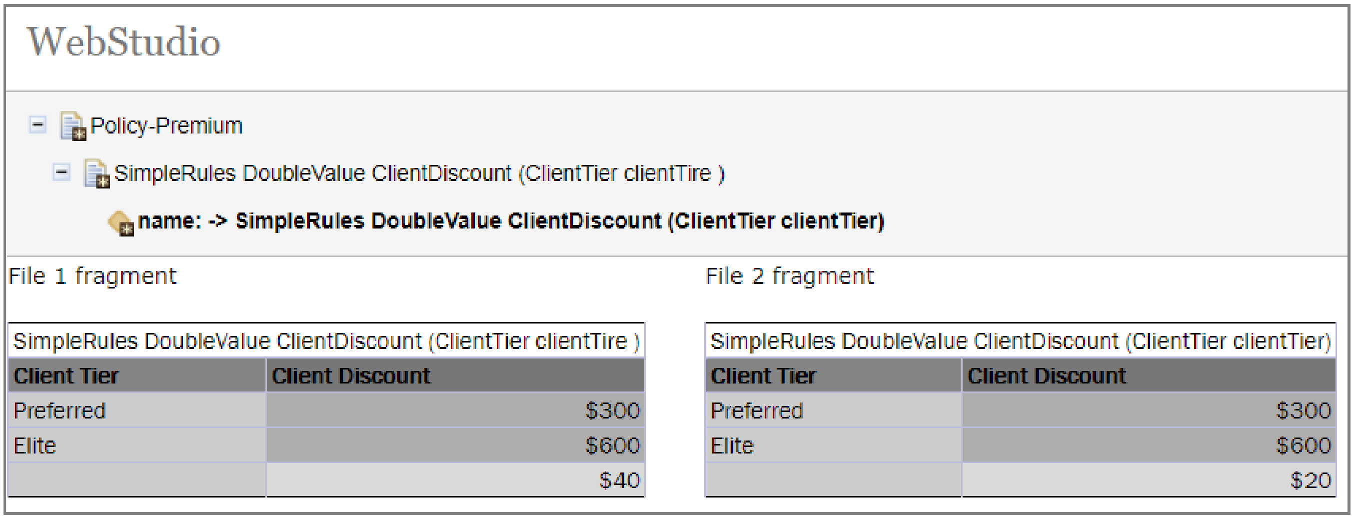



*Figure 28: The result of the module version comparison*

5. To revert module changes, for the required module version, click the **Restore** link and confirm the changes.

# Copying a Module

OpenL Tablets WebStudio allows creating a copy of the existing module, in Editor, in either **Project** page, or in the **Module** page. The following topics are included in this section:

- Copying a Simple Module
- Copying a Module Defined Using the File Path Pattern

## Copying a Simple Module

To create a copy of a module, proceed as follows:

1. Do one of the following:
   - To create a copy of a module using the **Project** page, in the project tree, select a project which module must be copied, in the modules list, put the mouse cursor over the selected module name, and click **Copy Module** .
   - To create a copy of a module using the **Module** page, in the project tree, select a module to be copied, put the mouse cursor over the module name, and click **Copy Module** .
2. In the window that appears, enter the new module name.

   When the new module name is entered, the **Copy** button becomes enabled.
3. Optionally, edit the **New File Name** field value.

   The file name can differ from the module name.
4. Optionally, to copy the module to the specific folder, in the **New File Name** field, enter the file name and its location.

   The original path cannot be modified other than by entering the specific path in the **New File Name** field. For example, if the original module is located in `folder1`, the new module will be copied to `folder1`. `Folder1` cannot be changed, but a user can define a new file name, such as `folder2/Bank Rating ver2.xlsx`, and then the new module will be created in `folder1/folder2/Bank Rating ver2.xlsx`.

5. Click **Copy**.
   A new simple module is displayed in the modules list.

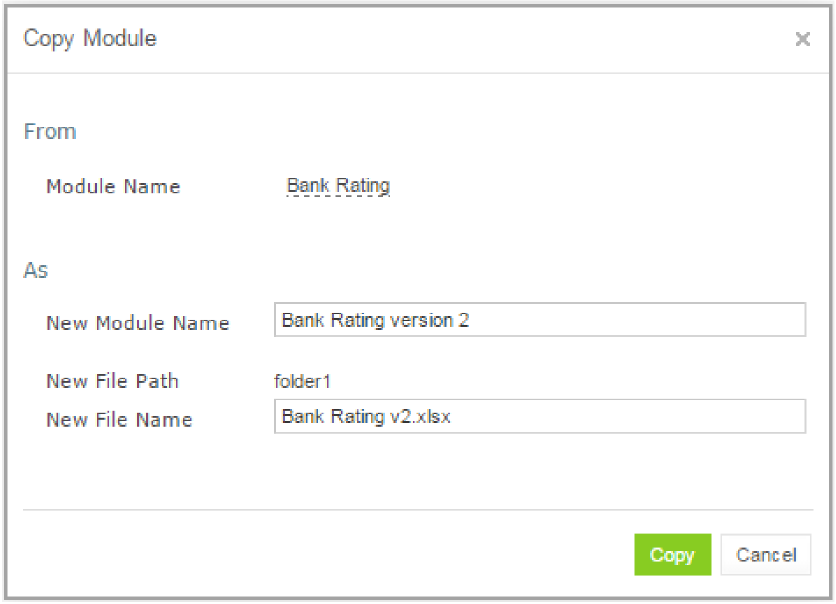



*Figure 29: Creating a copy of a module*

## Copying a Module Defined Using the File Path Pattern

If the module is defined using **File Path Pattern**, to copy such module, proceed as follows:

1. Do one of the following:
   - To create a copy of a module using the **Project** page, put the mouse cursor over multiple modules, click **Copy Module** , in the window that appears, click **Select module,** and in the **File Path** drop-down list, select the name of the module to copy.
   - To create a copy of a module using the **Module** page, in the project tree, select a module to copy, put the mouse cursor over the module name, and click **Copy Module** .
2. Click **Select module** and in the **File Path** drop-down list, select the name of the module to copy.
3. Enter the new module name.
4. Click **Copy**.
   The new module is displayed in the modules list.

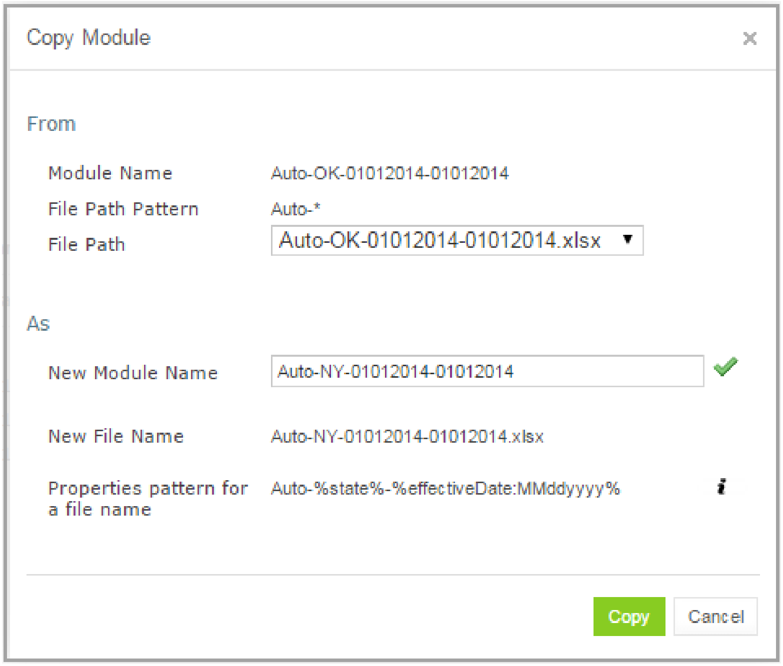



*Figure 30: Copying a module with the defined file path and properties patterns*

If the new module name does not match the properties pattern for the file name, no business dimension properties will be applied to the rules inside the module.

# 4.5 Defining Project Dependencies

A project dependency can be defined when a particular rule project, or **root project**, depends on contents of another project, or **dependency project**. Project dependencies are checked when projects are deployed to the deployment repository. OpenL Tablets WebStudio displays warning messages when a user deploys projects with conflicting dependencies.

To define a dependency on another project, proceed as follows:

1. In Rules Editor, in the project tree, select a project name.
2. If the project is not editable, make it editable as described in Editing and Saving a Project.
3. Put the mouse cursor over the **Dependencies** label and click **Manage Dependencies** .
4. In the window that appears, update information as required and click **Save**.

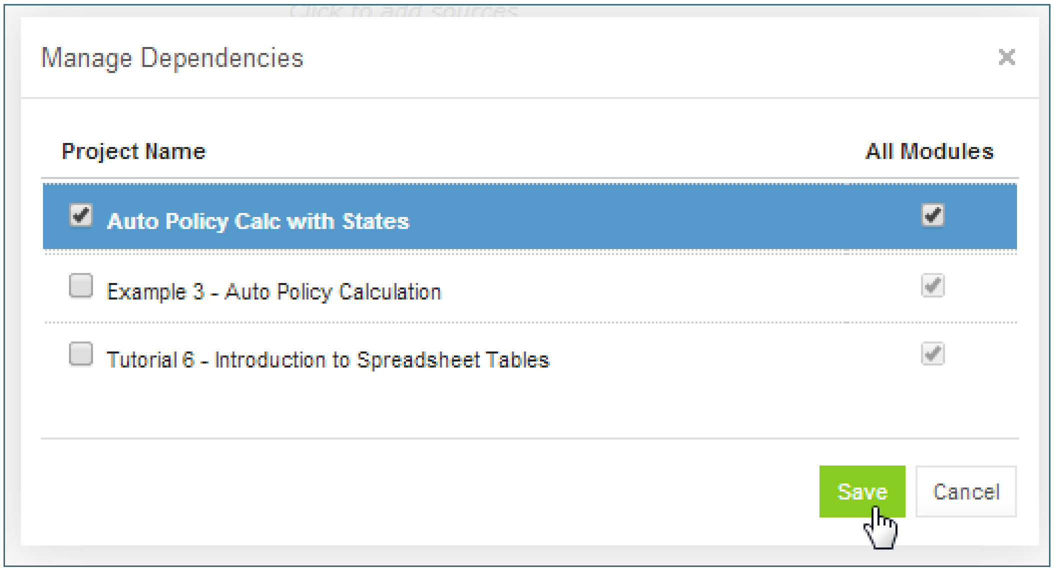



*Figure 31: Managing project dependencies*

If the **All Modules** option is selected in the multi-module mode, tables of all modules of the dependency project are accessible from any module of the root project.

If the **All Modules** option is cleared or the single module mode is selected, the root project module has access to the particular module of the dependency project only if an appropriate dependency is added in the **Environment** table of the root module.

**Note:** Module names of the root and dependency projects must be unique.

**Note:** Dependency projects must be available in Rules Editor to make dependency work.

For more information on project and module dependencies, see the **Project and Module dependencies** section in *[OpenL Tablets Reference Guide]*.

# 4.6 Viewing Tables

OpenL Tablets module tables are listed in the module tree. Table types are represented by different icons in Rules Editor. The following table describes table type icons:

| **Table type icons** | |
|---|---|
| **Icon** | **Table type** |
|  | Decision table. |
|  | Decision table with unit tests. |
|  | Column match table. |
|  | Column match table with unit tests. |
|  | Tbasic table. |
|  | Tbasic table with unit tests. |
|  | Data table. |

| Table type icons | |
|---|---|
| **Icon** | **Table type** |
| | Datatype table. |
| | Method table. |
| | Unit test table. |
| | Run method table. |
| | Environment table. |
| | Property table. |
| | Table not corresponding to any preceding types. Such tables are considered comments. |
| | Spreadsheet table, Constants table. |

For more information on table types, see ***[OpenL Tablets Reference Guide]***. If a table contains an error, a small red cross is displayed in the corner of the icon.

To view contents of a particular table, in the module tree, select the table. The table is displayed in the middle pane. If the project is not in the **In Editing** status, the table can be viewed but cannot be modified.

# 4.7 Modifying Tables

OpenL Tablets WebStudio provides embedded tools for modifying table data directly in a web browser. To modify a table, proceed as follows:

1. In the module tree, select the required table.
   The selected table is displayed in the middle pane in read mode.

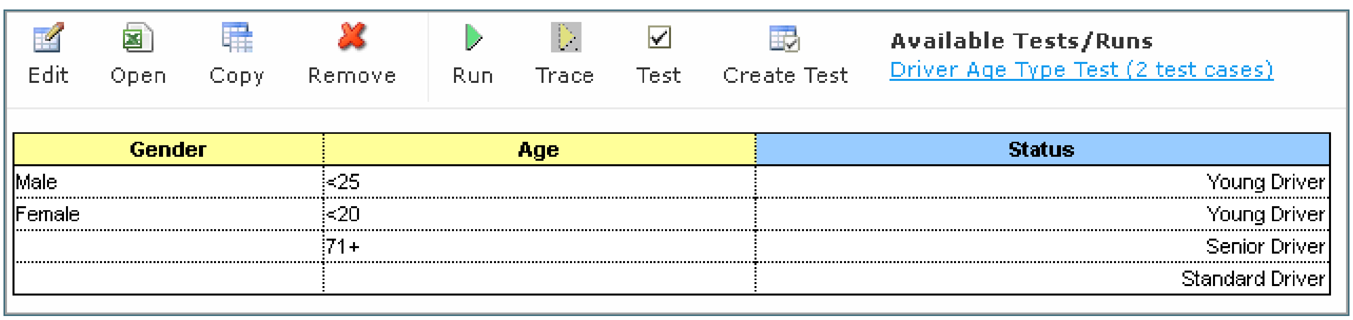



*Figure 32: Table opened in OpenL Tablets WebStudio*

2. To switch between simple and extended view, in **User settings**, select or clear the **Show Header** and **Show Formula** options as required.
3. To switch the table to the edit mode, perform one of the following steps:
   - Above the table, click **Edit**.
   - Right-click anywhere in the table and click **Edit**.
   - Double click the cell to edit.

   Alternatively, the file can be edited in Excel. In the local mode, the rule file is opened in Excel, and changes become available in OpenL Tablets WebStudio upon Excel file saving. In the remote mode or if a demo-policy file exists, clicking the **Open** button initiates file download. After editing the file locally, it can be uploaded back to the project in Rules Editor as described in [Exporting, Updating, and Editing a Module]() or via the repository.

The following table is switched to the edit mode:

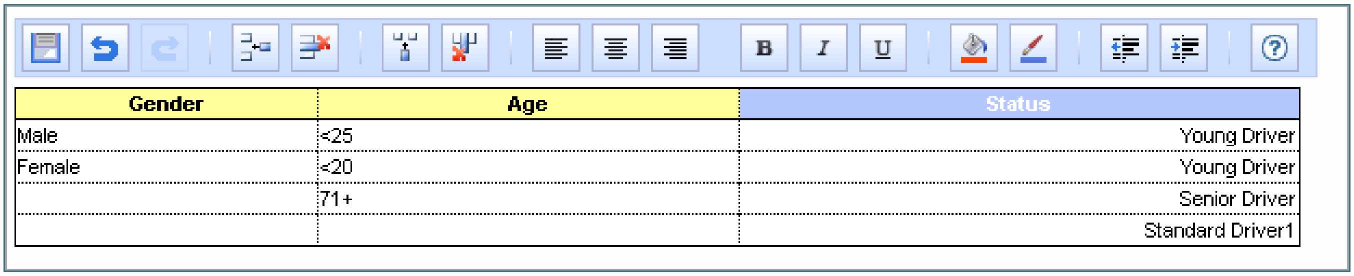
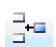
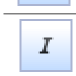
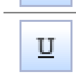
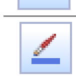



*Figure 33: Table in the edit mode*

The edit mode provides the following functional buttons:

| Table editing buttons | |
|---|---|
| **Button** | **Description** |
|  | Saves changes in table. |
|  | Reverses last changes. |
|  | Reapplies reversed changes. |
|  | Inserts a row. |
|  | Deletes a row. |
|  | Inserts a column. |
|  | Deletes a column. |
|  | Aligns text in currently selected cell with left edge. |
|  | Centers text in currently selected cell. |
|  | Aligns text in currently selected cell with right edge. |
|  | Make the text font **bold**. |
|  | Applies *italics* to the cell text. |
|  | Underlines the cell text. |
|  | Sets the fill color. |
|  | Sets the font color. |
|  | Decreases indent. |
|  | Increases indent. |

4. To modify a cell value, double click it or press **Enter** while the cell is selected.

5. To enter a formula in the cell, double click it, perform a right click, and select **Formula Editor.**
   Now a user can enter formulas in the selected cell.
6. To save changes, click **Save** .
   If a table contains an error, the appropriate message is displayed.

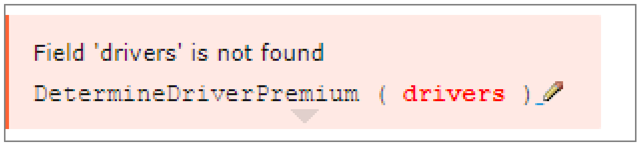



*Figure 34: Example of an error in a table*

The arrow under the message allows viewing all stack trace for this error.

```
Error: Field 'drivers' is not found
Openl Code Fragment:
======================
 DetermineDriverPremium ( drivers )
                          ^^^^^^^
======================
   at file:/home/eis/openl-tablets-demo/openl-demo/user-workspace/DEFAULT/Example%203%20-%20Auto%20Policy%20Calculation/AutoPolicyCalculation.xls?sheet=Calculation&cell=E50&start=27&end=33
java.lang.IllegalStateException: Field 'drivers' is not found
      at org.openl.binding.impl.ANodeBinder.assertNotNull(ANodeBinder.java:283)
      at org.openl.binding.impl.IdentifierBinder.bind(IdentifierBinder.java:37)
      at org.openl.binding.impl.ANodeBinder.bindChildNode(ANodeBinder.java:29)
      at org.openl.binding.impl.ANodeBinder.bindChildren(ANodeBinder.java:100)
      at org.openl.binding.impl.MethodNodeBinder.bind(MethodNodeBinder.java:86)
      at org.openl.binding.impl.ANodeBinder.bindChildNode(ANodeBinder.java:29)
      at org.openl.binding.impl.ANodeBinder.bindChildren(ANodeBinder.java:100)
      at org.openl.binding.impl.ANodeBinder.bindChildren(ANodeBinder.java:73)
      at org.openl.binding.impl.BlockBinder.bind(BlockBinder.java:31)
      at org.openl.binding.impl.ANodeBinder.bindChildNode(ANodeBinder.java:29)
      at org.openl.binding.impl.ANodeBinder.bindChildren(ANodeBinder.java:100)
      at org.openl.binding.impl.ANodeBinder.bindChildren(ANodeBinder.java:73)
      at org.openl.binding.impl.BlockBinder.bind(BlockBinder.java:31)
      at org.openl.binding.impl.ANodeBinder.bindChildNode(ANodeBinder.java:29)
      at org.openl.binding.impl.Binder.bind(Binder.java:93)
      at org.openl.engine.OpenLSourceManager.processSource(OpenLSourceManager.java:227)
      at org.openl.engine.OpenLCodeManager.makeMethodWithUnknownType(OpenLCodeManager.java:141)
      at org.openl.engine.OpenLCellExpressionsCompiler.makeMethodWithUnknownType(OpenLCellExpressionsCompiler.java:89)
      at org.openl.rules.calc.SpreadsheetStructureBuilder.extractCellValue(SpreadsheetStructureBuilder.java:196)
      at org.openl.rules.calc.SpreadsheetStructureBuilder.makeType(SpreadsheetStructureBuilder.java:144)
      at org.openl.rules.calc.element.SpreadsheetCellField.getType(SpreadsheetCellField.java:44)
      at org.openl.binding.impl.FieldBoundNode.getType(FieldBoundNode.java:108)
      at org.openl.binding.impl.ANodeBinder.getTypes(ANodeBinder.java:150)
      at org.openl.binding.impl.MethodNodeBinder.bind(MethodNodeBinder.java:90)
      at org.openl.binding.impl.ANodeBinder.bindChildNode(ANodeBinder.java:29)
      at org.openl.binding.impl.ANodeBinder.bindChildren(ANodeBinder.java:100)
```

*Figure 35: Error stack trace example*

## 4.8 Referring to Tables

OpenL Tablets WebStudio supports references from one table to another table. A referred table can be located in the same module where the first table resides, or in the different module of the same project.

Links to the following tables are allowed:

- data table
- datatype table
- rule table types

Links to the rule tables are underlined and marked blue. When a mouse cursor is put over the link, a tooltip with method name and input parameters with types is displayed.


*Figure 36: A tooltip for the linked method to a decision table*

Links to the data and datatype tables are underlined with a dotted line and has an appropriate tooltip with description.

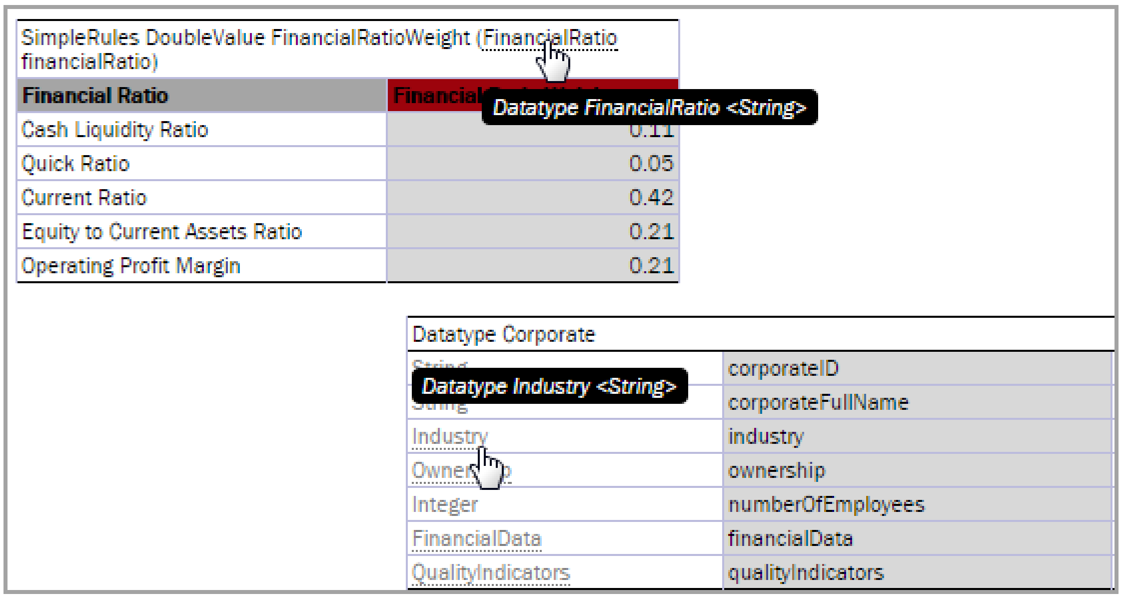



*Figure 37: Links to the datatype tables from the decision and datatype table*

All fields of the datatype tables are also linked and contain tooltips.

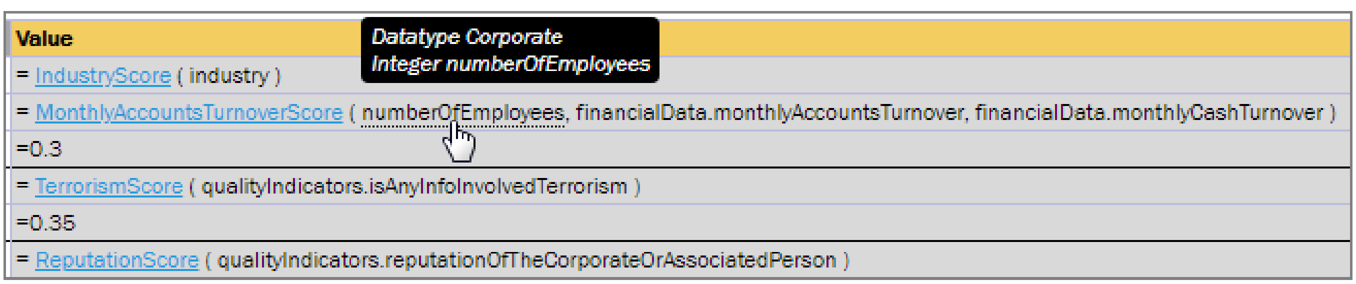



*Figure 38: A link to the field of the Corporate datatype table*

# 4.9 Managing Range Data Types

OpenL Tablets WebStudio provides a special tool, **Range Editor**, for adding and editing range data types, such as IntRange and DoubleRange, in rule tables and test tables.

This section briefly introduces Range Editor and provides examples of its functionality.

The main Range Editor goal is to move to a single range format in OpenL rules, namely, the '..' format. For more information on ranges on OpenL Tablets, see the **Range types in OpenL** section in ***[OpenL Tablets Reference Guide]***.

Consider the following principles while working with Range Editor:

- The default range format is set to '..' in OpenL Tablets WebStudio.
- When a new range is created, the '..' format is used.
- When a range format other than '..' is edited, if only range values are edited, the format remains the same. If any editor control is used, for example, a check box or the **Done** button, the range format is set to '..'.

The following example displays the decision table with data represented as a range:

| Rules String Greeting3 (Integer hour) | |
|---|---|
| C1 | RET1 |
| hour | greeting + ", World!" |
| IntRange | String greeting |
| **Hour** | **Greeting** |
| 0 - 11 | Good Morning |
| 12 - 17 | Good Afternoon |
| 18 - 21 | Good Evening |
| | Good Night |

*Figure 39: Decision table with a range data type*

In this table, the **Hour** column contains hours with the IntRange Data type. All range sells are filled except for the last one. This example is used further in this section to demonstrate how Range Editor works.

The following controls are available in Range Editor:

- **From** — indicates the left border of the range
- **To** — indicates the right border of the range
- **Include** — indicates whether the border is included in the range
- **'>'** — indicates values greater than the specified border
- **'<'** — indicates values smaller than the specified border
- **'='** — indicates a constant
- **'-'** — indicates a range

To create a range, proceed as follows:

1. Double click the cell to be edited.

   For example, edit the cell containing 18-21. The table is extended by the pop-up window with a set of controls for editing the range.

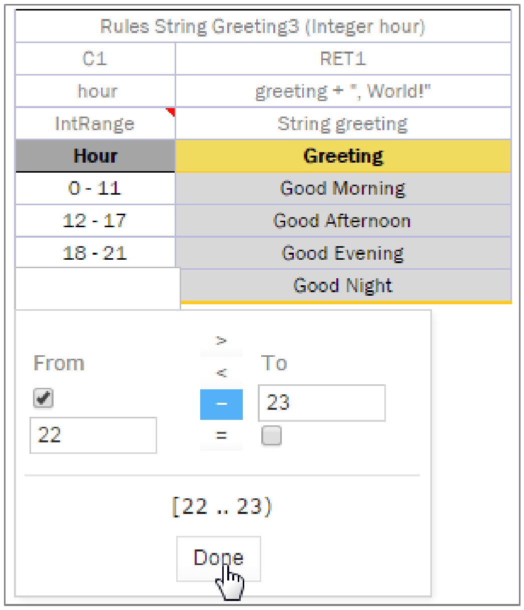



*Figure 40: Creating a range in Range Editor*

2. In the **From** field, enter the left border of the range, which is 22 for the example described in this section.
3. In the **To** field, enter the right border of the range.

   In this example, the **To** value must be 24, but an erroneous value 23 is entered for further editing of this border.
4. Clear the **Include** check box.
5. Click **Done** to complete.

   The last cell in the **Hour** column is filled as follows:

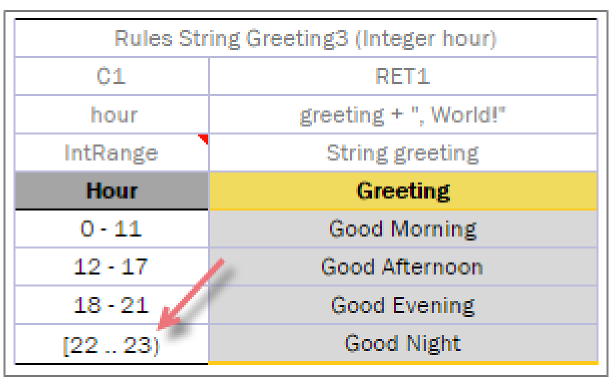



*Figure 41: New range created in Range Editor*

6. To modify the range in Range Editor, double click the cell with the [22-23) range.

   The table resembles the following:

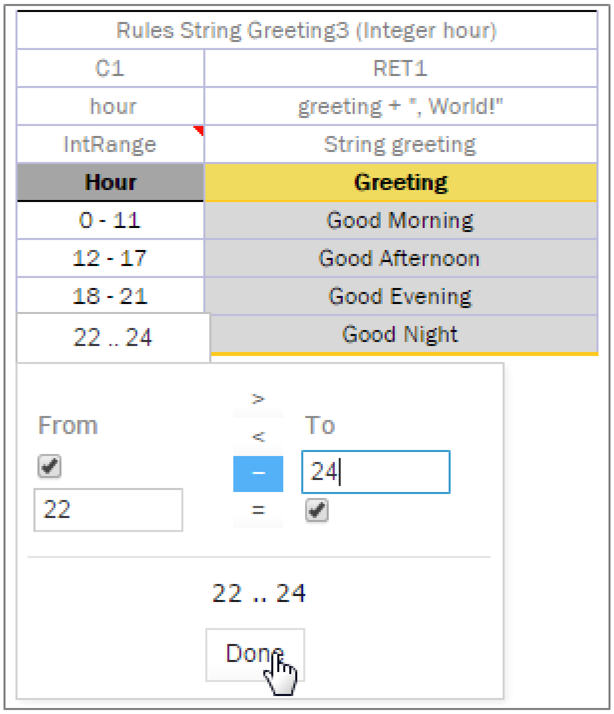



*Figure 42: Editing a range in Range Editor*

7. Select the **To** field, set the right border to 24, and select **Include**.
8. Click **Done** to save the work.
   The range resembles the following:

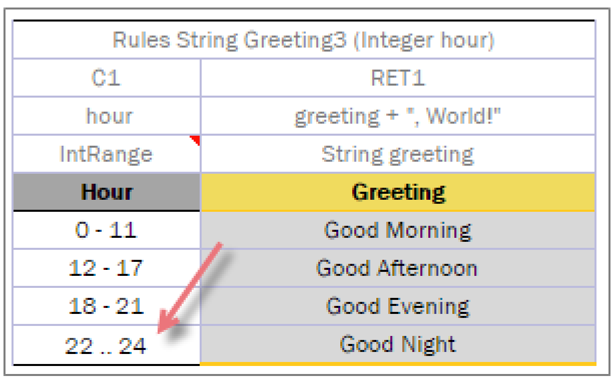



*Figure 43: The range edited in Range Editor*

A range can also be modified using '>', '<' and '=' controls as described in the beginning of this section.

# 4.10 Creating Tables by Copying

A table can be created based on another table using one of the following methods:

- Copying the Existing Table
- Creating a New Version of the Table
- Creating a Table as a New Business Dimension Version

## Copying the Existing Table

To create a table as a copy of the existing table, proceed as follows:

1. In the module list, select a table to copy.
2. Click the **Copy Table** icon .
   The system displays the **Copy Table** form with **New Table** selected by default.

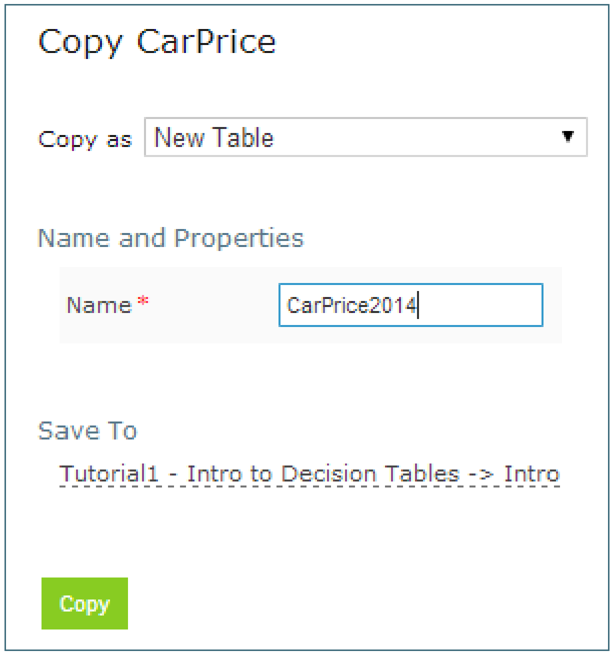



*Figure 44: Copying the existing table*

3. If necessary, modify the **Name** field value.
4. To change the workbook and worksheet where the copy must be saved, click the link in the **Save To** area and in the corresponding drop-down list, select the required module and category.
5. To save the copied table in a new category, use the **New** option.
6. Click **Copy** to save your changes.
   The table appears in the module list.

## Creating a New Version of the Table

To create a new version of the existing table, proceed as described in Using Table Versioning. In that case, dimensional properties of a new version are exactly the same as for the original one. OpenL Tablets allows creating an overloaded table from an existing one.

## Creating a Table as a New Business Dimension Version

To create a table as a new business dimension version, proceed as follows:

1. In the module list, select a table and click the **Copy Table** icon.
2. In the **Copy as** list, select **New Business Dimension Version**.
3. Specify business dimension properties as required.

4. If necessary, modify the workbook and worksheet values in the **Save as** area.
5. Click **Copy** to save the table.

# 4.11 Performing a Search

OpenL Tablets WebStudio provides search functionality to look through all module tables data for a particular project. The following topics describe search modes in OpenL Tablets WebStudio:

- Performing a Simple Search
- Performing an Advanced Search

## Performing a Simple Search

In the **simple search** mode, the system looks for a particular word or phrase in all tables within the given module.

To perform a simple search, in the **Search** field, enter a word or phrase and press **Enter**.

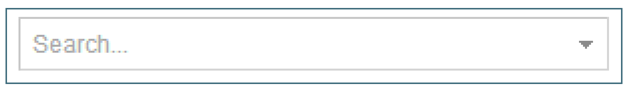



*Figure 45: Starting a simple search*

OpenL Tablets WebStudio displays all tables containing the entered text. Above each table, there is the **Open Table in Excel** link redirecting to the Excel file containing the entered text. The **Edit Table** link opens the table in Rules Editor in the editing mode.

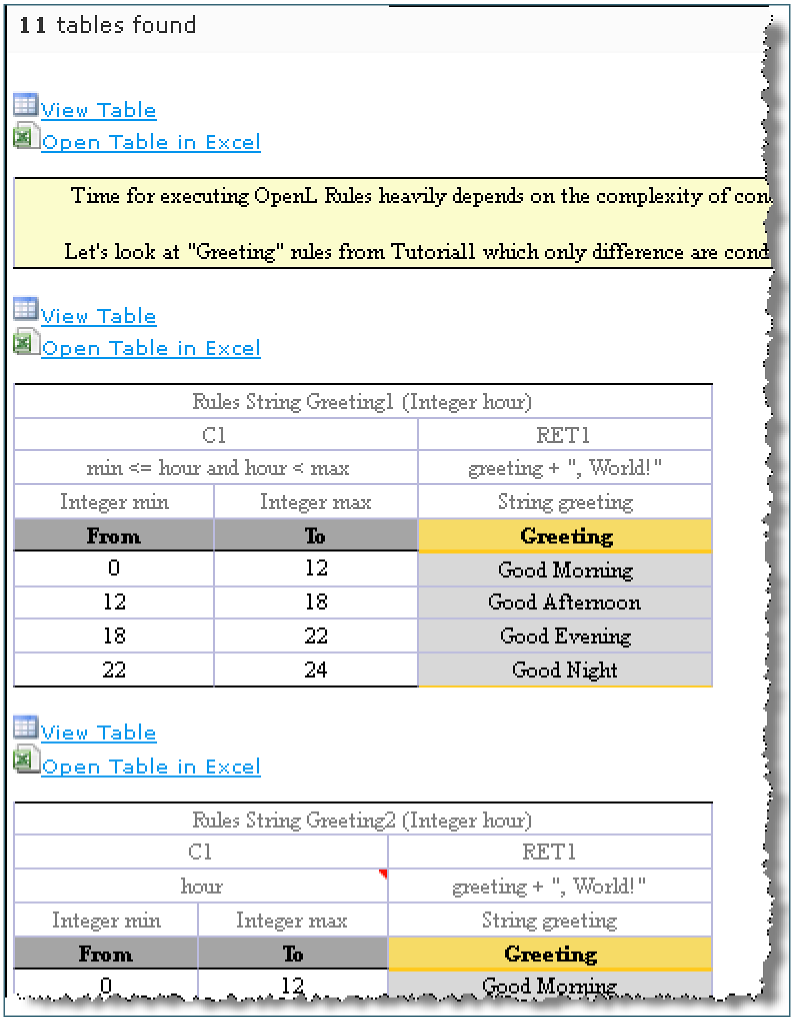



*Figure 46: Search results*

To search for any cell contents, right click the cell and in the context menu, select **Search**. The table is opened in the read mode.

## Performing an Advanced Search

Advanced search allows specifying criteria to narrow the search through tables. To limit the search, specify the table type, text from the table header, and table properties as described further in this section.

1. To launch an advanced search, click the arrow to the right of the search window.

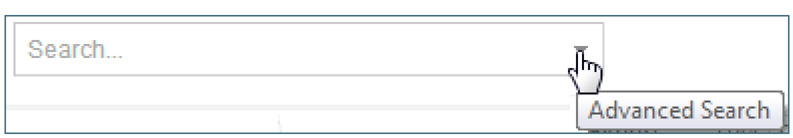



*Figure 47: Initiating the advanced search*

2. In the filter form, click the **Table Types** field and select the required table type or select **Select All** to search in all table types.
3. In the **Header contains** field, enter the word or phrase to search for.

4. Expand the **Table Properties** list, select the required table property, and then click the **Add** button on the right.
   The text field for entering the property name appears.
5. Enter the property name.
6. In the similar way, add as many table properties as required.
7. To remove a property, click the cross icon to the right of the property.

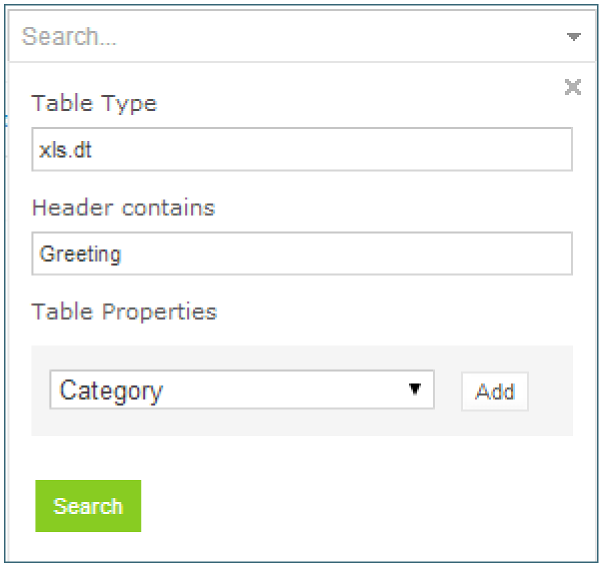



*Figure 48: A filled form for advanced search*

8. Click **Search** to run the search.

As a result, the system displays the tables matching the search criteria along with links to the relevant Excel files and the **Edit Table** links leading to the table editing page.

View Table
Open Table in Excel

| Rules DoubleValue CarPrice (Car car, Address billingAddress) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| properties | effectiveDate | 1/1/09 | | | | | | |
| | expirationDate | 1/1/10 | | | | | | |
| Rule | C1 | C2 | HC1 | HC2 | RET1 | | | |
| | country | region | brand | model | | | | |
| | Country | String | CarBrand | String | | | | |
| # Rule | Country | Region | BMW | | Porche | | | |
| | | | Z4 sDrive35i | Z4 sDrive30i | 911 Carrera 4S | 911 Targa 4 | 911 Carrera Cabriolet | 2009 Audi R8 4.2 quattro Auto |
| R1 | USA | Pacific West | $51,650 | $45,750 | $93,200 | $90,400 | $87,000 | $121,500 |
| R2 | | West | $52,000 | $44,050 | $93,200 | $90,400 | $87,000 | $121,500 |
| R3 | | Mid Atlantic | $52,450 | $46,550 | $93,200 | $90,400 | $87,000 | $121,500 |
| R4 | GreatBritain | England | $53,650 | $47,750 | $94,200 | $91,400 | $88,000 | $121,500 |
| R5 | | Wales | $53,650 | $47,750 | $95,200 | $92,400 | $89,000 | $121,500 |
| R6 | | Scotland | $53,650 | $47,750 | $96,200 | $93,400 | $90,000 | $121,500 |
| R7 | Belarus | Minsk | $56,650 | $49,750 | $93,200 | $90,400 | $87,000 | $121,500 |
| R8 | | Vitebsk | $56,650 | $49,750 | $93,200 | $90,400 | $87,000 | $121,500 |
| R9 | | Grodna | $56,650 | $49,750 | $93,200 | $90,400 | $87,000 | $121,500 |

*Figure 49: Advanced search result*

# 4.12 Creating Tables

OpenL Tablets WebStudio allows creating tables of the following types:

- datatype table
- datatype alias table
- data table
- test table
- properties table
- simple rules table

Tables are created via the wizard initiated by clicking the **Create Table** button Create Table. The wizard creates a table for the current module. The table is available for all included modules and modules linked by dependencies. For more information on dependencies, see the **Project and Module dependencies** section in *[OpenL Tablets Reference Guide]*.

The following topics are included in this section:

- Creating a Datatype Table
- Creating a Data Table
- Creating a Test Table and Defining the ID Column for Test Cases
- Creating a Simple Rules Table

## Creating a Datatype Table

To create a datatype table, proceed as follows:

1. In OpenL Tablets WebStudio, click **Create Table**.

2. In the list of table types, select **Datatype Table** and click **Next**.

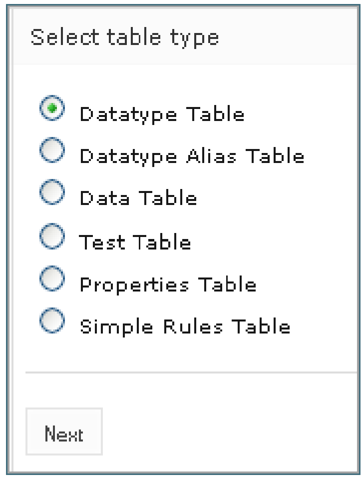



*Figure 50: Creating a Datatype table*

3. Enter the data type name and if necessary, select the existing data type as a parent.
   If a parent data type value is specified, the newly created data type will have access to all fields defined in the parent data type as described in the **Inheritance in Data types** section in ***[OpenL Tablets Reference Guide]***.
   This option is unavailable if no custom data types are created in the module.

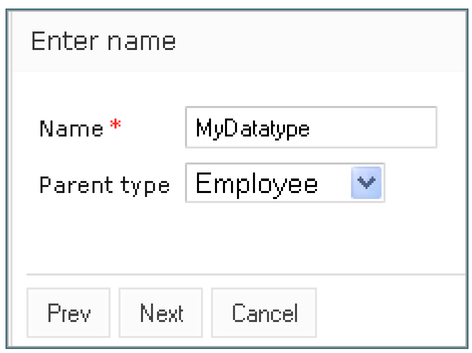



*Figure 51: Specifying the data type name and parent type*

4. To define data type fields, click **Add parameter**, specify values as required, and then click **Next**.

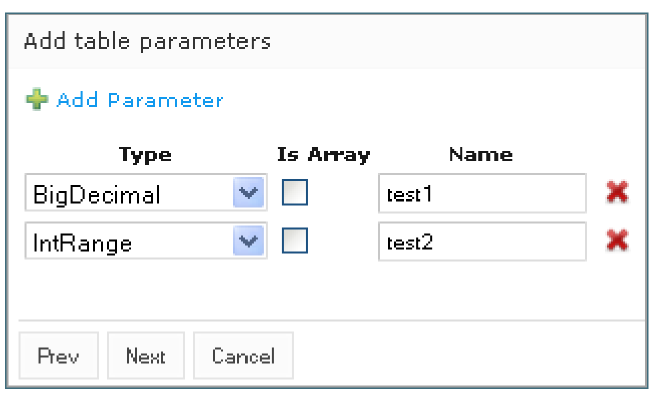



*Figure 52: Defining data type fields*

5. To indicate the new datatype table location, in the **Select destination** window, select an existing sheet, or in the **New** field, enter the new sheet name.
   The **Module** value cannot be changed. All created tables go to the current module.

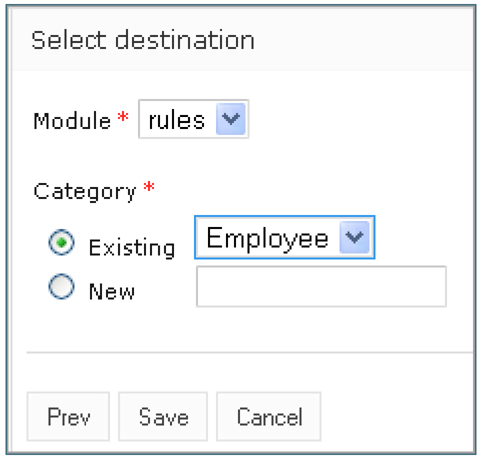



*Figure 53: Specifying table location*

6. Click **Save** to complete table creation.
   The datatype table is created and becomes available in OpenL Tablets WebStudio.

## Creating a Data Table

Creating a data table resembles creating a datatype table described in [Creating a Datatype Table](). Proceed as follows:

1. In OpenL Tablets WebStudio, click **Create Table**.
2. Select the **Data Table** item and click **Next**.

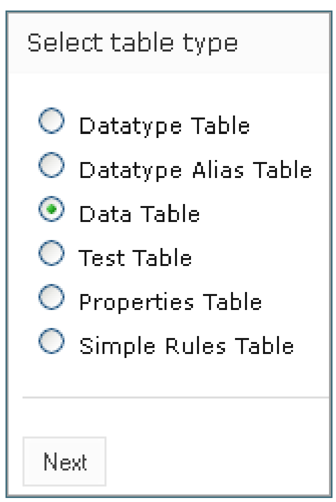



*Figure 54: Initiating data table creation*

3. Select the table type, enter the table name, and click **Next**.

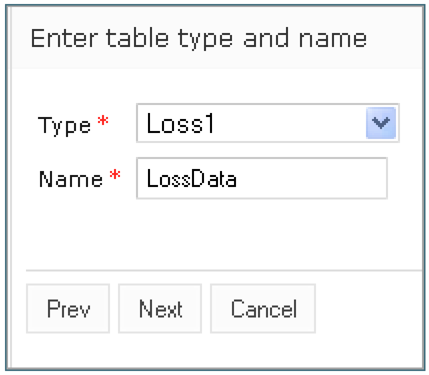



*Figure 55: Defining table type and name*

4. Define the table columns configuration.
   For the **Loss1** type selected in the previous window, column configuration resembles the following:

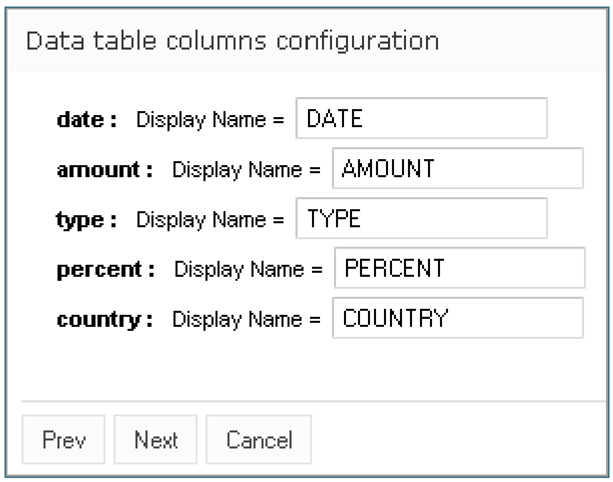



*Figure 56: Defining column configuration*

5. To indicate new data table location, in the **Select destination** window, select an existing sheet, or in the **New** field, enter the new sheet name.
   The **Module** value cannot be changed. All created tables go to the current module.

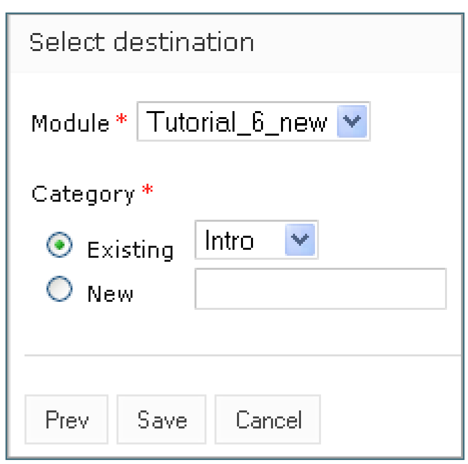



*Figure 57: Specifying table location*

6. Click **Save** to complete table creation.

The new data table is created and can be modified as needed.

OpenL Tablets supports array value definition in data tables as described in the **Defining Array Values in Test and Data Tables** section in ***[OpenL Tablets Reference Guide]***.

# Creating a Test Table and Defining the ID Column for Test Cases

This section describes how to create a test table and define the ID column for test cases and includes the following topics:

- Creating a Test Table
- Defining the ID Column for Test Cases

## Creating a Test Table

To create a test table, proceed as follows:

1. In OpenL Tablets WebStudio, click **Create Table**.
2. Select **Test Table** and click **Next**.

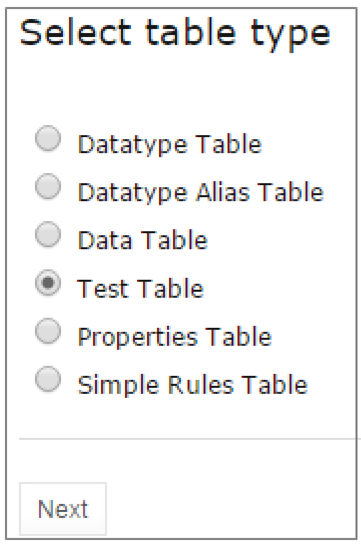



*Figure 58: Creating a test table*

3. In the **Select table** window, select the rule table and click **Next**.

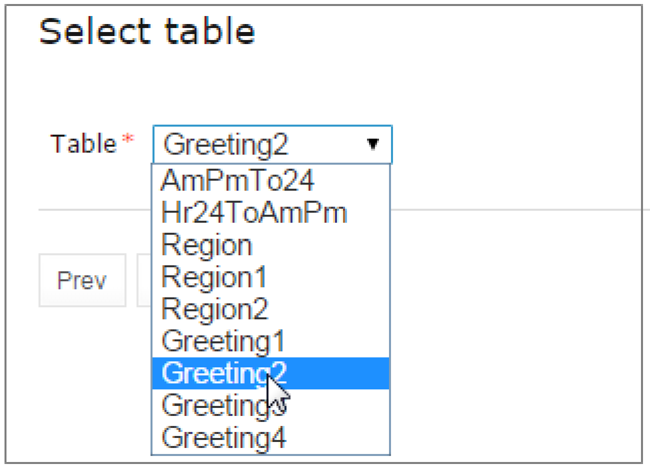



*Figure 59: Selecting a rule table to create a test table for*

**Note:** If there is no rule table available in this module, a test table cannot be created, and an error message is displayed.

4. In the **Input name** window, if necessary, modify the generated test table name and click **Next**.

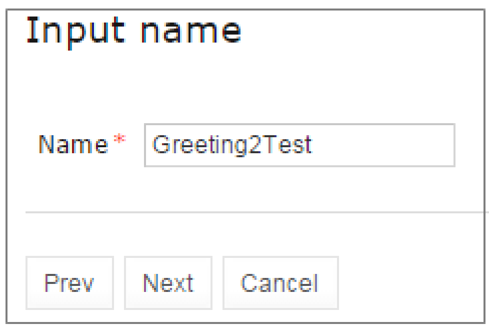



*Figure 60: Reviewing the test table name*

5. To define the test table location, in the **Select destination** window, select an existing sheet, or in the **New** field, enter the new sheet name.
   The **Module** value cannot be changed. All created tables go to the current module.

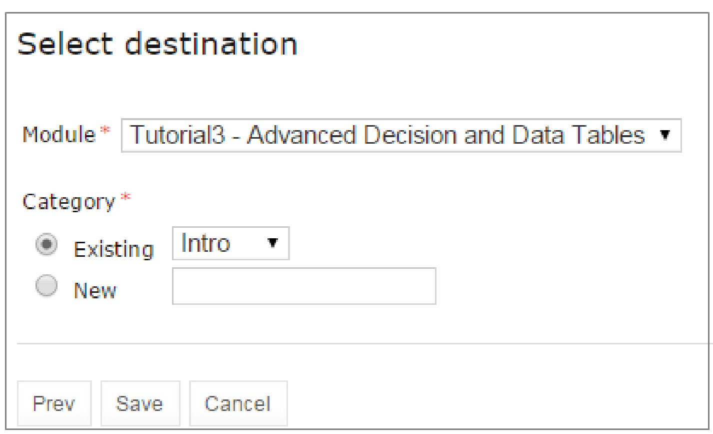



*Figure 61: Specifying table location*

6. To complete table creation, click **Save**.

The test table is created and becomes available in OpenL Tablets WebStudio.

OpenL Tablets supports array value definition in test tables as described in the **Defining Array Values in Test and Data Tables** section in ***[OpenL Tablets Reference Guide]***.

## Defining the ID Column for Test Cases

The ID column is not mandatory in a test table. A user can define the ID column and set the appropriate unique value for each test case.

| Test AmPmTo24 AmPmTo24Testv2 | | | |
|---|---|---|---|
| _id_ | ampmHr | ampm | _res_ |
| **ID** | **Hour** | **AM/PM** | **24 Hr** |
| TC1 | 3 | AM | 3 |
| TC2 | 12 | AM | 0 |
| TC3 | 12 | PM | 12 |
| TC4 | 3 | PM | 15 |

*Figure 62: A test table with the ID column defined*

If the ID column is not defined for the test table, default numeric values are displayed beside each test case.

When running a test table, to run the test cases, expand the additional settings for the **Run** button and select the required cases.

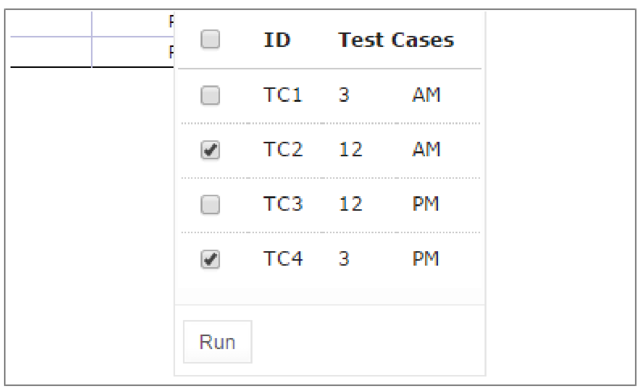



*Figure 63: Running the specified test cases*

To use ranges of IDs for executing the required cases, enable the **Use the Range** setting and in the **Range of IDs** field, specify the ID values separated by dash or comma.

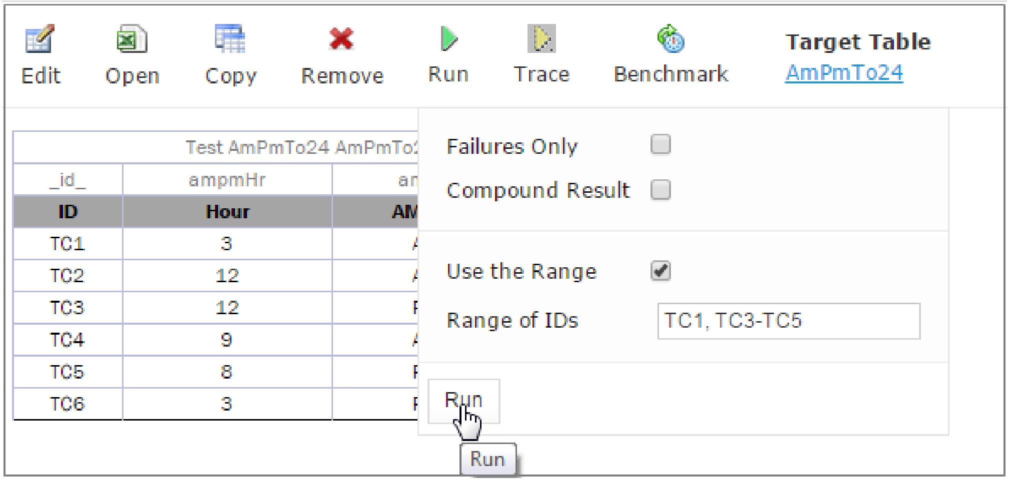



*Figure 64: Specifying test cases ID range*

## Creating a Simple Rules Table

This section describes how to create a new simple rules table in OpenL Tablets WebStudio.

1. In OpenL Tablets WebStudio, click **Create Table**.
2. Select **Simple Rules Table** and click **Next**.

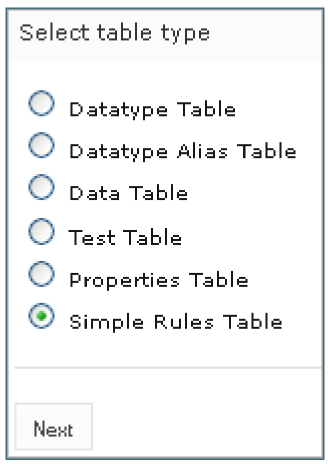



*Figure 65: Initiating table creation*

3. Enter table name and select the required data type to return.
4. Click **Add Input Parameters** and specify values as required.

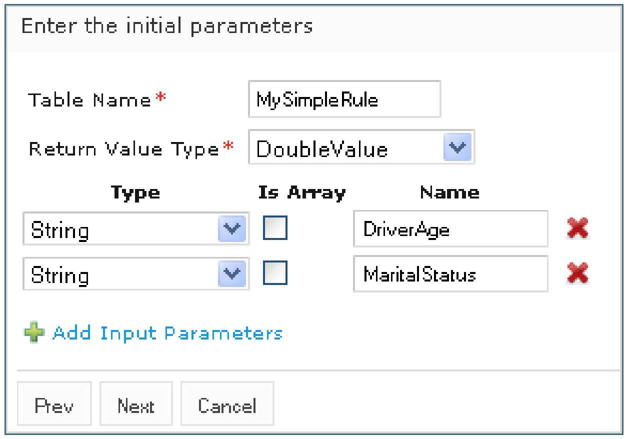



*Figure 66: Specifying table parameters*

5. When finished, click **Next**.

   In the **Construct a table** window that appears, a blank simple rules table with the header constructed based on the previously entered values appears.

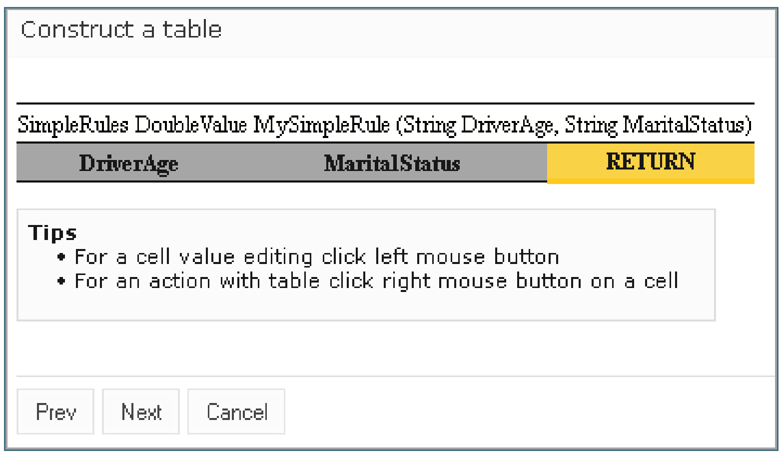



*Figure 67: Adding data to a table*

   Now the table can be filled with data.

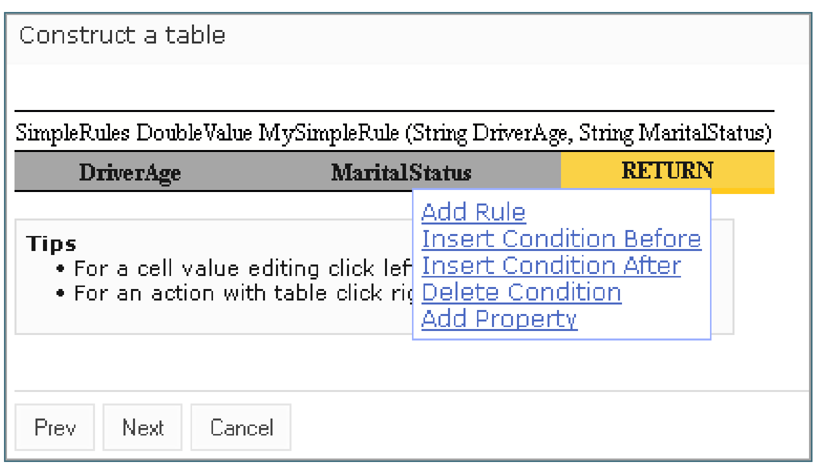



*Figure 68: Selecting an action from the context menu*

6. Right click any cell and select one of the following actions:

<table>
<tr><th colspan="2">Actions available for simple rules table</th></tr>
<tr><th>Action</th><th>Description</th></tr>
<tr><td>Add Property</td><td>Appears after selecting a property in the drop-down list and indicating its value.</td></tr>
<tr><td>Add Rule</td><td>Allows entering data. An example is as follows:
<table>
<tr><td colspan="3">SimpleRules DoubleValue MySimpleRule (String DriverAge, String MaritalStatus)</td></tr>
<tr><th>DriverAge</th><th>MaritalStatus</th><th>RETURN</th></tr>
<tr><td>Young Driver</td><td>Married</td><td>200</td></tr>
</table>
Figure 69: Entering table data<br>
This action can be repeated as many times as required.</td></tr>
<tr><td>Insert Condition Before / Insert Condition After</td><td>Adds a condition column to the specified position. An example of the added DriverOccupation condition column is as follows:
<table>
<tr><td colspan="4">SimpleRules DoubleValue MySimpleRule (String occupation, String DriverAge, String MaritalStatus)</td></tr>
<tr><th>DriverOccupation</th><th>DriverAge</th><th>MaritalStatus</th><th>RETURN</th></tr>
<tr><td>Teacher</td><td>YoungDriver</td><td>Married</td><td>200</td></tr>
<tr><td>IT</td><td>SeniorDriver</td><td>Single</td><td>220</td></tr>
</table>
Figure 70: Adding a condition column</td></tr>
<tr><td>Delete Condition / Delete Rule</td><td>Removes a condition or rule.</td></tr>
</table>

7. When finished, click **Next**.
8. To indicate new table location, in the **Select destination** window, select an existing sheet, or in the **New** field, enter the new sheet name.

   The **Module** value cannot be changed. All created tables go to the current module.
9. Click **Save** to save the changes.

   The new simple rules table is created and appears in the project.

# 4.13 Comparing Excel Files

OpenL Tablets WebStudio supports comparing contents of Excel files displaying tables and Excel elements that are modified. To compare two Excel files, proceed as follows:

1. In OpenL Tablets WebStudio Rules Editor, in the top line menu, select **More > Compare Excel Files.**

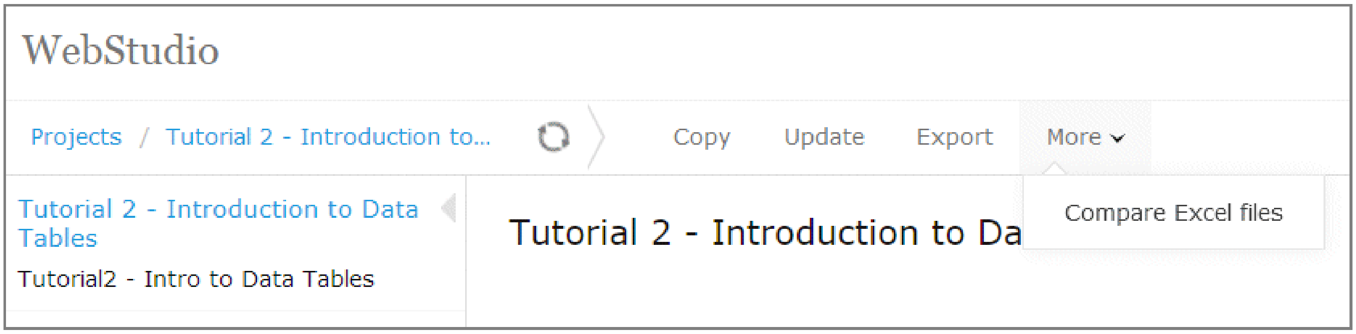



*Figure 71: Initiating Excel comparison functionality*

2. In the window that appears, click **Add** and select two Excel files to compare.
3. Click **Upload** and wait until file status is changed to **Done.**

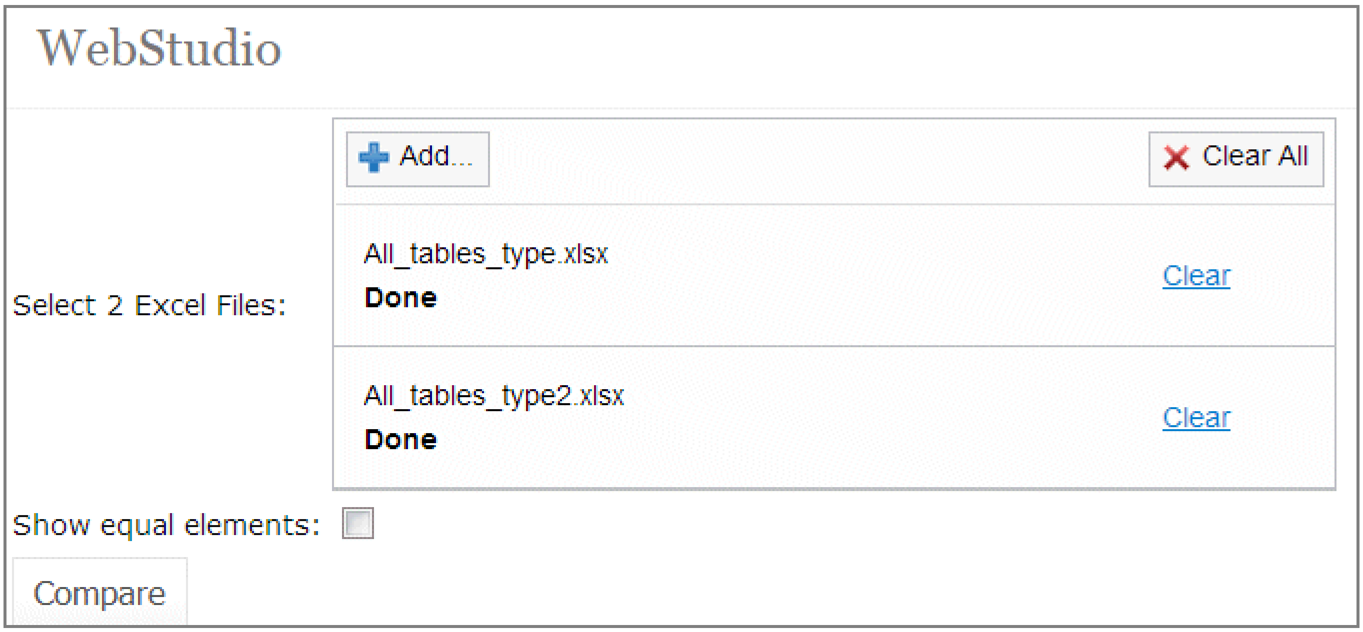



*Figure 72: Excel files ready for comparison*

4. To display tables and other Excel file elements that differ in the selected Excel files, click **Compare.**
   The list of tables and Excel elements is displayed, grouped by Excel sheets. Clicking on the table or element in the list displays the changes in the section below.

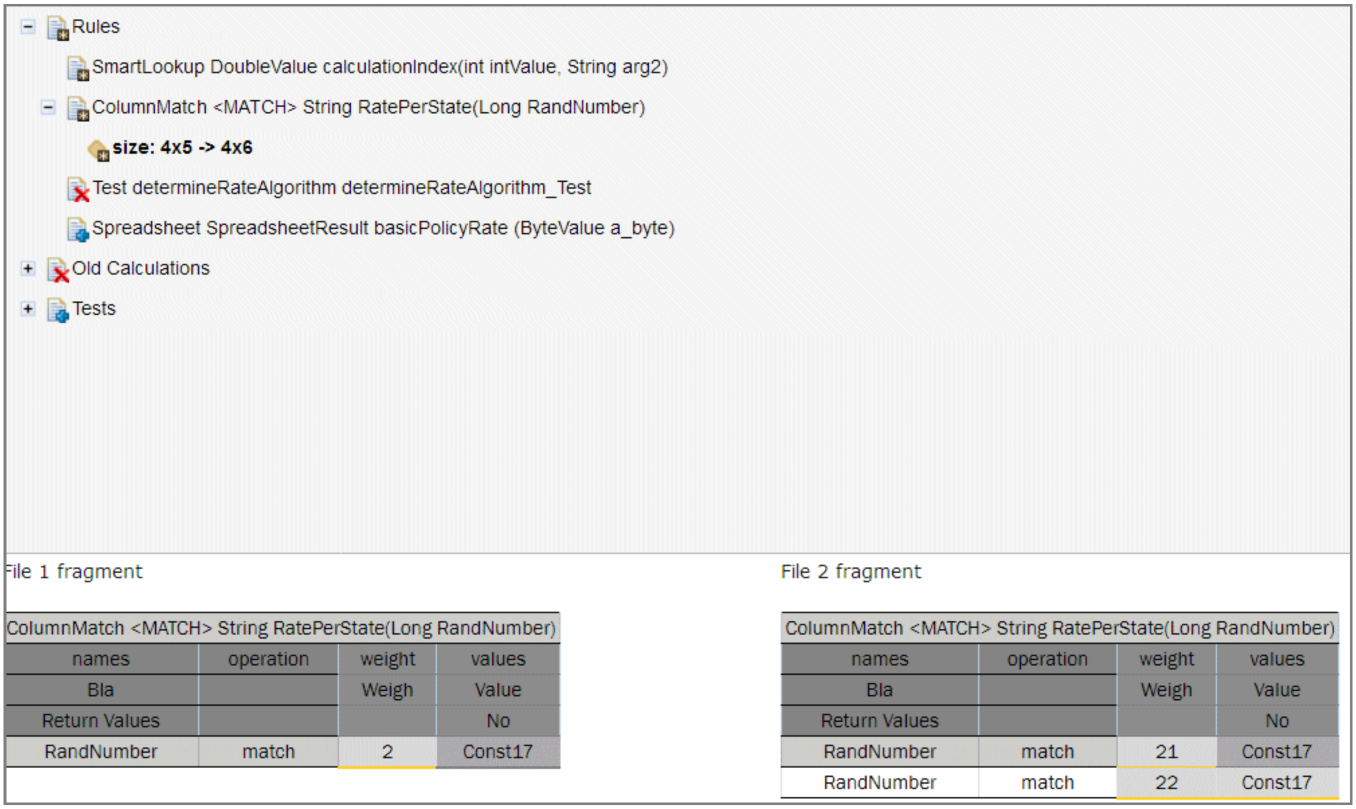



*Figure 73: Excel file comparison results*

Elements and tables that changed the location or contents are marked with the asterisk icon . Added elements are marked with the plus sign icon . Removed elements or tables are marked with the deletion icon .

5. To display all equal tables and Excel file elements in the selected Excel files, select **Show equal elements** check box and click **Compare.**

   All elements that are equal in the selected Excel files are displayed, grouped by Excel sheets. Elements that are relocated, added, or removed are marked with an appropriate icon.

# 5 Editing and Testing Functionality

This chapter describes advanced OpenL Tablets WebStudio functions, such as table editing, performing unit tests, rule tracing, and benchmarking. The following sections are included in this chapter:

- Editing Tables
- Using Table Versioning
- Performing Unit Tests
- Tracing Rules
- Using Benchmarking Tools

## 5.1 Editing Tables

This section describes table editing and includes the following topics:

- Editing a Comma Separated Array of Values
- Editing Default Table Properties
- Editing Inherited Table Properties

### Editing a Comma Separated Array of Values

OpenL Tablets WebStudio allows editing comma separated arrays of values. A multi selection window displaying all values appears enabling the user to select the required values.

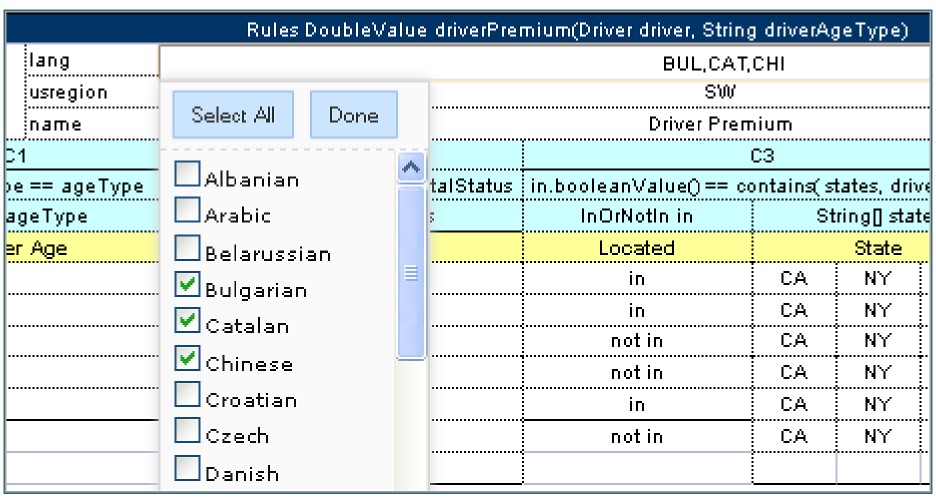



*Figure 74: Editing comma separated arrays*

### Editing Default Table Properties

This section describes table properties available in OpenL Tablets WebStudio. For more information on table properties, see the **Table Properties** section in *[OpenL Tablets Reference Guide]*.

If default property values are defined for a table, they appear only in the right hand **Properties** section, but not in the table. In the following example, there are **Active = true** and **Fail On Miss = false** default properties.

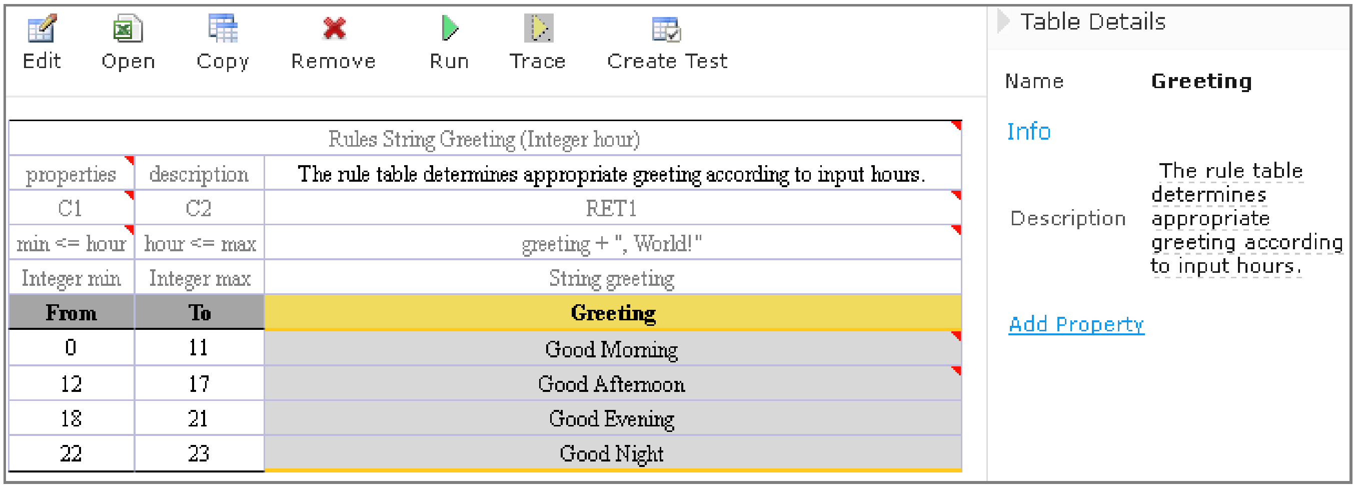



*Figure 75: Default table properties example*

Default properties can be overridden at the table level; in other words, they can be changed as follows:

1. In the **Properties** section, click the default property to be changed.
   Instead of the property value, a checkbox appears:

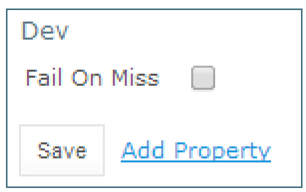



*Figure 76: Updating a default property*

2. Select or deselect the checkbox as needed and click the **Save** button.
   The property appears in the table with its new value:

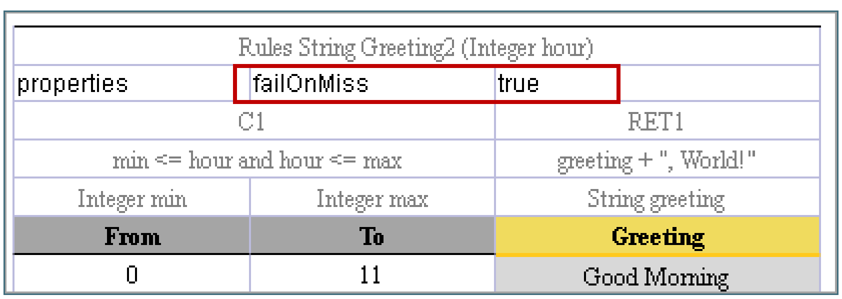



*Figure 77: Default property was updated by a user*

## Editing Inherited Table Properties

Module or category level properties are those inherited from a **Properties** table as described in the **Properties Table** section in ***[OpenL Tablets Reference Guide]***. In the **Properties** section of the given table, inherited properties appear in a different color and are accompanied with a link to the **Properties** table where they are defined. The values of the inherited properties are not stored in the table, they are displayed in the **Properties** section, since they are inherited and applied to this table. Inherited properties can be overridden at a Table level, i.e. they can be changed.

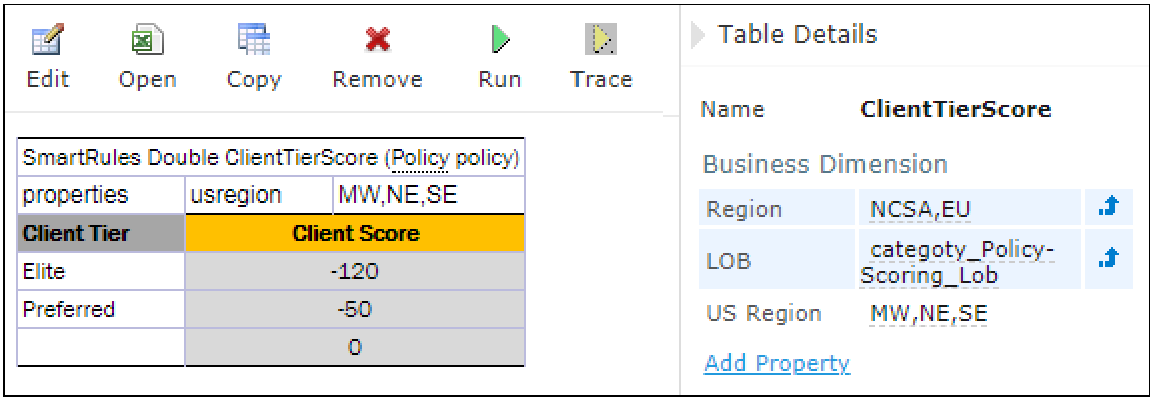



*Figure 78: An example of inherited category-level properties*

To change an inherited property, perform the following steps:

1. In the **Properties** section, click the inherited property to be changed.
2. Enter or select the desired values from the drop-down list and click **Save**.

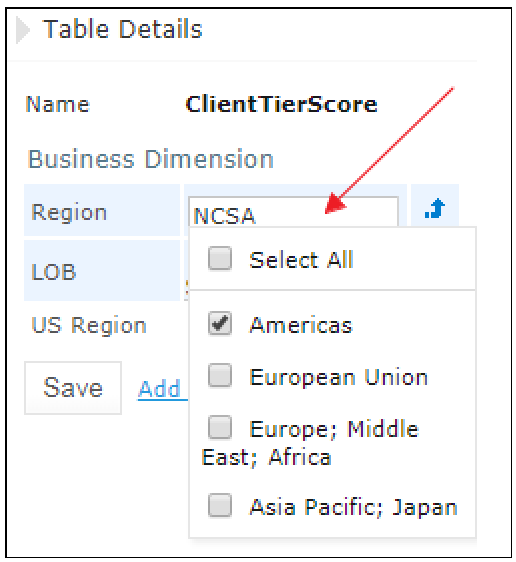
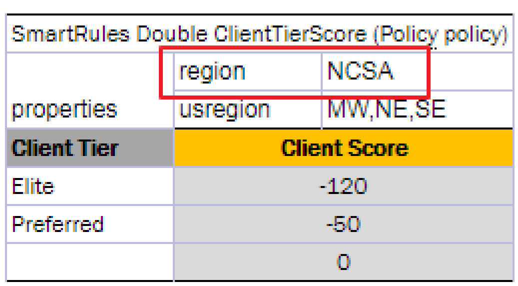



*Figure 79: Updating an inherited property*

The system displays the property in the table:



| SmartRules Double ClientTierScore (Policy policy) | | |
|---|---|---|
| properties | region | NCSA |
| | usregion | MW,NE,SE |
| **Client Tier** | **Client Score** | |
| Elite | -120 | |
| Preferred | -50 | |
| | 0 | |

*Figure 80: Inherited category-level property updated by a user*

The following topics are included in this section:

- [Editing System Properties]
- [Editing Properties for a Particular Table Type]

## Editing System Properties

By default, OpenL Tablets WebStudio applies system properties to each created or edited table. For information on how to switch off this option, please refer to Managing Common Settings. The values of the System properties are provided in the table and in the Properties section.

The **modifiedBy** property value is set using the name of the currently logged in user. The **modifiedOn** property is set according to the current date. These properties are applied upon each save.

The **createdBy** property value is set using the name of the currently logged in user. The **createdOn** property is set according to the current date. These properties are applied on the first save only while creating or copying a table in OpenL Tablets WebStudio.

The **createdBy** and **modifiedBy** properties are only applied in the multi-mode as described in Security Overview.

System properties cannot be edited in UI. The OpenL Tablets WebStudio users can delete those properties if required.

| Rules DoubleValue driverAccidentPremium(Driver driver, String driverRisk) | | |
|---|---|---|
| properties | modifiedOn | 10/26/12 |
| | modifiedBy | snm |
| C1 | | RET1 |
| driverRisk == risk | | accidentPremium * driver.numAccidents |
| String risk | | DoubleValue accidentPremium |
| **Driver Risk** | | **Per Accident Premium** |
| | | $160 |

*Figure 81: An example of system properties*

## Editing Properties for a Particular Table Type

Some properties are only applicable to particular types of tables. When opening a table in OpenL Tablets WebStudio, the properties section displays properties depending on the type of the table.

For example, such property as **Validate DT** is available for Decision Tables. That means it can be selected in the drop-down list after clicking the **Add** link at the bottom of the **Properties** section. The following figure shows properties applied to a Decision Table:

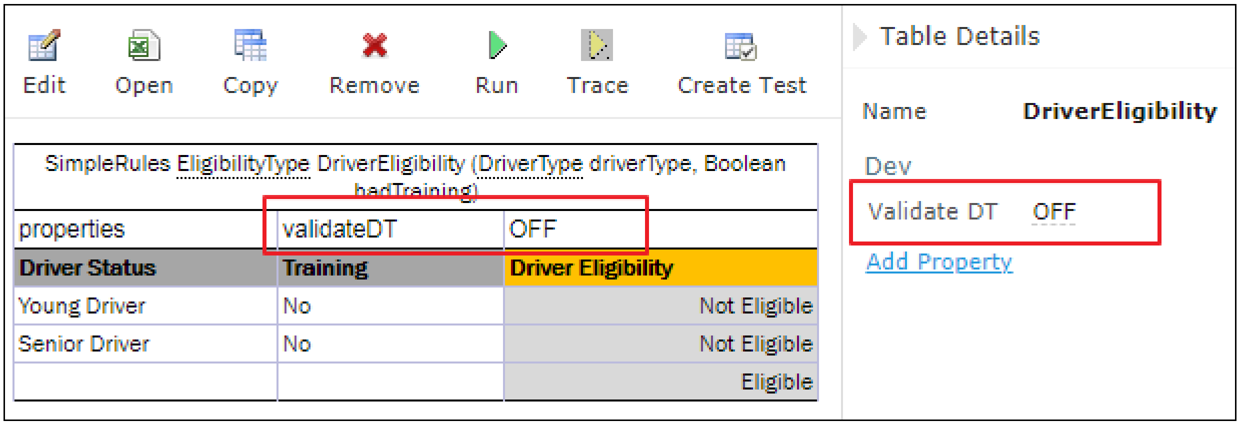



*Figure 82: Properties for the Decision table type*

When opening a Data Table in the same project, these properties are not available for selecting from the drop-down list in the **Properties** section.

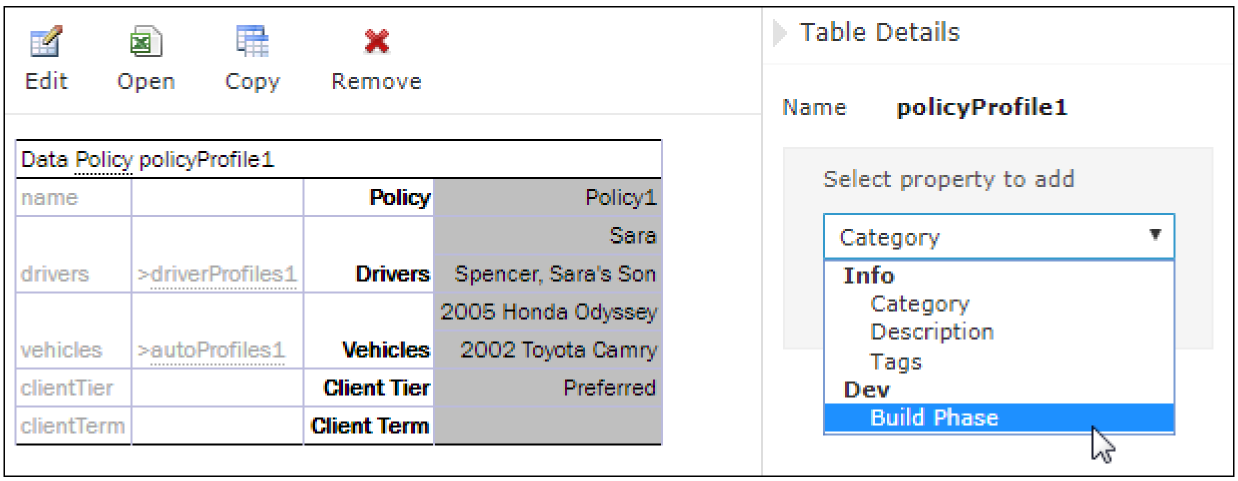



*Figure 83: The Decision table properties that are not available for a Data table*

When performing the “Copy” action, properties unsuitable for the current table type do not appear in the wizard.

To add a new property for the selected table, perform the following steps:

1. In the **Properties** pane, click the **Add Property** link.

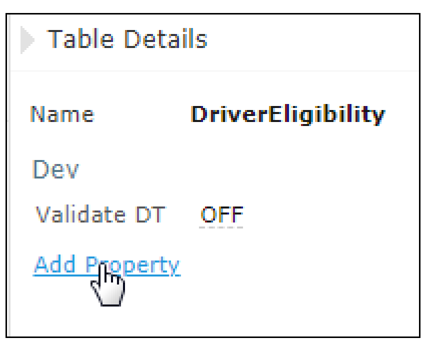



*Figure 84: Add new property for the current table*

2. Enter the desired property or select it from the drop-down list and click the **Add** button.

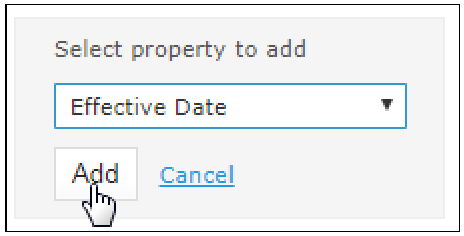



*Figure 85: Selected table property to be added*

3. Specify the property value and then click the **Save** button to complete.
   All steps are collected in the following figure:


*Figure 86: Saving a new property for the current table*

## 5.2 Using Table Versioning

The table versioning mechanism is based on copying the existing table and is initiated in OpenL Tablets WebStudio by clicking the **Copy** button. Then select **New Version** in the **Copy as** list, enter the data as needed and click **Copy** to save.

A new table version has the same identity, that is, signature and dimensional properties of the previous version. When a new table version is created, the previous version becomes inactive since only one table version can be active at a time. By default, all tables are active. The following is an example of an inactive table version.

| SimpleRules DriverType DriverAgeType (Gender gender, Integer age) | | |
|---|---|---|
| properties | version | 0.0.2 |
| | active | true |
| **Gender** | **Age** | **Driver Status** |
| Male | <25 | Young Driver |
| Female | <20 | Young Driver |
| | 71+ | Senior Driver |
| | | Standard Driver |

*Figure 87: An inactive table version*

Versions of the same table are grouped in the module tree under the table name. Clicking the table name displays the active version. If all tables are set to inactive, the latest created version is displayed.

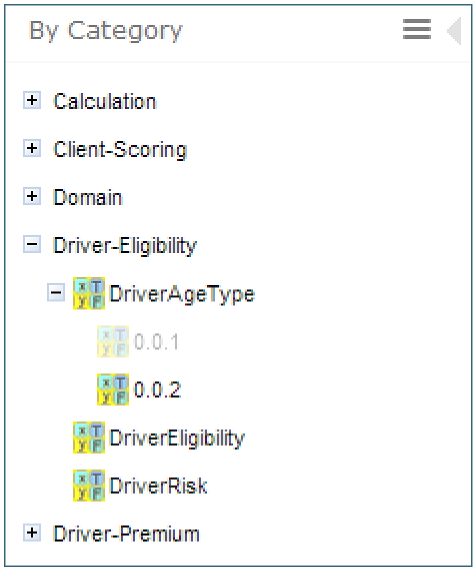



*Figure 88: Displaying table versions in the module tree*

The table version is defined in a three digit format, such as 4.0.1. Table versions must be set in an increasing order.

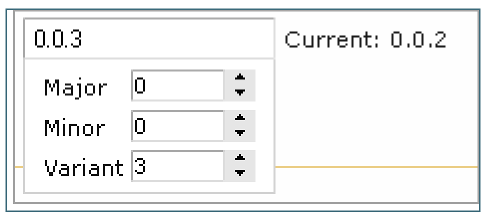
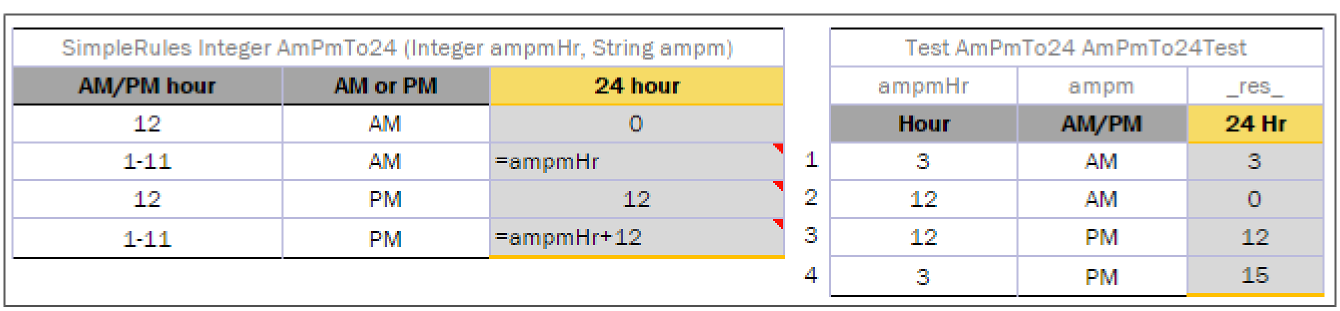



*Figure 89: Entering a new version number*

## 5.3 Performing Unit Tests

Unit tests are used in OpenL Tablets to validate data accuracy. OpenL Tablets Test tables with predefined input data call appropriate rule tables and compare actual test results with predefined expected results.

For example, in the following diagram, the table on the left is a decision table but the table on the right is a unit test table that tests data of the decision table:



| SimpleRules Integer AmPmTo24 (Integer ampmHr, String ampm) | | |
|---|---|---|
| **AM/PM hour** | **AM or PM** | **24 hour** |
| 12 | AM | 0 |
| 1-11 | AM | =ampmHr |
| 12 | PM | 12 |
| 1-11 | PM | =ampmHr+12 |

| | Test AmPmTo24 AmPmTo24Test | | |
|---|---|---|---|
| | ampmHr | ampm | _res_ |
| | **Hour** | **AM/PM** | **24 Hr** |
| 1 | 3 | AM | 3 |
| 2 | 12 | AM | 0 |
| 3 | 12 | PM | 12 |
| 4 | 3 | PM | 15 |

*Figure 90: Decision table and its test table*

OpenL Tablets WebStudio supports visual controls for creating and running project tests. Test tables can be modified like all other tables in OpenL Tablets WebStudio. For information on modifying a table, see Modifying Tables. Test results are displayed in a simple format directly in the user interface.

The following topics are included in this section:

- Adding Navigation to a Table
- Running Unit Tests
- Creating a Test

## Adding Navigation to a Table

OpenL Tablets WebStudio adds a view navigation link to the appropriate test table and vice versa. See the following example:

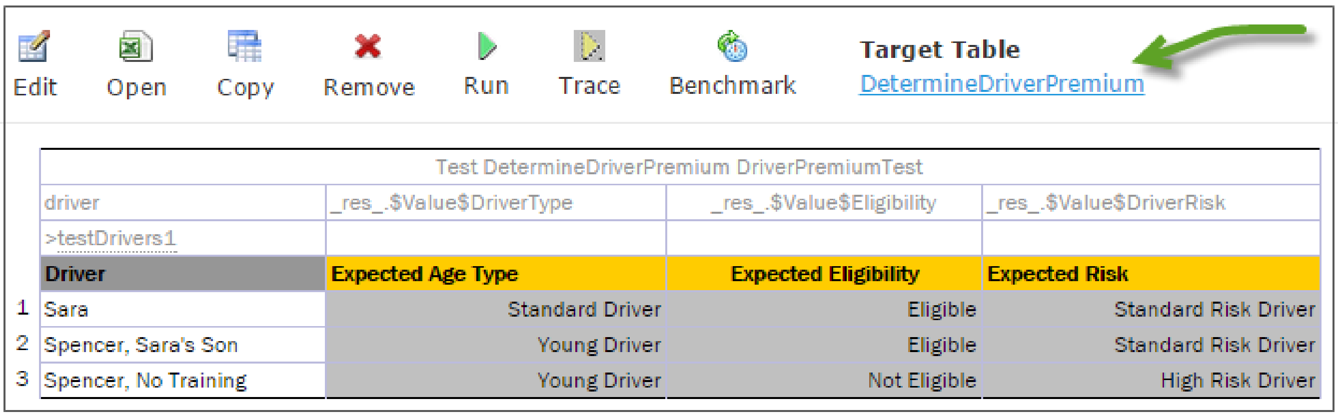



*Figure 91: Navigation link to target table*

## Running Unit Tests

This section provides the methods used to run unit tests. The following topics are included in this section:

- Executing All Module Tests at Once
- Executing Tests for a Single Table
- Displaying Failures Only
- Displaying Compound Result

### Executing All Module Tests at Once

The system automatically executes all test runs, test cases, in every unit test in a module, including tests in module dependencies, and displays a summary of results.

Test results display resembles the following sample:

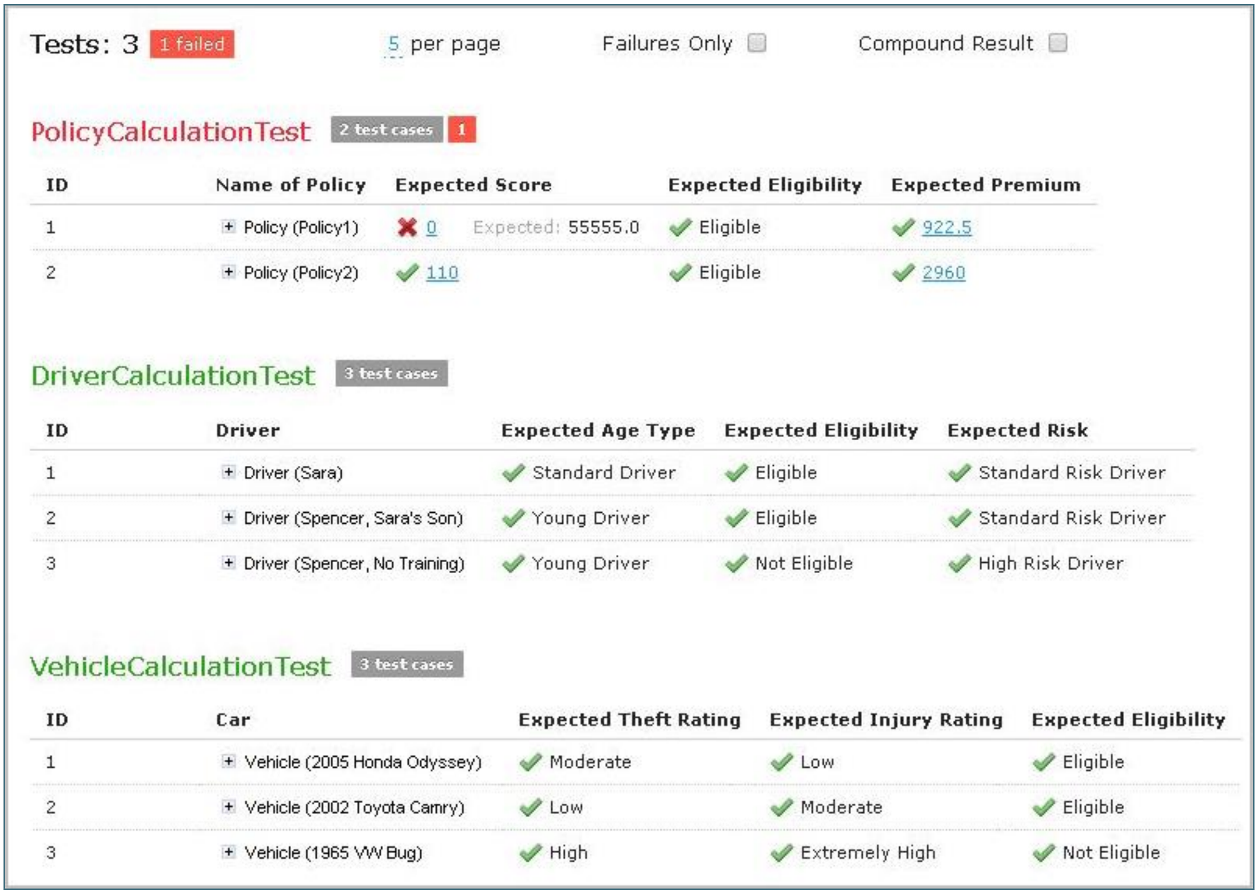



*Figure 92: Results of running all project tests*

To run all module tests, click the **Run Tests** 

icon in the top line menu of Rules Editor.

Failed test cases are represented by  mark. Passed tests are represented by  mark.

**Note:** If all tests are run in Multi-module mode, the system executes all tests of the project, including project dependencies.

In the example above, test results are displayed with five test tables, unit tests, per page. This setting is configured for each user individually in User Profile as **Tests per page** setting.

To change the setting for a particular test run without updating user settings, click the arrow to the right of the **Run Tests**  and choose a desired number of **Tests per page**. There is an alternative way: the same setting options are displayed on the top of the window after executing all tests. The following picture provides an illustration:

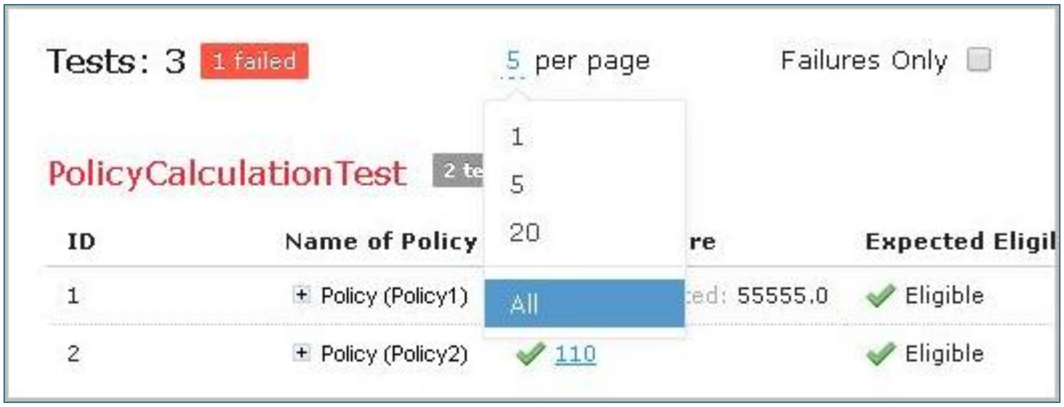



*Figure 93: Number of tests per page setting*

To export test results into an Excel file, in the **Run** or **Test** drop-down menu, select **Run into File** or **Test into File.** The generated file contains both results and input parameters.

## Executing Tests for a Single Table

This section describes test execution. Proceed as follows:

1. To execute all test runs for one particular rule table, select the rule table in the module tree and, in the upper part of the middle pane, click **Test** ☑.

   Test results resemble the following:


*Figure 94: Results of executing all test runs for one rule table*

2. To test a rule table even if no tests have been created for the given table yet, proceed as follows:
   1. In the module tree, select the required rule table and click the green **Run** arrow ▷ above the table.

      The form for entering required values to test rule table appears.

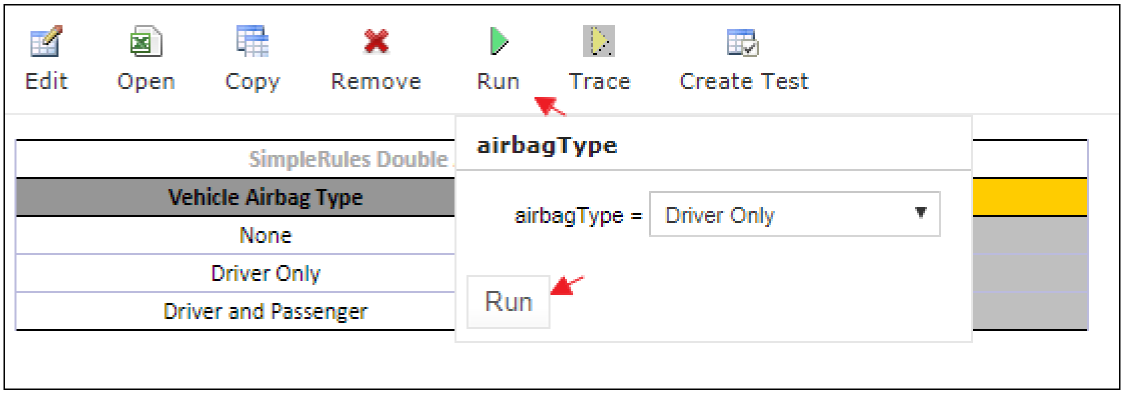



*Figure 95: Testing a rule table without tests*

2. In the pop-up window, click **Run**.
   The results of the testing are displayed.

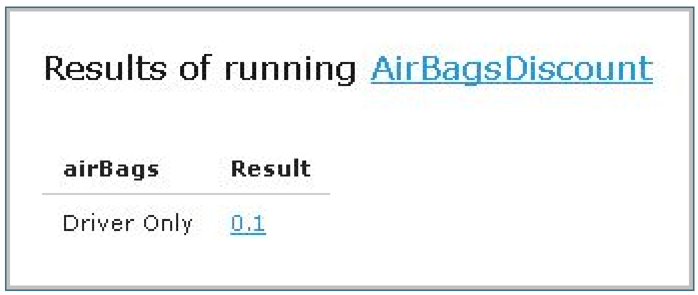



*Figure 96: Result of running virtual test*

3. For Test tables, to select test cases to be executed, proceed as follows:
   1. Navigate to the **Run** button above the Test table and click the small black arrow .
   2. In the pop-up window that appears, select or clear the check boxes for the appropriate IDs, and to run several particular test cases, define them in the **Use the Range** field.

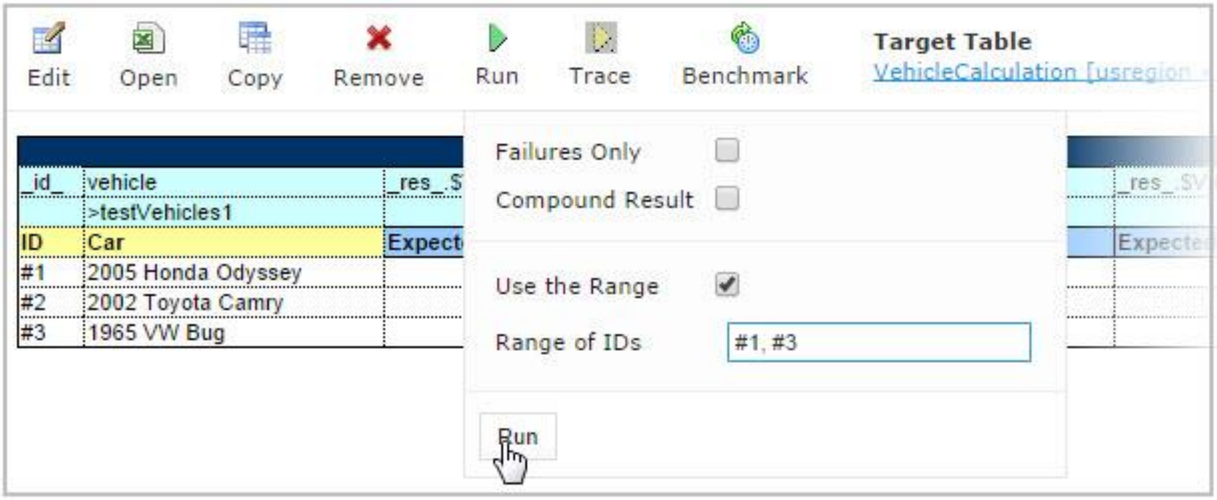



*Figure 97: Select test cases via Range field to be executed*

3. In the pop-up window, click **Run**.
   Only the selected test cases are executed.

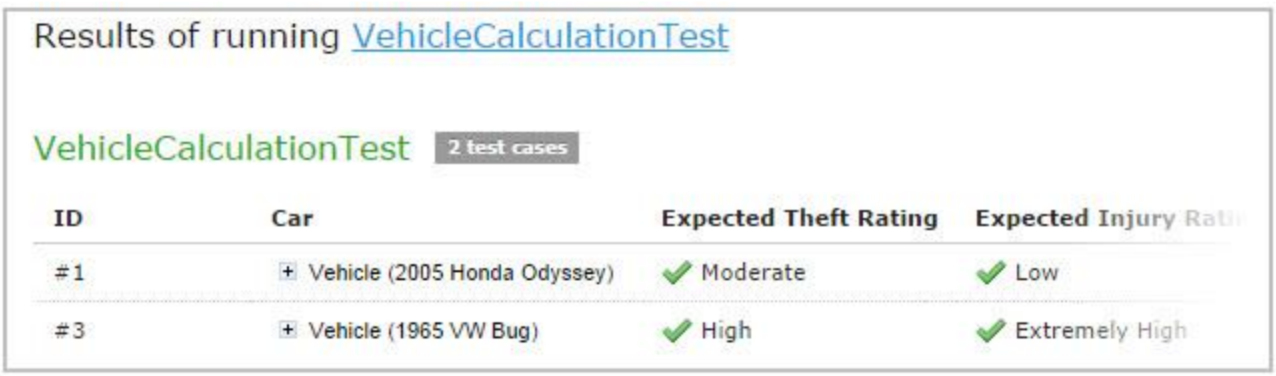



*Figure 98: Result of selective testing*

4. To export test results into an Excel file, click **Test** and select **Test into File.**

## Displaying Failures Only

There are cases when a user wants to examine results of failed test cases only. For example, the project contains a test with more than 50 test cases and a user just needs to know whether project rules are operating correctly, that is, whether all test cases are passed. If a user runs the test, a huge table of results is returned, which is difficult to review and find failures to correct the rule or case. For such situations, OpenL Tablets WebStudio provides an option to display failed test cases only.

This option is configured for each user individually in User Profile as the **Failures Only** setting. There are multiple ways to change the setting for a particular test run without updating user settings:

- Click the arrow to the right of the **Run Tests**  and in a pop-up window that appears, clear or select **Failures only**.

- Select the Test table, navigate to the **Run** button above the table, click the **Run** arrow , and in the pop-up window that appears, select or clear **Failures only**.
- Select or clear the **Failures only** setting that appears on the top of the window upon executing all tests at once as displayed in Figure 93: Number of tests per page setting.

Additionally, the number of failed test cases displayed for one unit test can be limited. For example, a user is testing rules iteratively and is interested just in the first several failures in order to analyze and correct them, and re-execute tests, sequentially correcting errors. To do this, change **All** on an appropriate value next to **Failures per test** label or **first** label (for method 3). The setting is available only if **Failures only** is selected.

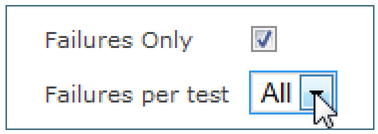



*Figure 99: Settings for displaying failed test cases only*

## Displaying Compound Result

The result of a rule table execution can be a single value or compound value such as spreadsheet. A test table specifies what is tested, full result or particular parts of it, and their expected results of each test case. In the following example, *IncomeForecastTest* is intended to check Minimal and Maximal Total Salary values in the resulting spreadsheet:

| Test IncomeForecast IncomeForecastTest | | | |
|---|---|---|---|
| bonusRate | sharePrice | _res_.$TotalAmount$MinSalary | _res_.$TotalAmount$MaxSalary |
| **Bonus Rate** | **Share Price** | **Min Total Salary** | **Max Total Salary** |
| 15% | $15 | $94,500 | $108,675 |
| 10% | $25 | $94,500 | $103,950 |
| 5% | $35 | $94,500 | $99,225 |

*Figure 100: Testing tables with compound result on*

After running the test, OpenL Tablets WebStudio displays each test case with input values and actual results marked as passed or failed.

IncomeForecastTest 3 test cases

| ID | Bonus Rate | Share Price | Min Total Salary | Max Total Salary |
|---|---|---|---|---|
| 1 | 0.15 | 15 | ✔ 148500 | ✔ 185775 |
| 2 | 0.1 | 25 | ✔ 148500 | ✔ 188350 |
| 3 | 0.05 | 35 | ✔ 148500 | ✔ 190925 |

*Figure 101: Testing spreadsheet result*

In cases when test result is complex (compound), there is an option to display the full result of running test cases as well, not only values which are being tested. It is configured for each user individually in User Profile as "**Compound Result**" setting. If the option is switched on, the result of running *IncomeForecastTest* looks as follows:

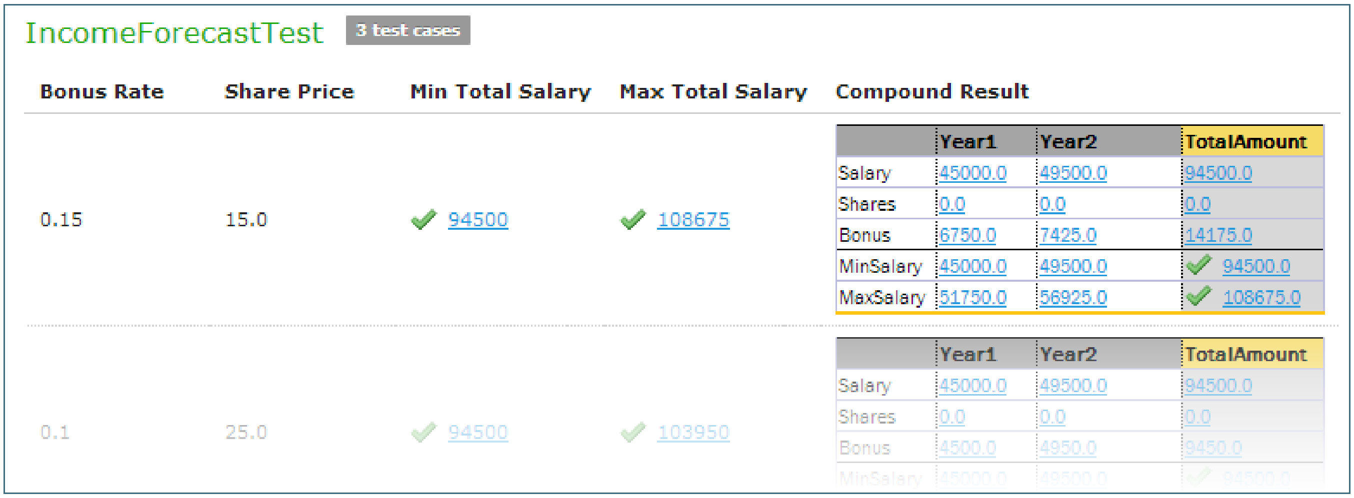



*Figure 102: Displaying compound result*

This setting for a particular test run (without updating user settings) can be changed in the same ways as it is described in Displaying Failures Only.

## Creating a Test

OpenL Tablets WebStudio provides a convenient way to create a new test table.

When an executable table, such as Decision, Method, Spreadsheet, ColumnMatch, or TBasic table, is created, the **Create Test** item becomes available.

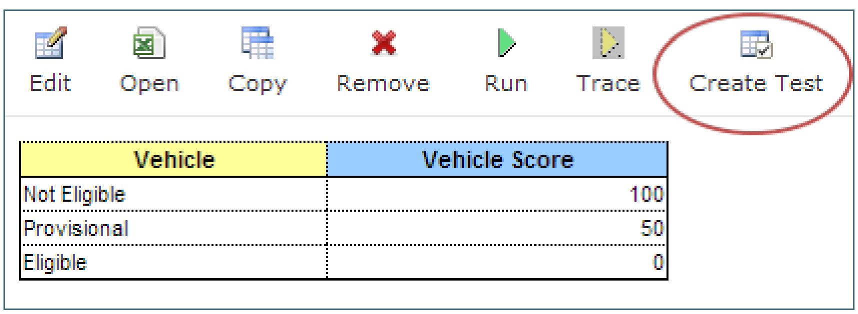



*Figure 103: Create new test table*

Proceed as follows:

1. To create a Test table for the current table, click the **Create Test** button.
   OpenL Tablets WebStudio runs a two-step wizard for creating an appropriate Test table.
2. Enter test input values and expected result values to complete the Test table.

# 5.4 Tracing Rules

OpenL Tablets WebStudio provides a rule tracing view for all appropriate OpenL Tablets methods. These methods include the following:

- All test tables
- All Rule tables with the possibility of specifying input parameters

- Method tables with preset parameters

Tracing of a rule enables users to determine how results for complex rules are obtained.

**Note:** Before tracing, ensure that the browser does not block pop-up windows. Otherwise, trace results will not be displayed. For more information on how to unblock pop-up windows, refer to the specific browser Help.

When using the tracing functionality, users can check the result of each step of the rule and how the result was obtained without creating test cases. For that, perform the following steps:

1. In Rules editor, open a rule table to be traced and click **Trace** in the middle pane.
2. Enter parameters to be traced in the pop-up window.

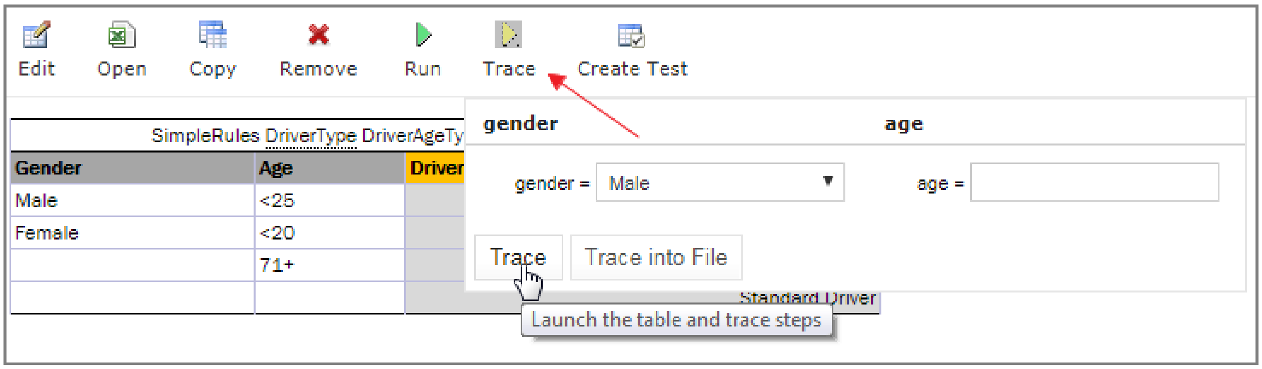



*Figure 104: Tracing a rule for a rule table*

3. To use JSON data and prefill fields with data extracted from log or provided by developers, select **JSON.**

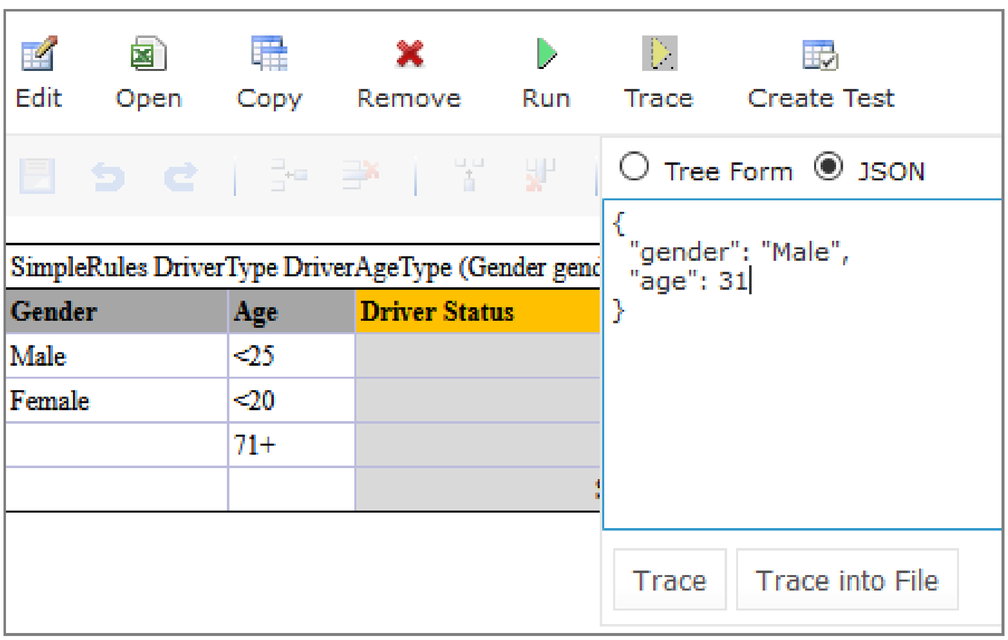



*Figure 105: Selecting the JSON option for tracing*

The same functionality is available for running rules.

4. Click the **Trace** button.

If there is a set of test cases and the result of each step of the rule and how the result was obtained need checking, trace the Test table as follows:

1. Open the desired Test table and hover the mouse pointer over the **Trace** button.
2. To open a pop-up with test cases to be traced, click the small right-hand black arrow.

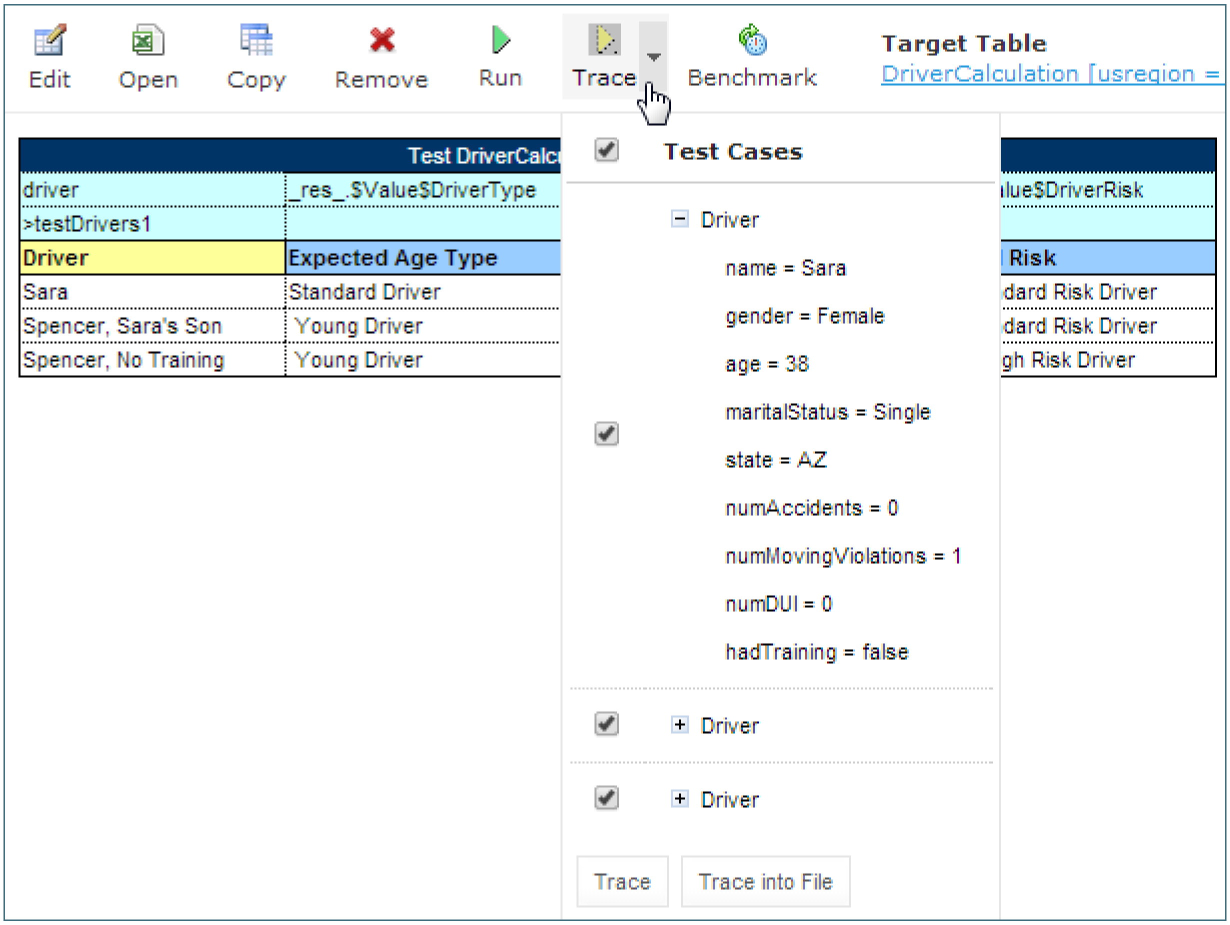



*Figure 106: Tracing of a Test table*

3. In the pop-up, select or deselect the test cases as needed.
   By default, all cases are selected. All test cases can be checked or unchecked by using the checkbox on the left of **Test Parameter(s)**.
4. Click **Trace** to start the process.
   The system displays the tracing results in a separate browser window as illustrated in the following example:

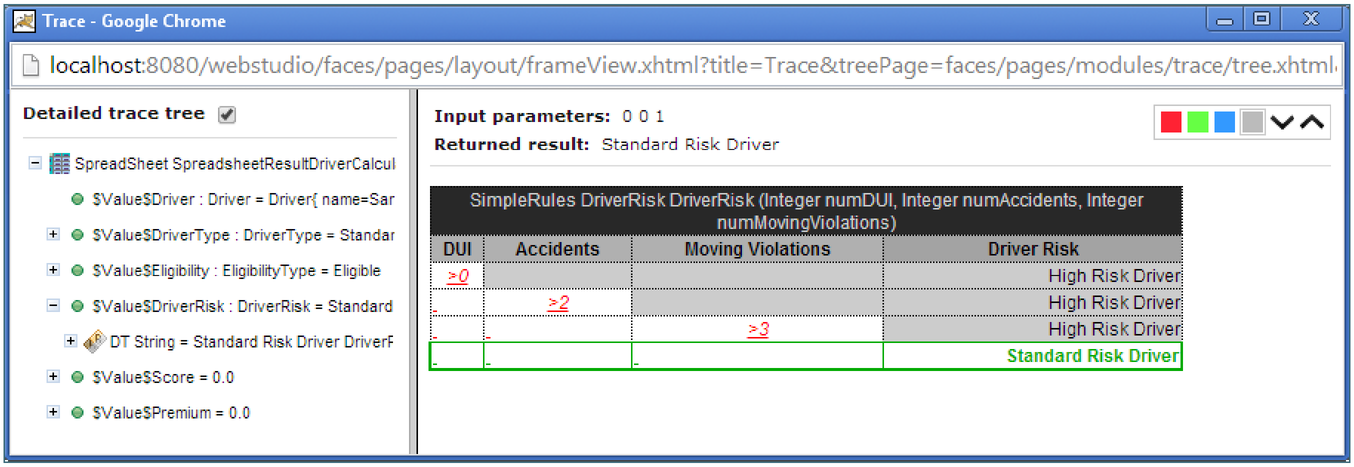



*Figure 107: Tracing results*

The left side displays a tree consisting of rule tables as tree nodes and fired rule rows as tree leaves. Selected **Detailed trace tree** option enables to view all rule calls.

- If that option is cleared, only successful calls will be displayed.

  This option can only be used for a Decision table or if a Decision table is used in complex rules.
- If an element in the tree is selected, the corresponding rule table is displayed in the right pane.

  The fired rule rows are highlighted using the specified color. The highlight color and transparency level can be configured by clicking the  buttons above the rule table. Note that the gray button is selected by default.

In addition, the right pane displays the actual parameters used in the particular rule call and the returned result.

The example above demonstrates the results of tracing Decision table. For other rule tables, the picture slightly differs but the meaning is essentially the same.

For a Decision table, the tracing results are shown as follows:

- The rules that were traced are not highlighted and appear as white rows.
- Successfully completed or returned rules are boxed with green lines.
- The failed rules are displayed in red.

# 5.5 Using Benchmarking Tools

OpenL Tablets WebStudio provides benchmarking tools for measuring execution time for all appropriate OpenL Tablets elements. In OpenL Tablets, everything that can be run can be benchmarked too. Benchmarking is useful for optimizing the rule structure and identifying critical paths in rule calculation.

The benchmarking icon is displayed above the table to be traced.

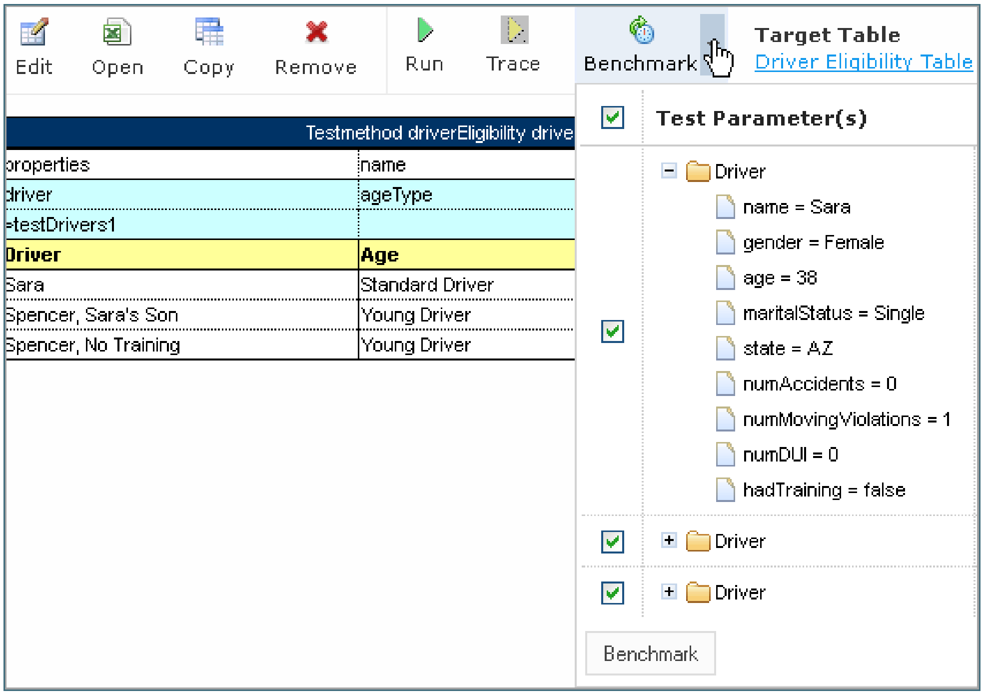


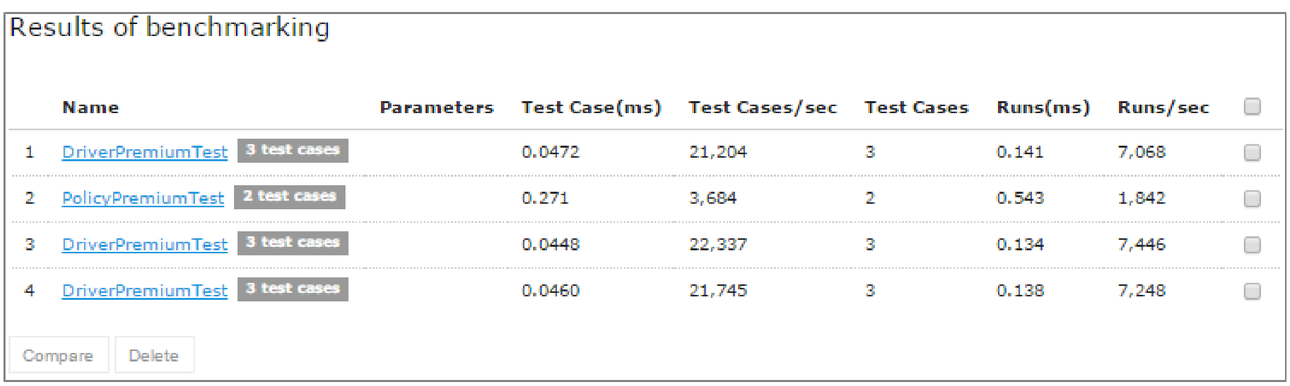

*Figure 108: Controls for measuring performance*

For a Test table, select the test cases as follows:

1. Open the desired Test table.
2. Navigate to the **Benchmark** button above the Test table and click the small right-hand black arrow to open a pop-up with test cases as needed.
3. Select or deselect the test cases as needed.

   By default, all cases are selected. All test cases can be also checked or unchecked by using the checkbox on the left of **Test Parameter(s)**.
4. Click the **Benchmark** button within the pop-up.

   Clicking the benchmarking icon runs the corresponding method or set of methods and displays the results in a table.



Results of benchmarking

| | Name | Parameters | Test Case(ms) | Test Cases/sec | Test Cases | Runs(ms) | Runs/sec | |
|---|---|---|---|---|---|---|---|---|
| 1 | DriverPremiumTest 3 test cases | | 0.0472 | 21,204 | 3 | 0.141 | 7,068 | |
| 2 | PolicyPremiumTest 2 test cases | | 0.271 | 3,684 | 2 | 0.543 | 1,842 | |
| 3 | DriverPremiumTest 3 test cases | | 0.0448 | 22,337 | 3 | 0.134 | 7,446 | |
| 4 | DriverPremiumTest 3 test cases | | 0.0460 | 21,745 | 3 | 0.138 | 7,248 | |

Compare Delete

*Figure 109: Benchmarking results*

Benchmark is displayed using the following parameters:

| Benchmarking results parameters | |
|---|---|
| **Parameter** | **Description** |
| Test Case (ms) | Time of one test case execution, in milliseconds. |
| Test Cases/sec | Number of such test cases that can be executed per second. |
| Test Cases | Number of test cases in a Test table. |
| Runs (ms) | Time required for all test cases of the table, or rule set, execution, in milliseconds. |
| Runs/sec | Number of such rule sets that can be executed per second. |

OpenL Tablets WebStudio remembers all benchmarking runs executed within one session. Every time a new benchmark is run, a new row is added to the results table.

Benchmarking results can be compared to identify the most time consuming methods. Select the required check boxes and click **Compare** to compare results in the results table.

Comparison results are displayed below the benchmarking table.

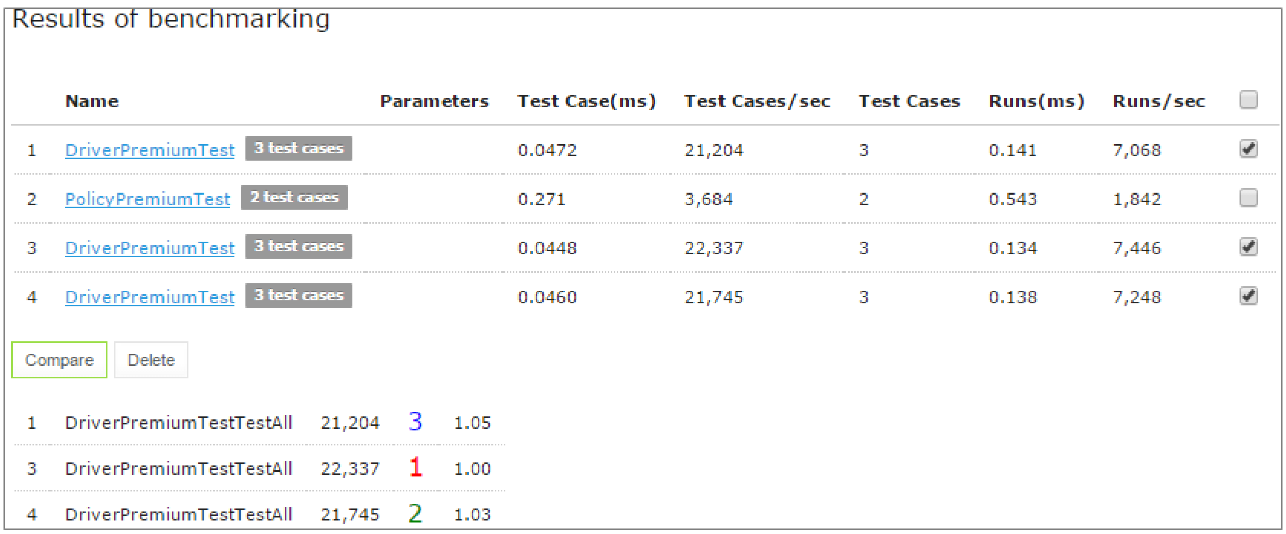



*Figure 110: Comparing benchmarking results*

# 6 Using Repository Editor

This chapter describes tasks that can be performed in repository editor. For general information on repository editor, see Introducing Repository Editor.

The following topics are included in this chapter:

- Browsing Design Repository
- Filtering the Project Tree
- Creating Projects in Design Repository
- Opening a Project
- Closing a Project
- Saving a Project
- Viewing a Project Properties
- Modifying Project Contents
- Copying a Project
- Removing a Project
- Deploying Projects
- Comparing Project Versions
- Exporting a Project or a File
- Unlocking a Project
- Browsing the Deployment Repository

## 6.1 Browsing Design Repository

Repository editor displays all projects in user's workspace and Design repository. The project tree is organized into the following categories:

| **Categories in the project tree** | |
|---|---|
| **Category** | **Description** |
| **Projects** | Contains OpenL Tablets rule projects. |
| **Deploy Configurations** | Contains deploy configurations for deploying rule projects to deployment repository.<br>For information on using deploy configurations, see Deploying Projects. |

The status of each project in the tree is identified by a specific icon. The following table describes the icons in the project tree:

| **Project icons in repository editor** | |
|---|---|
| **Icon** | **Description** |
|  | Project is closed.<br>It is available only in Design repository and must be opened to copy it to user's workspace. |
|  | Project is opened for viewing.<br>It is copied to user's workspace and can be modified. |
|  | Project is edited by the current user.<br>It is copied to user's workspace and is modified. Other users cannot edit the project. To save changes, the project must be saved. |

<table>
<tr><th colspan="2">Project icons in repository editor</th></tr>
<tr><th>Icon</th><th>Description</th></tr>
<tr><td></td><td>Project is closed by the current user but edited by another user (Closed – Locked).<br>Current user cannot edit the project.</td></tr>
<tr><td></td><td>Project is opened for viewing by the current user but edited by another user (No changes – Locked).<br>Current user cannot edit the project but can browse the project in Rules Editor.</td></tr>
<tr><td></td><td>Project exists only in user's workspace but not in Design repository (Local).<br>Other users do not see this project. User can delete the project or import it into Design repository as described in the Creating Projects in Design Repository.</td></tr>
<tr><td></td><td>Project is marked for deletion.<br>In OpenL Tablets WebStudio, deletion of a project takes place in the following phases:
<table>
<tr><th>Phase</th><th>Description</th></tr>
<tr><td>Deleting a project</td><td>Project is removed from user's workspace and marked for deletion. In this phase, the project can be restored using the undelete function.<br>For information on deleting a project, see Deleting a Project.</td></tr>
<tr><td>Erasing a project</td><td>Deleted project is permanently removed from Design repository. After this phase, the project cannot be restored.<br>For information on erasing a project, see Erasing a Project.</td></tr>
</table></td></tr>
</table>

# 6.2 Filtering the Project Tree

Projects in the repository editor are filtered the same way as in Rules Editor. To filter projects by name, enter the name in the filter text box. All projects matching the name are displayed in the **Projects** list.

An advanced filter can also be applied to the project tree so that only files of particular types are displayed:

1. Click the **Advanced Filter** icon to the right of the filter text box.
2. In the **Advanced Filter** pop-up window, in the **Filter files by extensions** field, enter a list of file extensions separated by semicolon.

   An example is `xls;properties;txt`
3. Select the **Hide deleted projects** option if required.
4. Click **Apply**.

   The project tree is filtered so that only files of the specified extensions are displayed. Project folders are always displayed.

**Note:** To reset the filter, clear the previously entered file extensions and click **Apply**.

# 6.3 Creating Projects in Design Repository

OpenL Tablets WebStudio allows users to create new rule projects in the Design repository in one of the following ways:

| Ways of creating new rule projects | |
|---|---|
| **Way** | **Section** |
| Create a rule project from a template | Creating a Project from Template |
| Create a rule project from Excel files | Creating a Project from Excel Files |
| Create a rule project from a zip archive | Creating a Project from ZIP Archive |
| Import a rule project from workspace | Importing a Project from Workspace |

Whatever the way used, new projects are created in the **No Changes** status that means they are open and can be modified.

# Creating a Project from Template

This section describes how to create a project using a template and includes the following topics:

- Creating a Project Using a Default Repository Template
- Creating a Project Using a Custom Template

## Creating a Project Using a Default Repository Template

This is the easiest way to create a rule project in the Design repository that must be preferably used for demonstration or introductory purposes.

While creating a project from template, use the following template types:

| Template types | |
|---|---|
| **Template type** | **Description** |
| **Simple Templates** | Include the following:<br>• **Sample Project** is a very simple project consisting of one rule table and hence, one Excel file.<br>• **Empty Project** allows creating a project with an empty Excel file. Open the project and create tables as needed. |
| **Examples** | Provide several simple projects demonstrating how OpenL Tablets can be used in various business domains. |
| **Tutorials** | Represents projects designed to familiarize users with OpenL Tablets step-by-step, from simple features and concepts to more complex ones. |

Projects represented as Examples and Tutorials can be used not only to learn how they are organized and work, but also to create user's own projects from them.

To create a new project from template, proceed as follows:

1. In the top line menu, click **Create Project**.
   The **Create Project from** window appears.
2. Clicks the **Template** tab.

**Note:** This tab is normally selected by default.

   All project templates are organized into three areas: Simple Templates, Examples and Tutorials described above in this topic.
3. Navigate to the desired template and click its name.
   The name appears in the **Project Name** field. The following example demonstrates creating a Simple project:

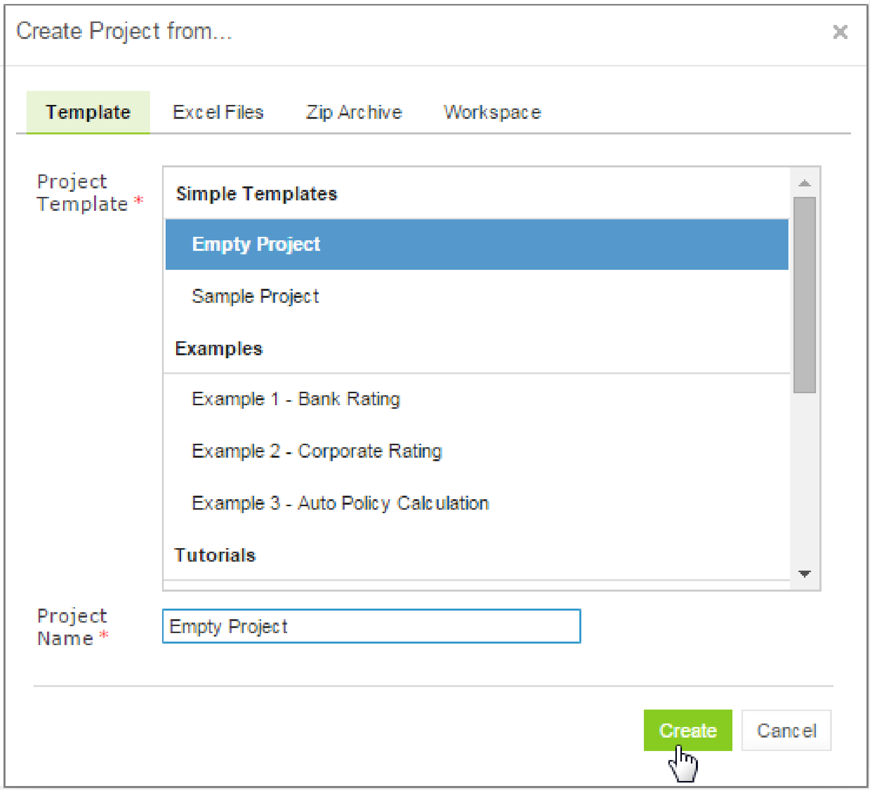



*Figure 111: Creating Simple project from template*

4. Click **Create**.

   A new project is created in Design repository. Initially, project structure corresponds to the selected project template but can be constructed manually.
5. To construct the project structure, add folders and upload files as described in Modifying Project Contents.

## Creating a Project Using a Custom Template

A custom project template can be created and then used during new projects definition. To create a new custom project template, proceed as follows:

1. If the OpenL Tablets WebStudio home directory `\<OPENL_HOME>`, create the following directory:
   `\<OPENL_HOME>\project-templates`
2. Create a subfolder with a template category name.

   An example is `\<OPENL_HOME>\project-templates\My Custom Templates`.
3. For project templates that store files with project rules, create subfolders.

   For example, `\<OPENL_HOME>\project-templates\My Custom Templates\MyRule1\rating.xlsx` will be presented as the **MyRule1** template project in the `My Custom Templates` category containing the `rating.xlsx` file.

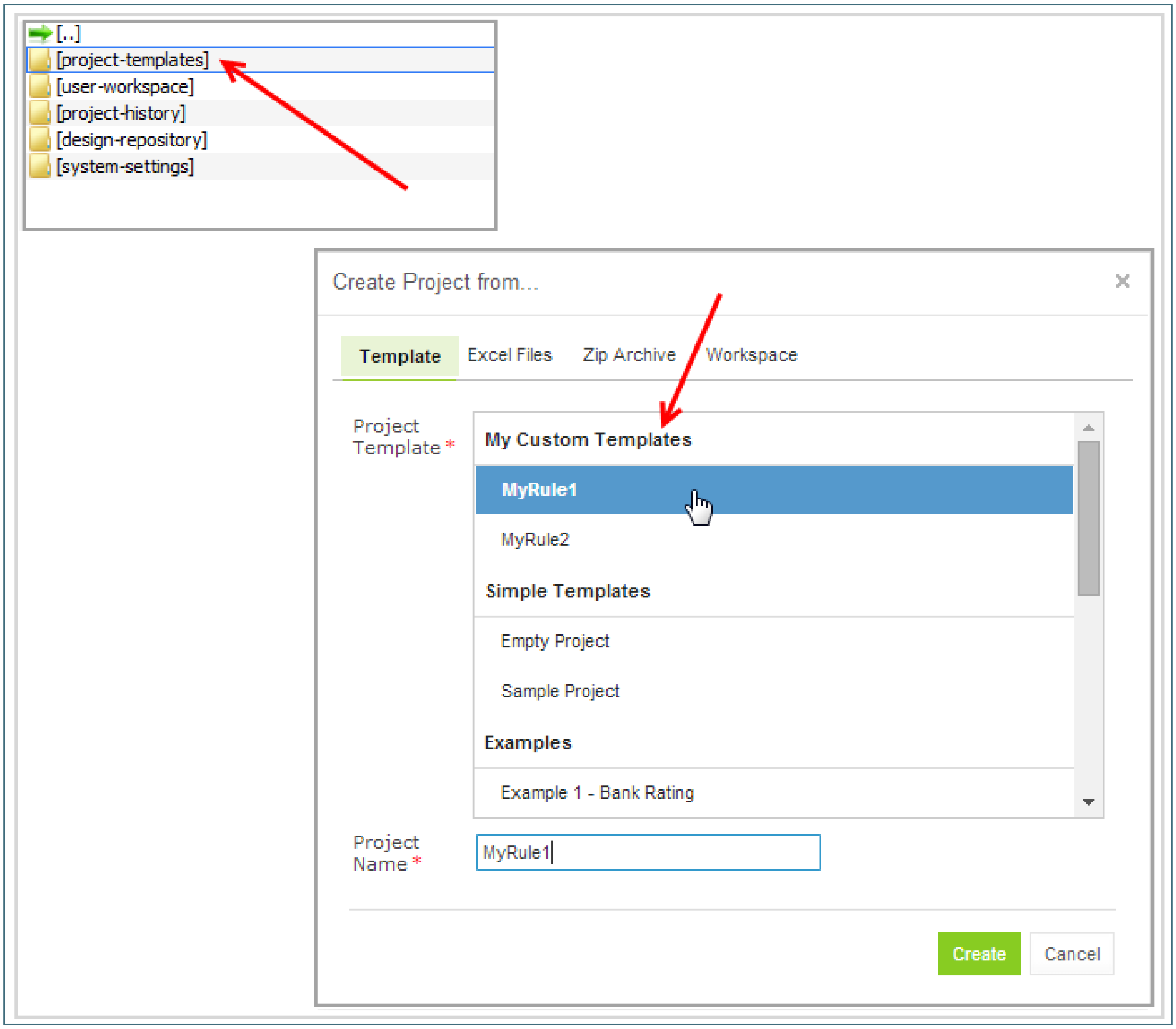



*Figure 112: Creating a custom project template*

## Creating a Project from Excel Files

A rule project in the Design repository can be created by loading one or more Excel files that contain OpenL rule tables or entire rule projects.

Proceed as follows:

1. Click **Create Project** in the top line menu.
2. In the **Create Project from** dialog, click the **Excel Files** tab.
3. Click the **Add** button, locate the desired Excel file in a file system and click **Open**.
4. If required, repeat the previous step to add more files for the project.

   All files will be listed in the **File** area.

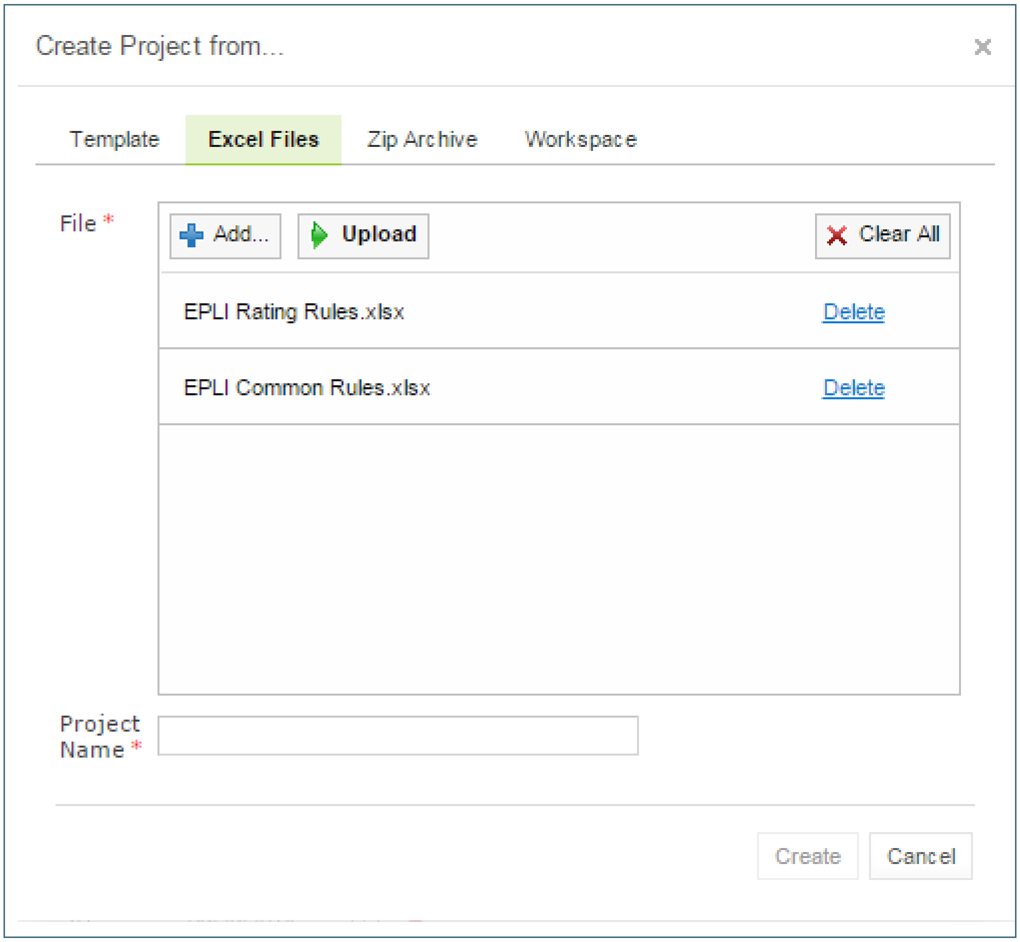



*Figure 113: Creating a project from Excel files*

A file can be removed from the list by clicking the corresponding **Delete** link. To delete all files, click **Clear All**.

5. After adding all the required files, click **Upload** to load the files into the repository.
   Each file can be uploaded separately but it is not recommended.
6. In the **Project Name** field, enter the name by which the project must be represented in Design repository.
7. Click **Create** to complete.

## Creating a Project from ZIP Archive

OpenL Tablets WebStudio provides a control for loading rule projects archived in a ZIP file into Design repository. The procedure is similar to creating a project from Excel files described above although there are a few differences.

A project can only be created from a `zip` archive. The `.rar` or `.7zip` archives cannot be used.

1. Click **Create Project** in the top line menu.
2. In the **Create Project from** dialog, click the **Zip Archive** tab.
3. Click the **Add** button, locate the desired zip archive and click **Open**.

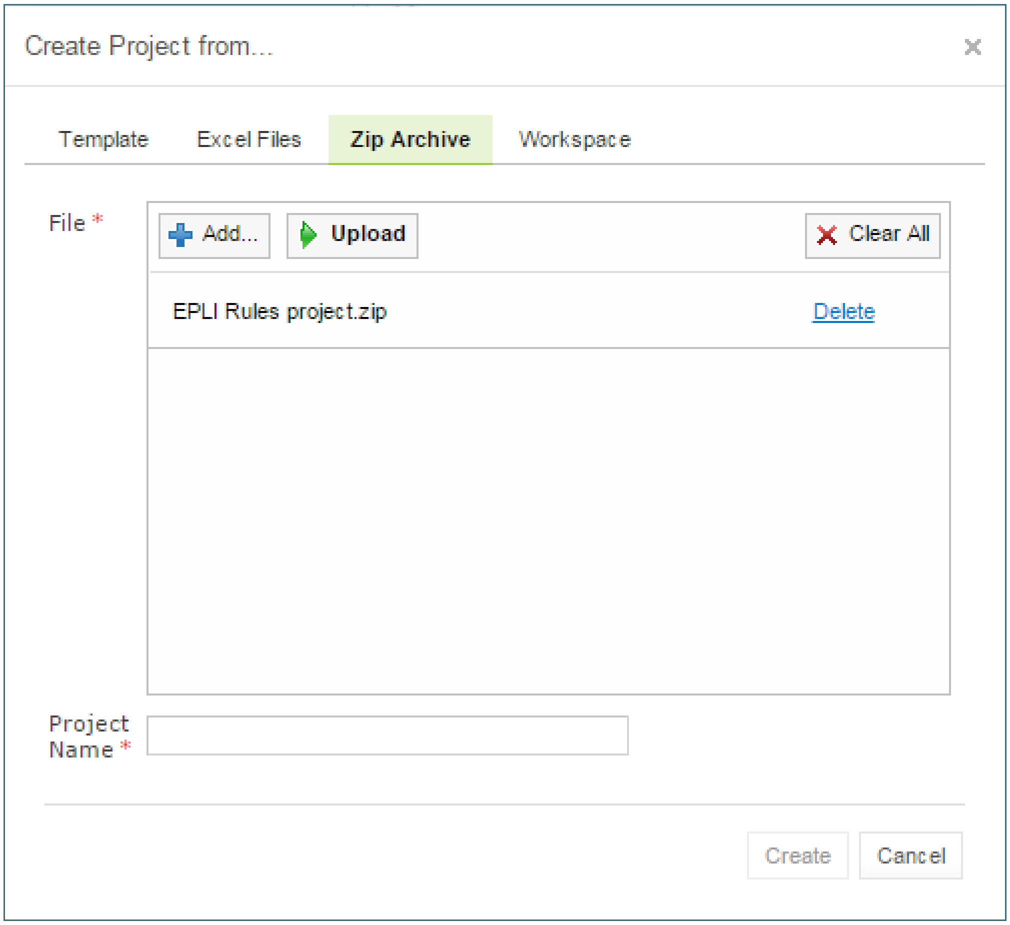



*Figure 114: Creating a project from ZIP file*

4. Click **Upload** to proceed.
5. **Project Name** text box is automatically populated with the project name defined in `rules.xml`, if the uploaded ZIP file contains `rules.xml`, or with the file name.

   The name can be changed, and it will be updated in `rules.xml` accordingly.
6. Click **Create** to complete.

## Importing a Project from Workspace

A new project can be created in Design repository by loading a project with the **Local** status from user workspace.

1. Click **Create Project** in the top line menu.
2. In the **Create Project from** dialog, click the **Workspace** tab.

   The system displays rule projects available in the workspace:

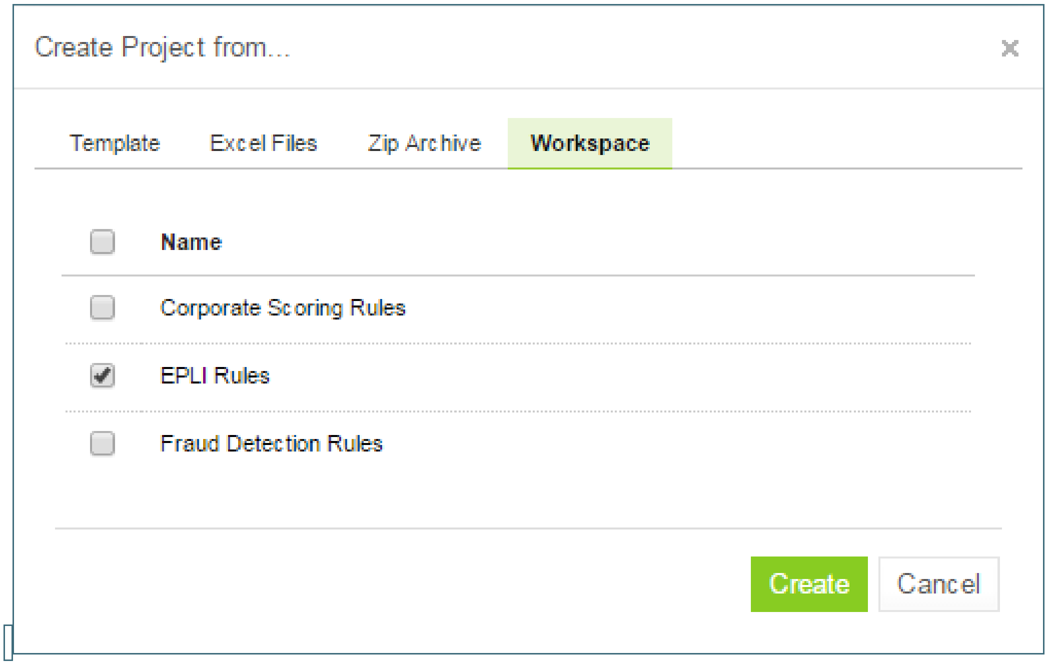



*Figure 115: Creating a project from Workspace*

3. Select checkboxes for projects to be uploaded.
4. To complete creation, click **Create**.

# 6.4 Opening a Project

An opened project is copied to user's workspace and becomes available for selection in Rules Editor. The project is opened for viewing and can be modified if it is not locked by another user. When a user modifies a project, its status is set to **In Editing** and it becomes locked for other users who now can only view it.

To open a project, in the project tree, select the project and, in the right pane, click one of the following buttons as required:

| **Buttons for opening a project** | |
|---|---|
| **Button** | **Description** |
| **Open** | Opens the latest revision of project. |
| **Open Revision** | Displays window where user can specify which project revision must be opened. |

Any project revision can be opened, with the project status set to **Viewing Revision**, as follows:

- Opening a Project Revision Using the Open Revision Button
- Opening a Project Revision Using the Revisions Tab

## Opening a Project Revision Using the Open Revision Button

To open a project revision using the **Open Revision** button, proceed as follows:

1. Click the **Open Revision** button.
2. In the **Project Revisions** field, select the required revision.

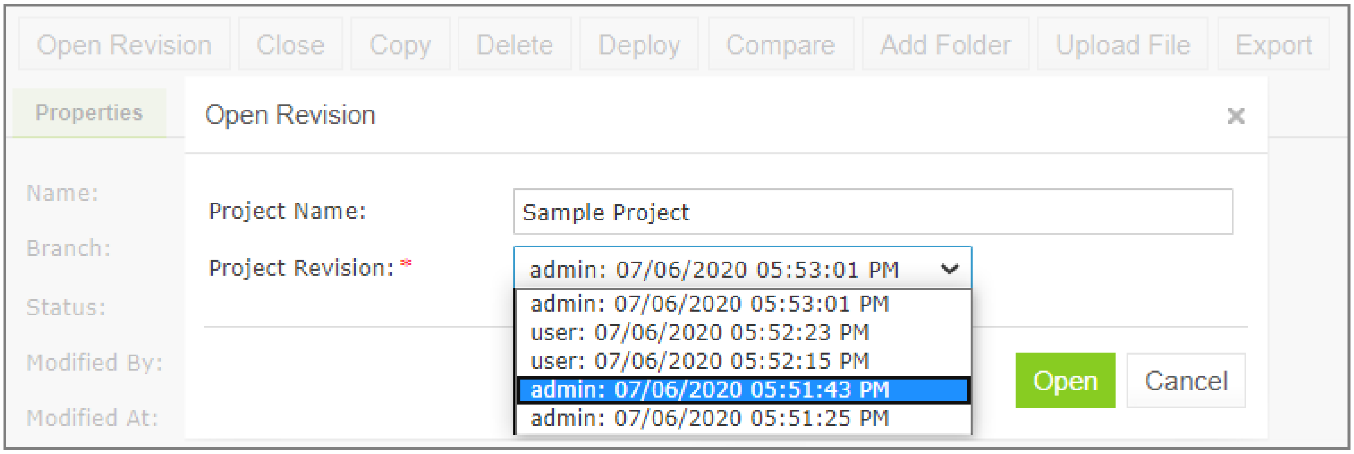



*Figure 116: Opening a project revision using the Open Revision button*

3. Click **Open**.

## Opening a Project Revision Using the Revisions Tab

To open a project revision using the **Revisions** tab, proceed as follows:

1. In the **Projects** tree, select a project.
2. Click the **Revisions** tab.
   A list of revisions appears.


*Figure 117: List of project revisions*

3. Navigate to the revision that needs to be opened and click the corresponding magnifier icon in the **Action** column.
4. In the information message, click **OK**.

If a project has the **Viewing Revision** status, the opened project revision becomes available for viewing and modifying, not the latest revision.

If user tries to modify an old revision of the project, the system displays the warning message, “**You are trying to edit old revision of the project. Do you want to overwrite newer revision?”** When user modifies the old revision, it becomes the current version of the project, and its status changes to **In Editing**.

## 6.5 Closing a Project

Closing a project deletes it from the user's workspace. No changes made to the project will be applied and stored. From that point, the project is not available for selection in Rules Editor. Users can still browse closed projects in repository editor.

To close a project, in the project tree, select the project and, in the right pane, click **Close**.

## 6.6 Saving a Project

A modified project is saved and copied from the user's workspace to Design repository as a new revision.

To save a project, proceed as follows:

1. In the project tree, select the project, and, in the right pane, click **Save**.
   The **Save changes** window appears:


*Figure 118: Save changes in a project*

   The number of a revision is updated automatically and is specified in the **Next Revision** field.
2. Enter comments if needed and click **Save**.

An editable project can be saved and closed directly from Rules Editor as described in [Editing and Saving a Project](#).

## 6.7 Viewing a Project Properties

Each rule project has a set of properties displayed in the **Properties** tab when a project is selected.

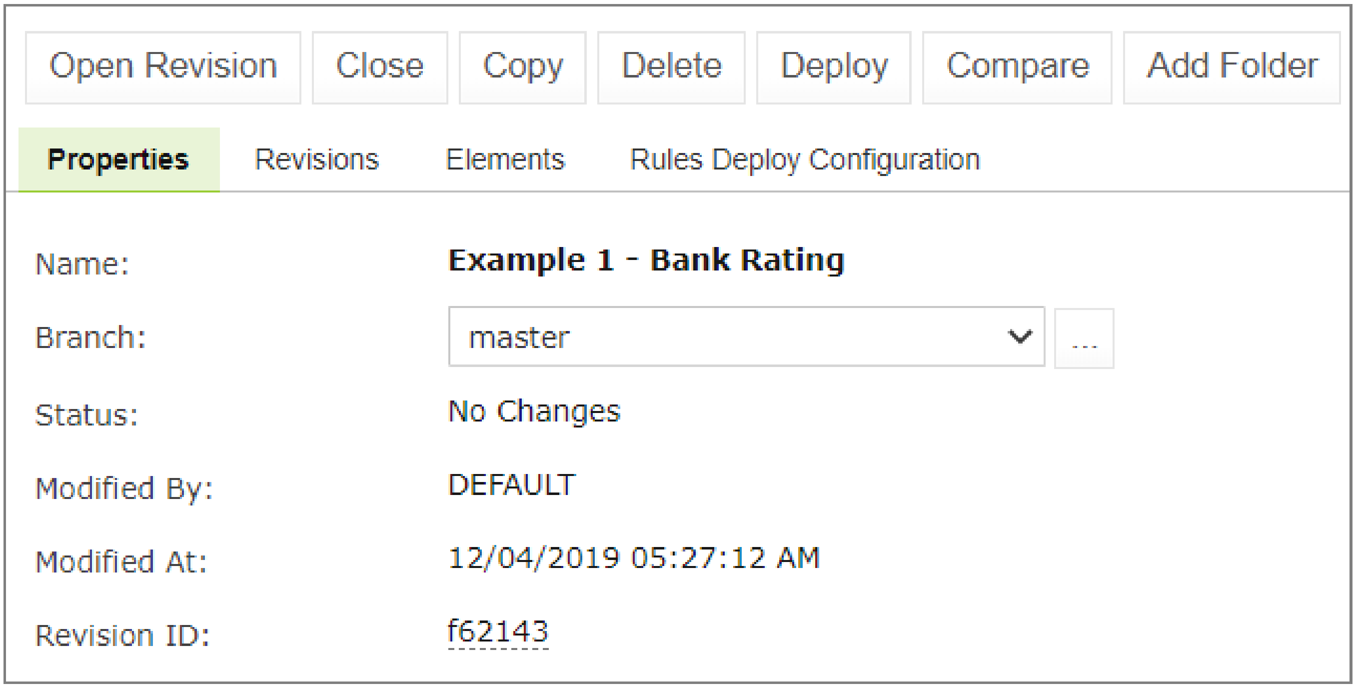



*Figure 119: Project properties*

These properties, such as Name and Created At / Created By, are updated automatically by the system, and users cannot edit them in the OpenL Tablets WebStudio UI.

# 6.8 Modifying Project Contents

This section describes modifying the physical structure of the project and includes the following topics:

- Creating a Folder
- Uploading a File
- Updating a File
- Deleting a Folder or a File
- Copying a File

## Creating a Folder

To create a new folder in the project structure, proceed as follows:

1. In the project tree, select the parent folder in which the new folder must be created.
   To create a root level folder, the project name must be selected in the project tree.
2. In the right pane, click **Add Folder**.
3. In the **Add Folder** window, enter the folder name and click **Add**.

## Uploading a File

To upload a file to a project folder, proceed as follows:

1. In the project tree, select the folder where the file should be uploaded.
   To upload a file to the root level, the project name must be selected in the project tree.

2. In the right pane, click **Upload File**.
   The **Upload File** window appears:

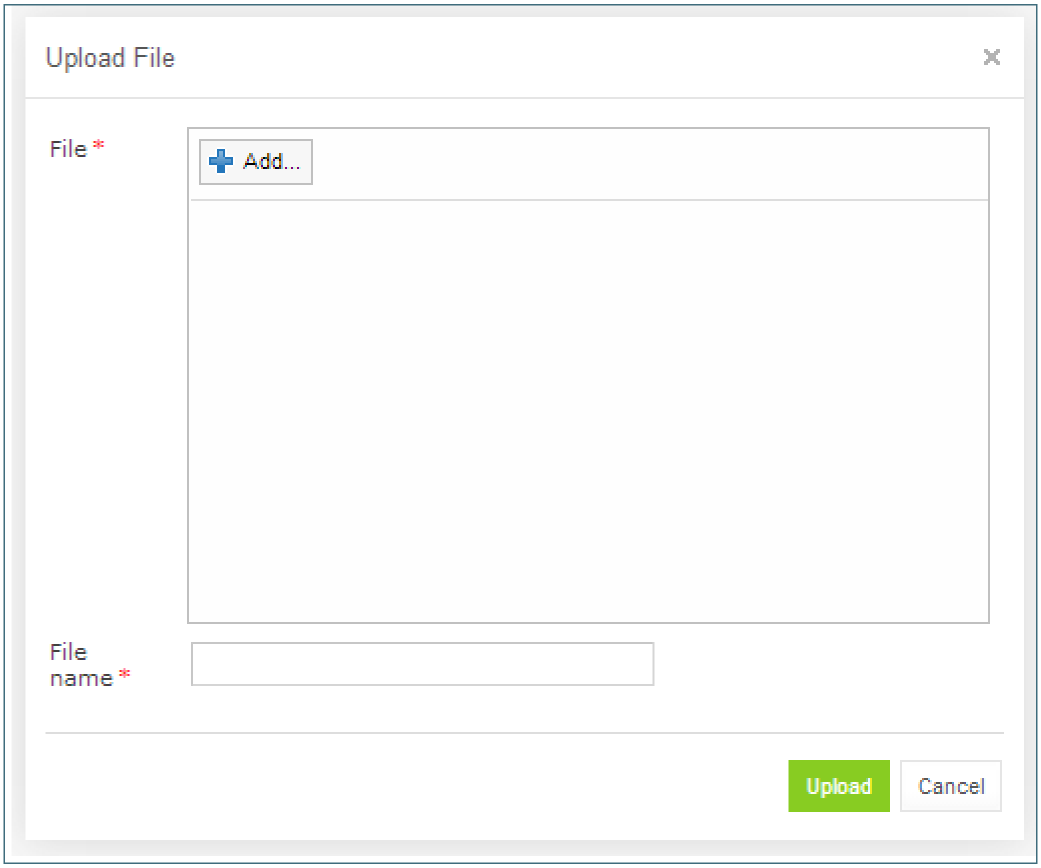



*Figure 120: Uploading a file*

3. Click **Add** in the **File** area and select the file to be uploaded.
4. Click the upper **Upload** button (with a green arrow).
5. In the **File name** field, enter or modify the name of the file to be used in Design repository.
6. Click the **Upload** button at the bottom.

## Updating a File

To update a file of a project via repository editor, proceed as follows:

1. In the project tree, select the file to be updated and, in the right pane, click **Update file**.
2. In the window that displays, click **Add** and choose the required file for updating.
3. Click the **Upload** button to load the file.
4. Click **Update** to end the action.

## Deleting a Folder or a File

To delete a folder or a file in the project structure, proceed as follows:

1. Perform one of the following steps as required:
   - Expand the project tree, select the folder or file to be deleted and, in the right pane, click **Delete**.

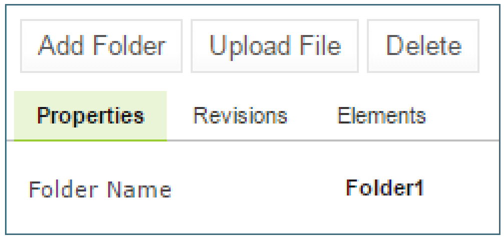



*Figure 121: Deleting a project element*

- To delete an element inside the parent folder, select that folder, click **Elements** to expand the folder and then click **Delete** at the right of the item to be deleted.

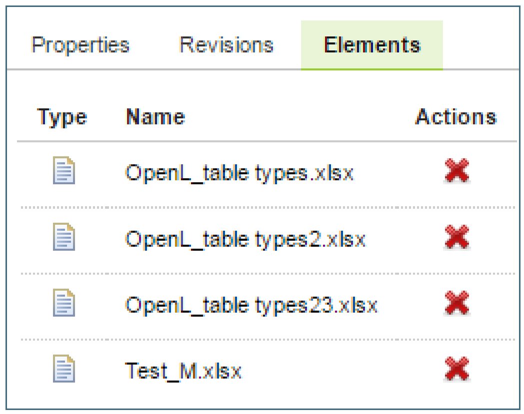



*Figure 122: Deleting project elements from the* ***Elements*** *tab*

2. In the confirmation window, click **OK**.

## Copying a File

A user can create a copy of a file using the repository editor. The current revision of the file or any revision stored in the repository can be used for copying. Proceed as follows:

1. Select a project that contains a file to copy and in the files tree, select the required file.
2. In the upper left corner of the page, click **Copy file** Copy file .
3. In the window that appears, select the **Current Revision** or clear it and in the **File Revision** field, select a value.
4. Optionally, enter the **New File Path** property value.
5. In the **New File Name** field, enter the file name.

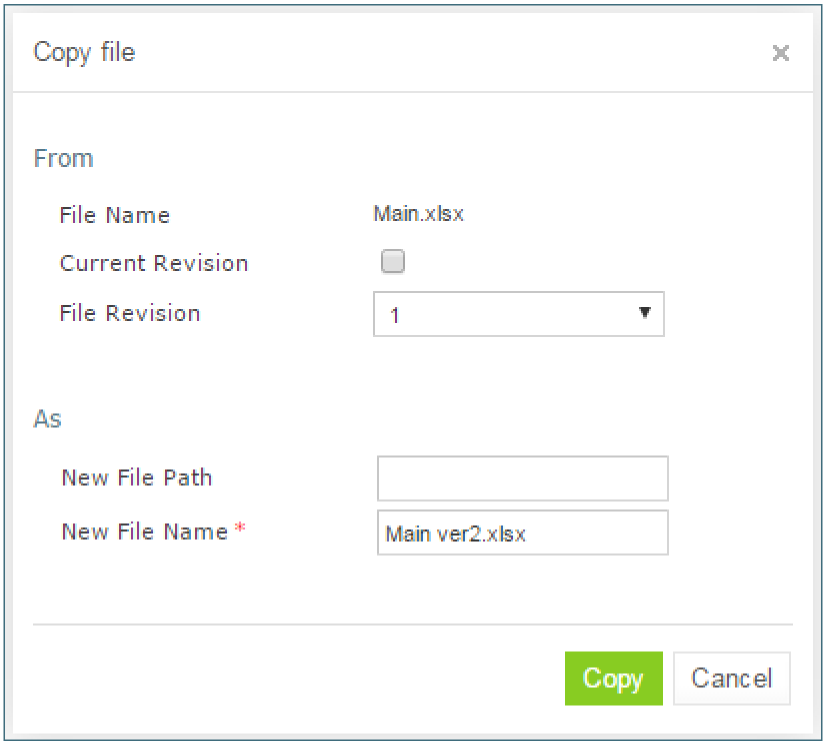



*Figure 123: Copying a file in repository editor*

6. Click **Copy**.
   The newly created file appears in the file tree.

# 6.9 Copying a Project

Copying a project creates a new project with identical contents and a different name in Design repository. This function can be used for copying local projects to Design repository with a different name.

To copy a project, proceed as follows:

1. Perform one of the following steps as required:
   - In the **Projects** tree, select the desired project and, in the right pane, click the **Copy** button.
   - Click **Projects** in Navigator to get a list of projects, navigate to the project you want to copy and click the corresponding **Copy** item  on the right.
2. In the **Copy Project** window, enter the new project name.
3. Select whether a new project must be linked to the origin project.
   In case of linked projects, a new project branch is created. For more information on branches, see Working with Project Branches.
4. Specify whether old revisions must be copied to the newly created project.
5. Optionally, provide comments.
6. Click **Copy**.
   The new project appears in the list of projects.

# 6.10 Removing a Project

Removing a project is executed in the following phases:

- Deleting a Project
- Erasing a Project

## Deleting a Project

A deleted project is removed from user's workspace and marked as deleted in Design repository. All users can see that a project is deleted. Physically, it still remains in Design repository.

**Note:** Projects in the **Local** status that were not uploaded to Design repository will be removed physically and cannot be restored.

To delete a project, proceed as follows:

1. Perform one of the following steps as required:
   - In the **Projects** tree, select the project and, in the right pane, click **Delete**.
   - Click **Projects** in Navigator to get a list of projects, navigate to the project you want to remove and click the corresponding **Delete** item on the right.
2. In the confirmation window, click **Delete** or **OK**.

Deleted projects, except for those in the **Local** status, can be restored by using the **Undelete** button.

To make deleted projects visible, uncheck the **Hide deleted projects** checkbox in the filter pop-up window, which appears after clicking the **Filter** button above the **Projects** tree, and click **Apply**.

To restore a deleted project, proceed as follows:

1. Navigate to the deleted project in the **Projects** tree.
2. Click the **Undelete** button in the right pane.
3. Click **Undelete** in the confirmation window.

## Erasing a Project

Erasing a project permanently removes it from Design repository.

**Warning:** Erased projects cannot be restored.

To erase a project, proceed as follows:

1. Delete the project as described in Deleting a Project.
2. In the **Projects** tree, select the project and, in the right pane, click **Erase**.
3. In the confirmation window, click **Erase**.

# 6.11 Deploying Projects

This section describes tasks related to deploying rule projects to deployment repository.

The following topics are included in this section:

- Creating a Deploy Configuration

- Defining Projects to Deploy
- Deploying a Deploy Configuration
- Opening Deployed Configurations
- Redeploying Projects
- Configuring Additional Rules Deploy Configuration Settings
- Defining Rule Service Version

## Creating a Deploy Configuration

Deployment to deployment repository is performed by using deploy configurations. A deploy configuration is a list of rule projects and specific project revisions to be deployed together to deployment repository. Deploy configurations are useful for recording the history of project deployments.

Deploy configurations are listed in the **Deploy Configurations** tree. Like rule projects, deploy configurations are stored in Design repository and can be versioned.

To create a deploy configuration, proceed as follows:

1. Click **Create Deploy Configuration** in the top line menu.
2. In the **New Deploy Configuration** window, enter the deploy configuration name and click **Create**.
   The new deploy configuration appears in the **Deploy Configuration** tree.
3. Define deploy configuration projects as described in Defining Projects to Deploy.

## Defining Projects to Deploy

A Project to Deploy is a reference to one specific revision of a rule project to be included in the deploy configuration. Project to Deploy must be added to the deploy configuration specifying which rule projects and project revisions are deployed.

To add a new project to deploy to the deploy configuration, proceed as follows:

1. In the **Deploy Configurations** tree, select the deploy configuration and, in the right pane, select the **Projects to Deploy** tab.

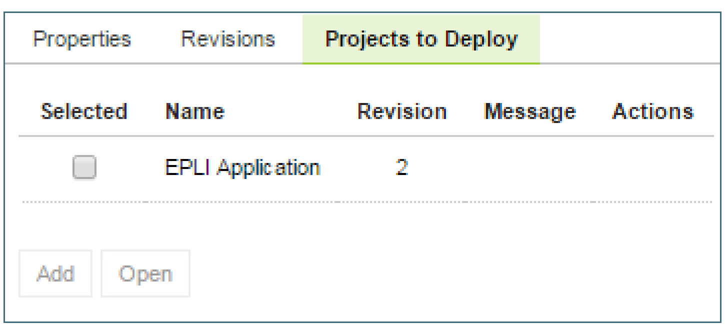



*Figure 124: Deploy configuration with projects to deploy*

   The **Projects to Deploy** tab displays existing projects to deploy of the selected deploy configuration.
2. To add a new project to deploy, click **Add** and specify the project and revision to be included in the deploy configuration.
3. Repeat this procedure to add as many projects as required.

## Deploying a Deploy Configuration

To deploy a deploy configuration, click **Deploy**.

**Note:** The **Deploy** button is disabled if deploy configuration is in the **In Editing** status.

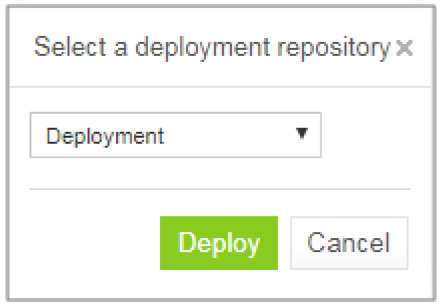



*Figure 125: Deploying configuration to deployment repository*

The specified projects are deployed to deployment repository and a deployment message is displayed.

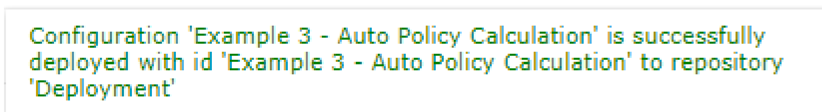



*Figure 126: Deployment message*

**Note:** Deploy configuration cannot be deployed if any dependency projects are missed in it. Check messages on the **Projects to Deploy** tab.

## Opening Deployed Configurations

Deploy configurations provide the means for tracking the deployment history of project revisions. OpenL Tablets WebStudio provides functionality for quickly opening the deployed configuration revisions. This is especially useful when some time has passed since deployment and a review of files during specific deployments is desired.

To open the specific project revisions included in a deploy configuration, proceed as follows:

1. In the **Deploy Configuration** tree, select the deploy configuration.
2. In the right pane, select the **Projects to Deploy** tab.
3. In the **Selected** column, select the check boxes for projects to be opened.
4. Click **Open**.

The selected project revisions are opened in repository editor.

## Redeploying Projects

OpenL Tablets WebStudio provides a function that allows a simple update and redeployment of many related deploy configurations when a particular rule project is modified. This function considers the revision of the opened rule project and works correctly, even with older project revisions.

To update related deploy configurations and redeploy a rule project, proceed as follows:

1. In the Projects tree, select the modified rule project.
2. In the right pane, click **Deploy**.

**Note:** The **Deploy** button is disabled if the selected project has the Local status or if it is edited.

The **Auto Deploy** window appears listing all existing deploy configurations which's latest revision contains a reference to the selected rule project. Deploy configurations marked for deletion are not displayed.

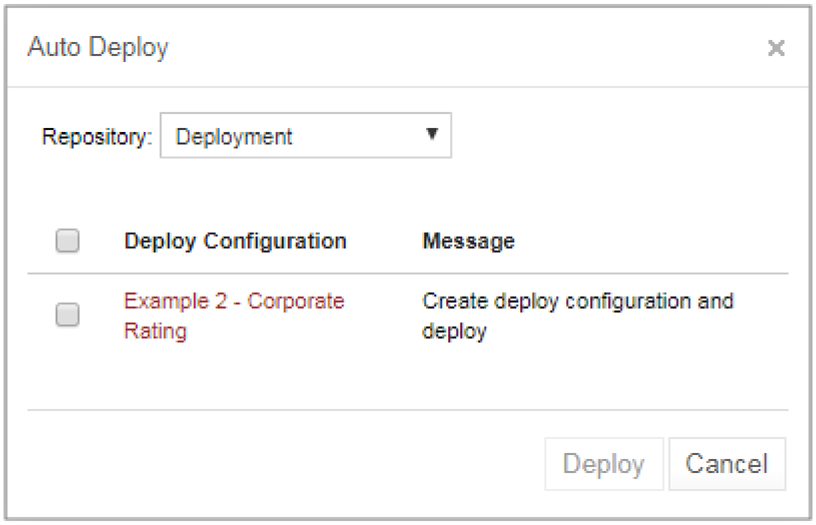



*Figure 127: Deploying a project*

The **Message** column displays the current status of displayed deploy configurations. If a particular deploy configuration cannot be deployed, the check box is gray. Possible reasons for a deploy configuration to be disabled are the following:

- The deploy configuration is saved.
- The deploy configuration is locked by another user and cannot be updated.

If the selected rule project is not referenced by any existing deploy configuration, the system offers to create a new deploy configuration containing only the rule project with an identical name.

3. Select check boxes for the deploy configurations that must be updated and deployed.
4. Click **Deploy**.

   Update and deployment results are displayed in the user interface.

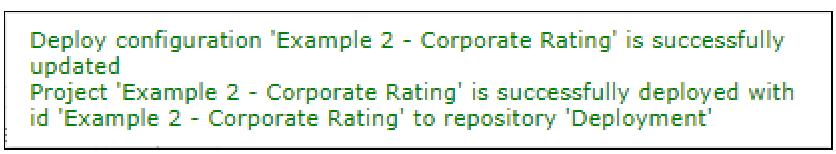



*Figure 128: Redeployment results*

## Configuring Additional Rules Deploy Configuration Settings

Deployment rules can be added before deploying a project to deployment repository. If a project already has the `rules-deploy.xml` configuration file, it can be edited via the **Rules Deploy Configuration** menu.

Proceed as follows:

1. In the top line menu, click **Rules Deploy Configuration**.
2. Click **Create rules deploy configuration**.
3. In the window that appears, enter the following information about the rules:
   - Provide runtime context.

- Use the Rule Service runtime context.
- Define variations.
- Create services specifying the versions of web services to support, which is either the SOAP service, or the RESTful service, or RMI, or all of them.
- Enter the service name.
- Define the service class.
- Define an RMI service class.
- Define the service version. For more information on service version definition, see Defining Rule Service Version.
- Enter URL of the service.
- Define an intercepting template class.
- Define an annotation template class.
- Define comma separated service groups.
- Add configuration description to the XML file.

For more information on the **Rules Deploy Configuration** tab settings configuration, see ***[OpenL Tablets Rule Services Usage and Customization Guide]***, section **OpenL Tablets Rule Services Customization > Service Configurer**.

4. Click **Save Configuration**.

The selected rules are displayed in the **Rules Deploy Configuration** tab.

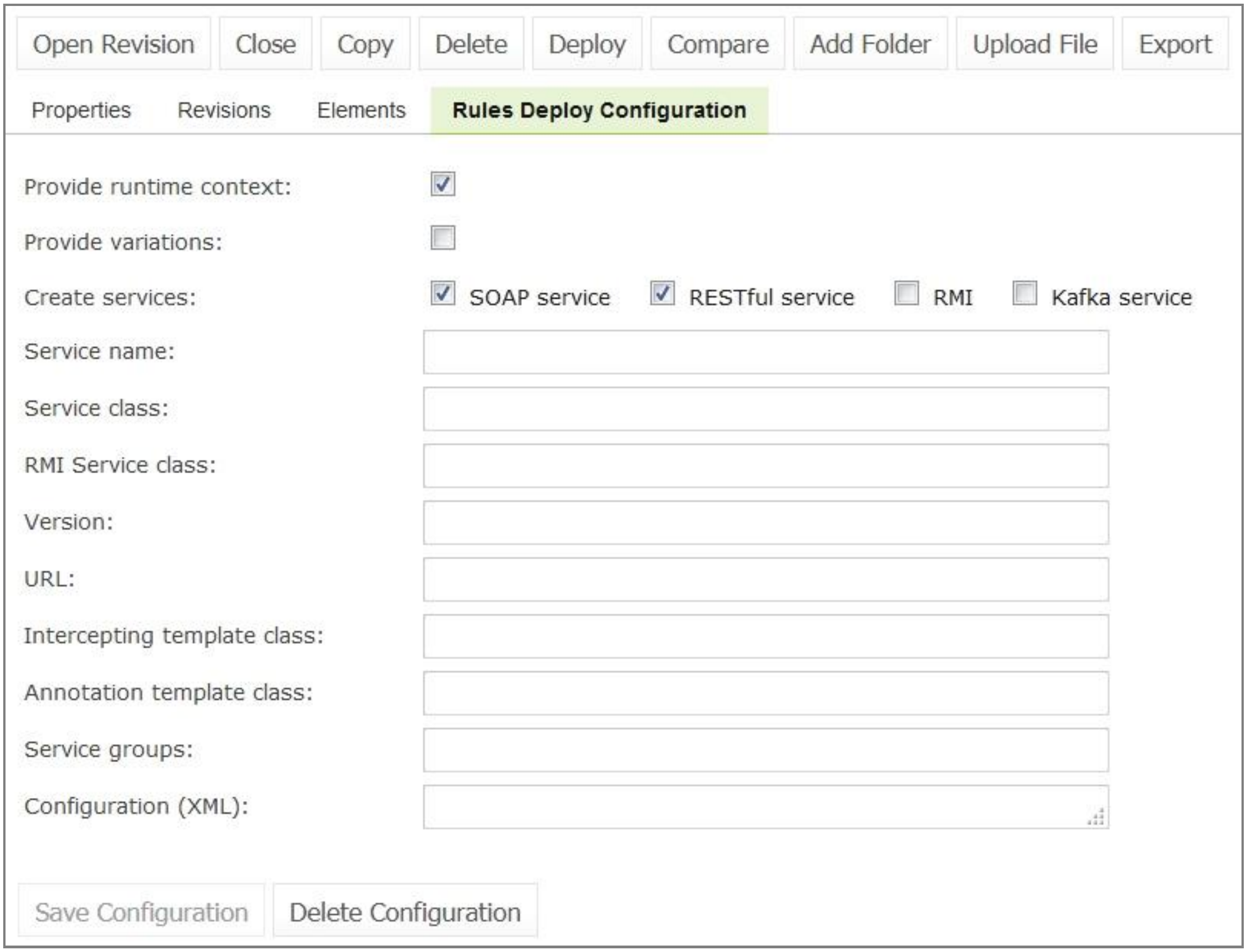



*Figure 129: Defining rules deploy configuration settings*

# Defining Rule Service Version

OpenL Tablets WebStudio supports versioning definition for rule services. This functionality allows specifying a version for the project revision to be deployed. The required version of the deployed project can be called from deployment repository. All specified versions of the project appear on the OpenL Tablets Rule Services page with a version number defined in brackets.

To check the services version deployment, in OpenL Tablets Rule Services, find the name of the deployed project. Services version is set both in the services header and in the services URL.


*Figure 130: Services header and URL with the version number*

To define the rule service version, proceed as follows:

1. In the **Projects** tree, select a project.
2. In the top line menu, click **Rules Deploy Configuration**.
3. In the window that appears, click the **Version** field.
   By default, the **Major 0, Minor 0** scroll list appears.
   For more information on how to configure deployment configuration settings, see Configuring Additional Rules Deploy Configuration Settings.
4. In the scroll list, select the services version.
   For example, to create the services version 1.0, Major = 1 and Minor = 0 must be selected.

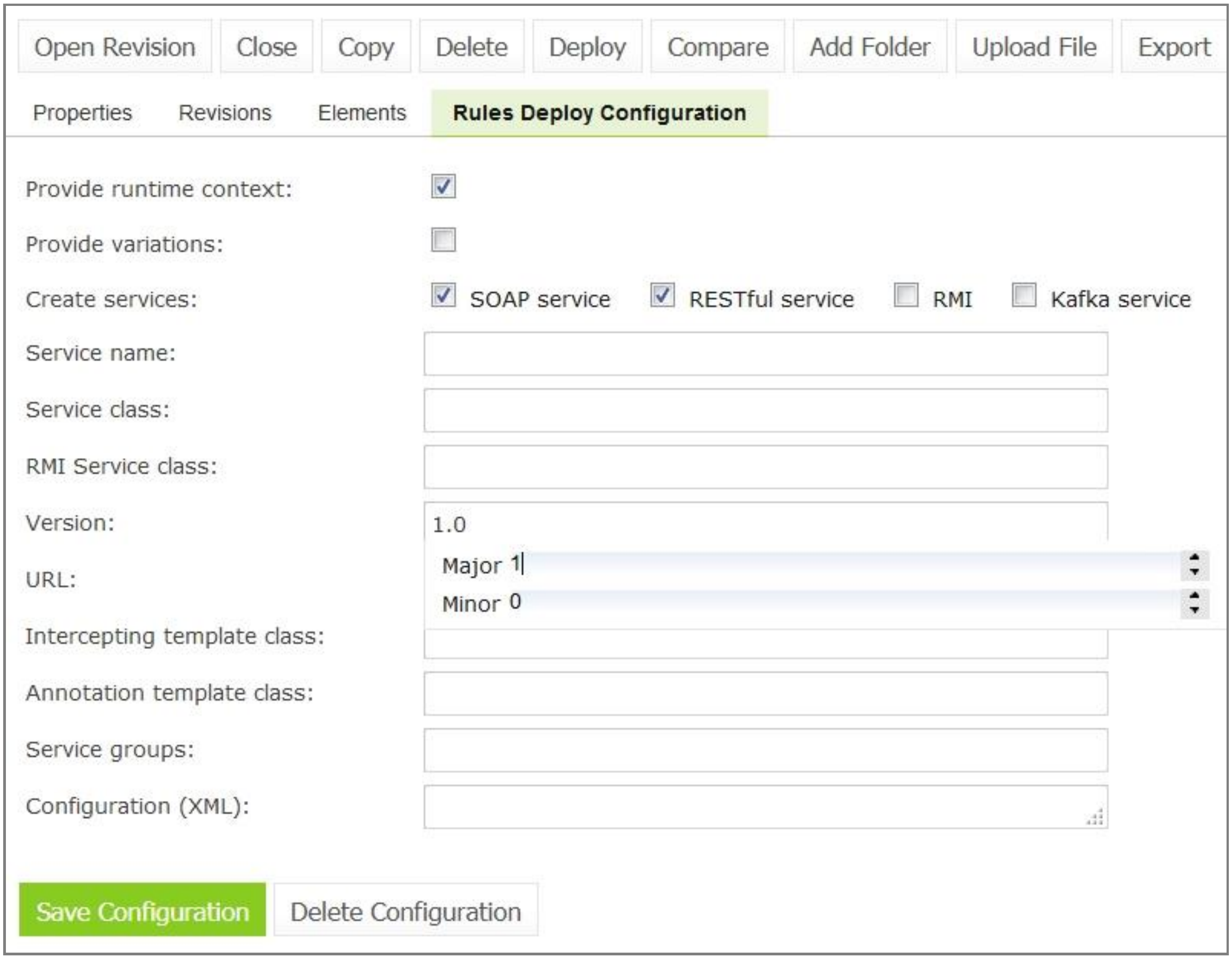



*Figure 131: Defining services versioning*

5. Click **Save Configuration**.

   The selected services version is displayed in **Rules Deploy Configuration** for the selected project. For the example displayed in this section, the project version is 1.0.

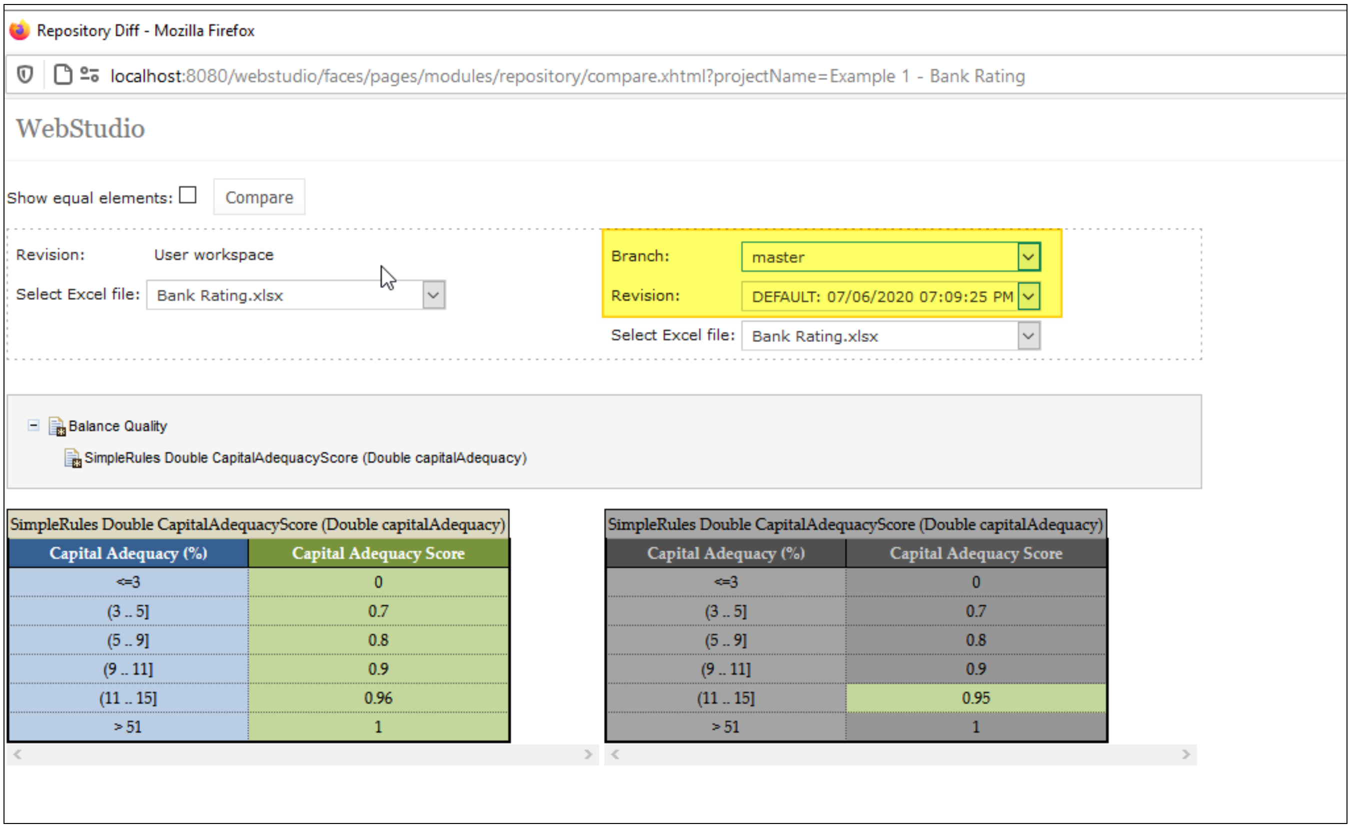
# 6.12 Comparing Project Revisions

OpenL Tablets WebStudio provides a function for comparing files and sheets in Excel files between two project revisions.

To compare contents of the currently opened project revision with any other revision, proceed as follows:

1. In the project tree, select the project.
2. In the right pane, click **Compare**.

   A window appears listing contents of the currently opened project version on the left side and contents of another project revision on the right side.




| SimpleRules Double CapitalAdequacyScore (Double capitalAdequacy) | |
|---|---|
| Capital Adequacy (%) | Capital Adequacy Score |
| <=3 | 0 |
| (3 .. 5] | 0.7 |
| (5 .. 9] | 0.8 |
| (9 .. 11] | 0.9 |
| (11 .. 15] | 0.96 |
| > 51 | 1 |

| SimpleRules Double CapitalAdequacyScore (Double capitalAdequacy) | |
|---|---|
| Capital Adequacy (%) | Capital Adequacy Score |
| <=3 | 0 |
| (3 .. 5] | 0.7 |
| (5 .. 9] | 0.8 |
| (9 .. 11] | 0.9 |
| (11 .. 15] | 0.95 |
| > 51 | 1 |

*Figure 132: Comparing the current project revision from user workspace to the second project revision*

3. To compare the current project revision with a different revision, select the branch and revision.

# 6.13 Exporting a Project or a File

To export a project from repository editor, proceed as follows:

1. In the project tree, select the project.
2. In the right pane, click **Export**.
3. In the displayed window, select the required project revision, click **Export** and a full project in the selected revision will be exported.

To export any revision of a file from Repository, proceed as follows:

1. In the project tree, select the project.
2. Expand the project tree and select the file to be exported.
3. In the right pane, click **Export file**.
4. In the displayed window, select the required file revision and click **Export**.

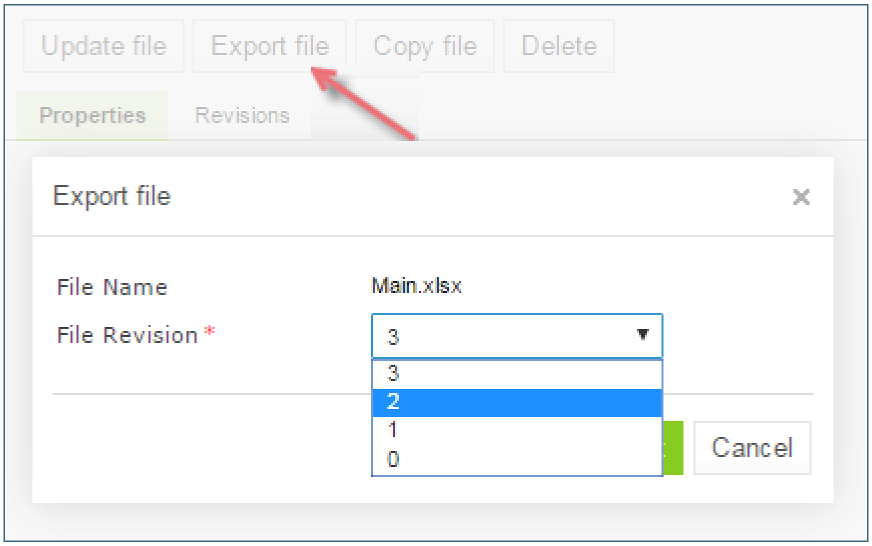



*Figure 133: Exporting a file from a project*

**Note:** If the project is in the Local status, these options are not available.

# 6.14 Unlocking a Project

OpenL Tablets WebStudio provides a function for a user to unlock a project which is edited and, therefore, locked by another user. Be aware that after unlocking, all unsaved changes made by another user will be lost and the project will be closed.

To unlock a project, proceed as follows:

1. Perform one of the following steps as required:
   - In the **Projects** tree, select the project and, in the right pane, click **Unlock**.
   - Click **Projects** in Navigator to get a list of projects, navigate to the project that needs to be removed and click the corresponding **Unlock** item on the right.
2. In the confirmation window, click **OK**.

It is recommended to grant permission to the “Unlock” functionality only for administrators.

# 6.15 Browsing the Deployment Repository

**The Deployment repository** contains project deployments and is also the location from where solution applications use them. OpenL Tablets WebStudio allows connecting several deployment repositories. For information on how to configure deployment repositories, refer to Managing Repository Settings.

To browse a deployment repository, proceed as follows:

1. Switch from the **Design repository** view to the **Deployment repositories** view by clicking **Deployment** in the top of the left pane.
2. In the project tree, select the deployment repository to be browsed (repositories are marked by icon).

The list of project deployments or deployed configurations – deploy configurations which consist of rule projects and specific project revisions and deployed to the selected deployment repository – are displayed in the middle pane.

3. If needed, expand the repository tree and browse project deployments.

OpenL Tablets WebStudio displays only the latest revisions of each deployed configuration in the deployment repository.

Also, when browsing deployed configurations in the deployment repository, users can see their content, namely what rules projects are deployed.

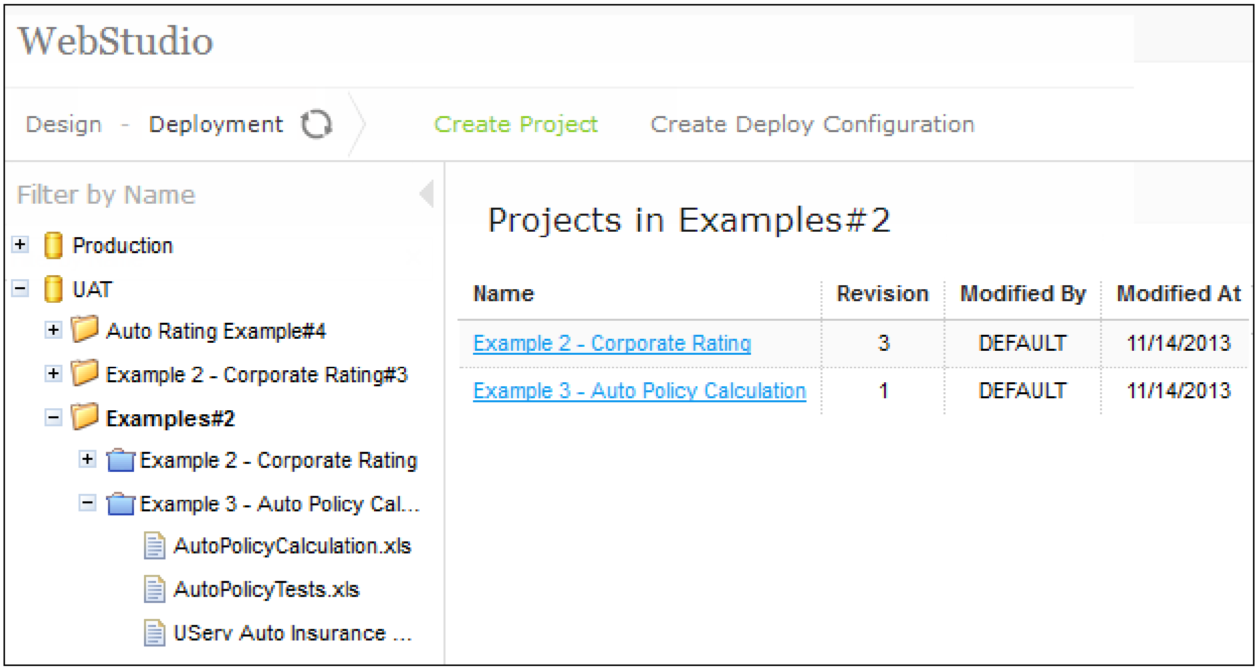



*Figure 134: Deployment repository with deployed projects*

# 7 Working with Project Branches

This section introduces project branches and describes how to use them. Branches are useful when several users work on the same project simultaneously and then merge the changes or keep them as separate project versions.

The following topics are included in this section:

- Creating a Branch
- Working with Branches
- Resolving Conflicts

## 7.1 Creating a Branch

A branch is created by copying an existing project. Both predefined and user-defined names can be used for a branch. For more information on name patterns, see Setting Up Git Repository Functionality.

Proceed as follows:

1. In OpenL Tablets WebStudio, in the editor or repository, select a project.
2. Click **Copy.**
3. In the **New Branch Name** field, proceed with the default value or enter a new branch name.
   The newly created project branch is displayed as an active branch and ready for work.

**Note:** When a project in the **Closed** status is copied, the project in the newly created branch has the **No changes** status.

## 7.2 Working with Branches

This section describes how to view existing branches, switch between them in the editor and repository, enable and disable branches, and delete branches. Proceed as follows:

1. To display a current project branch, in OpenL Tablets WebStudio, in the editor or repository, open a project.
   The current project branch is displayed.
2. To switch between branches in the editor, click the last link in the address bar identifying the branch name and in the list that appears, select the required branch.

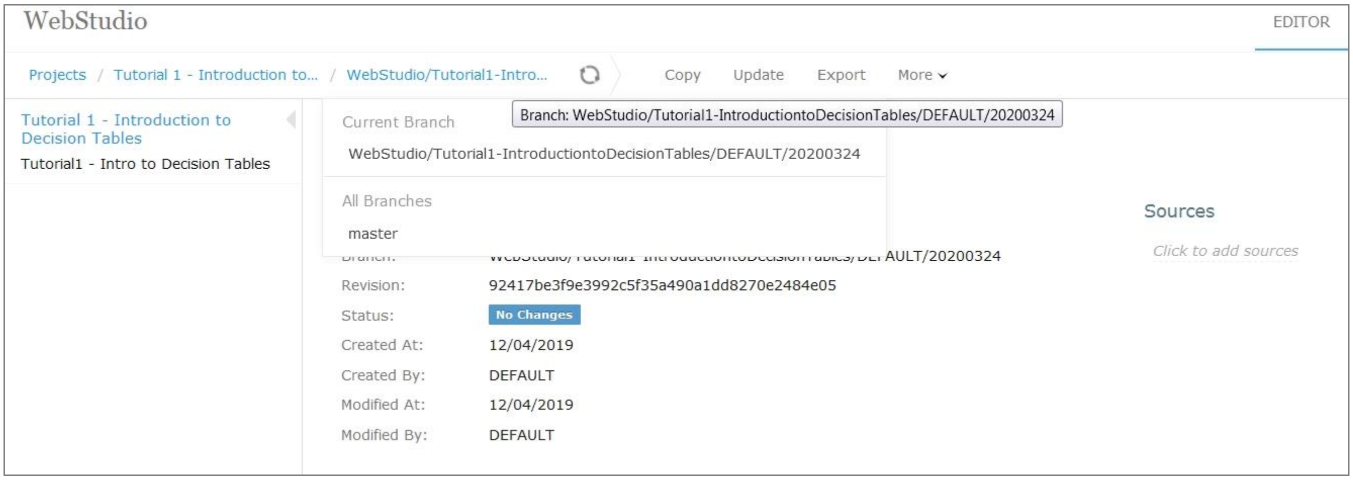



*Figure 135: Switching between branches in the editor*

3. To switch between branches in the repository, for a project, in the **Branch** field, select the required branch.
4. To disable or enable a branch for a project, in the repository, click the dots next to the branch name field and in the window that appears, clear or select the appropriate branch check box.
   This list also contains branches created outside of OpenL Tablets WebStudio.


*Figure 136: Enabling and disabling branches for a project*

5. To delete a branch, in the editor, select the required project branch and click **Delete.**
   The project is archived and disappears from the list of active projects.
6. To completely delete the branch or restore the archived project, proceed as follows:
   1. Click the filter icon and clear the **Hide deleted projects** check box.
   2. In the branch drop-down list, select an archived branch.
   3. To restore an archived project, select it in the list of projects and click **Undelete.**
   4. To completely remove the project and the branch, click **Erase.**
7. To merge two branches, click **Sync** and select one of the following options:

| Options for merging branches | |
|---|---|
| **Option** | **Description** |
| Receive their updates | Changes from a selected branch are copied to the currently active branch. |
| Send your updates | Changes from the currently active branch are uploaded to the selected branch. |

If upon saving there is a conflict due to updates in the same modules, the **Resolve conflicts** window appears.

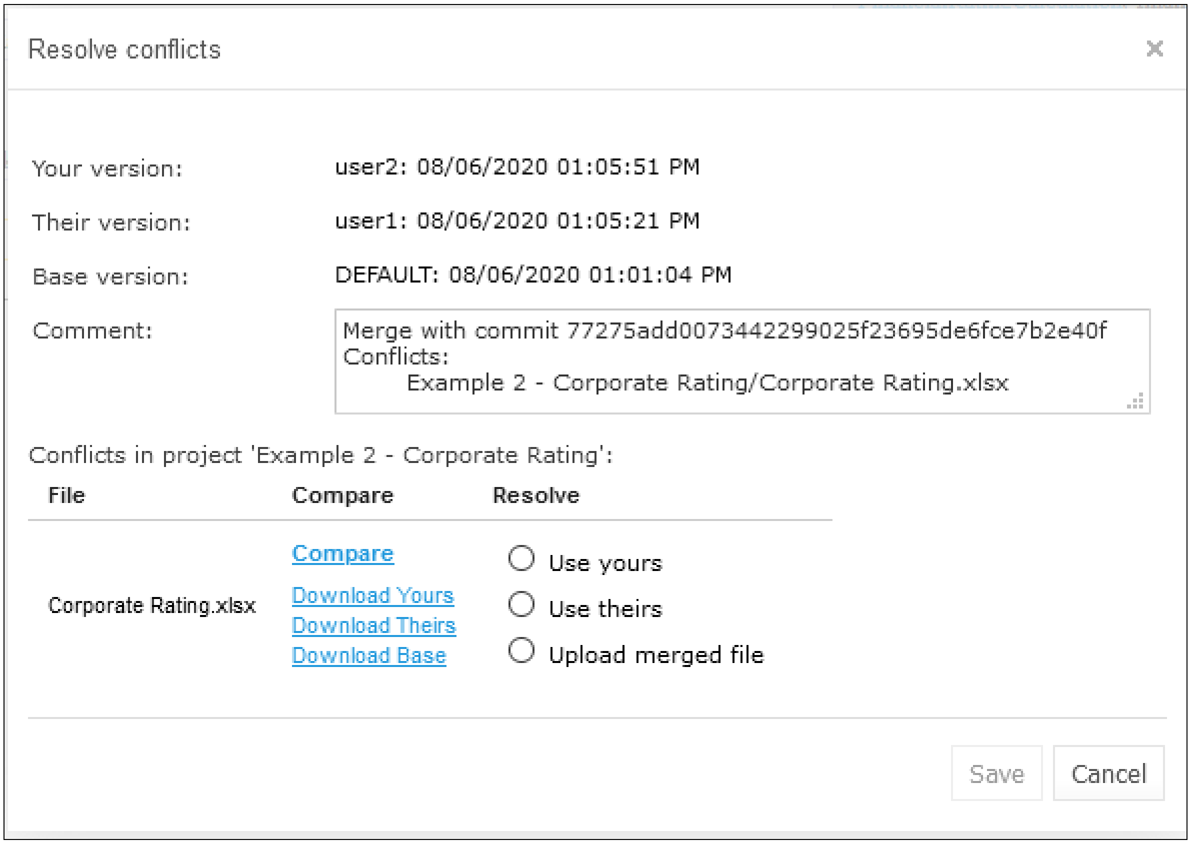



*Figure 137: Resolving conflicts on merging branches*

Conflicts can be resolved by selecting one of the following options:

| Options for resolving conflicts when merging branches | |
|---|---|
| **Option** | **Description** |
| Use yours | Changes in the currently active branch are applied on merge. The changes applied by another user are lost. |
| Use theirs | Changes in the selected branch are applied on merge. The changes made by you are lost. |
| Upload merged file | Depending on the selected merging options, changes in the manually updated and uploaded file override changes in the branch. |

8. To view the changes made by another user, compare them to your changes, or view the base version of the file, select a corresponding option in the **Compare** column.

# 7.3 Resolving Conflicts

If the same version of the project is edited by several users, upon submitting their changes using different clients, the conflict error message appears asking which version must be saved. Changes in other versions are discarded.

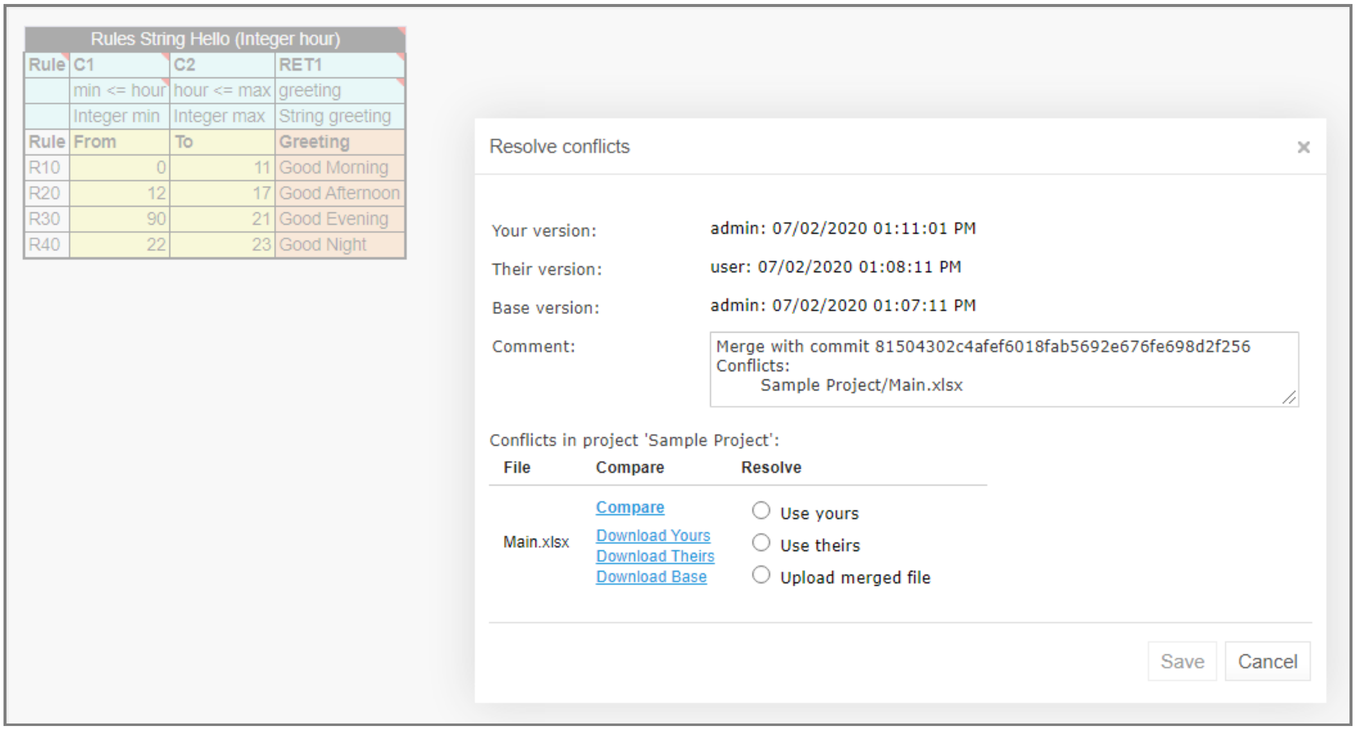



*Figure 138: An error message upon saving conflicting versions*

The error message contains the **Compare** link that allows viewing both conflicting versions for comparison.

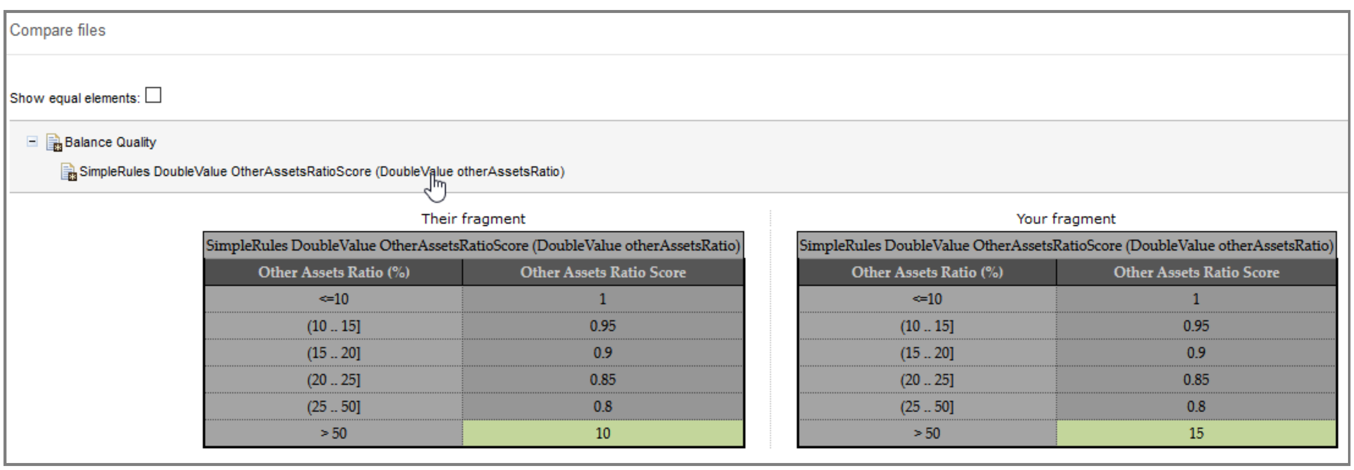



*Figure 139: Comparing conflicting versions*

# 8 Using Administration Tools

This section explains how to view and control OpenL Tablets WebStudio system settings and manage user information in the system.

To perform administration tasks, in the top line menu, click **Administration**.

By default, the **System Settings** tab is displayed. The system settings are organized into the **Common**, **Repository**, and **System** groups. To open the group, click the corresponding icon on the left.

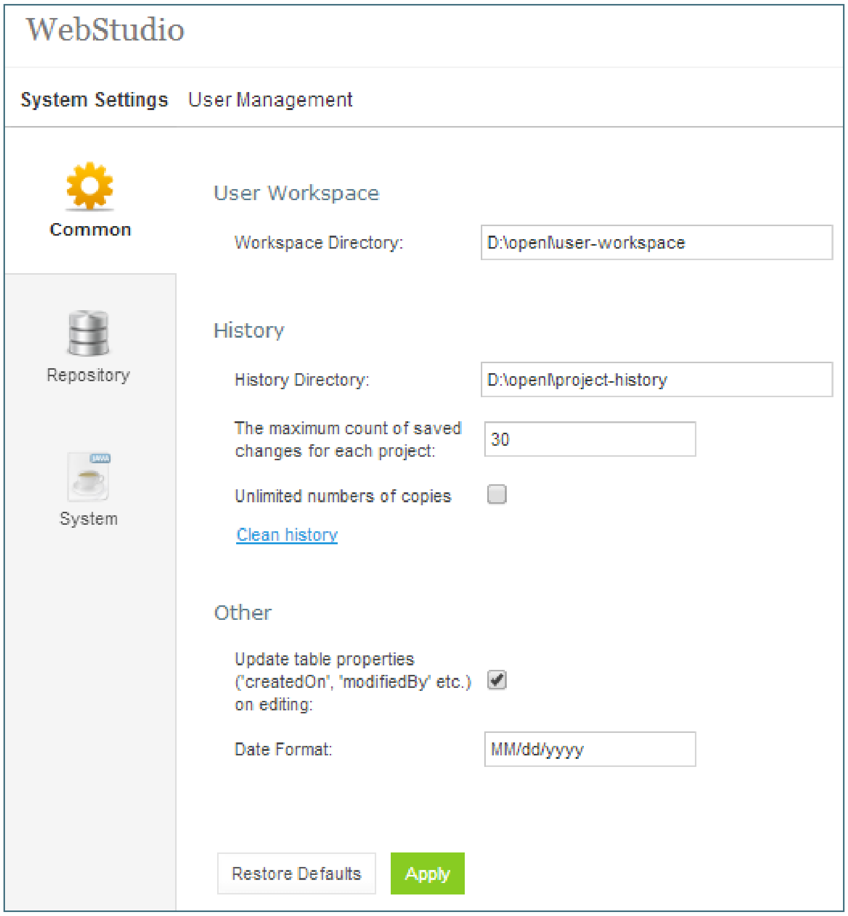



*Figure 140: OpenL Tablets WebStudio administration*

Normally, the default settings are recommended, but users with appropriate permissions can change them as required. After making changes, click **Apply** and refresh the page. To restore the original settings, click the **Restore Defaults** button.

The following topics are included:

- Managing Common Settings
- Managing Git Repository Settings
- Managing System Settings
- Managing User Information

# 8.1 Managing Common Settings

The **Common** section defines the following general OpenL Tablets WebStudio settings:

- Managing User Workspace Settings
- Managing History Settings
- Managing Other OpenL Tablets WebStudio Settings

## Managing User Workspace Settings

The **User Workspace** section is used to define the workspaces directory where user projects are located.

## Managing History Settings

To manage history settings, proceed as follows:

1. In the **History Directory** field, specify the directory where user project history files are stored.
   These files are used to track, compare, and revert the changes made in projects.
2. To set up the number of history files, proceed as follows:
   1. Clear the **Unlimited numbers of copies** check box.
      Clearing this check box enables to define the number of history files stored for a project.
   2. In the **The maximum count of saved changes for each project** field, enter the required number.
      By default, this field value is set to 30.
   3. Click **Apply** to save changes.
   4. In the confirmation dialog, click **OK**.
3. To clean all history files for the project, perform the following steps:
   1. Click the **Clean history** link.
   5. In the **Clean projects history** form, select the particular project check box.
   6. Alternatively, to clear history for all projects, select **Name**.
   7. Click **Clean** to complete.

## Managing Other OpenL Tablets WebStudio Settings

The following table describes other general OpenL Tablets WebStudio settings:

| Other general OpenL Tablets WebStudio settings | |
|---|---|
| **Option** | **Description** |
| **Update table properties** | Indicates whether table properties controlled by the system must be updated and can be viewed in OpenL Tablets WebStudio UI.<br>If this option is cleared, information about the time of table creation and modification and changes authors, such as **Created By/On**, **Modified By/On**, is not added to the table properties. |
| **Date Format** | Enables changing the date format in the OpenL Tablets WebStudio UI. |

# 8.2 Managing Git Repository Settings

**Git** is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. For more information on Git, see https://git-scm.com/.

A **Git repository** is the `.git/` folder inside a project. This repository tracks all changes made to files in the project, building a history over time.

This section describes how to set up a connection to a Git repository, configure Git functionality, and resolve conflicts when modifying the same version of the project, and includes the following topics:

- Setting Up a Connection to a Git Repository
- Setting Up Git Repository Functionality

## Setting Up a Connection to a Git Repository

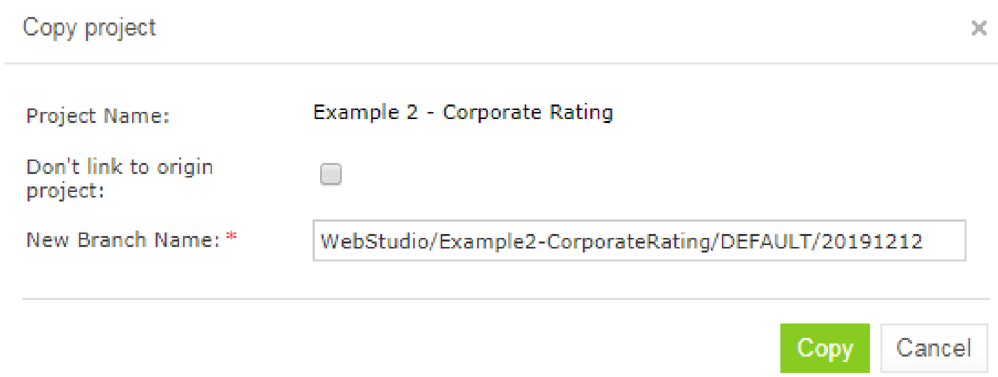
In the **Administration** tab, in the **Repository** section, define values for the following connection properties:

| **Git repository connection settings** | |
|---|---|
| **Parameter** | **Description** |
| Name | Repository name. This value cannot be modified. |
| Type | Type of the repository. The value must be set to **Git.** |
| Remote repository | Identifier of whether a Git repository is stored remotely. |
| URL | URL for the remotely located Git repository or file path to the repository stored locally. |
| Login | Username for accessing a remote Git repository. |
| Password | Password for accessing a remote Git repository. |
| Local path | Folder where a Git repository local copy is stored. |
| User display name | Username to which Git commits are associated.<br>If username validation is enabled on the Git server, commits must be allowed for a username specified in this field.<br>If a username is not specified, OpenL Tablets WebStudio username is used for Git commits. |
| User email | Email to which Git commits are associated. |

## Setting Up Git Repository Functionality

To set up Git repository functionality, in the **Administration** tab, in the **Repository** section, select the **Remote repository** check box and define values for the following properties:

| **Git repository setup parameters** | |
|---|---|
| **Parameter** | **Description** |
| Branch | Project branch that is used by default. |

| Git repository setup parameters | |
|---|---|
| **Parameter** | **Description** |
| New branch pattern | Pattern for a new branch name.<br>• {0} is replaced by a project name.<br>• {1} is replaced by a username.<br>• {2} is replaced by the current date.<br>For example, for the `WebStudio/{0}//{1}/{2}` pattern, real branch name can resemble the following.<br><br>*Figure 141: Branch name example* |
| Changes check interval | Repository changes check interval in seconds. The value must be greater than 0. |
| Connection timeout | Repository connection timeout in seconds. The value must be greater than 0. |
| Customize comments | Comment message template for Git commits. Comments can be customized using the following placeholders:<br>• {username}<br>• {user-message}<br>• {commit-type}<br>• {project-name}<br>• {revision}<br>By default, all commits are submitted to Git with a message in the following format:<br>{user-message} Author: {username}. Type: {commit-type} |
| Flat folder structure | Flag that denotes repository structure.<br>For a flat structure, all projects are stored in the directory specified in the **Path in repository** property, each project in its own folder.<br>Otherwise, projects can reside in folders and subfolders defined by a user upon project creation or copying, with each project having its own level of nesting, and the **Folder config file** field contains the property file listing all projects and paths to them.<br>Folder name limitations are the same as those applied to folder names by the used OS. |
| Path in repository | Directory where all flat repository structure projects are stored. |

# 8.3 Managing Repository Settings

The **Repository** section contains connection settings of design and deployment repositories. To modify the repository settings, proceed as follows:

1. In the **Name** field, enter the repository name to be displayed in repository editor.
2. Select the connection type and enter corresponding location of the repository to be used as a data source as described in the following table.

| **Connection types for setting up design and deployment repositories** | |
|---|---|
| **Type** | **Description** |
| **Git** | The repository can be configured on the local or remote machine. |
| **Database** | The repository is located in a local or remote database. **Repository URL** field displays URL for access to the database.<br>A user can create connection to different databases, such as MySQL, MS SQL, Oracle etc. For more information on supported versions, see http://openl-tablets.org/supported-platforms. |
| **AWS S3** | The repository is located in Amazon Simple Storage Service (AWS S3). A "bucket" is a logical unit of storage in AWS S3, and is globally unique. Choose a region for storage to reduce latency, costs, etc. An Access key and a Secret key are needed to access storage. If empty, the system retrieves it from one of the known locations as described in AWS Documentation. Best Practices for Managing AWS Access Keys. The Listener period is the interval in which to check repository changes, in seconds. |

For more information on repository settings, see the **Data Source Configuration** section in ***[OpenL Tablets Rule Services Usage and Customization Guide]***.

The following table provides examples of deployment repository URL values for different databases.

| **Examples of deployment repositories URL values for different databases** | |
|---|---|
| **Database** | **URL value sample** |
| **MySQL, MariaDB** | jdbc:mysql://localhost:3306/prodRepository,<br>jdbc:mariadb://localhost:3306/ prodRepository (for MariaDB driver) |
| **PostgreSQL** | jdbc:postgresql://localhost:5432/ prodRepository |
| **MS SQL** | jdbc:sqlserver://localhost:1433;databaseName=prodRepository;integratedSecurity=false |
| **Oracle** | jdbc:oracle:thin:@localhost:1521:prodRepository |

3. To set up a secure connection for connecting to remote or database-located repositories, select the **Secure connection** check box and fill in the login and password fields.

   For more information on repository security, see the **Configuring Private Key for Repository Security** section in ***[OpenL Tablets Installation Guide]***.

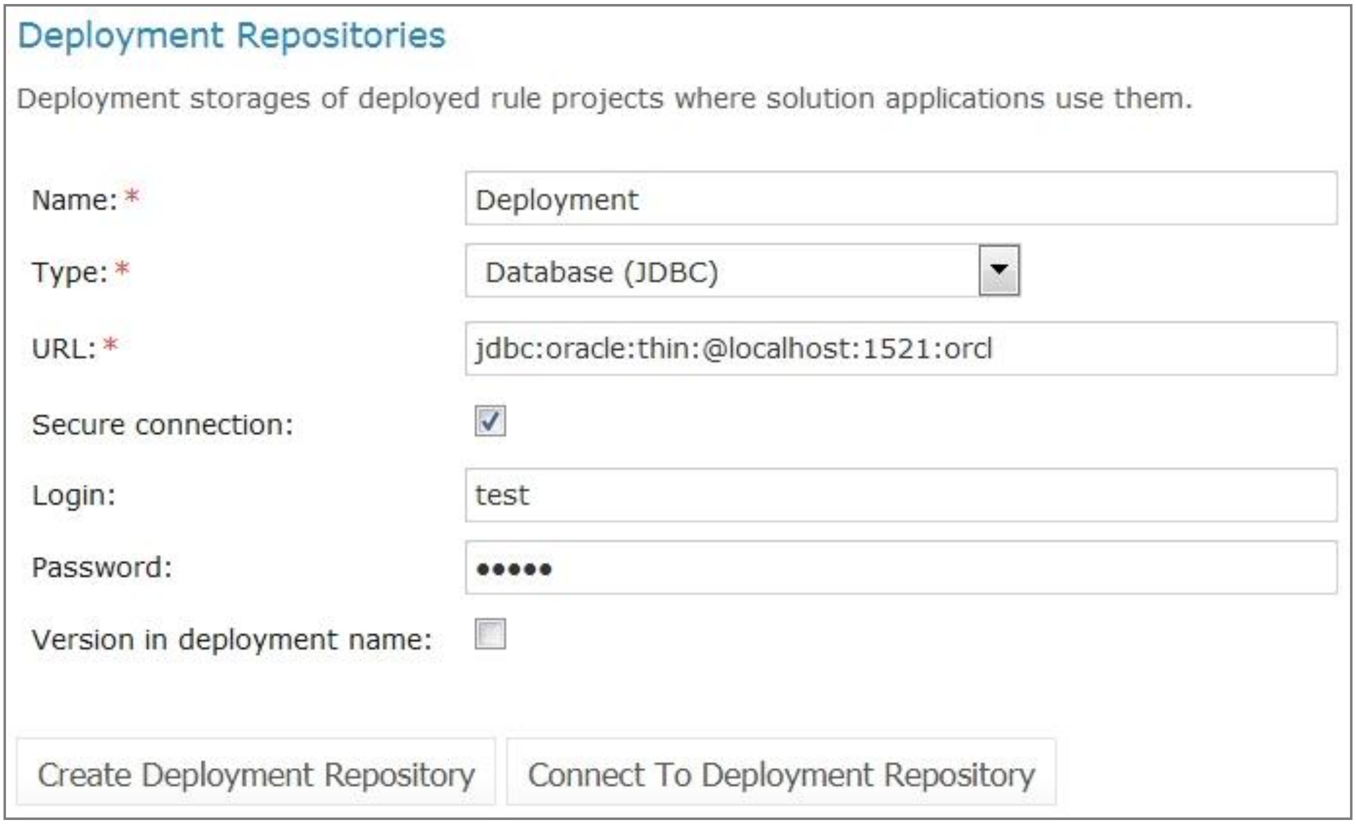



*Figure 142: Configuring deployment repository settings*

Connection settings can be changed by editing the tab or deleted by clicking the red cross .

Connection to a local deployment repository is configured by default.

4. To connect to other deployment repositories, click the **Connect To Deployment Repository** button, enter the repository parameters, and click **Connect**.

   Note: In cases where the Deployment repository was created in OpenL versions older than 5.20, the check box “Version in deployment name” must be checked for backward compatibility.

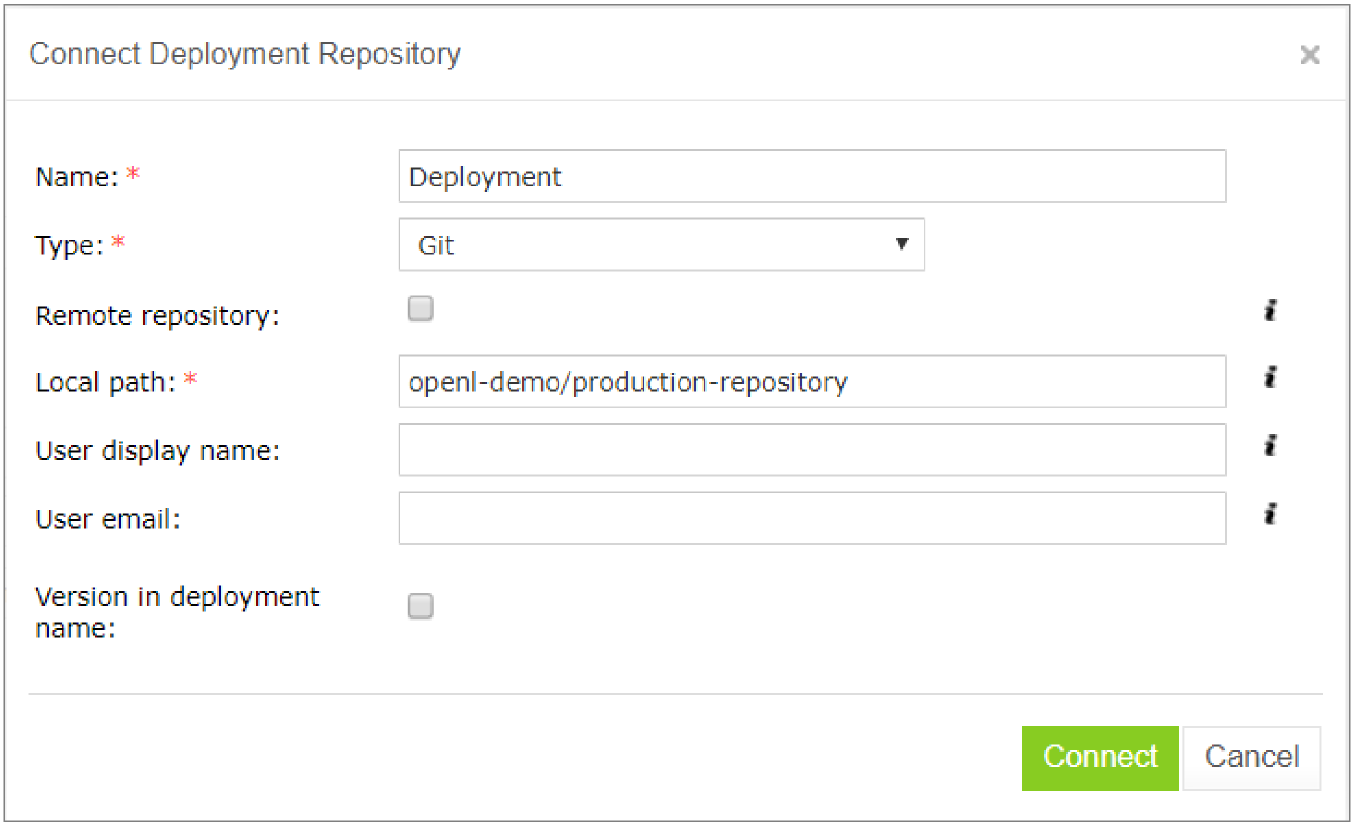



*Figure 143: Connecting to a deployment repository*

5. To create a local deployment repository, click the **Create Deployment Repository** button, enter new repository parameters, and click **Create**.
6. To store Deploy configurations in a separate repository, not in Design repository, clear the **Use Design Repository** check box and provide required parameter values.

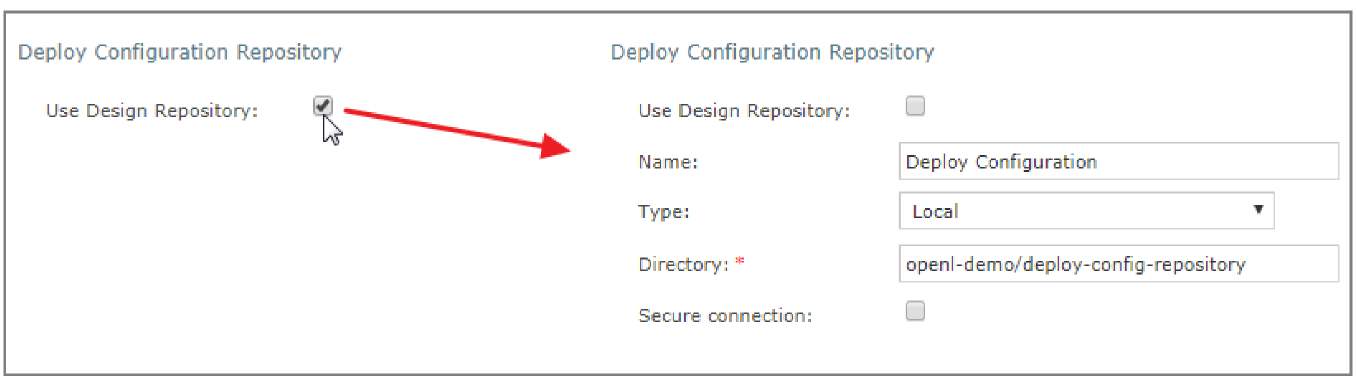



*Figure 144: Using another repository for deployment configurations*

7. When finished, click **Apply** to save the changes and refresh the page.

## 8.4 Managing System Settings

The **System** tab enables modifying core, project, and testing options and includes the following topics:

| System settings management | | |
|---|---|---|
| **Section** | **Property** | **Description** |
| Core | **Dispatching Validation** | Setting turns on/off the mechanism of dispatching for a rule table where the only one version of this rule table exists.<br>By default, the **dispatching.validation** value is set to **true** in OpenL Tablets WebStudio. For information on dispatching validation, see the **Table Dispatching Validation Mode** section in ***[OpenL Tablets Rule Services Usage and Customization Guide]***. |
| | **Verify on Edit** | Allows turning on/off checking of rules consistency and validity on each edit in Rules Editor.<br>By default, the check box is selected. Automatic checks are executed after each edit.<br>If this option is cleared, the verification process does not launch automatically when the **Save** button is clicked. Instead, a **Verify** button appears in Rules Editor, and the user must verify manually by clicking this button. |
| Testing | **Run test cases of the test in parallel** | Enables reducing the time spent on executing test cases of a test table by configuring the number of parallel threads in the field below.<br>If this option is cleared, all test cases are executed one by one. |
| | **Thread number for tests** | Indicates the number of test cases executed simultaneously.<br>By default, four threads are set. It means that after running a test table or all tests, up to four test cases will be in progress at the same time. When they are calculated, the next four test cases will be executed. |

# 8.5 Managing User Information

This section describes how to control user access in the OpenL Tablets WebStudio application based on users and user groups. All privileges in the system are assigned at a group level and will be granted to a particular user after he or she is included in a particular group.

Users and groups are managed from the **User Management** tab which, in turn, is divided into **Users** and **Groups & Privileges** tabs. Only members of the **Administrators** group have rights to manage users and groups in OpenL Tablets WebStudio.

The following topics are included in this section:

- Managing Groups
- Managing Users

## Managing Groups

This section explains how to create, modify, and delete a user group with a certain set of privileges. The **Administrators** group cannot be deleted from the system.

The following topics are included in this section:

- Viewing a List of Groups
- Adding a Group
- Editing a Group
- Deleting a Group

- [Managing a Group in Case of Third Party Identity Provider]

## Viewing a List of Groups

To view a list of groups, proceed as follows:

1. In the **Administration** tab, click **User Management** on the top-left of the screen.
2. Click **Groups & Privileges** on the left.

   The system displays a list of groups similar to the following one:

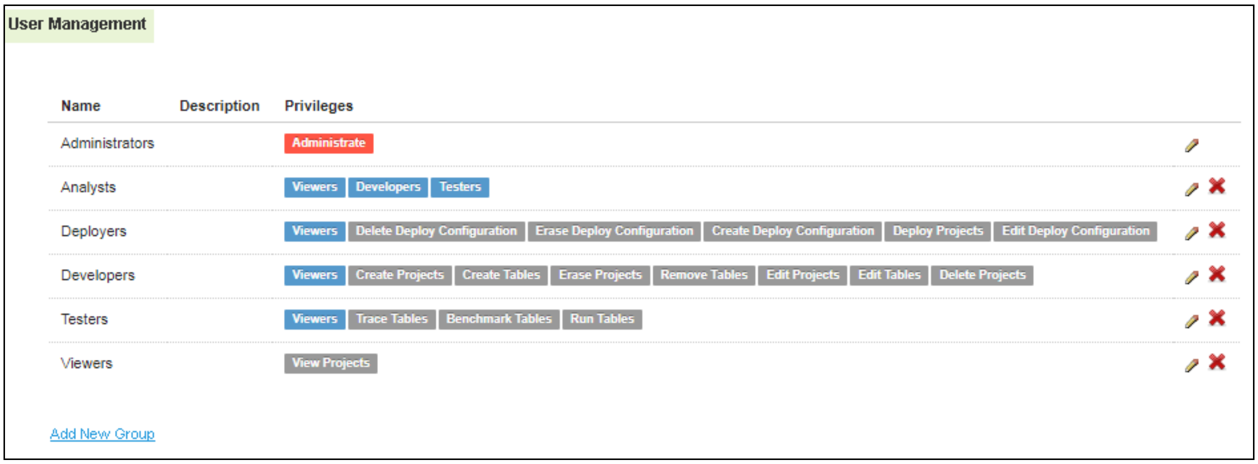



*Figure 145: User groups in the **Groups & Privileges** tab*

3. To create a new group, proceed as described in Adding a Group.
4. To edit a group, proceed as described in Editing a Group.
5. To delete an existing group, proceed as described in Deleting a Group.

## Adding a Group

To add a new group, proceed as follows:

1. Click the **Add New Group** link.

   The **Add New Group** form appears.
2. Enter the group name in the **Name** field.
3. Optionally, provide group description in the **Description** text box.
4. In the **Privilege** area, define the privileges as needed.

   To assign a set of privileges for a group, click the group name above the list of privileges, such as Developers, Testers, or Administrators. The **Viewers** group is selected for a new user group by default.

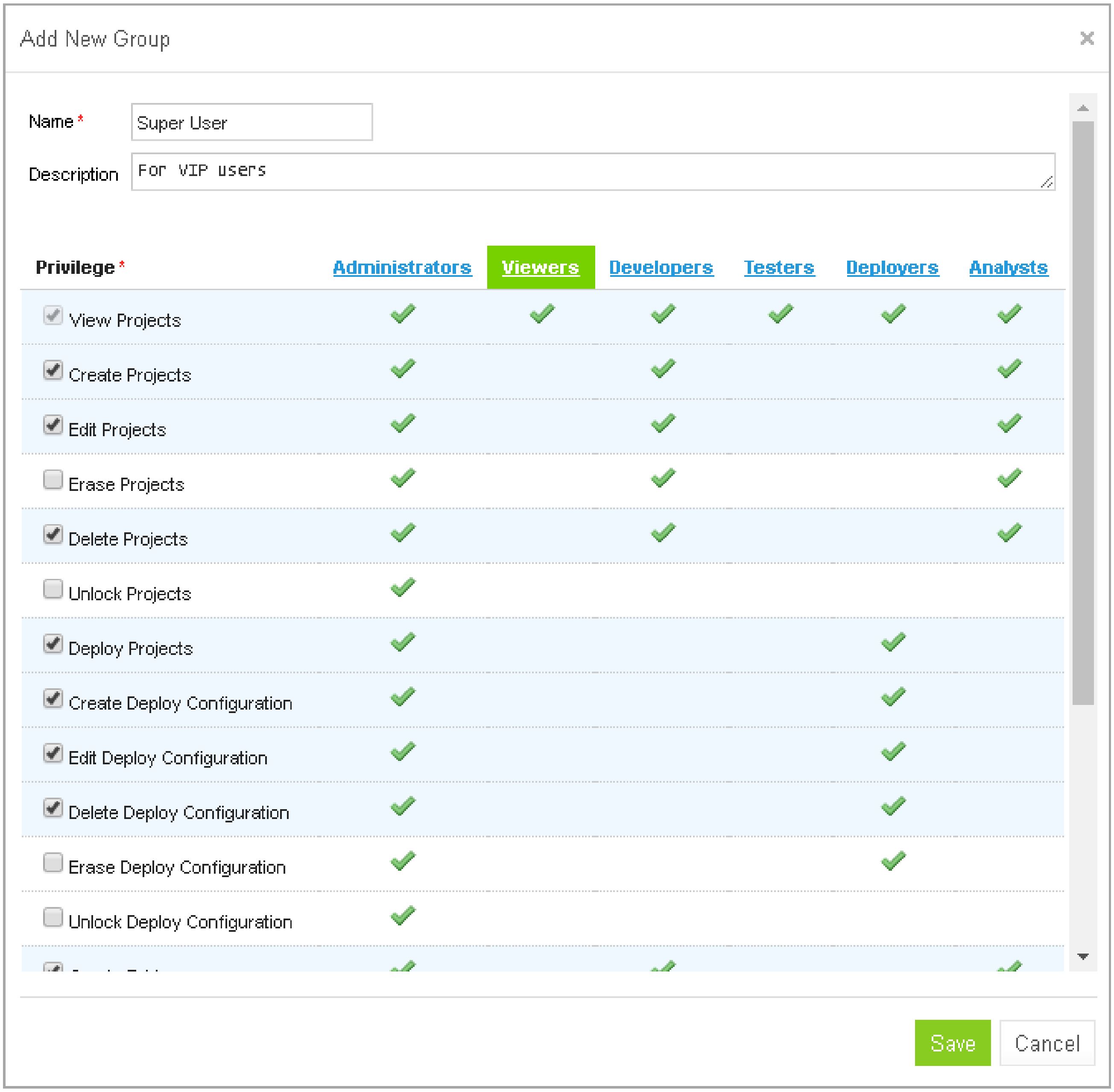



*Figure 146: Add a new user group with required set of privileges*

5. Click **Save**.

## Editing a Group

To modify a user group, proceed as follows:

1. In the list of groups, locate the group that needs to be changed and click the **Edit** icon .
2. In the **Edit Group** form, change the group name, add or modify its description, and change privileges as needed.
3. Click **Save** to complete.

### Deleting a Group

To delete a user group, proceed as follows:

1. Locate the group to be deleted and click the red cross on the right: .
2. Click **OK** in the confirmation dialog.

### Managing a Group in Case of Third party Identity Provider

If OpenL Tablets WebStudio is installed with the option to sign in via a third party identity provider, such as SSO or AD, and the **External user management** option is selected, groups created and edited in OpenL Tablets WebStudio must have the same names as available Active Directory or SSO groups.

## Managing Users

Users get access to OpenL Tablets WebStudio functions by including them in particular groups.

By default, there are the following users in OpenL Tablets WebStudio predefined in Demo mode:

| **OpenL Tablets WebStudio users** | | |
|---|---|---|
| **User name** | **User password** | **Groups** |
| user | user | Viewers |
| u0 | u0 | Testers |
| u1 | u1 | Developers, Analysts |
| u2 | u2 | Viewers |
| u3 | u3 | Viewers |
| u4 | u4 | Deployers |
| a1 | a1 | Administrators |
| admin | admin | Administrators |

On the first start of the OpenL Tablets WebStudio in Multi user mode, users are provided with `admin/admin` login/password pair that gives them Administrator's permissions. Users can then set up their own users in OpenL Tablets WebStudio as needed. For information about the permissions of the groups, refer to Managing Groups.

The following topics are included in this section:

- Viewing a List of Users
- Creating a User
- Editing a User
- Deleting a User
- Managing Users in Case of Third Party Identity Provider

### Viewing a List of Users

To view a list of users, proceed as follows:

1. In the **Administration** tab, click **User Management** on the top-left of the screen.
2. Click **Users** on the left.

The system displays a list of OpenL Tablets WebStudio users.

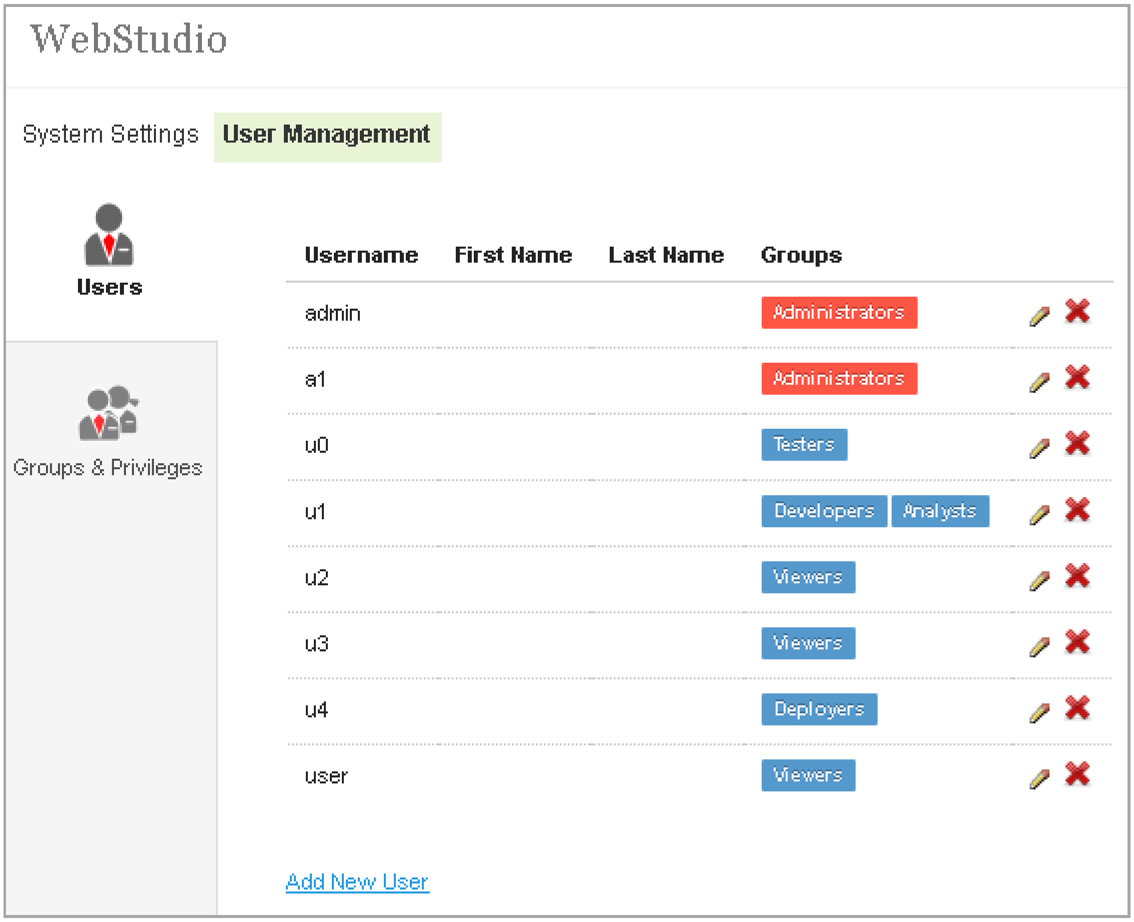



*Figure 147: List of OpenL Tablets WebStudio users*

3. In the **Users** tab, perform either of the following:
   - To create a new user, proceed as described in Creating a User.
   - To edit a user, proceed as described in Editing a User.
   - To delete a user from the system, proceed as described in Deleting a User.

## Creating a User

While creating a new user, make sure to include the user in at least one group.

To create a new user, proceed as follows:

1. Click the **Add New User** link.
   The system displays the **Add New User** form.

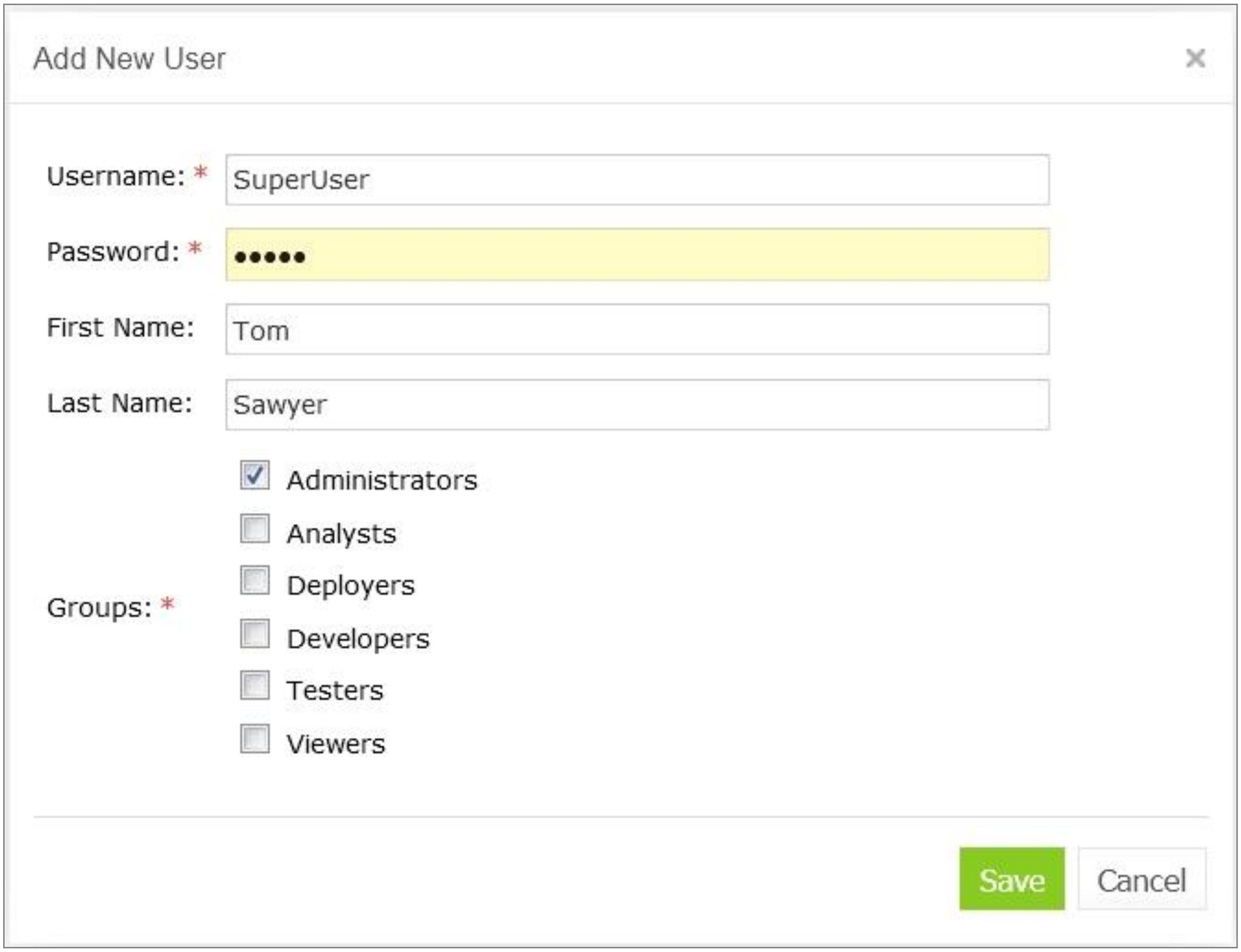



*Figure 148: Creating a new user*

2. Specify the user’s login in the **Username** field and enter the password in the **Password** field.
3. Optionally, enter the user’s first and last name.
4. Select one or more groups to assign the user to.
5. Click **Save** to complete.
   The system displays the new user in the **Users** list.

## Editing a User

To edit a user, proceed as follows:

1. In the **Users** list, locate the user that needs to be modified and click the **Edit** icon: .
2. In the **Edit User** form, change user first name or last name and specify the groups to which a user belongs.
   Users with the administrator privilege can also reset passwords for other users. For users exported from Active Directory or SSO, only group information is editable.
3. Click **Save** to save the changes.

### Deleting a User

The **Administrators** group in OpenL Tablets WebStudio must contain at least one administrator user. That it, the only OpenL Tablets WebStudio administrator cannot be deleted.

To delete a user, proceed as follows:

1. In the **Users** list, locate the user for deletion and click the **Delete** icon: .
2. Click **OK** in the confirmation dialog.

### Managing Users in Case of Third Party Identity Provider

There are some differences in managing users when OpenL Tablets WebStudio is installed with an option to sign in with a third party identity provider.

If the **External user management** option is selected on OpenL Tablets WebStudio installation:

- For SSO, the **Users** section is not available.
- For Active Directory, the **Users** section is available for managing only local users stored in OpenL Tablets WebStudio database, and not in Active Directory.

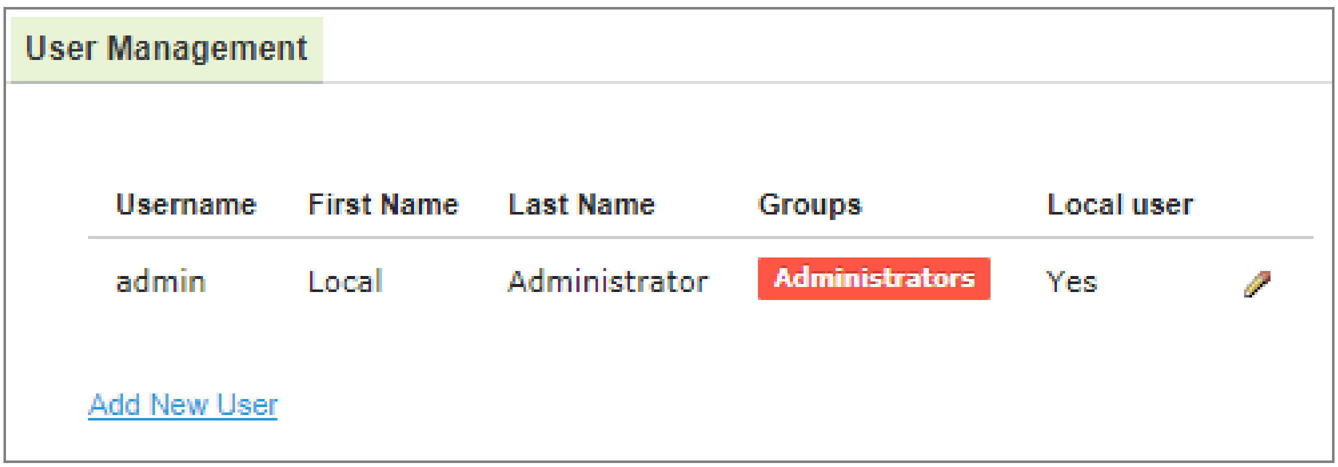



*Figure 149: Local user list*

**Note:** On the first start of OpenL Tablets WebStudio, a local user with `admin/admin` login and password and administrator's permissions is created.

If the **Local user management** option is selected on OpenL Tablets WebStudio installation:

- For SSO**,** in the **User** section, groups can be assigned or unassigned to users only.
  When creating an SSO user, the username in OpenL Tablets WebStudio must match the username in SSO.
- For Active Directory, in the **Users** section, for local users, all information can be managed, but for Active Directory users, only their groups can be updated.
  When creating an Active Directory user, the username in OpenL Tablets WebStudio must match the username in Active Directory.

A local user is added in the **Add New User** window by selecting the **Local user** check box .

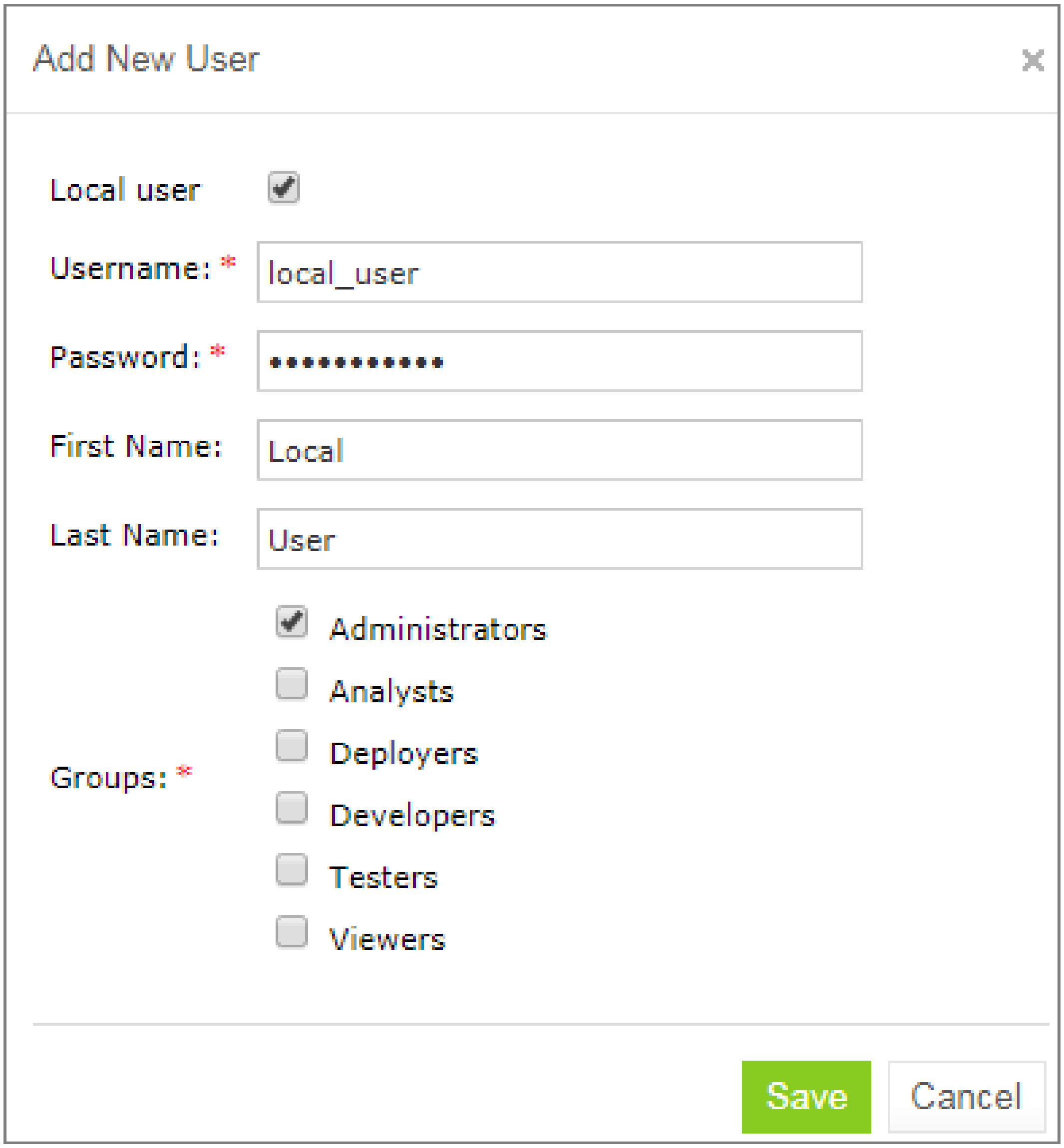



*Figure 150: Creating a local user*

# 9 Working with Design Repository via REST

This section describes how to work with Design repository via REST and includes the following topics:

- Commands for the Design Repository REST Service
- Authentication and Authorization

## 9.1 Commands for the Design Repository REST Service

OpenL Tablets WebStudio provides REST services to work with Design repository programmatically using another client application, such as cURL.

The following table describes commands available for the Design repository REST service, for the BASIC authentication method:

**Commands for the Design repository REST service**

| Method | Path | Description | Sample cURL command |
|---|---|---|---|
| GET | /rest/repo/projects | Returns a list of project descriptions. | `curl http://localhost:8080/webstudio/rest/repo/projects` |
| GET | /rest/repo/project/{name} | Returns the latest version of the specified project. | `curl http://localhost:8080/webstudio/rest/repo/project/myproject -output myproject.zip`<br>In this example, the last version of the myproject project is returned and written to the `myproject.zip` file. |
| GET | /rest/repo/project/{name}/{version} | Returns the specified version of the project. | `curl http://localhost:8080/webstudio/rest/repo/project/myproject/42 -output myproject.zip`<br>In this example, the myproject project is returned with revision 42 and written to the `myproject.zip` file. |
| POST | /rest/repo/project/{name} | Uploads a zipped project using `multipart/form-data` with a comment and name.<br>A new project or project revision is created if it exists already. | `curl http://localhost:8080/webstudio/rest/repo/project/example3 -F file=@d:\projects\Example3-datasource.jar -F comment="Initial from the script"` |
| POST | /rest/repo/project/ | Uploads a zipped project using `multipart/form-data`.<br>The project name is retrieved from `rules.xml` if it exists; otherwise, “406 Not acceptable” status is returned.<br>A new project or project revision is created if it exists already. | `curl -u admin:admin http://localhost:8080/webstudio/rest/repo/project -F file=@d:\projects\Example3-datasource.jar -F comment="Initial from the script"` |

| Commands for the Design repository REST service | | | |
|---|---|---|---|
| **Method** | **Path** | **Description** | **Sample cURL command** |
| POST | /rest/repo/lockProject/{name} | Locks the specified unlocked project for editing by other users. | `curl http://localhost:8080/webstudio/rest/repo/lockProject/myproject` |
| POST | /rest/repo/unlockProject/{name} | Unlocks the specified project locked by the current user. | `curl http://localhost:8080/webstudio/rest/repo/unlockProject/myproject` |

# 9.2 Authentication and Authorization

If OpenL Tablets WebStudio is configured to authenticate users, the REST client must provide username and a password. For example, in cURL, it can be done using the `-u` key as follows:

```
curl -u mylogin:mypassword http://localhost:8080/webstudio/rest/repo/projects
```

In this command, `mylogin` is used as a username and `mypassword` as a password when retrieving a list of projects.

- To retrieve a project list or download a specific project, the **View Projects** privilege must be assigned.
- To create a project, the **Create Projects** privilege is required.
- To update, lock, or unlock an existing project, the **Update Projects** privilege is required.

If a user has no required privilege assigned, the **403 Forbidden** HTTP status is returned.

Unlocking is possible only if the project is locked by the current user. When trying to unlock a project locked by another user, the **403 Forbidden** HTTP status is returned.

# Appendix A: ZIP Project Structure

ZIP projects described in this section can be imported into OpenL Tablets WebStudio. The following topics are included:

- Single Project Structure
- Single Project Structure #2

## Single Project Structure

A single project must be archived into ZIP file and have the following structure:

```
my-project.zip:
    rules.xml               OpenL Tablets project descriptor
    rules-deploy.xml        OpenL Tablets project deployment configuration
    *.xlsx                  Excel files with rules
```

OpenL Tablets project descriptor and project deployment configuration are optional and can be skipped in a single project structure.

## Single Project Structure #2

For a special case when an archive contains a single folder in the root, use the following structure:

```
my-project.zip:
    my-project               Folder with OpenL Tablets project inside
        rules.xml
        rules-deploy.xml
        *.xlsx
```

This type of archive is supported by OpenL Tablets WebStudio only.

# Index


## A

advanced functionality, 58

## B

benchmarking, 74

## D

deploy configuration
- creating, 92
- defining projects to deploy, 92
- deploying, 93

Deploy configurations repository
- definition, 10

deployed project
- opening, 93

Design repository
- browsing, 77
- definition, 10

## F

file
- deleting, 88
- uploading, 87

folder
- creating, 87
- deleting, 88

## G

guide
- audience, 6
- related information, 6
- typographic conventions, 6

## M

module
- opening, 23

## O

OpenL Tablets WebStudio
- components, 10
- definition, 8
- logging in, 12
- security, 11
- working with projects, 8

OpenL Tablets WebStudio, 8

## P

Production repository
- definition, 10

project
- closing, 86
- compare versions, 28
- copying, 90
- creating, 78, 82
- deleting, 91
- deploying, 91
- deployment, 92
- editing, 25
- erasing, 91
- exporting, 88, 99
- managing, 24
- modifying contents, 87
- modifying properties, 86
- opening, 84
- redeploying, 93
- removing, 91
- saving, 25, 86
- unlocking, 100

project dependency
- defining, 33

project tree
- filtering, 78

project versions
- comparing, 98

## R

**repository editor**, 20
- using, 77

Repository editor
- definition, 10

**rules editor**
- managing projects, 24
- opening a module, 23
- **overview**, 15
- using, 22
- view modes, 17

Rules Editor, 15
- definition, 10

## S

search
  advanced, 44
  performing, 43
  simple, 43
security, 11

## T

tables
  modifying, 35
  viewing, 34
tracing, 71

## U

unit tests, 64
User workspace
  definition, 11

## V

view modes, 17