# SoapUI Document

SoapUI official site document – 2019.4.25 12:22 PM

# SoapUI Projects

In SoapUI, your work is organized into projects, which are displayed under the root node in the workspace navigator. A project can contain any number of functional tests, and service simulations required for your testing purposes.

**Looking for more powerful functional test automation for your SOAP and REST APIs and web services?** Download a free trial of ReadyAPI.

# Project Formats

SoapUI offers two project formats: standalone projects and composite projects.

## Standalone Projects

The default project format is standalone. SoapUI stores them as a single XML file that contains all project artifacts – such as interfaces, tests, mock services, scripts, and so on.

## Composite Projects

SoapUI Proffers a suaim of the composite projects is team testing. These allow several people to work on the project at the same time.

# Project Types

There are three main types of projects in SoapUI:

* SOAP projects
* REST projects
* Generic projects

The project type indicates the main type of the tested service. However, you can manually combine service types in any projects you create.

## SOAP Project

SOAP projects can be created from a WSDL file or single service. You can use these projects to test every aspect of you SOAP services, verify that the services support commonly used standards such as WS-Security, WS-Addressing, and MTOM, create functional and load tests, and much more.

To learn more about SOAP projects in SoapUI, see the corresponding section of the documentation.

## REST Projects

REST projects can be created from WADL file or, directly, URI and its parameters. You can use these projects to test RESTful services, create various requests and check the information you receive, give a try to a multitude of methods and operations, and so on.

To learn more about RESTful project in SoapUI, see the corresponding section of the documentation.

## Generic Projects

Generic projects are all-purpose project for services with varied interfaces and methods. They many combine tests you create for REST and SOAP services, as well as data-driven tests, mockups, and all the functionality SoapUI provides.

# Add Project to Workspace

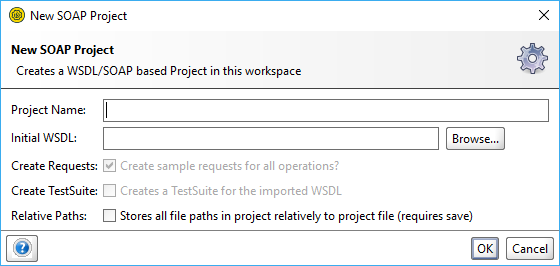
There are two ways to add a project to the workspace:

1. By creating a new project
2. By importing an existing *local*, *composite*, *packed*, or *remote* project

## Create New Project

* To create a new *SOAP project*, select **File > New SOAP Project**.

Specify the name for your new project and the WSDL file that SoapUI will use for the initial configuration, the select the necessary options.



Click **OK**. SoapUI will generate the service and, optionally, test suite and mock service.

* To create a new REST project, select File > New REST project.

Specify an URI to use for the project generation in the subsequent dialog and Click OK.

You can also Import WADL to switch to the New WADL project dialog.

With this dialog, you create a REST project based on WADL.

* To create a new empty project, select File > Create Empty Project. It will immediately appear in the Navigator, and the editor for the project will become available in the Workspace.

In ReadyAPI, you can also create projects from the OpenAPI or Swagger definition, or use the built-in API discovery feature. To try it out, download a free trial of ReadyAPI.

## Import Project

### Local Project File

To import an existing file from you computer:

1. Select File > Import Project.
2. Specify a path to the project XML file, or click Browse and navigate to the file manually.
3. Click OK.

### Local Packed Project

To import an existing packed project ZIP file from your computer:

1. Select File > Import Packed project.
2. Specify a path to the Project ZIP, or click Browse and navigate to the file manually.
3. Click OK.

### Local Composite Project

To import a composite project consisting of multiple files:

1. Select File > Import Project.
2. Specify a path to the fold that contains the project files, or click Browse and navigate to the folder manually.
3. Click OK.

### Remote Project

To import a remote project (for example, a tutorial, demonstration, bug report, and so on) using HTTP:

1. Select **File** > **Import Remote Project**.
2. In the subsequent dialog, specify an URL that leads to the project.
3. Click **OK**.

**Note**: SoapUI saves all changes you apply to the imported remote project to the local copy.

### Resolve Projects

When you import a project, SoapUI checks it to verify it is consistent and has all necessary external dependencies available. This process is called **resolving**.

If errors are found, the **Resolve Project** dialog appears. It shows all errors, so you can resolve them as you need.

To take a closer look on a specific item, double-click it in the list, and then corresponding editor will appear.

Resolve Project Dialog

In the Resolve Project dialog, you handle conflicting resources in your project when you import it.

To resolve a potential problem

1. Select a needed action from the drop-down list to the right of each item.
2. Enter parameters or values as needed.
3. Click **OK**.

### Import Postman Collection

To create a project based on the Postman Collection:

1. Select **File > Import Postman Collection**.
2. In the subsequent dialog, specify the path to the collection file.
3. Click **OK**.

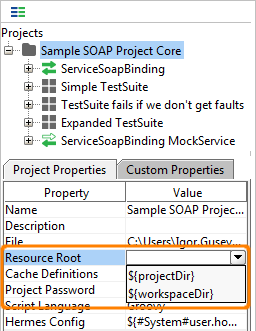
To learn more about the Postman integration in SoapUI, see Postman Collections.

# Resource Root Property

## Description

The **Resource Root** property controls how SoapUI handles paths for the project resources.

If the project contains references to external resource, they will need to be included with project when distributed.



|  |  |
| --- | --- |
| **Option** | **Description** |
| An absolute path | Use this absolute path. |
| ${projectDir} | Resolves files relative to the project folder. |
| ${workspaceDir} | Resolves files relative to the folder containing the workspace file. |

**Note**: SoapUI also applies this setting to external files the project uses.

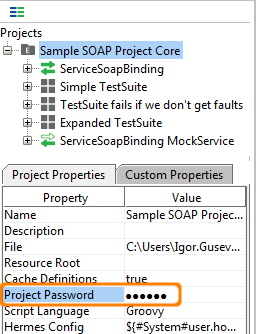
## Resource Root Examples

For example, you have the testdata.xls file you need access from the project. The d:\project folder contains workspace files, and d:\projects\webservice\ is the folder of the project.

* **Absolute path** – The d:\data path resolves to d:\data\testdata.xls.
* **Project path** – The ${projectDir}\data path resolves to d:\data\testdata.xls.
* **Workspace path** – The {workspaceDir}\data path resolves to d:\projects\webservice\data\testdata.xls.

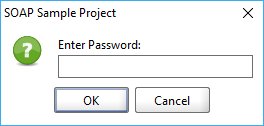
# Project Encryption

You can use the **Project** **Password** project property to encrypt the entire contents of the project file.



When you save such project, the project icon will contains a small E character to indicate it has been encrypted.

When you open the password-protected project, SoapUI will suggest to decrypt the project before using it.



To remove the encryption, clear the project password value and save the project.

**Important**: Do not lose your password. SoapUI provides no means to recover an encrypted project file if the password has been lost.

No one knowns APIs better than SmartBear. Find out what our Pro version SoapUI can do to improve your testing.