Creating a Reactive Web App - Exercise

Table of Contents

Outline

In this exercise, we will create our first reactive web application. This application will manage movies and people involved in those movies, such as cast and crew.

Over the next exercises, we will progressively build the application with new functionality, but for now we'll start by creating the application and two modules, one for the interface and the other for Logic/Entities.

- Reactive Web App Name: OSMDb_<your_initials>
- Interface module: OSMDb_<your_initials> - Reactive web
- Core module: OSMDb_Core_<your_initials> - Blank module

After the application is created, we want to create a simple action that sets a static text to a variable in the Blank module, and then display that message in a Screen in the Reactive Web module.

Resources

For this exercise, we will use an image as the icon of the application, *OSMDb-icon.png*. The image can be found in the Resources folder of the Boot Camp materials.

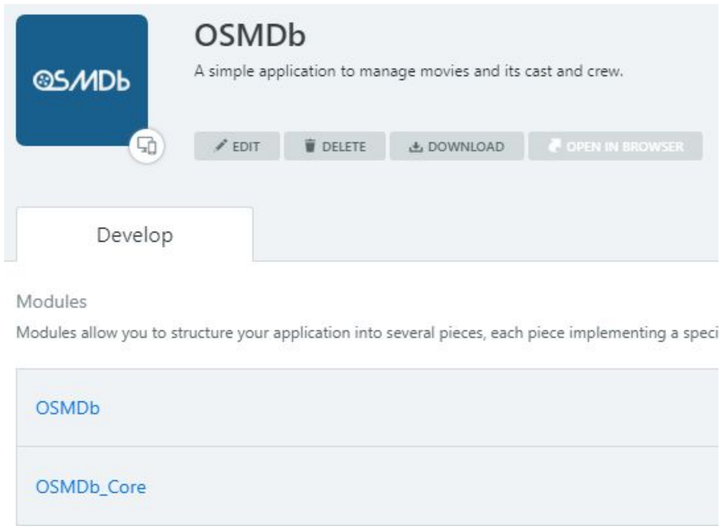
Hands-on

This is the first exercise of a set of exercises that will progressively build a reactive web application called OSMDb, a simple application to manage movies and its cast and crew.

Create the Application

In this exercise, we will create the application, using the OSMDb icon and with a simple description of what the application does.

This application needs to have two modules: **OSMDb**, a reactive web module where the screens and the UI of the application will be created; **OSMDb_Core**, a blank module where the database tables and some of the business logic will be defined. Since the Boot Camp scenario is shared between all the students, *the application and module names should be followed by your initials* to distinguish them.



OSMDb

A simple application to manage movies and its cast and crew.

Modules

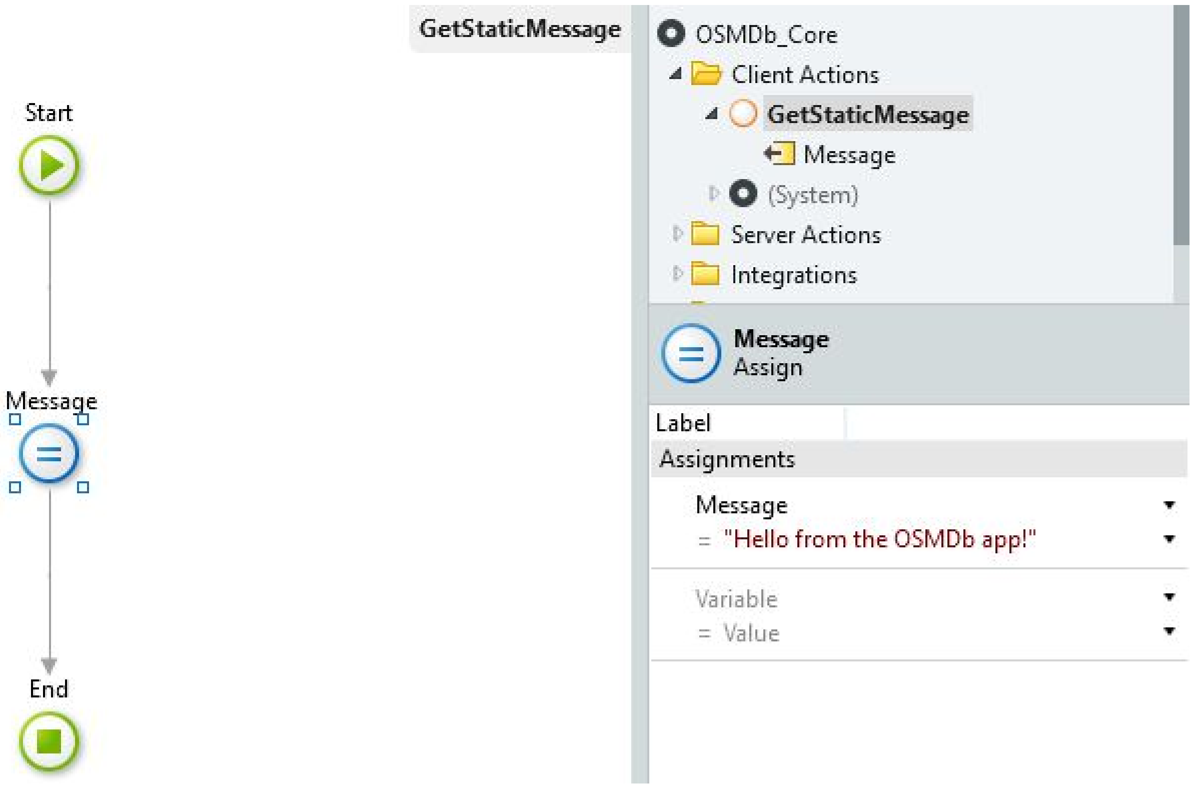
Modules allow you to structure your application into several pieces, each piece implementing a speci

OSMDb

OSMDb_Core

Create the First Client Action

In this exercise we will also start sharing features between modules. We'll create a Client Action in the OSMDb_Core Module that will simply set an output parameter to a static Text: the *GetStaticMessage* Action. Having the Action and the output parameter, *Message*, we need to use an Assign to accomplish that.



Message
= "Hello from the OSMDb app!"

This Action will be used in the reactive web module, to display the message directly on a Screen. For that, the Action should be **Public**, to make it visible and usable in your Reactive Web module, and set as a **Function**. You can see these properties by clicking on the GetStaticMessage Action.

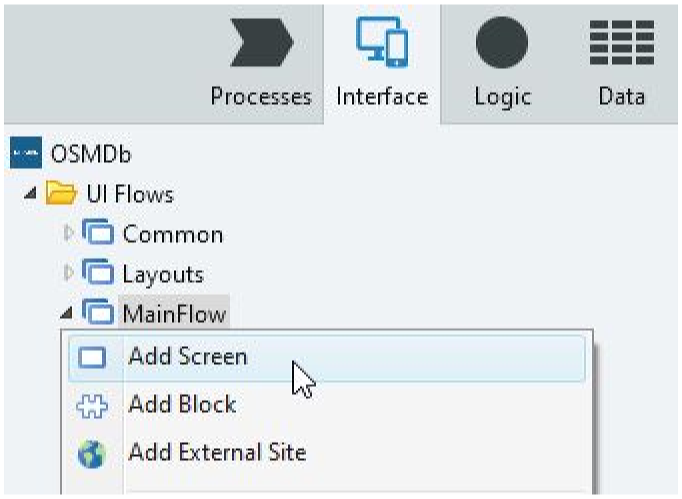
After it's done, don't forget to publish the module to save the changes, by clicking on the 1-Click Publish button at the top of the Screen!

Reference the Action in the UI module

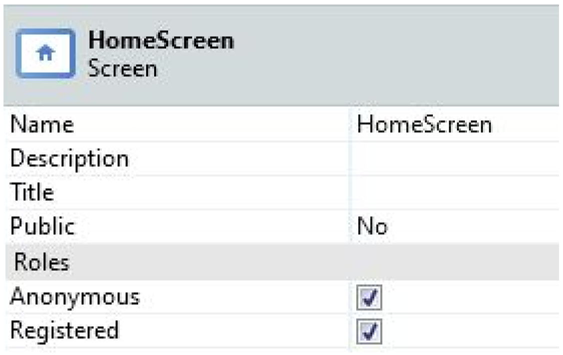
Now, on the Reactive Web module, we want to display this static message on a new Screen.

First, we need to open the OSMDb module and reference the Action by using the Manage Dependencies option (it will only be visible if the Core module was published!).

Then, we need to create an Empty Screen and use an expression to display the value in the output parameter of the Action.



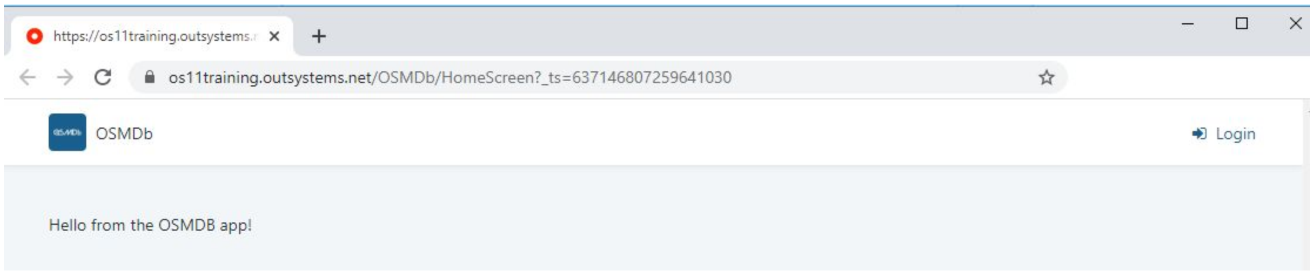
At this time, we're not concerned about security issues, so remember to check the Anonymous checkbox on the screen properties.



HomeScreen
Screen

Name	HomeScreen
Description	
Title	
Public	No
Roles	
Anonymous	✓
Registered	✓

At the end, we can publish the module and open it in the browser. The Screen should look like this:



os11training.outsystems.net/OSMDb/HomeScreen?_ts=637146807259641030

OSMDb

Login

Hello from the OSMDB app!