SCP3C01 – Creating a HANA Cloud Platform Developer Account

|  |  |
| --- | --- |
| **Product**  HANA Cloud Platform  **Level**  Undergraduate/Graduate  Beginner  **Focus**  HANA Cloud Platform  **Author** Ross Hightower | MOTIVATION  This case describes registering for a developer trial account on SAP HANA Cloud Platform configuring the WebIDE and creating a first application.  **PREREQUISITES**  None |
|  | |

# Creating an Account

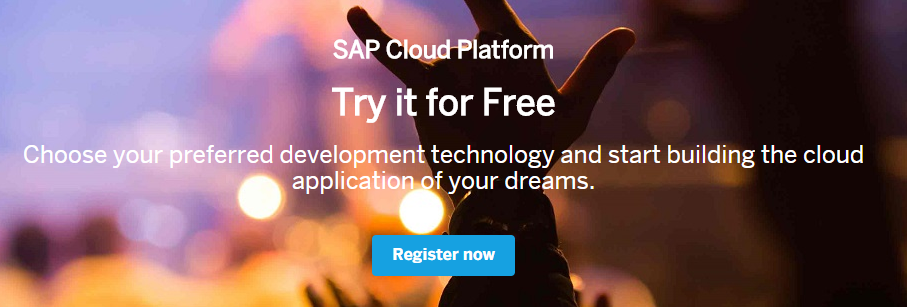
The **HANA Cloud Platform** trial account is a free developer account. It has no time limit but has certain limitations such as a 1 GB storage limit. One significant limitation from a instructional point of view is that it only allows one account making it difficult to share the content you create. Otherwise, it provides almost all the capabilities of the full HANA Cloud Platform. In this section, you will register for an account and then configure the WebIDE development platform

## Registering for an Account

Navigate to <https://cloudplatform.sap.com/index.html>. If you already have an id on the SAP Community Network, you don’t need to register. You can logon with your SCN credentials.



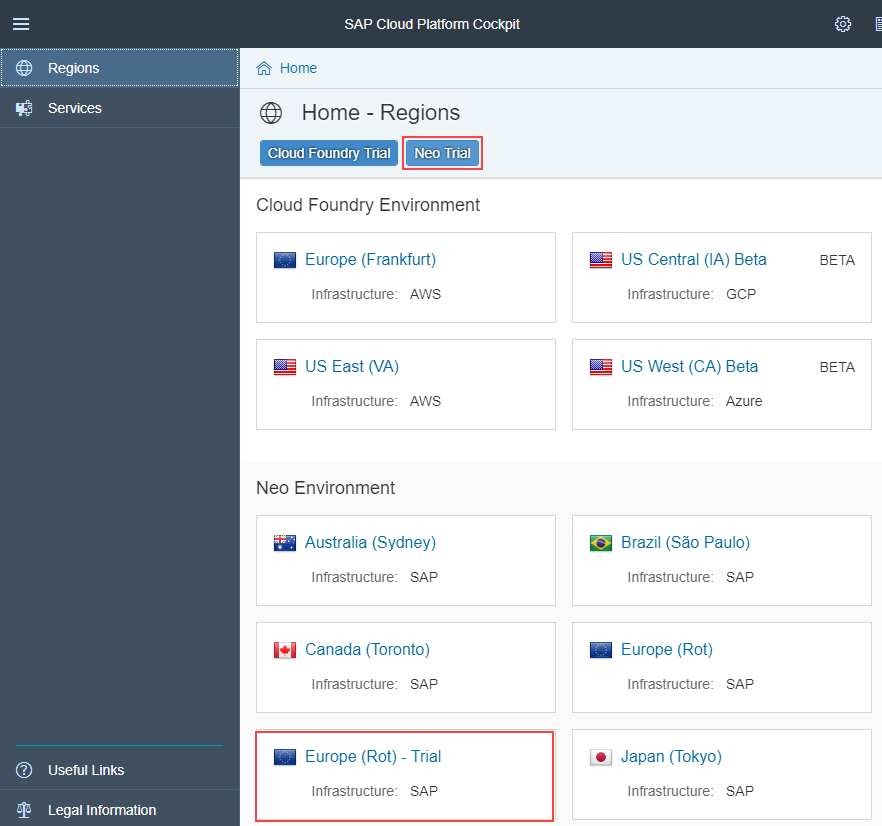
If you don’t have an SAP account, click on **Register now**.



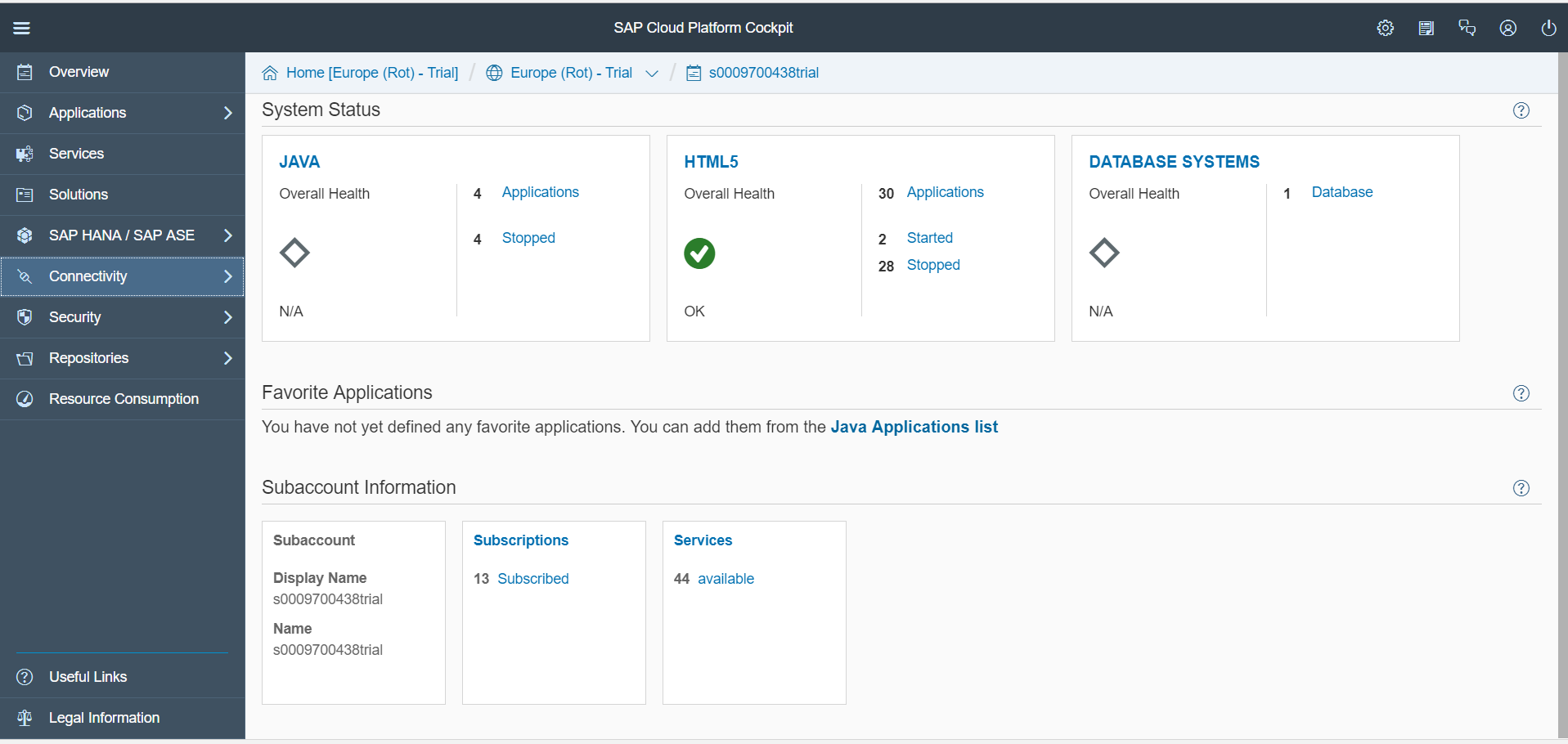
Once you register, you will receive an email with a link to activate your account.

Click **Log On** and enter your credentials when prompted.

The first screen you will see shows the various hosting centers around the world. Click the Europe (Rot) – Trial tile.

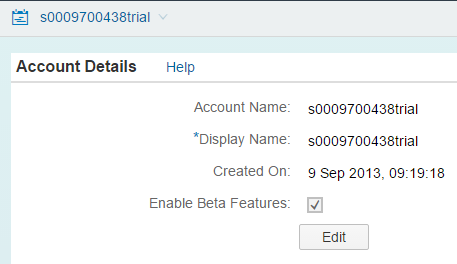


This will bring you to the SAP Cloud Platform Cockpit.



The menu on the left side of the screen is called the navigation area.

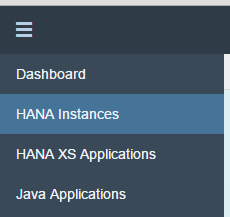
Click the Account menu item near the bottom of the screen.



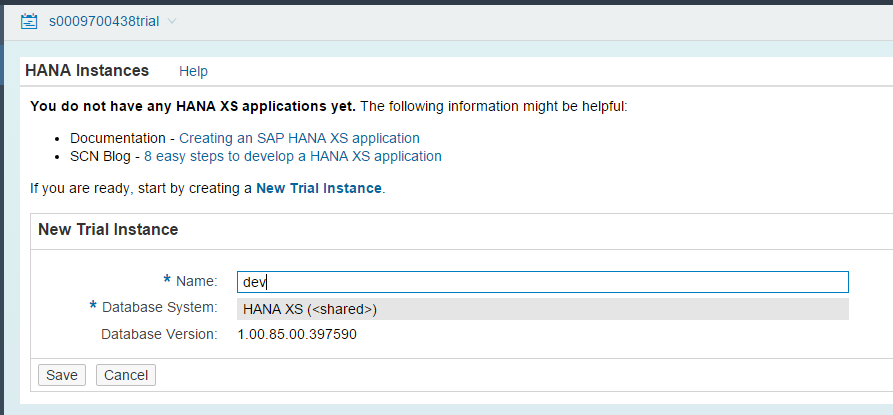
The account name is your SCN id plus the workd trial.

## Create a HANA Instance

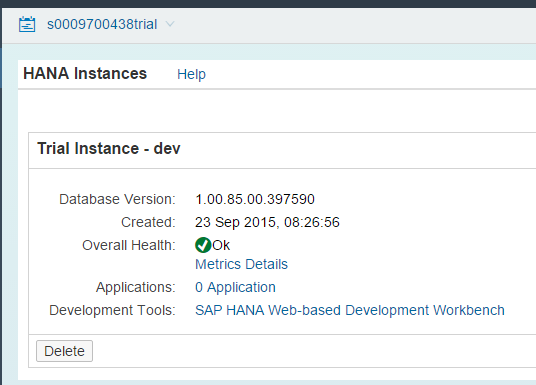
The first step is to create a HANA instance. Click on **HANA Instances** in the list on the left side of the screen.



Click **New Trial Instance,** enter the name **dev** and click **Save**.



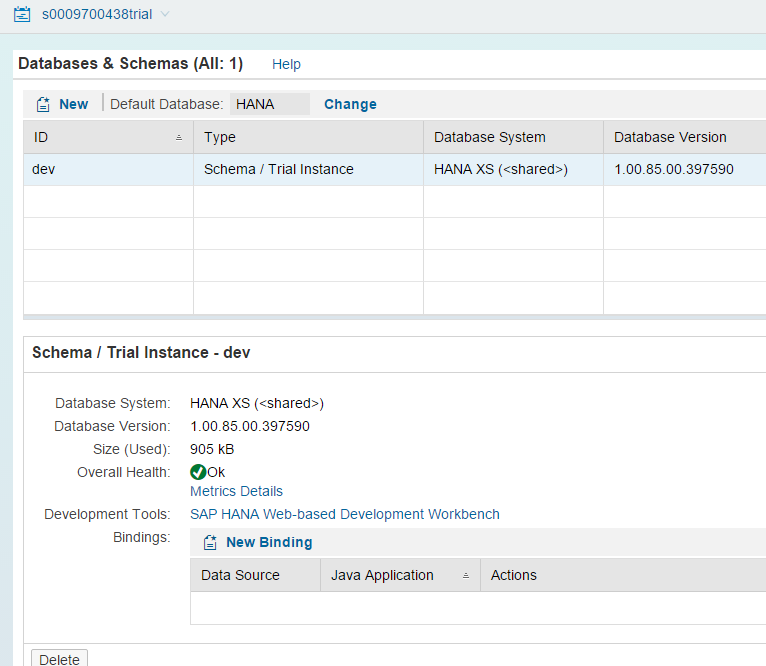
The instance is created.



Click on **Dashboard**. Let’s take a look at some of the screens you can access via the navigation area.

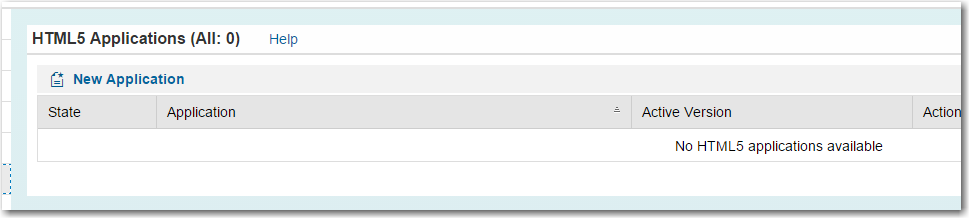
### Databases and Schemas

The database instance was created when you created the HANA instance. One of the limitations of the trial account is a 1GB limit on the database. If, at some point in the future, you wish to start over, you can delete the database instance and create a new one.



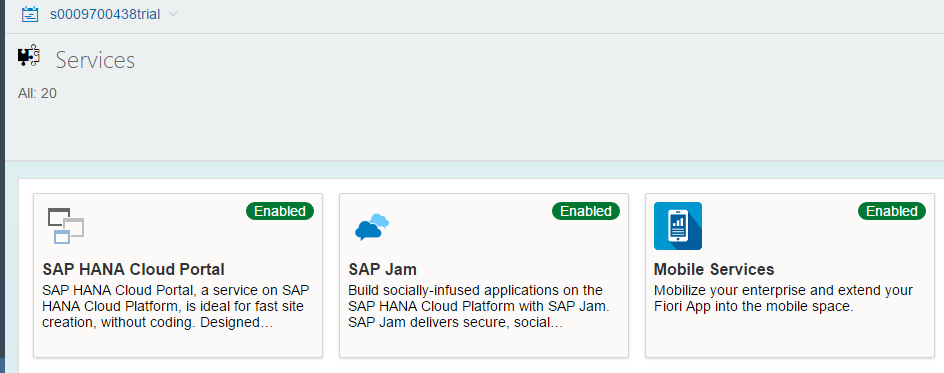
### HTML5 Applications

This screen will show applications that you deploy in later cases.



### Services

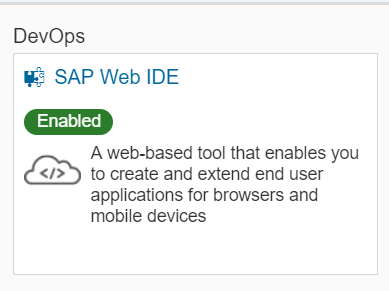
Services provide additional functionality. Some of these are already enabled and you can access the service’s documentation from this screen. Some must be enabled. Later cases will use some of these including the SAP HANA Cloud Portal.



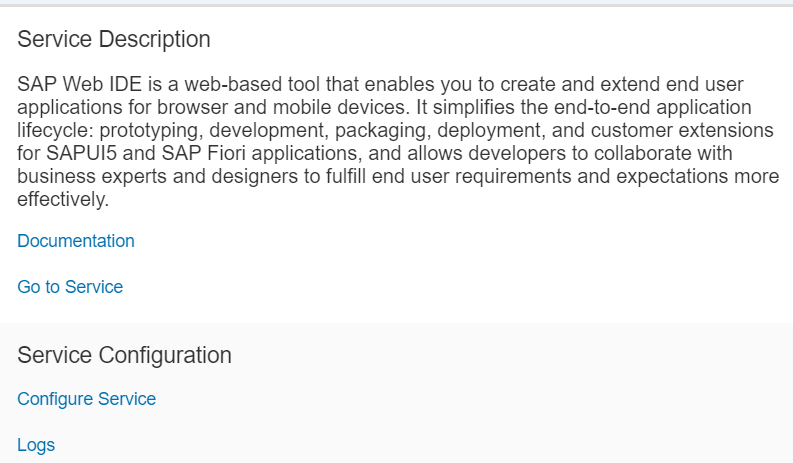
## Open the WebIDE

The WebIDE is the development platform used in this curriculum. You can develop using Eclipse as well which provides a more polished enviroment. However, SAP has made it known the WebIDE is their platform of the future and it offers a number of conveniences when developing in the cloud.

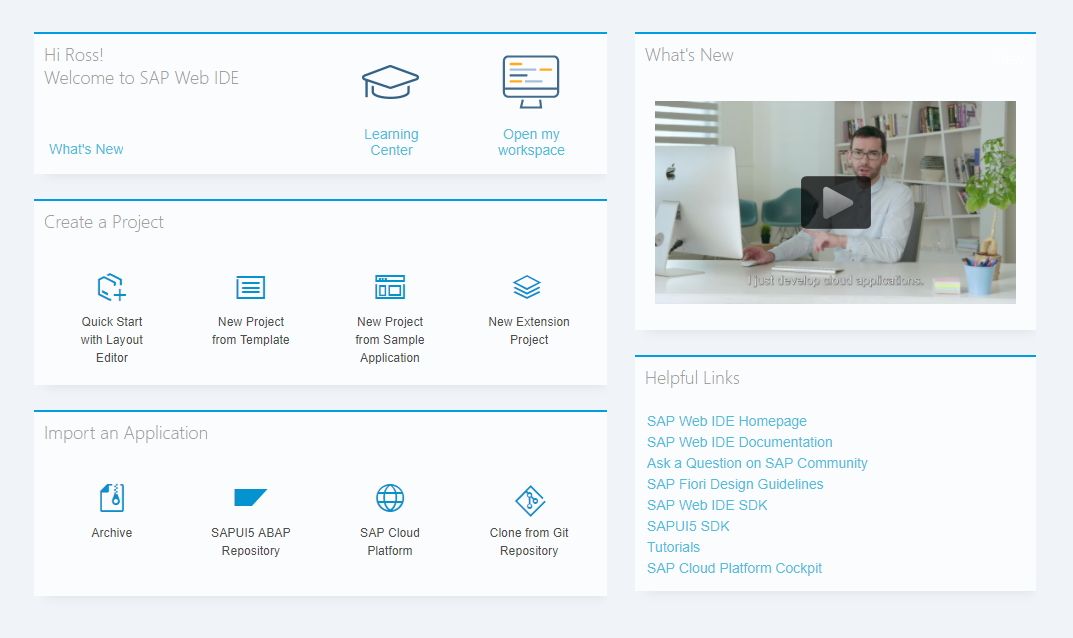
Click on **Services** in the navigation area. Scroll down or use the search box to find the Web IDe tile. Click the tile.



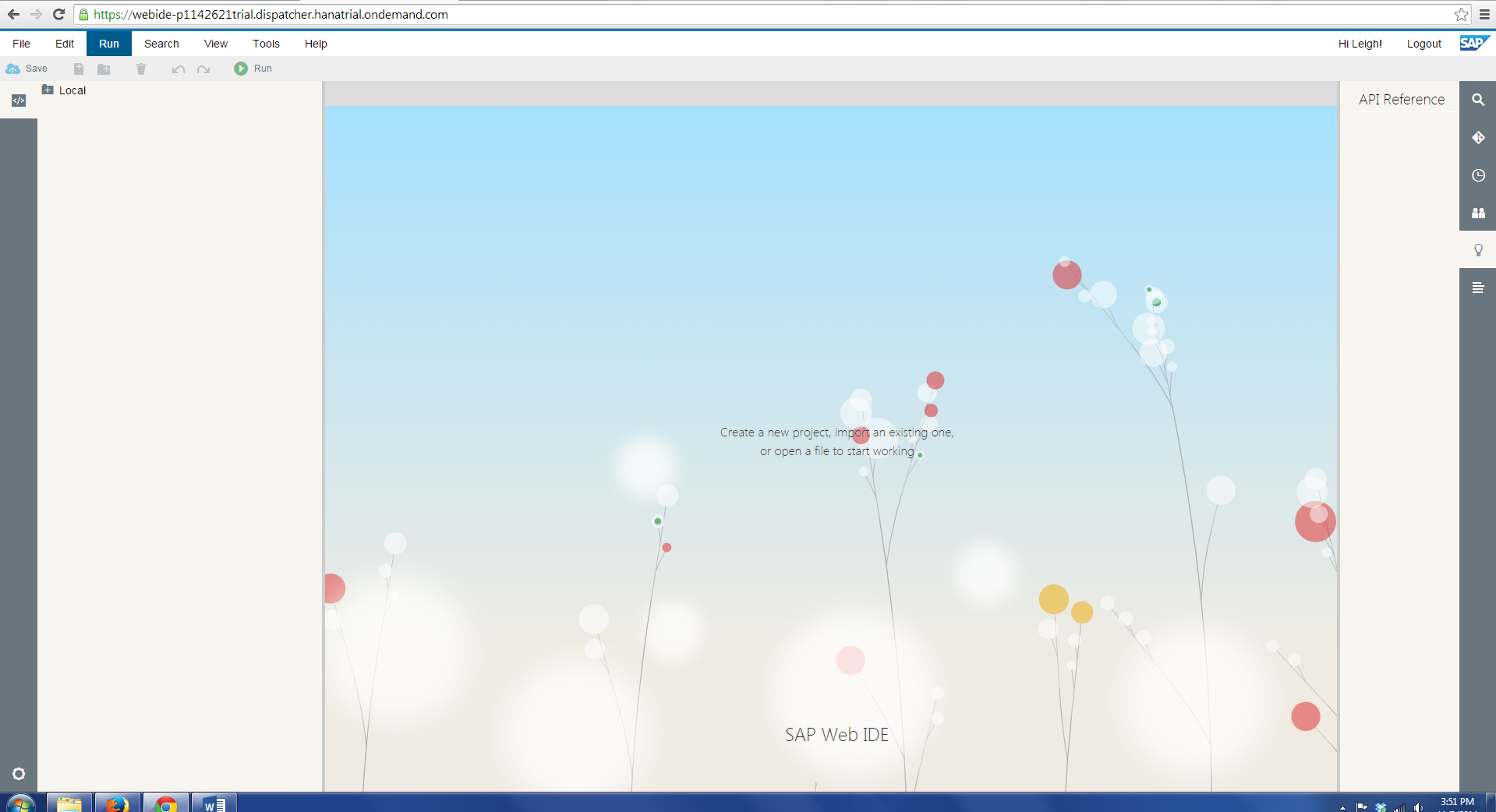
Click **Go to Service**.



The first thing that opens is the Welcome screen. You can bookmark this link to return directly to the Web IDE.

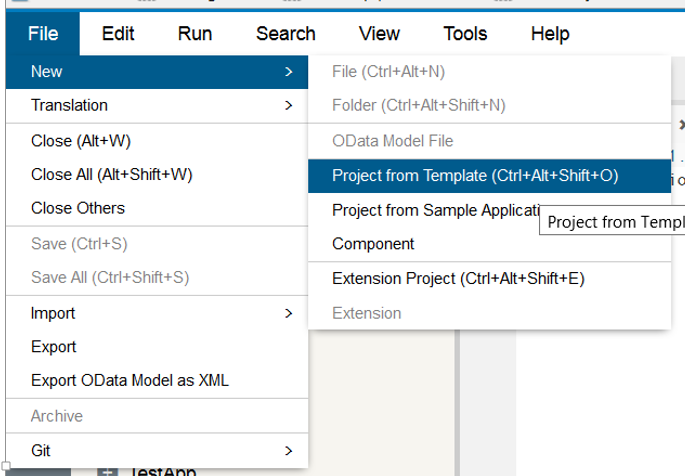


Click the  icon in the toolbar on the left side of the screen to open the editor

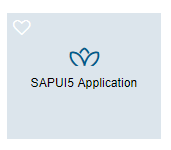


# Create the Hello World App

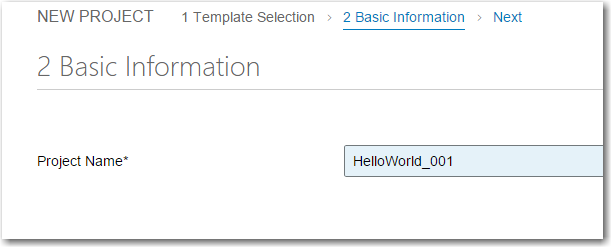
The traditional first app in programming environments is the Hello World app. This is a simple app that is used to demonstrate the mechanics of creating applications. In the WebIDE, click **File→New→Project From Template.**



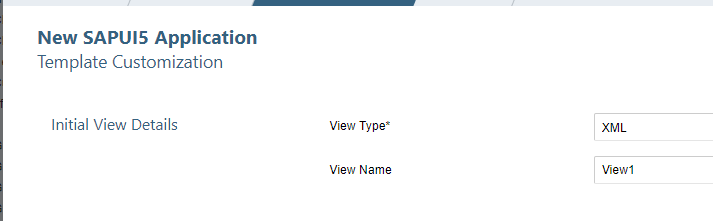
Select the **SAPUI5 Application Project** and click **Next**.



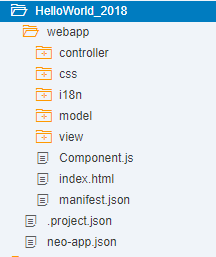
Enter a project name as **HelloWorld\_###** where ### is the three digit id assigned by your instructor:



Leave the Template Customization defaults as they are and complete the wizard:



The project is created.



## Model-View-Controller

The model-view-controller or MVC convention divides the application components according to their function:

**Model** – The model accesses and manages the data. In SAPUI5 the data in models can be bound to controls in views so that the flow of data between the model and the interface is handled automatically but SAPUI5. You can defined models based on XML, JSON or oData data sources in SAPUI5.

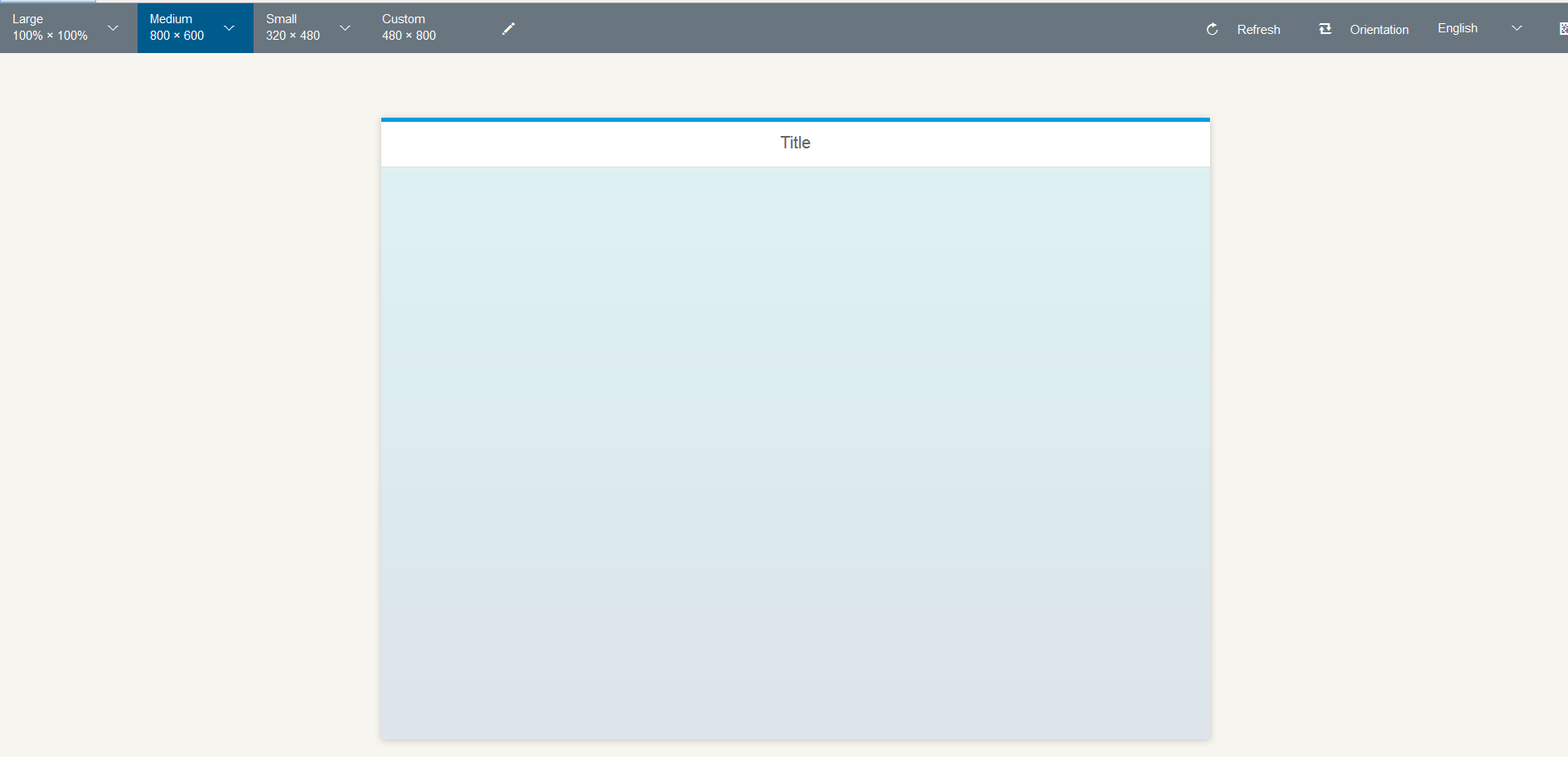
**View** – The view is used to create the user interface. SAPUI5 allows you to define views in JavaScript, XML or JSON. The recommended method is to use XML because it reinforces the separation of the application logic in the controller and the use interface in the view. Also, the WYSIWIG editor under development works with XML views. Fiori apps use XML views.

**Controller** – The controller is used to define the application logic. In SAPUI5, the controllers are defined in JavaScript and are tied to specific views.

You can find more information about MVC [here](https://help.sap.com/saphelp_nw74/helpdata/en/91/f233476f4d1014b6dd926db0e91070/content.htm?frameset=/en/95/8ead51e2e94ab8bcdc90fb7e9d53d0/frameset.htm&current_toc=/en/5c/be4e5b4a19479a92b1d32ff23b7b63/plain.htm&node_id=60&show_children=false).

## Run the Application

You can run the application by selecting and part of the project and clicking the run icon .



You can view the application in different screen sizes and orientations. If you click the QR Code icon on the top right, you will shown a QR Code you can use to open the app on your phone or tablet.

## Add Some Content

Next, you’ll add little content to make it more interesting.

### View1.view.xml

Open the View1.view.xml which is located in the webApp/view folder of your project and add the code shown below between the <content> and </content> tags.

|  |
| --- |
| <Button text="Hello XML View!" press="doIt" /> |



The next case provides some explnation of the structure of the code in views.

### View1.controller.js

Add the following code to the View1.controller.js file.

|  |
| --- |
| doIt : function(oEvent) {    jQuery.sap.require("sap.m.MessageToast");  sap.m.MessageToast.show("Pressed: " + oEvent.getSource().getId());    } |



Now refresh or run the application again.

|  |  |
| --- | --- |
|  |  |