

PROGRESS EXCHANGE 2013

DISCOVER. DEVELOP. DELIVER.

OpenEdge Mobile

An introductory guide to the fundamentals of building mobile applications using OpenEdge Mobile 11.3

Executive Summary

Using a timesheet entry system as an example, we'll take you through an entertaining step-by-step guide to let you build your first Mobile OpenEdge application.

In this workshop, you'll learn how to:

- > **Articulate your ideas** - Easily design interactive mobile user interfaces appropriate for different users running on different devices.
- > **Build your application** – Using the power of OpenEdge Mobile you'll soon be building a real mobile application.
- > **Create new opportunities** – OpenEdge Mobile opens the door not just to mobile devices but to new business opportunities.

It's as easy as learning your A, B, C's!

Table of Contents

Executive Summary	1
Table of Contents	2
Preface	3
Introduction	4
Timesheet Overview	6
Chapter 1 – Setting up your environment	7
Chapter 2 – Building the Mobile UI	22
Chapter 3 – Building the Business Logic	47
Chapter 4 – Binding the Mobile User Interface to the Business Logic	56
Chapter 5 – Packaging and Deploying your application for specific devices	68
Chapter 6 – OpenEdge Mobile Express	70
Conclusion.....	74
Appendix A – Code pieces	75
Appendix B – Tips and tricks	78
Appendix C – Getting a Progress login.....	79

Preface

This guide is designed to quickly introduce you to the basics of building mobile business application using the power of OpenEdge Mobile. We'll only cover some of the core functionality in this guide, but remember OpenEdge Mobile is a powerful platform for developing mobile applications.

HOW TO USE THIS GUIDE

This guide is designed to flow from one lesson to the next, with each lesson building on the previous. Starting with a brief overview of the basics of building a mobile user interface, we will then connect what you have built to some real OpenEdge business logic, and finally we will use the built in functionality to deploy and execute your application so you can use it instantly from your mobile device.

To best leverage this guide, it is recommended you perform the operations as you read them, taking breaks between lessons as required. After you've completed all the lessons, the guide will make for a practical reference that you can dive into at any point to refresh yourself on the concepts.

CONVENTIONS USED IN THIS GUIDE

This document uses the following conventions to distinguish elements of text:

- > **Bold** – Indicates new terms and titles of commands
- > ***Bold italics*** – Notes and tips that alert you to important information.
- > **Bold Courier font** – Text that you should type.
- > *Footnotes* – Expand on details and used to define terminology (e.g. timesheet entry terms)

Introduction

JUST WHAT IS A MOBILE APPLICATION?

A mobile application is an application that can be run on a mobile device, like a smartphone or tablet computer. There are three types of mobile applications:

- **Mobile Web Applications**
This type of application is based on HTML5/JavaScript/CSS and simply runs in a web browser. This application can potentially also be used from a desktop machine when started in a web browser. Although this is a very flexible way to make the application available to users – you only have to provide the URL – it has its limitations. Accessing most of the device specific capabilities, such as camera, microphone, contact, etc. are simply no options. Most web browsers now support geolocation and there's also some level of off line support as part of HTML5. But they aren't real apps with their proper icons and can't be installed from e.g. AppStore.
- **Hybrid Applications**
When you use this approach, you start by writing a mobile web application, but deploying this in a native device container, you can deploy it as a real app on e.g. Android or iOS. This entails that you also have access to device specific capabilities, like the camera, microphone, contacts, off line usage, etc. Best of all is that you write the app once and you can deploy it on different platforms.
- **Native Applications**
This kind of applications requires you to learn the development environment of the platform you are developing for. So this means that if you want/need to deploy on different platforms, you have to develop multiple apps. It does allow you to take full advantage of all the features of the device platform you are developing for.

WHAT IS OPENEDGE MOBILE?

OpenEdge Mobile takes the Hybrid approach since research has shown that this is the best approach for business applications. The write once, deploy anywhere capability combined with being able to use the native services of the different deployment platforms gives OpenEdge a huge advantage. OpenEdge Mobile allows you to talk to your existing business logic through RESTful services that sit on top of the OpenEdge AppServer. RESTful services have been added in the OpenEdge 11.2 release.

MUCH MORE THAN A USER INTERFACE DESIGNER

OpenEdge Mobile is more than just a user interface designer:

- > **Articulate ideas** – You can use the Mobile App Builder to build interactive prototypes of your mobile application. With this you can very quickly validate your design against the user's expectations with regards to look and feel, user experience, usability design, etc. without having to build the complete application.

- > **Build the application** – Once you have the “go” from your customer or end user, you can complete the application by building/integrating the mobile app with backend business logic.

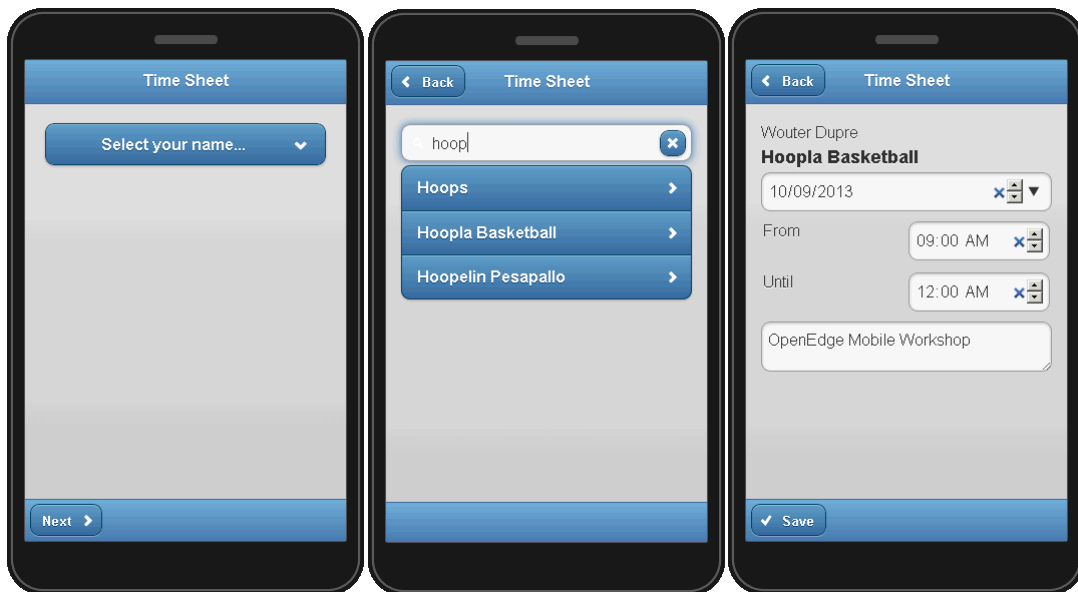
- > **Create business opportunities** – Smartphones, tablets and mobile applications are a part of our world today. App stores provide new ways to reach new global markets. It's a new way of doing business where high quality and high volume at low cost determine the success rate of apps.

Timesheet Overview

ABOUT THE TIMESHEET ENTRY APPLICATION USED IN THIS GUIDE

OpenEdge Mobile is robust enough to develop and prototype virtually any business focused mobile application, from supply chain management and customer care, to order taking and inventory management. To best illustrate this, we'll take you through an example that describes how an employee can enter their timesheets for the work they've done. While we make no claims as to the veracity of the example, it will nonetheless provide you an excellent grounding in the mobile application builder tool that you can abstract upon to quickly build and test your own business focused mobile applications.

To give you an idea of what you're going to build, here's an example of what the finished mobile application will look like:



This is an overly simplified business application that lets an employee enter their timesheet. An employee will select their name from a list of colleagues, they will select the customer that they are currently doing work for and finally will enter some details relating how much time they have been working for this customer.

As you can see, this is not meant to be a fully-featured mobile application, and indeed this workshop is not intended to be a full training class; but by following this document you will gain experience on how to work with OpenEdge Mobile to extend your existing OpenEdge applications or even to build new ones.

Chapter 1 – Setting up your environment

Getting started with Progress Arcade

WHAT IS PROGRESS ARCADE?

Progress® Arcade™ is a web portal where you can deploy and manage Progress applications in a Cloud-based environment. You can use Arcade to migrate existing Progress applications to a public Cloud environment, run and test the application in the Cloud, and then deploy the application. Deployment includes the ability to host demos or to publish applications that are available to subscribers.

Progress Arcade is also a site where you can interact with other Arcade users, and find out about services offered by various Progress Technology Partners.

HOW DO I ACCESS THE FEATURES OF PROGRESS ARCADE?

The features of Progress Arcade are grouped into functional areas, represented by six panels on the Arcade home page. The panels include:

- > **Stage & Test** – Gain experience in running your application in the public cloud.
- > **Deploy** - Deploy your application into full production for customer use directly from the public cloud (not currently available).
- > **Demo** – Publish your application for demonstration purposes using the public cloud.
- > **Expo** – Search for information on complementary products and services offered by the Progress community.
- > **Community Café** – Access Progress product resources and network with others in the Progress community.
- > **Product Showroom** – Learn about the features and benefits of Progress products. Specify what technology you are interested in, and within minutes get a public cloud-based machine dedicated to your exclusive use.

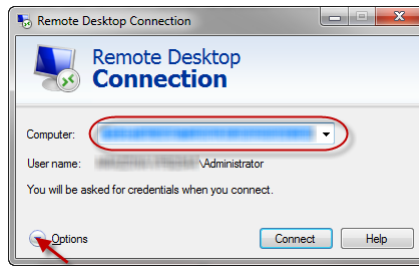
PROGRESS LOGIN CREDENTIALS

In order to be able to do this workshop, you'll need a Progress Login (PSDN, Communities, ESD,...). If you don't have one yet, don't worry. Just go to Appendix C – Getting a Progress login and follow the described steps.

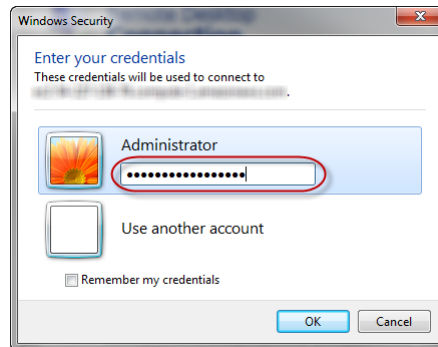
LAUNCHING YOUR OPENEDGE MOBILE DEMO MACHINE ON PROGRESS ARCADE

To launch your OpenEdge Mobile Demo machine:

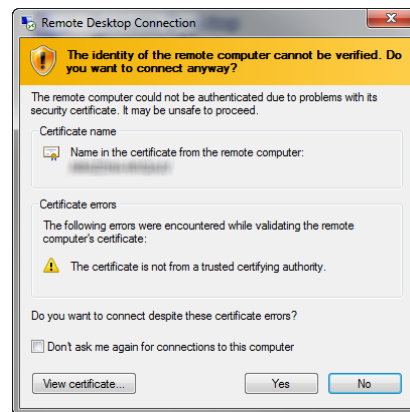
1. Several machines have been started for this workshop. Each of you will be given a unique DNS address by the workshop host at this point which points to a virtual windows server running on Progress Arcade.
2. On your laptop, choose **Start | Accessories | Remote Desktop Connection** and paste your unique DNS into the **Computer** field. Click the **Connect** button. You may need to click the button left of Options to change (erase) the domain name.



3. Enter **Administrator** as the Username, and **Exchange2013** as the Password.



4. Click the **Yes** button to say you trust this remote connection.



5. You should now be connected to the demonstration machine running in Progress Arcade.

Note: The machine instance has may have a different keyboard setting than what you are using. If your laptop has a different keyboard, you may need to set this up yourself on the machine instance through **Control Panel| Region and Language** settings.

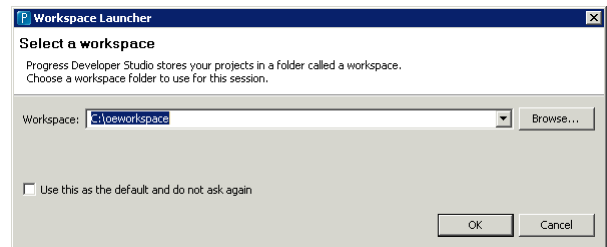
Launch and Configure the Development Environment

STARTING PROGRESS DEVELOPER STUDIO

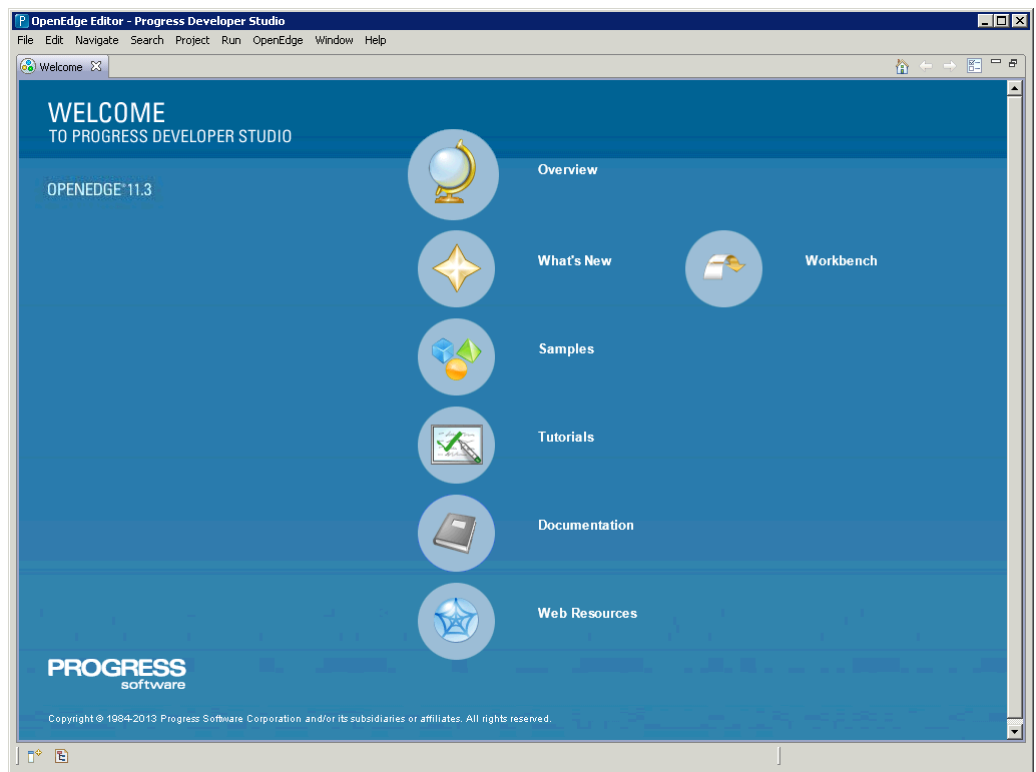
1. Start Progress Developer Studio:
Start → All Programs → Progress → OpenEdge 11.3 → Developer Studio

The dialog Workspace Launcher will pop up

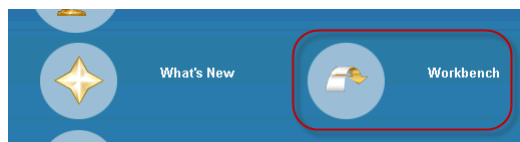
2. Create a new workspace:
C:\oeworkspace



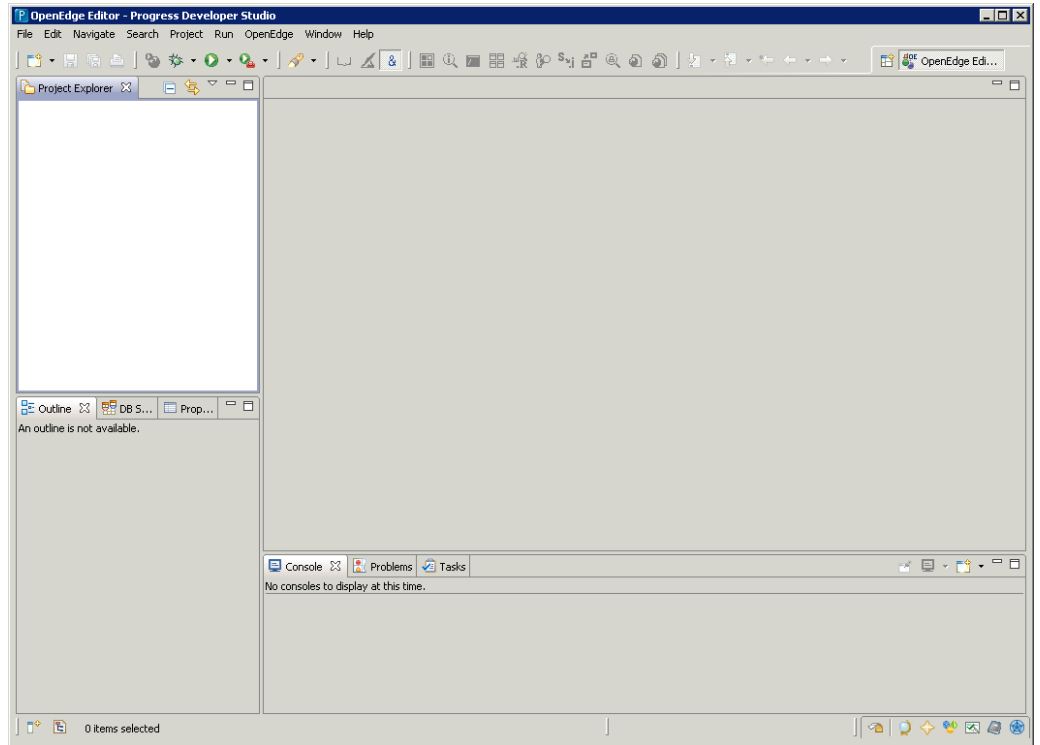
3. After loading the environment, the Welcome screen will appear.



4. Open the Workbench.



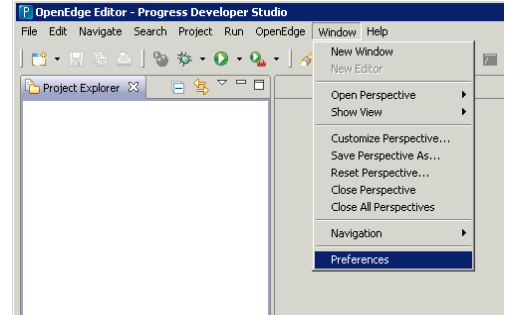
5. The Workbench will appear.



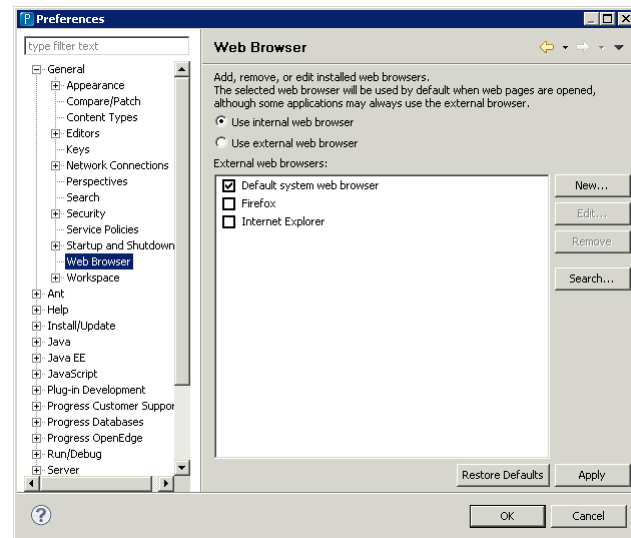
ENVIRONMENT CONFIGURATION

Now we have to set up our environment for the application we are going to develop.

1. Go to *Window* → *Preferences*

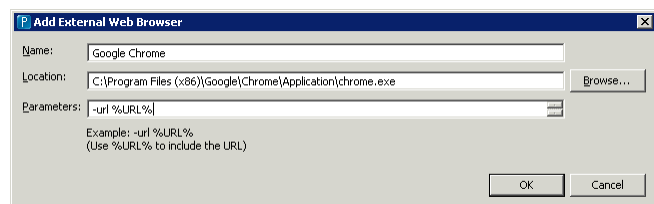


2. Go to *General* → *Web Browser*

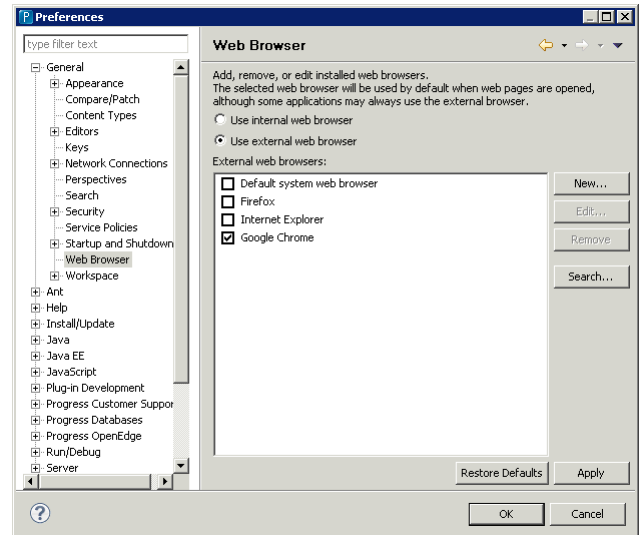


3. Click on the **New...** button to add a new External Web Browser
Enter the following details:

- Name:
Google Chrome
- Location:
**C:\Users\Administrator\AppData\Local\Google\Chrome
Application\chrome.exe**
or
**C:\Program Files
(x86)\Google\Chrome\Application\chrome.exe**
- Parameters:
-url %URL%

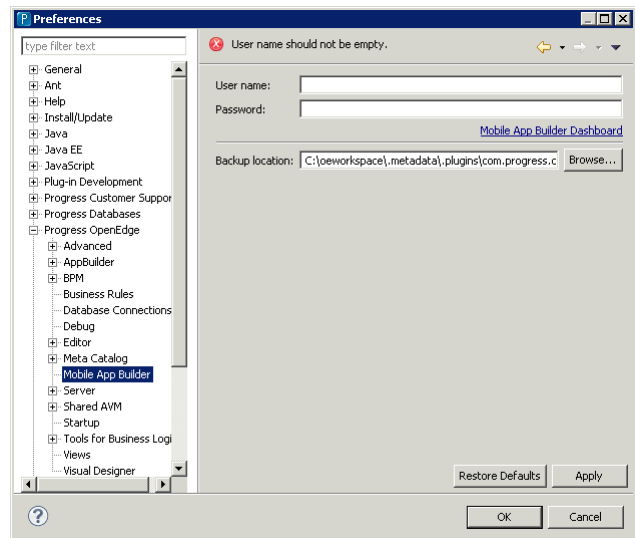


4. Click **OK**, select **Use external web browser**, check **Google Chrome** and click **Apply**



5. Go to *Progress OpenEdge* → *Mobile App Builder*
Enter your credentials and click **Apply**
The credentials should be your PSDN or Progress Communities User name and Password.

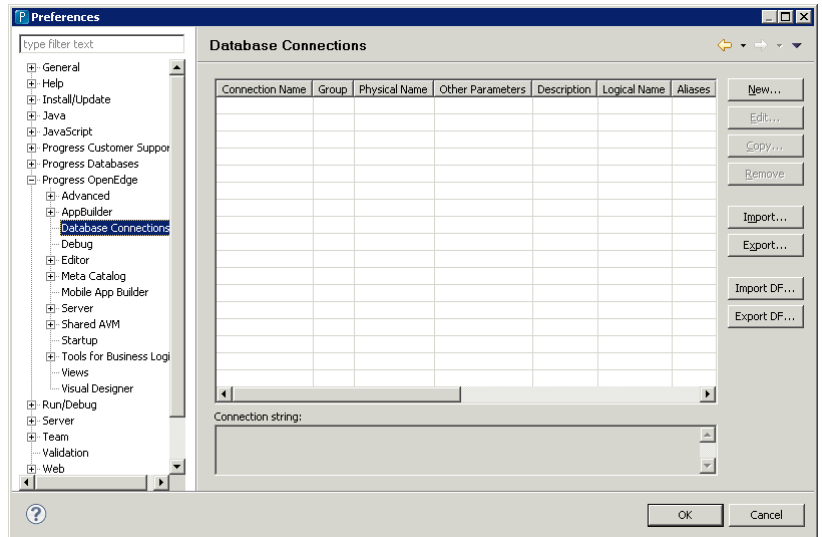
When you click **Apply**, your credentials will be validated, so if you don't get an error message, your credentials should be ok.



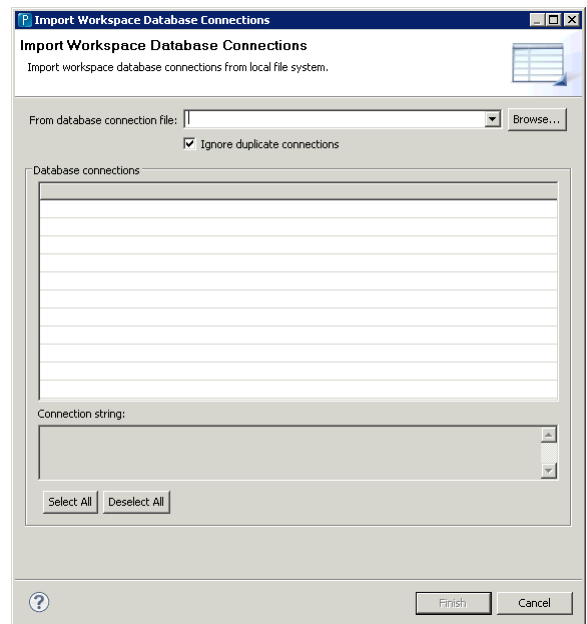
In some cases you may need to log in to the Mobile App Builder Dashboard with your credentials first before you can continue. You can use the link below the password for this.

6. Go to Progress OpenEdge → Database Connections

Click **Import...** to import database connection profiles

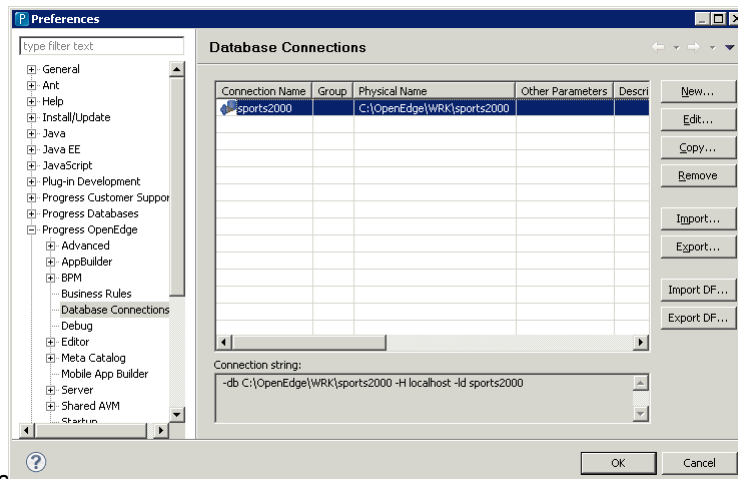


7. Use the **Browse...** button to find the file **databaseconnections.xml** in the folder **C:\OEMobile** and click **Open**

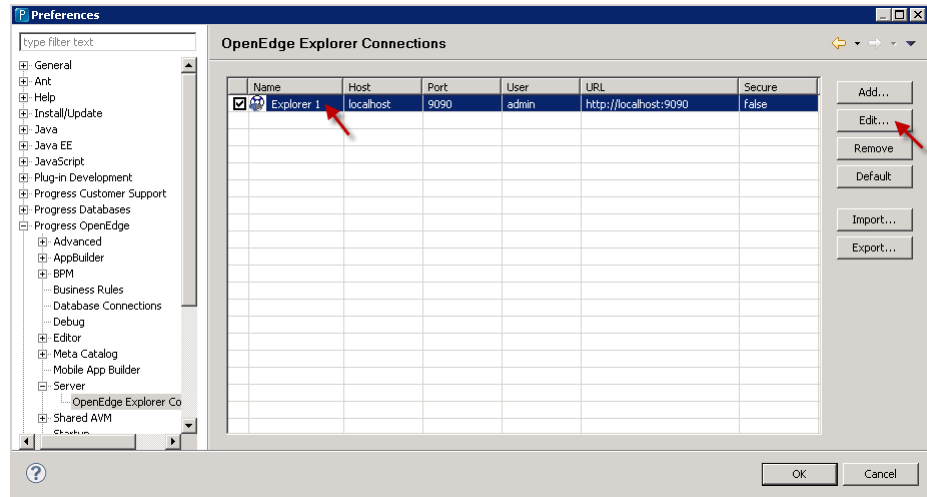


You should see the sports2000 database connection and a marked checked box at the beginning of the line.

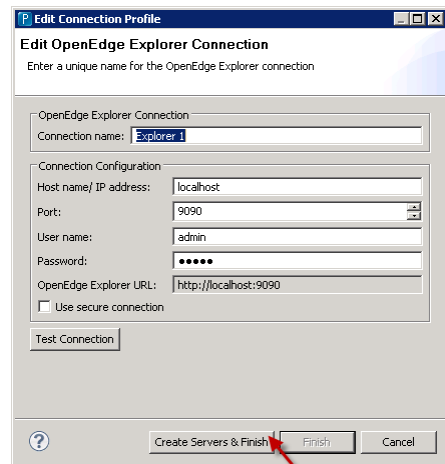
8. Click **Finish** and you should see the following screen.



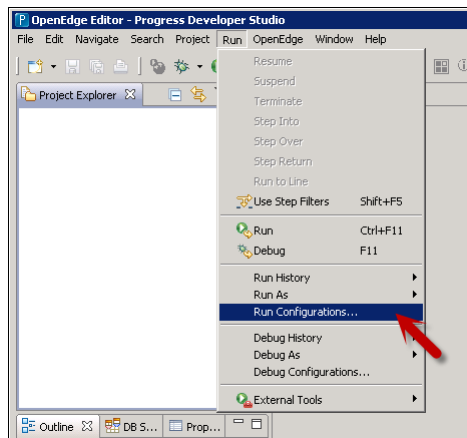
9. Go to the node **Progress OpenEdge → Server → OpenEdge Explorer Connections**
10. Click on **Explorer 1** and then **Edit...**



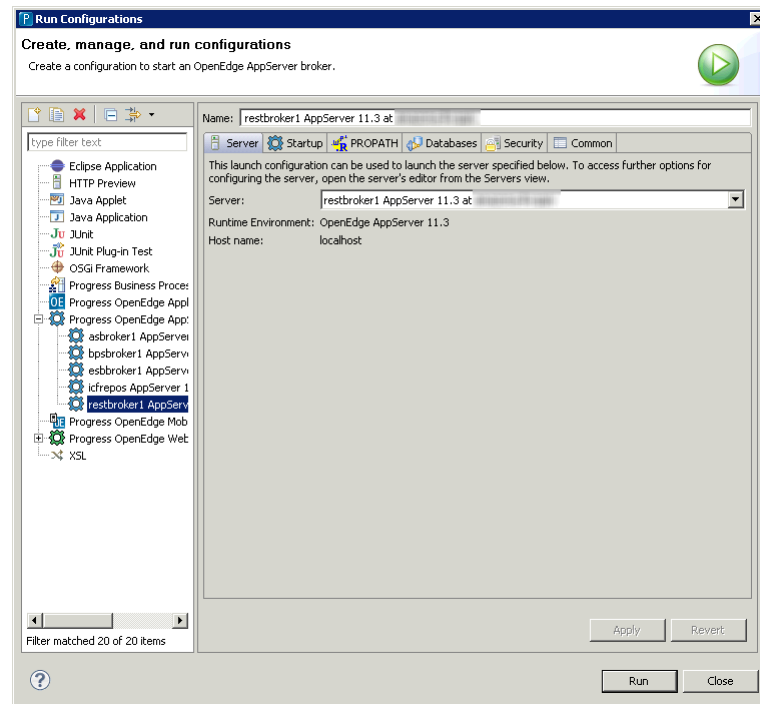
11. Click **Create Servers & Finish**



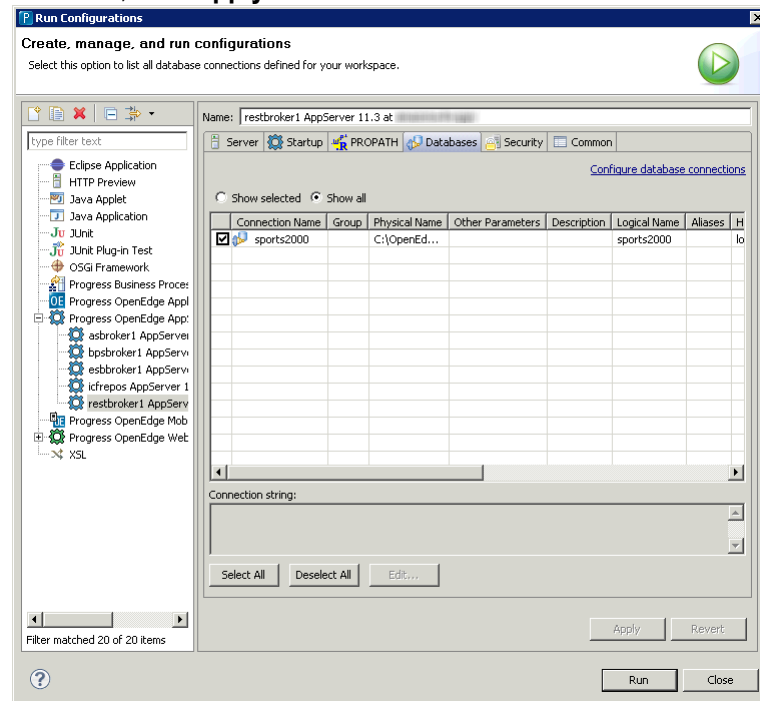
12. Click **OK** to close the Preferences window.
13. Go to **Run → Run Configurations...**



14. Select **restbroker1 AppServer** as shown here:



15. Select the **Databases** tab, click **Show all**, check the **sports2000** database, click **Apply** and **Close**.



16. Before you can run your Mobile app (later on), you must start the OE Web Server. To start the web server, open the **Servers** view:

Window → Show View → Servers

or if it's not listed there

Window → Show View → Other → Server → Servers

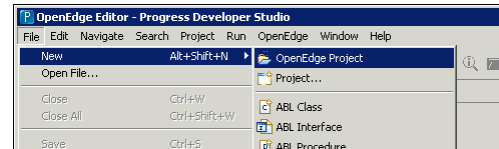
17. In the **Servers** view, right-click on **restmgr1 OE Web Server**, and choose **Start**. This will take a couple of minutes, and you can watch the progress in the bottom right corner.

Creating an OpenEdge Project

Now we have the development correctly configured we're ready to start to build our OpenEdge Mobile application. Firstly, we need to create an OpenEdge project to store our application in.

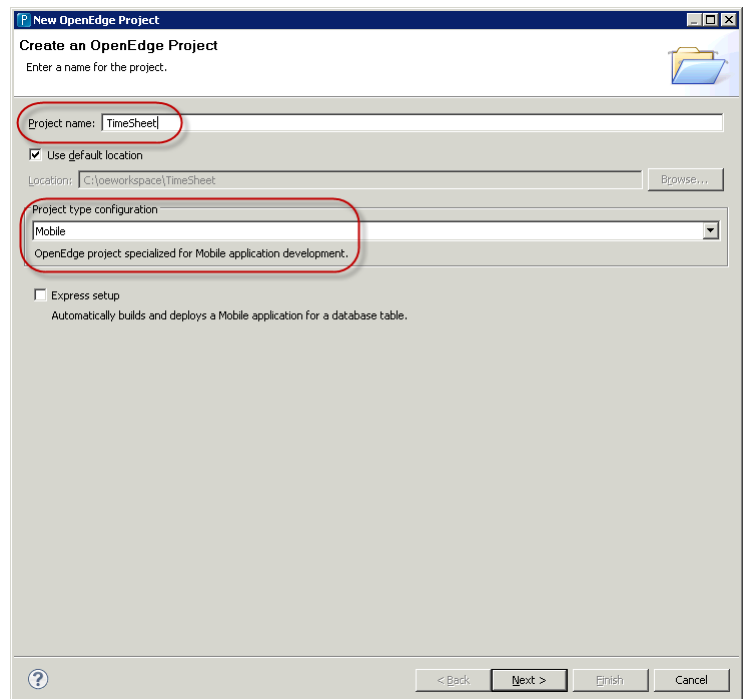
1. Click **File** → **New** → **OpenEdge Project**

The New OpenEdge Project dialog appears.

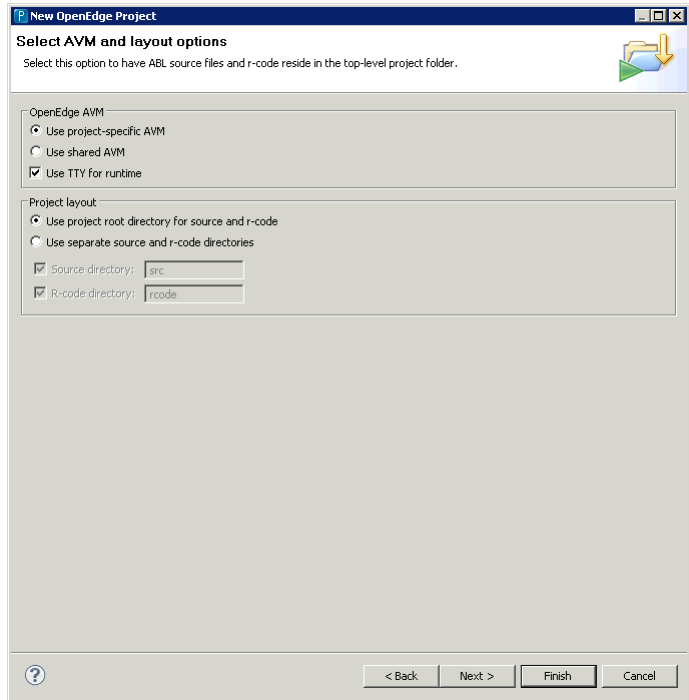


2. Enter the Project name, select the Project type configuration and click **Next >**

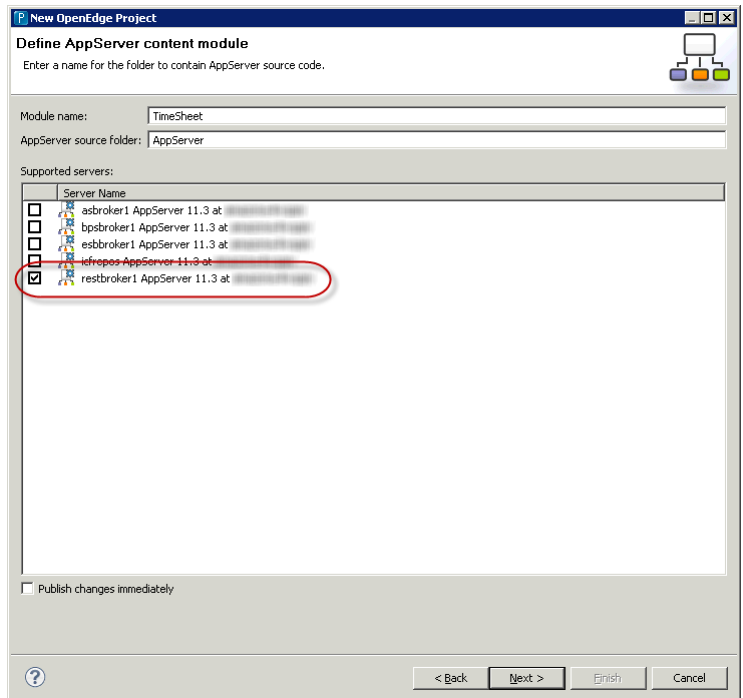
- Project name:
TimeSheet
- Project type configuration:
Mobile



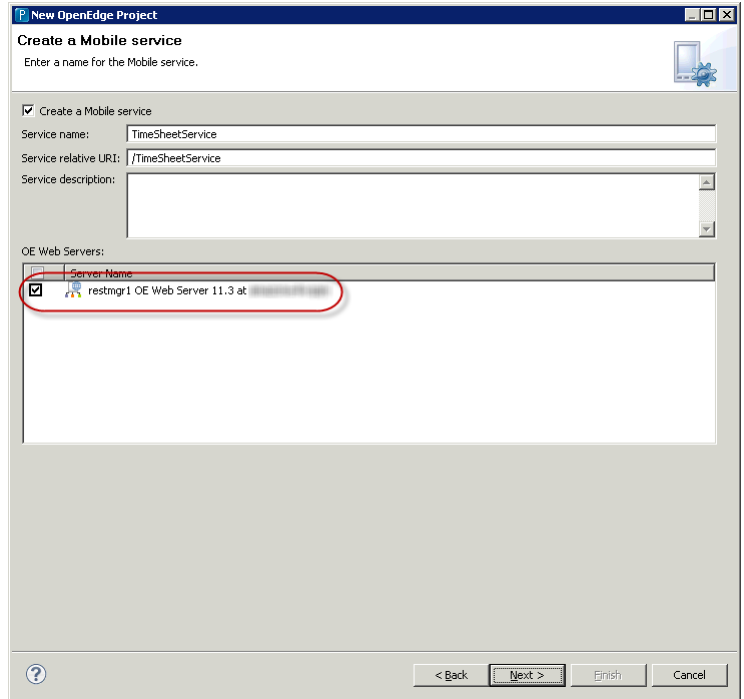
3. On the Select AVM and layout options page, click **Next >**



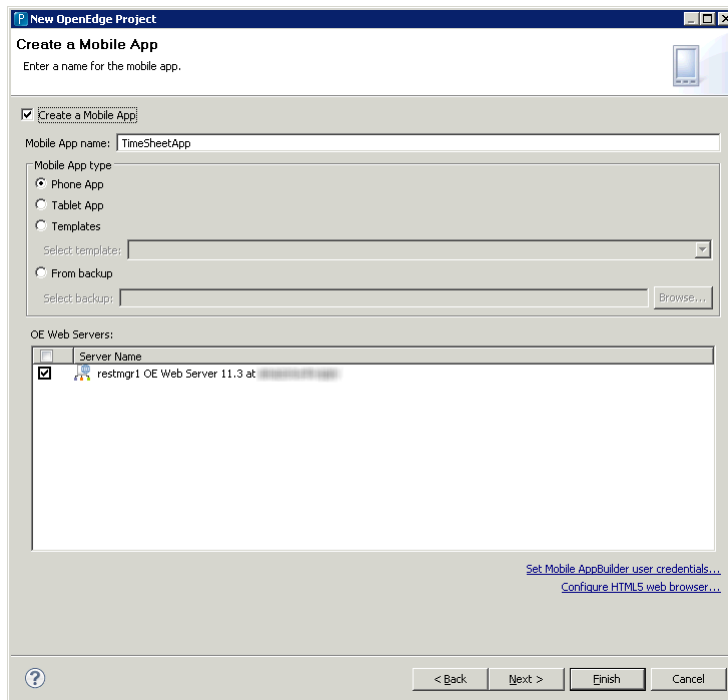
4. On the Define AppServer content module page, check **restbroker1** and click **Next >**



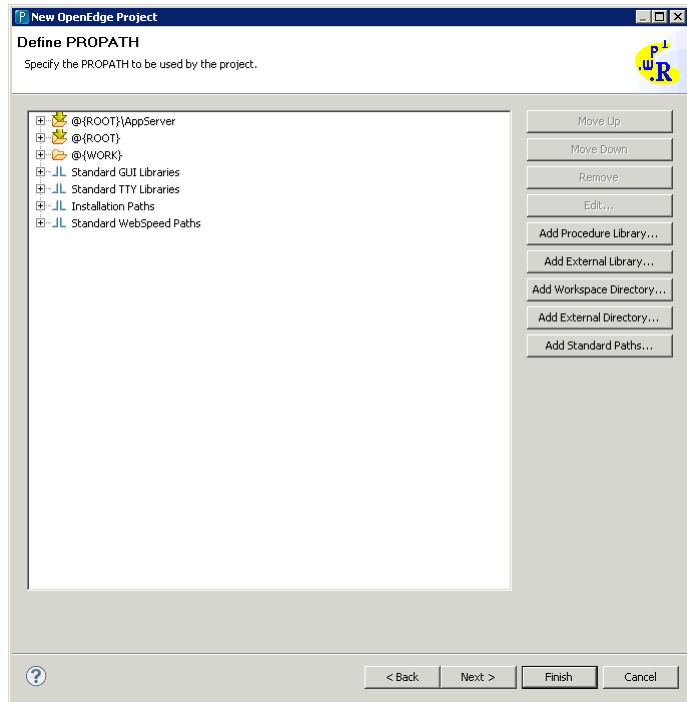
5. On the Create a Mobile service page, check **restmgr1** and click **Next >**



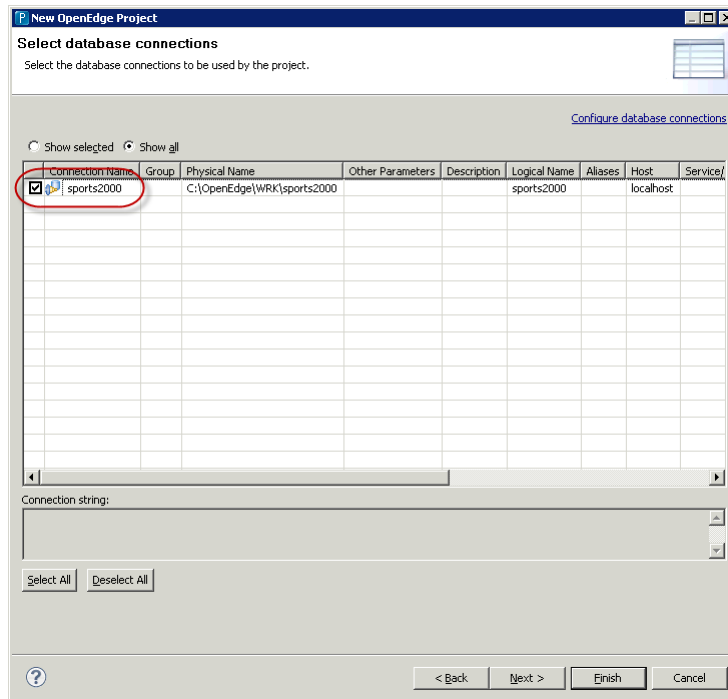
6. On the *Create a Mobile App* page, check **restmgr1** and click **Next >**



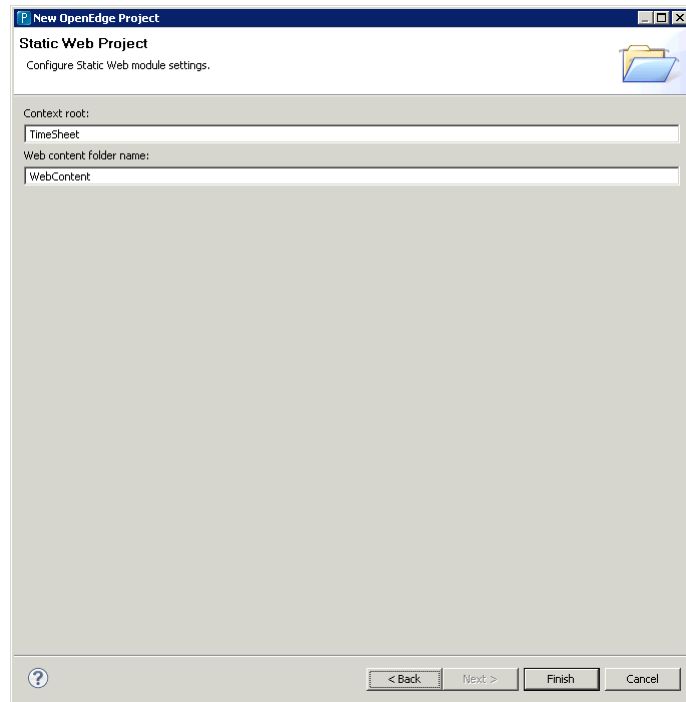
7. On the *Define PROPATH* page, click **Next >**



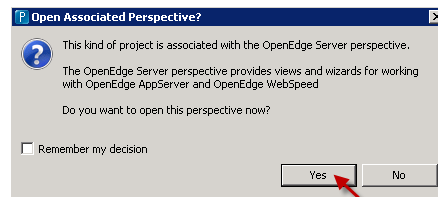
8. On the *Select database connections* page, check sports2000 and click **Next >**



9. On the *Static Web Project* page, click **Finish**.



After you have clicked **Finish**, You may see a dialog asking to open the associated perspective. Click **Yes**.

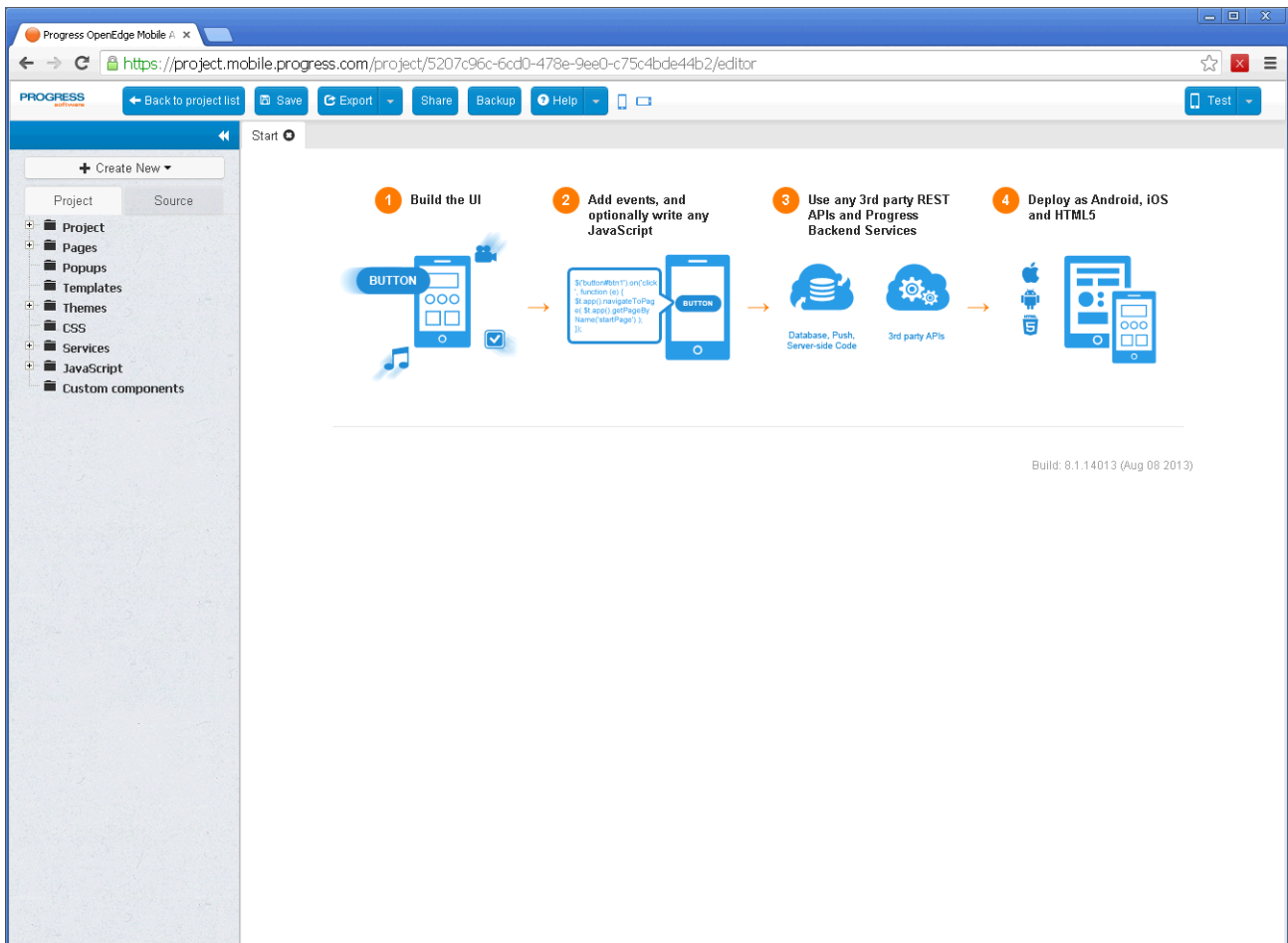


The OpenEdge Mobile App Builder will start once you've entered your credentials.

Chapter 2 – Building the Mobile UI

Creating Pages

When you have correctly entered your credentials for the OpenEdge Mobile App Builder, after ‘Loading awesome stuff...’. You should will the following screen:

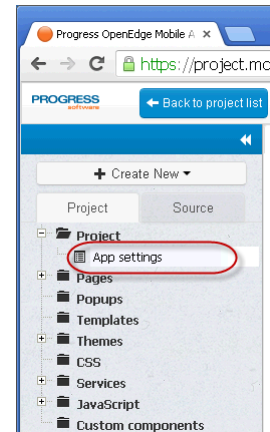


There's a toolbar at the top of the environment. The Save button will be the most often used button. On the left you have your project overview in a treeview format. This tree can be collapsed using the << button at the top.

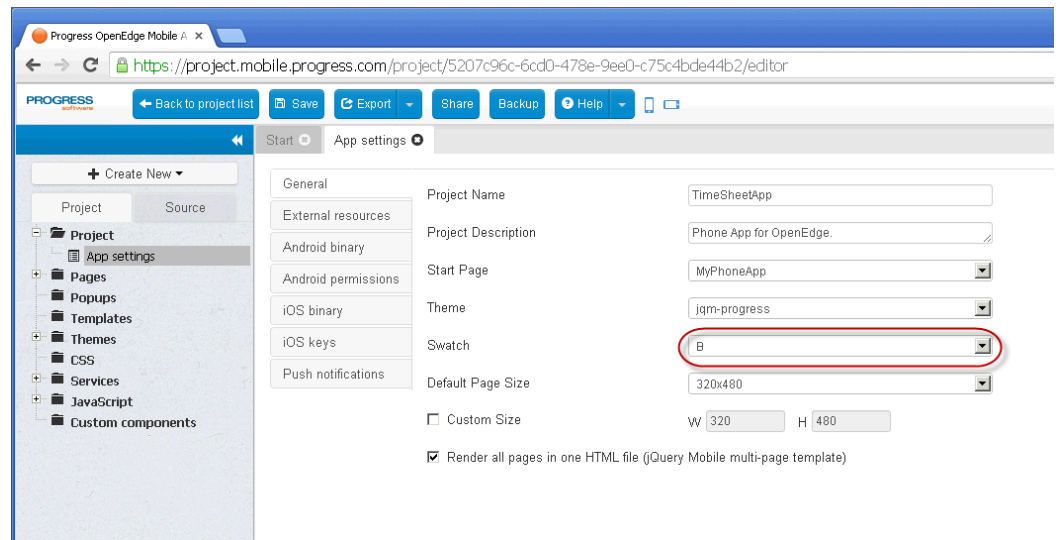
THE HOME PAGE

To continue the creation of the Mobile UI, follow the steps below:

1. Expand the **Project** node and click on **App settings**

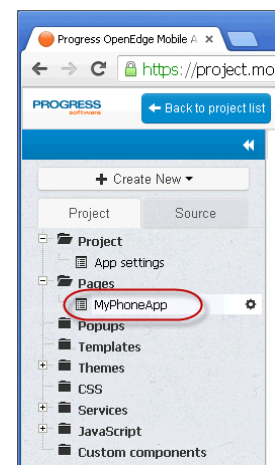


This will open a new tab page with the name **App settings**:

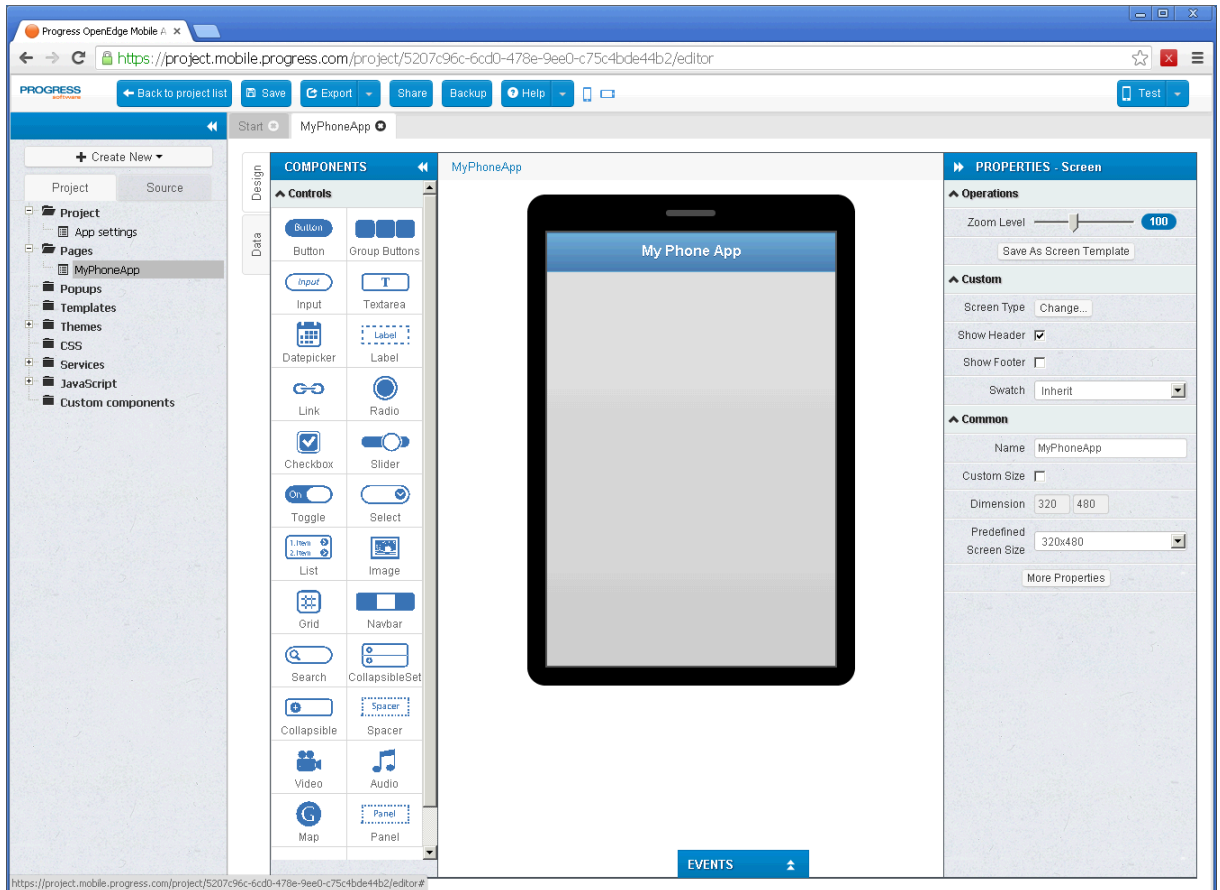


2. Change the Swatch to **B** and close the App settings tab. This will change the default color scheme of the application.

3. Expand the **Pages** node and click on **MyPhoneApp**



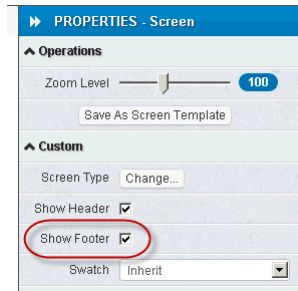
This will open a new tab page with the name **MyPhoneApp**:



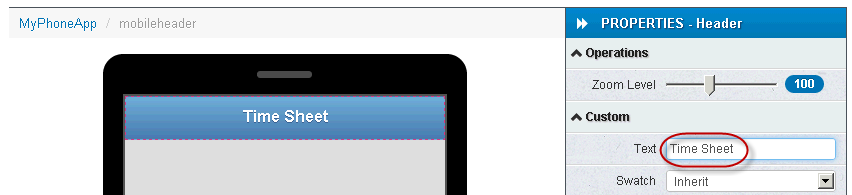
You'll notice some important parts on the screen:

- Collapsible panels:
 - a. COMPONENTS (expanded, on the left)
 - b. PROPERTIES (expanded, on the right)
 - c. EVENTS (collapsed, at the bottom)
- Two vertical tabs on the left:
 - a. Design (activated)
 - b. Data
- Breadcrumbs at the top of the page design (currently it just says *MyPhoneApp*)

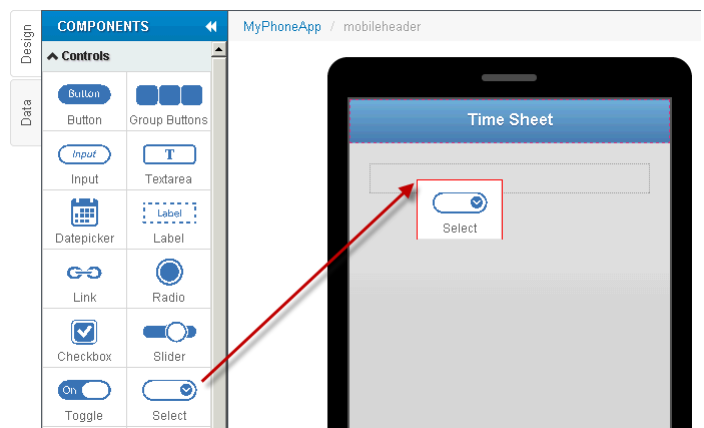
4. Check the **Show Footer** option in the Properties panel.



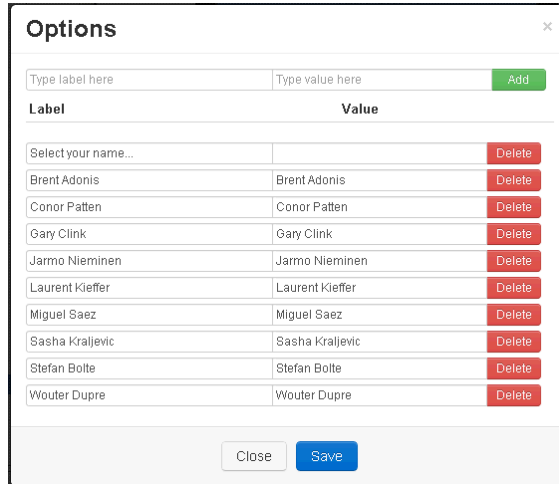
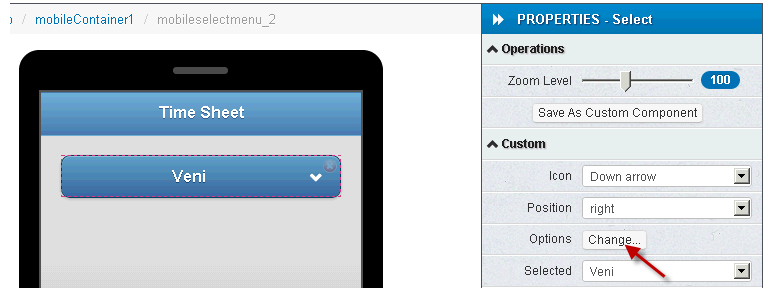
5. Click on the **Header** (MyPhoneApp) and change the Text property to Time Sheet



6. Drag and drop a **Select** component onto the body of the page

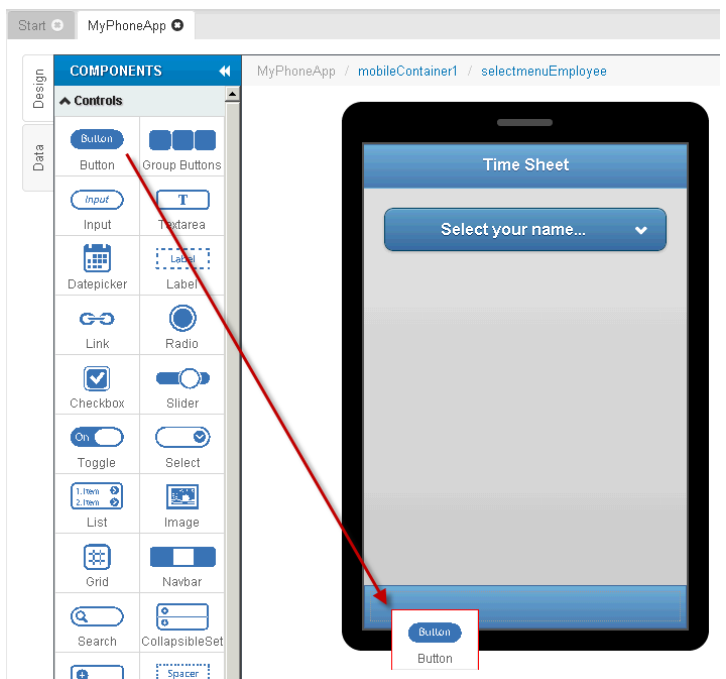


- Click on the **Change...** button next to **Options** in the **Properties** panel and set the options as shown below. You can use your own set of names if you want.



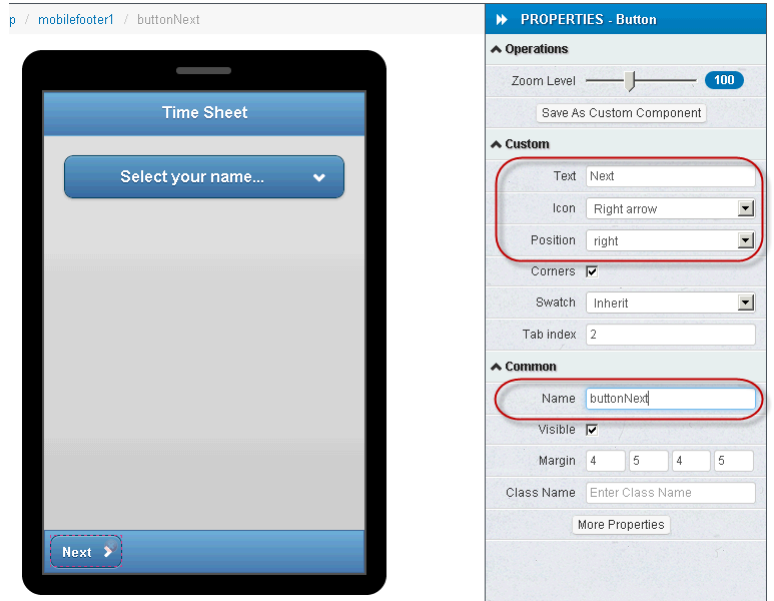
You can use the **Add** and **Delete** buttons to add or remove lines. When you're done setting the options, click the **Save Options** button

- Set the **Name** property of the selectmenu to **selectmenuEmployee**
- Drag and drop a **Button** component into the footer area of the page



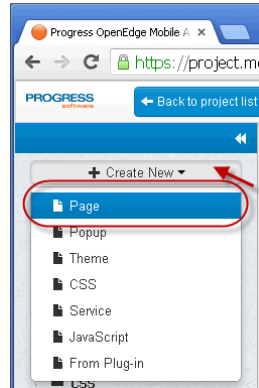
10. Set the properties of the button as below:

- Text:
Next
- Icon:
Right arrow
- Position:
right
- Name:
buttonNext

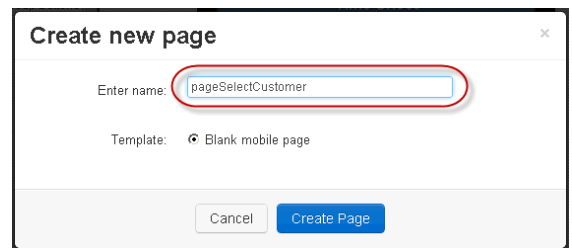


THE CUSTOMER SELECTION PAGE

1. Create a new page by clicking on the **+ Create New** button and then on **Page**,

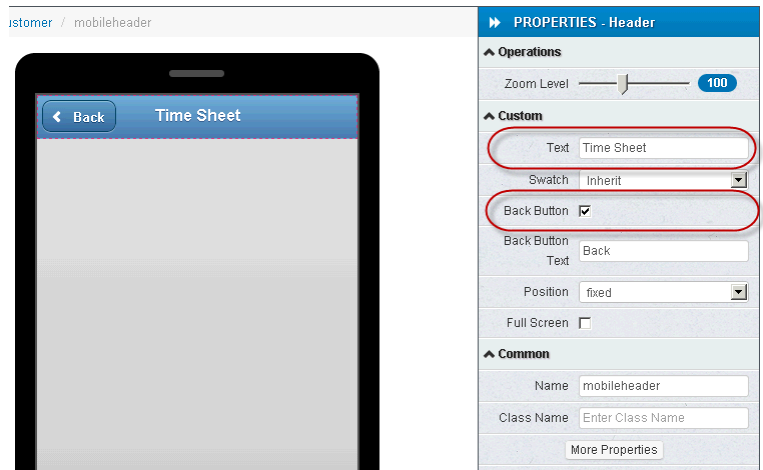


2. Name the page **pageSelectCustomer** and click **Create Page**



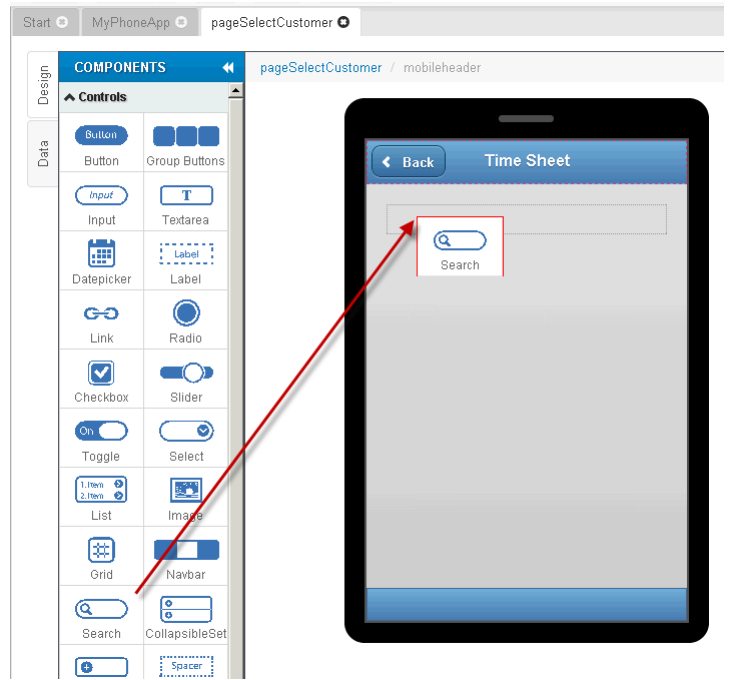
3. Click on the header and set its properties as below:

- Text: **Time Sheet**
- Back Button: **Checked**

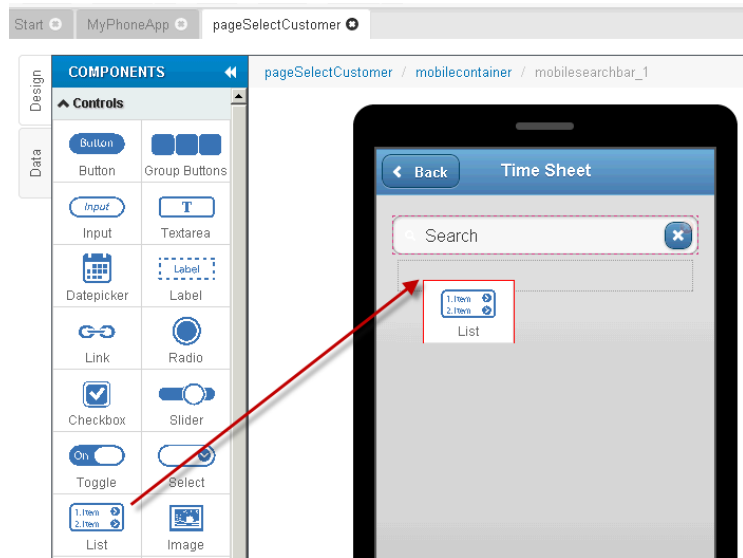


Notice that a back button is automatically added when you check the Back Button checkbox.

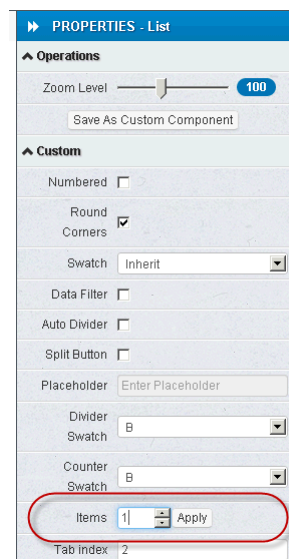
4. Drag and drop a **Search** component to the body of the page



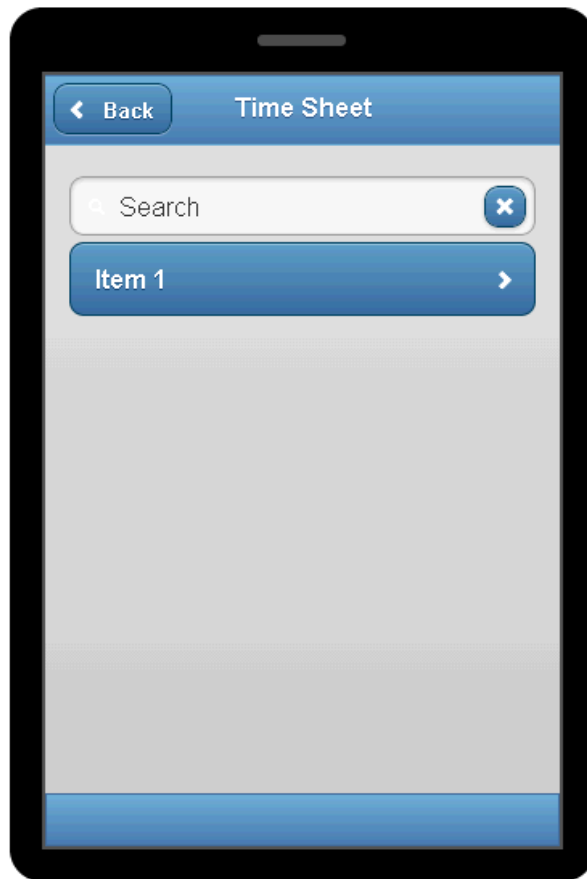
5. Drag and drop a **List** component to the body of the page



6. Change the **Items** property to **1** and click **Apply**



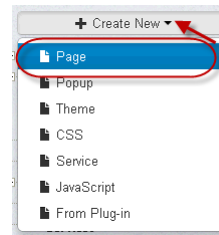
The completed page should look like this.



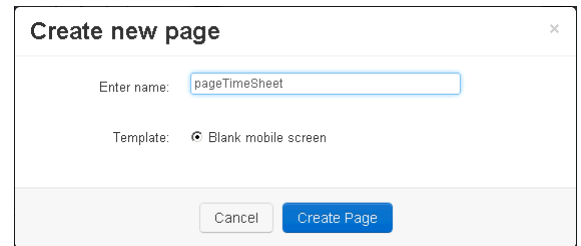
TIME SHEET DETAILS PAGE

Next up is the page where you'll enter the Time Sheet details. To create this page perform the following steps.

1. Create a new page by clicking on the **+ Create New** button and then on **Page**,



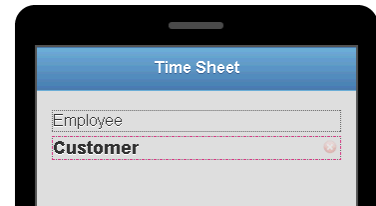
2. Name the page **pageTimeSheet** and click **Create Page**



3. Change the text of the header to **Time Sheet**
4. Add two **Label** components to the page and set their properties:

Label 1:

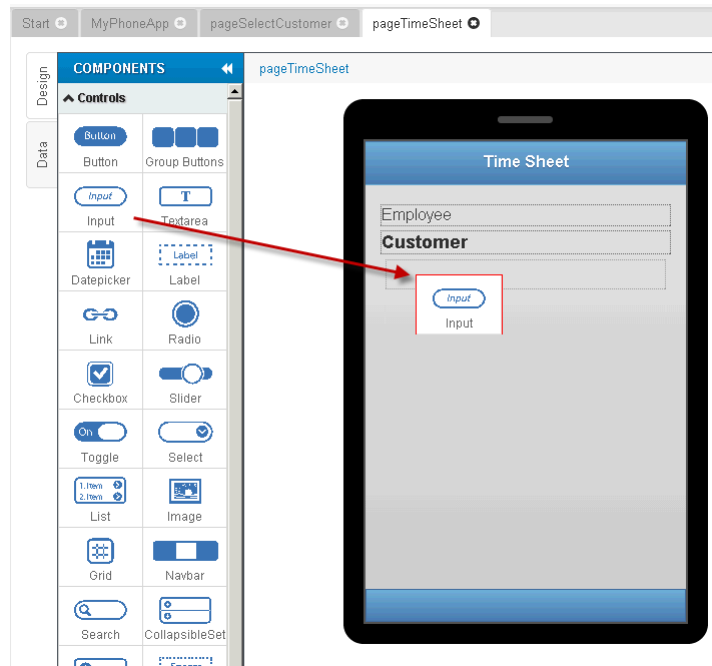
- Text:
Employee
- Name:
labelEmployee
- Font:
Normal



Label 2:

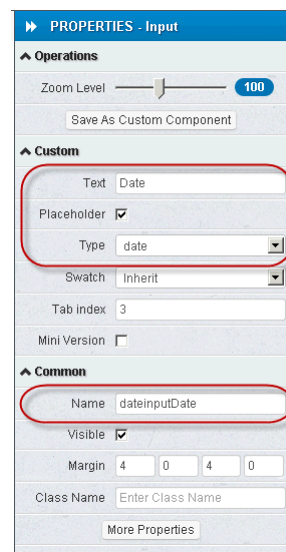
- Text:
Customer
- Name:
labelCustomer
- Size:
18
- Font:
Bold

5. Add an **Input** component to the page



6. Set the properties of the **Input** component

- Text:
Date
- Placeholder:
- Type:
date
- Name:
dateinputDate



7. Drag and drop a **Grid** component on the page

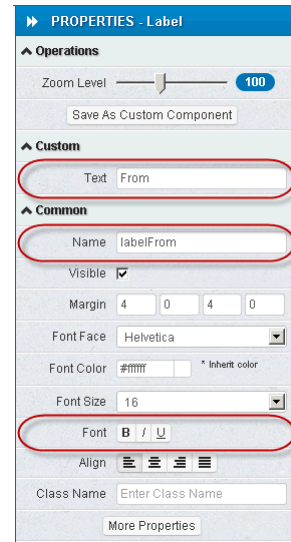


8. Add a **Label** component to the upper left cell of the grid

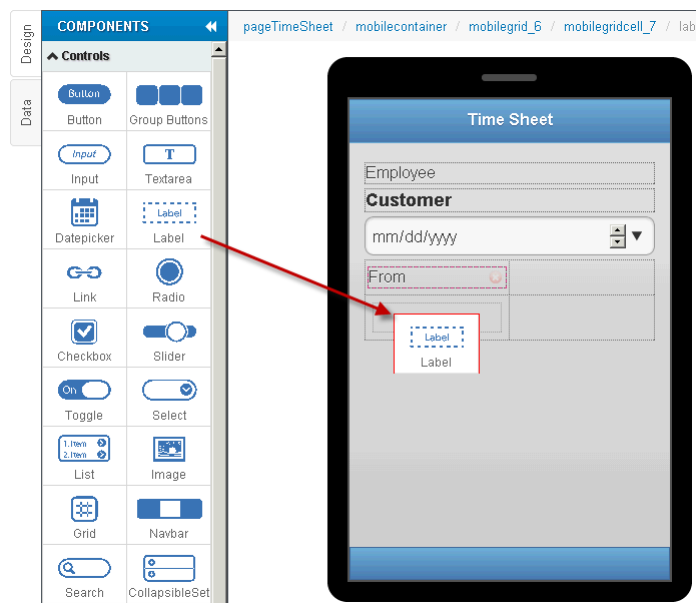


9. Set the properties to:

- Text:
From
- Name:
labelFrom
- Font:
Normal

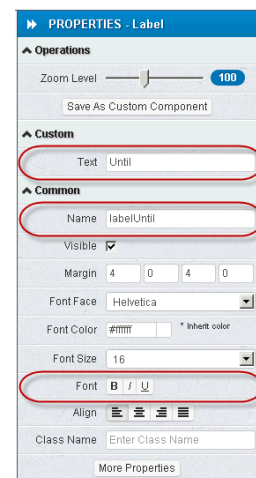


10. Add a **Label** component to the bottom left cell of the grid

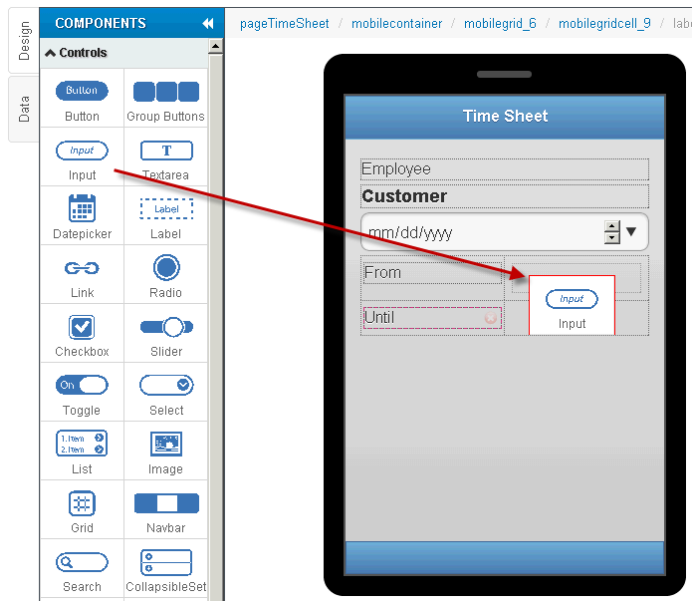


11. Set the properties of the label to:

- Text:
Until
- Name:
labelUntil
- Font:
Normal

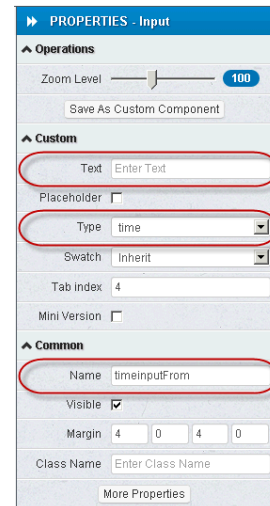


12. Add an **Input** component to the right top cell of the grid

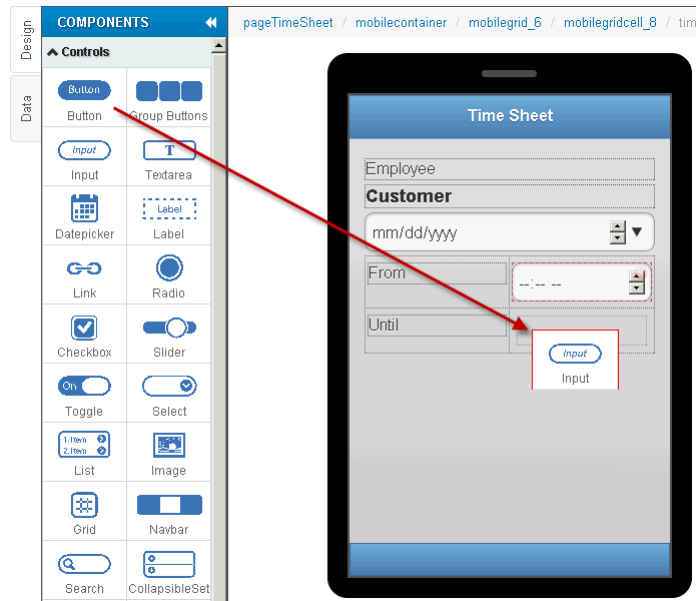


13. Set the properties to:

- Text:
<blank>
- Type:
time
- Name:
timeinputFrom

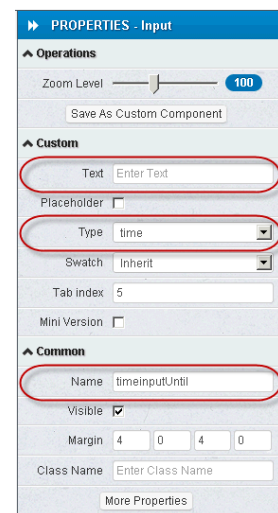


14. Add an **Input** component to the right bottom cell of the grid



15. Set the properties to:

- Text:
<blank>
- Type:
time
- Name:
timeinputUntil



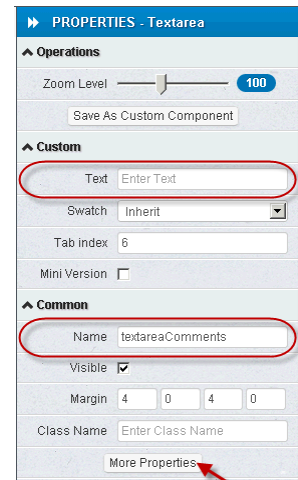
16. Add a **Textarea** component to the screen below the grid



17. Set the properties of the textarea to:

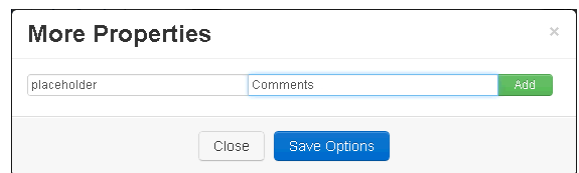
- Text:
<blank>
- Name:
textareaComments

and click on **More Properties**



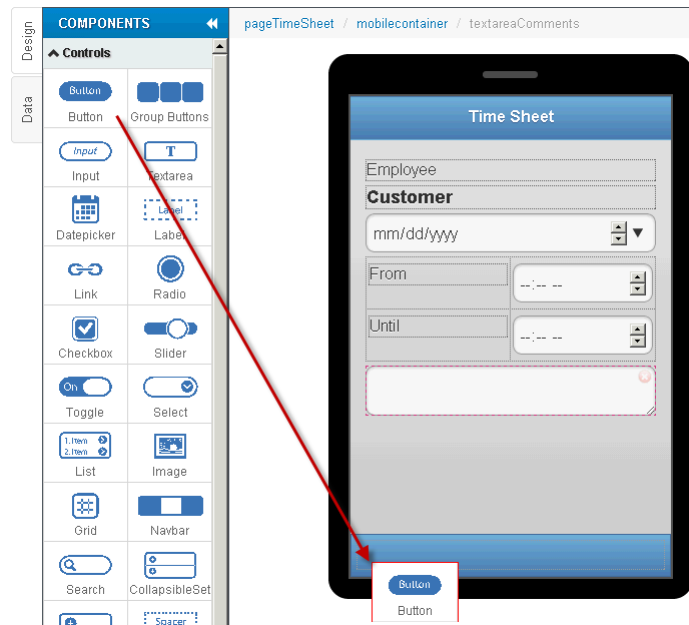
18. Add a property (Careful! Names are case-sensitive!):

- Name:
placeholder
- Value:
Comments



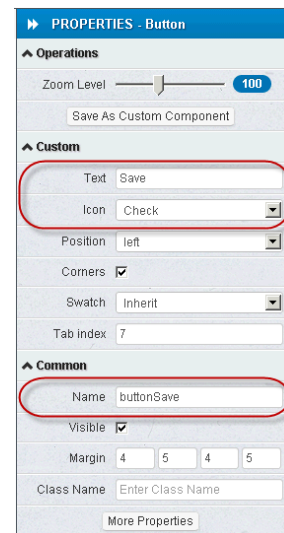
and click **Add** to add the property and click **Save Options**

19. Add a **Button** component to the footer of the page

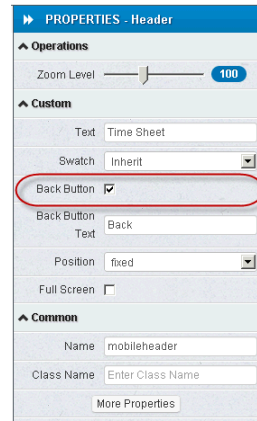


20. Set the properties of the button to

- Text:
Save
- Icon:
Check
- Name:
buttonSave



21. Click on the header of the page and check the **Back Button** property in the properties panel



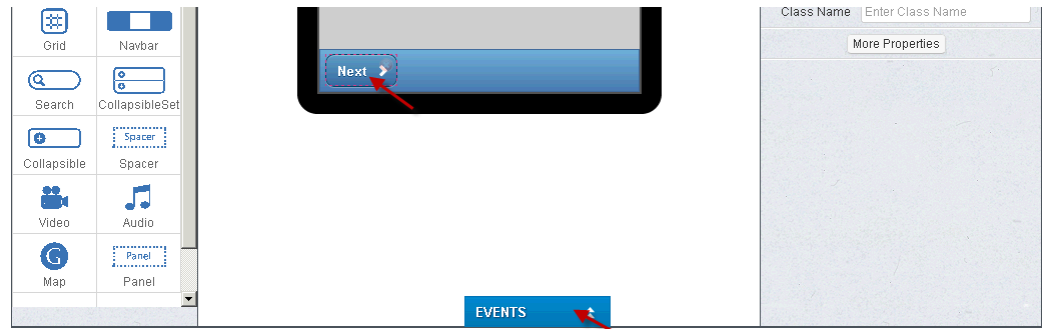
22. When you have completed these steps you should see the following completed page



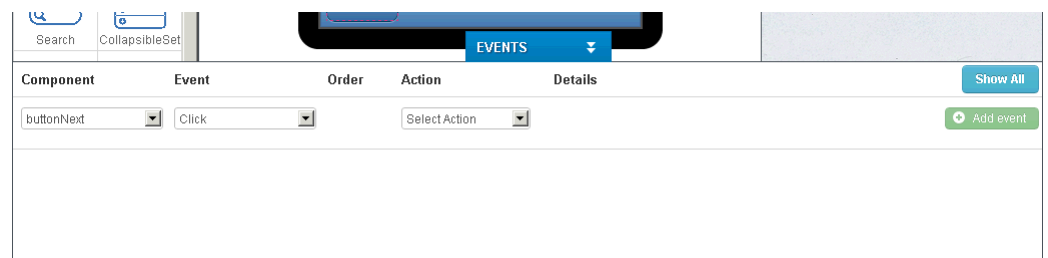
Adding Events

Congratulations, you now have successfully created a number of pages with the OpenEdge Mobile App Builder, well done! Next thing you'll do is add some events to make the pages more interactive. Please follow the steps in this chapter to add the events.

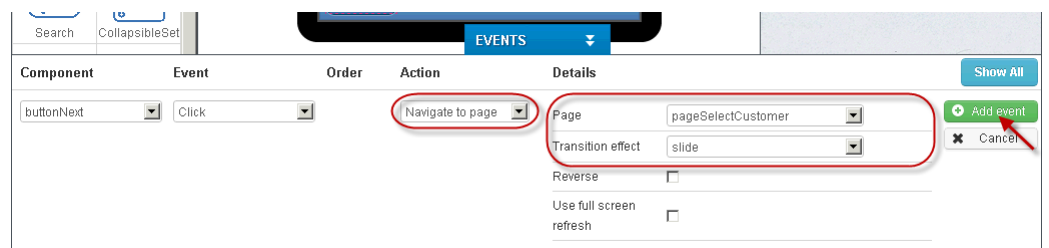
1. Go to the **MyPhoneApp** page, click on the **Next** button in the footer and then on **EVENTS** at the bottom of the window



The EVENTS panel expands



2. Choose **Navigate to page** in the **Action** dropdown, choose **pageSelectCustomer** for the **page**, **slide** for the **Transition effect** and click on **Add event**



3. Select the **selectmenuEmployee** component on the screen and add a **Value change** event

This event should have Set local storage as Action and the other settings must be

- Variable name:
EmployeeName
- Bind to component:
- Target Component:
selectmenuEmployee
- Property name:
Selected

and click **Add event**.

4. On the page **pageSelectCustomer**, click on the **mobilelistitem** component and add a **Click** event with **Set local storage variable** as the **Action**. Make sure the **Variable name** is set to **CustomerName** (case-sensitive), **Bind to component** is checked, the **Target component** is the **mobilelistitem** and the **Property name** is **Text**. Then click **Add event**.

- 5.

the **mobilelist** (**Careful! Use the breadcrumbs if needed!**) on the page and add a **Click** event.

The **Action** should be **Navigate to page**, set the **Page** to **pageTimeSheet** and the **Transition effect** to **slide**. When you're done, click **Add event**

6. Go to the page **pageTimeSheet**.
7. Add an event to the page
 - Component:
pageTimeSheet
 - Event:
Page show
 - Action:
Set property
 - i. ComponentName:
labelEmployee
 - ii. PropertyName:
Text
 - iii. Read from local storage variable:
 - iv. Value
EmployeeName
8. Add a second event to the page
 - Component:
pageTimeSheet
 - Event:
Page show
 - Action:
Set property
 - i. ComponentName:
labelCustomer
 - ii. PropertyName:
Text
 - iii. Read from local storage variable:
 - iv. Value
CustomerName

9. The last event we'll add right now is a **Click** event for the **Save** button on the **pageTimeSheet**. Set the **Action** to **Navigate to page** and set the other properties to:

Component	Event	Order	Action	Details
buttonSave	Click		Navigate to page	Page: MyPhoneApp Transition effect: fade Reverse: <input type="checkbox"/> Use full screen refresh: <input type="checkbox"/>

- Page:
MyPhoneApp
- Transition effect:
fade

Don't forget to click the **Add event** button!

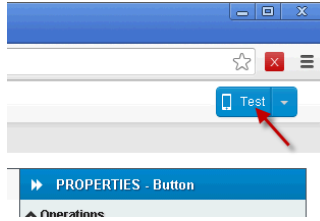
Later in this workshop we'll have to do some more work with events, but for now we are done building the Mobile UI. All that remains is saving our work and do some preliminary tests.

10. Save the Mobile app by clicking on the **Save** button at the top of the screen



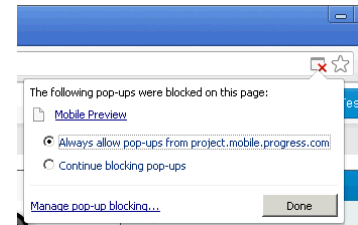
Testing the User Interface

14. To test our application we can click on the Test button on the right.

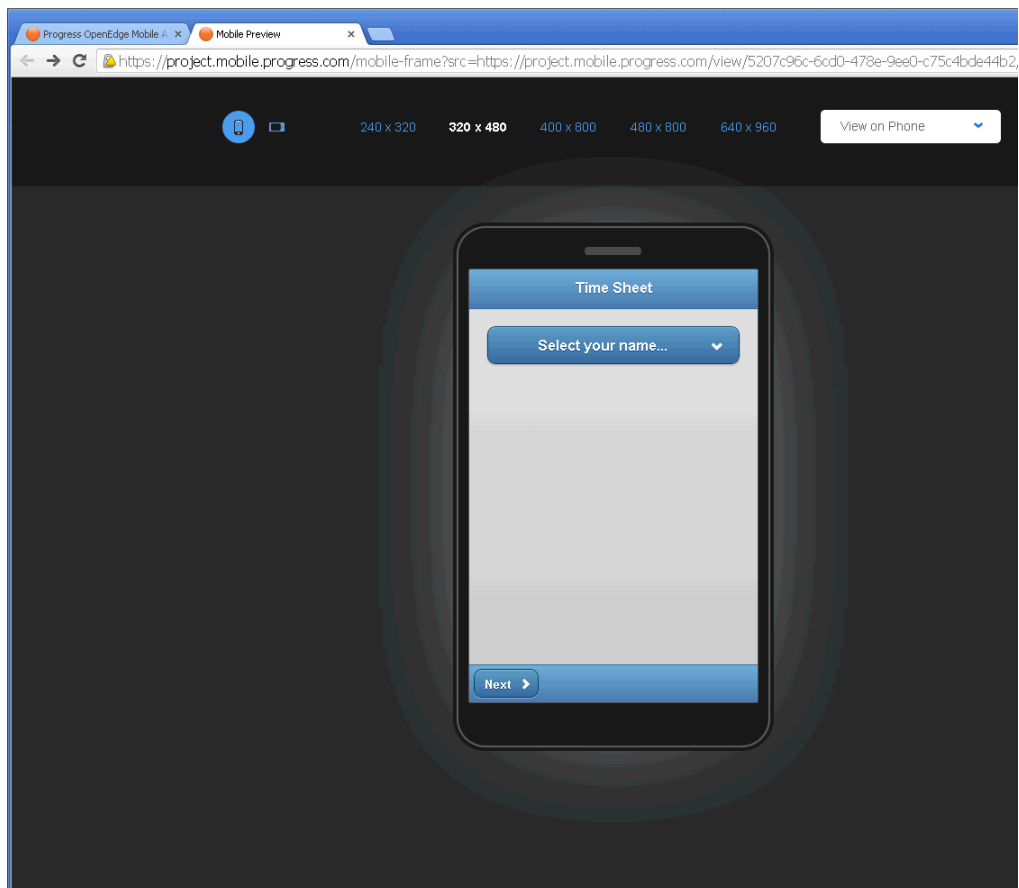


You may see a Pop-up blocked message.

If you do, click on it and set it to **Always allow pop-ups from project.mobile.progress.com**. Click **Done**.

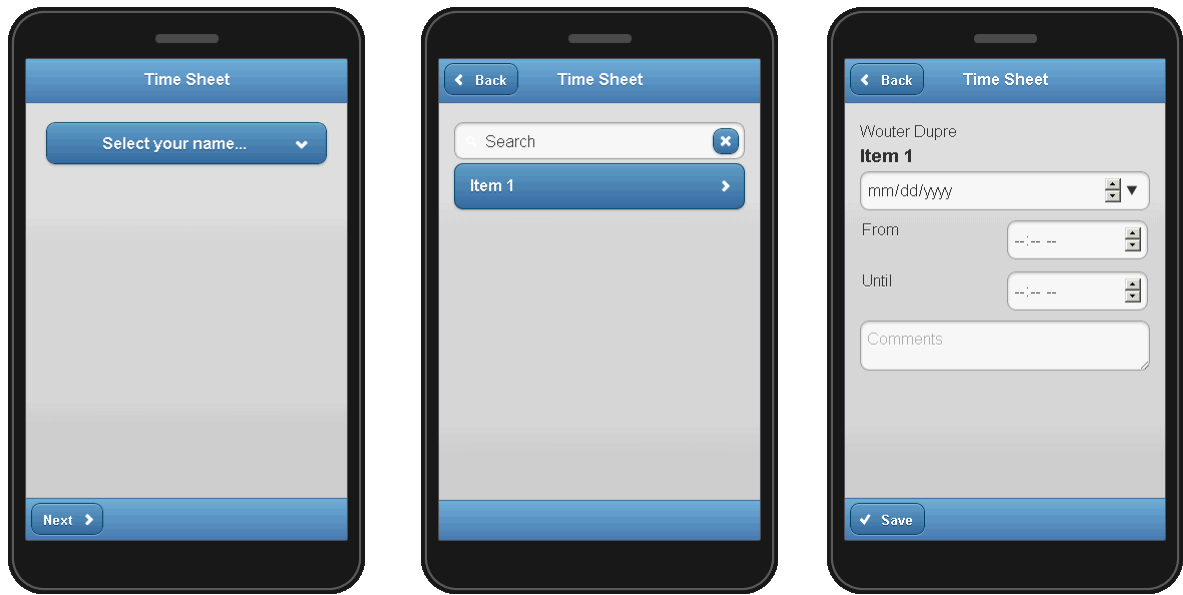


This will open up a new tab in the browser with a device emulator where you can see what the app looks like and how it behaves.



You can also change the size and orientation of the emulator screen by using the option in the toolbar.

The completed screens should look something like this:



You should be able to navigate through the screens using the **Next**, **Back** and **Save** buttons.

At this point you have successfully built a prototype that you can take to your customer and validate the behavior and look and feel of the application without any backend coding!

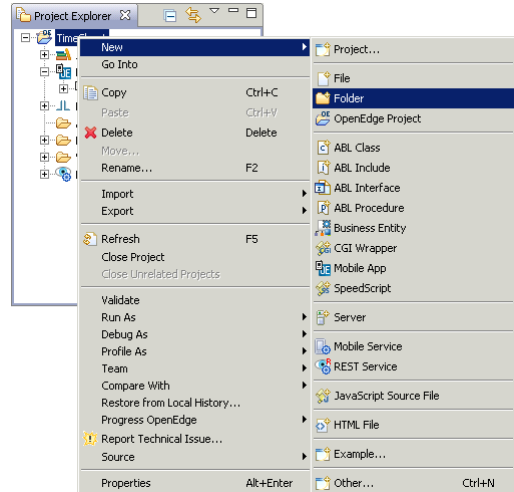
Chapter 3 – Building the Business Logic

What we have built so far is the user interface that will be displayed on the mobile device, but it doesn't do that much yet. We need to prepare some things on the backend to provide interaction with the frontend mobile application.

To start building the backend logic, go back to Progress Developer Studio for OpenEdge and follow the instructions below.

1. Right-click on the project name in the Project Explorer view and choose **New → Folder** from the context menu

The New Folder dialog appears

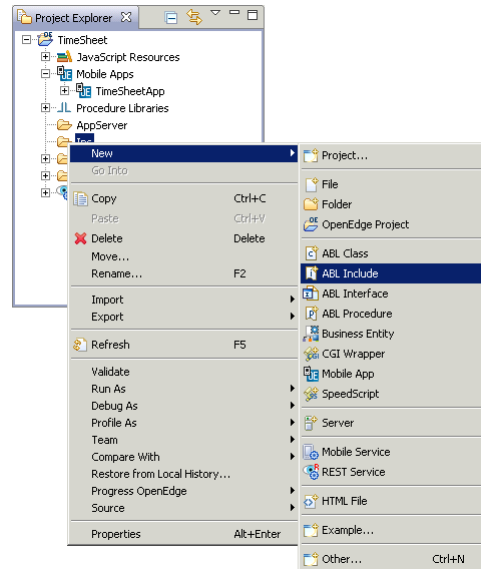


2. Enter **Inc** in the **Folder name** field and click **Finish**

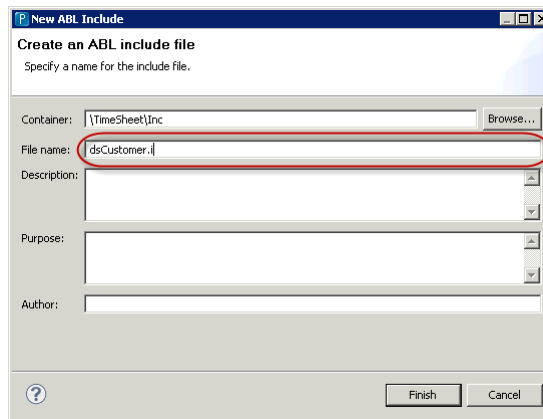
A folder with the name **Inc** is created under the project's root



3. Right-click on the **Inc** folder and choose **New → ABL Include** from the context menu
The New ABL Include dialog appears



4. Enter **dsCustomer.i** as the File name and click **Finish**



5. Enter the temp-table and dataset definition in the Definitions section of the include file. You can copy/paste the [code](#) from the appendix in this document or from the **OE Mobile Workshop.txt** file located in the folder **C:\OEMobile**.

Save the file.

```

dsCustomer.i
Notes
:
-----*/
/* ***** Definitions ***** */
@DEFINE TEMP-TABLE eCustomer NO-UNDO
BEFORE-TABLE eCustomerBefore
FIELD CustNum AS INTEGER
FIELD Name AS CHARACTER
FIELD Address AS CHARACTER
FIELD City AS CHARACTER
FIELD State AS CHARACTER
FIELD Country AS CHARACTER
FIELD Phone AS CHARACTER
FIELD Contact AS CHARACTER
FIELD SalesRep AS CHARACTER
FIELD Comments AS CHARACTER
FIELD CreditLimit AS DECIMAL
FIELD Balance AS DECIMAL
FIELD Terms AS CHARACTER
FIELD Discount AS INTEGER
FIELD PostalCode AS CHARACTER
FIELD Fax AS CHARACTER
FIELD EmailAddress AS CHARACTER
INDEX CustNum IS PRIMARY UNIQUE CustNum
INDEX Comments IS WORD-INDEX Comments
INDEX CountryPost Country PostalCode
INDEX Name Name
INDEX SalesRep SalesRep.

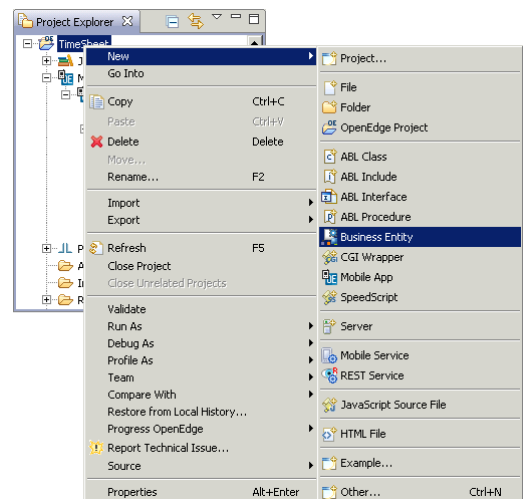
DEFINE DATASET dsCustomer FOR eCustomer.

/* ***** Preprocessor Definitions ***** */

```

6. Right-click on the project name in the **Project Explorer** view and choose **New → Business Entity**

The New Business Entity dialog appears



7. Enter **beCustomer** as the **Business entity name** and click **Next >**

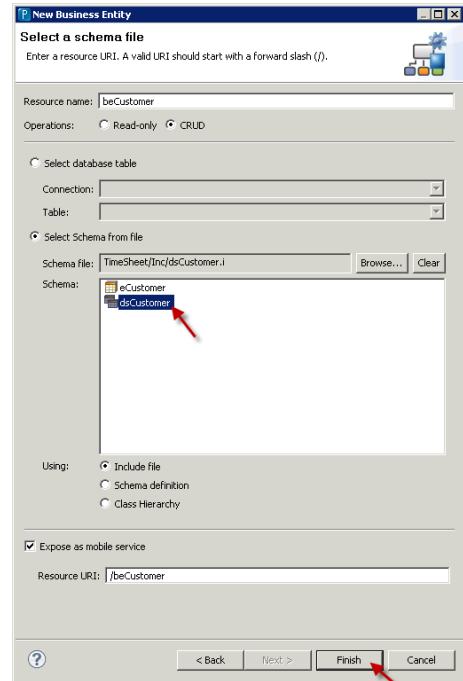
The screenshot shows the 'New Business Entity' dialog box with the title 'Create a Business entity class'. The 'Business entity name' field is highlighted with a red circle and contains the text 'beCustomer'. Other fields include 'Package root' (\\TimeSheet\AppServer), 'Package' (empty), 'Inherits' (empty), and 'Implements' (empty). There are checkboxes for 'Final', 'Abstract', and 'Widget pool'. Below are sections for 'Specify the code elements to generate' and 'Specify the return value for generated methods'. At the bottom are buttons for '< Back', 'Next >', 'Finish', and 'Cancel'.

8. On the **Select a schema file** page, click the **Browse...** button and select the **dsCustomer.i** include file in your **C:\oeworkspace\TimeSheet\Inc** folder.

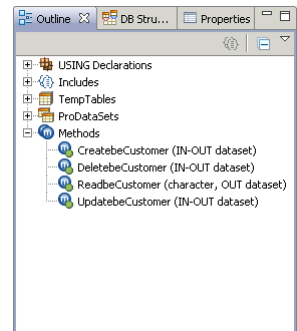
The screenshot shows the 'New Business Entity' dialog box with the title 'Select a schema file'. The 'Resource name' field contains 'beCustomer'. The 'Operations' section has 'Read-only' and 'CRUD' radio buttons, with 'CRUD' selected. There are three options for selecting a schema: 'Select database table', 'Select Schema from file', and 'Using'. The 'Select Schema from file' option is selected. The 'Schema file' field is empty, and a red arrow points to the 'Browse...' button next to it. The 'Expose as mobile service' checkbox is checked. The 'Resource URI' field contains '/beCustomer'. At the bottom are buttons for '?', '< Back', 'Next >', 'Finish', and 'Cancel'.

9. Select **dsCustomer** in the Schema section and click **Finish**

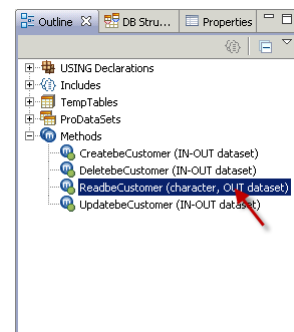
A new business entity is created.



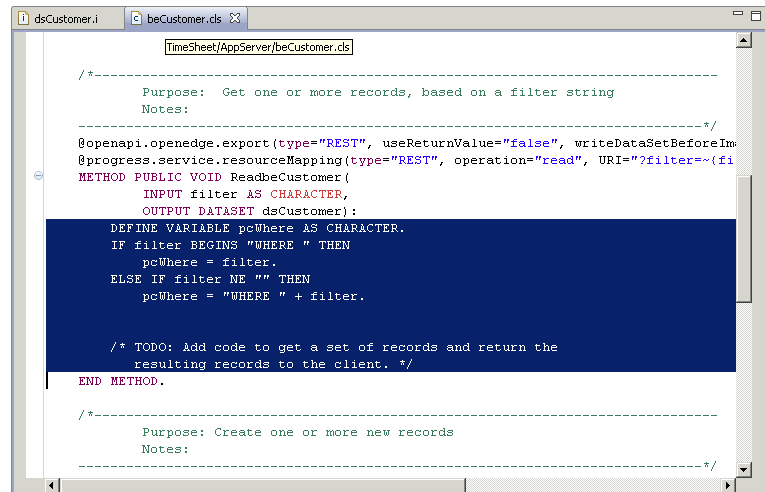
10. Check the **Outline** view and expand the **methods** node. You should see a **create, delete, read and update** method for this business entity



11. Select the **readbeCustomer** method



12. Select and delete the code in body of the method



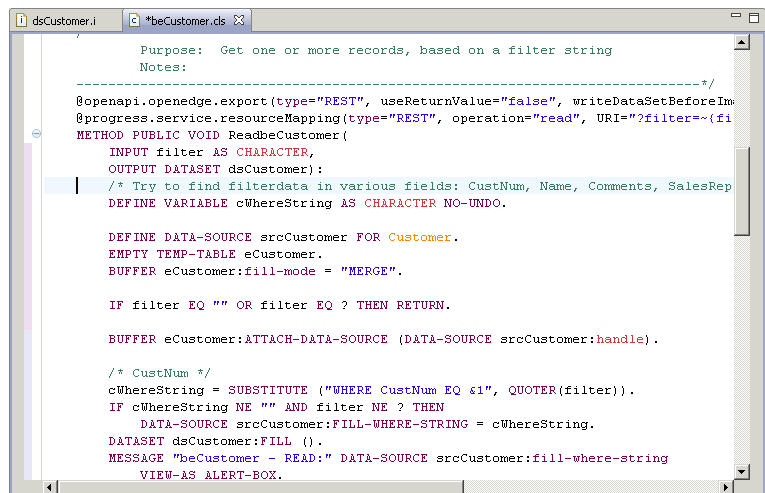
```
TimeSheet/AppServer/beCustomer.cls

/*-----*/
Purpose: Get one or more records, based on a filter string
Notes:
/*-----*/
@openapi.openedge.export(type="REST", useReturnValue="false", writeDataSetBeforeIm
@progress.service.resourceMapping(type="REST", operation="read", URI="?filter=~(fi
METHOD PUBLIC VOID ReadbeCustomer(
    INPUT filter AS CHARACTER,
    OUTPUT DATASET dsCustomer):
    DEFINE VARIABLE pcWhere AS CHARACTER.
    IF filter BEGINS "WHERE " THEN
        pcWhere = filter.
    ELSE IF filter NE "" THEN
        pcWhere = "WHERE " + filter.

    /* TODO: Add code to get a set of records and return the
    resulting records to the client. */
END METHOD.

/*-----*/
Purpose: Create one or more new records
Notes:
/*-----*/
```

13. Copy/paste the [code](#) for the readbeCustomer method from the appendix or the **OE Mobile Workshop.txt** file into the body of the readbeCustomer method.



```
dsCustomer.i *beCustomer.cls

Purpose: Get one or more records, based on a filter string
Notes:
/*-----*/
@openapi.openedge.export(type="REST", useReturnValue="false", writeDataSetBeforeIm
@progress.service.resourceMapping(type="REST", operation="read", URI="?filter=~(fi
METHOD PUBLIC VOID ReadbeCustomer(
    INPUT filter AS CHARACTER,
    OUTPUT DATASET dsCustomer):
    /* Try to find filterdata in various fields: CustNum, Name, Comments, SalesRep
    DEFINE VARIABLE cWhereString AS CHARACTER NO-UNDO.

    DEFINE DATA-SOURCE srcCustomer FOR Customer.
    EMPTY TEMP-TABLE eCustomer.
    BUFFER eCustomer:fill-mode = "MERGE".

    IF filter EQ "" OR filter EQ ? THEN RETURN.

    BUFFER eCustomer:ATTACH-DATA-SOURCE (DATA-SOURCE srcCustomer:handle).

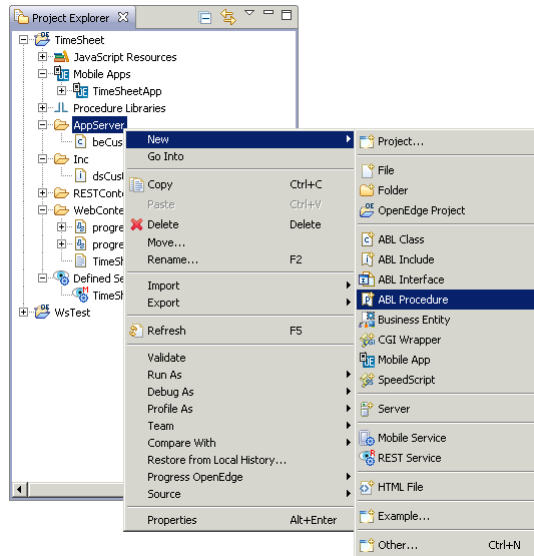
    /* CustNum */
    cWhereString = SUBSTITUTE ("WHERE CustNum EQ &1", QUOTER(filter)).
    IF cWhereString NE "" AND filter NE ? THEN
        DATA-SOURCE srcCustomer:FILL-WHERE-STRING = cWhereString.
    DATASET dsCustomer:FILL ().
    MESSAGE "beCustomer - READ:" DATA-SOURCE srcCustomer:fill-where-string
    VIEW-AS ALERT-BOX.
```

To indent the code properly you can deselect any text and press **CTRL-I**.

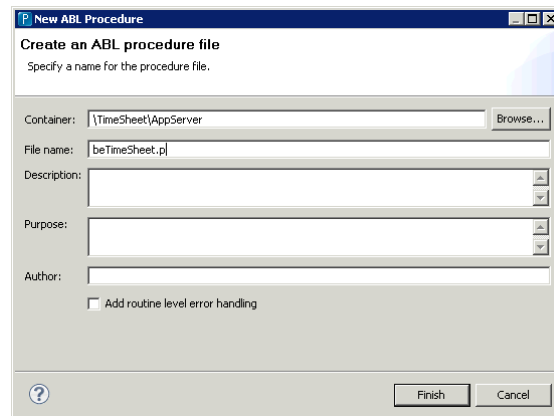
14. Check the syntax (**CTRL-SHIFT-C**) and save (**CTRL-S**) the code

This is all the code we need for the beCustomer business entity. Next we'll create a persistent procedure to allow us to save the Time Sheet details.

- Right-click on the **AppServer** node in the **Project Explorer** view. Choose **New -> ABL Procedure**



- Enter the **beTimeSheet.p** as the file name of the procedure and click **Finish**



- Copy/paste the procedure **SaveTimeSheet** from the appendix or from the file **OE Mobile Workshop.txt** located on your desktop.

```

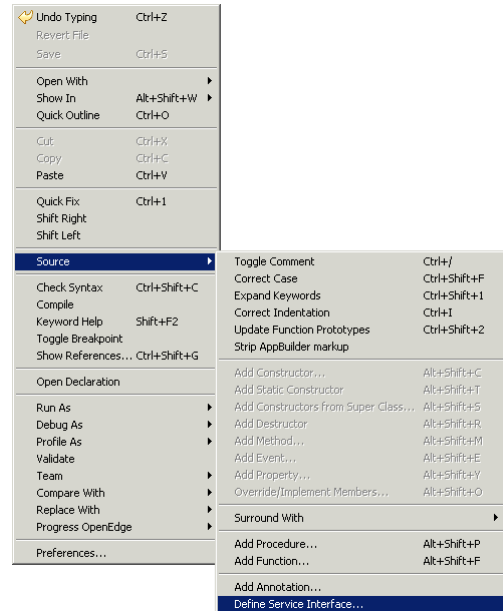
/* ***** Main Block ***** */

PROCEDURE SaveTimeSheet:
    DEFINE INPUT PARAMETER ipchEmployee AS CHARACTER NO-UNDO.
    DEFINE INPUT PARAMETER ipchCustomer AS CHARACTER NO-UNDO.
    DEFINE INPUT PARAMETER ipchDate AS CHARACTER NO-UNDO.
    DEFINE INPUT PARAMETER ipchFrom AS CHARACTER NO-UNDO.
    DEFINE INPUT PARAMETER ipchUntil AS CHARACTER NO-UNDO.
    DEFINE INPUT PARAMETER ipchComments AS CHARACTER NO-UNDO.

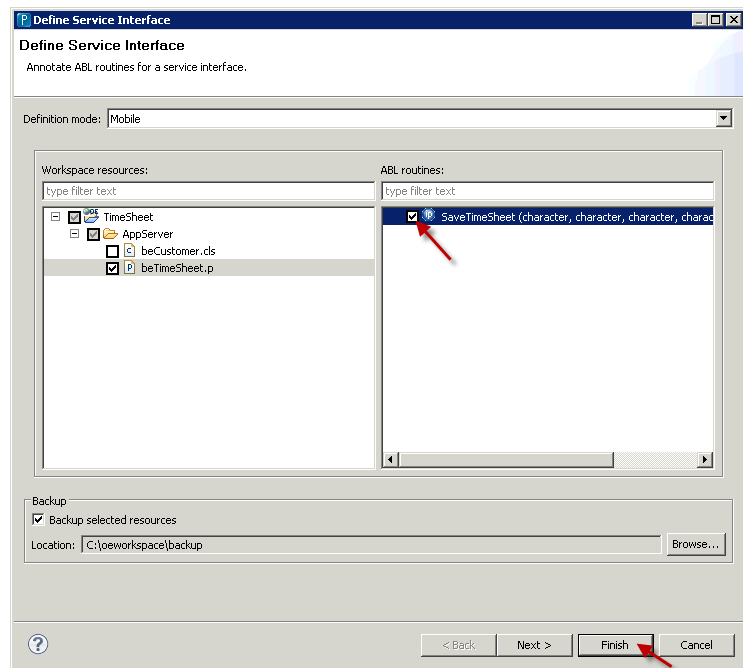
    MESSAGE
        "Employee:" ipchEmployee SKIP
        "Customer:" ipchCustomer SKIP
        "Date:" ipchDate SKIP
        "From:" ipchFrom SKIP
        "Until:" ipchUntil SKIP
        "Comments:" ipchComments
        VIEW-AS ALERT-BOX.
END PROCEDURE.

```

18. Right-click in the code and choose **Source** → **Define Service Interface...** from the context menu. Alternatively you can find the same option under **Progress OpenEdge**.

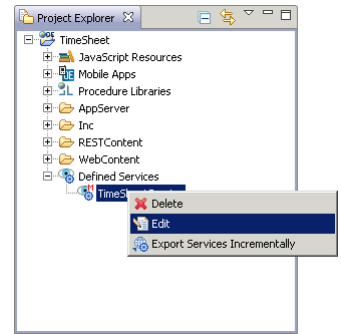


19. Check the **checkbox** at the beginning of the line **SaveTimeSheet** under **ABL routines** and click **Finish**

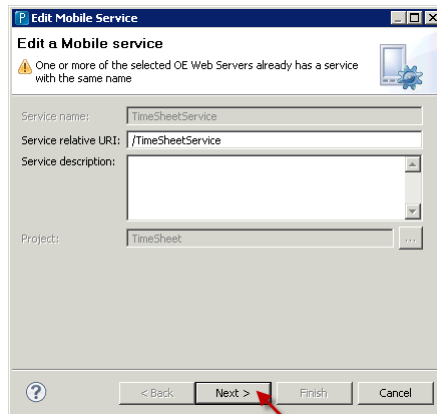


20. Save your code by pressing **<CTRL-S>**.

21. Now we need to add the business entities to the mobile service. To do this you expand the **Defined Services** node in the **Project Explorer** view, right-click on the **TimeSheetService** node and choose **Edit**

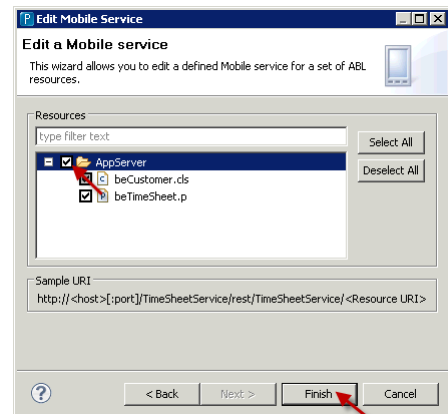


The Edit Mobile Service dialog appears.



22. Click **Next >**
23. **Check** the **AppServer** node and click **Finish**

At this moment a new file (**TimeSheetService.json**) is being created with the description of the service under the **WebContent** folder. We'll need this file in our Mobile App Builder later.



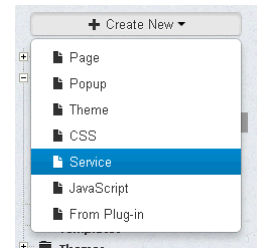
Annotations have been added at the top of the procedure and just before the SaveTimeSheet procedure.

Chapter 4 – Binding the Mobile User Interface to the Business Logic

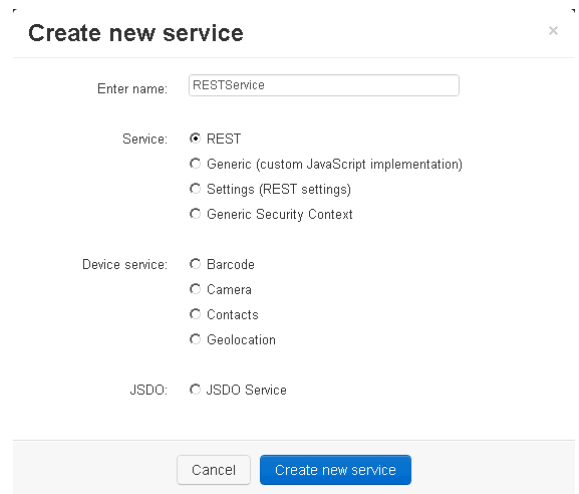
So far so good! We have built a mobile UI and some backend logic. All we need to do now is bind the two together. Follow these steps to complete this task.

If you have closed the OpenEdge Mobile App Builder, you can double-click on the TimeSheetApp node under Mobile Apps in the Project Explorer view of PDSOE.

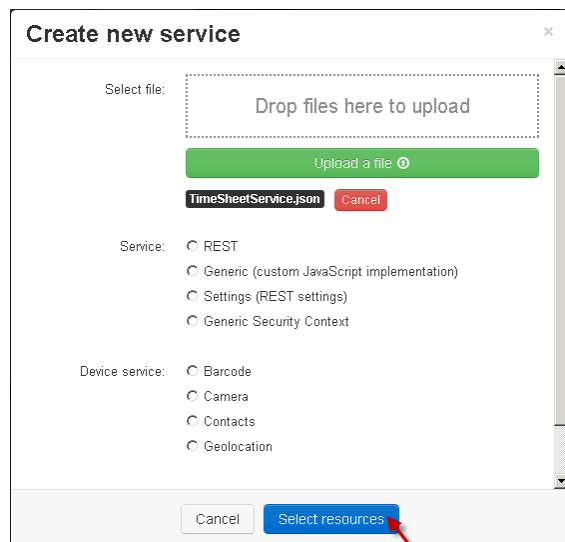
1. In the OpenEdge Mobile App Builder, create a new service by clicking on **+ Create New** and then **Service**



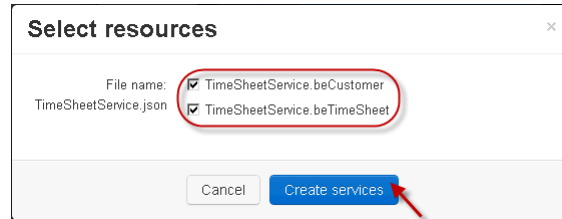
2. This will open the **Create new service** dialog



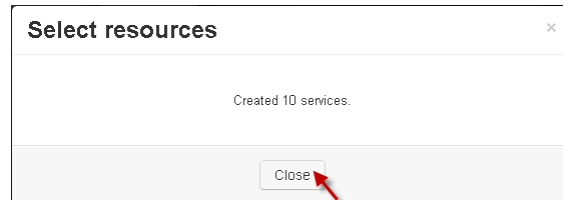
3. Select **JSDO Service**, click on **Upload a file** and browse for **TimeSheetService.json**
This file is located PDSOE in your workspace -> project -> WebContent folder
and click on **Select resources**



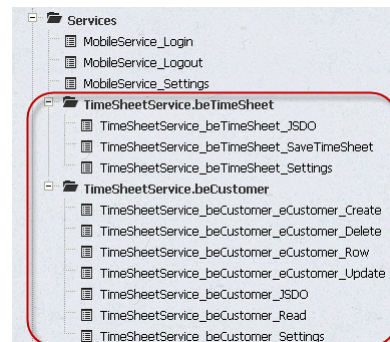
4. Check the checkboxes **TimeSheetService.beCustomer** and **TimeSheetService.beTimeSheet** and click **Create services**



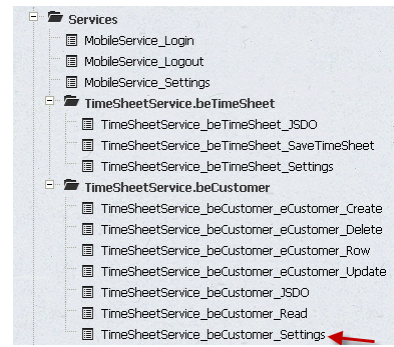
5. Click **Close** when you see the **Created 10 services** message



6. In the Project overview panel, you should be able to see the 10 newly created services



7. Click on the node **TimeSheetService_beCustomer_Settings** node

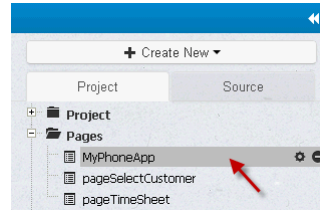


8. Enter the following values:

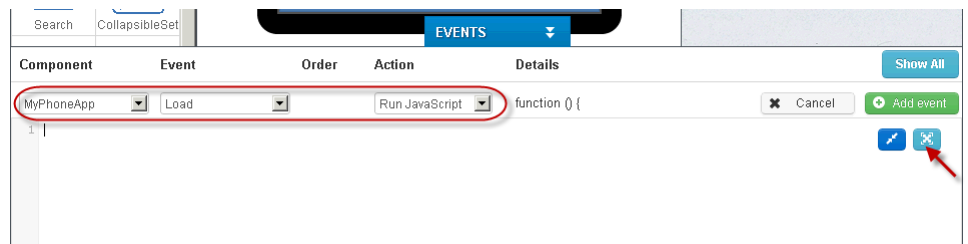
Name	Default value
catalogURI	http://<yourpublicmachinename>.amazonaws.cc
serviceURI	http://<yourpublicmachinename>.amazonaws.cc
resourceName	beCustomer

- Before the current value of catalogURI:
http://<yourpublicmachinename>:8980/TimeSheetService
so the full value becomes:
http://<yourpublicmachinename>:8980/TimeSheetService/static/mobile/TimeSheetService.json
Replace <yourpublicmachinename> with the real public machine name.
This is the name of your Arcade Instance that was given to you at the beginning of the workshop and starts with **ec2-**.
- serviceURI:
http://<yourpublicmachinename>:8980/TimeSheetService
Again, replace <yourpublicmachinename> with the real public machine name.


- Now we have to load the JSDO service to the catalog. We'll do this on the home page of the application. Open the home page by clicking on the **MyPhoneApp** item in the project view under **Pages**



- Expand the **Events** panel, if needed and add the following event



- Component:
MyPhoneApp
- Event:
Load
- Action:
Run JavaScript


- Click on the full screen button  to expand the text area
- Copy/paste the [code](#) for adding the catalog from the appendix in this document or from the **OE Mobile Workshop.txt** file.

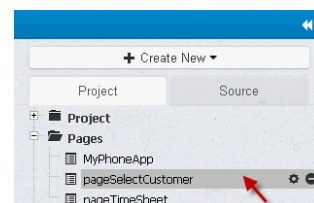
```

1 /* if the first time, or not logged in, try to log in */
2
3 if ($.ProgressSession == undefined || $.ProgressSession.loginResult != progress.data.Session.LOGIN_SUCCESS) {
4
5     /* TimeSheetService_beCustomer_Settings is the Settings Service created when
6     the catalog was loaded. */
7
8     var settings = TimeSheetService_beCustomer_Settings;
9
10    /* Putting the session object into Mobile App Builder's namespace
11    so that it can be accessed later, if needed */
12
13    var pdsession = $.ProgressSession;
14
15    if (pdsession == undefined)
16        pdsession = $.ProgressSession = new progress.data.Session();
17
18    if (settings.serviceURI == undefined || settings.serviceURI == '')
19        console.log("serviceURI was not specified." +
20        " catalogURI: " + settings.catalogURI +
21        " resourceName: " + settings.resourceName);
22
23    var loginResult = pdsession.login (settings.serviceURI, "", "");
24    if (loginResult != progress.data.Session.LOGIN_SUCCESS) {
25        var msg = "ERROR: Login failed with code: " + loginResult;
26        console.log(msg);
27        alert (msg);
28        return;
29    }
30    // load catalog
31    pdsession.addCatalog(settings.catalogURI);
32 }

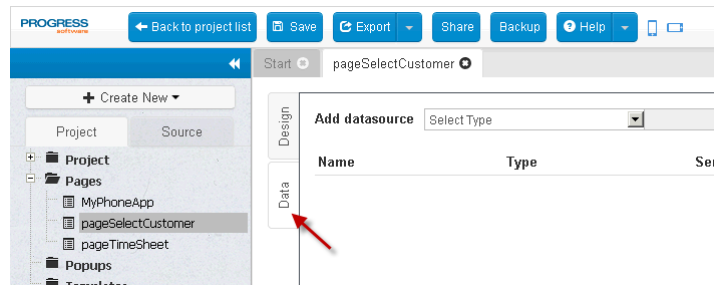
```

If you have used a different name for your project and/or business entity, you'll have to change your code accordingly.

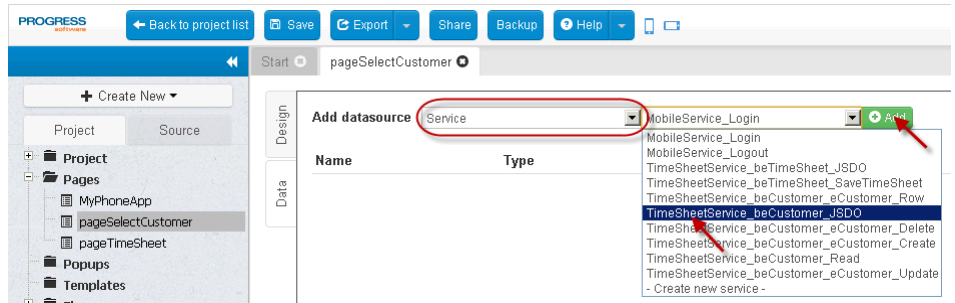
- Click the exit full screen button  and click **Add event**. In case you don't see the Add Event button, you need to collapse the components panel.
- Open the pageSelectCustomer page



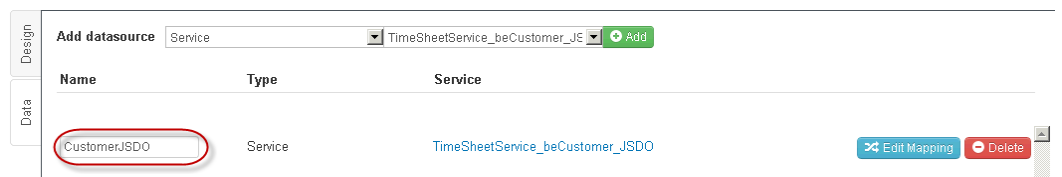
15. Click on the **Data** tab, just below the **Design** tab



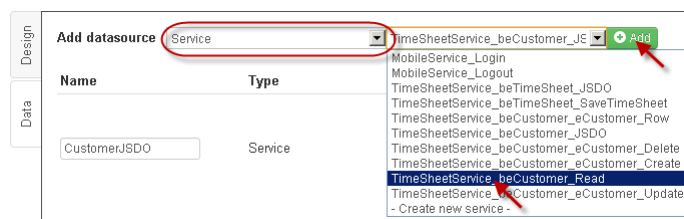
16. Select **Service** in the Add datasource dropdown and then **TimeSheetService_beCustomer_JSDO** and click **Add**



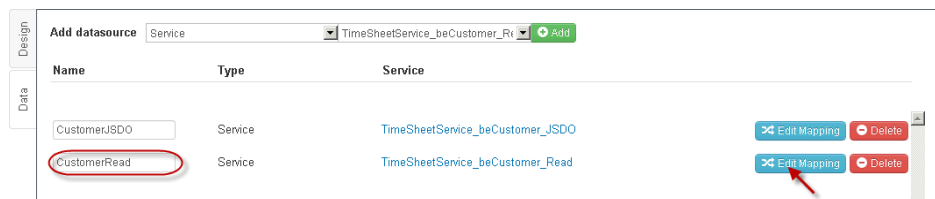
17. Rename the service to **CustomerJSDO**



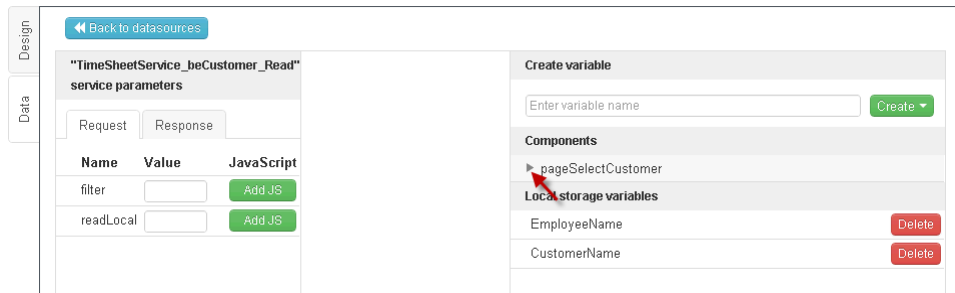
18. Add another datasource of type **Service**: **TimeSheetService_beCustomer_Read**



19. Rename the service to **CustomerRead** and click **Edit Mapping**



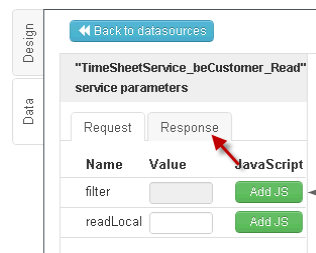
20. In this screen you'll connect the UI components to the datasource.
You can click on the **triangle** left of the components to expand or collapse nodes.



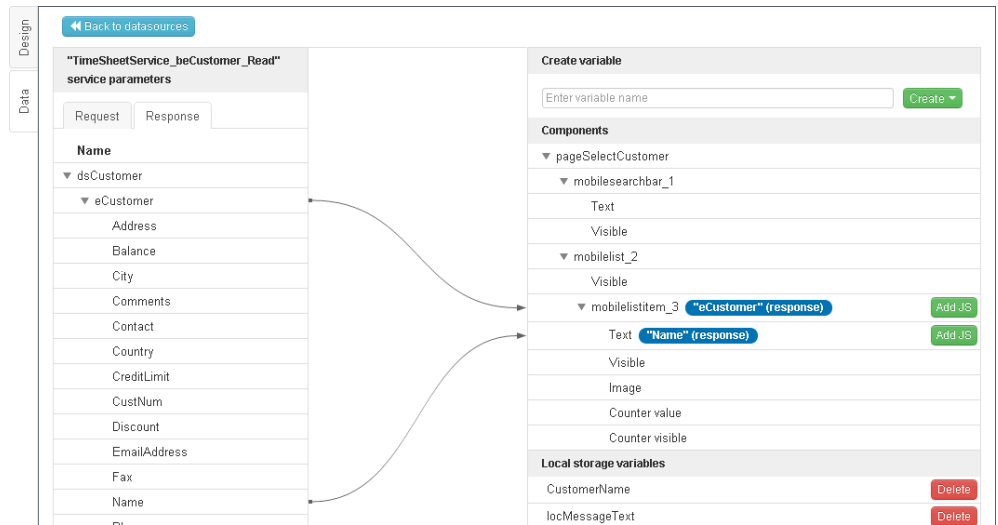
21. Expand the **mobilesearchbar1_2** and Drag and drop the **Text** node to the **filter** node in the **Request** tab



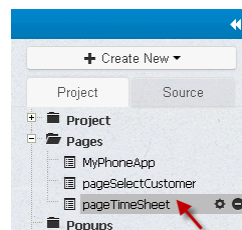
22. Click on the **Response** tab



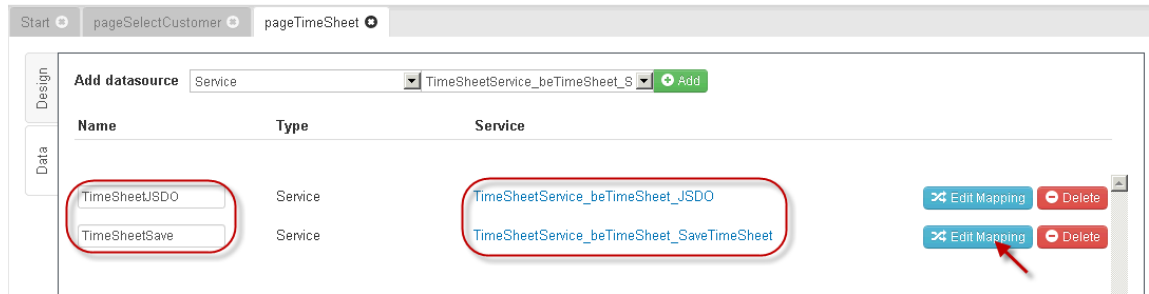
23. Drag and drop the **eCustomer** node from the Response to the **mobilelistitem** node
 Drag and drop the **Name** node from the Response to the **Text** Node of **mobilelistitem**



24. Click **Back to datasources**
25. Click on the **Design** tab and select **pageSelectCustomer** from the breadcrumbs
26. Expand the **Events** panel if needed and add a **Load** event for the page. Choose **Invoke service** from the **Action** dropdown and select **CustomerJSDO** as **Datasource**. Click **Add event**.
27. Select the **mobilesearchbar** on the page
28. Add a **Value change** event to the **mobilesearchbar**, select **Invoke service** as the **Action**, **CustomerRead** as the **Datasource** and Click **Add event**
29. Open the **pageTimeSheet** page



30. Add **datasources** to **pageTimeSheet**

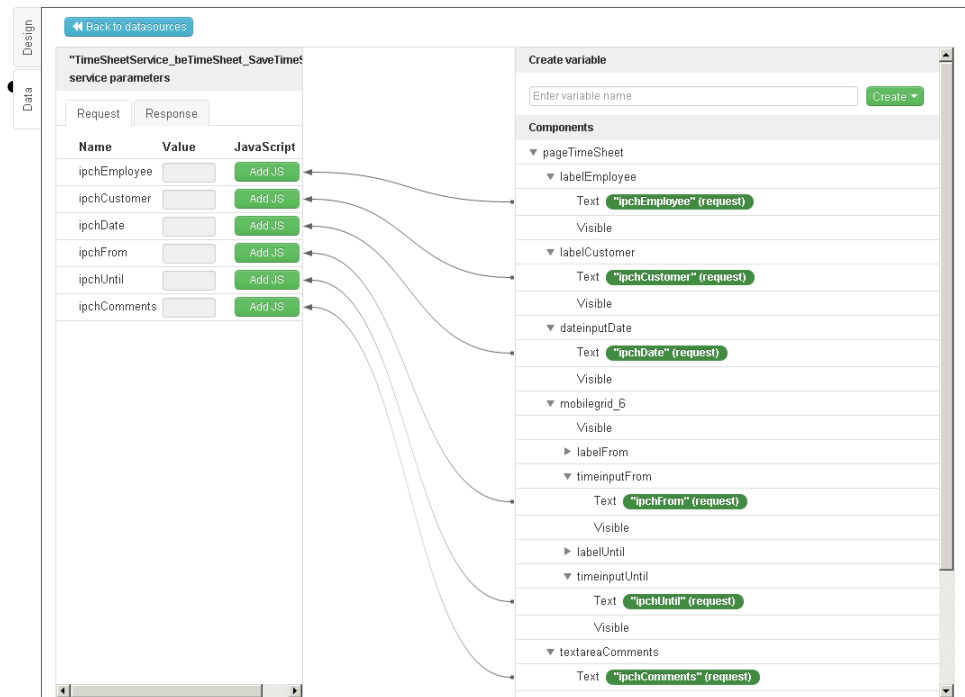


Add datasource **TimeSheetService_beTimeSheet_JSDO** to **pageTimeSheet**, name: **TimeSheetJSDO**

Add datasource **TimeSheetService_beTimeSheet_SaveTimeSheet** to **pageTimeSheet**, name: **TimeSheetSave**

31. Click **Edit Mapping** for the **TimeSheetSave** service.

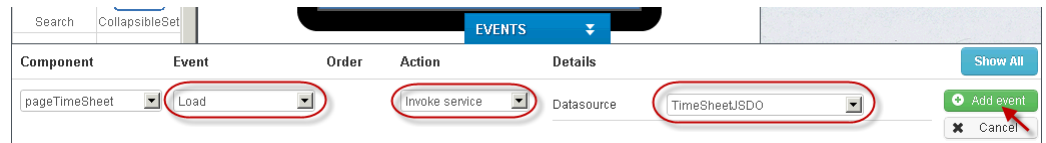
32. On the **Request** tab map the components as shown here:



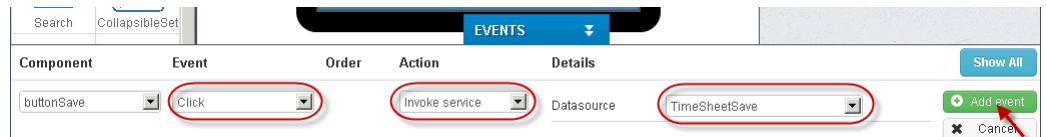
b
elEmployee.Text

- ipchCustomer ← labelCustomer.Text
- ipchDate ← dateinputDate.Text
- ipchFrom ← timeinputFrom.Text
- ipchUntil ← timeinputUntil.Text
- ipchComments ← textareaComments.Text

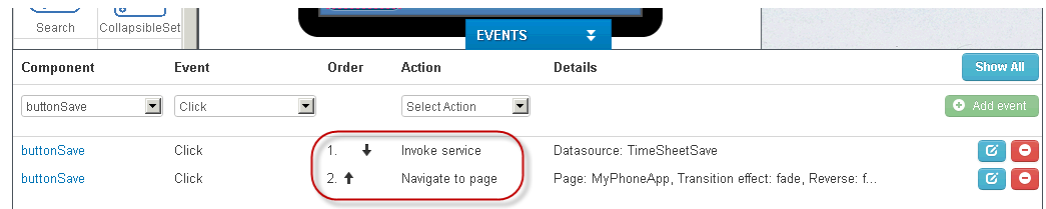
33. Click the **Design** tab, expand the **Events** panel and add a **Load** event for **pageTimeSheet**. Set **Invoke Service** as the **Action** and **TimeSheetJSDO** as the **Datasource**. Click **Add Event**.



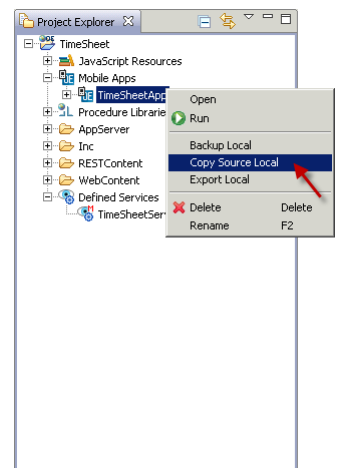
34. Add a **Click** event for **buttonSave**, set **Invoke Service** as the **Action** and **TimeSheetSave** as the **Datasource**. Click **Add event**.



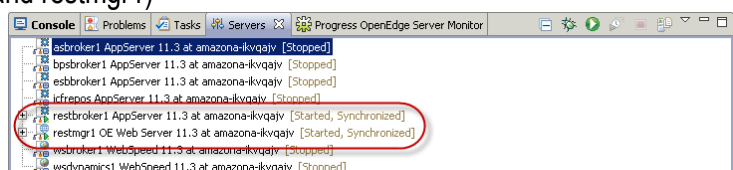
35. Reorder the events for **buttonSave** so that the **Invoke Service** event takes place before the **NavigateToPage** event using the **up** and/or **down** arrows in front of the event actions.



36. Save the application
37. Go back to Progress Developer Studio
38. In the **Servers** view, right-click on **restbroker1 AppServer**, and choose **Start**.
39. Right-click on the **TimeSheetApp** node in the Project Explorer view and choose **Copy Source Local**

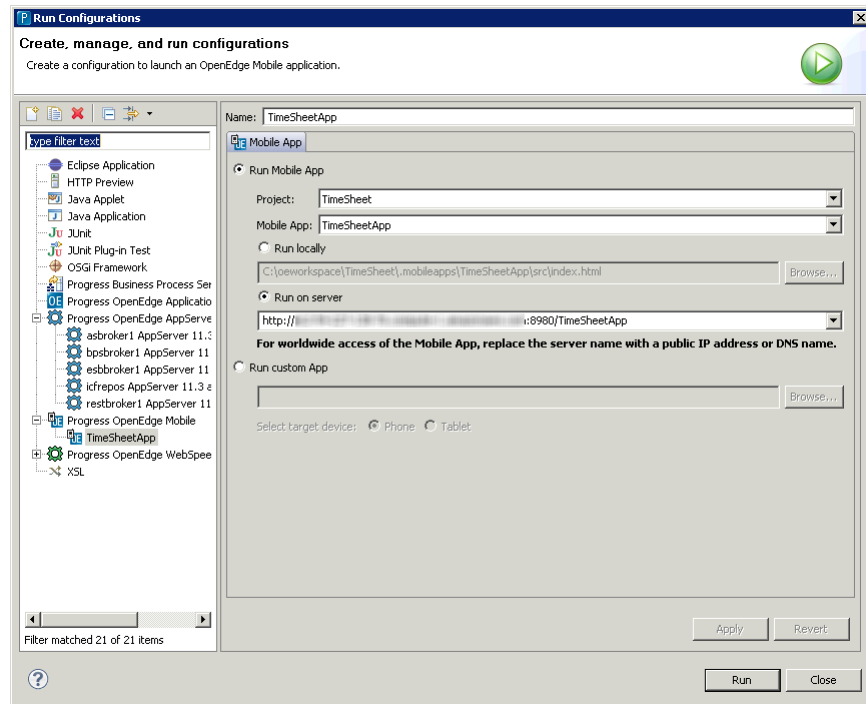


Wait until the code is published (synchronized) to the servers (restbroker1 and restmgr1)



40. Go to **Run** → **Run Configurations...**

41. Double-click **Progress OpenEdge Mobile** in the project type on the left.



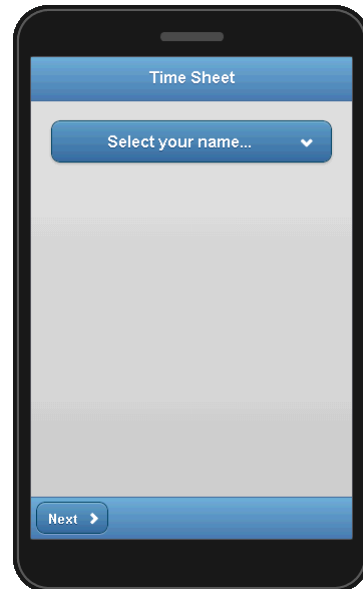
42. Enter **TimeSheetApp** as the **name** for the configuration

43. Choose the project **TimeSheet** from the **Project** drop down and the mobile app **TimeSheetApp** from the **Mobile App** drop down.

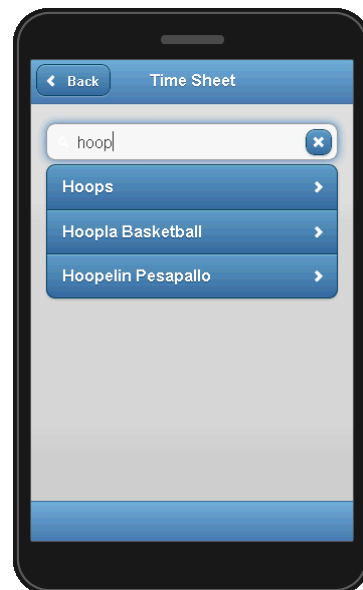
44. Choose **Run on server**, which will populate a correct URL by default. Potentially you can also enter a public IP or hostname instead of the machine name. That would make the application accessible from you smartphone or tablet (or any other computer). In the screenshot above the machine name (default behavior) was replaced by the public domain name.

45. click **Apply** and **Run**.

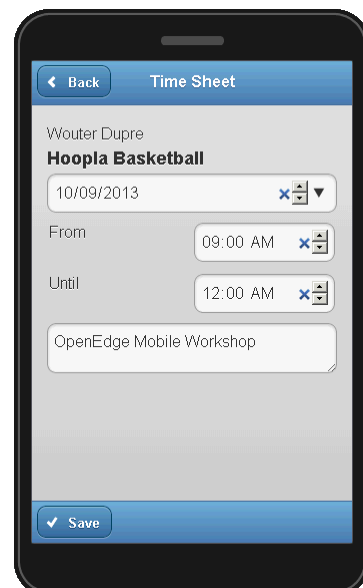
46. Choose an Employee from the select menu and click **Next**.



47. Enter some search criteria to lookup a customer (can be based on name, number or salesrep) and click on one to select it.



48. Choose a **Date**, **From** and **Until** times, enter some **comments** and click **Save**.



49. Check the AppServer log file (*C:\OpenEdge\WRK\restbroker1.server.log*) for messages that indicate you have successfully saved the time sheet data.

If you can see the data that you entered through your mobile user interface then congratulations, you have successfully configured your mobile application! If you were writing an application for real, at this point you would run your business logic to write the data back to your database using standard ABL code.

Chapter 5 – Packaging and Deploying your application for specific devices

OpenEdge Mobile uses some clever technology to allow the applications that you've built to be deployed either as a WebApp (i.e. being run using standard browser technology) or packaged as an application for a specific platform or device.

In this workshop we have been developing and deploying the TimeSheet application through our browser. What is needed to deploy the application as a real app to e.g. the Apple AppStore, is beyond the scope of this workshop. However here are some notes about this topic.

Deploying to Android

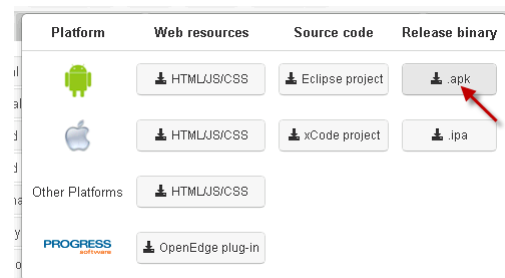
Before you can successfully deploy to an Android device, you'll need to package (build) your application for this platform. To do this you'll first want to have a look at the Project → App settings. There you'll find 2 vertical tabs related to Android deployment:

- Android binary
- Android permissions

The Android binary tab allows you to set things like the label and icon for the app, the version information, minimum Android versions, etc.

The Android permissions tab allows you to set a load of options specific to the Android platform. For more details on these options we'll have to refer to the Android documentation.

Once you're happy with these settings, you need to build the application using the Export button in the Mobile App Builder. Alternatively you can use the export option on the Mobile app in Developer Studio.



If all goes well, you'll get a file with the extension .apk which you need to get on your Android device.

Deploying to iOS

When you want to deploy an app to iOS, the steps are somewhat similar, but there are also some significant differences.

First of all, you'll need to register yourself as an iOS developer on the Apple website (<http://developer.apple.com>). Once that's done, you need to create a distribution certificate, and a provisioning file for the application. More information can be found on the Apple website.

Then you also have 2 tabs in the App settings specific for iOS, but similar to the ones for Android:

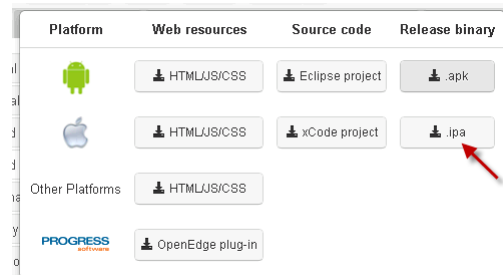
- iOS binary
- iOS keys

The iOS binary tab also has settings for the label, version, icons, etc. Specific to iOS are the Bundle ID (unique id for your app), the Distribution certificate, and the Provisioning profile.

Next you have the iOS keys tab, which allows you to a load of iOS specific settings.

When that's all ok, you can also export the app as an .ipa file using the button shown on the right.

Alternatively you can use the export option on the Mobile app in Developer Studio.



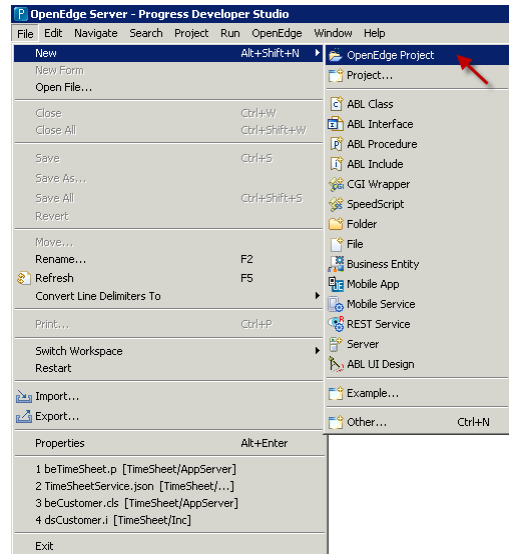
That .ipa file needs to be imported in iTunes, and then you use iTunes to sync the app to your device. Alternatively you can also scan the QR code with your iOS device which allows you to install the app over the air.

This process is only valid for testing purposes. When you want to upload your application to the AppStore, we have to point you to the Apple website for more information or you can get help from our Professional Services team.

Chapter 6 – OpenEdge Mobile Express

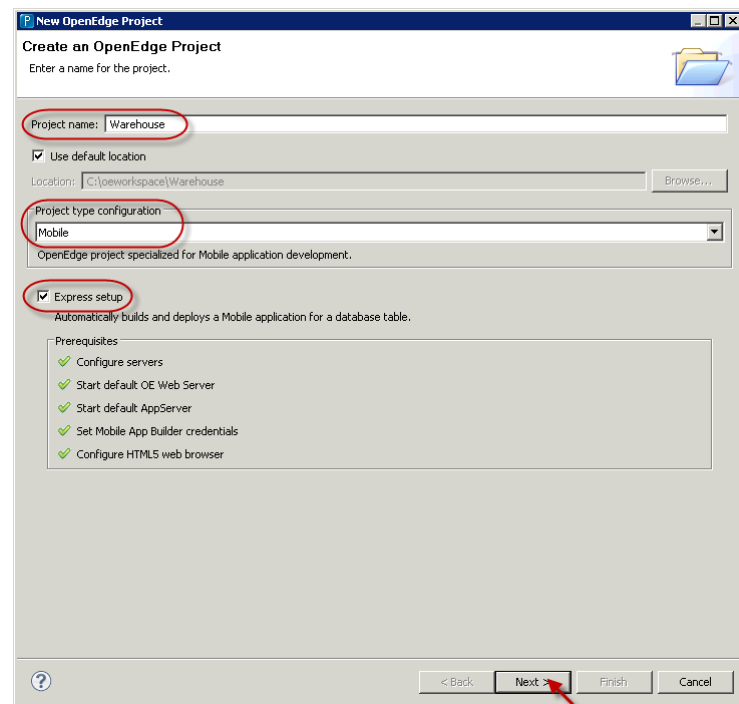
This chapter introduces a new feature of OpenEdge 11.3, Mobile Express. With this feature you can very quickly create a complete mobile app from a database table. When you have followed the steps below, you'll get a feeling of what it is.

1. Create a new OpenEdge Project



2. Enter the details for the new project:

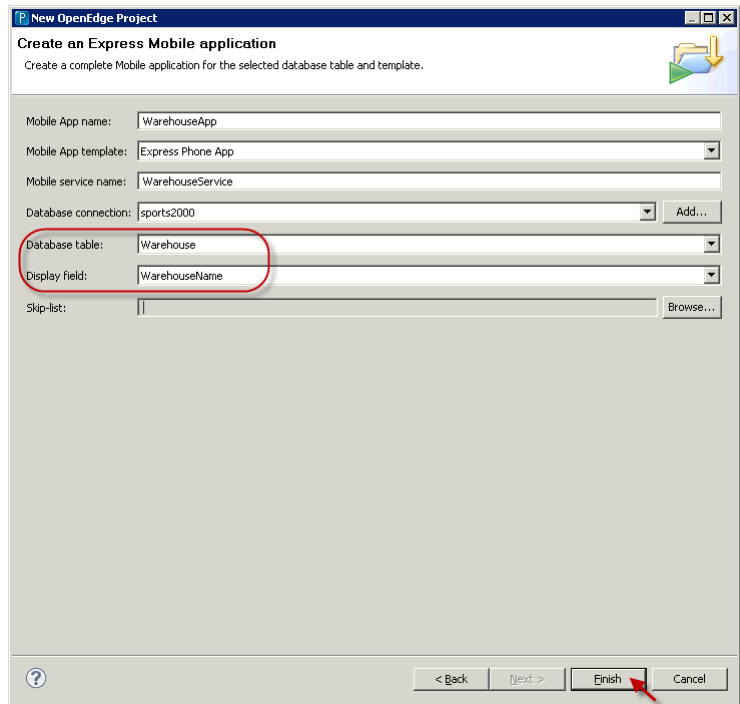
- Project name
Warehouse
- Project type configuration
Mobile
- Express Setup



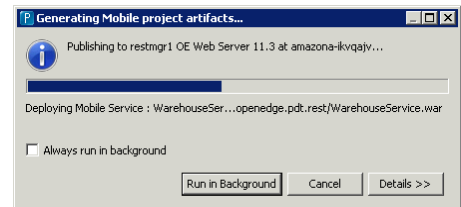
and click **Next**.

3. Select **Warehouse** as the Database table and **WarehouseName** as the Display field.

Click **Finish**.



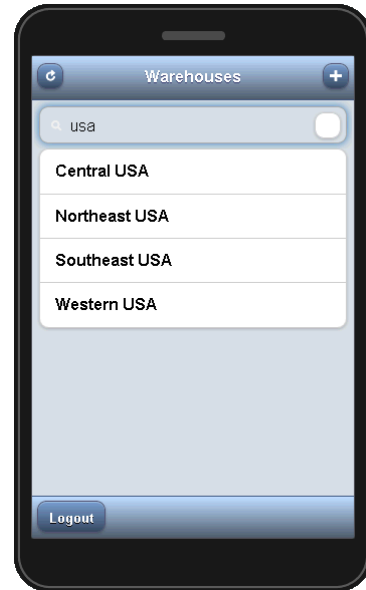
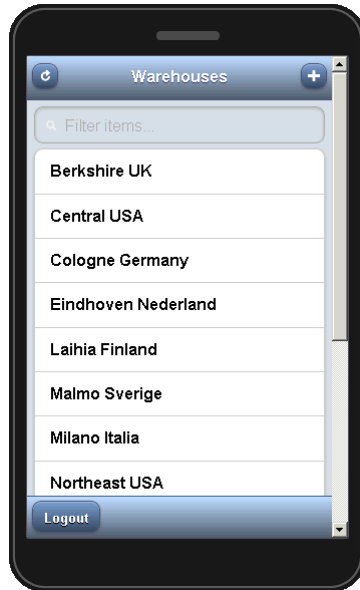
After you've clicked Finish it will take some time before everything is done. You will see a window with the status of the generation process.



When the process is complete the application will be launched automatically in the browser and the first page you'll get is a login page in the application. There's no login process behind it. It's a placeholder where you can put in your own authentication process. The generated application has full CRUD capabilities. You can follow the guided instructions below to check some of this out, or you can try it out by yourself.

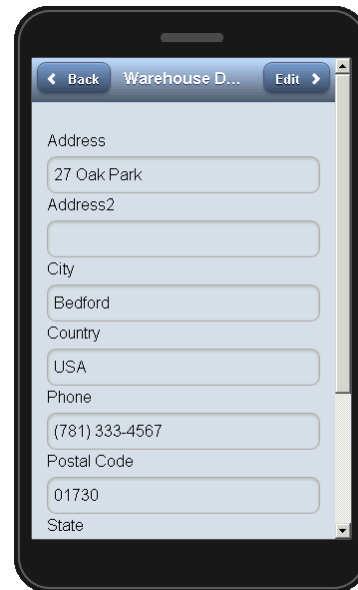


4. Click **Login** and you'll be presented with an overview of all the warehouses in the database on which you can filter using the field at the top of the screen.

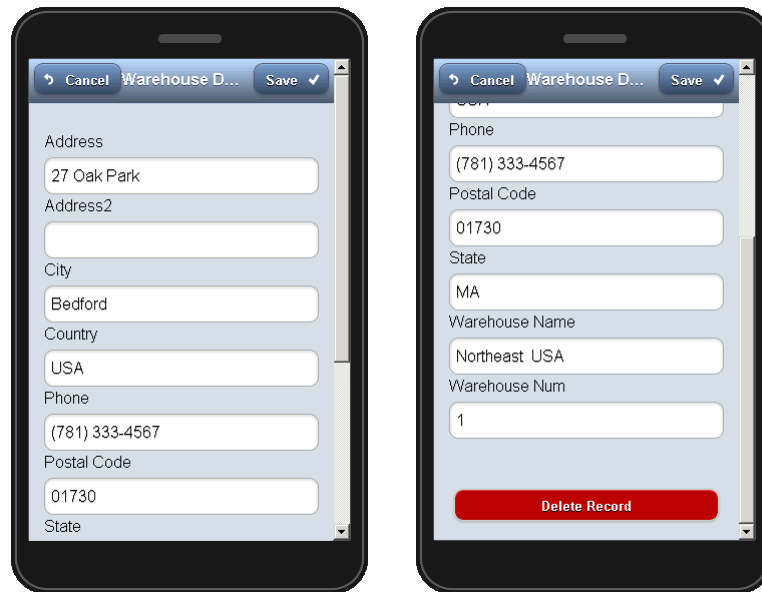


The "+" button allows you to add a new warehouse.
The **Logout** button brings you back to the login screen.

5. Click on a warehouse (e.g. **Northeast USA**) to go to the details page.



6. Click the **Edit** button to go into update mode.



The **Back** button changes into **Cancel**, and the **Edit** button change to **Save**. When you scroll down, you'll also see a **Delete Record** button, which is not there in read mode.

If you want to fine tune the application, e.g. change the page titles, use your own designed details page, use different images, etc. you can change settings in the **appconfig.js** file in your project. There is (or there will be soon) a white paper available on our Communities web site with more details on how to use the **appconfig.js** file.

Congratulations! You have now successfully completed this workshop.

Conclusion

As you've seen in our lessons, OpenEdge Mobile allows you to quickly build mobile applications for a variety of devices and platforms. Here are some suggestions to expand upon your learning:

FUN THINGS TO TRY WITH THE TIMESHEET EXAMPLE

- > If you have an Android device, try to deploy the TimeSheet application to it.
- > Use some of the in-built location based services to add a location to your timesheet directly from your mobile device
- > Find out how far away your colleagues are from you by placing their location on an interactive map
- > Add pictures and video to your application as proof of attendance at a customer site

FUN THINGS TO CREATE ON YOUR OWN

- > OpenEdge Mobile can do much more than just interact with your OpenEdge services, it is a powerful platform that will allow you to create many different types of applications, not just data centric ones. You can build apps to interact with stock tickers, weather feeds, twitter feeds, video capture, location services....
- > You can always access a completed time sheet application like the one you've built during this workshop on <http://tinyurl.com/TimeSheetExchange2013>.



ADDITIONAL TRAINING AND DEMONSTRATIONS

- > This workshop has given you an introduction to OpenEdge Mobile but we don't want you to stop there. To help you get started with OpenEdge Mobile talk to your progress account rep to discuss your training and services needs.

Appendix A – Code pieces

In this appendix you can find all the code you have to use in this workshop. You can copy/paste from here or you can use the file **OE Mobile Workshop.txt** located in this folder on your machine: **C:\OEMobile**.

DSCUSTOMER.I – TEMP-TABLE AND DATASET DEFINITION

```
DEFINE TEMP-TABLE eCustomer NO-UNDO
  BEFORE-TABLE eCustomerBefore
  FIELD CustNum AS INTEGER
  FIELD Name AS CHARACTER
  FIELD Address AS CHARACTER
  FIELD City AS CHARACTER
  FIELD State AS CHARACTER
  FIELD Country AS CHARACTER
  FIELD Phone AS CHARACTER
  FIELD Contact AS CHARACTER
  FIELD SalesRep AS CHARACTER
  FIELD Comments AS CHARACTER
  FIELD CreditLimit AS DECIMAL
  FIELD Balance AS DECIMAL
  FIELD Terms AS CHARACTER
  FIELD Discount AS INTEGER
  FIELD PostalCode AS CHARACTER
  FIELD Fax AS CHARACTER
  FIELD EmailAddress AS CHARACTER
  INDEX CustNum IS PRIMARY UNIQUE CustNum
  INDEX Comments IS WORD-INDEX Comments
  INDEX CountryPost Country PostalCode
  INDEX Name Name
  INDEX SalesRep SalesRep.

DEFINE DATASET dsCustomer FOR eCustomer.
```

BECUSTOMER – READ METHOD

```
/* Try to find filterdata in various fields: CustNum, Name, Comments, SalesRep */
DEFINE VARIABLE cWhereString AS CHARACTER NO-UNDO.

DEFINE DATA-SOURCE srcCustomer FOR Customer.
EMPTY TEMP-TABLE eCustomer.
BUFFER eCustomer:fill-mode = "MERGE".

IF filter EQ "" OR filter EQ ? THEN RETURN.

BUFFER eCustomer:ATTACH-DATA-SOURCE (DATA-SOURCE srcCustomer:handle).

/* CustNum */
cWhereString = SUBSTITUTE ("WHERE CustNum EQ &1", QUOTER(filter)).
IF cWhereString NE "" AND filter NE ? THEN
    DATA-SOURCE srcCustomer:FILL-WHERE-STRING = cWhereString.
DATASET dsCustomer:FILL ().
MESSAGE "beCustomer - READ:" DATA-SOURCE srcCustomer:fill-where-string
    VIEW-AS ALERT-BOX.

/* Name */
cWhereString = SUBSTITUTE ("WHERE Name BEGINS &1", QUOTER(filter)).
IF cWhereString NE "" AND filter NE ? THEN
    DATA-SOURCE srcCustomer:FILL-WHERE-STRING = cWhereString.
DATASET dsCustomer:FILL ().
MESSAGE "beCustomer - READ:" DATA-SOURCE srcCustomer:fill-where-string
    VIEW-AS ALERT-BOX.

/* Comments */
cWhereString = SUBSTITUTE ("WHERE Comments CONTAINS &1", QUOTER(filter)).
IF cWhereString NE "" AND filter NE ? THEN
    DATA-SOURCE srcCustomer:FILL-WHERE-STRING = cWhereString.
DATASET dsCustomer:FILL ().
MESSAGE "beCustomer - READ:" DATA-SOURCE srcCustomer:fill-where-string
    VIEW-AS ALERT-BOX.

/* SalesRep */
cWhereString = SUBSTITUTE ("WHERE SalesRep EQ &1", QUOTER(filter)).
IF cWhereString NE "" AND filter NE ? THEN
    DATA-SOURCE srcCustomer:FILL-WHERE-STRING = cWhereString.
DATASET dsCustomer:FILL ().
MESSAGE "beCustomer - READ:" DATA-SOURCE srcCustomer:fill-where-string
    VIEW-AS ALERT-BOX.

BUFFER eCustomer:DETACH-DATA-SOURCE ().

RETURN.
```

ABL PROCEDURE SAVETIMESHEET

```
PROCEDURE SaveTimeSheet:
    DEFINE INPUT    PARAMETER ipchEmployee AS CHARACTER NO-UNDO.
    DEFINE INPUT    PARAMETER ipchCustomer AS CHARACTER NO-UNDO.
    DEFINE INPUT    PARAMETER ipchDate     AS CHARACTER NO-UNDO.
    DEFINE INPUT    PARAMETER ipchFrom     AS CHARACTER NO-UNDO.
    DEFINE INPUT    PARAMETER ipchUntil    AS CHARACTER NO-UNDO.
    DEFINE INPUT    PARAMETER ipchComments AS CHARACTER NO-UNDO.

    MESSAGE
        "Employee:" ipchEmployee SKIP
        "Customer:" ipchCustomer SKIP
        "Date:"      ipchDate      SKIP
        "From:"      ipchFrom      SKIP
        "Until:"     ipchUntil     SKIP
        "Comments:" ipchComments
        VIEW-AS ALERT-BOX.
END PROCEDURE.
```

JAVASCRIPT – ADD CATALOG

```
/* if the first time, or not logged in, try to log in */

if ($t.ProgressSession == undefined || $t.ProgressSession.loginResult !=
progress.data.Session.LOGIN_SUCCESS) {

    /* TimeSheetService_beCustomer_Settings is the Settings Service created when
    the catalog was loaded. */

    var settings = TimeSheetService_beCustomer_Settings;

    /* Putting the session object into Mobile App Builder's namespace
    so that it can be accessed later, if needed */

    var pdsession = $t.ProgressSession;

    if (pdsession == undefined)
        pdsession = $t.ProgressSession = new progress.data.Session();

    if (settings.serviceURI == undefined || settings.serviceURI == '')
        console.log("serviceURI was not specified." +
            " catalogURI: " + settings.catalogURI +
            " resourceName: " + settings.resourceName);

    var loginResult = pdsession.login (settings.serviceURI, "", "");
    if (loginResult != progress.data.Session.LOGIN_SUCCESS) {
        var msg = 'ERROR: Login failed with code: ' + loginResult;
        console.log(msg);
        alert(msg);
        return;
    }
    // load catalog
    pdsession.addCatalog(settings.catalogURI);
}
```

Appendix B – Tips and tricks

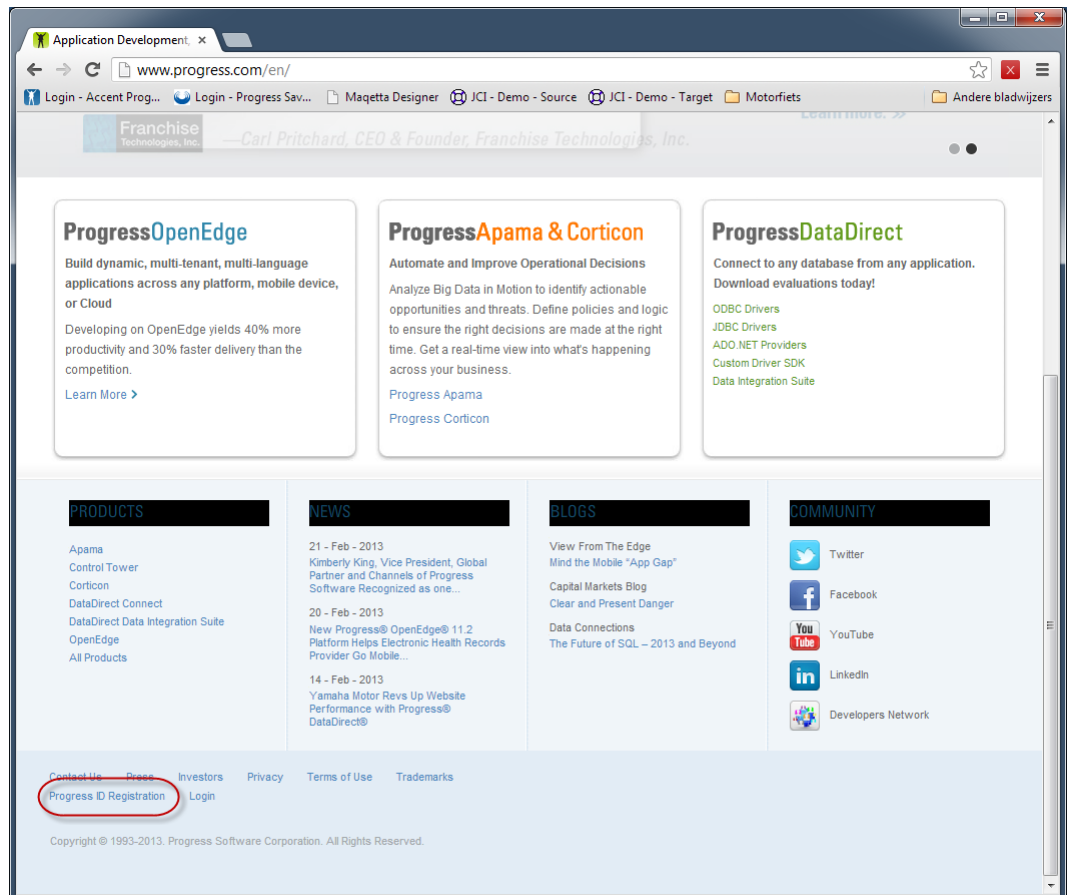
In this appendix you'll find some tips and tricks to help you while you are developing OpenEdge Mobile Apps.

1. To avoid caching issues you can add the cache killer to Google Chrome <https://chrome.google.com/webstore/category/home?hl=en>
2. When you accidentally close the Mobile App Builder, you can always reopen it from the Project Explorer View by right-clicking on the *Mobile Applications* → *TimeSheetApp* node and choosing the **Open** option. Double-clicking on the *TimeSheetApp* node will also work.
3. If you get a blank screen when you start the mobile app, you most likely have a problem with the script that add the catalog to the application. Check for typo's, especially in the references to the Settings, and in the values you had to enter for the CatalogURI and the ServiceURI.
4. Check if restbroker1 and restmgr1 have been started (Servers view in Progress Developer Studio)
5. When you don't see your changes in the mobile app when you run it, clear the browser cache (see also tip 1)
6. In Google Chrome you can use Inspect Element (right-click in the browser or press F12) to start debugging. Look at the Console tab first and check for any error messages
7. Other places to look for errors are the various log files:
 - a. C:\OpenEdge\WRK\restbroker1.broker.log
 - b. C:\OpenEdge\WRK\restbroker1.server.log
 - c. C:\Progress\OpenEdge\servers\tomcat\webapps\TimeSheetService\WEB-INF\adapters\logs\TimeSheetService.log
 - d. C:\Progress\OpenEdge\servers\tomcat\webapps\TimeSheetApp\WEB-INF\adapters\logs\TimeSheetApp.log

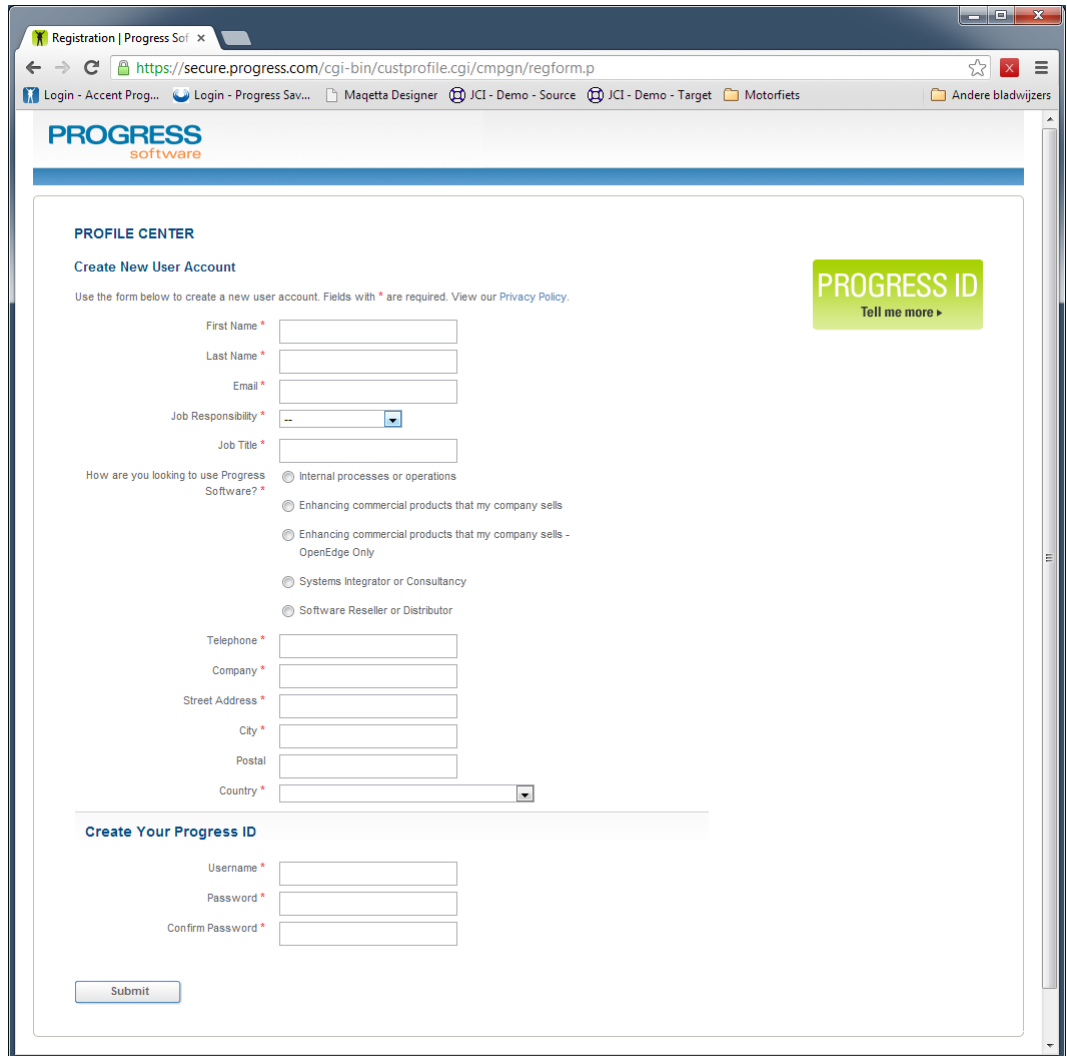
Appendix C – Getting a Progress login

If you don't have a Progress login yet, follow these steps to get one.

1. Go to <http://www.progress.com>, scroll to the bottom of the page click **Progress ID Registration**.



2. Complete the form and click **Submit**



The screenshot shows a web browser window with the URL <https://secure.progress.com/cgi-bin/custprofile.cgi/cmpgn/regform.p>. The page header features the Progress Software logo. The main content area is titled "PROFILE CENTER" and "Create New User Account". It includes a "PROGRESS ID" badge with a "Tell me more" link. The registration form consists of several sections:

- Personal Information:** First Name *, Last Name *, Email *, Job Responsibility *, and Job Title *.
- Usage Intent:** "How are you looking to use Progress Software?*" with radio button options: Internal processes or operations, Enhancing commercial products that my company sells, Enhancing commercial products that my company sells - OpenEdge Only, Systems Integrator or Consultancy, and Software Reseller or Distributor.
- Contact Information:** Telephone *, Company *, Street Address *, City *, Postal, and Country *.
- Account Creation:** "Create Your Progress ID" section with Username *, Password *, and Confirm Password *.

A "Submit" button is located at the bottom left of the form area.

You will receive an email with your Progress Login Credentials.
Your Mobile App Builder evaluation period of 60-days will start automatically when you log on to the Mobile App Builder for the first time.



Progress Software

Progress Software Corporation (NASDAQ: PRGS) is a global software company that simplifies the development, deployment and management of business applications on premise or on any Cloud, on any platform and on any device with minimal IT complexity and low total cost of ownership.

Worldwide Headquarters

Progress Software Corporation, 14 Oak Park, Bedford, MA 01730 USA

Tel: +1 781 280-4000 Fax: +1 781 280-4095 On the Web at: www.progress.com

For regional international office locations and contact information, please refer to the Web page below: www.progress.com/worldwide

Progress and [ALL PRODUCT NAMES LISTED IN PUBLICATION] are trademarks or registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and other countries. Any other marks contained herein may be trademarks of their respective owners. Specifications subject to change without notice.

© 2013 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.