

Developing for Multiple Mobile Platforms Using Web Technologies in RadPHP XE2

By Al Mannarino
Embarcadero Technologies

Americas Headquarters

100 California Street, 12th Floor
San Francisco, California 94111

EMEA Headquarters

York House
18 York Road
Maidenhead, Berkshire
SL6 1SF, United Kingdom

Asia-Pacific Headquarters

L7. 313 La Trobe Street
Melbourne VIC 3000
Australia

Table of Contents

Introduction.....	- 1 -
How to create a Mobile App using RadPHP XE2	- 6 -
Part 1 -Install the Android Project Tools on your computer	- 6 -
Part 2 - Set-up your InterBase database connection to the osCommerce database	- 7 -
Part 3 - Connect to InterBase Database using RadPHP XE2.....	- 8 -
Part 4 - Create a new Mobile Application.....	- 10 -
Part 5- Create an Android application.....	- 20 -
Part 6 - Alternate method to Manually Start your Android Emulator	- 24 -
Part 7- Create the iOS application.....	- 25 -
Part 8 - Create the BlackBerry application	- 33 -
Appendix A - How to create the InterBase osCommerce database	- 41 -
Part 1 - InterBase XE Developer Edition Installation	- 41 -
Part 2 - Setting up InterBase Database.....	- 47 -

Corporate Headquarters
100 California Street, 12th
Floor
San Francisco, California
94111

EMEA Headquarters
Thames House
17 Marlow Road
Maidenhead, Berkshire
SL6 7AA, United Kingdom

Asia-Pacific Headquarters
Level 9, 390 St Kilda Road
Melbourne VIC 3004
Australia

INTRODUCTION

Over the past decade, the mobile landscape has changed rapidly. Smartphones have been dominating the market and mobile phones and tablets have become the go-to devices for consumers and business customers alike.

Platforms like iOS and Android have been showing double digit growth year over year, and mobile application development has presented a viable business opportunity for individual developers and enterprise companies alike.

When the iPhone was first introduced to consumers in 2007, it changed the mobile landscape forever. It provided a totally new, interactive mobile experience unlike anything the industry had ever seen before.

With the introduction of the iOS SDK for mobile application development in early 2008, and the launch of the Apple App Store a few months later, a new tech craze was born. Developers around the world could now create mobile app versions of their popular desktop applications. Hundreds of thousands of new applications were developed, allowing users to be productive, creative, and to have fun on their mobile devices. However, back then developers were limited to Apple's tools and only able to code their applications in Objective-C using Xcode.

In 2011, Apple loosened its restrictions and allowed developers to build iOS apps and sell them through Apple's App Store without using Objective-C and Apple's development tools. This spurred new technologies, including mobile web app development frameworks that allow users to create applications for multiple mobile platforms including iOS, Android and BlackBerry using standard web programming languages such as HTML, Javascript and CSS.

Mobile applications built on web technologies can access device sensors and services using platforms such as PhoneGap and can be packaged as native applications. Several Javascript libraries have moved to the forefront of web and mobile web application development. One of those Javascript libraries is jQuery. jQuery has long been a popular JavaScript library for creating rich, interactive websites and web applications. However, since it was designed primarily for desktop browsers, it does not have many features specifically designed for building mobile web applications. jQuery Mobile is a project that addresses this shortfall. It is a framework built on top of jQuery that provides a range of user interface elements and features for you to use in your mobile applications.

PhoneGap is an application platform based on HTML5 that enables developers to build native mobile applications with access to device sensors and services such as the camera, GPS and accelerometer using the aforementioned web technologies. PhoneGap leverages web technologies that developers already know such as HTML, JavaScript and CSS and allows them to build applications with a native look and feel or a common UI across all the target platforms. Applications built with PhoneGap using web technologies can be distributed on the most popular mobile marketplaces, like the Apple App Store and Android Marketplace.

With the release of RadPHP XE2, Embarcadero introduced visual mobile web application development using jQueryMobile and PhoneGap making it easy to work with those technologies. RadPHP XE2 wraps jQuery Mobile features into mobile components, and provides a wizard for PhoneGap. You can create mobile web applications directly from RadPHP, use jQuery Mobile components to create touch-friendly interfaces and PhoneGap components to interact with mobile device hardware, and deploy your applications to Android, iOS and BlackBerry devices.

RadPHP XE2 for Mobile App Development allows you to:

- Build mobile-optimized touch-friendly web applications for most mobile devices in the market using new jQuery Mobile components
- Visual Mobile Designer to see how apps will look on the real device
- Convert PHP apps into standalone mobile apps for iOS, Android and BlackBerry
- Interact with the mobile device hardware, like GPS, accelerometer, camera, etc.
- Expanded Zend Framework component set with email, barcode, Google, RSS and more
- Give your mobile apps access to device features like cameras, accelerometers and GPS in an easy component-based way

RadPHP is unique for mobile app development since it's the only visual Rapid Application Development (RAD) environment for PHP. Other development environments may include a visual HTML designer but only RadPHP provides a visual designer that incorporates HTML and HTML templates with visual PHP components that you can drag and drop to create user interfaces and database connectivity. It also has an integrated component class library that lets developers quickly and visually create PHP web applications and integrate PHP open source packages, source code and components.

The RadPHP Visual Component Library (RPCL) adds mobile components and wizards to leverage your PHP skills to create native mobile optimized Android, iOS and BlackBerry applications.

RadPHP XE2 includes dozens of jQuery Mobile components.

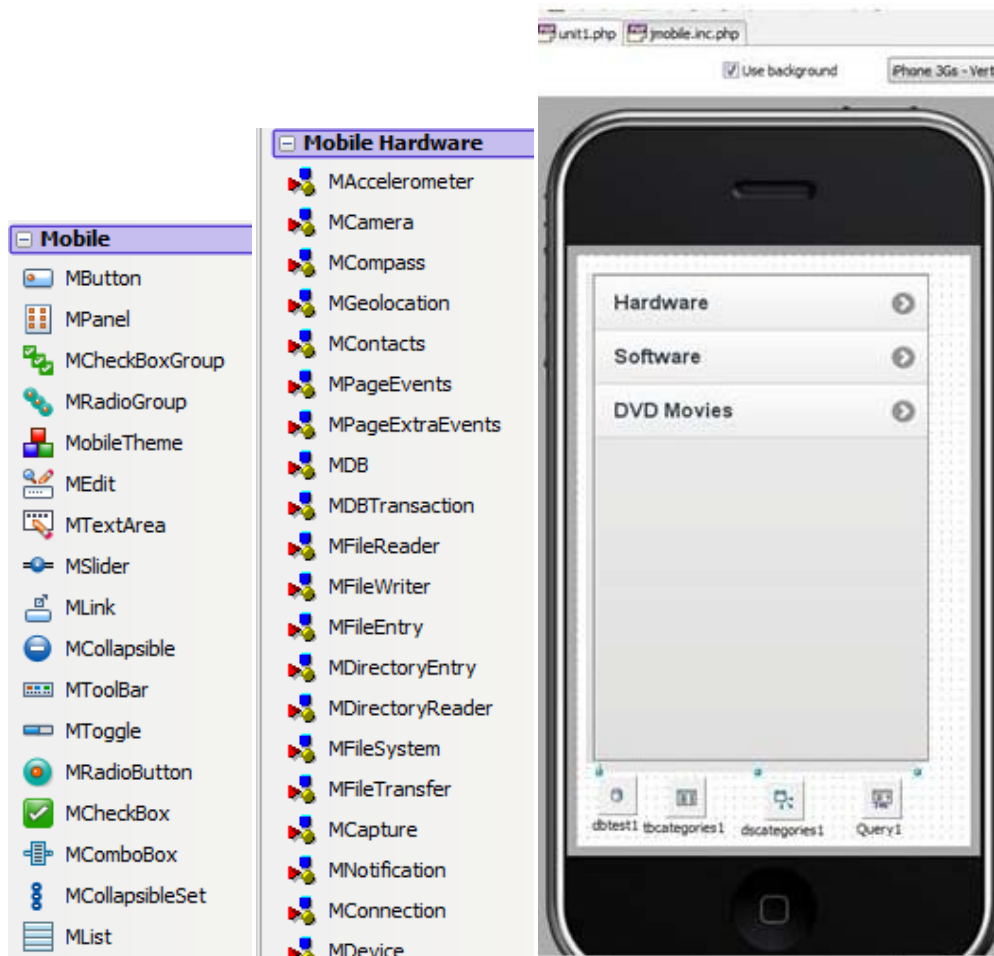


Figure 1

Figure 1 The RadPHP IDE lets you quickly create mobile web apps directly from the IDE and use the jQuery Mobile components to create touch friendly interfaces. You can simulate iOS (iPhone, iPad and iPod Touch), Android and BlackBerry devices and you can add new devices, such as Android tablets.

Figure 2 shows what the RadPHP XE2 IDE looks like for a Mobile Application project. In this screenshot you see an iPhone with a **Mlist** component that connects to a database that will display parent – child data on the mobile device.

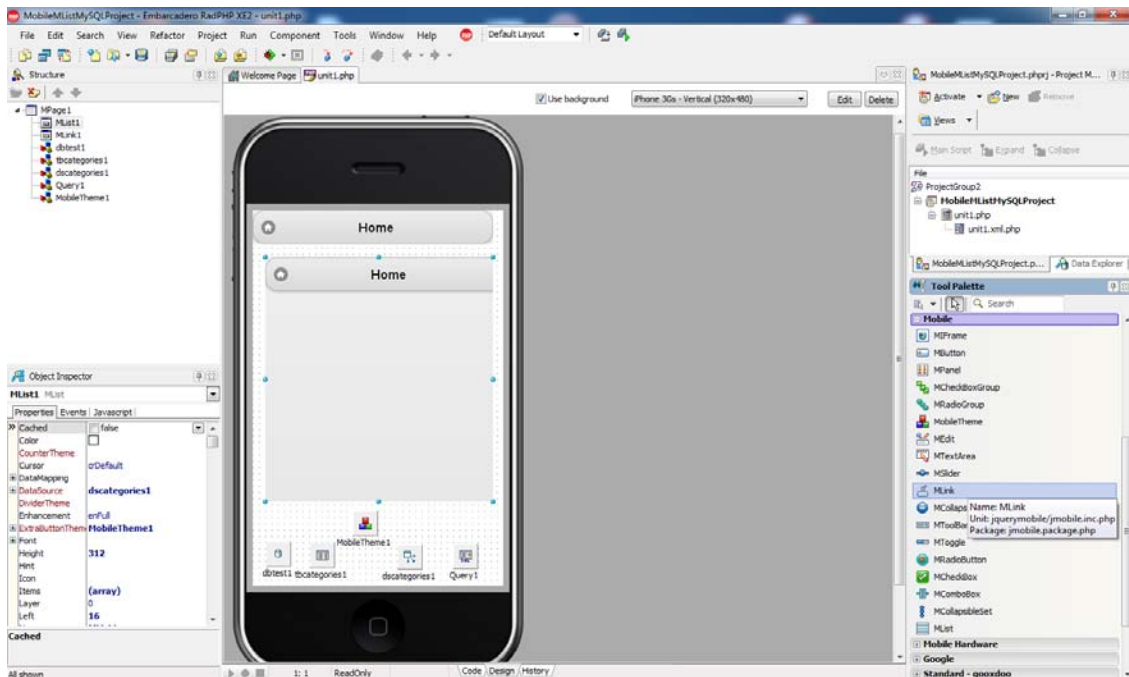


Figure 2

And the code is 100% pure PHP, JavaScript and HTML code! In Figure 3 you see the jQuery Mobile include file for all the jQuery Mobile components.

```

unit1.php
- use_unit("jquerymobile/jmobile.inc.php");
- use_unit("dtables.inc.php");
- use_unit("db.inc.php");
10 use_unit("dbgrids.inc.php");

- //Class definition
- class MPage1 extends MPage
- {
-     public $MList1 = null;
-     public $dbtest1 = null;
-     public $tbcategories1 = null;
-     public $dscategories1 = null;
-     public $Query1 = null;
20     public $MLink1 = null;
-     public $MobileTheme1 = null;

-     function MPage1Create($sender, $params)
-     {
-         $this->MLink1->Link = $this->curPageURL() . '?id=0&list=' . $GET["list"];
-     }
27     class MLink extends CustomMLink - jmobile.inc.php (2865,7)
-     {
-         Links are rendered as buttons but they don't submit forms.
-         All local pages referenced in links are loaded via Ajax. A
-         transition can be established when clicking the Link and will be
-         shown when the page is loaded via Ajax. Ajax can also be
-         disabled. @see Label @example JQueryMobile/mlink.php
30     }
-     function c
-     {
-         $pageUR
-         if($_SE
-         $pageURL .= "://";
-         if($_SERVER["SERVER_PORT"] != "80")
-         {

```

Figure 3

After we create our Mobile app in the RadPHP IDE, we'll use the Wizard for PhoneGap to generate all the files we need to create our native iOS, Android and/or BlackBerry mobile app based on web technologies.

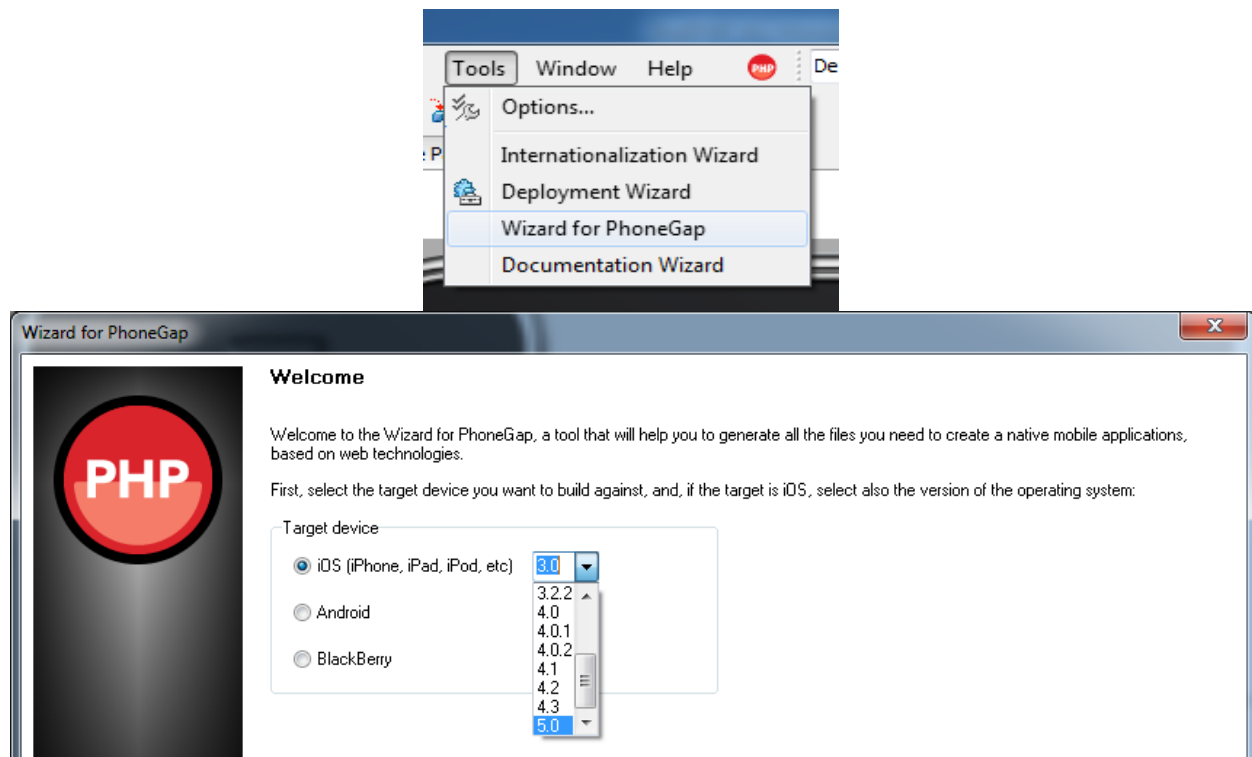


Figure 4

With RadPHP, you build your mobile app once, and the Wizard for PhoneGap creates the native mobile apps for iOS, Android and BlackBerry.

HOW TO CREATE A MOBILE APP USING RADPHP XE2

This section provides step-by-step instructions to create a mobile app using RadPHP XE2, and deploy the app to Android, iOS and BlackBerry.

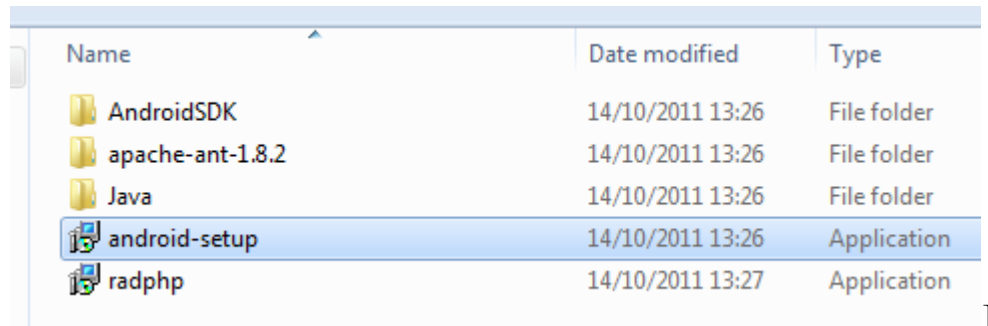
PART 1 -INSTALL THE ANDROID PROJECT TOOLS ON YOUR COMPUTER

Use **android-setup** from RadPHP XE2 installation files:

Android Setup

Before you can build your mobile application for Android, you will need to setup the Android SDK and some additional software. RadPHP includes an *installation*

wizard to ease this process. In the folder or CD where you first run the RadPHP installer, you will find this wizard under the name **android-setup.exe**. See (Figure 5).



Name	Date modified	Type
AndroidSDK	14/10/2011 13:26	File folder
apache-ant-1.8.2	14/10/2011 13:26	File folder
Java	14/10/2011 13:26	File folder
android-setup	14/10/2011 13:26	Application
radphp	14/10/2011 13:27	Application

Figure 5

So, run **android-setup.exe** and let it guide you through the installation process. When you run the installer, you will be required to provide administration permissions. Your current user must be the administrator and you must provide administration permissions with it. Do not provide administration permissions through a different user account.

Please see http://docwiki.embarcadero.com/RadPHP/en/Android_Setup for the detail steps for installing the RadPHP Android Project Tools, Java SE Development Kit, Android SDK Tools and Apache Ant.

At the end, another wizard window will pop up, and will give you the option to reboot. Click **Finish** to reboot.

Note: You will not be able to build Android applications until you reboot your system.

PART 2 - SET-UP YOUR INTERBASE DATABASE CONNECTION TO THE OSCommerce DATABASE

This mobile app will use an InterBase osCommerce database created from Create_osCommerce.sql, located in **Appendix A -How to create the InterBase osCommerce database**.

A. Download and Install

If you don't already have InterBase installed from the RadPHP installation, then download and Install the **InterBase XE Developer Edition** from:
<https://downloads.embarcadero.com/free/interbase>

InterBase XE Developer Edition

InterBase XE Developer Edition provides all developers the best cross-platform database to build and test database applications for embedded and SME segments. Developers are free to choose the standard connectivity they want, and make use of a mature SQL92 compliant database for their applications. Database Performance monitoring allows the developer to streamline applications for optimal deployment.

Warning: This version of InterBase is licensed for development use only. Deployment Edition must be purchased separately.

B. Registration

You can also access the download link for InterBase XE Developer Edition along with the registration serial number by going to:

C:\Program Files (x86)\Embarcadero\RadPHP\4.0\Welcomepage\install.htm file.

C. Follow the steps in Appendix A - How to create the InterBase osCommerce database.

PART 3 - CONNECT TO INTERBASE DATABASE USING RADPHP XE2

In RadPHP, using the **Data Explorer** tab, create a new InterBase connection to your osCommerce database (created from Appendix A).

1. Right-click on InterBase | Add New Connection | Connection Name = **IB_OSCOMMERCE**

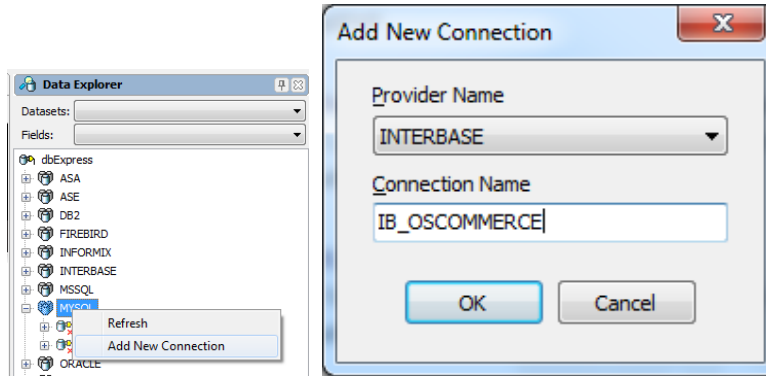
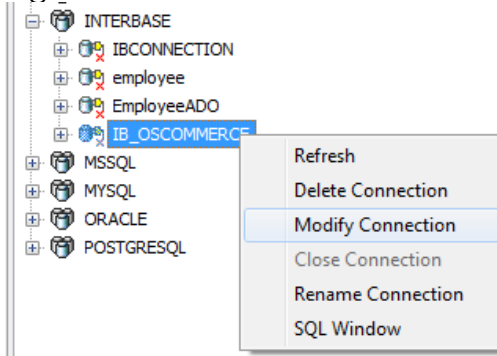


Figure 6

Click OK.

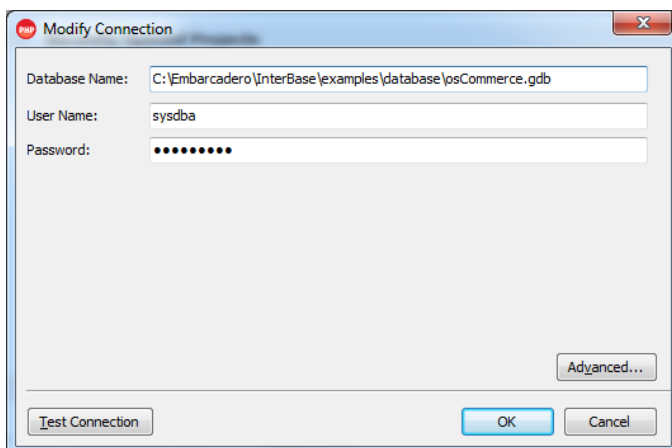
Right-click on the IB_OSCOMMERCE node >> Modify Connection.



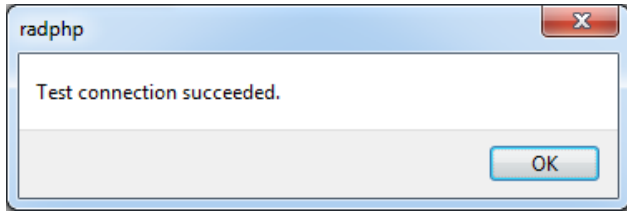
2. Enter your connection values for your InterBase instance.

Database Name =

C:\Embarcadero\InterBase\examples\database\osCommerce.gdb



3. Click Test Connection to verify connection.



Click OK.

4. In Data Explorer, expand your "IB_OSCOMMERCE" connection and verify you have your tables:

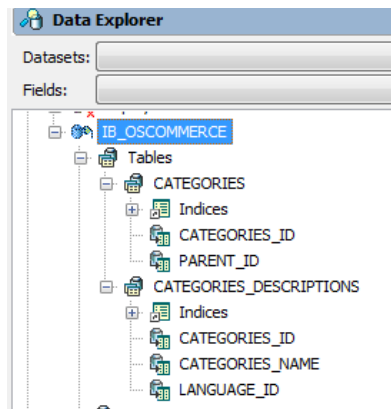


Figure 7

5. As an optional feature to explore, right-click on IB_OSCOMMERCE | **SQL Window** | and select and drag categories and categories_descriptions onto the designer. Look at the "code" and "data" tabs.

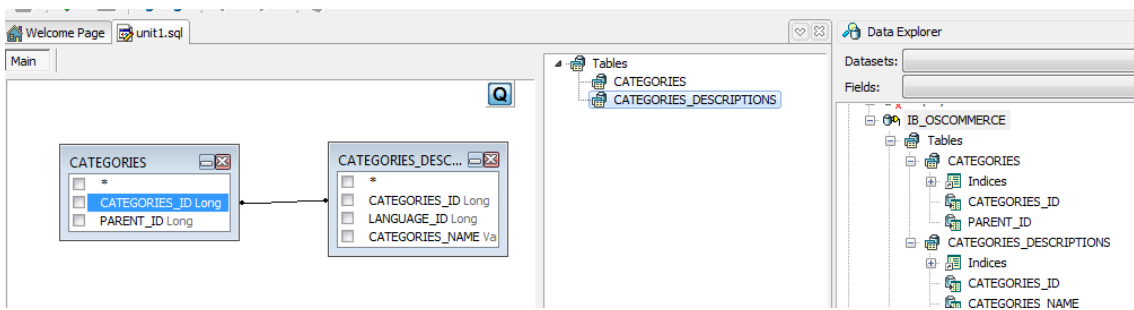


Figure 8

PART 4 - CREATE A NEW MOBILE APPLICATION

1. Click "New Items" icon.
2. Select RadPHP Projects | Mobile Application

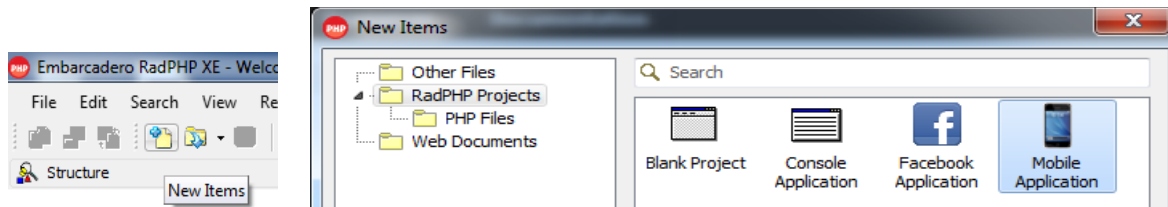


Figure 9

Click OK.

Your Mobile Application gets created and you should see your **jQuery Mobile** components, like this:

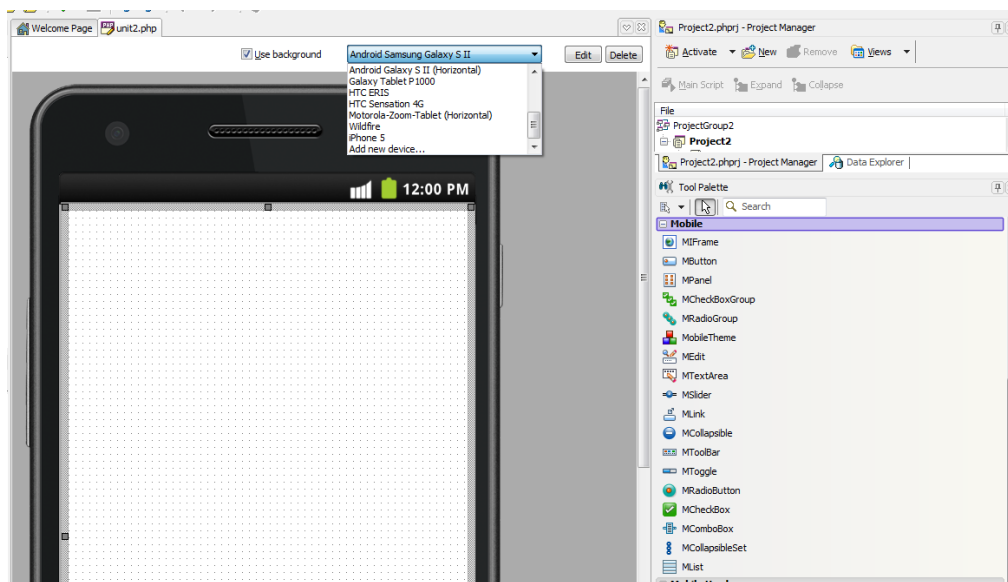


Figure 10

Note: You have the option to select several background images for mobile devices for Android, iOS and BlackBerry devices, plus the option to add a New Device to your project. RadPHP XE2 includes many device images located here: C:\Program Files (x86)\Embarcadero\RadPHP\4.0\repository\images.

Note: It does not matter which background you select, you will get to target Android, iOS and BlackBerry device from the Wizard for PhoneGap.

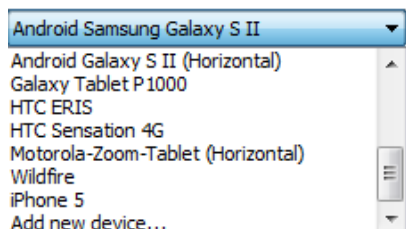
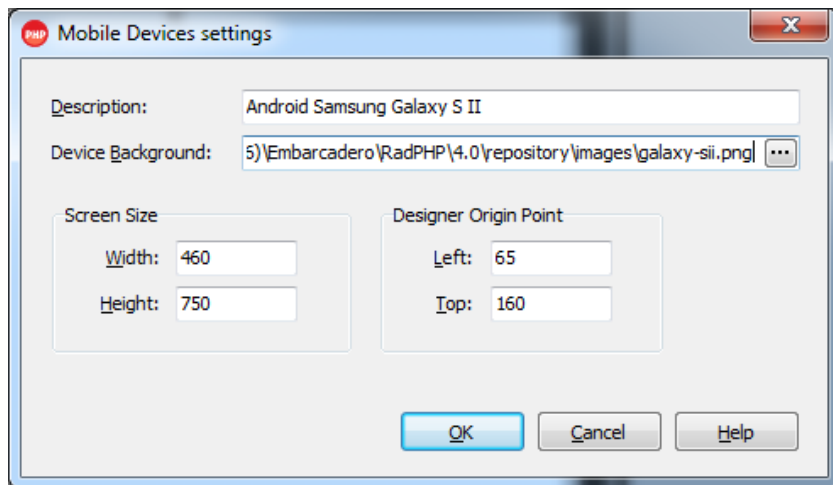


Figure 11

We'll use an **Android Samsung Galaxy S II** for this project.



To this project, we will add an **MList** component. A **MList** is a dynamic list that we can fill up with data, either entered manually or we can set-up a database connection and retrieve the data from a database.

MList allows you to create an ordered or unordered list. Every element on the list can have up to two links, the secondary link will render as an Extra Button. MLists can be nested.

- Control the **Theme** and color variation of a MList with the MobileTheme component associated to it.
- With **DividerTheme** a MobileTheme can be associated to all the dividers included in the MList
- **ExtraButtonTheme** allows you to associate another MobileTheme component to the Extra Button of every element on the list.
- Change the **SystemIcon** of the Extra Button to a different icon.
- Select your own **Icon** for the Extra Button.
- Indicate the **Type** of list: tOrdered or tUnordered
- With **isFiltered** we'll add a filter Bar on the top of the list
- **isWrapped** wraps the list with a grouped style.
- Use **Items** to add elements to the List. Every item has the following attributes:
 - **Caption** Text to display, HTML are tags allowed.
 - **Link** URL of the page to load.
 - **MList** Select a Mlist from a list of the available MLists on the form to include a nested MList.
 - **ExtraButtonHint** Alternative text for the Extra Button.

- o **ExtraButtonLink** URL of the Extra Button.
- o **isDivider** Will make this element a list divider.
- o **CounterValue** Numeric input that will display on the right side of the element.
- o **Thumbnail** Image to use as thumbnail on the right of the element
- o **isIcon** indicates that the Thumbnail image is an icon and will be displayed in a different way.

3. Add an **MList** component onto the page and expand the component to fill the page, like this:



Figure 12

4. Next, to get the data, we'll use a database connection. We'll be using the sample **IB_OSCommerce** database. From **Data Explorer**, drop the **categories** table onto the mobile device, like this:

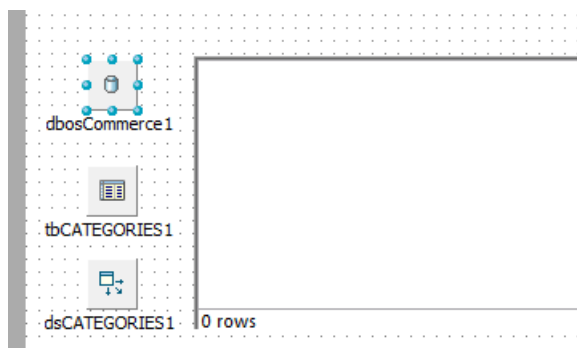


Figure 14

This adds four components to the page (**Database, Table, Datasource and DBGrid**) onto the page.

5. Run the application to test your InterBase connection. (Press F9)
You should be asked to **save** your unit:

- a. Create a new folder for the project, such as **RadPHPMListInterBase**

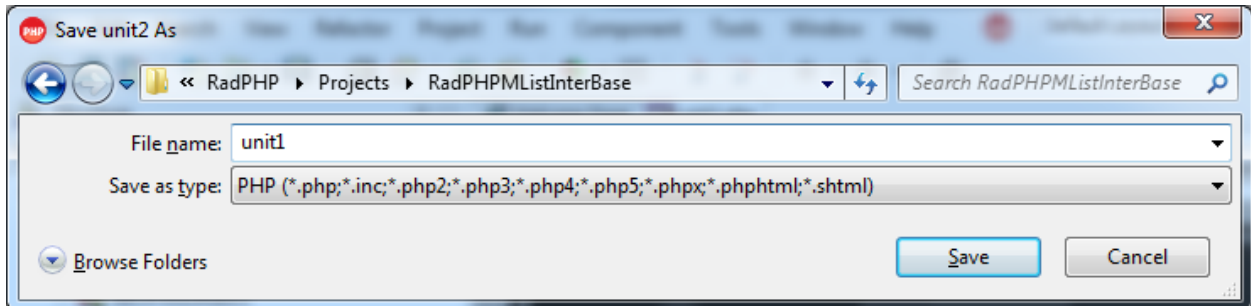


Figure 15

Click **Save** to save your unit1.php

- b. **Save** your Project, such as **RadPHPMListInterBaseProject**

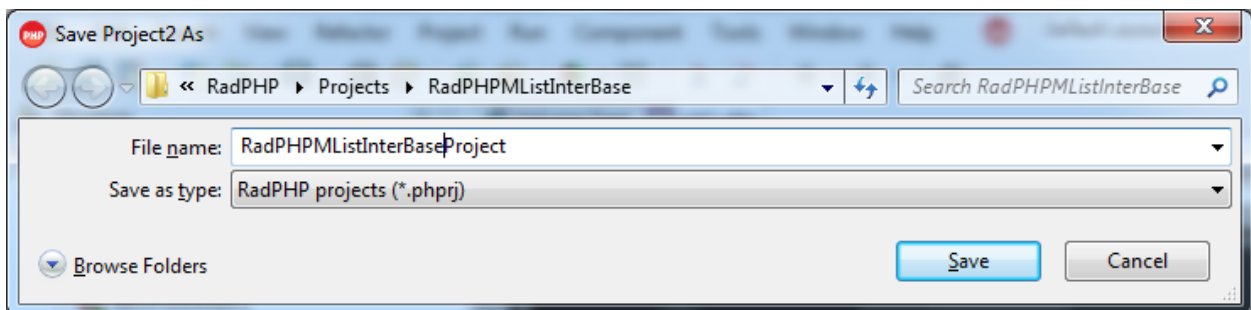


Figure 16

Click **Save**.

The project should now run and you should see this:

CATEGORIES_ID	PARENT_ID
1	0
2	0
3	0
5	1
6	1
7	1
8	1
16	1
17	1

10 rows

Figure 17

This was a good test to verify you can connect and get data returned from your **categories** table.

You can close the web page.

6. Delete the DBGrid component. We don't need the DBGrid for our mobile application. We'll be using an MList to display the data on our mobile device.

7. Add a **Query** component (from Data Access) onto the page, and set **Database = dbosCommerce1**, set **Active = true**.



Figure 18

8. On the same Query component, double-click the **SQL** property and add this SQL:

```
select * from categories
left join categories_descriptions on
categories_descriptions.categories_id=categories.categories_id
and categories_descriptions.language_id=1
```

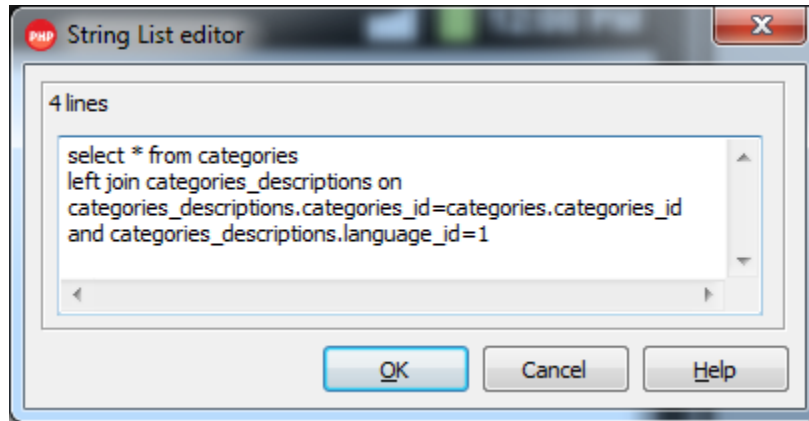


Figure 19

This query returns all the categories and their category descriptions for language = English (1).

Click **OK**.

9. Select the **MList** component. Set **DataSource** = **dsCATEGORIES1**, Set **Dataset** = **Query1**

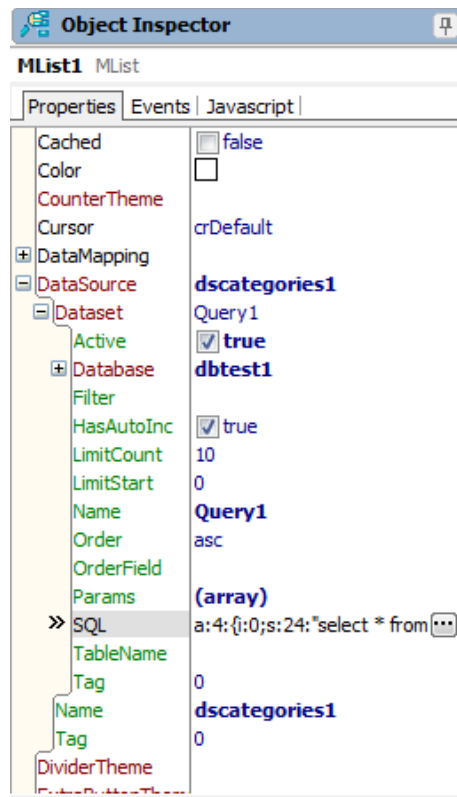


Figure 20

10. Set **DataMapping** on **MList** to correspond to your database table values to display on the **MList**, like this:

- a. The **BaseParentFieldValue = 0** will be the first value (column) to display on our MList.
- b. Set **Caption** to be what we want to use for each element on the MList = **CATEGORIES_NAME**
- c. Set **idField = CATEGORIES_ID**
- d. Lastly we will establish the **ParentField = PARENT_ID**

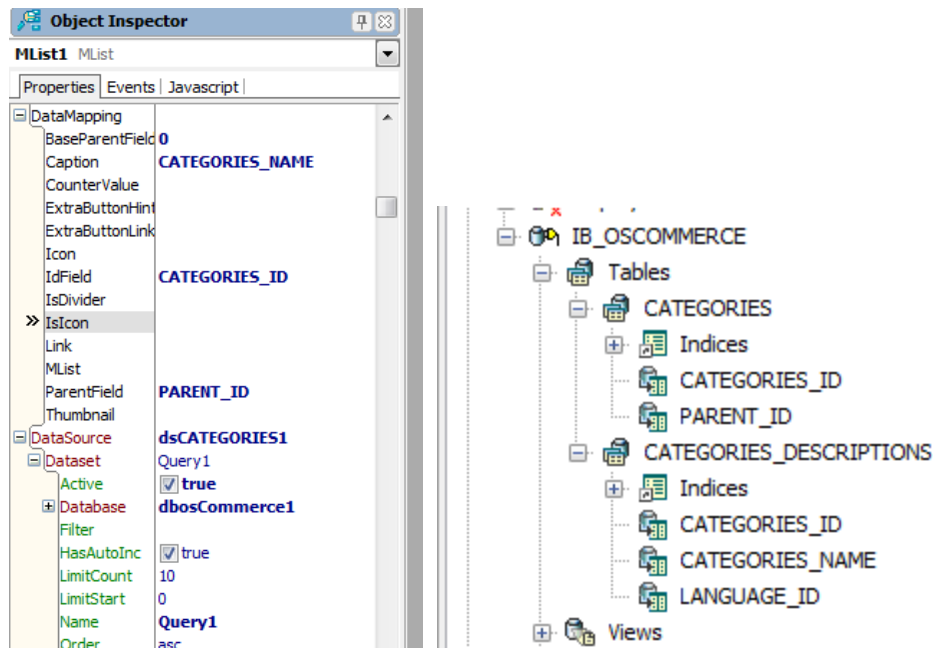


Figure 21

11. Run the mobile application (F9). Your MList should display your CATEGORIES values, like this:

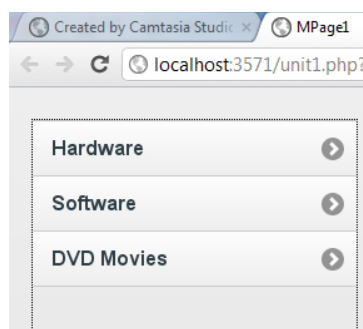


Figure 22

12. Click on the Hardware category, and you will drill down to its sub-categories, like this:



Figure 23

Congratulations!

NOTE: This sample osCommerce database only has this one level of drill down data, but if it had more, the application would continue to drill down as needed. If you click on any of these sub-categories, you will just get a blank screen, and that's normal expected behavior.

13. Next, we'll create a mobile application that will display this data on Android, iPhone and BlackBerry. First, we'll create an Android application.

For our Android mobile application, we cannot put our large InterBase database, nor can we process the PHP files on the mobile device, so we'll have our application make some AJAX calls to another application hosted on a server that will provide the data. We will be using the same PHP Apache Server that we just started in step 11.

So the first thing we need to do is tell our main application to use AJAX.

14. Set **UseAjax = true** on the **MPage1** component:

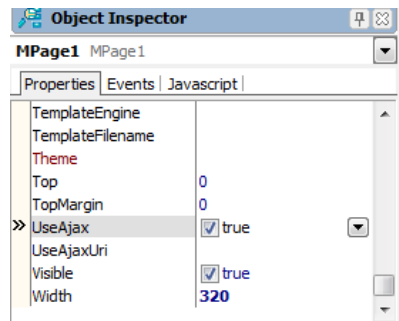


Figure 24

15. Next, we'll establish the AJAX URI (**UseAjaxUri**) that we will connect to and collect the data.

NOTE: For the URI, do not use localhost, use your real IP address, such as the output from running ipconfig, for example:

IPv4 Address. : 10.1.1.72

For this demo, I'll use the <http://10.1.1.72:3571/unit1.php> that we have already been using in this example.



Figure 25

16. Save ALL.

17. Run the app, and verify you connect to <http://10.1.1.72:3571/unit1.php>

18. **IMPORTANT:** Keep your PHP Web Server running (do not close the web window). The mobile apps we create will connect to this running PHP Web Server.

19. Next, we'll use the new "**Wizard for PhoneGap**" feature to create our Android mobile application.

Tools | Wizard for PhoneGap

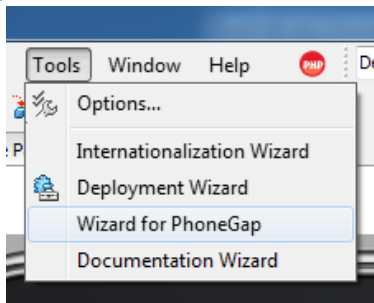


Figure 26

Here we have the option to create either an iOS, Android or BlackBerry mobile application. This wizard will generate all the files you need to create a native iOS, Android and/or BlackBerry mobile application based on web technologies.

PART 5- CREATE AN ANDROID APPLICATION

1. From **Tools** | **Wizard for PhoneGap** select and create an **Android** application.

Select Android

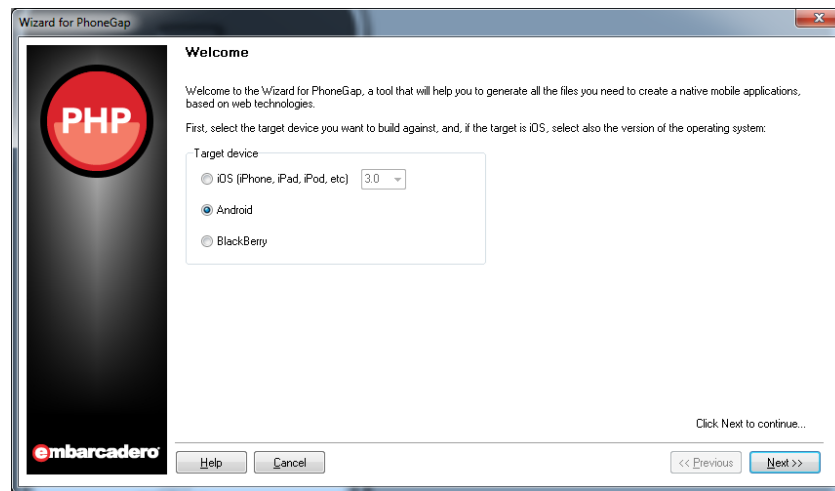


Figure 27

Click Next.

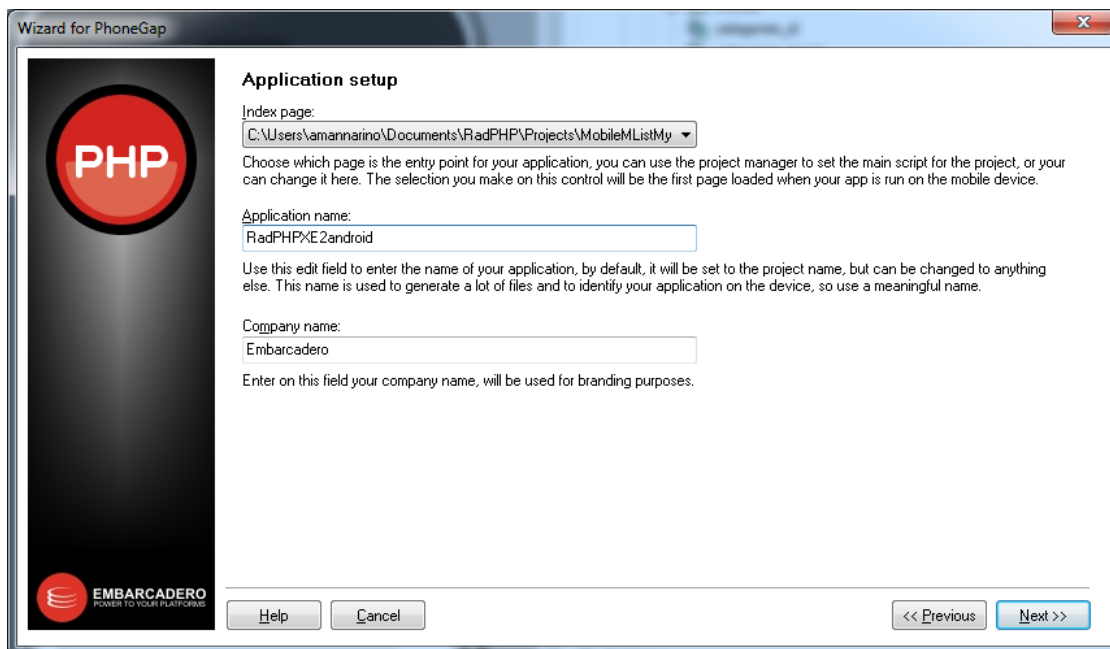


Figure 28

2. On the above "**Application setup**" screen:

The Index page is our **unit1.php** from our current project.

The **Application name**, we will call **RadPHPXE2android**.

Company name; enter your Company name, for this example I'm using **Embarcadero**.

Click Next.

3. You can adjust Android Graphics on this next screen, but for this example, we'll just keep the defaults.

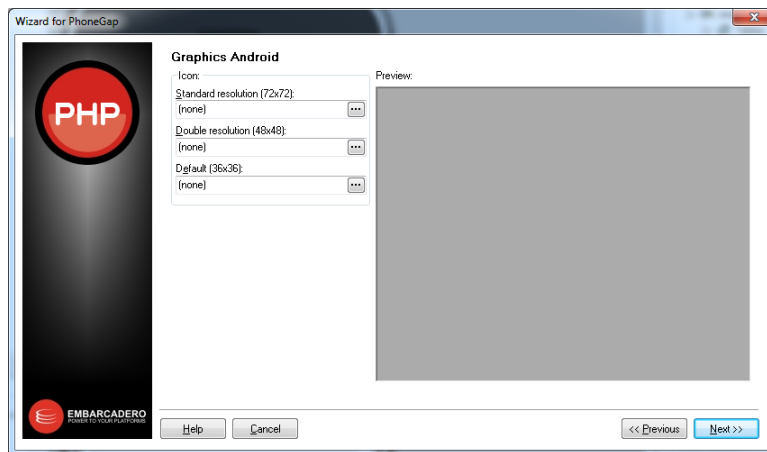


Figure 29

Click Next.

On Target Destination screen, here you can choose either **Debug** or **Release** Compile mode. Select **Debug** mode.

Select a destination folder for the generated Android files, such as:
C:\Users\amannarino\Documents\RadPHPMListInterBaseProject

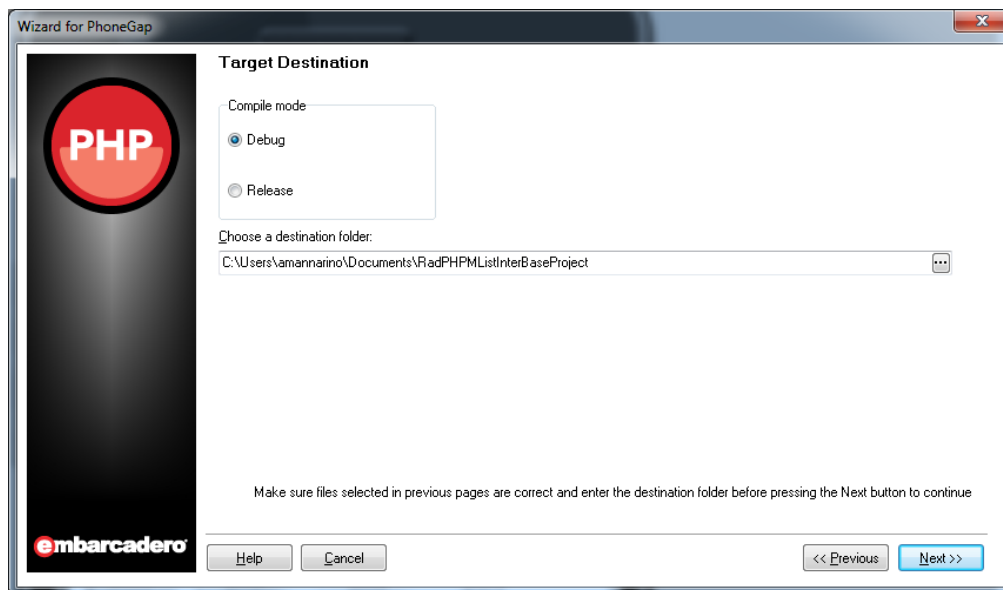


Figure 30

Click Next.

Your Android files get created in your **RadPHPMListInterBaseProject** folder.

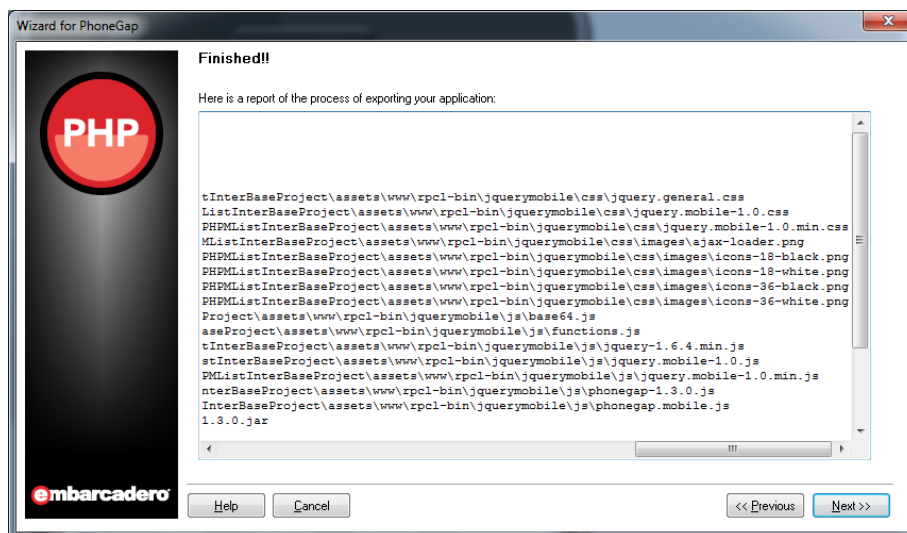


Figure 31

Click Next.

Here you can choose to Run your App on a Real Device, or on an Emulator. Select **Android emulator** to use (from the Part 1 -Install Android-Setup). In Figure 32 we are using Android 2.3.1 emulator.

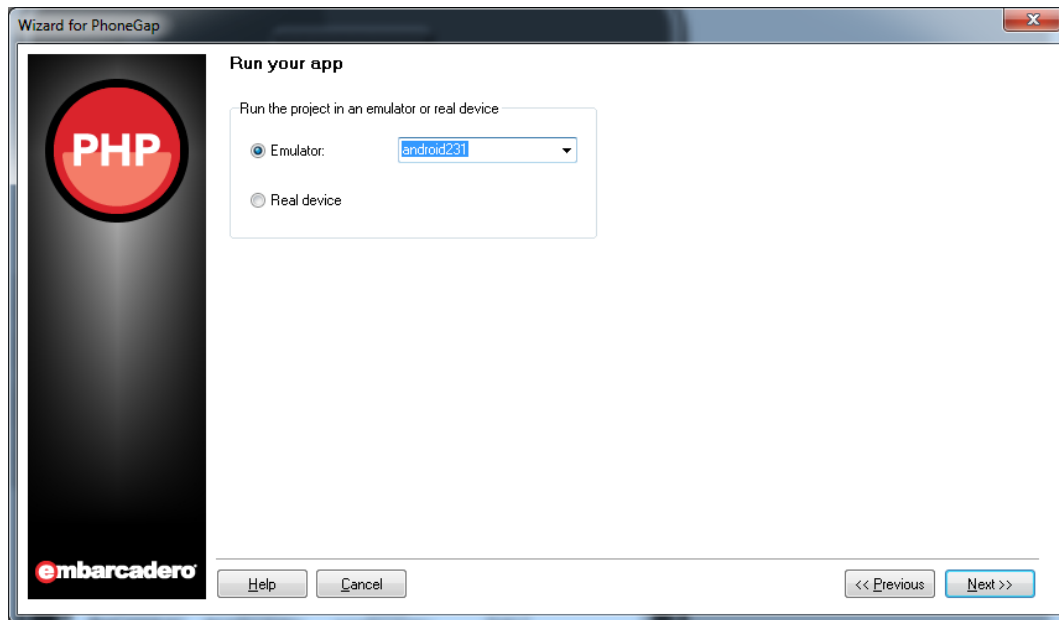


Figure 32

Click Next.

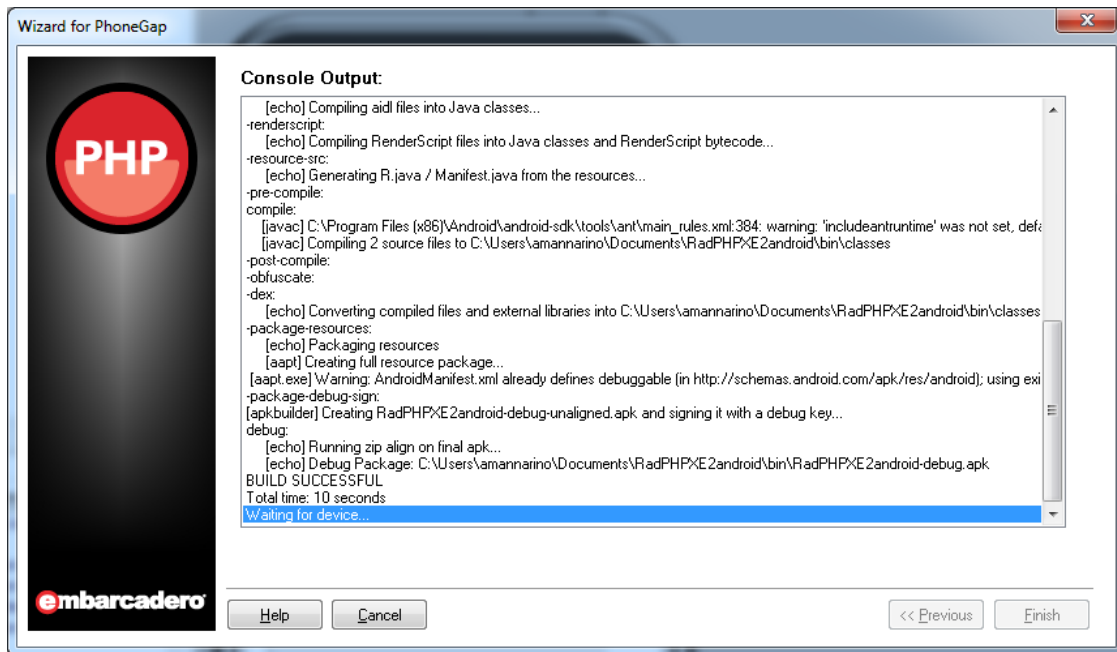


Figure 33

The Android adb server starts, SDK Tools launch, the Android project builds and your final Android .apk file gets created.

NOTE: If you do not have your Android Emulator running, you will get the message "Waiting for device", like you see in the screen above. The Android Setup from Part 1 should automatically start your Android Emulator, load and run the app like this:

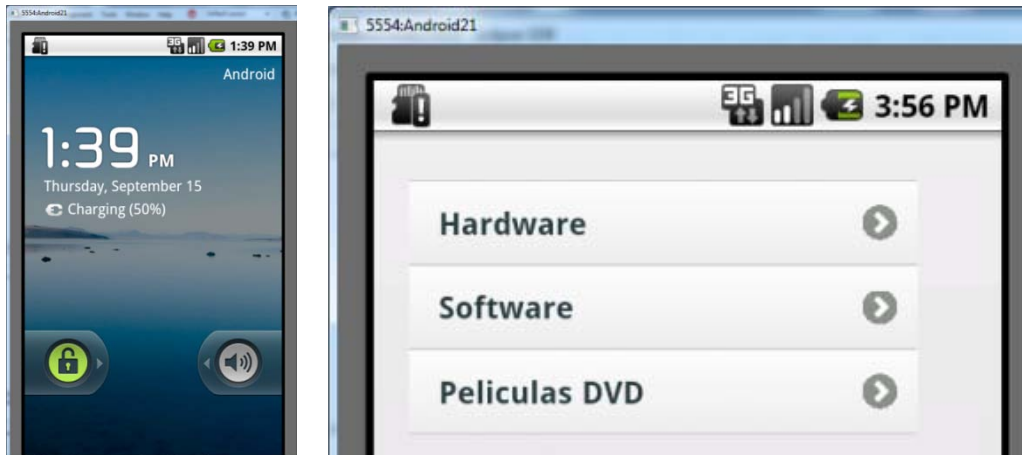


Figure 34

PART 6 - ALTERNATE METHOD TO MANUALLY START YOUR ANDROID EMULATOR

Execute the **android.bat** file from C:\Program Files (x86)\Android\android-sdk\tools.

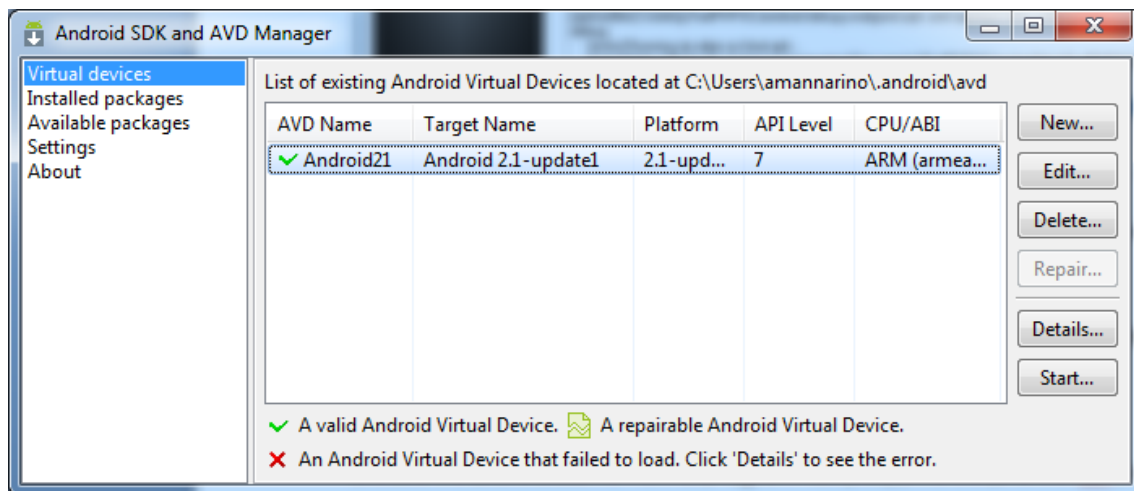


Figure 35

1. Select your Android Emulator, such as the Android 2.1 on this screen.
2. Click Start and Launch

3. With Android device started, the Console Output shows:

```
"Installing RadPHPXE2android-debug.apk..."
Process Completed.  Unlock your device.
BUILD SUCCESSFUL
Total time: 10 seconds
Waiting for device...
Installing RadPHPXE2android-debug.apk...
```

Click Finish.

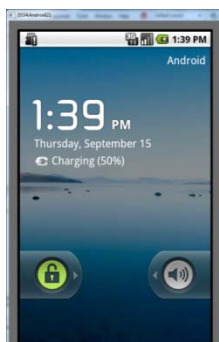


Figure 36

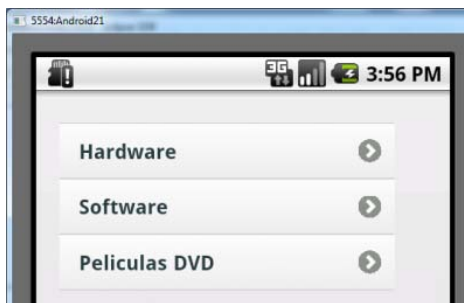


Figure 37

Slide the Lock from Left to Right, to unlock your Android device. Note: Unlocking your emulator (varies by type of emulator you are using).

Your Android application will run on your Android Emulator, like above Figure 36 and Figure 37.

Congratulations!

PART 7- CREATE THE IOS APPLICATION

Using the same project, we'll create the app for an iOS device.

Tools | Wizard for PhoneGap

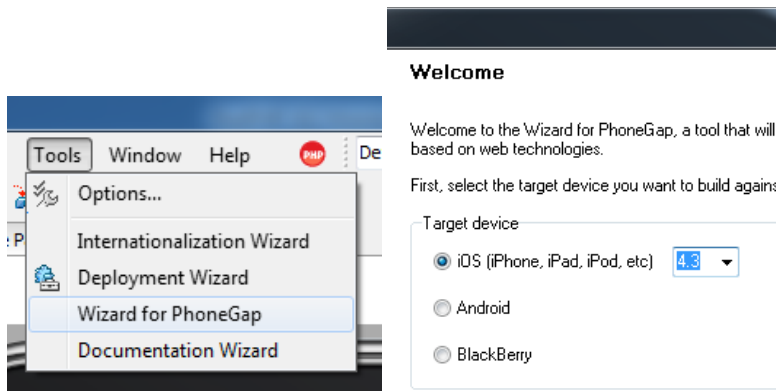


Figure 38

For this example, I'll select iOS version 4.3.

Click Next.

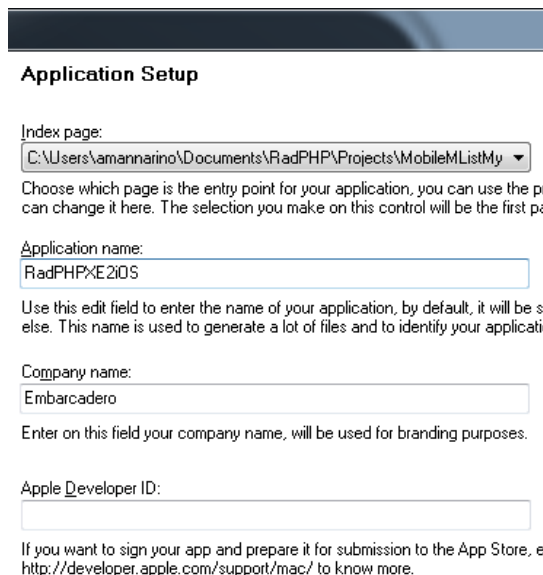


Figure 39

Application Name = RadPHPXE2iOS.

Optionally, type your developer ID in Apple Developer ID field. Note: If you plan to upload your application to the App Store, you will need to sign it, for which you will need to have an Apple Developer ID. You can find more information about the topic at <http://developer.apple.com/support/mac/>.

Click Next.

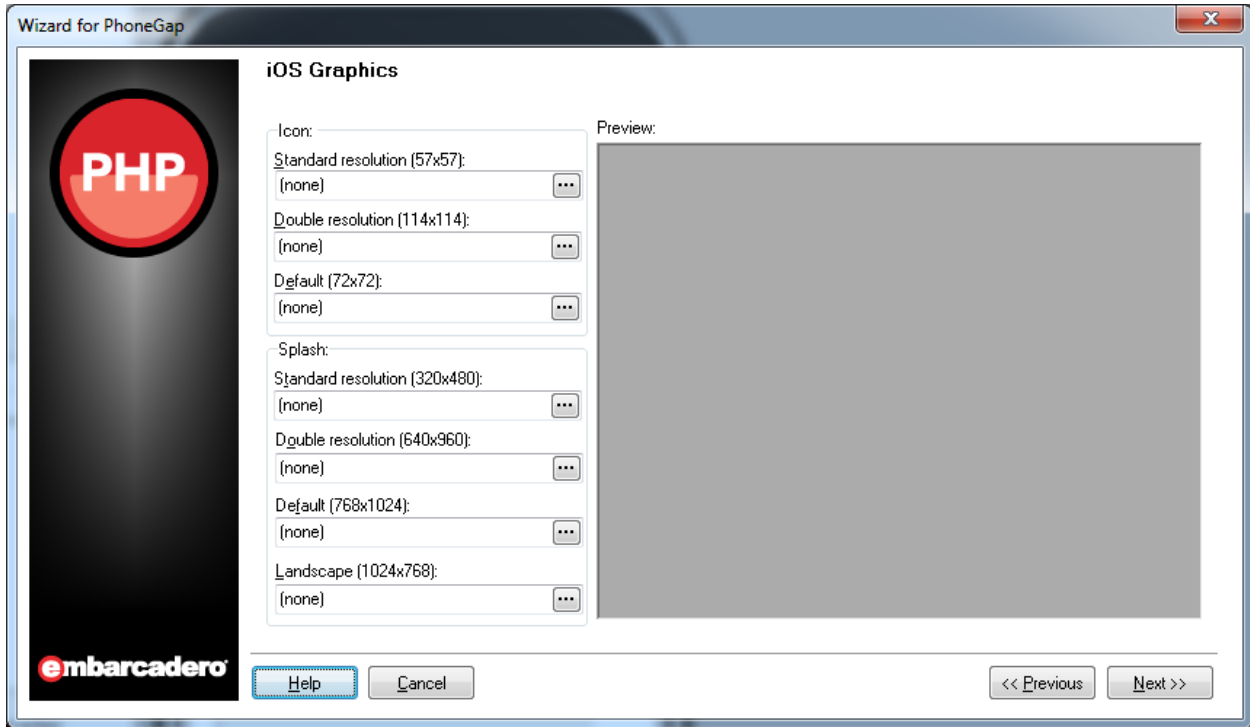


Figure 40

On the **iOS Graphics** page, you can setup icons and splash screens for your application. For this example, we will leave all as their defaults.

Click Next.

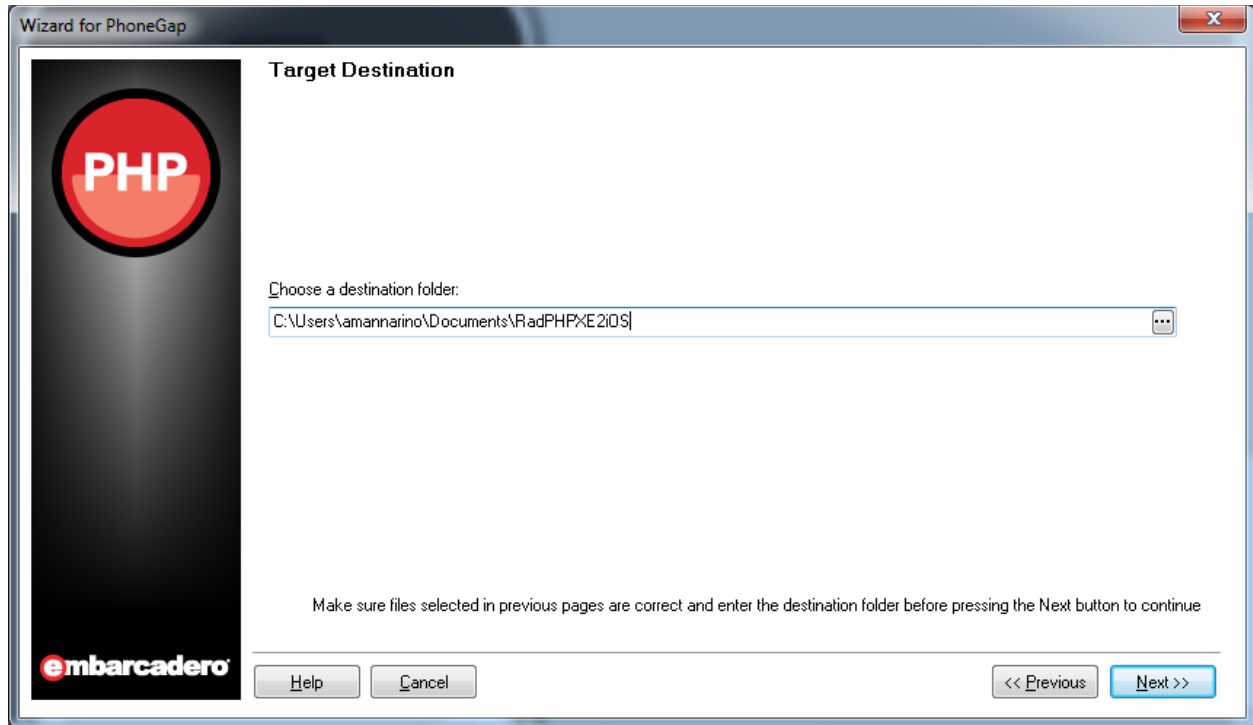


Figure 41

Destination Folder = RadPHPEX2iOS

1. On the **Choose Destination Folder** page:

Note: *The folder you choose will be where all files and folders will be generated. You might want to create a new folder for your build.*

1. Optionally, change destination folder:
 1. Click
 2. Go to folders location, and click **OK**.
2. Click **Next**.

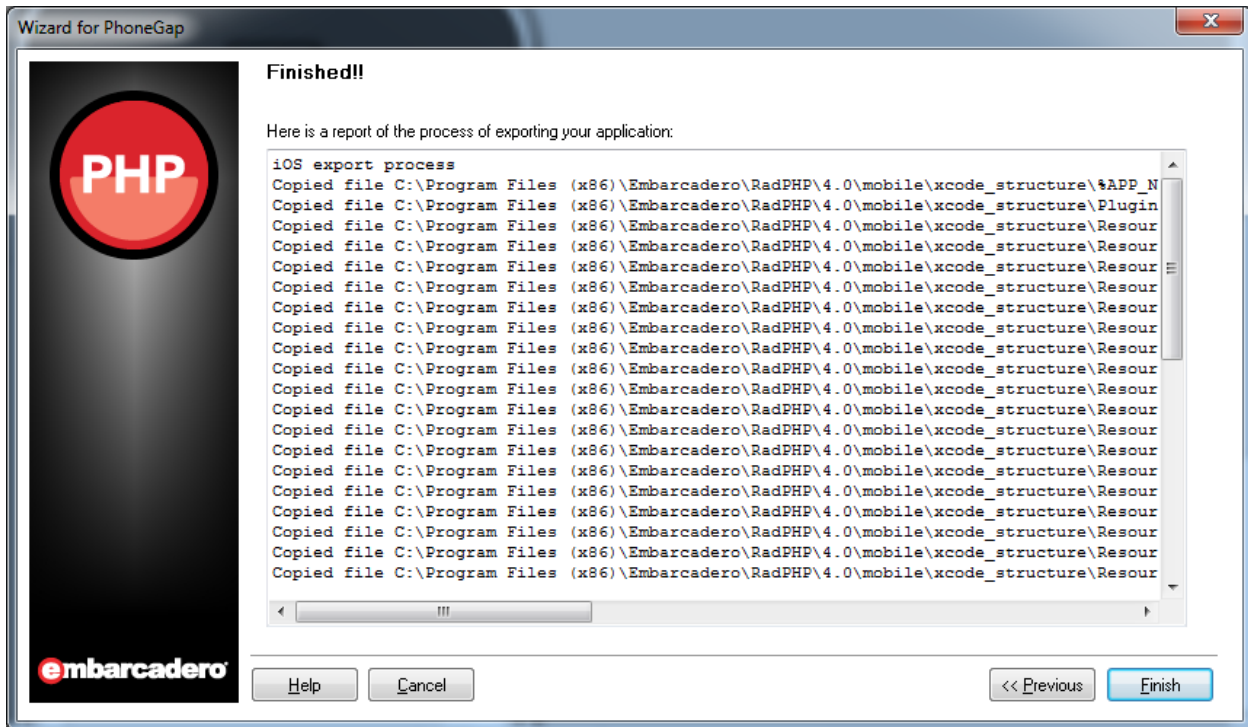


Figure 42

2. On the **Finished!!** page, wait for the build process to end and then click **Finish**.

NOTE: This does not apply to this sample project, but if you used any **external resources**, you must copy any **external resources** you included in your application, like image files, into **www** PhoneGap folder. Check [this video tutorial](#) (around minute 6:25) for details. Then proceed to move your generated PhoneGap folder to your Mac OS X system to [complete the build steps](#).

iOS Setup - Prerequisites

Before you can build the mobile application for iOS, you will need to setup your Mac OS X system.

Xcode and iOS SDK Installation Steps on Mac OS X

It is assumed that you have Xcode and the iOS SDK already installed on your Mac. If you don't, you will need to install both. For more information, visit developer.apple.com/xcode.

PhoneGap Installation Steps on Mac OS X

Install PhoneGap 1.3.0 for iOS:

1. Download PhoneGap from github.com/callback/phonegap/zipball/1.3.0.
2. Inside downloaded folder, find **iOS/PhoneGap-1.3.0.dmg** and open it.
3. Inside the **.dmg** you will find a **PhoneGap-1.3.0.pkg** file. Run it.

PhoneGap installation wizard will then pop up.

1. On the **Introduction** page, you can check PhoneGap change log. Click **Continue**.
2. On the **Read Me** page, you will be shown the content of the ReadMe file. Click **Continue**.
3. On the **Destination Select** page, select who you want to install PhoneGap for, and click **Continue**.
4. On the **Installation Type** page, you can decide the installation folder for PhoneGap.
 1. If you want to customize installation folder, click **Change Install Location...**, select your new installation folder and click **OK**.
 2. Click **Install**.
5. On the **Summary** page, you will be shown PhoneGap installation notes. You can now click **Close**.

PhoneGap 1.3.0 for iOS is now installed in your system.

XCode - Build and Run iOS Xcode project on Mac

Warning: Before you continue, do not forget to [setup your system](#) steps above.

Once you are in your Mac OS X system, open the PhoneGap folder you generated with RadPHP, and inside it you will find RadPHPiOS.xcodeproj. Open it with XCode.

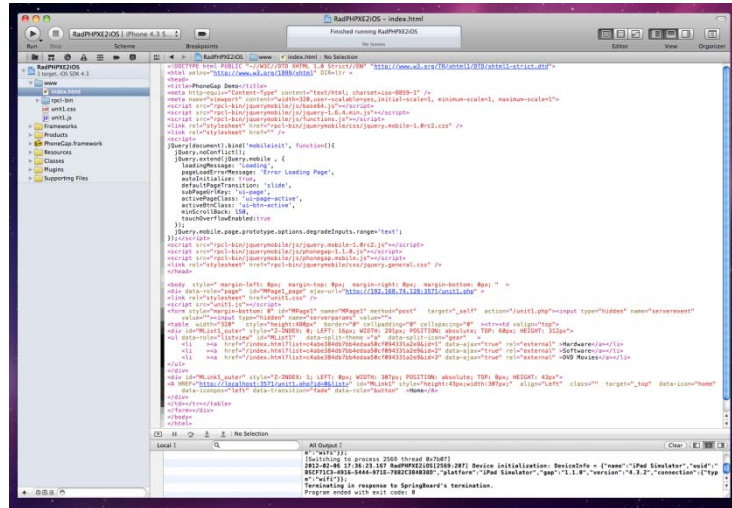


Figure 43

Run

Once you open your project in XCode, follow these steps to run it in a real device or a simulator:

Device drop-down list.

1. Set the device you want to use in the **device drop-down list** (top left). By default you can choose an iPad simulator, an iPhone simulator, or a real iOS device. For our example, we'll use **iPhone 4.3**.
2. Click **Run** button.

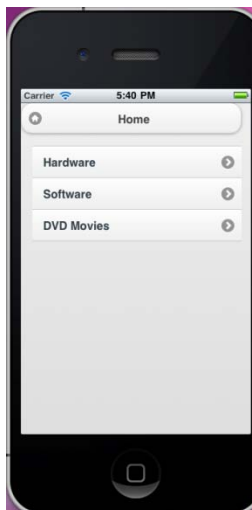


Figure 44

Note: With simulators, you can stop it at any time by clicking the **Stop** button.

iOS App Store Package

To create a package you can upload to the App Store, follow these steps:

1. Choose **iOS Device** in the *device drop-down list*.
2. Go to **Product > Archive**. Once the build is done, you will be taken to the *Organizer*, in the *Archive* tab.
3. Click the **Share** button.
4. On the dialog that will open:
 1. Set **Contents** to *iOS App Store Package (.ipa)*.
 2. Choose your developer **Identity** you want to use for the package.
 3. Click **Next**.
5. Choose a name for your package, a destination directory, and when you are done click **Save**.

You can now upload that package to the App Store for distribution. For that you need an **iOS developer account**, so if you have not one already, get it from developer.apple.com/programs/start/ios/ and follow the instructions you will be given by Apple. **Note:** For additional information on deployment, check the *XCode Manual*: developer.apple.com/library/mac/#documentation/ToolsLanguages/Conceptual/Xcode4/UserGuide/DistApps/DistApps.html

PART 8 - CREATE THE BLACKBERRY APPLICATION

Before you can build your mobile application for **BlackBerry**, you will need to setup the BlackBerry WebWorks SDK and some additional software.

Java SE Development Kit (32-bit)

Install the 32-bit version of the Java SE Development Kit, even if you are on a 64-bit system. In the folder or CD where you first run RadPHP installer, you will find 32-bit Java installer under **Java** directory, by the name of **jdk-6u26-windows-i586.exe**. Start it and follow the steps below:

1. In the **Java SE Development Kit** wizard, click **Next** to start installation setup.
2. On the **Custom Setup** page, you can customize what parts will be installed in your system, and where.
 1. Optionally, you can skip source code installation:
 1. Right-click **Source Code**.
 2. Click *Don't install this feature now*.
 2. Optionally too, you can change installation path for any item:
 1. Select an item, and click **Change**.
 2. Choose a new installation path, and click **OK**.
 3. Click **Next**.
3. On the **Destination Folder** page, you can customize JRE installation page.
 1. Optionally change installation path:
 1. Click **Change**.
 2. Browse to a new path, and click **OK**.
 2. Click **Next**, installation will start.
4. On the last page, click **Finish**.

Note: An Oracle web page might be loaded in your web browser, asking you to register. You do **not** need to register. You can safely close the page.

BlackBerry WebWorks SDK

First, **download** the installer from bdsc.webapps.blackberry.com/html5/download/sdk, where you must choose the **Smartphone SDK** version. Once downloaded, **run** it and follow the steps below:

1. On the **Introduction** page, click **Next**.
2. On the **License Agreement** page, check *I accept the terms of the License Agreement* and click **Next**.

3. On the **Choose Install Folder** page:
 1. Optionally change installation path:
 1. Click **Choose....**
 2. Go to the desired path, and click **OK**.
 2. Click **Next**.
4. On the **Pre-Installation Summary**, check everything is right and click **Install**.

Once installation is complete, click **Done**.

Blackberry Signatures

In order to deploy your application to real devices, it must be signed, so you must have a signing key. If you do not have one already, go to bdsc.webapps.blackberry.com/html5/signingkey and follow the instructions you will find there.

Blackberry Drivers

To test your application in either an emulator or a device, you will need Blackberry drivers. They are included in the Blackberry Desktop Software which you can download from us.blackberry.com/apps-software/desktop/.

Packaging with the BlackBerry WebWorks SDK

In a command window (**Start > Run > cmd**), change directories to the BlackBerry WebWorks SDK installation directory.

Note: Replace `<version #>` with the version number of the SDK you installed (for example, 2.3.0.9).

```
cd "C:\Program Files\Research In Motion\BlackBerry WebWorks SDK  
<version #>"
```

To build an application archive found at "C:\myapp.zip", type the following command:

```
bbwp "C:\myapp.zip"
```

Create the RadPHP XE2 BlackBerry app

To compile your application for **Blackberry**, you must first *export it as a PhoneGap project*:

1. Tools | Wizard for PhoneGap

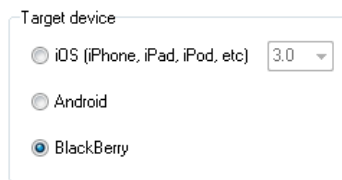


Figure 45

Select **BlackBerry**. Click Next.

Application Setup

Index page:

Choose which page is the entry point for your application, you can use the page can change it here. The selection you make on this control will be the first p

Application name:

Use this edit field to enter the name of your application, by default, it will be else. This name is used to generate a lot of files and to identify your applical

Figure 46

Application Name = **RadPHPXE2BlackBerry**. Click Next.

BlackBerry Graphics

Icon:
 Icon (80x80):
 ...

Hover Icon (80x80):
 ...

Loading Screen (200x200):
 ...

Figure 47

You can keep the defaults for BlackBerry Graphics or setup your own custom icons and splash screens. Click Next.



Figure 48

Target Destination folder = **RadPHPXE2BlackBerry**. Click Next.

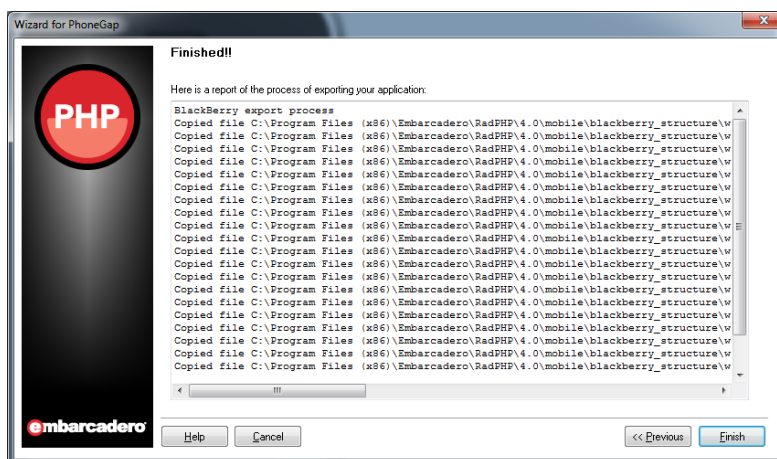


Figure 49

Wait for **Finished!!** to complete. Click Finish.

Now, in your destination folder you will find a **www** folder and a **RadPHPXE2BlackBerry.zip** archive.

If you included in your application any **external resources**, like image files, you must copy them into **www**. Check [this video tutorial](#) (around minute 6:25). Then you have to generate a new **.zip** archive with the contents of the **www** folder. There are [several tools](#) you can use for that task, like 7-zip.

Warning: You must create the archive from the contents of the **www** folder, not from the folder itself.

Build the app using BlackBerry WebWorks SDK

1. Open a terminal window: **Start > cmd**

Warning: In Windows Vista and Windows 7 you must run it **as administrator**.

2. Run `cd SDK_directory`, where **SDK_directory** is:
 - o In 32-bit systems: C:\Program Files\Research In Motion\BlackBerry WebWorks SDK **version**
 - o In 64-bit systems: C:\Program Files (x86)\Research In Motion\BlackBerry WebWorks SDK **version**

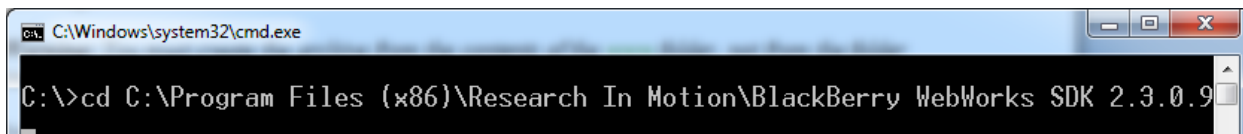


Figure 50

3. Run `bbwp zip_file -o destination_folder -g password`, where:
 - o **zip file** is the path to the **.zip** file with **www** contents inside.
 - o **destination folder** is the folder where you want built application to be moved.
 - o **password** is your password.

*Note: You can omit the **-g password** part if you are only building for testing.*

bbwp

`C:\Users\amannarino\Documents\RadPHPXE2BlackBerry\RadPHPXE2BlackBerry.zip -o C:\Users\amannarino\Documents\BlackBerryBuilds`

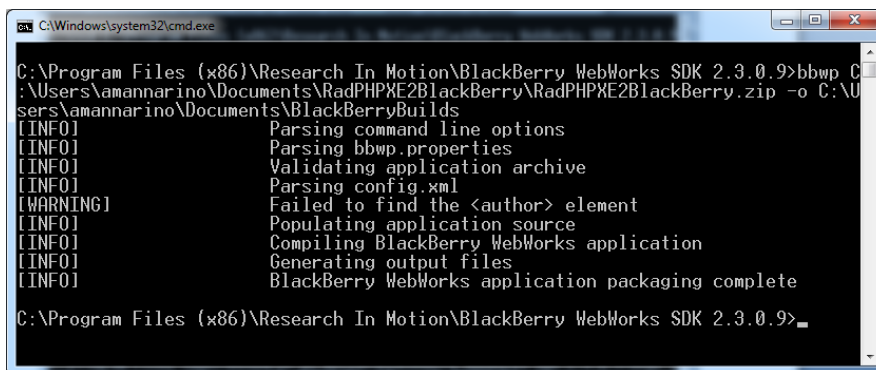


Figure 51

You will find your application in **destination_folder\StandardInstall\zip_filename.cod**. You can now deploy that **.cod** file to the **App World**. To do so, check the steps in BlackBerry Documentation: us.blackberry.com/developers/appworld/.

Name	Type
RadPHPXE2BlackBerry.alx	ALX File
RadPHPXE2BlackBerry.cod	COD File
RadPHPXE2BlackBerry	CodeSite Log File
RadPHPXE2BlackBerry.cso	CSO File

Figure 52

Emulate the app using BlackBerry WebWorks SDK

You will find emulator launcher, fledgelauncher.exe, in the following folder:

- In 32-bit systems: C:\Program Files\Research In Motion\BlackBerry WebWorks SDK **version**\simpack**simversion**
- In 64-bit systems: C:\Program Files (x86)\Research In Motion\BlackBerry WebWorks SDK **version**\simpack**simversion**

C:\Program Files (x86)\Research In Motion\BlackBerry WebWorks SDK
2.3.0.9\simpack\7.0.0.318

Start the launcher, select the emulator you want to launch, and click **Go**.

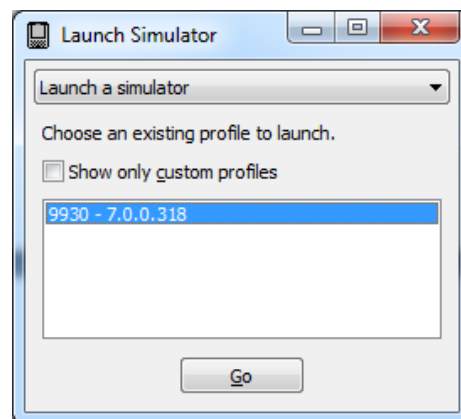


Figure 53

The BlackBerry 9930 Simulator starts.

File | Load BlackBerry Application

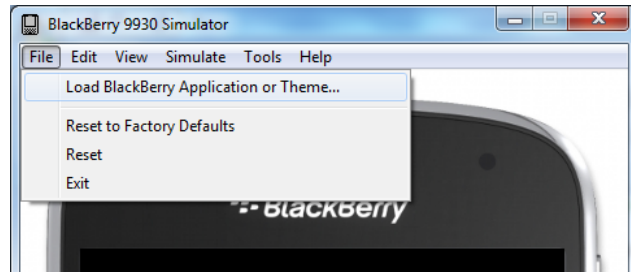


Figure 54

Select your **RadPHPXE2BlackBerry.cod** from
C:\Users\amannarino\Documents\BlackBerryBuilds\StandardInstall.

Select "**All**" on the BlackBerry simulator, and locate your RadPHP BlackBerry App
(Red Circle with PHP)

Click the **PHP** app. The BlackBerry app will access your PHP Web Server running
in RadPHP XE2 and connect to the InterBase database.



Figure 55

Congratulations!

CONCLUDING REMARKS

Embarcadero® RadPHP™ XE2 provides the fastest way to build Web, Facebook®, and mobile applications with the only visual PHP framework and IDE that supports Web and mobile deployment. RadPHP XE2 offers over 200 drag-and-drop components for building UIs, displaying and interacting with data and services such as Google maps and Facebook, along with integrated coding, debugging, performance profiling, and database connectivity. To learn

more about RadPHP XE2, or to download a trial, visit
www.embarcadero.com/radphp

APPENDIX A - HOW TO CREATE THE INTERBASE OSCOMMERCE DATABASE

PART 1 - INTERBASE XE DEVELOPER EDITION INSTALLATION

Download your platform specific version of InterBase XE from:

<https://downloads.embarcadero.com/free/interbase>

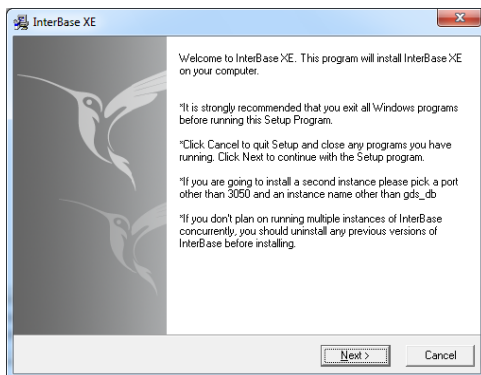
Select either the 32-bit or 64-bit Developer Edition depending on your specific OS platform:

InterBase XE **64-bit Developer** Edition for Windows

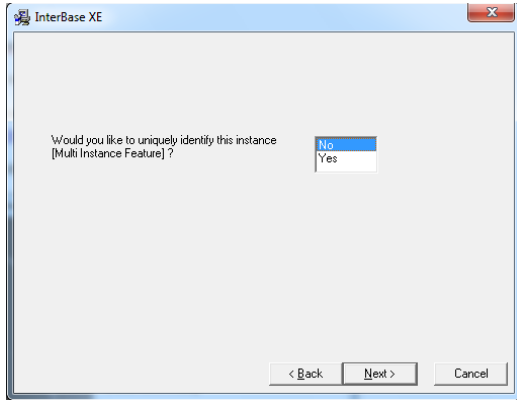
InterBase XE **32-bit Developer** Edition for Windows

Note: The download may email you the **Serial Number** to **Register** your product or you can use the serial number from your RadPHP installation `install.htm` file located at `C:\Program Files (x86)\Embarcadero\RadPHP\4.0\Welcomepage\install.htm`.

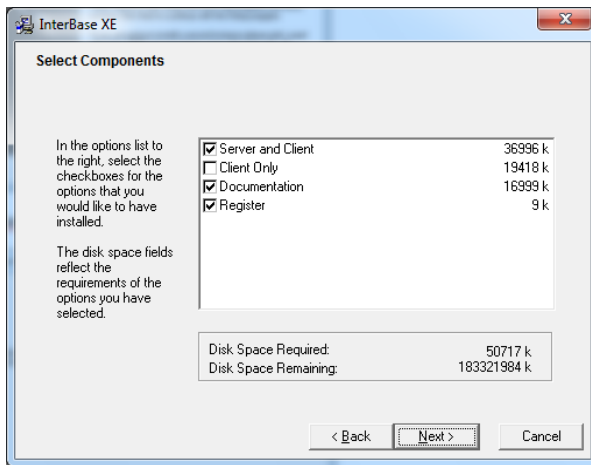
Run the `ib_install.exe` from the downloaded `InterBase_XE_Win32.zip` file.



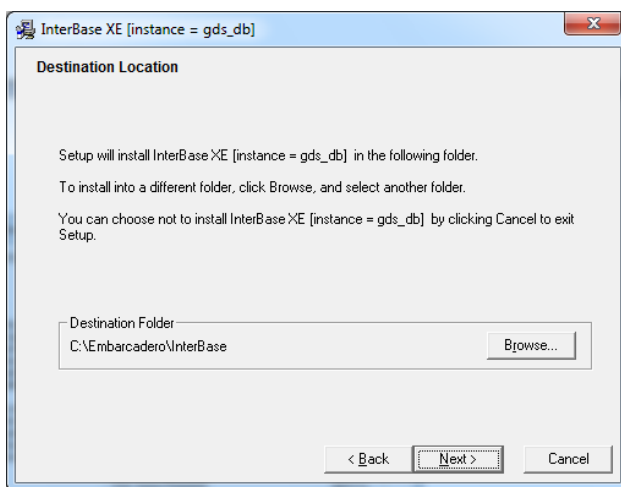
InterBase XE installs on port 3050 with an instance name of `gds_db`



Assuming you don't have any other versions or instances of InterBase running on your machine, select No for "Would you like to uniquely identify this instance?".



Select "Server and Client", Documentation and Register

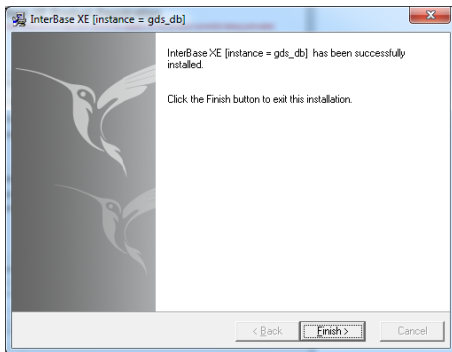


Click Next



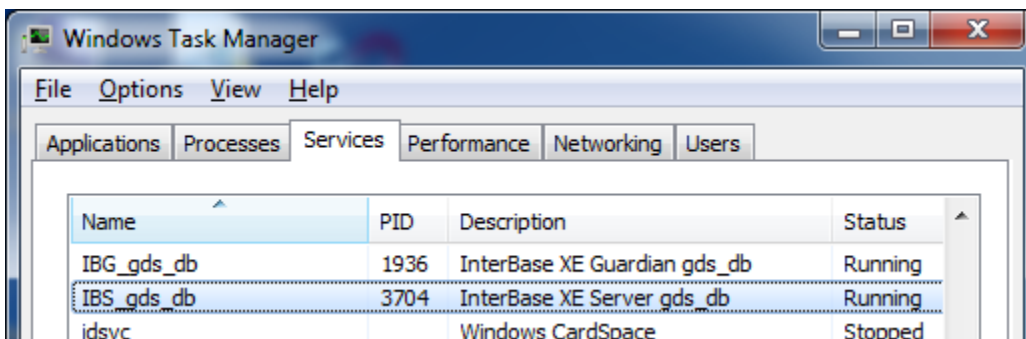
Enter valid Serial Number and your DN login and password.

Click **Register**

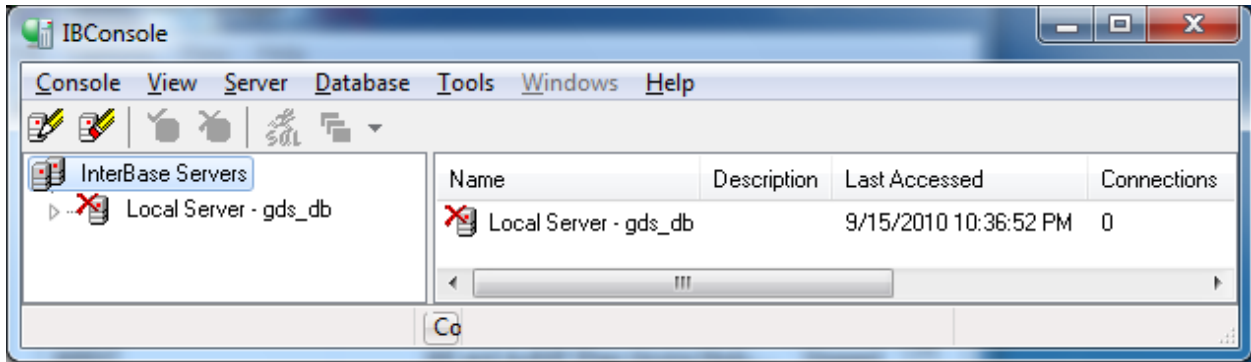


Click Finish.

1. Verify the two InterBase Services are running (InterBase XE Guardian gds_db and InterBase XE Server gds_db).



2. Run the IBConsole (from C:\Embarcadero\InterBase\bin\IBConsole.exe)



The IBConsole is an all-in-one database tool. It combines database administration, interactive SQL and communications testing capabilities in one easy to use application.

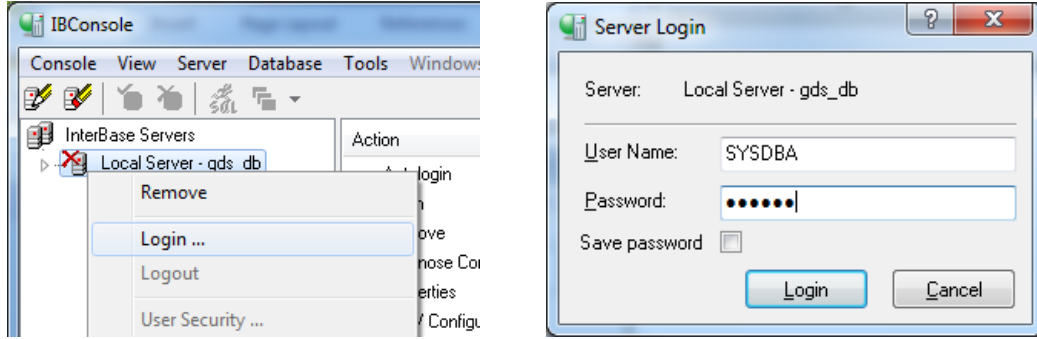
From this IBConsole, you can:

- Manage local and remote servers.
- Manage server security. Authorize new users, change user passwords, and remove users.
- Manage server certificates (licenses).
- Manage databases. Create new databases and set database properties.
- Backup and restore databases
- Perform database maintenance. Validate the integrity of a database, repair a corrupted database, sweep a database, and recover "limbo" transactions.
- Shut down and restart a database
- Execute SQL
- View database metadata in DDL script format

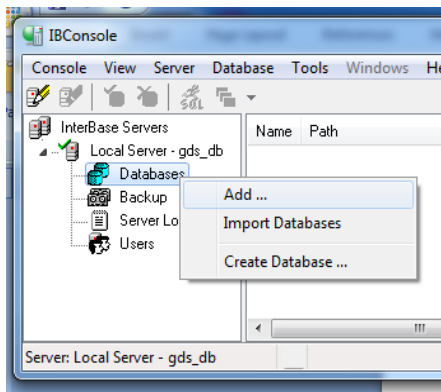
IBConsole consists of:

- A Main Window, the control center of IBConsole
- An Object View Window, which displays detailed information and database objects
- SQL Window, which can be used to execute SQL statements and scripts.
- A Text Viewer, used to display metadata, logfiles, etc.
- Several Visual Editors, to create and alter database object definitions

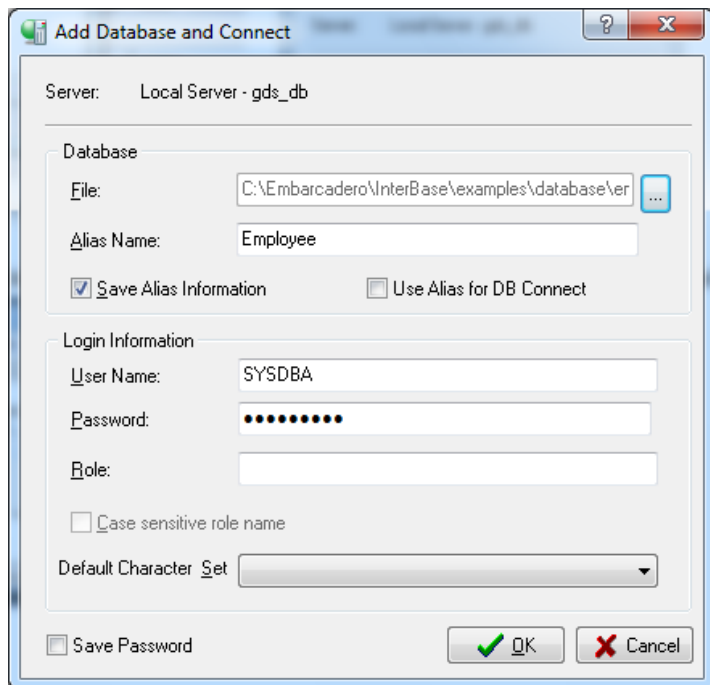
3. Right-click on Local Server – gds_db >> Login (User Name = SYSDBA , Password = masterkey)



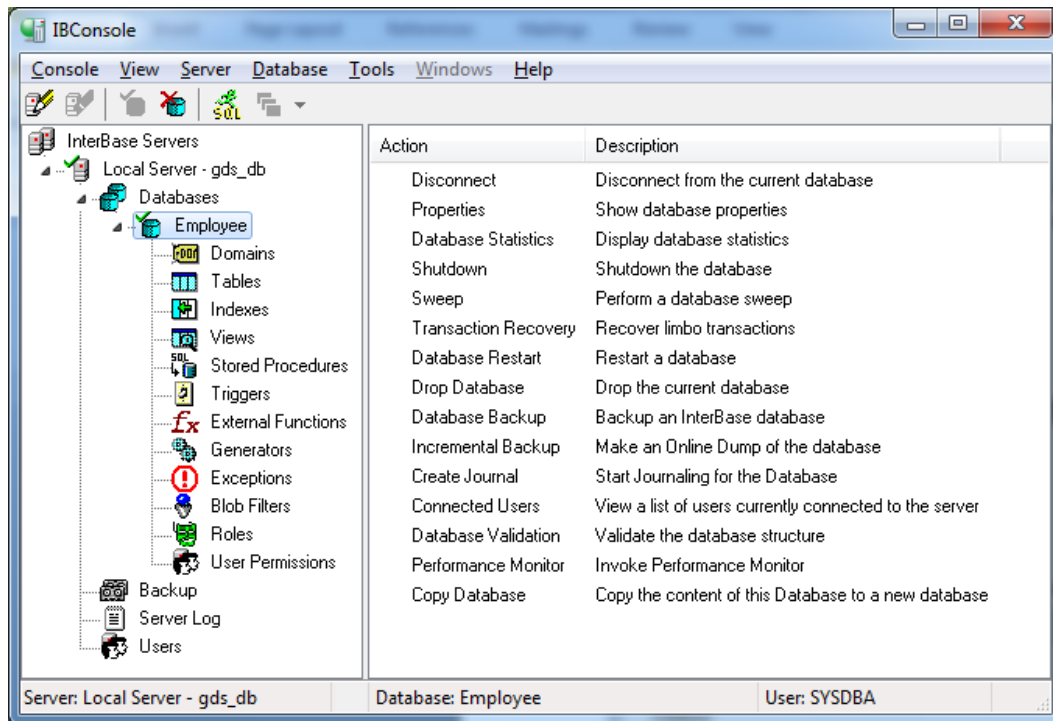
- 4. Right-click Databases >> Add >>
C:\Embarcadero\InterBase\examples\database\employee.gdb
- 5.



Login Information (User Name = SYSDBA , Password = masterkey)



Click OK.

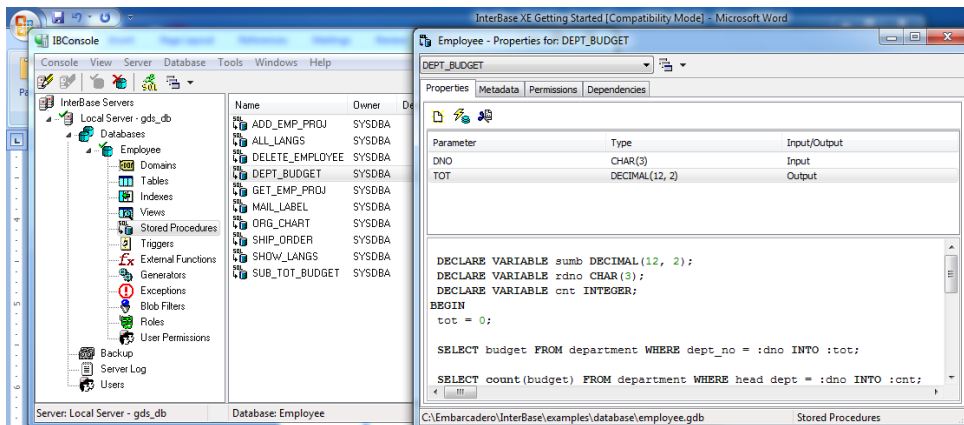


From the IBConsole, you have access to all your database objects (such as Domains, Tables, Indexes, Views, Stored Procedures, Triggers, External Functions, Generators, Exceptions, Blob Filters, Roles, and User Permissions).

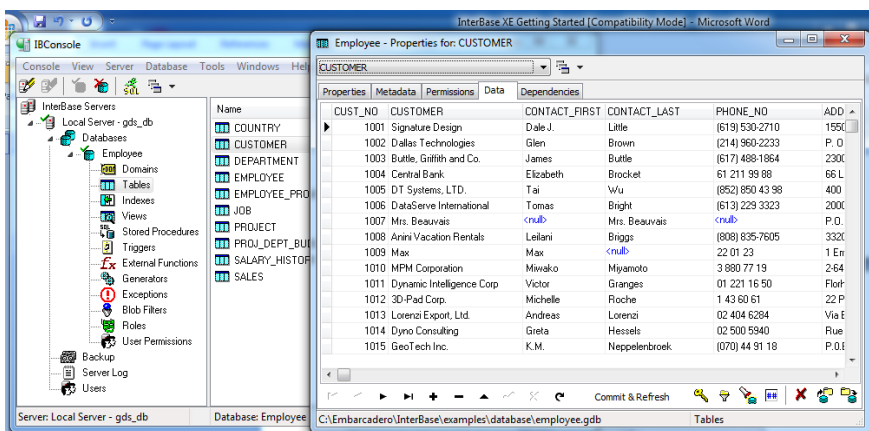
For example, to see all the Stored Procedures for the Employee table, select Stored Procedures.

Right-click on DEPT_BUDGET gives you your options.

Double-click the DEPT_BUDGET procedure to see its properties.



To see the data in a table, select Tables >> double-click Customer >> Data tab



This was a quick overview on how to install InterBase and how to use the IBConsole.

InterBase also includes the Wise Installer to make it easy to embed InterBase with your application, to run the installer in silent mode.

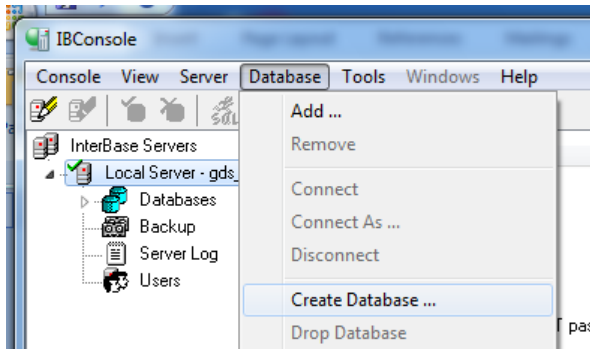
Now that you are familiar with the IBConsole, we'll create the new osCommerce database, tables, indexes and populate the tables with data.

PART 2 - SETTING UP INTERBASE DATABASE

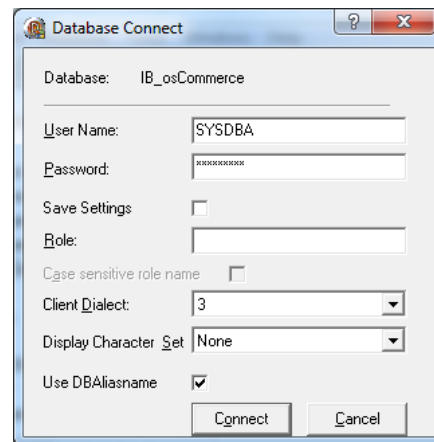
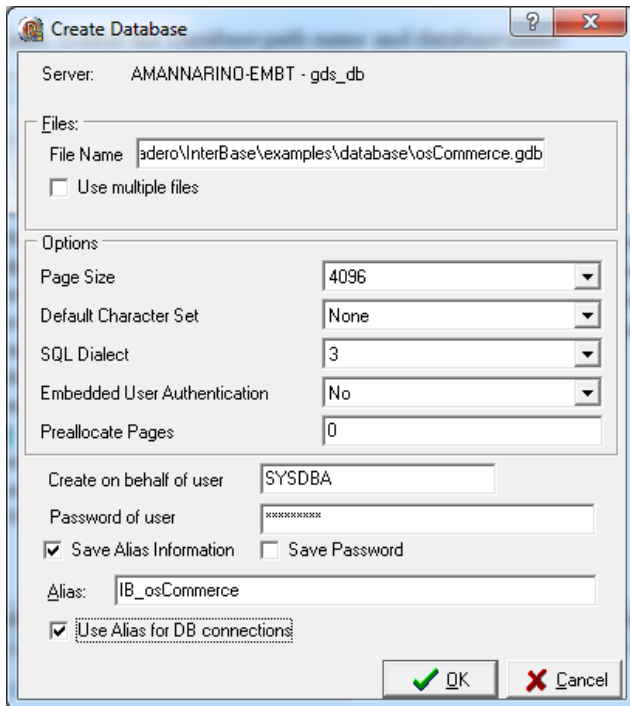
2.1 Create the database

1. First, we'll create the osCommerce database. This is the file structure within which InterBase will subsequently store both the table structure, indexes etc. (this is called the Metadata), and the data itself (called the Data).
2. To create the database:

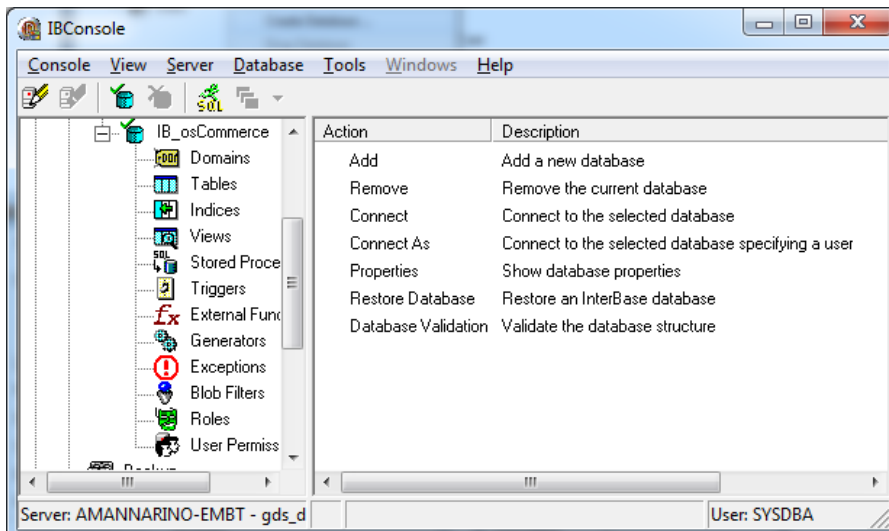
Open IBConsole
Click Database >> Create Database...



1. In File Name, Enter the Database path name and database name:
C:\Embarcadero\InterBase\examples\database\osCommerce.gdb



2. Enter user name (SYSDBA) and password (masterkey). Alias = IB_osCommerce. Check "Use Alias for DB connection". Keep all other default options.
3. Click OK.
4. Database Connect User Name (SYSDBA) and Password (masterkey).
5. Click "Connect".
6. Your new **osCommerce** database gets created.



2.2 Create the tables

1. Next, we'll create the table structures. This can be achieved in several ways. The easiest way is to take a simple text file, fill in the table structure, and "import" the structure to InterBase using IBConsole. For this database we'll use this Create_osCommerce_Tables.sql file.

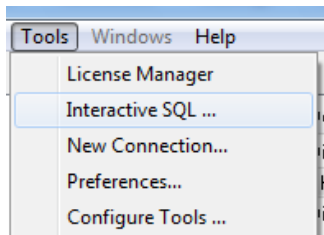
```
/* Create_osCommerce_Tables.sql */
/* Create the InterBase tables for the OSCOMMERCE database*/
```

```
/* Table: CATEGORIES, Owner: SYSDBA */
CREATE TABLE CATEGORIES ( CATEGORIES_ID INTEGER NOT NULL,
PARENT_ID INTEGER NOT NULL );
```

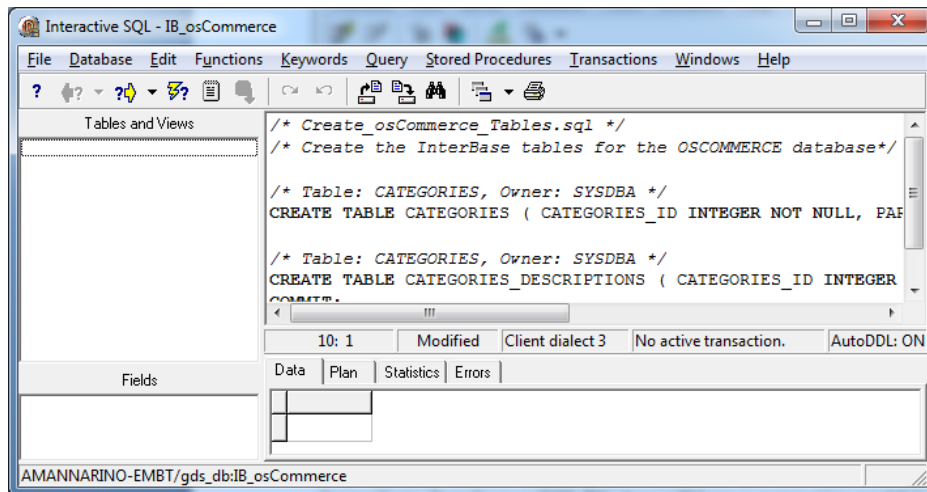
```
/* Table: CATEGORIES, Owner: SYSDBA */
CREATE TABLE CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID INTEGER NOT
NULL, LANGUAGE_ID INTEGER NOT NULL, CATEGORIES_NAME
VARCHAR(32) NOT NULL );
COMMIT;
```

2. Assure you are logged into your osCommerce database, and it's selected on IBConsole.

We'll use **Tools >> Interactive SQL** from the menu bar of the Main Window of the IBConsole to create a new SQL Window.



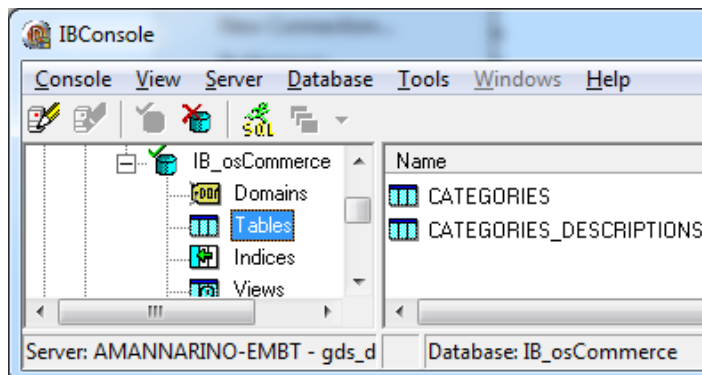
If the currently selected branch in the connections tree is a connected database or one of its descendants, then the new SQL Window will have a connection established to that database. If you have an external SQL file to load, then use would use **Query >> Load Script**. For this example, we'll copy and paste the above **Create_osCommerce_Tables.sql** code into the **SQL Window** like this:



3. **Query >> Execute** (or F5 key)

4. Select **Tables** from your osCommerce database should show you your created databases.

(Note: If you do not see your tables, try disconnect and reconnect to the osCommerce database; right-click Disconnect and Connect).



2.3 Create the indexes

Next we'll create some Indexes. Indexes allow InterBase to locate data (dramatically) more quickly. An index has two key components to it - the field(s) that you will want to search on and whether the field(s) are unique (e.g. a Reference number will probably need to be unique, but you may well need to accommodate several people sharing a birth date or a last name).

2.3.1 Primary key index

Next, we'll create the tables Primary key indexes. The Primary key index is usually needed on the Field(s) to uniquely identify a record within a table (e.g. the unique reference number given to each record, or a Social Security ID, or a post code and House number/name combination within an Address table). For example the CATEGORIES table has PRIMARY KEY (CATEGORIES_ID) ;

For the indexes we'll use this **Create_osCommerce_Indexes.sql** file:

```
/* Create_OSCOMMERCE_Indexes.sql */
/* Create the Indexes for InterBase tables for the
OSCOMMERCEdatabase*/

CREATE UNIQUE INDEX CATEGORIES_ID ON CATEGORIES (CATEGORIES_ID)
;
CREATE INDEX LANGUAGE_ID ON CATEGORIES_DESCRIPTIONS
(LANGUAGE_ID) ;

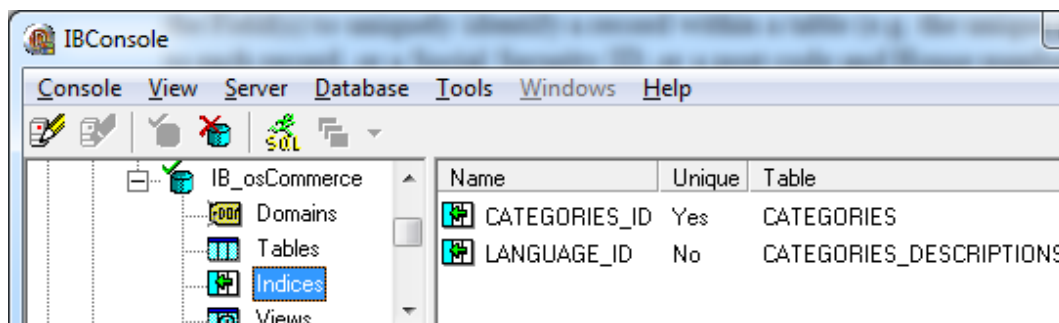
COMMIT;
```

Tools >> Interactive SQL

Copy and Paste the above "**Create_osCommerce_Indexes.sql**" into the SQL Window.

Query >> Execute (or F5 key)

Select Indexes from osCommerce, and verify your Indexes were created:



2.4 Populate the tables

For far we created the database, tables and indexes. Next we want to fill up the database with test data. This can be achieved through the "old-fashioned"

technique of manually entering data into the database (such as through a Client application which allows data entry).

A more robust technique is to create a series of SQL commands that insert data to the table within a simple text file, and import the SQL file to InterBase.

The advantages of this approach include the ability (a) to copy, paste and update lines to achieve a methodical selection of all types of data more easily, (b) to re-enter the data whenever you choose to clear all the data from the database and start again and (c) to reuse relevant test data within new database applications in future.

For the data we'll use this **Insert_osCommerce_data.sql** file:

```
/* Insert_osCommerce_data.sql */  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (1,0);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (2,0);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (3,0);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (5,1);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (6,1);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (7,1);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (8,1);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (16,1);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (17,1);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (18,2);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (19,2);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (20,2);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (10,3);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (11,3);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (12,3);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (13,3);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (14,3);  
INSERT INTO CATEGORIES (CATEGORIES_ID, PARENT_ID) VALUES (15,3);
```

COMMIT;

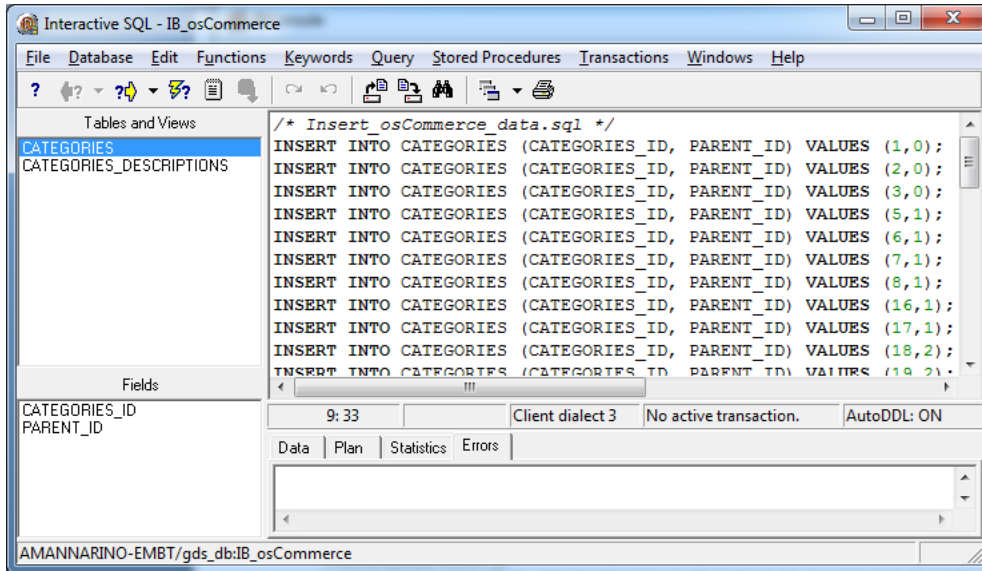
```
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,  
CATEGORIES_NAME ) VALUES (1,1,'Hardware');  
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,  
CATEGORIES_NAME ) VALUES (2,1,'Software');  
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,  
CATEGORIES_NAME ) VALUES (3,1,'DVD Movies');  
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,  
CATEGORIES_NAME ) VALUES (4,1,'Graphics Cards');
```

```
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (5,1,'Printers');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (6,1,'Monitors');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (7,1,'Speakers');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (8,1,'Keyboards');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (9,1,'Mice');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (10,1,'Action');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (11,1,'Science Fiction');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (12,1,'Comedy');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (13,1,'Cartoons');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (14,1,'Thriller');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (15,1,'Drama');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (16,1,'Memory');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (17,1,'CDROM Drives');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (18,1,'Simulation');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (19,1,'Action');
INSERT INTO CATEGORIES_DESCRIPTIONS ( CATEGORIES_ID, LANGUAGE_ID,
CATEGORIES_NAME ) VALUES (20,1,'Strategy');
COMMIT;
```

To populate the database using this method:

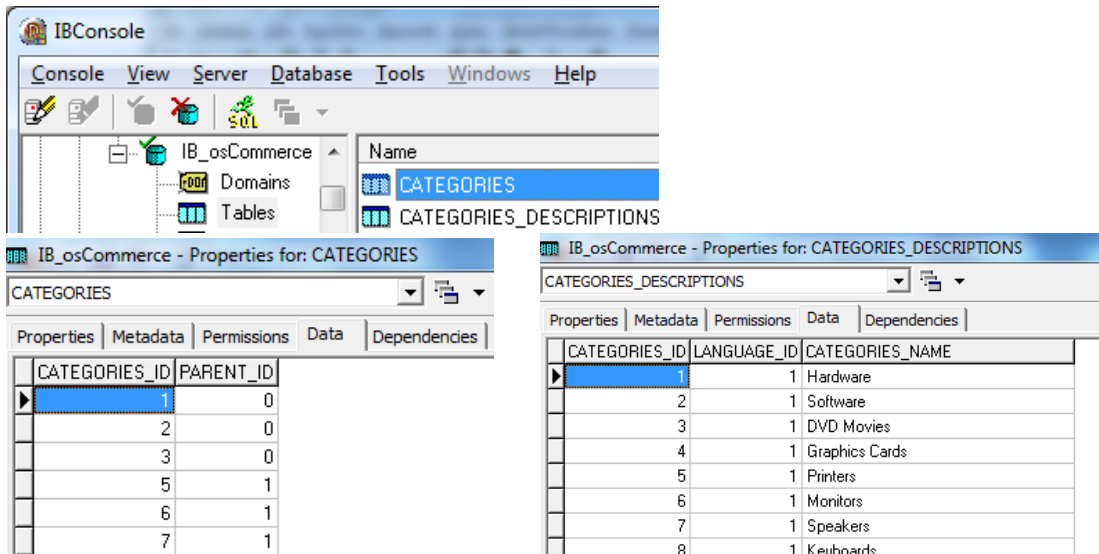
Tools >> Interactive SQL

Copy and Paste the "**Insert_osCommerce_data.sql**" file



Query >> Execute (or F5 key)

Select **Tables** >> double-click **CATEGORIES** >> **Data** tab from osCommerce, and verify your data was inserted:



5. Close out the Data tab.
 6. Console | Exit out of the IBConsole.
- Congratulations! Your InterBase osCommerce database has been created and populated with data!



Embarcadero Technologies, Inc. is the leading provider of software tools that empower application developers and data management professionals to design, build, and run applications and databases more efficiently in heterogeneous IT environments. Over 90 of the Fortune 100 and an active community of more than three million users worldwide rely on Embarcadero's award-winning products to optimize costs, streamline compliance, and accelerate development and innovation. Founded in 1993, Embarcadero is headquartered in San Francisco with offices located around the world. Embarcadero is online at www.embarcadero.com.