

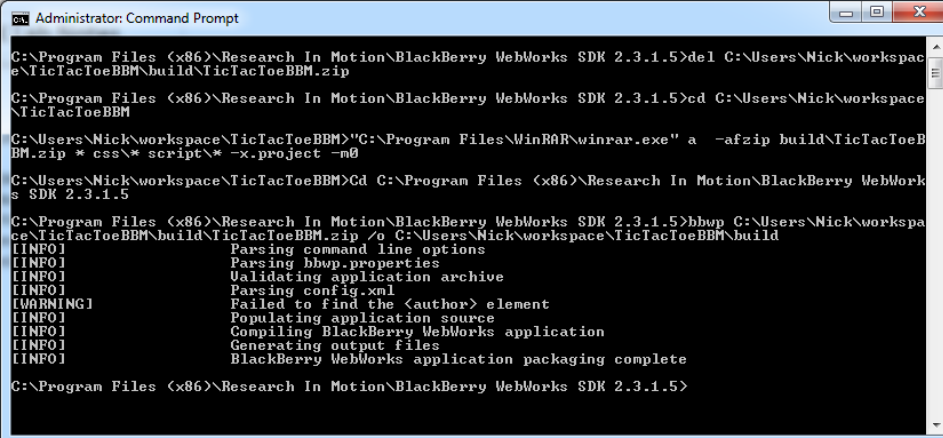
## Developing Socially Connected Apps with the BlackBerry Messenger API

### Lab # 2: Creating a BBM Enabled Application with WebWorks

The objective of this lab is to walk you through the steps required to convert any existing BlackBerry WebWorks application to a BBM enabled application using the BlackBerry Messenger (BBM) SDK. For this lab, we will be converting a Tic Tac Toe game into a game which posts the players progress on their BBM profile.

Step 1: Take a look at what we have so far

- In Lab 1, you set up your development environment in a way that you can develop BBM enabled applications. Let's go through some steps to setup the application to use the API.
- You can choose to develop the lab within Eclipse or any other programming environment of your choice. The screenshots included in the lab will be from the Eclipse development environment.
- Before we make any changes to the application, let's load the application into the Simulator to see how it looks and works beforehand.
- First, compile the application. In order to compile the application, you will need to zip all the files in your project. After zipping, the 'bbwp' file must be used in order to convert all project files into a .cod file for loading into the simulator.
- If you have WinRAR installed, zipping and compiling the application can be done with the following commands. Note that the build folder must be created in your project directory before the commands can be executed.
  - cd C:\Users\Administrator\workspace\TicTacToeBBM
  - "C:\Program Files\WinRAR\winrar.exe" a -afzip build\TicTacToeBBM.zip \* css\\* script\\* -x.project -m0
  - Cd C:\Program Files (x86)\Research In Motion\BlackBerry WebWorks SDK 2.3.1.5
  - bbwp C:\Users\Administrator\workspace\TicTacToeBBM\build\TicTacToeBBM.zip /o C:\Users\Administrator\workspace\TicTacToeBBM\build
- The result of the command line window should look close to the following:



```

Administrator: Command Prompt

C:\Program Files (x86)\Research In Motion\BlackBerry WebWorks SDK 2.3.1.5>del C:\Users\Nick\workspace\TicTacToeBBM\build\TicTacToeBBM.zip

C:\Program Files (x86)\Research In Motion\BlackBerry WebWorks SDK 2.3.1.5>cd C:\Users\Nick\workspace\TicTacToeBBM

C:\Users\Nick\workspace\TicTacToeBBM>"C:\Program Files\WinRAR\winrar.exe" a -afzip build\TicTacToeBBM.zip * css\* script\* -x.project -m0

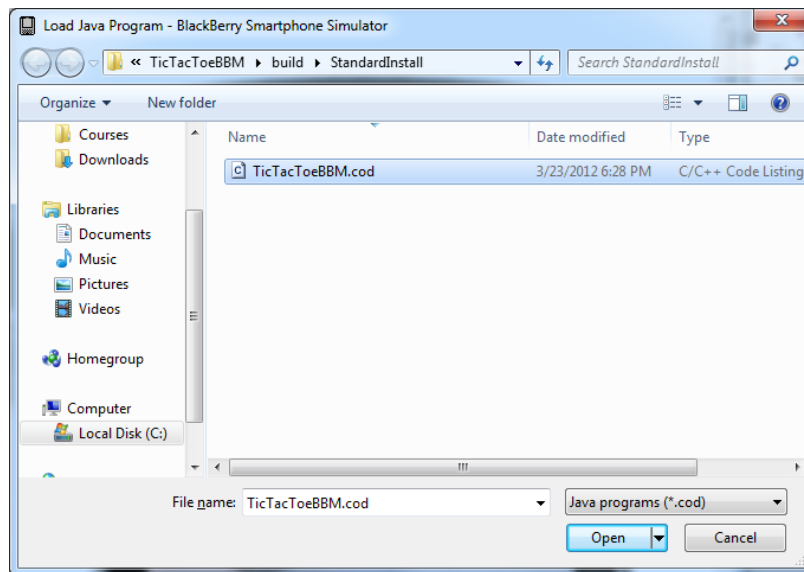
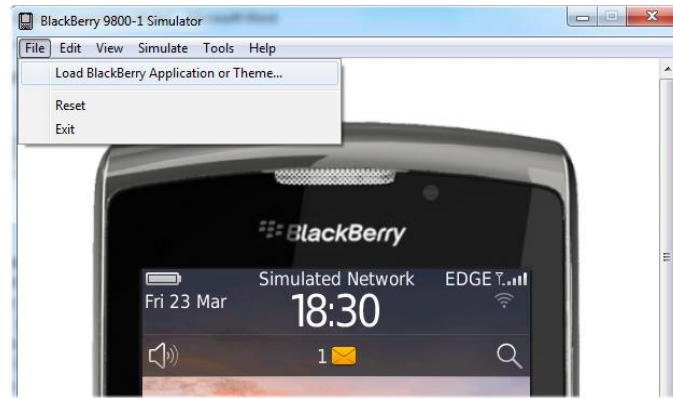
C:\Users\Nick\workspace\TicTacToeBBM>Cd C:\Program Files (x86)\Research In Motion\BlackBerry WebWorks SDK 2.3.1.5

C:\Program Files (x86)\Research In Motion\BlackBerry WebWorks SDK 2.3.1.5>bbwp C:\Users\Nick\workspace\TicTacToeBBM\build\TicTacToeBBM.zip /o C:\Users\Nick\workspace\TicTacToeBBM\build
[INFO]          Parsing command line options
[INFO]          Parsing bbwp.properties
[INFO]          Validating application archive
[INFO]          Parsing config.xml
[WARNING]       Failed to find the <author> element
[INFO]          Populating application source
[INFO]          Compiling BlackBerry WebWorks application
[INFO]          Generating output files
[INFO]          BlackBerry WebWorks application packaging complete

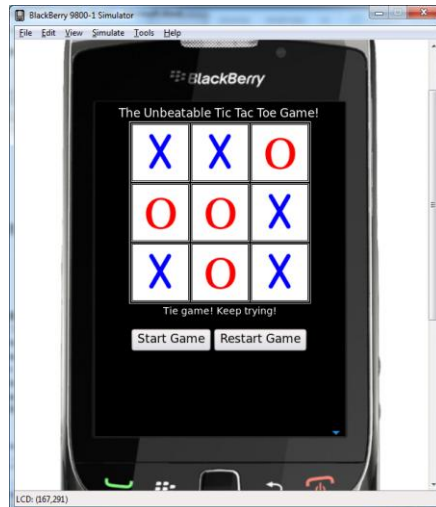
C:\Program Files (x86)\Research In Motion\BlackBerry WebWorks SDK 2.3.1.5>
  
```

- The result of these commands will place a BBMTicTacToe.cod file in the build directory where your project files are located. Let's see how the application looks on the simulator.

- Open the simulator by navigating to the folder where you installed the BBM enabled simulators (default is C:\Program Files (x86)\Research In Motion\BlackBerry Smartphone Simulators 6.0.0\6.0.0.313 (9800) for the BlackBerry 9800 simulator) and clicking on either sim1.bat or sim2.bat
- Once loaded, we can load the .cod file on the simulator by clicking on File->Load BlackBerry Application or Theme.



- The application will now be installed on the BlackBerry device. You can now open the application by clicking on the icon, press 'Start Game' and attempt to beat the BlackBerry Device at Tic Tac Toe (you will not be able to win 😊).



- After you've finished experimenting with the application, it's time to begin integrating this fantastic game with BBM.

## Step 2: Integrating the Application with BBM

- The first thing that will need to be done is changing the HTML code to include a field where BBM information can be shown. In the 'content-frame' div, underneath the 'title' div, add a div which shows BBM information with the text 'Connecting the BBM...' as the default text. Include the class 'content' so that the appropriate formatting will be done by the CSS file included in the lab.

```
<div id="start" class="content">
  <center>
    <h5>Connecting to BBM...</h5>
  </center>
</div>
```

- This will be the only change that needs to be made in the HTML file, all other changes in the lab will need to be done to the JavaScript file.
- In the tictactoe.js file, let's start by adding global variables which will be used for the lifetime of the application. Add the following variables on the top of the JavaScript file.

```
var triesCounter;
var userWins;
var userLoses;
var userTies;
```

- triesCounter is used to count how many times the user played the game, userWins counts the number of times the user won, userLoses counts how many times the user lost the game and userTies counts how many times the user had a tied game. These variables will be used to display information on the users BBM profile.
- When the application is launched, the setupApplication function is called by the following line in the HTML file:

```
o <body onload="javascript:setupApplication()">
```

- This function is used to reset the board, and initialize all variables. The first thing that will need to be added to this function is to initialize the variables we just created. Add the following lines to the top of the function in the JavaScript file.

```
//Reset the number of tries counter
triesCounter = 0;

//Reset the win,lose,tie counters
userWins = 0;
userLoses = 0;
userTies = 0;
```

- When the user exits the application, we want their BBM profile to be updated with their statistics from the last games that they played. In order to do this, we must change the callback function for the BlackBerry system event onHardwareKey. The callback function currently looks like the following:

```
blackberry.system.event.onHardwareKey(blackberry.system.event.KEY
_BACK, function ()
{
    blackberry.app.exit(); //Exit the application
});
```

- What we want to do instead is update the BBM profile item for this application to show how many times the user won, lost and tied in the game. Add the following code to the callback function:

```
try
{
    //Update stats on profile here
    var itemText = "Gave up on playing Tic Tac Toe with
"+userWins+" wins, "+userTies+" ties and "+userLoses+ "
loses! ";
    var itemIcon = "local:///exclaim.jpg";
    var itemOptions = { text: itemText, icon: itemIcon };
    blackberry.bbm.platform.self.profilebox.addItem
(itemOptions);
}
catch (e)
{
    alert(e);
}
blackberry.app.exit(); //Exit the application
```

- From the code, you can see that we create a variable which holds the message we want to put inside the profile box. We also include an icon which will go beside the text, and finally we create a JSON variable called itemOptions which holds both the text and the location of the icon. Afterwards, we call the addItem function of the Profile Box package in the BBM SDK to add the text and icon to the package.

- The last step that will need to be added in the setupApplications function is to clear the profile box of any old entries from the last time the user had the application open. To do this, simply add the following line as the last statement in the function: `blackberry.bbm.platform.self.profilebox.clearItems();`
- Regularly in order to call these functions, we'll need to add the features to the application using the config.xml file which must be included in all WebWorks applications. If you open the file included in the lab, you will see the line `<feature id="blackberry.bbm.platform" required="true" version="1.0.0.0"/>`. This line means that our application can access the BBM platform SDK and all functions within it.
- In order for us to do anything with BBM, we also need to register the application with the platform. Create a function called registerApplication which will be responsible for registering the application with the BBM platform. Before we register the application, we'll need to add some callback functions to handle events which can occur.
- The first callback function we will add handles events when the applications access changes. In the registerApplication function, add the following callback function

```

/**
 * Required
 * This is called when the application's access to BBM platform changes.
 */
blackberry.bbm.platform.onaccesschanged = function (accessible, status)
{
    // If allowed, initialize the application
    if (status === "allowed")
    {
        var startDiv = document.getElementById("start");
        startDiv.innerHTML = "<center><h5>Connected to BBM</h5></center>";
    }
    // If not allowed, show error
    else
    {
        alert("Failed to connect to BBM: "+getStatusMessage(status));

        /*
         * If blocked by the user, add a button to prompt the user to
         * reconnect to BBM.
         */
        if (status === "user")
        {
            var startDiv = document.getElementById("start");
            startDiv.innerHTML = "<center><h5><button
            onclick='showBBMAppOptionsAndRegister()'>Connect to
            BBM</button></h5></center>";
        }
        else
        {
            var startDiv = document.getElementById("start");
            startDiv.innerHTML = "<center><h5>Connect to BBM
            failed</h5></center>";
        }
    }
};

```

- This callback function will alert the user once they are connected to BBM by changing the text in the 'start' div on the HTML page. Likewise, if the connection fails, the user will be presented with a button which reconnects to BBM. We'll define the function

showBBMAppOptionsAndRegister next which will prompt the user to reconnect the application to BBM after we finish with the registerApplication function.

- The next step that is required in the registerApplication function is to actually register the application. We can do this by adding the following lines of code to the end of the registerApplication function:

```
//Register the application
try {
    blackberry.bbm.platform.register({
        // TODO You must define your own UUID
        uuid: "uuid"
    });
} catch (e) {
    alert("You must define your own UUID.");
}
}
```

- You'll notice that we have a UUID field when we register. A UUID is basically a unique identifier for the application. You'll need to generate you own UUID to register the application. An online UUID generator can be found here: <http://www.guidgenerator.com/online-guid-generator.aspx>
- Once generated, your UUID string should look something like: `uuid: "707968e9-ae6d-4dbb-b0f5-058a08df8275"`.
- Finally, add a call to the registerApplication function in the setupApplication function after the call to the setupMenuItems function. The reason we do not place the call to register at the end of the function is because our application needs to be registered before we call any BBM API related functions, such as `blackberry.bbm.platform.self.profilebox.clearItems();` which is at the end of the setupApplication function.
- Now let's add the function to reconnect to BBM in case the connection fails for any reason. You may remember we said the function would be called showBBMAppOptionsAndRegister. Let's create it now. Add the following function to your JavaScript code.

```
function showBBMAppOptionsAndRegister()
{
    blackberry.bbm.platform.showBBMAppOptions(function ()
    {
        registerApplication();
    });
}
```

- This function will show a window allowing the user edit the BBM options for the application. After the window closes, the registerApplication function will be called.
- Another function we'll need to add is the getStatusMessage function. If an error occurs, there is an associated reason. This function will change the error code into a reason the user can understand.

```

/**
 * Returns an access status message to be displayed to the user.
 * @param {String} status The status code.
 */
function getStatusMessage(status)
{
    if (status === "user") {
        return "You decided not to connect " +
            blackberry.app.name + " to BBM";
    } else if (status === "rim") {
        return blackberry.app.name + " has been banned by
            RIM.";
    } else if (status === "resetrequired") {
        return "A device restart is required to use this
            application.";
    } else if (status === "nodata") {
        return "There was no data coverage. Please try again
            when you are in data coverage.";
    } else if (status === "temperror") {
        return "A temporary error occured. Please try again
            in 30 minutes.";
    }
}
}

```

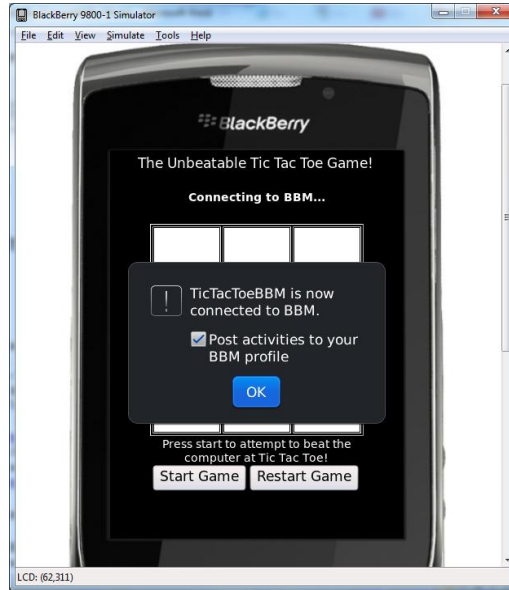
- Now let's test the application. Re-compile, open the MDS simulator and load the application into the simulator. Note: The WebWorks SDK and simulators are *extremely* buggy. You'll likely need to close the simulator and open it again each time you want to reload the application to avoid errors/crashing.
- Once the simulator is loaded, and you have reloaded the application, open it. You'll see something a little different this time.



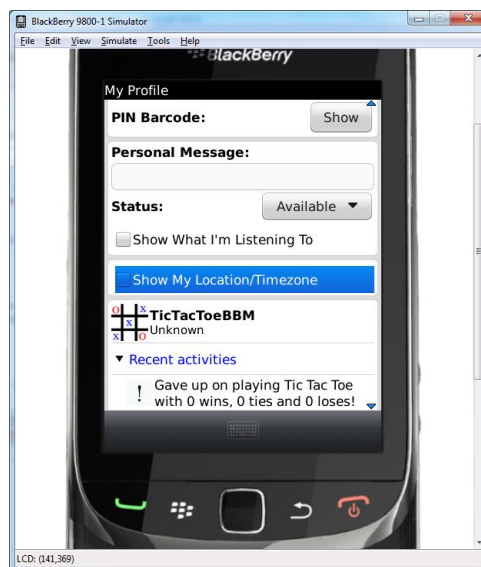
- The BlackBerry simulator is now requesting permission for the application to connect to BBM. Click connect. You'll notice that the BlackBerry simulator will download an update for BlackBerry ID. This is normal. Wait until the process completes, then restart the simulator. Once the simulator is restarted, open the application and initiate the connection to BBM. You'll now need to login with a BlackBerry ID to be able to connect

the application to BBM. A BlackBerry ID can be obtained by visiting <http://us.blackberry.com/id/> if you don't already have one.

- After you've created a BlackBerry ID and logged in on the simulator, you'll see the following:



- Be sure the checkbox is checked so that we can see items that we post from the application within BBM. Once the application is connected to BBM, simply quit the application by pressing the back button.
- Now let's check the BBM profile to make sure something was updated to the profile box. Open BlackBerry messenger on the simulator, click the BlackBerry menu button and then choose the View my Profile option. Scroll to the bottom, and you should see the BBM application at the bottom of the profile. Under Recent Activities, the message we posted when the user closed the application should be visible.





- Now that we know how to post message to the profile, we can use this knowledge to post more messages about the user's progress in the game.
- The next thing we'll do is actually increase the counters when the user won, tied to lost the game. To do this, we need to go to the alertWinner() function.
- Inside the first If-statement, we can include the line of code `userWins++;` as the variable *winner* will be equal to 1 if the user won. Inside the else-if portion, where we check if *winner* is equal to 2, we can include the line of code `userLoses++;` as this portion will be true if the user lost the game. Finally, if the game was a tie, we can include `userTies++;` in the else statement.
- Since we also need to keep track of the number of games the user played in order to display it in their BBM profile, we also need to increment the `triesCounter++;` variable at the end of the alertWinner function, as this function gets called each time the game is over.
- Now let's work on posting to the users profile when each of these events occurs. In the first if-statement, add the following lines of code to post to the users profile when they won a game.

```
var itemText = "Beat the unbeatable Tic Tac Toe game!";
var itemIcon = "local:///thumbsup.jpg";
var itemOptions = { text: itemText, icon: itemIcon };
blackberry.bbm.platform.self.profilebox.addItem(itemOptions);
```

- In the else-if statement which handles the event if the user lost the game, add the following lines of code.

```
var itemText = "Failed to beat the unbeatable Tic Tac Toe game!";
var itemIcon = "local:///thumbsdown.jpg";
var itemOptions = { text: itemText, icon: itemIcon };
blackberry.bbm.platform.self.profilebox.addItem(itemOptions);
```

- Finally, in the else statement which handles tied games, insert the following code.

```
var itemText = "Was so close to beating the unbeatable Tic Tac
Toe game, but sadly the game was tied!";
var itemIcon = "local:///thumbsdown.jpg";
var itemOptions = { text: itemText, icon: itemIcon };
blackberry.bbm.platform.self.profilebox.addItem(itemOptions);
```

- Another good thing to have is an update if the user has not beaten the game after a certain amount of times playing. To accomplish this, we add another if-statement to determine if the triesCounter variable is a multiple of 5. If it is a multiple of 5, we would post a message on the users profile letting everyone on their contact list know how many attempts they made. To do this, we need the following lines of code at the end of the alertWinner function.

```

if(triesCounter%5==0)
{
    var itemText = "Has tried to beat the unbeatable Tic Tac
    Toe game "+triesCounter+" times since opening the
    application!";
    var itemIcon = "local:///exclaim.jpg";
    var itemOptions = { text: itemText, icon: itemIcon };
    blackberry.bbm.platform.self.profilebox.addItem
    (itemOptions);
}

```

- After all the changes, the entire alertWinner function should look like the following.

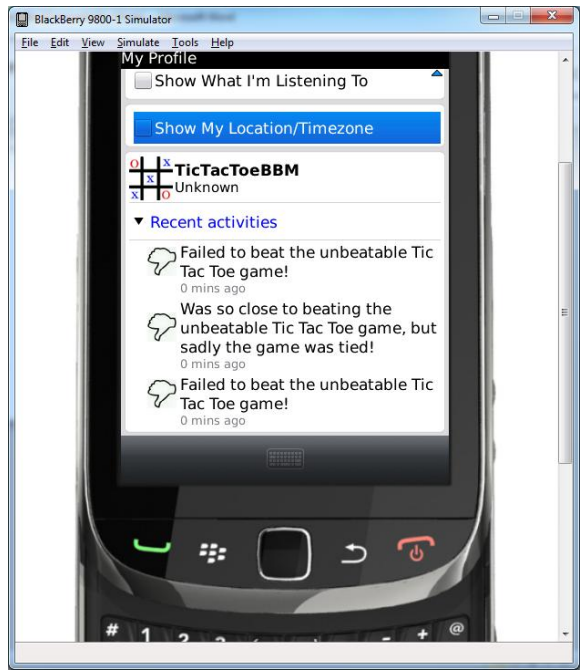
```

function alertWinner()
{
    //if winner == 1, player X won. If winner == 2, player O won. if winner
    == 3, game was tied
    if(winner==1)
    {
        alert("You beat the unbeatable Tic Tac Toe game! Impressive!");
        document.getElementById("infomessages").innerHTML = "You beat the
        unbeatable Tic Tac Toe game! Impressive!";
        var itemText = "Beat the unbeatable Tic Tac Toe game!";
        var itemIcon = "local:///thumbsup.jpg";
        var itemOptions = { text: itemText, icon: itemIcon };
        blackberry.bbm.platform.self.profilebox.addItem(itemOptions);
        userWins++;
    }
    else if(winner==2)
    {
        alert("You lost the game!");
        document.getElementById("infomessages").innerHTML = "You lost the
        game!";
        var itemText = "Failed to beat the unbeatable Tic Tac Toe game!";
        var itemIcon = "local:///thumbsdown.jpg";
        var itemOptions = { text: itemText, icon: itemIcon };
        blackberry.bbm.platform.self.profilebox.addItem(itemOptions);
        userLoses++;
    }
    else
    {
        alert("Tie game! Keep trying!");
        document.getElementById("infomessages").innerHTML = "Tie game!
        Keep trying!";
        var itemText = "Was so close to beating the unbeatable Tic Tac Toe
        game, but sadly the game was tied!";
        var itemIcon = "local:///thumbsdown.jpg";
        var itemOptions = { text: itemText, icon: itemIcon };
        blackberry.bbm.platform.self.profilebox.addItem(itemOptions);
        userTies++;
    }
    triesCounter++;
    gameover = true;
    gameInProgress = false;
    if(triesCounter%5==0)
    {
        var itemText = "Has tried to beat the unbeatable Tic Tac Toe game
        "+triesCounter+" times since opening the application!";
        var itemIcon = "local:///exclaim.jpg";
        var itemOptions = { text: itemText, icon: itemIcon };
        blackberry.bbm.platform.self.profilebox.addItem(itemOptions);
    }
}

```

}  
}

- Recompile and run the application again and play a few games. Afterwards, open the BBM profile again. You'll notice now each time you win, lose or tie a game, an update will be placed on the BBM user's profile.



- The lab is now completed, and you have successfully converted a regular BlackBerry WebWorks application into a BBM connected application. The API has many more features which are possible for applications to incorporate, however the purpose to this lab was simple to give you a taste of what's possible.