Android - Using native pages

fork and edit tutorial (https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/6.3/adding-native-functionality/android-using-native-pages-hybrid-applications.html) | report issue (https://github.ibm.com/MFPSamples/DevCenter/issues/new)

Overview

In this tutorial, integration of native and web "pages" in an Android application will be explained by using the `WL.NativePage.show()` API to open a native page from JavaScript. With this method, data can be sent from JavaScript to the opened native page, and specify a callback to call after the native page closes.

Connecting to the plugin from the JavaScript code

As a first step, `WL.NativePage.show()` needs to be implemented in order to open the native page:

```
function openNativePage(){
    var params = {
        nameParam : $('#nameInput').val()
    };
    WL.NativePage.show(nativePageClassName, backFromNativePage, params)
;
}
```

- `nativePageClassName`: The name of a native Android Activity to start.
- `backFromNativePage`: A callback function to call when the native page closes.
- `params`: optional custom parameters object to pass to the native code.

To handle the callback function:

```
function backFromNativePage(data){
 alert("Received phone number is: " + data.phoneNumber);
}
```

- `backFromNativePage(data)`: After the native closes, it can pass data back to the web part of an application.

Creating a native page

In Android, the native page must be implemented as an Android Activity, or extend an existing Activity. **Step 1** As with any Activity, it must be declared in the `AndroidManifest.xml` file. For example:

```
<activity android:name=".HelloNative"/><activity>
```

Step 2 To retrieve custom data parameters that are passed from the web view, an `intent` should be used:

```
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);

String name = getIntent().getStringExtra("nameParam");
```

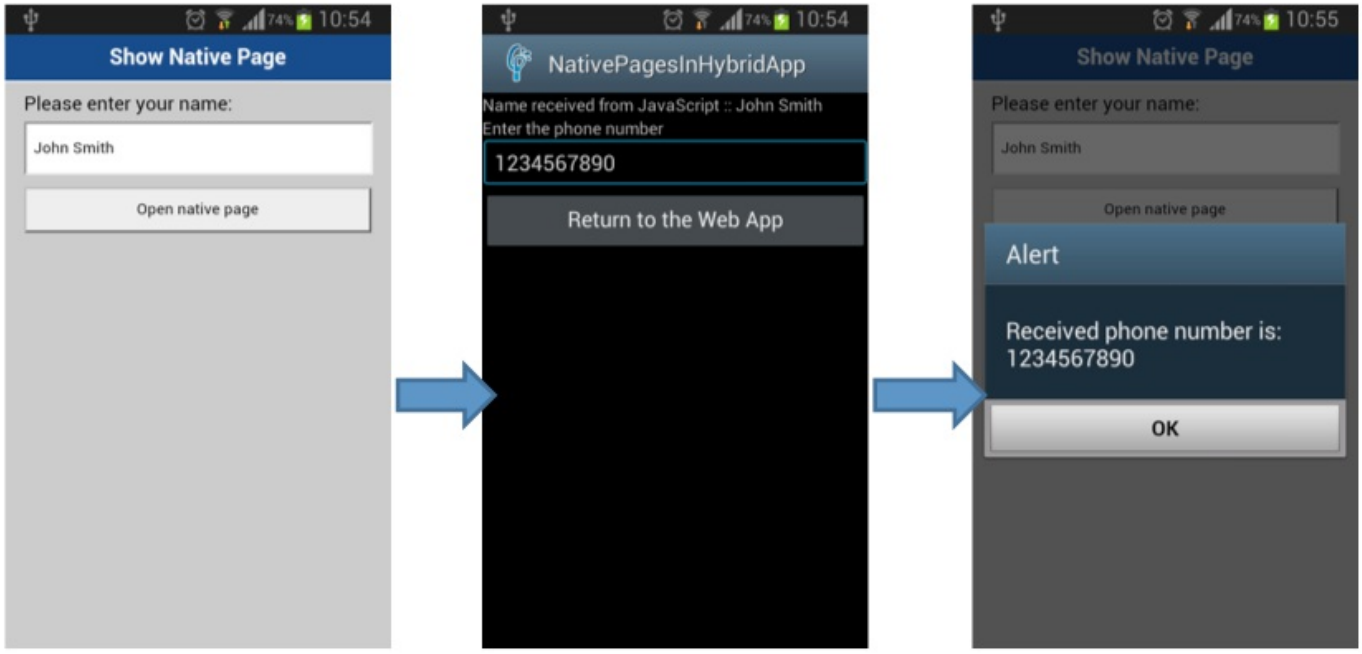
Returning control to the web view

When the native page switches back to the web view, the `finish()` function is called for the Activity. To pass data back to the web view can be done by using an `Intent` object. For example: **Java:**

```
String phoneNumber = editText.getText().toString();
Intent phoneNumberInfo = new Intent();
phoneNumberInfo.putExtra("phoneNumber", phoneNumber);
setResult(RESULT_OK, phoneNumberInfo);
finish();
```

JavaScript:

```
function backFromNativePage(data){
    alert("Received phone number is: " + data.phoneNumber)
;
}
```

Sample application

Click to download (http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v630/UsingNativePagesInHybridAppsProject.zip) the Studio project.