Hands-On Lab

The Windows Store

Lab version: 1.0.0

Last updated: 6/20/2012



Contents

[Overview 3](#_Toc327165546)

[Exercise 1: Detect Trial Versions 4](#_Toc327165547)

[Task 1 – Add WindowsStoreProxy.xml 4](#_Toc327165548)

[Task 2 – Modify the About Page 5](#_Toc327165549)

[Task 3 – Test the Results 6](#_Toc327165550)

[Exercise 2: Simulate App Purchases 8](#_Toc327165551)

[Task 1 – Add Purchasing Code to about.js 8](#_Toc327165552)

[Task 2 – Purchase the App 9](#_Toc327165553)

[Exercise 3: Simulate Product Purchases 9](#_Toc327165554)

[Task 1 – Modify the Item-Detail Page 9](#_Toc327165555)

[Task 2 – Make a Product Purchase 11](#_Toc327165556)

[Summary 12](#_Toc327165557)

Overview

One of the most compelling reasons to write Metro style apps is the ease with which you can publish them to the Windows Store. With more than 500 million PCs worldwide currently running Windows 7, and with each of those PCs representing a potential upgrade to Windows 8, the market – and revenue potential – is both huge and diverse. With a revenue sharing plan that directs up to 80% of the sales proceeds to the author, there is no shortage of motivation for developers to write great apps and offer them to the world.

The store has very flexible monetization options: You can do trials, one-time purchases, in-app purchases, third-party commerce, and advertisement.   
  
For trials you can use the Windows Store APIs in the Windows.ApplicationModel.Store namespace to detect if app is running in trial license. WinRT also offers other APIs make it easy to upgrade from trial versions to paid versions, support in-app purchases of additional products, retrieve licensing information, and more. The Windows Runtime’s CurrentAppSimulator class provides a handy means for simulating purchases and testing code that relies on Windows Store APIs in a controlled environment.

In this lab, you will use the Windows Store APIs to monetize Contoso Cookbook.   
First you will modify the about box to detect trial versions and include a purchase button if the app has not been paid for. Next, you will use CurrentAppSimulator to simulate a purchase when the purchase button is clicked. Finally, you will simulate in-app purchases by offering Italian recipes as a paid add-on rather than for free.

# Objectives

This lab will show you how to:

* + Detect whether your app is running as a trial version
  + Simulate purchases of the app from within the app itself
  + Simulate in-app purchases of additional products
  + Retrieve licensing information regarding apps and products

# System Requirements

* 1. You must have the following items to complete this lab:
  + Microsoft Windows 8 Release Preview
  + Microsoft Visual Studio 2012 RC

# Setup

* 1. You must perform the following steps to prepare your computer for this lab:
  2. Install the Microsoft Windows 8 Release Preview
  3. Install the Microsoft Visual Studio 2012 RC

# Exercises

This Hands-On Lab comprises the following exercises:

* 1. Detect Trial Versions
  2. Simulate App Purchases
  3. Simulate Product Purchases

Estimated time to complete this lab: **30 to 40 minutes**.

Exercise 1: Detect Trial Versions

1. In this exercise, you will use the Windows Runtime’s store APIs to customize the content of Contoso Cookbook’s about page. If the app has been purchased, you will display licensing information. If it hasn’t been purchased – that is, if it’s running as a trial version – then you will display a purchase button instead. Moreover, the price displayed on the purchase button won’t be hardcoded, but will come from listing information retrieved from the Windows Store.

Task 1 – Add WindowsStoreProxy.xml

We’ll be using the CurrentAppSimulator class to simulate purchases, retrieve licensing information, and more. To make the simulation as realistic as possible, we’ll use a file named WindowsStoreProxy.xml to provide information about pricing, expiration dates, and more to CurrentAppSimulator.

* 1. Open the ContosoCookbook project you finished in Lab 7 in Visual Studio. If you didn’t complete Lab 7 or would like to start with a reference copy, you will find a completed version of the lab in the starting materials.
  2. If the project doesn’t already have a data folder, create one.
  3. Right-click the data folder and use the **Add - Existing Item** command to import license.xml from the data folder of the starting materials. This is the file from which we’ll create WindowsStoreProxy.xml.
  4. Open default.js and add the following statements to the other statements like it at the top of the file:
     1. JavaScript
     2. var appmodel = Windows.ApplicationModel;
     3. var storage = Windows.Storage;
  5. Now add the following statements to the activated event handler before the “if (app.sessionState.history)” statement:
     1. JavaScript
     2. appdata.current.localFolder.createFolderAsync("Microsoft\\Windows Store\\ApiData", storage.CreationCollisionOption.openIfExists).then(function (folder) {
     3. appmodel.Package.current.installedLocation.getFileAsync("data\\license.xml").then(function (file) {
     4. folder.createFileAsync("WindowsStoreProxy.xml", storage.CreationCollisionOption.replaceExisting).then(function (newFile) {
     5. file.copyAndReplaceAsync(newFile);
     6. });
     7. });
     8. });
     9. **Note:** The code you just added use the Windows Runtime’s storage API to create WindowsStoreProxy.xml in the specific location in which CurrentAppSimulator expects to find it and copies the contents of license.xml into it each time the app starts up. You can simulate app purchases and product purchases without a proxy file, but if you want a richer simulation that retrieves prices and other information, you must include WindowsStoreProxy.xml.
  6. Open license.xml and take a moment to inspect its contents. The <ListingInformation> element contains information about the app itself and about the Italian-recipes product we’ll offer for purchase in Exercise 3. <LicenseInformation> contains licensing information about the app and the product. In real life, all of this information would come from the Windows Store. In a simulation, however, the information comes from WindowsStoreProxy.xml.

Task 2 – Modify the About Page

Now let’s modify the about page you created in Lab 5. Currently, the words “Trial Version” appear underneath the app title in the about page. We will use store APIs to determine whether this is indeed a trial version and customize the page’s content based on the results.

* 1. Open about.html.
  2. Find the H4 element containing the text “Trial Version” and add an id=”info” attribute:
     1. HTML
     2. <h4 id="info">Trial Version</h4>
  3. Add the following BUTTON declaration after the H4 element:
     1. HTML
     2. <button id="purchase" class="purchase-button"></button>
  4. Open about.css and add the following CSS class to style the button:
     1. CSS
     2. .purchase-button {
     3. width: 225px;
     4. height: 120px;
     5. margin-top: 24px;
     6. }
  5. Open about.js and add the following statement after the “use strict” statement:
     1. JavaScript
     2. var app = Windows.ApplicationModel.Store.CurrentAppSimulator;
  6. Open about.js and add the following statements to the ready function:
     1. JavaScript
     2. if (app.licenseInformation.isTrial) {
     3. // Show the purchase price on the purchase button
     4. var button = document.querySelector("#purchase");
     5. app.loadListingInformationAsync().then(function (listing) {
     6. button.textContent = "Upgrade to the Full Version for " + listing.formattedPrice;
     7. });
     8. }
     9. else {
     10. // Show the expiration date and hide the purchase button
     11. document.querySelector("#info").textContent = "Valid until " + app.licenseInformation.expirationDate.toLocaleDateString();
     12. document.querySelector("#purchase").style.visibility = "hidden";
     13. }
  7. Take a moment to examine the code you just added. It uses CurrentAppSimulator.licenseInformation.isTrial to determine whether the app is running as a trial version, and it calls CurrentAppSimulator.loadListingInformationAsync to retrieve the price that it displays on the purchase button. The returned ListingInformation object contains other information about the app as well. See Windows.ApplicationModel.Store.ListingInformation in the SDK documentation for details.

Task 3 – Test the Results

Now let’s test these changes to see CurrentAppSimulator and WindowsStoreProxy.xml in action.

* 1. Press F5 to launch the app.
  2. Display the charms bar and tap the Settings charm.
  3. Tap About in the settings menu to display the about page.
  4. Confirm that a purchase button appears in the about page, and that the purchase price is $12.99, as shown in Figure 1.
     1. 
     2. Figure 1
     3. The about page for the trial version of the app
  5. Return to Visual Studio and stop debugging.
  6. Open license.xml and change the purchase price in the <Price> element in the <App> section from $12.99 to $8.99.
  7. Launch the app again and go to the about page. What’s the price shown on the purchase button now?
  8. Return to Visual Studio once more and stop debugging.
  9. Open license.xml again and change the price back to $12.99. Also change <IsTrial> from true to false.
  10. Launch the app and go to the about page. Verify that the purchase button is gone and that you now see the “Valid until Saturday, December 31, 2022” message in Figure 2.
      1. 
      2. Figure 2
      3. The about page for the purchased version of the app
  11. Return to Visual Studio and stop debugging.
  12. In preparation for the next exercise, change <IsTrial> from false to true in license.xml.

Exercise 2: Simulate App Purchases

1. You can use WindowsStoreProxy.xml to test changes to your UI based on whether the app is a trial version, but that’s no substitute for being able to simulate actual purchases. In this exercise, you will write a handler for the purchase button so you can “purchase” the app from the Windows Store.

Task 1 – Add Purchasing Code to about.js

To simulate app purchases, we will call CurrentAppSimulator.requestAppPurchaseAsync when the user clicks the about page’s purchase button.

* 1. **Note:** The requestPurchaseAppAsync method submits an asynchronous request to purchase the app from the Windows Store. To determine when (if) the user purchases the application, you listen for licensechanged events. When that event fires, you can verify that the purchase was made by checking CurrentAppSimulator.licenseInformation.isTrial.
  2. Open about.js.
  3. Add the following statements to the end of the “if (app.licenseInformation.isTrial)” clause, before the else statement:
     1. JavaScript
     2. // Handle clicks of the purchase button
     3. button.onclick = function () {
     4. app.requestAppPurchaseAsync(false);
     5. };

Task 2 – Purchase the App

Pretty simple, huh? Now let’s simulate an app purchase. Be aware that CurrentAppSimulator stores information regarding purchases and changes in licensing status in memory; it doesn’t record them in WindowsStoreProxy.xml. Therefore, once you purchase the app, it will remain “purchased” as long as the app is running, but when you restart the app, you will once more be running a trial version.

* 1. Press F5 to launch the app.
  2. Go to the about page and tap the purchase button to simulate an app purchase.
  3. Complete the simulated purchase by tapping the Continue button in the Windows Store dialog.
  4. Display the about page again and confirm that the purchase button has disappeared.
     1. **Note:** Contoso Cookbook doesn’t expose additional functionality to the user once it has been purchased; it merely replaces the purchase button with licensing information. In real life, you might choose to limit what the user can do with a trial version and only expose the full functionality once a purchase has been made.
  5. Return to Visual Studio and stop debugging.

Exercise 3: Simulate Product Purchases

1. In addition to allowing one-time purchase for apps, the Windows Store supports in-app product purchases, too. For example, a game could allow users to purchase additional levels of the game as they complete previous levels. In the Windows Store object model, features purchased this way are known as *products*, and the Windows Runtime provides the APIs you need for product purchases, as well as APIs for determining which products have been purchased, the licensing status of those products, and more.
2. In this exercise, you will modify Contoso Cookbook so that Italian recipes are no longer free, but must be purchased. You will add a simple UI for purchasing them that relies on CurrentAppSimulator, and logic that prevents Italian recipes from being shown in full until after a product purchase has been made.

Task 1 – Modify the Item-Detail Page

The first step is to add a purchase button to the item-detail page. When the page loads with an Italian recipe, we will display the purchase button if the Italian-recipe product hasn’t been purchased, or cooking directions if it has. While we are at it, we will add code to the purchase button allowing an in-app purchase to take place.

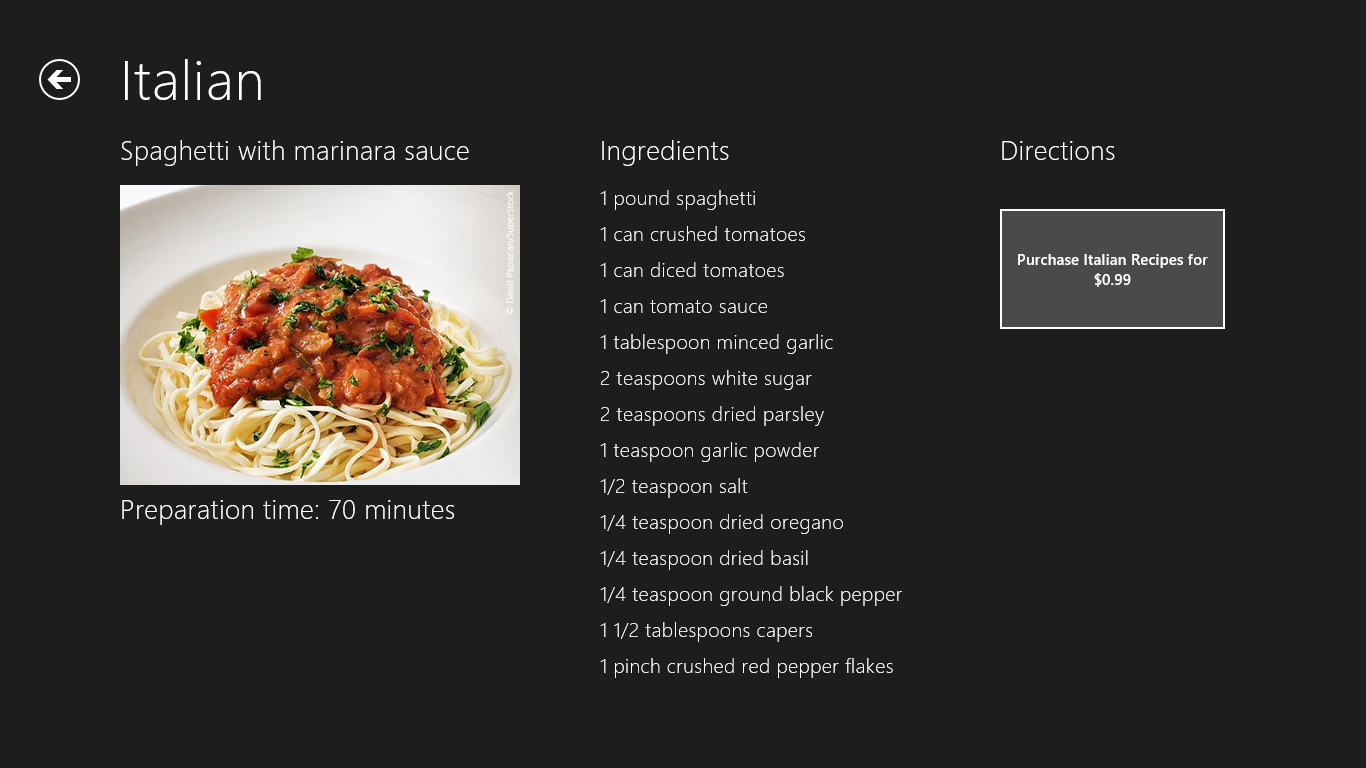
* 1. Open itemDetail.html and find the DIV with the class=”directions” attribute.
  2. Insert the BUTTON element below into the H2 element whose is class is “item-directions:”
     1. HTML
     2. <button id="purchase" class="purchase-button"></button>
  3. Open itemDetail.css and add the following statements to style the button you just added:
     1. CSS

.purchase-button {

* + 1. width: 225px;
    2. height: 120px;
    3. margin-top: 24px;
    4. background-color: rgba(255, 255, 255, 0.2);
    5. }
  1. Open itemDetail.js and replace the line of code that displays cooking directions (the statement that assigns item.directions to the textContent property of an HTML element) with the following statements:
     1. JavaScript
     2. // Determine whether Italian recipes have been purchased
     3. var app = Windows.ApplicationModel.Store.CurrentAppSimulator;
     4. var license = app.licenseInformation;
     5. if (license.productLicenses.ItalianRecipes.isActive || item.group.key !== "Italian") {
     6. // Display cooking directions
     7. element.querySelector("article .item-directions").textContent = item.directions;
     8. }
     9. else {
     10. // Show the purchase price on the purchase button
     11. var button = document.querySelector("#purchase");
     12. app.loadListingInformationAsync().then(function (listing) {
     13. var price = listing.productListings.ItalianRecipes.formattedPrice;
     14. button.textContent = "Purchase Italian Recipes for " + price;
     15. });
     16. // Handle licensechanged events
     17. app.licenseInformation.onlicensechanged = function () {
     18. if (license.productLicenses.ItalianRecipes.isActive) {
     19. // Replace purchase button with cooking directions
     20. var directions = element.querySelector("article .item-directions");
     21. if (directions !== null) {
     22. directions.textContent = item.directions;
     23. }
     24. }
     25. };
     26. // Handle clicks of the purchase button
     27. button.onclick = function () {
     28. app.requestProductPurchaseAsync("ItalianRecipes", false);
     29. };
     30. }
     31. **Note:** There’s a lot going on here, to be sure. We are first checking for a product license named “ItalianRecipes”. That product is defined in WindowsStoreProxy.xml. When the purchase button is clicked, we are calling CurrentAppSimulator.requestProductPurchaseAsync to purchase the product. That call is asynchronous, so we handle licensechanged events and replace the purchase button with cooking directions if the event fires and an additional license check indicates that the product license is now active.

Task 2 – Make a Product Purchase

All that remains now is to test your changes and see a product purchase in action.

* 1. Press F5 to launch the application.
  2. Tap one of the Italian recipes to go to the item-detail page.
  3. Confirm that a purchase button appears in place of the cooking directions, as shown in Figure 3.
     1. 
     2. Figure 3
     3. UI for making a product purchase
  4. Tap the button to initiate a product purchase.
  5. Tap Continue in the Windows Store dialog to simulate a product purchase.
  6. Confirm that the button disappears and cooking directions appear in its place.
  7. View some of the other Italian recipes and verify that cooking directions are shown as normal.
  8. Return to Visual Studio and stop debugging.

Summary

The exercises you performed in this lab demonstrate some of the most important aspects of the Windows Store API – how to detect trial versions, how to simulate purchases of an app, how to simulate in-app product purchases, and how to retrieve information about those products. Of course, in a real app, you will want to replace calls to CurrentAppSimulator with calls to CurrentApp. With that, you will have all the tools you need to monetize your app. Go forth and generate revenue!