Hands-On Lab

Application Bars and Media Capture

Lab version: 1.0.0

Last updated: 6/20/2012



Contents

[Overview 3](#_Toc327336048)

[Exercise 1: Add an Application Bar 4](#_Toc327336049)

[Task 1 – Add an Application Bar to the Item-Detail Page 4](#_Toc327336050)

[Task 2 – Test the Results 5](#_Toc327336051)

[Exercise 2: Add Photo Capture 6](#_Toc327336052)

[Task 1 – Use CameraCaptureUI to Capture Photos 6](#_Toc327336053)

[Task 2 – Enable Webcam Access 8](#_Toc327336054)

[Taks 3 – Test the Results 8](#_Toc327336055)

[Exercise 3: Add Video Capture 9](#_Toc327336056)

[Task 1 – Use CameraCaptureUI to Capture Video 9](#_Toc327336057)

[Task 2 – Test the Results 10](#_Toc327336058)

[Summary 11](#_Toc327336059)

Overview

* 1. The Windows Runtime’s Windows.Media.Capture namespace contains a handy set of classes for capturing media and incorporating it into a Metro-style app. In particular, the CameraCaptureUI class makes it a breeze to use a webcam to snap photos or shoot videos. It’s even cognizant of devices that include two cameras (one front-facing, the other back-facing) and provides a simple and intuitive UI for switching between the two.
  2. In this lab, you will enhance Contoso Cookbook by allowing users to capture photos and videos of their favorite recipes and share them with other applications. You will also add an application bar that provides shortcuts to these features and learn how to incorporate popup menus into application bar commands.

# Objectives

* 1. This lab will show you how to:
  + Implement application bars in a Metro style app
  + Add commands and menus to the application bar
  + Use the Windows Runtime to snap photos
  + Use the Windows Runtime to capture videos
  + Share photos and videos using sharing contracts

# 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

* 1. This Hands-On Lab comprises the following exercises:
  2. Add an Application Bar
  3. Add Photo Capture
  4. Add Video Capture
  5. Estimated time to complete this lab: **30 to 40 minutes**.

Exercise 1: Add an Application Bar

1. Before we add features allowing Contoso Cookbook users to capture and share photos and videos, we need to modify the user interface to provide access to those features. An application bar is the perfect tool for the job.

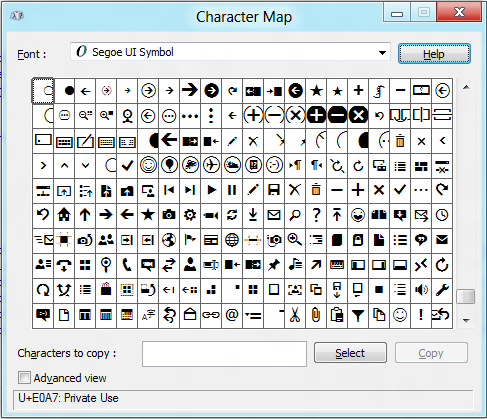
Task 1 – Add an Application Bar to the Item-Detail Page

* 1. We’ll begin by adding an application bar to the item-detail page. We’ll include a Brag command for capturing photos and videos, and we’ll use a popup menu to let the user choose between the two.
  2. Open the ContosoCookbook project you finished in Lab 3 in Visual Studio. If you didn’t complete Lab 3 or would like to start with a reference copy, you’ll find a completed version of the lab in the starting materials.
  3. Open itemDetail.html and add the following statements just before the closing BODY tag to declare an application bar and a popup menu to go with it:
     1. HTML
     2. <div id="appbar" data-win-control="WinJS.UI.AppBar">
     3. <button data-win-control="WinJS.UI.AppBarCommand" data-win-options="{id:'brag', label:'Brag', icon:'emoji2', section:'selection', type:'flyout', flyout:'bragFlyout'}"></button>

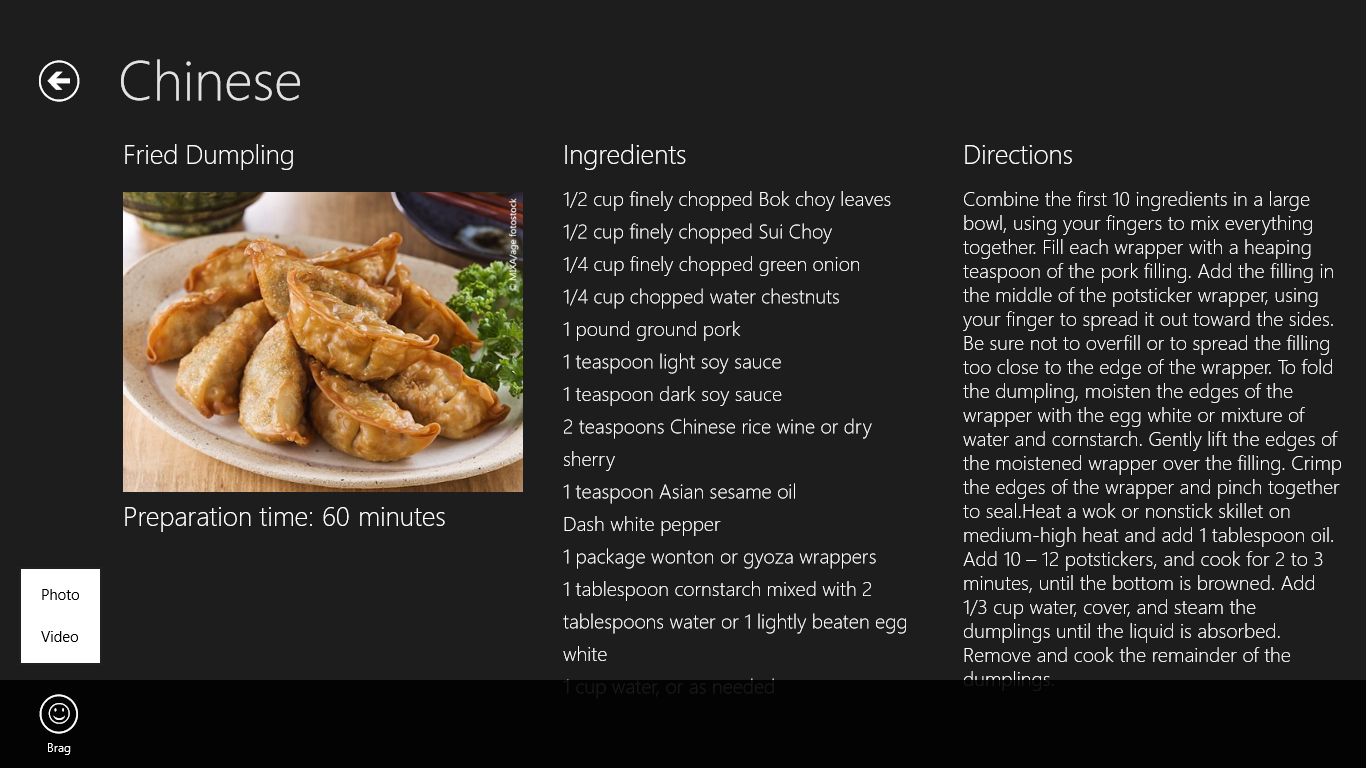
</div>

* + 1. <div id="bragFlyout" data-win-control="WinJS.UI.Menu">
    2. <button data-win-control="WinJS.UI.MenuCommand" data-win-options="{id:'photo', label:'Photo'}"></button>
    3. <button data-win-control="WinJS.UI.MenuCommand" data-win-options="{id:'video', label:'Video' }"></button>

</div>

* + 1. **Note:** An AppBarCommand’s label property specifies the text that appears at the bottom of the command, while the icon property identifies the icon that appears on its face. Names such as “emoji2” are defined in WinJS’s ui.js file and map to character codes in the Segoe UI Symbol character set. To see these characters for yourself, fire up Windows 8’s Character Map application, select Segoe UI Symbol in the drop-down font list, and scroll to the bottom. Instant iconography!
    2. ****
  1. Open itemDetail.js and add the following statements to the end of the ready function to handle clicks of the menu items in the flyout:
     1. JavaScript
     2. // Handle click events from the Photo command
     3. document.getElementById("photo").addEventListener("click", function (e) {
     4. dtm.showShareUI();
     5. });
     7. // Handle click events from the Video command
     8. document.getElementById("video").addEventListener("click", function (e) {
     9. dtm.showShareUI();
     10. });
     11. **Note:** The showShareUI method displays the share UI, which is the same UI that appears when you select the Share charm in the charms bar. You’re stubbing out the handlers for the Photo and Video commands here to show the share UI.

Task 2 – Test the Results

* 1. Now let’s see the application bar in action.
  2. Press F5 to run the application.
  3. Tap a recipe to go to the item-detail page.
  4. Display the application bar by swiping upward from the bottom of the screen, right-clicking with the mouse, or pressing Win-Z.
  5. Verify that the application bar contains a Brag command, and that tapping it displays a popup menu, as shown in Figure 1.
     1. 
     2. Figure 1
     3. The item-detail page’s application bar
  6. Tap the Photo command in the menu and verify that the sharing pane appears.
  7. Return to Visual Studio and stop debugging.

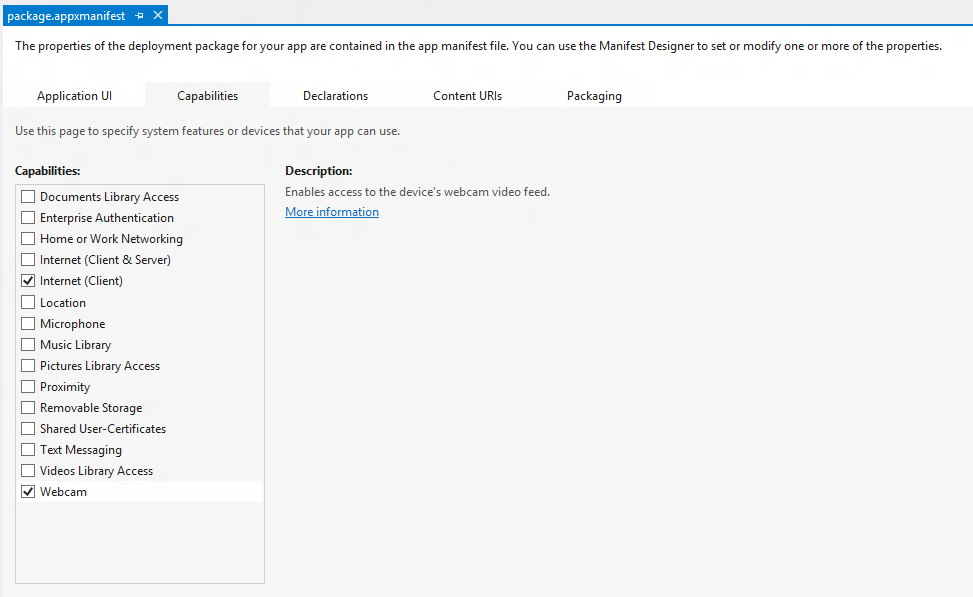
Exercise 2: Add Photo Capture

1. The UI for snapping a photo is in place. Now let’s modify the code to allow the user to take a photo and share it with other applications. The Windows Runtime’s Windows.Media.Capture namespace includes a class named CameraCaptureUI class that provides a high-level interface to camera hardware and makes interfacing with cameras about as simple as it could possibly be.

Task 1 – Use CameraCaptureUI to Capture Photos

* 1. CameraCaptureUI has a captureFileAsync method that makes short work of snapping photos, so let’s put it to work in Contoso Cookbook. While you’re at it, let’s modify the sharing code in the item-detail page so that it can share captured photos as well as recipe images.
  2. Go back to itemDetail.js and add the following statements at the top of the file, right after the statement that defines a variable named dtm:
     1. JavaScript
     2. var capture = Windows.Media.Capture;
     3. var \_photo;
     4. var \_video;
  3. Find the click handler for the Photo command that you added in Exercise 1 and modify it to look like this:
     1. JavaScript
     2. // Handle click events from the Photo command
     3. document.getElementById("photo").addEventListener("click", function (e) {
     4. var camera = new capture.CameraCaptureUI();
     5. // Capture a photo and display the share UI
     6. camera.captureFileAsync(capture.CameraCaptureUIMode.photo).then(function (file) {
     7. if (file != null) {
     8. \_photo = file;
     9. dtm.showShareUI();
     10. }
     11. });
     12. });
  4. Find the onDataRequested event handler that you added in the previous lab. We need to modify it to share a recipe if \_photo is null, or share a photo if it’s not null. To do that, rewrite the function as follows:
     1. JavaScript
     2. onDataRequested: function (e) {
     3. var request = e.request;
     4. request.data.properties.title = item.title;
     5. if (\_photo != null) {
     6. request.data.properties.description = "Recipe photo";
     7. var reference = storage.Streams.RandomAccessStreamReference.createFromFile(\_photo);
     8. request.data.properties.Thumbnail = reference;
     9. request.data.setBitmap(reference);
     10. \_photo = null;
     11. }
     12. else {
     13. request.data.properties.description = "Recipe ingredients and directions";
     14. // Share recipe text
     15. var recipe = "\r\nINGREDIENTS\r\n" + item.ingredients.join("\r\n");
     16. recipe += ("\r\n\r\nDIRECTIONS\r\n" + item.directions);
     17. request.data.setText(recipe);
     18. // Share recipe image
     19. var uri = item.backgroundImage;
     20. if (item.backgroundImage.indexOf("http://") != 0)
     21. uri = "ms-appx:///" + uri;
     22. uri = new Windows.Foundation.Uri(uri);
     23. var reference = storage.Streams.RandomAccessStreamReference.createFromUri(uri);
     24. request.data.properties.thumbnail = reference;
     25. request.data.setBitmap(reference);
     26. }
     27. },
  5. Press F5 to run the application and tap a recipe to go to the item-detail page.
  6. Display the application bar and tap the Photo button. What happens?
  7. Return to Visual Studio and stop debugging.

Task 2 – Enable Webcam Access

* 1. Metro-style apps must have permission to access webcams. That permission comes through the application manifest, which contains metadata about the app. The next step, then, is to edit Contoso Cookbook’s manifest to indicate that it requires access to webcams.
  2. In Solution Explorer, double-click package.appxmanifest to open it for editing.
  3. Go to the Capabilities tab and check the Webcam box, as shown in Figure 2.
     1. 
     2. Figure 2
     3. Enabling webcam access in the application manifest

Taks 3 – Test the Results

* 1. Now that webcam access is enabled, let’s capture a photo.
  2. Press F5 to run the application and tap a recipe to go to the item-detail page.
  3. Display the application bar and tap the Photo button. Click “Allow” if asked if the app can use your webcam.
  4. When the camera-capture UI appears, tap the screen to snap a photo.
  5. Tap the OK button in the lower-right corner of the screen to accept the photo.
  6. When the share UI appears, select a share target such as Share Target Sample App.
  7. Confirm that the photo you just captured is accepted by the share target.
  8. Return to Visual Studio and stop debugging.

Exercise 3: Add Video Capture

1. Contoso Cookbook users can now snap photos and share them with other applications. In this exercise, you’ll add support for capturing videos, too. You’ll change the parameter passed to captureFileAsync to indicate that you want to capture video rather than photos, and use CameraCaptureUI’s VideoSettings property to indicate what format you want to capture in.

Task 1 – Use CameraCaptureUI to Capture Video

* 1. The same captureFileAsync method that captures a photo can be used to capture video, too. Let’s modify the click handler you stubbed out earlier to demonstrate.
  2. Find the click handler for the Video command that you added in Exercise 1 and modify it to look like this:
     1. JavaScript
     2. // Handle click events from the Video command
     3. document.getElementById("video").addEventListener("click", function (e) {

var camera = new capture.CameraCaptureUI();

camera.videoSettings.format = capture.CameraCaptureUIVideoFormat.wmv;

* + 1. // Capture a video and display the share UI
    2. camera.captureFileAsync(capture.CameraCaptureUIMode.video).then(function (file) {
    3. if (file != null) {
    4. \_video = file;
    5. dtm.showShareUI();
    6. }
    7. });
    8. });
  1. Add an else-if clause to the onDataRequested function so the app can share videos as well as photos:
     1. JavaScript
     2. onDataRequested: function (e) {
     3. var request = e.request;
     4. request.data.properties.title = item.title;
     5. if (\_photo != null) {
     6. request.data.properties.description = "Recipe photo";
     7. var reference = storage.Streams.RandomAccessStreamReference.createFromFile(\_photo);
     8. request.data.properties.Thumbnail = reference;
     9. request.data.setBitmap(reference);
     10. \_photo = null;
     11. }
     12. else if (\_video != null)
     13. {
     14. request.data.properties.description = "Recipe video";
     15. request.data.setStorageItems([\_video]);
     16. \_video = null;
     17. }
     18. else {
     19. request.data.properties.description = "Recipe ingredients and directions";
     20. // Share recipe text
     21. var recipe = "\r\nINGREDIENTS\r\n" + item.ingredients.join("\r\n");
     22. recipe += ("\r\n\r\nDIRECTIONS\r\n" + item.directions);
     23. request.data.setText(recipe);
     24. // Share recipe image
     25. var uri = item.backgroundImage;
     26. if (item.backgroundImage.indexOf("http://") != 0)
     27. uri = "ms-appx:///" + uri;
     28. uri = new Windows.Foundation.Uri(uri);
     29. var reference = storage.Streams.RandomAccessStreamReference.createFromUri(uri);
     30. request.data.properties.thumbnail = reference;
     31. request.data.setBitmap(reference);
     32. }
     33. },
  2. Go to the Capabilities section of the app manifest and check the “Microphone” box. This is necessary because when you capture video, CameraCaptureUI uses the microphone as well as the camera.

Task 2 – Test the Results

* 1. Now let’s test the code you just added.
  2. Press F5 to run the application and tap a recipe to go to the item-detail page.
  3. Display the application bar and tap the Video button.
  4. If asked whether the app can use your camera and microphone, click “Allow.”
  5. When the camera-capture UI appears, tap the screen to begin capturing video.
  6. After a few seconds, tap the screen again to stop capturing video.
  7. Tap the OK button in the lower-right corner of the screen to accept the video.
  8. When the share UI appears, select a share target such as Share Target Sample App.
  9. Confirm that the video you just captured is accepted by the share target.
  10. Return to Visual Studio and stop debugging.

Summary

* 1. On some platforms, including photo and video capture capabilities in an application is a difficult undertaking, requiring you to interface with cameras at the device level. The Windows Runtime makes media capture extraordinarily easy by providing the core UI and logic in the CameraCaptureUI class. In Contoso Cookbook, we don’t do anything with captured image and video files other than share them out. However, you could easily use types in the Windows.Storage namespace to save these files to the file system and allow the user to create libraries of recipe photos and videos.
  2. You may not have noticed, but the operating system did something pretty cool for you when checked the Webcam and Microphone boxes in the manifest. If you run the app, select the Settings charm, and select “Permissions,” you’ll see that the permissions page now contains a toggle button for turning camera and microphone access on and off. Apps that use cameras and microphones are required to allow access to be disabled by the user, and you didn’t have to write a single line of code to make it happen.