Xamarin.Android – Activity Lifecycle Demo

Part 1 – Implementing Activity Lifecycle Methods

In this exercise, you will override the activity lifecycle methods, and append the status to the StatusTextView within the Activity's layout.

1. Open the solution in the Start folder
2. In MainActivity.cs, override the following methods: OnStart, OnResume, OnPause, OnStop
3. In each of the override methods, including the OnCreate method, call the private method SetStatus, passing in the state based on the method (ex: Created, Started, Resumed, Paused, Stopped)
4. Build and run the application. Test the application by:
    a. Launching the CTTDNUG meetup site
    b. Pressing the back button to return to the application
    c. Rotating the device or emulator from Portrait to Landscape
    d. Rotating the device or emulator from Landscape to Portrait

What did you observe?

5. Stop debugging

Part 2 – Saving and Restoring Instance State

In this exercise, you will override the OnSaveInstanceState to save the list of statuses from the Text field of StatusTextView. You will then override the OnRestoreInstanceState method to set the text of StatusTextView to the list of statuses that were saved.

1. In MainActivity.cs, override the OnSaveInstanceState method
2. In OnSaveInstanceState:
    a. Call SetStatus passing in state (Example: "Saving instance state")
    b. Save the text from StatusTextView to the Bundle, outstate
3. Override the OnRestoreInstanceState method
4. In OnRestoreInstanceState:
    a. Retrieve the text from the Bundle, savedInstanceState
    b. Set the Text of StatusViewText to the string retrieved from savedInstanceState

    c. Call SetStatus passing in state (Example: “Restored instance state”)

5. Build and run the application. Test the application by going through the same steps described in Part 1, Step #5.
6. Stop debugging and return to the project in Visual Studio (or Xamarin Studio)

Part 3 – Adding a Second Activity

1. In Visual Studio (or Xamarin Studio), right-click on the project and select Add > New Item…
2. In the dialog, select **Activity**. Name the Activity, **NotMainActivity.** Notice that the Activity template already includes an override of OnCreate. However, we don’t have a view to load into this activity. Let’s create one.
3. Right-click on the layout folder (found within the Resources folder), and select Add > New Item…
4. In the dialog, select **Android Layout**. Name the layout, **NotMain**.
5. In NotMain layout, add a TextView and name it NotMainStatusTextView. Be sure to set its layout_width and layout_height to match_parent.
6. In NotMainActivity.cs, add the necessary code within the OnCreate method to load the NotMain layout.
7. Retrieve a handle to NotMainStatusTextView by using the FindViewById<TextView> method. Refer to MainActivity.cs for an example.
8. Similar to MainActivity, create a SetStatus method to append a status string to NotMainStatusTextView.
9. Override the following methods: OnStart, OnResume, OnPause, OnStop
10. In each of the override methods, including the OnCreate method, call the private method SetStatus, passing in the state based on the method (ex: Created, Started, Resumed, Paused, Stopped). Optionally you can append the Activity name to the status, if you wish.
11. In the MainActivity.cs, add code to the Click handler of the launchActivityButton to navigate to NotSoMainActivity.
    a. Hint: StartActivity(typeof(NotMainActivity));
12. Build and run the application. Test the application by:
    a. Launching the CTTDNUG meetup site
    b. Pressing the back button to return to the application
    c. Rotating the device or emulator from Portrait to Landscape
    d. Rotating the device or emulator from Landscape to Portrait
    e. Navigating to the new activity
    f. Rotating the device or emulator
    g. Returning back to the main activity