Android App Development in Xamarin (Part 2)

I. Creating Layouts in Xamarin

A. User Interfaces in Xamarin
- .XAML files usually found inside the ***Resources folder*** of an app project
- Can be created in the Android Designer workspace via the Toolbar
- Usually contains one (1) root item: a ***LinearLayout*** object to put child items of the corresponding UI
- A ***.CS file*** (usually, the ***MainActivity.cs file***) puts logic on the UI activity and handles events of the .XAML file's elements

B. What are ***Layouts***?
- Basically, the "structure" of a specific activity (window or part) of an Android app
- Contains the different controls that execute and manage an app's primary and secondary functions

C. Types of Android App Layouts
- ***LinearLayout*** aligns all children in a single direction, vertically, or horizontally. A scrollbar is created if the length of the window exceeds the length of the screen.
- ***RelativeLayout*** displays child views in relative positions. The position of each view can be specified as relative to sibling elements (such as child A to the left of child B) or in positions relative to the parent ***RelativeLayout*** area (such as aligned to the top of the parent).
- ***GridLayout*** displays items in a two-dimensional scrollable grid. This layout is similar to the Home menu of most Android devices.
- ***FrameLayout*** blocks out an area on the screen to display a single item. Generally used to hold a single child view, because it can be difficult to organize child views in a way that's scalable to different screen sizes without the children overlapping each other.
- ***Fragments*** are small, individual activities embedded within other layouts and activities.

D. Fragments
- Self-contained, modular components that are used to help address the complexity of writing applications that may run on screens of different sizes;
- They can be embedded within other activities and layouts to provide reusability of user activities/layouts with minimal, segregated coding.
- They are also versatile, allowing them to be designed to specifically adhere to a certain structure or layout (such as tab row menus that contain other app activities and controls).

II. Xamarin UI Controls

A. Common Selection Controls

- ***Spinners*** are simple drop-down menus containing a list of pre-determined values.
- ***DatePicker*** is used when date values (date, month, year) must be specified and selected as data.
- ***TimePicker*** works in a similar manner as DatePicker, except that it specifies/selects time values (hour, minute, second).
- ***SeekBar*** is used whenever value adjustments are needed (such as in audio volume, rewind/fast-forwarding of audio tracks, or any other specified value). It uses a bar with control that can be dragged left or right.
- ***Switch/ToggleButton*** is used to turn features and other functions on or off.
- ***CheckBox*** is used when multiple choices are allowed to be selected in an activity.
- ***RadioButton*** is used when only one (1) choice is allowed to be selected in an activity.
- ***Button*** is a control that can be tapped (clicked) in order to execute various operations.
- ***TextField*** is an input control that allows for inputting of alphanumeric values (numbers, letters, certain symbols) via a generated keyboard on the device.

III. Activities and Views in Xamarin

A. What is an ***activity***?

- The main building block of Android applications
- A singular focused part of an app that a user can interact with
- In Xamarin, it is a C# class containing a single user action often found behind a full-screen layout (or ***View***)
- Follows an ***activity state life cycle*** managed by various callback methods to ensure app integrity during operation

B. Android Life Cycle Callback Methods

- ***onCreate()*** – fires when the ***system first creates the activity***; created activity enters the ***Created*** state. Basic application startup logic that should happen only once for the entire life of the activity can be performed in this state/method.
- ***onStart()*** – invoked when the activity enters the ***Started*** state. The call ***makes the activity visible to the user***, as the app prepares for the activity to enter the foreground and become interactive.
- ***onResume()*** – invoked when the activity enters the ***Resumed*** state in which the app interacts with the user. The app stays in this state until something happens to take focus away from the app.
- ***onPause()*** – called during the ***Paused*** state as the first indication that the user is leaving the activity. The method is used to pause operations such animations and music playback that should not continue while the activity is in the Paused state, and that you expect to resume app interaction shortly.
- ***onStop()*** – invoked during the ***Stopped*** state, meaning the activity is no longer visible to the user. This may occur, for example, when a newly launched activity

covers the entire screen. The system may also call the method when the activity has finished running, and is about to be terminated.

- ***onDestroy()*** - Called before the activity is destroyed. This is the final call that the activity receives. The system either invokes this callback because the activity is finishing or because the system is temporarily destroying the process containing the activity to save space.

C. Android App Views
- Represents the basic building block for user interface components
- Works similarly to selection controls, only that some of them aren't necessarily related to input
- Occupies a rectangular area on the screen and is responsible for drawing and event handling
- The base class for ***widgets***, which are used to create interactive UI components (buttons, text fields, etc.)

D. Commonly Used Views in Android Apps:
- ***Button*** is a view user interface element that the user can tap or click to perform an action. It makes use of event listeners and handlers in order to receive input from users and execute its corresponding code.
- ***TextView*** displays customized, non-editable text that can be used to label certain controls. It can contain properties that alter its appearance, such as size and color.
- ***EditText*** is a user interface element for entering and modifying text. The ***TextView_inputType*** attribute must be specified to determine the type of input keyboard to be generated (such as keyboards used for entering email addresses, for example).
- ***ImageView*** displays image resources, for example, ***Bitmap*** or ***Drawable*** resources. It is also commonly used to apply tints to an image and handle image scaling.

References:

- Hermes, D. *Xamarin Mobile Application Development*. Retrieved August 2, 2017.
- Petzold, C. (2016). *Creating Mobile Apps with Xamarin.Forms*. Retrieved August 2, 2017.
- Microsoft Docs. *C# Guide*. Retrieved August 17, 2017 at https://docs.microsoft.com/en-us/dotnet/csharp/index.
- Android Developers – *Views* (n.d.). Retrieved August 25, 2017 at https://developer.android.com/reference/android/view/View.html.
- Android Developers – Activities (n.d.). Retrieved August 25, 2017 at https://developer.android.com/guide/components/activities.