VLC Android – MediaLibrary

VLC Architectural Pattern

Since our group has chosen an Android app, this actually makes it easier to find the architectural pattern since most Android apps mainly use a couple of patterns and those patterns are similar to MVC but changed a little bit to fit Android. The problem with MVC is that it is tightly tied with the Android APIs so this makes it difficult to unit test. Another problem of MVC is that the controller is tightly coupled with the view and this basically make it an extension of the view. If we change the view we must change the controller too. Because of this and the research I have done in to the VLC code and online I have concluded that the architectural pattern that the Android version of VLC uses is MVP (Model-View-Presenter).

Why MVP?

With MVC there are too many issues that just don’t work with Android. MVP almost gets rid of most of these problems without there being a massive difference from MVC. MVP has clear separation of responsibilities between components. This allows for an easier understanding and maintenance of the code. As stated before there is not a massive difference, so MVP still has all the good stuff that MVC offers such as modularity, which allows you to change a component without changing other components and because the separation is well defined with components, testing become a lot easier because you can test a component in isolation.

The M in MVP is still the same as MVC. It is still the model. It is the data + state + business logic layer, so nothing has changed.

The V in MVP still stands for view. The view still has same description of the view in MVC but instead the Activity which is part of Android, is now considered part of the view. This gets rid of coupling to any specific view and allows test with a mock implementation of the view.

The P in MVP stands for presenter. The presenter is basically the controller except it’s not all tied to the view, it is just an interface. This helps with the modularity and testability that MVC has a problem with. The presenter just displays “what” it needs to not “how” to display it.

From all this information I have gathered and by studying the code and other examples of MVP, I saw a big similarity where the example codes and the VLC code used an “interface” section with code looking very similar to each other. This was a major reason for me to decide this was an MVP architecture pattern, but I can’t really say why because there are problems with MVP and it is not perfect either, but I guess it just comes down to the preference of the team making this.