**Implementation details**

* **The layout of your user interface should work on both tablets and phones, in portrait and landscape modes.**
* **You must use the contents of words.plist (iOS) or words.xml (Android) as your universe of possible words. You’re welcome, but not required, to transform it into some other format (e.g., SQLite).**
* **You must implement your app’s two strategies for gameplay (evil and non-evil) in two separate model classes called EvilGameplay and GoodGameplay both of which must implement a protocol (Swift) or interface (Java) called Gameplay. In other words, based on whether evil is enabled or disabled, your app should pass messages to an instance of one class or the other.**
* **Your app must come with default values for the app’s three settings; those defaults should be set through a PreferenceActivity (Android) or a separate view controller (iOS). Those defaults should be set in NSUserDefaults with registerDefaults: (iOS) or as an android:defaultValue in a SharedPreferences XML file (Android).**
* **You must implement each of the numeric settings with a slider control. Each slider should be accompanied by at least one label that reports its current value (as an integer).**
* **You must implement the evil toggle with a switcher control.**
* **You must obtain a user’s guesses via a text filed (and the on-screen keyboard that accompanies it). For the sake of aesthetics, you are welcome, but not required, to keep that text field hidden (so long as the on-screen keyboard works). You are also welcome, but not required, to respond to user’s keypresses instantly, without waiting for them to hit return or the like, in which case textField:shouldChangeCharactersInRange:replacementString in the UITextFieldDelegate protocol (iOS) might be of some interest.**
* **You must implement the display of high scores in a controller called HistoryViewController (iOS) or HistoryViewActivity (Android) that presents itself at game’s end via a UIModalTransitionStyleCoverVertical transition (iOS).**
* **You must implement methods with which to store and retrieve high scores in a model called History. You must store high scores persistently, as in a property list other than words.plist (iOS) or in a shared preference (Android), or in some other format (e.g., SQLite).**
* **Your app must work within the iPhone or Android Simulator; you need not test it on actual hardware.**