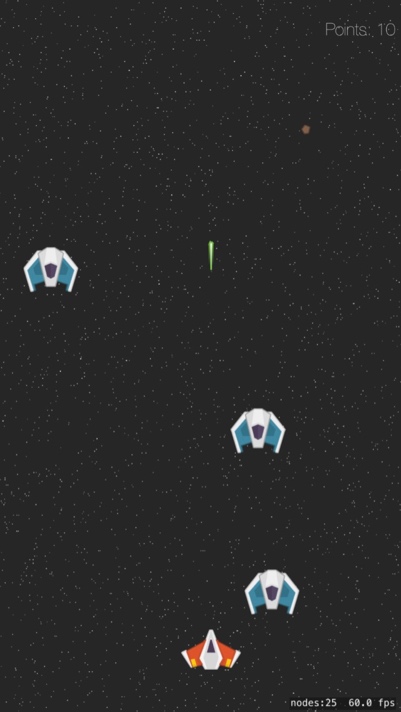
### Purpose of the application

The purpose of this project was to create an innovative game which utilized the sensors on an Apple Watch to control player motion on the screen of a paired iPhone. In order to do this an iOS app running on an Apple iPhone, iPad or iPod Touch and a WatchOS app running on a paired Apple Watch was needed. After much research and consideration, we came to this idea as we wanted both a challenge and to create something that had previously not been done before.

Our chosen project was going to be a challenge for a couple of reasons. One we had not previously done any native development on either iPhone or Apple Watch, and on top of this the language we would be writing in was going to be Swift, which we did not have any experience in. The second reason that this would be difficult is that during our initial research there were many aspects to creating this kind of application that were either unclear if they were possible or there was a very limited amount of information that we could draw from online.

### Gestures identified as appropriate for this application - (35%)

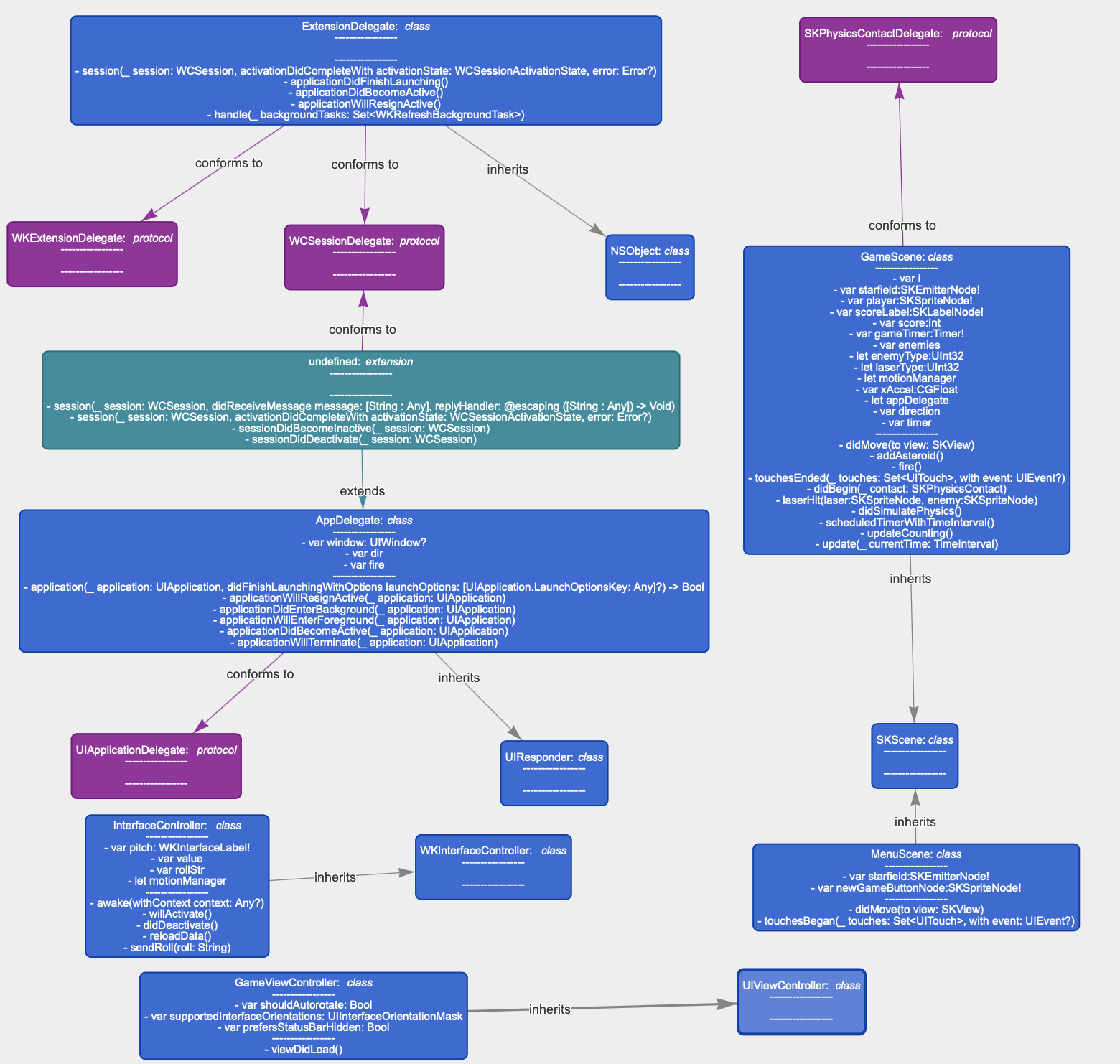
consider how gestures can be incorporated into the application, providing a justification for the ones that you pick. This is an important research element for the project and needs to explain how the gestures fit into the solution you are creating.

### Hardware used in creating the application

### Architecture for the solution

The full architecture for the solution, communications and distributed elements that you are creating. The architecture must make sense when the gestures and the hardware are combined. Justification is necessary in the documentation for this.

### Class Diagram



### Data Models

The data that we are recording from the devices sensors is the pitch attitude. This is being converted to a Double on the Watch app and is sent over WatchConnectivity to the iPhone AppDelegate as a dictionary. This dictionary is made up of a key and value pair.

### Communication

### Libraries

* WatchKit

The WatchKit framework contains the classes used to create a WatchOS app. A WatchOS app may contain elements such as tables, buttons, sliders etc. and these visual components are manipulated by WatchKit to respond to user interactions.

* WatchConnectivity

This framework is used to transfer data between the paired Apple Watch and iPhone. For this project we utilized WatchConnectivity to send live data from the Watch to the phone.

* CoreMotion

CoreMotion is used to report environmental data taken from an iOS devices onboard sensors such as the gyroscope, barometer, manometer and G-meter.

* GameplayKit

Used for building iOS games, the GameplayKit is an Object-Oriented framework that includes tools for designing games with reusability and functionality in mind. This framework is used for the player movement, physics, collisions, emitters and spawning to name a few.

* UIKit

This is the framework that provides the basic infrastructure for iOS apps. The main app loop window and view architecture and event handling for input are all features controlled by UIKit.

### Conclusions & Recommendations

Conclusions are what you have learned from this project and the associated research. Recommendations are what you would do differently if you were to undertake the project again. The Reflective Piece – what I learned and “enjoyed”! This gives scope for a critical evaluation of the project and the objective that you tried to achieve.