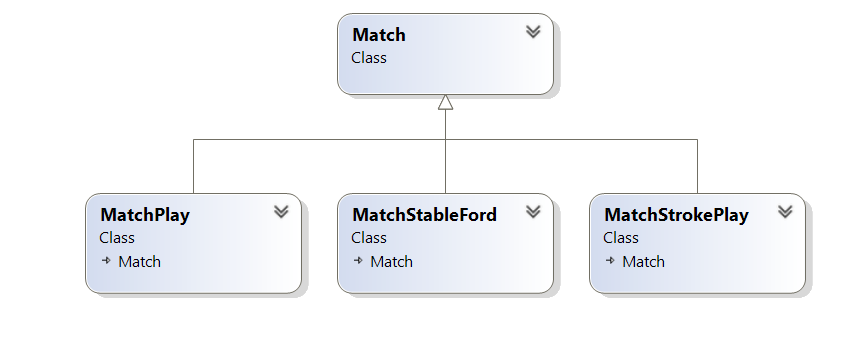
Design

Why I chose an Object Oriented Approach?

Object Oriented coding is based purely on the concept of “Objects”. This means that you can store the data that the physical object would have, just like the sheet of paper that you would write you’re scores on. In the program these objects are fields. The most important reason that I chose to use object oriented Programming techniques is the ability to call methods on an object. For example, “Create A New Golfer” which creates a new instance of the class / object of golfer. This means that it is just like the physical object and makes it very useful in thinking what the main functionality requires. You can also extract a lot of data on objects as it is all in the same place. In addition, as the client if they are not use to programming it can be easier for them to comprehend what is happening.

Class Diagram showing inherence for matches.



This class diagram shows that I have deicded that MatchPlay,MatchStableFord and MatchStrokePlay all inherit data and behavior from parent class. This parent class is “Match” and is also known as the super class. The written terms this effectively means MatchPlay “is a “Match and the same for MatchStableFord and MatchStrokePlay. This is helpful in when they all have a certain same method and there for I can reduce the code duplication.

Objects:

Course: This is just as the name suggests it’s a course. Each course has a few properties that are just data about that course for example a name location address and so on. Course also has a collection of holes this means there is a one to many relationship between course and holes. I have made Hole a separate object because that its self has data about it and its own functions.

Golfer: A golfer is the account that you as a user will create when joining the application. You will need to add handicap but will not have to add properties like emails are optional.

Hole: As mentioned before holes interact with course but other than that they have no method on them selve just optional data. This will effective be the exact some rows that are on a paper score card.

HoleScore: this score for each hole and portrayed as a table within the interface. This is the actual score sheet that you can see.

Match: Match is the super class for the other match types it contains the collection of golfers showing another many to one relationship as a golfer can only be in one match at a time. It also ahas the one to one relationship for course.

MatchStableFord, MatchPlay, MatchStrokePlay: these are the subclasses and contain the algoriths for calculating the scores and the handicap effects. Since each of these are different to each other It was best to separate them into subclasses of match.

For each of the main objects there is also a Configuration class which creates new instances and or methods like show all golfers.