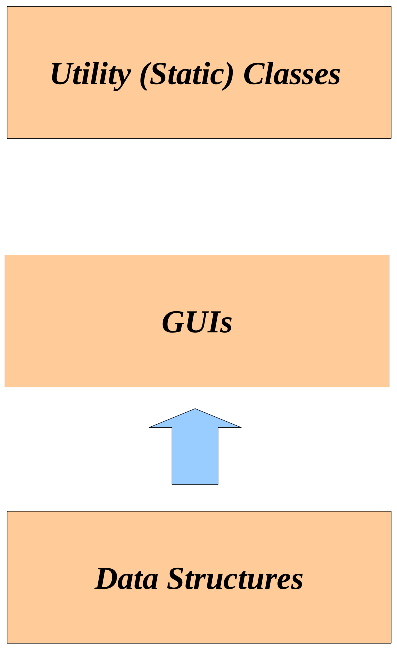
NewObject – Design and Architecture

Table of Contents

Overview

Our design document is split up into different categories, each giving a high-level graphical representation of each sub-system shown below.



Each sub-system shown will be explained in more detail. A brief explanation of the purpose of each sub-system will help the reader understand our goal of creating or grouping our application in the way we have. Priorities will explain what the sub-system must do to a chieve its purpose proposed. Design issues will also be clarified, and course of action explained for how the issue was resolved. Also UML diagrams provided to show a little more detail of each sub-system.

GUI Sub-system

Purpose

Our GUI Sub-system describes the design decisions involved in implementing the GUI for the application and deciding how to present the information to the user. Also decides how the user is to interact with the application.

General Priorities

1. Make it easy to use.
   - Ease of usability, common tasks are easier accessed, multiple ways to do tasks etc
2. Aesethically pleasing
   -
3. Continuous integration with other Sub-systems
   - Linking of Sub-system is well-defined, each is its own seperate entity.

Design Issues

Issue 1: C# vs Java, Language Decisions

Normally we would use Java, but we thought that possibly using C# would make GUI building easier.

Option 1

Java. We would have easy access to tools and utilities that have been tested and proven, and used frequently by other groups in the same course. Also, the assistance from the course co-ordinator would be instantaneous rather than delayed.

Option 2

C#. GUI design would be easier using Visual Studios©™ drag and drop feature rather than using javas swing classes. There is some proven tools for use with C#.

Decision

Java wins the fish.

Issue 2: Design of initial startup

We talked about a few ways that the application could startup (as far as look and feel goes), and the various pros and cons of each.

Option 1

Discussed having the main GUI load straight from execution. This allowed a user instant access to the GUI, but didnt provide functionality for easy team setup.

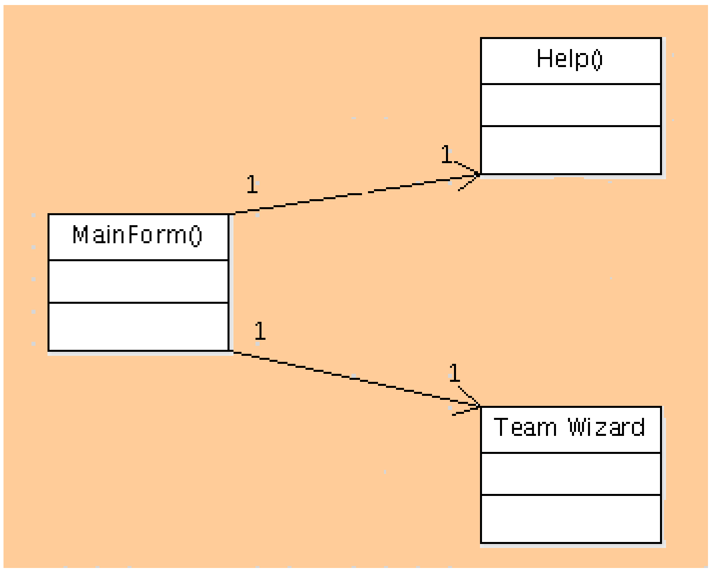
Option 2

Discussed having team wizard appear before main GUI if application was run from an executable. Provides ease of setup for players and teams associated with groups of players. Also allows user to load in previously created team data.

Decision

Option 2 was decided to be better as we could incorporate an easy way of loading a team in the wizard.

Gui Sub-system UML Diagram


Help()
1
1
MainForm()
1
1
Team Wizard