Project 1 - Proposal

Main Goal

To design an interactive, but simple, theatre utility that will simulate a play or theatre production.

Schedule

- **Week 0** : Shoes (Planning) : [2/18/09 - 2/23/09]
  - Plan and write out a proposal of project requirements and deadlines.
  - Discuss and decipher program’s “wants” and “needs” with respect to the project’s main goal in focus.
- **Week 1** : Skateboard (Alpha 1) : [2/23/09 - 2/27/09]
  - Complete a working prologue of the program.
  - This week should result in a completed first chunk of functionality.
- **Week 2** : Go-Cart (Alpha 2) : [3/2/09 - 3/6/09]
  - Complete a working prototype.
  - This week should result in a completed skeleton of the program’s core, in which all the program’s basic “needs” are implemented, functioning, and this satisfied.
- **Week 3** : Car (Beta) : [3/9/09 - 3/13/09]
  - This week should result in a completed version of the program that is ready to be tested and debugged.
  - All features that should be implemented will be functioning at a testable level.
- **Week 4** : Rocket-Ship (Final) : [3/23/09 - 3/27/09]
  - This week should result in a completed product.
  - The testing and debugging phase will be completed within this week, and the program will be available for public use.
Outline

The time constraints set for the development of this program have divided all possible aspects and features of the program into two sets. The first, and more important of the two sets, is the “needs”. The second is the set of the program’s “wants”.

The program’s “needs” will be all the features of the program that must be functioning and debugged by the time of the project’s completion date. These features are the raw necessities and the most important of all the program’s features that will be implemented upon completion.

The program’s “wants” will be all the features of the program that should only be implemented if, and only if, they do not interfere with the outcome and scheduling of the program’s “needs”. These features are mostly for show, to enhance the user’s immersion with the program, and to polish off the interface of the program.

The following table is a spreadsheet example with the program’s “wants” and “needs” all laid out according to category.

<table>
<thead>
<tr>
<th>Project's &quot;wants&quot; vs. &quot;needs&quot;</th>
<th>Wants</th>
<th>Needs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SS Stuff - Bucket Selection</td>
<td>User selection</td>
<td>Predefined selection</td>
</tr>
<tr>
<td>SS Stuff - Upload/Download</td>
<td>User abilities</td>
<td>Manually do it</td>
</tr>
<tr>
<td>Images</td>
<td>Auto/user scaling images (e.g., backgrounds for sets)</td>
<td>Manually scaling images before uploading them to the bucket.</td>
</tr>
<tr>
<td>Sounds</td>
<td>N/A</td>
<td>MP3</td>
</tr>
<tr>
<td>Players</td>
<td>Customized poses/postures features</td>
<td>Some representation of Actors,Actresses on screen.</td>
</tr>
<tr>
<td>Scene</td>
<td>Ability to add props, etc.</td>
<td>Ability to compose a scene that consists of a set, players, and sounds.</td>
</tr>
<tr>
<td>Record/Play Back</td>
<td>Ability to record a scene.</td>
<td>N/A</td>
</tr>
<tr>
<td>3D</td>
<td>3D image rendering.</td>
<td>N/A</td>
</tr>
<tr>
<td>Lighting</td>
<td>Shading</td>
<td>Some representation of lighting.</td>
</tr>
<tr>
<td>Actions</td>
<td>Ability to have players perform custom gestures and actions.</td>
<td>Ability to have players perform default actions, such as walking.</td>
</tr>
<tr>
<td>Real Time Play</td>
<td>Ability to have user manipulate players in real time via keyboard, microphone, or mouse</td>
<td>N/A</td>
</tr>
<tr>
<td>Camera</td>
<td>3D camera angles, or multiple camera angles.</td>
<td>Default front view.</td>
</tr>
</tbody>
</table>
For the program’s scene setting feature, the idea of a basic 2D set will be implemented. Ideally, a 3D rendering of a stage would better represent the idea of an actual set; however, in an effort to meet the demands set by the project’s schedule, a simple 2D set will suffice. To show how the set will be represented on-screen, I provided a simple drawing of the program’s stage/scene interface.

The program will divide its interface into three separate UIs, which will allow the user to manage the main pieces of their project with ease. The idea of the tabs is to keep the users focus on one aspect of the play at a time. This will result enhance the organization of the user’s play, or project.

The first tab is the stage tab. This tab will be used to let the user manipulate the features of the main set, such as backdrops and lighting. The second tab is the cast tab. This tab will be used to allow the user to manipulate and select the actors and actresses in their play. The last tab is the props tab. This tab, like the cast tab, will give the user the ability to select and manipulate the props that they want to use in their play.
For the program’s actors and actresses, an avatar will give the user a graphical representation of the player. The avatar, or player, will be a 2D representation of the actor, or actress. It will have a numerous fields, most of which will be for on-screen rendering. To better explain this, I drew up an encapsulation diagram of the Player object.

```
Player
Fields:
- Name - Actor/Actress name.
- Gender - Specifies actor or actress.
- Position - Stage position of player.

Face
Fields:
- Ears, eyes, nose, mouth, etc.

Limbs
Fields:
- Arms, legs, (Torso), etc.
```

[Please excuse the uneven shapes, Adobe Photoshop wasn’t being cooperative]

The program will also have a similar component set up for props objects, although the fields will be more much more basic since props have no life-like features. It may be also prudent to design a parent object that both props and player can inherit from to preserve an underlying universal component structure.
The program’s main stage UI will be set up will have a main stage object, this object will be responsible for providing an organized representation of the main stage’s features, both graphical, or on-screen, and internal. This object will also be responsible for managing these features in a systematic and simple manner. The main stage will be a singleton, to keep management flaws to a minimum. To aid in the description of this object, I have provided the following encapsulation diagram.
**Week 0 - Shoes**

This week will be reserved for outlining the major concerns of the program. Specifically the project’s main goal, structure of components from an object-oriented standpoint, and overall design of program’s main foundation and GUI.

**Week 1 - Skateboard**

By the end of this week, the program should have a fully developed base. The functionality of this base should include some sort of singleton object implementation that will act as a global execution manager for the program’s internals. Accompanying this should be a very basic interface that will provide functioning interaction between the global execution management object and the user’s input.

Besides this basic functionality, certain aspects of the program should be also carefully chosen by this point, such a specified frame-rate for the flash interface, as well as other variables that correspond to the specifics of the language.

**Week 2 - Go-Cart**

This week is basically a continuation of week 1. It will result in the program’s alpha 2 release, which will have very limited functionality, just as alpha 1 had in week 1. The difference is that by this week, the program’s main internal functionality will be complete and sufficient for building on top of. This means that all the kinks in the framework have been resolved; thus, a strong platform is ready to have interactive features constructed upon it.

**Week 3 - Car**

This is the week in which the program really begins to take off. The foundation was laid during weeks 1 and 2, and now the beta version for the program should be complete and ready for testing. This version of the program has all features from the “needs” category implemented and functioning.

This phase will probably have a lot of bugs in it, and thus it will require heavy testing. To help with testing it may be a good idea to get the opinions and reactions of test users with multiple levels of computer and GUI experience. This will aid in the exposing of hidden bugs, useless or uncanny features, and UI faults, such as confusing layouts, etc.
Week 4 - Rocket-Ship

This is the week that the product is debugged and released as a finished product. All features of the project have been implemented and a running smoothly, all requirements have been met, and the project's main goal has been achieved.