TDDC74 - Projektspecifikation

Projektmedlemmar:
Henrik Österman henos134@student.liu.se
Erik Bäcklund Ekvall eriek286@student.liu.se

Handledare:
Johannes Schmidt johannes.schmidt@liu.se

14 april 2015

Innehåll

1 Projektplanering

The project is supposed to result in a game-engine made specially for making 2D roleplaying games. The focus is on making tools to rapidly produce simple games.
The code will be object-oriented primarily, using multiple inheritance but functional code will also be a part of the project. As an example we intend to write our own scripting language which will be functional.

1.1 Kort projektbeskrivning

A 2D RPG game engine with tools. Examples of tools are a scripting language for making dialogue, pre-existing class structure which allows quick and easy creation of new objects etc.
We recommend looking at IceBlink Engine and Final Fantasy 4-6 for examples of the type of engine we will try to achieve.

1.2 Utvecklingsmetodik

We will work separately the majority of the time and later have follow-up meetings. In practice this will result in us working in the same room but separately so we can communicate progress as it happens. We will use git for revision control. The git repo can be found at www.github.com/zappater/riverengine/.

1.3 Grov tidplan

We will implement one version of both the graphics engine and game engine every week. By the half time meeting we will be working on version 3. Information about what the versions contain can be in the document 'Version.org'.

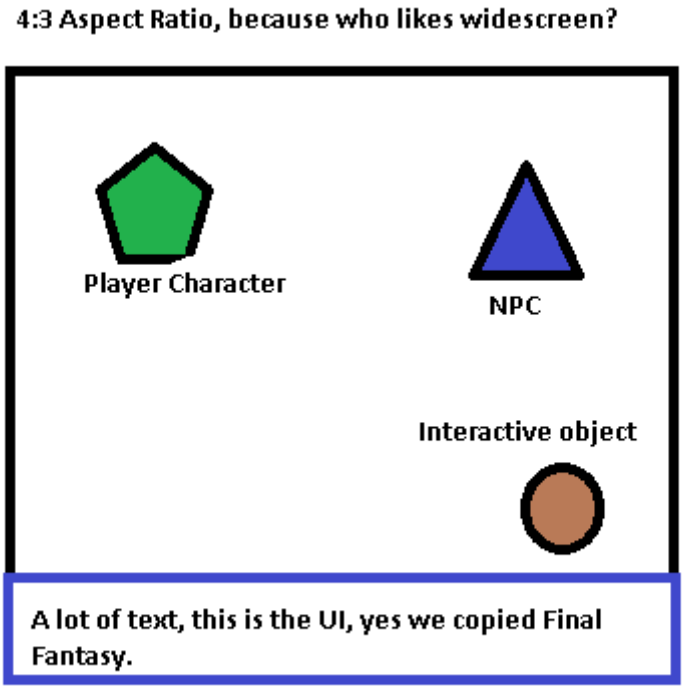
1.4 Betygsambitioner

We will only attempt to achieve the grade 3 on the project as we both already have a 5 in the course from the exams. This allows us to let other courses take more time if they need to. However we expect to achieve a 5 on the project if we aren't distracted by other courses.

2 Konceptskiss

For the developer: A lot of code, more like a library than anything else.

For the player: A lot like Final Fantasy, look at the picture and use your



imagination.

2.1 Kravlista

#	Beskrivning	Prioritet
1	NPC dialogue read from file and scripted using its own language	A
2	Scripting language for making objects	A
3	Simple graphic editor	B
4	2D graphics engine	A
5	Scripting language for controlling UI	A
6	Support for animations	B

3 Implementation

3.1 Abstrakta datatyper eller klasser

Act: Contains information about what should be read in to memory. It has a type (game or UI). An act contains a n-dimensional vector (in the mathe-

matical meaning) of levels. A game is a collection of acts.
Level: A level is a collection of pointers to objects which are in the same interactionlevel and will be drawn in the same layer by the graphics engine. The difference between interactionlevel and layer is that a interactionlevel determines what the player can interact with while a layer determines when the graphics engine draws the object.
Classes: Items, NPCs, Player Character, World Objects, Sprites, Background. All of these with their own subclasses.
Dialogue will be its own type of ADT. In essence it will be a graph implemented using linked-lists and contained in one or several textfiles.

3.2 Testning

Unit tests will be created as the engine is developed, we are not yet able to specify unit tests.
We will test the system by making a simple techdemo in it. It will contain 1 PC, 2 NPCs with dialogue, 1 interactive world object, 1 item. It will be made in 2 acts.

4 Tidrapportering

Can be found in separate spreadsheet, which will be found in our git repository. So far we have each spent 20 hours on planning this project and this specification.