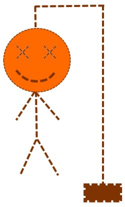
# HANGMAN THE GAME

## SOFTWARE DEVELOPMENT PROJECT

**RUSLAN ABDULIN**

5 mar 2019



# Contents

# 1 Revision History

| Date | Version | Description | Author |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |

## 2 General Information

| Project Summary | |
|---|---|
| Project name | Project id |
| HANGMAN THE GAME | |
| Project manager | Main client |
| Ruslan Abdulin | Linnaeus University |
| Key Stakeholders | |
| Tobias Andersson Gidlund, Daniel Toll, Tobias Olsson | |
| Executive Summary | |
| | |

# 3 Vision

The software project ”Hangman” is a game in text based fashion. The main purpose of developing the game is to get familiarise with the entire process of software development. The basic idea of Hangman is that the player is going to guess a word by suggesting letter after letter.

## 3.1 The game process

The player starts with stating his/her name. Then he/she gets a menu which allows him/her to start or to continue the previous game. Also, game configuration is be available.

Once started a new game the player is presented with the number of letters in the word but for every wrong guess the game is building a part of a man getting hanged. The player will loose if the hangman is completed. If the player guesses the word he gets some points to his score and presented with a new word and so on.

Every player has his own high score and the table of top high scores is available which encourage a player to beat the highest score.

## 3.2 Vision reflections

Vision is an idealistic view at the outcome of a project which visualize the main idea or main benefits of the project at the very early stage. The idea behind the vision may be both to engage people in the project and to show how important the project is. Also it is a basis for future discussions about the project. Vision doesn't focuses on specific details, technologies or people involved.

# 4 Project Plan

## 4.1 Introduction

The main purpose of the project is to develop a text-based guessing game. The game should be designed in 4 iteration, be tested and well documented. Overall time to develop the game almost 2 months.

## 4.2 Justification

The project goes through the entire process of software development. Creating this simple game allows to train such skills as project management, programming, testing, time management, risk management and creating the documentation. When it is done the project manager and the programmer will be ready to work on more complicated and difficult projects.

## 4.3 Stakeholders

- Tobias Andersson Gidlund
- Tobias Olsson
- Daniel Toll

## 4.4 Resources

a programmer - Ruslan Abdulin
a poject manager - Ruslan Abdulin
a tester - Ruslan Abdulin

1 laptop for programming, designing and keeping track of the project
1 computer for testing the result always on Internet connection
20 hours of the project manager to design a project
45 hours of the programmer to develop the game
15 hours of the tester to test the product
10 hours of the programmer to write documentation

## 4.5 Hard- and Software Requirements

To develop:
Processor Intel Core i5
Memory 6Gb of RAM
Solid State Drive 256Gb
Operating system Windows 7 or higher
Sofwtware:
Java 8 SDK
Libre Office
Notepad ++
Eclipse IDE

To run:
Java 8
Windows
Windows 10 (8u51 and above)
Windows 8.x (Desktop)
Windows 7 SP1
Windows Vista SP2
Windows Server 2008 R2 SP1 (64-bit)
Windows Server 2012 and 2012 R2 (64-bit)

RAM: 128 MB
Disk space: 124 MB for JRE; 2 MB for Java Update
Processor: Minimum Pentium 2 266 MHz processor

Mac OS X
Intel-based Mac running Mac OS X 10.8.3+, 10.9+

Linux
Oracle Linux 5.5+1
Oracle Linux 6.x (32-bit), 6.x (64-bit)2
Oracle Linux 7.x (64-bit)2 (8u20 and above)
Red Hat Enterprise Linux 5.5+1, 6.x (32-bit), 6.x (64-bit)2
Red Hat Enterprise Linux 7.x (64-bit)2 (8u20 and above)
Suse Linux Enterprise Server 10 SP2+, 11.x

Suse Linux Enterprise Server 12.x (64-bit)2 (8u31 and above)
Ubuntu Linux 12.04 LTS, 13.x
Ubuntu Linux 14.x (8u25 and above)
Ubuntu Linux 15.04 (8u45 and above)
Ubuntu Linux 15.10 (8u65 and above)

## 4.6 Overall Project Schedule

- 1 - 6 feb - creating the project pan
- 6-8 feb - creating skeleton of code Basic functions: Text-based interface Test data base of words Picking up words and display the number of dashes Drawing the hangman Right-Wrong guessing Points calculating
- 8 feb - deadline for delivering release 1
- 9-12 feb - modelling new features using UML Features: Menu Registration Name checking Difficulty options High-Score database
- 12 feb - 17 feb Implementing the features
- 18 feb- 21 feb - Creating documentation
- 21 feb deadline for delivering release 2
- 21feb - 25feb - code review
- 26feb - 28feb - planing tests
- 29feb-6mar perform testing
- 7 mar - documenting tests
- 8 mar deadline for delivering release 3
- project plan review
- minor bugfix
- final testing

- complete documentation
- deadline for delivering release 4

## 4.7 Scope, Constraints and Assumptions

Main features to be implemented:
Drawing the hangman in text mode
Alphabet shows what the letters were tried
Messages to user if a letter was correct or not

Additional features:
Menu - options - player
Hihg score table
Player registration
Saving/Loading the game
Settings - diffuculty
Short word deffinition or topic

Constraints:

- The process of the development is divided into 4 iterations.
- GitHub must be created and users dntoll, tobias-dv-lnu, DTAG-lnu, marietherese and kennyek must be added as coloborators.
- All the files should be prepared and release should be made before the deadline.
- All the code should be documented/commented as fully as possible.
- The structure should be simple and understandable. All the files should have descriptive names.
- Every iteration should contain a time-log.
- Diagrams and figures should go in files with descriptions and not to be standalone.

- The project - plan should be full and up-to-date
- all the releases should be made strict before the deadline.

## 4.8 Project Plan Reflections

A Project Plan is a major document in plan-driven development. The plan sets goals and restrictions and mark up the resources available to the project, the work breakdown, and a schedule for carrying out activities. The details of project plans vary depending on the type of project and organization.

# 5 Iterations

Plan for four iterations, including this. This is a fine-grained plan on what is to be done in each iteration and with what resources. To begin with, this is a plan of what we expect to do, update this part with additions (never remove anything) when plans do not match up with reality. Also make time estimates for the different parts. In this course the overall planning has in some ways already been decided, so use the template to provide more details on specific tasks that define your project. Remember that you can plan to add features to any of the phases as long as the main focus is also met. The first assignment is to complete iteration one.

## 5.1 Iteration 1

The first iteration is this project plan along with some degree of implementation. Complete the documentation first so that the implementation goals are met in code. You need to implement an idea and some skeleton code for your project to work with. This is assignment one.

Iteration one includes this plan, the first version of the game with very basic functional, the table of risks and timelog.

## 5.2 Iteration 2

In this iteration you need to add some features to the game but after you have first modelled them using UML. All diagrams need to be included in the project documentation and should be implemented in the way modelled.

## 5.3 Iteration 3

You may include additional features to the game in this iteration, but the main focus is on testing. Plan, perform and document your tests in this iteration.

## 5.4 Iteration 4

The outcome of this iteration is the complete game. Reiterate the steps in iteration 1 – 3 for a set of new features but also remember to see the project as a whole, not only its parts.

# 6 Risk Analysis

Raw lists are sections 6.1 and 6.2 whereas the table which gather them presented in section 6.3

## 6.1 List of Risks

1. Key staff are ill at critical times in the project and aren't able to work on the project for a while - Low probability - Tolerable effects

2. Huge load on other projects which make it difficult to work on this project - Medium probability - Serious effects

3. The project or its requirements have changed requiring major design rework - Medium probability - Catastrophic effects

4. The laptop is broken or The information is lost - Low probability - - Catastrophic effects

5. The Internet isn't available - Low probability - Insignificant effects

6. It is difficult/time consuming to implement additional features - Medium probability - Tolerable effects

7. The time required to develop the software is underestimated - Medium probability - Serious effects

8. The code generated by software code generation tools is Inefficient - Low probability - Insignificant effects

## 6.2 Strategies

1. Plan and control activities in advance

2. Check the list of deadlines to estimate time better; Plan other activities in advance

3. Check the requirements, arrange tutoring sessions to make sure that everything is correct. Follow latest news in MyMoodle

4. Keep everything backed up, commit to GitHub after every significant change

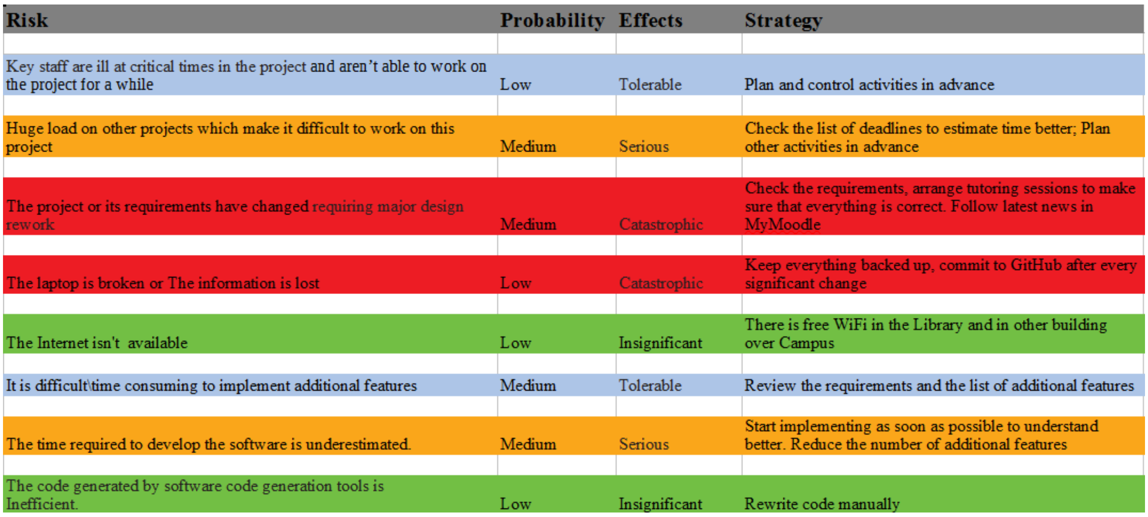
5. There is free WiFi in the Library and in other building over Campus

6. Review the requirements and the list of additional features

7. Start implementing as soon as possible to understand better. Reduce the number of additional features

8. Rewrite code manually

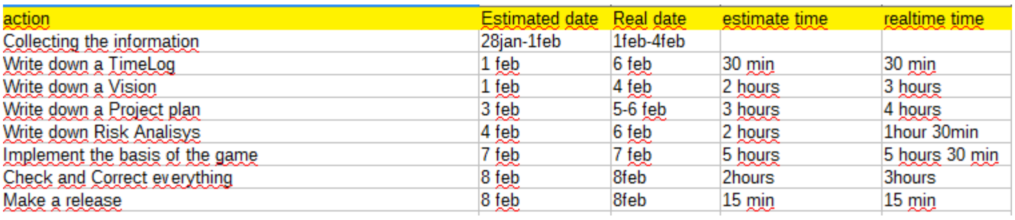
## 6.3 table of risks



| Risk | Probability | Effects | Strategy |
|---|---|---|---|
| Key staff are ill at critical times in the project and aren't able to work on the project for a while | Low | Tolerable | Plan and control activities in advance |
| Huge load on other projects which make it difficult to work on this project | Medium | Serious | Check the list of deadlines to estimate time better; Plan other activities in advance |
| The project or its requirements have changed requiring major design rework | Medium | Catastrophic | Check the requirements, arrange tutoring sessions to make sure that everything is correct. Follow latest news in MyMoodle |
| The laptop is broken or The information is lost | Low | Catastrophic | Keep everything backed up, commit to GitHub after every significant change |
| The Internet isn't available | Low | Insignificant | There is free WiFi in the Library and in other building over Campus |
| It is difficult\time consuming to implement additional features | Medium | Tolerable | Review the requirements and the list of additional features |
| The time required to develop the software is underestimated. | Medium | Serious | Start implementing as soon as possible to understand better. Reduce the number of additional features |
| The code generated by software code generation tools is Inefficient. | Low | Insignificant | Rewrite code manually |

*the original table in doc folder

# 7 TimeLog



| action | Estimated date | Real date | estimate time | realtime time |
|---|---|---|---|---|
| Collecting the information | 28jan-1feb | 1feb-4feb | | |
| Write down a TimeLog | 1 feb | 6 feb | 30 min | 30 min |
| Write down a Vision | 1 feb | 4 feb | 2 hours | 3 hours |
| Write down a Project plan | 3 feb | 5-6 feb | 3 hours | 4 hours |
| Write down Risk Analisys | 4 feb | 6 feb | 2 hours | 1hour 30min |
| Implement the basis of the game | 7 feb | 7 feb | 5 hours | 5 hours 30 min |
| Check and Correct everything | 8 feb | 8feb | 2hours | 3hours |
| Make a release | 8 feb | 8feb | 15 min | 15 min |

*the original timelog table in doc folder