




Dungeon Battle

A rogue-like game

 5 people

Project Description

Dungeon crawls, or more specifically rogue-likes are turn-based computer games in which a single player evolves through a procedurally generated dungeon, fighting creatures, finding various objects, acquiring experience. The final objective may be to reach the top (or bottom, if the dungeon is a cave) of the dungeon and come back, or to find a special object. Death is typically permanent, and plays can be very short, especially for beginners.

Skills

Scenario

Real Time programming

GUI

(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 0 Where everything must start.

- + 500 xp Management
Project created on project management platform (e.g. github).
- + 500 xp Communication
Mailing list for developers, with all developers subscribed.
- + 500 xp Compilation
Standard build system, documented in a README/INSTALL file.

Level 1 You may now pursue to the level 1 of the project.

- + 1000 xp Documentation Required for lvl 1 validation
Source code documentation generated from build system and documented code.
- + 1000 xp Bug fighting Required for lvl 1 validation
Testing framework integrated in build system, including tests.
- + 500 xp Verification Required for lvl 1 validation
Continuous integration for all relevant build targets, including tests.
- + 1000 xp Board
The game is played on a series of dungeon levels which are simple square grids. The grid cells may be empty, walls, or floor.
- + 3000 xp Graphics
The graphics can either be 2D or text based. There is a window where the board is displayed.
- + 1000 xp Controls
The player is controlled with zqsd and can move through the level.
- + 1500 xp Monsters
Monsters appear on the map, and move along some predefined path.
- + 2000 xp Fighting
The hero can attack monsters, and kill them.
- + 1500 xp Experience
The hero can progress and level up doing more and more damages the more monsters he kill.
- + 1500 xp Intelligent Monsters
Monsters will go towards the hero, without getting stuck.



Dungeon Battle

A rogue-like game



5 people

Project Description

Dungeon crawls, or more specifically rogue-likes are turn-based computer games in which a single player evolves through a procedurally generated dungeon, fighting creatures, finding various objects, acquiring experience. The final objective may be to reach the top (or bottom, if the dungeon is a cave) of the dungeon and come back, or to find a special object. Death is typically permanent, and plays can be very short, especially for beginners.

Skills

Scenario

Real Time programming

GUI

(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 2

Level 1 must be unlocked to read this section

- + 1500 xp **Developer Documentation** Required for lvl 2 validation
Document your project (not necessarily only in the source code) so that a newcoming developer could understand and contribute to the code.
- + 500 xp **Release** Required for lvl 2 validation
Produce a release as a source archive or git tag. The release files should have up-to-date README and INSTALL files and more generally allow anyone to deploy the application.
- + 1000 xp **Package** [optional]
Project packaging (e.g. debian, opam).
- + 1500 xp **More NPCs**
The game should feature more than one kind of non-playable character: NPCs should have different appearances and characteristics, e.g. strength, specific movement or resistance. Some NPCs should be friendly, e.g. a dog that follows the character and attacks nearby enemy NPCs.
- + 500 xp **Body snatch**
The player can acquire a spell by picking up a special item. The spell allows the player (once) to exchange his character with another one. After the exchange the player controls the other character and has his characteristics, and conversely.
- + 3000 xp **Multiplayer** [optional]
Multiple players can play together on different computers.
- + 1000 xp **Save/restore**
Make it possible to save and restore a game state.
- + 1500 xp **Lighting**
All characters should have a limited field of vision. A decent lighting algorithm should be used to determine it. It should be developed in a TDD way.
- + 2000 xp **Dynamic world generation**
The dungeon level should not always be created upon initialization but possibly lazily as the player explores it. Use this capability to create an infinite level, a potion resetting the level, and optionally a dynamic worst-case labyrinth.
- + 1500 xp **Story mode**
A notion of story should be created to trigger events based on conditions such as meeting a specific character, acquiring enough experience, etc. so as to represent a quest, with possible side quests. The story should include text that is displayed on screen. A boss should be implemented in a special (dynamically generated) room where the boss periodically moves to the next room, until he is killed, at which point the next room is an exit.
- + 3000 xp **Ascii mode** [optional]
Text-based UI with the same capabilities as the graphical UI. Unicode and colors are allowed.
- + 1000 xp **Memory profiling**
Profile the memory usage of the application, on a specially created game instance that allocates a lot. Fix what needs fixing until a reasonable result is obtained. Demo the evaluation and fixes.



Dodge man

A one screen game



3/4 people

Project Description

The player is stuck inside the window, and objects are falling from the sky. If the player is touched, he dies, and if a dodges an object, the object disappears after touching the ground. The longest the player stays alive, the highest the score, but the hardest it gets.

Skills

GUI

Real time programming

(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 0

Where everything must start.

- + 500 xp Management
Project created on project management platform (e.g. github).
- + 500 xp Communication
Mailing list for developers, with all developers subscribed.
- + 500 xp Compilation
Standard build system, documented in a README/INSTALL file.

Level 1

You may now pursue to the level 1 of the project.

- + 1000 xp Documentation Required for lvl 1 validation
Source code documentation generated from build system and documented code.
- + 1000 xp Bug fighting Required for lvl 1 validation
Testing framework integrated in build system, including tests.
- + 500 xp Verification Required for lvl 1 validation
Continuous integration for all relevant build targets, including tests.
- + 3000 xp Window
A character,a background, and objects can be rendered
- + 1500 xp Objects
Objects are generated in semi random fashion, and are falling to the ground
- + 1000 xp Score
There is a score system, with high score and leaderboard.
- + 2000 xp Gameplay
The more the player progresses, the harder the game gets. This imply to have the objects falling in a smart way which can support the acceleration.
- + 500 xp Controls
The character can be controlled with q/s/d, left/right/crouch
- + 1500 xp Death
When hit, the character dies.
- + 500 xp Jump
The character can jump.
- + 1000 xp Animations
The character is animated, with several images for each actions



Dodge man

A one screen game



3/4 people

Project Description

The player is stuck inside the window, and objects are falling from the sky. If the player is touched, he dies, and if a dodges an object, the object disappears after touching the ground. The longest the player stays alive, the highest the score, but the hardest it gets.

Skills

GUI

Real time programming

(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]


Level 2 Level 1 must be unlocked to read this section

- + 1500 xp **Developer Documentation** Required for lvl 2 validation
Document your project (not necessarily only in the source code) so that a newcomers developer could understand and contribute to the code.
- + 500 xp **Release** Required for lvl 2 validation
Produce a release as a source archive or git tag. The release files should have up-to-date README and INSTALL files and more generally allow anyone to deploy the application.
- + 1000 xp **Package** [optional]
Project packaging (e.g. debian, opam).
- + 1000 xp **Disorientation**
Include a mode where the screen is flipped (horizontally and/or vertically) and one where the screen is continuously rotated. Controls are unchanged: this is only a modification of the display. It could be triggered per-level or by a special item.
- + 1000 xp **Multiplayer**
Make it possible for two players to play on the same computer, each with his own character. Player characters can collide. The last Tim standing wins.
- + 1500 xp **Moving platforms**
Include moving parts in some levels, such as platforms or elevators.
- + 1500 xp **Performance evaluation**
Evaluate the performance of the game during a play, in space and optionally in time. Fix what needs fixing, until a good result is obtained. Demo the evaluation, and the fixes.
- + 2000 xp **Replay**
Make it possible to replay a game, e.g. when selecting it among high-scores. Optionally, export a video.
- + 3000 xp **Time manipulation**
Add the ability to slow down or invert the flow of time for non-player objects. Inversion should re-create destroyed items. The effect could be triggered by the player, after that ability has been gained through special items.
- + 1500 xp **Liquids**
Add a level where the player floats/swim in some liquid. Waves should be induced by the player and falling objects. The liquid level may be affected by special items.
- + 1500 xp **Complex collisions**
Add polygonal non-player objects, e.g. falling Tetris pieces. Handle collisions of non-circular objects, inducing torsors.



Paint is not dead

A drawing software

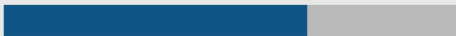
 4/5 people

Project Description

This software enables the user to draw pictures on its computer. It can be freehand drawing, but the drawing may also be guided by the computer in order to draw specific shapes (circles, squares...). It can contain more advanced features, with for example edge detections and automatic color filling.

Skills

Image manipulations



GUI



(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 0

Where everything must start.

- + 500 xp Management
Project created on project management platform (e.g. github).
- + 500 xp Communication
Mailing list for developers, with all developers subscribed.
- + 500 xp Compilation
Standard build system, documented in a README/INSTALL file.

Level 1

You may now pursue to the level 1 of the project.

- + 1000 xp Documentation Required for lvl 1 validation
Source code documentation generated from build system and documented code.
- + 1000 xp Bug fighting Required for lvl 1 validation
Testing framework integrated in build system, including tests.
- + 500 xp Verification Required for lvl 1 validation
Continuous integration for all relevant build targets, including tests.
- + 3000 xp Freehand drawing
A simple window where one can draw with point and click.
- + 500 xp GUI
A user interface allows to create a new drawing of specified size, and also to select the drawing tool.
- + 1000 xp Save
The user can save and load its drawing.
- + 1000 xp Shapes
A tool assists in the creation of some shapes, like squares and circles, with a live pre-visualization.
- + 1500 xp Antialiasing
An algorithm allows to reduce aliasing in small resolution.
- + 1000 xp Auto select
The user can automatically select a part of the drawing which is of the same color.
- + 1000 xp Filling
It is possible to fill a selected space with a color or a gradation.
- + 1000 xp Types
When drawing a shape or a line, the user can select different options, such as thickness, color, filling,...



Paint is not dead

A drawing software



4/5 people

Project Description

This software enables the user to draw pictures on its computer. It can be freehand drawing, but the drawing may also be guided by the computer in order to draw specific shapes (circles, squares...). It can contain more advanced features, with for example edge detections and automatic color filling.

Skills

Image manipulations



GUI



(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 2 Level 1 must be unlocked to read this section

- + 1500 xp **Developer Documentation** Required for lvl 2 validation
Document your project (not necessarily only in the source code) so that a newcoming developer could understand and contribute to the code.
- + 500 xp **Release** Required for lvl 2 validation
Produce a release as a source archive or git tag. The release files should have up-to-date README and INSTALL files and more generally allow anyone to deploy the application.
- + 1000 xp **Package** [optional]
Project packaging (e.g. debian, opam).
- + 2000 xp **Layers**
The objects can be on different layers. Some layers can be hidden, duplicated, merged... Different mode of compositions between layer should be available (transparency, difference, mask,...)
- + 1000 xp **Effects**
Effects can be applied to a layer or an object, with for example a blur.
- + 2000 xp **History**
It is possible to display the history of modifications, and come back to previous versions.
- + 3000 xp **Team Effort**
In the spirit of etherpad, a server host each drawing at a specific url, and several people can edit at the same time a drawing, with an historic of modifications made by who.
- + 1500 xp **Performance**
Test and demonstrate the performances in term of space and time, with huge drawings or quick actions.
- + 1500 xp **Smart Paint TDD**
Through test driven development, implement a function which can detect if there is a square like shape in the image.



Click and Run

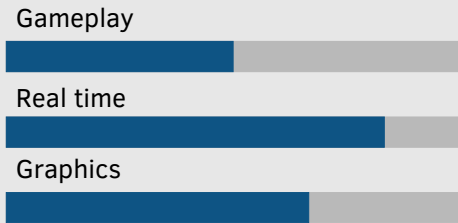
A canabalt-like game

3/4 people

Project Description

In this game, a player-controlled character is constantly running through a world, always in the same direction. The only control that the player has is to make the character jump to avoid various sorts of threats. The objective is to run for as long as possible. The world is generated procedurally and on the fly.

Skills



(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 0 Where everything must start.

- + 500 xp Management
Project created on project management platform (e.g. github).
- + 500 xp Communication
Mailing list for developers, with all developers subscribed.
- + 500 xp Compilation
Standard build system, documented in a README/INSTALL file.

Level 1 You may now pursue to the level 1 of the project.

- + 1000 xp Documentation Required for lvl 1 validation
Source code documentation generated from build system and documented code.
- + 1000 xp Bug fighting Required for lvl 1 validation
Testing framework integrated in build system, including tests.
- + 500 xp Verification Required for lvl 1 validation
Continuous integration for all relevant build targets, including tests.
- + 1000 xp Basic game mechanics
Display a character running through an empty world, with some control that allows to jump.
- + 2500 xp Procedural world generation and display
Generate an infinite world with a fair amount of variety, and display it on the start screen, which should be scrolling from left to right through the world.
- + 1500 xp Complete game mechanics
Put the previous two items together and detect death conditions, at least collision but perhaps also falling out of the screen.
- + 500 xp Menu
The start screen should still display a generated world, and also display a menu for starting a new game; after death, the start screen is displayed again.
- + 1000 xp Animation
Animate the character, its jumps, and perhaps its death(s).
- + 2000 xp Background
Generate an infinite background for the world, and display it with a slower scroll, simulating a parallax effect.
- + 1000 xp Score
Compute a score, display it during the game, and have a leader board which prompts for a player name when a new high-score is reached.
- + 1000 xp Adaptative difficulty
Introduce new difficulties, in the form of new items or landscape features, when the score reaches predefined levels.



Click and Run

A canabalt-like game



3/4 people

Project Description

In this game, a player-controlled character is constantly running through a world, always in the same direction. The only control that the player has is to make the character jump to avoid various sorts of threats. The objective is to run for as long as possible. The world is generated procedurally and on the fly.

Skills

Gameplay



Real time



Graphics



(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 2

Level 1 must be unlocked to read this section

- + 1500 xp **Developer Documentation** Required for lvl 2 validation
Document your project (not necessarily only in the source code) so that a newcoming developer could understand and contribute to the code.
- + 500 xp **Release** Required for lvl 2 validation
Produce a release as a source archive or git tag. The release files should have up-to-date README and INSTALL files and more generally allow anyone to deploy the application.
- + 1000 xp **Package** [optional]
Project packaging (e.g. debian, opam).
- + 500 xp **Adaptative speed**
Change the scrolling and running speed depending on the score or its derivative.
- + 1000 xp **Items**
Add special items that can be picked (by running over them): one that instantly kills the player, one that gives an extra life (displayed on a life counter on screen), one that kills the player unless an antidote is picked within 20 seconds. Add items for temporarily slowing down or speeding up.
- + 500 xp **Visual effect items**
Add items that temporarily cause visual effects: one for reducing the visibility (affecting the luminosity and contrast) and one for changing the colors (perhaps while at the same time making the player immune to enemies, cf. next item).
- + 2000 xp **Enemies**
Add enemies to the map. A sort of enemy should walk towards the player character, another jump following a predictable pattern.
- + 500 xp **Theme**
Create an alternative theme which changes the way the player and world are displayed. This should be more than a simple change of color.
- + 1000 xp **Two player mode**
Add a two-player mode where players are on the same machine, each one controlling a character.
- + 1000 xp **Gravity**
Change the strength of gravity based on special items or locations. Add a control for inverting the gravity; in two player mode it should be shared by both players. For this to be interesting, the map should also contain platforms on the roof.
- + 2000 xp **Replay**
Add the possibility to replay past games. For instance, high score runs could be displayed on the start screen.
- + 1000 xp **Performances**
Evaluate the performances of the game, both in space and time, using profiling, and demonstrate the limits at high speed with many enemies.



Hungry penguins

Strategy board game



2/3 people

Project Description

In the board game “Hey, that’s my fish!”, several players play turn-by-turn, moving at each turn one of their penguins on a hexagonal grid. After moving, the penguin eats the fish present on the cell where he has arrived, and the cell he left thaws. The game is over when nobody can move anymore. The goal is to have eaten as much fish as possibly by then. The complete rules can easily be found online.

Skills

Graphics

Algorithms

(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 0 Where everything must start.

- + 500 xp Management
Project created on project management platform (e.g. github).
- + 500 xp Communication
Mailing list for developers, with all developers subscribed.
- + 500 xp Compilation
Standard build system, documented in a README/INSTALL file.


Level 1 You may now pursue to the level 1 of the project.

- + 1000 xp Documentation Required for lvl 1 validation
Source code documentation generated from build system and documented code.
- + 1000 xp Bug fighting Required for lvl 1 validation
Testing framework integrated in build system, including tests.
- + 500 xp Verification Required for lvl 1 validation
Continuous integration for all relevant build targets, including tests.
- + 1000 xp Text-based display
Define the datatype for the board, a game state, and display it in a text-based fashion.
- + 1000 xp Multiplayer game
Implement a multiplayer game (maybe text-based) where all players play on the same machine.
- + 2500 xp Graphical interface
A graphical UI should allow to start a game, selecting the number of players, number of penguins per player, and grid size using a menu, then play the game in multiplayer.
- + 500 xp Animation
Animate penguins when they wait, move, and perhaps eat.
- + 3000 xp End-game evaluation
When a connex piece of the board belongs to a single player, compute exactly how much fish he can eat there.
- + 2500 xp Computer player
Implement computer-controlled players: first a greedy one; then think of a smarter strategy.



Hungry penguins

Strategy board game

 2/3 people

Project Description

In the board game “Hey, that’s my fish!”, several players play turn-by-turn, moving at each turn one of their penguins on a hexagonal grid. After moving, the penguin eats the fish present on the cell where he has arrived, and the cell he left thaws. The game is over when nobody can move anymore. The goal is to have eaten as much fish as possibly by then. The complete rules can easily be found online.

Skills

Graphics



Algorithms



(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 2 Level 1 must be unlocked to read this section

- + 1500 xp **Developer Documentation** Required for lvl 2 validation
Document your project (not necessarily only in the source code) so that a newcoming developer could understand and contribute to the code.
- + 500 xp **Release** Required for lvl 2 validation
Produce a release as a source archive or git tag. The release files should have up-to-date README and INSTALL files and more generally allow anyone to deploy the application.
- + 1000 xp **Package** [optional]
Project packaging (e.g. debian, opam).
- + 2000 xp **Client-server**
The game should be played using a client-server infrastructure. The protocol should be the same for all “Hungry Penguins“ projects.
- + 500 xp **Server-side game management**
Clients should be able to create new games on the server, with various settings such as number of players, password, etc. If the game settings allow it, a client may disconnect and reconnect later to play its move.
- + 500 xp **Server-side user management**
The server has a notion of user, keeps a global score based on the games played by the user, and can display a ranking of users.
- + 1000 xp **Tournament**
The server can spontaneously propose games to idle connected clients. A tournament can be organized in this way, matching users depending on their kind (humain/AI) and level (global score).
- + 1000 xp **Puzzle mode**
A single-player (non-networked) puzzle mode challenges users to achieve a specific task on a specific map: eat 10 fishes, reach this spot, survive as long as possible, etc.
- + 1500 xp **Dynamic cells**
In the puzzle mode, special cells can move or alter their features depending on external events. Using this, implement a pseudo-randomly floatting cell, a door/switch system, a ferry for passing a river.
- + 2000 xp **Puzzle solvers**
Puzzles should feature solvers, at least for the three kinds of puzzles listed above, and for levels including the kinds of dynamic cells listed above. Solvers can be invoked to find and display a solution when the player gives up.
- + 1500 xp **Map editor**
The GUI should make it possible to create maps: select where the floats are, how many fishes they contain, etc. It should also be possible to edit maps with special cells for the puzzle mode.



\$\$\$

Accounting for friends



2 people

Project Description

The software should allow to keep track of money spent between friends, e.g. during a holiday. It should help them to settle their accounts.

Skills

Web



Databases



(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 0

Where everything must start.

- + 500 xp Management
Project created on project management platform (e.g. github).
- + 500 xp Communication
Mailing list for developers, with all developers subscribed.
- + 500 xp Compilation
Standard build system, documented in a README/INSTALL file.

Level 1

You may now pursue to the level 1 of the project.

- + 1000 xp Documentation Required for lvl 1 validation
Source code documentation generated from build system and documented code.
- + 1000 xp Bug fighting Required for lvl 1 validation
Testing framework integrated in build system, including tests.
- + 500 xp Verification Required for lvl 1 validation
Continuous integration for all relevant build targets, including tests.
- + 2000 xp Basic features
A command-line tool should allow to: create a new database; create payers; add a transaction; view past transactions and current balance.
- + 2000 xp Resolution
The tool should propose solutions to balance the account: optimizing either the total number of transactions, or the maximum number of transaction per user.
- + 4000 xp Web interface
All of the above features should be usable through a web interface. The web service should require to log in. Each user of the service can create various projects, and grant rights to other users on each project (addition/deletion/modification of transaction items).
- + 1000 xp Web API
The web service should provide a RESTful API, so that a command-line tool (equivalent to the first one) can be made to use the system.
- + 1000 xp PDF report
Generate a PDF showing a transaction log, balance, and resolution.



\$\$\$

Accounting for friends



2 people

Project Description

The software should allow to keep track of money spent between friends, e.g. during a holiday. It should help them to settle their accounts.

Skills

Web

Databases

(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 2 Level 1 must be unlocked to read this section

- + 1500 xp **Developer Documentation** Required for lvl 2 validation
Document your project (not necessarily only in the source code) so that a newcoming developer could understand and contribute to the code.
- + 500 xp **Release** Required for lvl 2 validation
Produce a release as a source archive or git tag. The release files should have up-to-date README and INSTALL files and more generally allow anyone to deploy the application.
- + 1000 xp **Package** [optional]
Project packaging (e.g. debian, opam).
- + 2000 xp **Reactive User Interface**
When entering a username, a dropdown list should allow to select the user directly. Update made serverside should be instantly displayed on the user browser.
- + 2000 xp **Handling events**
Tag transactions as being part of events, display events as single transactions that can be detailed upon request. The creation of an event should allow to create in one page many sub transactions, with different repartitions. The deletion of an event should delete the transactions.
- + 1500 xp **Multiple currencies**
The transactions, balance, etc should be expressible with multiple currencies. The exchanges rates should be captured in live on the web.
- + 3000 xp **Peer to peer** [Optional]
Several instances of the website can be ran on different servers, and they synchronize between themselves through the Rest API.
- + 500 xp **Resolution by TDD**
Implement the smart resolution algorithm in a test driven way.
- + 1500 xp **Performances**
Demonstrate the performances by creating instances with as many transactions and users as possible.