Team AC: Daniil Gannota, German Mikultski, Aksel … , Gabriel Vanca .

**Introduction**

The main aim of our project is to teach and inspire a complete beginner to learn basic programming skills. Therefore, there are three main areas we concentrated on: simplicity of the material, general entertainment experienced and independence from any particular language (the teaching materials should visually illustrate abstract notations and rules with no accent upon a singular language such as C++ or Java).

Our team has decided to make those three points to be our main as due to the fact that this is the way we would like to had been taught, initially. In our view, this approach will encourage for further computer science research or at least opens the basics of computing to beginners; thus, help them learning any programming language later.

The way our project follows the requirements and corresponds to what is stated on Moodle is by having a Unity program, a game. The game, named 'Last Hope', is a visualiser of such abstract statements as “if” or “while” loops, etc. The application tries to educate a person, who is completely new to programming, those basic logic statements showing the result displayed in a game: it has a character – that is affected by the commands – and a field/board full of obstacles to be overcame. The purpose is to make use of “if” or “while” commands so that the user understands the logic behind all these rules.

As the user completes levels, they become more and more complex. At the same time, we introduce other logic statement, for example, “do while” or “if else”. As a result, we intend to make users think and develop their understanding of the concept behind it, increasing the number of steps that have to be predicted in order to complete new levels.

Furthermore, the application should inspire young minds to compute. This is due to the fact that as they play the game and see the way programming affects the behaviour of the character, they might start wondering of other ways that programming can be used as well as research the way our application is made. Our team concluded such behaviour based on the personal feelings about the game and project itself.

**MoSCoW and Team Management**

In order to be as efficient as possible, we first established features that ‘must be’, ‘should be’ or ‘could be’ as well as ‘will not be’ in the project to prioritise the objectives and key factors of the game.

**General description:** a game, where movements of a player are controlled by dragging and dropping specific blocks, representing pieces of code, e.g. "goForward" function that will make the character to move forward.

**Must:**

* See the map split into cells (fields)
* Some cell inaccessible (obstacles)
* Compose code from drag and drop blocks
* Characters behaviour determined by the code blocks
* Press "Run" button to run code
* Combine "walk", "turn", "if", "while"
* Complete levels by reaching a point
* Decide on platform

**Should:**

* Add "if else", "do while"
* Have fancy graphics
* Be able to save game progress
* Display C code + some explanation
* Hint/model answer if stuck
* "Jump", "Attack" functions  Could:

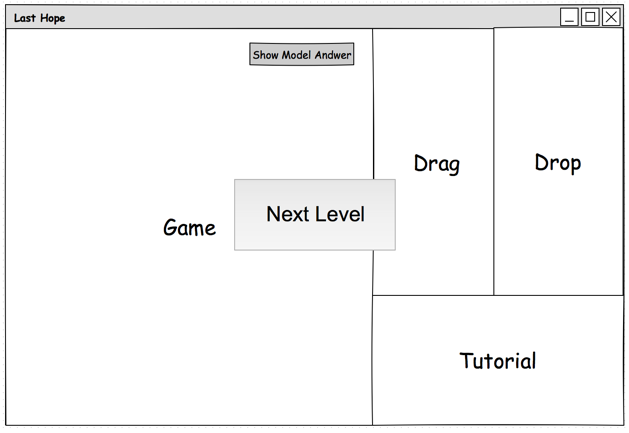
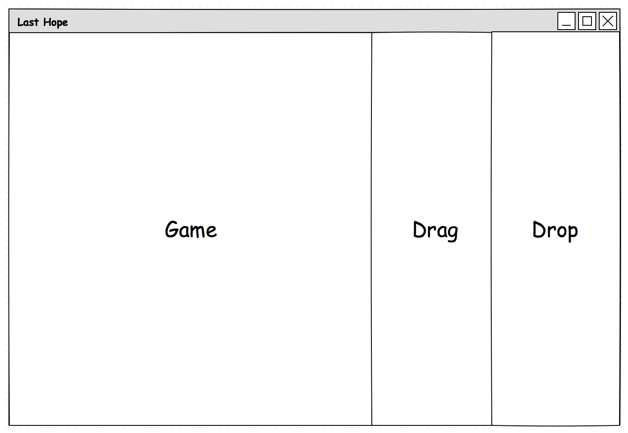
**Could:**

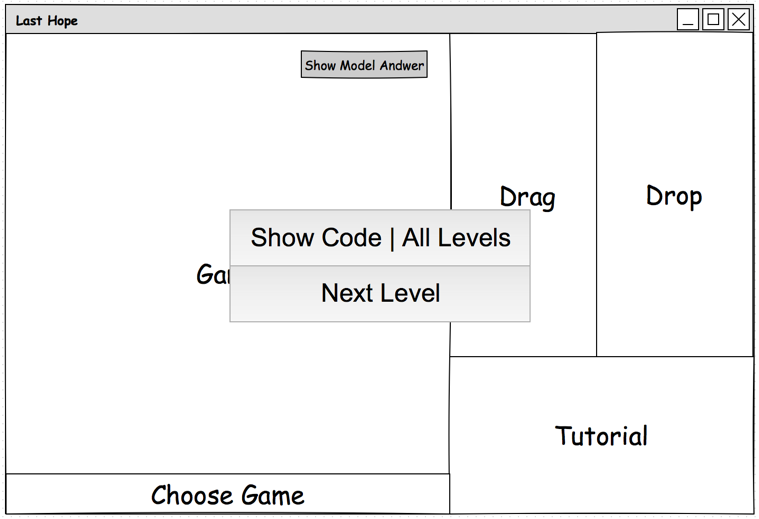
* Add "compile" feature
* Tutorial – description of each statement in programming world
* Support different languages: Java, C#…
* Several points to reach - rating of how well performed
* Several textures for the environment/player
* When the user enters certain obstacles, animation occurs
* Ability to see ‘Model Answer’ – easiest way to complete each level

**Won't:**

* Must not be too complex in terms of programming
* Is not based on one language

Furthermore, we created user interfaces that reflect each major iteration. Note that those steps have been changed throughout the project and only the final version is attached. Steps 1, 2 and 3, respectively.





Last but not least, we conducted our work by adding each task to a task manager (Todoist), where each person is assigned a certain piece of work and everyone else is able to track team progress. This and other software that provides multiple users sharing capabilities of files and code, such as Google Drive or GitHub, enabled us to make team coding much more efficient.

**Current State and Future Steps**

Currently, we have completed the most difficult part of our work. That is creating the base for each new level: starting menu, ‘drag/drop’ functions, and character behaviour as well as all obstacles and decorations. This allows us to conveniently use those basic features in the development of new levels, which will only require a creation of new maps.

Our next steps are to add more features to the user interface as well as more teaching materials that would describe each piece of code or showed, for instance, “if else” statements in C# or Java. Therefore, we intend to not only show the abstract concept of logic statements, but also relate to some programming languages. This is not to say the application will be singular computing language based, we are enthusiastic to implement a lot more examples. Thus, by pushing ‘Show Code’ button the user will be able to select preferable language(s) (e.g. Java or C) and see the actual code in it.

Finally, we are inspired to continue the development of the project even after the final submission. There are a few features that may take a lot more time to be implemented. However, despite our huge plans, good planning sets the course of the work so that we primarily deliver finished product, and only then iterate new features.