Casters & Coders

By: sddec23-13

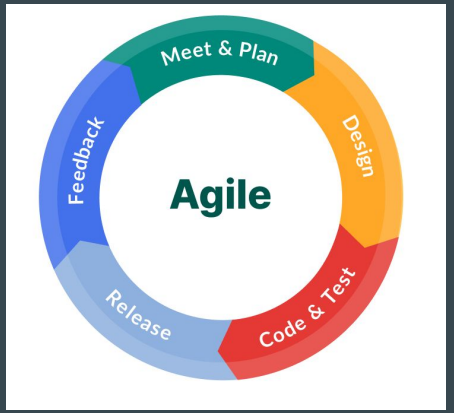
Problem Statement

Learning programming with little to no background knowledge can be a difficult mountain to climb, so we set out to create a game which:

- Has coding as a main gameplay mechanic
- Teaches basic programming concepts
- Requires little to no background knowledge
- Gamefies programming to make the process of learning raw technical information more fun and less frustrating
- Is fun to play

Project Management

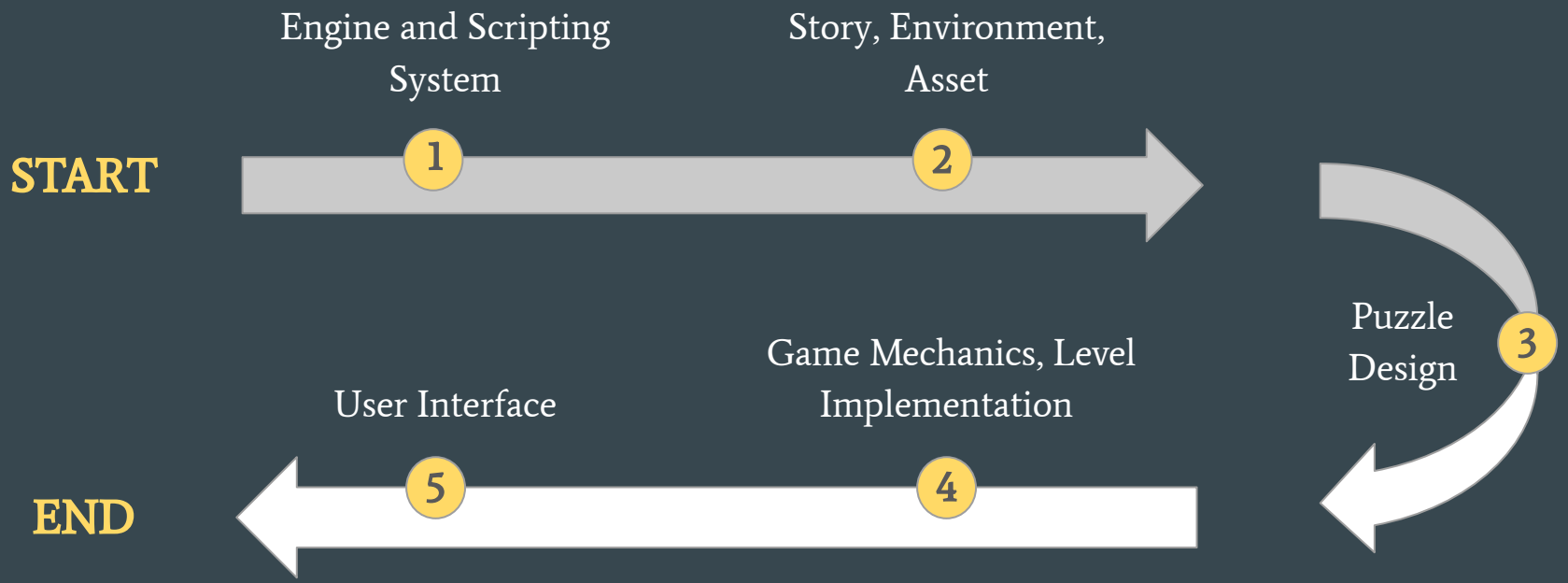
- We will adhere to the agile project management style
- With a video game there will be a lot of testing and debugging
- Requirements are just guidelines
- Features can be dropped and added throughout the project



Task Decomposition

- Engine and Scripting System
- Story, Environment, Design and Asset Creation
- Puzzle Design
- Mechanics of the Game and Level Implementation
- User Interfaces

Project Milestones



Project Timeline

- This chart assumes a sixteen-week timeline and 1-week sprints
- Some subtasks have strict dependencies on other subtasks.

  These are factored into the chart and denoted in parentheses by the subtask.
- Some tasks are expected to be finished before others

Task/Subtask	Sprint															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Engine and Scripting System																
1.1 Prototype scripting systems	X	X														
1.2 Choose engine, tool, and language			X													
1.3 Design scripting API			X													
1.4 Scripting MVP				X	X											
1.5 Improvements						X	X	X	X							
Story, Environment Design, and Asset Creation																
2.1 Storyline	X	X	X													
2.2 Game Map (3.2)				X	X	X										
2.3 Dialogue							X	X	X	X						
2.4 Source assets											X	X	X			
Puzzle Design																
3.1 Puzzle framework	X	X														
3.2 Curriculum	X	X														
3.3 Introductory puzzles		X	X	X												
3.4 Exploratory puzzles					X	X	X									
3.5 Challenge puzzles								X	X	X						
3.6 User feedback and polish								X	X	X	X	X	X			
Mechanics																
4.1 Player Controller	X	X														
4.2 Tileset environment	X	X														
4.3 Interactable objects			X	X	X											
4.4 Single puzzle & systems (1.4)(5.2)						X	X	X								
4.5 Additional puzzles									X	X	X	X	X	X	X	X
User Interfaces																
5.1 Style Guide	X															
5.2 IDE MVP		X	X	X												
5.3 Dialogue system					X	X	X	X								
5.4 Game menus									X	X	X	X				
5.5 IDE Polish													X	X	X	X

Risks and Risk Management

1. Engine and Scripting System.
    a. We need to make sure early in development that the game engine and scripting system are reliable and robust.
2. Story, Environment Design, and Asset Creation.
    a. The storyline may see some rewriting to fit the key concepts we are trying to teach.
3. Puzzle design
    a. We need to make sure that the puzzles are difficult to stay fun, while not being too difficult and frustrating the player.
    b. Puzzles must build off of each other and reuse old concepts to solidify the player's understanding of coding.
4. Mechanics of the Game and level implementation;
    a. When putting together many disparate systems from other tasks, we may run into unforeseen bugs, causing delays.
    b. We need to make sure that the core of the game's systems and mechanics use coding concepts with the intent of teaching people said coding concepts.
    c. It is likely that we will run into time constraints when trying to implement all the designed puzzles.

Personal Effort Requirements

Engine and Scripting System
104 hours

Story, Environment Design, and Asset Creation
56 hours

Puzzle Design
166 hours

Mechanics
130 hours

User Interfaces
78 hours

The End

...