Box God

(Originally titled 'You Raise Me Up')

Matias Rietig (mjr9r), Seven Starosta (sbs3bx), Corey Chen (czc3wa), Matt Lee (mjl4nk)

Github Repository:
https://github.com/UVA-CS4730-F17/game-project-you-raise-me-up

Game Pitch

Most commercial platformer games demand the player to beat the level by jumping, running and using various other skills their character; however, in Box God, formerly titled "You Raise Me Up," the player *becomes the level*. The player will help a computer-controlled character beat puzzles by positioning platforming objects in real time, cleverly unveiling new paths to finish the level and trying to protect the character simultaneously.

How to Play

Use the mouse to click, drag, and drop items from the toolbox in the top-left hand corner of the screen into the level to help the green box reach the blue, diamond-shaped portal at the end of each level. You will get maximum stars if the box makes it through a level with a good time, keeping full health, and using as few objects as possible.

Content

Box God contains 21 levels. Players can play these levels sequentially or navigate through them via the Level Select menu. Each individual level takes around 15 seconds to complete, if and only if the player is able to complete the level the first time. While this sounds like an incredibly short experience, most levels will take players several tries to successfully complete. Level difficulty increases over the course of the game, with end-game levels being very difficult both to figure out how to solve and to execute. Players will be introduced to new mechanics as objects in their toolbox as the game goes on. These objects include planks, springs, bombs, landing pads, and reverse blocks, and all items can be used in tandem to solve the various puzzles of Box God.

Additionally, the player is incentivized to replay levels with a three-star rating system. Players gain stars upon completion of the level based on the box's amount of health remaining, number of objects used by the player, and the amount of time taken to reach the end. This system will force even advanced players to think outside of the box to get three stars in every level. Level 21 can only be reached after accomplishing level 20, and is a hidden boss level.

Playtesting Report

In a laboratory session on 11/17 we playtested Box God to other students in CS 4730. These playtesters constructively commented that our visual designs for each draggable object were not necessarily intuitive to its function. While redesigning the visuals would run counter to the visual simplicity of Box God we still took this feedback heavily into account. Several levels were then designed to teach the players the ways to utilize each object to the optimal effect. In some of these, an object will drop into the level automatically to show that the player is able to drag it into the level. In another, there is a small cutout in the stage where a specific object will fit perfectly, and will show the player how to overcome a new type of obstacle, fall damage.

Additional play testing was conducted over the course of Thanksgiving break with 11 subjects who were not students in CS 4730. These subjects were sorted based on amount of gaming experience and were rated on a scale of 1-3, 1s being non-gamers and 3s being avid gamers. Subjects were observed with no instruction as they played six different levels of the game. Quantitatively, number of attempts each subject would take to complete the level and number of stars achieved (from 1-3) upon said completion were recorded. Other qualitative observations were made as well.

Subjects with a gaming experience of 1 struggled the most with completing levels 1, 3, and 6. Subjects with a gaming experience of 2 struggled with levels 1 and 6. Subjects with experience of 3 struggled most with levels 2 and 6. Level 1 showed perhaps the steepest learning curve in the game. With no instruction from the test-giver, many subjects (particular 1 and 2-rated subjects) were unclear on how they needed to interact with the game to complete the level successfully. To remedy this, level 1 was altered to show the player that objects can be dropped into the level by having a yellow plank drop into the level at the start. This did not affect 3-rated playtesters who were easily able to grasp what they needed to do to complete the level. Levels 2 and 3 were deemed the correct amount of challenging for where the players should be in the game at that point and were left unchanged. Level 6 in this playtesting demo was by far the most difficult for any of the subjects to complete. This was to be expected, as it was designed as a level for advanced and requires additional knowledge of the game that playtesters would not have the time or levels to learn. Still, 3-rated subjects were still able to understand what was required to complete the level, but mostly struggled with execution. Later adjustments made to alter the difficulty of level 6 in this demo have made it slightly, but not much, easier.

What We Learned

The Box God team learned the importance of making both modular code as well as modular game objects such as the prefabs in unity. We ran into many problems and slowdowns in our development because we had initially made several levels using identical objects that were not created from prefabs. This meant that future revisions of the game would have to be manually implemented in several levels. The use of prefabs would also allow for better coordination of efforts between multiple developers, as the design of game objects would be more streamlined, preventing merge conflicts and allowing easier distribution of work.

Our team also learned some lessons about scope. Our initial pitch was far too broad, and even after substantial reduction in the scale of our game, we still found that there was an ample amount of content to create and bugs to fix. We found that creating more levels without fixing existing bugs could cause more problems later on due to merge conflicts in github and the need for complete level redesign.

This also led our team to learn hands-on how “perfect is the enemy of good enough,” and how Box God, even though at times flawed, would simply be best if we continued working with what we have rather than restarting from scratch. The problems exposed throughout development only highlighted the need for extensive planning of the design beforehand.