POST MORTEM

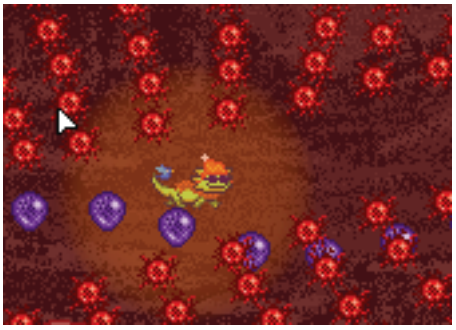
Pixelotl

Introduction

Pixelotl is a game created by Dan Lin, Scott Kulp, and Srivatsav Kunchakarra for the Video Game Design and Programming course at Rutgers University in Spring 2010. Initially, each person in the class created four or five prototypes and three, including Pixelotl, were chosen to be developed into full games.

Pixelotl is a 2D puzzle-platformer Flash game in which the player controls a salamander-like creature who is on a mission to save his friends from being eaten by a hungry stork so that he can have a party. He does this by going through fifty different levels split into five different worlds, each with different themes and challenges.

Unique in this game is the use of elastic platforms/blocks and a controllable gravitational field. While most platformers have very rigid and stationary platforms, we instead have different colored blocks which can move fluidly and be pushed around by the player. When the user holds down a key, he can increase the gravity around him, making him much heavier and triggering interactions with blocks around him.



Pixelotl using the field to raise purple blocks

Specs

Name: Pixelotl
Developer: Team XAOS
Team members: Dan Lin, Scott Kulp, Srivatsav Kunchakarra
Development time: 9 weeks
Platform: PC
Software Used: Flash, Actionscript 3, FlashDevelop, Flixel, Audacity, Gimp, TortoiseSVN, GoogleDocs
Hardware Used: Windows 7 PCs
Lines of code: 6,244
Source code files(not including Flixel): 71

What Went Right

Artwork

From the beginning, we had a very clear direction on the artwork style. Dan was well-experienced in pixel art, and so he did not need to go through many iterations for the graphics. The style for the different blocks, characters, interfaces, etc, were generally consistent throughout. The storyline slides matched the artstyle and were very well-received by our playtesters, as well.

Player sprites and different blocks

Accessories

The accessories provided the player greater motivation to save the friends and continue with the game. We were honestly quite surprised at the very large positive reaction to the accessories. They also let the user personalize the character and become more attached to it. We made adding accessories very easy and so we were able to add many different kinds of them.

Some accessory sprites and the customized player

Teamwork

We usually met three times a week in person, and then we chatted over instant messenger almost every day. Because of this, each one of us knew exactly what the others were working at the moment. We set concrete goals and expectations for each person at the beginning of each week. Each team member was very competent in their assigned tasks. Overall, we got along very well and there were no notable miscommunications or misunderstandings.

Model-View-Controller

Flixel was not designed for a Model-View-Controller framework. So, coming up with the design for the framework was tricky. However, we got this working in the first week, and afterwards, adding and removing game objects, such as blocks, friends, and other sprites, became very easy. This made the code very organized and decreased the necessary volume of code.

Overall Reaction

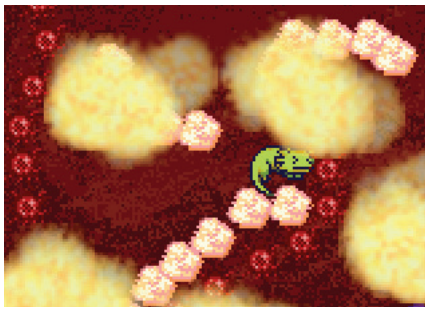
We intentionally were aiming for all age groups in making our game. From our overrall reaction from playtesters, it is clear that we have achieved this. On Rutgers Day alone, we had about 100 kids and young teenagers play the game and give extremely positive feedback. Additionally, we showed our game to many of our college-age friends, who then passed it along to their friends. Overall, from this we had over 250 unique visitors from all over the world visit the website to play.

Kids gathering around the big screen TV at Rutgers Day

What Went Wrong

Started Playtesting Late

We wanted to finish all fifty levels before we started playtesting. However, we did not realize that nearly all the levels we made would eventually be scrapped or redone. On average, we changed or removed each level four or five times. So, the delay in playtesting only forced us to rush at the end, and we may have been able to refine the levels more if we playtested earlier.



Some levels were too difficult!

Difficulty Curve Complications

Because of our previously-mentioned lack of outside playtesting towards the beginning of the development cycle, we had to test the levels ourselves as we were making them. We did not realize, though, that we were much better at the game than the average new user. So, we accidently made most of the levels extremely difficult, and so the difficulty curve was far too steep, and this took a long time to fix.

Player States

When we first started making the game, the player controller class was very simple. So, we did not use any player states, since we felt that it would have overcomplicated the code. However, as the weeks went on, we added many tweaks to the controls, such as scampering and block interactions, forcing us to steadily add more and more if statements, creating the exact problem we were hoping to avoid.

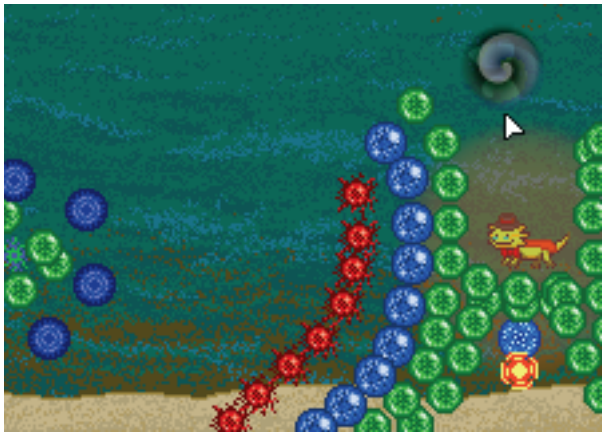
Pixel-Perfection

When designing the levels, we realized that the position of some blocks would cause the player to collide and get hurt or stuck when it was not necessarily his fault, making the game feel glitchy. Also, these problems with pixel-perfection sometimes caused animations to become jittery. These issues were probably exasperated by the aforementioned lack of player states.

Sound

We initially had some issues with sound creation. None of us had much experience in sound or music, and so it was not used to its fullest potential. We were originally hoping to have looping in-game music, but we found that any music we would have made would be of rather low quality and detract from the experience. So, we decided to just have in-game sound effects and have music limited to the intro and ending story slides.

Chilling out with friends at the level hub.



Using the gravity field to go down.

Conclusion

Overall, we feel that the game was a great success. While this game is complete, the process in creating the game and the levels also produced many different ideas that could be taken and expanded on further in the future. This project also gave us a better idea of scoping what is possible in a given amount of time, as well as experience in Flash and game design and programming.