Chromecast Video Game

Personnel

Begley, Nathaniel
Coburn, Roxanne
Bentley, Nikki

CS 499 **University of Kentucky** **April 2017**

Disclaimer:

This project has been designed and implemented as a part of the requirements for CS-499 Senior Design Project for Spring 2017 semester. While the authors make every effort to deliver a high quality product, we do not guarantee that our products are free from defects. Our software is provided "as is," and you use the software at your own risk.

We make no warranties as to performance, merchantability, fitness for a particular purpose, or any other warranties whether expressed or implied.

No oral or written communication from or information provided by the authors or the University of Kentucky shall create a warranty.

Under no circumstances shall the authors or the University of Kentucky be liable for direct, indirect, special, incidental, or consequential damages resulting from the use, misuse, or inability to use this software, even if the authors or the University of Kentucky have been advised of the possibility of such damages

Abstract

The Chromecast video game is just that, a game that can be ran using a Chromecast. A Chromecast is a small piece of hardware that plugs directly into a television via the HDMI port. This device acts as a receiver for applications that want to wirelessly “cast” videos, music, web pages, games, etc. The final game created for this project is a version of the arcade classic Centipede. This game is controlled using buttons displayed on a webpage, where all actions are wirelessly sent to the television screen, causing the player to move/shoot.

Introduction

Need

The customer came with a project to develop a video game that could be played using a Chromecast. They wanted to use an existing HTML/JS game, and modify it so that it could be controlled remotely. There have been various other projects done over this project, and no group has been able to get the Chromecast connectivity enabled.

Solution

The solution for the project proposed by the customer, was to create various web pages, that can be accessed with a sender (PC/phone) and a receiver (Chromecast). The webpages will communicate to each other in a way that allows button presses on the sender to be registered with the receiver, causing an action to take place on the screen. This involves the use of Google's Cast API. If time allowed, the group would move on and create their own game that could be ran on the Chromecast. The ultimate goal; however, was to just have a quality product that could communicate with the Chromecast correctly.

Customer

The customer for this project is Chad McQuillen, Senior Technical Staff Member at Lexmark International, Inc. He has a Bachelor's Degree in Computer Engineering. While he is a representative of Lexmark, this project is more of a personal project for him.

Users

The users of this project will be anyone who enjoys playing games, and has access to a Chromecast device. The game is meant for people of all ages and skill levels. The only limitation of access, will be the user's access to a Chromecast device, as the game cannot be run without that as well as a good internet connection.

Product Requirements

Using Chromecast, which is a wireless communication device that enables users to display media from one device to a television, implement a web based game that allows the user to use their device as a controller and their television as a display. Our main goal is to update an existing game to add Chromecast support. Currently the game is played via a web browser, using keyboard controls. Our project will allow the game to be displayed on a television via a Chromecast and played with a controller displayed via chrome. Our tasks are as follows (in priority order):

- Add Chromecast support to a pre-existing javascript game called Centipede
- Allow user to display the game on a television screen while controlling the game remotely via the chrome web browser
- Implement a controller displayed via Chrome web browser that communicates with the Chromecast device.
- Update existing game to respond to a graphic controller displayed on desktop or mobile through chrome, as opposed to a keyboard
- If everything else is completed, then the team can create custom game, **Netcast** (a play on the popular ASCII graphics based game, NetHack), that can be played via Chromecast and uses pixel-based graphics via processing
  - Create a game board with several levels that increase in difficulty
  - Create a sprite that can be moved forward, backward, and jump using arrow directions displayed on the controller and that has an inventory of items
  - Create enemies that spawn randomly on the game-board that sprite can attack by tapping button on the controller
  - Create items that are scattered across the board that can be equipped to the sprite
  - Add sprite leveling based on dungeon levels beat and number of enemies defeated

The game being used is called Centipede. This game is an arcade classic where a centipede continuously comes from the top of the screen, slowly making its way towards the player. The player has to shoot the centipede to kill it, but every time it is hit, the centipede will break into two separate centipedes. There are other enemies as well, that have varying abilities. There are also mushrooms all throughout the page that prevent player movement. The centipede is also affected by the mushrooms, but the other enemies are not damaged, and in fact, actually modify the mushrooms on screen.

Product Planning

In this section, the planning of the project will be discussed. This includes all of the logistics behind the project. Although, everything in this section was a rough estimate of how the project would go, overall we followed our initial guidelines relatively well.

Platform

Programmed using Javascript, HTML, and CSS, using Google's Chromecast SDK. Designed for Chrome Browsers that support casting tabs and the first generation Chromecast.

As mentioned, the main focus of this project was to get the game connected to the Chromecast. To do this, the group had to use Google's chromecast SDK. This is freely available online through Google's developer portal. Once there, they have 'Getting Started' guides and small tutorials to help developers get started. They also have many different examples available to the public through their github account. You can download these examples to understand just how it all works. The team used these examples as a base for much of their project. It was extremely helpful to have these at their disposal. These examples were not perfect though, and required a lot of modification for it to work how the team needed it to.

Size Estimation

Sender Size Estimate: 400 lines
Receiver Size Estimate: 200 lines
Existing Game size: 2000 lines

Actual Sender Size: 355 lines
Actual Receiver Size: 155 lines
Existing Game Size: 2000 Lines

When estimating our code, we took into account several examples of different existing chromecast apps. We considered which parts of the example apps would be necessary to integrate the existing game with the chromecast. We also considered typical sizes of interactive html pages, as the sender would need to implement this. With these considerations in mind, our size estimate was fairly accurate.

Schedule

February 10
-Project web page due
February 12
-First Project Meeting
-Drafted requirements
-Started Project Design
February 19
-Finished Project Design

February 24
-Chromecast setup for development
-Started Project Code
February 26
-Create basic custom receiver
March 5
-Started Testing Plan
March 8
-Design Due
-Midterm Presentation
March 20
-Tailor custom receiver for game input
April 2
-Redesign game to meet current UX Standards
-Finished Testing Plan
April 5
-Testing Meeting with Instructor
April 9
-Display playable game via Chromecast
-Finished Project Code
-Started Module Testing
April 12
-Code Review with Instructor
April 19
-Complete Module Testing

Design

A. Overview

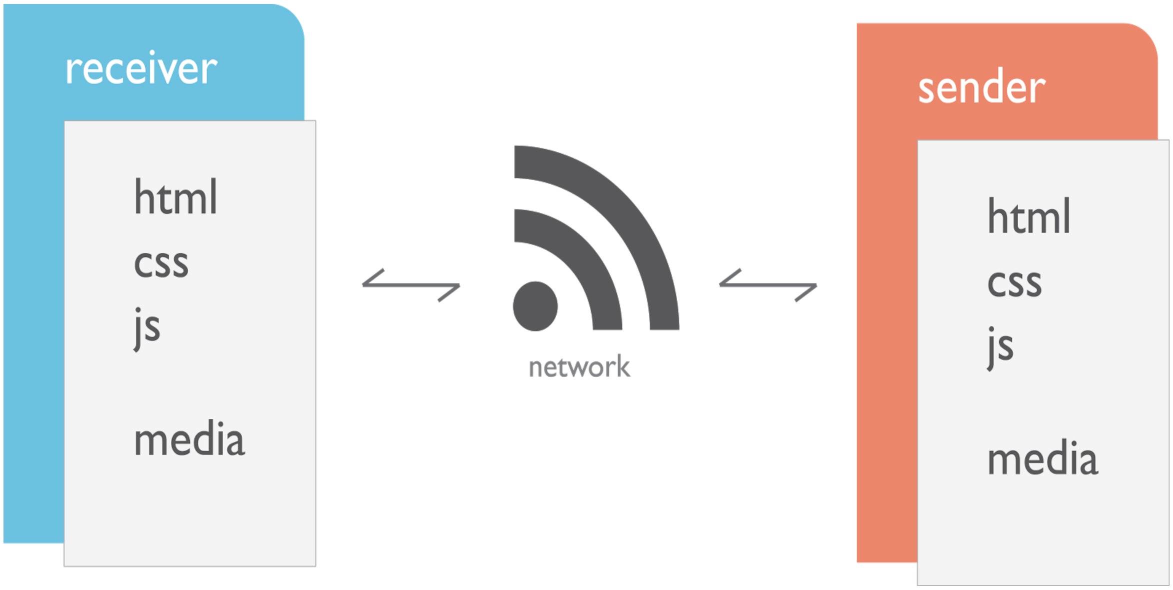
The Chromecast is a piece of technology developed by Google that streams digital media to a television via a small dongle. This device can be controlled by a mobile phone or a computer. On its own, a Chromecast cannot do many things, but when paired with something else it turns a “dumb” TV into a “smart” TV. The goal of the project is to create a game that can be ran on the chromecast. The controller will be the touch interface of the device and the screen where the actual game is will be the TV. The ultimate goal is to have a chromecast implementation of an existing Centipede game. This game is a recreation of the the classic arcade game Centipede. The game will undergo a thorough redesign after the chromecast implementation is done. The plan is to update the graphics, give it a dynamic window size that changes depending on the size of the screen, and add functional home and pause screens.

The main goal of this project is to learn programming with the Chromecast. The online documentation is there, the team just has to find a way to apply all of the information available to the situation at hand. None of the group members have experience developing with the chromecast, so this will be a great learning experience for everyone.

B. Environment

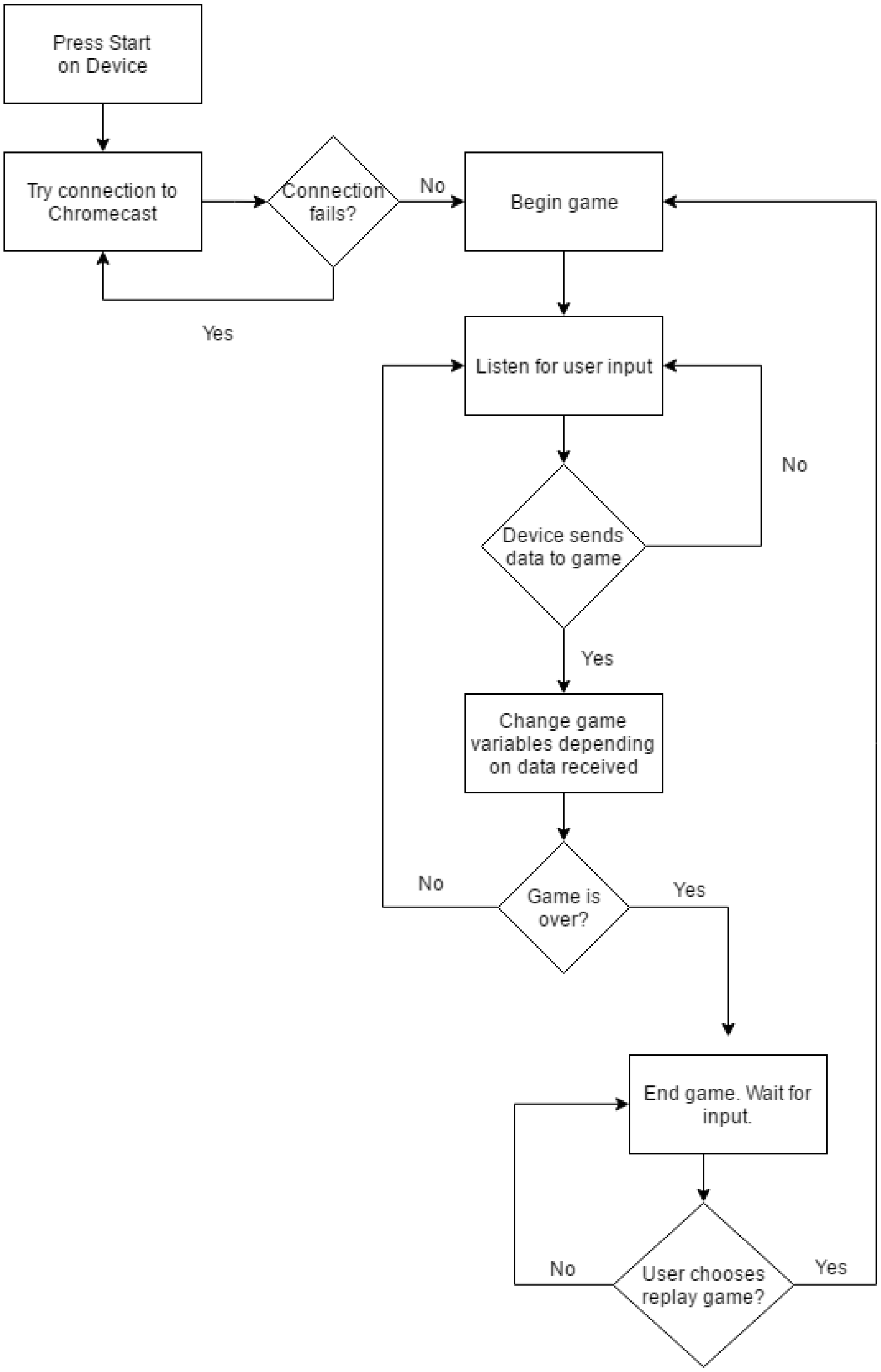
The environment that will be used for this project is the Chrome Web Browser (versions are listed in the testing portion). The applications that we will be making will be web apps hosted through github. There are many different parts that must work together harmoniously for the project to function correctly. The operating system can be anything that runs chrome smoothly. For testing, we will be using macOS Sierra, Windows 10, and ChromeOS.

There are many different languages that will be intermingling and functioning together. The following will be included in the project: js, html, css. The designs for the sprites will be made in Photoshop and Illustrator.



There will be two separate web pages that will “speak” to each other. They are called the sender and receiver. The sender is the device being used that sends the information and actions to the receiver so that it can handle the information and display actions. Everything for this project will be hosted through github and available on the Chrome web browser.

C. Module Descriptions and Data Flow


Press Start on Device
Try connection to Chromecast
Connection fails?
No
Yes
Begin game
Listen for user input
Device sends data to game
No
Yes
Change game variables depending on data received
Game is over?
No
Yes
End game. Wait for input.
User chooses replay game?
No
Yes

There will be two major pieces for this project, a sender and a receiver.

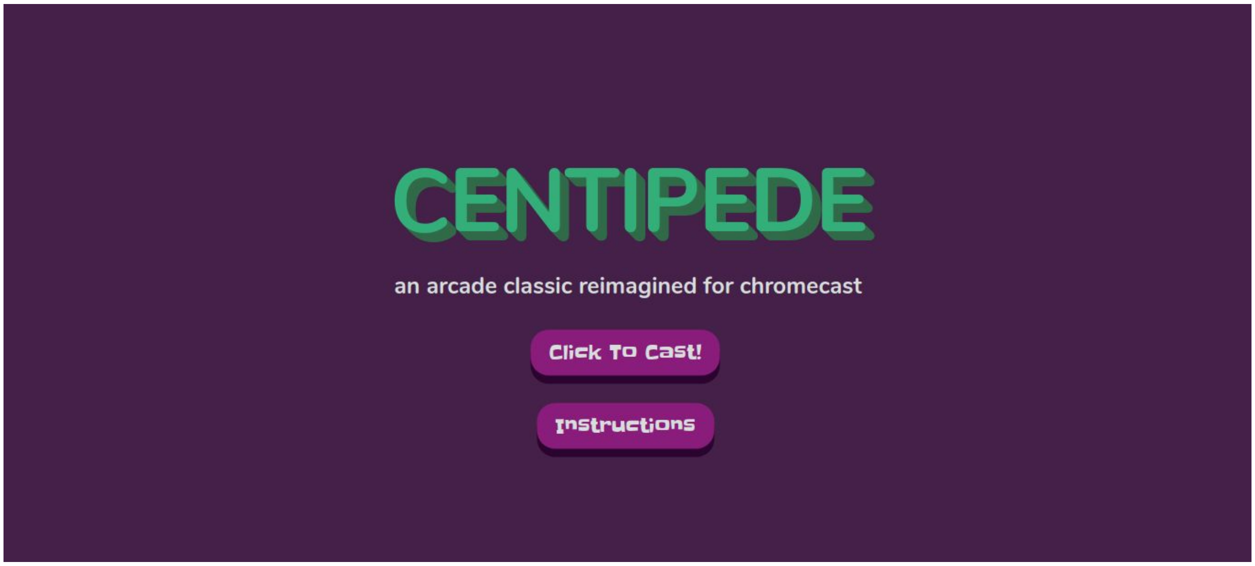
Sender Pseudocode

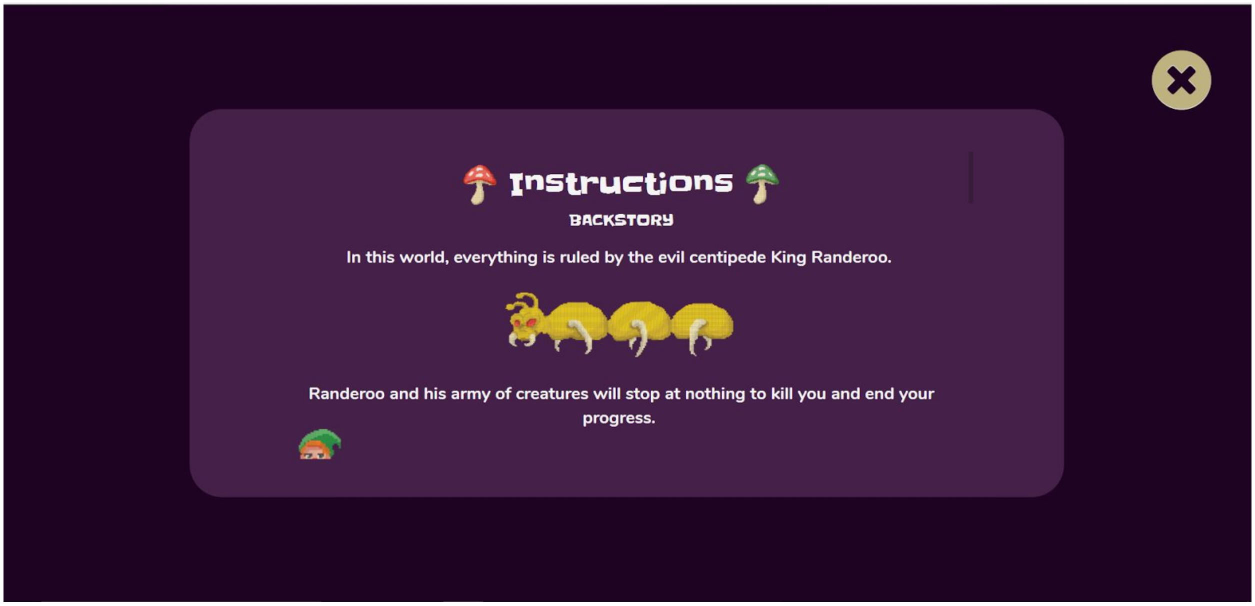
```
● Display user controls
● When screen is cast:
    ○ Establish connection with receiver
    ○ Listen for user input
    ○ While receiving input:
        ■ Package input into message
        ■ Send message to receiver
    ○ Close connection
```

Receiver Pseudocode

```
● When screen is cast:
    ○ Load game
    ○ Establish connection with sender
    ○ On receive message:
        ■ Determine command
        ■ Execute command
        ■ Update Screen
    ○ Close connection
    ○ Display final game screen
```

User Screens



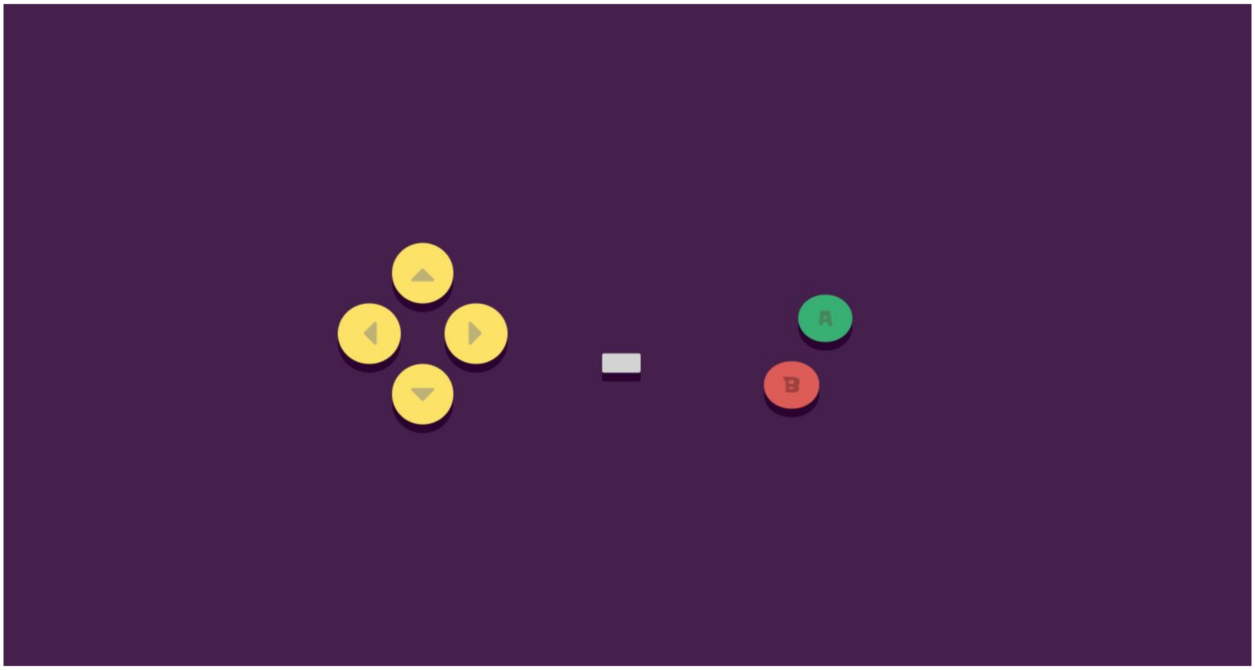


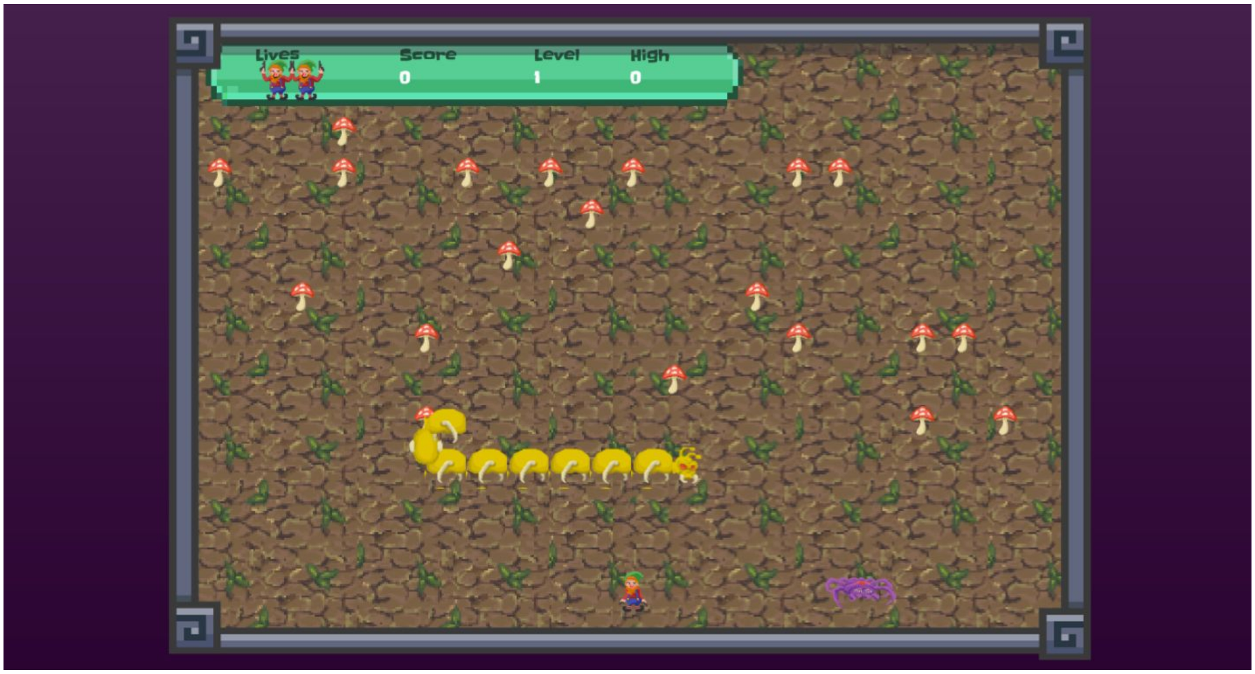
Instructions

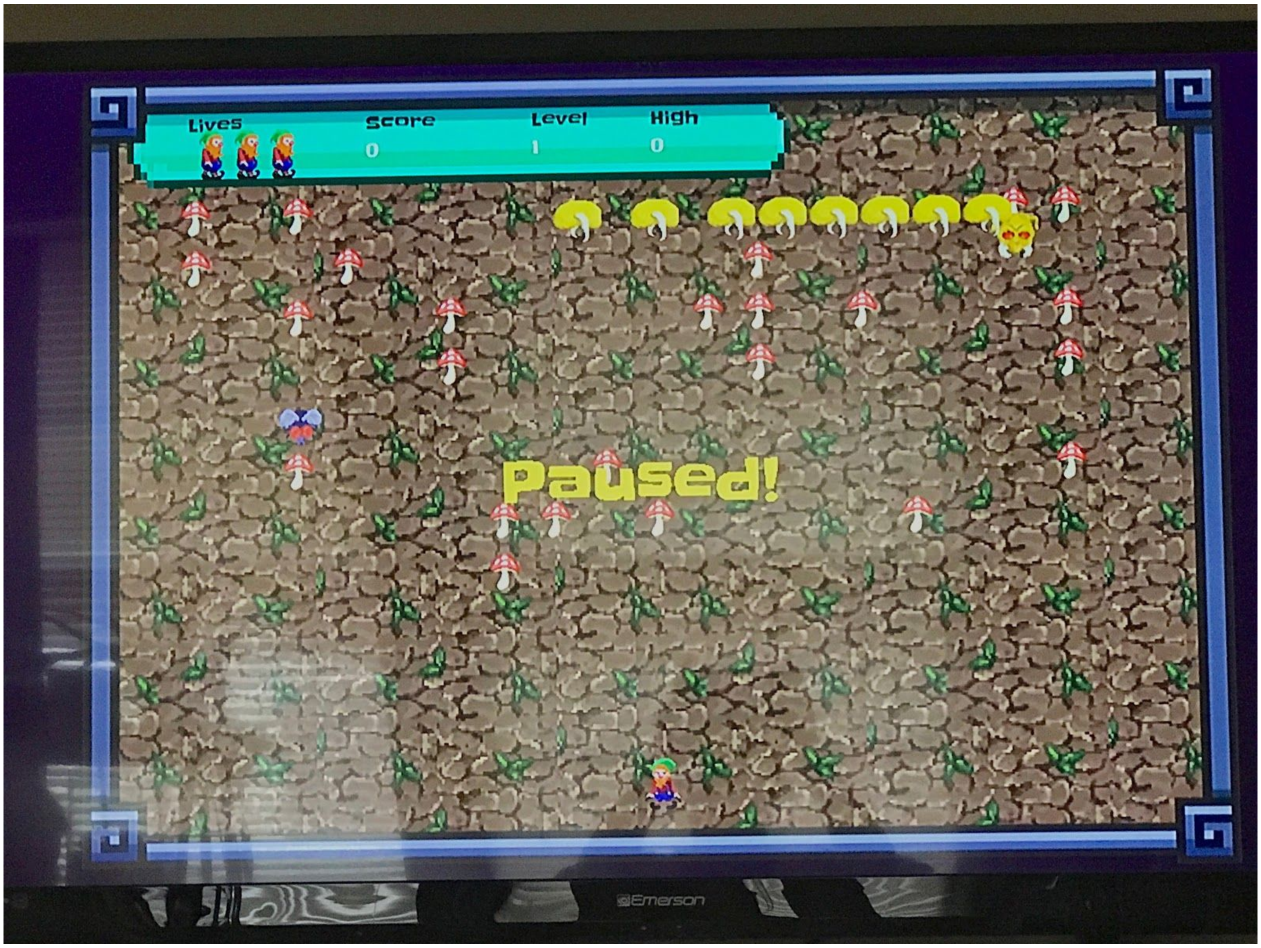
BACKSTORY

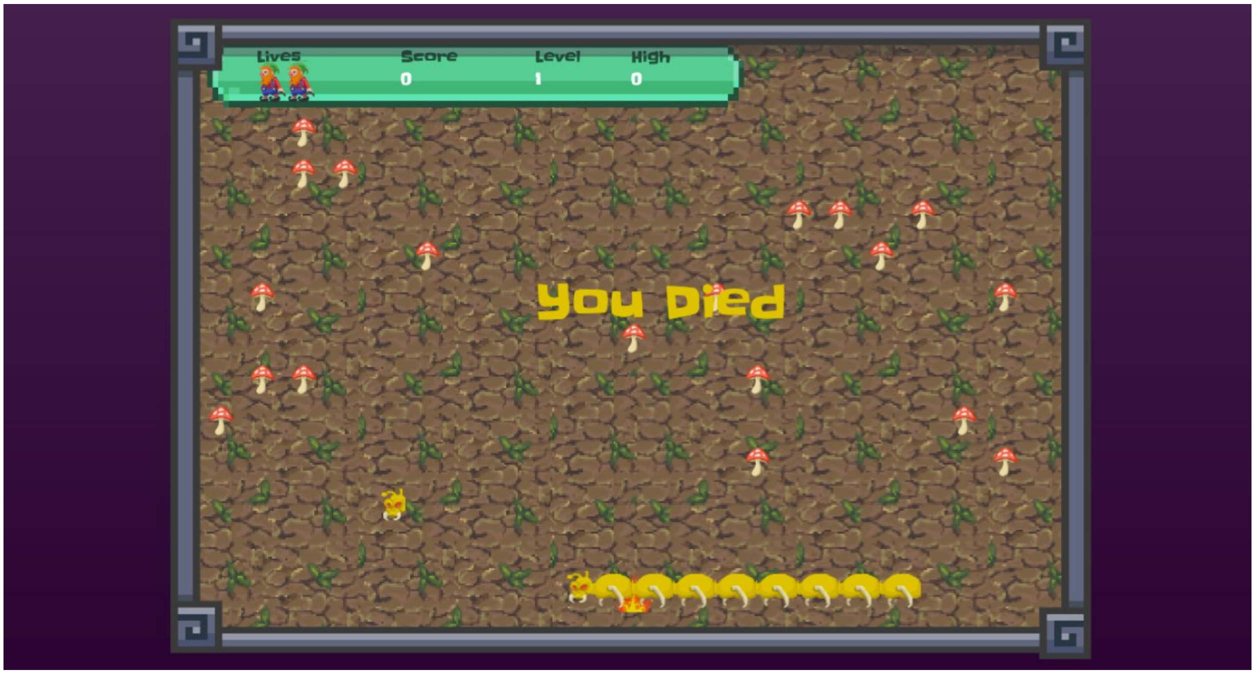
In this world, everything is ruled by the evil centipede King Randeroo.

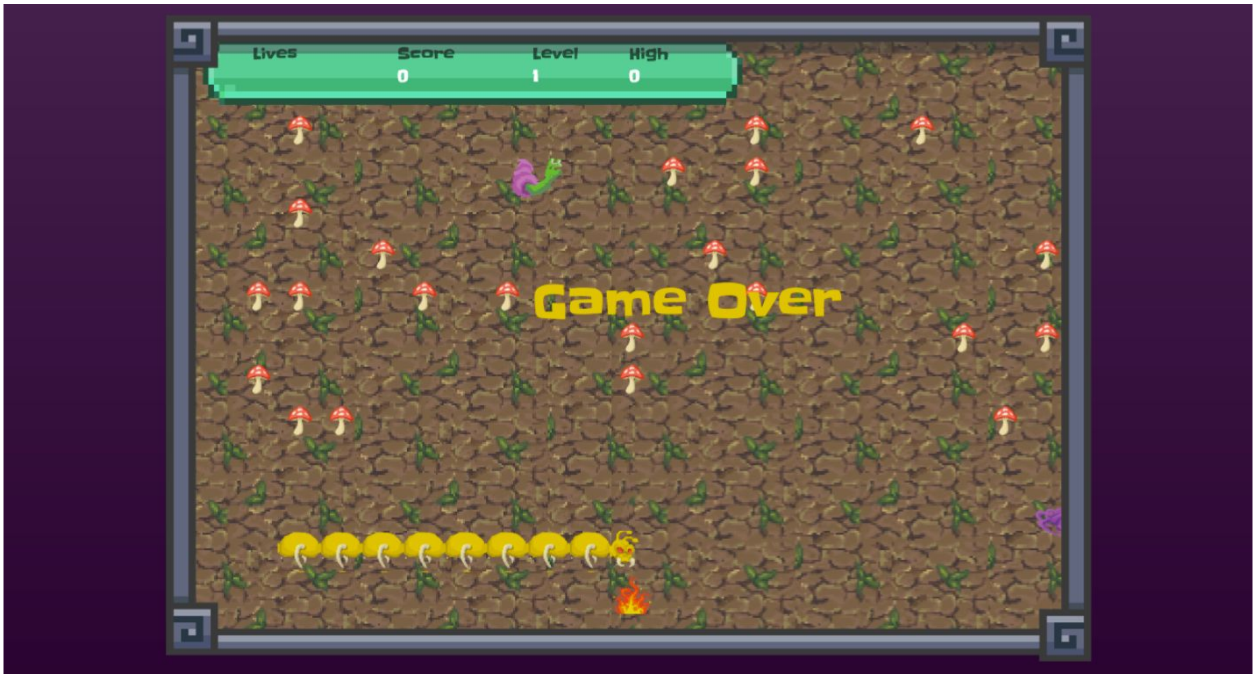
Randeroo and his army of creatures will stop at nothing to kill you and end your progress.




Lives
Score
0
Level
1
High
0


Lives
Score
0
Level
1
High
0
Paused!
Emerson





D. User Scenarios

- When user opens the link in Chrome on their phone, Chromecast immediately shows the home screen.

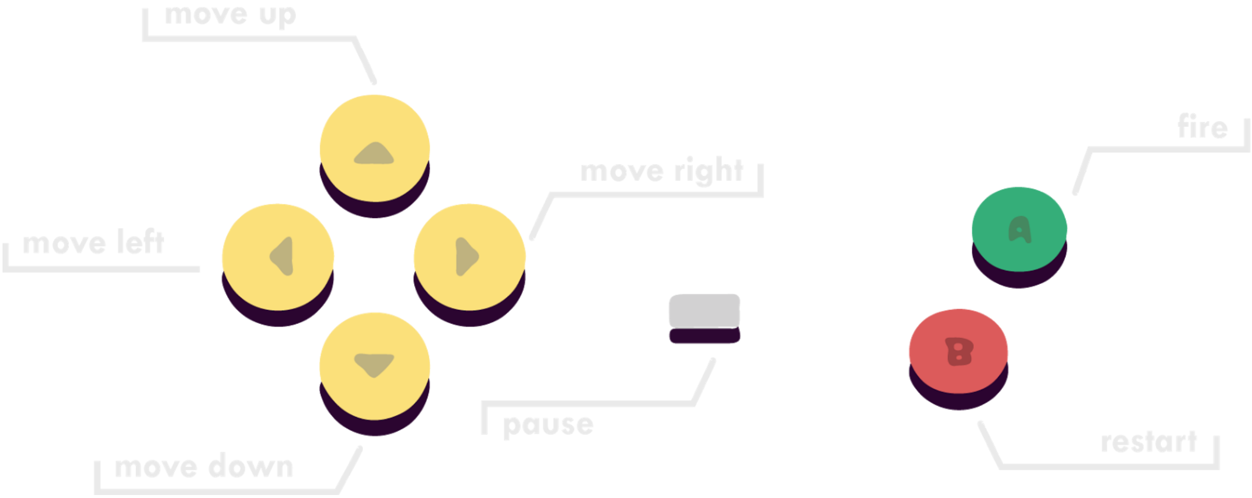
- Wait for user to press a button
    - If they press ‘Click to Cast’, then a list of available devices drop down, then it connects.
    - If they press ‘Instructions’, then the instructions are listed on the sender.
- If a connection is successful, then the game will start. The controller is displayed on controller, and the game screen starts on the TV.
- User hits left, right, up, or down arrow on the controller, then the sprite moves left, right up, or down on the monitor.
- User hits “A” button on the controller, sprite fires upwards on monitor
- User hits rectangle button on controller, game pauses
- When game is paused, must hit rectangle button again to restart game.
- If the connection is dropped, the game ends.
- User passes a level, must wait for next level to appear on screen
- User dies, must wait for next life to start.
- User loses game, must press “A” button to restart game.

Implementation

A. Implementation Decisions

The group decided that it would be most efficient to make a sender app for the chrome browser. This would allow the application to be played via any browser that supports the casting of tabs.

We also spent significant time on the design process. We decided that in order to make a high quality game that could one day be marketable to a wide audience we needed to make it attractive as well as functional. To this end, we focused primarily on usability. We wanted the controller we made (which is displayed on the sender app) to be easy to use, functional on a variety of devices, and to allow the user to enjoy gameplay. To do this, we docked the arrow pad on the left hand side of the screen, and the A/B buttons on the right hand side and the pause button in the middle. We made the buttons scale depending on the device screen size as well as used brightly colored buttons that are easy to see and in a format that is easy for the average gamer to understand.



In addition to this, we also added a colorful and engaging introduction screen that gives a backstory to the game which in turn engages the audience in the game and gives them a purpose behind playing the game. We not only worked to simply instruct the user how to play the game, but why they should play the game.

We also updated the original game graphics. We changed the dimensions to be easier to view on the standard TV, and the sprites to be larger and easier to target upon encounters. We changed the overall color scheme to match the controller in order to make both parts seem integrated and succinct as to not distract the user. We also worked on making the actual sprites more detailed so users are visually pleased as well as pleased with the actual gameplay. We also updated the menu bar to have larger text in the same font style as the rest of the project to again join the two parts in unity.

We think these design changes enhances the overall game experience considerably and take this project from a basic experiment in the capability of the Chromecast hardware to an actual viable product.

B. Problems During Implementation

One of the biggest difficulties in implementing this design was understanding the registration process to set up a development environment. The most time consuming part of the process this semester was registering as developers and configuring the hardware. Beyond that, we did not initially assume that a device had to support tabular casting to work with the sender application. Also, due to the hardware restrictions, there was a significant time delay between altering the code and having the changes reflected.

C. Restrictions

Currently, the largest restriction is that the game is only fully functional when using a desktop version of chrome. Additional app development will be required to add functionality to android and ios.

Testing

The main purpose of this project is to produce a game that is playable on a television using a Chromecast device. Because of this, there will be an extra focus on the connection between the device and the Chromecast with our testing.

Unit Testing:

There are many different modules in the project that must work together to give the finished product. To confirm that all modules are working they will be tested independent from one another. The primary tests for this portion will be as follows, each bullet point is representative for a module being tested:

- Connecting the sender module to the receiver module.
  - Manually calling appropriate functions to connect.
- Registering keypresses when connected.
  - Pressing any buttons on screen, which are then handled.
- Use of “Cast” button.
  - Pressing the “Click to Cast” button, then connecting receiver and sender.
- Displaying splash screen and all associated buttons.
  - When the URL is accessed, display screen with functioning buttons
- Use of “Instructions” button.
  - When “instructions” is pressed, a pop-up appears with the background.
- Movement of player sprite using clickable controller.
  - Buttons clicks are handled, and turned into movement on screen.
- Movement of player sprite using touchable controller.
  - Buttons presses are handled, and turned into movement on screen.
- Player and enemy sprite hit detection.
  - If a player is hit, then they die. If an enemy hits a mushroom/other enemy, the appropriate action takes place.
- Player shooting animation and enemy hit detection.
  - When firing, if an enemy/mushroom is shot, the appropriate action takes place.
- Dynamic screen size.
  - All screen sizes/resolutions are compatible.

- Level changing.
    - After each level is finished, the level is reset and difficulty increased.
- Displaying pause screen
    - When rectangle is pressed, “Paused” is shown on screen and all animations/events are stopped.
- Game Over functionality
    - Display game over, and wait for click to restart.

Integration Testing:

In this stage all of the individual modules will be combined and tested with each other. There is a lot of communication between the separate modules, so it is vital that they all function together properly. The same tests will be conducted as above, but with all of the parts working concurrently. This is what the end user will be using, so full functionality is a necessity. In this step, the combined project will be tested on the most stable platform to ensure that all issues present (if any) are caused due to coding errors.

System Testing:

After full integration is complete, the project as a whole will be analyzed to see whether it meets all specifications and requirements set out at the beginning. The main goal here is to test as many different devices as possible to confirm that the design will function universally. The browser that will be tested is the Chrome browser, as it is the only browser that is officially Chromecast supported. The devices being tested will include: Apple iPhone, Apple MacBook Pro, Windows PC, Chromebook, and Android Phone. Extensive testing will be done on all platforms. Some of the platforms do not have any real functionality, but it was chosen to leave them in for continuity. Below, Table 1 contains the results of the testing. The tests being conducted correspond to the individual modules listed above, but altogether this time on multiple platforms.

Customer Testing:

Once all other aspects have been tested, the project will be opened to users. Focus groups will be created that will test the project as a finished product and give us feedback. There will be a wide range of users in these groups, so that we can have the best overall experience for the average person. The main areas we will be looking at in user testing will be: functionality of the game on other devices than what was initially tested, performance for each individual user, and overall quality of the experience for the user. The results of this testing can be found in Table 2 below.

Key: Y= Functional N= Not Functional N/A=Does Not Apply

TABLE 1	MacBook	Windows PC	Chromebook	iPhone	Android
Connecting Sender to Receiver	Y	Y	Y	N	Y
Registering Keypresses	Y	Y	Y	N	N
“Click to Cast” Functionality	Y	Y	Y	N	Y
Splash Screen Functionality	Y	Y	Y	Y	Y
“Instructions” Functionality	Y	Y	Y	Y	Y
Player movement via clicking	Y	Y	Y	N/A	N/A
Player movement via touching	N/A	Y	Y	N	N
Player and enemy hit detection	Y	Y	Y	N	Y
Player firing and enemy shot detection	Y	Y	Y	N	Most likely would work, but cannot be tested due to no connection.

Dynamic Screen Size	TV DEPENDENT	“”	“”	“”	“”
Level Changing	Y	Y	Y	N	Most likely would work, but cannot be tested due to no connection.
Pause Screen	Y	Y	Y	N	N
Game Over	Y	Y	Y	N	Y

(Scale from 1-5, 5 being highest)

	User Info			Testing Results (Quotes from Users)			
	Age	Sex	Game Experience	Design	Controls	Interest	Overall (comments)
User 1	21	M	5	5	4	4	The game is beautiful, and I really like it a lot. Only issue I see is control-scheme being hard to grasp.
User 2	21	F	3	5	2	4	The game works well, however the controller was difficult to use.
User 3	21	F	5	5	3	5	I like the colors, the pixel art is pretty too. I just wish it would work on a smaller device.

User 4	20	F	3	5	3	2	I love the games look, it's really cool. I had never heard of this game, but I like it. Controls were a little weird.
User 5	48	M	2	4	3	5	I played this game when I was younger, so this is really cool to see done with this new technology.
User 6	45	F	1	5	2	5	This used to be in the arcade I worked at, I love the concept. It is also really pretty. I have trouble controlling the guy though.
User 7	28	M	2	5	3	3	The game is hard to control, but when you get past it, it is a beautiful game.
User 8	27	M	5	5	4	5	I used to have a version of this. This is a great version of it, well done.
User 9	18	F	2	5	4	3	I haven’t seen this game before, but I really like it. I would download this.
User 10	21	M	4	5	5	5	This game is so cool, I haven’t seen any kind of game like this on Chromecast before.

Future Enhancements/Maintenance

A. Android App

The current game is only supported using the desktop version of chrome that supports tabular casting. Assuming that chrome does not change the functionality of their chrome browsers, an android app would be necessary to play the game on an android phone. The android app would function as the sender side of the project. The code would be similar to the current code in the chrome sender. There will be slight differences caused by the dependencies that would be required in the android app. After the development of the app, the receiver side and the rest of the project should remain unchanged.

B. Hosting

Currently the game is hosted on Github using gh-pages. This is a free hosting solution, and works perfectly for the scope for this project. The team can all individually add to the project and test it remotely, which has been a huge help overall. The only issue is that this is not a permanent solution. In the future it would be nice to move the hosting to a more secure and reliable host. This would involve getting a domain/server for the game to run off of which would cost time and money to get set up. If this were to ever go commercial, then that would be an important step.

C. Multiplayer

During testing the idea of multiplayer capabilities came up. Currently the game works with only one player, but works very well for what the team initially had planned. If it were possible to have multiple players playing as the gnome at the bottom, then the game would be on another level. Chromecast gaming lends itself to multiplayer use,because it is a “living room/media center” device. This means that usually there are more than one person in the room. If we could get everyone involved in the game, then it would have a better marketability.

D. SDK updates

Currently, the most recent SDK available is SDK version three. This, quite obviously, means that there have been two other version before this. With each new version release, comes a set of small challenges to overcome for the project to remain relevant. This will not be too difficult to implement, but it will take time and someone will have to do it in the future for maintenance.

E. Refined control scheme

The game controls as expected given the time allotted, but one of the main complaints the game received during user testing was the control scheme. Almost everyone that tested the game said the controls were a little off. So in the future it would be nice to change the control scheme a little more refined and user friendly. It is hard to control with a mouse, but with the touch screen it is a little better. With the touch screen PC, the buttons are in the middle of the screen, making them hard to reach. Overall, just refining the controls will be a nice addition to the project.

Conclusion

Throughout the semester, the group gained an in-depth knowledge of the google chromecast SDKs. In addition, we gained more experience working with javascript, using github to collaborate, and developing games.

Most of the changes in the original design stemmed from lack of experience developing for the chromecast as well as a lack of experience developing mobile apps. As a result of these changes, the group was able to dedicate more time to refining the user experience and updating the graphics of the game and the controller.

Given the advanced knowledge of developing for chromecast that we currently possess, if doing a similar project, the group would budget time differently. Less time would be spent on setting up a development environment, and more time would be spent on perfecting the chrome sender and potentially developing an android or iOS sender app to enable mobile gaming.

Overall, the project was successful in creating a game that can be ran and controlled using a Chromecast device. There were some issues along the way that caused design changes to be made, but as a whole the goal was met. The framework is laid out for another group to continue the work we have done and create a mobile application of some sort.

References

1. Nicholls, Leon. "Casthellotext-Chrome". *GitHub*. N.p., 2017. Web. 10 Feb. 2017.
2. "Cast | Google Developers". *Google Developers*. N.p., 2017. Web. 10 Feb. 2017.
3. Burt, Randolph. "Writing An HTML5 Game With No Game Engine – Just Angularjs And The HTML5 Canvas!". *The adventures of Randeroo*. N.p., 2014. Print.

User Manual

In this world, everything is ruled by the evil centipede King Randeroo. Randeroo and his army of creatures will stop at nothing to kill you and end your progress. You, a simple gnome, must kill Randeroo before he reaches you, if he touches you then you will die. Use your missiles to shoot Randeroo and his creature army. But watch out, each time you hit Randeroo he splits into two faster moving versions of himself! Each level he gets harder and harder to kill, so be prepared. There are mushrooms throughout the level that no-one can dodge, even Randeroo himself, but you can shoot these mushrooms to get a better shot at the king. His creature army, while small, can really have an impact though. There is Sabrina the Spider, he will always be near where you are at the bottom. He eats mushrooms, which can be really *helpful for you. But, he also eats people so never get too close. Freddy the Fly will also be coming for you. He flies from the top to the bottom of the level, spreading more mushrooms in his wake. He leaves long lines of mushrooms to disrupt everything. Syd the snail will slowly make her way across the level, never daring to come close to you. She does, however, give mushrooms strength, making them much harder to get rid of. It is up to you to topple the Randeroo empire… you are the last hope for your home, the gnome village of Pittsburg, good luck.*

A. Set-Up

There is very little set-up to begin playing this game. One must first setup and update their chromecast using the instructions included with the device. After the chromecast is connected to the user's preferred device, the user should open their up-to-date chrome browser and navigate to the game's homepage. From there, the user has the option of reading the game's instructions or starting the game by the "Click to Cast" button.

B. Home Screen

On the homescreen, there are two options for the player. The "Instructions" button will display details about the game and how to play. If there is a chromecast available, "Click to Cast" will activate a prompt to connect to the chromecast device. Upon connection, the game will automatically start and the controller will be displayed on the screen.

C. Gameplay

a. Moving

The player is able to move in all four directions using the yellow directional buttons. The player must press the buttons multiple times in order to move continuously. The player must move within a certain area at the bottom of the game board in order to avoid the obstacles present on the board.

b. Shooting

Use the green "A" button in order to fire missiles. It may take multiple missiles to remove an obstacle.

c. Pausing

Pressing the Grey Button will allow you to stop and restart the game at will.

d. Winning

In order to advance to the next level, the player must completely eliminate the centipede before the centipede can reach the player.

e. Losing

After each death, "You Died" will be displayed. If the player has more lives, then the game will automatically restart at the beginning of the current level. If the player is out of lives, “Game Over” will be displayed on the screen. In order to restart a new game from the first level, the player should press the red “B” button.