Developers Journal

ORBliteration

# Pre-Production Phase

## Requirements

* Game Design Document
* Technical Design Document
* Alpha Sprint Backlog
* Product Backlog
* Pre-Production Phase

## 31 July 2015

### Scrum Meeting:

N/A

### Tasks:

* High Concept Document
* Concept pitch Presentation
* Decide on Scrum master
* Decide on organisation tools

### Update:

The High Concept document was completed. We then presented our concept pitch. It was a successful pitch as we were given the Go-Ahead to move onto the Pre-Production phase of our project – ORBliteration.

It was decided that that for the Pre-production Phase that J. Griffith Will assume the task of Scrum Master. The use of Trello alongside the Wall would be used to organise the team’s production.

## 4 Aug 2015

### Scrum Meeting:

J. Griffith led today’s Scum meeting and it was decided to start writing up the Game Design Document. First we made a decision to follow the template from Iron Belly Studios to create the GDD, and follow up with the guidelines given to us. After which sections of the GDD were delegated to each team member to complete.

### Tasks:

* Write the GDD

### Update:

Initially we split up the sections of the GDD and each team member worked on them. When a member had finished there section they would help another team member with theirs.

Once completed we added each section together and as a team we edited it to create a flow and concurrency with in the document, as well as correctness.

## 6 Aug 2015

### Scrum Meeting:

J. Griffith led today’s Scum meeting and it was decide that we needed to get our GDD approved by the product owner – Asma Shakil. So we emailed A. Shakil to go over the GDD.

### Tasks:

* Get Approval on GDD from product owner

### Update:

A. Shakil had a look at the document and gave us the following advice on how to edit our GDD.

* Add in a table of changes
* Format heading to include numbers to clearly show sub headings
* Remove target platform and add it to the TDD
* Move Visual/Audio style heading to the assets heading.
* Rename “Starting Out” to “Game stages”
  + Augment by adding images
* Add super heading “Game Play” and move mechanics section to it.
  + Also put Heading “Game Play” before “Game stages”
* Add supper Heading of “Game Design” for “Game stages” and “Assets”
* Add Summary at the end
* Think about adding taught/untaught mechanics

## 7 Aug 2015

### Scrum Meeting:

J. Griffith led today’s Scum meeting and it was decided that a tech Demo or prototype was needed. So a basic implementation of the balls hitting into each other and knocking each other off a platform. The platform will also need to be decreasing in size over time. Power ups were deemed unnecessary at this point. It was decided that Unity was going to be used to create the prototype.

### Tasks:

* Tech Demo/Prototype
  + 2 Players (Local)
  + A surface that decreases in size over time
  + Player movement that was based on acceleration
  + A collision between players that caused a transference of force
    - i.e. when a player knocked the other it would go flying

### Update:

Due to the simplicity of the required prototype, and the ease of using Unity. As a team we used one machine to create the prototype.   
The prototype was completed with basic gameplay and had all 4 requirements. Creating the prototype taught us a few things about what we could be facing in the future when creating the actual game. Such as;

* The Size of players to play surface
* The balancing of the acceleration speed
  + so that it’s not too fast to be unplayable, but fast enough to notice a speed increase
  + Also to have less control when going very fast.
* The Force imparted when colliding will be an interesting challenge to overcome as well

## 11 Aug 2015

### Scrum Meeting:

J. Griffith led the meeting and it was decided that we need to complete the GDD with advised changes. As well as get started on the Technical Design Document (TDD).

### Tasks:

* Update GDD
* Begin Work on TDD

### Update:

We had started out with implementing the changes to the GDD, which did take longer than initially anticipated but we had completed the changes. We also went through the document multiple times editing the GDD continuously to give the document flow and continuity.

At which point we then started work on the TDD. We focused on implementing the coding standards and basic bullet points structure for the document.

## 18 Aug 2015

### Scrum Meeting:

J. Griffith has done a bit of research on how to implement the fonts with in the game, and realised that we needed to change the way we do the fonts. This needed to be shown with in the GDD, and therefor GDD needed to be updated. During the week we discovered that the burn down charts along with the alpha sprint back log was needed for this phase as well. The day will be used to get started on these tasks as well as continue with the TDD.

### Tasks:

* Update GDD
* Continue Work on TDD
* Create Back Log
* Create burn down Chart

### Update:

Started out with editing the GDD accordingly to the change in how we are planning on implementing the fonts in the game. The Burn down chart was created along with a template to use for future burn down charts.

The User stories for the product back log were started. We got a number of them written down, but have not yet added them to the back log itself.

Continued work on the TDD. A large chunk of the TDD got completed though questions arose to the correctness of the document, so it was decided to get advice on how to continue and complete the document.

## 21 Aug 2015

### Scrum Meeting:

We held a quick meeting, to focus the team back onto the task at hand. Which was to continue work on the TDD and product back log. We also had a meeting with A.Shakil regarding a few sections in the TDD. According to the meeting we were on track with our TDD. Regarding the TDD it was decided to focus on getting the flow charts done

### Tasks:

* Continue Work on TDD – Focus on the flow charts
* Continue Work Back Log

### Update:

The product back log was completed with little problems. Next Tuesday we will decide the ordering of the back log as well as the Alpha One Sprint backlog.

Majority of the flowcharts required by the TDD were completed. The main game loop and its sub systems are the only flow charts left.

Minor Edits to GDD to make sure it was formatted correctly.

## 25 Aug 2015

### Scrum Meeting:

In today’s Scrum meeting it was noted that this is the final week for the pre-production phase. Therefore the TDD and Back Log needs to be completed today. Then from the product back log we need to decide the alpha sprint back log from it. Then once those are completed we need to come up with a presentations for the pre-production phase

### Tasks:

* Finish TDD
  + Flow Charts
  + Acceptance Test plan
* Finish product Back Log
* Create Alpha sprint Back Log
* Create Pre-production Presentation

### Update:

The team worked long and hard and managed to complete all required documents for the preproduction phase. We then when onto preparing a presentation to show how far we have come with the project and what our plan for the next phase was.

Alpha Sprint One Phase

# Production Phase - Alpha Sprint One

# Requirements

* The Updated Game Design Document
* The Updated Technical Design Document
* The Updated Product Backlog Sheet
* Alpha Sprint 2 Backlog
* Product Source & Release

# 15 Sep 2015

## Scrum Meeting:

In today’s Scrum meeting lead by Callan Moore as the new Scrum master, we came together to discuss the required tasks for this Phase. After which the tasks were then delegated out to each person. The tasks were discussed in great length to make sure each person knew what they needed to accomplish as well as the order that they needed to be completed in. In the end it was split up fairly and logically in such a way that the required task for each step would be completed before the next person would need to it for their tasks.

First tasks was to decide on whose current frame work we will be using. Since most of the graphical components fell on Callan and Juran, a combination of their frames works was decided on. So they would start the day working on combining the two frameworks as needed.

The tasks that befell on me at this included the following:

* The physics system
* The game pad input system
* The player game mechanics
* The multiplayer system (including the multiple input controls)
* The Win/Loss states

The rest of the tasks were split up between Juran and Callan Accordingly (For more info on their actual tasks see their development journals)

The first task that I need to complete is the game pad input system so today will be dedicated to getting started on that. Then followed by getting multiple controllers support after that

## Tasks:

* Research and implement the Game Pad input system
* Multiple Controller support

## Update:

I have had quite the successful day

I decided to handle the Xbox 360 controller input using the built in DirectX XInput. I was able to find very useful information and examples online on how to implement XInput in order to handle input from an Xbox 360 controllers.

I spent the first half of the day creating a class to hold the gamepad input functionality. This was done in my own frame work as the team were still busy merging their frameworks to create our team frame work.

The timing of finishing up the input class could not have been better as Callan and Juran were able to have a roughly merged Framework ready for me to add the gamepad input class to the teams frame work.

After adding the gamepad input class to the frame work I then proceeded to test its functionality, making sure that I was able to read input from the Xbox controller. The framework already had a smaller cube rendered on top of a larger cube so I used that to test the controller to move the smaller cube around the larger one. The input system worked quite well with very little adjustments. After testing the movement with the left thumb stick I thought it be a good idea to test the button mapping. Which I found was all correct. So at this point the input system was now complete for a single player implementation.

After having the single controller working it was time to work on implementing the multiple controller support. This required some tweaking and changing with in the input class but was not difficult and didn’t take me as long as I thought it would, and testing it, it worked straight away with no issues.

# 16 Sep 2015

## Scrum Meeting:

Today’s scrum meeting was quite short. We just discussed what we all did yesterday and what we were planning on doing today. Everybody was going to continue working on their tasks from the day before with minor tweaks and clean ups. Also updating the Trello befell on me as well as I was ahead in my tasks.

My planned tasks for today will be to update the Trello first. After which I was asked to add more functionality to the input class, as J. Griffith needed d-pad like controls for the analogue sticks. Followed by creating the player orb class and implementing the physics based movement.

## Tasks:

* Update Trello
* Added Functionality to the Input Class
* Create and Implement the Player Orb Class
  + Implement physics based movement

## Update:

First task was to update the Trello. I first added people’s names to each card that was assigned to them, adding a few of the missing cards and putting them in the correct places.

I then needed to implement the functionality with in the input class that J. Griffith required. As the analogue sticks controls return an axis of the direction the stick is in J. Griffith requested a more d-pad control base for the analogue sticks to simply see if the stick has been pushed towards one of the 4 directions. This wasn’t too hard as I just added a simple check on the axis of the stick to see if it was passed a certain threshold in the requested direction. This worked like a charm and worked exactly to how J. Griffith needed.

Upon adding the functionality I noticed that there were a few clean ups required with in the input system with changing how the tutorial done things to how I usually do things so I updated and cleaned up the Game pad class.

My next task was to implement the player Orb class and its movement mechanics. First part was to get the orbs drawing to the screen. I decided that implementing the orbs as simple single colored cubes first as I already had the mesh for cubes ready for me. So I created the Orb class to first be able to render a cube. Then added in the simple movement test that I had yesterday for the test cube that I had and moved the cube around. This changed how ever as we wanted the physics based movement. I decided that I was going to create this physics based movement without the use of a physics API as I understood what I needed to do to create the physics movement and would find it much simpler to implement it myself rather than try integrate a physics API into the code.

As first I wanted to create the movement using forces, and mass, but in the after a while decided that all I needed was acceleration and velocity. So after adding the physics movement I needed to tweak the speed values so that the orb (cube) wasn’t flying across the screen so fast.

I decided to play around with a bit of rotations as well trying to make the Cubes roll as they were moving around. I kind of got the cube rotating but it wasn’t quite right and took me some time to figure out what the issue was. I first thought it was the speed of rotations was correct or I was basing the rotations on the wrong variables, I had them based on acceleration and velocity but found the position held better results. Thou it was still not correct after a while I realised the problem was that the rotations where happening on the cubes local axis and not on the world axis and that was why it would not look right some of the time. So at this point I decided to put the rotations on hold and removed it from the Orb.

I then decided to finish up tweaking and implementing the movement correctly trying to get a balance with the speed of movement and speed increase etc.

After I got to a movement that felt somewhat close to what we wanted I decided to add the other players and see if I could control separate orbs with separate controllers. Which worked quite well as the correct orbs where being controlled.

So with that I decided to call it a day.

# 17 Sep 2015

## Scrum Meeting:

Today’s Scrum meeting continued the same way as the last as we discussed what we done so far and what we had planned to implement for today.

As Callan had implemented a basic arena as separate tiles my tasks for today was going to be based around the interaction between the player Orbs and the arena tiles. This required me to calculate which tile orb was on. And then decide how they would interact, as we wanted the three different tiles that affected the Orbs movement in by slowing it down in varying degrees, with the rough tile slowing the orbs down the most and the slippery tiles barely slowing them down, with the last tile somewhere in-between.

## Tasks:

* Orb - Tile Collisions
* Orb - Tile Interactions
* Win/Los States

## Update:

First task for today was to calculate which tile the orb was on at any given time. I first implemented this by looping though each Player Orb then for each Player Orb I looped through the list of arena tiles and checked if the player orb was colliding with the tile. After almost completing the collision check I was reminded of a more optimised and quicker way of calculating which tile the orb was on. In a previous project that dealt with a gridded area I needed to calculate where in the grid a user had clicked, I calculated which tile was clicked based on the mouse position and the screen position of the tiles, then converted the x and y coordinate of the mouse position into the row – column position with in the 2D array of tiles to get which tile was clicked. I decided to implement the same idea with in the ORBliteration. By taking the position of an Orb and knowing that the area tiles are centred on world origin. I use the x and y coordinate of the orbs centre to calculate the row – column position with in the 2D array of arena tiles. I had first slightly miss calculated the each tile area, but a small tweak I had calculated which tile any Orb was on correctly and efficiently.

Since I was now able to get which tile I was on I could now check what type of tile I was on. And calculate the friction of the surface. I then released that I haven’t given added the friction to the orbs movement yet. This took me quite some time to work out as most of the ideas I thought of had major flaws, or only work if moving in a positive axis direction. Though eventually I was able to figure this out. The solution was staring me right in the face the entire time. I just needed to take the current velocity normalise it the inverse the direction. Then multiply the inversed velocity unit vector by some friction value. I then subtract this friction vector from the current velocity and viola there I have it slowing down based on a friction value. After I had gotten the orbs to slow down over time and I was able to get which tile I was on I then updated the friction value based on which tile the orb was on.

The last thing on my agenda for today was the win and lose states, the win state was easy to implement as it was a simple check to see if only one orb was still alive.   
At first I thought that the lose state would be just as simple but I kept getting vector out of bound errors, it was at this time that I thought of a better idea for handling the destroyed tiles, instead of destroying them we can instead set a state on them, this way I can do I check on the tile that the orb is on and see if its alive or not, and if it’s not alive the orb has lost and therefore gets set to dead as well, and therefore ceases all processing and rendering of that orb. This required Callan to update his tiles accordingly, so we help each other with that and then I continued to implement the lose states for the orbs.

# 18 Sep 2015

## Scrum Meeting:

Today’s scrum meeting continued the tradition of the last few meetings with us just touching base with where each team member was with what they have done and what they planned to continue with.

As today is pretty much the final day we have to work together on the project until the final week we need to have most of the game playable by the end of the day. Which means I need to get the Orbs Colliding and bouncing off each other and implement the power–up affects as well.

UPDATE:

I called a second meeting to discuss some changes that I would like to implement with in the game to focus the core gameplay. I discussed with the team the idea of removing the power-ups (at least for the time being) and instead implement two abilities instead, a boost and a phase ability. This would focus our game back to a core gameplay and help us focus on trying to make the main game loop enjoyable. The team agreed that this was a good idea and allowed me then to focus on implementing the two proposed abilities.

## Tasks:

* Orb – Orb Collisions
* Bounce Forces
* Add powers ups

UPDATE

* Boost ability
* Phase ability

## Update:

First point of order was to calculate the Orb to Orb collisions. This wasn’t as hard as I initially thought it was going to be. As I remembered that I can do circle to circle collision detection instead of sphere to sphere collision detection.

After successfully calculating if two orbs where colliding it was time bounce them. At first I just swapped the velocities of the two orbs that collided. This kind of had a small bounced effect but not quite what I was hoping for. So I spent quite a bit of time trailing different was to do the bounce (such a different multiples of the velocities and various other techniques). None of them really worked as I wanted but I needed to move on to implementing the power-ups.

Though before I did that I asked Austin O’Brien to have a quick play and tell me what he thought about the game so far, he agreed with me that the bounce didn’t feel quite right but we also had a bit of a chat about what the core of our game was and how the power ups would affect the game. He suggested that we hold off on the power ups for now and have a think about what the core gameplay loop was, and think about instead of adding extra features instead try and develop the core gameplay loop.

So we continued to discuss what the core gameplay loop was? Where most of the game was being played on the arena floor and why. As well as what the actions where. After a few more play throughs we agreed that the battle between the players was the core gameplay loop, the baiting of players toward the edge and trying to dodge them and or counter attacking to get the other Orb of the arena. This arose the discussion of instead of power ups why not add in an active dodge/side-step ability and with that an active attack ability like a boost.

This is when I took these ideas back to the rest of the team. The team agreed with the changes, and that’s when the boost and phase ability was decided on.

Also During the discussion with Austin I changed the bounce mechanic to instead of swapping velocities I now calculate which orb has the higher velocity and then with that I impart a greater force to the orb with the lower velocity and for now just nulling the velocity of the Orb with the higher velocity.

So instead of adding in the power ups I now needed to add in these two abilities, since we didn’t have much time left I just implemented basic boost and phase abilities, with some kind of charge bar slash cool down. Though these abilities needs more balancing.

# 28 Sep 2015

## Scrum Meeting:

In today’s scrum meeting it was decided that we need to updates to the documentation. We discussed the different parts of the GDD and TDD that needed to be updated. It was decided that it would be best if the person who had worked on that particular section of the project would then update it in the documentation.

## Tasks:

* Update GDD – to incorporate the new changes
* Update TDD – to incorporate the new changes
* Help Callan with flow charts – esp. Game loop
* Help Juran with test cases in TDD
* Proof read and add the flow charts to the TDD

## Update:

We had completed most of the documentations todays. We just need to update the backlogs and burn down charts left to do. We had a pretty productive day getting through the documentations.

# Production Phase - Alpha Sprint Two

# Requirements

* The Updated Game Design Document
* The Updated Technical Design Document
* The Updated Product Backlog Sheet
* Beta Sprint Backlog
* Product Source & Release together with a Demo

# 05 Oct 2015

## Scrum Meeting:

Today’s scrum meeting we discussed the different updates that we need to complete with in the documents to get it ready for alpha sprint one submission.

## Tasks:

* Finish documentation for submission.

## Update:

We had spent the day working together to complete the remaining tasks required for the submission.

# 13 Oct 2015

## Scrum Meeting:

Today’s scrum we came together to discuss the remaining tasks (mainly features) that where left to be implemented into the game.   
The game was mostly done, most of the features were already implemented, a few fixes here and there.

## Tasks:

* Correct the Blue tile processing (loss of control)
* implement temp Victory Screen
* Spawn in correct locations
* Prevent Spawning tile to be slippery
* implement pause screen
* Implement the Rumble of the controller
* Implement Sound effects in the game
* Fix release build error
* No controller Connected

## Update:

The first thing I started with today was to implement the loss of control on the blue tiles, I had it just set to zero friction, at first I decided to change the handleInput() function to instead of looping thought the number of controllers to taking in the player number, to handle that players controller. Then within the process I check the tiles and only call the handleInput() function if a player is not on a blue tile.

With having added the loss of control I decided to spawn the players in the correct locations, I decided to spawn the players in the corners (In one tile). I initially went about overthinking the implementation, by trying to reverse engineer the calculation to find what tile a player is on, but most way through I realised that I can just take the position of the tile I want and set the Orbs position based on that and slightly higher, ended up being much simpler.

So having the Orbs spawned in the correct locations I could now test the Blue tiles properly, while trying to test the blue tiles I kept getting spawned on a blue tile so I decided to quickly fix that even thou it was a task assigned to Callan. This was a simple fix, as after all the tiles are created and spawned I added the functionality to change their type. Then changed the types of the four tiles that Orbs could spawn on and set them to a Standard tile.

Now I was able to test the blue tile more efficiently. While testing it I realised that it wasn’t working quite as well as I wanted to, also had a little bug where if you got stopped on a blue tile you couldn’t then move. So I decided to have another look at how to implement the loss of control. I decided to give the Orbs a tile, so that the Orb can decide whether or not it can update it acceleration based on the tile it’s on. So every frame I set the tile that the Orb is on to the Orb, then when the handleInput() function calls the setAcceleration() function the Orb then checks two things first check if the tile it’s on is not a blue tile, and then if it is checks its current Velocity if the velocity is 0 or the tile is not blue then allow the acceleration to be updated else do not update it. This was the final version of the loss of control as it worked quite well, it may need some tweaking, with the velocity check instead of zero maybe have a less than instead, because the movement is real slow when you get stuck on a blue tile. But more on playing with this is Beta phase maybe.

Next I decided to implement a temporary victory screen on the end of the game, most of the work for this was already done I just needed to render a victory screen. Since 2D assets and rendering was part of Juran’s tasks so I got him to give me a quick hand in just rendering a temp victory screen.

After having just render a temp version of the victory as Juran plans on implementing a UI manager class to handle the UI, I decided to move onto implementing the vibrations on the controller. Since we have an option of turning it on and off I needed to update the input class with a new variable of allow vibrate and have that dictate whether the controller vibrates. Then I set that variable based on the choice in the Options menu.   
Then actually implementing the vibrations took some time, it was either not vibrating or wouldn’t stop vibrating. But after a few trails I finally got it to work, by setting a timer on the vibrations, so it would vibrate for a half a second then stop.

While I was implementing the vibration I had a chat with A. Shakil about our release version bug to see if she had any input in being able to fix it. Sadly a fix for the bug was not found, and the bug is still in the game, we did however find out why it was happening, and it was due to the compiler optimisation, running the handle collision code in a weird way.

After adding the vibrations I quickly added the sound system that Juran created into the game and added the sound effects on the collisions and death of an Orb. Due to the way Juran wrote the sound system this was extremely easy to implement.

The Next step was for me to handle the scenario when the correct amount of controllers are not present. This took some time to implement, but in the end I got a rough version of it working by getting the HandleInput() to return a Boolean if that controller is not connected. Then if that is the case I stop the processing of the game and render a temp image to indicate that the controller is not connected. This worked but was still very basic.

So I decided to implement the pause system in game, so that I could utilise that when the controller is not connected. This however required a change to the way the game class would work, I decided to implement a state based implementation to the game class. Adding in the different states the game can be in and base the process on what state the game is currently in. With the state based implementation the pause system was much simpler to implement. So now when the player presses the start button the game state will change and render a temp image over the top, and not process the Orbs or tiles, but still handle input to get out of pause.

After having implement the temp pause system I got Juran to help me with implement the actual pause screen and functionality into the game, so the two of us worked together to get the pause screen that would pause the game return to the game and return to the main menu from the game. The instructions and options menu was not yet implemented.

# 15 Oct 2015

## Scrum Meeting:

Today’s scrum we realised that the game is still not feature complete, with one or two features still missing, as well as the code needed a little bit of cleaning up and commenting as well. The main feature that needed to be implemented was the game timers and the full screen option.

## Tasks:

* Tidy up Code
  + Switch cases on game states
* Add Comment headers

## Update:

Callan decided that he would take the task of implementing the timers, which gave me time to tidy up my code a little and add in any missing comments and comment headers.

I started with adding in the missing comment headers in the classes that I was working on. Which was found to only be in the Orb Class, so this didn’t take me too long

After that I Decided to clean the game class up a bit, first by removing the Tiles vector form the Game class, this required to add a few functions in the arena Floor class.

After having done that I moved onto cleaning up the process and render functions to have a switch case based on the game state instead of multiple if else statements. This took some time as there was a few places that need to be updated accordingly. While doing this I found a major flaw with my physics with in the game.

Spent the next few hours tinkering with the movement and Collision code, frustrated and getting nowhere I requested Callan help. Then spent a while trying to catch Callan up and help him understand what it was that I was doing. Together we tried to fix these issues but again both frustrated and still not getting anywhere. We would fix one area but the other would bug out. So we decided that we would spilt the task up, Callan had an idea with the collisions so he took on the task of handling the collisions, I took on the task of correcting the movement.

I spent some time trying different versions of movement calculations, but they didn’t work quite as well as I wanted. I decided to revert back to the original movement, then stripped it back to just moving, I removed the frictions and the boost. I then started tinkered around with the ordering and values but kept the same approach to the calculation, eventually I got to a point in where the movement felt semi correct, still not as well as I wanted but it was smooth enough. I then moved onto to getting the friction working again. After which I then moved onto getting the boost mechanic back in. The mechanic changed to be an increase in speed for a set amount of time after the button was pressed.

Then together with Callan we combine our codes and tested it. We had finally fixed the issue. It was a great victory.

# 16 Oct 2015

## Scrum Meeting:

Today’s scrum meeting we discussed the remaining tasks required before the deadline of alpha phase. This involved a few bug fixes and the final merge.

## Tasks:

* Fix tile edge
* Fix Controller Disconnected
* Fix Physics
* Update Boost
* Merge
* Presentation

## Update:

First task for today was to fix a small bug to do with the calculating which tile the Orb was on as there were certain places that an orb would not die. The fix was to do with converting a float to an integer and loosing data. Easy fix.

While fixing the tile edge bug Juran mentioned a bug with the disconnected controllers. After looking at the bug, I immediately saw what the issue was, and the solution was a little complicated but simple to implement, and worked as intended with in the fish couple of tests.

We then got a few people to play test the game along with us and noticed a few bugs were still present with the physics along with Callan we made minor fixes to get the game to a playable state. Along with this edit we changed the boost ability again this time it felt more of a punch instead of a speed boost which gave it more of an attack type feel to it.

We then merged our codes together after that moved onto getting our presentation ready.

After the presentation we were given some good ideas to think about, as we move on to the next phase.

# Production Phase - Alpha Sprint Two

# Requirements

* The Updated Game Design Document
* The Updated Technical Design Document
* The Updated Product Backlog Sheet
* Test Log
* Gold Sprint Backlog
* Product Source & Release together with a Demo

# 27 Oct 2015

## Scrum Meeting:

In today’s scrum meeting we discussed the testing plan forward.

## Tasks:

* Conduct initial end to end testing of the game
* Update acceptance Test plan with any new test

**Bugs to Fix**

* Stop Vibration on Death
* Bring Up Pause menu When Controller is plugged out

## Update: