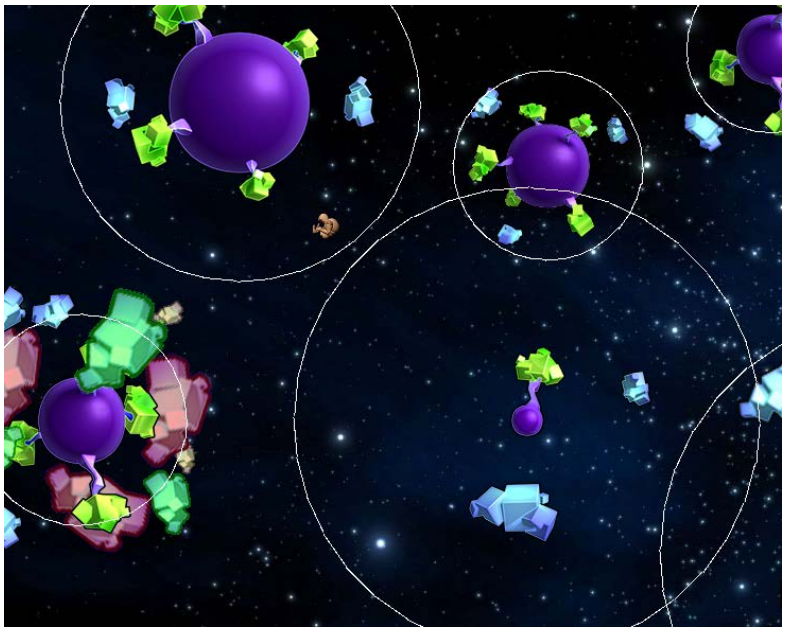
Game and Technical design - Cubelicious

Game Mechanics and core game play

The player jumps between planets scoring points for each completed jump. Tension and a win condition is added through a simple game timer.

To add challenge to the player the player can run completely around each circle planet, with some jumps to the next planet only possible through running and jumping from the correct angle.



Each planet has its own visible gravity well that allows the player to see where possible inter-planet transitions are possible.

Game Physics and player movement

There are two types of controls that affect the player, movement in space and movement on the player. Movement on the planet is defined by a constant speed and calculations to allow the player to move around the circle of the planet.

Movement in space is defined by a custom simple physics solution that allows the player to build up velocity based on key/touch press length.

User interface

The user interface is minimal to avoid taking away from the beauty and simplicity of the game. The score and current time are displayed in the bottom left of the screen for easy sighting.

Functional Requirements

1. Splash screen state

Allows behind the scenes loading to take place and lets the user know the game is ready to go

2. Main Game state

The main game, the player is dropped onto a planet at the left most of the screen, they will have to make their way through the level to gain scores and achieve a place on the leader board

3. Game over state

Checks if the player has achieved a high score and displays the players result

4. Add score screen state

Accepts a name from the player to save against their score

Cross Platform

This project is capable of being built for Playstation Vita and Android devices. It does this by having three projects.

To achieve this I have used Monogame, Xamiran and PSM studio.

1. VS project
    a. Contains core code and builds for PC
    b. Uses straight Monogame
2. PSM
    a. Contains references to VS project
    b. Contains PSM wrapper
    c. Uses PSM SDK and Monogame
3. Android
    a. Contains references to VS project
    b. Contains Android wrapper
    c. Uses Xamarin for deployment

Considerations

The project had to be able to handle three different input methods, to work with this simple movement functions where created such as:

- MoveLeft
- MoveRight
- Jump

This allowed each custom input method to call these base movement functions with ease.

Android is purely touch, I opted to split the screen into thirds

1. Move left
2. Jump
3. Move right

This allows for intuitive gameplay.

Signed and approved by:

Date:

Name:

_____

Signature:

_____