Design Document

The world is yours

Version 4.0

Gustavo Martins

Document Overview

Overview

This is the design document for *Planets!*. This document is broken into major sections, each describing a particular aspect of the game. The appendices offer supporting information for the development of the game.

Revision History

This is a brief description of this document. A list of the major changes is provided following each revision number. Furthermore, a list of any outstanding topics or any topic that needs further details is provided.

Revision	Date	Change Description
1.0	May 9, 2012	• First Draft Pass.
2.0	May 23, 2012	• Gameplay and Artificial Intelligence sections added.
3.0	Jun. 6, 2012	• Game Modes and Heads Up Display sections added.
4.0	Jun. 18, 2012	• Table of Contents, Figures and Tables added, Appendix section added, Instructor feedback incorporated.

Table of Contents

Table of Figures

Table of Tables

Game Concept – Planets!

Overview

Everyday a world is born, and life follows soon afterwards.

High Concept

The world is yours

Planets! is a 2D life simulation game for iOS and Android devices that puts the player in control of an entity called the Designer. Tasked with creating a new world, and thus create life, the players will make the calls regarding the planet's initial characteristics, ranging from mass, size, gravity, temperature, composition, and watch as the events of creation unfold. Once life appears on the planet, the player will assume the role of Nature and will guide the life form to new heights via a method of goal setting, which will set the parameters for the artificial life to follow, guiding them across a technological tree towards the ultimate goal.. Players also have the options to utilize one of Nature's many abilities in order to aid their artificial life, such as the ability to make rain and other climatic events. The final objective of the game is to evolve life to an advanced enough stage in order to be able to initiate a space program and reach the stars.

Common Questions

Why Create this Game?

The game is being created as a means to research emergent behavior in artificial intelligence. While the game features 2D graphics similar to those found in many iOS games, the artificial intelligence under the hood is extremely complex, simulating a population of semi-sentient beings with growing understanding of the world around it. The AI will be derived from S.T.A.L.K.E.R.'s A-Life engine, inasmuch as it will feature a similar system of goal generation and agency to achieve those goals. This will be covered in more detail in the respective section. The game will also be used as a tool to evaluate a possible approach to building and evolving AI. The game is a form of sandbox that is not encountered in the App or Android stores. It is a reminiscent of the old Sim games of the 90s, and will fill a potential hole in the current market, as well as provide entertainment for the player's idle hours, opening a literal world of possibilities. The goals set forth in creating this game include, but are not limited, to the following:

- A new way of experimenting and researching AI and emergent behavior.

- A sandbox game unlike anything currently on the market.
- Combines gameplay with real world physics and psychology, allowing players the possibility to experiment with their own theories and ideas of perfect worlds.
- Increase awareness of artificial intelligence and how they can impact our lives, as well as encourage people to venture further in this field.

What is the Target Hardware?

iOS and Android devices.

Who Is The Target Audience?

ESRB: Everyone (E)

Geographic: The game is aimed primarily at North America and Europe due to high iOS and Android adoption, as well as the possibility to use English as the language for the product, cutting initial costs.

Demographic: The game is aimed towards teens aged between 13 and 17, as well as young professionals who have just started their careers, and have some spare time that they need to fill. They have their own cellphones, either provided by their parents or themselves, and either live at home with reasonable allowances or have enough spending cash each month.

Psychographics: They are used to playing casual games on their phones often, are more interested in playing alone but able to show off their achievements to others, and are consistent web users.

Behaviors: Will use the internet and contribute to discussions regarding the games and their experiences. Optimally, they will also be interested in research regarding artificial intelligence.

Feature Set

This section describes the features that make up the overall gameplay experience of Planets!.

Major Features

Endless possibilities for world generation

At the start of the game, players will have access and will be able to tweak several planetary variables that will have direct impact in how the game plays out, including size, mass, composition, gravity and temperature. A unique world will then be generated based on the player's selection.

Real world metrics

The resources available to a planet, the possible personalities of the artificial life and the innumerous evolutionary paths a player can guide their species through are all based on real world theories and metrics, making them not only more solid as a system overall, but also more interesting.

- **Resources:** A bigger planet will inevitably have more resources than a smaller one, but it's larger gravitational field will hamper a species' evolution as well as make it harder to achieve flight.
- **Personalities:** The overall population of the planet will be divided into personality categories, each with their own modifiers towards their personal goals, as well as stances on conflicts and other happenings.
- **Environment-dependent Evolution:** Evolution follows the laws of biology and physics, encouraging the player to create multiple worlds in order to see what Nature can come up with. A sea planet will be more likely to spawn a water-based species than a desert planet.

Ever evolving artificial life

The artificial life featured in the game is controlled by a complex artificial intelligence system that encompasses several algorithms in order to expand its functions and behaviors.

Paradigm	Use	Effect
Global and Local Goals	Governs the motivation of the artificial actors.	Allows them to consider if their current course of action benefits themselves or the society better, and acts on it.
Recursive Algorithm	Governs the resolution leading to actions.	Allows the AI to react better to changes in their environment, perform better actions more often.
Evolutionary Programming	Governs the source code for the individual artificial actors, dictating how they should develop and what should change with every generation.	Allows for a more organized way of coding, streamlining the entire process, making it easier to troubleshoot and displaying a true evolution in the artificial life.
Neural Network	Governs the distribution of goals and conflict resolution.	Allows the artificial life to be more easily manageable by the processor, focusing the efforts on a single entity that in turn governs the results of the remaining actors.

Minor Features

Open ended

In *Planets!* the player starts with a planet created by himself, and over the course of the game unlocks the ability to explore his star system. *Planets!* possesses only one goal: Colonize another planet. Once that has been achieved, the player can continue playing and colonizing more planets, but the game never truly ends. Players can simply start new worlds in order to experiment new ideas.

Pick up and play

Planets! is very accessible to start playing, and play can be stopped and resumed at any time. Players have the options to let the game continue the simulation in the background, utilizing only a small amount of computational power but speeding up the growth of the game. Due to the self-reliance that the artificial intelligence will have, the player does not need to be constantly micromanaging everything, making this game an ideal candidate for the casual market.

Game Characters

Overview

This section describes the main character, the NPC's as pickups and obstacles found in *Planets!.*

Playable Character(s)

Overview

There are two playable main characters in *Planets!*, the Designer and Nature.

Main Character – The Designer

The designer is the first playable main character in the game, and is responsible for creating the planet the game will take place on. He accounts for the first encounter with the game, as well as for the smaller portion of it. He is able to tailor the world to his visions, having complete control over the variables available at creation. His decisions are very simple to effect, requiring only he push of a button or the slide of a slider, but can have a long term impact on gameplay. He is the personification of an omnipotent God, however he is not seen during gameplay.

Figure 1: Concept Art for The Designer.

Main Character – Nature

Nature is the second playable main character in the game, and is responsible for guiding the artificial life on the planet the game takes place on. She accounts for the remaining portion of the game. She has the power to interact with the environment on the planet, sending events that will either help or harm the artificial life living on it. As the life progresses, more options are unlocked, allowing Nature to have a direct impact on the evolution of life, being able to set species-wide goals for it, which will serve as the basis for the further development of life. She is the personification of Mother Nature; however she is not seen during gameplay.

Figure 2: Concept Art for Nature.

Non-Playable Character(s)

Overview

Non-playable characters consist of the life that will appear in the planet created by the player. They are not directly playable, but their existence will be impacted upon by the player's decisions as the game progresses. These are divided into Main Actors and Supporting Cast.

Main Actors

The main actors are the artificial living beings who possess an advanced artificial intelligence, and are the main focus of the game. They are randomly created depending on the planet's characteristics, which will dictate how they look and their base psychology.

Figure 3: Concept art for Creatures.

Supporting Cast

The supporting cast is composed of characters that don't directly interfere with the gameplay, and are added only to add more diversity to the artificial intelligence. They make up the rest of the life on the planet, and keeping a balanced planet will be sure to produce a larger diversity of species on it, which will lead to overall happier main actors.

Figure 4: Supporting Cast Dog Concept Art.

Enemy Character(s)

Overview

Planets! doesn’t features enemies in the traditional sense. Instead, it features disaster events that the can either be summoned by the player or by chance. They are divided into Planetary Disasters and Cosmic Disasters.

Planetary Disasters

Planetary disasters are disasters that affect the environment and are caused by interior agent.

Table 1: Table of Planetary Disasters.

Planetary Disaster	Effect
Flood	Reduces population.
Drought	Decreases food reserves.
Global Warming	Multiple effects.
Pandemic	Reduces population by a large amount.
Genetic Mutation	Has a random effect on the next generation.

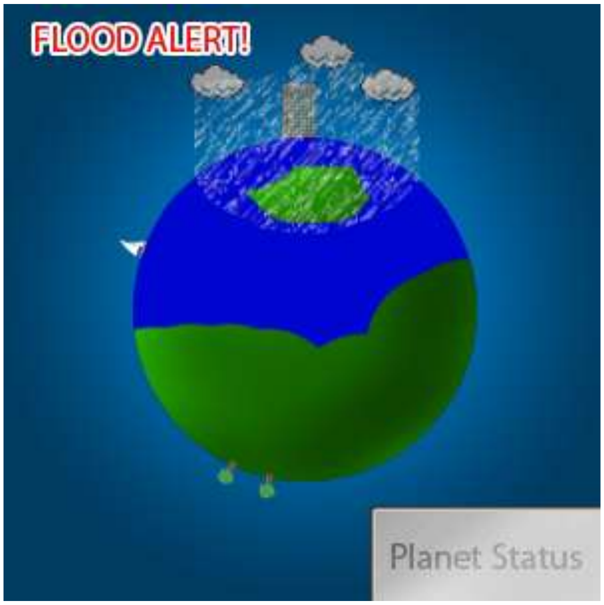


Figure 5: Flood Concept Art.

Cosmic Disasters

Cosmic disasters are disasters that affect the planet as a whole, and are caused by exterior agents. They usually feature devastating results, such as a meteor that wipes life off the face of the planet, but can be avoided with proper research and warning.

Table 2: Table of Cosmic Disasters

Cosmic Disaster	Effect
Meteor	Exterminates part of the population.
Black Hole	Destroys the planet.
Asteroid	Increases the population’s fear level.
Solar Flares	Temporarily reduces technological progress.
Planetary Collision	Destroys the planet and part of the Star System.

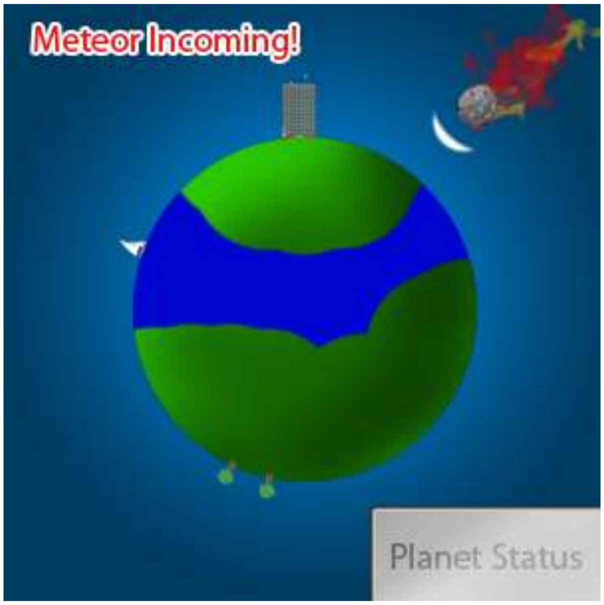


Figure 6: Meteor Concept Art.

Camera

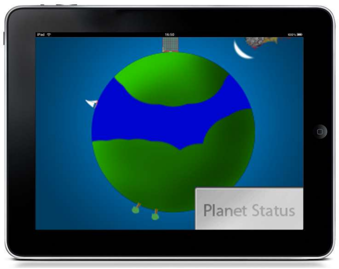
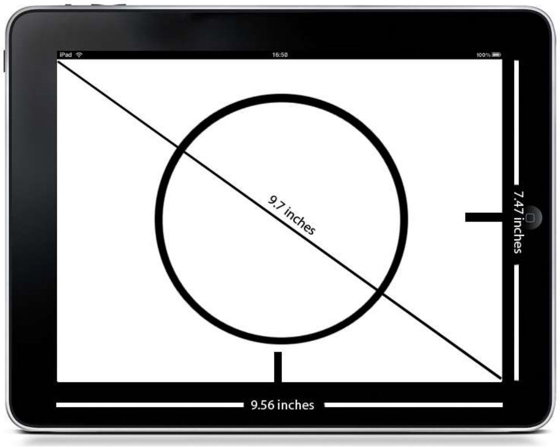
Overview

The camera system is designed around a first-person perspective, from the player's point of view, looking at the world ahead of them.

Main First-person Camera

The object of interest of the first-person camera in *Planets!* is the planet itself. Therefore, the camera will have its focus locked on the planet. The size of the planet is irrelevant to the camera, inasmuch as the camera will keep the ratio of planet to empty screen constant regardless of the planet size. The difference will be in the level of detail observed in the planet. The planet will be perfectly centered on the screen. It's center will be positioned according to half the screen's width for X and half the screen's height for Y. The distance between the edge of the planet and the edge of the screen height will be approximately 10% of the total screen height. This camera is static, and the player can't move around or zoom into the planet.

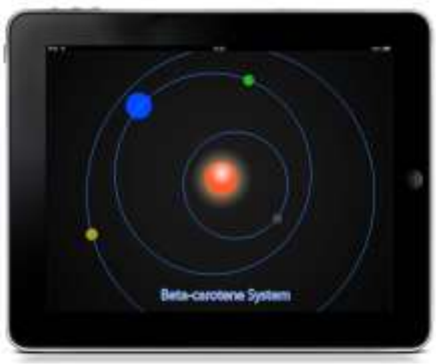
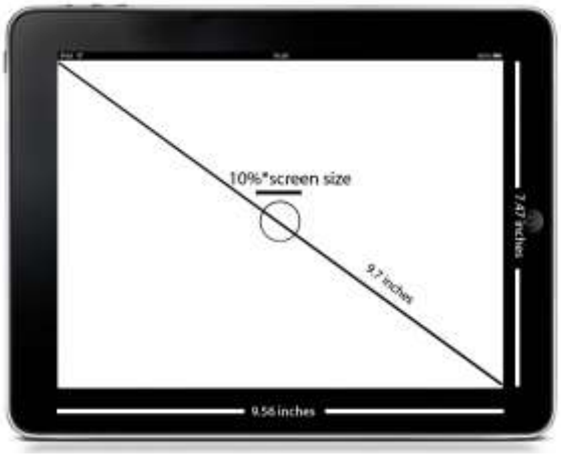
Table 3: Table of Main Cameras

Camera Example	Description
	**Main First-person Camera** Mockup showing the approximate position of the planet on the iPad screen.
	**Main First-person Camera** Schematics showing the position of the planet on the iPad screen.

System First-person Camera

The galaxy first-person camera is the view that appears when the player navigates the system map in order to find new planets. It displays a top-down view of the system the player is currently in, viewed from a first-person perspective. The camera will be centered on the central star of the system, which will always occupy an area equivalent to a circle with diameter equaling 10% of the total screen real estate, and will always be positioned in the center. This camera permits some panning movement across the x and y axis, but will not permit zooming into planets and will always bounce back to its original position if the player pans to a region out of bounds, and alternatively can bounce back to its original position by double tapping the screen.

Table 4: Table of System Cameras

Camera Example	Description
	**System First-person Camera** Mockup showing the approximate position of the central Star and its orbiting planets.
	**System First-person camera** Schematics showing the position of the central star on the iPad screen.

Controls

Overview

Planets! is built with the touchscreen interface of Android and iOS devices in mind. Therefore, all the action in the game is commanded via touch inputs.

Selection

Selection is the main action available for the player. It is the root of all the action, and as the name says it selects what action the player wishes to perform. Selection is performed by touching the desired spot on the screen and then releasing it. Holding the touch down for a longer period of time has no side effects.

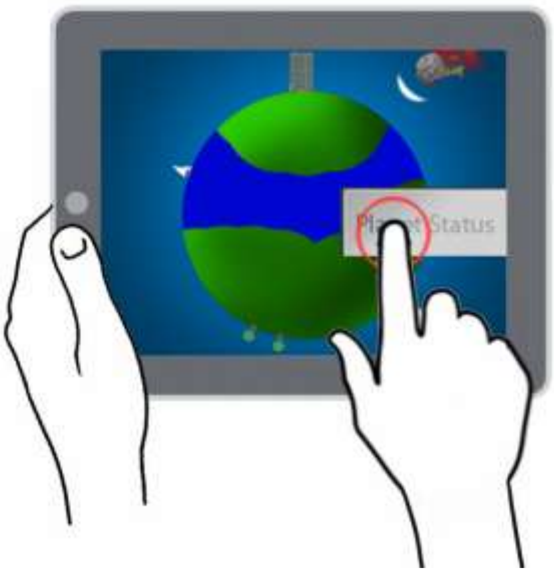


Figure 7: Selecting a button by touching.

Pan

The pan action is accessible on the Star System Map. It allows the player to pan the camera around to see the rest of the Star System. Pan is performed by touching a spot on the map, holding it down and sliding the finger on the desired direction, while still touching the screen.

When the player ceases the touch, the camera will either stay in the same place if the area is not out of bounds, or it will bounce back to its original position if the area is out of bounds.

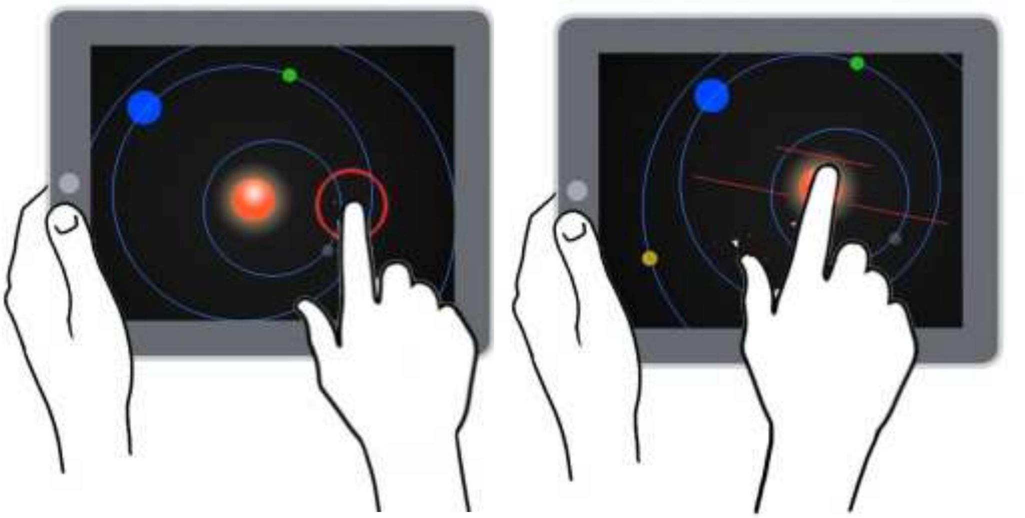


Figure 8: First the player taps the desired spot then he slides his finger towards the desired position

Return

The return action is accessible on any screen on the game, and allows the player to return to his previous screen. This is useful for navigating quickly back from the map or from a menu, when the same does not take place automatically. To return from a menu, the player must tap the screen twice with his finger, with a delay of at least half a second between the first touch stopping and the second touch commencing.

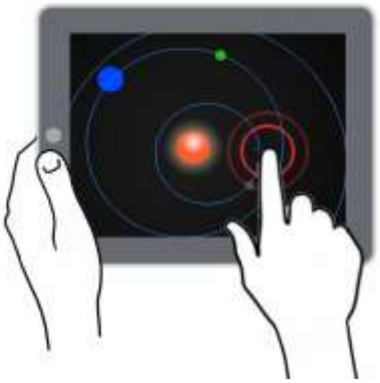


Figure 9: The return action being performed. Players must separate their two touches by at least half a second.

Gameplay Details

Overview

This section describes the gameplay and mechanics found in *Planets!*, and contains topics describing the mechanics, how they are used, and their implementation into the game.

Planet Creation

The first piece of gameplay the player will encounter is the Planet Creation. This is the first vital step the player takes into his game, as the decisions made here will have an extreme impact on the remainder of the game. To that end, the player has a number of characteristics they can tweak when creating their planets. The ratings generated by the planet creator do not affect the creation of the planet itself, but will impact the outcome of the events that will happen in the future.

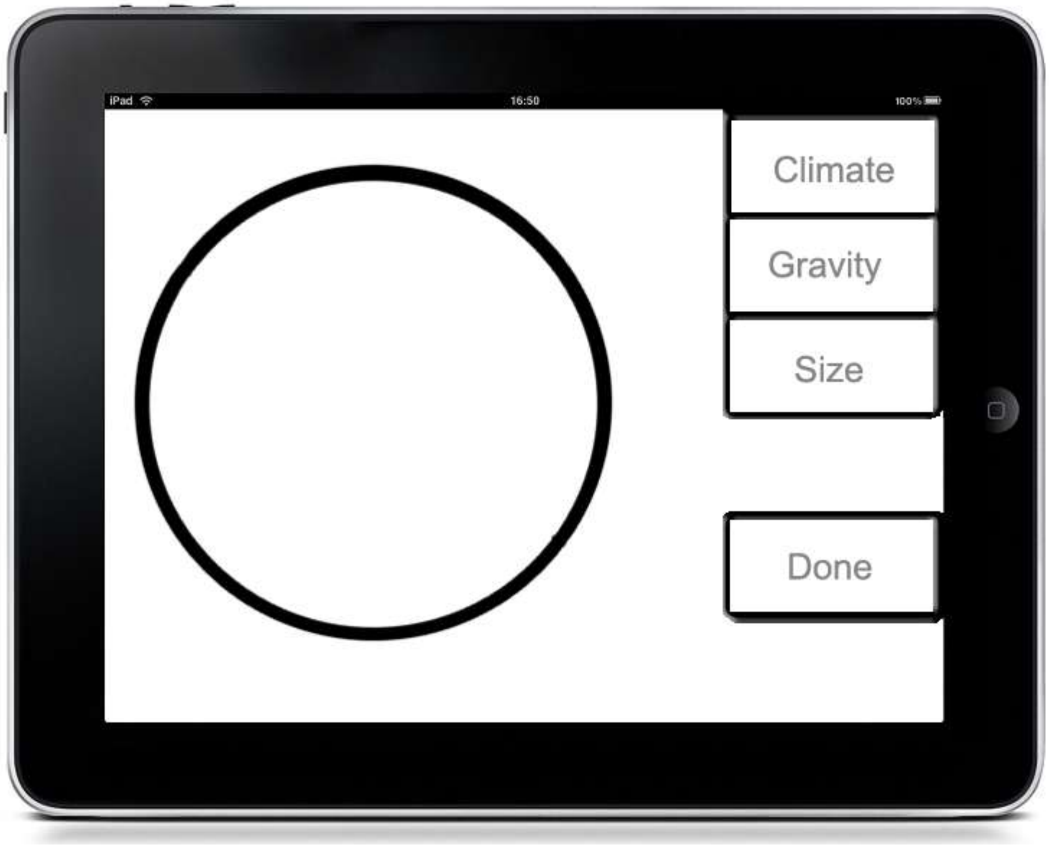


Figure 10: Schematics showing the planet creation screen.

Climate

The climate of the planet dictates how easy it is to survive in that environment. This will have a direct impact on natural selection, as the harder it is to survive in an environment, the harder it is for weak members of a species to survive. The player tweaks the climate by selecting its average temperature, which is given in Celsius or Fahrenheit, depending on the player’s preference. Planets are assigned a numerical rating according to the value selected for the planet’s climate.

Table 5: Table of Climate

Climate	Rating	Definition
<= -10^{c}	1	Frozen
> -10^{c} and <= 10^{c}	2	Cold
> 10^{c} and <= 20^{c}	3	Average
> 20^{c} and <= 35^{c}	4	Tropical
> 35^{c}	5	Desert

Players can define the climate by means of using the touching the on-screen thermometer and sliding his finger to a desired position on the scale. The information on climate, rating and definition will be updated as the player changes the temperature on the thermometer. Sliding the finger up increases the temperature, while sliding the finger down decreases the temperature.

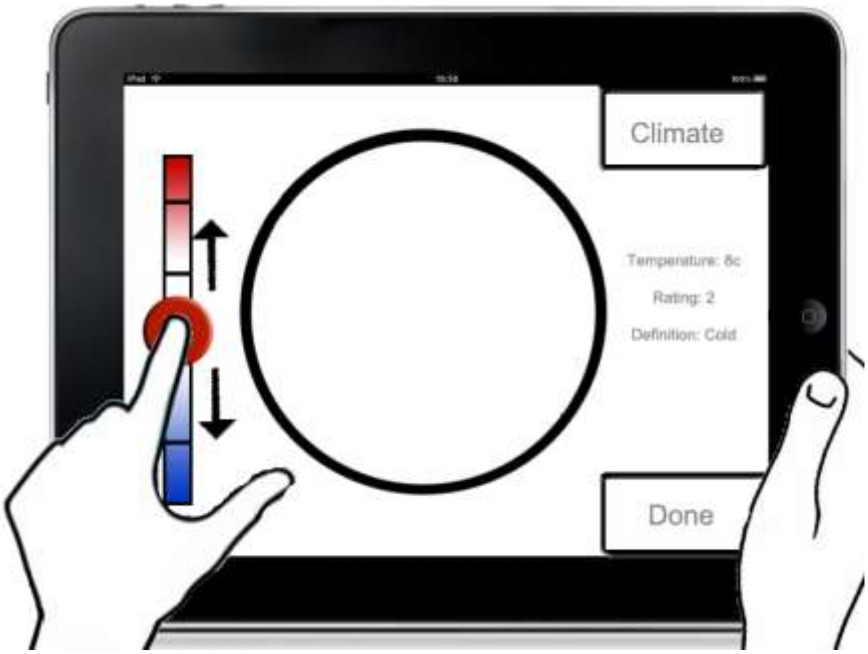


Figure 11: Gestures for decreasing and increasing temperature.

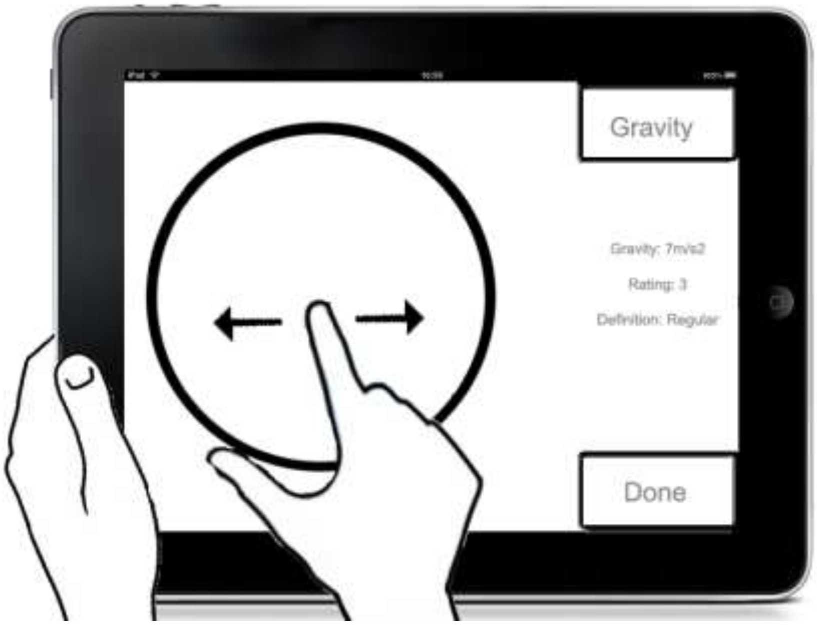
Gravity

The gravity of the planet dictates what sort of life will be developed on the planet. It also has an impact on how hard it is for life to survive on the planet, impacting the evolution of the species in different ways. Planets are assigned a numerical rating according to the value selected for the planet's gravity. The unity for gravity is m/s^2.

Table 6: Table of Gravity

Gravity	Rating	Definition
<= $1m/s^2$	1	Weightless
> $1m/s^2$ and <= $5m/s^2$	2	Light
> $5m/s^2$ and <= $15m/s^2$	3	Regular
> $15m/s^2$ and <= $20m/s^2$	4	Moderate
> $20m/s^2$	5	Heavy

Players can define the gravity of the planet by means of using the swipe gesture on the planet to generate speed. The information on the gravity, rating and definition will be updated as the player changes the speed of the planet. Swiping left or right increases the speed in that direction, therefore decreasing it in the opposite direction.



Gravity: 7m/s2
Rating: 3
Definition: Regular

Figure 12: Gestures for decreasing and increasing planet gravity.

Size

The size of the planet dictates how much life it is capable of sustaining. Planets are assigned a numerical rating according to the selected size for the planet. Planet size is provided using either kilometers or miles, according to the user's preference.

Table 7: Table of Sizes

Radius	Rating	Definition	Population Modifier
<= 100km	1	Dwarf Planet	-50%
> 100km and <= 1.000km	2	Small Planet	-20%
> 1.000km and <= 10.000km	3	Average Planet	0%
> 10.000km and <= 50.000km	4	Large Planet	+30%
> 50.000km	5	Titanic Planet	+100%

Players can define the size of the planet by means of using the pinch gesture on the planet to increase or decrease the size. The information on the radius, rating and definition will be updated as the player changes the size of the planet. Closing the pinch decreases the size of the planet, while opening the pinch increases the size of the planet.

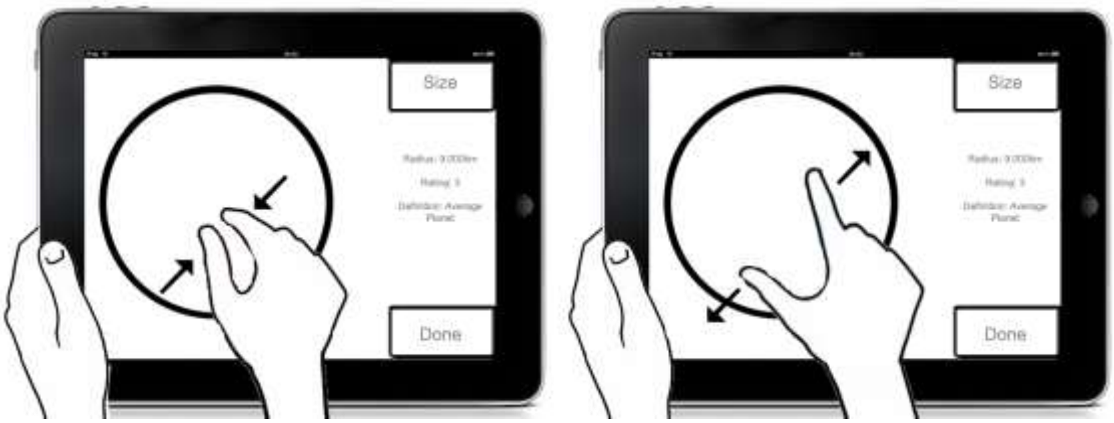


Figure 13: Gestures for decreasing and increasing planet size.

Life Generation

Life generation is the next step in gameplay. During life generation, the game attempts to create life based on the planet ratings until it manages to find a form of life that can sustain its population. The player is largely a spectator on this step, watching the game come up with different life forms and the statistics concerning their status. The game never fails to produce a life form, no matter how much of a disadvantage it will have in the world. Life generation follows the procedure stated in the following flowchart:

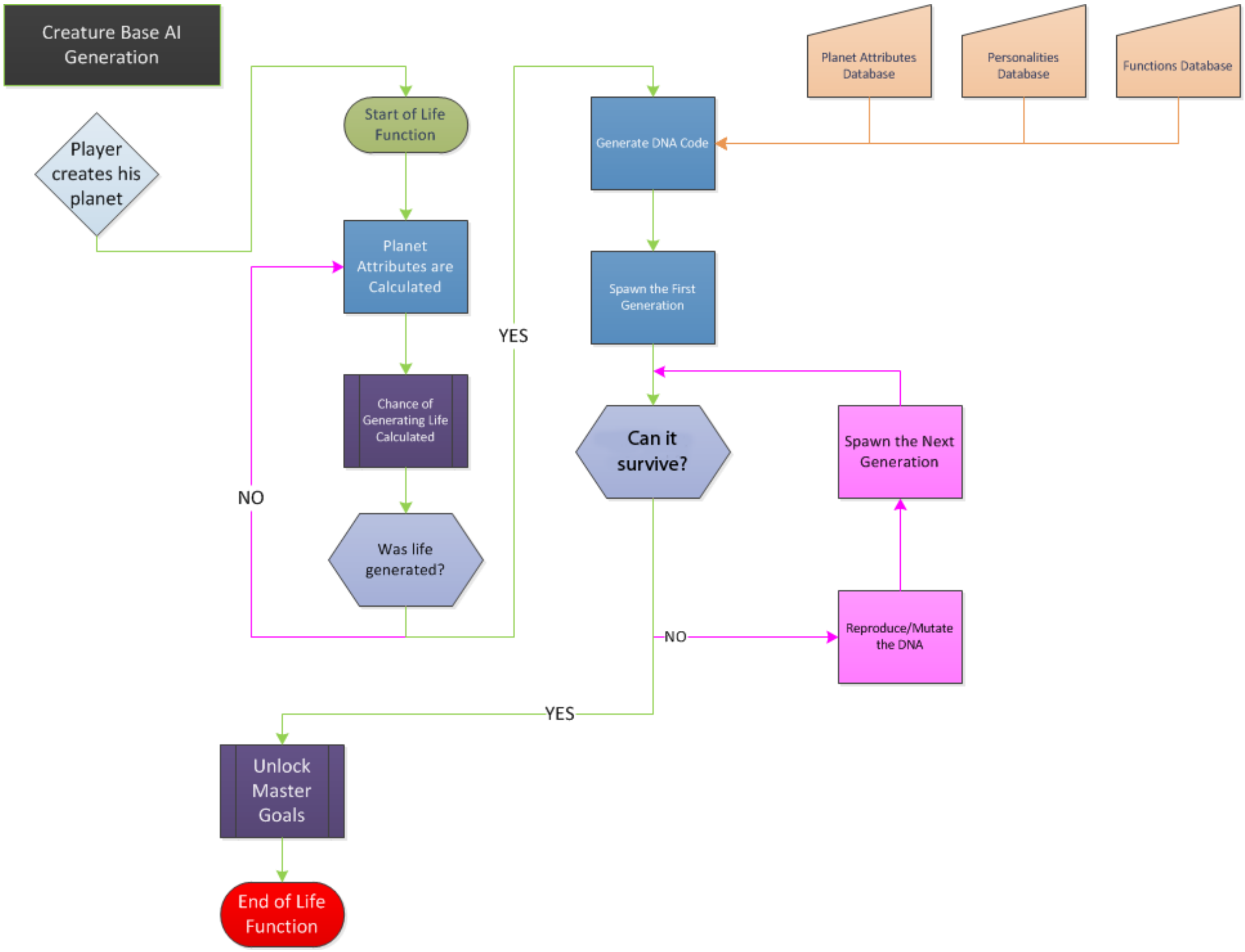

Creature Base AI Generation
Player creates his planet
Start of Life Function
Planet Attributes are Calculated
Chance of Generating Life Calculated
Was life generated?
NO
YES
Generate DNA Code
Planet Attributes Database
Personalities Database
Functions Database
Spawn the First Generation
Can it survive?
Spawn the Next Generation
Reproduce/Mutate the DNA
NO
YES
Unlock Master Goals
End of Life Function

Figure 14: Flowchart depicting the generation of life.

Calculation of Planet's Attributes

The first step after the function has started is the calculation of the planet's attributes, using the planet's ratings as a base. The attributes being calculated are:

-Ease of Spacefaring:

Ease of spacefaring is governed by the gravity characteristic, and it dictates the difficulty of reaching space in the future. Varies from 1 to 10, and increases the time needed to research space colonization by 5% per rating.

-Hardiness:

Hardiness is governed by the gravity characteristic, and it dictates how physically strong the life forms can be. Varies from 1 to 5, and gives a bonus of 1 per skill level to the intensity of the physical event check.

-Weather Patterns:

Weather patterns are governed by the climate characteristic, and it dictates what sort of weather events can happen on the planet. Varies from 1 to 5, and adds a bonus of 1 per skill level every time an event is done that includes a weather.

-Hospitability:

Hospitability is governed by the climate characteristic, and it dictates the toughness level of natural selection, impacting on the evolution of the species. Varies from 1 to 5, and gives a bonus of 1 per skill level to the intensity of the health event check.

-Year Length:

Year length is governed by the size characteristic, and it dictates how long it takes for the year to pass in the planet in relation to Earth, as well as the length of every generation.

-Quantity of Life Forms:

The quantity of life forms is governed by the size characteristic, and it dictates how many possible life forms the planet can have, impacting on the overall psychological health of the species. Varies from 1 to 10. Quantities of species in a planet is determined by the following formula:

Planet size rating * (random value between 10 and 200) * (quantity of life forms integer)

Calculation of the Chance of Generating Life

After the planet's attributes have been calculated, the function progresses on to attempt to generate life based on them. It weighs the numerical values of the characteristics and outputs a percentage value. This value is then presented to the player, along the option to generate life based on it and the option to recalculate the planet's attributes. If the player chooses to generate life, the function generates a random number between 1 and 100. If the number is bigger than the chance of failure, which is the percentage value minus 100, then life is successfully generated. Otherwise the player is prompted to either retry to generate life or recalculate the planet's attributes.

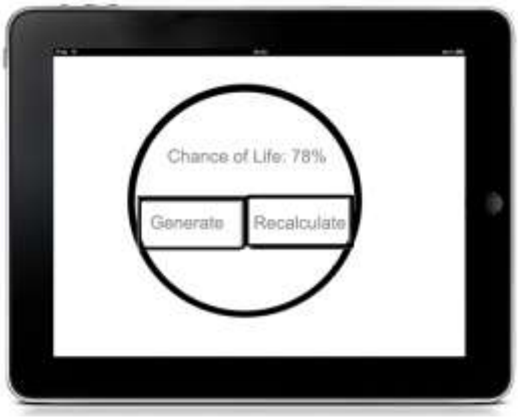


Figure 15: Scheme depicting the possible outcome and choices after the calculation of the chance of generating life.

DNA Code Generation

Once life has been successfully generated, the function moves on to generate the DNA code that will be used as a base for the species. It follows the following flowchart:

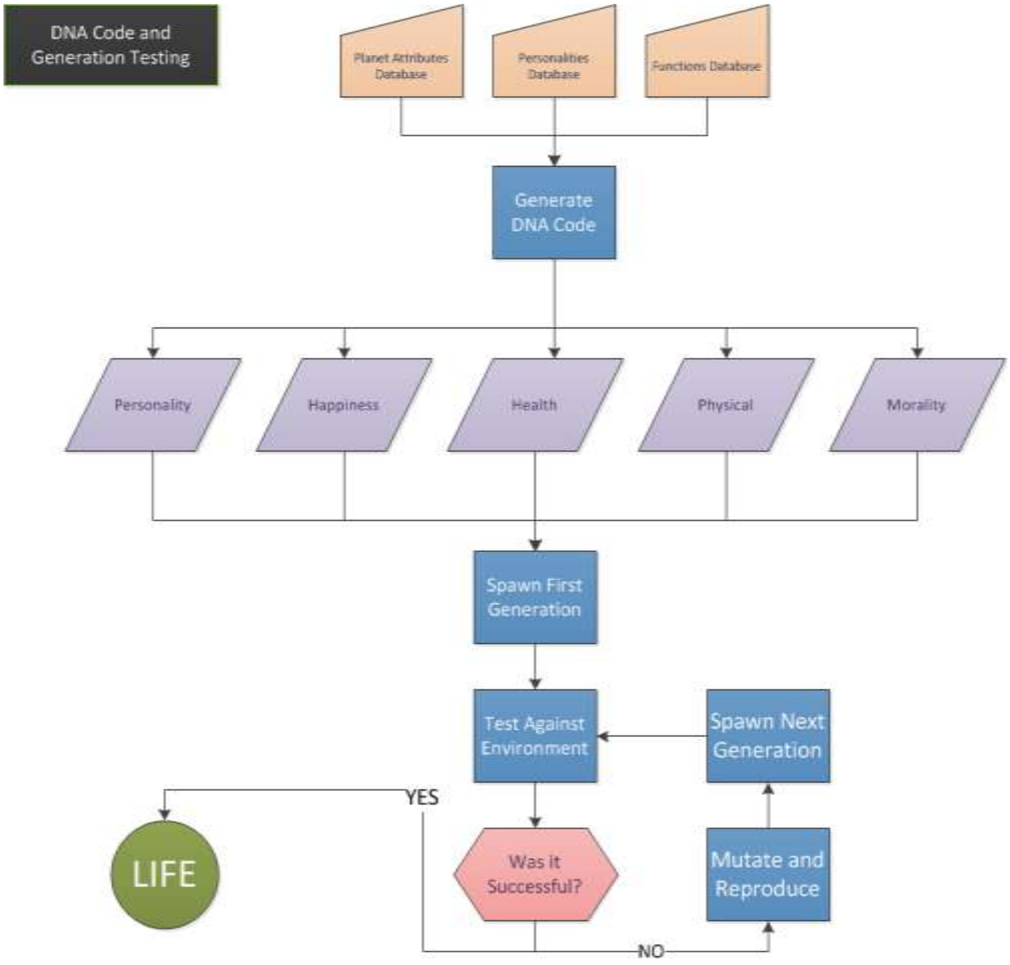


Figure 16: Flowchart depicting the generation of DNA code.

-Databases

The first step in generating the DNA code lies in referencing the three databases in the game. The data acquired from them will determine the parameter for the generation of the DNA.

-Planet Attributes Database:

Encompasses the attributes calculated at the earlier stage of the life generation algorithm.

-Personalities Database:

This database encompasses a list of first names, last names and personality types, which are added to a creature. First and last names are only of cosmetic importance, serving as a way to distinguish between different creatures when they achieve something important. Personality types provide a bonus to a gene function when that function is calling, making it have the best possible effect. There are two different personality types for every gene.

Table 8: Table of Names

First Name	Last Name
Gustavo, Jonathan, Mario, Steve	McSmith, McMurdo, Clarkson, Angelis
Armin, Johanna, Maria, Mordecai	Pavlovsky, Tornado, Hurricane, Da Trinta
Kate, Ashley, Daniel, Jessica	Timtom, Tont, Fowler, Goosenakis
David, Bill, Ulf, Brad, Jean	Martini, Johnston, Worth, Kanamaki
Raphael, James, Carl, Tabitha	Clouveau, Schmitt, Bamba, Mikkelsen
Ryan, Anne, Angela, Caitlin	Jarvinen, Espen, Gonzalez, Holm
Mike, Nadine, Noelle, Emma	Kalopolous, Zapata, English, Explosion
Lana, Marvin, Yuri, Theresa	Schwarzchild, Kami, Tanasuka, Pongo
Boris, Anastasia, Peter, Nadia	Eremeev, Schilin, Aliev, Kapoor
Zander, Frank, Sarah, Lily	Heath, Pindor, Stone, Kunis

Table 9: Table of Personalities

Type	Gene Affected	Effect
Friendly	Happiness	Defaults to the best effect for a good base pair
Hostile	Happiness	Defaults to the best effect for a bad base pair
Evil	Morality	Defaults to the best effect for a bad base pair
Good	Morality	Defaults to the best effect for a good base pair
Loner	Personality	Defaults to the best effect for a bad base pair
Social	Personality	Defaults to the best effect for a good base pair
Resilient	Physical	Defaults to the best effect for a good base pair
Weak	Physical	Defaults to the best effect for a bad base pair
Healthy	Health	Defaults to the best effect for a good base pair
Unhealthy	Health	Defaults to the best effect for a bad base pair

-Effects

Each function or gene will have an effect that ranges in degree from best to worst. The range is also assigned a numerical value of 1 to the amount of possible effects, with 1 being the worst. If a creature has a personality type that affects the gene that would be affected by the event, then instead of throwing a random number at the player, it will instead default to the best or worst effect for that particular base pair.

For example:

-Environment Moral Action: Rating of 1

-Rating of 1 passed into the Moral gene sets the random number between 1 and 3, in this case it was a 1.

-1 is the worst effect that can happen to a creature.

-A random number between 1 and 3 is generated, to select the base pair that will be affected. In this case it was a 1, which for this case was the Rescue Kitten base pair.

-Ordinarily, the number created on step 2 would be passed into the base pair, which will then produce its effect, which would lead to the death of the creature, since the number was 1, the worst effect, and that effect was “Dies when rescuing a cougar”.

-However, this particular creature has the Friendly type of personality. In this case, instead of passing the number 1 from the calculation on step 2, it will default to the best effect, meaning it will pass in the number 3, which is the best possible rating for an environmental moral action of rating 1. So the effect would be “Rescues his own kitten”.

-Functions Database:

This database encompasses the list of possible functions that can be assigned to a DNA sequence. These functions are treated as base pairs for the genes. They are detailed in tables 9 through 13.

-Genes

The DNA sequence is a combination of 5 different genes. World events affect genes by passing in a number with which the genes will activate a function connected to the quality of the event. Each gene is composed of 3 base pairs which are binary with 1 determining a good base pair and 0 determining a bad base pair. Base pairs are the functions that are called whenever a world event happens. The outcome of those functions will affect the creature’s life. The genetic sequence is stored in array, to be accessed by the world functions at any given time.

Each gene is stored in the order: Personality, Happiness, Health, Physical and Morality. Each gene has a total of 8 different combinations of base pairs, for a total of 32.768 different combinations per gene sequence.

A base pair is in effect a bit of information, being either 1 or 0, that will determine which function will be activated when that base pair is called. It is generated randomly on the first generation of creatures, and from then on it is inherited by its parents. On generation, there’s a 50% chance it will either be a 1 or a 0, 1 being a good function and 0 being a bag function.

When a base pair is activated, there is a 0.01% chance that a remarkable consequence will happen, engraving the name of the creature that triggered it in the history of the planet.

The number provided by the environment is utilized to set the threshold for the generation of the random number to be utilized to decide the effect on the creature.

For example:

-Environment Happy Action: Rating of 5

-Rating of 5 passed into Happiness gene sets the random number between 4 and 5, in this case it was a 4

-A random number between 1 and 3 is generated, to decide which of the 3 base pairs of the happiness gene will activate. In this case it was a 2

-The random number from step 2 is passed into the happy function

-Result: Creature "X", with gene sequence 100-010-110-011-111, performs Tell a Joke function, and gets the result "Tells the funniest joke". The gene sequence is detailed in the next item.

-Event is over

NOTE: The rating for the environment action is taken from the planet's ratings. It will take in the planet's gravity, climate or size rating, and use that for the action rating. For instance, a planet with a gravity rating of 4, climate rating of 3 and size rating of 1 would only be able to activate environment actions with ratings of 1, 3 or 4.

-Gene Sequence Generation

The Gene Sequence is first generated from scratch when the first generation of creatures is created in the Generate DNA Code step of the flowchart. It is an array that stores 15 values that are either 0 or 1. On the first generation, each value of the array is generated randomly, with a 50% chance that it will be a 0 and a 50% chance that it will be a 1. Afterwards, the sequences are generated based on the parents of the creature, as detailed in the Crossover and Gene Conversion sections of this document.

-Personality Gene:

It is the gene that responds to world events pertaining to personality. It has the following possible base pairs:

Table 10: Table of Personality Functions

Function	Quality	Effect(Best to Worst)
Throw a Party	Good	-Throws a birthday party for a friend -Throws a party for friends and family -Throws a party for the dog -Throws a party but no one shows -Hangs self at a party
Drink alone	Bad	-Drinks water alone -Drinks wine alone -Drinks beer alone -Drinks whisky alone -Drinks self to death
Give a Speech	Good	-Gives a graduation speech -Gives an emotional speech -Gives a motivational speech -Gets shot during a speech
Heckle a Speech	Bad	-Heckles a motivational speech -Heckles an emotional speech -Heckles a graduation speech -Gets shot by the mob
Write an Article	Good	-Article for Clock magazine -Article for a newspaper -Article for Sci-Magazine -Article for Hustler -Papercut to death
Graffiti a Wall	Bad	-Graffiti of a cartoon -Graffiti of a motivational line -Graffiti of a swear word -Death by ink

-Happiness Gene:

It is the gene that responds to world events pertaining to happiness. It has the following possible base pairs:

Table 11: Table of Happiness Functions

Function	Quality	Effect(Best to Worst)
Eat Ice Cream	Good	-Eats ice cream with his entire family -Eats ice cream with his friends -Eats his favorite ice cream -Eats ice cream alone -Suicides over ice cream
Kick a Dog	Bad	-Kicks a very small dog -Kicks a large dog -Kicks a rabid dog -Is killed by a tiger
Tell a Joke	Good	-Tells the funniest joke -Tells a “knock-knock” joke -Tells a “Yo-Mamma” joke -Is murdered by hecklers
Swear at Someone	Bad	-Swears in his mind -Swears at kids -Swears at a deaf person -Is murdered by a gang of skaters
Play	Good	-Plays a video game -Plays a sport -Plays a board game -Dies after exhaustion
Trick	Bad	-Trick or treats on Halloween -Tricks a friend -Starts a Ponzi scheme -Dies after a prank goes wrong

-Health Gene:

It is the gene that responds to world events pertaining to health. It has the following possible base pairs:

Table 12: Table of Health Functions

Function	Quality	Effect(Best to Worst)
Eat Vegetables	Good	-Eats freshly picked vegetables -Eats canned vegetables -Eats spoiled vegetables -Dies of food poisoning
Eat Junk Food	Bad	-Eats freshly made burgers -Eats yesterday's burgers -Eats bacon-fried fat -Dies of obesity
Take Medication	Good	-Takes the right medication -Takes the wrong medication -Overdoses on barbiturates
Eat Herbs	Bad	-Eats the right herbs -Eats poisonous herbs -Overdoses on mushrooms
Work Out	Good	-Runs with friends -Swims at the lake -Goes to a gym -Dies after exhaustion
Laze	Bad	-Lazing on a Sunday afternoon -Lazes with friends at the beach -Lazes with friends at the mall -Arrested for loitering -Killed in jail by Machete

-Physical Gene:

It is the gene that responds to world events pertaining to physical condition. It has the following possible base pairs:

Table 13: Table of Physical Functions

Function	Quality	Effect(Best to Worst)
Run	Good	-Runs the marathon -Runs to the store -Runs to the car -Fatal heart attack
Walk	Bad	-Walks 500 miles -Walks to the sidewalk -Walks to the bathroom -Fatal embolism
Lift	Good	-Lifts 1000 pounds -Lifts a beer mug -Lifts self out of bed -Fatal stroke
Squat	Bad	-Squats 300 pounds -Squats to pick up a penny -Squats into a shantytown -Fatal squatting
Jump	Good	-Jumps on a bouncy castle -Jumps into a swimming pool -Jumps up and down -Jumps on a trampoline -Fatal broken neck
Climb	Bad	-Climbs a mountain -Climbs a hill -Climbs an escalator -Climbs to the roof -Fatal broken spine

-Morality Gene:

It is the gene that responds to world events pertaining to physical condition. It has the following possible base pairs:

Table 14: Table of Morality Functions

Function	Quality	Effect(Best to Worst)
Rescue Kitten from Tree	Good	-Rescues a kitten for a little girl -Rescues his own kitten -Rescues a stray kitten -Dies when rescuing a cougar
Put Kitten on Tree	Bad	-Puts your own kitten on a tree -Puts a stray kitten on a tree -Puts a little girl's kitten on a tree -Dies when putting a cougar on a tree
Save the Princess	Good	-Saves the princess from a dragon -Saves the princess from a strip club -Killed by a bouncer
Kidnap the Princess	Bad	-Kidnaps the princess from a strip club -Kidnaps the princess for the dragon -Killed by an adventurer with a moustache
Saint	Good	-Builds a charity -Donates to charity -Helps a homeless person -Gives change to a beggar
Psycho	Bad	-Harasses a beggar -Cons people off their money -Commits murder -Burns down a hospital

-First Generation

Once all the variables have been calculated, the game spawns 1000 creatures. Each will have their own DNA, First and Last names, and personality. They are the real actors in the game, and the numbers of the population will be extrapolated from them.

-Environmental Test

Once the first generation of 1000 creatures has been spawned, they will be tested against the variable. Testing in this case means that the game will dispatch a world event, and it will act upon every individual of those 1000 creatures. Passing the test means a mortality rate smaller than 30%. The test works as follow:

-World dispatches a random event of rating according to the planet's characteristics.

-Event activates the specific gene sequence of the creatures.

-The base pair is selected inside the gene sequence and the event rating is passed into it.

-Creatures will either be dead or alive after the event, depending on their personality and which base pair was activated.

-The total number of dead creatures is removed from the total number of creatures.

-If this number is more than 30% of the total, the creatures have failed. Otherwise, they have succeeded.

If the creatures fail the test, the remaining creatures will enter reproduction in order to refresh the gene pool as well as create more creatures, replacing those that were destroyed. During reproduction, the DNA of both parents in combined via genetic recombination, performing either crossover or gene conversion.

-Reproduction

How a creature chooses its mate is detailed further on the Artificial Intelligence section of the document. For now it is important to make a distinction on how the transmission of genes works.

For a new creature to be generated, it needs two parents. The parents will contribute their sequence to the process of reproduction, that is, their DNA array composed of 15 integers that are either 1 or 0. These DNAs are used in either a crossover or a conversion, which will result in the DNA for two creatures that will be the offspring of the parents. The chance for either crossover or reproduction is 50%, but one of them must always happen during reproduction.

-Crossover

Crossover guarantees that both offspring will be different from their parents. In this method a breaking point is needed, which will determine which area of the DNA will be swapped between the parents for their offspring DNA. This breaking point is an integer randomly generated, between 1 and 14, guaranteeing that at least one base pair will be switched for the offspring. This is better illustrated on the following picture that has a breaking point of 9.

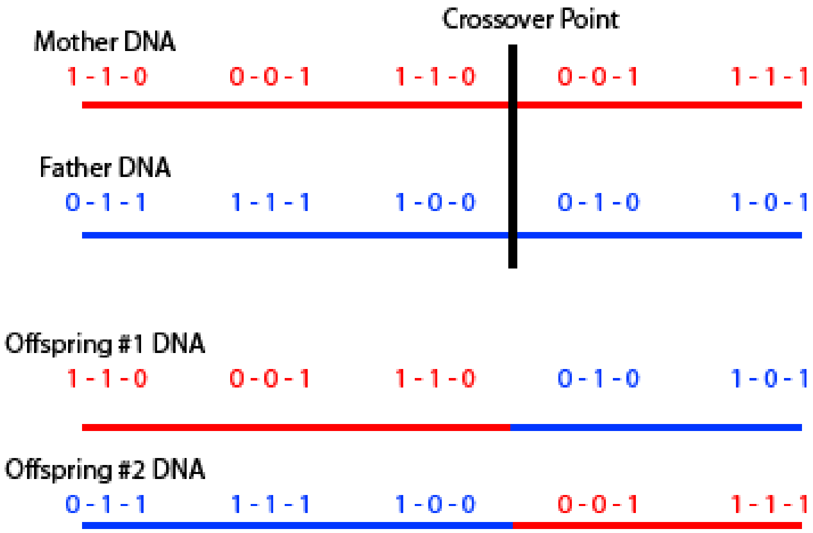


Figure 17: Crossover of the DNA.

-Gene Conversion

Gene conversion guarantees that one creature with a new DNA will be created, whereas the other one will be a copy of the parent. This method requires two breaking points, one to determine the start of the sequence to be copied and one to determine the end of the sequence. The first will be generated between 1 and 14, with the second one being generated between the first + 1 and 14.

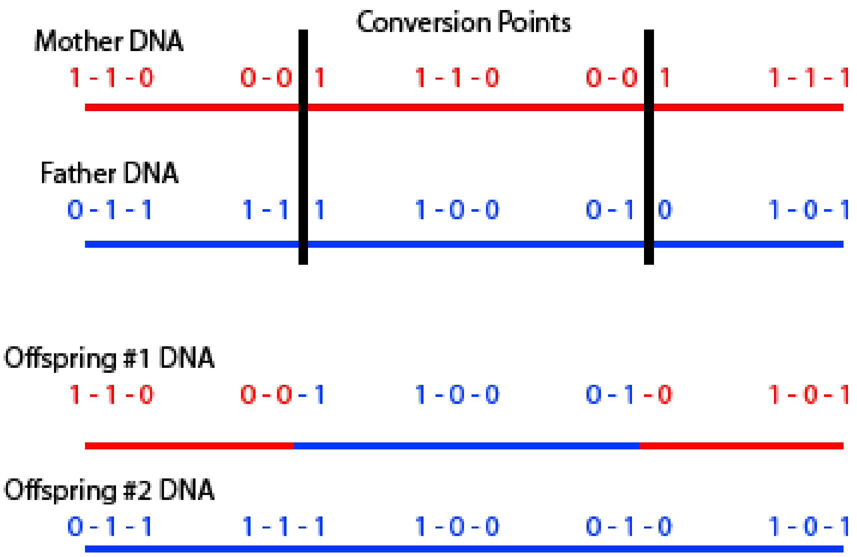


Figure 18: Conversion Points of the DNA.

-Mutation

Mutation can occur at a chance of 0.01%. This means that one in 10000 reproductions will result in a mutation in one of the genes. This mutation is a simple switching of a value from a 0 to 1 and vice versa. However, the mutation chance can increase based on choices the player takes as well as the path the planet takes towards the achievement of space flight. These include, but are not limited to, sending events that can result in radioactive waste surfacing around the planet, which can increase the chance from mutation randomly from the standard 0.01% to as high as 1%, depending on the rating of the radioactive event. The rating influences the interval for the random number to be generated for the chance. Table 15 illustrates the minimum and maximum values for the generation of the new mutation value according to the rating of the event.

Table 15: Event rating versus Minimum/Maximum Values Table

Event Rating	Minimum and Maximum values for the Mutation
1	0.01 and 0.05
2	0.01 and 0.2
3	0.05 and 0.5
4	0.1 and 0.75
5	0.5 and 1

Once the creatures pass the environmental test, they are locked in and will start evolving. At this point, the player can start setting goals for their artificial life.

NOTE: The gene that suffers a mutation is selected randomly by the computer. During a reproduction, the chance of 0.01% is calculated to see if the creature being created has suffered a mutation or not. If it hasn't, the mutation process ends. If it has, then a further random integer is generated between 1 and 15, which will determine which base pair will be mutated.

Goal System

The goal system is one of the ways the player can interact with his artificial life, and it sets a target for the artificial life to achieve. This element is divided into distinct steps:

Step 1: Finding the Goal

The first thing the player does is finding the goal he desires his creatures to achieve. This is accomplished by pressing the Status button on the game screen, which will open up the options that the player can perform as well as statistics, and pressing the Goal button, which will lead you to the screen with the possible goals that the player can select.

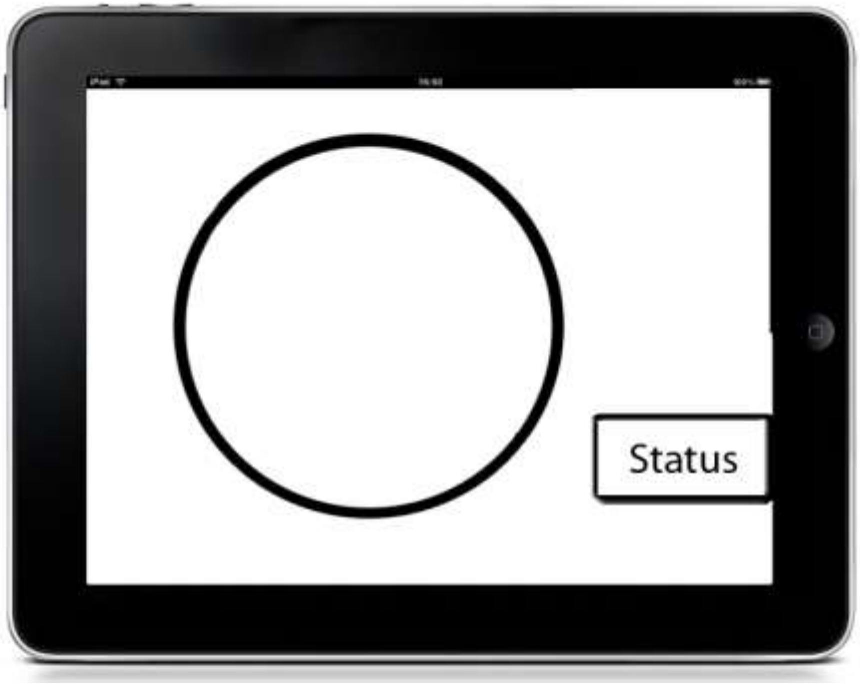


Figure 19: Scheme of the game screen with the status button.

Step 2: Setting the Goal

The second thing the player does is selecting the desired goal. On the goal selecting screen, the player has a view of the available technology as well as how long it is estimated for the AI to achieve that goal. The time is displayed in real world hours, whereas the AI will take a relative number of generations to achieve the goal.

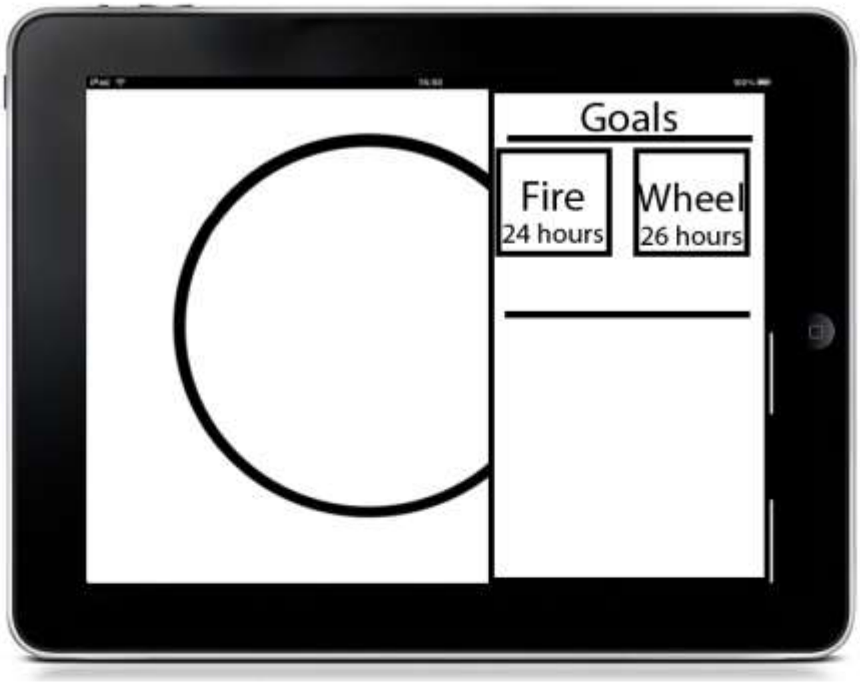


Figure 20: Scheme of the game screen with the available goals.

Step 3: Helping the Goal

The third and optional thing the player can do is helping the AI achieve their goal by sending world events towards the AI. These events trigger the event generator, creating a random event based on the type of event selected. These events also occur randomly in the world, but the player can spend their in game credits to send them himself. These events will have either positive, negative or both types of repercussions in the AI, and may help them achieve their goal faster or slow them down considerably. A list of goals is provided at the end of this section.

Step 4: Achieving the Goal

Once the AI realizes the goal, one of the AI creatures is selected to be named as the person who achieved the discovery.

Table 16: Table of Species Goal

Goal	Base Time	Effect
Fire	10000 real world hours divided by the amount of inhabitants on the planet	-Increases the hospitability attribute of the planet by 10%
Wheel	30000 real world hours divided by the amount of inhabitants on the planet	-Increases maximum population by 10%
Cloth	60000 real world hours divided by the amount of inhabitants on the planet	-Increases the hospitability attribute of the planet by 10%
Surgery	600000 real world hours divided by the amount of inhabitants on the planet	- Increases the rating of a health event by 1
Alcohol	300000 real world hours divided by the amount of inhabitants on the planet	- 1 point bonus to events that affect the drinking alone base pair
Combustion Engine	1000000 real world hours divided by the amount of inhabitants on the planet	-Unlocks the Smog event, a rare automated event called by the AI that can elevate the level of pollution in the world
Flight	2000000 real world hours divided by the amount of inhabitants on the planet	-Increases maximum population by 10%
Nuclear Power	7000000 real world hours divided by the amount of inhabitants on the planet	-Enables nuclear meltdown event, a rare automated event called by the world AI that can increase the rate of mutation
Computers	10000000 real world hours divided by the amount of inhabitants on the planet	- 1 point penalty to events that affect the social base pair
Space Flight	100000000 real world hours divided by the amount of inhabitants on the planet	-Enables space colonization goal
Space Colonization	1000000000 real world hours divided by the amount of inhabitants on the planet	-Player wins the main game

NOTE: An increase in hospitability affects the health event ratings that can be generated. The percentage increase works as a second calculation that is applied to the rating of the event. What this means is that every time a health event happens, with a 10% increase in hospitability there is a 10% chance of adding a 1 to the event rating after the rating has been calculated.

Table 17: Table of Events

Events	Effect
Forest Fire	-Dispatches a world personality event, to test the personality of the creatures, to see if they care enough to do something about it
Blizzard	-Dispatches a world morality event, to test the resolve of the creatures when being separated from society due to the blizzard
Heat Wave	-Dispatches a world health event, to test the health of the creatures
Storm	-Dispatches a world happiness event, to test people’s reactions to the storm
Stampede	-Dispatches a world physical event, to test the creatures’ physical ability when faced with that challenge
Viral Epidemic	-Dispatches a world health event, to test the creatures’ health when facing viruses
Overpopulation	-Dispatches a world morality event, to test the creatures’ morality level when faced with such possibilities
Nuclear Meltdown	-Dispatches a world health event -Dispatches a world morality event To check if they can survive the event and to check what they decide to do about it
Apocalypse	-Dispatches one event of every kind
Frenzy	-This is a unique event that the player can dispatch which will force the reproduction event upon the entire race

Event Generator

The event generator works to create a dynamic environment for the AI to live and survive in. The event generator works by randomly creating a name for the event, assigning a type to it and then dispatching it. The AI then interacts with the environment, and gives feedback as to how they reacted through the function that was activated in their genes by the environment.

-Event Name

The name of the event is generated by randomly choosing one word from the objects list and one word from the qualifier list.

Table 18: Table of Event Names

Objects	Qualifier
Potato, Alcohol, Idol, Banana	Revolution, Conflict, War
Tomato, Blizzard, Summer	Massacre, Peace, March
National, November, April	Prohibition, Moon, Prime
Hunger, Harvest, Liberty	Festival, Coup, Outbreak
Freedom, Rainy, Sunny	Uprising, Sale, Festivities
Orange, Candy, Jasmine	Strike, Rebellion, Jubilee
Rose, Red, October, Bulldozer	Celebration, Feast, Day
Nuclear, Tulip, Green	Truce, Armistice, Concord
International, Industrial, Ice Cream	Truce, Union, Promenade

-Event Hero

The event hero is randomly selected from the pool of creatures to be named as the person who influenced the celebrations. This only takes place if when the AI function is called to respond the event, the AI triggers the hidden random hero effect, which has a 0.01% chance of happening. The calculation takes place before the function moves on to identify the correct gene for the event.

For the 0.01% chance, each creature gets assigned a random number between 1 and 10000. After the calculations for the results of the event have taken place, the environmental intelligence generates another random number between 1 and 10000 and compares with the population. If there is a match, that creature gets selected as the event hero. If more than one creature has a match, they are collectively awarded the event hero as a team. If the creature that has a match was killed by the event, he is posthumously awarded the event hero. If no creatures had a match, no one is awarded the event hero.

Artificial Intelligence

Overview

This section describes the artificial intelligence found in *Planets!*, and contains topics describing the how they are used, and their implementation into the game. Artificial Intelligence found in *Planets!* can be broken down in two: Environmental Intelligence and Individual Intelligence.

Environmental Intelligence

This is the intelligence that governs the world, the events that are sent as well as make sure everything is happening according to the rules of the game. The AI checks for changes in the game every 10 seconds as long as the game is active, otherwise if the game is inactive or running on the background, the AI will check for changes in the game every hour. The AI function decision tree is described by the following image:

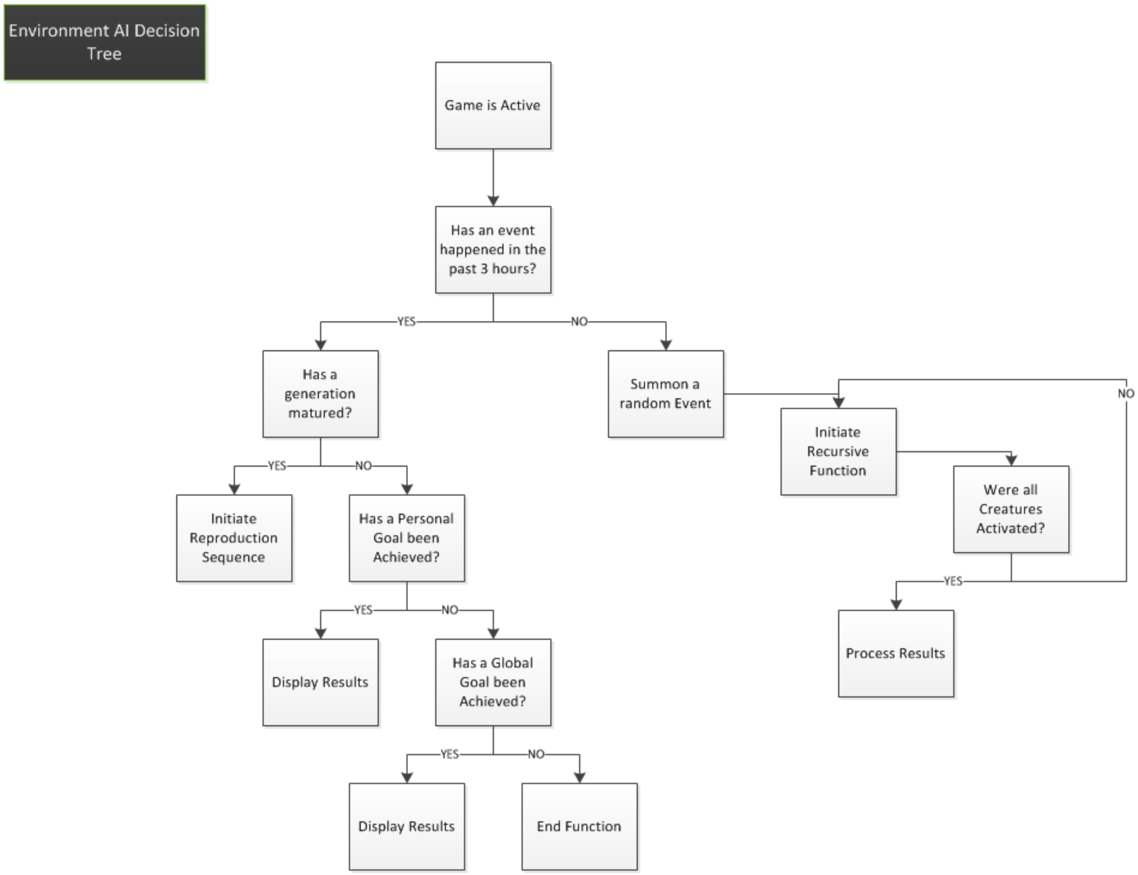

Environment AI Decision Tree
Game is Active
Has an event happened in the past 3 hours?
YES
NO
Has a generation matured?
Summon a random Event
Initiate Recursive Function
NO
YES
NO
Initiate Reproduction Sequence
Has a Personal Goal been Achieved?
Were all Creatures Activated?
YES
NO
YES
Display Results
Has a Global Goal been Achieved?
Process Results
YES
NO
Display Results
End Function

Figure 21: Flowchart of the Environmental AI Decision Tree.

-Event Check

The first step the AI takes is checking if it has been longer than 3 hours since the last AI generated event. 3 hours is the threshold between events, and is reset every time a new event happens. If there wasn't an event in the past 3 hours, a new random event is created.

The AI then calls its recursive function that will activate all the AI creatures, activating the related gene on them. If a meaningful result happens, such as an AI having the low chance of great work activated, then it stops and calculates the result of the great work, and displays it to the player. Then it resumes the recursive function until all the AI creatures have been affected, and reverts to waiting mode.

-Maturity Check

If an event has happened, it moves on to check the maturity of the current generation. All AI creatures possess a maturity level that is closely linked to their age and their genes. If the generation is mature enough, it will initiate a reproduction sequence.

In the reproduction sequence, each member of the species is assigned a number between 1 to 10 in order to group them by their fitness level, with 10 being the fittest and 1 being the weakest. They are then tested to see if they are eligible to reproduce. This is done by assigning them a random number between 1 and 100.

Table 19: Table of Reproduction Eligibility

Fitness Score	Chance	Number Needed
1	30%	70 to 100
2	40%	60 to 100
3	50%	50 to 100
4	60%	40 to 100
5	65%	35 to 100
6	70%	30 to 100
7	80%	20 to 100
8	85%	15 to 100
9	90%	10 to 100
10	95%	5 to 100

If the creature succeeds in the test, his fitness score is added to the pool and he is given an amount of numbers relative to his fitness that correspond to his place in line.

For example:

5 organisms: A, B, C, D and E

A fitness level = 1, pool number = 1

B fitness level = 2, pool numbers = 2 and 3

C fitness level = 2, pool numbers = 4 and 5

D fitness level = 4, pool numbers = 6, 7, 8 and 9

E fitness level = 10, pool numbers = 10, 11, 12, 13, 14, 15, 16, 17, 18, 19

Total pool number: 19

Then the AI goes through every creature from the fittest and therefore highest number, down until there are no AI left to reproduce. It then generates a random number between 1 and the total pool number, and mates the two creatures. If a creature generates a random number that is equal to his pool numbers, nothing happens.

-Personal Goal Check

If there was no reproduction, the AI moves on to check the personal goals of every creature. If a creature has achieved a goal, the AI then calculates the results of that goal, processes the results and shows them to the player.

-Global Goal Check

If no personal goal has been achieved, the AI moves on to check the global goal set by the player. If a global goal has been achieved, the AI calculates the results of that goal, processes the results and shows them to the player.

Individual Intelligence

This is the intelligence that governs the behavior of every individual creature and makes sure everything is happening according to the rules of the game. It deals with the genetic workings of the creatures, their goals and behavior. It works according to the following flowchart:

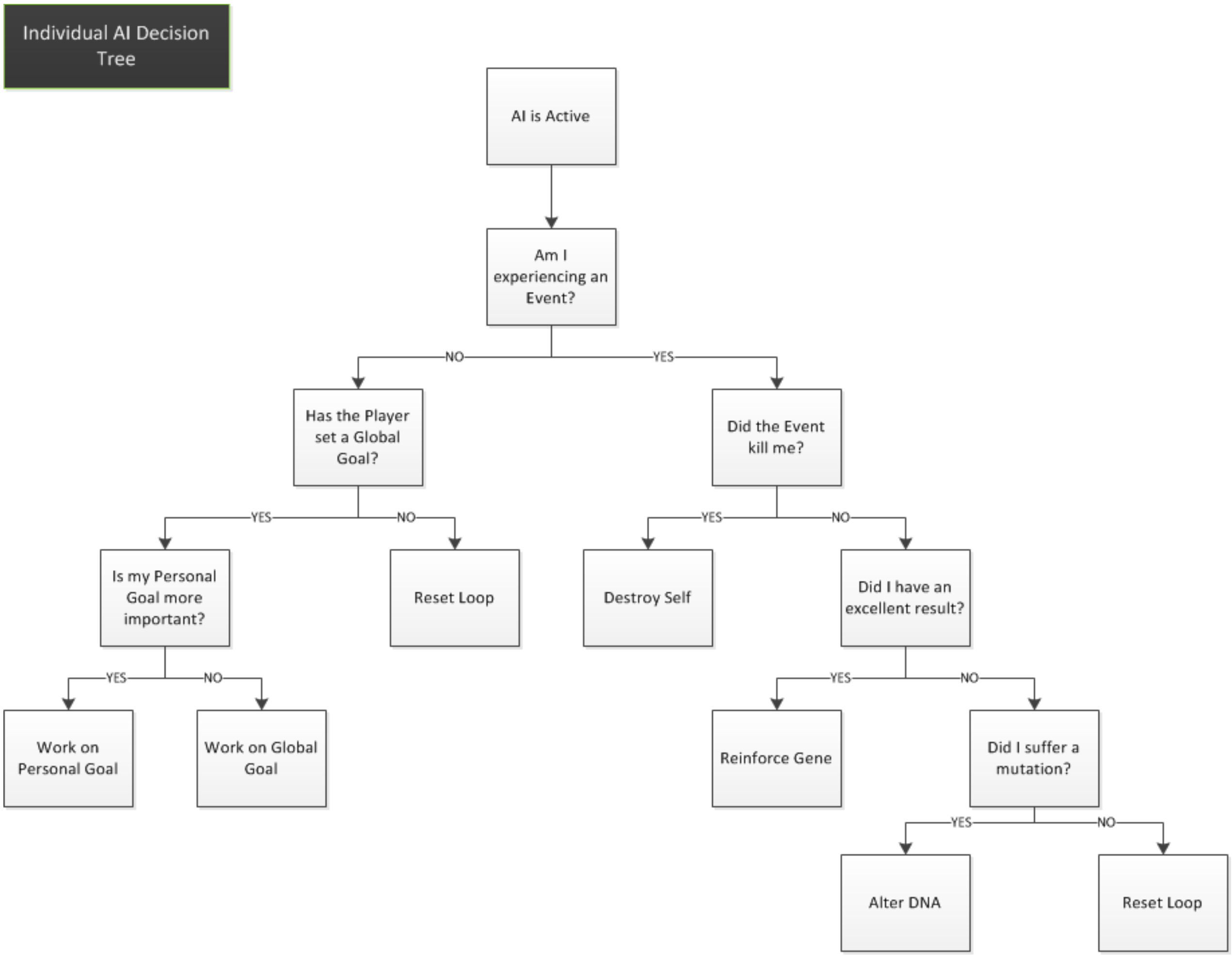


Figure 22: Flowchart of the Individual AI Decision Tree.

-Event Check

If the AI was activated by an Event, this will return true, and it will continue down the right branch of the tree.

-Kill Check

The AI then checks to see if the Event that happened had a fatal outcome for itself. If it returns false, it continues on. Otherwise, the AI is deleted from the game. This is determined by a Boolean that is assigned to every creature upon creation, determining that they are not dead. It will return false as long as the creature has not been killed by the event, otherwise it returns true.

-Excellence Check

The AI then checks to see how well it did on the event. If the AI activated the 0.01% chance for the remarkable result, the AI then memorizes this result and expands its threshold for that gene by adding another number to the random number generator used to check which base pair gets activated for the event, and assigning that number to the base pair that was reinforced. It also reduces the chance of that base pair being mutated by adding a number to the other base pairs when doing a mutation check.

This has the effect of raising the chance of that base pair being activated from 33% to 50% and of reducing the chance of mutation from 33% to 20%.

Base pairs can be reinforced an infinite number of times, reinforcing the good behavior and guaranteeing a stronger creature in the gene pool. The reinforcement is also passed down to the offspring.

-Mutation Check

If there was no excellent result, the AI then checks the mutation chance to see if anything has happened to its DNA. If a mutation has happened, the AI swaps the functions and resets any reinforcement that that base pair might have had before the mutation.

-Global Goal Check

If there wasn’t an event, the AI checks to see if the player has put a global goal in place. If there is no goal, it defaults back to the start of the function. If there is a goal, it proceeds down the tree.

-Personal Goal Check

The AI checks to see which goal has more importance: Personal or Global. It then applies its work to the respective goal. The AI is assigned a personal goal upon being created, utilizing the Random Personal Goal Generator, or R.P.G.G. for short. This is a goal that the AI will pursuit through its entire lifetime, coming in conflict with the global goal.

When an AI is created, it has an altruism attribute that is randomly generated between 1 and 5. This attributes dictates how often the AI will work towards its personal goal and how often it will work towards the global goal.

The check uses a D20 system, where one dice is rolled for the global goal difficulty, and one dice is rolled for the AI altruism check, with his altruism attribute being added to this result.

If the results match, the AI devotes its work hour for the Global Goal. Otherwise it devotes its time to the personal goal.

Every real life hour equals 3000 hours in game. Therefore, an AI creature can contribute up to 3000 hours to the goal every real life hour.

-Random Personal Goal Generator

The R.P.G.G works similarly to the random event generator, in that it takes two words and creates the title of the goal for a specific AI. This serves merely as an aesthetic qualifier and to give the AI a better appearance of personality. It takes one object and one qualifier to create the title of the goal.

Table 20: Table of Goal Generator Objects

Objects	Qualifier
Rock n Roll, Blues, Surfing, Banana	Engineer, Doctor, Lawyer
Classic, Automotive, Pizza	President, Dictator, Poet
Pasta, French, Business	Musician, Leader, Mime
Tropical Country, Minor, Liberty	Scientist, Research, Vendor
Mega, Senior, Candy	Designer, Salesman, Cook
House, Trance, Product	Actor, Student, Artist
Project, Method, Drunk, Lifelong	Party Animal, Bum, Vagrant
Paint, Shakespearian, Keynesian	Officer, General, Manager
Mafia, Cigar, Chinese Food	Painter, Celebrity, God

Game Modes

Overview

This section of the document details the free-play game mode featured in *Planets!*.

Free-Play Game

Planets! features only a single-player free-play game mode. The player takes on the role of the Designer and is tasked with building a planet from the ground up and guiding the life that appears to space. The player has the freedom to select the events that will impact on the way their creatures advance through the ages. It is a sandbox game, where the player is given a set of individual artificial intelligence creatures and he can toy with their environment in order to impact changes on the creatures themselves.

Story

The player play as the Designer, an entity that has the power to create planets and life. The story is liquid in that it is not completely locked. It has a main frame that is the evolution of the life and their pursuit of spaceflight. However, the player has control over how the life progresses in order to achieve that goal, making every play-through different from each other.

Story Breakdown

The player can track his story progression via cut-scenes that show a major breakthrough, and therefore a jump in the technological age, as well as the different graphics for the buildings in the planets that will get modernized as the player progresses.

The story begins with the creation of the planet. Afterwards, a small cut-scene will illustrate the generation of life on the planet, letting the player know that life has been created. The player will then progress through the ages, and this progress will be communicated via cut-scenes and different graphics.

Free-Play Progression

This section of the document details how the player progresses through the story on free-play mode. The progression in *Planets!* happens in three fronts; cosmetic, events and population.

Cosmetic Progression

The cosmetic progression is visible to the player in that it shows on the graphics used to display the current state of the civilization. The function of the cosmetic progression is to visually show to the player that his creatures are evolving and advancing through the ages. These vary according to the age the civilization encounters itself in, and varies according to the following items:

-Ancient Age

The ancient age is the first age the player will enter. It is the age of cavemen, where the artificial life is evolved enough to start hunting and gathering, but not enough that it can build advanced objects and buildings.

The ancient age planet is depicted as somewhat barren, as there is an absence of buildings. The players will mostly see forests and mountains on their planet, with the occasional sighting of his creature. Once he researches fire, fire pits will start to appear in the planet to indicate that the creatures are active. The Ancient Age ends when the player researches the wheel.

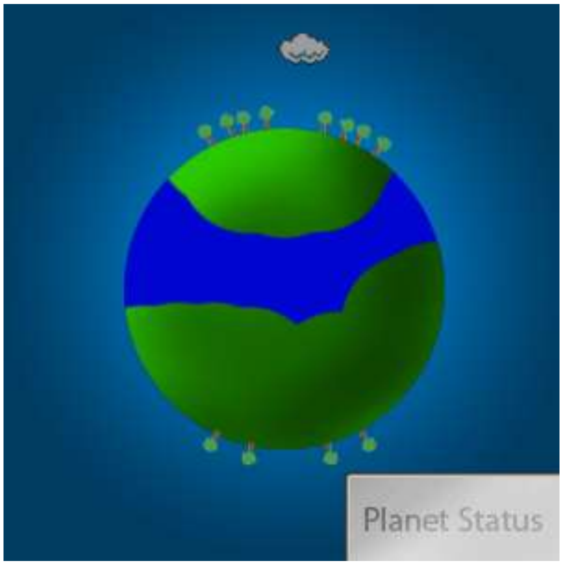


Figure 23: Ancient Age Planet.

-Classic Age

The classic age is the second age the player will enter. It is the age of metals, where the artificial life is evolved enough to start building homes.

The classic age is depicted by buildings that look like traditional Greek and Roman architecture. The players will start to see buildings coming up and down as the years pass as well as the construction of roads. The Classic Age ends when the player researches surgery.

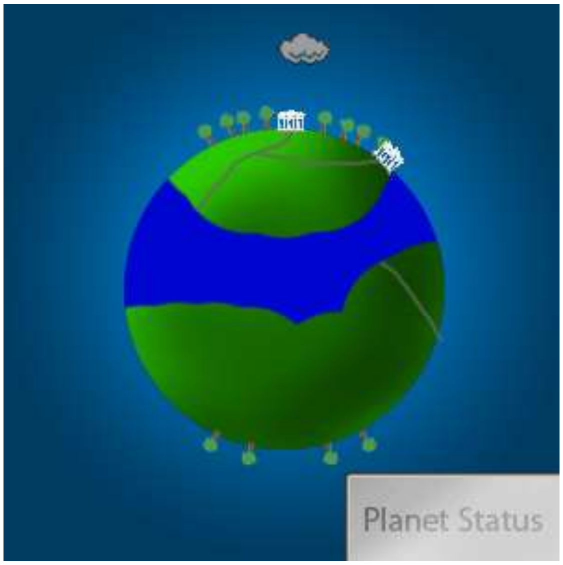


Figure 24: Classic Age Planet.

-Medieval Age

The medieval age is the third age the player will enter. It is the age of castles, where the artificial life is evolved enough to start building massive castles.

The medieval age is depicted by buildings that look like traditional Medieval architecture. The players will start to see castles coming up and down as the years pass. The Medieval Age ends when the player unlocks the combustion technology.

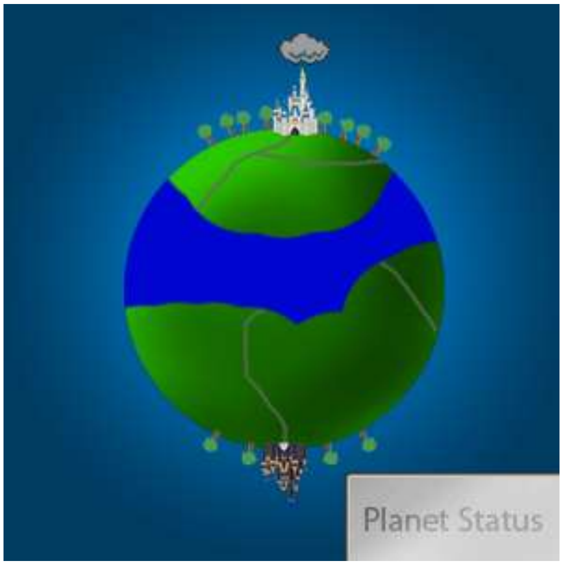


Figure 25: Medieval Age Planet.

-Industrial Age

The industrial age is the fourth age the player will enter. It is the age of industry, where the artificial life is advanced enough to start building industries, allowing the population to boom.

The industrial age is depicted by buildings that look like traditional Victorian architecture. The players will start to see cities coming up and down as the years pass, as well as the apparition of the first railroads across the planet. The Industrial Age ends when the player completes the flight technology.

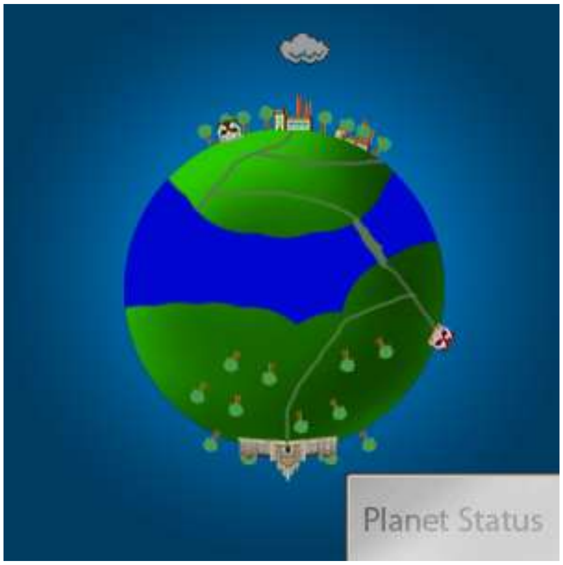


Figure 26: Industrial Age Planet.

-Modern Age

The modern age is the fifth age the player will enter. It is the age of computers, where the artificial life is advanced enough to start building taller structures as well as airplanes.

The modern age is depicted by buildings that look like traditional Modern architecture. The players will start to see tall buildings coming up and down as the years pass, as well as the apparition of the first airplanes across the planet. The Modern Age ends when the player completes the computers technology.

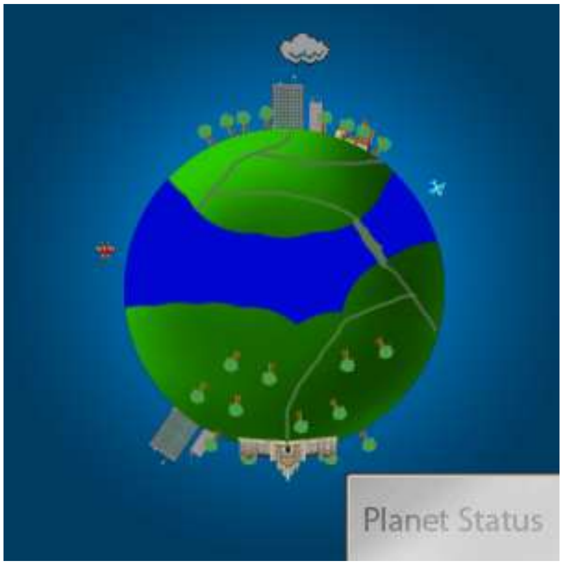


Figure 27: Modern Age Planet.

-Space Age

The space age is the final age the player will enter. It is the age of spaceflight, where the artificial life is advanced enough to start building spaceships and start investigating the cosmos.

The space age is depicted by buildings that look futuristic. The players will start to see taller buildings coming up and down as the years pass, as well as the apparition of the first spaceships attempting to reach the cosmos. It is the final age the player can achieve, and successfully landing on a planet triggers the final victory condition.

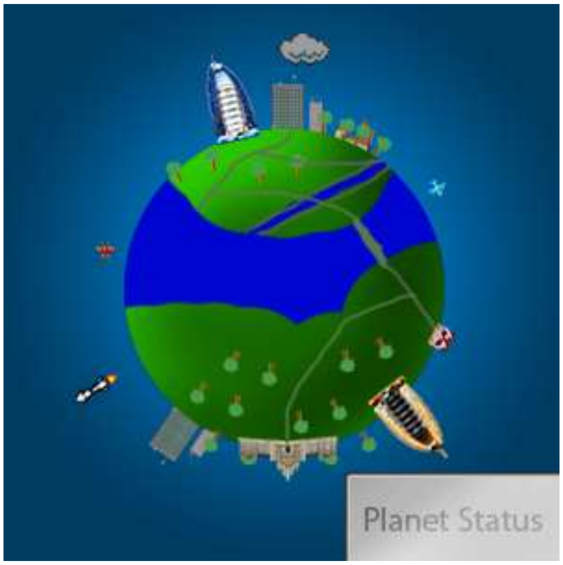


Figure 28: Space Age Planet.

Event Progression

As detailed in the event sections, completing certain goals will enable new events that the player can use, as well as possibly modify the rating for previous events that the player can dispatch.

Population Progression

As the player advances through the ages, the population of the planet becomes larger. Events that the player can send can affect the size of the population, which can lead to overpopulation. The more advanced a race is, the more inhabitants they can support without problems, and the quicker they can research the next step. The player can also send a Frenzy event to force the reproduction of the species. The population also progress via natural selection, with the susceptible to bad events being killed off in the process while good characteristics are reinforced through the population, leaving the player in the end with a strengthened population in some direction (both good and bad).

Free-Play Time Projection

The following chart is a projection of the time needed for the player to go through the different civilization ages and reach space travel. Due to the free-play nature of the game, players make take longer to achieve the goals due to just playing around with the sandbox.

Table 21: Table of Free-Play Time Projection

Stage	Technology to Complete	Average Time for Completion (hours)
Ancient Age	Wheel	10
Classic Age	Surgery	15
Medieval Age	Combustion	20
Industrial	Flight	30
Modern Age	Computers	40
Space Age	Space Flight	50

Free-Play Beat Chart

The beat chart for the game shows the difficulty of sustaining the population at a planet before attaining the technology that simplifies that process, if only temporarily. The biggest challenge the player will face is making sure the population is at an adequate level for the planet, which is explained in table 22.

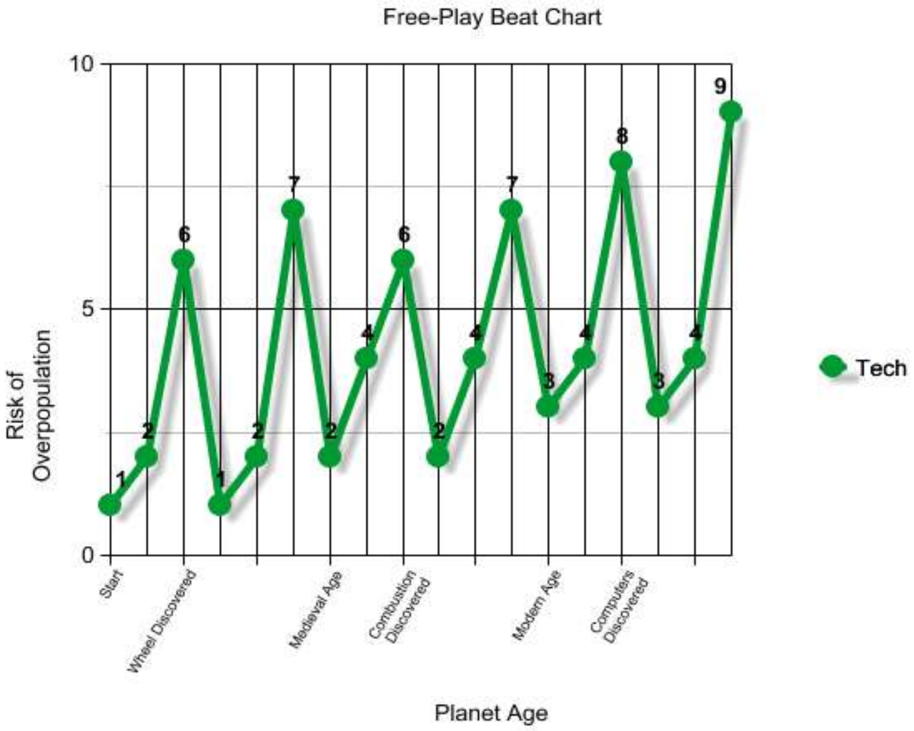


Figure 29: Free-Play Beat Chart

Table 22: Base values for Maximum Population according to Age

Age	Maximum Population
Ancient	2000
Classic	40000
Medieval	100000
Industrial	200000
Modern	500000
Space	1000000

NOTE: It is important to notice that the values displayed in Table 22 are only base values, and do not take into account the size of the planet. The planet size modifiers are explained earlier in the document in the Planet Creation settings. These modifiers are applied to these base values.

Risk of overpopulation varies from 1 to 10 and is directly related to the ratio between current population and maximum population. It increases progressively with time, reaching its apex before the player researches the technology that will lead them to the next age. The chart represents the average risk the player will face just before getting to the next age. A risk rating of 1 means that the player should be at about 30% of the population, a risk rating of 9 means that the player should be at about 98% population, with the rest of the ratings being assigned a percentage according to the scale.

The way that this works is that the latter stage technologies take a considerable amount of time to be achieved, but that time is split among the population of the planet. The beat chart provides the optimum levels for the optimum completion of the game. Players will deviate from this in that they will take longer to complete the game or go for shorter but riskier wins.

In effect it means, according to the beat chart, that on average the player will have a population of 980000, a rating of 9, by the time he reaches the final technology.

Victory Conditions

This section of the document will detail the victory condition featured in *Planets!.*

Space Travel

Planets! features a single victory condition, and that is successfully colonizing a planet after reaching the space age. This condition can be accessed when the player finishes researching space travel. The player will then be able to set the species goal as the colonization of a planet. The player will be warned when the species have decided to send a ship to the nearest planet, and will be given the choice of allowing them to do so or to get them to keep researching if the deems his species' ship not space worthy.

The player will then follow the journey of the ship and their landing. The condition can fail if the ship wasn't properly built, if the planet was too hostile or if another accident happened during the way. Otherwise, the player will be rewarded with a cut-scene showing the species safely colonizing the new planet, and the player will be given the option to keep playing or to start a new planet.

Game World

Overview

This section of the document will detail the game world featured in *Planets!.*

Planet

Overview

This topic provides a description the key components of the world including scale, time of day, travel, weather regarding the main world, the planet itself. Planets does not feature different worlds or levels, rather only a single game world that the player can play around with.

Scale

When it comes to the interface, the size of the plane is irrelevant as it will always be positioned and displayed in the same size relative to the size of the screen, as shown on figure 30.

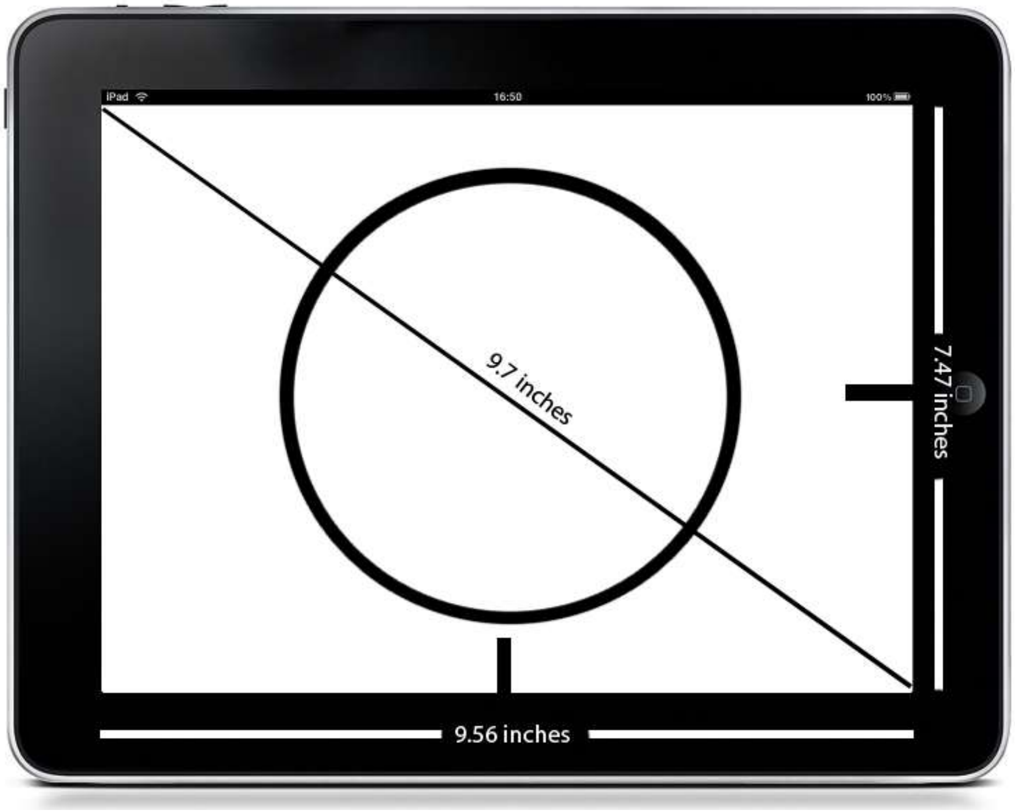


Figure 30: The absolute position and size of the planet

When it comes to the graphics being displayed on screen, the difference will be on the scale of the game objects being displayed to the user. Buildings, vehicles and all other graphics will be scaled up or down by multiplying the size for the scale value according to the size of the planet, as shown in table 23.

Table 23: Planet Size versus Scale Table

Planet Size	Scale
Dwarf	2.0
Small	1.5
Average	1.0
Large	0.6
Titanic	0.2

Weather

The game world will display the current weather events on the game by means of animated graphics around the planet as exemplified by figure 31. These graphics are always the same size relative to the size of the planet, meaning you never get cartoonish big clouds over a dwarf planet.

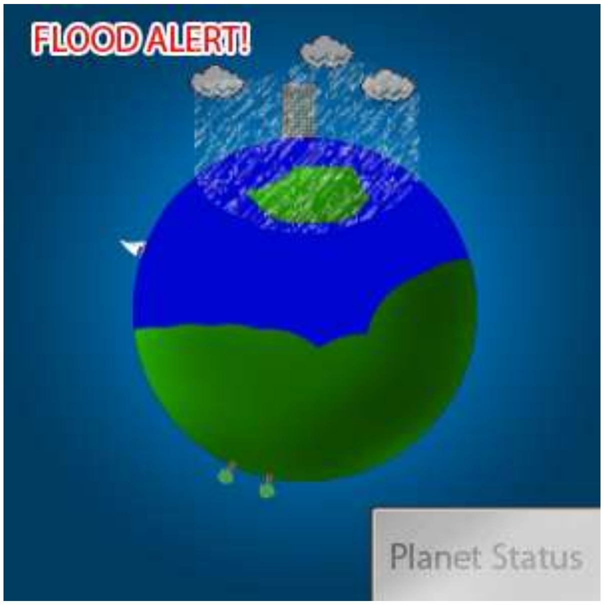


Figure 31: Example of a heavy rain weather affecting the game world

Time of Day

There is no visual progression of the time of day in the game world, due to the perspective of viewing the world from outside its atmosphere.

Travel

The only mode of travel featured in *Planets!* is navigating the system map to see what is around the player's planet. The system map is featured in detail in the System First Person Camera section earlier in this document.

Appendix A – Art Guidelines

Overview

This section includes an overview of the art philosophy, as well as some sample images.

Art Design

The art for *Planets!* should consist of relatively simple graphics with a simple interface that's easy to navigate and does not take the focus away from the gameplay. Graphics are based on the pixel art style. The graphics should be colorful enough so as to make them appealing. The use of bright colors is recommended. An example is shown on figure 32.

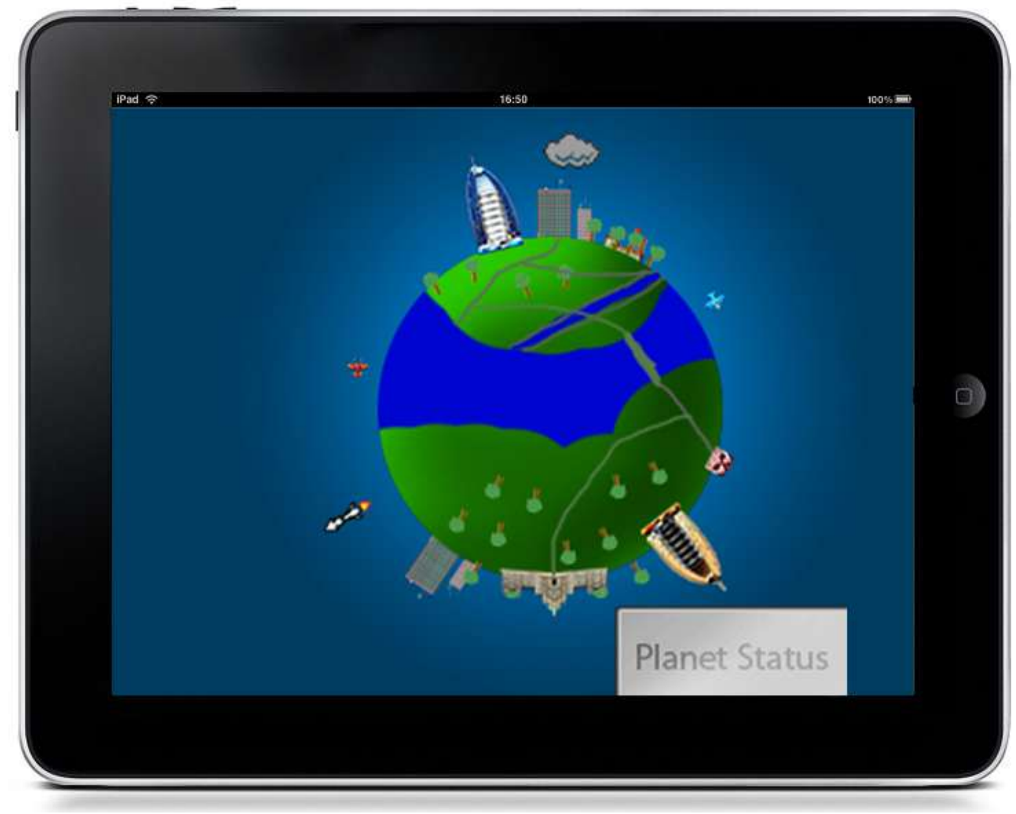


Figure 32: Art Design Example

Colorful Graphics

Game screens should utilize bright colors with simple gradients to give the game a stylized look.

Multiple Palettes

There should be multiple color palettes for every building in game, in order to add more variety while saving on space. The base graphics will be designed using a grayscale palette, in order for a color palette to be applied later.

Appendix B – Audio Guidelines

Overview

This section includes the audio philosophy and guidelines for *Planets!*.

Sound Design

The audio assets will consist of soothing background music and sound effects for button presses, alarms and events.

Background Music

Planets! will feature a music bank of 10 looping instrumental tunes whose tempo range between 66 and 76 bpm. The music will not speed up or slow down at any point during gameplay, therefore they don't need to be scalable. They shall not feature lyrics. The tunes will also need to be at least 5 minutes long, and should be able to transition between them seamlessly, either through the use of a fade or ending and beginning roughly with the same notes.

Button Press

Button presses sound effects should be space themed, such as the beeps of a machine. Different class of buttons will play different sounds. Buttons will always play the same sound.

Alarm

Alarm sound effects consist of looping siren sounds that will be played when a catastrophe is about to happen. They should have a higher bpm, circa 200 bpm, to give the player a sense of urgency. They fade out once the player has tended to the catastrophe instead of just stopping.

Events

The sound of a bell ring will play once an event happens, letting the player know that something has taken place and requires his attention. This sound will only play if the player is actively playing the game, and only plays once per event.

Appendix C – Interface Wireframes

Overview

This section includes wireframes for the main menu and menu screens for *Planets!.*

In-Game Screens

In-Game HUD

Planets! HUD features the main gameplay object, the planet itself, in the central area of the screen. The only object the player can interact with is the “Planet Status” button that will bring up the menu with all the information on the planet as well as possible courses of action.

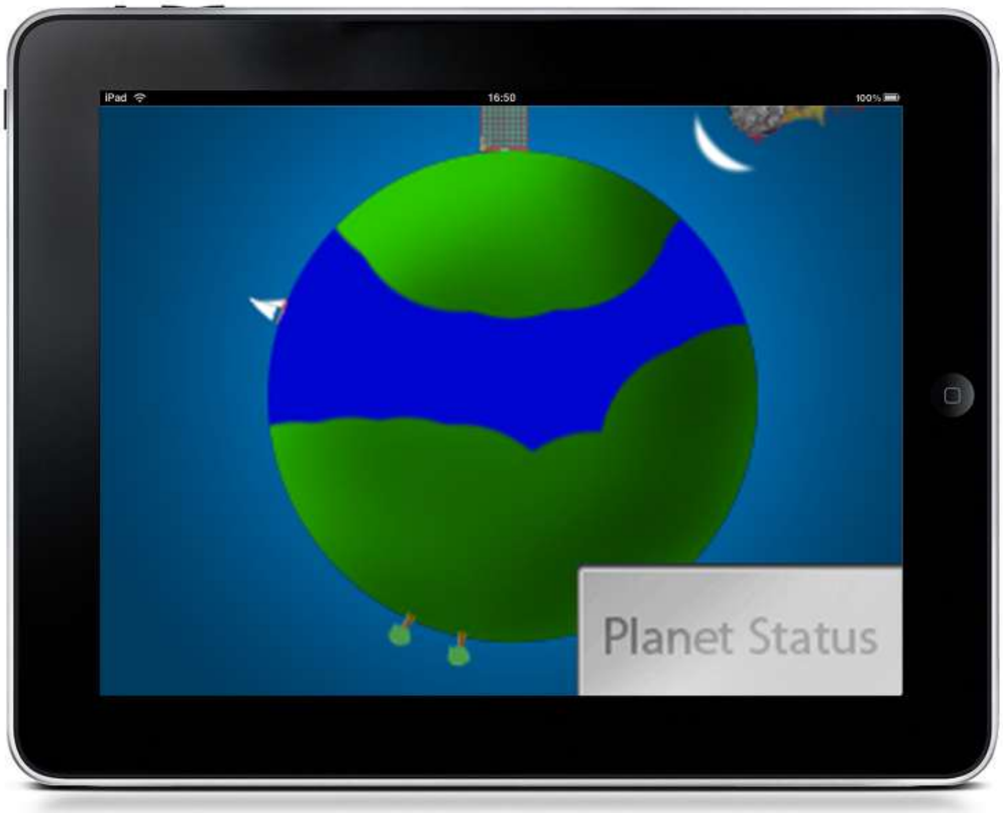


Figure 33: In-Game HUD Mockup.

The player also receives information about the status of the planet via graphics that will appear on the surface and around the planet. These will be portraying climate events, disasters, new buildings etc. These serve only to give feedback to the player if an event is happening that merits the player's attention. Their position on the screen is random.

In-Game Wireframes

In-Game HUD Wireframe

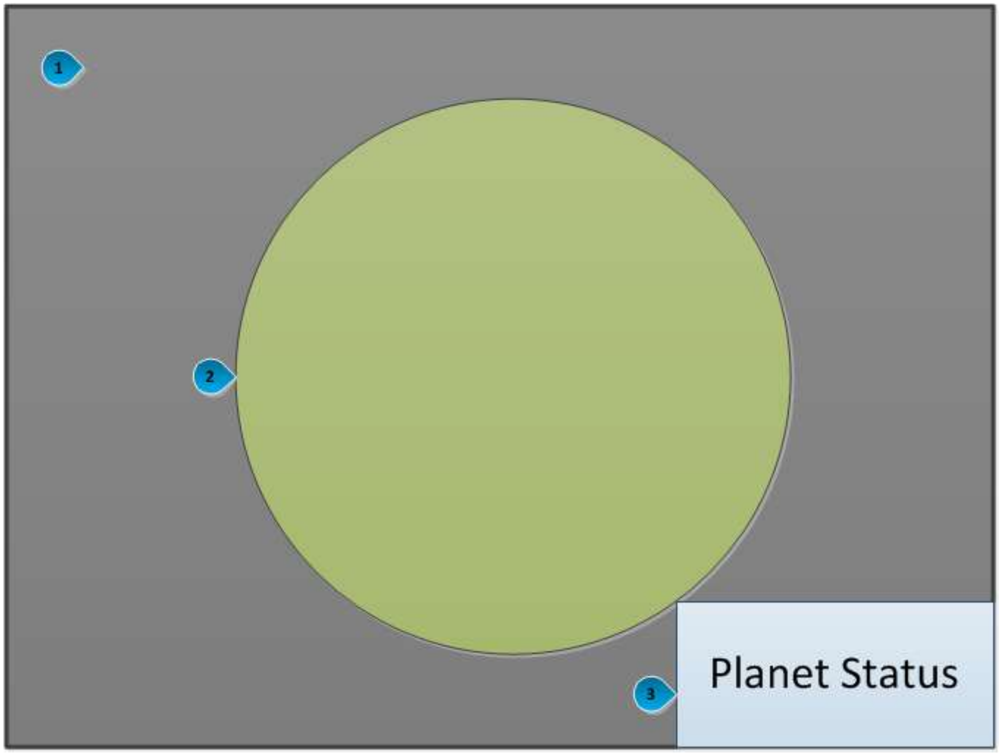


Figure 34: Wireframe of the HUD

Table 24: In-Game HUD Table

Item	Name	Details
1	Environment	Shows the universe behind the planet
2	Planet	The player's planet, with its graphics
3	Planet Status	The button that opens up the planet status menu, allowing the player more actions

Planet Status Wireframe

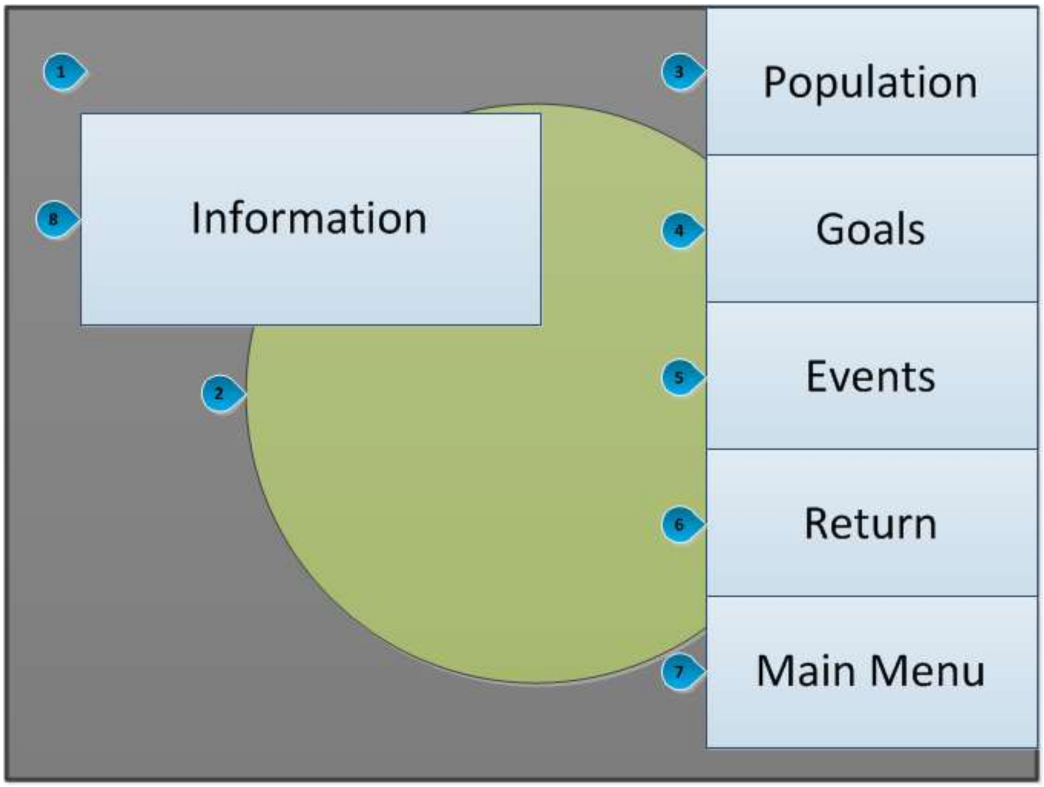


Figure 35: Wireframe of the Planet Status Menu

Table 25: Planet Status Table

Item	Name	Details
1	Environment	Shows the universe behind the planet
2	Planet	The player's planet with its graphics
3	Population	The population button fills the information box with data on the population
4	Goals	The goals button fills the information box with data on the goals
5	Events	The events button fills the information box with data on events
6	Return	The return button navigates back to the previous screen
7	Main Menu	The main menu button navigates out of the game and into the main menu
8	Information	Displays the information on the selected button, such as population size, events to send, goals to assign.

As detailed on the Control section, navigating through the menus is done using the touch input on the screen of the user's device.

Main Menu Wireframe

The main menu begins with a loading screen displaying the Vancouver Film School screen, followed by a loading screen featuring the game title as well as copyright notices. It will then check to see if the player has a previous game.

If he does, then it will move to the loading screen, get the last checkpoint and start the game.

If he doesn't, it will progress to the main menu screen, where the player will have the options to start a new game, quit the game or visit the credits screen.

Starting a new game moves the player to the loading screen and then into the game.

Quitting the game will prompt the player to confirm that he wishes to quit or go back to the main menu.

Credits will take the player to the credits screen, displaying the name of the people who have worked in the game. The player can exit this screen by pressing the return button.

NOTE: The main menu is only shown automatically when the player launches the game for the first time or doesn't have a previous save game. If the player wishes to activate the main menu again, he must do so through the in game menu.

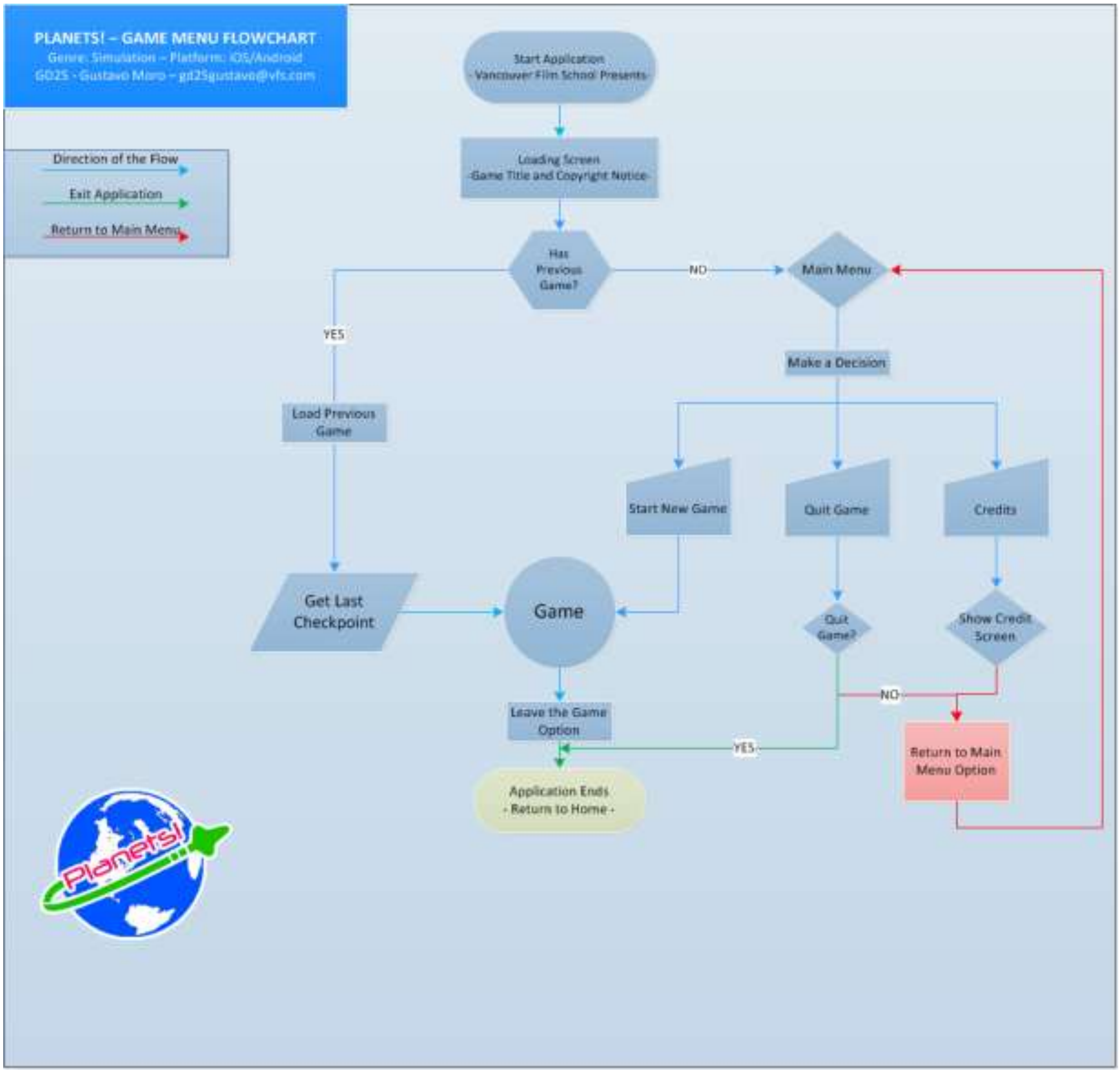


Figure 36: Flowchart for the Main Menu Screen

Menus Screens

Main Menu

The main menu allows you to play the game or navigate to the credits screen that shows the people that have worked in the game, as well as navigate back to the home screen of the user's device.

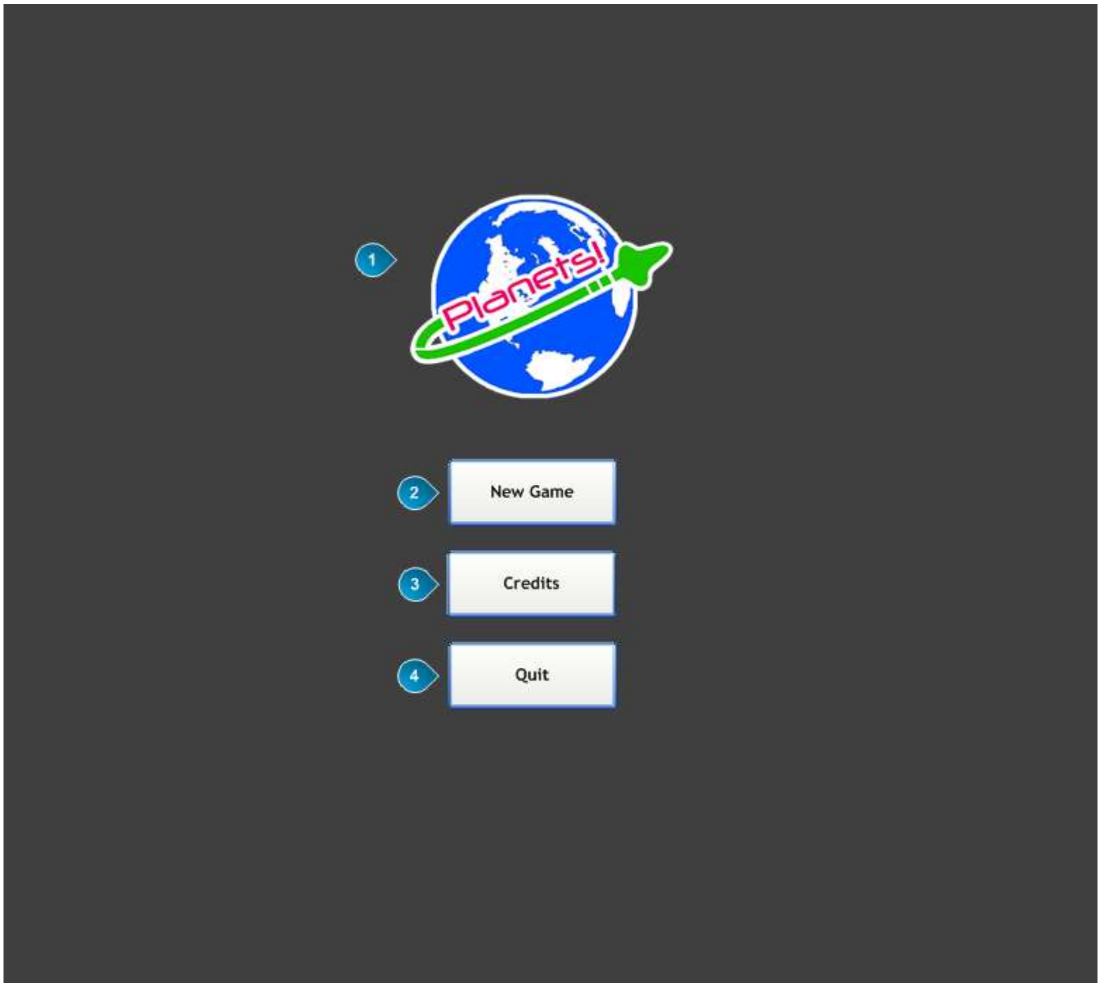


Figure 37: Main Menu Wireframe

Table 26: Main Menu Table

Item	Name	Details
1	Logo and Background	The background of the Main Menu will be an image that represents the universe full of stars. The logo will be inserted over the stars, with a glow from behind it as if being illuminated by a star.
2	New Game	Pressing the New Game button will start a new game
3	Credits	Pressing the Credits button will take the player to the credits screen
4	Quit	Pressing the quit button will prompt the player on his decision to quit, and if so take him to the home of the device

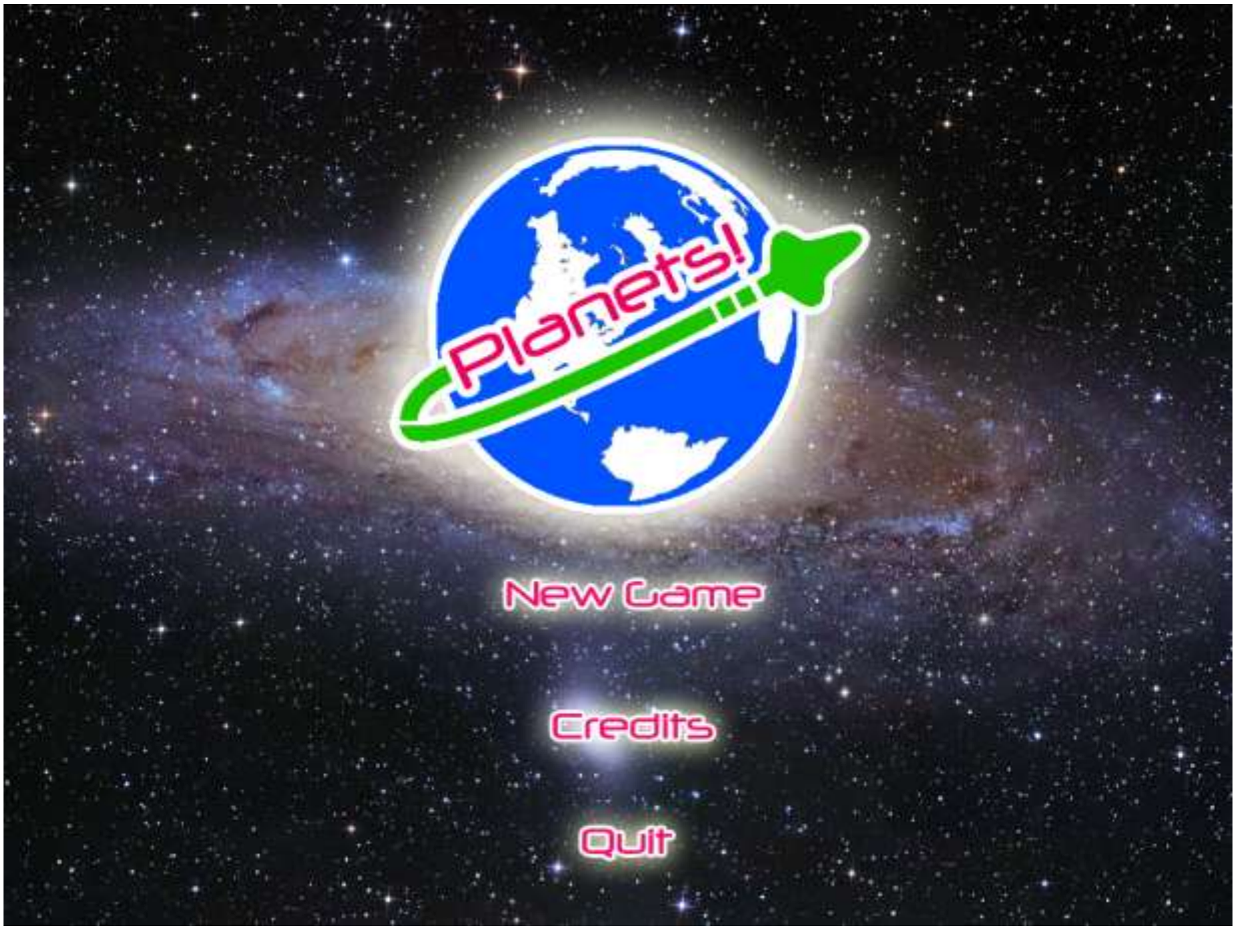
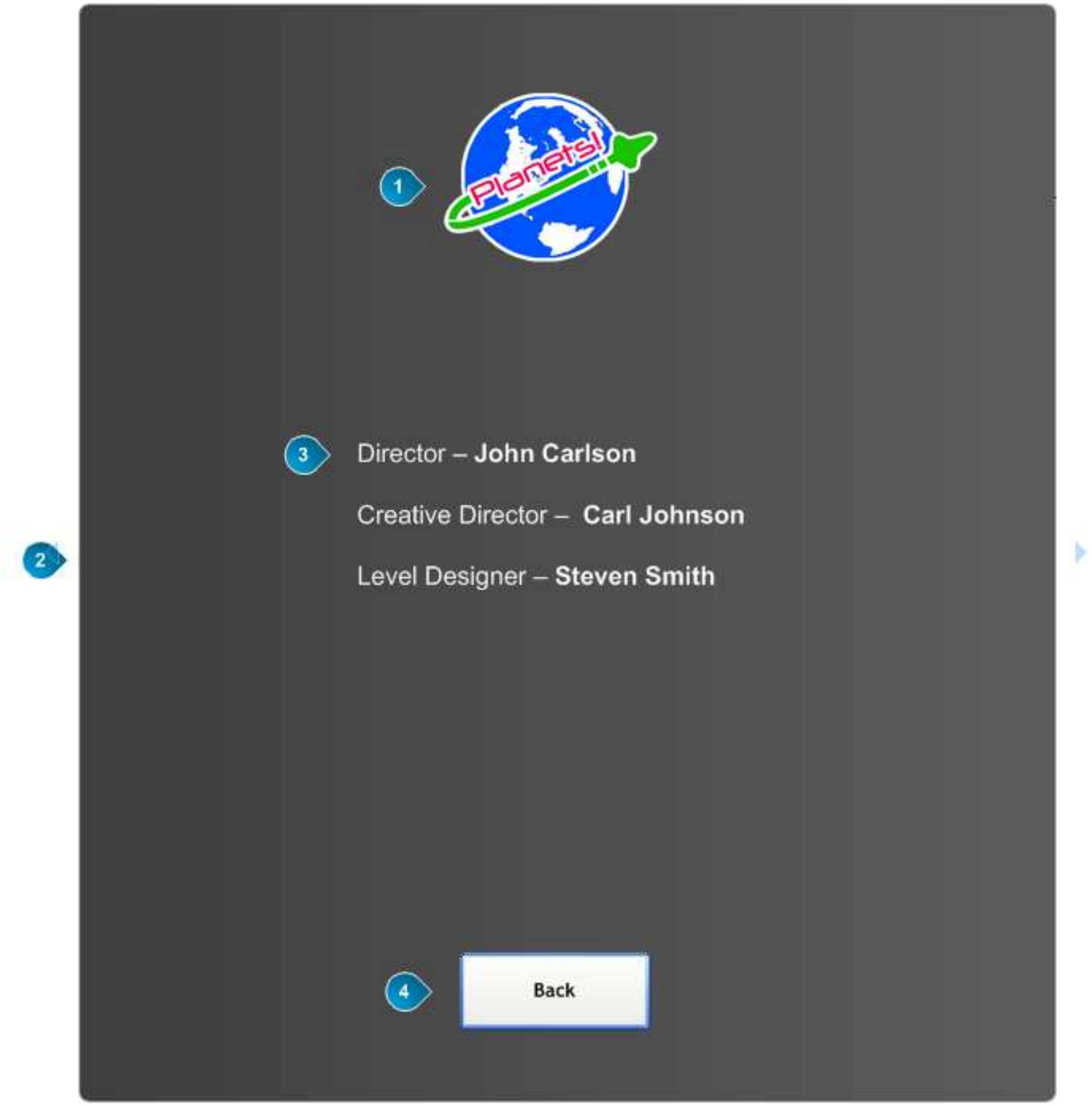


Figure 38: Mockup of the Main Menu Screen

Credits

The credits screen allows the player to see who has contributed to the game and enables the player to navigate back to the main menu when he so desires.



Director – **John Carlson**

Creative Director – **Carl Johnson**

Level Designer – **Steven Smith**

Figure 39: Credits Screen Wireframe

Table 27: Credits Table

Item	Name	Details
1	Logo	This component displays the Logo of the game.
2	Background	An animated image of a starry universe will appear as the background. The animation consists of stars shining, expanding and contracting their size.
3	Credits Text	This section displays the text, the credits of the game. As the stars shine, a name will fade into appearance alongside a planet designed by the person being credited. It will then fade out, leaving room for another name and planet to appear.
4	Back	This button allows the user to return to the Main Menu Screen.

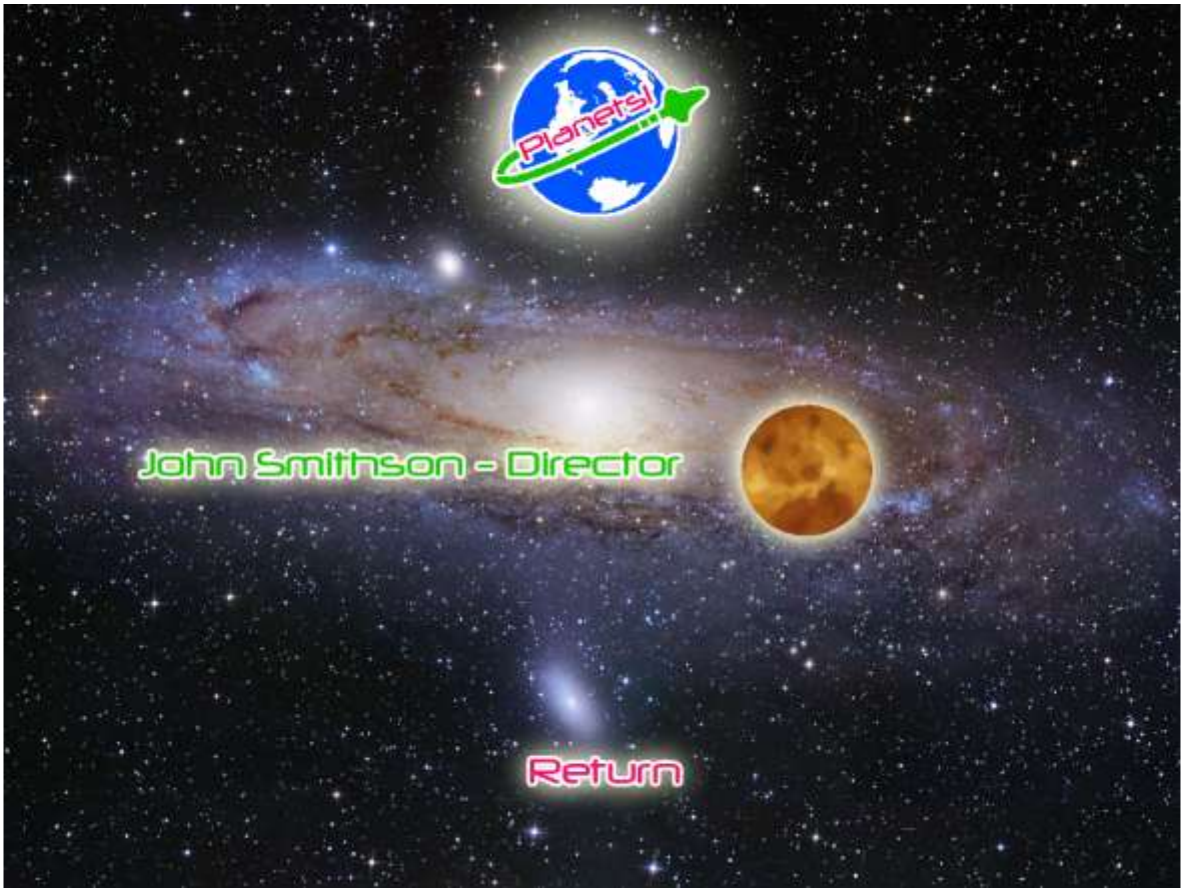


Figure 40: Credits Screen Mockup

Loading

The loading screen is displayed while the game is loading. It's not interactive, and displays a progress bar to show the player that the game is in fact loading. Loading can be cancelled by pressing the home button or equivalent on the device.

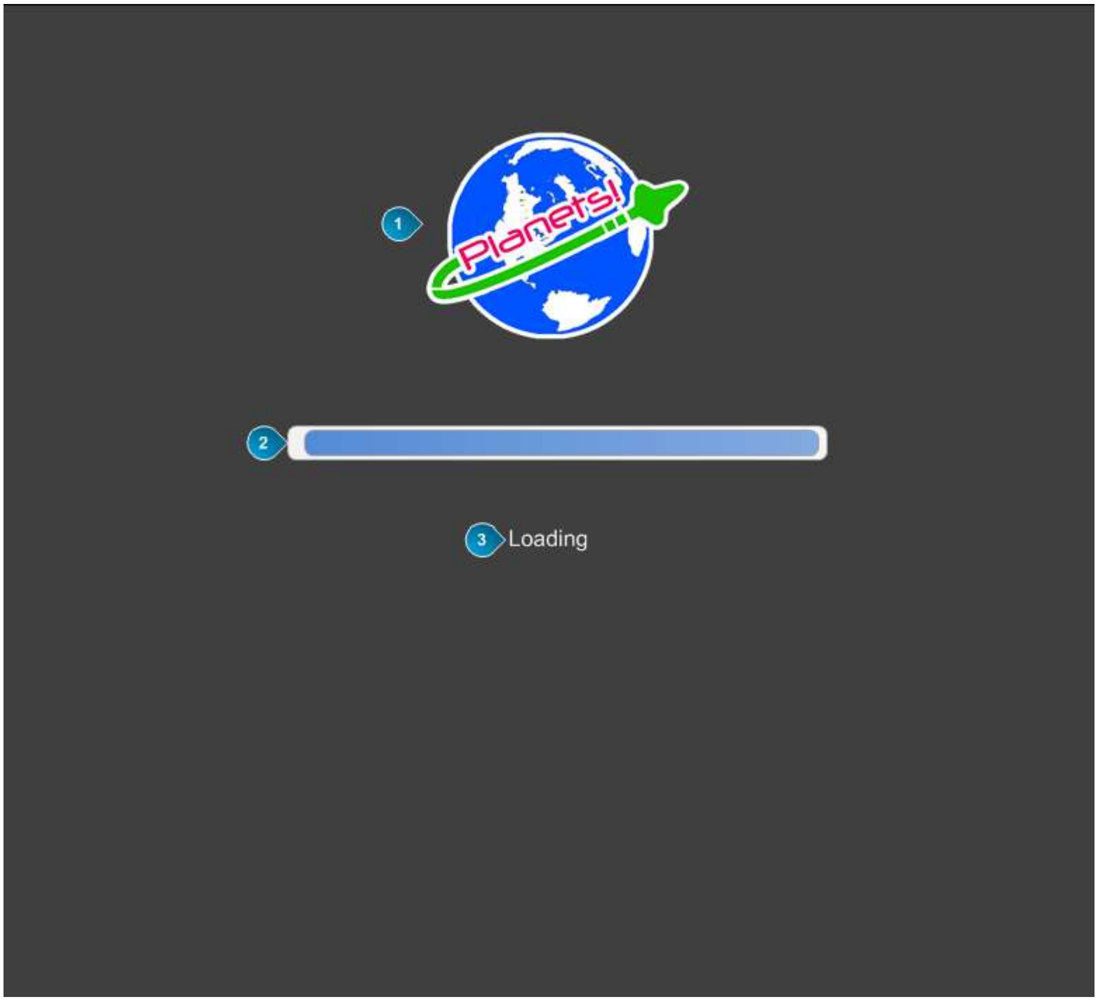


Figure 41: Loading Screen Wireframe

Table 28: Loading Screen Table

Item	Name	Detail
1	Logo and Background	Displays the logo of the game as well as the background
2	Progress Bar	Animated bar that is filled up as the game is done loading
3	Loading Text	Static text that serves to remind the player that the game is loading

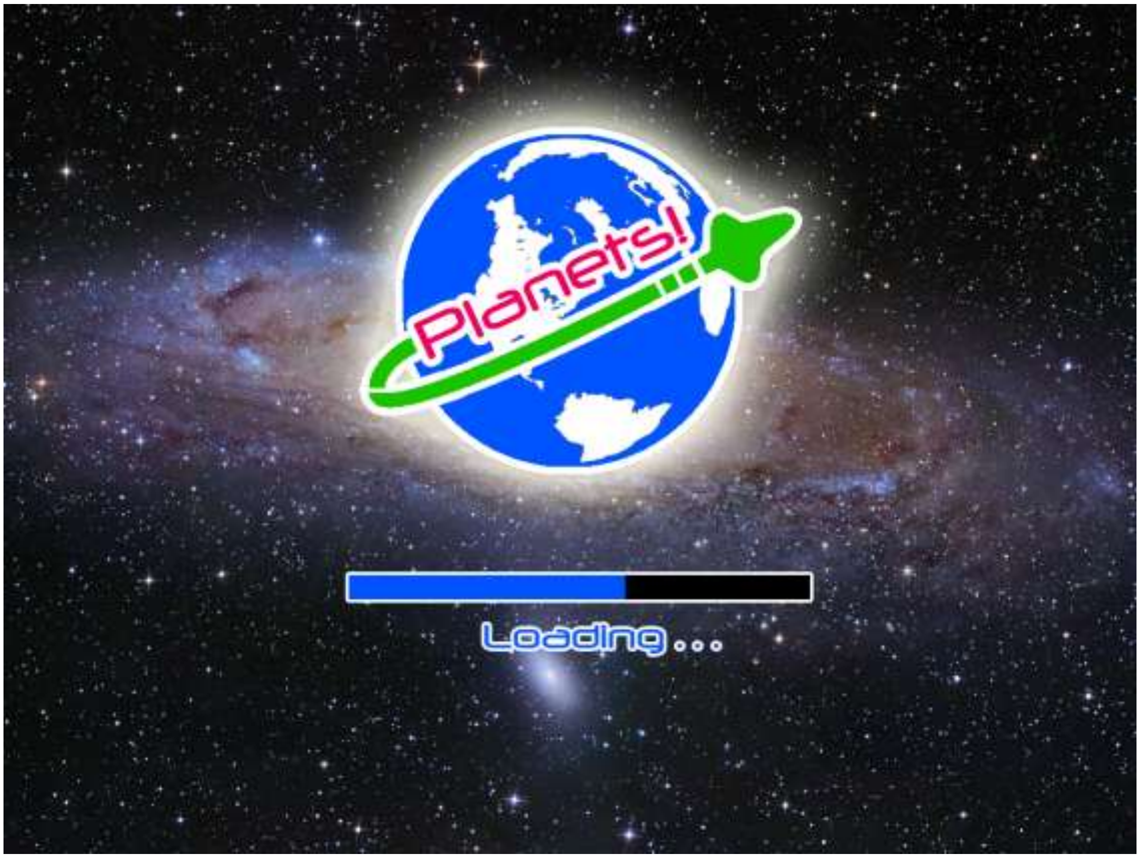


Figure 42: Loading Screen Mockup