**Brave New Galaxy**

**The Design Document**

Creator: Bill DeHaven

Version #: 1.0.1

Date: February 25, 2015

Contents

[Version History 3](#_Toc412996324)

[Game Overview 4](#_Toc412996325)

[Design Language 5](#_Toc412996326)

[In-Game World/Object Design: Xeno-Grunge 5](#_Toc412996327)

[Graphical Style: Illustrative Retro-Noir 5](#_Toc412996328)

[Game Mechanics 6](#_Toc412996329)

[Procedural World Generation 6](#_Toc412996330)

[Galactus Prime 6](#_Toc412996331)

[Planets 8](#_Toc412996332)

[Sandbox-Style Gameplay 8](#_Toc412996333)

[Massively-Multiplayer Interaction 8](#_Toc412996334)

[Game World 9](#_Toc412996335)

[The Galaxy 9](#_Toc412996336)

[Star Systems 9](#_Toc412996337)

[Planets 9](#_Toc412996338)

[Ecosystems 9](#_Toc412996339)

[Races 9](#_Toc412996340)

[History 10](#_Toc412996341)

[Technology 10](#_Toc412996342)

[Travel 10](#_Toc412996343)

[Extras 11](#_Toc412996344)

[Resources 12](#_Toc412996345)

[Future Features 13](#_Toc412996346)

# Version History

* 1. – (Oct. 12, 2014) Design Document first published. (Oct. 12, 2014)
     1. – (Feb. 25, 2015) Some explanation of galactic generation provided, as well as list of alien species.
     2. – (March 1, 2015) Added information on:
* Updated Game Mechanics section with the following:
  + Massively-Multiplayer Gameplay and World-Persistence based on utilization of a fast read/write database system.
  + Methods and strategies for optimizing World-Generation mechanics.
* Updated Game Overview with more descriptive genres – also modified MMORPG to MMOG and added possible expansion titles.
* Updated In-Game Graphical Style: Grunge to provide more description of how the style will be used within the game itself.

# Game Overview

Project: Brave New Galaxy

Possible Expansion Titles: Beyond the Stars, Dark Matter, Sagittarius,

Media Type: Java Desktop Game

Target Audience: Males, 12- to 30-year-olds, MineCraft players, sci-fi fans, rpg players, MMO players.

Genre(s): MMOG, Voxel, Sandbox, Sci-Fi

Screen Projection: 3D

Programming Language/Environment: Java w/JOGL for OpenGL

# Design Language

According to Wikipedia:

"A design language or design vocabulary is an overarching scheme or style that guides the design of a complement of products or architectural settings. Designers wishing to give their suite of products a unique but consistent [look and feel](http://en.wikipedia.org/wiki/Look_and_feel) define a design language for it, which can describe choices for design aspects such as materials, colour schemes, shapes, patterns, textures, or layouts. They then follow the scheme in the design of each object in the suite" (http://en.wikipedia.org/wiki/Design\_language, accessed October 11, 2014).

## In-Game World/Object Design: Xeno-Grunge

“Grunge” – a style deriving influence from an emotionally dark, hard rock music genre. Centered in Seattle, this music inspired art in a variety of northwest locations including coffee shops and pubs. It searches for beauty in the decay and disorder of reality. The game will utilize these themes of the Grunge style to elicit an old and decaying universe which has developed an immensely complicated array of organisms, ecosystems, and societal structures.

“Xeno” – pertaining to aliens, from the greek for *guest*, strange. “Strangeness” will be another overarching theme of the game, as each of the races within the game, alien or humanoid, live in a very different way than we do today. Even human beings in the game will be living in a time several thousand centuries in the future, and their worldview, though with similarities, will effectively be alien to our own.

## Graphical Style: Illustrative Retro-Noir

Illustrative Retro-Noir is both moody and epic, semi-realistic but pixeled -- somewhat decayed. Similar to Quake 2 and the 3D games of that era, it should allude to the sophisticated but dilapidated technology systems featured in the game.

The main GUI elements should replicate the technology of the player's race, whether that be digital, biological, chemical, nuclear, mechanical, or mental. GUI elements should also make intuitive use of transparency and allow the player to align the various menus and control their attributes in a layout that makes sense to them.

Each 3D model and mesh should use enough polygons to create a simple detail within the game, but only that many. The game's graphical style does not require realistic graphics, but will strive to shy away from 'block' graphics.

Each race will also utilize a style, or styles, of architecture that make sense with their history and values. Each race will also have its own color palette, though items used by or manufactured by that race do not necessarily always need to use just those colors. Above all else, the graphical style for the world and the objects within it must seem natural – not a realistic natural, but a kind of used, worn down, actually-existing-within-a-world-of-entropy kind of natural.

# Game Mechanics

The game focuses on several core Game Mechanics in order to produce an **Open-Ended game world with Integrated Interstellar, Planetary, and Abstract domains**: **Procedural World Generation, Sandbox-Style Gameplay, and Massively-Multiplayer Interaction.** The goal of the game-design process is to create an underlying architecture which brings all of these mechanics together seamlessly, without interrupting the player's *Suspension of Disbelief*. Many MMOGs allow the use of these elements but not without the player delving extensively into the menu-system or using game-defined interfaces in order to accomplish their chosen task; for example, players usually cannot move uninterrupted from one area of the game-world to another or from one instance of a map-area to another – often, a pause in the gameplay is required in order to load map details and resources.

The following Venn diagram illustrates the relationships between each of the three main game mechanics:

## Procedural World Generation

Procedural generation is a process whereby an algorithm and a seed number are used to initialize a complex object – in the realm of game world creation, the algorithms and world seed are used to generate an entire game world. Various games utilize this method at different complexities and levels – for instance, some games use generation algorithms to produce seemingly random terrain and resource placement, whereas other games use generation algorithms at an even deeper level to produce various kinds of in-game objects or storylines. The galaxy and all planets will utilize Procedural Generation, a process of using consistently applied seeds and algorithms to generate, and later reconstruct, structures and features.

Both the in-game Galaxy objects and World objects will be composite objects – components made up of even more components. Naturally, this creates a tree hierarchy which can serve very well to generate a realm procedurally, especially at higher and higher resolutions as the need arises.

Galaxy objects are composed of System objects which are composed of Stellar objects which are composed of World objects. It is important to note that an object may not necessarily be the same kind of celestial body as is implied by the level in the hierarchy it belongs to – for instance, a System object can just as easily be a star system as a nebula or a gas cloud. Likewise, a Stellar object can just as easily be a black hole as much as a star.

Planet objects are composed of

### Galactus Prime

**Galactus Prime will be the test-bed galaxy for all testing of procedural generation. Eventually, multiple galaxies may exist across one or more servers, each representing a shard on which players can play *Brave New Galaxy*.**

**Each galaxy will have its own Primary Key, serving as its unique identifier. In addition, it will also have its own Seed, allowing it to be procedurally generated every time the generation algorithms are run. Galactus Prime will have the following properties:**

* **Primary Key: 0**
* **Name: Galactus Prime**
* **Seed: 20151963**
* **Size: 1024**
* **Type: Elliptical**

When first constructed, the galactic generation algorithm will be used to create a finite number of System Level objects, which are the primary components of the galaxy. Each system will be recorded and maintained in a database table representing the galaxy – this is the only part of the game which will not be dynamically generated at run-time. Each System Level object will have the following properties assigned to it: Primary Key, Seed, System Type, X-Coord., Y-Coord. The procedure for generating these objects, and the participants, will be as follows:

1. Within the 'WORLD\_GENERATION' game state, initialize the GalaxyGenerator director object with an EllipticalGalaxyBuilder builder object assigned to it. The GalaxyGenerator will request the EllipticalGalaxyBuilder to accomplish the following steps:
   1. Initialize GalaxyRoot as the root object for the galactic structure. This object is a composite, as are its children, and will be returned to the game by the GalaxyGenerator once generation is complete.
   2. Set the number of arms, by using a noise function with the galaxy's seed fed into it. The number should be between 1 and 7 inclusive, and should gravitate toward numbers closer to 4. The field armCount will be set with this number. Even though the arms are not necessary for some galaxy types (Elliptical, for instance, or irregular), this value will still be preserved – however, it may be ignored if necessary.
   3. The system's seed and the galaxy's seed will then be used to determine the system's type (Star System, Nebula, Dust Cloud, Black Hole, etc.).
   4. Starting at 0, use another noise function to initialize the characteristics of each system. The noise function should initially be fed the galaxy's seed and the system's primary key to find the system's seed, then the system's seed and an incrementing counter should be used in a noise function to generate random X and Y coordinates, as well as a dice with 100 sides. Each set of coordinates will be compared to the nearest arm to determine the probability of it existing there – if the probability is lower than the dice roll, those coordinates become that system's location; otherwise, the counter is incremented and the next set of coordinates and dice roll are generated.

### Planets

## Sandbox-Style Gameplay

## Massively-Multiplayer Interaction

The game

# Game World

## The Galaxy

A procedurally generated galaxy akin to our own Milky Way Galaxy.

## Star Systems

Star Systems can include any number of stars, though higher numbers become increasingly less likely. The following is the current list of objects which can be found within a star system:

1. Star
2. Asteroid Belt
3. Comet
4. Nebula
5. Planet
6. Black Hole

Stars can range from white to red and from super giant to dwarf. A few other star types exist too, including wolf-rayet, neutron, and quasar.

## Planets

The following planets will exist in the game:

1. Terran (Rocky, Earth-like)
2. Gas (Jovan)
3. Icy
4. Ocean
5. Volcanic
6. Carbon

## Ecosystems

Bears

## Races

Humans

Androids

Plants, Gaia

Rock-Based (Basalt?)

Jellyfish

Crustacean-like, Methane breathers

Cuttlefish? (Not playable)

## History

## Technology

## Travel

# Extras

# Resources

# Future Features