





JEFF SALVAGE, CCI ADVISOR



Laura Mo *Team Lead*



Colan Biemer *Programming Lead*



Gabby Getz *Programmer*



Shreya Patel *Programmer*



Jasmine Marcial *Programmer*



Alliy Frauenpreis *Programmer*



DR. FRANK LEE, STAKEHOLDER & DIGM ADVISOR



Matt Bodner *DIGM lead*



Dave Petersen *Artist*



Michael Rodriguez *Artist*



Keano Jan Osmillo *Artist*



Andrew Mylet *Artist*



Joseph Santos *Artist*

PROJECT MOTIVATION



2015-2019 GLOBAL GAMES MARKET

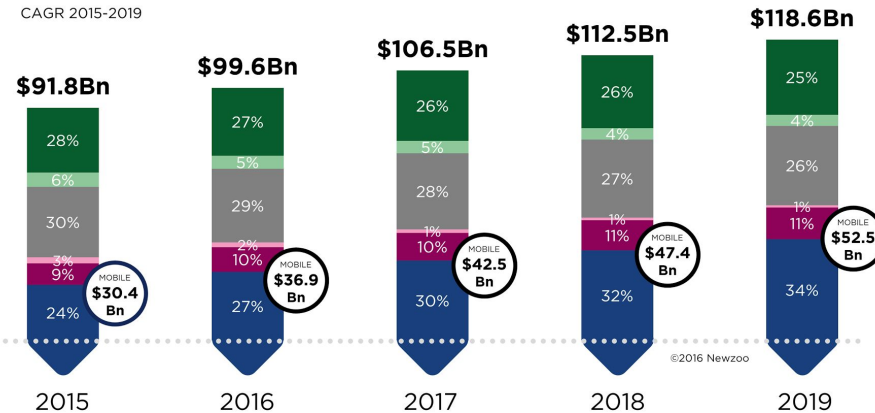
FORECAST PER SEGMENT TOWARD 2019

TOTAL MARKET

+6.6%

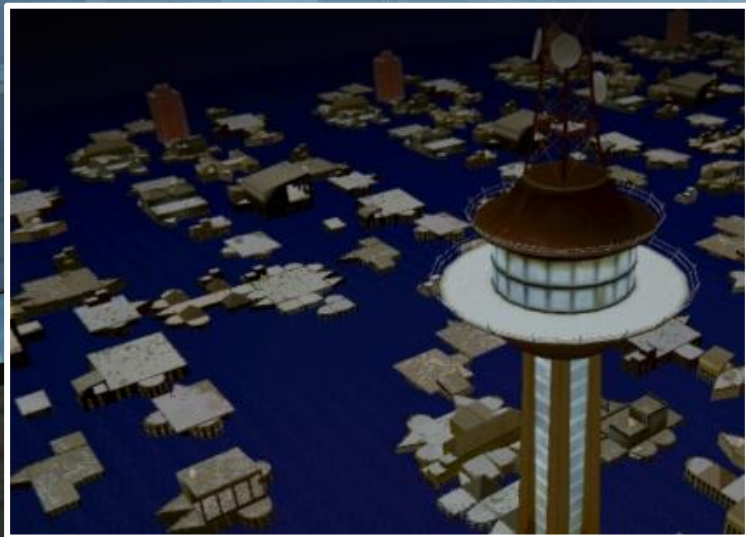
CAGR 2015-2019

Smartphone Tablet Handheld TV/Console Casual Webgames PC/MMO



HIGHWATER IS

A roguelike survival game



Survival

Manage health, warmth, and hunger

Permanent death

Procedural Cityscape

Different environment every playthrough



Intuitive Crafting System

Multipurpose and dynamic items



Discoverable Story

Integrated into the environment and radio system



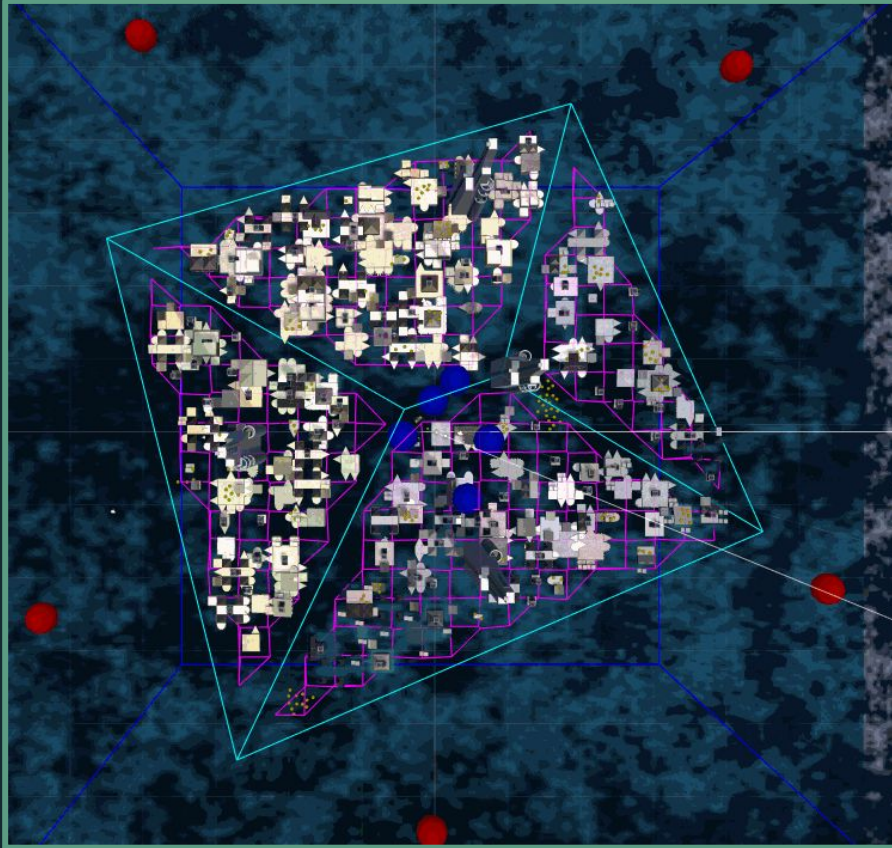
Simulated Weather

Realistic weather system the player can predict
City slowly filling with water

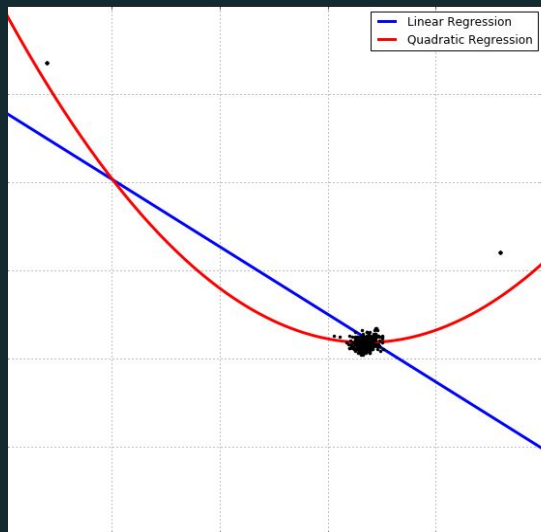


	Call of Duty	Banished	Highwater
Precipitation	✓	✓	✓
Water	✓	✓	✓
Temperature		✓	✓
Weather Based Events		✓	✓
Wind Speed			✓
Based on Real Data			✓
Dynamic Pressure System			✓

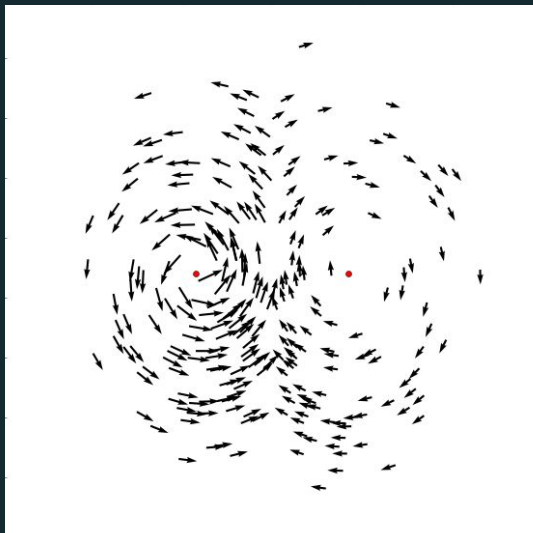
PRESSURE SYSTEMS AND WEATHER



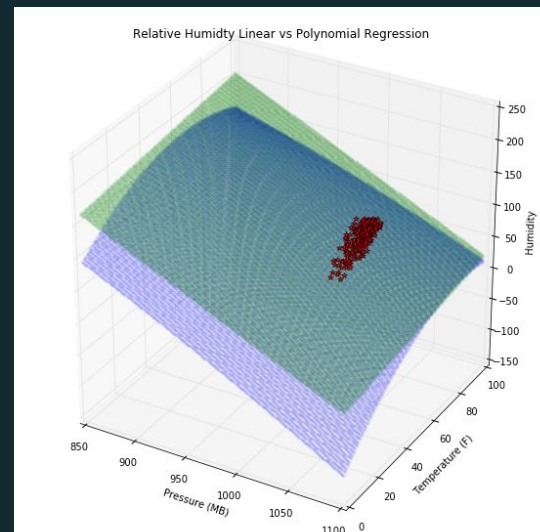
WEATHER SYSTEM



Sea Level Pressure to Wind Speed

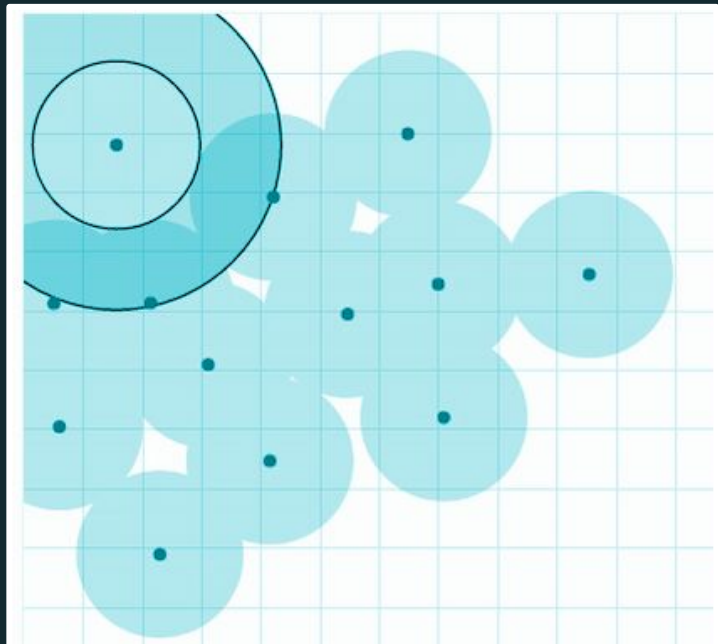


Wind Speed Direction Compared to Pressure Centers

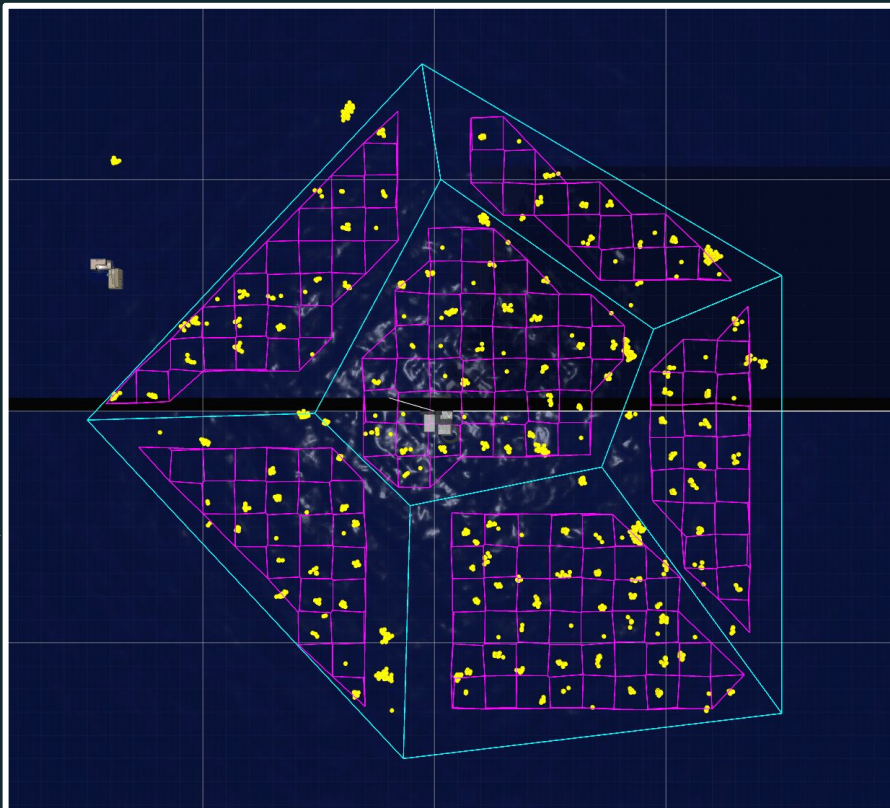


Relative Humidity from Pressure and Temperature

SAMPLING POINT GENERATION

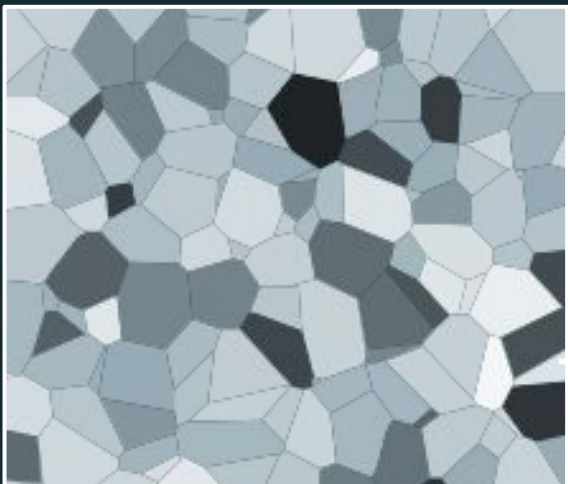


Poisson Sample Point Generation

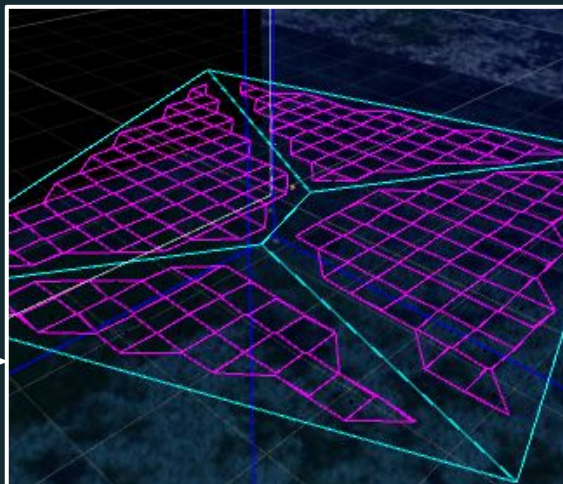


Item Placement

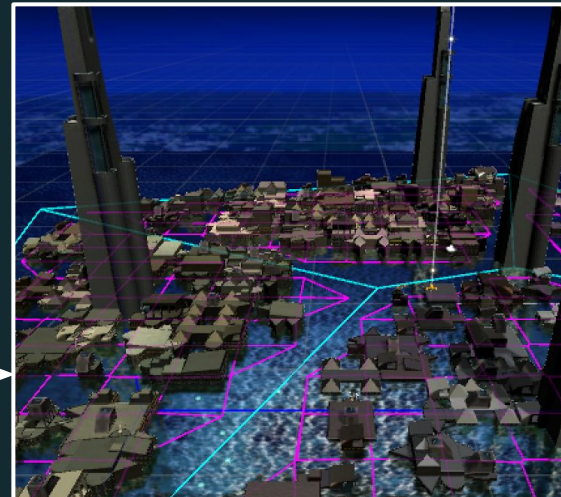
PROCEDURAL CITY GENERATION



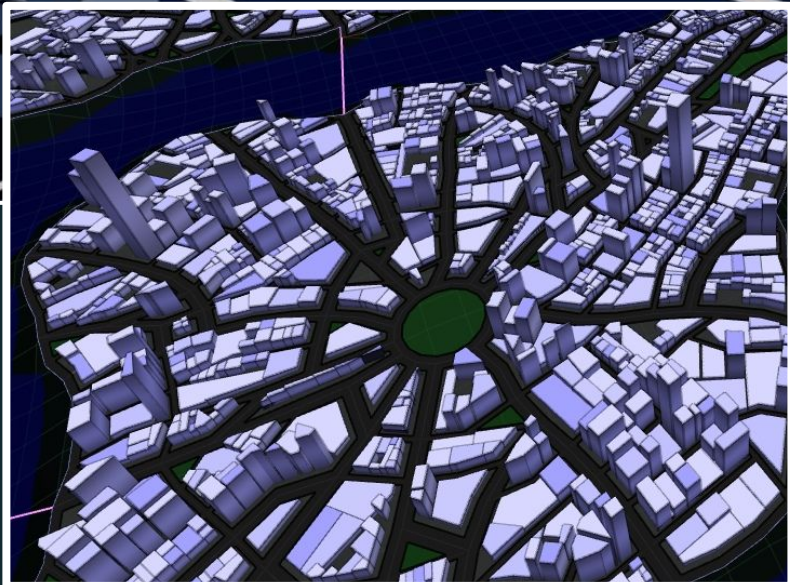
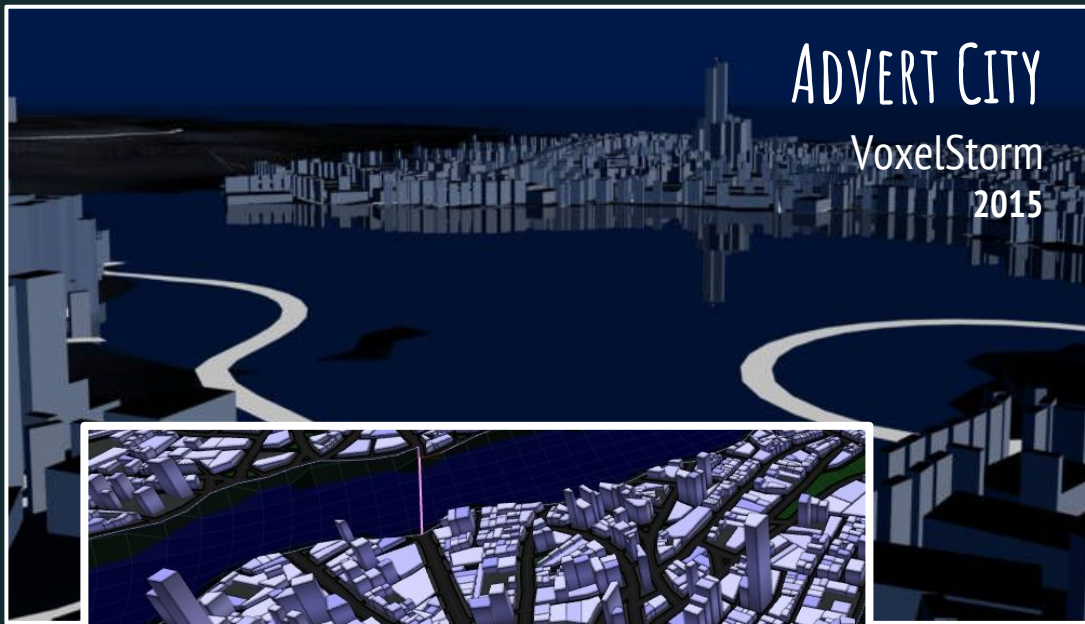
Voronoi Diagram with seed points created with **Poisson Distribution** to generate districts and city edges.



Voronoi Diagram with regularly spaced seed points to create city blocks.



Generate building meshes and pack buildings onto edges of each block



COMPARISONS

Also block-based city layouts

Buildings are **voxel** which are more easily managed in memory

SUBVERSION

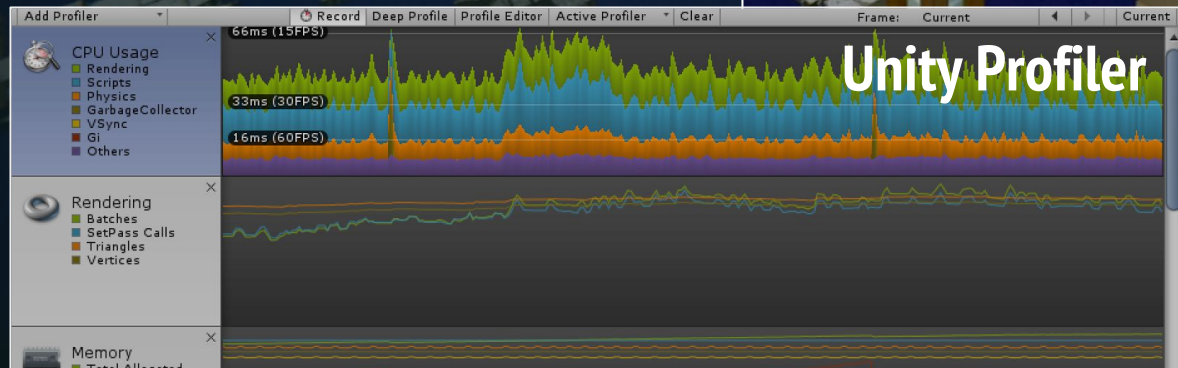
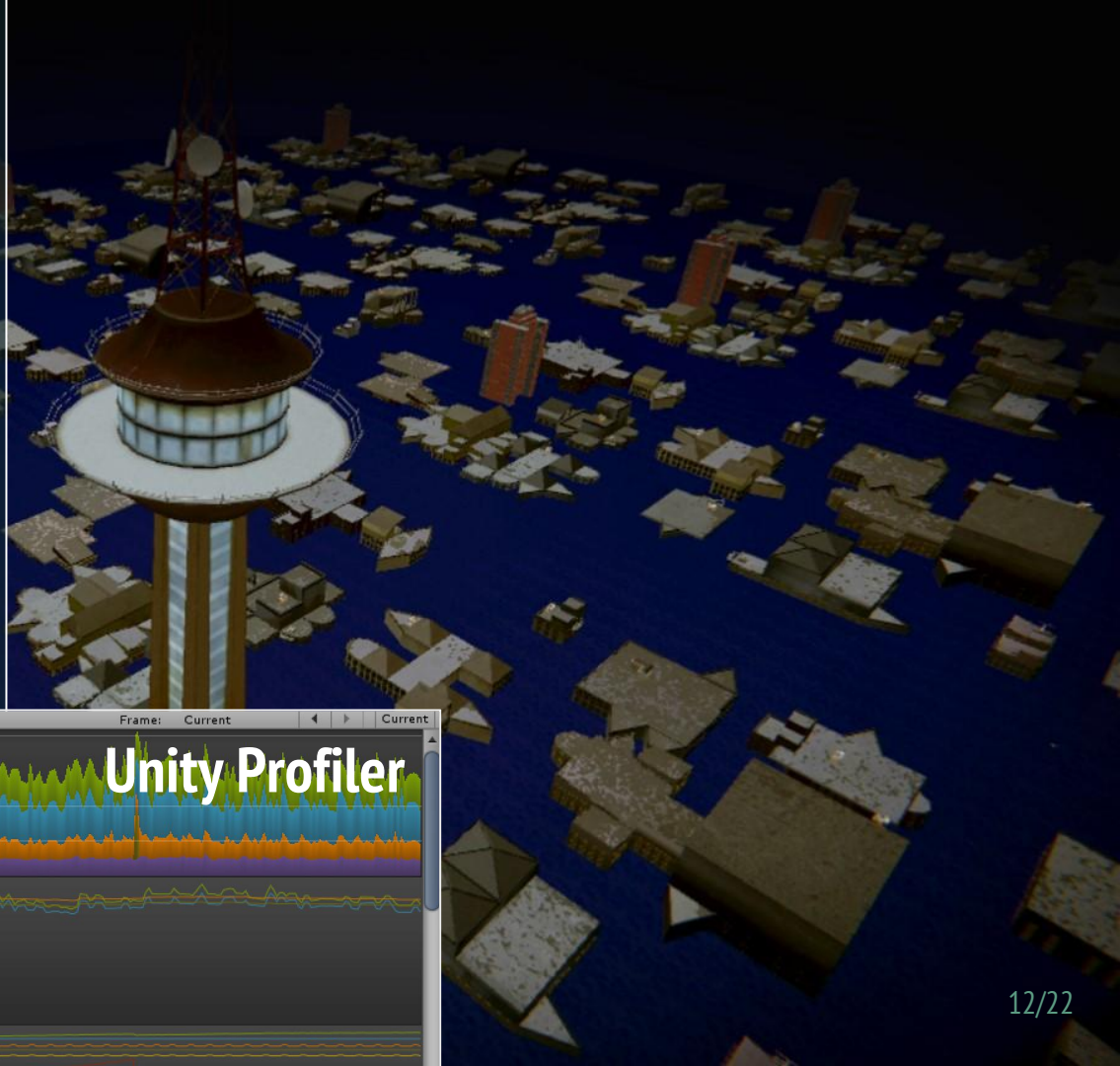
Introversion Software
(Unreleased)

CITY OPTIMIZATION

Load city buildings in **chunks**

Creature and item **pools**

Hide buildings with fog to save on
rendering complex meshes in
distance



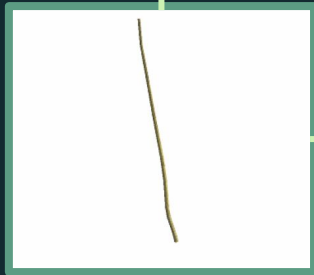


50,000 FT VIEW
ARCHITECTURAL DIAGRAM

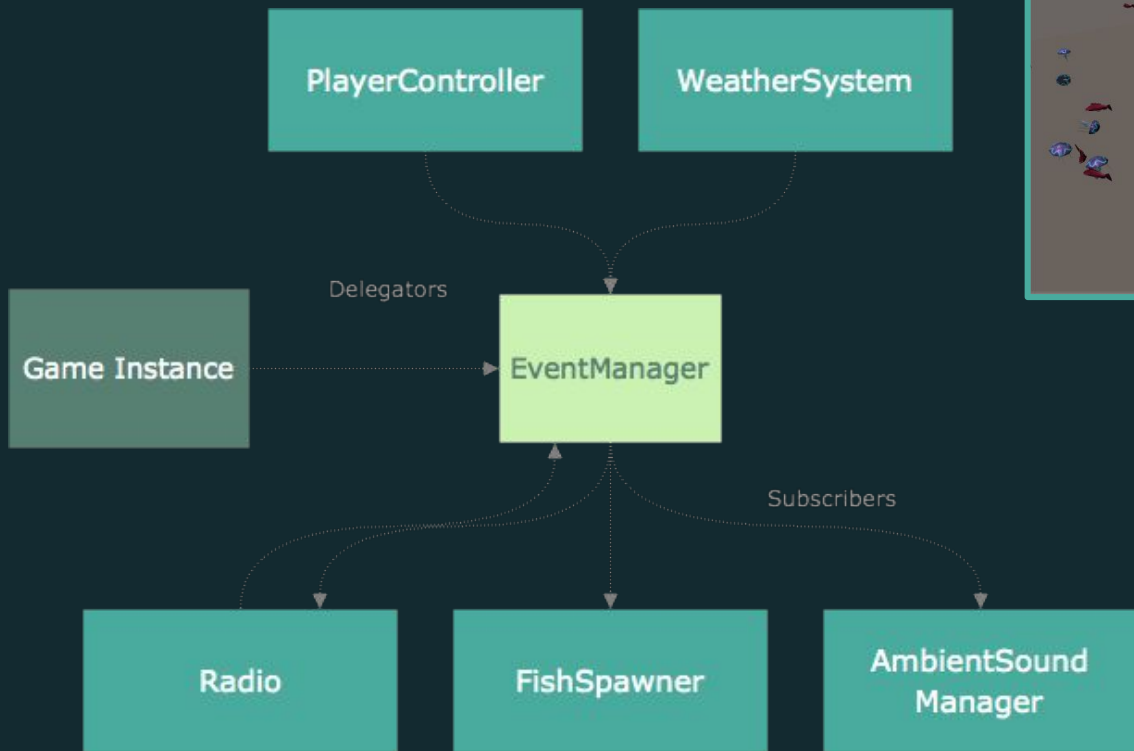
ITEM CRAFTING



Config
Files



EVENT MANAGER




DEVELOPMENT PIPELINE




user creates pull request

reviewer approves or denies



user merges pull request





**Changes requested**

Hide all reviewers



1 review requesting changes and 2 approved reviews by reviewers with write access. [Learn more.](#)

 **bi3mer** requested changes


[See review](#) [Dismiss review](#)

 **ggetz** approved these changes

[See review](#) [Dismiss review](#)

 **mpb84** approved these changes

[See review](#) [Dismiss review](#)

**Merging is blocked**

Merging can be performed automatically once the requested changes are addressed.

Squash and merge

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

Game Performance

Run at 60 FPS on machine with recommended requirements

XML Documented

All code required standard C# XML documentation for public methods and members

Coding Style

Followed Microsoft C# Coding Conventions as well as own Style Guide

Unity Best Practices

Proper asset naming, project settings, and directory structure, use of prefabs

SOFTWARE & GAME DEVELOPMENT STANDARDS



Designer Friendly

All values were exposed to be changed by designers through Unity and configuration files

GOF Design Patterns

Including Bridge, Controller, Decorator, Factory, Observers, Singleton, etc.

Playtesting

Qualitative game testing in addition to software tests

Dynamic UI

Scalable, responsive, and user tested

TESTING

NUNIT & UNITY TEST TOOLS

NUnit Framework 2.6 for backend code tests

Qualitative testing done through **playtesting** in tandem with the DIGM Team

```
1 using UnityEngine;
2 using UnityEditor;
3 using NUnit.Framework;
4
5 [TestFixture]
6 public class BaseItemTests
7 {
8     [Test]
9     public void AddItemCategoryMethodShouldUpdateGetItemCategoryList()
10    {
11        //Arrange
12        BaseItem stick = new BaseItem("Sample Stick");
13        SolidCategory solid = new SolidCategory ();
14        stick.AddItemCategory (solid);
15        Stack stickStack = new Stack (stick, 4, "");
16
17        // Act
18        stickStack.Item.GetItemCategories();
19
20        // Assert
21        Assert.AreEqual (1, stickStack.Item.GetItemCategories().Count);
22    }
23 }
24
```

CONTINUOUS INTEGRATION

UNIT TESTING & UNITY CLOUD BUILD



Highwater » Cloud Build ▾

Gabby Getz | 56983c99-0609-42d1-82cd-4807a2aea8a4 | PLAN: Personal

Start New Builds ▾

History

Stats

Collaborators

Config

Notifications

Target: All ▾


Status: All ▾

Per Page ▾

Columns ▾

Other Options ▾

Delete Selected Builds

<input type="checkbox"/>	Fav	Status#	Target	Commit	Wait Time	Build Time	Time Since	Size	Details	Install
<input type="checkbox"/>	☆	✓ #16	 Default WebGL	01c3581...	01:00:35	00:11:23	7 days ago	8.2 MB	Share Summary Changes: 56 FILES Full	Play ▾
									Log Compact Log	

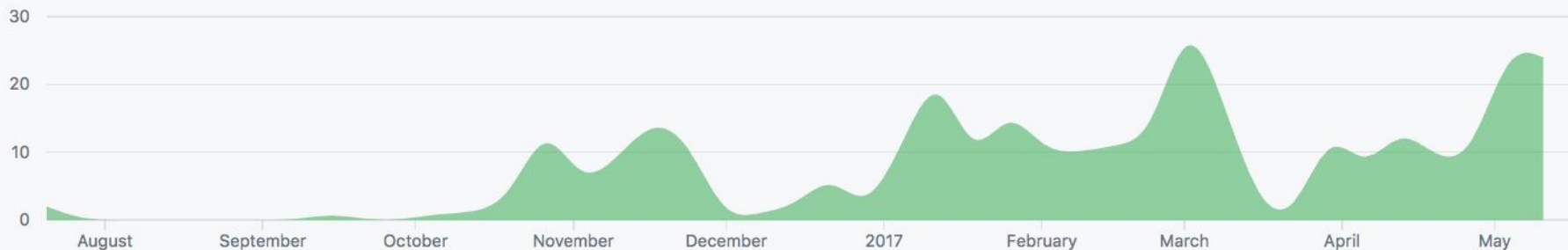
STATISTICS

📁 357 commits

🌿 35 branches

📦 0 releases

👤 12 contributors



23,219

lines of code

1041

lines of unit tests

368

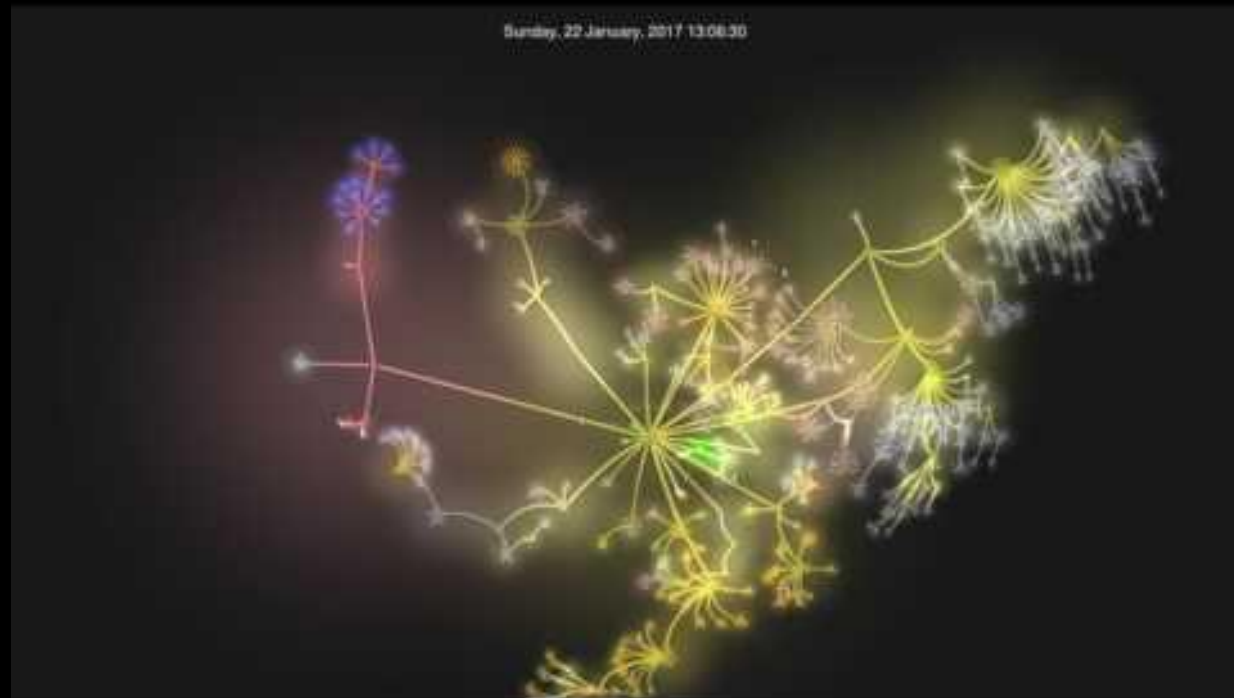
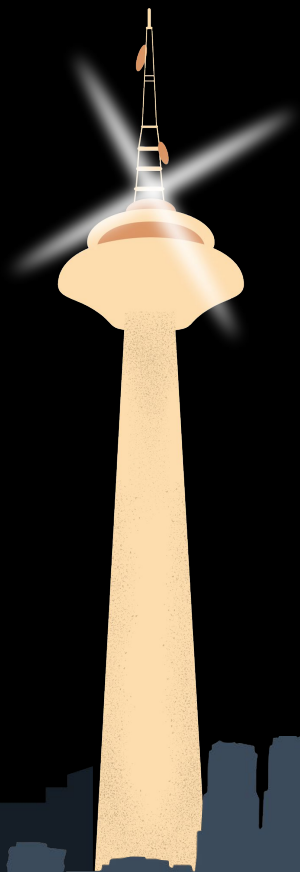
closed pull requests

422

closed issues

46

play-tests



A humpback whale is shown swimming in the ocean, viewed from below. The whale's body is dark grey with a lighter, white patch on its side. Its long, curved tail is visible at the bottom left. The background is a deep blue ocean with some lighter blue patches. The text "THANK YOU!" is overlaid in the center of the image.

THANK YOU!