# P2: Hash Landscapes

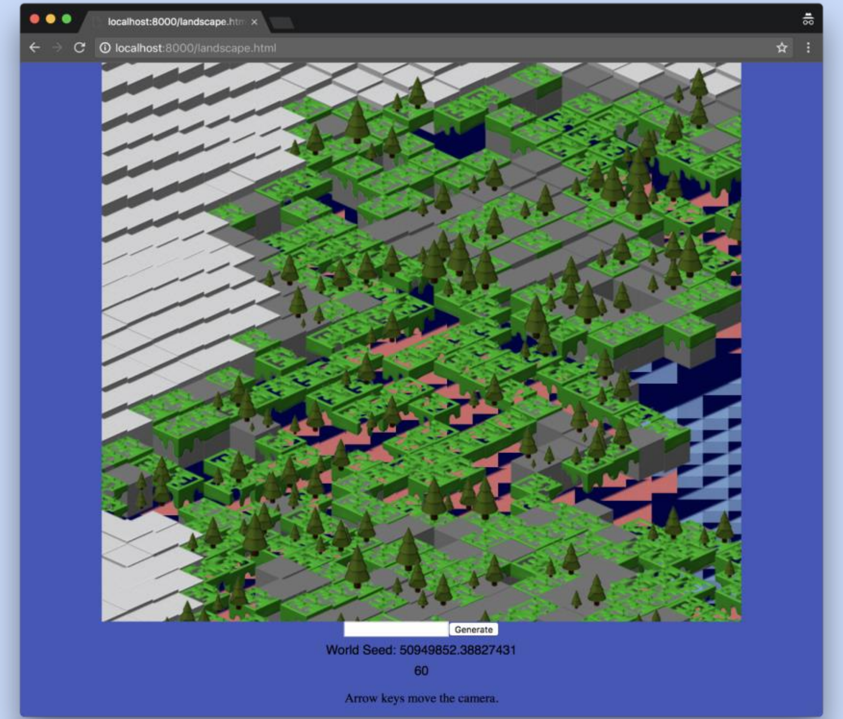
Due 10/25/2017

## Overview

In this programming assignment, you'll create a **pseudo-infinite**, **pseudo-random** terrain design exploiting the idea of hash trees. Every minute detail of your world should be uniquely determined by a user-provided worldgen key.

Practice goals:

- Use noise functions to create smooth variations
- Use hash trees to make repeatable random choices.
- Derive both continuous and discrete choices from the discrete state of the hash.



World Seed: 50949852.38827431
60
Arrow keys move the camera.

## Support code

**https://drive.google.com/file/d/0B_1RQdh4LMQ3UGUxYVRwaXE3d3c/view?usp=sharing**

**The zipped folder contains:**

landscape.html

world.js

support.js

example_tile.png

xxhash.min.js

p5.js

P5.min.js

/addons/dom.js

/addons/dom.min.js

## You should implement...

function myPreload()

function mySetup()

function myTileHeight(i, j)

function myTileVariation(i,j, height)

function myDrawTile(i,j, variation)

function myHandleClick(i,j)

function myHandleWorldgenStringChange(key)

Placeholder implementations of these functions are provided in **world.js** of the support code. Feel free to modify any aspects of **support.js** (which builds on **p5.js** and **xxhash.js)** you need to achieve the desired effect.

Add your artist statement to **landscape.html. You may also optionally modify** it to customize the look of the page outside of the P5js canvas.

## function myPreload()

Does rendering your landscape require drawing any images from files? Load them in this function just like you (probably) did in **preload()** for P1.

## function mySetup()

Does your world require any one-time setup (such as transforming the preloaded images)? Do that in this function just like you (probably) did in **setup()** for P1.

## function myTileHeight(i, j)

The support code will call this function to determine the vertical offset of tile **i,j**.

To implement smoothly rolling terrain, use scaled versions of **i** and **j** as the two arguments to P5js’s **noise(x,y)** function. If you want **noise(x,y)** to be sensitive to the worldgen key, make sure to call **noiseSeed(seed)** at some point.

## function myTileVariation(i, j, height)

Your tiles shouldn’t just be identical floating cubes. Customize this function to derive the specific look for tile **i**, **j**. The support code will give the value returned by **myTileHeight(i, j)** to this function. The datatype for variations is up to you: it could be a boolean, a number, or some more complex JavaScript object (even including the hash object).

The variation of a tile should be influenced by user interaction. See **myHandleClick(i,j)**.

This function shouldn’t do any drawing itself. It should just return the information that **myDrawTile(i, j, variation)** will use.

## function myDrawTile(i, j, variation)

Here’s where you actually draw a specific tile with a specific visual appearance variation. The support code uses **translate(x,y)** to move the drawing origin to an appropriate location on the screen for drawing images with **image(img,x,y)**. If you make use of **translate(x,y)** in your **myDrawTile(i,j,variation)** function, make sure to un-translate back when you are done.

Be creative with the P5js API to give the tiles in your project a unique look. You could draw multiple layers of images and overlay them with other shapes and lines. To bring your tiles to life, access the current time with **millis()** and use a scaled copy of it with **noise()** to get signals that wiggle in time (for sparkling lights, waving grass, or quaking earth).

## function myHandleClick(i, j)

The user has just clicked on tile **i**,**j**. Your implementation of this function should somehow remember this. Here’s how Adam suggests counting clicks on a tile:

```
let clickCounts = {};

function myHandleClick(i,j) {
	clickCounts[[i,j]] = 1 + (clickCounts[[i,j]]|0);
}

function getClickCount(i,j) {
	return clickCounts[[i,j]]|0;
}
```

# function myHandleWorldgenStringChange(key)

The user has just entered a new worldgen **key** in the box below the canvas. You implementation should somehow remember this.

Consider setting a global variable (worldSeed?) based on the hash of this string. This global variable should influence how **myTileVariation(i,j, height)** and other functions work.

Here is the documentation for the xxHash library used in our support code: https://github.com/pierrec/js-xxhash

## Artist Statement

Help us understand your project by writing a brief statement (1 paragraph).

If you don’t know what to write, consider these guildines for middle-school artists: https://www.theartofed.com/content/uploads/2015/09/Artist-Statement-Flow-Chart-final1.pdf Only write what’s meaningful for your project; don’t ramble just to take up space. At the same time, don’t just write code without thinking about the bigger picture.

## Inspiration

## Stuff you can try to represent

**Indoor**

Hardwood flooring

Carpet

Kids toys

Furniture

People

Pets

...

**Outdoor**

Water

Lava

Dirt

Grass

Trees

Roads

Buildings

...

**Elsewhere**

Cybertronic gridwork

Energy orbs

Fire

Crystals

Bio foams

Abstract tree things

...

## Techniques (try these if you need ideas)

Feeding one generator into another (perhaps height changes drive another effect?)

Combining multiple functions or noise values

Filtering the noise function

Use biomes to create regions with shared values or varied generation approaches

Layer multiple tiles on top of each other, using the transparency/alpha channel

Use arrays of images, or animate them

Use other mathematical functions

**More advanced:**

Using point noise or splatting

Procedurally generate the tile graphics (in part or completely).

Dividing the world into chunks and doing some post-processing on the chunks (Minecraft does this)

Looking at neighboring tiles to drive properties, like using height differences to determine if this tile is a sharp cliff

Tiles that connect to each other, like roads

## Grading Criteria

[1pt] P5js's noise() function is hash-influenced by worldgen key.
[1pt] Tile height is hash-influenced by worldgen key.
[1pt] Tile variation is hash-influenced by worldgen key.
[1pt] Tile variation is hash-influenced by height.
[1pt] Tile variation is hash-influenced by user clicks.
[1pt] There are at least two visibly distinct variations in tile appearance.
[1pt] The appearance depends on a meaningful modification of the tile template image (either by manually painting on the image or drawing procedural detail on top of it).
[1pt] The appearance of a tile somehow depends on time even without user interaction.
[1pt] An artist statement is present in your html page.
[1pt] The project demonstrates something you uniquely added: can be an aesthetic look, a technical implementation, or something else. Does not have to be big or grand, it just has to be something that makes this project uniquely yours. (You should probably mention what it is in the artist statement.)