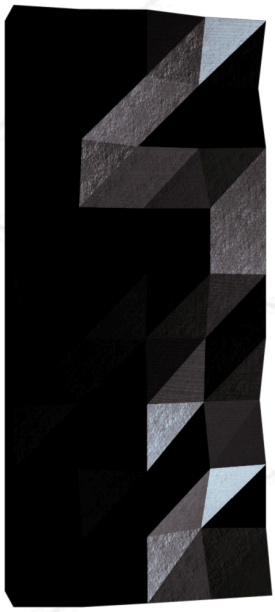


ft_minecraft

pimp my world

Summary: ft_vox but harder

Version:

Contents

Chapter I

Preamble

Mojang Studios is a Swedish video game developer based in Stockholm. It was founded by the independent video game designer Markus Persson in 2009 as Mojang Specifications for the development and release of Persson's sandbox and survival video game Minecraft. The studio inherited its name from a previous video game venture Persson had left two years prior. Following the game's release, Persson, in conjunction with Jakob Porsér, incorporated the business as Mojang AB in late 2010 and hired Carl Manneh as the company's chief executive officer. Other early hires included Daniel Kaplan and Jens Bergensten.

Minecraft became highly successful, eventually the best-selling game of all time, giving Mojang sustained growth. With a desire to move on with the game, Persson offered to sell his share in Mojang, and the company was acquired by Microsoft through Xbox Game Studios (then known as Microsoft Studios) in November 2014. Persson, Porsér, and Manneh subsequently left Mojang, with Jonas Mårtensson replacing Manneh. In May 2020, Mojang was rebranded as Mojang Studios.

As of 2021, the company employs approximately 600 people. Executives include chief executive officer Mårtensson and studio head Helen Chiang. Apart from Minecraft, Mojang Studios has developed Caller's Bane, a digital collectable card game, Crown and Council, a turn-based strategy game, and the dungeon crawl game Minecraft Dungeons. It also released various smaller games as part of game jams organised by Humble Bundle.

In 2011, Persson and Kaplan envisioned a hybrid of Minecraft and Lego bricks and agreed with the Lego Group to develop the game as Brickcraft, codenamed Rex Kwon Do (in reference to the film Napoleon Dynamite). The game has also been described as a first-person shooter. Mojang hired two new programmers to work on the game, while a prototype was created by Persson. However, Mojang cancelled the project after six months. Upon announcing the cancellation in July 2012, Persson stated that the move was performed so that Mojang could focus on the games it wholly owned. Daniel Mathiasen, a Lego Group employee at the time, later blamed the cancellation on a series of legal hurdles that the Lego Group had put in place to protect the product's family-friendly image. Kaplan lamented that the staff at Mojang had felt more like consultants on the project, rather than its designers. The Lego Group also considered acquiring Mojang at this point but later decided against doing so as they had not foreseen that Minecraft would become as popular as it would at one point be.

Chapter II

Introduction

This project is the logical continuation to ft_vox.
But ft_minecraft will be way more advanced, especially on the Procedural Generation part, and the beauty of the result.
You will also do a bit of network, to allow multiplayer.

Chapter III

Objectives

This project will focus on two major aspects :

- Advanced Procedural Generation with different biomes, vegetation, river, 3d clouds, minerals in cavern, ..
- Advanced rendering effect (lightning, shadows, ssao,..)

Chapter IV

General instructions

- You're free to use whatever language, but keep an eye on its performances. (If you can't choose, c/c++/rust are suggested).
- You can use Vulkan, Metal or OpenGL/CL for your GPU calculations. You cannot use a library to work for you.
- You can use a library to load pictures, a windowing library and a mathematics library for your matrix/quaternions/vectors calculations. You must not push them in your repo. Instead, you must write your own download/install scripts.
- Using libraries to create noise or whatever technique to create the terrain or biomes is completely forbidden. You MUST re-code everything.
- The render should always be SMOOTH, meaning a minimum of 30 fps is expected.
- Any crash (Uncaught exception, segfault, abort ...) will disqualify you.
- Your program will have to run at 1080p or higher. Reducing the default frame buffer is prohibited.

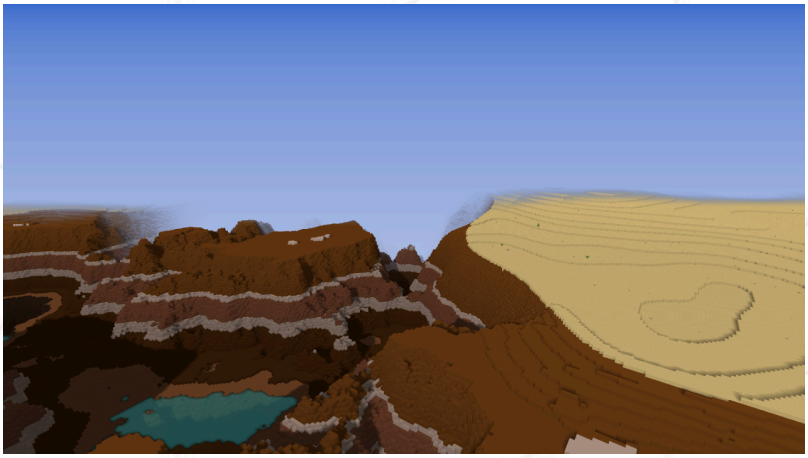
Chapter V

Mandatory part

V.1 The world

The world must be generated on demand.
You should be able to visit through atleast 5.000.000 cubes on the XZ axis.
You're free to manage as you like how exceeding the limit is handled *(invisible wall to keep the player from moving, mirror the world...)*

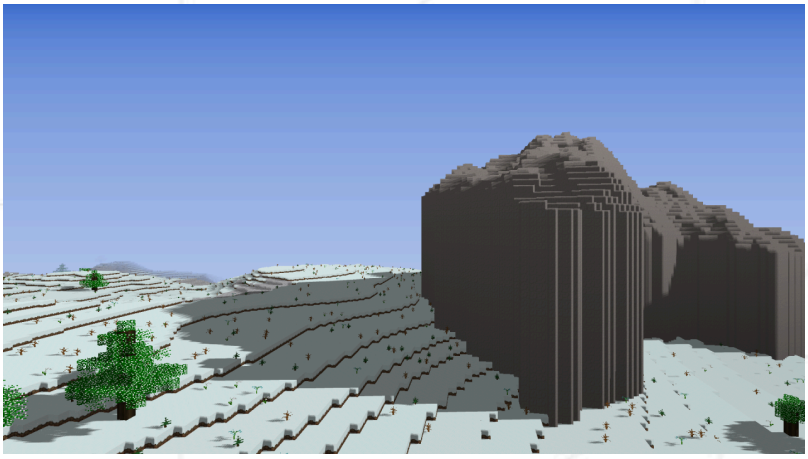
You must have a minimum of 10 biomes *(ex: Hill, Mountain, Desert, Canyon, Swamp, Sequoia forest, Island, Savanna, ..)*.
A biome is a region with unique geography, vegetation and other characteristics.
Each biome must feel really unique.
You can take as an example the biomes in Minecraft game.



Here are two biomes. A Canyon next to a Desert

The whole world cannot share the same terrain elevation, you must have mountain/-canyon/island biomes where the terrain is different from other biomes.
You must have at least 4 biomes with unique terrain elevation generation *(ex: canyon, mountain, island)*.

The transition between the biomes must be clean, by that we mean that the terrain must feel natural and be progressive *(see example below)*.



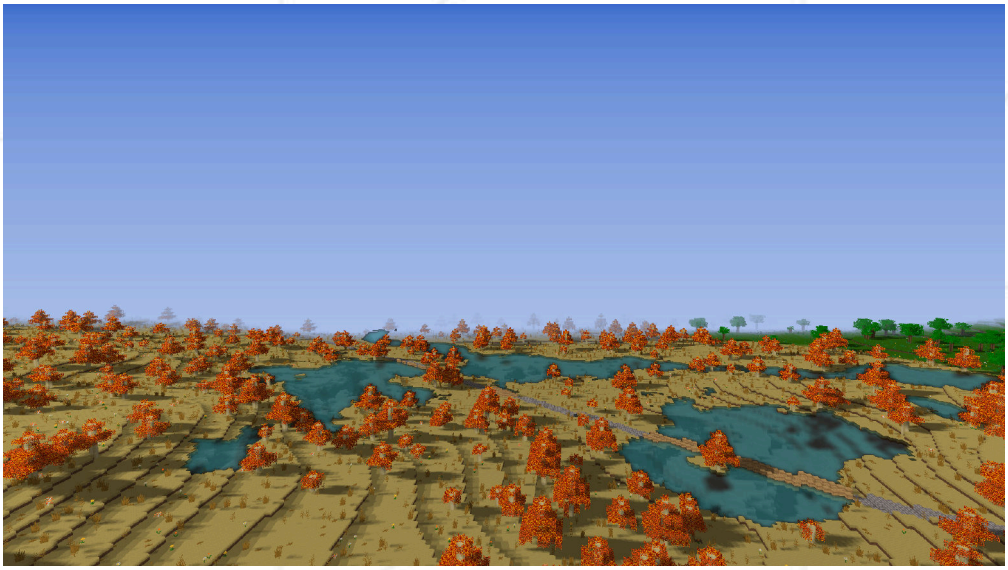
This is **wrong**

This is **right**

Small plants / flowers / mushrooms around the world.
Procedural trees *(trees/leaves must be blocks like others. You can't just load some 3d .obj and change scale. They must be procedural around multiple parameters (shape, height, width, leaf density, etc..)).*

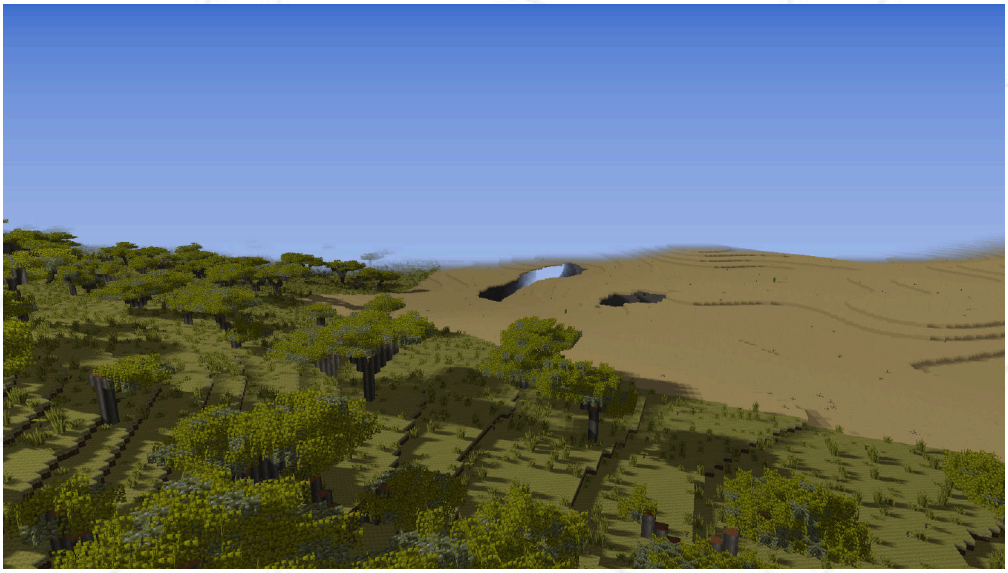
Here are Jungle trees next to Hill trees. Both are different in shape, but each individual tree is also unique.

There must be sea and rivers running around the world.



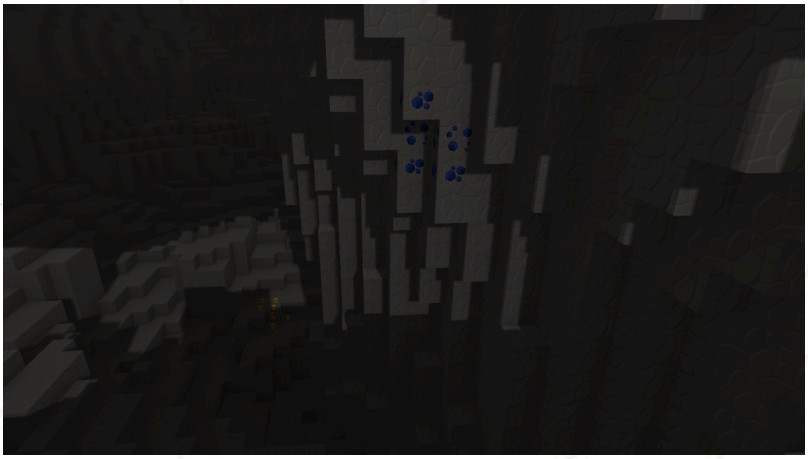
Lake and river, with a road/bridge passing through *(road/bridge isn't required)*

Cave with natural entry formation, accessible from outside. A player should be able to see them and access them without digging.



natural cave entrance

Realistic caverns (wormhole), not just a simple 3d noise.
The cavern must be filled with some chunks of rare minerals, such as gold, diamond..



This must be chunks of minerals as in minecraft. Not a simple probability on each block.

Monsters (creepers / zombies) that chase you when you get close.

3d clouds around the world, it can either be some type of block with no interaction possible (only visual) or some shader.

You can destroy blocks that you can collect (as in minecraft) and place them wherever you want.
Destroyed/placed blocks must be persistent, meaning even if you unload the chunk, and reload it after, the modifications should still be there.

V.2 Graphic rendering

What would a voxel world be without any cubes displayed ?
You will need to implement everything that was asked in ft_vox, but better.
Differences from ft_vox :

- The minimal render distance increase from 160 to 260
- You can use a skyshader instead of a skybox, if you want

But now you're getting good, to challenge you, we ask you to implement the following :

- Directional lightning
- Shadows
- SSAO (ambiant occlusion)
- Transparent water
- Far distance fog

`Be careful, your FPS should never get under 30.`

If you want your render to be smooth, you should manage the workload so it is equally shared between the CPU and GPU.

V.3 Camera

If we can move, that would be cool.
The keyboard should allow you to move forward, backward, strafe right, left, according to the camera.
You wil also need a key to jump and run.
The mouse should allow you to turn 360 degrees on the Y axis, and look up and down.
The camera speed should be set to around 1 cube per second when walking, and 2 cubes per second when running.
You can press a button to toggle fly-mode, and your "running" speed should be x20.

V.4 Sounds

There must be multiple ambiant musics. Each biome must have his own theme, and the transition between themes must be smooth. The music should not change brutally from one theme to another.
There also need to add overall sounds for walking, attacking, jumping, swimming,.. not only from the player, but also other player and monster around. That also means the sounds volume should be reduced according to the distance from the player.

V.5 Multiplayer and server

You don't want to feel lonely in such a big world, you'll need friends, that's why you will need to make your world multiplayer.
At least 4 players can join your server.
You should be able to see them in the world, and doing pretty much any actions (animation, destroying blocks, and dying from some weird monster).
All modifications of the world (destroying / adding blocks) must be synced between all players, and saved.

You're free to handle the server-side as you want, meaning you can procedurally generate the world on the server then dispatch it to the clients, or each clients will generate the world and sync it with the other players modifications.

Be careful, each methods has it's own pros and cons, think about it carefully

V.6 Interface

Because using command line arguments is so 2000s, you will at least need a basic interface.
At the start of the game, an interface to join a server must be displayed.
A menu to see connected players and be able to be teleported to them.
You're free to display the inventory as a quick action bar at the bottom of the screen or a full fledged interface.
FPS, triangles, cubes and chunks count must be displayed on-screen with a key toggle.

V.7 Other

Because we're getting closer to a real game, you will also need some basic qefqef :
A simple gravity and collision through the blocks (obviously, the water should not be collidable).
Be able to swim and dive, you're free to decide if you want the player to slow down his velocity when he's in the water.
Adapt your visual rendering when you are under water (color filter and/or reduced visibility).
Very simple animation (minecraft-like, as simple as what's asked in humangl) for walking / attacking / jumping.

Chapter VI

Bonus part

Possibilities are huge, here are some examples :

- Procedurally generated villages
- Crafting system
- Minecraft water (water that try to fall and spread when possible)
- Growing plants (from seed)
- Bow and arrow similar to minecraft
- Nether portal that actually teleport you in the nether
- Cross-Platform (Mac, Windows and Linux must be supported)
- Stereo sounds
- An online interface to navigate the world map (like Minecraft addon Dynmap)

Chapter VII

Turn-in and peer-evaluation

As usual, turn in your work on your repo GiT. Only the work included on your repo will be reviewed during the evaluation.