

# Portfolio

Sam Sully

[Website](#) | [GitHub](#) | [CV](#)

# Space Balloons

Build hardware and software for high altitude ballooning hobby.

Uses Raspberry Pi, LoRa radio, RTTY radio, high altitude GPS and Pi Camera.

Sends location and images via radio to receiver on chase car to facilitate retrieval.

Requires extreme fault tolerance and redundancy to cope with interruptions in radio contact.

Links: [code](#), [older code](#).



Payload to be attached to balloon (top left, bottom right), picture taken by camera at 33km altitude (top right), picture from a launch (bottom left).



# Cloud Software

I run a self-hosted microservice-based cloud, communicating using gRPC across services, all built in Golang.

Fully monitored using prometheus/grafana, automatic configuration of prometheus targets.

Uses code-generation for creation of new services quickly and to reduce repeat work.

Responsible for running my website and a number of personal tools (e.g. cctv service, calendar synchronisation service, home automation).

# Voxel Populi

Decentralised voxel-based online game engine.

Peer-to-peer architecture based on a DHT ([Kademlia](#)), extremely fault tolerant and scalable.

Designed to spread game computation across many servers and transparently transition between servers as the player moves around the world.

Links: [video demo](#), [code](#).

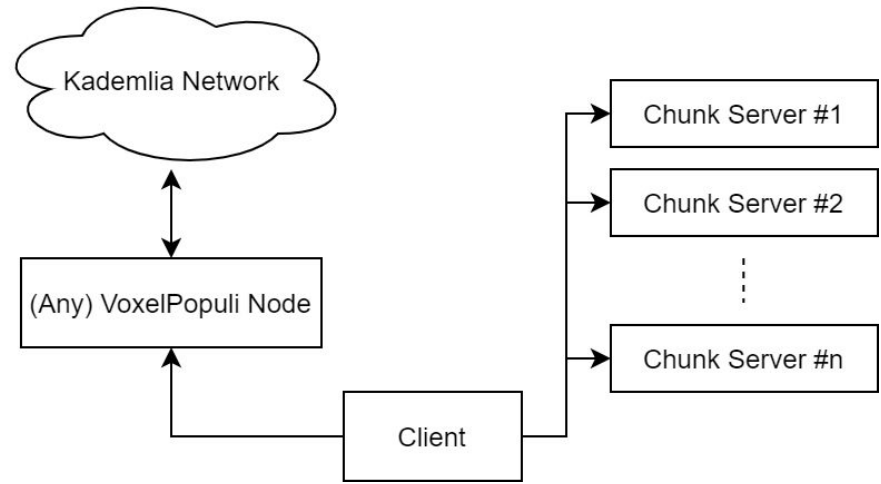
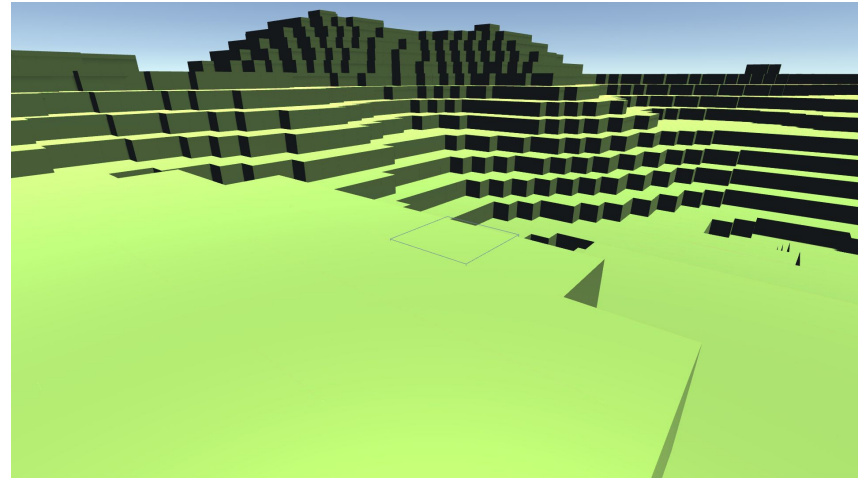


Diagram showing architecture of Voxel Populi (top), screenshot of Voxel Populi client displaying a world spread across 9 decentralised servers (bottom).



# Minecraft

Wrote several plugins for Minecraft servers to overhaul the game into an MMORPG and for administration of a large Minecraft server.

Designed custom spells, abilities, weapons, etc. and implemented their logic within the constraints of Minecraft server-side only modifications.

Scaled the modified game to hundreds of concurrent players on my public server.

Links: [code](#).



A screenshot of one of my custom spells (fireball) being used in the game, you can see on the right too that there is some information displayed about the player's MMO-style statistics and class.



# Game Dev

I enjoy hobby game development and have built a few games. My largest project is a multiplayer 2D survival game called Perilous.

This is a client-server multiplayer game with hand-drawn art implemented using Lua and a 2D game library called Love2D.

I wrote a number of tools to simplify my game development for me, this was because I identified a number of repeat tasks I kept doing.








A screenshot from Perilous.

# Game Dev Tools

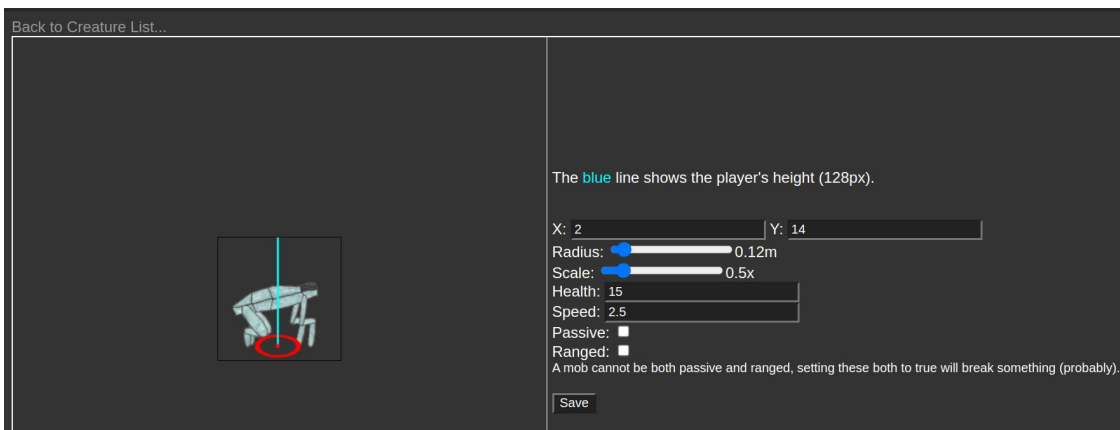
I found that I was performing certain workflows repeatedly when making games so I decided to automate them or make tools to simplify them.

I wrote a an editor for my game which allowed me to add content to my game without writing any code. This meant that new items and entities could be added to my game quickly without having to write any code. It also allowed me to configure things like collisions, textures & scaling, AI settings and item crafting recipes using a UI rather than having to write them out in code.

This massively sped up the game development process once the underlying game logic had been implemented.

	Jack's Hammer	Edit Add Component	3 x  (x), 2 x  (x),
	Ice	Edit Add Component	
	Berries	Edit Add Component	
	Stick	Edit Add Component	
	Pila	Edit Add Component	2 x  (x), 1 x  (x),
	Hazel Stick	Edit Add Component	
	Axe	Edit Add Component	4 x  (x), 1 x  (x),

Game item list (top), creature editor (bottom).

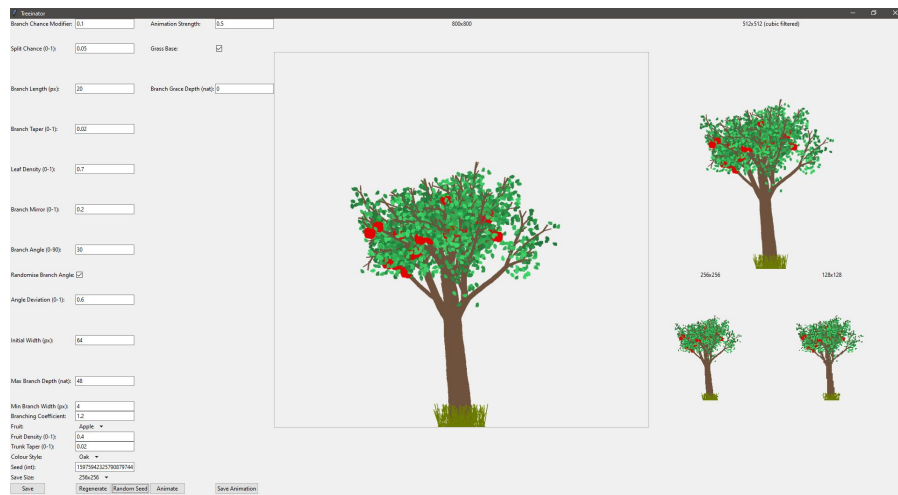


# Game Dev Tools

I also wrote some handy tools to help me with animations and textures. In one project I needed a lot of different tree textures so I wrote a piece of software to generate 2D trees procedurally. To do this I tried to emulate the natural growth process of trees to generate realistic looking trees.

In another project I wanted to create animations for a lot of different creatures and game objects quickly. To do this I decided I would write a piece of software to allow me to generate animations based on harmonic oscillations translating and rotating textures. This meant I had only to draw the texture once (in parts) and could then stitch these together and bring them to life in the animator.

Links: [code \(tree generator\)](#), [code \(animator\)](#).



Tree generator UI (top), harmonic animator part selection (bottom left), harmonic animator oscillation editor (bottom right).

**ANIMATE**



**SAVE**

**PLAY/PAUSE**

TIME	3
XOSC	0
XAMP	0.00
XPHA	0.00
YOSC	0
YAMP	0.00
YPHA	0.00
TOSC	2
TAMP	0.15
TPHA	0.00

SIZE: 256X256

