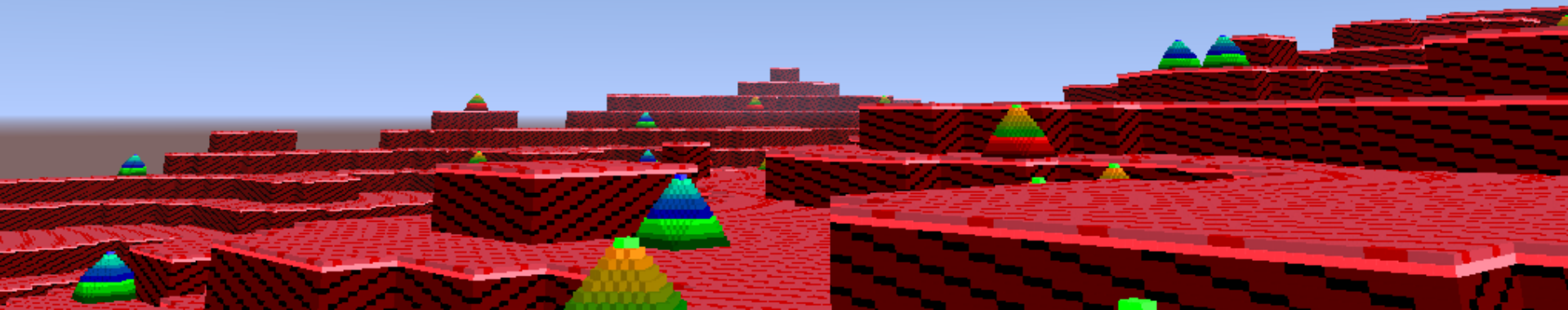


MP451 Project: Cubes

Kevin Reid
reid@clarkson.edu
Fall 2011



Project

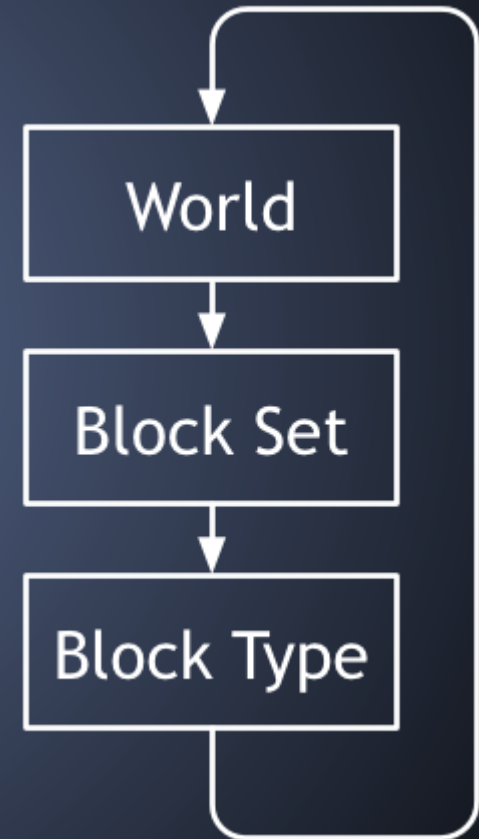
- **Inspiration:** “Minecraft, but with the blocks made out of blocks.”
- **Refinement:** Define the entire gameplay by assembling blocks (within the game).
Game = game engine?

Technical: Use WebGL, JS, HTML(5)

- OpenGL without porting
- Zero-installation game

Data Model

- **World**
Three-dimensional gridded space containing blocks which are from a block set.
- **Block**
Element of a world. Blocks, like real atoms, are nearly all absolutely identical to all others of the same block type.
- **Block type**
Visual and functional design for a block; defined by a miniature world.
- **Block set**
Collection of block types; what block types are available in the world. Defines the sort of game being played.



Rendering / Engine

- Starting point: learningwebgl.com; refactor to meet my needs.
- Packed data in JavaScript: **Uint8Array** and other WebGL typed arrays
- Abstracting procedural geometry with **RenderBundle**
- Shaders (are awesome)
- World chunks

User Interface Tech

- Mouselook in the browser: *no pointer capture!* Workaround: acceleration.
- Build HUD/menus in HTML — minimum complexity cost, or is it?
- A programmable game needs visibility of state
- HCI project reviewing block set interface

What I learned: Topics

- Voxel raycasting
- OpenGL ES 2: GLSL shaders, buffer objects, renderbuffers
- “HTML5”/New Browser APIs:
WebGL, Web Audio, localStorage, binary data
- Some audio synthesis
- Precise frustum culling
- CPU-limited environment:
JavaScript microoptimization

ONE MILLION VERTICES!

Future Work

- Reusing block sets across worlds
- Make circuits usable and capable
- Moving objects/vehicles/mobs
- Interesting procedural world generation
- Server-side save/load (“5MB is not enough”)
- Multiplayer