Software Design

This document outlines the software design portion of building the physicalized game interface.

Goals and Responsibilities

Goals

- Working, portable software
- User-accessible
  - No command line UI
  - Pre-compiled binaries
- Separate editor and display programs

Responsibilities

- Format user's input into apriltag(s)
  - Use UI to generate data
  - Process into data structures
  - Format data into apriltag
- Read in data from apriltags
  - Read in image from webcam
  - Detect where apriltag is in frame
  - Read apriltag data
  - Associate data with renderables
- Reproject output
  - Take in renderables
  - Render to a buffer/screen
  - Send image to projector to be displayed
- Calibrate display/camera
  - Provide calibration apriltag
  - Read in calibration apriltag
  - Project camera-perceived position in-frame
  - User manually adjusts projector

Editor Architecture

Model

Data Requirements:

- Active scene(s)
  - Scene ID
  - Scene graph
- User actions
  - Undo/redo stacks
- Scene Graph
  - Renderables
  - Groups
    - Canvas
      - Resolution
    - Transforms
- Transform
  - Position
  - Rotation
  - Scale
  - Renderable
- Renderable
  - Geometry
    - Stroke color?
    - Fill color?
  - Sprite
    - Animation?
  - Text
    - Typeface
    - Style
    - Weight
- Encoded scene

View

Requirements:

- Display scene graph
- Accept user input
  - On scene graph
  - Interface for saving/loading

Controller

Requirements:

- Pass scene graph data to view
- Process user input
  - Modify model
  - Encode model

Display Architecture

Model

- Active scenes
  - Scene ID
  - Scene data
- Active Tags
  - Location
  - Scene ID

View

- Draw scene data at relative location

Controller

- Read camera data
- Find apriltags
- Send locations and IDs to model
- Read active tag information from model