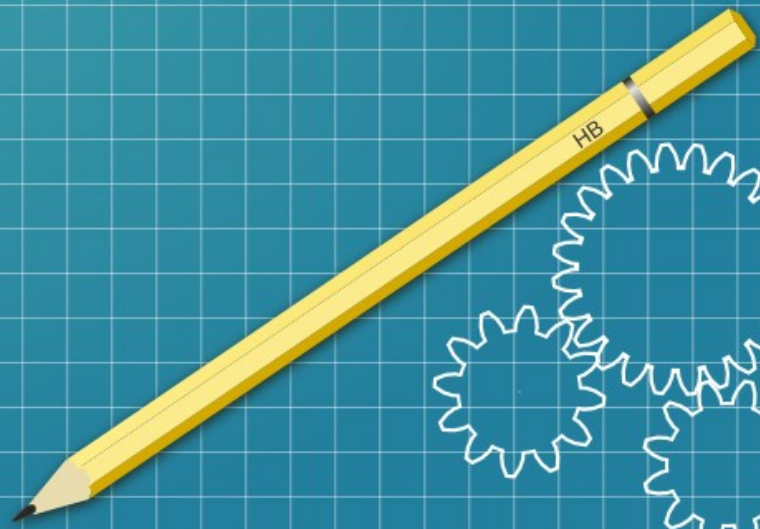
Creating a simple fitness game with MoveNet and TypeScript

By Klaus Kestel



Extra Materials

You’re welcome to follow along in the companion repo:
https://gitlab.com/lapiscode/fitness-game

Prerequisites

- Node 18 or later
- Yarn 1.22 or later
- Editor of you choice

Setting up the project

- We're using a Vite setup for sane defaults
- Quick start: yarn create vite fitness-game --template react-ts
- If you don't like React, you could also use the vue-ts, svelte-ts, or any other template from the official guide

Creating a camera mirror

- Use getUserMedia to capture the camera
- Set the resulting MediaStream as the srcObject of a hidden <video> element
- Use requestAnimationFrame and CanvasRenderingContext2D to draw the flipped video onto the canvas
- Question: Why not just directly show the video and flip it using a CSS transform?

Pulling in TensorFlow to analyze the video

- We're using the the MoveNet wrapper of the TensorFlowJS project
- Make sure you have all peer dependencies installed and import tfjs-core and a rendering backend as specified in the documentation
- Create a detector and throw the HTMLVideoElement at it using estimatePoses()

Adding sprites to the renderer

- Let's create a Sprite Type and some simple CRUD methods for the game elements
- I just used a Map to manage elements that are drawn on every render (see snippet)

Using MoveNet's keypoints to move the sprites

- The wrapper makes use of the so called COCO Keypoints
- We need some naming convention to manage the various sprites associated to the keypoints
- If you use multipose, ensure the number of sprites stays consistent
- Note: Pay attention to the dimensions of the canvas and the video!

Adding some assets to make it look good

- Using some hidden HTML elements does the job

Questions?