SmartPeerJS: A library for turning smartphone in interactive controller

Emma Poliakova - 2389098p

December 17, 2021

1 Status report

1.1 Proposal

This project builds on my summer project: WebRTCSmartPhoneController and smartpeer. This was focused on creating demos to illustrate the idea of a website controlled by smartphone. The goal is now to create an open-source library that packages the code needed to establish the peer-to-peer communication between the computer and phone websites and simplify the process of creating new types of smartphone controllers.

1.1.1 Motivation

Virtually everyone has a smartphone nowadays. Harnessing the large array of sensors smartphones offer to create controllers avoids the need of any expensive equipment or software installations. The idea is to turn your phone into a controller such as joystick, nes controller or hand tracker among others, by scanning just a single qr code.

1.1.2 Aims

The aim of the project is to have a library that requires only a few lines of code to support a peer-to-peer connection and data management between a phone controller and a PC browser. The process of controller creation should be easy and standardized. Working directly with users of the library to learn what functionality should be supported.

1.2 Progress

- Posted a survey to figure out a library name and keywords that would make searching for it easier.
- Created a Github organization to store and manage the code.
- Made a logo for the organization.
- Decided on the code structure, how will different classes complement each other.
- Created base classes for browser, controller and phone.
- Host and maintain three specific controllers Joystick, Touchpad and Nes Controller to be used on the phone.

- Made simple demos to demonstrate the use of the library.
- Written basic documentation.
- Published a version to npm compatible with different ways of importing.

1.3 Problems and risks

1.3.1 Problems

- The main problem was finding a way to export the library to npm and support different kinds of imports.
- Some of the code turned out to be redundant and needed to be refactored.
- Efficient way of bundling files with webpack.

1.3.2 Risks

- Cannot prevent multiple players to scan the same QRcode and have the same player ID. Mitigation: Provide a way to decide which player should keep the player id.
- Two peers cannot have the same peer ID. This could happen if a user decides to set the peer IDs manually. Mitigation: Make sure this is clearly explained in the documentation.

1.4 Plan

- Week 1-2: Create a support for player IDs.
- Week 3: Implement stats and update frequency support.
- Week 4-5: Finish writing documentation.
- Week 6-7: Upgrade the existing demos to show different functionalities of the library, easy to follow and learn from.
- Week 8-10: Finishing up dissertation.