App Uploader for Remote Embedded Devices

Craig Murray

2473420m

Proposal

Motivation

Currently, programs created for espruino devices can be uploaded either through the espruino web IDE, or through an app loader built into specific devices. Both of these methods work well enough, however neither are as straightforward as they could be. Uploading code through the web IDE requires users to write/download code, go to an external website, copy the code over, then upload it. The app loader only works with espruino smartwatch devices, and apps must be submitted and accepted onto the service. This project aims to create a new method of uploading code – one which would allow users/developers to upload code stored at a remote source (e.g. a GitHub repo) onto any espruino device, directly from any within any webpage.

Aims

The project aims to create an API that developers can implement to easily upload code to an espruino devices directly from a webpage, without the user having to visit an external service.

Progress

- Setup environment and tooling required for project
- Researched into API design best practices
- Researched and decided upon how to develop/distribute the uploader (As an NPM package and as a javascript API)
- Uploader currently takes in a URL containing source code for the devices, and uploads that code to the device
- Created various demos to test the functionality of the project on different espruino devices
- Collated ideas for “proof of concept” projects that can show the benefits/usage of the remote uploader

Problems and risks

Problems

Initially I had misinterpreted the goals of the brief; My original idea was that I would need to create a low-level system with which code could be written to the espruino hardware. However, after more research and discussion with my supervisor, I was able to clarify what exactly was required, and learned that there exists various APIs that can handle the lower-level side of development for me. These APIs have made both understanding the project, and implementing the uploader much easier. In terms of development, I have encountered a few roadblocks. As of writing this update, I am currently attempting to implement functionality allowing the uploader to confirm that code has been successfully uploaded so that control can be passed over to the user. My current solution is to upload a simple function onto the device, which can return data back to the uploader, however this has proven to be more challenging than expected and debugging errors has been difficult.

Risks

It is possible that there may not exist an intuitive method of notifying the user that code has been successfully uploaded. I am actively researching this problem however and hoping to find a logical solution. It would also be nice to add options for how the user wants to save code onto the device, for example saving to the RAM or onto flash memory; however from a user perspective, this may not be viable – it is likely that a user may specify in their code that they want certain functions/variables to be saved onto flash memory, whilst saving other parts to RAM, which could get complicated and introduce bugs if the uploader allows for such options separately.

Plan

- I hope to have all core functionality implemented and the uploader published before the next supervisor meeting.
- The rest of January would be spent on “proof of concept” demos demonstrating potential use cases of the uploader. Current idea is a “smart hub” using a pixl.js as a control panel to monitor and control smart devices in the home. Code for this would be written, stored on a GitHub repo, and then anyone can upload the code to their espruino device using the app uploader I developed. A demo like this would also allow me to identify and implement extra features into the uploader that users may find useful.
- February would be spent gathering user feedback, and writing up a first draft of the dissertation
- During March I aim to revise and finalize my dissertation writing, and implement any features/changes that may become apparent as a result of user feedback