Robot Programming App

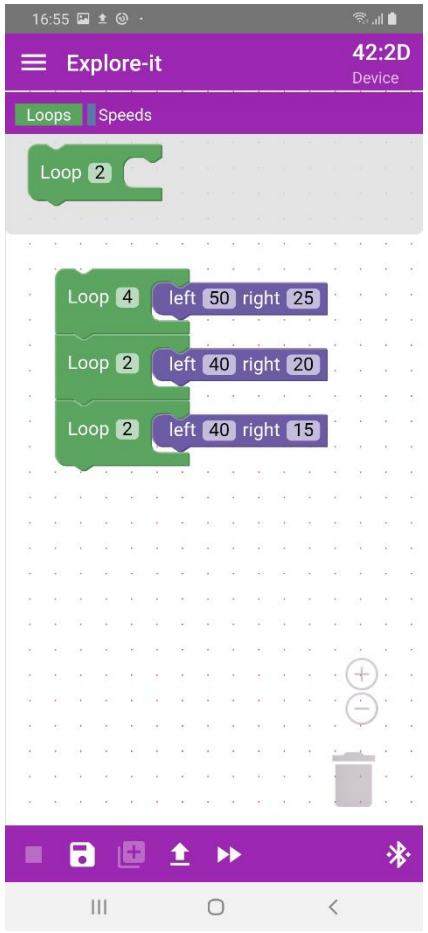
React Native

IP5 Blockly Based Approach

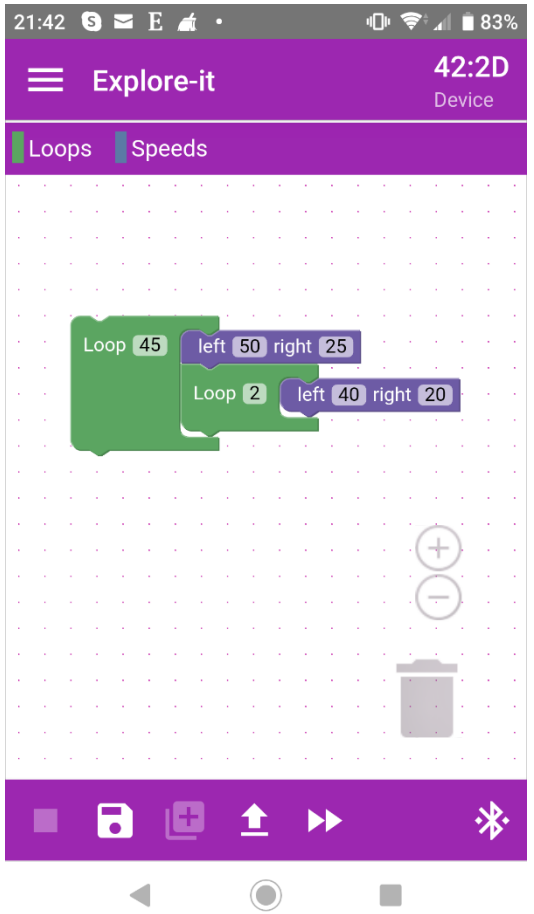
Borbely Sabina

Summary

- Demo
- Blockly
- Blockly and React Native Integration.
- Implementation
- Overview



Samsung Galaxy A70, OS: Android 9.0



Sony Xperia XA2, OS: Android 9.0

Blockly Advantages & Disadvantages

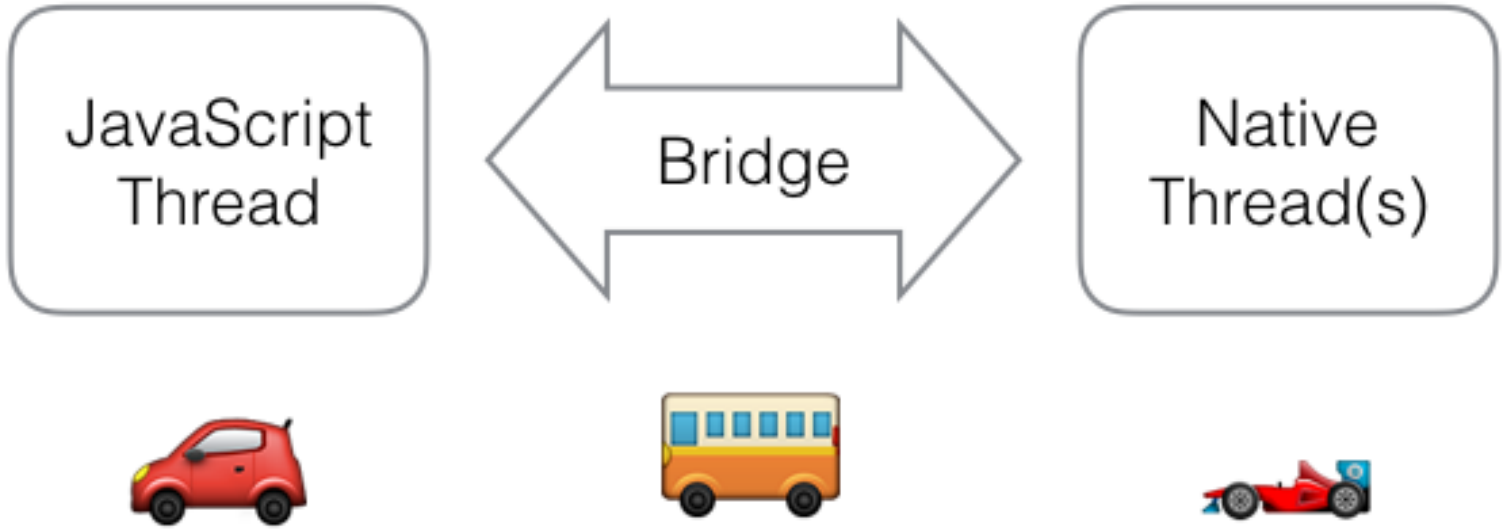
- Put together colourful blocks to build a program
- intuitive user interface
- simplicity
- input of a large number of similar steps
- loops and speeds

- Blockly integration – more difficult

Performance

The performance of a React Native App

- **JavaScript optimization**
- **Native components**
- **Optimizing the performance of custom UI components**



Blockly Blocks

- Blocks - localized to the user's language

Blockly

Block Definition

Multilanguage text

Block Javascript code

Toolbox definition

Blockly Interpreter

Device/Board functions

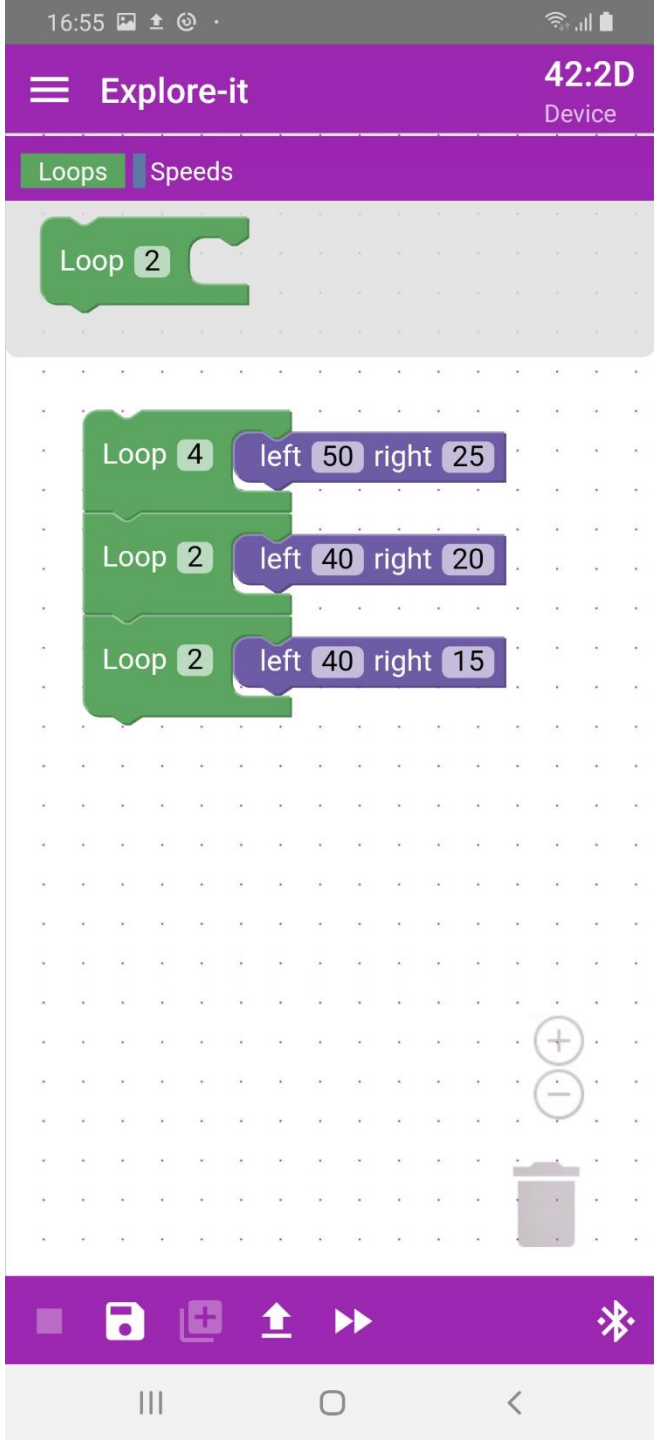
Blockly
Custom Blocks

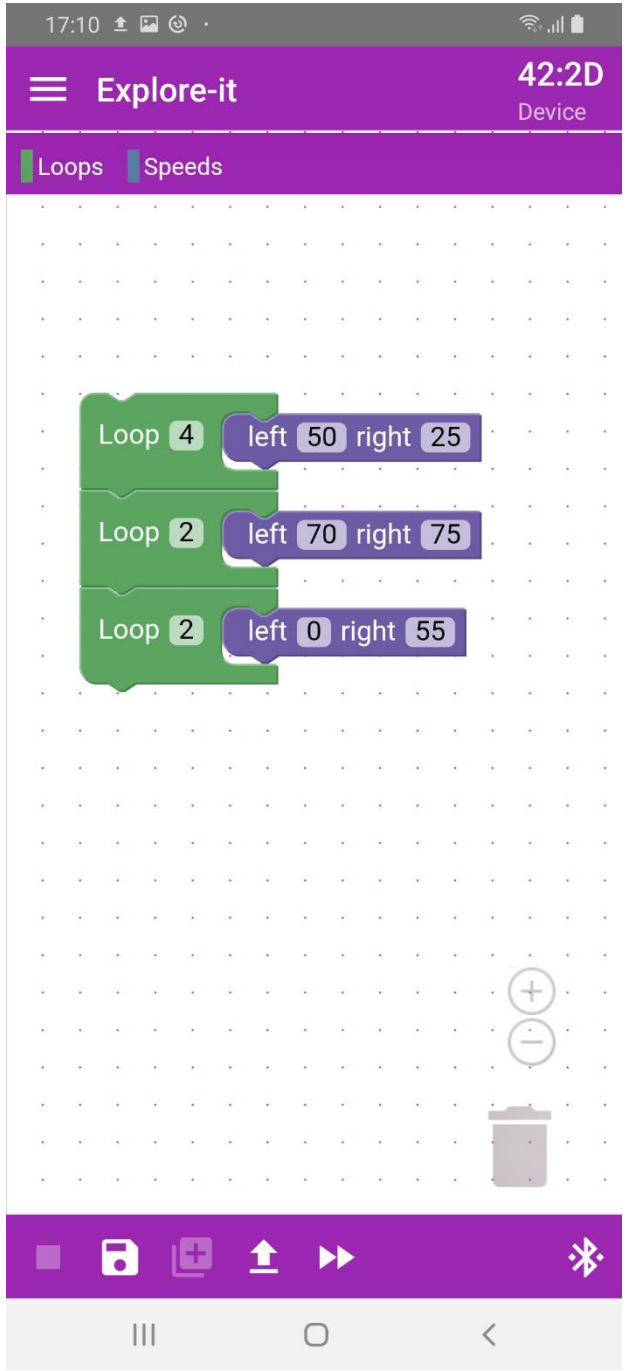
Loops

- Repeat x times

Speeds

- Left & Right

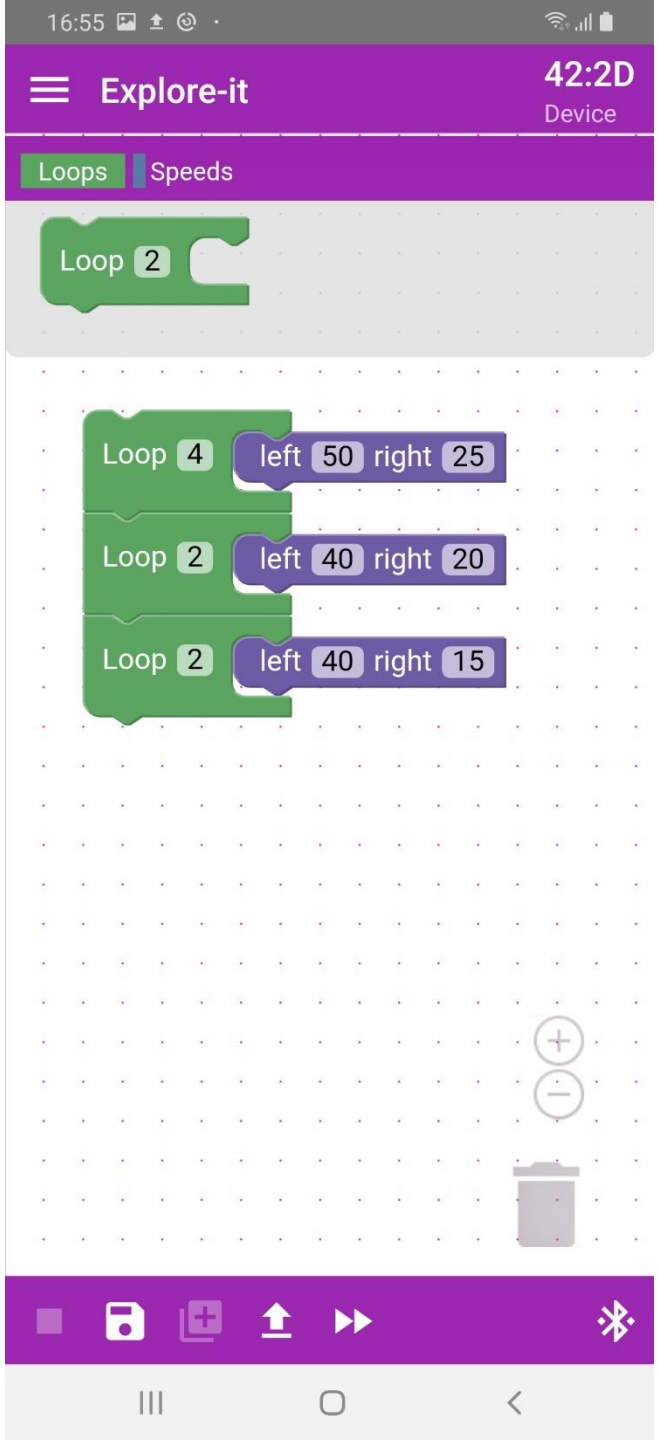




Blockly Javascript Interpreter

Generated Code as per the custom block definitions

```
for (let count = 0; count < 4; count++) {
      addStep({left:50, right:25});
}
for (let count = 0; count < 2; count++) {
      addStep({left:70, right:75});
}
for (let count = 0; count < 2; count++) {
      addStep({left:0, right:55});
}
```



React Native – Blockly Communication

- **onSave**
- **BlockPr-->BlockComp-->BlocklyWebview**
- the program is considered done. The generated code -final
- send request to the web app for the generated code (refs and RN WebView injectJavascript property)
- Receive (onMessage) and eval the code to construct an array of steps in React Native

```
onMessage={event => {
  const { data } = event.nativeEvent;
  { receiveCodeAsString(data) };
```

- Save the current speeds in the BlockComponent (speeds to be uploaded)

Blockly– RN Communication postMessage()

```
function sendGeneratedCodetoRN() {
  let res_code = Blockly.JavaScript.workspaceToCode(workspacePlayground);
  console.log(res_code);
  window.ReactNativeWebView.postMessage(res_code);
}
```

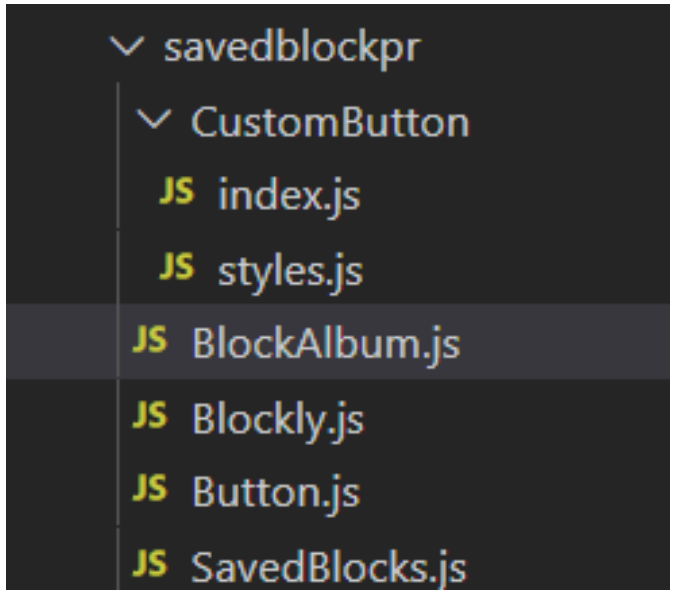
- Receive (onMessage) and eval the code to construct an array of steps in React Native

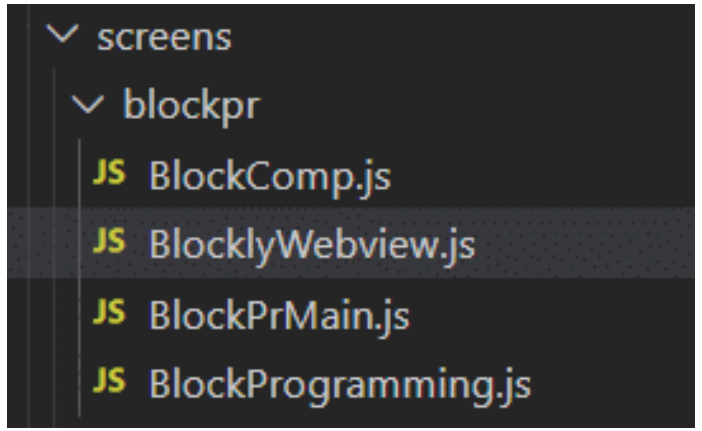
```
onMessage={event => {
  const { data } = event.nativeEvent;
  { receiveCodeAsString(data) };
```

React Native – Blockly Communication

- React Native -> Web: injectedJavaScriptBeforeContentLoaded
- 1. React Native -> Web: The injectedJavaScript prop
- 2. React Native -> Web: The injectJavaScript method
- 3. Web -> React Native: The postMessage method and onMessage prop
- Receive (onMessage) and eval the code to construct an array of steps in React Native

```
onMessage={event => {
  const { data } = event.nativeEvent;
  { receiveCodeAsString(data) };
```

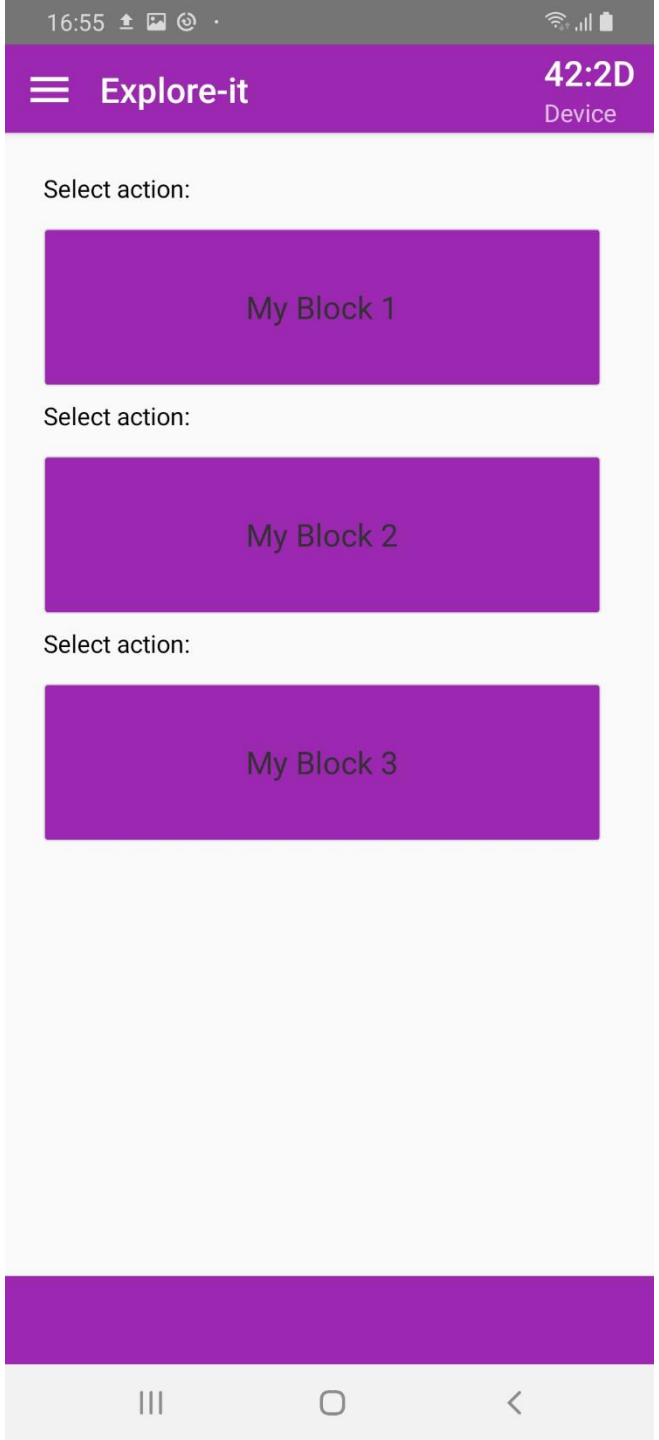




Block Programming

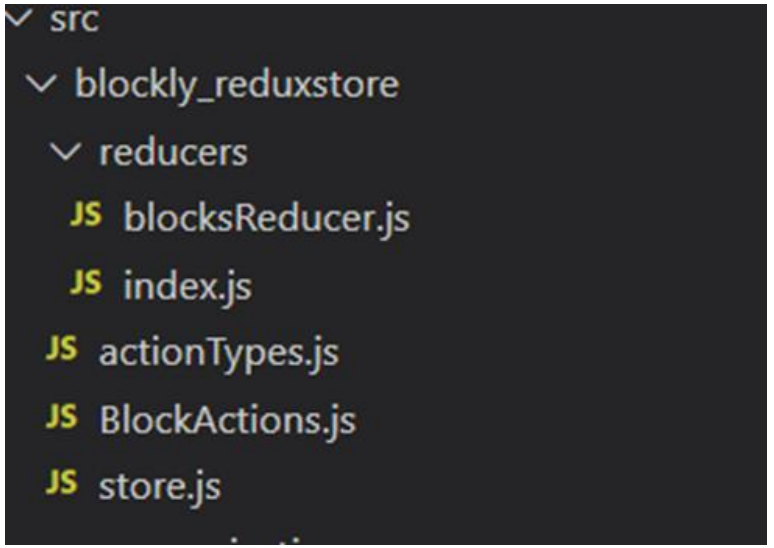
- **BlockProgramming-->BlockComp->BlocklyWebview**
- **BlockProgramming → Redux Store**
- **BlockAlbum → Redux Store**
- **Props:**
  - addBlock
  - removeBlock(block_name)
  - updateBlock
  - getBlock(block_name)

  - blocks=*state.blocksReducer*



Saved Blocks

- Menu Options-Select action
- 1. delete a block
- 2. open Blockly- in a pop-up or modal
- The list is updated based on the blocks from redux store
- New blocks are automatically named My Block 1, My Block 2 …




```
<Provider store={store}>
    <PersistGate loading={<LoadingIndicatorView />} persistor={persistor}>
        <BlockAlbum />
    </PersistGate>
</Provider>
```

```
    <Provider store={store}>
      <PersistGate loading={null} persistor={persistor}>
        <BlockProgramming />      You, 3 minutes ago • Unco
      </PersistGate>
    </Provider>
```

Redux Persist

Folder Structure
BlockReducer

- Redux Persist
- BlockPrMain- Container of BlockProgramming

Overview

Blockly Based Approach –with React Native Community WebView

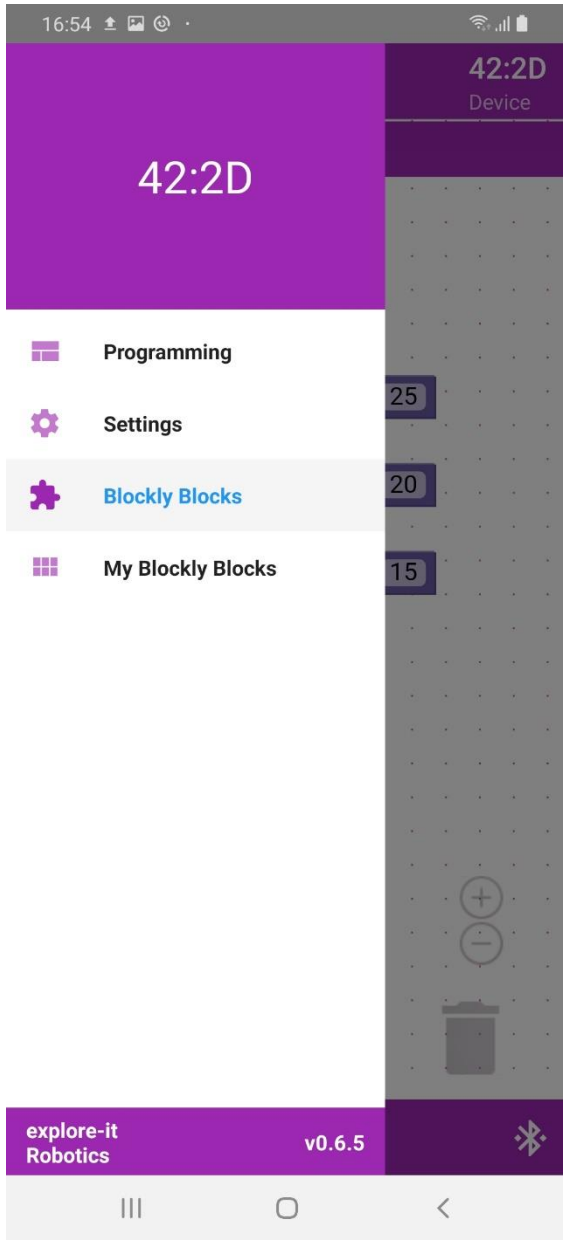
- Web App and React Native

Possible improvements

- Custom block names
- German text
- Using a database for saving blocks

Another possible implementation approach to try:

- npm Blockly package https://www.npmjs.com/package/blockly
- autogenerate the types declarations for Blockly with https://github.com/trodi/blockly-d.ts
- TypeScript and React Native



Questions. Robot Programming App