Rummikub

Web-Technologien

Kira Koch Julian Riegraf

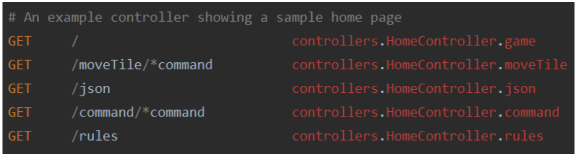
HTWG Konstanz

January 22, 2020

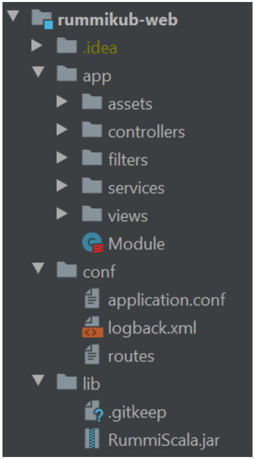
Play Server

Task 01: Create a Play project

Play is a web application framework, written in Scala and for Scala projects.



```
# An example controller showing a sample home page
GET     /                       controllers.HomeController.game
GET     /moveTile/*command      controllers.HomeController.moveTile
GET     /json                   controllers.HomeController.json
GET     /command/*command       controllers.HomeController.command
GET     /rules                  controllers.HomeController.rules
```

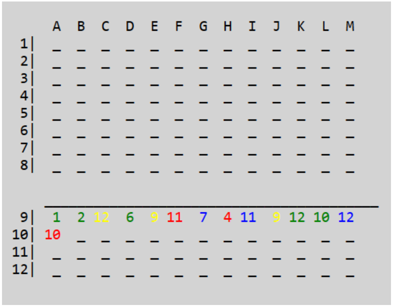


```
▼ rummikub-web
  ▶ .idea
  ▼ app
    ▶ assets
    ▶ controllers
    ▶ filters
    ▶ services
    ▶ views
      Module
  ▼ conf
      application.conf
      logback.xml
      routes
  ▼ lib
      .gitkeep
      RummiScala.jar
```

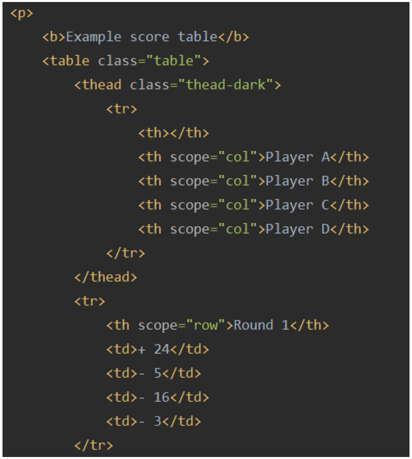
HTML

Task 02: Create HTML Pages

HTML is a Hyper Text Markup Language for content on the web.



```
    A  B  C  D  E  F  G  H  I  J  K  L  M
 1| _  _  _  _  _  _  _  _  _  _  _  _  _
 2| _  _  _  _  _  _  _  _  _  _  _  _  _
 3| _  _  _  _  _  _  _  _  _  _  _  _  _
 4| _  _  _  _  _  _  _  _  _  _  _  _  _
 5| _  _  _  _  _  _  _  _  _  _  _  _  _
 6| _  _  _  _  _  _  _  _  _  _  _  _  _
 7| _  _  _  _  _  _  _  _  _  _  _  _  _
 8| _  _  _  _  _  _  _  _  _  _  _  _  _

   _______________________________________
 9| 1  2 12  6  9 11  7  4 11  9 12 10 12
10| 10 _  _  _  _  _  _  _  _  _  _  _  _
11| _  _  _  _  _  _  _  _  _  _  _  _  _
12| _  _  _  _  _  _  _  _  _  _  _  _  _
```

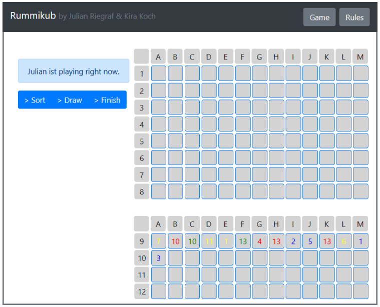


```
<p>
    <b>Example score table</b>
    <table class="table">
        <thead class="thead-dark">
            <tr>
                <th></th>
                <th scope="col">Player A</th>
                <th scope="col">Player B</th>
                <th scope="col">Player C</th>
                <th scope="col">Player D</th>
            </tr>
        </thead>
        <tr>
            <th scope="row">Round 1</th>
            <td>+ 24</td>
            <td>- 5</td>
            <td>- 16</td>
            <td>- 3</td>
        </tr>
```

LESS

Task 03: Add CSS to your Pages

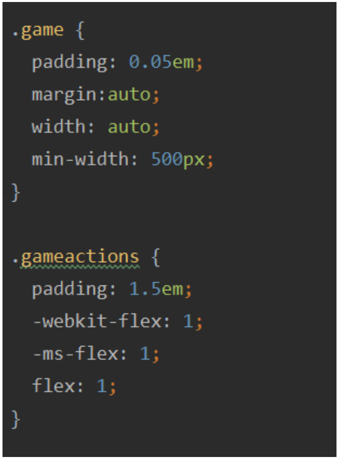
LESS: It's CSS, with just a little more.



Julian ist playing right now.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2													
3													
4													
5													
6													
7													
8													

	A	B	C	D	E	F	G	H	I	J	K	L	M
9	7	10	10	11	1	13	4	13	2	5	13	6	1
10	3												
11													
12													

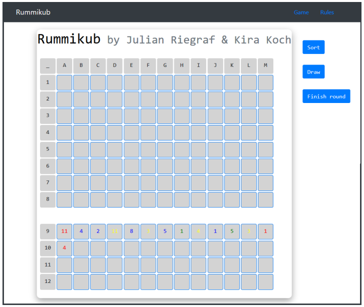


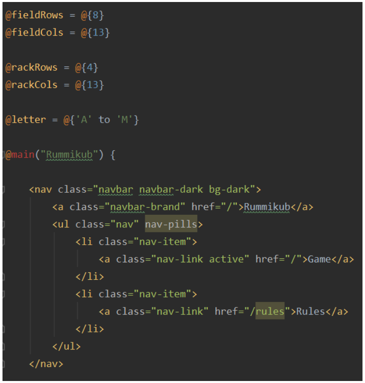
```
.game {
  padding: 0.05em;
  margin:auto;
  width: auto;
  min-width: 500px;
}

.gameactions {
  padding: 1.5em;
  -webkit-flex: 1;
  -ms-flex: 1;
  flex: 1;
}
```

Bootstrap

Task 04: Add Controls to your game using Bootstrap

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS.



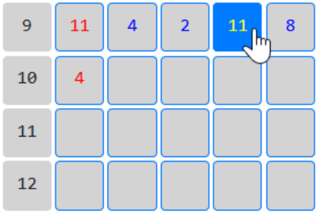


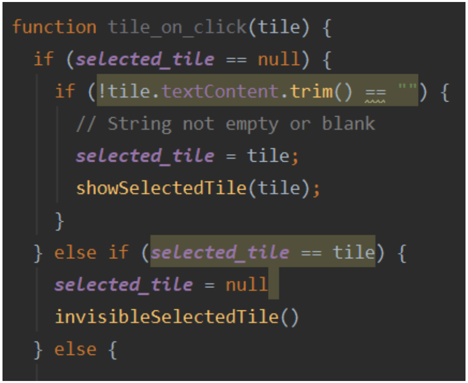
```
@fieldRows = @{8}
@fieldCols = @{13}

@rackRows = @{4}
@rackCols = @{13}

@letter = @{'A' to 'M'}

@main("Rummikub") {

    <nav class="navbar navbar-dark bg-dark">
        <a class="navbar-brand" href="/">Rummikub</a>
        <ul class="nav" nav-pills>
            <li class="nav-item">
                <a class="nav-link active" href="/">Game</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="/rules">Rules</a>
            </li>
        </ul>
    </nav>
```

JavaScript

Task 05: Add Interaction using JavaScript

JavaScript is a very widely used programming language.



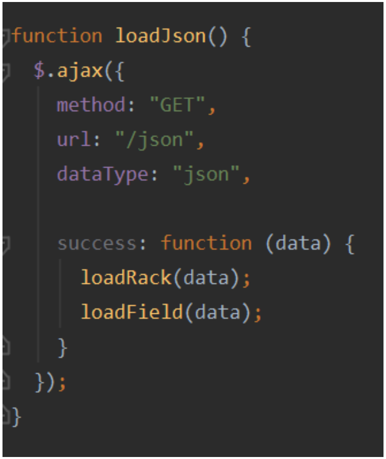


```
function tile_on_click(tile) {
  if (selected_tile == null) {
    if (!tile.textContent.trim() == "") {
      // String not empty or blank
      selected_tile = tile;
      showSelectedTile(tile);
    }
  } else if (selected_tile == tile) {
    selected_tile = null
    invisibleSelectedTile()
  } else {
```

jQuery and AJAX

Task 06: Use jQuery to react to events

jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for a rapid web development.

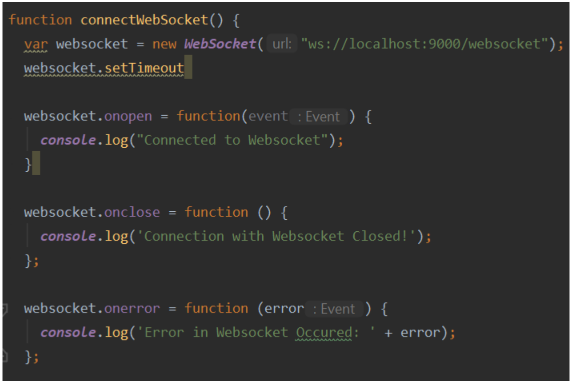


```
function loadJson() {
  $.ajax({
    method: "GET",
    url: "/json",
    dataType: "json",

    success: function (data) {
      loadRack(data);
      loadField(data);
    }
  });
}
```

WebSockets

Task 07: Build a ServerPush Mechanism

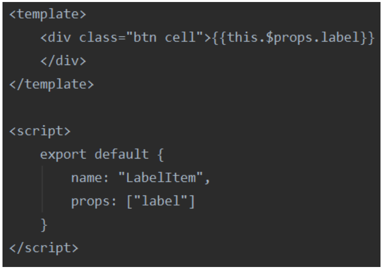
WebSocket is a protocol for full-duplex data transfer



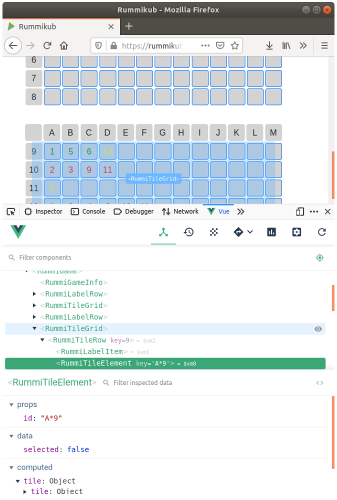
```
function connectWebSocket() {
  var websocket = new WebSocket( url: "ws://localhost:9000/websocket");
  websocket.setTimeout

  websocket.onopen = function(event : Event ) {
    console.log("Connected to Websocket");
  }

  websocket.onclose = function () {
    console.log('Connection with Websocket Closed!');
  };

  websocket.onerror = function (error : Event ) {
    console.log('Error in Websocket Occured: ' + error);
  };
```

Vue.js Components

Task 08: Use and Build Custom Web Components



```
<template>
    <div class="btn cell">{{this.$props.label}}
    </div>
</template>

<script>
    export default {
        name: "LabelItem",
        props: ["label"]
    }
</script>
```



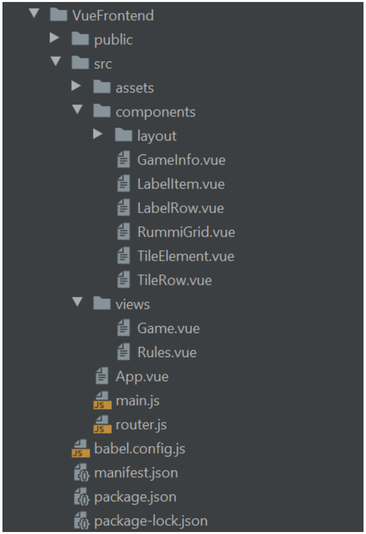
```
<RummiGameInfo>
▸ <RummiLabelRow>
▸ <RummiTileGrid>
▸ <RummiLabelRow>
▾ <RummiTileGrid>
  ▾ <RummiTileRow key=9> = $vm2
      <RummiLabelItem> = $vm1
      <RummiTileElement key='A*9'> = $vm0
```

```
<RummiTileElement>
▾ props
    id: "A*9"
▾ data
    selected: false
▾ computed
  ▾ tile: Object
    ▸ tile: Object
```

Vue.js

Task 09: Implementing a SPA using Vue

A SPA is a web application or web site that interacts with the user by dynamically rewriting the current page rather than loading entire new pages from a server.
State management with Vuex.

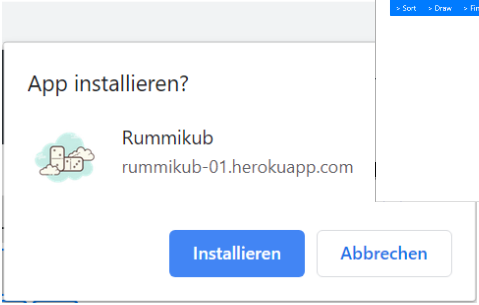


```
▼ VueFrontend
    ▶ public
    ▼ src
        ▶ assets
        ▼ components
            ▶ layout
              GameInfo.vue
              LabelItem.vue
              LabelRow.vue
              RummiGrid.vue
              TileElement.vue
              TileRow.vue
        ▼ views
              Game.vue
              Rules.vue
          App.vue
          main.js
          router.js
      babel.config.js
      manifest.json
      package.json
      package-lock.json
```

Progressive Web App

Task 10: Build a progressive Web App

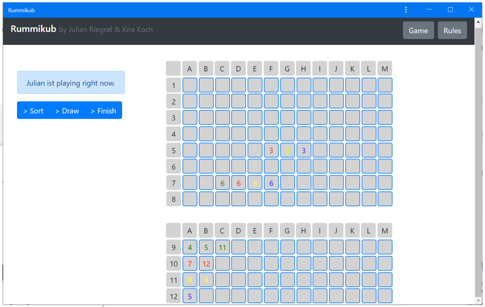
PWAs bring Web Technologies to mobile device apps.



App installieren?

Rummikub

rummikub-01.herokuapp.com



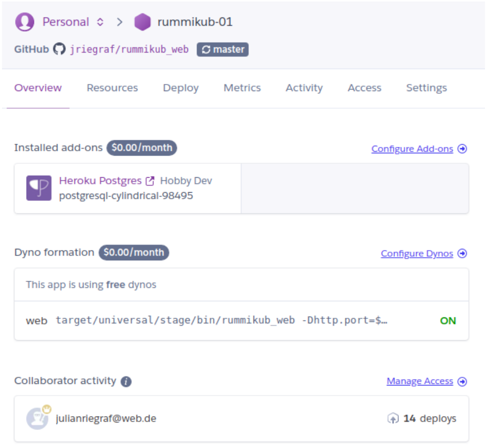
Rummikub by Julian Riegraf & Kira Koch

Julian ist playing right now.

Deployment - Heroku

Task 11: Deploy your game with Heroku

Web application run on a server, connected by HTTP


Personal > rummikub-01

GitHub jriegraf/rummikub_web master

Installed add-ons $0.00/month Configure Add-ons

Heroku Postgres Hobby Dev
postgresql-cylindrical-98495

Dyno formation $0.00/month Configure Dynos

This app is using **free** dynos

```
web  target/universal/stage/bin/rummikub_web -Dhttp.port=$…   ON
```

Collaborator activity Manage Access

julianriegraf@web.de **14** deploys