FS18

WEB ENGINEERING & DESIGN 3

Introduction

T. Bocek

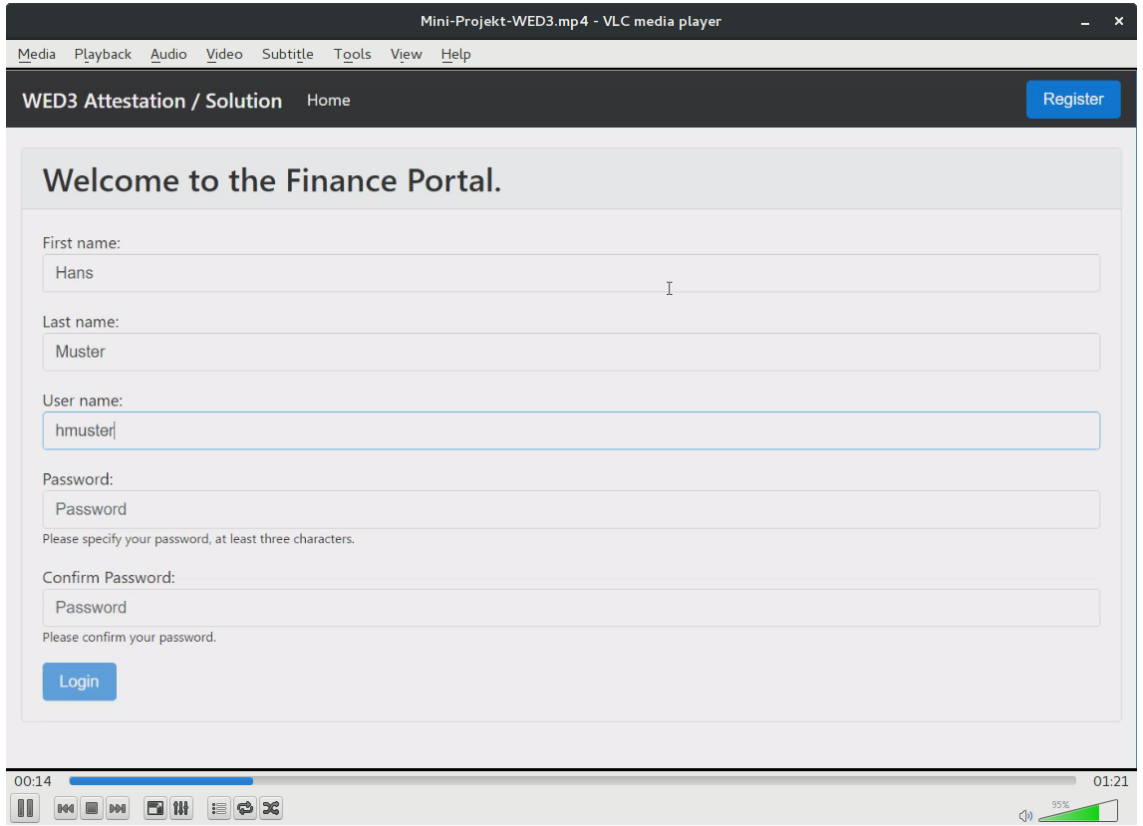
thomas.bocek@hsr.ch

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

WED3 Administration – Mini Project

- **Mini Project**
  - Description Angular
  - Description React
    - Video
  - Signup here
  - Demo here (user1/1234)



Single Page Application (SPA) - In the News

- **25.02.2018:** **Web application from scratch, Part I**
  - HTTP Server
    - Protocol, request
    - Simple example in Python
  - Serve files
  - Keep an eye on further parts
- **25.02.2018:** **This SVG always shows today's date**
  - SVG (also with scripts) will be discussed in an upcoming lecture
  - <svg onload="init(evt)“
- **25.02.2018:** **Webpack 4**

URL: b.socrative.com/student/

Room: HSR

Revisiting Exercises

- **Exercise 1 – Crypto Portfolio Manager**
- **Server had 3 endpoints**
  - GET /portfolio – returns an unused unique ID. Take care, this is not transactional!
  - Not needed, new example only 2 endpoints

```
@SpringBootApplication
@RestController
public class Application {
    final private static Map<String, String> portfolio = new ConcurrentHashMap<String,
String>();
    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }

    @GetMapping("/portfolio/{uuid}")
    public String get(@PathVariable("uuid") String uuid) {
        String retVal = portfolio.get(uuid);
        if(retVal == null) {
            return "[]";
        }
        return retVal;
    }
    @RequestMapping(value = "/portfolio/{uuid}", method = RequestMethod.POST)
    //@PostMapping("/portfolio/{uuid}")
    public void set(HttpEntity<String> httpEntity, @PathVariable("uuid") String uuid) {
        String json = httpEntity.getBody();
        portfolio.put(uuid, json);
    }
}
```

Revisiting Exercises

- **Exercise 1 – Crypto Portfolio Manager**
- **Frontend**
  - Added dropdown
  - Added total from vue-crypto
  - Added remove button
  - Added fetching rates in periodically
  - Dev Tools
- **Recap**
  - "…it adds all the properties found in its data object to Vue's reactivity system"
  - Two way data-binding, e.g., selected

```
data: {
        coinmarketcapData: [],
        userCoinPortfolio: [],
        uuid: '',
        selected: ''
}
```

```
<p v-for="item in userCoinPortfolio">
  <input v-model="item.symbol" placeholder="BTC">
  <input v-model="item.amount" placeholder="0">
  {{item.amount * find(item.symbol)}}
  <button @click="remove(item.symbol)">remove</button>
</p>
<p>Total: {{total | round}} USD</p>
<select v-model="selected">
  <option disabled value="">Please select one</option>
  <option v-for="item in coinmarketcapData" :value="item.symbol">
    {{ item.name }}
  </option>
</select>
<button @click="userCoinPortfolio.push({'symbol' : selected,
'amount': '0'})">add</button>
```

Revisiting Exercises

- **Exercise 1 – Crypto Portfolio Manager**
- **Two way data binding**
  - JSON.stringify(vm.$data.userCoinPortfolio)
  - `[{"symbol":"BTC","amount":"434342.3"},`
    `{"symbol":"ETH","amount":"434"},`
    `{"symbol":"XRP","amount":"43422"}]`
  - v-model binds input to elements in the array
  - Array operations, push/remove on userCoinPortfolio directly reflected in the UI

```
data: {
        coinmarketcapData: [],
        userCoinPortfolio: [],
        uuid: '',
        selected: ''
}
```

```
<p v-for="item in userCoinPortfolio">
  <input v-model="item.symbol" placeholder="BTC">
  <input v-model="item.amount" placeholder="0">
  {{item.amount * find(item.symbol)}}
  <button @click="remove(item.symbol)">remove</button>
</p>
<p>Total: {{total | round}} USD</p>
<select v-model="selected">
  <option disabled value="">Please select one</option>
  <option v-for="item in coinmarketcapData" :value="item.symbol">
    {{ item.name }}
  </option>
</select>
<button @click="userCoinPortfolio.push({'symbol' : selected,
'amount': '0'})">add</button>
```

Revisiting Exercises

- **Exercise 1 – Crypto Portfolio Manager**
- **Watch variables**
  - UUID: fetch data from backend. Needs to be done once UUID is set
  - userCoinPortfolio: (deep watch array) if something changes in the array, store it
    - Store every single character/number change - overhead

```
watch: {
  uuid: function(newUuid, oldUuid) {
    window.location.hash = '#' + newUuid
      fetch('/portfolio/'+this.uuid)
        .then(response => response.json())
        .then(body => {
          this.userCoinPortfolio = body;
        });
  },
  userCoinPortfolio: {
    handler: function(newUuid) {
      fetch('/portfolio/' + this.uuid, {
        method: 'POST',
        body: JSON.stringify(this.userCoinPortfolio)
      })
    }, deep: true
  }
}
```

URL: b.socrative.com/student/

Room: HSR

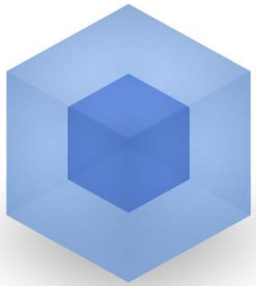
Vue.js Components

- **Components**
  - Reusable
  - Extend basic HTML elements
  - Ticker example
- **Single File Components**
  - Now: everything in one file
    - Can get messy
  - vue-for-idea

```
// register
Vue.component('my-component', {
  template: '<div>A custom component!</div>'
})

// create a root instance
new Vue({
  el: '#example'
})

// html file
<div id="example">
  <my-component></my-component>
</div>

//will produce
<div id="example">
  <div>A custom component!</div>
</div>
```

Vue.js Single File Components

- **ticker.vue**
- **How to create (split) html/js/css files?**
- **Bundler!**
  - In the JavaScript world: understand the concepts not the tools
  - E.g., Spesen Project as Master project started (task runner - minify, uglify, linting, compiling)
  - May 2015: Grunt for project (supervision, state-of-the-art)
  - Dec 2015: Gulp for pdfsign.js (done by me)
  - … Browserify (missed that one :)
  - June 2016: Webpack for modum.io (done by me + Vue.js)
  - January 2017: Webpack 2 for modum.io (supervision)
  - March 2018: Still Webpack 2? Rollup?



URL: b.socrative.com/student/

Room: HSR

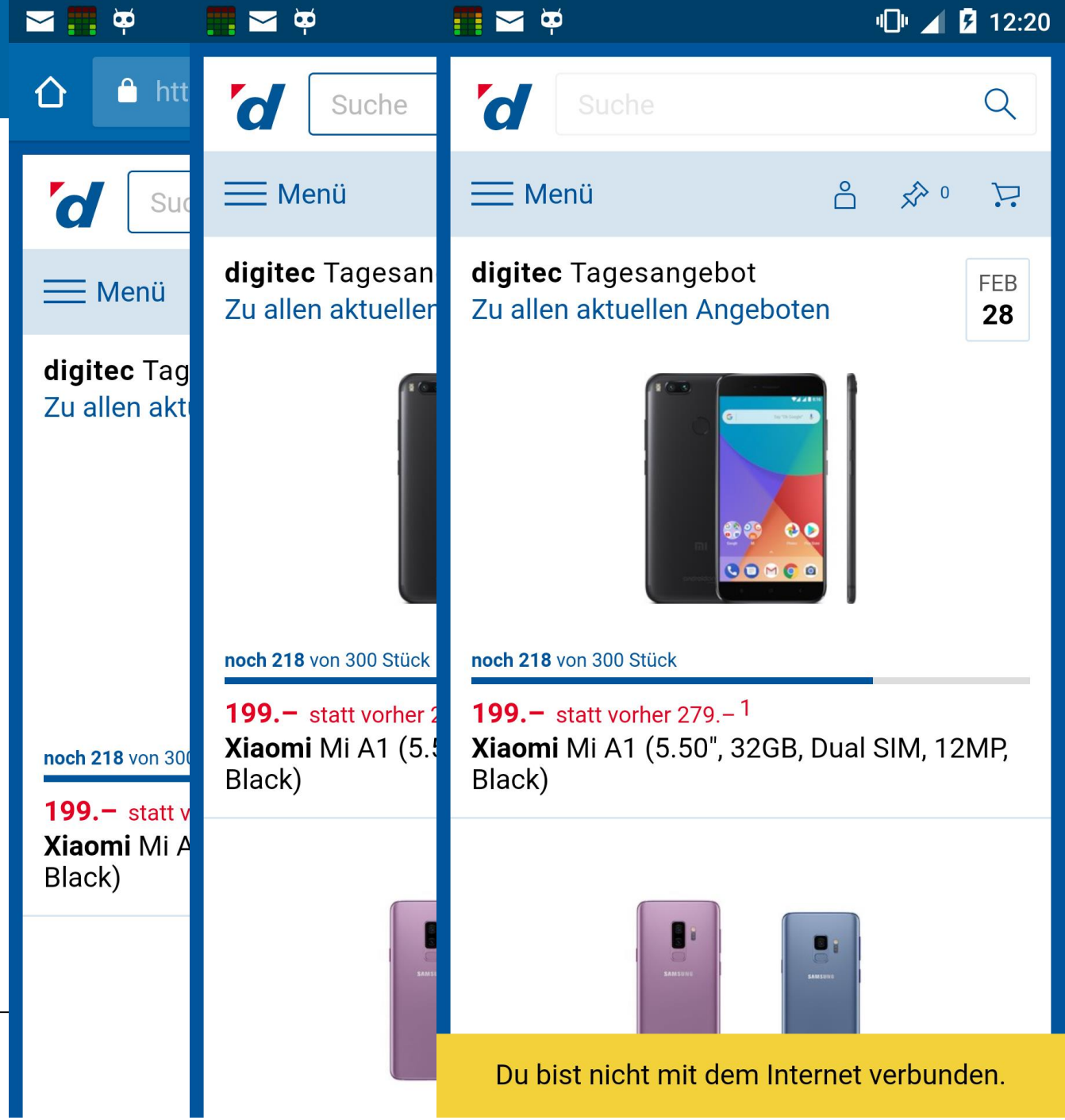
Webpack

- **Bundler**
  - “A JavaScript bundler is a tool that puts your code and all its dependencies together in one JavaScript file”
  - Dependencies (package.json) Install yarn (or npm)
    - yarn install
    - Dev dependencies: css-loader, vue-loader, vue-template-compiler, webpack,webpack-cl, webpack-dev-server
  - Webpack (webpack.config.js)
    - Entry: which files to start the bundling process
    - Output: where (and how) the generated bundle are stored
    - Modules, rules, loaders: packages that transform files matching a pattern (plugins)
    - The resolve object configures how module resolution works (ES2017).

```
const UglifyJSPlugin = require('uglifyjs-webpack-plugin');

module.exports = {
    entry: {
        app: __dirname + '/src/main/app.js',
    },
    output: {
        path: __dirname + '/src/main/resources/static/dist',
        publicPath: "/dist/",
        filename: 'bundle.js',
    },
    module: {
        loaders: [{
            test: /\.vue$/,
            loader: 'vue-loader',
        },]
    },
    resolve: {
        alias: {
            'vue$': 'vue/dist/vue.esm.js'
        }
    },
    plugins: [
        new UglifyJSPlugin()
    ]
};
```

PWA – Progressive Web Apps

- **“Progressive web apps (PWAs) are web applications that are regular web pages or websites, but can appear to the user like traditional applications or native mobile applications…” – Wikipedia**
- **To be a Progressive Web App, a site must:**
  - Originate from a Secure Origin, TLS.
  - Reference a Web App Manifest with at least the following properties:
    - name
    - short_name
    - start_url
    - display with a value of standalone or fullscreen
    - An icon at least 144×144 large in png format.



PWA – Progressive Web Apps

- **To be a Progressive Web App, a site must:**
  - Load while offline (even if only a custom offline page). This means that Progressive Web Apps require Service Workers.
    - Service workers: scriptable network proxy in the web browser to manage the web/HTTP requests programmatically
- **Firefox, Chrome, Edge (soon)**
- **Advantages**
  - Better User Experience
  - One codebase + language for mobile+desktop
- **Disadvantages**
  - Yet another standard ☺
  - Hardware access depends on standards

```
if ('serviceWorker' in navigator) {
  navigator.serviceWorker.register('service-worker.js');
}
```

```
if (event.request.mode === 'navigate' ||
 (event.request.method === 'GET' &&

event.request.headers.get('accept').includes('text/html'
))) {
  console.log('Handling fetch event for',
event.request.url);
  event.respondWith(
    fetch(event.request).catch(error => {
      console.log('Fetch failed; returning offline page
instead.', error);
      return caches.match(OFFLINE_URL);
    })
  );
}
```

URL: b.socrative.com/student/

Room: HSR

Exercise 2

- **Reuse Exercise 1 (github)**
- **Refactor it to use**
  - Single File Components
  - Webpack
- **Convert it to a PWA**
  - (show offline.html, when offline)

```
if ('serviceWorker' in navigator) {
  navigator.serviceWorker.register('service-worker.js');
}
```

```
if (event.request.mode === 'navigate' ||
 (event.request.method === 'GET' &&

event.request.headers.get('accept').includes('text/html'
))) {
  console.log('Handling fetch event for',
event.request.url);
  event.respondWith(
    fetch(event.request).catch(error => {
      console.log('Fetch failed; returning offline page
instead.', error);
      return caches.match(OFFLINE_URL);
    })
  );
}
```

Questions?

Prof. Dr. Thomas Bocek
thomas.bocek@hsr.ch