Frontend Best Practices 2021
September Event (3/3)

Leightweight Design Systems

Markus Stolze

Director of Studies - CAS Frontend Engineering, MAS HCID, CAS FEE Advanced
Agenda

18:00 Presentations
   - Intro: Markus Stolze, OST (10 min)
   - Lightweight Design Systems - A Consultants View: Simon Halimonov (Required)
   - Lightweight Design System @ Crowdhouse: Kondrat Shmoylov (Crowdhouse)
   - Lightning Talks
     - Lightweight Design Systems with Atomic Kitchen: Thomas Sokolowski (Snowflake Productions)
     - Lightweight Design System @ Altoo AG
       Owen Benwell & Manuel Alabor (Altoo AG)

19:45 Discussion

20:00 Meet the Speakers - Wonder.me (Open-End)
Frontend Best Practices
Welcome

• Cont. Education **MAS HCID** since 2006

• Cont. Education HSR/OST: **CAS FEE** since 2014
AngularJS -> Angular X, Svelte, UX, Security, Clean Coding, …
Next start Spring 2022

• Cont. Education **CAS FEE Advanced** start Spring 2022

• Twitter **CAS_FEE@OST** since 2016

• Frontend Best Practices «Conference» since 2017
(Meetup since 2019)
  • ’17: Frameworks: Angular oder React
  • ’18: Webpack, Chatbots, Performance
  • ’19: Plain-JS @ Frontify, Vue, GraphQL
  • ’20: XState, SEO
  • ’21: (1/3) Design Systems, (2/3) Vue, (3/3 Design Systems)
CAS FEE // CAS FEE Advanced

**CAS FEE**
- Wednesdays 17:15 – 21:50 (Rapperswil)
- Start: KW 17: Wed. April 27th 2022
- Duration: 38 Weeks / 8 Months

**CAS FEE Advanced -> Team Lead**
- Wednesdays 17:15 – 21:50 (Rapperswil)
- Start: KW 9: Wed. March 2nd 2022
- Duration: 33 Weeks / 7 Months
Info-Events

• CAS FEE - Info-Event Wed Sept. 8th 18:30
  https://www.ost.ch/de/weiterbildung/weiterbildungsangebot/informatik/cas-frontend-engineering

• CAS FEE Advanced - Info-Event Wed Nov. 17th 18:30
  https://www.ost.ch/de/weiterbildung/weiterbildungsangebot/informatik/software-engineering-testing/cas-frontend-engineering-advanced

• MAS HCID – Info-Event Fri Sept. 24th. 13:30 – 14:30
  https://www.ost.ch/de/weiterbildung/weiterbildungsangebot/informatik/mas-human-computer-interaction-design
Mark Your Calendar

Frontend Best Practices 2022 Feb 22

Frontend Best Practices ‘22

**Tuesday Feb. 22nd 2022; Dienstag 22 Feb 2022 - 18:00 – 20:00**

Topic TBD

Looking for Speakers & Suggestions / Referent:innen gesucht

@CAS_FEE; #FBP21
Code of Conduct

• Online conference/meeting: Mute your mics, video if you want (network?)
• Using welcoming and inclusive language (also Chat)
• Being respectful of differing viewpoints and experiences (also Chat)
• Gracefully accepting constructive criticism (also Chat)
• Showing empathy towards other community members (also Chat)
• Focusing on what is best for the community (also Chat)

• **Q&A / Discussions**
  Try to learn, not to proof something
  Start with “QUESTION”
  + Short description in Chat

@CAS_FEE; #FBP21
What is a «Design System»
(AKA «Living Styleguide», «Component Library», «DesignOps»)

Coined by Brad Frost «Atomic Design»
“Standard” Component Libraries

**Button**

**Names**
- Button - Ant Design, Adidas, Carbon Design System, Evergreen, UI Fabric, FAST, Primer Design System, Semantic UI, Standout UI, ...
- Buttons - Auth0 Styleguide, Bootstrap, Lightning Design System, Material Components Web

**Button group** - Bootstrap

**Button Icons** - Lightning Design System

**Anatomy**
- icon
- root
- content

**Concepts**
- Show descriptions

<table>
<thead>
<tr>
<th>Size</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>small</td>
<td></td>
</tr>
<tr>
<td>field</td>
<td></td>
</tr>
<tr>
<td>mini</td>
<td></td>
</tr>
<tr>
<td>tiny</td>
<td></td>
</tr>
<tr>
<td>medium</td>
<td></td>
</tr>
<tr>
<td>big</td>
<td></td>
</tr>
<tr>
<td>huge</td>
<td></td>
</tr>
<tr>
<td>massive</td>
<td></td>
</tr>
</tbody>
</table>

Some things to think about before you decide to create your own component library.

When you started your last software project, did you write your own programming?

Making buttons accessible

Material Design spec advises that touch targets should be at least 48 x 48 px. To meet this requirement, add the following to your button:

```html
<div class="mdc-touch-target-wrapper">
<button class="mdc-button mdc-button--touch">
  <span class="mdc-button__ripple"></span>
  <span class="mdc-button__touch"></span>
  <span class="mdc-button__label">My Accessible Button</span>
</button>
</div>
```

Note: The outer `mdc-touch-target-wrapper` element is only necessary if you want to avoid potentially overlapping touch targets on adjacent elements (due to collapsing margins).

**Text button**

Text buttons are typically used for less-pronounced actions, including those located in dialogs and cards. In cards, text buttons help maintain an emphasis on card content.

**Text button example**
Previous FBP: («Corporate») Design Systems
February 23rd 2021

18:00 Presentations
  • Intro: Markus Stolze, OST (10 min)
  • Design System @ Post AG: Leena Majaranta & Carla Hofer, Post AG (25 min)
  • Design System @ SBB: Yves Tscherry, SBB / Unic (25 min)
  • Design System @ Axa: Luca Mele, Axa (20 min)
  • Design System @ Bank Vontobel: Robert Denus & Pavlos Vos, Bank Vontobel (15 min)

19:45 Discussion 15 min
  • Presenters
  • Daniel Demel (Frontify)
  • Philippe Morier (Locatee)
  • Fabian Mettler (TX Group)

Summary: Roland Siegenthaler, echtpraktisch.ch

20:00
Exchange in Wonder.io (Open-End)
Agenda

18:00  Presentations

  • Intro: Markus Stolze, OST (10 min)
  
  • Lightweight Design Systems - A Consultants View: Simon Halimonov (Required)
  
  • Lightweight Design System @ Crowdhouse: Kondrat Shmoylov (Crowdhouse)
  
  • Lightning Talks
    – Lightweight Design Systems with Atomic Kitchen: Thomas Sokolowski (Snowflake Productions)
    – Lightweight Design System @ Altoo AG
      Owen Benwell & Manuel Alabor (Altoo AG)

19:45  Discussion

20:00  Meet the Speakers - Wonder.me (Open-End)
Questions / Discussion Topics

• How do you introduce an Design System incrementally -- or is this not possible / needed for running a "lightweight" Design System? Do you introduce the developer-facing side (component-libs etc.) together with the design-facing side? Or one after the other? Then: which one first?

• Do you need a component library for running a successful design system -- or is having a Figma-Library already enough?

• What are success-cases, helpful practices, do's and dont's from experiences in successful projects?

• When does "using Bootstrap" become "using a design system"?

• Is it an option to build (instead of buy/adapt) a component library when a lightweight (and nimble) approach to design system introduction is required?

• Are there sensible alternatives to exposing a component library as a Storybook? What else do you need for a lightweight DS infrastructure

• What criteria do you use when evaluating a 3rd party component library: License, support options, future-safety, framework match, completeness/coverage, DX, stylability, extensibility, accessibility, i18n, out-of-the-box integration with Figma and Storybook, size (tree-shakeability) -- in this order? more?

• What are the advantages and disadvantages of using a framework-agnostic component library (e.g. Bootstrap, Material Design, Spectrum CSS) vs. framework-specific (e.g. Material-UI, Spectrum React, Ionic) vs. Multi-Framework (e.g. Talwind Headless UI, Teleric Kendo UI) -- OR is a component-library based on Web-Components (based on Svelte or Lit) the way to go even when "lightweight" is required?