Take Advantage of Conway's Law with React

Or: Teams VS Technology….fight!

“The basic thesis of this article is that organizations which design systems (in the broad sense used here) are constrained to produce designs which are copies of the communication structures of these organizations.”

How Do Committees Invent?

Melvin E. Conway, 1968

“The basic thesis of this article is that organizations which design systems (in the broad sense used here) are constrained to produce designs which are copies of the communication structures of these organizations.”

How Do Committees Invent?

Melvin E. Conway, 1968

“Don’t ship the org chart.”

Steven Sinofsky, ????

“Remember Sinofsky's "don't ship the org chart"? It is a lie. You cannot avoid it. You always ship the org chart. So the real question is, what is the org going to look like so that we ship something good-looking?”

Hacker News

An “Amazon Engineer”, 2011

What does this mean?

- We build teams
- Those teams build software
- The software’s structure resembles our teams’ structure
- This is bad!

What does this mean?

- We build teams
- Those teams build software
- The software’s structure resembles our teams’ structure
- This is bad!

- Except when it’s good?

Consider these ideas

- Separation of Concerns
- Write Everything Twice? Don’t Repeat Yourself.
- Not Invented Here

Examples

Company A (departments)

- UI Design is a single group at the company
- All product/feature teams share the same pool of designers
- UI Designers work closely with each other, but not colocated with other engineers

Company B (matrix)

- UI Designers are embedded in engineering teams
- A product/feature team might have one or two designers dedicated to their projects
- UI Designers work closely with other engineers, but not with other Designers

Code reuse will help

- Person A wrote something that works
- Person B needs to reuse some of that work
- Person A or B can extract it to import in their component
- Person C will see the benefit too

CSS examples 1/3

- External CSS files
- Your component’s style is defined in CSS files deployed separately
  - CSS changes can happen out of band with application deploys
  - Application deploys can happen with no change in CSS style (don’t update classes!)
  - But...nothing is optional

This page is intentionally left blank.

CSS examples 2/3

- CSS loaders (ex: Webpack)
- Your component has an imported style elsewhere in your application
  - CSS changes are separated from component changes, but still deployed together
  - Your CSS can be global or local, depending on need
  - But...bad dependency management leaves the risk of orphan code

JS

```
import styles from './app.css'

let element = `
  <div class="${styles.element}">
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Consequatur laudantium
  </div>
`

document.write(element);
```

CSS examples 3/3

- Styled Components https://www.styled-components.com/
- Your component has an inline style that is scoped only to the component
  - Changes are less likely to affect other components
  - Styles are colocated to the component using them
  - Removing the component also cleans up the component’s CSS
  - But...your component is less readable

```
1  import styled from 'styled-components';
2
3  const Text = styled.div`
4    color: white,
5    background: black
6  `
7
8  <Text>Hello CSS-in-JS</Text>
```

css-in-jss.js hosted with ❤️ by GitHub view raw

Conclusions

- There is no Right Answer™
- Sorry.

Thank you!

- Pablo Nevares
- GitHub: @pnevares
- LinkedIn: @pnevares
- Twitter: @PabloNevares