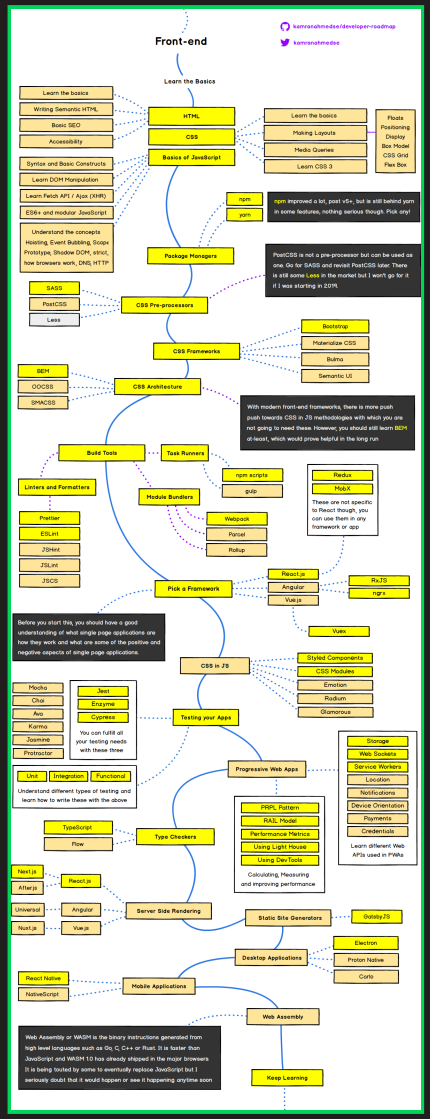
SMAKI JAVASCRIPTU

Prowadzący Mariusz Witkowski

SPIS TREŚCI

ROADMAP



Source: https://github.com/kamranahmedse/developer-roadmap

FRAMEWORKI

DLACZEGO POTRZEBUJEMY FRAMEWORKÓW?

- Synchronizacja UI ze stanem
- Złożoność logiki biznesowej
- Duplikacja kodu
- Łatwość utrzymania
- WebComponents są niewystarczające

FUNDAMENTY

- Komponenty
  - widoki
  - reużywalne bloki
- Zarządzanie stanem
  - lokalnie
  - globalnie
- Router
- Http request

KOMPONENT

jest to mały, potencjalnie wielokrotnego użytku zestaw elementów logicznych, zachowań i elementów interfejsu (UI or API)

ROUTER

Jest to oprogramowanie odpowiedzialne za porządkowanie stanów aplikacji, przełączanie między różnymi widokami. Na przykład router początkowo wyświetli ekran logowania, a po pomyślnym zalogowaniu przejdzie do ekranu powitalnego użytkownika.

HTTP REQUEST

Zarządzanie połączeniem interfejsu użytkownika z backendem API

jQuery

- pomocnicze funkcje JS
- manipulacje drzewem DOM
- rozbudowane efekty animacji
- manipulacja eventami
- zapytania Ajax
- kompatybilność z wieloma przeglądarkami

jQuery - RELEASES

- Initial release **kwiecień 2006**
- Stable release **maj 2020**

`jQuery` - CZY POTRZEBNY?

- ~~pomocnicze funkcje~~ - JS ECMAScript
- ~~manipulacje drzewem DOM~~ - `querySelector`
- ~~rozbudowane efekty animacji~~ - CSS3 animations
- ~~manipulacja eventami~~ - Streams/Observable
- ~~zapytania Ajax~~ - fetch
- ~~kompatybilność~~ - Chromium/FF/Safari

ANGUARJS

- wprowadził komponenty
- uporządkował logikę
- dodał dyrektywy
- wbudowany ajax
- routing (SPA)

ANGUARJS - DLACZEGO KONIEC?

ANGULAR *(2+)*

One framework. Mobile & desktop.

ANGULAR ON GITHUB

- Organizacja: **Google**
- Stars: **61k+**
- Contributors: **1100+**
- Issues: **2800+**
- Forks: **16,6k+**

FUNDAMENTY FRAMEWORKU

- Komponenty - `core`
- Stan
  - lokalnie - `this`
  - globalnie - `rxjs/Subject`, `providers`
- Router - `RouterModule`
- Http request - `HttpClientModule`

ANGULAR BUNDLE SIZE

- @angular/core@9.1.7 - 304.2kB*
- @angular/common@9.1.7 - 63.7kB*
- @angular/router@9.1.7 - 87kB*

*  *tree-shakeable*

HOW TO START

```
npm install -g @angular/cli

ng new my-dream-app

cd my-dream-app

ng serve
```

TWORZENIE KOMPONENTU

```
<h2>Products</h2>

<div *ngFor="let product of products">

  <h3>
      {{ product.name }}
  </h3>

</div>
```

```
import { Component } from '@angular/core';

import { products } from '../products';

@Component({
  selector: 'app-product-list',
  templateUrl: './product-list.component.html',
  styleUrls: ['./product-list.component.css']
})
export class ProductListComponent {
  products = products;

  share() {
    window.alert('The product has been shared!');
  }
}
```

PRZYKŁAD

```
<div class="product-card">
  <img
    class="product-card__image"
    src={{data.image.src}}
    alt={{data.image.alt}}
  />
  <span class="product-card__name">
    {{data.name}}
  </span>
  <span class="product-card__discount">
    {{data.discount}}
  </span>
  <span class="product-card__price">
    {{data.price}}
  </span>
  <span class="product-card__old-price">
    {{data.oldPrice}}
  </span>
  <button
    class="product-card__button"
    (click)="addToCart()"
  >
    Add to cart
  </button>
</div>
```

```
import { Component, Input } from '@angular/core';

import ProductsStoreService from './ProductsStoreService'

@Component({
  selector: 'product-card',
  templateUrl: './product-card.component.html',
  styleUrls: ['./product-card.component.css']
})
export class ProductCardComponent {
  @Input data: Product;

  constructor(public productsStore: ProductsStoreService)

  addToCart() {
    this.productsStore.addToCart(this.data.id)
  }
}
```

```html
<div class="product-card">
  <span class="product-card__image-skeleton" />
  <span class="product-card__name-skeleton" />
  <span class="product-card__price-skeleton" />
  <span class="product-card__button-skeleton" />
</div>
```

```ts
import { Component } from '@angular/core';

@Component({
  selector: 'product-card-skeleton',
  templateUrl: './product-card-skeleton.component.html',
  styleUrls: ['./product-card-skeleton.component.css']
})
export class ProductCardSkeletonComponent {}
```

```html
<div class="product-card">
  <span class="product-card__error-message">
    Something went wrong :(
  </span>
  <button
    *ngIf="shouldRetry"
    class="product-card__button"
    (click)="retryFetch()"
  >
    Retry load
  </button>
</div>
```

```ts
import { Component } from '@angular/core';

import ProductsStoreService from './ProductsStoreService'

@Component({
  selector: 'product-card-error',
  templateUrl: './product-card-error.component.html',
  styleUrls: ['./product-card-error.component.css']
})
export class ProductCardErrorComponent {
  private retryCount = 0;

  constructor(public productsStore: ProductsStoreService)

  get shouldRetry() {
    return this.retryCount < MAX_NUMBER_OF_ATTEMPTS
  }

  retryFetch() {
    this.productsStore.retryFetch()
    this.retryCount++
  }
}
```

```
<div>
  <ng-template
    [ngIf]="products"
    [ngIfElse]="isLoading"
    [ngIfElse]="isError">
    <product-card
      *ngFor="let product of products"
      [data]="product"
    ></product-card>
  </ng-template>
  <ng-template #loading>
    <product-card-skeleton></product-card-skeleton>
  </ng-template>
  <ng-template #error>
    <product-card-error></product-card-error>
  </ng-template>
</div>
```

```
import { Component, OnInit } from '@angular/core';

import ProductsStoreService from './ProductsStoreService'

@Component({
  selector: 'product-list',
  templateUrl: './product-list.component.html',
  styleUrls: ['./product-list.component.css']
})
export class ProductListComponent implements OnInit {
  constructor(public productsStore: ProductsStoreService)

  ngOnInit() {
    this.productsStore.fetch()
  }

  get isLoading() {
    return this.productsStore.state === LOADING_STATE
  }

  get isError() {
    return this.productsStore.state === ERROR_STATE
  }

  get products() {
    return this.productsStore.products
  }
}
```

REACT

A JavaScript library for building user interfaces

REACT ON GITHUB

- Organizacja: **Facebook**
- Stars: **149k+**
- Contributors: **1300+**
- Issues: **490+**
- Forks: **28,8k+**

FUNDAMENTY FRAMEWORKU

- Komponenty - `core`
- Stan
  - lokalnie - `useState`, `mobx` *itp.*
  - globalnie - `useContext`, `redux`, `mobx` *itp.*
- Router - `react-router` *itp.*
- Http request - `fetch`, `axios`, `graphql` *itp.*

REACT BUNDLE SIZE

- react@16.13.1 - 6,3kB
- react-dom@16.13.1 - 114.6kB
- react-router@5.2.0 - 20.9kB*
- redux@4.0.5 - 7,3kB*

** tree-shakeable*

HOW TO START

```
npx create-react-app my-app
cd my-app
npm start
```

HOW TO START

lub z użyciem nextjs

```
npm init next-app nextjs-blog --example "https://github.com/zeit/next-learn-starter/tree/master/learn-star
cd nextjs-blog
npm run dev
```

TWORZENIE KOMPONENTU

```
// Native

const Hello = (props) => {
  return React.createElement(
    'div',
    null,
    `Witaj, ${props.toWhat}`
  );
}

ReactDOM.render(
  React.createElement(Hello, { toWhat: 'Świecie' }, null),
  document.getElementById('root')
);
```

```
// JSX

const Hello = (props) => {
  return <div>Witaj, {props.toWhat}</div>;
}

ReactDOM.render(
  <Hello toWhat="Świecie" />,
  document.getElementById('root')
);
```

WADY JSX

- słabo imituje html
- nie działa natywnie w przeglądarce
- `className` zamiast `class`
- css inline jako obiekt

```
<button
  className="btn"
  style={{ color: 'red', marginLeft: 10 }}
>
  Click me!
</button>
```

PRZYKŁAD

```jsx
import React, { useCallback, useContext } from 'react'
import ProductsContext from './ProductsContext'

const ProductCard = ({ data }) => {
  const productsContext = useContext(productsContext)
  const addToCart = useCallback(() => {
    productsContext.addToCart(data.id)
  }, [data.id, productsContext])

  return (
    <div className="product-card">
      <img
        className="product-card__image"
        src={data.image.src}
        alt={data.image.alt}
      />
      <span className="product-card__name">{data.name}</span>
      <span className="product-card__discount">{data.discount}</span>
      <span className="product-card__price">{data.price}</span>
      <span className="product-card__old-price">{data.oldPrice}</span>
      <button
        className="product-card__button"
        onClick={addToCart}
      >
        Add to cart
      </button>
    </div>
  )
```

```jsx
import React from 'react'

const ProductCardSkeleton = () => (
  <div className="product-card">
    <span className="product-card__image-skeleton" />
    <span className="product-card__name-skeleton" />
    <span className="product-card__price-skeleton" />
    <span className="product-card__button-skeleton" />
  </div>
)
```

```jsx
import React, { useState, useCallback, useContext } from 'react'
import ProductsContext from './ProductsContext'

const ProductCardError = () => {
  const [retryCount, setRetryCount] = useState(0)
  const productsContext = useContext(productsContext)
  const retryFetch = useCallback(() => {
    productsContext.retryFetch()
    setRetryCount(currentRetry => currentRetry + 1)
  }, [productsContext, setRetryCount])

  const shouldRetry = retryCount < MAX_NUMBER_OF_ATTEMPTS

  return (
    <div className="product-card">
      <span className="product-card__error-message">
        Something went wrong :(
      </span>
      {shouldRetry ? (
        <button
          className="product-card__button"
          onClick={retryFetch}
        >
          Retry load
        </button>
      ) : null}
    </div>
  )
```

```jsx
import React, { useEffect, useContext } from 'react'

import ProductsContext from './ProductsContext'
import ProductCard from './ProductCard'
import ProductCardSkeleton from './ProductCardSkeleton'
import ProductCardError from './ProductCardError'

const ProductCardList = ({ state, data }) => {
  const productsContext = useContext(productsContext)
  useEffect(() => {
    productsContext.fetch()
  }, [])

  const isLoading = productsContext.state === LOADING_STATE
  const isError = productsContext.state === ERROR_STATE

  return isLoading ? (
    <ProductCardSkeleton />
  ) : isError ? (
    <ProductCardError />
  ) : productsContext.map(product => <ProductCard key={product.id} data={product} />)
}
```

VUE.JS

The Progressive JavaScript Framework

VUE.JS ON GITHUB

- Twórca: **Evan You**
- Stars: **164k+**
- Contributors: **290+**
- Issues: **300+**
- Forks: **24,8k+**

FUNDAMENTY FRAMEWORKU

- Komponenty - `core`
- Stan
  - lokalnie - `data()`, `computed`
  - globalnie - `vuex`
- Router - `vue-router`
- Http request - `fetch`, `axios`, `graphql` *itp.*

VUE.JS BUNDLE SIZE

- vue@2.6.11 - 63.5kB*
- vuex@3.4.0 - 11kB*
- vue-router@3.1.6 - 25.9kB*

* *tree-shakeable*

HOW TO START

```
npm install -g @vue/cli
vue create hello-world
# simple form

cd my-app
npm run dev
```

TWORZENIE KOMPONENTU

```
<div id="app">
  {{ message }}
</div>
```

```
var app = new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue!'
  }
})
```

SINGLE FILE COMPONENTS

```
<template>
  <p>{{ greeting }} World!</p>
</template>

<script>
export default {
  data: function() {
    return {
      greeting: "Hello"
    };
  }
};
</script>

<style scoped>
p {
  font-size: 2em;
  text-align: center;
}
</style>
```

PRZYKŁAD

```vue
<template>
  <div class="product-card">
    <img
      class="product-card__image"
      :src="data.image.src"
      :alt="data.image.alt"
    />
    <span class="product-card__name">
      {{ data.name }}
    </span>
    <span class="product-card__discount">
      {{ data.discount }}
    </span>
    <span class="product-card__price">
      {{ data.price }}
    </span>
    <span class="product-card__old-price">
      {{ data.oldPrice }}
    </span>
    <button
      class="product-card__button"
      @click="addToCart"
    >
      Add to cart
    </button>
  </div>
</template>
```

```vue
<script>
export default {
  name: 'ProductCard',
  props: {
    data: {
      type: Object,
      required: true,
    },
  },
  methods: {
    addToCart() {
      this.$store.dispatch('addToCart', this.data.id)
    },
  },
};
</script>
```

```vue
<template>
  <div class="product-card">
    <span class="product-card__image-skeleton" />
    <span class="product-card__name-skeleton" />
    <span class="product-card__price-skeleton" />
    <span class="product-card__button-skeleton" />
  </div>
</template>

<script>
export default {
  name: 'ProductCardSkeleton',
};
</script>
```

```vue
<template>
  <div class="product-card">
    <span class="product-card__error-message">
      Something went wrong :(
    </span>
    <button
      v-if="shouldRetry"
      class="product-card__button"
      @click="retryFetch"
    >
      Retry load
    </button>
  </div>
</template>
```

```vue
<script>
export default {
  name: 'ProductCardError',
  data() {
    return {
      retryCount: 0
    }
  },
  computed: {
    shouldRetry() {
      return this.retryCount < MAX_NUMBER_OF_ATTEMPTS
    }
  },
  methods: {
    retryFetch() {
      this.$store.dispatch('retryFetch')
      this.retryCount++
    },
  },
};
</script>
```

```vue
<template>
  <div>
    <ProductCardSkeleton v-if="isLoading" />
    <ProductCardError v-else-if="isError" />
    <ProductCard
      v-else
      v-for="product in products"
      :key="product.id"
      :data="product"
    />
  </div>
</template>
```

```vue
<script>
import ProductCard from './ProductCard'
import ProductCardSkeleton from './ProductCardSkeleton'
import ProductCardError from './ProductCardError'

export default {
  name: 'ProductCardList',
  components: {
    ProductCardSkeleton,
    ProductCardError,
    ProductCard,
  },
  computed: {
    isLoading() {
      return this.$store.state === LOADING_STATE
    },
    isError() {
      return this.$store.state === ERROR_STATE
    },
    products() {
      return this.$store.products
    }
  },
  created() {
    this.$store.dispatch('fetch')
  },
};
</script>
```

OTHERS

- Svelte
- Flutter

BENCHMARKS

https://github.com/krausest/js-framework-benchmark

TRENDY

NPM TRENDS

https://www.npmtrends.com/react-vs-vue-vs-@angular/core

GOOGLE TRENDS

https://trends.google.com/trends/explore?geo=US&q=%2Fg%2F11c6w0ddw9,%2Fm%2F012l1vxv,%2Fg%2F11c0vmgx5d

STACK OVERFLOW TRENDS

https://insights.stackoverflow.com/trends?tags=angular%2Creactjs%2Cvue.js

BEST OF JS

- (Angular)[https://bestofjs.org/projects/angular]
- (React)[https://bestofjs.org/projects/react]
- (Vue)[https://bestofjs.org/projects/vuejs]

STATE OF JS

https://stateofjs.com/

CO DALEJ?

- TDD
- TypeScript
- GraphQL
- Functional programming *(RxJS)*

SOURCES

- https://blog.daftcode.pl/lexical-functional-programming-jargon-and-naming-convention-a4f0cf559fd
- https://github.com/kamranahmedse/developer-roadmap
- https://medium.com/dailyjs/the-deepest-reason-why-modern-javascript-frameworks-exist-933b86ebc445
- https://github.com/krausest/js-framework-benchmark

FRAMEWORKI

- Angular - https://angular.io/
- React - https://reactjs.org/
- Next.js - https://nextjs.org/
- Vue - https://vuejs.org/
- Svelte - https://svelte.dev/
- Flutter - https://flutter.dev/

PYTANIA?

THANK YOU!