



Ekosystem React

JavaScript Developer 2020/2021 (*jeśli przetrwamy*)

ZACZNIJMY OD PODSTAW:

Magiczny świat Front End



Trudne początki

1990

The WWW Virtual Library

- [Agriculture](#)
Agriculture, Beer & Brewing, Gardening...
- [Computer Science](#)
Computing, Graphics, Languages, Web...
- [Communications and Media](#)
Communications, Telecommunications, Journalism...
- [Education](#)
Education, Cognitive Science, Libraries, Linguistics...
- [Engineering](#)
Civil, Chemical, Electrical, Mechanical, Software...
- [Humanities](#)
Anthropology, Art, Dance, History, Museums, Philosophy...
- [Information Management](#)
Information Sciences, Knowledge Management...
- [International Affairs](#)
International Security, Sustainable Development, UN...
- [Law](#)
Law, Environmental Law...
- [Business and Economics](#)
Economics, Finance, Transportation...
- [Recreation](#)
Recreation, Games, Gardening, Sport...
- [Regional Studies](#)
Asian, Latin American, West European...
- [Science](#)
Biosciences, Medicine & Health, Physics, Chemistry...
- [Society](#)
Political Science, Religion, Social Sciences...

Mirrors: [Stanford](#) (USA), [Penn State](#) (USA), [East Anglia](#) (UK) [Geneva](#) (CH), [Geneva-2](#) (CH), [Argentina](#).

[About the VL](#) | [Alphabetical listing](#) | [VL keyword search](#) | [What's New](#)

Last update Nov 23 1998

2010

zune products music video podcasts my social search

zune hd

ZUNE HD 16
ZUNE HD 32
ZUNE HD 64
LEARNING CENTER

bigger on the inside
The new Zune HD 64 has enough room for a music and movie collection. And with Smart DJ right tracks, you can sit back and let the good s Buy now at Zune Originals

Take your entertainment experience to new levels Watch video

touchscreen
Enhanced entertainment is vivid with an easy-to-use 3.3 inch touchscreen for all navigation. Navigate Zune HD touchscreen

HD Radio
Get instant access to a variety of FM stations at no additional cost with crystal clear, CD-quality sound. Discover the benefits of HD Radio

HD video out
Watch HD movies and TV shows, or shop for music from Zune Marketplace, all on your big screen HDTV. (Zune HD AV dock required, sold separately.)* Learn more

wi-fi
Buy and stream music in wi-fi hotspots, sync wirelessly to your PC, and surf the Internet on wireless networks.* Shop Zune Marketplace on wi-fi

be appy
We continue adding more free apps for Zune HD so you can connect to friends on Facebook and Twitter, play cards, bowl, race cars, and much more. See what's new

get personal with zune originals
Put your own spin on your Zune HD. Choose from five colors, artist designs and logos, and make it yours with text engraving. Personalize Zune HD

*Offer limited to U.S. only.
**Zune HD AV Dock and an HDTV (all sold separately) are required to view video at HD resolution. Supported 720p HD video files play on the player, downscaled to fit the screen at 480 x 272 – not HD resolution. Zune Pass subscription required; streaming via wi-fi available in U.S. only. HD Radio™ is a proprietary trademark of iBiquity Digital Corp. Learn more about HD Radio [here](#).

©2010 Microsoft Corporation. All Rights Reserved

Zune Media Player and Flat Design in 2009

2000

Netscape: Welcome to Batman Forever

File Edit View Go Bookmarks Options Directory Help

Back Forward Home Reload Images Open Print Find Stop

WELCOME TO GOTHAM

GALLERY MAIL LIBRARY CINEMA RADIO

WHAT'S NEW FRODLES PATHFINDER HELP

<http://www.batmanforever.com/welcome/gotham.map?442.143>

MacBook Air

Nowość Uskrzydłony mocą. Już od 5199 zł Kup Dowiedz się więcej >

MacBook Air

Nowość 13-calowy

Oto przyszłość Maca.

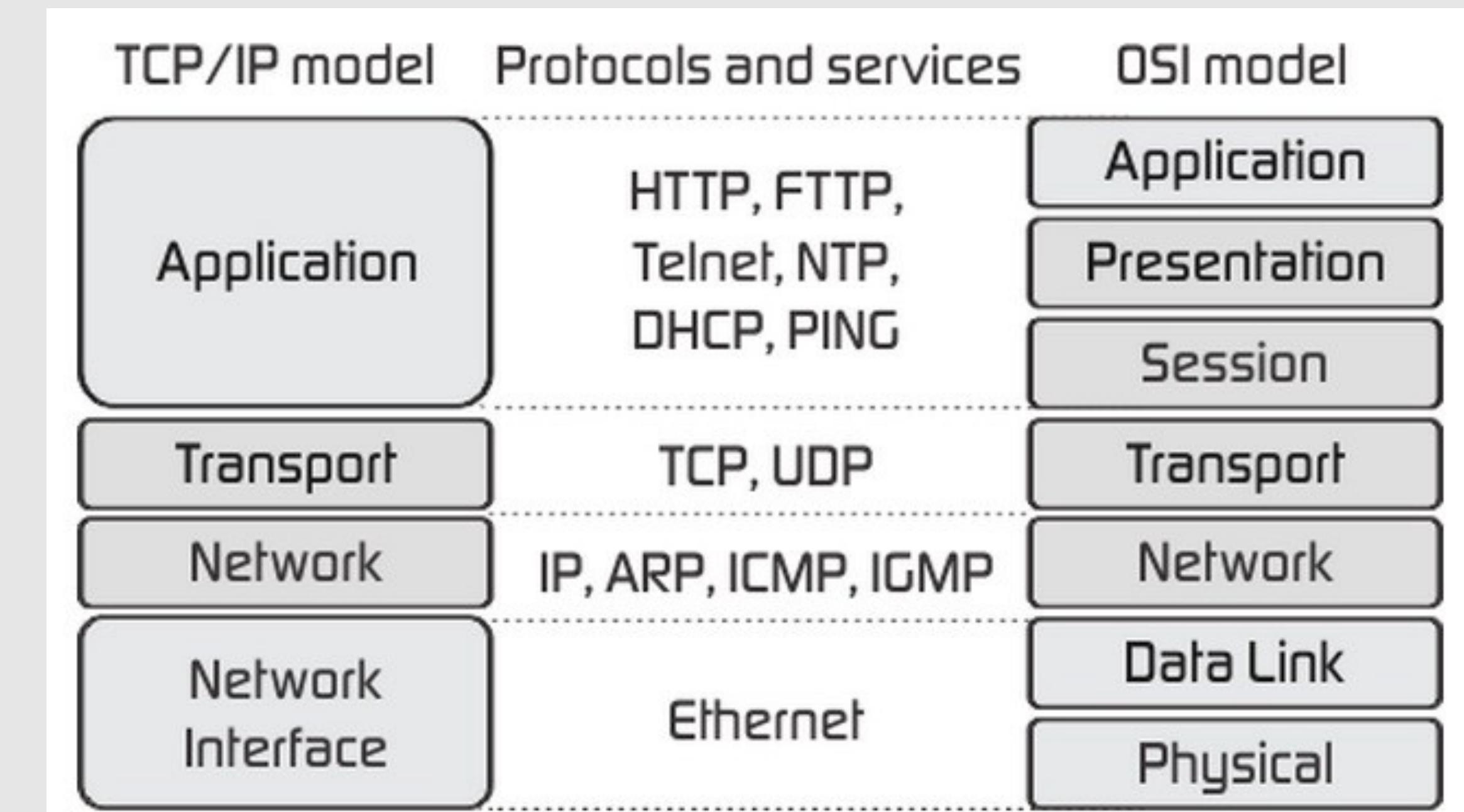
Protokół HTTP

HyperText Transfer Protocol

PROTOKÓŁ KOMUNIKACYJNY W WARSTWIE APLIKACJI MODELU OSI

Wymienia dane w zestawionym połączeniu

Wymiana odbywa się przy wykorzystaniu akcji **request** i **response**



HTTP Request

- METHOD:
 - GET
 - POST
 - PUT
 - PATCH
 - DELETE
 - PATH
 - VERSION
 - HEADERS
 - BODY
- ## Example **GET** request
- Request URL:** chrome-extension://mmgplondnjekobonklacmemikcnhklla/logo.png
- Request Method:** GET
- Status Code:** 200 OK
- Referrer Policy:** strict-origin-when-cross-origin
- Response Headers (4)
- ▼ Request Headers
- ⚠ Provisional headers are shown
- Referer
- User-Agent:** Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.183 Safari/537.36

HTTP Response

- STATUS CODE
- STATUS MESSAGE
- VERSION
- HEADERS
- BODY

Example **GET** response

```
Request URL: chrome-extension://mmgplondnjekobonklacmemikcnhklla/logo.png
Request Method: GET
Status Code: 200 OK
Referrer Policy: strict-origin-when-cross-origin

▼ Response Headers
cache-control: no-cache
Content-Security-Policy: script-src 'self' 'unsafe-eval'; object-src 'self';
Content-Type: image/png
ETag: "/miQmUQaAyPA1uXlgIEZqkqfhgs="
```

HTML

HyperText Markup Language

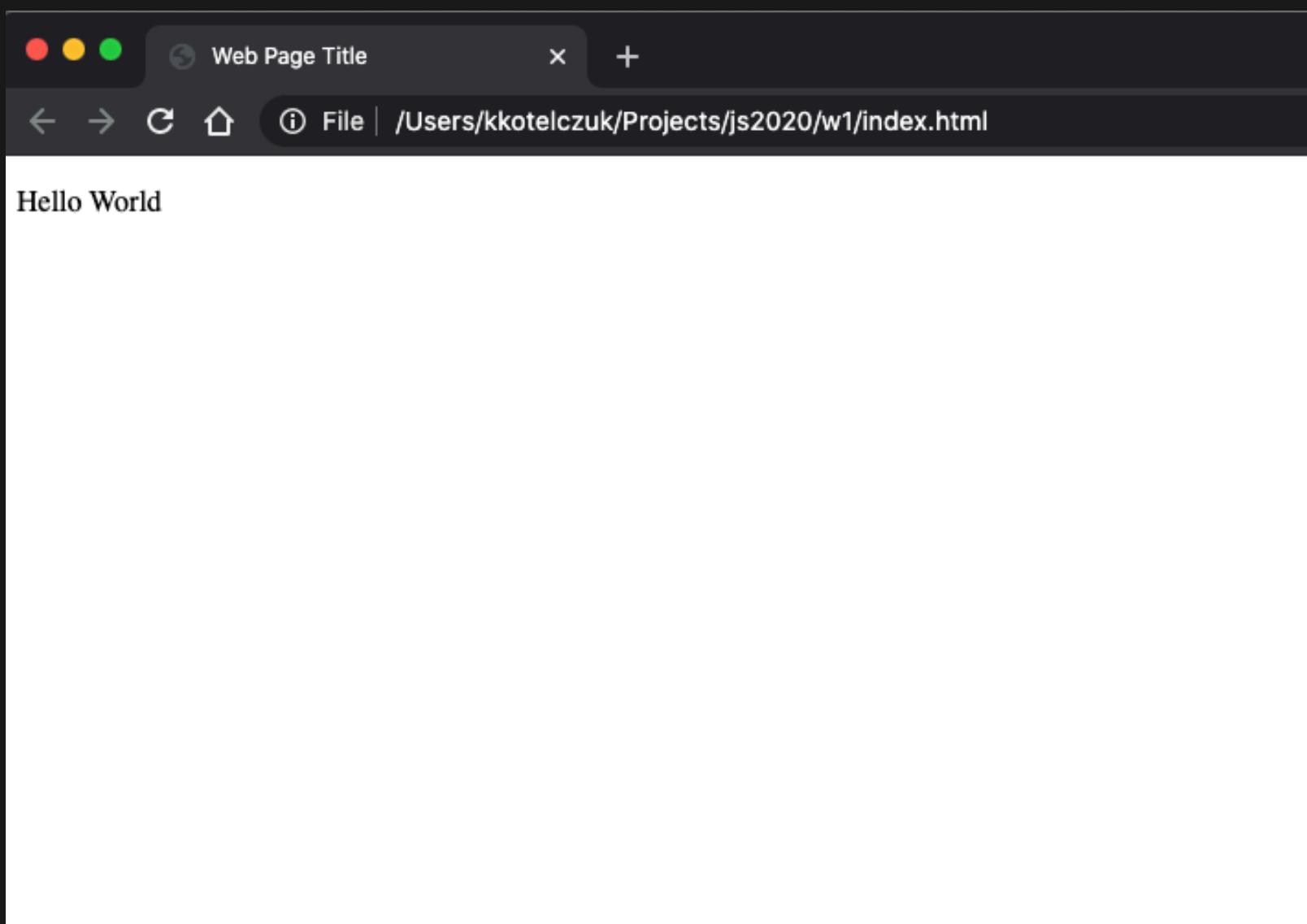
Podstawowa struktura HTML

```
<!DOCTYPE html>

<html>
  <head>
    <title>Web Page Title</title>
  </head>
  <body>
    To też zadziała
    <p>Hello World</p>
  </body>
</html>
```

```
<Deklaracja typu pliku>

<Ramka dokumentu >
  <Znacznik sekcji nagłówka strony>
    <Znacznik nazwy strony >Web Page Title</Znacznik zakończenia nazwy strony>
  </Znacznik zakończenia sekcji nagłówka strony>
  <Znacznik "cała" dokumentu>
    <Znacznik paragrafu>Hello World</Znacznik zakończenia paragrafu>
  </Znacznik zakończenia "cała" dokumentu>
</Ramka dokumentu >
```



Filter Hide data URLs **All** XHR JS CSS Img Media Font Doc WS Manifest Other Has blocked cookies Blocked Requests

10 ms	20 ms	30 ms	40 ms	50 ms	60 ms	70 ms	80 ms	90 ms	100 ms	110
-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	-----

Name x Headers Preview Response Initiator Timing

index.html

Request URL: file:///Users/kkotelszuk/Projects/js2020/w1/index.html
Referrer Policy: strict-origin-when-cross-origin

Request Headers

⚠ Provisional headers are shown
Upgrade-Insecure-Requests: 1

Filter Hide data URLs **All** XHR JS CSS Img Media Font Doc WS Manifest Other Has blocked cookies Blocked Requests

10 ms	20 ms	30 ms	40 ms	50 ms	60 ms	70 ms	80 ms	90 ms	100 ms	110
-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	-----

Name x Headers Preview Response Initiator Timing

index.html

Hello World

Filter Hide data URLs **All** XHR JS CSS Img Media Font Doc WS Manifest Other Has blocked cookies Blocked Requests

10 ms	20 ms	30 ms	40 ms	50 ms	60 ms	70 ms	80 ms	90 ms	100 ms	110
-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	-----

Name x Headers Preview Response Initiator Timing

index.html

```
1 <!DOCTYPE html>
2
3 <html>
4   <head>
5     <title>Web Page Title</title>
6   </head>
7   <body>
8     <p>Hello World</p>
9   </body>
10 </html>
11
```

CSS

CASCADING STYLE SHEETS

```
p {  
    color: blue;  
}  
  
body {  
    font-size: xx-large;  
}
```



CSS

```
<!DOCTYPE html>

<html>
  <head>
    <title>Web Page Title</title>
  </head>
  <body>
    <p>Hello World</p>
  </body>
</html>
```

```
p {
  color: blue;
}

body {
  font-size: xx-large;
}
```

The screenshot shows a browser's developer tools interface, specifically the Network tab. The timeline at the top indicates a total duration of 1800ms. Two requests are listed in the table:

Name	Headers	Preview	Response	Initiator	Timing
index.html					200000 ms - 1000000 ms
styles.css					1000000 ms - 1800000 ms

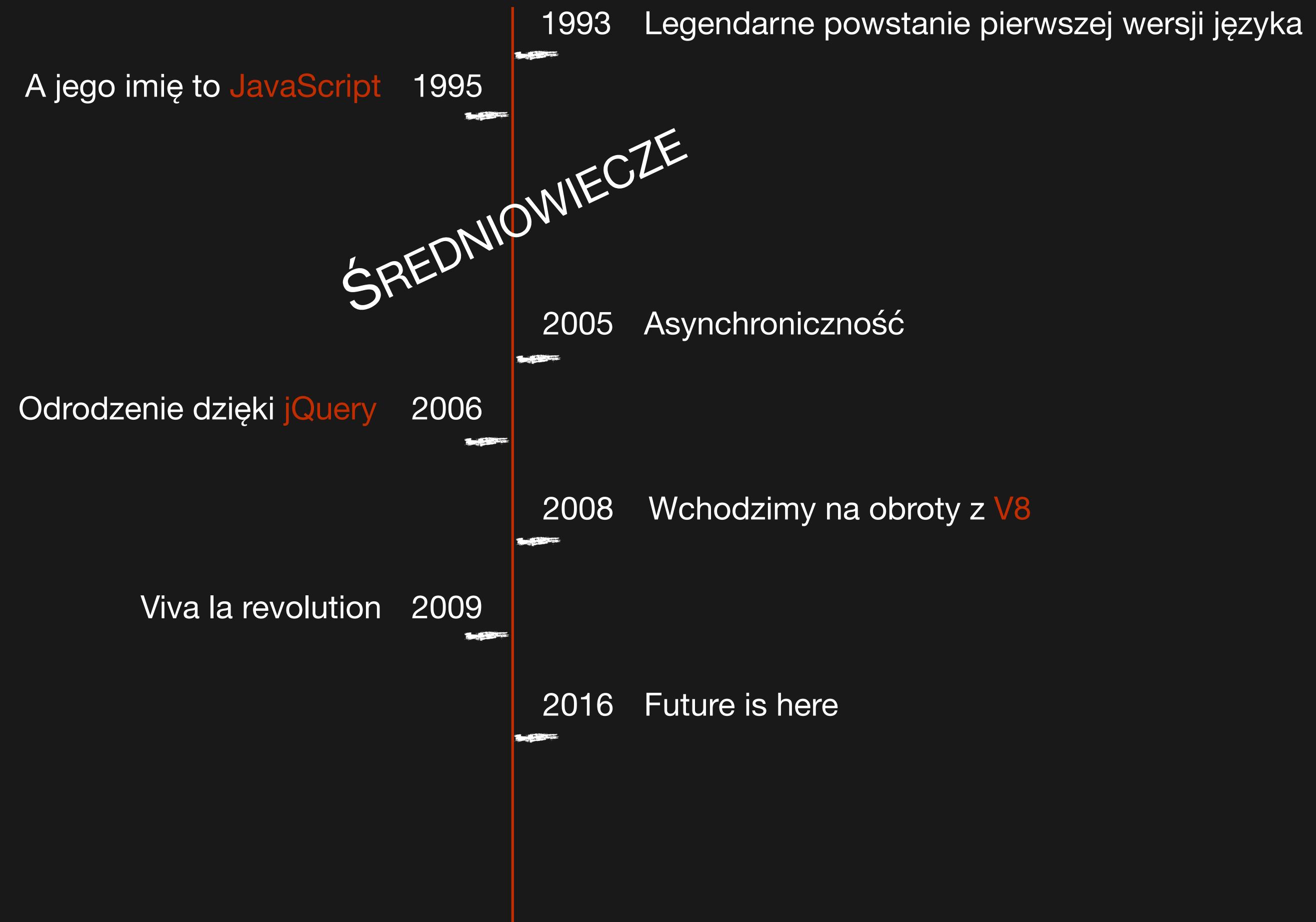
The 'index.html' request shows the following content:

```
1 p {
2   color: blue;
3 }
4
5 body {
6   font-size: xx-large;
7 }
```

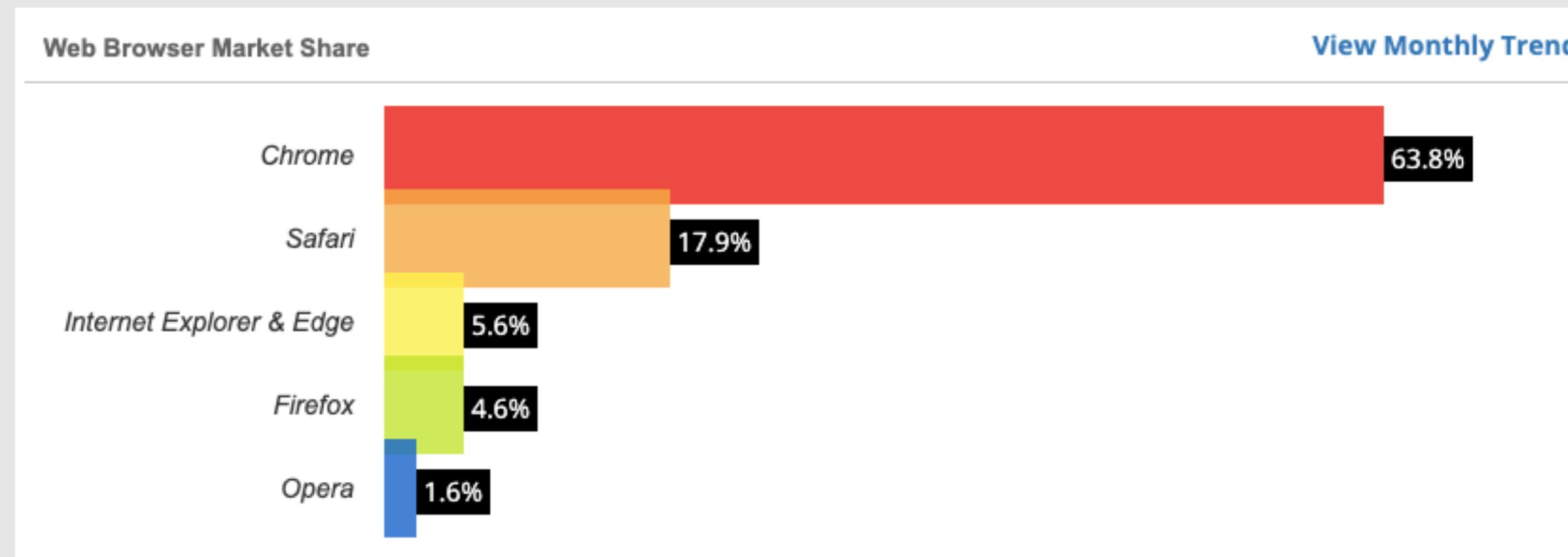
JavaScript

Tu już prawdopodobnie wszystko wiecie

Krótka lekcja historii



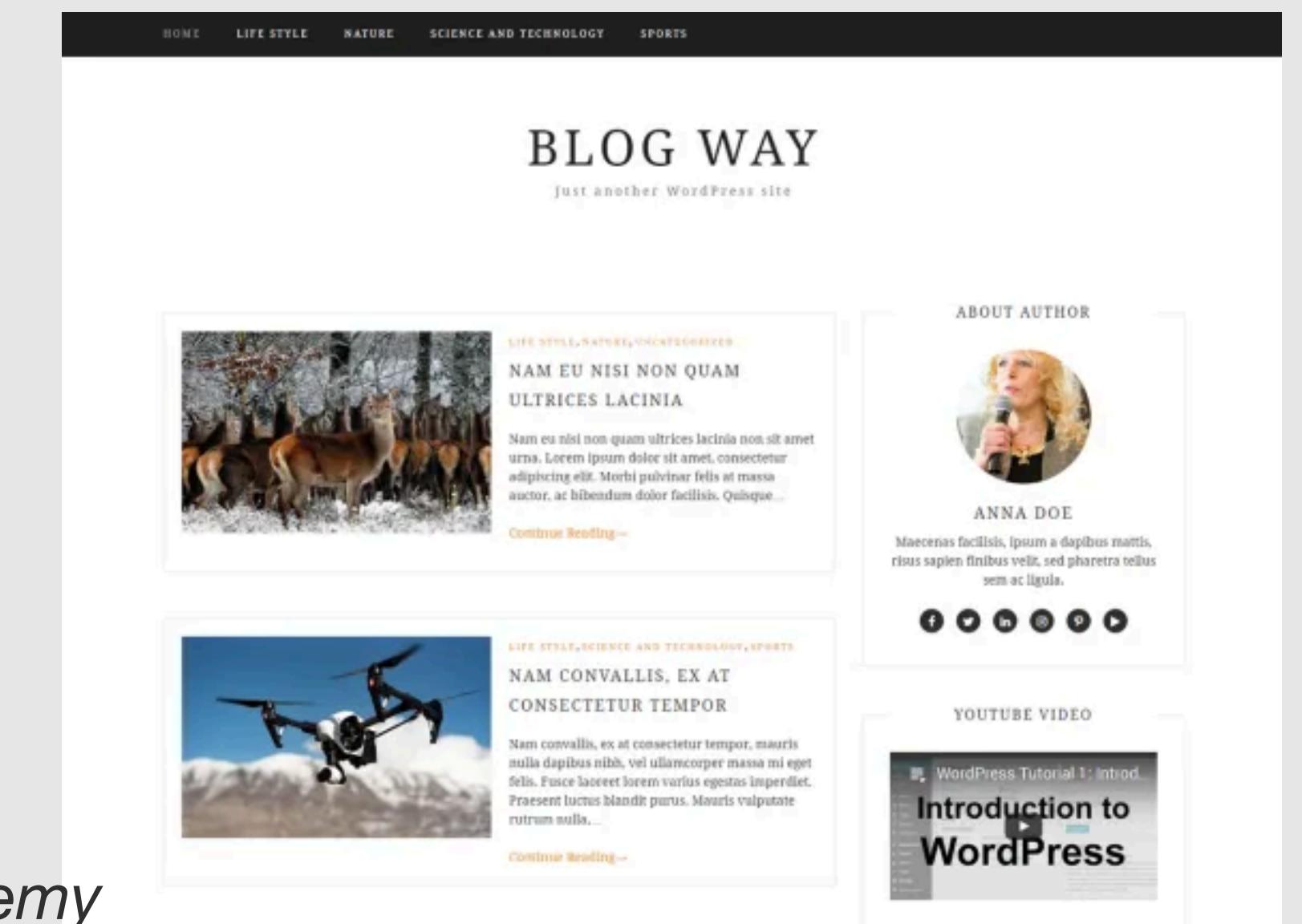
BROWSERS WARS





Multi Page Application

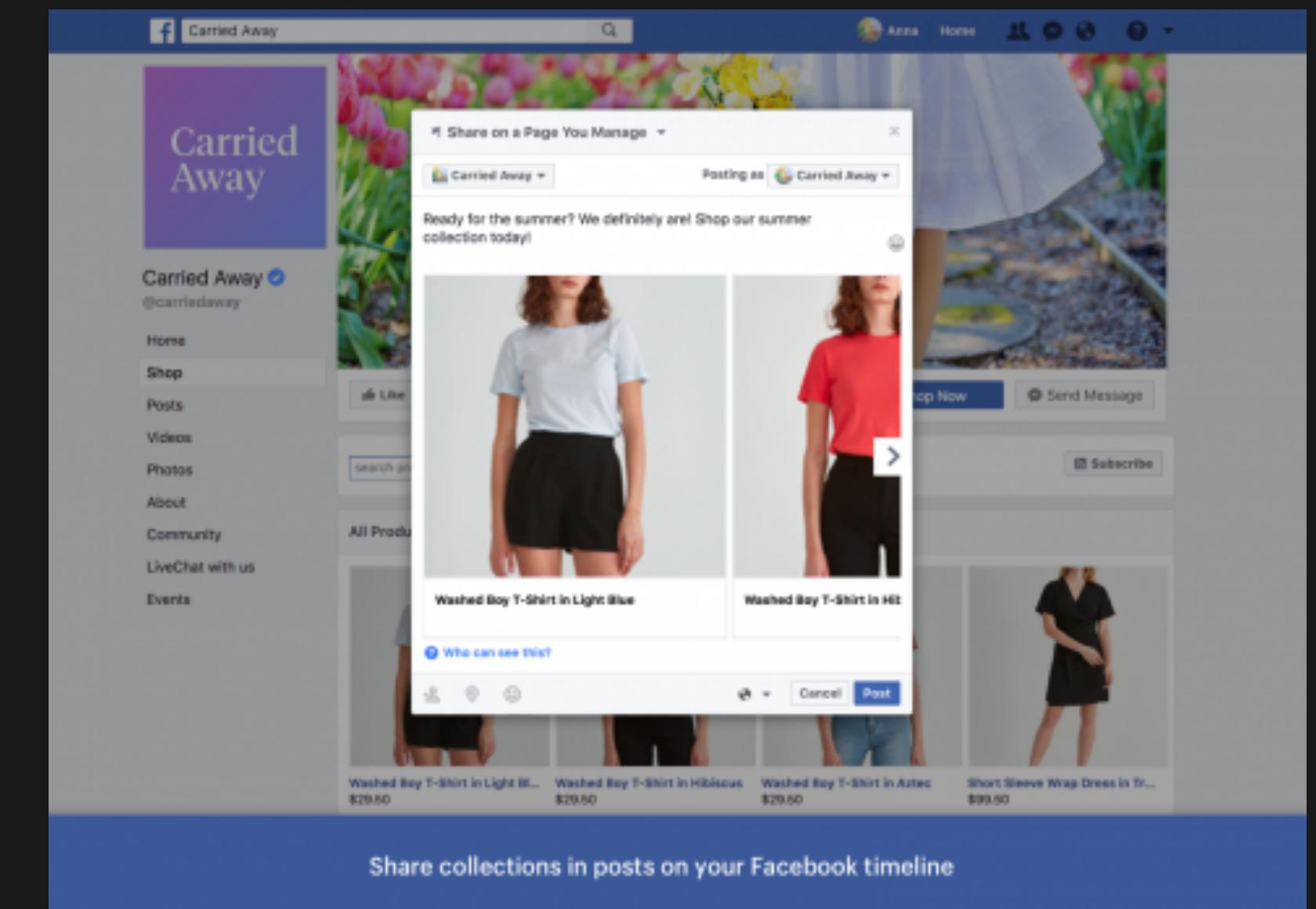
- Klasyczne strony
- Każde przejście na podstronę wymaga przeładowania zasobów
- Łatwe do pozycjonowania
- Klarowna struktura aplikacji



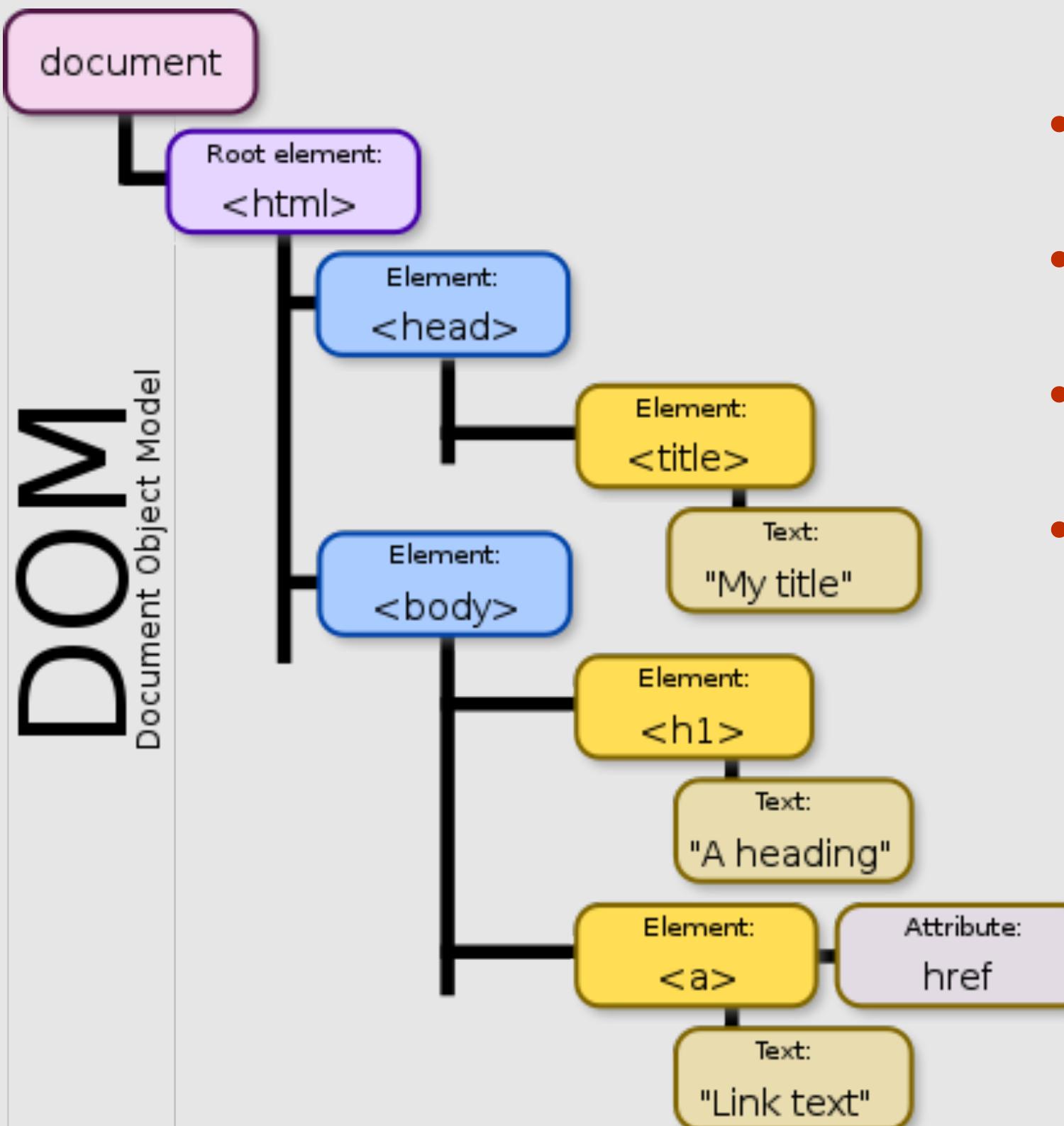
Dziś, wykorzystujemy głównie tam gdzie potrzebujemy dobrego SEO oraz nie mamy potrzeby często zmieniać zawartości strony

Single Page Application

- Strona składa się z 1 pliku js + 1 pliku HTML (zazwyczaj)
- Przechodzenie po stronie nie wymaga ciągłego ładowania wszystkich danych
- Znaczna część ciężaru aplikacji jest po stronie komputera użytkownika
- Pierwsze ładowanie trwa dłużej niż w MPA
- Proces twórczy jest bardziej skomplikowany



DOCUMENT OBJECT MODEL



- INTERFEJS
- Reprezentacja dokumentu HTML w formie drzewa logicznego
- Tworzony jest przez przeglądarkę
- Umożliwia JavaScript na modyfikowanie HTML'a
- Oraz dynamiczne reagowanie na wydarzenia

React

Rys historyczny



React v0.3.3

June 21, 2013 dodane przez [Paul O'Shannessy](#)

We have a ton of great stuff coming in v0.4, but in the meantime we're releasing v0.3.3. This release addresses some small issues people were having and simplifies our tools to make them easier to use.



React v15.0

April 07, 2016 dodane przez [Dan Abramov](#)

We would like to thank the React community for reporting issues and regressions in the release candidates on our [issue tracker](#). Over the last few weeks we fixed those issues, and now, after two release candidates, we are excited to finally release the stable version of React 15.

Biblioteka // framework

Kluczowe koncepcje

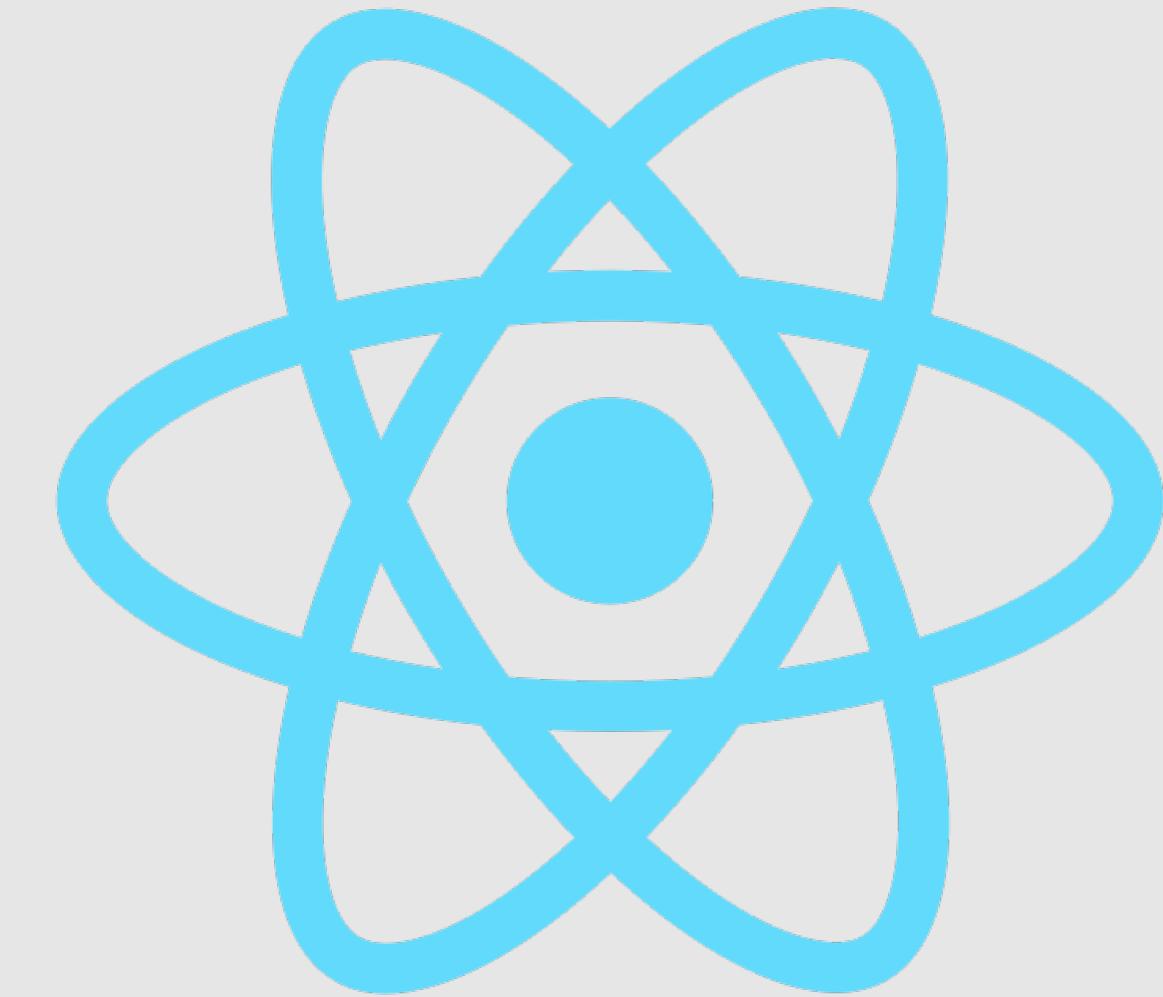
JSX

Wolność wyboru

One way **data** binding

Komponenty

Virtual **DOM**



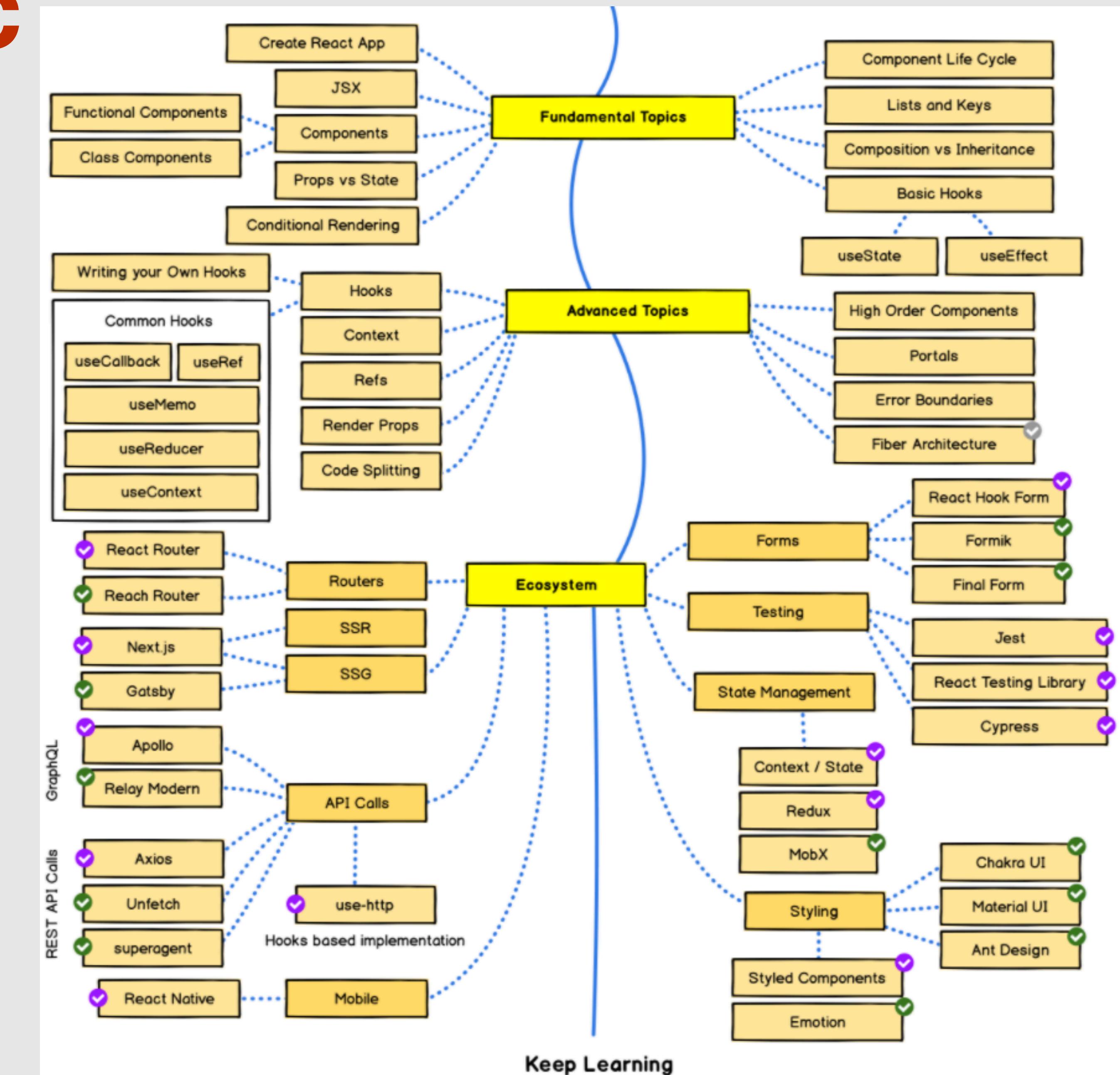
JSX

```
renderMyComponent = () => {
  const someData = [1,2,3,4,5,6]
  return (
    <div>
      {someData.map(item => <span>{item}</span>)}
    </div>
  )
}
```

```
<div>
  <span>1</span>
  <span>2</span>
  <span>3</span>
  <span>4</span>
  <span>5</span>
  <span>6</span>
</div>
```

Od czego zacząć

“Strive for progress, not perfection.”
— Unknown.



BABEL

Input:

```
// ES2020 nullish coalescing
function greet(input) {
  return input ?? "Hello world";
}
```

Output:

```
function greet(input) {
  return input != null ? input : "Hello world";
}
```

Środowisko programistyczne

01.

Przeglądarka

- Chrome, Safari, Edge
- Firefox, Opera, Internet Explorer
- ? Brave, Vivaldi

02.

System kontroli wersji Git

- Zainstalowany w systemie
- ? Aplikacja desktopowa
- ? Git dodany do konsoli

03.

Edytor tekstu

- Kartka papieru + ołówek
- Word, Wordpad, OpenOffice
- ? Notatnik
- ? VS CODE, Atom, WebStorm, VIM etc

04.

Niezbędne oprogramowanie

- Node.js
- ? Xcode command line tools
- ? Windows build tools

05.

Pomocnicze oprogramowanie

- ? Slack - desktop app
- ? M\$ Teams - desktop app
- ? Bash for command line

06.

Warto znać

- ? Skróty klawiaturowe
- ? Strona MDN
- ? Strona z dokumentacją React

Git Flow

Repozytorium

SSH

Fork

Pull Request

Remote

Merge

Local

Pull

Push

Fetch

Renderowanie elementów

ReactDOM

`ReactDOM.render(element, rootElement)`

rootElement = `document.getElementById('root')`

```
function tick() {  
  const element = (  
    <div>  
      <h1>Hello, world!</h1>  
      <h2>It is {new Date().toLocaleTimeString()}</h2>  
    </div>  
  );  
  ReactDOM.render(element, document.getElementById('root'));  
}  
  
setInterval(tick, 1000);
```

Komponenty

Koncepcyjnie, komponenty są jak javascriptowe funkcje.

Przyjmują one arbitralne wartości na wejściu (nazywane “właściwościami” (ang. *props*))
zwracają reactowe elementy opisujące, co powinno się pojawić na ekranie.

```
function Welcome(props) {  
  return <h1>Cześć, {props.name}</h1>;  
}
```

```
class Welcome extends React.Component {  
  render() {  
    return <h1>Cześć, {this.props.name}</h1>;  
  }  
}
```

React traktuje komponenty zaczynające się od małej litery jako tagi drzewa *DOM*.
Na przykład, `<div />` reprezentuje *HTML*-owy znacznik ‘div’,
ale już `<Welcome />` reprezentuje komponent i wymaga, aby Welcome było w *zasięgu (ang. scope)*.

Kompozycja komponentów

```
function Welcome(props) {  
  return <h1>Cześć, {props.name}</h1>;  
}
```

```
function App() {  
  return (  
    <div>  
      <Welcome name="Sara" />  
      <Welcome name="Cahal" />  
      <Welcome name="Edite" />  
    </div>  
  );  
}
```

```
ReactDOM.render(  
  <App />,  
  document.getElementById('root')  
>;
```

Wyodrębnianie komponentów

```
function Comment(props) {
  return (
    <div className="Comment">
      <div className="UserInfo">
        <img className="Avatar"
          src={props.author.avatarUrl}
          alt={props.author.name}>
        />
        <div className="UserInfo-name">
          {props.author.name}>
        </div>
      </div>
      <div className="Comment-text">
        {props.text}>
      </div>
      <div className="Comment-date">
        {formatDate(props.date)}>
      </div>
    </div>
  );
}
```

```
Comment(props) {
  (
    className="Comment">
      className="UserInfo">
        Avatar user={props.author} />
        <div className="UserInfo-name">
          {props.author.name}
        </div>
      />
      div className="Comment-text">
        {props.text}
      />
      div className="Comment-date">
        {formatDate(props.date)}
      />
    >
```

```
function Avatar(props) {
  return (
    <img className="Avatar"
      src={props.user.avatarUrl}
      alt={props.user.name}
    />
  );
}
```

```
function UserInfo(props) {
  return (
    <div className="UserInfo">
      <Avatar user={props.user} />
      <div className="UserInfo-name">
        {props.user.name}
      </div>
    </div>
  );
}
```

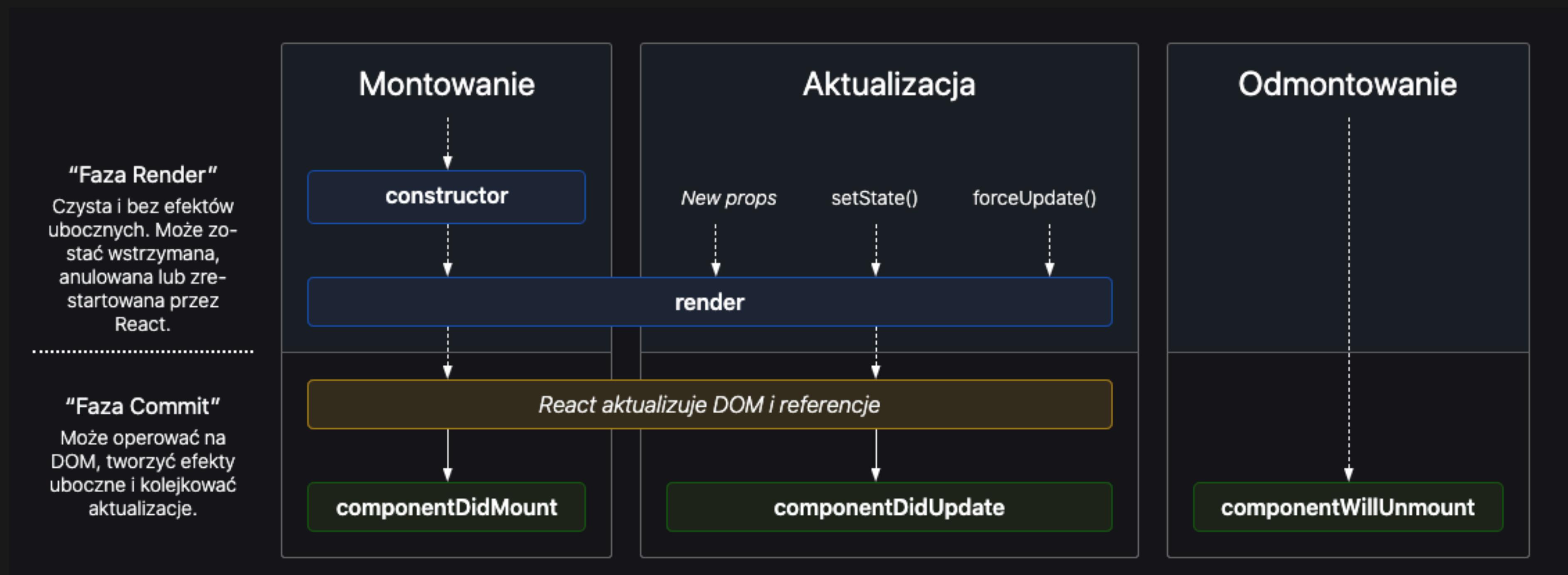
```
function Comment(props) {
  return (
    <div className="Comment">
      <div className="UserInfo">
        <img className="Avatar"
          src={props.author.avatarUrl}
          alt={props.author.name}
        />
        <div className="UserInfo-name">
          {props.author.name}
        </div>
      </div>
      <div className="Comment-text">
        {props.text}
      </div>
      <div className="Comment-date">
        {formatDate(props.date)}
      </div>
    </div>
  );
}

function Comment(props) {
  return (
    <div className="Comment">
      <div className="UserInfo">
        <img className="Avatar"
          src={props.author.avatarUrl}
          alt={props.author.name}
        />
        <div className="UserInfo-name">
          {props.author.name}
        </div>
      </div>
      <div className="Comment-text">
        {props.text}
      </div>
      <div className="Comment-date">
        {formatDate(props.date)}
      </div>
    </div>
  );
}
```

Pure functions

Wszystkie komponenty reactowe muszą zachowywać się jak pure functions - nie mogą modyfikować swoich propsów

Uproszczony cykl życia komponentu



Kompozycja nad dziedziczenie

Stan komponentu

```
function Clock(props) {
  return (
    <div>
      <h1>Witaj, świecie!</h1>
      <h2>Aktualny czas: {props.date.toLocaleTimeString()}</h2>
    </div>
  );
}

function tick() {
  ReactDOM.render(
    <Clock date={new Date()} />,
    document.getElementById('root')
  );
}

setInterval(tick, 1000);
```

```
ReactDOM.render(  
  <Clock />,  
  document.getElementById('root')  
);
```

```
function Clock(props) {  
  const [time, setTime] = useState(new Date())  
  return (  
    <div>  
      <h1>Witaj, świecie!</h1>  
      <h2>Aktualny czas: {time.toLocaleTimeString()}.</h2>  
    </div>  
  );  
}
```

```
useEffect(() => {
  this.timerID = setInterval(
    () => this.tick(),
    1000
  );
  return () => clearInterval(this.timerID);
})
```

```
function Clock(props) {
  const [time, setTime] = useState(new Date())

  useEffect(() => {
    this.timerID = setInterval(
      () => this.tick(),
      1000
    );
    return () => clearInterval(this.timerID);
  })

  return (
    <div>
      <h1>Witaj, świecie!</h1>
      <h2>Aktualny czas: {time.toLocaleTimeString()}</h2>
    </div>
  );
}
```

```
ReactDOM.render(
  <Clock />,
  document.getElementById('root')
);
```

```
class Clock extends React.Component {
  constructor(props) {
    super(props);
    this.state = {date: new Date()};
  }

  componentDidMount() {
    this.timerID = setInterval(
      () => this.tick(),
      1000
    );
  }

  componentWillUnmount() {
    clearInterval(this.timerID);
  }

  tick() {
    this.setState({
      date: new Date()
    });
  }

  render() {
    return (
      <div>
        <h1>Witaj, świecie!</h1>
        <h2>Aktualny czas: {this.state.date.toLocaleTimeString()}</h2>
      </div>
    );
  }
}

ReactDOM.render(
  <Clock />,
  document.getElementById('root')
);
```

Obsługa zdarzeń

Przez zdarzenie rozumiemy **obsługę akcji** wykonanych przez użytkownika

Zdarzenia reactowe pisane są **camelCasem**, a nie małymi literami.

W JSX procedura obsługi zdarzenia przekazywana jest **jako funkcja**, a nie łańcuch znaków.

```
<button onclick="activateLasers()">  
  Activate Lasers  
</button>
```

██████████

```
<button onClick={activateLasers}>  
  Activate Lasers  
</button>
```

```
<a href="#" onclick="console.log('Kliknięto w link.'); return false">
    Kliknij mnie
</a>
```

```
function ActionLink() {
    function handleClick(e) {
        e.preventDefault();
        console.log('Kliknięto w link.');
    }
    return (
        <a href="#" onClick={handleClick}>
            Kliknij mnie
        </a>
    );
}
```

SyntheticEvent

Enkapsulacja

Czyli renderowanie warunkowe

```
function UserGreeting(props) {
  return <h1>Witamy ponownie!</h1>;
}

function GuestGreeting(props) {
  return <h1>Proszę się zarejestrować.</h1>;
}
```

```
function Greeting(props) {
  const isLoggedIn = props.isLoggedIn;
  if (isLoggedIn) {
    return <UserGreeting />;
  }
  return <GuestGreeting />;
}
ReactDOM.render(
  // Spróbuj zmienić na isLoggedIn={true}:
  <Greeting isLoggedIn={false} />,
  document.getElementById('root')
);
```

&&

```
function Mailbox(props) {
  const unreadMessages = props.unreadMessages;
  return (
    <div>
      <h1>Cześć!</h1>
      {unreadMessages.length > 0 &&
        <h2>
          Masz {unreadMessages.length} nieprzeczytanych wiadomości.
        </h2>
      }
    </div>
  );
}

const messages = ['React', 'Re: React', 'Re:Re: React'];
ReactDOM.render(
  <Mailbox unreadMessages={messages} />,
  document.getElementById('root')
);
```

Uwaga - wyjątek

```
render() {  
  const count = 0;  
  return (  
    <div>  
      { count && <h1>Messages: {count}</h1>}  
    </div>  
  );  
}
```

Inline if-else

```
render() {  
  const isLoggedIn = this.state.isLoggedIn;  
  return (  
    <div>  
      The user is <b>{isLoggedIn ? 'currently' : 'not'}</b> logged in.  
    </div>  
  );  
}
```

If Guard

```
function WarningBanner(props) {  
  if (!props.warn) {  
    return null;  
  }  
  
  return (  
    <div className="warning">  
      Warning!  
    </div>  
  );  
}
```

Listy i klucze

```
const numbers = [1, 2, 3, 4, 5];
const doubled = numbers.map((number) => number * 2);
console.log(doubled)
```

//logged: 2, 4, 6, 8, 10

```
const numbers = [1, 2, 3, 4, 5];
const listItems = numbers.map((number) =>
  <li>{number}</li>
);
```

```
ReactDOM.render(
  <ul>{listItems}</ul>,
  document.getElementById('root')
);
```

```
function NumberList(props) {  
  const numbers = props.numbers;  
  const listItems = numbers.map((number) =>  
    <li>{number}</li>  
  );  
  return (  
    <ul>{listItems}</ul>  
  );  
}  
  
const numbers = [1, 2, 3, 4, 5];  
ReactDOM.render(  
  <NumberList numbers={numbers} />,  
  document.getElementById('root')  
)
```

```
function ListItem(props) {
  const value = props.value;
  return (
    <li key={value.toString()}>
      {value}
    </li>
  );
}

function NumberList(props) {
  const numbers = props.numbers;
  const listItems = numbers.map((number) =>
    <ListItem value={number} />
  );
  return (
    <ul>
      {listItems}
    </ul>
  );
}

const numbers = [1, 2, 3, 4, 5];
ReactDOM.render(
  <NumberList numbers={numbers} />,
  document.getElementById('root')
);
```

```
function ListItem(props) {  
  return <li>{props.value}</li>;  
}  
  
function NumberList(props) {  
  const numbers = props.numbers;  
  const listItems = numbers.map((number) =>  
    <ListItem key={number.toString()} value={number} />  
  );  
  return (  
    <ul>  
      {listItems}  
    </ul>  
  );  
}  
  
const numbers = [1, 2, 3, 4, 5];  
ReactDOM.render(  
  <NumberList numbers={numbers} />,  
  document.getElementById('root')  
>;
```

Formularze

```
<form>
  <label>
    Name:
    <input type="text" name="name" />
  </label>
  <input type="submit" value="Submit" />
</form>
```

```
const NameForm = () => {
  const [value, setValue] = useState('');

  const handleChange = event => {
    setValue(event.target.value);
  };

  const handleSubmit = event => {
    alert('A name was submitted: ' + value);
    event.preventDefault();
  };

  return (
    <form onSubmit={handleSubmit}>
      <label>
        Name:
        <input type="text" value={value} onChange={handleChange} />
      </label>
      <input type="submit" value="Submit" />
    </form>
  );
};
```

```
<textarea>  
  Hello there, this is some text in a text area  
</textarea>
```



```
<textarea value={this.state.value} onChange={this.handleChange} />
```

```
<option selected value="coconut">Coconut</option>
```

```
<select value={value} onChange={handleChange}>  
  <option value="grapefruit">Grapefruit</option>  
  <option value="lime">Lime</option>  
  <option value="coconut">Coconut</option>  
  <option value="mango">Mango</option>  
</select>
```

```
<select multiple={true} value={[ 'B' , 'C' ]}>
```

```
class Reservation extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      isGoing: true,
      numberOfGuests: 2
    };

    this.handleInputChange = this.handleInputChange.bind(this);
  }

  handleInputChange(event) {
    const target = event.target;
    const value = target.type === 'checkbox' ? target.checked : target.value;
    const name = target.name;
    this.setState({
      [name]: value
    });
  }

  render() {
    return (
      <form>
        <label>
          Is going:
          <input
            name="isGoing"
            type="checkbox"
            checked={this.state.isGoing}
            onChange={this.handleInputChange} />
        </label>
        <br />
        <label>
          Number of guests:
          <input
            name="numberOfGuests"
            type="number"
            value={this.state.numberOfGuests}
            onChange={this.handleInputChange} />
        </label>
      </form>
    );
  }
}
```