

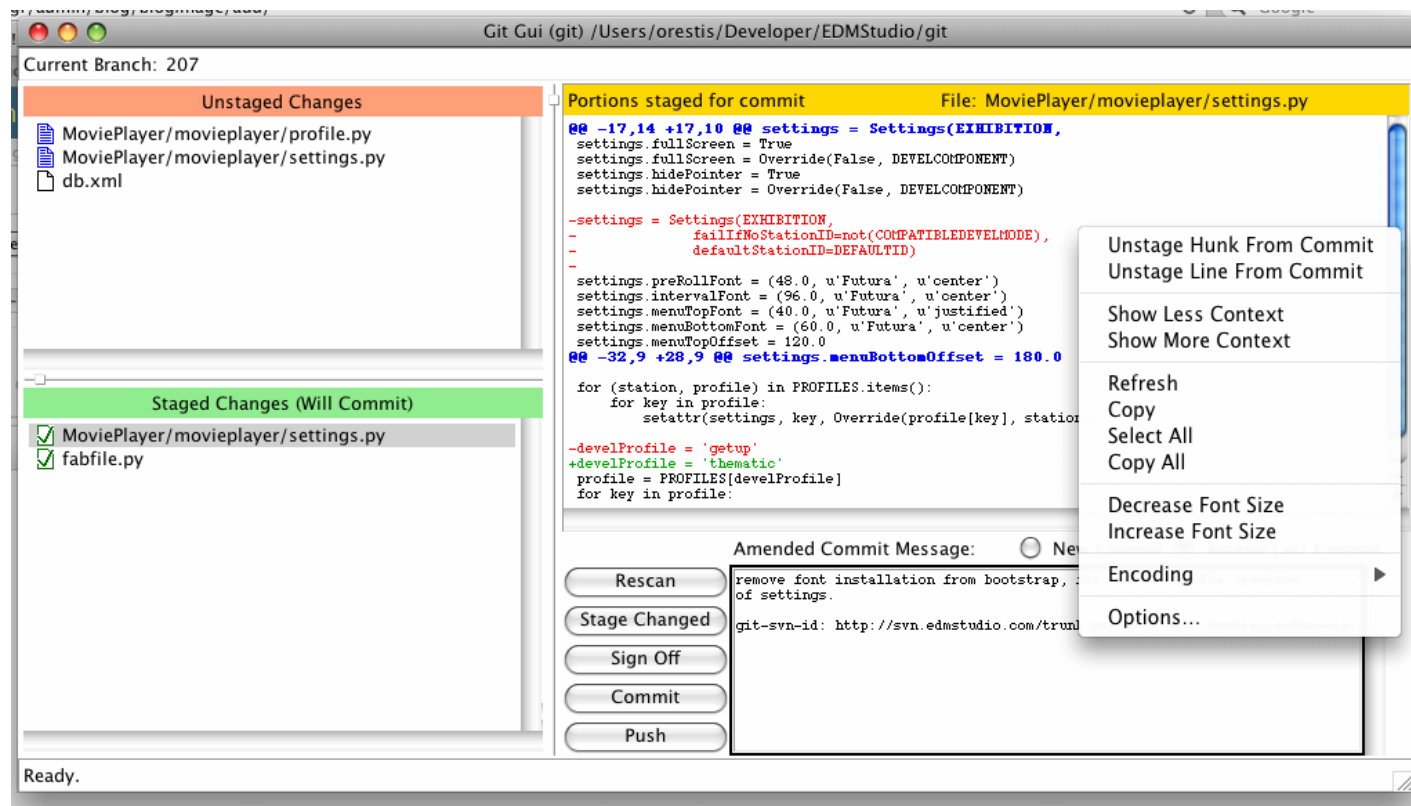
# IŠ PRAEITOS PASKAITOS

- su `git add` pridėtas pakeitimas yra valdomas `git`
  - norint jį pašalinti, reikia ne tik ištrinti iš katalogo, bet ir atlikti `git rm`
- konfigūracijos yra (pvz `.gitconfig` failams)
  - linux `/home/<username>`
  - windows `%UserProfile%`
  - mac `/Users/<username>`
- mac `kdiff3` instaliuoti reikia su `brew`  
<http://macappstore.org/kdiff3/>



# IŠ PRAEITOS PASKAITOS

- ne visuomet, tačiau dažniausiai greičiau ir patogiau nei komandinė eilutė yra git gui / git-gui



# IŠ PRAEITOS PASKAITOS

- git versija virtualioje mašinoje

```
$ git --version // git version 2.17.0
```

- pasiimti vieną failą iš kito branch'o

```
$ git checkout 516e831 failas.txt  
$ git checkout master failas.txt
```

- klonuojant repozitoriją ateina tik master branch'as, o kitus reikia pull'intis
- jei nesigauna - galite supakuoti ir atsiųsti visą repozitoriją .zip formatu ir parašyti, kad tiksliai neišeina





# AKADEMIJA.IT

INFOBALT IR TECH CITY

## WEB EVOLIUCIJA. REACTJS

Andrius Stašauskas

andrius@stasauskas.lt

<http://stasauskas.lt/itpro2018/>

# TURINYS

- Web evoliucija
  - Javascript istorija
  - Modernūs įrankiai
- React
  - JSX
  - stiliai ir kiti resursai
  - komponentai

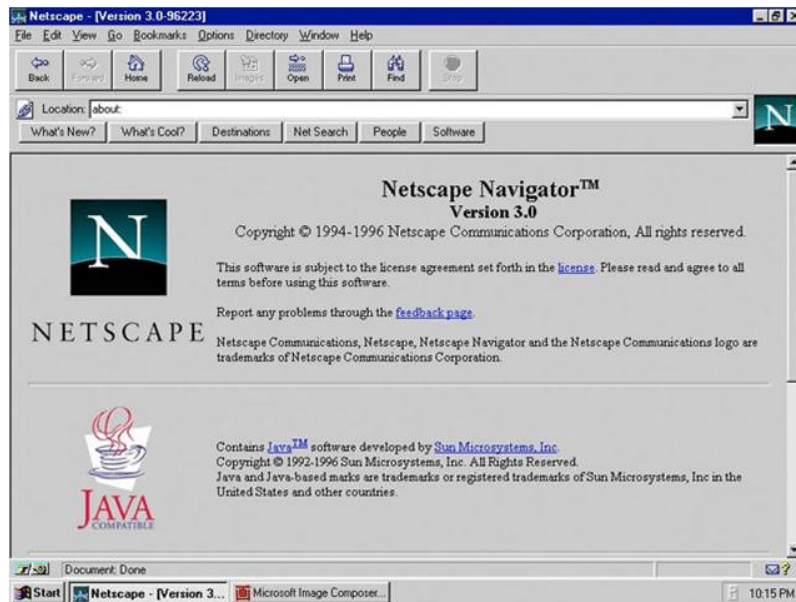


# INTRO

- 1991 m. Berners-Lee sukūrė ir paleido pirmąjį web puslapį
  - <http://info.cern.ch/>
- Dramatiškas web technologijų pasikeitimas
- Modernus web kūrimas = daug įrankių, daug konceptų, galybė būdų padaryti tą patį
- Įrankiai sudėtingi, bet leidžia kurti tokias aplikacijas, kurių prieš tai negalėjome
  - ir daug greičiau
  - ir daug lengviau valdomų (maintain)



# WEB 1.0 (1991-2001)



AKADEMIJA.IT

INFOBALT IR TECH CITY

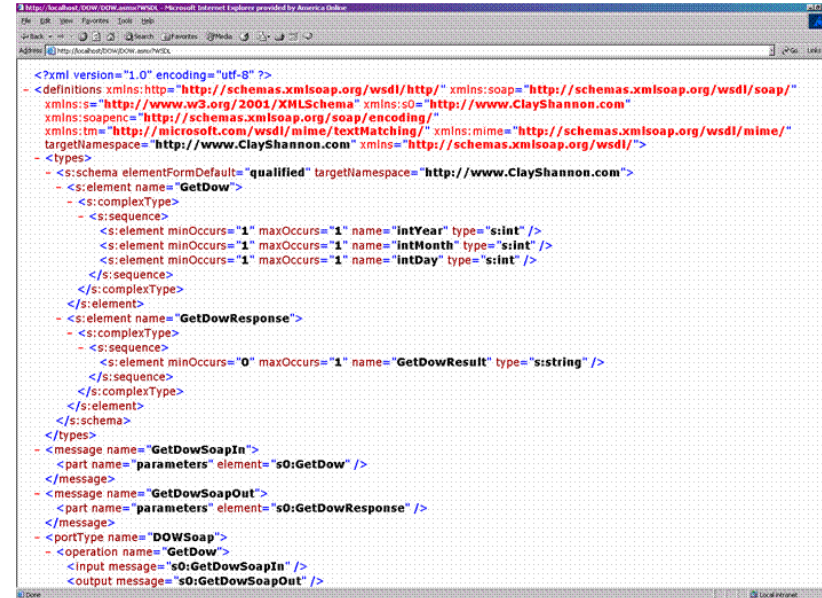
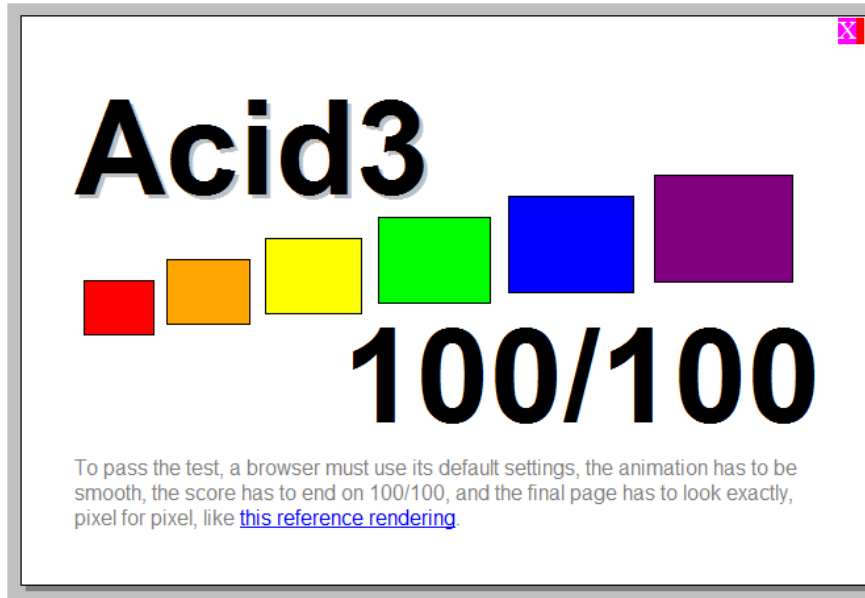
# WEB 1.0 (1991-2001)

- Tipiškas puslapis: static HTML, patalpintas GeoCities, parašytas naudojant FrontPage ar DreamWeaver
- Technologijos: /CGI-BIN -> Perl -> PHP/ASP -> Java / .NET
- Aplikacijos: skriptai ir vidutinio dydžio serveriai
- Interaktyvumas: Java apletai, ActiveX, DHTML
- Naršyklės: Mosaic, Netscape, IE
- Esminiai pasiekimai: gimė JS / HTML / CSS, išrasta Java ir Flash, multimedija per img ir iframe
- Kodo dalybos: "script list" svetainės
- Duomenų formatas: HTML su lentelėmis





# WEB 2.0 (2001-2010)



ADOBE  
FLASH PLAYER



Microsoft®  
Silverlight™



AKADEMIJA.IT

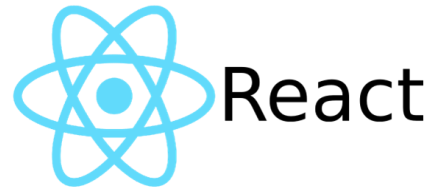
INFOBALT IR TECH CITY

# WEB 2.0 (2001-2010)

- Tipiška svetainė: puslapiai, surenderinti serveryje, MySQL duomenų bazė, naudojama J2EE ar Rails
- Serveris: J2EE, ASP.NET, Rails, Django, LAMP, monolitinė architektūra
- Klientas: AJAX, MooTools/Prototype, jQuery
- Naršyklės: Mozilla -> Firefox, IE6 -> IE8, Opera, Chrome
- Rich Internet aplikacijos (Flash, Silverlight, GWT, EXT, YUI)
- Esminiai pasiekimai: AJAX, mobile - išmanieji įrenginiai, JSON, Stack Overflow
- Kodo dalybos: SourceForge, Google Code
- Duomenų formatas ir perdavimas: XML per SOAP/WS



# MODERNI ERA (2010-DABAR)



AKADEMIJA.IT

INFOBALT IR TECH CITY

# MODERNI ERA (2010-DABAR)

- Tipiška svetainė: kliento JS, JSON API, mikroservisai su Node/Java paleisti per Amazon AWS, NoSQL DB
- Serveris: Node/Express, AWS, Java, ASP.NET MVC, Heroku
- Klientas: HTML5, Backbone, Angular, React, Bootstrap
- Naršyklės: Chrome, FF, Safari, Edge, visos "evergreen"
- Vieno-puslapio aplikacijos
- HTML5: canvas / WebGL, WebSockets, Web Workers, data storage, CSS flexbox, CSS grid, OS/native APIs
- Didžiausi pasiekimai: Node, naršyklės pluginų mirtis
- Kodo dalybos: decentralizuota versijų kontrolė, i.e. Github
- Duomenys: JSON -> binariniai per HTTP/WebSockets



# JAVASCRIPT ISTORIJA



**AKADEMIJA.IT**

INFOBALT IR TECH CITY

# JAVASCRIPT ISTORIJA

- 1995: Javascript kalba išrasta Brandon Eich iš Netscape
- 1999: ES3 specifikacija; XMLHttpRequest išrastas Microsoft
- 2001: JSON formatą populiarina Douglas Crockford
- 2005: "AJAX" termino populiarinamas
- 2006: jQuery išranda John Resig
- 2008: ES4 bando įtraukti tipus (atšauktas)
- 2008: "Javascript: the Good Parts" parašyta Douglas Crockford



# JAVASCRIPT ISTORIJA

- 2009: ES5 specifikacija; Node.js išranda Ryan Dahl
- 2010: Underscore.js, Backbone.js ir Coffeescript išranda Jeremy Ashkenas
- 2012: Microsoft sukuria TypeScript
- 2014: Facebook sukuria Flow (silpnėsni tipai)
- 2015: ES6 specifikacija (dar vadinama ES2015)
  - klasės, moduliai, iteratoriai, arrow funkcijos, binariniai duomenys, kolekcijos, generatoriai/promise'ai..



# JAVASCRIPT ISTORIJA

- 2016: ES7 specifikacija užbaigiama (ES2016)
- 2017: ES8 specifikacija užbaigiama (ES2017)
  - await/async kurie dirba su generatoriais/promise'ais
- 2018: ES9 specifikacija užbaigiama (ES2018)
- Brian Terlson (ECMAScript redaktorius):

I do not believe types are in the cards for the near future. <...>  
Just adopting TypeScript as the standard would of course be great  
for TypeScript users <...> don't expect anything near term





# MODERNUS WEB KŪRIMAS



Kas tai yra??



AKADEMIJA.IT

INFOBALT IR TECH CITY

# JAVASCRIPT IŠŠŪKIAI

- JS projektavimo tikslas buvo pasiekti, kad užvedus pelet ant beždžionės ji imtų šokti
- dažniausiai vienos eilutės skriptai
- 10 eilučių buvo normalūs
- 100 eil. buvo jau dideli
- 1000 eil. niekas net negirdėjo apie tokius
- JS nebuvo projektuotas didesniam programavimui
  - viskas tuo ir buvo paremta



# JAVASCRIPT IŠŠŪKIAI

- nėra modulių sistemos
- nėra enkapsuliacijos
- prototipais paremtas paveldimumas
- nėra statinių tipų ar kompiliavimo
- objektai ir duomenys dinamiškai modifikuojami
- minimali standartinė biblioteka
- skirtingos naršyklės ir jų galimybės
- dokumentų atvaizdavimo modelis perdarytas aplikacijoms



# JS PROGRAMUOTOJO TIKSLAI

- minimizuoti siunčiamų baitų skaičių
- susitvarkyti su naršyklių suderinamumu
- užpildyti JS standartinės bibliotekos ir JS kalbos spragas
- dalintis kodu tarp aplikacijų
- kurti vis sudėtingesnes pilnai paruoštas aplikacijas, kurios .. tiesiog yra naršyklėje
- aplikaciją kurti naršyklei yra sunku
- reikia paskirti tiek pat laiko kiek ir kuriant duomenų bazę ir schemą ar servisų sluoksnį
- pagarbos UI!



# MODERNŪS ĮRANKIAI

- Babel - JS kompiliatorius (ES6/ES2015 -> ES5)
- TypeScript - ES6, bet statiškai tipizuotas
- SASS/LESS
- Modulių formatai: asinchroninis AMD, sinchroninis CommonJS, ES6 statinis analizavimas
- Pakavimas/leidimas
  - Node.js - V8 JS Chrome Engine
  - NPM - paketų manageris su Node



# MODERNŪS ĮRANKIAI - SURINKIMAS

- Kompiliavimas: ES6/Typescript į ES5
- Grupavimas: daug failų į vieną
- Minifikavimas: pašalinami tarpai, komentarai, trumpiau pervadinami kintamieji
- Kodo atskyrimas: mažiau reikalinga - užkrauta vėliau
- Grunt: keletas užduočių per įskiepius
- Gulp: transformuoti duomenis žingsniais
- Browserify: CommonJS moduliai į naršyklę
- Webpack: bet kokius modulių (AMD/CJS/ES6 modules, CSS, images, ... ) transformuoti naršyklei ir ne tik

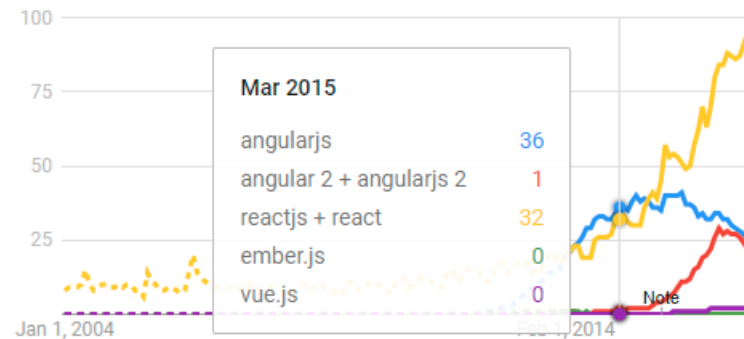


# KĄ NAUDOJA PROGRAMUOTOJAS

- Failų stebėjimas, kompiliavimas ir atnaujinimas
- Sourcemaps ir kiti naršyklės įrankiai (F12)
- Hot ("karštas") atnaujinimas
  - modulių pakeitimas, redagavimais gyvai
- Mocha, Jasmine, Tape, Jest ar Karma testai
- Chai, Expect, Sinon, JSDOM assert'ai
- Selenium, PhantomJS
- Klaidų/standartų tikrinimas-lintinimas: ESLint
- Bibliotekos: jQuery, Underscore/Lodash



# JAVASCRIPT KARKASAI (FRAMEWORKS)





# ATEITIS

- "X as a Service"
- "Serverless" backends - depends on BaaS backend as service
- Microservices, Containers
- Large-scale data transfer schemas/tools
- "Isomorphic" / "Universal" Javascript apps
- Server rendering
- Javascript everywhere
- Cross-platform toolkits (Cordova, Ionic)
- Desktop (Electron), Mobile (React Native)



# ATEITIS

- Component-based architectures
- "Virtual DOM"
- CSS-in-JS
- WebGL, Web Workers, Service Workers
- Functional Programming
- Immutable Data
- Reactive Programming / Observables
- Static typing



# UŽDUOTIS 1 - REDAKTORIAI

- MousePad arba Notepad++
- Eclipse senesnis
  - `apt-get install eclipse openjdk-8-jdk`
  - Help > Install New Software...
    - Eclipse Java EE Developer Tools
    - Eclipse Web Developer Tools
  - m2e - Maven Integration Plugin (maven projektui)
  - Eclipse failus su jsx kodu vadiname .jsx



# UŽDUOTIS 1 - REDAKTORIAI

- Eclipse naujesnis
  - <https://eclipse.org/downloads/eclipse-packages/>
  - Execute `ecclipse-inst` per Xubuntu file manager
  - Help -> Eclipse Marketplace plugin'ai:
    - Eclipse Web Developer Tools
    - TypeScript IDE
  - Projektą atidaryti
    - File -> Open Projects From Filesystem -> Directory



# UŽDUOTIS 1 - REDAKTORIAI

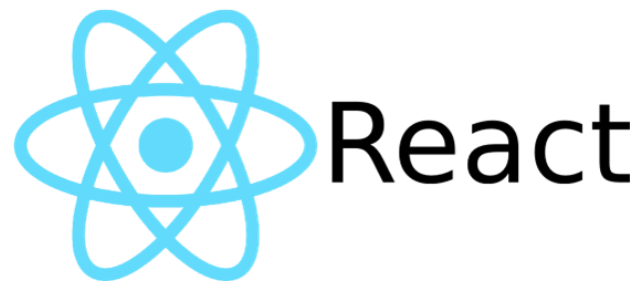
## Visual Studio Code

```
$ sudo add-apt-repository ppa:ubuntu-desktop/ubuntu-make  
$ sudo apt-get update  
$ sudo apt-get install ubuntu-make  
$ sudo umake ide visual-studio-code  
$ cd /.local/share/umake/bin  
$ ./visual-studio-code  
$ File -> Open Folder
```



# VARTOTOJO SĄSAJOS PROGRAMAVIMAS SU REACTJS





- MVC vaizdas, biblioteka, o ne karkasas
  - MVC ir susijusios architektūros - vėliau
- reikia pasirinkti dalis tam tikrai situacijai (būsenos valdymui, navigacijai, ..)
- įtakotas funkcinio programavimo (  $UI = f(būsenos)$  , vienkrypčiai duomenys)
- uždari komponentai ir žymėmis paremta JSX sintaksė ("HTML in JS")

# KODĖL BUVO SUKURTAS REACT

- Sunku buvo apjungti duomenis su UI
- Blogas UX naudojant "kaskadinius" DOM medžio atnaujinimus
  - React virtualus DOM per JS nes JS greitas
- Daug besikeičiančių duomenų
- Sudėtinga Facebook UI architektūra
- Parėjimas toliau nuo MVC mentaliteto





# PAPRASTAS 'HELLO WORLD' PAVYZDYS

```
<html>
<head>
  <title>E-Shop </title>
</head>
<body>
  <p>Labas</p>
</body>
</html>
```

```
$ firefox index.html
```



# ĮTRAUKIAME STILIUS IR JS

```
<html>
<head>
  <title>E-Shop </title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <p class="custom-paragraph">Labas</p>

  <!-- <script src="app.js"></script> -->
</body>
</html>
```



## app.js

```
console.log('Pasileido')

for (var i = 0; i < 10; i++) {
  console.log('Pasileido ' + i)
}
```

## styles.css

```
.custom-paragraph {
  color: red
}
```



# REACT KOMPONENTAS

- Web aplikacijos blokai, panaudojami ne vieną kartą
- React komponentas - tai funkcija
  - funkcijai paduodama informacija, gaunamas output
- Funkcija apibrėžia UI laike
- Kuriami naudojant `React.createClass()`
- vienintelis būtinas metodas yra `render()`
- įterpiami naudojant `React.renderComponent()` arba `ReactDOM.render()`



# REACT KOMPONENTAS

Home

Admin


Username

UsernameComponent

CartSummaryComponent

0 items

MenuComponent




Samsung Phone 1

Desc

20.6 Eur

Details




Samsung Phone 2

Desc

14.6 Eur

Details




Samsung Phone 3

Desc


14.6 Eur

Details




Samsung Phone 4

Desc



Samsung Phone 5

Desc



Samsung Phone 6

Desc

AKADEMIJA.IT

INFOBALT IR TECH CITY

37

## UŽDUOTIS 2 - PRIDEDAM REACT

Į head dalį:

```
<link
  rel="stylesheet"
  href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/boot
  integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXo
  crossorigin="anonymous"
>
```

Body pabaigoje:

```
<!-- // atkomentuoti savo kode
<script src="https://unpkg.com/react@16/umd/react.development.js"><
<script src="https://unpkg.com/react-dom@16/umd/react-dom.developme
<script src="https://unpkg.com/prop-types@15.6/prop-types.js"></scr
<script src="https://unpkg.com/babel-standalone@6/babel.min.js"></s
<script type="text/babel" src="app.js"></script>
-->
```



# PIEŠIMAS (REACTDOM.RENDER)

index.html

```
<div id="root"></div>
```

app.js

```
const component = <h1>Hello, world</h1>;

ReactDOM.render(
  component,
  document.getElementById('root')
);
```

- Susikurti branch'ą, nepamiršti git add , git commit , git push
  - kiekvieną dieną



# REACT KOMPONENTAI NAUDOJANT JAVASCRIPT

```
var HelloComponent = React.createClass({
  render: function() {
    return React.createElement(
      "div",
      null,
      "Hello ",
      this.props.name
    );
  }
});

ReactDOM.render(React.createElement(
  HelloComponent,
  { name: "Jane" }),
  document.getElementById('root')
);
```





## JSX PAGALBA

```
<MyButton color="blue" shadowSize={2}>  
  Click Me  
</MyButton>
```

## sukompiliavus

```
React.createElement(  
  MyButton,  
  {color: 'blue', shadowSize: 2},  
  'Click Me'  
)
```



# JAVASCRIPT JSX

- JSX galima naudoti Javascript'ą

```
<Komponentas atributas1={2+3} atributas2={someVar} atributas3={{a:
```

- Šiuo atveju atributas1 paduodama JS išraiška, ji bus įvykdyta
- atributas2 priskiriama JS išraiška su kintamuoju. Jeigu kintamasis egzistuos JS kode, jo reikšmė bus priskirta
- atributas3 tiesiog yra priskiriamas objektas {a: 'b'}



## PASTABOS

- JSX leidžia rašyti "html" žymes, bet kai kurie dalykai daromi kitaip.
  - Norint nurodyti CSS klasę, vietoj atributo `class` rašome `className`
  - komponentų atributai dažnai keičiasi į formą, panašią java'ai:
    - pvz. iš `background-url` į `backgroundUrl`



# KOMPONENTAS NAUDOJANT JSX

```
var HelloComponent = React.createClass({  
  render: function() {  
    return <div>Hello {this.props.name}</div>;  
  }  
});  
  
ReactDOM.render(React.createElement(  
  HelloComponent,  
  { name: "Jane" } ),  
  document.getElementById('root')  
);
```



# KOMPONENTAS NAUDOJANT JSX IR ES2015

```
class HelloComponent extends React.Component {  
  render() {  
    return <div>Hello {this.props.name}</div>;  
  }  
}  
  
ReactDOM.render(  
  <HelloComponent name="Jane"/>,  
  document.getElementById('root')  
) ;
```



## UŽDUOTIS 3

- pasibandyti visus pavyzdžius
  - su F12 pažiūrėti, ką jie sukuria
- JSX pabandyti sudėtingesnę HTML kodą
  - kas neveikia ar veikia ne taip?
    - klausti arba googlinti



# KOMPONENTŲ KOMPOZICIJA



# KOMPONENTŲ KOMPOZICIJA #1

```
var AvatarComponent = React.createClass({  
  render: function() {  
    return (  
      <img className="Avatar"  
        src={this.props.user.avatarUrl}  
        alt={this.props.user.name}  
      />  
    )  
  }  
});
```





## KOMPONENTŲ KOMPOZICIJA #2

```
var CommentComponent = React.createClass({
  render: function() {
    return (
      <div className="Comment">
        <div className="UserInfo">
          <AvatarComponent user={this.props.author} />
          <div className="UserInfo-name">
            {this.props.author.name}
          </div>
        </div>
      </div>
    );
  }
});
```



# KOMPONENTO ATRIBUTAI (PROPS) #1

- `this.props` - komponentui perduoti atributai
  - gali būti programuotojo sugalvoti
  - React'o persiunčiami pagal nutylėjimą (`props.children`)
- Papildomai kiekvienas komponentas gali nusakyti kokių atributų tikisi
- `PropTypes` rodo klaidas tik tada, kai naudojamas React development režimas

```
ProductCardComponent.propTypes = {  
  id: React.PropTypes.number.isRequired,  
  image: React.PropTypes.string.isRequired,  
  title: React.PropTypes.string.isRequired,  
  description: React.PropTypes.string.isRequired,  
  price: React.PropTypes.number.isRequired,  
};
```



## KOMPONENTO ATRIBUTAI (PROPS) #2

kaip tokį komponentą iškviesti:

```
<ProductCardComponent  
  key={index}  
  id={product.id}  
  image={product.image}  
  title={product.title}  
  description={product.description}  
  price={product.price}  
/>
```



# KOMPONENTAS SU ATRIBUTU

```
class HelloComponent extends React.Component {  
  render() {  
    return (<div>Hello {this.props.name}</div>);  
  }  
}  
  
HelloComponent.propTypes = {  
  name: PropTypes.string.isRequired  
}  
  
ReactDOM.render(  
  (<HelloComponent name="Jane"/>),  
  document.getElementById('root')  
);
```



## UŽDUOTIS 4

- pakeisti propTypes
  - iš `PropTypes.string` į `PropTypes.number`
  - ką rodo konsolė naršyklėje?
- pakeisti `name` : į `surname` :
- pakeisti `name="` į `vardas="`



## STILIAI

- ReactJs galima rašyti taip vadinamus `inline styles`, naudojant `style` atributą
- Vietoj to, kad aprašytume stilius CSS faile, jie yra rašomi Javascript'u
- Pagrindinė nauda - komponento stiliai įtakoja tik tą komponentą, kuriame jie aprašyti
- Ne viskas, kas įmanoma CSS'e, yra įmanoma `inline styles`



## STILIAI - PAVYZDYS

```
var styles = {  
  container: { background: 'red' },  
  greetingText: { color: 'green' }  
};  
var Component = React.createClass({  
  render: function() {  
    return (  
      <div style={styles.container}>  
        <p style={styles.greetingText}>Tekstas yra tokas</p>  
      </div>  
    );  
  }  
});
```



# NAUJO KOMPONENTO ŠABLONAS

```
class Component extends React.Component {  
  render() {  
    return (  
      <div>  
        <!-- Component view -->  
      </div>  
    );  
  }  
};  
  
Component.propTypes = {  
  // Properties JSON };
```





# ŠAKNINIO KOMPONENTO NUPIEŠIMAS

```
ReactDOM.render(  
  <Component prop1={prop1Value} />,  
  document.getElementById('root')  
);
```

- Dažniausiai būna vienoje vietoje aplikacijoje
- Vaikiniai komponentai automatiškai nusipiešia



## UŽDUOTIS 5 - PIRMIEJI KOMPONENTAI #1

- Užduočiai turėtų užtekti 2 failų sukūrimo - index.html ir app.js
- Galite pasiimti bet kokį paveikslėlį iš interneto
- Norint nurodyti CSS klasę, vietoj atributo `class` rašome `className`
- Sukurkite statinį produktų sąrašo vaizdą



## UŽDUOTIS 5 - PIRMIEJI KOMPONENTAI #2

- Sukurkite 2 komponentus:
  - ProductCardComponent - mokantis piešti vieną produkto kortelę
  - ProductListComponent - mokantis piešti daug produktų kortelių
- Vienas produktas sąraše: paveikslėlis, pavadinimas, kaina, mygtukas į detales
- Užpildykite vaizdą testiniais duomenimis
- Kortelėms galite naudoti

<https://getbootstrap.com/docs/4.1/components/card/>



## KAIP ĮGALINTI IMPORT'US ?

- vis tik norime būtų modernūs - naudoti ES6 import
  - kitiems JS moduliams, stiliams, paveikslukams
- reiktų modulių administratoriaus
- reiktų prijungti pvz. RequireJS patiems, jei norėtumėm CommonJS formato..
  - bet tam reiktų loaderių..
    - o tada reiktų modulių administratoriaus..
      - be to, per script naudojamas babel negali matyti failų sistemos, reiktų kitaip prijungti babel..
- .. ir t.t.

**STOP!**



## UŽDUOTIS 6

- Pirmąsias užduotis padarysime, kad žinotume, kaip tai veikia, bet daugiau patys link ir script neberašysime
- Toliau naudosime modernius įrankius
  - es2015, npm, babel, webpack
- CSS ir paveiksliukus importuosime su import

```
import './App.css'; // iš src/ katalogo
import pav from '../public/favicon.ico';
var style = { backgroundImage: 'url(' + pav + ')', width: 100, height:
var styleAlt = { backgroundImage: `url(${pav})`, width: 100, height:
var jsx = (<img src={favicon} className="klase-is-app-css"/>);
// arba tiesiog į CSS įdedam background-image: url(./logo.png);
```



## UŽDUOTIS 6 - BIBLIOTEKOS PER NPM

- Kaip pasileisti

```
$ apt-get install nodejs-legacy npm
$ npm install -g create-react-app
$ create-react-app hello-world
# Pridedam bootstrap priklausomybę į package.json dependencies
# "bootstrap": "4.1.3",
# "jquery": "1.9.1",
# "popper.js": "1.14.3"
$ npm install
$ npm run build
$ npm start
```

- "Sugadinti" projektą, kad atsirastų griaučiai:

```
$ npm run eject
```

- Pastaba: atgal grįžti nepavyks, tai tik TESTAS, ir neskirta realiam projektui



## UŽDUOTIS 6 - BIBLIOTEKOS PER NPM

- Windows ypatingai svarbu versijos

```
$ node --version // v8.10.0  
$ npm --version // 3.5.2  
$ create-react-app --version // 2.1.1
```

- NPM dar reikalauja ir windows build tools / linux build-essential

```
$ npm install -g windows-build-tools // windows  
$ apt install build-essential // linux
```

- pasidalinkite su kitais, kurios node ir npm versijos jums veiks Windows
- taip pat gali tekti leisti iš Git Bash. Ir įsidėti node/npm katalogą į environment variables rankiniu būdu



# PATIKRINTI AR VEIKIA ES2015/ES6 IR BABEL'IS

## Modulis/Modulis.js - kataloge Modulis failas Modulis.js

```
class Polygon {  
  constructor(height=2, width=3) {  
    this.height = height;  
    this.width = width;  
  }  
  get area() {  
    return this.calcArea()  
  }  
  calcArea() {  
    return this.height * this.width;  
  }  
}  
export default Polygon;
```

## index.js

```
import Polygon from './Modulis/Modulis';  
console.log(new Polygon().calcArea());
```





## PATIKRINTI, AR VEIKIA JEST TESTAI

- npm test paleidžia App.test.js per Jest

```
import Polygon from './Modulis/Modulis';  
it('calculates area correctly', () => {  
  expect(new Polygon().calcArea()).toEqual(6);  
});
```

- <https://github.com/facebookincubator/create-react-app/blob/master/packages/react-scripts/template/README.md#running-tests>



# PATIKRINTI AR VEIKIA BOOTSTRAP

index.js

```
import '../node_modules/bootstrap/dist/css/bootstrap.min.css';
```

app.js

```
<p><button className="btn btn-primary" role="button">Reload</button><
```



# KITOJE PASKAITOJE

Javascript moduliai. Arrow funkcijos. Map/filter/Reduce.  
Kolekcijos

