Client-Side Libraries & Framework

IF3110 – Web-based Application Development
School of Electical Engineering and Informatics
Institut Teknologi Bandung

Motivation

- Java Script is becoming THE main technology for building the future Internet (i.e., Internet of Things)
- Java Script is a “strange” and flexible language
  - Type
  - Code Style
  - Architecture
  - Handler

Framework vs Library

Framework – this describes a given structure of "how" you should present your code, such as a code-template, along some helpers, constructors etc. to solve/simplify a specific problem or bring your architecture in "order".

Examples, ”ReactJS", ”AngularJS", ”VueJS”.

Library – a collection of function that we can call which highly abstracts different layers, like browsers / DOM models / etc. In addition to be a good toolkit, it offers a lot of tools and neat stuff to work with, which in general, simplify your coding experience.

Examples "jQuery", "MooTools", "YUI"

Rational to Use

- Don’t reinvent the wheel
- Do more with less code
- Save time
- It is likely; we are not THE EXPERT
- Embrace community for your code’s sustainability
- Enforce norms (structure/architecture/convention) among your team members

The good thing about frameworks is…

they do lots of things for developers

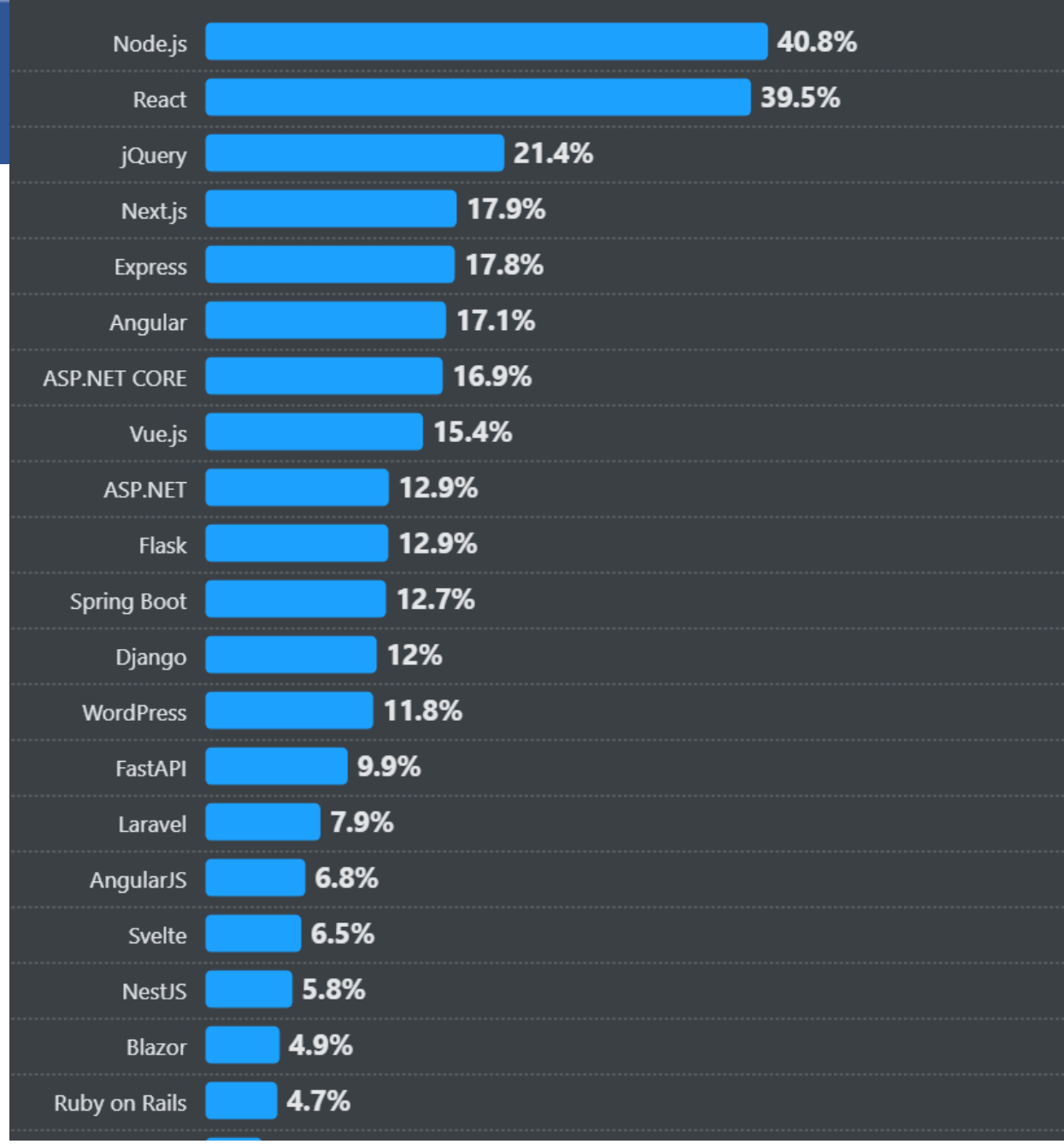
… and the bad thing about frameworks

they do lots of things for developer

Popular Web Framework

Source

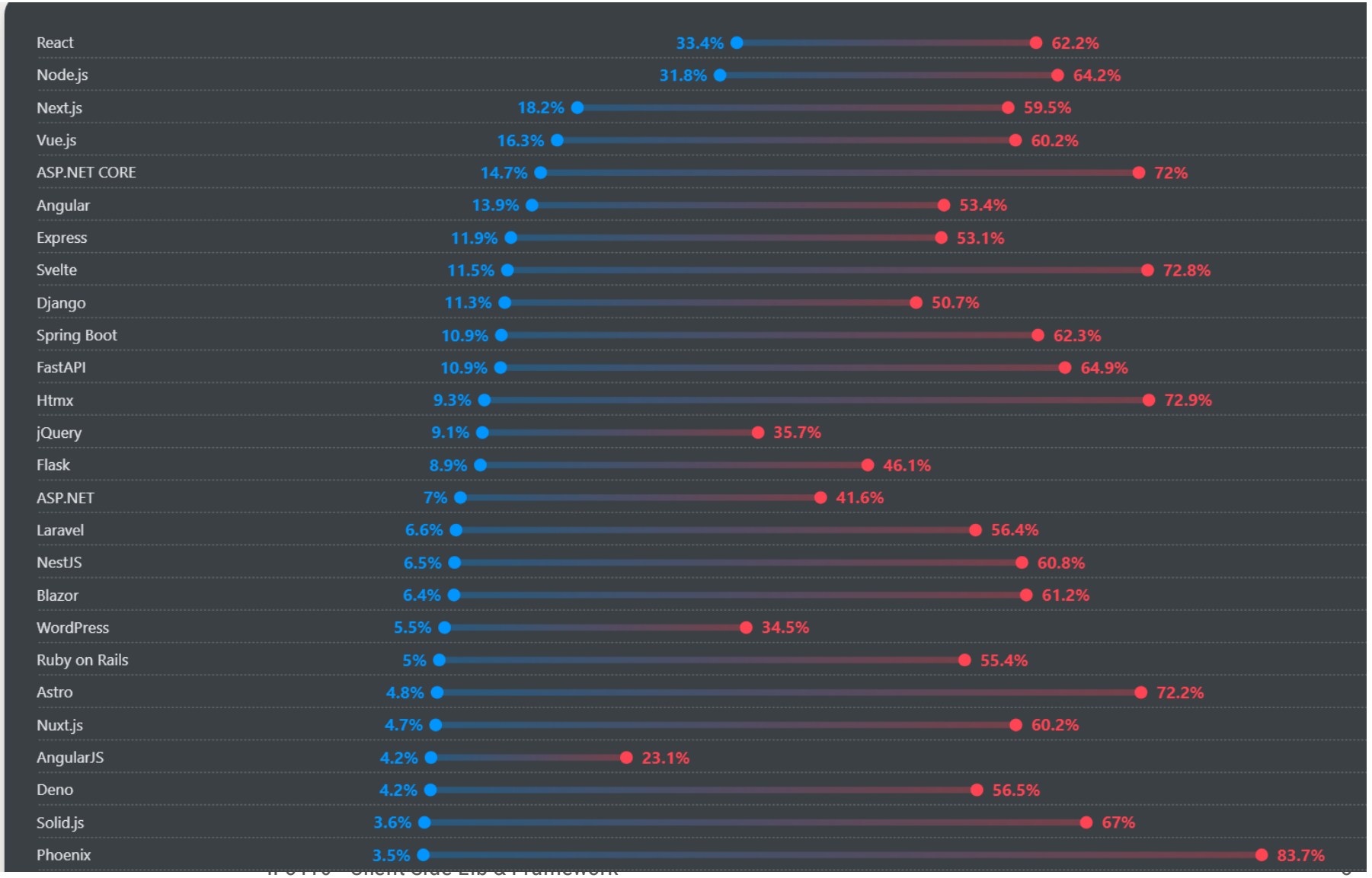
Stackoverflow Developer Survey 2024
https://survey.stackoverflow.co/2024/technology



Web Framework – Admired vs Desired

Source

Stackoverflow Developer Survey 2024
https://survey.stackoverflow.co/2024/technology



jQuery

jQuery

- Library Javascript, website: http://www.jquery.com
- Memiliki ukuran kecil (247K ~ 28K minified and compressed)
- Fitur
  - Memudahkan akses dan manipulasi DOM dengan fungsi $()
  - AJAX
  - Event handling
- Include:

```
<script type="text/javascript" src="jquery.js"></script>
```

jQuery: $()

$(): mengembalikan elemen atau array elemen, dengan menggunakan string selector dengan format mirip CSS selector

Contoh	Return
$('a')	elemen dengan tag a
$('#myid')	elemen dengan id myid
$('div#divStatus')	elemen dengan tag div dan id divStatus
$('p.content')	elemen dengan tag p dan class content

Berbagai method dari jQuery objects dapat dipanggil, misalnya:

- text(), html(), hide(), show()
- fadeIn(), fadeOut(), slideIn(), slideOut(), animate()
- click(), dblclick(), mouseenter(), mouseleave()

DOM Manipulation with jQuery

Common DOM Operations:

- .append() - Add content at the end
- .prepend() - Add content at the beginning
- .after() - Insert after element
- .before() - Insert before element
- .remove() - Remove elements
- .empty() - Remove child elements
- .html() - Get/Set HTML content
- .text() - Get/Set text content

AJAX with jQuery

AJAX Methods:

- $.ajax() - Low-level interface
- $.get() - GET request
- $.post() - POST request
- $.getJSON() - Load JSON data
- load() - Load HTML into element
- Promise callbacks (.done, .fail, .always)
- Error handling
- Data types support

jQuery: $.get()

$.get(): memanggil Ajax request dengan get

```
$.get("path/to/data.php?name=value",
      function (sData, sStatus) {
         alert(sStatus + ":" + sData);
      });
$.get("path/to/data.php", { name: "value" },
      function (sData, sStatus) {
          alert(sStatus + ":" + sData);
});
```

Callback parameter: Data & Status

jQuery: $.post()

$.post() serupa dengan $.get()

```
$.post("path/to/data.php", { name: "value" },
        function (sData, sStatus) {
                alert(sStatus + ":" + sData);
        }
);
```

jQuery: $.ajax()

$.ajax(): low level method untuk Ajax request

Menerima 1 parameter berupa object (associative array)

- type: get atau post
- url: url request
- data: url encoded data
- dataType: tipe data response: script, xml, html, json
- success: callback untuk success
- error: callback error
- complete: callback untuk complete

```
// jQuery (Library) - You call it when needed
$("#button").click(function() {
  $.ajax({
        url: "api/data",
        success: function(result) {
            $("#div").html(result);
        }
    });
});
```

jQuery vs Vanilla JavaScript

jQuery

- Shorter syntax
- Cross-browser support
- Built-in animations
- Simplified AJAX
- Plugin ecosystem

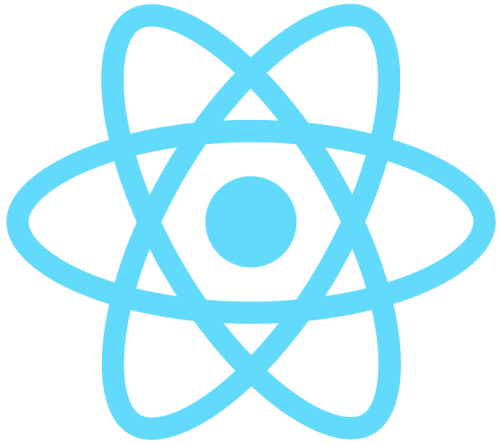
Vanilla JavaScript

- Better performance
- No dependencies
- Modern API support
- Smaller file size
- Direct browser APIs

Best Practices and Performance Tips

- Cache jQuery selections
- Use proper selectors
- Avoid global variables
- Utilize event delegation
- Minimize DOM manipulation
- Use chaining when possible
- Combine multiple operations
- Optimize animations

ReactJS



Intro to ReactJS

- React: Javascript library untuk membangun user interface
- React menjadi V dari MVC
- React dibuat oleh Facebook.
- Facebook dan Instagram menggunakan React.

Keunggulan React

- Simpel untuk melakukan update DOM saat data berubah
- Proses update DOM efisien. React melakukan **reconciliation** saat memperbarui DOM.

Core Concepts

Virtual DOM

- Lightweight copy of actual DOM
- Efficient updating mechanism
- Reduces direct DOM manipulation
- Improves performance

Component Architecture

- Reusable UI pieces
- Modular development
- Better code organization
- Easy maintenance

Including React in HTML

```
<script src="https://fb.me/react-0.14.2.js"></script>
<script src="https://fb.me/react-dom-0.14.2.js"></script>
```

Virtual DOM

- Mengubah DOM dapat menjadi operasi yang mahal.
- ReactJS mengoptimasi proses pembaruan DOM dengan menggunakan virtual DOM.
- Operasi pada ReactJS mengubah virtual DOM, bukan DOM browser.
- Saat akan memperbarui tampilan, ReactJS membandingkan virtual DOM yang lama dengan yang baru dan menentukan apa saja yang perlu diubah di DOM browser. Proses ini disebut rekonsiliasi.

JSX

- JSX: a XML-like syntax extension to ECMAScript without any defined semantics
- React dapat menggunakan plain Javascript atau JSX
- Direkomendasikan menggunakan JSX, karena lebih mudah mendefinisikan tree dan kodenya lebih mudah dibaca.

Dengan JSX

```
var LikeComponent = React.createClass({
    getInitialState: function(){
        return {liked:false};
    },
    handleClick: function(){
        this.setState({liked: !this.state.liked});
    },
    render: function() {
        var text = this.state.liked ? 'like' : "don't like";
        return (
            <p onClick={this.handleClick}>
                You {text} {this.props.name}. Click to toggle.
            </p>
        )
    }
});
```

Tanpa JSX

```
'use strict';

var LikeComponent = React.createClass({
    displayName: 'LikeComponent',

    getInitialState: function getInitialState() {
        return { liked: false };
    },
    handleClick: function handleClick() {
        this.setState({ liked: !this.state.liked });
    },
    render: function render() {
        var text = this.state.liked ? 'like' : "don't like";
        return React.createElement(
            'p',
            { onClick: this.handleClick },
            'You ',
            text,
            ' ',
            this.props.name,
            '. Click to toggle.'
        );
    }
});
```

Dengan JSX

```
var likeElement = <LikeComponent name='pizza'/>

ReactDOM.render(likeElement, mountNode);
```

Tanpa JSX

```
var likeElement = React.createElement(LikeComponent, { name: 'pizza' });

ReactDOM.render(likeElement, mountNode);
```

Dengan JSX

```
var App = (
  <Form>
    <FormRow>
      <FormLabel />
      <FormInput />
    </FormRow>
  </Form>
);
```

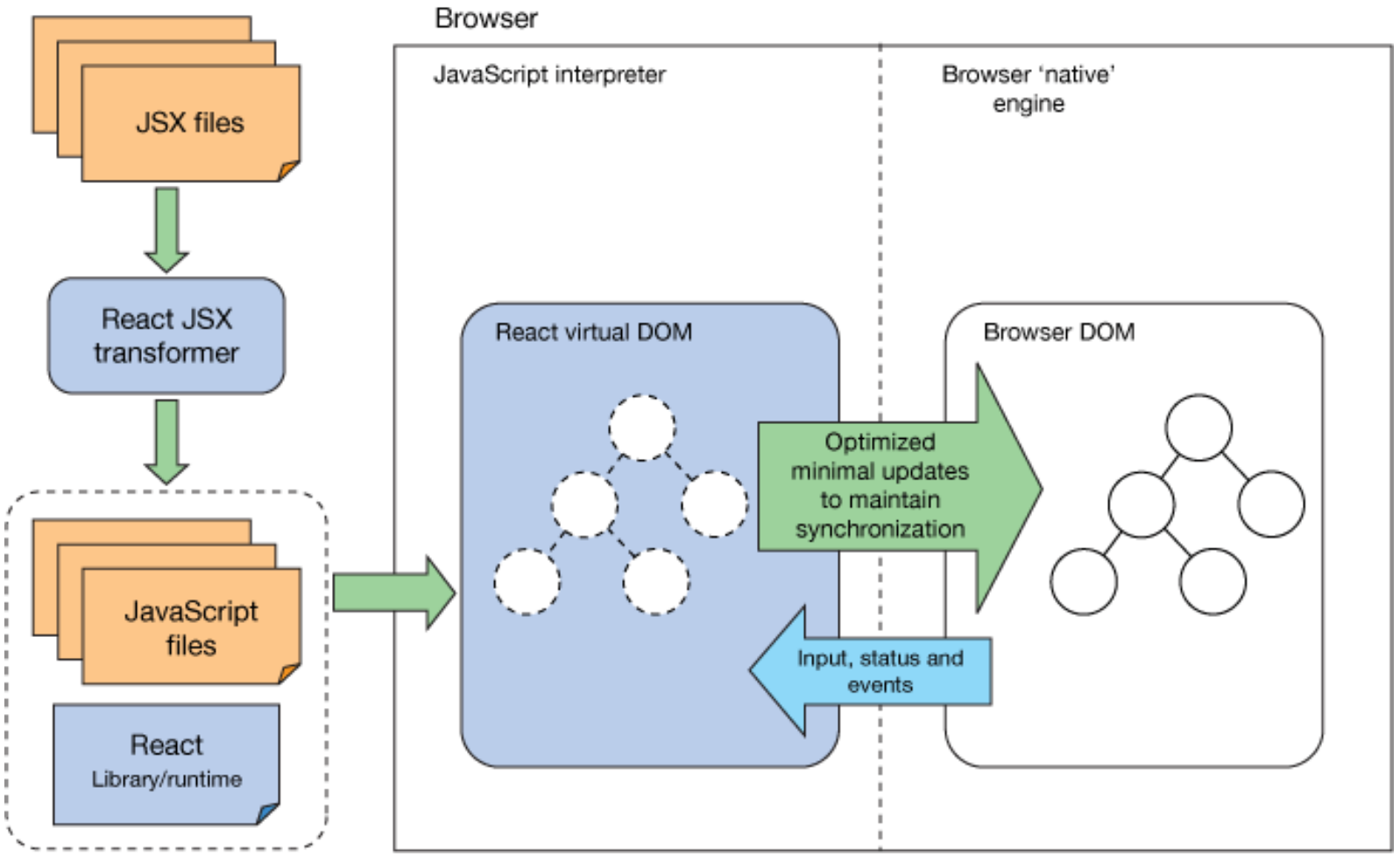
Tanpa JSX

```
var App = (
  React.createElement(Form, null,
    React.createElement(Form.Row, null,
      React.createElement(Form.Label, null),
      React.createElement(Form.Input, null)
    )
  )
);
```

JSX

- JSX harus di-compile sebelum dapat dieksekusi oleh browser.
- Salah satu compiler yang dapat digunakan: BabelJS
- BabelJS dapat meng-compile JSX langsung di browser atau sebelum di-load di browser
- Untuk production JSX tidak disarankan di-compile di browser

How it works


Browser
JavaScript interpreter
Browser 'native' engine
JSX files
React JSX transformer
JavaScript files
React
Library/runtime
React virtual DOM
Browser DOM
Optimized minimal updates to maintain synchronization
Input, status and events

React Element

React Element : Analog dengan elemen HTML

- DOM pada browser terdiri dari elemen HTML
- Virtual DOM terdiri dari React Element

Supported Tag:

```
a abbr address area article aside audio b base bdi bdo big blockquote body br
button canvas caption cite code col colgroup data datalist dd del details dfn
dialog div dl dt em embed fieldset figcaption figure footer form h1 h2 h3 h4 h5
h6 head header hr html i iframe img input ins kbd keygen label legend li link
main map mark menu menuitem meta meter nav noscript object ol optgroup option
output p param picture pre progress q rp rt ruby s samp script section select
small source span strong style sub summary sup table tbody td textarea tfoot th
thead time title tr track u ul var video wbr
```

Contoh React Elemen

Tanpa JSX

```
var title = React.createElement('h1', {}, 'Hello world');

ReactDOM.render(title, document.getElementById('example'));
```

Dengan JSX

```
var title = <h1>Hello World!</h1>;

ReactDOM.render(title, document.getElementById('example'));
```

React Components

- React Components: Custom React Element
- React Component dibuat dengan memanggil fungsi React.createClass(object specification)
- Masukan fungsi createClass adalah JavaScript object yang berisi berbagai fungsi yang mengatur spesifikasi dan lifecycle React Components
- Nama variable Component harus diawali huruf kapital

Contoh React Component

Huruf Kapital

Fungsi render

```
var Comment = React.createClass({
    render: function() {
        return(
                <div>
                    <h3>Judul komentar</h3>
                    <p>Isi komentar</p>
                </div>
        );
    }
});

var title = <Comment />;

ReactDOM.render(title, document.getElementById('example'));
```

React Components

Ada beberapa fungsi yang mengatur spesifikasi dan lifecycle React Components.

Fungsi terkait spesifikasi komponen:

- render
- getInitialStates
- getDefaultProps
- displayName

Props dan State

- Setiap komponen dapat mempunyai props atau state.
- Props digunakan untuk memberikan data dan event handler kepada child components
- State digunakan untuk menyimpan data yang diperlukan oleh komponen.
- Perubahan pada props dan state mengakibatkan komponen tersebut direnderer ulang.

Props dan State

Props

```
var likeElement = <LikeComponent name='pizza'/>
```

```
var LikeComponent = React.createClass({
    getInitialState: function(){
        return {liked:false};
    },
```

State

Perbedaan Props dan State

Props

- Immutable
- Better performance
- Use this to pass data or event handler to child components
- Use case: component configuration

State

- Mutable
- Worse performance
- Don’t access this from child component
- Component-specific
- Use case: Triggers re-renderes

Fungsi Render

```
ReactDOM.render(title, document.getElementById('example'));
```

Fungsi ReactDOM.render menerima masukan virtual DOM dan DOM browser

Saat fungsi tersebut dipanggil, proses rekonsiliasi dimulai

Rekonsiliasi

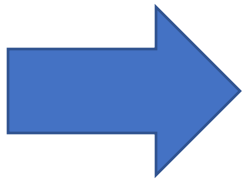
- Rekonsiliasi adalah proses dimana React melakukan update DOM browser
- Saat fungsi ReactDOM.render() dipanggil, React membandingkan virtual DOM yang baru dengan dengan yang lama.
- Dari perbedaan ini, React menentukan bagaimana mengubah DOM browser agar proses update efisien.
- Setelah itu, DOM pada browser diubah dengan cara yang telah ditentukan pada langkah sebelumnya
- Walaupun terlihat dua kali kerja, proses ini lebih cepat dibandingkan dengan langsung mengubah DOM browser.
- Perubahan DOM pada browser membuat browser harus menghitung ulang margin, padding, dll, dan mengakibatkan perubahan DOM menjadi lambat.

Proses Rekonsiliasi

Ketika mengupdate komponen, React berasumsi bahwa komponen yang sama akan menghasilkan pohon yang sama. Ini berpengaruh ke bagaimana React mengupdate DOM browser.

Proses Rekonsiliasi

```
// Render Pass 1
<Card>
  <p>Paragraph 1</p>
  <p>Paragraph 2</p>
</Card>
```



```
<Card>
  <p>Paragraph 2</p>
</Card>
```

Secara intuitif, `<p>Paragraph 1</p>` dihapus dan `<p>Paragraph 2</p>` dibiarkan

Namun, React tidak menghapus `<p>Paragraph 1</p>`, melainkan mengubahnya menjadi `<p>Paragraph 2</p>` dan menghapus `<p>Paragraph 2</p>`

Perilaku ini dapat menimbulkan masalah untuk komponen yang mempunyai state.

Proses Rekonsiliasi

Untuk menghindari masalah dengan komponen yang mempunyai state, berikan props key untuk komponen tersebut.

Jika komponen diberikan key, React menjamin komponen tersebut tidak akan di-reuse melainkan hanya diurutkan ulang atau dihancurkan.

```
<Message content="Hi..." key="somekey" />
```

Performance Optimization

- Memoizing
  - React.memo
  - useMemo
  - useCallback
- Code Splitting
- Lazy Loading
- Virtual Lists

Referensi ReactJS

https://reactjs.org/docs

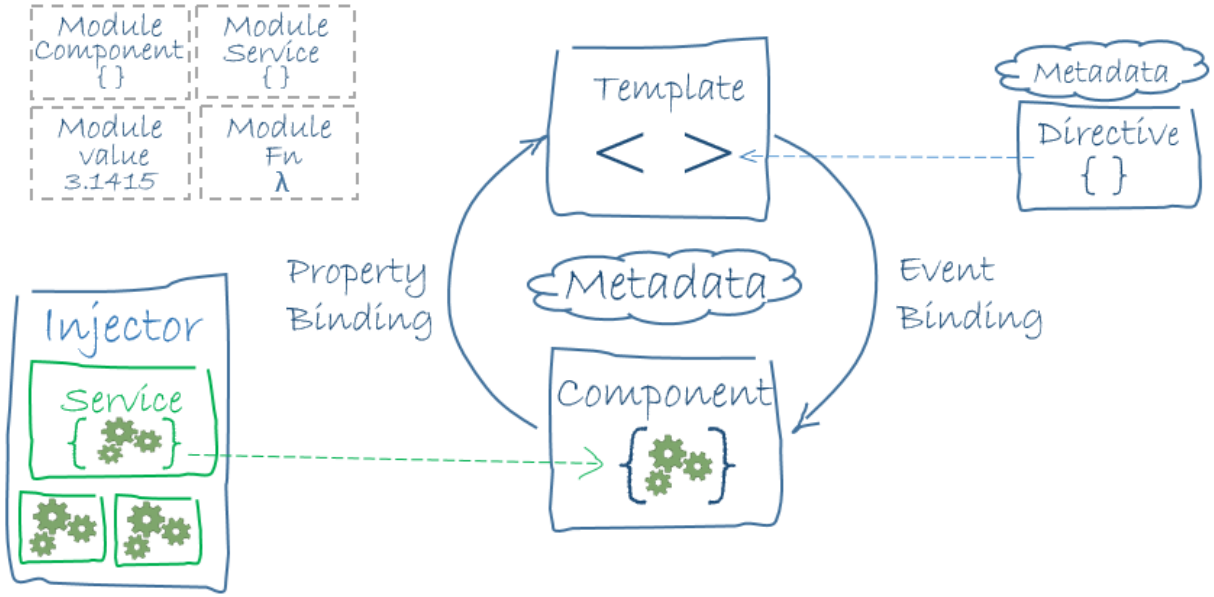
https://www.digitalocean.com/community/tech-talks/getting-started-with-react

http://conferences.oreilly.com/fluent/fluent2014/public/schedule/detail/32395

http://stackoverflow.com/questions/21109361/why-is-reacts-concept-of-virtual-dom-said-to-be-more-performant-than-dirty-mode

Angular 2+

- Using TypeScript (a language developed & maintained by Microsoft)
- Maintained and led by team at Google
- Very rapid releases
  - Version 2 (September 14, 2016) ~ Version 12 (May 12, 2021)
- Angular does not have a concept of "scope" or controllers; instead, it uses a hierarchy of components as its primary architectural characteristic.



https://en.wikipedia.org/wiki/Angular_(web_framework)

Evolution from AngularJS to Angular 2+

- AngularJS (Angular 1.x)
  - Released in 2010
  - JavaScript-based framework
  - Two-way data binding

- Angular 2+ (Complete Rewrite)
  - Released in 2016
  - TypeScript-based framework
  - Component-based architecture
  - Improved performance
  - Better mobile support
  - Enhanced tooling