Webtech@scopevisio

Choosing your webtech poison

Jörg Haas | Stepan Rutz | 2019

SCOPEVISIO

Web Technologien Trends

Online Informationsquellen:

- Slashdot
- Dzone
- Syntax.fm
- Stackoverflow (Developer Survey)
- IRC
- Github + https://githut.info/

Top-Sprachen im Laufe der Zeit

Ubiquitous JavaScript

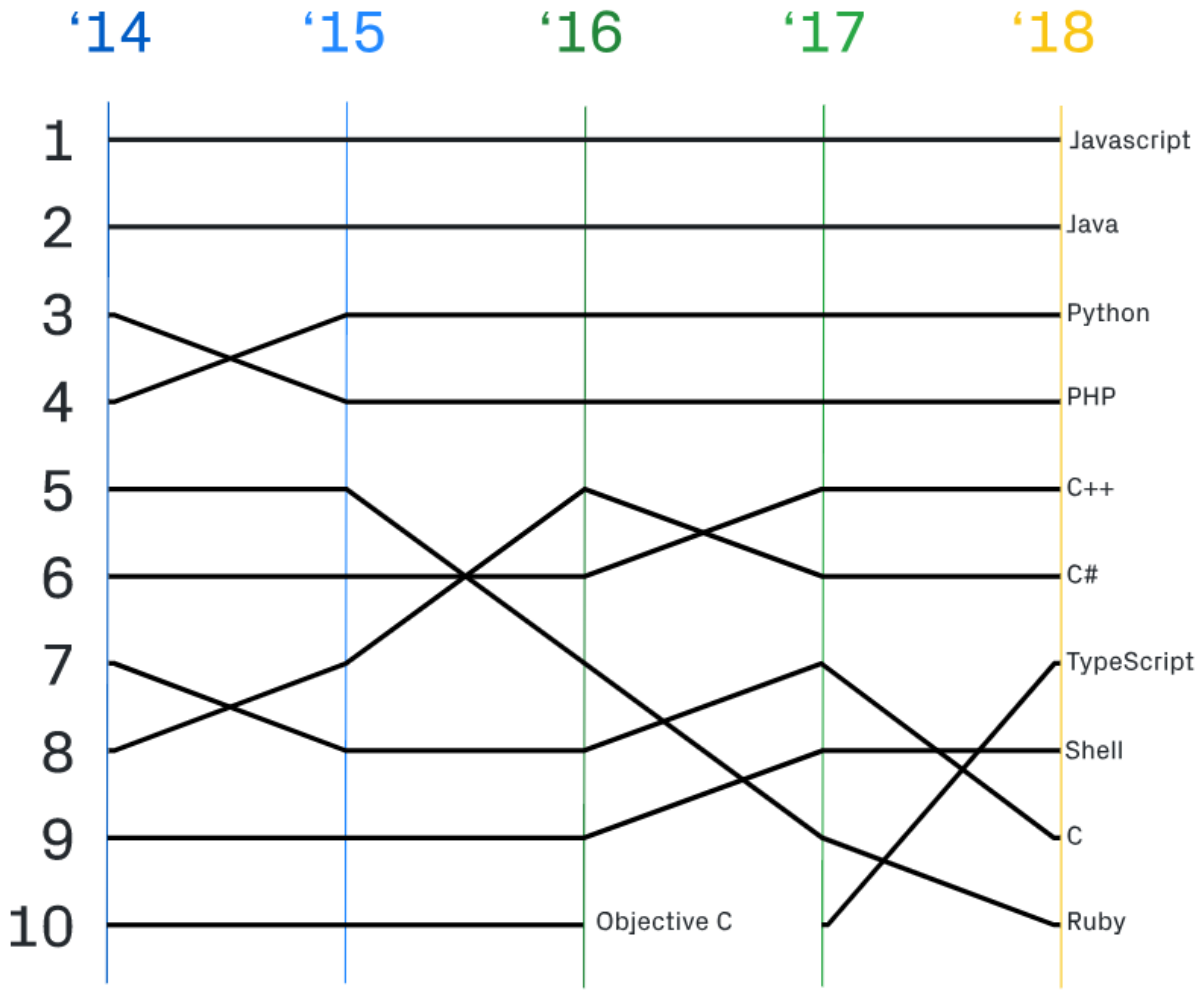
"Software is eating the world, the web is eating software, and JavaScript rules the web."

https://octoverse.github.com/projects#languages

„Stack Overflow 2018 report states that 69.8% of their 78,334 respondents-developers use JavaScript”

It can do Frontend and Backend

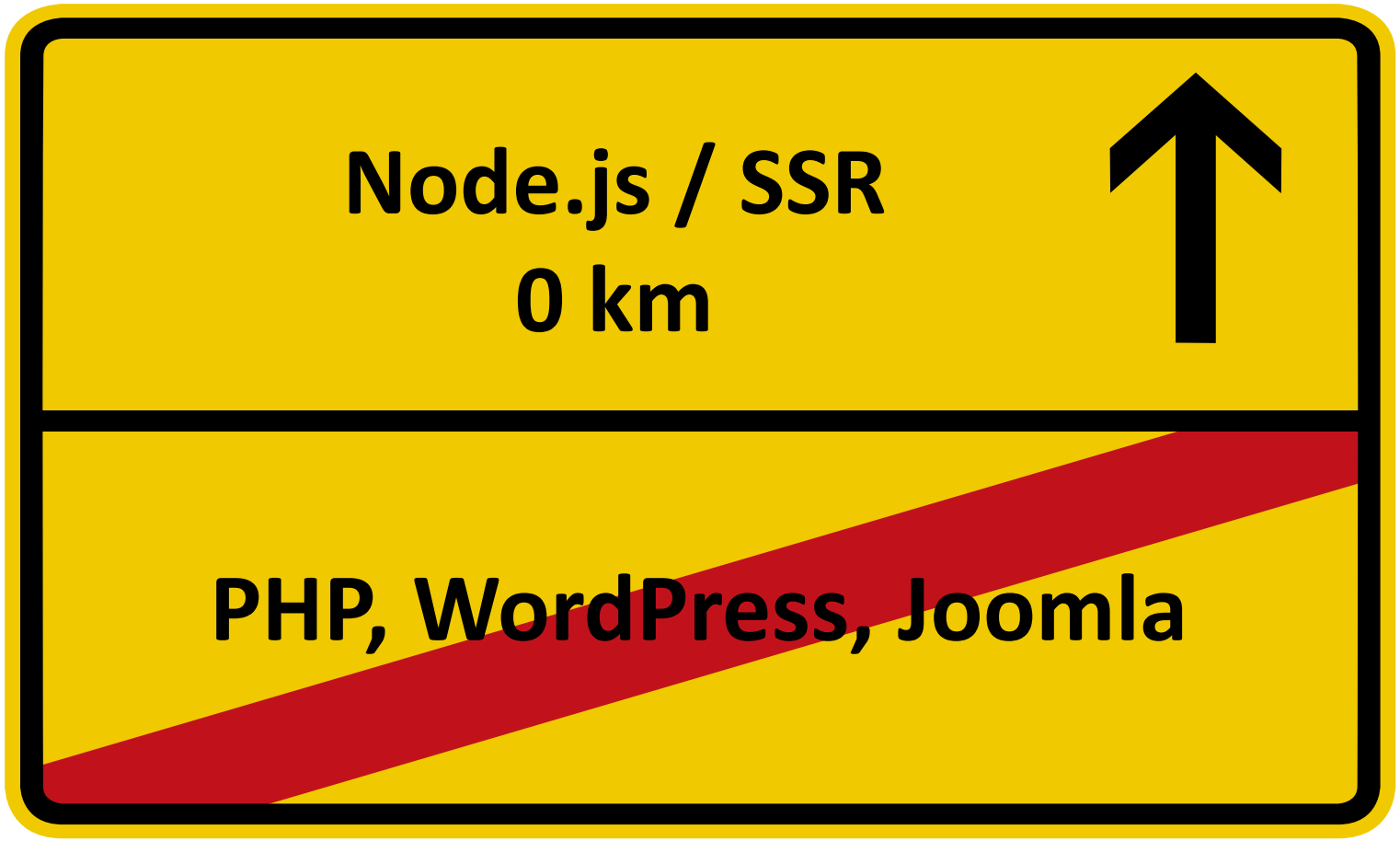
Server : **node.js** killed Php
Mobile : **Hybrid Apps** killed Swift, ObjectiveC, Android-Nativ



General Trends

1. CMS Systems
2. APIs and Backends
3. Richclients
4. Mobile Apps

General Trends: Websites + eCommerce



General Trends: Websites + eCommerce

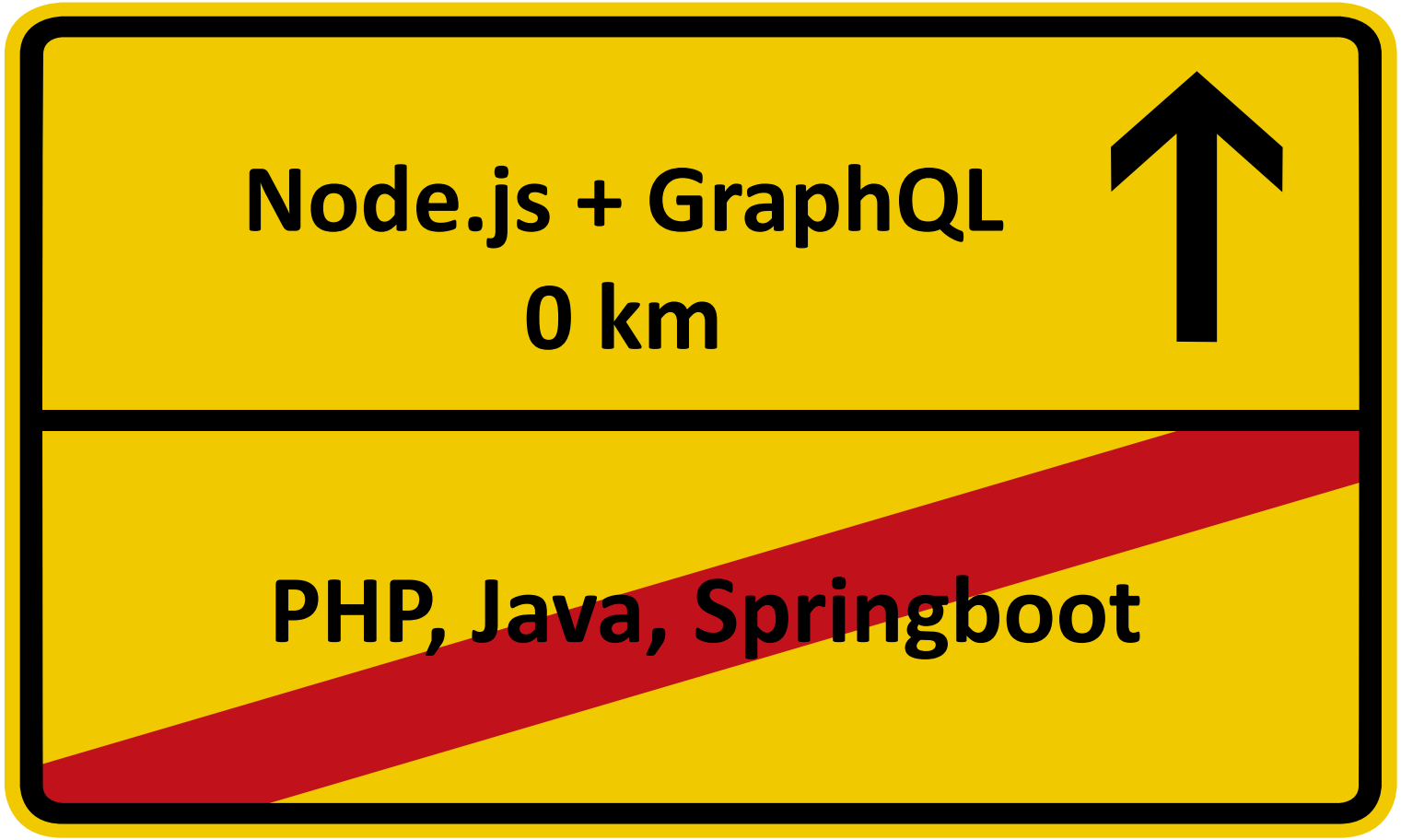
Node.js + SSR bedeutet zum Beispiel:

- Gatsby.js
- Next.js
- Nuxt.js
- React-Static
- Wintersmith
- Gridsome

… mit Technologien die Entwickler (heute) lieben:

Webpack, Javascript, GraphQL, Vue.js, React.js, PWA, HMR

General Trends: Backends + APIs



General Trends / Backends + APIs

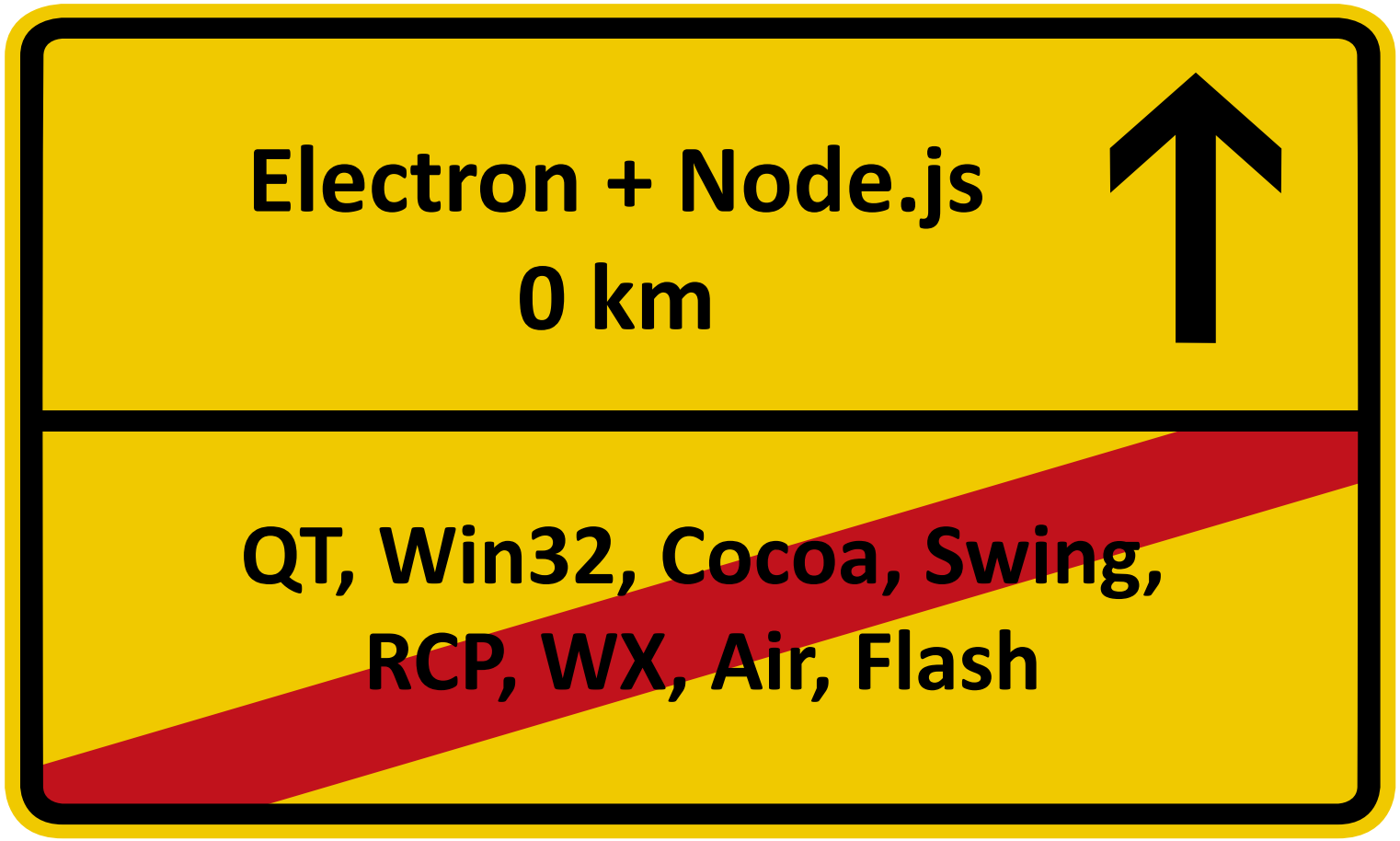
Serverside … bedeutet zum Beispiel:

GraphQL, Node.js, Express

… mit Technologien die Entwickler (heute) lieben:

Node.js, Typescript, GraphQL

General Trends: Richclient Applications



General Trends: Mobile Apps

Summary

Javascript + HTML everywhere

Richclient, Web, Mobile, Backends, CMS, Games

Scopevisio Decision

Webtech Overview

4 Layer-Bereichen werden unterschieden:

1. **Base Tech-Layer**
   Gleich/ähnlich für alle Webtech Anwendungen aller Ausprägungen.

2. **Design Layer**
   Design via CSS/Saas und Css-Frameworks

3. **Application Layer**
   Basic Ecosystems and Higher Level Frameworks

4. **Special Purpose Frameworks**
   Cordova (Mobile), Electron (Desktop Apps), Nuxt/Gatsby
   (Mostly Static Sites, eCommerce Sites and Blog Sites)

1 / 4 Base Tech Layer

Besteht aus:

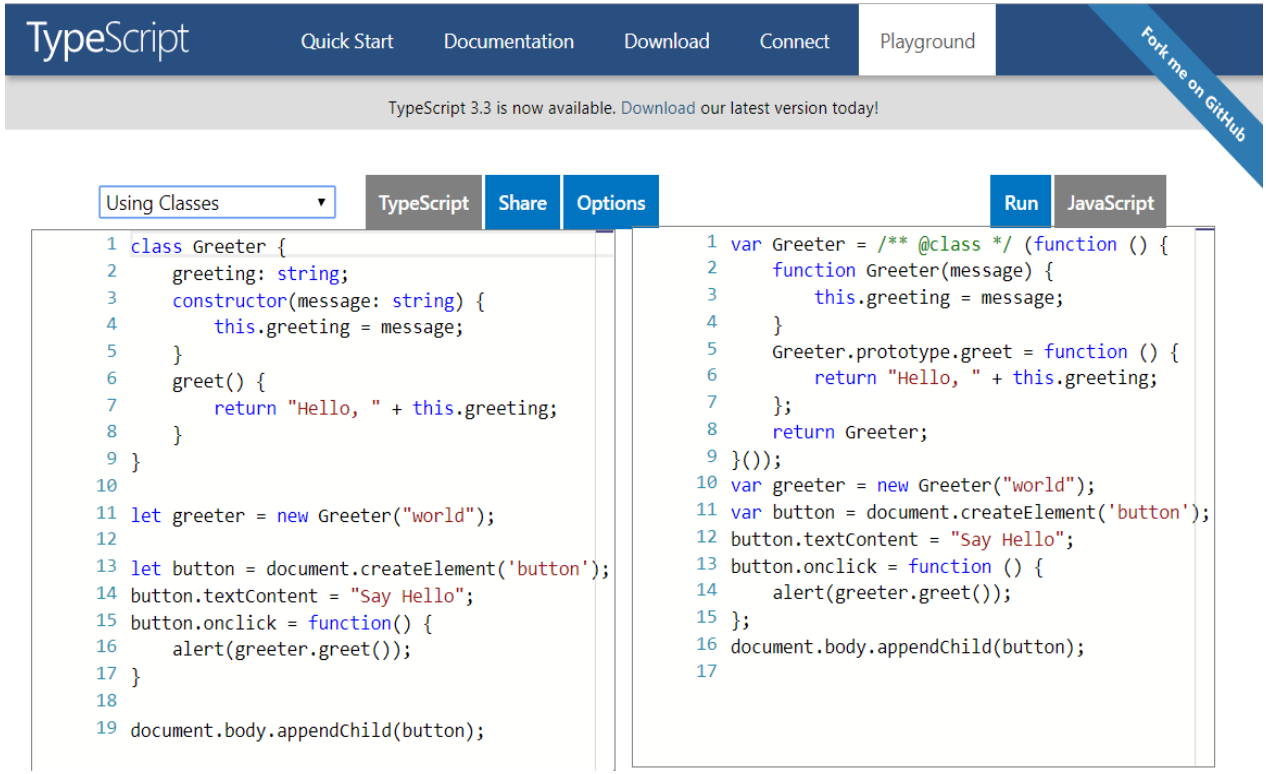
- Typescript
- NPM
- Webpack
- Loadash
- VisualStudioCode (VSCode)

Die Verwendung von ES6 Modules ist zwingend vorgeschrieben.

1 / 4 Base Tech Layer „TypeScript“

Typescript ist:

- Typsicheres Javascript
- ES6 Modules (down to ECMAScript 3)
- Static verification
- Transpiler zu älteren JS Versionen (down to ECMAScript 3)
- Enthält Polyfills für Async/Await, Enums, Decorators etc
- Verwendbar in Node.js Backends
- Typings for existing libraries and frameworks



Ersetzt folgende Dinge: ~~Babel.js, ESLint, require.js~~

1 / 4 Base Tech Layer „NPM“

Node Package Manager (NPM) liefert:

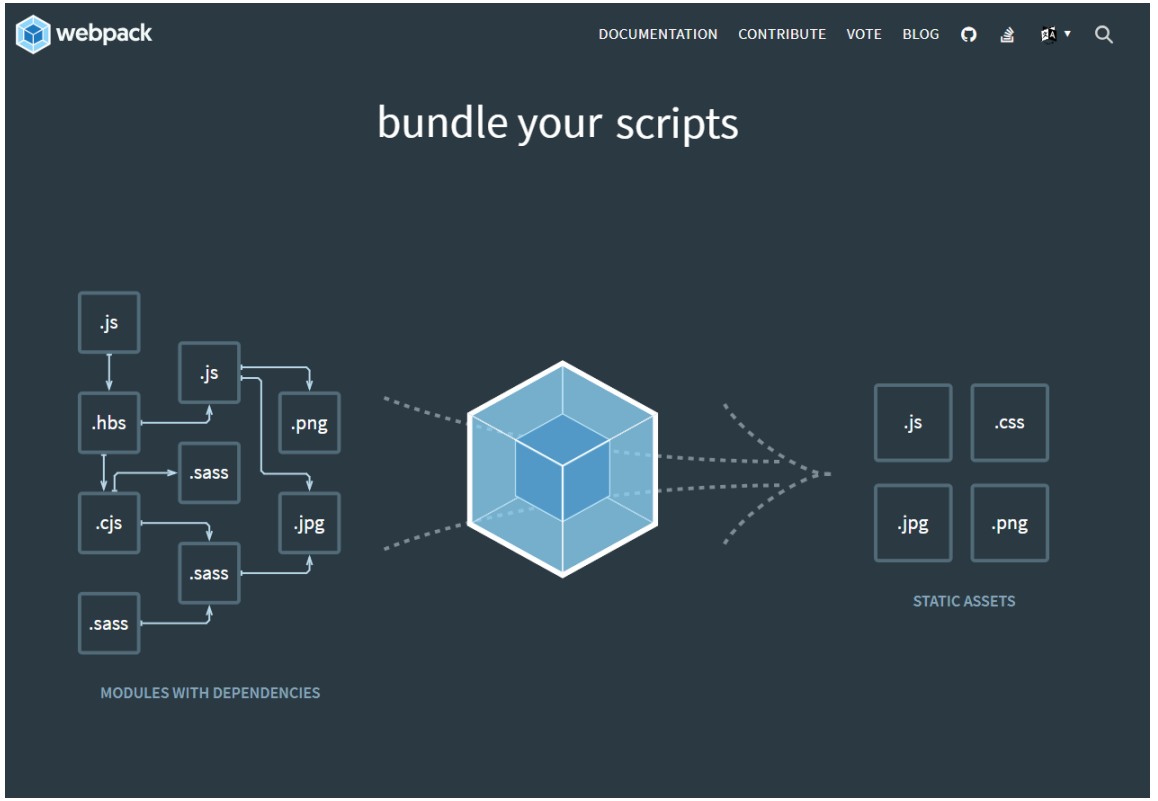
- Standard installation for developers via
  - Npm install
  - Npm run
  - npm watch
  - …
- Standard installation of development dependencies, run dependencies and Typescript-@Typings
- Node.js standard

Replaces: ~~Grunt, Make~~

1 / 4 Base Tech Layer

Webpack ist der Packager erfordert ES6 Modules und liefert eines oder mehrere der folgenden Dinge:

- Packaging
- Watchmode
- Build CSS from Saas
- Build JS from Typescript
- HMR (Hot Module Reloading) + Devserver
- Asset Management
- Code Splitting, Lazy loading, Tree shaking
- PWAs



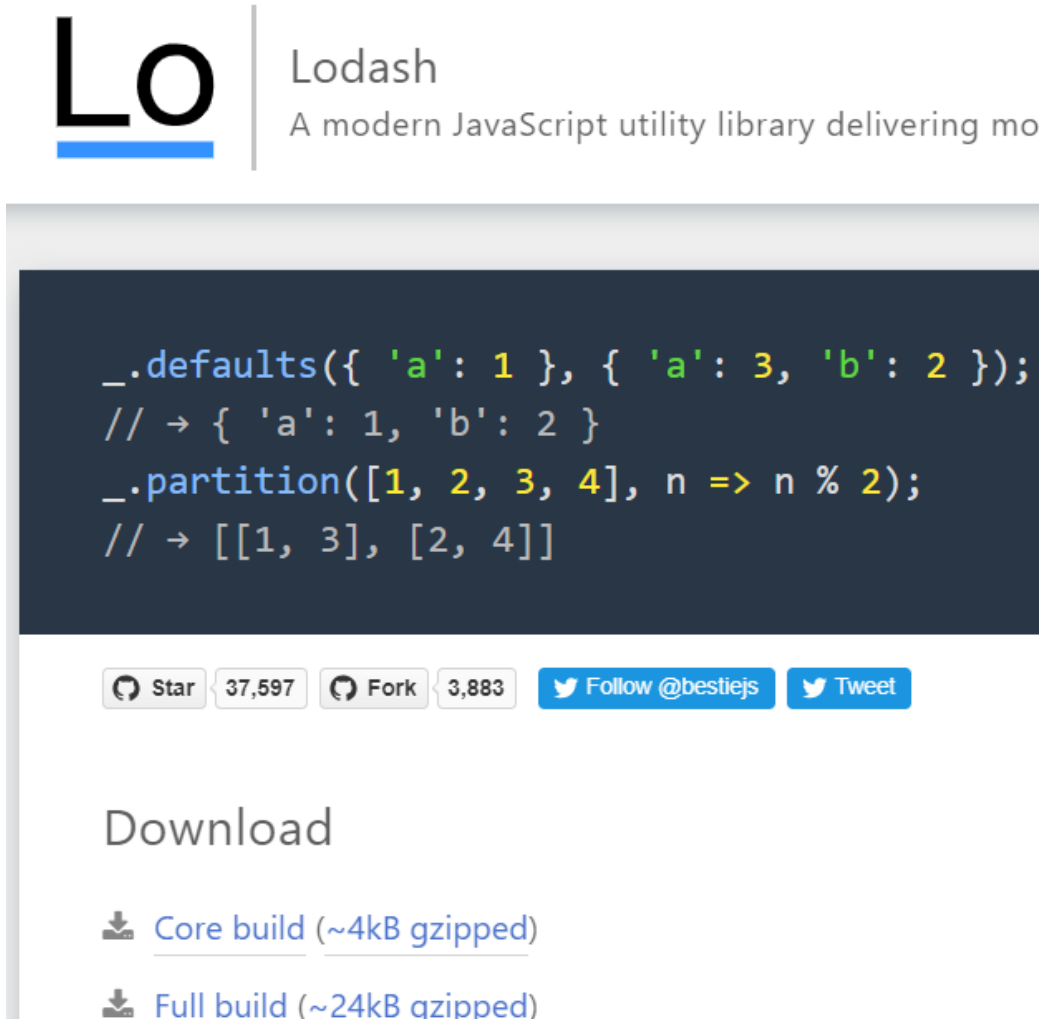
Replaces: ~~Make, Gulp, Grunt, Broccoli, Brunch, Devservers, Uglyfiers~~

1 / 4 Base Tech Layer „Loadash“

Loadash, als JavaScript utility library

- Utility library
- Modularization
- Polyfills

Replaces: ~~jQuery~~

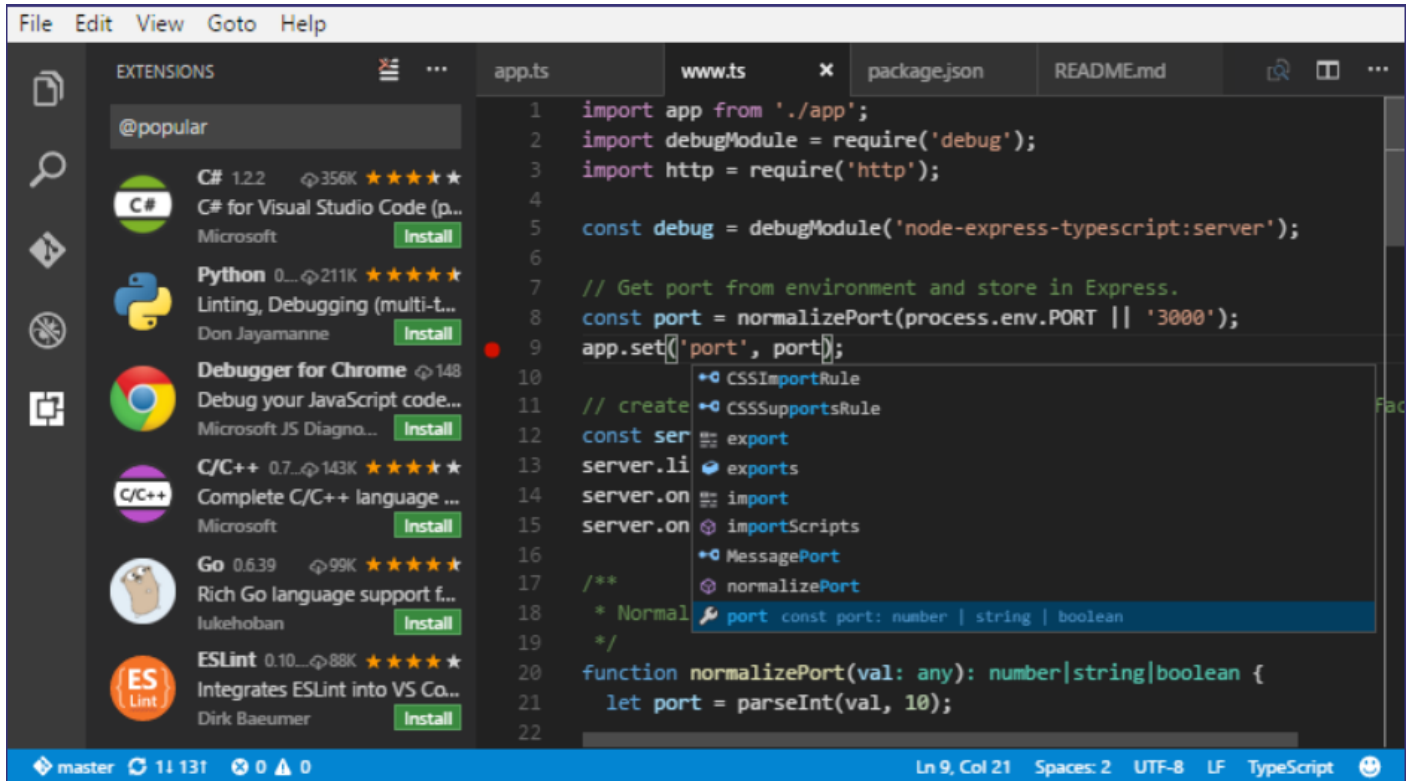


1 / 4 Base Tech Layer „VSCode“

VisualStudioCode (VSCode)

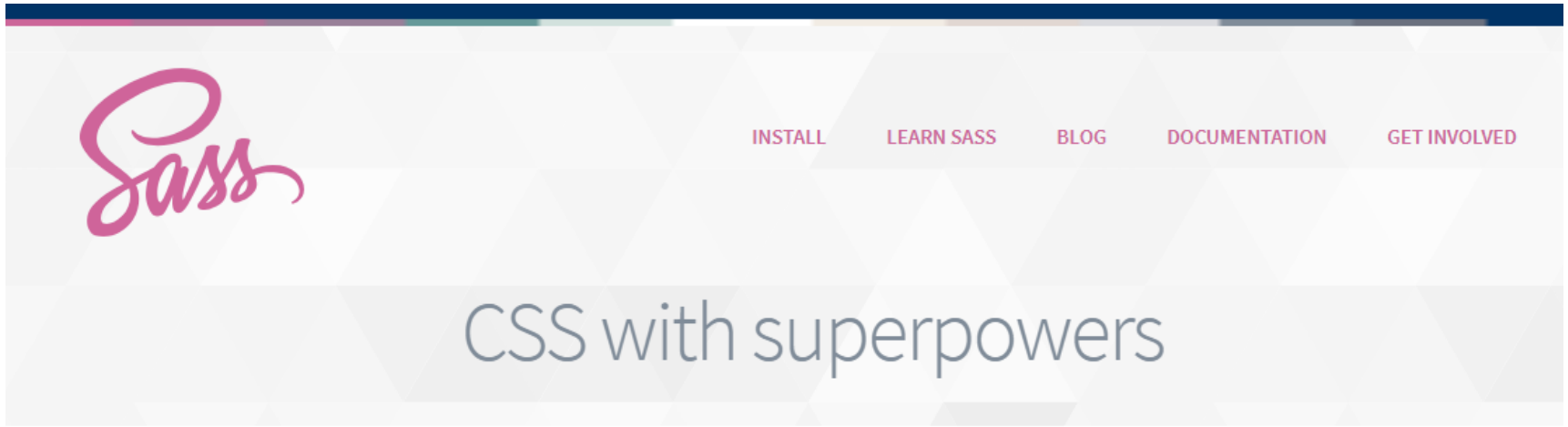
- Standard IDE
- Best Typescript Support
- Standard Editor
- Standard Tasks and configs can be shipped with the application and autostarted

VSCode ist optional aber „strongly recommended“.



2 / 4 Design Layer „Sass“

- Design wird mit Sass implementiert
- Scopevisio liefert eine Customization on top of Bulma.css
- Dieser Basisstyle kann im Projekt abgeändert und customized werden. Durch Sass erreichen wir eine gute Developer Experience und ein hohen Reuse-Faktor
- Plain CSS ist somit nur im Rahmen von Single-File-Components/Styles-Components (Vue/React) erlaubt.

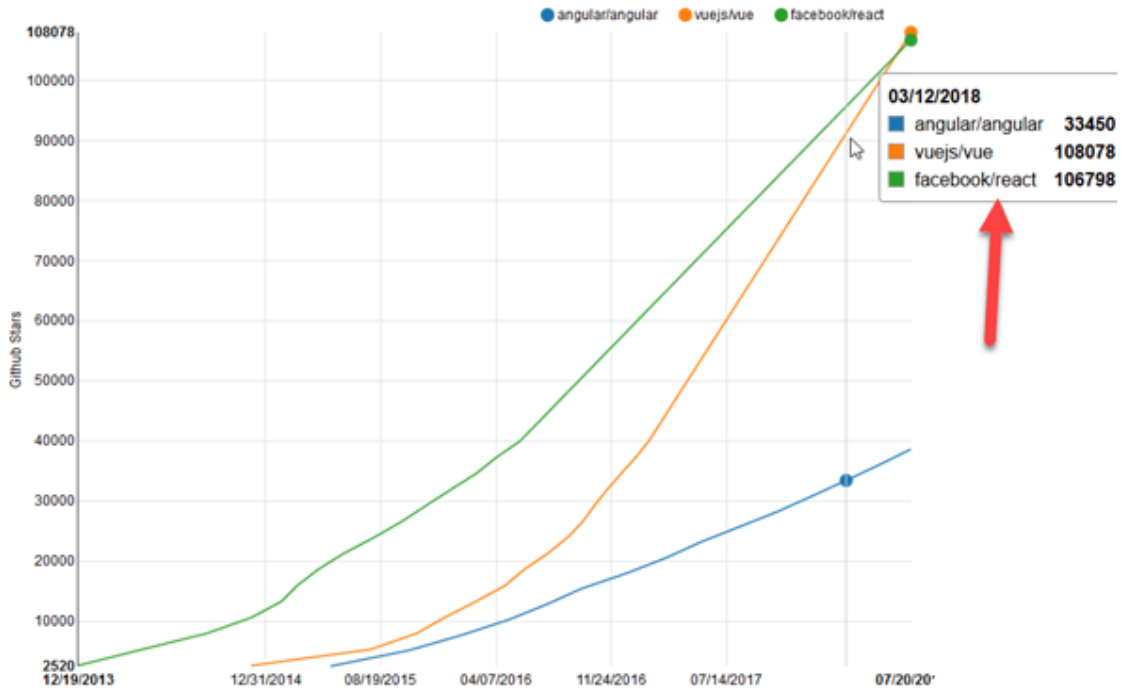


3 / 4 Application Layer „Vue.js und React.js“

Dieser Layer ist sehr Kontrovers und Subjektive.

SPA, Mobile Applications, PWAs etc. können auf folgende Arten gebaut werden:

- Plain Javascript
- Webcomponents
- Vue.js
- React.js
- Angular 2
- Ember.js



Vue.js und React.js sind bei den Entwicklern am attraktivsten und haben noch Momentum.

https://medium.com/front-end-weekly/react-vs-angular-vs-vue-js-a-complete-comparison-guide-d16faa185d61

3 / 4 Application Layer „Vue.js und React.js“

Primär wird entschieden in der HWP CB AG Gruppe Vue.js benutzt werden!

Begründung:

- Vue ist popular bei Entwicklern
- Vue kann Koexistieren mit allen anderen frameworks und auch nur partiell verwendet werden.
- Vue hat Single File Components (SFC)
- Vue hat sehr guten Typescript und VSCode support
- Vue hat auch eine Mobile-Native Version Extension und mit Nuxt.js ein High-Level Framework

React.js und das **React.js Ecosystem** (router + redux) wird ausnahmsweise dann verwenden, wenn man im Projekt React-Native, React Extensions, Next.js oder Gatsby.js benutzt.

4 / 4 Special purpose Frameworks

- Usecase: **Static Site Generation**
  Gatsby.js (uses React, Typescript, GraphQL, Webpack)
- Usecase: **Mobile App in Appstore**
  Apache Cordova (Many Plugins, Scanbot support)
- Usecase: **Charts**
  Chart.js
- Usecase: **3D**
  ThreeJs
- Usecase: **Maps**
  Google Maps and MarkerClusterer for Google Maps V3
- Usecase: **Codeviewer Integration**
  Monaco Editor
- Usecase: **Desktop Application**
  Electron Framework (enthält DOM und Node.js)

Integrated Development Environment (IDE)

1. **Datagateway** – Desktop Anwendung für Schnittstellenbetrieb

   ⇒ Typescript, NPM, Webpack, Electron/Node.js, Sass, Bulma, Vue.js

2. **Belegscanner-App** – Anwendung für den Belegscan

   ⇒ Typescript, NPM, Webpack, Cordova, Bulma, Vue.js

3. **Scopevisio-2GOs** – Bestehende 2Go Anwendungen

   ⇒ Typescript, jQuery, jQueryMobile, Cordova

4. **Gatsby.js Website** (zum Beispiel https://airbnb.io/)

   ⇒ Typescript (optional), React.js, Webpack, GraphQL

IDE der HWP CB AG

5. **Microsoft Teams**

   ⇒ Typescript, React.js, Webpack, Electron (when run as Application)

6. **Microsoft Visualstudiocode**

   ⇒ Typescript, Gulp/Yarn, Electron, Monaco

7. **Scopevisio-Webclient** – (Noch nicht entwickelt)

   ⇒ Typescript, Webpack, NPM, Loadash, Vue.js (oder React)

That‘s it, Thanks for your time.

Scopevisio AG
Rheinwerkallee 3
53227 Bonn
Germany

T +49 228 4334-3000
F +49 228 4334-3200
info@scopevisio.com
www.scopevisio.com

SCOPEVISIO

Appendix 1: Der Grund für Typescript und die Zwitterrolle als Javascripterweiterung und gleichzeitigem Ersatz von anderen Transpilern

Javascript-Transpiler:
Ein Transpiler (Compiler der wiederrum Sourcecode ausgibt) ist im Webtech Bereicht historisch notwendig gewesen, um aktuelle Sprachfeatures verwenden zu können und diese bei Bedarf auf frühere Javascript Versionen wie ES5 unter Berücksichtigung von Eigenheiten bestimmter Runtime abbilden zu können. Entwickeln kann man so unter ES6 und auch teilweise mit Verwendung von experimental Features wie Async/Await.
Babel ist der verbreiteste ES5 zu ES6 Transpiler.
Typescript wiederrum ist eine Spracherweiterung (somit voll abwärtskompatibel) von Javascript, welche typsichere mit Javascript (nun Typescript) erlaubt. Typescript kann auch alle relevanten Babelfeature wie Downcompiling auf ES5 mit erledigen. Man braucht also nur entweder Babel oder Typescript.

Appendix 1: Der Grund für Typescript und die Zwitterrolle als Javascripterweiterung und gleichzeitigem Ersatz von anderen Transpilern

… continued

Typescript ist notwendig um mittlere und größere Projekte sinnvoll langfristig warten zu können. Auch kleinere Programme die häufige und kritische Updates benötigen profitieren von Typescript Typescript ist mittlerweile sehr verbreitet. Für Libraries benötigt man sogenannte Typings, also Typinformationen, damit diese Libraries typsicher verwendet werden können. Optional kann man auch immer im Einzelfall auf die Typsicherheit verzichten. Die Typsicherheit von Typescript ist rein zur Compile-Time vorhanden. Die Ausgabe von Typescript ist herkömmliches Javascript in der gewünschten Version. Typescript erzeugt auch Polyfills und Code-Fragemente wenn dies für die eigenen Sprachfeatures notwendig ist.
Durch die Unterstützung von ES6 Modules in Typescript sind dann auch alle Programmteile prinzipiell wiederverwendbar.

Appendix 2: Dependencies and Modules

Für Dependencies und Modules ist NPM (Node Package Manager) in Verwendung bei Cordova, Vue, Vuxt, Gatsby, Electron und allen anderen bekannten Frameworks. Grunt und Gulp ist somit von NPM und/oder Webpack als aktueller Standard verdrängt worden.
Typescript Typedefinitionen werden auch im allgemeinen durch npm installiert, geupdated und verwaltet.

Appendix 3: Packager (HMR, Tree Shaking, Code Splitting)
Für eine gute Developer Experience benötigt man eine guten Watchmode um die Anwendung komfortabel und damit auch schnell entwickeln zu können. Außerdem sinnvoll sind prinzipiell:
- HMR (Hot Module Reload)
- Tree Shaking
- Code Splitting

Webpack ist hier der Standard für die aktuellen Frameworks. Außerdem gibt es Support für Sass, Typescript und die oben genannten Features wie HMR.