

Úvod

kurz Vývoj
progresívnych
webových aplikácií

Eduard Kuric



Prednášky/cvičenia

- **Ing. Eduard Kuric, PhD.**
- **eduard.kuric@stuba.sk**
- **miestnosť 4.42**

Všetky informácie:

<https://github.com/kurice/vpwa24>

Projekt

- 1., a 2. fáza projektu sa odovzdáva do AIS
 - nedeľa 23:59
 - oneskorené odovzdanie
 - max. 3 dni, za každý oneskorený deň 25% dole z pôv. maxima
- kontrolný bod
 - 10. týždeň semestra
 - neodovzdáva sa
 - odprezentujete mi na cvičení, že máte implementovaných aspoň 7 z 11 prípadov použitia podľa požiadaviek

Hodnotenie

- Semester
 - projekt: **47 bodov**
 - **12 bodov**: 1. fáza – SPA prototyp s použitím Quasaru
 - **30 bodov**: 2. fáza – PWA aplikácia Quasar + AdonisJS
 - **5 bodov**: kontrolný bod v 10. týždni semestra
 - priebežný test **9 bodov** v 8. týždni semestra
- Skúška: **44 bodov**

Minimum za semester

- 29 bodov zo 47 bodov za projekt
- 3 body z 9 za priebežný test
- *Nutné minimum zo skúšky nie je.*

Pôjdeme postupne...

- Zostavenie na strane servera vs zostavenie na strane klienta
 - Server-Side Rendering (SSR) vs **Client-Side Rendering (CSR)**
 - **SPA (Single Page Application)**
- Vue.js - reaktivita, základné konštrukcie, komponenty, hooks, udalosti, Vuex (store, state management), ...
- Rámec Quasar
 - Single-page aplikácia (SPA)
 - Webpack, zavádzač (loader), transpilátor (Babel), balíkovač (bundler), minifikácia, zamlženie kódu (obfuscation)
- ECMAScript a TypeScript - úvod, základné konštrukcie, asynchrónne volania a udalosti
- Node.js, správa závislostí (npm, prerendering, hydratácia (client-side hydration), websocket
- Rámec AdonisJS - zostavenie na strane servera
- Transformácia SPA na PWA

WWW

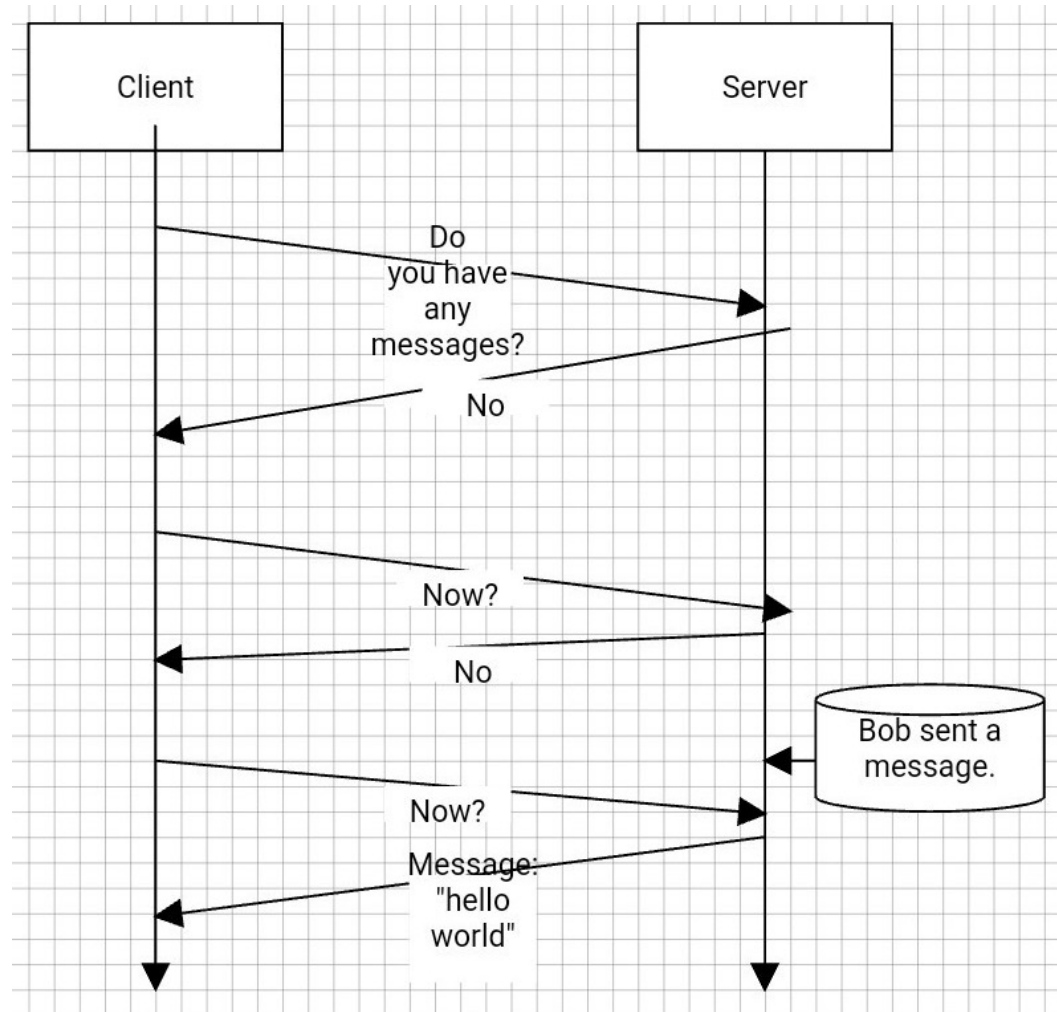
- **World Wide Web** (celosvetová sieť)
- **informačný priestor rôznych zdrojov** (dokumentov) na Internete **prístupných** prostredníctvom protokolu **HTTP(S)**
- autorom Webu je **Tim Berners-Lee**
- dokumenty - zvyčajne HTML
 - uložené na webových serveroch
 - štrukturované v HTML jazyku
 - prezeráme ich pomocou webových prehliadačov



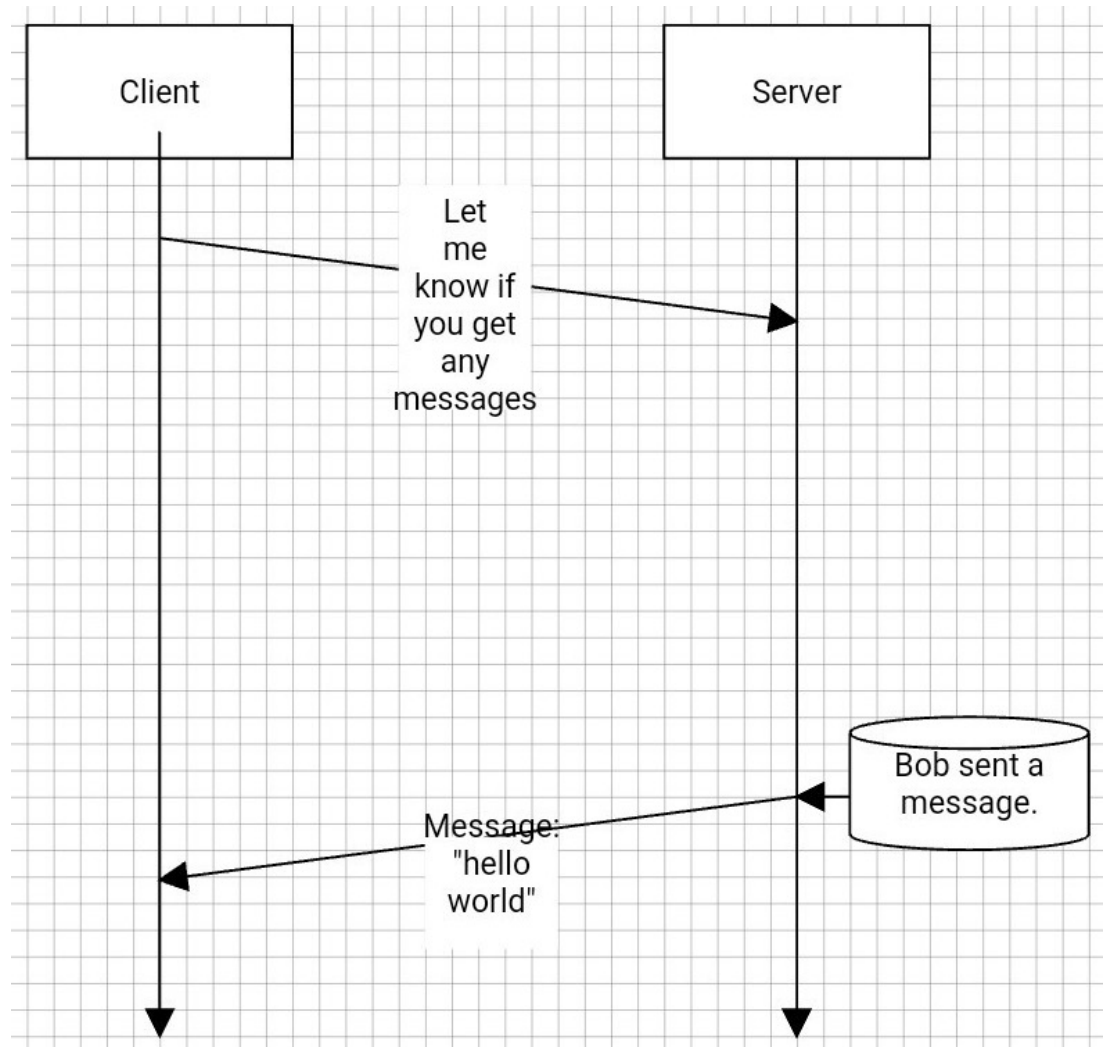
HTTP /RFC 2616

- **Hypertext Transfer Protocol**
- **internetový protokol na výmenu hypertextových dokumentov (HTML)**
- rozšírenie MIME (**M**ultipurpose **I**nternet **M**ail **E**xtensions) umožňuje prenášať akýkoľvek súbor
- implicitný port 80, https 443
- **používa URL** (Uniform Resource Locator) - jednoznačné umiestnenie zdroja

Websocket



Websocket /2



URL

- **Uniform Resource Locator**
- je referencia, ktorá určuje umiestnenie webového zdroja na Internete

scheme:[//[user[:password]@]host[:port]][/path][?query][#fragment]

<https://www.eshop.com/smartphones/apple/list?order=ASC#pagination>

Webový server

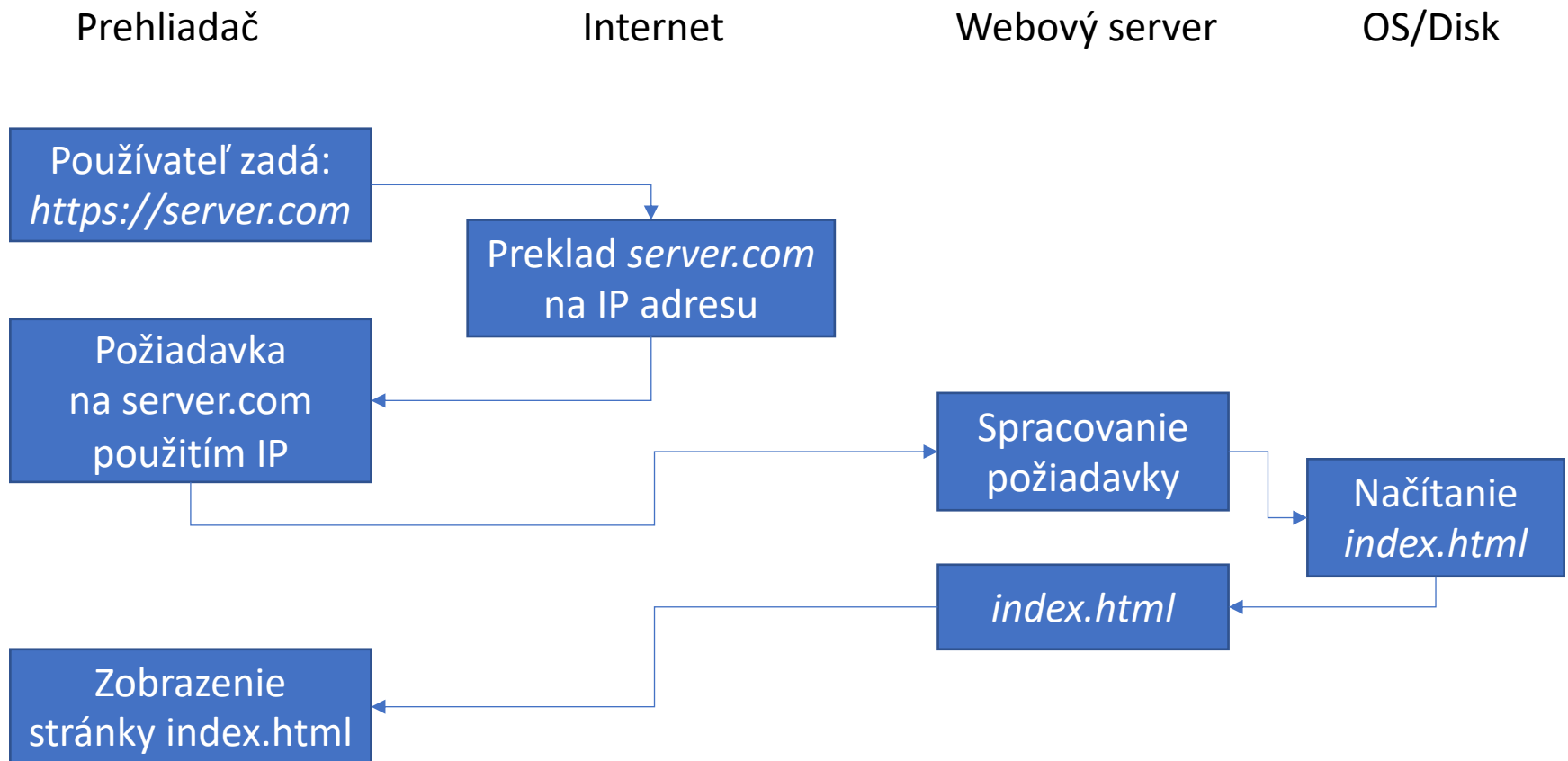
- počítač, ktorý vykonáva HTTP požiadavky od klientov (najčastejšie webový prehliadač)
- odpoveď: dokument, najčastejšie HTML
- [Apache HTTP](#)
 - open-source, cross-platform
- [Internet Information Services](#)
 - Microsoft, Windows
- [Node.js](#)
 - open-source, cross-platform

Web server - jazyky

- Apache + interpretované jazyky:
 - PHP
 - Perl
 - Ruby
 - Python
- IIS najmä ASP.NET
 - ASP.NET rámec (angl. framework)
 - kompilované programovacie jazyky C#, VB
- Node.js
 - JavaScript – interpretovaný/kompilovaný

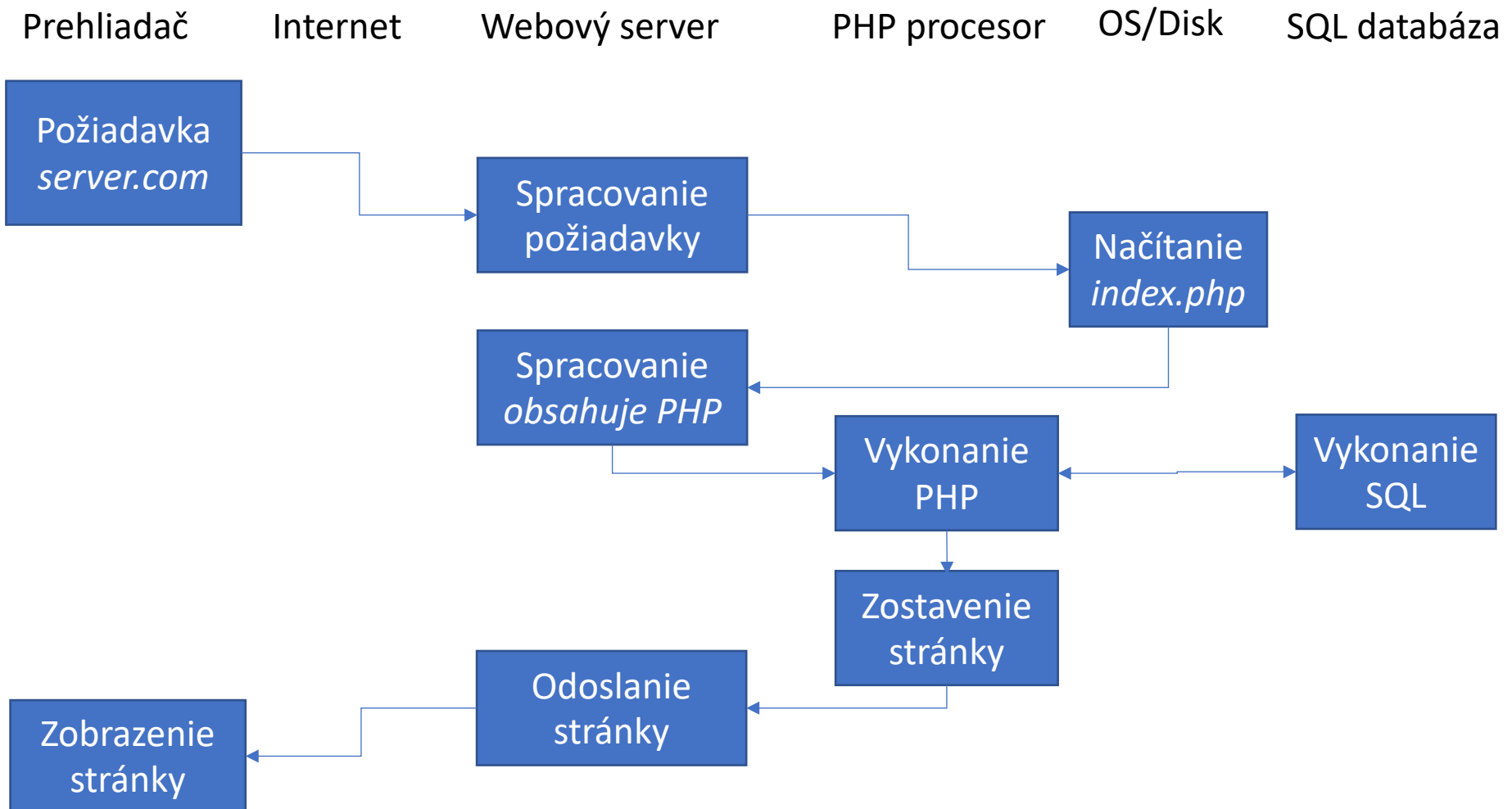
Statické stránky

klient/server požiadavka/odpoveď



Dynamické stránky

klient/server požiadavka/odpoveď



Zostavenie obsahu

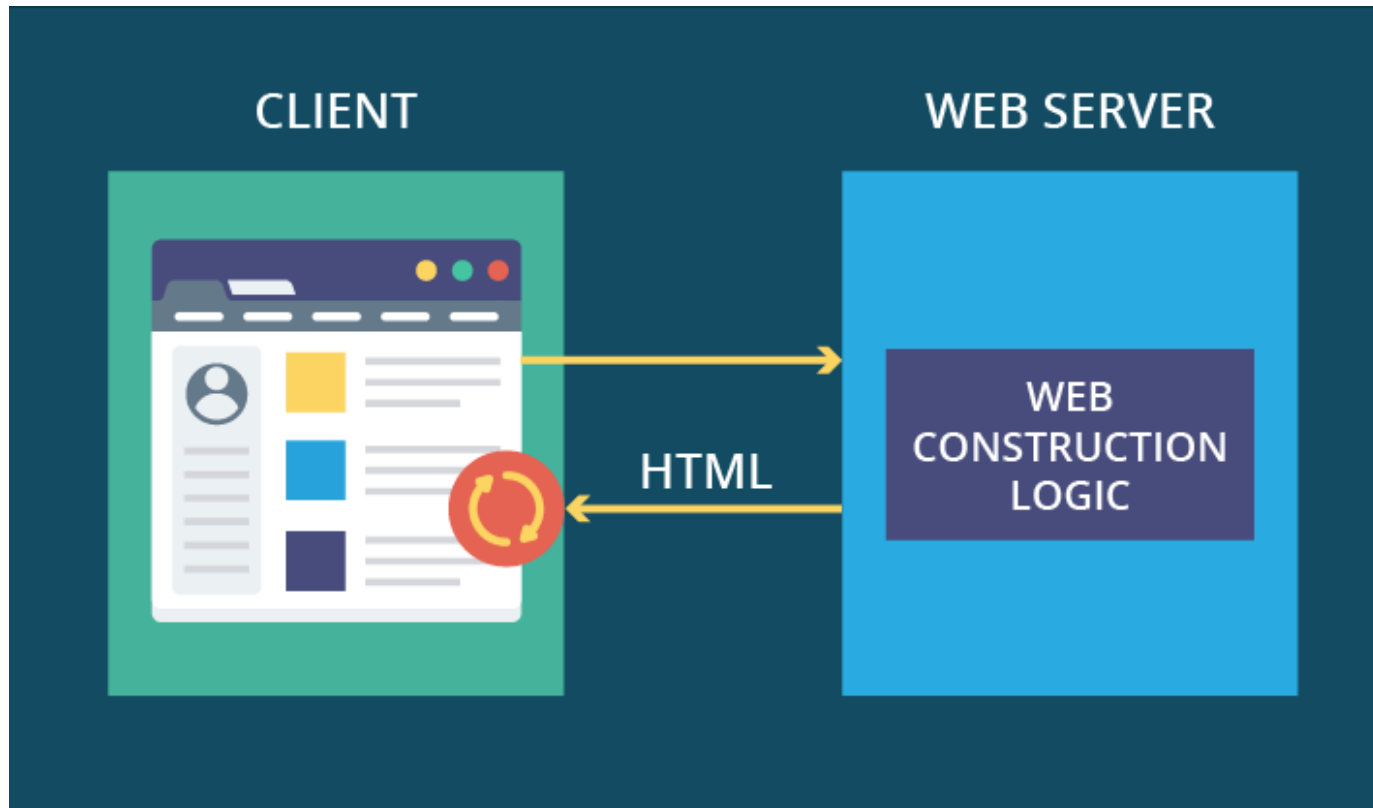
- na serveri (*angl. server side rendering*)
- na klientovi (*angl. client side rendering*)
- Na serveri
 - Apache
 - PHP (*Hypertext Preprocessor*)
 - Laravel rámec (*angl. framework*)
 - SQL databáza
- Na klientovi
 - Vue.js, React, Angular
 - [shadow DOM](#)

Architektúry web aplikácií

- **tri základné architektúry web aplikácií**
- **zostavenie/generovanie stránok na strane servera**
 - angl. Server-Side Rendering (server-side HTML)
- **dynamicky načítavané vybrané fragmenty/oblasti stránok cez JavaScript**
 - angl. JS widgets
- **zostavenie/generovanie stránok na strane klienta**
 - angl. client-side rendering,
 - reaktívne rámce, Vue, React, Angular
 - Single Page Application

Server-side rendering

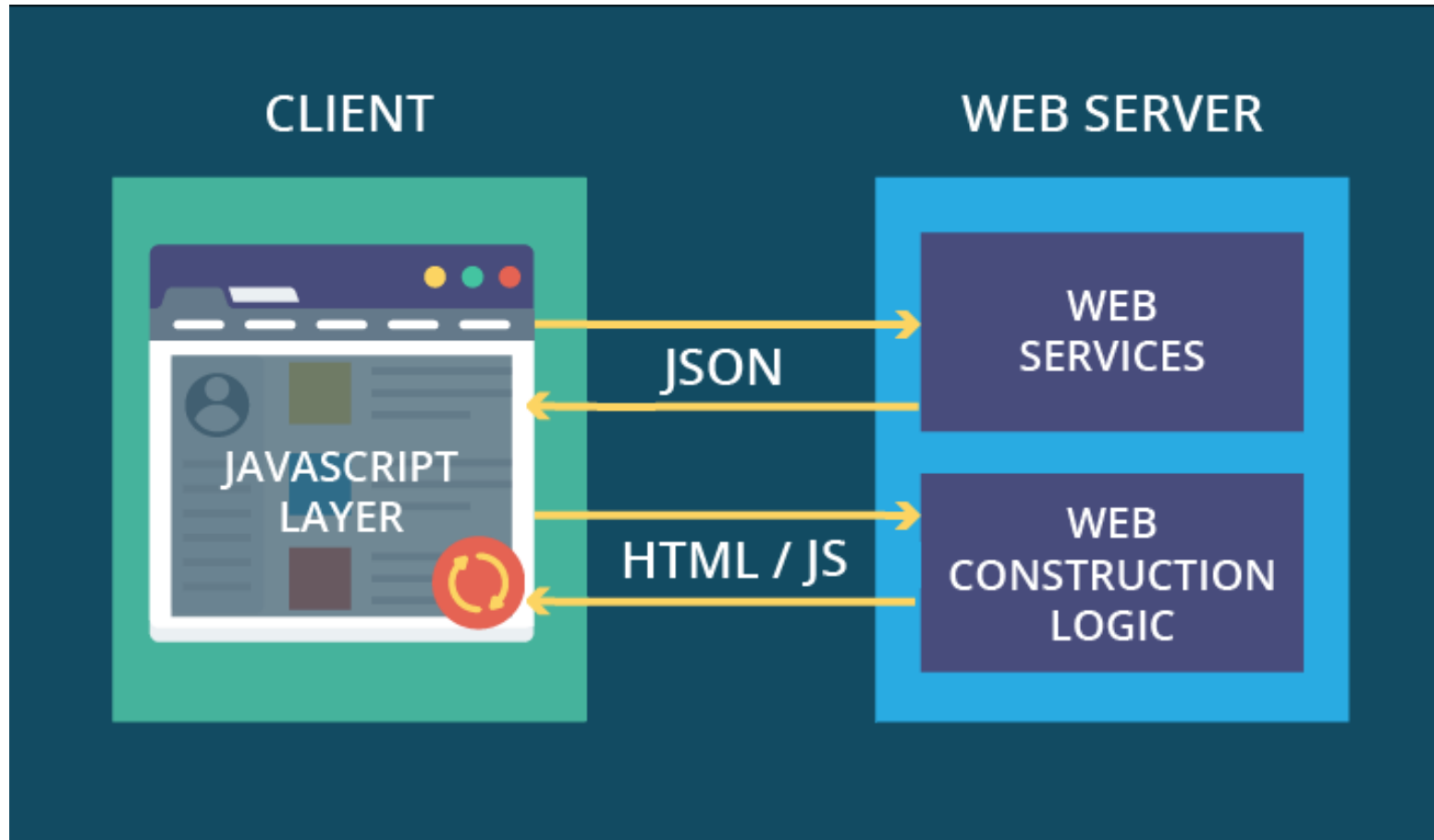
- **server generuje HTML stránky a posíla klientovi**



Zostavenie na strane servera

- najstaršia architektúra, zostavenie prebieha na výkonných serveroch
- na strane servera je veľká variabilita technológií/jazykov
- **plus:** bezpečnosť - biznis logika aplikácie je na serveri, vhodné aj z pohľadu „ochrany“ kódu pred neoprávneným použitím 3. stranou
- **mínus:** medzi klientom a serverom sa posiela množstvo dát, (ktoré sme už predtým prijali)

Client-side rendering Single Page Application - SPA



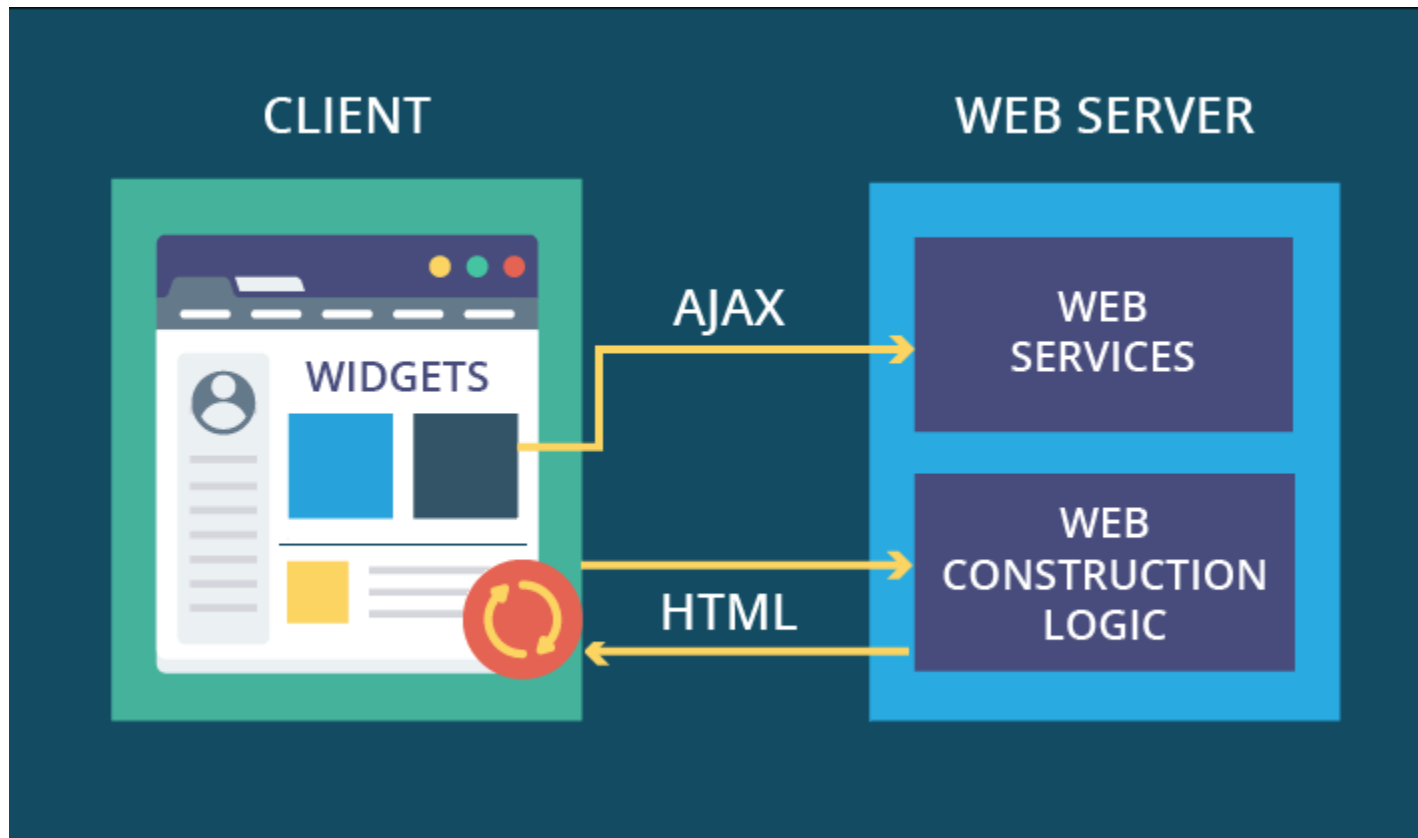
Client-side rendering

- **server vygeneruje HTML stránku, ktorá obsahuje koreňový element (kontajner) pre JavaScript aplikáciu/vrstvu**
- **spravidla celá biznis logika - konštrukcia/generovanie stránok (rozhrania) je na klientovi – tučný klient**
 - napísaná v JS, použitím rámca Angular, React, Vue...
 - JavaScript generuje HTML, aplikuje štýly, defiňuje správanie
- **server vystavuje webové služby, ktoré poskytujú iba údaje (napr. JSON), ktorými sa naplňa rozhranie aplikácie**

Client side

- <https://vuejs-i18n-demo.herokuapp.com/#/en>
- po úvodnom načítaní je množstvo prenášaných údajov minimálne, rozhranie kompletne generuje JavaScript
- **na vývoj je potrebná výborná znalosť JavaScriptu a špecializovaného rámca (Vue)**
- mínus: bezpečnosť – celá logika je na klientovi, poskytujeme kompletný kód

JS widgets



HTML

- HTML – **H**yper **T**ext **M**arkup **L**anguage
- **vyzerá ako text, je to text**
- **zjednodušene - obsah** (nadpisy, odseky, zoznamy, tabuľky...)
- nie je to programovací jazyk, je to **opisný značkovací jazyk**
- popisuje štruktúru stránky, slúži na štruktúrovanie stránok
- prehliadače nezobrazujú značky HTML, používajú ich na vykreslenie/zostavenie obsahu stránky
- samo o sebe je HTML fádny

CSS

- CSS - **Cascading Style Sheets**
- stará sa o **výzor stránky - prezentáciu, formátovanie**
 - sú to štýly, to, čomu zjednodušene hovoríme dizajn
 - písmo, farby, orámovanie, umiestnenie, pozadie,...
- HTML nebolo nikdy určené na to, aby formátovalo obsah
- HTML 3.2 špecifikácia priniesla element `` a atribút `color`
 - začala sa nočná mora pre vývojárov, každá stránka obsahovala písmo, farby – dlhý, zložitý, drahý vývoj
- CSS odstránilo formátovanie z HTML

JavaScript

- je to programovací jazyk HTML a Webu
- **na programovanie správania webových stránok**
- nemá nič s Javou

HTML+CSS+JavaScript

- **tvoria chrbticu**, neodlučiteľná trojica
- postačujú na vytváranie plnohodnotných webových aplikácií (+ nejaké tie rámce):
 - moderné
 - interaktívne
 - fungujúce v reálnom čase
- svet je pestrofarebný, treba poznať výhody/nevýhody technológií

Progresívne webové aplikácie

- **najväčší problém natívnych aplikácií je, že chcú príliš veľa hneď na začiatku**
- otvoriť obchod, vyhľadať, poprezerať opis - screenshots, čakať na stiahnutie, nainštalovať, udeliť prístupy
- výskum Google – pri každom z týchto krokov sa stratí 20% používateľov (1/4 dokončí proces)
- AppsFlyer - až 74,5% nainštalovaných aplikácií použijeme len raz

Progresívne webové aplikácie /2

- k webovým stránkam sa pristupuje jednoducho
 - v porovnaní s inštaláciou natívnej aplikácie,
 - ľahko sa zdieľajú, prostredníctvom odkazu URL
- natívne aplikácie sú lepšie prepojené s operačným systémom
 - je možné nainštalovať a používať offline
 - odkaz priamo na domovskej obrazovke
 - notifikácie

Progresívne webové aplikácie /3

- Fungujú pre každého používateľa, bez ohľadu na prehliadač/zariadenie
- Vyzerajú ako natívne aplikácie, správajú sa tak, sú „súčasťou plochy“ – push notifikácie
- Vždy aktuálne (najnovšia verzia) bez potreby sťahovania celej aplikácie
- Bezpečné (HTTPS)
- Jednoducho zdieľateľné cez URL
- Umožňujú pracovať offline, alebo na sieťach s nízkou prenosovou rýchlosťou

Príklady PWA

- [Google drive](#)
- Youtube



- prostredie (runtime systém) umožňujúce **vykonávať JavaScript na strane servera**
 - založený na [Google V8 engine](#) (ako Chrome)
- otvorený zdrojový kód (open source, MIT licencia)
- asynchrónne, udalosťami riadené API (event-driven)
- zabudované moduly (napr. http, url, events)
- [Node.js](#)



Vue.js

- populárny progresívny JS rámec na tvorbu **plne-interaktívneho používateľského rozhrania** webových stránok/aplikácií
- vytvoril ho Evan You po tom, ako pracoval v Googli, kde používal AngularJS
 - extrahoval a preniesol do Vue najlepšie koncepty z Angularu
 - prvá oficiálna verzia 2014
- <https://vuejs.org/>
- [porovnanie s inými rámcami](#)



- Komplexý BE rámec
- Nad Node.JS
- Inšpirovaný Laravelom
- MVC rámec
- TypeScript
- <https://adonisjs.com/>

DOM

- Document Object Model
- OO reprezentácia XML alebo HTML dokumentu
- je to API umožňujúce prístup/modifikáciu obsahu, štruktúry, alebo štýlu dokumentu
- pôvodne mali prehliadače vlastné špecifické rozhranie na manipuláciu s HTML elementami
- W3C štandardizácia
- DOM umožňuje prístup k dokumentu, ako ku stromovej reprezentácii

DOM /2

- na vykreslenie stránky používa väčšina **prehliadačov** interný **model podobný DOM**
- uzly v strome sú usporiadané v stromovej štruktúre – **DOM tree**
- **koreň** sa nazýva **document object**
- keď je stránka načítaná, prehliadač vytvorí DOM, ktorý funguje ako rozhranie medzi JavaScriptom a dokumentom
 - umožňuje vytvárať dynamické webové stránky
- JavaScript môže **pridávať, meniť, odstraňovať HTML elementy** a atribúty v stránke
- JavaScript môže **meniť** všetky **CSS štýly**
- JavaScript môže **reagovať** na všetky existujúce **udalosti** na stránke
- JavaScript môže **vytvárať** nové **udalosti** v rámci stránky



TypeScript

- Vytvorený a vyvíjaný Microsoftom
- Nadstavba nad JavaScriptom
 - rozširuje JS o statické typy a vlastnosti známe z OOP
 - Jednoduchší na debugovanie, spravidla rýchlejší vývoj
- Nepodporujú ho prehliadače, je transpilovaný do JS
- typescriptlang.org



Quasar

- FE výkonný rámec nad Vue.js
- Material design základná téma
- SPA, SSR, PWA, Mobile apps, Desktop, Browser extension
- Jednoducho prispôsobiteľný
- Bohatý katalóg komponentov UI
- quasar.dev