Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Katedra informačních technologií

Studijní program: Aplikovaná informatika

Obor: Informační systémy a technologie

Tvorba uživatelského rozhraní mobilního ovladače pro interaktivní obrazovky

DIPLOMOVÁ PRÁCE

Student : Bc. Jana Nová

Vedoucí : doc. Ing. Ota Novotný, Ph.D.

Oponent : Ing. Lukáš Daněk

2016

Prohlášení:

Prohlašuji, že jsem diplomovou práci zpracovala samostatně a že jsem uvedla všechny použité prameny a literaturu, ze které jsem čerpala.

V Praze dne 29.11 2016 . . . . . .

Jméno a příjmení

Poděkování

Ráda bych poděkovala vedoucímu práce doc. Ing. Otovi Novotnému, PhD. za vedení této práce.

Abstrakt

Cílem této diplomové práce je vytvoření responsivní frontendové části mobilního ovladače pro aplikace na interaktivních obrazovkách s využitím moderních frontendových technologií. V teoretických kapitolách je čtenář seznámen se základními principy přístupnosti a použitelnosti včetně představení testovacích nástrojů, kterými lze tyto principy kontrolovat. Dílčím cílem práce je vytvoření stručného průvodce moderními frontendovými nástroji a technologiemi jako jsou například frontendové frameworky, CSS preprocesory či nástroje pro automatizaci úloh na frontendu. V praktické části jsou také tyto popsané nástroje a technologie využity pro tvorbu mobilního ovladače.

Klíčová slova

Přístupnost, použitelnost, UX, frontend, uživatelská rozhraní

Abstract

The aim of this thesis is to create a responsive front-end part of the mobile controller for applications on the interactive screens using modern frontend technologies. The theoretical chapters introduce to the readers the basic principles of accessibility and usability, including testing tools, which can be used for checking these principles. The aim of the practical part is to create a concise guide of front-end modern tools and technologies such as front-end frameworks, CSS preprocessors and tools for automating tasks on the front-end. In the practical part are these described tools and technologies used to create a mobile controller.

Keywords

Accessibility, usability, UX, front-end, user interfaces

Obsah

Seznam obrázků

Seznam výpisů programů

1 Úvod

1.1 Cíle práce

Hlavním cílem této práce je vytvoření frontendové části universálního mobilního ovladače aplikací na interaktivních obrazovkách za použití moderních frontedových nástrojů a technologií. Dílčím cílem práce je seznámení čtenářů s těmito nástroji a technologiemi, které v dnešní době používá moderní frontend a přiblížení tak tohoto odvětví svým čtenářům. Dalším dílčím cílem této práce je představení základních principů a nástrojů z oblasti přístupnosti, použitelnosti a UX uživatelských rozhraní.

1.2 Cílová skupina

Tato práce je určena především zájemcům o odvětví webového frontendu. Předpokládá se zde základní znalost webových technologií HTML a CSS. Práce může čtenářům přinést nové postřehy a poznatky v souvislosti s přístupností, použitelností a UX u webových stránek. Čtenář také získá některé tipy na různé nástroje pro testování nejen přístupnosti a použitelnosti, ale také například responsivního zobrazení. Představeny zde budou také moderní frontendové nástroje a technologie, které mohou pomoci posunout začínajícího frontend kodéra na vyšší úroveň.

1.3 Použité metody

K dosažení hlavního cíle, tedy vytvoření frontendu mobilního ovladače pro interaktivní obrazovky je nutné začít krátkým představením možností, které interaktivní obrazovky nabízí. K návrhu rozhraní je třeba znát, kdo bude výsledný produkt využívat. Proto bude v práci krátce představen nástroj pro návrh uživatelských rozhraní - tzv. „persony“. Zmíněn bude také nástroj pro tvorbu drátěných modelů (wireframů), který bude využit pro nákres výsledné podoby rozhraní ovladače. Dále je třeba zmínit základní informace o přístupnosti a použitelnosti, které by měly být při tvorbě ovladače brány v potaz. Protože jeden z cílů práce je vytvořit ovladač pomocí moderních frontendových technologií, musí

být také tyto technologie a nástroje popsány. Po načerpání teoretických poznatků je pak možné přikročit k samotné praktické části, tedy konkrétnímu návrhu a kódování ovladače.

1.4 Struktura práce

Teoretická část práce začíná třetí kapitolou, která obsahuje stručné představení pojmu interaktivních obrazovek a jejich použití. Ve čtvrté kapitole je čtenář obecně seznámen s uživatelskými rozhraními včetně nástrojů, které při tvorbě rozhraní mohou pomoci i nástroji, kterými jsou rozhraní testovány.

Pátá kapitola se zaměřuje na přístupnost. Popsány jsou, v souvislosti s procházením webu, různé typy handicapů včetně možností, jak přístupnost vylepšit a nástroje, kterými lze přístupnost testovat.

Použitelnost webu je rozebrána v kapitole šest, kde jsou mj. uvedeny také nástroje pro testování použitelnosti a ukázány vybrané příklady použitelnosti.

Na kapitolu o použitelnosti navazuje související pojem UX, který je vysvětlen v kapitole sedm. Jsou zde také uvedena některá pravidla správného UX při tvorbě formulářů a responsivních webů.

Nejvíce rozsáhlá je kapitola osm, která pokrývá všechny nástroje a technologie, které byly při tvorbě ovladače použity. Velká pozornost je věnována frontendovým frameworkům Bootstrap a Foundation, které jsou představeny včetně ukázek implementace vybraných prvků. Další velkou část této kapitoly tvoří CSS preprocesory a pokročilé nástroje pro testování frontendu webových stránek.

Devátou kapitolou začíná praktická část, kdy je čtenář postupně seznámen s procesem návrhu ovladače včetně vytvoření drátěných modelů.

Na vytvořené drátěné modely (wireframy) navazuje kapitola deset, kde je popsán proces kodování frontendové části ovladače. Kapitola devět a deset tedy tvoří praktickou část celé práce.

2 Komentovaná rešerše informačních zdrojů

Tématika návrhu a tvorby uživatelských rozhraní je velmi rozsáhlá a existuje pro ni spousty informačních zdrojů domácích i zahraničních. Tato práce se zabývá tvorbou uživatelského rozhraní pomocí webových technologií, proto byly následující informační zdroje vybrány s ohledem na tyto technologie.

- Kniha: Steve Krug – Web design: nenuťte uživatele přemýšlet! (Krug, 2010)

Tato kniha je u odborné veřejnosti považována za téměř povinnou četbu pro každého, kdo se zabývá návrhem a tvorbou webových stránek. Je zaměřena především na použitelnost webu. Zmiňuje základní principy dobré použitelnosti a v její kapitole 9 uvádí také několik doporučení, jak použitelnost testovat. Okrajově se v kapitole 11 zaměřuje na přístupnost webu. Na téma přístupnost webu však lze doporučit knihu od Jakoba Nielsena viz dále.

- Kniha: Jakob Nielsen – web.design (Nielsen, 2002)

Jedná se o trochu starší knihu z roku 2000, ale zmíněná pravidla přístupnosti platí dodnes. Kniha vyšla v nakladatelství SoftPress a jedná se o překlad původního anglického znění: Design Web Usability: The Practice of Simplicity. Tato diplomová práce čerpá především z kapitoly 6 – Dostupnost pro handicapované uživatele, kde jsou popsány různé typy handicapů v souvislosti s používáním webu. Jakob Nielsen je uznávaný odborník na přístupnost. Nyní působí ve společnosti Nielsen Norman Group. Webové stránky této společnosti (www.nngroup.com) mohou být také dobrým informačním zdrojem. Web obsahuje spoustu odborných článků z oblasti přístupnosti, použitelnosti a UX.

- Kniha: Petr Staníček – Dobrý designer to všechno ví (Staníček, 2016)

Kniha byla vydaná v době psaní této práce a ukázala se nakonec jako vynikající zdroj zejména pro kapitolu 7 – UX a také pro kapitolu o technických scénářích. V knize je popsán celý proces návrhu webu a jsou zde vysvětleny také motivace všech zúčastněných stran, tedy klienta, návrháře a uživatele.

- Kniha: Jan Řezáč – Web ostrý jako břitva (Řezáč, 2014)

Jan Řezáč v současné době působí jako konzultant v oblasti dozoru webových stránek. Tato kniha je určena nejen pro tvůrce webů, kteří díky ní získají trochu větší nadhled na proces návrhu webu, ale slouží také zejména zadavatelům, kteří by si díky této knize měli ujasnit cíle a také pro koho svůj web hlavně dělají.

Všechny výše uvedené knihy jsem přečetla od začátku do konce a ani jednu kapitolu z nich nepovažuji za ztrátu času. Mohu tedy jedině doporučit. Navíc v knize od Jana Řezáče je možné nalézt odkazy na další hodnotné zdroje.

Webovým frontendem se zabývám od roku 2010, ale inspirací pro některé nástroje, které jsem v práci zmínila, mi bylo školení od Martina Michálka – Dnešní webová kodeřina, které jsem absolvovala v březnu roku 2016.

- Online: Martin Michálek – vzhurudolu.cz (Michálek, 2016)

Martin Michálek je český autor, který se zabývá frontendovými technologiemi a na toto téma také napsal několik elektronických knih a odborných článků, které publikuje také na svém blogu vzhurudolu.cz. Z tohoto blogu byly čerpány informace zejména pro kapitolu o CSS preprocesorech, ale také bylo odcitováno několik tipů, jak vylepšit použitelnost na mobilních zařízeních. Martin Michálek byl také jedním z řečníků letošního WebExpo 2016, které se konalo v Praze. Zde však zmínil jen některé praktické věci zaměřené spíše na CSS technologii Flexbox a SVG animace.

- Přednáška z konference WebExpo 2016: Jan Kvasnička - Nejčastější chyby při návrhu mobilního a responzivního webu prakticky (Kvasnička, 2016)

Dalším z řečníků na letošní WebExpo konferenci, které jsem se měla možnost zúčastnit, byl Jan Kvasnička, který se zabývá především UX. Z přednášky byla čerpána informace a o tom, v jaké orientaci a v jakých polohách je nejčastěji držen mobilní telefon. Tyto informace byly brány v úvahu při návrhu mobilního ovladače.

- Online: oficiální stránky frameworku Bootstrap a Foundation

K popisu jednotlivých komponent frameworku Bootstrap a Foundation byl nejvhodnější zdroj vždy jejich oficiální web. Kde lze také čerpat informace o dalších komponentách, které v této práci uvedeny nejsou.

- Online: Smashingmagazine.com

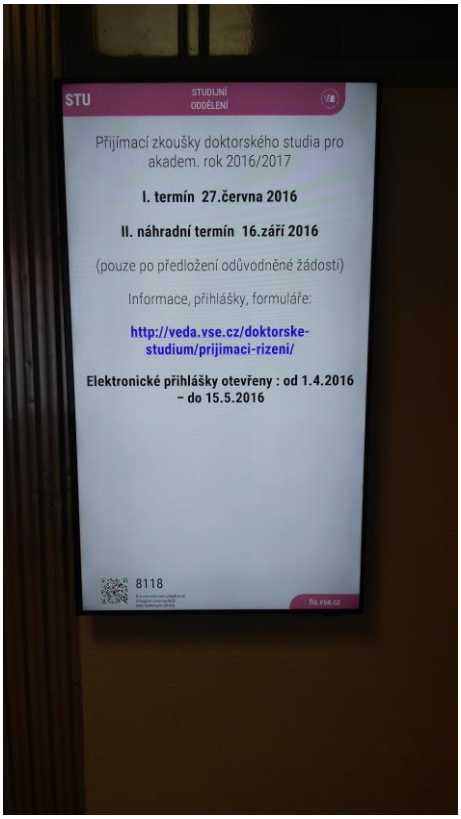
Dalším vhodným doplňujícím online zahraničním zdrojem může být také Smashing magazine, který pokrývá různá odvětví webu – od UX designu až po typografii.

3 Interaktivní informační obrazovky

V současné době je člověk obklopen velkým množstvím informací a reklam, které potkává na každém kroku. Ať už je v obchodě, v čekárně u lékaře, v restauraci, kině nebo třeba na nádraží. Firmy vymýšlejí stále nové způsoby, jak svého potenciálního zákazníka zaujmout případně předat jim nějakou informaci.

Jedním ze způsobů, jak lidem nabídnout svůj obsah je informační obrazovka. Na ní mohou být různá sdělení, od reklamy až po informace o jednotlivých výrobcích.

Na obrázku 3.1 je možné vidět informační obrazovku před dveřmi studijního oddělení Fakulty informatiky a statistiky Vysoké školy ekonomické v Praze.



Obrázek 3.1:

Informační obrazovka na VŠE

(Zdroj: vlastní fotografie)

3.1 Digital Signage

Pro různá digitální zobrazovací zařízení ve veřejných prostorách se vžil název „Digital Signage".

Výhoda zobrazení informací na digitální obrazovce oproti papírovému sdělení je jasná - obsah je možné okamžitě změnit, díky digitálním obrazovkám se může šetřit papír a toner (ale na úkor spotřeby elektřiny).

Typická oblast, kde lze pomocí digital signage ušetřit náklady je pohostinství. Pokud si představíme restauraci, kde nabízejí denní menu i minutky, informační obrazovka tomuto podniku každý den ušetří náklady za materiál, za přelepování a ještě navíc může digitální obrazovka obsahovat snadno vyměnitelné fotky pokrmů, které v rozlišení FULL HD budou jistě vypadat skvěle. Snadno také bude moci návštěvník zjistit počet kalorií či obsah alergenů. Další podstatná výhoda je ta, že lze pomocí obrazovky na člověka přenést podstatně více informací v mnohem kreativnější a zábavnější formě. Zobrazovaný obsah lze také řídit centrálně z jednoho místa. (Dmarketing.cz, 2013)

3.1.1 Software

K ovládání vysílaného obsahu je zapotřebí mít speciální software, který se postará o plánování, vysílání a ovládání obsahu na obrazovkách.

Kromě speciálně vyvinutých softwarů na míru existuje i řešení s licencí Open source.

Možná nejznámější je Xibo dostupné na http://xibo.org.uk/about/.

3.2 Interaktivita v obrazovkách

S příchodem dotykových technologií se výrazně rozšířila možnost interakce člověka s obsahem na obrazovce. Ať už se jedná přímo o dotykovou technologii na obrazovce nebo ovládání interaktivní obrazovky z mobilního telefonu či tabletu. Právě druhé možnosti ovládání se bude úzce zabývat tato diplomová práce v její praktické části, kdy bude cílem vytvořit universální ovládání pro aplikace na obrazovkách. (Mediar.cz, 2016)

Díky ovládání aplikací na obrazovkách mobilním telefonem či tabletem lze například krátit volnou chvíli v čekárně hraním her apod.

Softwary na podporu digital signage navíc také umí vyhodnocovat, na jaké pole uživatelé nejvíce klikají a které používají. Díky tomu lze následně vylepšovat uživatelské rozhraní. (e15.cz, nedatováno)

4 Uživatelská rozhraní

Jedna z mnoha definic uživatelského rozhraní je podle serveru TechTerms následující:

“A user interface, also called a "UI" or simply an "interface," is the means in which a person controls a software application or hardware device. (Techterms.com, 2009)

Volně přeloženo, uživatelské rozhraní, někdy též označované jen “rozhraní” případně “UI” je termín, který označuje prostředek, kterým dokáže člověk ovládat softwarovou aplikaci nebo hardwarové zařízení.

Lze také říci, že správně navržené uživatelské rozhraní zprostředkovává používající osobě uživatelsky přívětivou interakci se softwarem nebo hardwarem.

Tato kapitola má za cíl především ukázat čtenáři nástroje, které mohou pomoci při procesu návrhu uživatelských rozhraní a jejich následném testování a vylepšování.

4.1 Typy uživatelských rozhraní

Dva základní typy uživatelského rozhraní jsou:

- příkazový řádek (command line)
- grafické uživatelské rozhraní (GUI)

Historicky starší je příkazový řádek, který nebyl uživatelsky příliš přívětivý. V dnešní době toto rozhraní využívají spíše programátoři, kterým naopak umožňuje rychlé zadání příkazů k interpretaci. Pro nezkušeného uživatele je tento druh rozhraní ale prakticky nepoužitelný.

Naproti tomu grafické uživatelské rozhraní nabízí uživateli lepší a intuitivnější uživatelský zážitek v podobě různých ikon, dialogových oken a především v možnosti ovládání s myší, pokud se bavíme o uživatelském rozhraní v počítači, případně dotykem. (Publi.cz, nedatováno)

4.2 Návrh uživatelského rozhraní

Dobré uživatelské rozhraní chce čas a mnoho iterací, kdy bude podle skutečných potřeb modifikováno (například podle analýzy, kam uživatelé nejvíce klikají, uživatelským testováním apod.).

Podle Jana Řezáče (Řezáč, 2014) se proces návrhu webu skládá z fáze objevování, uživatelského výzkumu, návrhu webu a evaluace.

Před samotnou tvorbou bychom se měli ptát hlavně, kdo všechno bude uživatelské rozhraní používat a k čemu bude sloužit samotná aplikace. Je třeba také myslet na to, že rozhraní může používat široká skupina uživatelů od začátečníků, po středně pokročilé až po experty, kteří například preferují vyvolání určité nabídky pomocí klávesové zkratky. Každá skupina může s rozhraním pracovat jiným stylem.

Důležitá je také zpětná vazba od aplikace pro uživatele. Uživatel by měl být informován o výsledku určité interakce s aplikací. S tím souvisí také předcházení chyb, které může uživatel při práci s rozhraním způsobit. Lepší řešení než zobrazení informace o chybě je určité akce vůbec nepovolovat a například tlačítko, jehož stisknutím by se v určité fázi v aplikaci vyvolala chyba vůbec nezobrazovat. (Dostál, 2007)

Další dobré pravidlo, které jistě ocení každý uživatel je možnost vzít svojí akci zpět/vrátit se k původní stránce/zastavit akci. (Dostál, 2007)

V neposlední řadě bychom se měli vyvarovat příliš odborným a technickým pojmům k popisu jednotlivých ovládacích prvků a myslet opět na co největší intuitivnost a přehlednost celého rozhraní. (Dostál, 2007)

Platí také, že čím méně ovládacích prvků, tlačítek, přepínačů apod., tím lépe.

V této kapitole lze nalézt několik nástrojů, které mohou pomoci při návrhu uživatelských rozhraní. V kapitole 4.4 jsou zmíněné také různé testovací prostředky, díky kterým lze získat zpětnou vazbu a rozhraní následně vylepšovat.

4.3 Co nám může při tvorbě UI pomoci

V této kapitole bude představeno několik pomocných prostředků, díky kterým lze dosáhnout navržení promyšleného a dobře použitelného uživatelského rozhraní.

4.3.1 Use case diagramy

Use case diagram (diagram případů užití) je jedna z UML (Unified modeling language) modelovacích technik, jejímž cílem je popsání funkcionality aplikace ve vztahu k uživateli. Diagram se skládá z aktérů (v našem případě uživatelů aplikace), případů užití a vztahů mezi aktéry a případy užití. Aktéra může zastupovat například registrovaný uživatel v aplikaci. Případ užití tohoto aktéra je pak například napsání komentáře k článku, hodnocení článku apod. (Čápka, 2013)

4.3.2 Persony

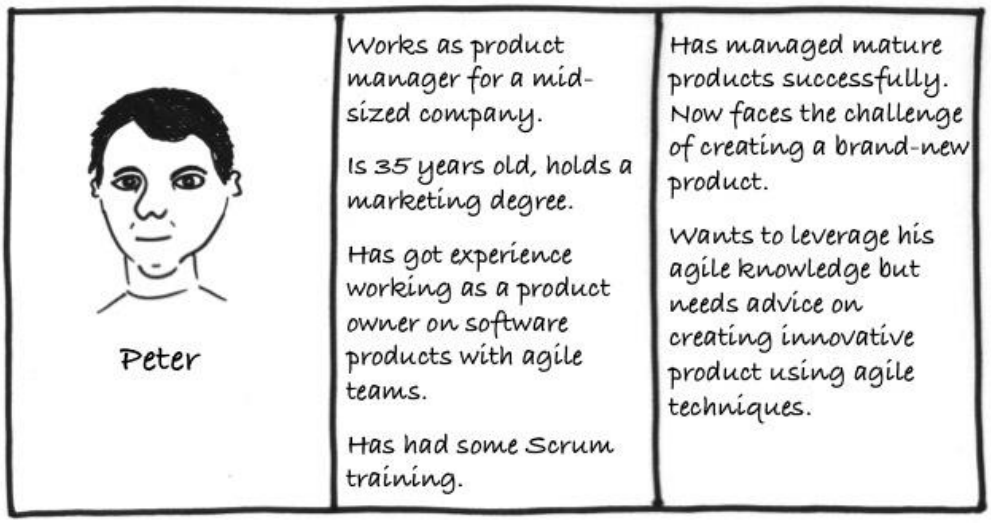
Technika zvaná persony může pomoci najít typické uživatele a dokáže představit jejich motivaci pro používání konkrétního webu či aplikace.

Persony by měly vycházet z uživatelského výzkumu uskutečněného pomocí dotazníků, rozhovorů apod. (Vernerová, 2013)

Ukázku persony znázorňuje obrázek č. 4.1, kde je možné vidět zástupce typické cílové skupiny uživatelů. Příklad dokládá, že persona je popsána velmi konkrétně včetně jména, věku, schopností a povolání.

Rozlišení podle věku může být důležité například podle vnímání různých symbolů. Pokud je potřeba nějakým symbolem vyjádřit ukládání dat, mladší generace tuto akci mohou mít spojenou s ikonou obláčku, který znázorňuje uložení v cloudu. Naopak generace starší, která ještě pamatuje uložení dat na klasické 3,5 palcové diskety, nebude mít problém ani s grafickým symbolem diskety, která mladší generace už znát nemusí. (Staníček, 2016)

Při návrhu je potřeba počítat i s povahovými rysy typického uživatele. Pokud naši typičtí uživatelé budou introverti, bude velmi těžké je zaujmout aplikací, která například pro účast v soutěži vyžaduje posílání vlastních fotografií apod. Když není tato vlastnost u person určena, měl by se produkt navrhovat tak, aby žádným zásadním způsobem neomezoval ani jednu z možných skupin. (Staníček, 2016)



Obrázek 4.1:

Příklad Persony

(Zdroj: http://www.romanpichler.com/wp-content/uploads/2012/05/SamplePersonaPeter.jpg)

4.3.3 Skicování nebo wireframy

Pokud již existuje jasná představa, jak má produkt fungovat, co je jeho cílem a kdo ho bude používat, je možné přikročit k prvnímu nákresu rozhraní. K převedení myšlenek do vizuální podoby lze použít tužku a papír a vytvořit skicu rozhraní. Tento nástroj je ideální pro okamžité zaznamenání myšlenky na papír. Kdo dává přednost digitální formě, má na výběr z velkého množství nástrojů na tvorbu drátěných modelů neboli wireframů. Příklady takových nástrojů může být myBalsamiq, Axure nebo UXpin.

Říká se, že čím kostrbatější a „ošklivější" je nákres modelu stránek, tím lépe. Je to hlavně z důvodu toho, že lidé mají někdy tendence považovat nákres drátěného modelu za grafickou podobu webových stránek a soustředí se pak zbytečně na grafické detaily místo na celkovou koncepci a strukturu. (Staníček, 2016)

4.4 Testování uživatelských rozhraní

4.4.1 Heuristická analýza

Heuristická analýza je jeden z typů testování uživatelského rozhraní odborníkem, nikoliv běžným uživatelem. Odborníci testují a porovnávají uživatelské rozhraní s obecně vyzkoušenými principy použitelnosti. Výhody tohoto druhu testování je získání poměrně rychlé zpětné vazby od odborníka. Nevýhodou může být vyšší cena experta. Doporučuje se také využít názorů více odborníků a jejich výsledky agregovat. (Usability.gov, nedatováno)

4.4.2 Uživatelské testování

Velmi důležitým prvkem pro vylepšování uživatelského rozhraní je uživatelské testování. Pokud se testování provádí, mělo by být iterativní a postupně zpracovávat vybrané připomínky od cílových uživatelů a ty znovu otestovat.

V knize „Nenuťte uživatele přemýšlet“ uvádí autor knihy a celosvětově uznávaný odborník na použitelnost, Steve Krug, několik velmi zajímavých podnětů ohledně testování: (Krug, 2010)

- *„Testování za pomoci jednoho uživatele je o 100% víc než vůbec žádné testování”.*
- *„Testování s jedním uživatelem na začátku projektu je lepší než testování s 50 uživateli těsně před koncem.”*
- *„Cílem testování není něco dokázat nebo vyvrátit. Má formovat váš názor”.*

4.4.3 A/B testování

A/B testování je užitečný nástroj při rozhodování, která varianta je ta správná. Na základě statistických dat se následně vybere vítězná varianta. Je možné si určit, na jak velkém vzorku se bude testovat - například jiná varianta se zobrazí každé desáté návštěvě. (H1.cz, nedatováno)

Testovat je možné například různé rozvržení layoutu, různé typy obrázků, textů, barev apod. (H1.cz, nedatováno)

5 Přístupnost

Dobře přístupný web je takový, který neklade návštěvníkům žádné překážky a na kterém dokáží návštěvnici bez problémů najít požadované informace.

Při návrhu a tvorbě webu by návrháři neměli brát v úvahu jen typického uživatele, ale měli by pozornost zaměřit také směrem k tomu, jak se web používá handicapovaným lidem (například sluchově nebo zrakově postiženým). Nad těmito návštěvníky webu nelze jen tak mávnout rukou, protože tvoří malé procento ze všech návštěv. Je třeba myslet i na ně a umožnit jim přístup ke stejným informacím, jako všem ostatním.

Vyhláška č. 64/2008 Sb., o formě uveřejňování informací souvisejících s výkonem veřejné správy prostřednictvím webových stránek pro osoby se zdravotním postižením (vyhláška o přístupnosti) dokonce upravuje povinnost na webových stránkách státních správy aplikovat základní pravidla přístupnosti. Jedná se celkem o 33 pravidel. (MVCR.cz, 2008)

Tato kapitola nejprve představí určité druhy handicapů v návaznosti na používání webových stránek. Následně bude zmíněno několik pravidel přístupnosti podle vyhlášky č. 64/2008 Sb. a podle pravidel přístupnosti WCAG20 dle organizace W3C. Na závěr kapitoly bude zmíněno opět několik nástrojů, kterými lze přístupnost testovat.

5.1 Druhy handicapů

Ne všechny druhy handicapů omezují uživatele v používání Internetu. Třeba takový vozíčkář je ve vztahu k prohlížení webu považován za zcela běžného uživatele. V této kapitole si uvedeme různé druhy handicapů, na které bychom měli při návrhu webu myslet a zpřístupnit těmto handicapovaným uživatelům informace a možnost procházení na webu tak, aby byli schopni web běžně používat.

Pokud porovnáme informace na webu a v tištěné podobě, mají informace uváděné na webových stránkách pro handicapované uživatele nesporné výhody - mohou si například převést text do mluvené řeči nebo zvětšit písmo. (Nielsen, 2002)

5.1.1 Zraková postižení

V dnešní době je většina webových stránek založena na vzhledu, největší problémy tak mohou mít hlavně zrakově postižení nebo nevidomí.

V souvislosti s lidmi se špatným zrakem je dobré brát v úvahu kontrast pozadí a textu. Určitě není vhodné umisťovat na pozadí různé vzory a textury, které tvoří s písmem příliš malý kontrast. O nástroji, který vypočítává poměr kontrastu barev textu a barev pozadí zde bude ještě řeč v kapitole 5.4.2. (Nielsen, 2002)

Důležité je také dodržení správného strukturování nadpisů, od <h1> - <h6> - pomůže to při procházení webu pomocí různých automatických čteček.

Další bod, který pomůže zrakově postiženým, je zvětšování velikosti písma. V dnešní době dokáží webové prohlížeče standardně na webových stránkách zvětšovat písmo. Ne všechny weby jsou na to ale připraveny a může se stát, že se díky zvětšení textu rozbije design určitých prvků na webu. Písmo by se mělo definovat v relativních jednotkách, tedy například em (výška jednoho řádku základního písma.) nebo %. Absolutní jednotky (mm, cm, in, pt) by se v přístupných webech neměly vyskytovat. (Nielsen, 2002)

Někteří uživatelé prochází web s vypnutými obrázky. Může to být třeba z důvodu šetření dat při prohlížení na mobilním telefonu apod. K tomu, aby i obrázky mohly nést určitou informaci, byl vymyšlen atribut ALT.

Příklad použití: `<img src=”images/fotoJN.jpg” alt=”fotografie autora” >` Atribut ALT při situaci, kdy má uživatel vypnuté obrázky, zobrazí text “fotografie autora”. Spousta webových tvůrců na atribut ALT stále zapomíná a znepřístupňuje tak informace handicapovaným uživatelům. Tento atribut by se naopak neměl používat v případě, když se jedná jen o obrázek, který je součástí designu a nenese žádný informační význam. (Nielsen, 2002)

5.1.2 Sluchová postižení

Problém pro sluchově postižené uživatele může nastat například při prohlížení videí, které nemají k dispozici titulky. Nebo při zvukovém záznamu, ke kterému není k dispozici přepis.

Titulky ve videu se zcela jistě budou hodit také cizincům, kterým může psaný projev pomoci více než mluvené slovo. (Nielsen, 2002)

5.1.3 Pohybová postižení

Uživatelé, kteří mají problém s přesným zacílením kurzoru myši na určitý odkaz, nemusí zoufat. Webové stránky je možné procházet i pomocí klávesnice. V HTML dokonce existuje atribut tabindex, který usnadňuje procházení a používání formulářových prvků. Stisknutím klávesy TAB přeskakuje kurzor po odkazech, které jsou na stránce umístěny. Stejně tak přeskakuje i u formulářových prvků. Pokud je formulář hodně složitý a pomocí kaskádových stylů i různě nastylovaný, hodí se atribut tabindex k přesnému určení pořadí vyplňování prvků ve formuláři. Je také díky tomu možné na webu, jehož primárním účelem je něco hledat, určit, že po prvním stisknutí tabulatoru, přeskočí kurzor hned do vyhledávacího pole.

Ukázka použití: `<input tabindex="1" type="text" name="hledani">`

5.2 Pravidla přístupnosti podle vyhlášky č. 64/2008 Sb.

- *„Každý netextový prvek nesoucí významové sdělení musí mít svou textovou alternativu.”* (Pristupnost.cz, nedatováno)

Typický případ tohoto pravidla je použití alternativního popisu u obrázků (html tagu <img/>). Dále je například u spousty formulářů na webu třeba ověřit, zda formulář odesílá opravdu člověk a je třeba opsat kód, který se vygeneroval v obrázku (tzv. Captcha). Leckdy má problémy přečíst tento kontrolní text i zdravý člověk a proto by přístupný web měl obsahovat například možnost zvukově přehrát text na obrázku, který je třeba opsat. (Pristupnost.cz, nedatováno)

- *„Pokud to charakter webových stránek nevylučuje, informace sdělované prostřednictvím skriptů, objektů, appletů, kaskádových stylů, cookies a jiných doplňků na straně uživatele, musí být dostupné i bez kteréhokoli z těchto doplňků a stránky musí být standardně ovladatelné. V opačném případě sdělí orgán veřejné správy tyto informace jiným způsobem.”* (Pristupnost.cz, nedatováno)

Zde se jedná především o to, že se některé přístroje (např. pro automatické čtení textu) nedokáží vypořádat s technologiemi různých skriptů (např. javascriptu apod.). Pomocí těchto technologií lze mj. měnit obsah webových stránek přímo u klienta. Cílem tohoto pravidla je umožnit dostupnost stejných informací i bez použití výše zmíněných technologií.

- *„Informace sdělované barvou musí být dostupné i bez barevného rozlišení."* (Pristupnost.cz, nedatováno)

Existují uživatelé, kteří nedokáží správně vnímat barvy. Konkrétní potíž by pro tento typ uživatelů mohla nastat v případě, že jsou například odkazy v textu odlišené pouze jinou barvou. Standardně by měly být odkazy také podtrženy, aby bylo zřejmé, že se opravdu jedná o odkaz.

- *„Barvy popředí a pozadí textu (nebo textu v obrázku) musí být vůči sobě dostatečně kontrastní, jestliže text nese významové sdělení."* (Pristupnost.cz, nedatováno)

K otestování tohoto pravidla slouží online nástroj, o kterém bude řeč v kapitole 5.4.2.

Pokud se jedná o neinformační text, který je použit spíše pro dekorační účely, nemusí být toto pravidlo dodrženo.

- *„Načtení nové webové stránky či přesměrování musí být možné jen po aktivaci odkazu nebo po odeslání formuláře."* (Pristupnost.cz, nedatováno)

Pokud nečekaně uživatele přesměrujeme někam jinam, bude se mu web používat špatně. Standardní přesměrování uživatel očekává při odeslání formuláře (například děkovná stránka při dokončení objednávky) nebo při kliku na odkaz na jinou stránku.

- *„Zvuk, který zní na webové stránce déle než tři sekundy, musí být možné na této webové stránce vypnout nebo upravit jeho hlasitost"* (Pristupnost.cz, nedatováno)

Toto pravidlo jistě ocení všichni uživatelé, bez ohledu na to, zda jsou handicapování nebo ne. Rozhodně není příjemné, když při prohlížení webových stránek máte otevřeno větší množství záložek a najednou se odněkud spustí zvuk. Některé webové prohlížeče, jako například Google Chrome zobrazují ikonu reproduktoru na záložce, odkud zvuk pochází (obr. 5.1).

Obrázek 5.1:

Ikona v záložce prohlížeče Google Chrome

(Zdroj: vlastní tvorba)

- *„Každá webová stránka (kromě úvodní webové stránky) musí obsahovat odkaz na vyšší úroveň v hierarchii webových stránek a odkaz na úvodní webovou stránku.“* (Pristupnost.cz, nedatováno)

Webové stránky jsou většinou tvořeny v určité struktuře kategorií a článků. Dobrým zvykem se stalo z jiné než úvodní stránky umožnit uživatelům prokliknout se přes logo webu na úvodní stránku. Toto pravidlo řeší prvek zvaný „drobečková navigace”.

5.3 Pravidla přístupnosti WCAG20 podle W3C

Pravidla z vyhlášky č. 64/2008 Sb. vycházejí z velké části z oficiálního doporučení organizace World Wide Web Consortium (W3C). V současné době (leden 2016) je k dispozici verze 2.0 z 11. 12. 2008, kterou je možné si prohlédnout na této adrese: https://www.w3.org/TR/WCAG20/

O český překlad tohoto dokumentu se postaral Radek Pavlíček. Český překlad je k dispozici zde: http://blindfriendly.cz/wcag20/

Celý dokument je rozdělen do 4 hlavních sekcí:

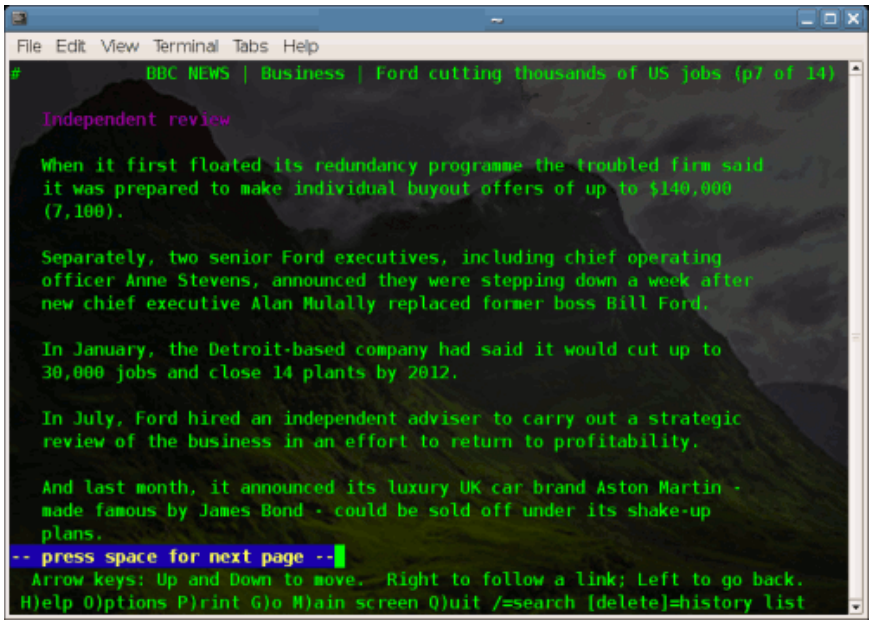
- **Vnímatelnost**: Informace a součásti uživatelských rozhraní musí být prezentovány tak, aby je uživatelé byli schopni vnímat. (Pavlíček, 2009)
- **Ovladatelnost**: Všechny součásti uživatelského rozhraní a všechny navigační prvky musí být ovladatelné. (Pavlíček, 2009)
- **Srozumitelnost**: Informace a ovládání uživatelského rozhraní musí být srozumitelné. (Pavlíček, 2009)
- **Robustnost**: Obsah musí být dostatečně robustní, aby mohl být spolehlivě interpretován širokou škálou přístupových zařízení. (Pavlíček, 2009)

5.4 Testování přístupnosti

V této kapitole bude představeno několik užitečných nástrojů, díky kterým lze vylepšit přístupnost webu. Existuje také několik online nástrojů pro testování přístupnosti.

5.4.1 Testování v prohlížeči Lynx

Pokud se web zobrazí správně v prohlížeči Lynx, bude se dobře používat zrakově postiženým uživatelům. Lynx je textový webový prohlížeč, který zobrazuje webové stránky zcela očištěné o veškerý design i obrázky. Je to dobrý nástroj pro převedení textu do mluvené podoby. Zobrazení webu BBC můžete nalézt na obrázku 5.2 níže. (Dickey, nedatováno)



Obrázek 5.2:

Ukázka textového prohlížeče Lynx

(Zdroj: http://tips.webdesign10.com/files/BBC-news-LYNX.png)

5.4.2 Testování kontrastu

K testování kontrastu jednotlivých barev (například barva textu vs. barva pozadí) slouží nástroj, který lze nalézt na adrese: http://juicystudio.com/services/luminositycontrastratio.php

Podle kritérií z WCAG 2.0 je minimální poměr mezi barvami 4.5:1. Pro představu poměr barev #ffffff (bílá) a #000 (černá) je tímto nástrojem vypočten na hodnotu 21.00:1. (Pavlíček, 2009)

5.4.3 Online validace přístupnosti

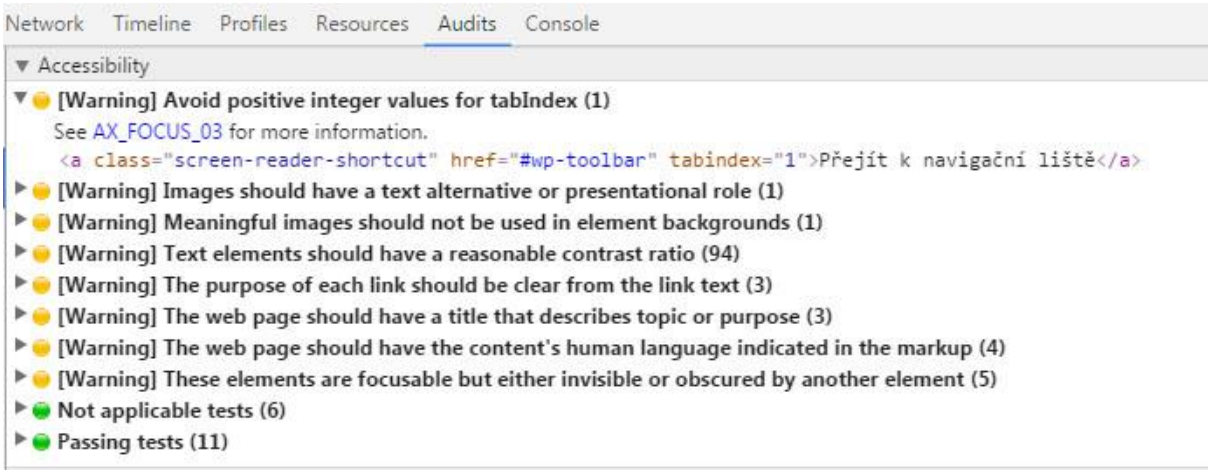
Existuje několik online validačních webových stránek, jejichž princip je velmi podobný. Na webu se nachází textové políčko pro vložení URL webové stránky, které chceme prohlédnout. Po vložení se analyzuje HTML kód z pohledu přístupnosti a jako výsledek nástroj nabídne doporučení, které lze zapracovat. Některé online validátory dokáží analyzovat i kontrast písma a pozadí.

Příklady takových online validačních nástrojů přístupnosti jsou:

- AChecker - dostupný na http://achecker.ca/checker/index.php
- Wawe - dostupný na http://wave.webaim.org/
- CyntihaSays - dostupný na http://www.cynthiasays.com/

5.4.4 Doplňky do prohlížeče

Pro kontrolu přístupnosti je možné využít také některé z doplňku do prohlížeče. V případě prohlížeče Google Chrome to může být například rozšíření Accessibility Developer Tool, které přidává do webového průzkumníka možnost spustit audit stránky (na záložce Audit) nejen z pohledu výkonu webových stránek, ale také z pohledu přístupnosti. Výsledkem je opět seznam doporučení rozdělených podle závažnosti. Znázorňuje to obr. č. 5.3.

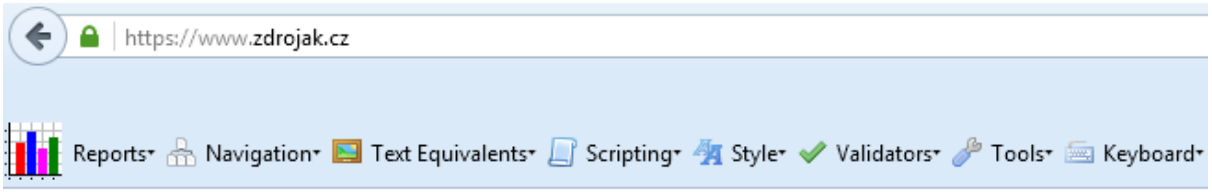


Obrázek 5.3:

Audit přístupnosti v nástroji Google Developer Tools

(Zdroj: vlastní tvorba)

Pro prohlížeč Mozzila Firefox tento doplněk k dispozici není, ale dobrou náhradou může být například Accessibility Evaluation Toolbar (obr. 5.4).



Obrázek 5.4:

Doplněk do prohlížeče Firefox Accessibility Evaluation Toolbar

(Zdroj: vlastní tvorba)

6 Použitelnost

Co to ta věčně zmiňovaná použitelnost vůbec je? Tuto kapitolu nelze začít jinak než definicí. Velmi pěkná definice použitelnosti je od uznávaného experta na použitelnost Steva Kruga:

„Something is usable if

- *A person of average (or even below average) ability and experience*
- *can figure out how to use the thing*
- *to accomplish some desired goal*
- *without it being more trouble than it's worth"* (Lireo.com, 2014)

Obecně lze říci, že dobře použitelný web dokáže průměrně (či podprůměrně) zkušený uživatel používat tak, že mu neklade žádné překážky v hledání požadovaných informací.

První Krugovo pravidlo použitelnosti zní: „Nenuťte mě přemýšlet". Tvrdí v něm, že uživateli by mělo být jasné, o čem stránky jsou a jakým způsobem by se měly používat, aniž by o tom musel uživatel nějak výrazně přemýšlet. Když uživateli není jasná struktura nebo význam některých prvků webu, může se stát, že velmi rychle odejde a dá přednost webu konkurence. Proto by použitelnost webu neměla být zanedbávaná část při procesu návrhu a tvorby webu.

Podle Kruga (Krug, 2010) uživatelé na webu nečtou. Takže hodiny copywritera, který vám připravuje poutavý a obsáhlý text, aby jej uživatel následně stejně ani nepřečetl, nakonec přijdou vniveč.

Další uznávaný expert na použitelnost je Jakob Nielsen, který se použitelností zabýval na webu www.useit.com (nyní www.nngroup.com - Nielsen Norman Group).

V této kapitole budou zmíněny základní principy a příklady použitelnosti a uvedeny budou také nástroje, které lze využít pro její testování.

6.1 Vybrané příklady použitelnosti

V této kapitole bude uvedeno několik příkladů, jak zlepšit použitelnost webových stránek.

6.1.1 Odkazování

Odkaz by měl být poznat. Pouze barevné odlišení nestačí. Jak bylo uvedeno v kapitole o přístupnosti, pokud je odkaz v textu označen pouze jinou barvou, může ho spousta uživatelů přehlédnout. Buď proto, že není dostatečně kontrastní, ale velice často i proto, že uživatele nenapadne, že by se mohlo o odkaz jednat. Během let používání webu se vžil zvyk, že odkazy, které jsou součástí textu, jsou podtržené. (Nielsen, 2002)

Z toho plyne, že obyčejný zvýrazněný text, který není odkaz, by neměl být nikdy podtržený.

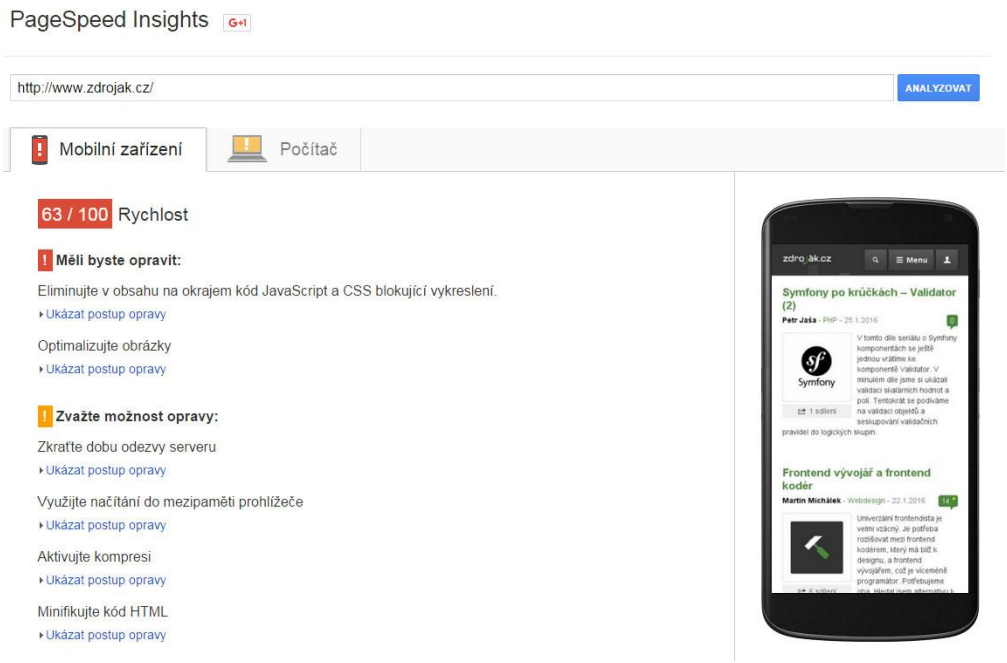
Další důležitou věc při odkazování tvoří tzv. anchor text, neboli z pohledu uživatele, ten text, který slouží jako odkaz. Špatný anchor text odkazu je například: „více informací o utkání Plzně se Spartou se dozvíte zde", kde jako odkaz slouží pouze text „zde." Lepší řešení by bylo „podívejte se na více informací o utkání Plzně a Sparty". Souvisí to s tím, co již bylo zmíněno výše - že lidé na webu nečtou, ale pouze prolétávají očima a narážejí na odkazy. (Nielsen, 2002)

6.1.2 Rychlost načítání

Rychlost načítání by se na webu měla řešit nejen proto, že to vypadá dobře, když se stránka načte okamžitě a nemusí se na ni čekat, ale také z hlediska SEO (Search engine optimization), neboli optimalizace pro vyhledávače. Rychlost načítání má pravděpodobně určitý vliv pro výpočet pozice v SERPu (výsledcích vyhledávání). Cílem Googlu by mělo být nabízet uživateli relevantní výsledky v rozumném čase. Vývojářům dokonce nabízí zdarma i tzv. PageSpeedModul, který zajistí optimalizaci rychlosti načítání částečně automaticky. (Jahoda, 2015)

Velmi dobrým nástrojem pro testování je další služba od Googlu - PageSpeed Insights.

Po zadání URL Google zhodnotí rychlost načítání podle různých kritérií a doporučí opravu, viz obr. 6.1.



Obrázek 6.1:

Výsledky z nástroje Google PageSpeed Insights

(Zdroj: vlastní tvorba z nástroje PageSpeed Insights)

Služba PageSpeed Insights testuje rychlost načítání také na mobilních telefonech. Zde bývá vzhledem k responsivním designům a mobilním verzím zcela jiný výsledek.

Následující seznam obsahuje jednotlivé faktory, které ovlivňují rychlost načítání.

- počet requestů neboli HTTP spojení se serverem

K eliminiaci počtu requestů nám může sloužit technologie zvaná Sprite, díky které je možné spojit načítání několika menších obrázků do jednoho.

Dále lze také doporučit používání font ikon místo obrázků (například obrázek šipky může být nahrazen vhodnou font ikonou). (Jahoda, 2015)

- datová velikost stahované stránky

Datovou velikost stahovaných stránek je možné vylepšit různými druhy komprese. (Jahoda, 2015)

- přesměrování

Pokud je to možné, je vhodné eliminovat přesměrování při načítání stránek na minimum. (Jahoda, 2015)

- blokující javascript nebo CSS

Když se stránka vykresluje, načítají se všechny skripty a styly z tagu <head>. Dokud prohlížeč nenačte tyto soubory, blokuje se vykreslování stránky. (Jahoda, 2015)

Rychlost načítání ovlivňuje ještě velká spousta faktorů. Jisté je jedno - pokud nebude uživatel na načtení stránky dlouho čekat, bude se mu web lépe používat a je velká šance, že se na požadovaný web opět vrátí. Naproti tomu, trvá-li načtení několik sekund, dříve nebo později dojde uživateli trpělivost. Je třeba pamatovat na to, že uživatel je od zavření okna pouhé jedno kliknutí.

6.2 Testování použitelnosti

V kapitole o uživatelských rozhraních již bylo krátce zmíněno uživatelské testování. V následujících kapitolách budou uvedeny konkrétní služby a nástroje, které jsou s testováním použitelnosti spojeny.

6.2.1 Usertesting.com

https://www.usertesting.com/

Jedná se o službu, která za několik desítek dolarů přiřadí vámi vytvořené testovací úkoly pro určitý web několika konkrétním reálným lidem. Po dokončení testování přijde nazpět video s průběhem testu. Testování je nemoderované a celé v angličtině.

6.2.2 Google webmaster tools

https://www.google.com/webmasters/tools/mobile-friendly/

Ani Google nezůstává při testování použitelnosti pozadu a uvolnil nástroj, který dokáže říci, zda je web připraven pro použití na mobilních zařízeních.

6.2.3 CrazyEgg

http://www.crazyegg.com/

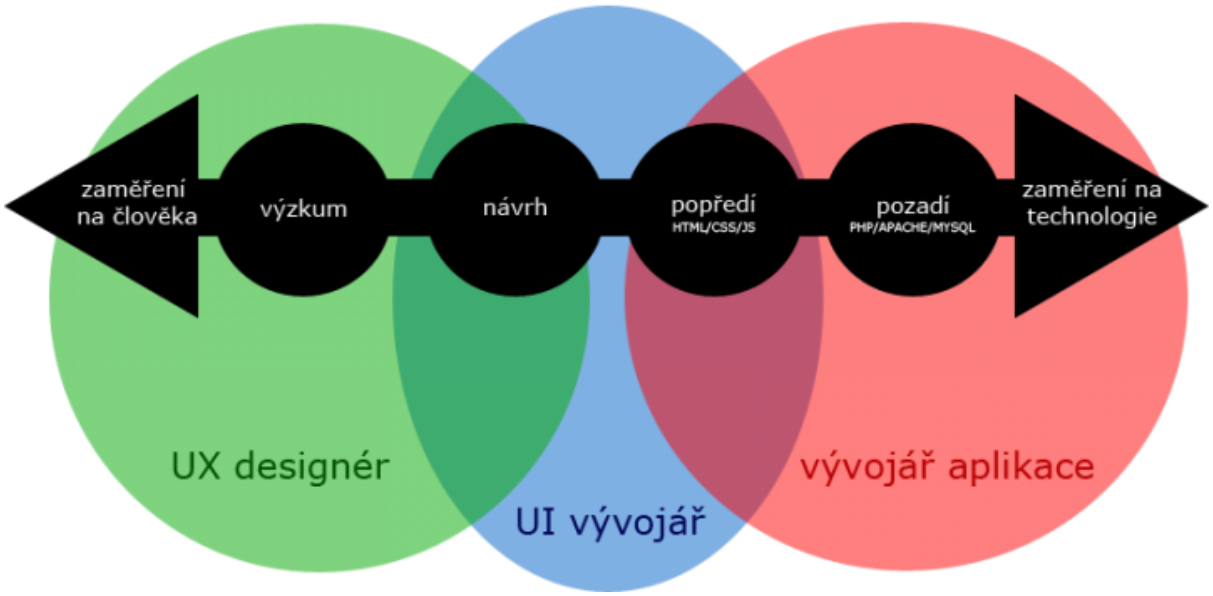
CrazyEgg je populární nástroj pro sledování uživatelových kliknutí na odkazy na webové stránce. Podporuje také tzv. heatmapy. Pokud na odkaz kliká velké množství uživatelů, je odkaz zabarven do sytě červené barvy. Pokud naopak na některé odkazy se kliká jen málo, jsou zabarveny většinou světle zelenou barvou.

Tento nástroj dokáže také ukázat, kolik lidí scroluje na stránce a kde většinou skončí.

7 UX

Co je vlastně UX? Definic, co znamená termín UX nalezneme v knihách i na webu spousty. UX je zkratka pro User eXperience, v překladu uživatelská zkušenost. Člověk, který se zabývá UX je UX designer.

Pěkně znázorňuje pojem „UX designer“ následující obrázek 7.1.



Obrázek 7.1:

Znázornění role UX designéra

(Zdroj: http://blog.lupa.cz/uxdesign/kdo-je-to-ux-designer/)

Jak je z obrázku patrné, UX designér je osoba, která se při návrhu produktu specializuje hlavně na uživatele a jeho prožitek z produktu. UX designer neřeší web jako takový. Je to pro něj jen jeden z dalších nástrojů. (Navrátil, 2012)

7.1 Tři úrovně designu

Ať už se jedná o design nábytku či webové stránky, vždy je třeba brát v úvahu tři základní úrovně designu:

- Základní úroveň - funkčnost a přístupnost,
- střední úroveň - použitelnost a ergonomie,
- nejvyšší úroveň - uživatelský prožitek (UX).

Budovat design by se mělo vždy od nejnižší úrovně, tedy v případě webové stránky by měl být web přístupný pro různé typy lidí s různými omezeními. Nelze řešit použitelnost něčeho, k čemu se lidé vůbec nedostanou. Všechny vylepšení vyšší úrovně by nikdy neměly zhoršit úroveň nižší. (Staníček, 2016)

7.2 Příklady správného UX u webových formulářů

Typický prvek webových stránek, kde je od uživatele vyžadována určitá interakce, jsou webové formuláře, které mohou na webu sloužit k nejrůznějšímu účelu, ať už se jedná o vyplňování údajů objednávky v e-shopu, registrace na různé události nebo jen odeslání zpětné vazby pomocí kontaktního formuláře.

V této kapitole bude uvedeno několik pravidel pro tvorbu formulářů, díky kterým by měl být uživatel schopen formulář bez problémů vyplnit a odeslat.

7.2.1 Žádejte po uživatelích jen údaje, které skutečně potřebujete

Spousta formulářů obsahuje zbytečná pole, která se zpracováním žádosti/objednávky/dotazu vůbec nesouvisí. Provozovatelé se možná chtějí o uživateli dozvědět co nejvíce, ale s každým zbytečným polem uživatelova chuť vyplňovat formulář klesá, proto by měl formulář obsahovat skutečně jen pole, která jsou potřebná. (Ilinčev, 2016)

7.2.2 Formuláře by si měly pamatovat již vyplněné údaje

Může se stát, že při validaci správnosti polí se pracně vyplněný formulář neodešle a požaduje po uživateli například opravu formátu zadaného data narození. Schopnost zpracovat různě zadané formáty dat je určitě plus, ale co dokáže uživatele spolehlivě otrávit je nepamatování si již vyplněných údajů. Pokud formulář neobsahuje javascriptovou validaci, která uživatele upozorní již při zadávání chybného vstupu a nechá uživatele formulář odeslat a následně ho vrátí prázdný pouze s chybovou hláškou, nutí uživatele vyplňovat všechny údaje znovu, což mu na radosti z používání formuláře rozhodně nepřidá. (Ilinčev, 2016)

7.2.3 Délka políček podle délky vstupu

Proč by mělo být pole pro zadání věku nebo pohlaví stejně dlouhé jako pole pro adresu, kde se předpokládá zadání většího počtu znaků? Krátké pole navíc fungují jako záchytné body, když uživatel v rychlosti prohlíží formulář. Ještě větší zmatení může nastat v případě nějakých potvrzujících kódu - například přihlášení do internetového bankovnictví. Uživatelé jistě ocení, pokud formulářové pole dovolí zadat přesný počet znaků, který potvrzovací kód obsahuje. (Ilinčev, 2016)

7.2.4 Ať nejasná pole obsahují vysvětlivky

Je vhodné uživatelům vysvětlit, za jakým účelem se po nich požaduje vyplnění požadovaného pole. Uživatelé ocení také upřesnění požadovaného formátu. Například u tvaru hesla je v zájmu provozovatele, aby si uživatel zvolil heslo bezpečné. Přesně v takovýchto situacích se u příslušného pole hodí připsat vysvětlivku, jak by takové bezpečné heslo mělo vypadat a co by mělo obsahovat. (Ilinčev, 2016)

7.3 Mobilní UX

Uživatelé, kteří prohlížejí web na mobilních zařízeních, jako jsou mobilní telefony a tablety mají na prohlížení jiné požadavky než lidé, kteří zobrazují web v počítači. U prohlížení na mobilním zařízení záleží podstatně více na rychlosti načítání a množství přenesených dat. Je to dáno omezením FUP, které stále mobilní operátoři hojně využívají.

Google také oficiálně potvrdil, že od dubna roku 2015 bude ve vyhledávání zvýhodňovat ty webové stránky, které budou optimalizovány pro mobily. (Takaki, et al., 2016)

Toho lze docílit prakticky dvěma způsoby - vytvářet weby responsivní nebo pro web udělat mobilní verzi. O porovnání responsivní verze a mobilní verze webu pojednává následující kapitola.

7.3.1 Responsivní verze vs. mobilní verze webu

V dnešní době se již weby vytvářejí povětšinou responsivní, tzn. dokáží se přizpůsobit velikosti obrazovky. Stejný web, by se měl tedy dobře zobrazit, jak na mobilním telefonu, tabletu, notebooku, tak na velkém monitoru.

Klasické webové stránky, které nejsou responsivní samozřejmě také lze prohlížet v mobilu, ale jejich používání není pohodlné, je třeba zoomovat na určité části a neustále web posouvat do stran apod.

Například nákup z mobilu na e-shopu, který není responsivní (nebo nemá mobilní verzi) může být velice obtížný a je zde nebezpečí, že zákazník raději odejde ke konkurenci a objedná si zboží na e-shopu, který optimalizovaný pro mobily je. O tom, zda jsou weby optimalizované pro mobily vůbec nutné, asi není třeba v dnešní době spekulovat.

V responsivní verzi lze upravit rozložení tak, že se ne vše, co je na stolním počítači bude zobrazovat také na mobilech.

Alternativou k responsivnímu webu je oddělená mobilní verze. Tu je možné umístit buď na samostatnou adresu (většinou m.adresawebu.cz) nebo využít technologii Dynamic serving, která podle zařízení, ze kterého uživatel přistupuje na web, pošle jiný obsah. K mobilní verzi webu se většinou tvůrci uchylují, pokud potřebují předat návštěvníkům z mobilních zařízení výrazně jiný obsah a lépe tak zacílit na mobilní klienty. (Vataha, 2015)

Mobilní verze mají často i svojí vlastní databázi. Celkově responsivní verze snižuje náklady na údržbu, protože se udržuje jen jeden obsah. U responsivní verze se navíc nemusí řešit přesměrování z adresy m. apod. (Vataha, 2015)

7.3.2 Vylepšení UX na mobilních zařízeních

K dosažení lepšího uživatelského zážitku při používání responsivního webu (nebo mobilní verze) mohou pomoci následující věci, které by měl brát každý designer v úvahu.

Carousely

Velmi často se stává, že uživatelé nepoznají, že se jedná o carousel, tedy posuvný kolotoč obsahu. Navigační šipky mohou být na telefonech skryty a tak některé informace mohou uživateli zůstat utajeny. (Michálek, 2016)

Pomalé načítání

O pomalém načítání již byla řeč v kapitole 6.1.2. Vylepšit rychlost načítání je možné také skrytím různých pluginů sociálních sítí. Většina operačních systémů v telefonech má sdílení již integrováno. (Michálek, 2016)

Aktivní telefonní číslo

Uživatelé jistě velmi ocení, když si při prohlížení v telefonu nebudou muset číslo pro vytočení kopírovat, ale bude jim stačit na něj jen kliknout. (Michálek, 2016)

Hamburger menu

Pro rozbalovací menu na mobilních zařízeních se vžil název „hamburger menu". Toto menu je charakteristické symbolem tří na sobě položených vodorovných čar. Ne všichni, ale tento symbol znají a jsou schopni si pod tím představit menu celého webu. Proto lze jen doporučit, aby byl k této ikoně připojen i popis „menu". Zvýší se tím šance, že uživatelé položky v rozbalovacím hamburger menu najdou. (Michálek, 2016)

Přidání možnosti vyhledávat

Zejména na větších informačních webech nebo internetových obchodech návštěvníka potěší, když bude mít možnost rychle zadat do pole pro vyhledávání konkrétní či obecný dotaz. Ještě lepší je, když hledání obsahuje našeptávač a v průběhu psaní dotazu se již pod vyhledávacím polem zobrazuje výpis nalezených položek. (Michálek, 2016)

Možnost zobrazení desktopové verze

Tvůrci mobilních či responsivních verzí by neměli zapomenout ani na odkaz na plnou verzi webu. Pokud si uživatel chce prohlédnout z mobilního zařízení plnou verzi webu, ale je neustále přesměrováván na mobilní verzi, dojde mu brzy trpělivost a odejde. (Michálek, 2016)

8 Technologie a nástroje použité při tvorbě ovladače

Tato kapitola popisuje, jaký postup, nástroje a frameworky byly při tvorbě rozhraní použity a obecně může být tak považována za jakéhosi průvodce pro pokročilejší tvorbu frontendových částí webu.

Píše se rok 2016 a dnešní moderní frontend není již jen HTML, CSS, to celé psané v jednoduchém či pokročilém poznámkovém bloku. Již nějaký čas se lze setkat s nástroji a technologiemi, které výrazně usnadňují každodenní práci frontend kodérů. V této praktické části budou jednotlivé nástroje a technologie postupně představeny. Všechny tyto technologie i nástroje byly při tvorbě universálního ovládání pro interaktivní obrazovky použity. Cílem této podkapitoly je ukázat začínajícímu frontend kodérovi, jakým způsobem lze snadno v dnešní době tvořit moderní frontendové části webových stránek (e-shopů) apod.

Jako první bude ukázáno pokročilejší vývojové prostředí (IDE), následovat bude spíše doplňující zmínka, ani ne tak týkající se frontendu, o webovém serveru, který byl při tvorbě ovladače použit. Pokud je řeč o moderním frontendu, nesmíme zapomenout ani na automatizační nástroje. Při tvorbě ovladače byl hlavně pro kompilaci SCSS do CSS použit nástroj Gulp, kterému se bude věnovat kapitola 8.4. Pro úplnost je nutno dodat, že existuje také velká řada nových technologií, založených na Javascriptu, jako například React. Javascriptovými frameworky se nicméně tato práce zabývat nebude.

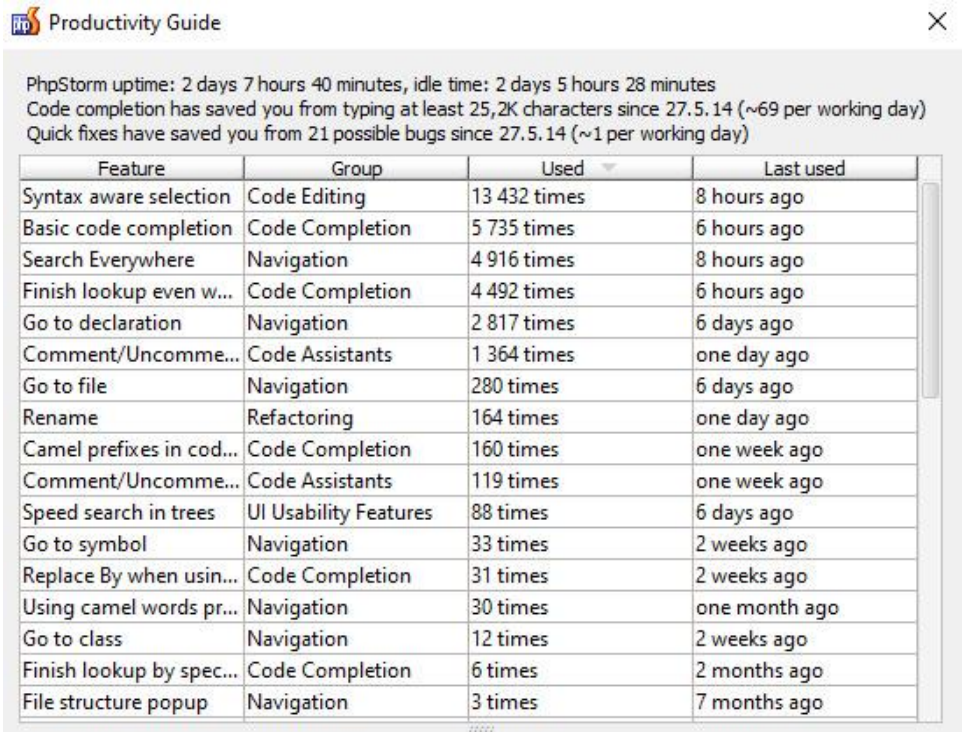
Kaptiola 8.5 obsahuje ukázky nástrojů, které lze využít při testování responsivního zobrazení. Následující kapitola 8.6 obsahuje krátkou zmínku o systému verzování zdrojových kódu, konkrétně verzovacím systémem GIT.

Představeny budou také CSS preprocesory SASS a LESS. Jejich popis lze nalézt v kapitole 8.7. Spolu s CSS preprocesory jsou další velkou pomocí pro kodéra známé HTML, CSS, JS frameworky - Foundation a Bootstrap, o kterých je možné se více dočíst v kapitole 8.8.

8.1 Vývojové prostředí

Pro tvorbu rozsáhlejších webových projektů je vhodné zvolit i účinné nástroje, které jejich tvorbu pomohou zefektivnit. Snad nikdo si již v dnešní době nedovede představit psát jakýkoliv kód delší než „Hello World" v poznámkovém bloku.

Moderní vývojová prostředí neboli IDE (Integrated development enviroment) obsahují nástroje pro debugování, rychlé hledání, doplňování kódu, navigace do deklarace funkcí, zvýraznění syntaxe a mnoho dalšího. Pro tvorbu praktické části diplomové práce byl použit PhpStorm (http://www.jetbrains.com/phpstorm/) od české firmy JetBrains. Na obrázku 8.1 je screenshot z PhpStormu, který autorka práce používá od roku 2014. Je možné vidět, kolikrát byla použita určitá funkce a kolik znaků PhpStorm storm autorce ušetřil v psaní v tomto případě to je něco přes 25 tisíc znaků.



Feature	Group	Used	Last used
Syntax aware selection	Code Editing	13 432 times	8 hours ago
Basic code completion	Code Completion	5 735 times	6 hours ago
Search Everywhere	Navigation	4 916 times	8 hours ago
Finish lookup even w...	Code Completion	4 492 times	6 hours ago
Go to declaration	Navigation	2 817 times	6 days ago
Comment/Uncomme...	Code Assistants	1 364 times	one day ago
Go to file	Navigation	280 times	6 days ago
Rename	Refactoring	164 times	one day ago
Camel prefixes in cod...	Code Completion	160 times	one week ago
Comment/Uncomme...	Code Assistants	119 times	one week ago
Speed search in trees	UI Usability Features	88 times	6 days ago
Go to symbol	Navigation	33 times	2 weeks ago
Replace By when usin...	Code Completion	31 times	2 weeks ago
Using camel words pr...	Navigation	30 times	one month ago
Go to class	Navigation	12 times	2 weeks ago
Finish lookup by spec...	Code Completion	6 times	2 months ago
File structure popup	Navigation	3 times	7 months ago

Obrázek 8.1:

Statistiky produktivity v programu PhpStorm

(Zdroj: vlastní tvorba)

Dalším příkladem vývojových prostředí vhodných k tvorbě frontendových aplikací může být například Aptana (http://www.aptana.com/). Toto IDE je dokonce open source a lze ho tedy využívat zadarmo.

Méně výkonnější alternativou mohou být například pokročilé textové editory jako Sublime (https://www.sublimetext.com/) nebo Atom (https://atom.io).

8.2 Webový server

Pro snazší a efektivnější editaci a konfiguraci celého ovladače bylo třeba vyčlenit opakující se části kódu do samostatného souboru (například opakující se záhlaví, zápatí, vysouvací menu apod.). K tomuto účelu byla využita technologie PHP. Aby bylo možné PHP soubory spouštět, bylo nutné nainstalovat webový server. Zvolen byl webový server Apache, který je součástí balíku XAMP. Tento balík obsahuje také databázi, ta ale nebyla v případě realizace praktické části potřeba, jelikož cílem bylo vyřešit pouze frontendovou část ovladače.

8.3 Webové technologie

Webový ovladač pro interaktivní obrazovky byl vytvořen pomocí základních webových technologií HTML5 a CSS3. Pro drobné programování na klientské straně byla využita javascriptová knihovna jQuery.

Pro případy dalšího rozšiřování ovladače se lze zamyslet, zda nevyužít některý ze šablonovacích systémů jako Smarty nebo Latte. Ty prozatím použity nebyly právě pro menší rozsah celého produktu.

Hlavním pomocníkem při tvorbě ovladače byly CSS preprocesory a frontendové frameworky. Konkrétně byl použit frontendový framework Foundation spolu s CSS preprocesorem SASS (respektive syntaxí SCSS).

CSS preprocesory budou dále rozebrány v kapitole 8.7. Více o frontendových frameworcích lze nalézt v kapitole 8.8.

8.4 Gulp

Jak bylo zmíněno v předchozí kapitole, pro tvorbu stylů byl použit CSS preprocesor SASS. Pro napojení do stránky ale je třeba zajistit výsledný zkompilovaný (a ideálně zminifikovaný) CSS soubor. Pro kompilaci kódu preprocesoru do CSS existuje spoustu nástrojů, ať už online či desktopových aplikací. Aby bylo možné docílit efektivní práce a kontinuální kompilace během psaní kódu, byl pro tento účel zvolen automatizační nástroj Gulp.

Gulp je open sourcový nástroj, který pro svůj běh potřebuje node.js ekosystém a je napsaný v Javascriptu. Jedná se o poměrně mladý nástroj, který vznikal v roce 2013. Jeho autorem je Eric Schoffstall. Ovládat jej je možné z příkazové řádky. (Ožana, 2014)

Obrázek 8.2:

Logo nástroje Gulp

(Zdroj: https://www.npmjs.com/package/gulp)

Díky Gulpu je možné sledovat změny v souborech a automaticky provádět kompilaci přímo při psaní kódu. Existuje velké množství rozšíření, které lze při tvorbě webu využít. Pro účely tvorby ovladače byly použity následující pluginy:

- `gulp sass`
- `browser-sync`
- `gulp-clean-css`

`Gulp sass` zajišťuje převod SASS kódu do výsledného CSS. Stejně tak obsahuje Gulp rozšíření pro kompilaci LESSu do CSS. Pomocí Gulpu je možné také přidat do projektu nástroj Browser-sync, který umožní jednoduché testování responsnivního zobrazení na různých zařízeních. Více o tomto nástroji lze nalézt v kapitole o testování 8.5. Nakonec bylo také využito rozšíření clean-css, které se postará o výslednou minifikaci souborů, díky čemuž se sníží datová velikost celého CSS souboru.

Alternativou pro nástroj Gulp je velmi podobný nástroj Grunt, který výše uvedené úkony umí též. Obecně záleží na každém vývojáři, který z těchto nástrojů mu vyhovuje více. Srovnání obou nástrojů ovšem není předmětem této práce.

8.5 Testovací nástroje

V procesu každého vývoje by se také měla objevit fáze testování. V případě vývoje frontendu webového ovladače se bude jednat hlavně o testování správného responsivního

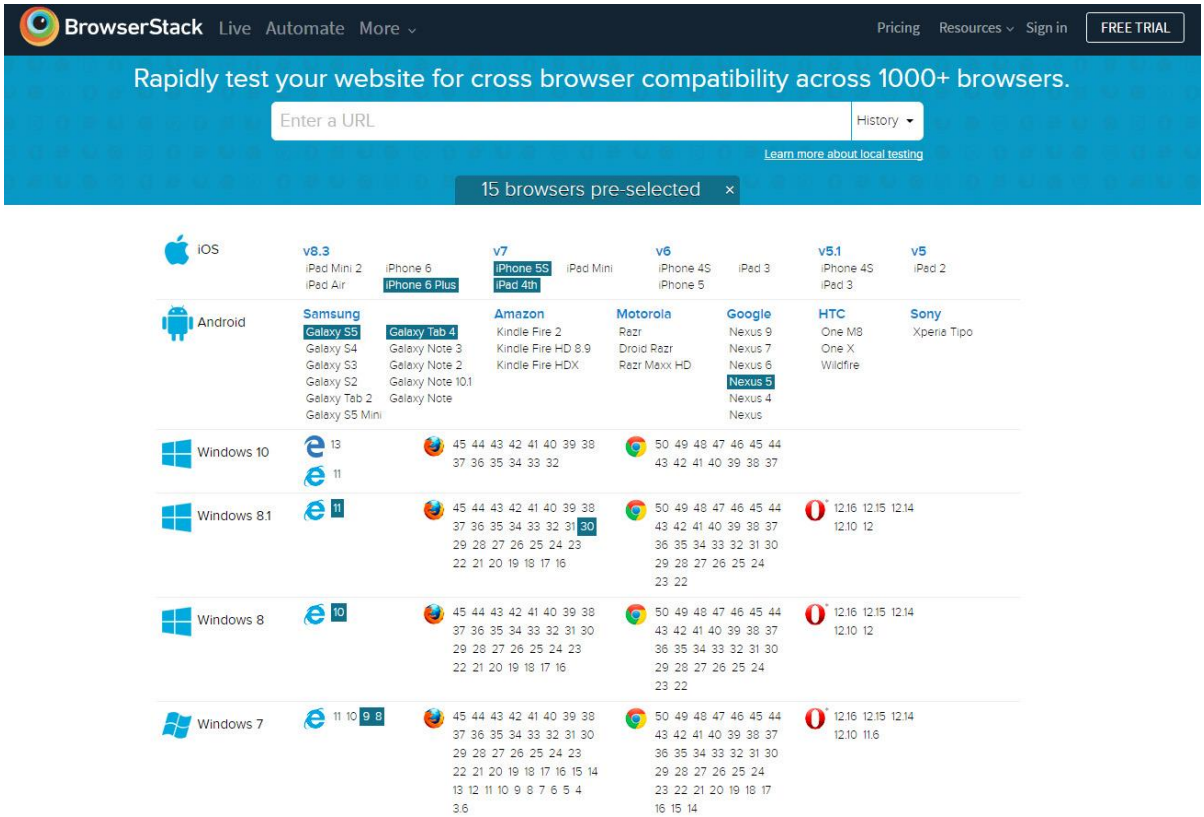
zobrazení, testování napříč možnými operačními systémy a prohlížeči a testování přístupnosti. K tomuto účelu byly vybrány následující nástroje, které budou v této kapitole blíže představeny. Jejich praktické využití bude popsáno také v kapitole 10.7.

8.5.1 Browserstack

Nástroj Browserstack je možné najít na adrese https://www.browserstack.com/. Je to placený nástroj, který ale nabízí některé své funkce také omezeně zdarma. Velkou výhodou tohoto nástroje je jeho možnost prohlížet webovou stránku v nejpoužívanějších mobilních zařízeních - tzv. live testování na fyzických zařízeních v cloudu.

V kombinaci s dev tools neboli vývojářskými nástroji prohlížeče Google Chrome, lze přímo provádět debugování na různých zařízeních. Samozřejmostí je také simulování překlápění z orientace portrait na landscape a zpět.

Další funkcí je provedení screenshotů stránky z opravdu velké škály prohlížečů a operačních systému viz obrázek 8.3.



Obrázek 8.3:

Možnosti testování v prohlížečích pomocí nástroje Browserstack

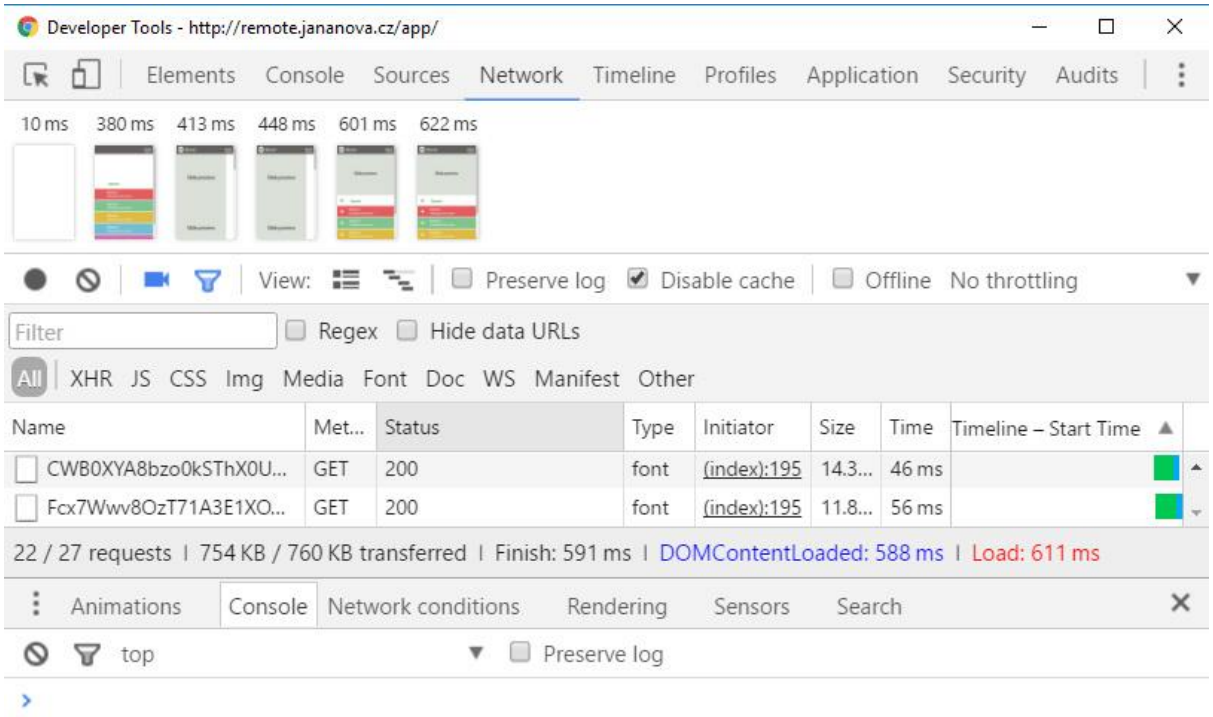
(Zdroj: https://www.browserstack.com/automate)

8.5.2 Quirktools

Pro testování responsivity je také k dispozici spoustu nástrojů, které jsou zcela zdarma. Jedním příkladem může být například Quirktools Screenfly dostupný na http://quirktools.com/screenfly/. Nevýhoda oproti nástroji Browserstack je ta, že nelze simulovat testování na reálných zařízeních. Nástroj pouze přepíná rozlišení, které odpovídá nejčastěji používaným zařízením, případně je možné nastavit i ručně konkrétní požadované rozlišení.

8.5.3 Developer Tools v Google Chrome

Sada Developer Tools neboli vývojářské nástroje v Google Chrome umějí mnohem více než jen otestovat responsiviní zobrazení. Na obrázku 8.4 lze vidět postupné fotografie stránky při tom, jak se načítá. Ve spodní části lišty je možné vidět informaci o tom, jak datově náročná je webová stránka a kolik obsahovala požadavků na server. Právě počet požadavků na server hraje důležitou roli při rychlosti načítání stránky. Na záložce „Console" se pak ukazují javascriptové chyby, chyby 404 apod.



Obrázek 8.4:

Vývojářské nástroje prohlížeče Google Chrome

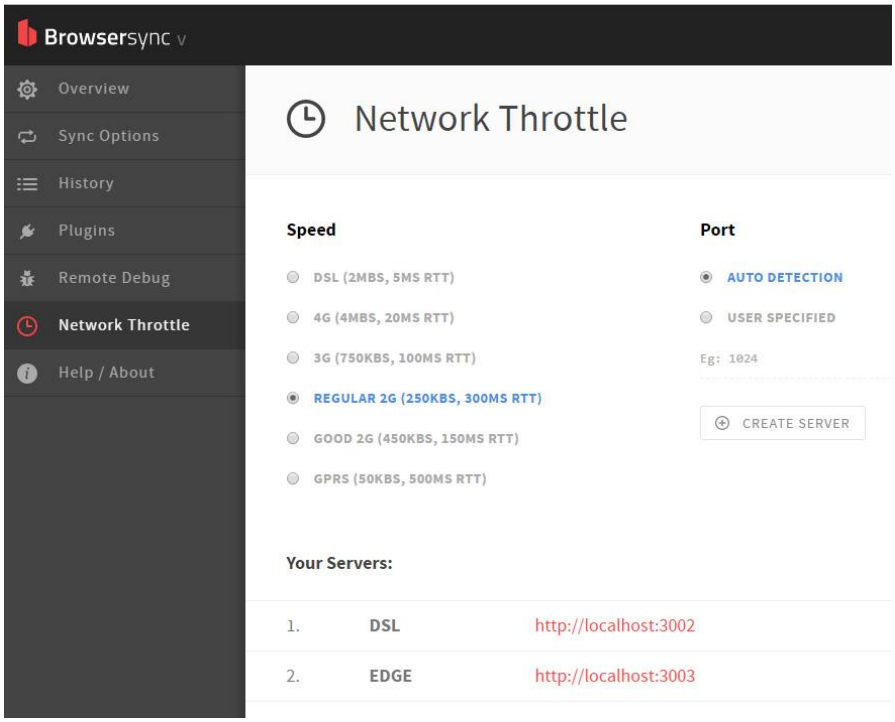
(Zdroj: vlastní tvorba)

Developer Tools dokáží také udělat audit z pohledu přístupnosti stránky. Stačí k tomu instalace vhodného doplňku. Konkrétně se jedná o rozšíření Accessibility Developer Tool, které již bylo zmíněno v kapitole 5.4.4.

8.5.4 BrowserSync

Je velice zdlouhavé testovat každou responsivní úpravu postupně v různých zařízeních. BrowserSync (dostupný na adrese https://www.browsersync.io/) umožňuje synchronizovat vyvíjenou stránku na různých zařízeních. Například tedy je možné mít stránku otevřenou zároveň na mobilním telefonu, tabletu, notebooku i velkém monitoru. Hlavní výhodou je ale vzájemná synchronizace, tzn. pokud například na mobilním telefonu bude provedeno scrollování, začne se automaticky scrollovat i na všech ostatních synchronizovaných zařízeních - je tedy možné okamžitě kontrolovat zobrazení na všech zařízeních, které jsou fyzicky k dispozici. Další funkce je například synchronizované odeslání formulářů, kdy se výsledek odeslání opět zobrazí na všech zařízeních.

V nástroji lze také simulovat pomalejší síťové připojení, kdy se na určitém portu vytvoří nová instance nástroje s požadovaným nastavení viz. obrázek 8.5.



Obrázek 8.5:

Nástroj pro testování BrowserSync

(Zdroj: vlastní tvorba)

Nástroj BrowserSync je možné do projektu nainstalovat pomocí Gulpu (nebo Gruntu) a jeho rozšíření.

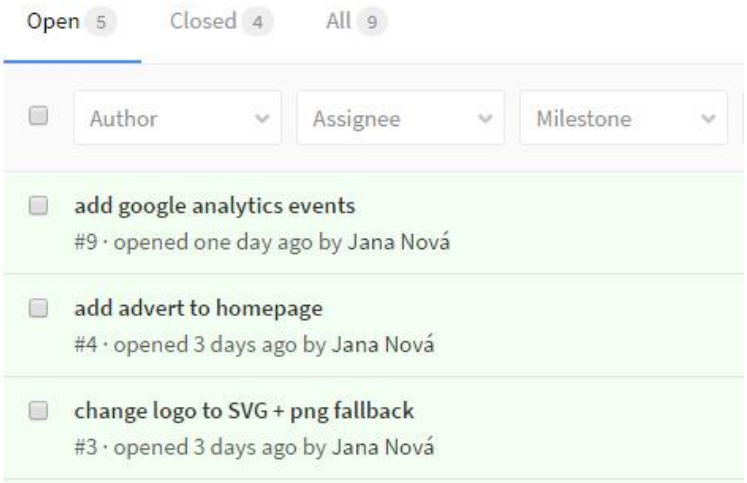
Jeho praktické použití bude ještě zmíněno v kapitole 10.7 o testování (v postupu kodování ovladače).

8.6 Verzování - GIT

Při práci v týmu je už téměř nezbytné zařadit do workflow vývoje verzovací systém, který se postará o synchronizaci souborů mezi členy týmu. Pro účely vytvoření ovladače byl použit verzovací systém GIT, který byl využit hlavně pro zvýšení přehlednosti a záloh verzí souborů. Na ovladači pracovala pouze autorka této práce a tak nebylo třeba řešit konflikty v souborech, které mohou nastat, když do repozitáře přispívá více vývojářů.

Již zmíněné vývojové prostředí PhpStorm má k dispozici také svojí verzi lokálního verzování souboru, kdy je možné podívat se do historie a vrátit se například k určité verzi, která existovala před určitým časem.

Při práci na ovladači bylo také využito webové rozhraní a soubor nástrojů pro práci s GITem - GitLab (dostupný na https://gitlab.com/). GitLab má podporu pro testování, code review, continues integration a mnoho dalších funkcí. Pro účely tvorby ovladače byl hlavně využit modul „issues", kde se udržoval seznam věcí, které je ještě nutné dořešit. Ukázka výpisu je na obrázku 8.6 , kde každá položka v sobě obsahuje základní informaci, co je třeba udělat, kdo tento požadavek zadal, kdy a bližší popis chyby (požadavku, otázky apod.). Každá položka může být spojena s konkrétním příspěvkem do repozitáře (s commitem) a kdokoliv může okomentovat každou řádku přispívaného kódu. Jednotlivé položky v modulu issues lze roztřídit do jednotlivých milníků a udržovat si tak větší přehled o seznamu nedodělků v určitých etapách vývoje.



Obrázek 8.6:

Výpis issue v nástroji GitLab

(Zdroj: vlastní tvorba z nástroje GitLab)

8.7 CSS preprocesory

Co vůbec CSS preprocesor je? Jedná se o další technologie, které do obyčejného CSS přidávají nové funkce jako proměnné, mixiny a další. V této kapitole budou představeny základní rysy CSS preprocesoru. Nejznámější z CSS preprocesorů jsou LESS, SASS a Stylus.

Pro ujasnění terminologie je třeba zmínit, že SASS jako takový obsahuje dvě syntaxe: SASS a SCSS, kde druhá zmíněná syntaxe je více podobná CSS. SASS naproti tomu je mnohem striktnější. Nepoužívá složené závorky a spoléhá na čistější coding standards, kde záleží na přesném odsazení. Pro účely vývoje ovladače byla zvolena syntaxe SCSS spolu s frontendovým frameworkem Foundation 5. (Giraudel, 2014)

Kód preprocesoru se kompiluje do výsledného CSS pomocí automatizačních nástrojů (Gulp, Grunt), speciálních aplikací (například Prepros - https://prepros.io/), příkazové řádky (v případě nainstalování vhodného prostředí) nebo přímo v IDE. (Michálek, 2014)

8.7.1 Proměnné

Věc známá z programovacích jazyků - proměnné, mají v preprocesorech svojí důležitou roli. Díky proměnným lze v CSS preprocesoru nastavit na jednom místě například hlavní

barvu celého webu, základní velikost fontu, hodnotu rozlišení v media queries apod. Pokud někdy vznikne potřeba přebarvit hlavní barvu na celém webu, pomocí proměnné se tak učiní velmi snadno na jednom místě, kde je proměnná definovaná.

Ukázka jednoduchého zápisu primární barvy v preprocesoru LESS:

```
@brand-primary: #ddd;
```

Podobně lze zapsat primární barvu i v SCSS.

```
$primary-color: #ddd;
```

8.7.2 Zanořování

Díky preprocesorům je možné zanořovat selektory do sebe. Je třeba ale dbát opatrnosti, aby se kód nestal díky mnoho násobným zanořením nečitelný. Optimální úroveň zanoření by měla být 3. (Michálek, 2014)

8.7.3 Zanořování v media queries

V CSS ve většině případů nelze zanořit deklaraci media queries do určitého selektoru. V preprocesorech to není problém a díky tomu lze psát styly pro responsivní zobrazení přímo do selektorů a organizovat tak styly do komponent. (Michálek, 2014)

8.7.4 Mixiny

Mixiny jsou části CSS kódu, které lze definovat jednou a v dalších místech kódu už jen vkládat název (případně parametry dané mixiny). Jako příklad mixiny s parametrem v syntaxi SCSS může být prefixovaný zápis CSS vlastnosti transition viz níže.

***Výpis 1**: Mixina pro prefixovanou CSS vlastnost transition*

```
@mixin transition($parametr) {
 -webkit-transition: $parametr;
 -moz-transition: $parametr;
 -o-transition: $parametr;
 transition: $parametr;
 }
```

V kódu se pak už jen použije vložení mixiny a do závorky se napíše hodnota parametru dané mixiny, tedy například:

```
.element {
 @include transition(all 100ms ease-out);
}
```

Dalšími vylepšeními mohou být například podmínky, cykly, případně funkce pro zesvětlení/ztmavení barev.

8.8 Frontendové frameworky

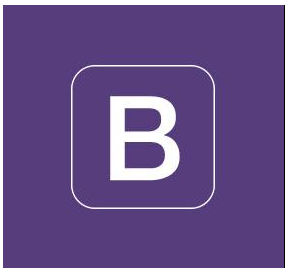
V dnešní době existuje velké množství frontendových frameworků, které mají mj. za úkol úsporu práce pro kodéra. Frontendové frameworky mohou obsahovat různé předem připravené komponenty, které může kodér jednoduše vzít, vložit do stránky a použít, případně upravit si je podle své představy. Mohou se ale také používat například k úvodnímu prototypování, k tvorbě uživatelských rozhraní a v neposlední řadě jsou velmi dobrou pomocí při tvorbě responsivních webů. Skutečnou sílu ale mají v kombinaci s některým z CSS preprocesorů, které již byly podrobněji rozebírány v předcházející kapitole.

Cílem této kapitoly je představit dva populární frontendové frameworky Bootstrap a Foundation ve verzích Bootstrap 3 a Foundation 5.

8.8.1 Bootstrap

Prvním frontendový framework, který zde bude představen je Bootstrap. Aktuálně je k dispozici ve verzi 3 (verze 4 je zatím stále Alpha). Bootstrap byl vytvořen v roce 2010 dvěma zaměstnanci Twitteru - designerem a vývojářem Markem Ottou a Jacobem Thorntonem. (Getbootstrap.com, nedatováno)

Původní použití bylo jako interní style guide, tedy jakási sada pravidel, jak by měly standardně vypadat určité prvky při vývoji nových produktů. Až v roce 2011 - konkrétně 19. srpna 2011 byl zveřejněn oficiálně pro ostatní pod licencí open source. (Getbootstrap.com, nedatováno)



Obrázek 8.7:

Logo frameworku Bootstrap

(Zdroj: http://getbootstrap.com/about)

Na obrázku 8.7 je logo Bootstrapu. Ten se ve svém brandovém manuálu od spojování s Twitterem distancuje. Dokonce výslovně upozorňuje na to, že by se v souvislosti s jeho logem neměl nikdy objevovat modrý Twitterový ptáček ani se celý framework označovat jako „Twitter Bootstrap". Spoluautor Mark Otto již v Twitteru nepracuje. V současné době působí jako vedoucí designu v GitHubu. (Getbootstrap.com, nedatováno)

Bootstrap 3 podporuje širokou škálu prohlížečů včetně nepopulárního Internetu Exploreru ve verzi 8. Toto je dobrá zpráva zejména pro uživatele operačního systému Windows XP, kteří odmítají alternativní prohlížeče, ale zároveň nemají již možnosti Internet Explorer aktualizovat. (Getbootstrap.com, nedatováno)

Velkou pomocí při návrhu a tvorby webu je mřížka (anglicky grid system nebo jen grid). Bootstrap využívá klasického 12ti sloupcového layoutu, kdy se se zvětšováním obrazové plochy zvětšuje rovnoměrně i šířka sloupců.

V základu má Bootstrap předdefinované gridové třídy jako řádku (`.row`), kontejner (`.container`) či čtyři základní breakpointy pro různá rozlišení xs (extra small), sm (small), md (medium devices) a lg (large devices).

Breakpointy

V Bootstrapu jsou definované 4 základní breakpointy:

XS - extra small - extra malé zařízení - s rozlišením do 768px

SM - small - malé zařízení - s rozlišením do 992px

MD - medium - středně velké zařízení - s rozlišením do 1200px

LG - large - velké zařízení - s rozlišením od 1200px

Tyto základní breakpointy jsou standardem pro Bootstrap 3, ale je samozřejmě možné je přepsat na libovolnou hodnotu. Toto se nejlépe dělá pomocí proměnných v preprocesoru, o kterých byla řeč v kapitole 8.7.

Kontejner

Třída `.container` má předdefinované styly pro pevné šířky kontejnerů, do kterých se vkládají příslušné řádky (`.row`). Kontejner může být buď pevný nebo plovoucí (`.container-fluid`) - ten nemá naopak definovanou pevnou šířku v pixelech, ale má jí nastavenou procentuelně na 100% šířky svého rodiče. Kontejner se také automaticky centruje na střed stránky pomocí CSS vlastnosti margin:0 auto.

Kontejnery mají nastavené pevné šířky podle breakpointu. Do 768px rozlišení není kontejner definován, respektive má automatickou (100% velikost). Pro sm breakpoint má nastaveno 750px šířku, pro md 970px a pro lg - velké zařízení 1170px. Samozřejmostí je možnost snadného přepsání hodnoty každé šířky na jednom místě pomocí proměnné v preprocesoru.

Řádka, sloupec

Řádka, definována třídou `.row` na sebe aplikuje styly záporných bočních odsazení (`margin-left` a `margin-right`), které jsou nastavené na velikost bočních paddingů u jednotlivých sloupců. Díky tomu se vyrovnává odsazení v řádce. Do řádků se následně vkládají sloupce s gridovými třídami tedy například `.col-xs-8` a `col-xs-4`. V zápisu HTML by to pak vypadalo nějak jako na obrázku 8.8:

```
<div class="container">
    <div class="row">
        <div class="col-xs-8"></div>
        <div class="col-xs-4"></div>
    </div>
</div>
```

Obrázek 8.8:

Základní použití Bootstrap gridu

(Zdroj: vlastní tvorba)

Označení `.col-xs-8` znamená sloupec, který se do breakpointu xs roztáhne na osm sloupců. Doplňuje ho sloupec `.col-xs-4`, který se roztáhne na zbývající 4 sloupce (z 12ti

sloupcového gridu). Každému bloku lze dát také třídy, které upravují jeho velikost na větších rozlišeních.

Pokud bude například do divu přidána třída `.col-sm-6` (jak znázorňuje obrázek 8.9), změní se layout stránky pro maximální rozlišení definované v breakpointu sm na rozvržení 50% pro každý sloupec (12/6).

```
<div class="container">
    <div class="row">
        <div class="col-xs-8 col-sm-6"></div>
        <div class="col-xs-4 col-sm-6"></div>
    </div>
</div>
```

Obrázek 8.9:

Rozšířené použití Bootstrap gridu

(Zdroj: vlastní tvorba)

Typografie

Základ dobrého frontendového frameworku je také typografie. Na tu má v základu Bootstrap 3 připravené předem definované třídy pro zmenšení, zvětšení, styl písma jako tučnost, kurzíva, kapitálky apod.). Samozřejmostí je také definovaná základní typografie pro sadu nadpisů h1 - h6.

Komponenty a třídy

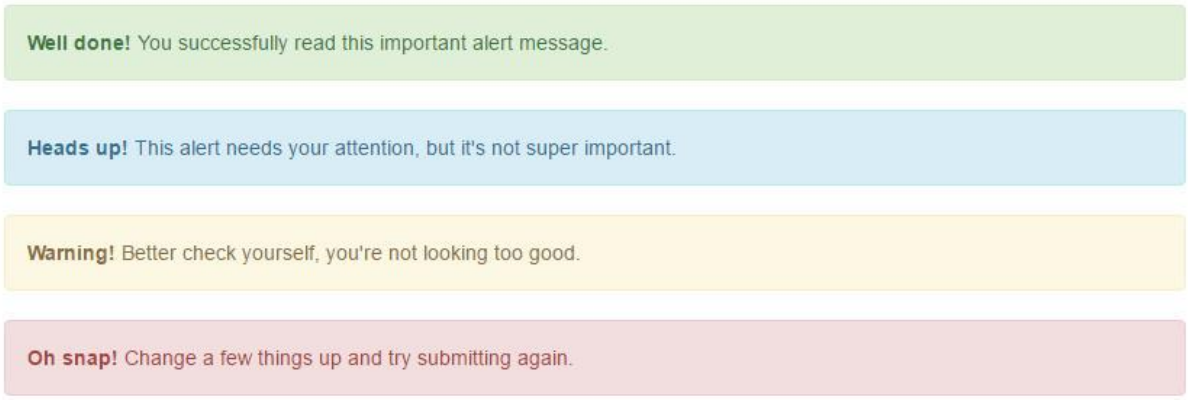
Silnou stránkou Bootstrapu je množství připravených komponent a předem definované třídy, které jsou připraveny k okamžitému použití.

Barvy

Základem pro další komponenty jsou kontextové barvy, které nabízejí hned několik základních stavů:

- success - navázáno na třídu .success, která udává úspěšně splněný stav (například odeslaný formulář apod.)
- info - navázáno na třídu .info, která reprezentuje neutrální oznámení
- warning - navázáno na třídu .warning, kterou lze označit nekritická varování
- danger - navázáno na třídu .danger, která označuje chybu, tedy například špatně odeslaný formulář

Přehled těchto barev je znázorněn na obrázku 8.10, kde byla pro ilustraci zvolena další Bootstrap komponenta alert.



Obrázek 8.10:

Bootstrap komponenta Alert

(Zdroj: http://getbootstrap.com/components/#alerts)

Tabulky

Jednoduše lze pomocí Bootstrapu formátovat také tabulky. Připravené jsou třídy:

`.table-striped` - přidá každému druhému řádku jemné podbarevení pro větší přehlednost

`.table-bordered` - orámuje každou řádku i buňku tabulky jemným šedým rámečkem

`.table-hover` - tato třída způsobí to, že po najetí myši na příslušnou buňku se orámuje celý řádek, do kterého daná buňka náleží

`.table-responsive` - poslední vybraná třída je table-responsive, která zajistí korektní zobrazení i na menších displejích

Tlačítka

Tlačítka jsou nezbytným elementem při tvorbě jakéhokoliv uživatelského rozhraní. Díky tlačítkům může uživatel interagovat s aplikací. Potvrzovat či odmítat různé akce, přecházet na další kroky apod.

Na obrázku 8.11 je možné vidět základních šest připravených tlačítek, které Bootstrap nabízí. Kromě již zmíněných barev je k dispozici i neutrální (default) tlačítko a také tlačítko primární, které by se v souvislosti s UX mělo používat na primární akce. Například u

internetového obchodu by primární akcí mělo být dokončení objednávky, proto by tlačítka „vložit do košíku" měla být právě primární barvou. Důležité také je, že tlačítka vypadají opravdu jako něco, na co se dá kliknout a nelze se je tedy splést s obyčejným textem. (Babich, 2016)

Obrázek 8.11:

Základní Bootstrap tlačítka

(Zdroj: http://getbootstrap.com/css/#buttons*)*

Třída, která zajišťuje, že se z HTML elementu <a>, <button> nebo <input> stanou odkazy je .btn.

Následnou barvu tlačítka poté můžeme definovat jednou z následujících tříd:

- `.btn-default`
- `.btn-primary`
- `.btn-success`
- `.btn-info`
- `.btn-warning`
- `.btn-danger`

Bootstrapová tlačítka mají také předem definované různé typy velikosti:

- `.btn-xs`
- `.btn-sm`
- `.btn-lg`

Jak je možné vidět, velikosti odpovídají definovaným breakpointům. Změnou velikosti tlačítka se změní velikost písma (`font-size`), řádkování (`line-height`) a vnitřní okraje (`padding`).

Formuláře

Stejně jako velikost tlačítek, lze upravovat dle stanovených breakpointů i velikost formulářových polí. Formulářové prvky lze také jednoduše vkládat do gridu. Tím se tvorba formulářů a jejich následné zobrazení na menších rozlišeních stává opravdu jednoduchou záležitostí.

Existuje také třída `.sr-only`, kterou je možné aplikovat například na formulářový prvek `<label>` a tím trochu zlepšit přístupnost formulářů. Tato třída totiž dokáže zobrazit prvek

jen pro čtečky obrazovky (screen reader). Všude jinde tento prvek nebude vidět díky nastavenému stylu: (Getbootstrap.com, nedatováno)

Výpis 2: *CSS kód třídy pro čtečky obrazovky*

Zdroj: http://v4-alpha.getbootstrap.com/getting-started/accessibility/

```
.sr-only {
  position: absolute;
  width: 1px;
  height: 1px;
  padding: 0;
  margin: -1px;
  overflow: hidden;
  clip: rect(0,0,0,0);
  border: 0;
}
```

8.8.2 Foundation

Za velmi populárním frontendovým frameworkem Foundation, který byl mj. použit i pro tvorbu ovladače, stojí společnost Zurb. Historie vytvoření frameworku se datuje do roku 2011. V roce 2013, kdy společnost Zurb existovala už patnáct let, bylo vydáno Foundation s pořadovým číslem 5, které bylo využito při tvorbě ovladače. Avšak v listopadu 2015 vyšlo zatím nejnovější Foundation 6, o kterém pojednává tato kapitola. (Zurb.com, nedatováno)

Foundation mj. obsahuje také podporu pro tvorbu responsivních emailů, kde dokáže pomocí vlastní syntaxe následně převést jednoduše do výsledné podoby tabulek, které jsou nezbytné pro zachování co nejvyšší podpory u starších klientů. (Zurb.com, nedatováno)

Také Foundation využívá CSS preprocesor - konkrétně SASS. Stáhnout lze ale také už zkompilované výsledné CSS a to používat.

Stejné tak používá Foundation také rozložení stránky do 12kového gridu. Zápis sloupců pomocí tříd je následovný: `<div class=”medium-2 columns”></div>`

Podobně jako v Bootstrapu jsou předpřipraveny tyto breakpointové třídy:

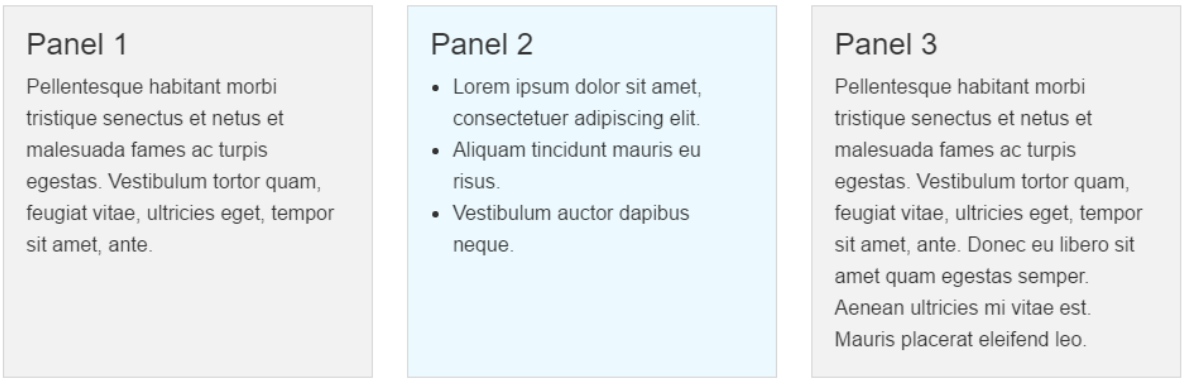
- .small-2.columns
- .medium-2.columns
- .large-2.columns

Foundation obsahuje velmi podobné prvky, které byly představeny v minulé podkapitole u frameworku Bootstrap. Pro ukázání co nejvíce možností tedy zde nebude probíhat srovnání syntaxe zápisu prvků Bootstrapu a Foundation, ale budou zde ukázány prvky takové, které v minulé kapitole představeny nebyly nebo je Bootstrap ani neobsahuje.

Equalizer

Díky funkci Equalizer je možné jen pomocí několika data atributů udržovat stejnou výšku určených bloků, které obsahují například různé množství textu (obr. 8.12).

Jednoduše stačí do obalovače jednotlivých bloků vložit data atribut `data-equalizer` a blokům, u kterých je požadováno udržovat stejnou výšku, vložit atribut `data-equalizer-watch.` Výška všech bloků pak bude závislá na výšce bloku nejdelšího.



Obrázek 8.12:

Použití rozšíření equalizer ve frameworku Foundation

(Zdroj: http://foundation.zurb.com/sites/docs/v/5.5.3/components/equalizer.html)

Off Canvas

Také vytvoření menu, které po kliku na ikonu (odkaz) vyjede z jedné nebo druhé strany viewportu a posune tak původní obsah je s pomocí Foundation opravdu jednoduché. Tato komponenta byla využita také při tvorbě ovladače. Zároveň je již připraveno spoustu SASS proměnných k přizpůsobení vzhledu modulu. HTML zápis tříd by vypadal takto:

Výpis 3:

Zápis off-canvas komponenty

Zdroj: http://foundation.zurb.com/sites/docs/v/5.5.3/components/offcanvas.html

```
<div class="off-canvas-wrap" data-offcanvas>
 <div class="inner-wrap">
   <a class="left-off-canvas-toggle" href="#" >Menu</a>
   <!-- Off Canvas Menu -->
   <aside class="left-off-canvas-menu">
       <ul>
         <li><a href="#">Item 1</a></li>
       </ul>
   </aside>
   <!-- main content goes here -->
 </div>
</div>
```

Magellan

Velmi užitečnou věc, která zvyšuje použitelnost webu je tzv. “magellan”. Jedná se o komponentu, která se zafixuje na horní část stránky - typicky pokud se jedná o navigaci na tzv. single page webu, tedy jednostránkových prezentacích, kdy jsou jednotlivé sekce webu seřazeny pod sebou a aktivní položka v navigaci se označuje dynamicky podle toho, jak moc uživatel scroluje a na kterou sekci se zrovna dívá.

V HTML zápisu se jedná následující kousek kódu:

Výpis 4: *Zápis magellan komponenty ve frameworku Foundation*

Zdroj: http://foundation.zurb.com/sites/docs/v/5.5.3/components/magellan.html

```
<div data-magellan-expedition="fixed">
 <dl class="sub-nav">
   <dd data-magellan-arrival="build"><a href="#build">Build with HTML</a></dd>
   <dd data-magellan-arrival="js"><a href="#js">Arrival 2</a></dd>
 </dl>
</div>
```

Jednotlivé cílové destinace, kde dojde ke změně aktivní položky navigace, se pak jednoduše označí data atributem, tedy například:

```
<h3 data-magellan-destination="arrival">Arrival</h3>
<a name="arrival"></a>
```

Přístupnost a Foundation

Ani Foundation nezapomíná na přístupnost a všechny její komponenty jsou vhodné pro čtečky obrazovky. Mimo to Foundation používá knihovnu „what-input", která dokáže detekovat uživatelův vstup, ať už se jedná o myš, klávesnici či dotyk. Na základě toho poté upraví jednotlivé CSS. Například tím může být upravena vlastnost outline, tedy orámování. Uživatelům, kteří používají myš, bude tato vlastnost skryta. Naopak uživatelé, kteří se na stránce pohybují pomocí klávesnice, toto orámování uvidí, protože jim bude ukazovat aktuálně vybraný prvek. (Foundation.zurb.com, nedatováno)

9 Návrh uživatelského rozhraní ovladače

Tato kapitola popisuje proces návrhu uživatelského rozhraní webového ovladače. V kapitole 4 bylo již zmíněno, jak by měl proces návrhu uživatelského rozhraní probíhat. V této kapitole je pak proces návrhu ukázán na konkrétním příkladu webového ovladače.

9.1 Požadavky na ovladač

Vzhledem k tomu, že se mohou vyskytovat různé scénáře použití - ať už se bude jednat o ovládání interaktivní obrazovky v obchodě, restauraci či v čekárně, bude ovladač používat široká veřejnost. Hlavní požadavek na ovladače je tedy jeho universálnost a snadné používání.

Používat ovladač může každý s nějakým mobilním zařízením. Jelikož je v době psaní této práce nepřeberně množství a typů mobilních zařízení, další hlavní požadavek je možnost přizpůsobení ovladače používanému viewportu, tedy velikosti obrazovky. Ovladač, který je vytvořen pomocí webových technologií musí být tedy responsivní. O způsobu testování responsivity pojednávala kapitola. 8.5.

Ovladač by také měl být připravený na různé scénáře, které lze na obrazovce ovládat - například ovládat média, pohyb, hodnotit, stahovat obsah atd. Ovládacích modulů, neboli, scénářů může být v budoucnu i více, proto další požadavek je snadné přidávání možnosti ovládání dalších scénářů.

9.2 Persony

Jak již bylo zmíněno v předchozí kapitole, ovladač může používat široká veřejnost, která vlastní některé mobilní zařízení (mobilní telefon, tablet) a je schopna ho alespoň základně používat k prohlížení webových stránek. Jelikož cílem nebylo vytvoření ovladače pro nějaký konkrétní scénář či podnik, je definování typických uživatelů velmi obtížné. K upřesnění person a jejich vlastností mohou sloužit data z měření používání ovladače (například ve službě Google Analytics) na konkrétním nasazeném případu. Z tohoto důvodu nebudou definovány konkrétní persony, ale pozornost bude soustředěna spíše na universál-

nost a snadné ovládání, které neklade uživateli žádné překážky a nenutí ho zbytečně přemýšlet.

9.3 Technické scénáře

Narozdíl od definice typických uživatelů a jejich modelů chování, je třeba při návrhu myslet i na různá omezení, které může produkt uživateli klást. Technické scénáře mají spíše funkci kontrolní a jejich cílem je zajištění přístupnosti a použitelnosti. Na technickém scénáři také může záležet volba technologií a frameworků při následné implementaci. Nezbytnou částí technického scénáře je negativní vymezení - tzn. specifikace limitů, které již nemusí být podporovány. (Staníček, 2016)

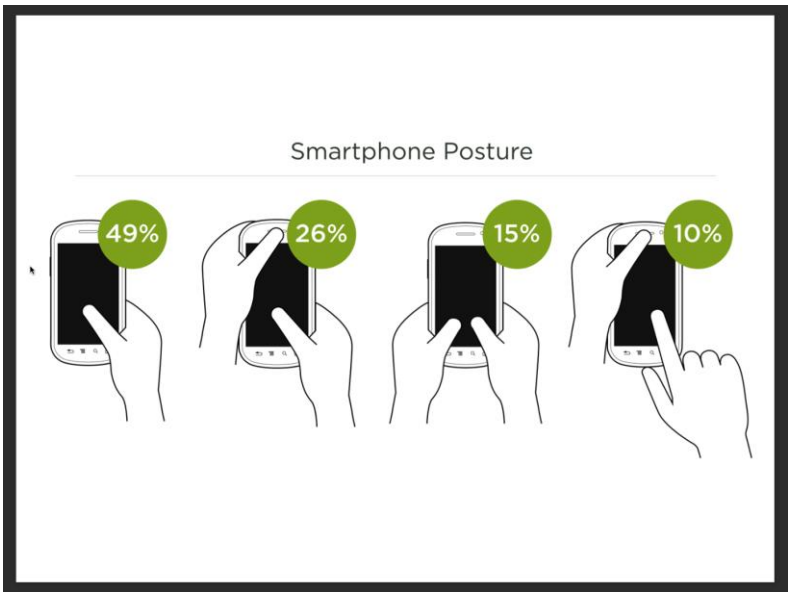
Typický technický scénář pro mobilní ovladač tedy může být například:

- Uživatel s mobilním telefonem/tabletem, který obsahuje webový prohlížeč, který podporuje technologie HTML, CSS, JavaScript. Má připojení k internetu a zařízení používá hlavně pomocí dotykových technologií.
- U displejů s rozlišením do 640px včetně bude hlavní nabídka jednosloupcová. Uživatelé s rozlišením větším než 640px budou mít nabídku na hlavní straně rozdělenou do dvousloupcového layoutu. U rozlišení větších než 1024px bude nabídka ve 4 stejně velkých sloupcích.
- Pokud zařízení nepodporuje technologie javascriptu, uživatel je o této skutečnosti informován.

9.4 Návrh wireframů

Při návrhu ovládání bylo vycházeno ze skutečnosti, že uživatelé preferují držení mobilního telefonu v 94% ve vertikálním držení (orientace portrait). Jen 6% zbývajících používá mobilní telefon horizontálně (orientace landscape). (Kvasnička, 2016)

Dalším zajímavým zjištěním jsou způsoby držení chytrých mobilních telefonů (obr. 9.1), kde převládá držení v jedné ruce (49%). (Kvasnička, 2016)



Obrázek 9.1:

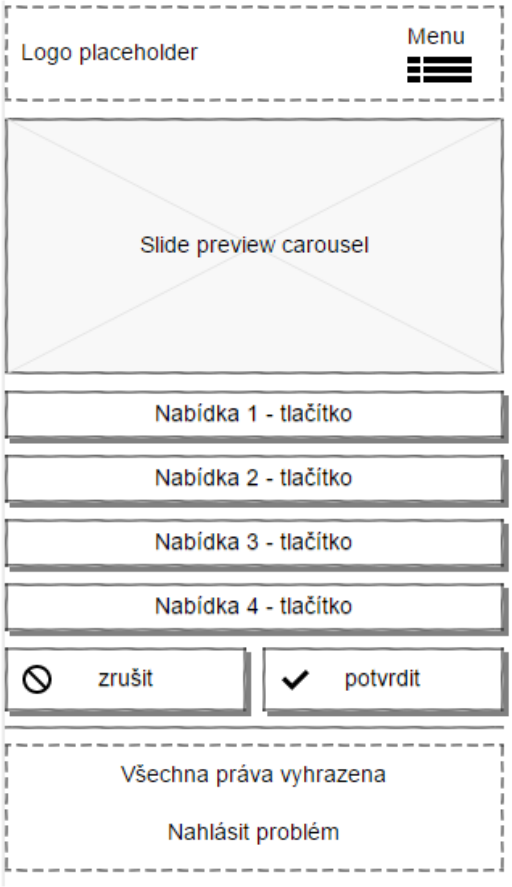
Způsob držení Smartphonu

(Zdroj: https://webexpo.cz/praha2016/prednaska/nejcastejsi-chyby-pri-navrhu-mobilniho-a-responzivniho-webu-prakticky/)

Tyto údaje představil na konferenci WebExpo 2016 v Praze odborník na UX - Jan Kvasnička.

Návrh základního layoutu tedy vycházel z toho, že většina uživatelů používá svůj mobilní telefon především v orientaci na výšku. Vycházelo se z přístupu „mobile-first", tedy rozložení se navrhovalo v první fázi především pro menší rozlišení. K návrhu wireframů byl použit nástroj Wires (dostupný na http://quirktools.com/wires/). Celkové rozvržení webu pro mobilní zařízení s 320px velikostí displeje bylo navrženo tak, jak znázorňuje obrázek 9.2.

Jednotlivé části návrhu budou podrobněji rozebrány v následujících kapitolách.



Obrázek 9.2:

Wireframe základního rozvržení ovladače

(Zdroj: Vlastní tvorba pomocí nástroje QuirkTools Wires)

9.4.1 Hlavička

V hlavičce je umístěno ilustrační logo a tzv. rozbalovací „hamburger menu", do kterého lze velmi snadno přidávat další odkazy na rozcestníky ovládání jednotlivých modulů, které se mohou využít na různé scénáře na obrazovce.

V levé horní části hlavičky je místo pro logotyp společnosti či organizace, která bude interaktivní obrazovky provozovat. A bude tak možné jednoznačně pomocí loga spojit celou aplikaci s konkrétní značkou. Jedná se tedy o prostředek pro podporu budování značky. Kromě této funkce má logo také funkci navigace, kdy je možné z jednotlivých podstránek přes logo prokliknout na hlavní rozcestník ovládání.

Pro ikonu rozbalovacího menu byl zvolen dnes již známý symbol tří na sobě poskládaných linek (někdy označované jako hamburger). Doplňkově pro snadnější pochopení funkce, kterou ikona zastává (po jejím kliku se otevře rozbalovací nabídka - obr. 9.3), byl ještě přidán popis "MENU".

Obrázek 9.3:

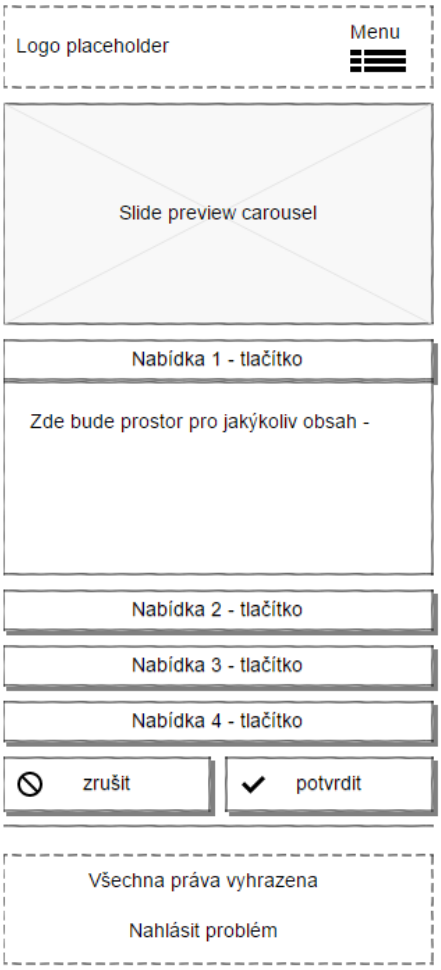
Wireframe rozbalovacího menu

(Zdroj: Vlastní tvorba pomocí nástroje QuirkTools Wires)

V rozbaleném menu lze vidět naznačené jednotlivé odkazy na konkrétní rozcestníky příslušných modulů.

9.4.2 Obsah

Hlavní složku ovladače ale bude vždy tvořit obsah. Ať už se bude jednat o prohlížení obrázků, hlasování v anketě, stažení receptů apod. Z tohoto důvodu obsahuje hlavní stránka několik naznačených rozbalovacích nabídek, jak znázorňuje obrázek 9.4.



Obrázek 9.4:

Wireframe rozbalené nabídky na hlavním rozcestníku ovladače

(Zdroj: Vlastní tvorba pomocí nástroje QuirkTools Wires)

9.4.3 Patička

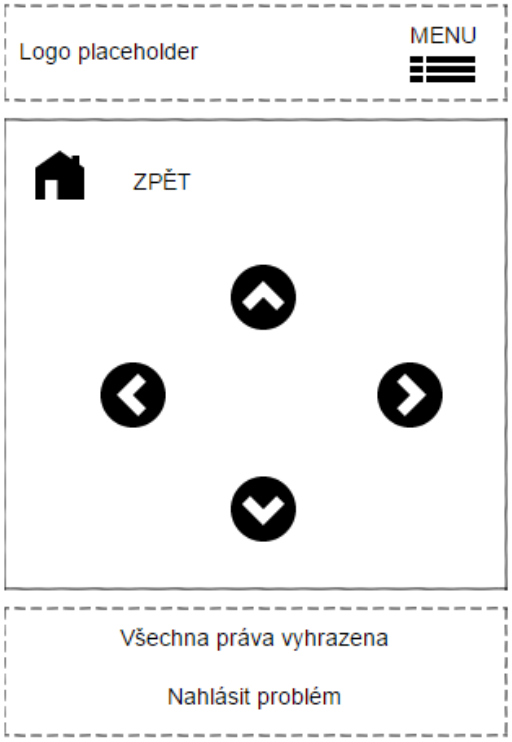
Patička ovladače je tvořena jednoduchým pruhem, který obsahuje informace o poskytovateli a odkaz na kontaktní formulář, díky kterému je možné sbírat zpětnou vazbu od uživatelů například na nefunkční zobrazení na určitých mobilních zařízení apod.

9.4.4 Rozcestník pro jednotlivé moduly

Každý scénář, který se může na obrazovce odehrávat, může vyžadovat různé množství ovládacích prvků. Aby byl zajištěný dostatečný prostor pro dotyk jednotlivých prvků, byl

návrh každého konkrétního modulu vytvořen na svém vlastním rozcestníku. Do rozcestníku modulu, jak již bylo zmíněno v minulých kapitolách, se lze prokliknout přes rozbalovací menu.

Na obrázku 9.5 je možné shlédnout rozcestník pro pohybové ovládání. V každém rozcestníku by zároveň měl být odkaz zpět na hlavní stránku ovladače (na návrhu doplněný typickou ikonou domu).



Obrázek 9.5:

Wireframe rozcestníku pro pohybové ovládání

(Zdroj: Vlastní tvorba pomocí nástroje QuirkTools Wires)

10 Kódování frontendu ovladače

V kapitole 8 již byly představeny technologie a nástroje, které byly při tvorbě ovladače použity. Tato kapitola popisuje technickou stránku procesu kódování frontendové části ovladače. Nejdříve bude v kapitole 10.1 vysvětleno, jakým způsobem byly napojeny základní knihovny (jquery), framework (Foundation) a nainstalován nástroj Gulp, který je zde využit zejména pro kompilace kódu preprocesoru do výsledného CSS a následné ladění pomocí nástroje BrowserSync. V další kapitole bude stručně popsán způsob tvorby layoutu pomocí frameworku Foundation. Kapitola 10.3 představí připravené moduly, které je možné použít na různé scénáře u aplikací na obrazovkách. Design ovladače a jednoduché možnosti jeho změny bude popsán v kapitole 10.6. Na závěr bude ukázán postup testování ovladače a konečná adresářová struktura výsledného produktu.

10.1 Základní napojení frameworku a knihoven

Jako první je třeba nainstalovat node.js (dostupný na https://nodejs.org/en/), tedy javascriptové prostředí, díky kterému je možné instalovat balíčky pomocí balíčkovacího manažera npm.

V příkazové řádce (lze i přímo ve vývojovém prostředí) je nutné zavolat: `npm install -g gulp` - tím dojde ke globální instalaci modulu gulp pro všechny projekty. Následně je třeba zavolat další příkaz: `npm install gulp --save-dev` , který nainstaluje Gulp přímo do projektu.

Při tvorbě byl použit také další balíčkovací systém, který je zaměřený především na frontend knihovny. Jmenuje se Bower (dostupný na https://bower.io/). Nejdříve je třeba, podobně jako v případě instalace Gulpu, nainstalovat Bower globálně příkazem: `npm install -g bower`.

Následně se provede instalace konkrétních knihoven do projektu. Například framework Foundation lze nainstalovat tímto příkazem: `bower install foundation`. Výhoda instalace knihoven tímto způsobem je ta, že Bower si automaticky hlídá závislé knihovny a jejich správné verze. Takže spolu s frameworkem Foundation se automaticky stáhla i například knihovna jQuery v odpovídající verzi, kterou Foundation využívá.

Pokud se následně zavolá příkaz `bower init`, a na příkazovém řádku vyplní několik informací, vygeneruje se soubor bower.json , který se již může verzovat například do Gitu. Naproti tomu složky node_modules a bower_components (které vznikly instalací balíčků popsaných výše) by se do verzovacího systému přidávat neměly. Je tedy třeba založit soubor s názvem .gitignore a do něj přidat cestu ke složkám node_modules a bower_components. (Michálek, 2016)

Aby bylo možné provádět kompilaci preprocesorového kódu do CSS, je třeba napsat tzv. „Gulp task", který toto zajistí.

Nejdříve je nutné vložit do nově založeného gulpfile.js tyto řádky:

```
var gulp = require('gulp');
var sass = require('gulp-sass');
```

Následně kód pro úlohu kompilace do CSS vypadá takto:

Výpis 5: *Kód úlohy pro kompilaci SCSS souborů do CSS*

Zdroj: vlastní tvorba

```
gulp.task('sass',function(){
    return gulp.src('app/assets/scss/*.scss')
           .pipe(sass())
           .pipe(gulp.dest('app/assets/css'))
});
```

Kód lze interpretovat tak, že po zavolání příkazu `gulp sass` se provede to, že se všechny soubory s koncovkou .scss v umístění `app/assets/scss/` zkompilují a vytvoří se v umístění `app/assets/css` jejich výsledná CSS verze. Vše je tím připraveno na stylování pomocí preprocesoru SASS.

10.2 Tvorba základního layoutu

Pro snadnější správu opakujících se částí layoutu jako je hlavička, patička, rozbalovací menu apod. byl využit jazyk PHP, který dovoluje do kódu vkládat (pomocí příkazu `include`) jakýkoliv kód. Při prací tedy bylo nutné mít vždy spuštění webový server, což zajistí již zmíněný program XAMPP.

Základ layoutu tvoří stránka index.php, do které se v hlavičce (`<head>`) vkládá soubor: `inc/inc_meta_tags.php`, který obsahuje napojení stylů a fontů. Napojení javascriptových souborů je provedeno až těsně před ukončovacím HTML tagem `</body>`v souboru `inc/inc_js_body_end.php` z důvodu neblokování vykreslování stránky.

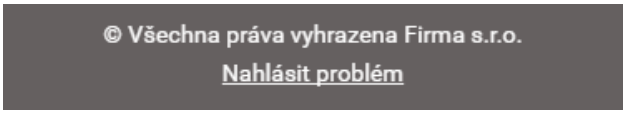
Vkládány jsou také soubory `inc_header.php` a `inc_footer.php`, kde se nachází opakující se hlavička (s rozbalovacím menu) a patička.

Obrázek 10.1:

Nakodovaná hlavička ovladače

(Zdroj: Vlastní tvorba)

Celá patička je tvořena jedním souborem inc_footer.php a obsahuje jednoduchou HTML strukturu, která je roztažena vždy na 100% šířku. V patičce se také nachází odkaz na kontaktní formulář, díky kterému je možné získávat od uživatelů zpětnou vazbu například na špatné zobrazení na některých mobilních zařízeních.



Obrázek 10.2:

Nakodovaná patička ovladače

(Zdroj: Vlastní tvorba)

Jak již bylo zmíněno v kapitole 9.4.2, hlavní část ovladače bude tvořit vždy obsah. Proto na úvodní stránce, tedy hlavním rozcestníku, je připraveno několik nabídek, které po kliknutí rozbalí další obsah, jak naznačuje obrázek 10.3. Počet nabídek lze jednoduše přidávat.

Obrázek 10.3:

Rozbalená nabídka na hlavním rozcestníku ovladače

(Zdroj: Vlastní tvorba)

Ilustrační obrázky, které obsahuje nabídka se díky lightboxu, tedy rozbalovací galerii z frameworku Foundation, nechají jednoduše po kliknutí roztáhnout na maximální možnou šířku podle šířky viewportu. Vše, co je třeba udělat, je vložit třídu `.clearing-thumbs` do HTML seznamu (`<ul>`), mít správně vložený javascriptový modul a přidán data atribut `data-clearing` taktéž do `<ul>` kontejneru jednotlivých obrázků.

Výše zmíněné prvky tvoří hlavní layout ovladače. Následující kapitola popisuje přehled jednotlivých modulů a způsob jejich přidávání do ovladače.

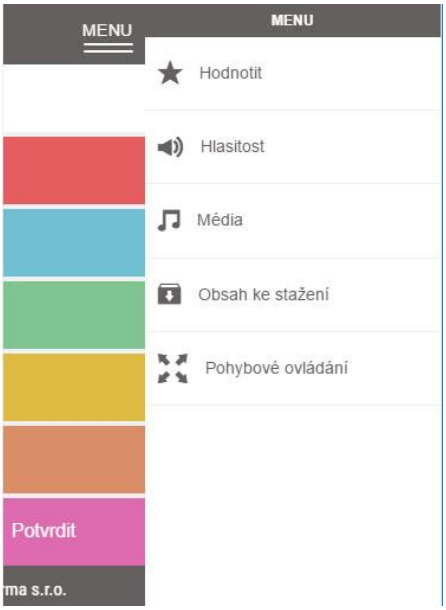
10.3 Přidávání jednotlivých modulů

Ovladač má aktuálně připravený frontend pro několik scénářů, které je možné na obrazovce s příslušnou backendovou podporou (která není součástí této práce) vykonávat. Jedná se o následující moduly:

- modul hodnocení
- modul pro ovládání hlasitosti
- modul pro přehrávání médií
- modul pro obsah ke stažení
- modul pro pohybové ovládání

K tomu, aby bylo možné jednotlivé moduly lehce zapínat a vypínat byl vytvořen konfigurační soubor `cfg.php`, který se vkládá na začátek každé stránky ovladače. Jednotlivý modul se pak zapíná jednoduše změnou logické hodnoty true/false, tedy například: `$GLOBALS['ENABLE_MOVE'] = false;`

Pokud je modul zapnutý, tak se v rozbalovacím menu (obr 10.4) přidá tlačítko s ikonou a názvem, které odkazuje do detailu ovládání modulu.

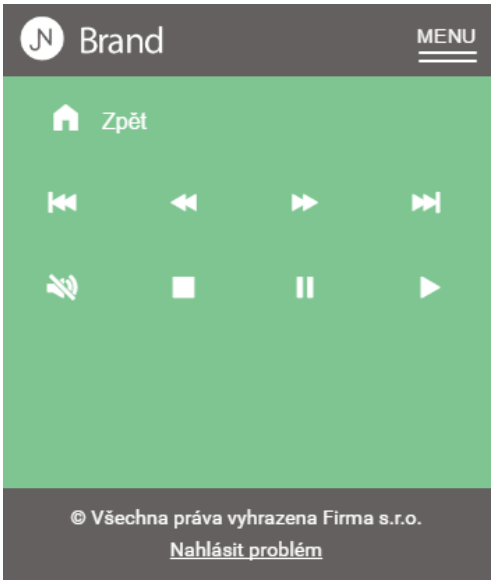


Obrázek 10.4:

Nakodované rozbalené menu ovladače

(Zdroj: Vlastní tvorba)

Samotný detail je poté tvořen vlastními ovládacími prvky a odkazem na úvodní stránku Na obrázku 10.5 je detail ovládání pro media. Detaily všech modulů lze nalézt v příloze A.



Obrázek 10.5:

Rozcestník pro modul media

(Zdroj: Vlastní tvorba)

Jednotlivé ovládací prvky jsou tvořeny vždy odkazem a font ikonou ze sady, kterou nabízí přímo Foundation na adrese http://zurb.com/playground/foundation-icon-fonts-3

V HTML struktuře vypadá zápis jednoho ovládacího prvku (jedné ikony) následovně:

```
<a class="small-3  columns item">
<em class="ico fi-previous i-fs-s"></em>
</a>
```

Pro přidání ikony stačí elementu `<em>` vložit třídu označující příslušnou font ikonu, tedy například: `.fi-previous` . Další třída `.i-fs-s` poté označuje velikost vložené ikony, která je nastavena třídě `.i-fs-s` v CSS pomocí vlastnosti `font-size`.

Jako doplňující byly vytvořeny ještě podstránky s kontaktním formulářem a chybová stránka 404.

10.4 Kontaktní formulář

Stránka `contact-us.php`, která obsahuje kontaktní formulář je odkazována z patičky každé podstránky ovladače. Smyslem tohoto formuláře je pohodlné zajištění zpětné vazby od uživatelů a případné reportování chyb v zobrazení na různých mobilních zařízeních.

Odesílání formuláře je zajištěno souborem `send.php` (na který je formulář napojen). V tomto souboru je zajištěna validace povinných polí a vlastní odesílání je realizováno PHP funkcí:

`mail( string $to , string $subject , string $message)`. Možné vylepšení do budoucna by mohlo spočívat v přidání formulářového pole pro nahrání souboru (například špatného screenu a napojení na pokročilejší PHP knihovnu pro posílání emailů - PHPMailer (dostupný na https://github.com/PHPMailer/PHPMailer).

Vzhled formuláře na telefonu a zobrazení stavu odeslaného formuláře, který neprošel validací, je možné shlédnout v příloze B.

10.5 Chybová stránka - 404

Při procházení webu se někdy stane, že stránka, na kterou se odkazuje, neexistuje. Může to být způsobeno různými situacemi. Ať už se jedná o překlep v adrese nebo situaci, kdy požadovaná stránka byla smazána či přemístěna, pro uživatele to znamená, že nenašel to, co hledal a musí se rozhodnout, co dále. Pokud se mu objeví nic neříkající kód - chyba 404

apod., může být velmi zmatený a ze stránky odejít. Aby se tak nestalo, měli by tvůrci webů pamatovat na možnost přizpůsobit chybové stránky. Minimum, které může každý tvůrce udělat je upravit chybovou stránku tak, aby zapadala do designu celého webu a obsahovala odkaz na úvod, případně rozcestník s navigací na nejdůležitější stránky webu.

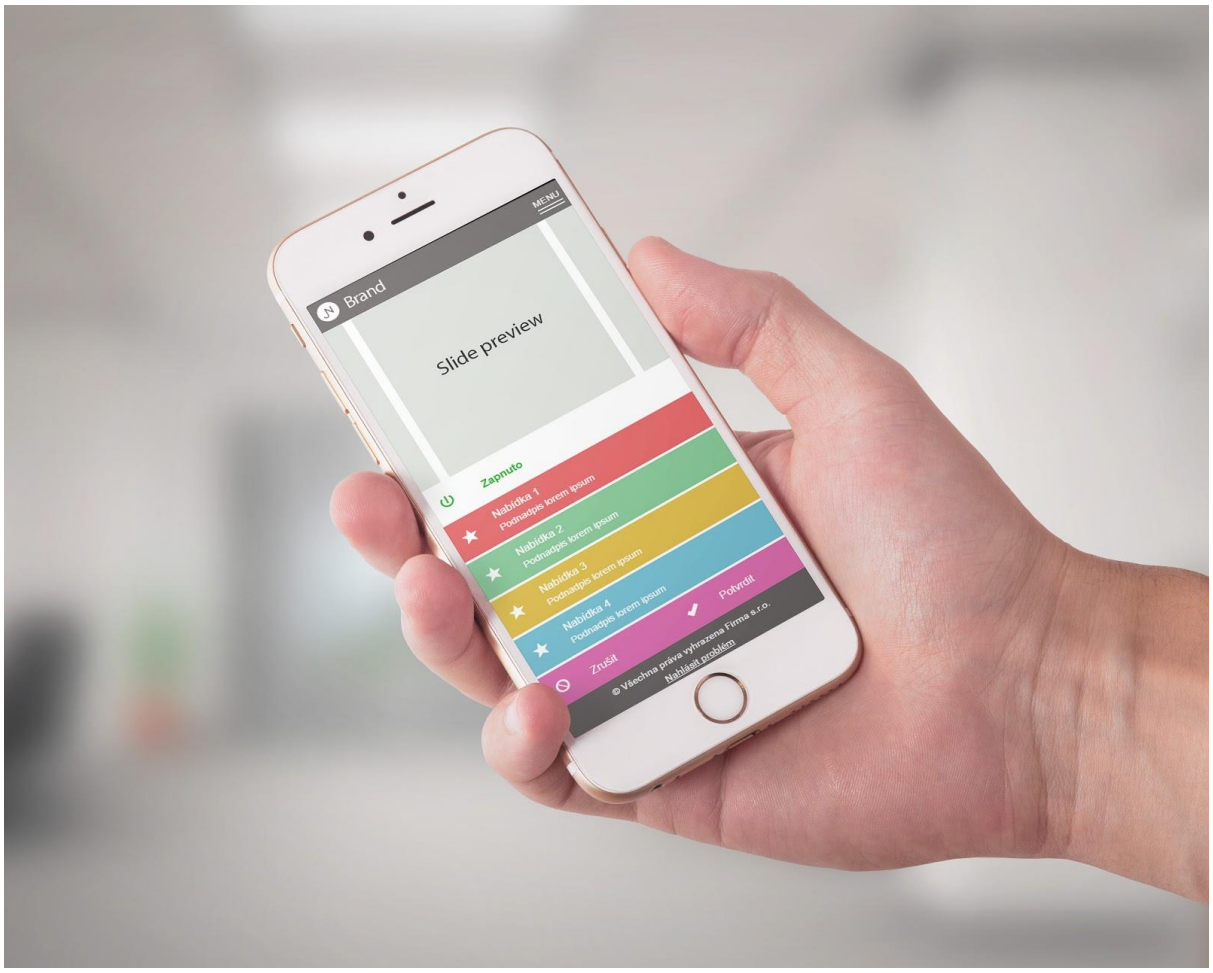
Také při tvorbě ovladače byla tato stránka realizována. Její podobu je možné shlédnout v příloze C.

Technická realizace není nikterak složitá. Stačilo vytvořit stránku `404.php` se základním rozložením a napojit tuto stránku do souboru `.htacces` (umístěný v rootu webu). Samotné napojení je zajištěné direktivou `ErrorDocument 404 /app/404.php`

10.6 Design ovladače

Konečná podoba ovladače byla tvořena přímo upravováním CSS stylů na navržené struktuře a lze jí vidět na obrázku 10.6. Pokud by se v praxi jednalo o větší a rozsáhlejší projekt - webovou stránku, e-shop, katalog atd., bylo by třeba, před samotnou implementací, vytvořit ještě grafický návrh. V případě ale takto malého rozsahu byl krok grafického návrhu vynechán.

Konečná podoba designu ovladače byla situována do tzv. flat designu, kde se klade důraz především na jednoduchost. Tedy v designu byla snaha o odstranění jakýchkoliv rušivých prvků jako stíny, rámečky apod. (May, 2016)



Obrázek 10.6:

Konečná podoba ovladače

(Zdroj: Vlastní tvorba pomocí MockupFree.co)

Díky SCSS proměnným je možné definovat na jednom místě hodnotu barvy a napříč celým designem používat jen tyto proměnné. V případě ovladače byly definovány následující hlavní barvy, které je možné použít například na barvy pozadí jednotlivých nabídek na úvodní stránce.

- `$color-red: #e55d5f;`
- `$color-blue: #71bfd3;`
- `$color-green: #7fc591;`
- `$color-yellow: #dfbb43;`
- `$color-orange: #d98e67;`
- `$color-purple: #de6bb0;`

V příloze D lze nalézt další náhled designu ovladače.

10.7 Testování

Jako hlavní nástroj pro testování správného responsivního zobrazení byl použit nástroj BrowserSync. Spouštění je možné provádět pomocí nástroje Gulp, kdy pro využití nástroje je třeba přidat do souboru gulpfile.js následující řádky:

1) Na začátek souboru zajistit vložení modulu.

```
var browserSync = require('browser-sync').create();
```

2) Do Gulp úlohy, která řeší kompilaci preprocesorového kódu vložit následující řádek

```
.pipe(browserSync.stream());
```

3) Do Gulp úlohy, která hlídá změny v souborech

```
browserSync.init({
          proxy: "http://localhost/_SKOLA/diplomka/app"
    });
```

a

```
gulp.watch("./*.php").on('change', browserSync.reload);
```

Celý soubor gulpfile.js je možné prohlédnout v příloze E.

Jako doplňkový nástroj pro testování správného responsivního zobrazení byl zvolen nástroj Browserstack, o kterém byla řeč v kapitole 8.5.1.

Obecně u webových stránek, které je požadováno prohlížet na mobilních zařízeních, kde může být připojení k internetu velmi omezené, je třeba testovat také množství přenesených dat a počet požadavků na server. Tyto dvě hodnoty je třeba co nejvíce minimalizovat. Testování počtu požadavků a množství dat lze velmi jednoduše změřit na záložce „Network" v Google DeveloperTools.

Pro účely snížení datového objemu výsledného CSS kódu slouží další z Gulp automatizačních úloh - gulp-clean-css. Tato úloha načte všechny CSS z určené složky a zminifikuje tyto soubory do jiné vybrané složky. Soubory je možné tak spojit do jednoho a snížit tak požadavky na server.

10.8 Adresářová struktura

Následující kapitola uzavírá praktickou část a tak v ní spíše jen pro doplnění bude uvedena adresářová struktura se stručným vysvětlením. Obrázek struktury z vývojového prostředí PhpStorm je uvedena v příloze F.

V kořenovém adresáři se nacházejí složky *app, bower_components* a *node_modules*. Poslední dvě jmenované složky obsahují různé knihovny, které byly do projektu přidány pomocí příkazů `npm install` a `bower install`. Nejdůležitější je složka *app*.

Složka *app* obsahuje dále složky *assets, docs a inc*. *Docs* slouží pouze pro ukládání v tomto případě ilustračních dokumentů ke stažení. Do složky *inc* se vkládají jednotlivé opakující se části, které tvoří layout.

Ve složce assets je 6 podsložek:

- *css* - zkompilované SCSS styly,
- *dist* - zminifikované styly ze složky css,
- *icons* - font ikony Foundation,
- *images* - ilustrační obrázky jako logo apod.,
- *js* - javascriptový soubor main.js,
- *scss* - styly preprocesoru SASS.

11 Závěr

Hlavním cílem práce bylo vytvořit responsivní frontendovou část univerzálního ovladače interaktivních obrazovek pomocí moderních frontendových technologií. Ke splnění hlavního cíle vedly především kapitoly osm devět a deset, kde byly postupně představeny nástroje a technologie, které se používají v odvětví dnešního frontendu. Dále byl v kapitole devět vytvořen návrh uživatelského rozhraní s využitím wireframů. Následně byl celý frontend nakódován a otestováno bylo také jeho responsivní zobrazení především pomocí nástroje BrowserSync a Browserstack. Oba zmiňované nástroje byly také teoreticky představeny v kapitole 8.5.

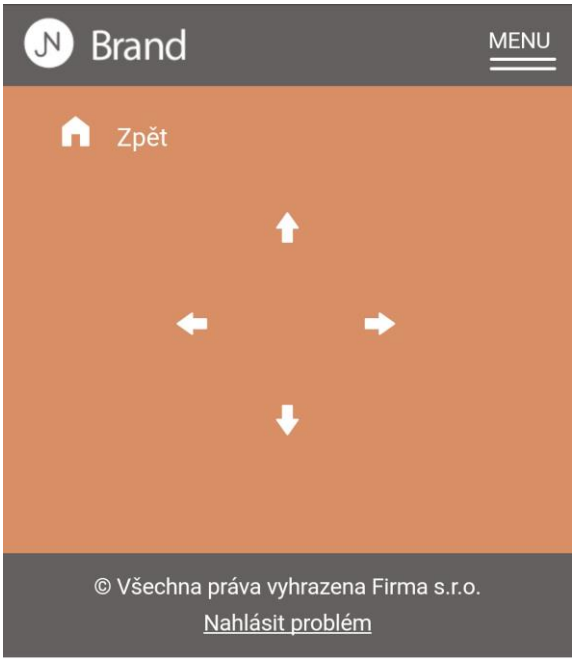
Možné rozšíření vytvořeného ovladače by mohlo spočívat v nasazení některého ze šablonovacích systémů (například Smarty nebo Latte) a samozřejmě také vytvoření dalších modulů pro různé scénáře.

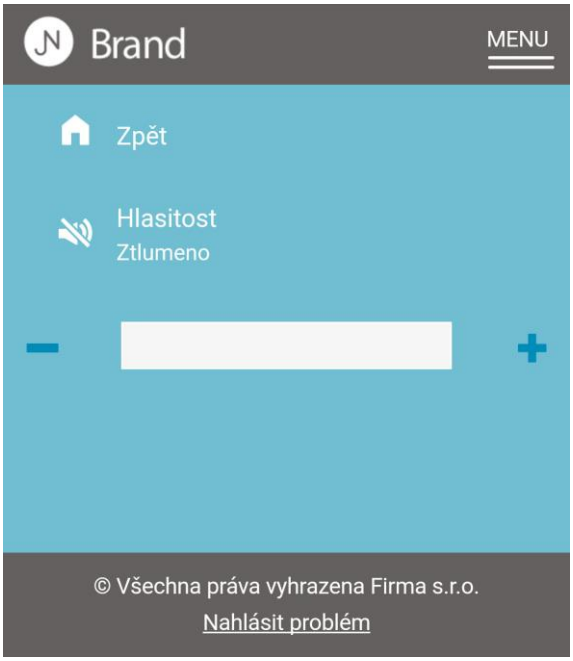
Dílčím cílem bylo vytvořit stručného průvodce nástroji a technologiemi, které využívá moderní frontend, jako jsou například CSS preprocesory, frontendové frameworky a nástroje pro automatizaci úloh na frontendu. Splnění tohoto cíle bylo dosaženo v kapitole osm, kde je v kontextu tvorby ovladače čtenář proveden od úplného začátku celého procesu kódování, tedy od výběru vhodného vývojového prostředí, přes představení nástrojů pro automatizaci úloh a nástrojů pro testování responsivního zobrazení až po možnosti verzování vytvořených zdrojových kódu. Kapitola osm dále přibližuje dvě hlavní části moderního frontendu, tedy CSS preprocesory, konkrétně LESS a SASS a frontendové frameworky Bootstrap a Foundation.

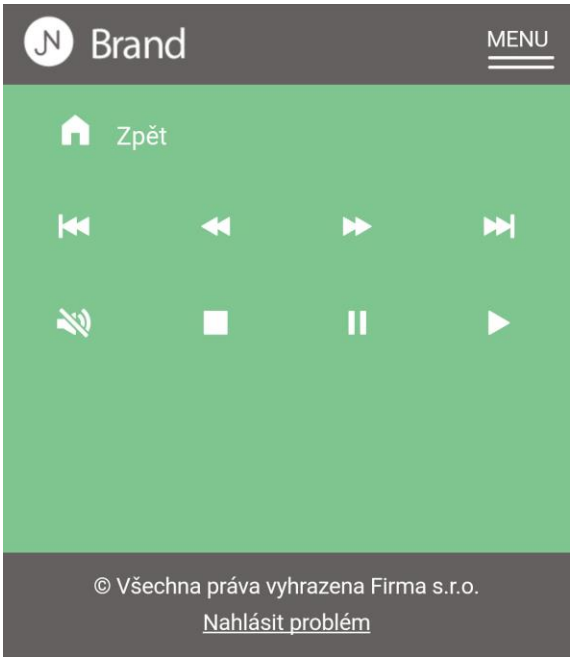
Dalším dílčím cílem bylo seznámení čtenářů se základními principy přístupnosti a použitelnosti i se zaměřením na mobilní weby a aplikace. Tento cíl byl naplněn v kapitole pět, šest a sedm. V kapitole pět byl čtenář seznámen s různými druhy handicapů v situaci procházení webových stránek. Dále zde byla uvedena konkrétní vybraná pravidla přístupnosti jak podle vyhlášky č 64/2008 Sb., tak také podle pravidel přístupnosti WCAG20 dle W3C. Vybrané příklady použitelnosti byly zmíněny v kapitole šest. V obou kapitolách pět i šest byly také představeny nástroje pro testování přístupnosti a použitelnosti. Kapitola sedm nakonec dává do souvislosti všechny tři pojmy – přístupnost, použitelnost a UX a definuje tzv. tři úrovně designu. Dále kapitola sedm popisuje, jaký je rozdíl mezi mobilní a responsivní verzí webu a uvádí také vybrané tipy pro lepší uživatelský zážitek při procházení webu na mobilních zařízeních.

Práce může být přínosná pro čtenáře, kteří se specializují na tvorbu především frontendových částí webových stránek. Díky velkému množství zmíněných nástrojů pro testování ať už přístupnosti, použitelnosti či responsivního zobrazení, může čtenář získat nové tipy na nástroje pro kontrolu webu z různých úhlů pohledu. Pro začínajícího frontend kodéra může mít tato práce velký přínos v podobě zvýšení efektivity své práce díky komplexnějšímu přehledu a vysvětlení moderních frontendových nástrojů a technologií, se kterými se nemusel čtenář doposud setkat.

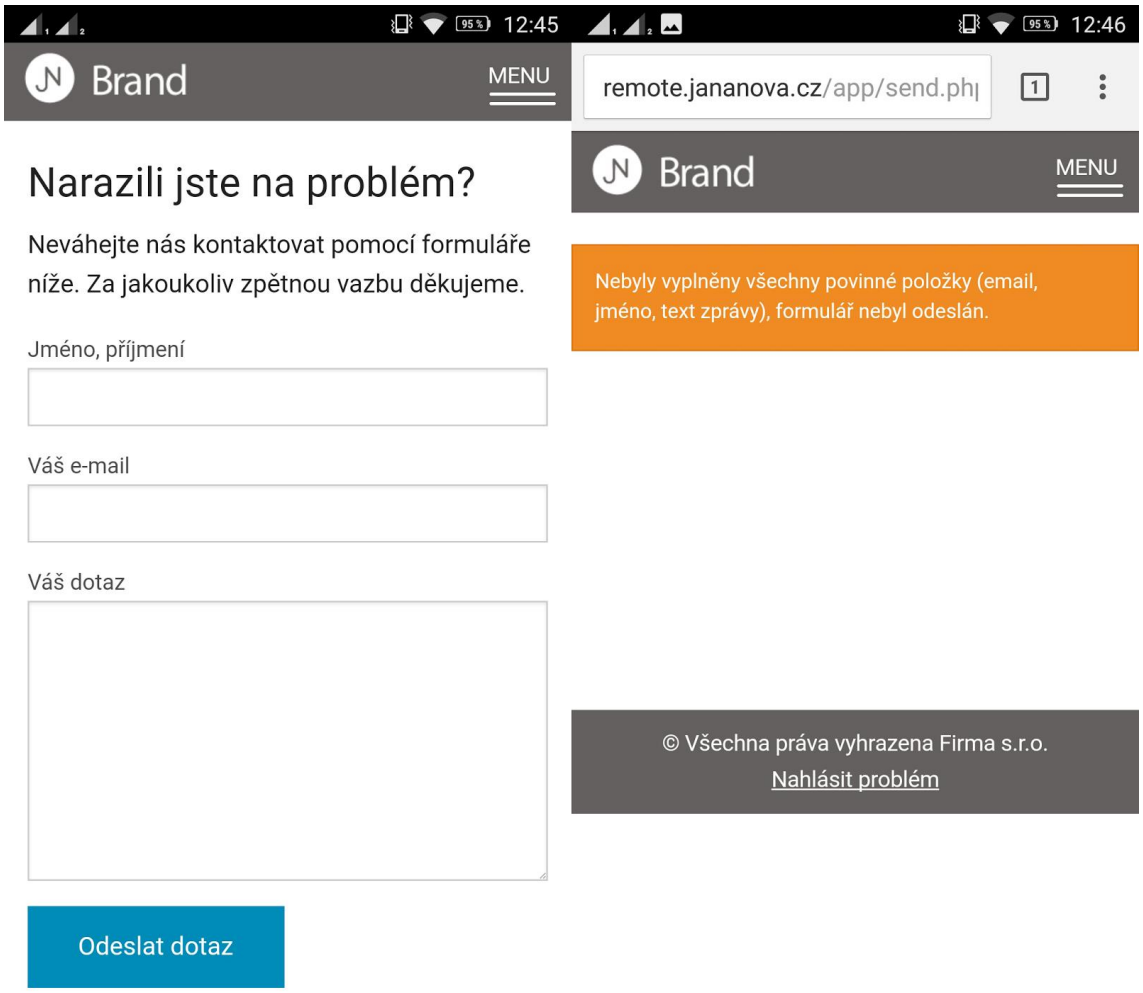
Příloha A: Screeny jednotlivých rozcestníků modulů





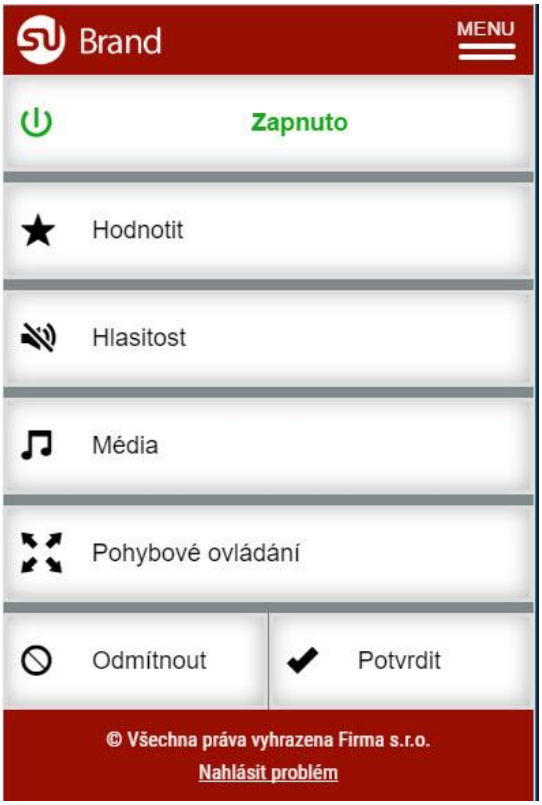


Příloha B: Kontaktní formulář a stav po jeho odeslání



Příloha C: Chybová stránka 404

Příloha D: Jiný design ovladače

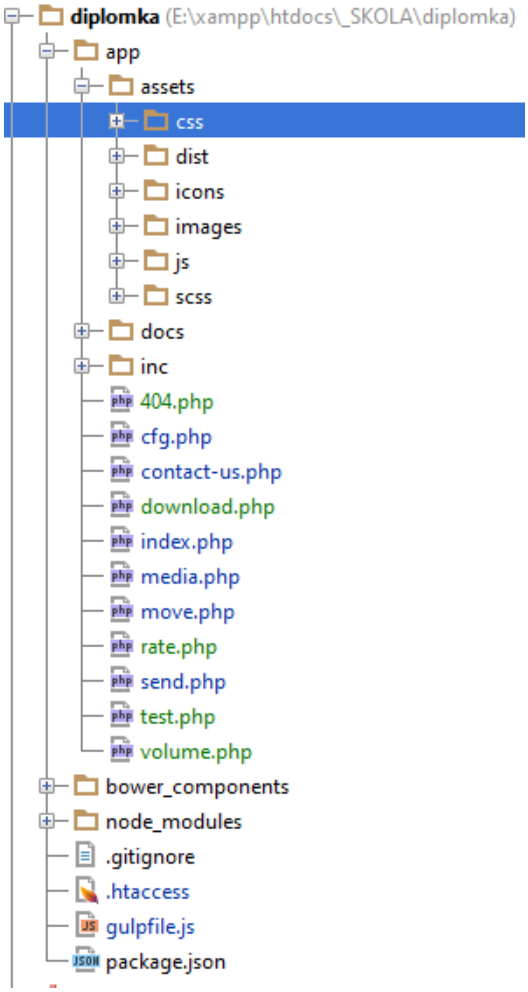


Příloha E: Soubor gulpfile.js

```
var gulp = require('gulp');
var sass = require('gulp-sass');
var browserSync = require('browser-sync').create();
var cleanCSS = require('gulp-clean-css');
gulp.task('sass',function(){
    return gulp.src('app/assets/scss/*.scss')
           .pipe(sass())
           .pipe(gulp.dest('app/assets/css'))
           .pipe(browserSync.stream());
});
gulp.task('watch',function(){
    browserSync.init({
          proxy: "http://localhost/_SKOLA/diplomka/app"
    });
    gulp.watch('app/assets/scss/*.scss',['sass']);
    gulp.watch("./*.php").on('change', browserSync.reload);

});
gulp.task('minify-css', function() {
    return gulp.src('app/assets/css/remote.css')
           .pipe(cleanCSS({compatibility: 'ie8'}))
           .pipe(gulp.dest('app/assets/dist'));
});
gulp.task('default',['watch']);
```

Příloha F: Adresářová struktura frontendu ovladače



Terminologický slovník

Termín	Zkratka	Význam [zdroj]
GIT	GIT	Jeden z typů verzovacího systému pro zdrojové kódy.
Vývojové prostředí	IDE	Sada nástrojů pro usnadnění práce vývojáře při vývoji softwaru. (Integrated Development Enviroment)
SASS	SASS	Typ preprocesoru CSS kódu. (Giraudel, 2014)
SCSS	SCSS	Jedna ze syntaxí preprocesoru SASS. (Giraudel, 2014)
Optimalizace pro vyhledávače	SEO	Sada technik, jejíž cíl je posunout webovou stránku na co nejlepší pozice ve výsledcích vyhledávání. (Search Engine Optimization)
Výsledky vyhledávání	SERP	Seznam výsledků vyhledávání vrácených vyhledávačem. (Search Engine Result Page)
Wireframe		Drátěný model – rozkreslený návrh (webu) používaný za účelem náčrtu struktury a uspořádání jednotlivých prvků.

Použité informační zdroje

Anon., 2014. *What's the Difference Between Sass and SCSS?.* [Online]
Available at: https://www.sitepoint.com/whats-difference-sass-scss/
[Přístup získán 30 Říjen 2016].

Babich, N., 2016. *Button UX Design: Best Practices, Types and States.* [Online]
Available at: http://babich.biz/button-ux-design-best-practices-types-and-states/
[Přístup získán 3 Listopad 2016].

Čápka, D., 2013. *2. díl - UML - Use Case Diagram.* [Online]
Available at: http://www.itnetwork.cz/navrhove-vzory/uml/uml-use-case-diagram/
[Přístup získán 26 Únor 2016].

Dickey, T. E., nedatováno *Lynx is the text web browser.*. [Online]
Available at: http://lynx.invisible-island.net/
[Přístup získán 30 Březen 2016].

Dmarketing.cz, 2013. *Digital signage v restauracích veřejného stravování..* [Online]
Available at: http://www.dmarketing.cz/2013/05/digital-signage-v-restauracich-verejneho-stravovani/
[Přístup získán 1 11 2016].

Dostál, M., 2007. *Základy tvorby uživatelského rozhraní.* [Online]
Available at: https://phoenix.inf.upol.cz/esf/ucebni/gui-dostal.pdf
[Přístup získán 5 Březen 2016].

e15.cz, 2012. *Jaká bude budoucnost digital signage.* [Online]
Available at: http://strategie.e15.cz/special/jaka-bude-budoucnost-digital-signage-938518
[Přístup získán 15 02 2016].

Foundation.zurb.com, nedatováno *Foundation and Accessibility.* [Online]
Available at: http://foundation.zurb.com/sites/docs/accessibility.html#foundation-and-accessibility
[Přístup získán 5 Listopad 2016].

Getbootstrap.com, nedatováno *About - Bootstrap.* [Online]
Available at: http://getbootstrap.com/about/
[Přístup získán 3 Listopad 2016].

Getbootstrap.com, nedatováno *Browser and device support - Bootstrap.* [Online]
Available at: http://getbootstrap.com/getting-started/#support
[Přístup získán 3 Listopad 2016].

Getbootstrap.com, nedatováno *Forms.* [Online]
Available at: http://getbootstrap.com/css/#forms
[Přístup získán 4 Listopad 2016].

Giraudel, H., 2014. *What's the Difference Between Sass and SCSS?.* [Online]
Available at: https://www.sitepoint.com/whats-difference-sass-scss/
[Přístup získán 30 Říjen 2016].

H1.cz, nedatováno *A/B testování.* [Online]
Available at: http://www.h1.cz/a-b-testovani
[Přístup získán 15 Březen 2016].

Ilinčev, O., 2016. *Příručka marketéra: Nejhorší chyby webových formulářů.* [Online]
Available at: http://tyinternety.cz/prirucka-marketera/prirucka-marketera-17-nejhorsich-chyb-webovych-formularu/
[Přístup získán 22 Září 2016].

Jahoda, B., 2015. *Vliv rychlosti webu na SEO.* [Online]
Available at: http://jecas.cz/seo-rychlost
[Přístup získán 3 Duben 2016].

Krug, S., 2010. *Nenuťte uživatele přemýšlet.* místo neznámé:COMPUTER PRESS. ISBN 80-251-1291-8

Kvasnička, J., 2016. *Nejčastější chyby při návrhu mobilního a responzivního webu prakticky.* [Online]
Available at: https://webexpo.cz/praha2016/prednaska/nejcastejsi-chyby-pri-navrhu-mobilniho-a-responzivniho-webu-prakticky/
[Přístup získán 11 11 2016].

Lireo.com, 2014. *Definition of Usability.* [Online]
Available at: https://www.lireo.com/definition-of-usability/
[Přístup získán 3 Duben 2016].

May, T., 2016. *The beginner's guide to flat design.* [Online]
Available at: http://www.creativebloq.com/graphic-design/what-flat-design-3132112
[Přístup získán 8 Listopad 2016].

Mediar.cz, 2016. *Digital signage: tahoun roku 2016 v místě prodeje.* [Online]
Available at: http://www.mediar.cz/digital-signage-tahoun-roku-2016-v-miste-prodeje/
[Přístup získán 15 02 2016].

Michálek, M., 2014. *Průvodce CSS preprocesory: co a jak?.* [Online]
Available at: http://www.vzhurudolu.cz/blog/12-css-preprocesory-1
[Přístup získán 30 Říjen 2016].

Michálek, M., 2016. *Instalace ekosystému Node.js pro použití na frontendu.* [Online]
Available at: http://www.vzhurudolu.cz/prirucka/node-instalace
[Přístup získán 9 Listopad 2016].

Michálek, M., 2016. *Jak zničit mobilní uživatele podruhé: zakažte zoomování nebo schovejte obsah.* [Online]
Available at: http://www.vzhurudolu.cz/blog/48-znicit-mobilistu-2
[Přístup získán 9 Srpen 2016].

Michálek, M., 2016. *Jak zničit mobilní uživatele? Pomalým načítáním, karusely nebo schováním navigace.* [Online]
Available at: http://www.vzhurudolu.cz/blog/47-znicit-mobilistu-1
[Přístup získán 10 Srpen 2016].

MVCR.cz, 2008. *Vyhláška č. 64/2008 Sb., o formě uveřejňování informací souvisejících s výkonem veřejné správy prostřednictvím webových stránek pro osoby se zdravotním postižením (vyhláška o přístupnosti) - Ministerstvo vnitra České republiky.* [Online]
Available at: http://www.mvcr.cz/clanek/vyhlaska-c-64-2008-sb-o-forme-uverejnovani-informaci-souvisejicich-s-vykonem-verejne-spravy-prostrednictvim-webovych-stranek-pro-osoby-se-zdravotnim-postizenim-vyhlaska-o-pristupnosti-10.aspx
[Přístup získán 16 Březen 2016].

Navrátil, P., 2012. *Kdo je to UX designér?.* [Online]
Available at: http://blog.lupa.cz/uxdesign/kdo-je-to-ux-designer/
[Přístup získán 15 Březen 2016].

Nielsen, J., 2002. *WEB.DESIGN.* místo neznámé:Softpress. ISBN 80-86497-27-5.

Ožana, R., 2014. *Gulp vs. Grunt: souboj bez vítěze a poraženého.* [Online]
Available at: https://www.zdrojak.cz/clanky/gulp-vs-grunt-souboj-bez-viteze-a-porazeneho/
[Přístup získán 25 Říjen 2016].

Pavlíček, R., 2009. *Web Content Accessibility Guidelines 2.0.* [Online]
Available at: http://blindfriendly.cz/wcag20/
[Přístup získán 30 Březen 2016].

Pristupnost.cz, nedatováno *Česká pravidla přístupnosti - Přístupnost.cz.* [Online]
Available at: http://www.pristupnost.cz/jak-tvorit-pristupny-web/pravidla-pristupnosti/ceska-pravidla-pristupnosti/
[Přístup získán 19 Březen 2016].

Publi.cz, nedatováno *Uživatelské rozhraní.* [Online]
Available at: https://publi.cz/books/11/12.html
[Přístup získán 22 02 2016].

Řezáč, J., 2014. *Web ostrý jako břitva.* místo neznámé:Baroque Partners. ISBN: 978-80-87923-01-6.

Staníček, P., 2016. *Dobrý designér to všechno ví!.* 1. editor Kamenné Žehrovice: vydáno vlastním nákladem autora. ISBN 978-80-260-9427-2.

Takaki, M., J. C. & Phan, D., 2016. *Finding more mobile-friendly search results.* [Online]
Available at: https://googlewebmastercentral.blogspot.cz/2015/02/finding-more-mobile-friendly-search.html
[Přístup získán 9 Srpen 2016].

Techterms.com, 2009. *User Interface: TechTerms.com.* [Online]
Available at: http://techterms.com/definition/user_interface

[Přístup získán 22 02 2016].

Usability.gov, nedatováno *Heuristic Evaluations and Expert Reviews.* [Online]
Available at: https://www.usability.gov/how-to-and-tools/methods/heuristic-evaluation.html
[Přístup získán 10 Březen 2016].

Vataha, P., 2015. *MOBILNÍ NEBO RESPONZIVNÍ – VŠE CO POTŘEBUJETE VĚDĚT.* [Online]
Available at: http://www.antstudio.cz/blog/mobilni-nebo-responzivni-vse-co-potrebujete-vedet/
[Přístup získán 9 Srpen 2016].

Vernerová, T., 2013. *UX techniky - Analýza.* [Online]
Available at: http://www.otestujweb.cz/2013/06/ux-techniky-2.html
[Přístup získán 26 Únor 2016].

Zurb.com, nedatováno *A brief timeline.* [Online]
Available at: http://zurb.com/about/history
[Přístup získán 5 Listopad 2016].

Zurb.com, nedatováno *Foundation for Emails 2.* [Online]
Available at: http://foundation.zurb.com/emails.html
[Přístup získán 5 Listopad 2016].