Pedagogická fakulta
Faculty of Education

Jihočeská univerzita v Českých Budějovicích
University of South Bohemia in České Budějovice

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH

Pedagogická fakulta

Katedra informatiky

Front-End webové aplikace vědeckého časopisu Studia Kinanthropologica

Front-End of the web application for the scientific journal Studia Kinanthropologica

Bakalářská práce

Vypracoval: Marek Musil

Vedoucí práce: PaedDr. Petr Pexa, Ph.D.

České Budějovice 2017

Prohlášení

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

V Českých Budějovicích dne 5. dubna 2017

Marek Musil

Abstrakt

Moje práce pojednává o vytvoření klientského rozhraní webové aplikace vědeckého časopisu Studia Kinanthropologica. Na druhé části webu (back-endu) pracoval můj kolega, se kterým jsem byl stále v kontaktu, abychom tyto dvě části průběžně spojovali do konečného redakčního systému pro nahrávání a správu časopisu. Beru v potaz i požadavky katedry tělesné výchovy, která tento časopis vydává. Samozřejmostí je využití nejmodernějších technologií pro tvorbu webu, jako je HTML5, CSS3, Javascript a jiné. Nechybí ani kompletní přizpůsobení pro všechny velikosti displejů a monitorů. Zaměřuji se také na optimalizaci, použitelnost a maximalizaci vizuálního výkonu aplikace za účelem efektivního zobrazení služeb a obsahu. Web obsahuje dvě jazykové verze (českou, anglickou), formuláře a také administrátorskou sekci. Tento redakční systém by měl usnadnit a zpřehlednit práci autorům článků a recenzentům časopisu.

Klíčová slova

web, front-end, html5, css3, javascript, AJAX, responzivita, redakční systém

Abstract

My thesis discuss creating a client interface of a scientific journal Studia Kinanthropologica web application. On the second part of the web (back-end) worked my colleague and I was in touch with him all the time, to regularly merge these two parts until the final content management system for uploading and managing the journal. I take into account the requirements of the department of sports studies, which publishes this journal. I surely use the newest technologies such as HTML5, CSS3, Javascript and the other. There also is a complete adaptation to all sizes of displays and monitors. I am also focusing on optimization, usability and maximizing visual performance of the application in order to effectively display services and content. The web contains two language versions (Czech, English), forms and also the administration section. This content management system should facilitate and improve the work of the authors of the articles and the reviewers of the journal.

Keywords

web, front-end, html5, css3, javascript, AJAX, responsivity, content management system

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH
Fakulta pedagogická
Akademický rok: **2015/2016**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Marek MUSIL**
Osobní číslo:	**P14224**
Studijní program:	**B7507 Specializace v pedagogice**
Studijní obor:	**Informační technologie a e-learning**
Název tématu:	**Front-End webové aplikace vědeckého časopisu Studia Kinanthropologica**
Zadávající katedra:	**Katedra informatiky**

Zásady pro vypracování:

Cílem bakalářské práce bude vytvoření klientského rozhraní (tzv. Front-end) webové aplikace vědeckého recenzovaného časopisu Studia Kinanthropologica, který je vydáván katedrou tělesné výchovy a sportu PF JU. Vlastní grafický návrh bude s využitím webových technologií HTML5, CSS3, jQuery, Javascript, AJAX aj. sestaven do výsledné prezentační vrstvy pro registrované autory článků, recenzenty a administrátory. Samozřejmostí bude i responzivita webu pro přehledné zobrazení i na mobilních zařízeních, autor se dále zaměří na optimalizaci, použitelnost, přehlednost a maximalizaci vizuálního výkonu webu za účelem efektivního zobrazení služeb a obsahu. Aplikace bude obsahovat i administrátorskou sekci, která bude dostupná po přihlášení.

Rozsah grafických prací: **CD ROM**

Rozsah pracovní zprávy: **40**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

1. **BÖTTIGHEIMER, Armin. Vytváříme grafiku webu ve Photoshopu: průvodce tvorbou ikon, bannerů a navigačních panelů. Vyd. 1. Brno: Computer Press, 2011. ISBN 978-80-251-3485-6**
2. **DAWSON, Alexander. Výjimečný webdesign: jak tvořit osobité, přitažlivé, použitelné weby. 1. vyd. Brno: Computer Press, 2012. ISBN 978-80-251-3719-2**
3. **SHARKIE, Craig a Andrew FISHER. Responzivní webdesign: okamžitě. 1. vyd. Brno: Computer Press, 2015. ISBN 978-80-251-4384-1**
4. **W3Schools [online]. [cit. 2016-03-31]. Dostupné z: http://www.w3schools.com/**
5. **CSS3 Tipy a Triky [online]. [cit. 2016-03-31]. Dostupné z: http://www.css.chobits.ch/**

Vedoucí bakalářské práce: **PaedDr. Petr Pexa, Ph.D.**
Katedra informatiky

Datum zadání bakalářské práce: **19. dubna 2016**

Termín odevzdání bakalářské práce: **28. dubna 2017**

Mgr. Michal Vančura, Ph.D.
děkan

L.S.

doc. PaedDr. Jiří Vaníček, Ph.D.
vedoucí katedry

V Českých Budějovicích dne 19. dubna 2016

Poděkování

Rád bych poděkoval panu PaedDr. Petru Pexovi, Ph.D. za pomoc a vedení této bakalářské práce. Cením si jeho stálé ochoty a rad, které mi v průběhu tvorby dával. Také si velice vážím veškeré podpory ze strany mé rodiny a přátel.

Obsah

1 Úvod

Bakalářská práce se zabývá tvorbou klientského rozhraní, tzv. front-endu, webové aplikace vědeckého časopisu Studia Kinanthropologica. Jedná se o částečné předělání v tuto chvíli již fungujícího webu časopisu, kde jsou základní informace o časopisu, archiv starších vydání, kontakt a další. Dále se pak hlavně věnuje vytvoření kompletně nové administrativní části webu, kam budou mít uživatelé přístup po přihlášení a kde budou moci provádět, na základě jejich práv, různé operace, jako například odesílat články ke schválení, mazat články a tak dále. To vše za pomoci technologií HTML5, CSS3, Javascript a AJAX.

V práci stručně uvádím jak vůbec takový projekt od základu vzniká, jeho definice a návrh, jaké mohou nastat problémy a jak je případně řešit. V další části rozebírám můj postup návrhu a implementace, kde porovnávám i jiné alternativní metody práce. Z hlediska vzhledu představuji tzv. Material design definovaný společností Google a vysvětluji jednotlivé kroky jeho přenesení do projektu. Zabývám se i problematikou implementace mého grafického rozhraní s back-endovou částí aplikace. Na závěr celý web testuji pro zajištění co nejlepší funkčnosti při různých rozlišení displeje a použitém prohlížeči.

Pro lepší přehlednost budu web, který je již vytvořen nazývat „úvodní web" a druhou část, kam se uživatel dostane až po přihlášení, „backoffice".

1.1 Východiska práce

Práce vychází z potřeb katedry tělesné výchovy, která by uvítala webovou aplikaci pro správu a publikaci svého vědeckého časopisu. Redakční systém je od základu vytvořen na základě požadavků katedry. Již běžící web časopisu obsahuje pouze jednoduché informace a je třeba jeho částečná úprava a přidání backoffice.

Redakční systém je software zajišťující správu dokumentů, nejčastěji webového obsahu. CMS (z anglického content management system) se uplatní všude tam, kde se obsah často mění, přidává ho více lidí, nebo je požadována jeho pohodlná správa. [3]

S pohodlnou správou zcela jistě souvisí vnější vzhled systému a pojem UX design. UX je zkratkou již zavedeného anglického pojmu User eXperience. Česky, ne zcela věrně, se překládá jako uživatelský prožitek. Navrhování a vytváření těchto prožitků se zabývá právě tím, aby ono používání bylo příjemné, užitečné, budilo pozitivní emoce, abychom si z něj odnášeli příjemný zážitek a pozitivní zkušenost a zanechalo v nás příjemnou stopu. [1]

K tomu dopomáhají nejen moderní technologie jako je například značkovací jazyk HTML5, či třetí verze kaskádových stylů CSS, ale i techniky responzivního

webdesignu (zkráceně RWD). Ty poskytují návštěvníkům stránek optimální prožitek a dokáží ušetřit spoustu času, protože nemusíte vytvářet jedinečná řešení pro každého uživatele a zařízení zvlášť. [2]

1.2 Cíle práce

Cílem práce je vytvoření klientského rozhraní webové aplikace tak, aby splňovala všechny stanovené normy v oblasti vývoje webů a obsahovala všechny funkce požadované katedrou tělesné výchovy. Ty následně implementuji do back-endové části aplikace. Dále se zaměřuji na design a používám takové postupy, aby byl web přístupný pro co nejširší spektrum uživatelů, bez ohledu jaké zařízení k prohlížení využívají.

Součástí je i navrhování logické, datové a navigační struktury. U některých kroků případně vysvětluji funkčnost a účel jednotlivých komponent. Z práce je patrné i s jakými částmi byly problémy a jak jsem je vyřešil, nebo mezi čím se rozhodovalo a proč jsem si nakonec vybral danou věc, či postup.

1.3 Metody práce

V mé práci analyzuji požadavky katedry a stručně představuji základní části designu. Dále se zmiňuji o variantách, které se dají použít při tvorbě front-endu. Součástí použité varianty je i stručný popis vytvoření layoutu v jazyce HTML5 a jeho nastylování v CSS3. Zmiňuji i jak probíhá vzájemná spolupráce s kolegou, který pracuje na back-endu a jakým způsobem se to vše implementuje. Klíčová je i fáze testování, kde například testuji responzivitu na různých zařízeních a funkčnost v rozdílných prohlížečích.

2 Návrh a realizace

2.1 Zadání projektu

Prvním krokem každého návrhu projektu je vždy zjistit od klienta maximálně přesné zadání. To znamená, jaké jsou jeho zájmy a cíle, a jaké jsou potřeby cílové skupiny uživatelů, kteří budou web navštěvovat. S vymezením těchto pojmů může pomoci například vzorový dotazník, kde klientovi pokládáme otázky jako, o jaký typ webu půjde, jaký účel bude plnit, jaká je cílová skupina, zda má klient určen nějaký obchodní model a tak dále. Je toho spousta, na co se můžeme ptát, ale tento krok je opravdu velmi důležitý ve specifikaci projektu. V mém případě se nejedná o tak rozsáhlý projekt a tak stačila osobní schůzka, případně e-mailová korespondence.

Jak v knize zmiňuje Petr Staníček, musíme zohlednit, že zde může nastat tzv. projektová mezera. Jedná se o chybějící propojení znalostních bází naší a našeho klienta. Ani jedna ze stran nerozumí byznysu strany druhé, neorientuje se v jejích specifikách, nezná její terminologii. Je právě na designérovi, aby tuto mezeru vyznačil a obě strany propojil do funkčního celku. Proto je třeba shromáždit takto velké množství informací, následně tyto informace analyzovat, vytvořit podrobné zadání, navrhnout datové i navigační struktury a chování jednotlivých částí. [1]

Až to vše bude přesně vyjasněno, zadáno a schváleno, pak se teprve přistupuje k samotnému grafickému návrhu aplikace. Mnohdy se stává, že se designér po úvodním zadání rovnou vrhne do navrhování rozložení prvků, barev a podobně, ale tímto uspěchaným rozhodnutím vznikají chyby, které se budou postupně nabalovat a stěžovat práci, až do konce projektu. Pokud se vůbec takovýto problémový projekt ke svému konci dostane.

V mém případě byla poptávka vytvořena katedrou tělesné výchovy a sportu, která potřebovala vytvořit webovou aplikaci pro jednoduchou správu svého několik let vydávaného časopisu. Návrh se týká i již běžícího webu časopisu, kde bylo třeba nejen přidělat formulář pro přihlašování, ale i částečně upravit rozložení prvků a lépe navrhnout responzivitu. Jelikož zadavatel nemá jiné požadavky k tomuto webu, zbytek zůstal ve stejném stavu. Svou pozornost jsem tedy zaměřil na vytvoření zcela nové stránky, na kterou se bude přihlašovat několik typů uživatelů s různými právy, co mohou a nemohou dělat. Například po přihlášení autora článků se mu zobrazí nabídky pro přidání, či editaci článku.

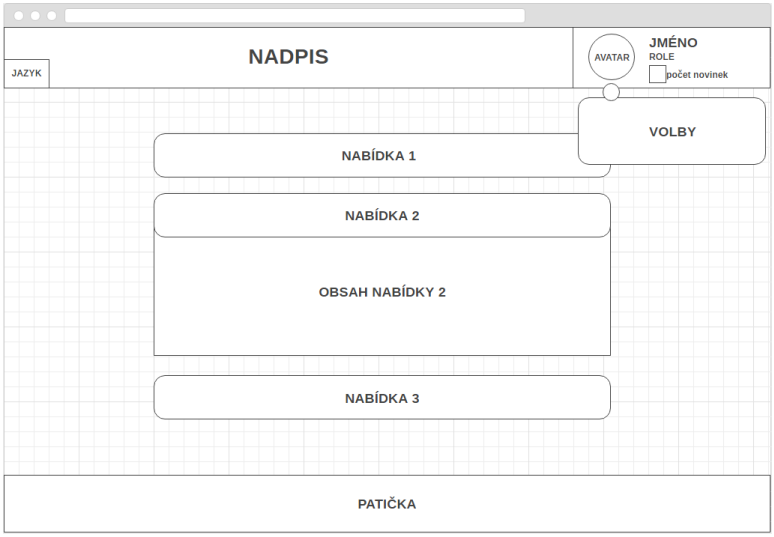
Jedním s důležitých požadavků je, aby stejně jako úvodní web, nabízel i backoffice dvě jazykové verze, a to českou a anglickou. Další věcí je přístupnost nejen z hlediska různých zařízení (responzivita), ale i z pohledu různých uživatelů. Tím je myšleno hlavně přizpůsobení webu pro osoby s postižením.

2.2 Analýza

V oblasti analýzy a návrhu webových, či jiných aplikací, se používá drátěný model (tzv. wireframe). Vytvořením tohoto modelu můžeme lépe definovat návrh a obsah aplikace, v mém případě webových stránek.

Wireframe definuje rozmístění funkčních prvků na stránce. Nejedná se v žádném případě o grafický návrh, neobsahuje obrázky a je tvořen pouze pomocí čar a textu. Nedoporučuje se ani použití barev, až na výjimky, které je potřeba odlišit. [5]

Ve svém návrhu drátěného modelu jsem využil *use case* diagram, vytvořený kolegou, jehož úkolem je back-endová část aplikace. Jeho diagram mi napověděl, jaké všechny funkce by měla aplikace obsahovat. Díky této informaci byl poté promyšlen počet, vzhled a velikost jednotlivých ovládacích prvků tak, aby byly dobře viditelné (například při různém rozlišení obrazovky), byla z nich jasně patrná jejich funkce, a aby zajistili uživateli co možná nejjednodušší a nejrychlejší cestu k jeho cíli.



Obrázek 1: Wireframe

Tímto mohu rovnou navázat na pojem CTA (Call to Action). V návrhu internetových stránek je CTA banner, tlačítko, nebo nějaký druh grafiky či textu, který vyzývá uživatele, aby na něj kliknul a nasměroval se tak blíže jeho cíli. CTA může být jednoduchý nenáročný požadavek jako „vyberte barvu", nebo „prohlédni si toto video". Je zřejmé, že se CTA používá k nabádání uživatelů ke koupi výrobku, nebo poskytnutí osobních údajů a kontaktních informací. [6]

2.3 Definice principů

Ze všeho nejdříve je třeba si vymezit mantinely, stanovit rámcové zásady, obecná pravidla a principy. Rozhodnutí, jaké bude celkové pojetí a styl webu, jaká bude komunikace s uživatelem a forma prezentace informace, je zde důležité. Podstatné je i stanovení priorit tak, aby se příslušná cílová skupina dostala co nejdříve k jádru sdělení. Vedle celkového stylu musíme specifikovat i formu, jakou chceme informace prezentovat. Například zda bude web komunikovat primárně textově, nebo spíše prostřednictvím obrázků či infografiky. Nedílnou součástí jsou i metastránky a různé podpůrné nástroje. Sem můžeme zařadit stránky, které stojí tak trochu mimo hlavní strukturu webu (přihlašování, odesílání zpráv, nebo třeba i různé formuláře a jiné). Nesmíme zapomínat ani na SEO optimalizaci pro vyhledavače. [1]

2.4 Dostupné technologie

Při realizaci tohoto projektu bylo zapotřebí promyslet, zda použiji nějaký framework, jako je například Bootstrap, či Foundation, nebo si vystačím s vlastním řešením. Již hotová řešení nabízí mnoho předpřipravených nástrojů, které se jednoduše vloží do stránky a nechybí ani nativní responzivita. Takto rychlému a vcelku jednoduchému vytváření aplikací se říká „rapidní prototypování". Ne vždy se ale takovýto přístup hodí zrovna pro náš projekt, pracovní postup, nebo styl programování.

V mém projektu se jedná o celkem jednoduchou strukturu prvků webu, a proto nepotřebuji využívat tolik různých funkcionalit nabízených právě frameworky. Na základě těchto informací jsem se rozhodl pro vlastní řešení, za použití nejnovějších standardů z oblasti vývoje webu a to HTML5, CSS3 a velmi zajímavou technologii asynchronního načítaní AJAX. Ta umožňuje aktualizaci webových stránek vyměňováním dat s webovým serverem. To znamená, že je možné aktualizovat části webu, bez načtení celé stránky znova. Vše se děje na pozadí, skrytě před uživatelem.

2.5 Material design

V souvislosti s tvorbou vzhledu jsem se nechal inspirovat tzv. Material designem, který byl v roce 2014 představen společností Google. [7]

V aplikaci využívám nejen jeho principů návrhu, ale i s tím spojenou sadu ikon, volně dostupných pod licencí Apache License Version 2.0. [8]

2.5.1 Úvod

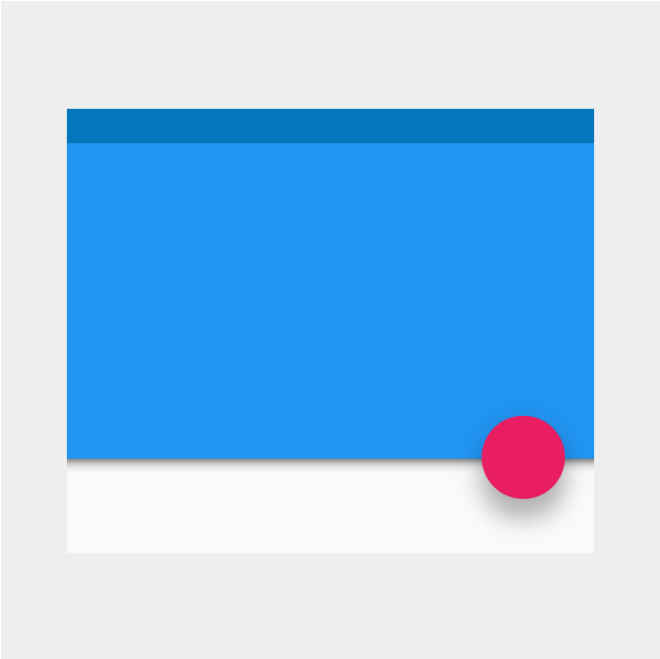
Material design je vizuální jazyk pro uživatele, jenž slučuje klasické principy dobrého designu s inovacemi a možnostmi technologie a vědy. Tato specifikace je aktualizována, její principy a specifika jsou neustále rozvíjeny. Jako cíl si klade vytvoření jednotného elementárního systému, který sjednocuje zážitek napříč platformami s různými velikostmi zařízení. Pravidla při vyvíjení na mobilní zařízení zůstávají podstatná, ale dotek, hlas, myš a klávesnice jsou nejvíce upřednostňovanými vstupními metodami. [9]

2.5.2 Principy

Tento design je teorie racionalizovaného prostoru a systému pohybu. Použití známých vlastností, týkajících se doteku, pomáhá uživatelům rychle pochopit účel prvku. Podstata světla, povrchu a pohybu jsou klíčem k pochopení, jak se objekty pohybují, komunikují, a jak existují v prostoru a ve vztahu k sobě navzájem. Realistické osvětlení ukazuje spoje, rozděluje prostor a indikuje pohyblivé části. Základní elementy zde tvoří typografie, mřížky, prostor, měřítko, barvy a metafory. Tyto prvky dělají mnohem více, než jen pěkný vzhled. Utváří hierarchii, smysl a zaměření. Klade se důraz na akce uživatelů, aby byla u prvku ihned jasně určena jeho funkcionalita. [9]

Material design má velmi mnoho různých principů a stylů návrhu. Z toho důvodu zmiňuji pouze některé z nich a zaměřuji se hlavně na ty, které jsem použil ve svém projektu.

Jedním z takových stylů jsou elevace a stíny. Ty dodávají objektům virtuální hloubku na jinak plochém displeji. Prvky s odlišnými funkcemi mají na stránce různé stíny, určující v jaké úrovni osy Z se nacházejí. Pomáhá to tak odlišit několik na sobě ležících a překrývajících se komponentů. [10]



Obrázek 2: Zobrazení stínů překrývajících se komponentů

V Material designu se také využívá pohyb, sloužící k zobrazení vztahu a hierarchie mezi prvky. Animace také navádí uživatelovu pozornost k další položce. Pohyb by měl být svižný a plynulý. [11]

Nezanedbatelnou součást návrhu tvoří i výběr vhodných barev a barevných kombinací, poskytujících celému webu jasný a logický systém. Když si vybereme nějakou barvu, je dobré na ostatní prvky používat podobný barevný tón, aby jsme se vyhnuli přemíře různých variací. V Material designu jsou doporučovány základní barevné palety, které od sebe vhodně oddělují prvky z různými funkcemi.

Důležité také je, aby žádná z informací nebyla sdělována pouze pomocí barvy. Tím se vyhneme problému, kdy mohou lidé s poruchami vidění a barvocitu tuto informaci špatně interpretovat či vůbec nezaregistrovat. Záleží i na vhodně zvoleném kontrastu všech prvků. Tímto pravidlem se zabývá norma WCAG (Web Content Accesibility Guidelines).

Zde mohu zmínit praktický web *Paletton* (www.paletton.com), v němž můžete snadno vybrat barevná schémata a navzájem se hodící odstíny. Web nabízí i ukázku simulací poruch barevného vidění či špatného zobrazení barev na nekvalitních zařízeních. Dalším významným pojmem při navrhování webu je typografie. I zde je nutné brát v potaz kontrast písma s pozadím, který zmiňuji o pár řádek výše, nebo používání různých úrovní nadpisů, velikostí písma, formátování a tak dále. To už bych ale zabíhal příliš do podrobností.

S typografií je spjatý i pojem „copywriting". To znamená použití takových slov a slovních spojení, aby byly texty dobře pochopitelné, poutavé a co nejefektivněji působily na cílovou skupinu. Proto je copywriting klíčovou součástí

marketingu. V mém projektu používám krátká a výstižná slovní spojení. V některých situacích si pomáhám i vhodně zvolenými ikonami.

3 Tvorba layoutu

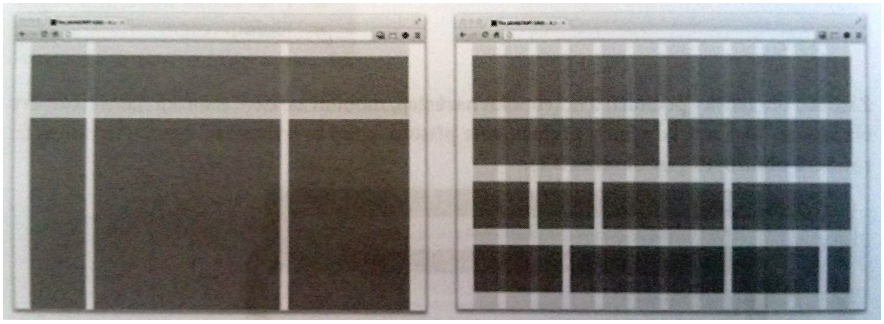
V této kapitole se už věnuji samotné tvorbě rozhraní. Celá aplikace obsahuje celkem dvě rozdílné části. První částí je úvodní web, kde se návštěvníci pomocí formuláře registrují, či přihlašují do druhé části, ve které se nacházejí nabídky pro práci s aplikací. Ty se liší podle jednotlivých rolí uživatelů. Například autor článků má na výběr jiné nabídky, než recenzent. Druhá část mimo jiné obsahuje sekci s novinkami, obsahující různá upozornění a změny od posledního přihlášení, jejíž obsah je zobrazován v modálním okně. Dále se zde také nachází, pro každého uživatele stejná, vyskakovací nabídka s možností odhlášení, či upravení svého účtu.

Již od samotného začátku jsem se zaměřoval na to, aby byly všechny prvky plně responzivní, přehledné a snadno ovladatelné i na menších displejích. Důležitým požadavkem katedry byla dostupnost celé aplikace ve dvou jazykových verzích. Vytvořil jsem proto pro úvodní web všechny html soubory dvakrát, s tím rozdílem, že text uvnitř se lišil. Změnu jazyka v backoffice tvořil kolega pomocí PHP. Ostatní prvky, jako formuláře a různá tlačítka, stále odkazovaly na stejné javascriptové funkce. Tím pádem se následná implementace s back-endovou částí aplikace příliš nekomplikovala.

3.1 Úvodní web

Jak je zmíněno již v úvodu, úvodní web nepotřeboval žádné velké změny. Hlavní věcí zde bylo přidání formuláře na přihlášení a registraci. Rozhodl jsem se také, že web částečně předělám a zlepším responzivitu. Obsahově vše zůstane tak, jako předtím.

Při vytváření layoutu jsem se nechal inspirovat moderními frameworky, které pro vytváření rozhraní využívají tzv. fluidní mřížky. Celý web je rozdělen na pomyslných šest svislých, stejně širokých sloupců. Ty jsou procentuálně nastavené, a tak se vždy přizpůsobí šířce okna. Rozvržení také obsahuje řádky, do kterých se vkládá samotný obsah. Pro lepší představu se můžete podívat na obrázek (Obrázek 3).



Obrázek 3: Fluidní mřížky

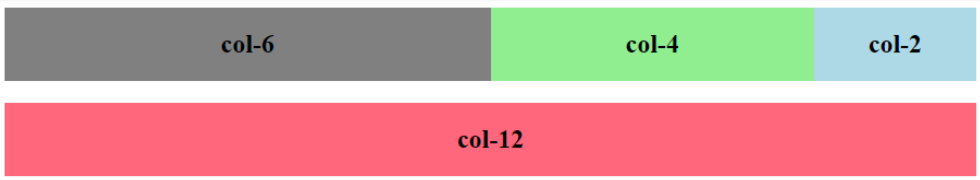
3.1.1 Fluidní mřížky

Prvním krokem při vytváření fluidních mřížek je nastavení kaskádových stylů. Na první pohled se mohou zdát poněkud složité, zejména pak v případě počítaní procent každého sloupce. Tato práce lze velmi usnadnit použitím webů, které vše vypočítají za vás. Poradí si i s přidáváním mezer mezi jednotlivé oddíly, což přidává na složitosti počítání. Já jsem použil web *Grid Generator* (www.responsivegridsystem.com/calculator), kde stačí vložit počet sloupců, procentuální mezera mezi divy a kliknout na *Calculate*. Odtud jsem si zkopíroval šířky všech druhů sloupců a nastavení řádky, kterou nazývají *section*.

Celý obsah se nachází v oddílu *container*, nastaveného na šířku 100 procent. Pomocí selektoru atributu jsou všem sloupcům nastaveny dané hodnoty.

```
[class*="col"]{
...
```

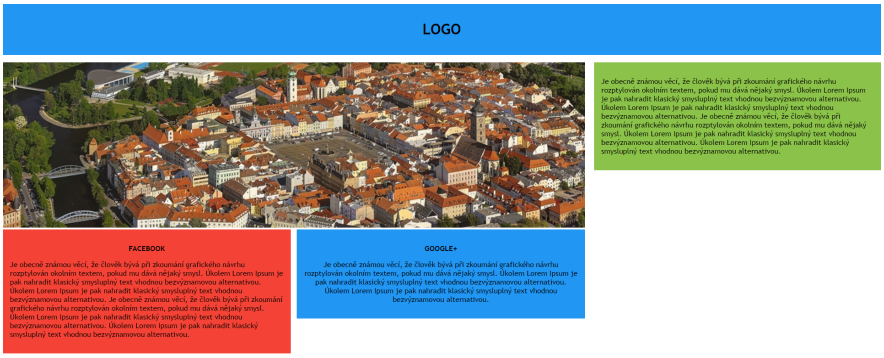
Layout se poté vytváří tak, že vždy začínáte oddílem řádky (*row*) a do něj následně umisťujete sloupce se stejnou, či rozdílnou šířkou. Celková šířka nesmí překročit počet stanovených sloupců.



Obrázek 4: Řádky ve sloupcovém layoutu

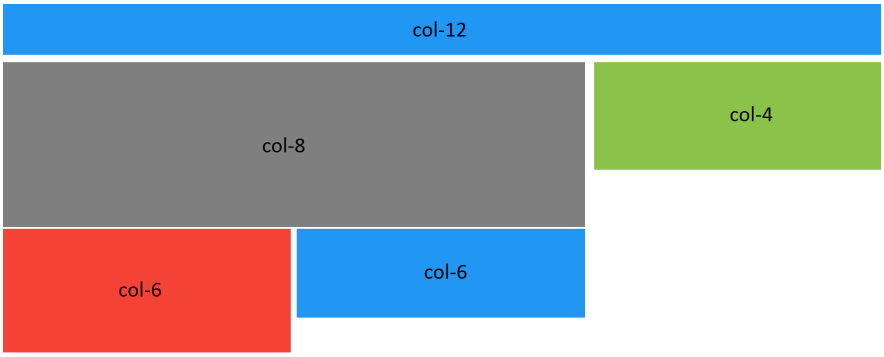
V mém případě jsem vytvořil layout o šesti sloupcích. První oddíl je široký pět a druhý jeden sloupec. Celkově to tedy dává zmíněných šest sloupců. Myslel jsem i na prohlížení na menších displejích, kde by byl druhý oddíl již příliš úzký, a tak se v určitém momentu všechny sloupce zařadí pod sebe a roztáhnou se přes celou šířku.

Myšleno je i na situaci, kdy je potřeba, aby se z jednoho oddílu stalo více menších a byly stále ve stejné řádce (Obrázek 5).



Obrázek 5: Oddíl rozdělený na více částí

To lze jednoduše vyřešit pomocí přidání dalšího řádku (*row*) dovnitř oddílu a vložením dalších sloupců. Následně je třeba myslet na to, že se šířka těchto sloupců bere jako kdyby byly na celé stránce. Když se kupříkladu v oddílu širokém dva sloupce nachází dva menší, nebude každý široký jeden, nýbrž tři sloupce (Obrázek 6).



Obrázek 6: Oddíl rozdělený na více částí s popisky

3.1.2 Zajímavé CSS vlastnosti

V průběhu tvoření jednotlivých stránek mě napadlo vyzkoušet jednu velmi zajímavou experimentální funkci, která může kodérům velice usnadnit práci. V sekci „Ke stažení" se totiž nachází seznam odkazů, kde si uživatel může prohlížet starší čísla časopisu. Tento seznam se mi nezdál příliš vábný pro oko uživatele, a tak jsem se rozmýšlel, jak ho vylepšit. Jedna z myšlenek byla rozdělit odkazy podle dílčích roků vydání a ty následně skrýt. Na stránce by zůstala pouze tlačítka s roky, která by až po kliku zobrazila související odkazy. Ostatně na stejném principu fungují i volby v backoffice části aplikace. Nicméně jsem takto neučinil i z důvodu komplikovanějšího vkládání nových odkazů, které má na starosti někdo jiný a použil

standardní seznam. Jak bylo zmíněno na začátku odstavce, vylepšení proběhlo díky experimentální funkci pro pozicování s názvem *sticky*. Jedná se o kombinaci relativního a fixního pozicování. Prvek je považován za relativně umístěný, až do chvíle překročení stanoveného bodu, na jehož místě je s ním zacházeno, jako s fixně umístěným. V syntaxi se neliší od ostatních vlastností *position*, pouze místo např. *absolute*, či *relative,* napíšeme hodnotu *sticky*. Je doporučeno i použití prefixů jednotlivých prohlížečů (*-webkit-sticky*, *-moz-sticky*, apod.). Jak bylo řečeno, nesmíme opomenout na druhou vlastnost týkající se pozice, která určuje, kdy objekt změní svoje chování. (Obrázek 7) [12]

```
position: sticky;

top: 5px;
```

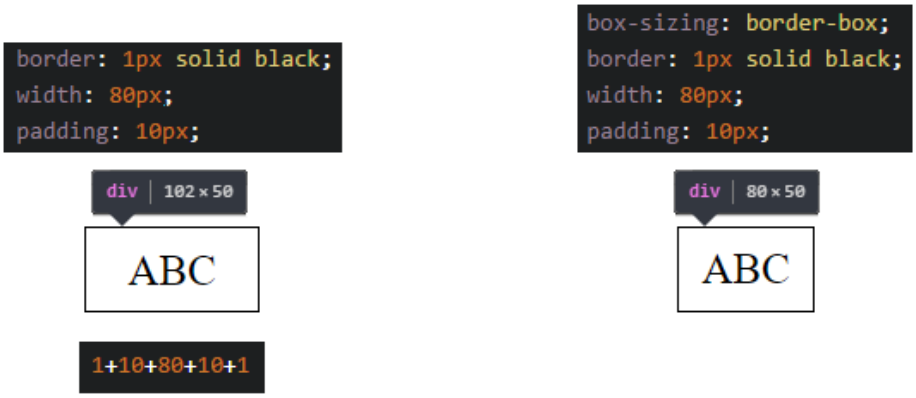
Obrázek 7: Ukázka před a po uchycení oddílu

Vlastnost *sticky* je poměrně nová záležitost a z toho důvodu nemá ještě tak velkou podporu v prohlížečích. V nejnovějších verzích Chrome, Firefox, Safari a Opera je funkce v provozu, v Edge je ve fázi schvalování, zatímco v Internet Exploreru není podpora zajištěna. To ale nevadí, protože je použita tak, aby její nepřítomnost neovlivnila funkčnost prvku. Slouží pouze jako vylepšení stávající situace. [12]

Největší výhodou této technologie je zcela jistě možnost kompletně vynechat Javascript, jehož kód, dělající totožnou činnost, by byl delší a spotřeboval více systémových prostředků. [13]

Tuto kombinaci relativního a fixního pozicování jsem aplikoval na všechny oddíly, určující rok vydání, a celý obsah umístil do okna, ve kterém lze rolovat myší. Seznam se stal více přehledným a je v něm snazší najít časopis v daném roce.

Další vlastností, kterou bych chtěl zdůraznit, je *box-sizing*. Ta určuje, zda-li se do celkových rozměrů prvku zahrnou i velikosti vnitřního okraje a rámečku. Implicitně se totiž tyto hodnoty přidají do finálních rozměrů.

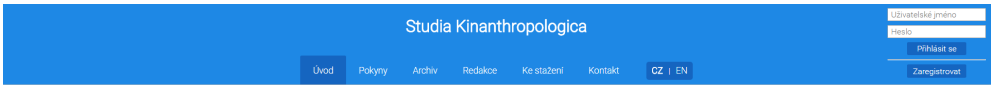


Obrázek 8: Příklad před a po použití vlastnosti box-sizing

Právě v případě tvoření fluidních mřížek mi byla tato vlastnost velmi nápomocna. Jak už jsem zmínil dříve, hodnoty šířek jednotlivých sloupců jsou přesně vypočítány. Při nepoužití hodnoty *border-box* bych musel při sebemenší změně vnitřního okraje, či rámečku všechny šířky přepočítat. Situace by se ještě zkomplikovala v momentě rozdílných jednotek velikostí. Například kdyby byla šířka prvku v procentech a vnitřní okraj v relativních *em* jednotkách. Jak poté odečíst em od procent, aby se zachovala stejná šířka? Velmi složitě. Proto je CSS vlastnost *box-sizing* vítaným pomocníkem.

3.1.3 Základní prvky

Základní prvky celé stránky tvoří hlavička (*header*), obsahující titulek, navigaci (*nav*) a formulář pro přihlášení, který se nachází vpravo nahoře. Pouze v momentě velmi malé šířky okna se přesune až pod navigaci na konec hlavičky.



Obrázek 9: Hlavička úvodního webu

Formulář je tvořen dvěma tagy *input*. Do prvního, typu *text*, uživatel vkládá svoje uživatelské jméno, které si vytvořil při registraci. Druhý je následně určen k vyplnění hesla, a proto je typu *password*. Použitím tohoto atributu je zajištěna i částečná bezpečnost. Vkládaný text se totiž automaticky mění na tečky, jak jsme zvyklí u mnohých formulářů.

Součástí oddílu je i tlačítko s názvem „Zaregistrovat", vyvolávající modální okno, sloužící k vytvoření nového uživatelského účtu. Jelikož již od začátku v celém projektu využívám výhod moderního jazyka HTML5, i formuláře těží z této skutečnosti. Lze si tak všimnout podle atributu *required*. Ten, po vložení dovnitř tagu, nabízí automatickou kontrolu, zda-li uživatel daný údaj vyplnil. Pokud tak

neučinil, zobrazí se mu upozornění a nebude možné data z formuláře odeslat. Toto je velmi užitečná funkce, která se obejde bez použití Javascriptu. Nevýhoda zde může být chybějící podpora ve starších prohlížečích bez HTML5. Podle mě ale není v projektu tak důležitou částí, aby měla velkou zpětnou kompatibilitu.

Co se týká formulářů a jejich chování, jako další zmíním atribut *placeholder*, jenž zobrazuje popisek uvnitř textového vstupu. V moderních verzích prohlížečů klasicky není problém. Snažil jsem se ale, aby byla aplikace přístupná i například v Internet Exploreru 9. Proto bylo zapotřebí několik úprav, protože prohlížeč atribut ignoruje a nic nezobrazí. Z toho se stává velký problém, poněvadž pak uživatel neví, kam jednotlivé údaje vložit. Situaci jsem vyřešil díky přidání popisného textu, vloženého uvnitř tagu *label*, ke každé položce formuláře. Defaultně se popisky nezobrazují, protože by se tam nacházely zbytečně dvakrát, ale díky stejné podmínce, jakou používám v backoffice HTML souboru, jsou zobrazeny pouze pokud se jedná o zmíněný IE 9. Menším bonusem může být, že se uživatel nemusí při vyplňování trefit přesně dovnitř tagu *input*, ale stačí i klik na popisek a výběr se taktéž provede.

Samozřejmostí navigace je, že se její položky responzivně přizpůsobí a při určité velikosti displeje se zařadí pod sebe a skryjí. Místo nich se zobrazí klasická ikona tzv. hamburgeru a po kliku na něj se menu vysune. Je několik možných způsobů, jak toho docílit. Já osobně použil variantu, při které se po kliku na ikonu menu spustí javascriptová funkce. Ta následně změní CSS třídy aplikované na navigaci a jeho jednotlivé položky. V momentě hodně malé velikosti se taktéž skryje i formulář pro přihlašování, který má opět svoji ikonu.

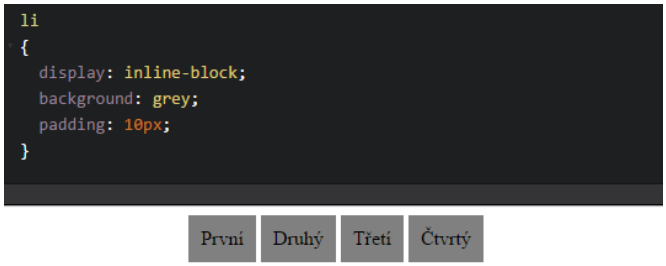
```
function menu(){
var n = document.getElementById("nav");
if (n.className === "menu"){
n.className += " responsive";
}
else{
n.className = "menu";
}
}
```

Zápis kódu je celkem jednoznačný. Jediné co bych zdůraznil je použití operátoru se třemi znamínky rovná se. Tento operátor identity, oproti klasické rovnosti (dvě rovná se), při porovnávání nepřevádí hodnoty. Aby se hodnoty rovnaly, musí být na obou stranách stejná hodnota se stejným datovým typem. Vysvětlím to na jednoduchém příkladu. [14]

```
'2' == 2 //true, protože se text '2' převede na stejný typ a poté porovná

'2' === 2 //false, zde k převodu nedochází, tím pádem se hodnoty nerovnají
```

Ještě než se přesunu k další části webu, rád bych zde zmínil ještě jednu věc, která může designerovi v některých případech vadit. Týká se to odrážkového seznamu v navigaci a hlavně jeho dílčích položek. Používáním těchto tagů při vytváření menu si můžete všimnout malinké mezery mezi jednotlivými položkami (*<li>*), i když v kódu není nic vloženo.



Obrázek 10: Mezery mezi položkami

Mohou za to elementy typu *inline-block*, které využívají vlastností jak řádkových, tak blokových elementů. Mezery mezi těmito bloky se berou jako mezery mezi slovy. Toto nastavení už je prostě tak dané, ale dobrá zpráva je, že existuje několik postupů, jak problém opravit. [15]

Jedním z takových, který při tvorbě webu používám, je velice jednoduchý. Spočívá v tom, že uvnitř HTML kódu všechny (*<li>*) tagy napíšeme tak, aby konec jednoho přímo bez mezery navazoval na začátek druhého. Upozornění, pokud jsou uvnitř další tagy, musí také přímo navazovat. Právě klasické „odentrování" za každou položkou, sloužící k zpřehlednění kódu, vkládá zmíněnou nechtěnou mezeru. Vnitřek tagu již na novém řádku začínat může. Je tedy na našem zvážení, zda-li napíšeme všechny tagy vedle sebe do jednoho dlouhého řádku, nebo jejich obsahy začínat na dalším řádku.

```
<li><a href="#">První</a></li><li><a href="#">
Druhý</a></li>
```

Pod hlavičkou se nachází hlavní část (*main*), uvnitř které je samotný obsah (*section*) a boční panel (*aside*), obsahující dvě loga a adresu redakce. I zde byl třeba řešit problém s automaticky vkládanou mezerou mezi elementy typu *inline-block*, popsaný výše. Zde bylo důležitější mezeru odstranit, než v případě navigace, kde šlo spíše o estetickou stránku věci. Při menším rozlišení zobrazovacího zařízení

se totiž boční panel přesune pod hlavní sekci, přičemž je každému logu nastavena šířka 50 % tak, aby se nacházela vedle sebe. V momentě, kdy bych s již zmíněnou nechtěnou mezerou nepočítal, by se měla loga tendenci zařazovat pod sebe a ne vedle sebe.

Zcela dole se tradičně zobrazuje patička (*footer*). Jak si můžete všimnout, jednotlivé části mají přiřazeny své sémantické elementy tak, jak je definuje standard HTML5.

3.1.4 SEO optimalizace

Jak jsem zmiňoval již na začátku práce, nesmíme zapomínat ani na SEO optimalizaci pro vyhledavače. Proč je důležitá? Už jen z toho důvodu, že některé z kroků optimalizace se prolínají s validitou HTML kódu. Příkladem může být správné použití a členění nadpisů (*<h1>*, *<h2>*, ...).

Základním krokem je také vhodné vyplnění meta tagů s atributem *name* v hlavičce dokumentu, popisující, co se na stránce nachází, klíčová slova s ní spjatá, autory apod.

```
<meta name="description" content="...">
<meta name="keywords" content="...">
<meta name="author" content="...">
<meta name="robots" content="index, follow">
<meta name="googlebot" content="index, follow, snippet, archive">
```

Poslední dva řádky v příkladu nastavují chování tzv. robotů, kteří procházejí internet a tímto je jim řečeno, že mohou stránku navštívit a zobrazovat ji poté ve vyhledávání. Nastavení lze provádět i pomocí textového souboru *robots.txt*, umístěného v kořenovém adresáři webu. Dalším ze souborů v adresáři je také mapa stránky (*sitemap*), která pomáhá robotům v orientaci po webu.

Součástí SEO optimalizace je nepochybně používání zpětných odkazů, vedoucích na nějaké vnější stránky. V mé práci se na úvodním webu nacházejí dvě loga univerzity, kdy se po kliku dostanete buď na stránku Jihočeské univerzity, nebo Pedagogické fakulty.

Výše zmíněné odkazy zvyšují *PageRank* webu, který vyjadřuje něco jako věrohodnost nebo důležitost stránky. Google, který toto hodnocení navrhl, si *PageRank* počítá (zjednodušeně řečeno) podle toho, kolik a jak důležitých stránek na tu počítanou stránku odkazuje. [20]

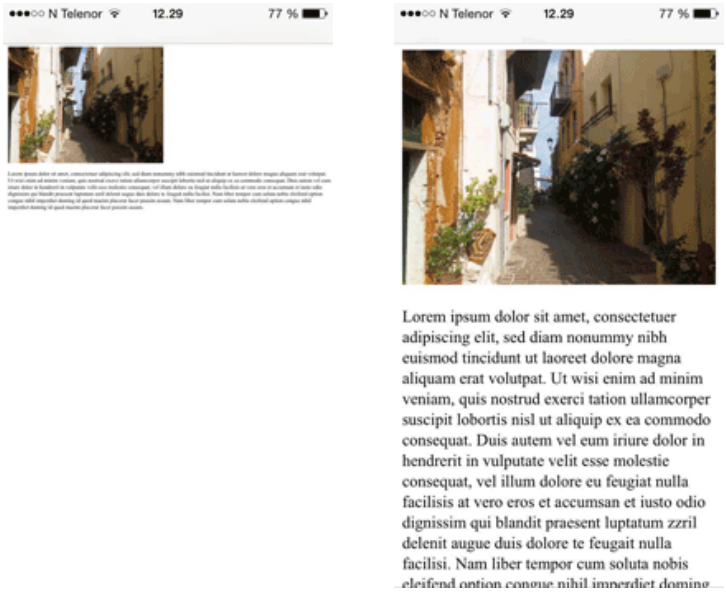
K dispozici je mnoho publikací a článků popisující tuto problematiku. Existují i společnosti, které se tímto přímo zabývají. Z toho je vidět, jak důležitá může

SEO optimalizace být, zvláště pak pro velké firmy, které vyžadují co největší počet oslovených zákazníků a návštěvnost jejich webu. Mnoho lidí při tvorbě webu SEO podceňuje, a proto bych chtěl rád tímto krátkým shrnutím poukázat na její potenciál.

3.2 Backoffice

V rámci mého projektu zmíním pouze některé části celkového kódu, které jsou nějakým způsobem nezbytné, nebo podstatné. Cílem práce není učit základy HTML a CSS, ale představit styl mé práce, na co si dávat pozor, ukázat s jakými problémy jsem se potýkal a jak je případně vyřešil.

Prvním důležitým krokem bylo použití meta tagu *viewport*. Ten zajišťuje správné zobrazení i na menších obrazovkách. Na obrázku (Obrázek 11) můžete vidět rozdíl mezi jeho použitím, či vynecháním.



Obrázek 11: Rozdíl před a po použití viewportu

3.2.1 Připojení stylů

U HTML kódu ještě zůstanu a podotknu, že existuje několik způsobů, jak k dokumentu připojit kaskádové styly. Prvním způsobem může být připojení několika souborů CSS pomocí tagu *link*, kde u každého souboru definujeme, při jaké velikosti displeje se má použít (viz. kód níže).

```
<link rel="stylesheet" media="device-width: 800px"
                    href="styly.css">
```

Osobně jsem použil druhý způsob, ve kterém vložím také několik souborů, ale šířku displeje zohledňuji až v CSS kódu.

V hlavním souboru (*styly.css*) se nacházejí všechna nastavení a šířky, při kterých se má nastylování pozměnit. Pro lepší rozložení na menších obrazovkách jsem zvolil 768 pixelů a níže.

```
@media all and (max-width:768px)
{
```

Druhý soubor (*stylyIE.css*) se díky podmínce použije pouze v případě, že uživatel používá prohlížeč Internet Explorer ve verzi 9 a nižší. Právě v těchto verzích IE vznikaly problémy se správným zobrazením některých prvků, které se použitím vlastních stylů opravily. Jiné, než uvedené prohlížeče, tento kód ignorují.

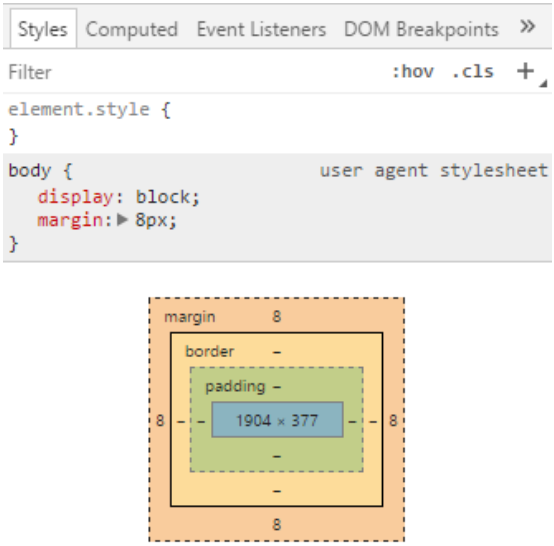
```
<!--[if lt IE 10]>
<link rel="stylesheet" href="css/stylyIE.css">
<![endif]-->
```

Je dobré zmínit, že nebylo zapotřebí psát většinu kódu znova, protože druhý CSS soubor nenahrazuje ten první, ale pouze doplňuje části, které v prvním souboru nebyly, či nahrazuje původní hodnoty novými.

3.2.2 Resetování

Ještě než jsem se pustil do vkládání jednotlivých prvků, bylo nutné vynulovat některé hodnoty stránky. Tento postup se využívá z důvodu různého chování prohlížečů. Každý z nich může mít totiž nastavené rozdílné výchozí styly, například pro vnější okraj (*margin*).

```
body { margin: 0; padding: 0; top: 0; }
```



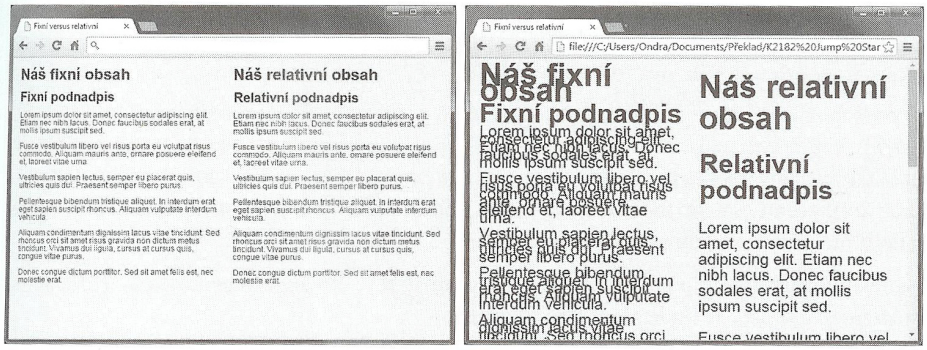
Obrázek 12: Defaultně nastavené hodnoty prohlížeče Chrome

3.2.3 Relativní jednotky

U prohlížečů ještě zůstanu, a to kvůli jednotkám velikosti jako jsou pixely, *em* a *rem*. Každý moderní prohlížeč má výchozí velikost písma 16 pixelů. Při vývoji se ale nepoužívají tyto fixní jednotky, nýbrž relativní (*em*, *rem*). Důvodem je, že když si bude chtít například uživatel se zrakovým postižením zvětšit písmo, relativní jednotky se automaticky přepočítají. Tyto hodnoty se dají aplikovat nejen na velikost písma, ale třeba i na vnější a vnitřní okraje. [2]

Rozdíly mezi *em* a *rem* nejsou tak zásadní. Hodnota jednotky *rem* je určena velikostí písma elementu *html*, zatímco jednotka *em* je odvozena od velikosti písma rodičovského elementu, ve kterém je použita. [4]

V mém projektu jsem použil výhradně relativní jednotky *em*, které se vždy vhodně přepočítají, a tím si vzhled stránky stále zachová stejné rozvržení. V ostatních případech jsou velikosti nastaveny pomocí pixelů, či procent tam, kde se hodnoty vztahují například k zobrazovacímu zařízení, nebo porovnávání dvou prvků.

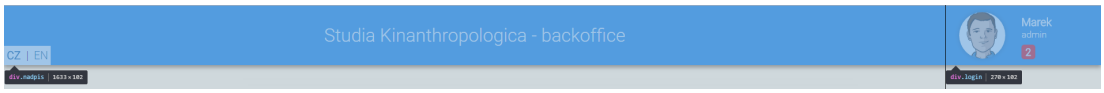


Obrázek 13: Fixní versus relativní jednotky (vpravo dvojnásobná velikost písma)

3.2.4 Rozložení prvků

Vše na stránce se nachází v oddílu *container*, který dále obsahuje tři dílčí části. Zcela nahoře je hlavička s nadpisem a *loginem*, na opačném konci pak patička a mezi těmito dvěma prvky je hlavní obsah. Každá tato část je sémanticky rozdělena prostřednictvím HTML5 tagů *header*, *main* a *footer*.

V hlavičce bylo potřeba vyřešit dva vedle sebe se nacházející prvky. Samo o sobě to nebylo až tak složité, stačilo použít vlastnost obtékání (*float*). Problém nastal v momentě, když měl mít jeden prvek pevnou šířku a druhý by automaticky doplnil zbývající prostor. Pokud bych nastavil procentuální hodnoty, abych zachoval poměr, kde je nadpis širší než *login*, zmenšující se šířka *loginu* by při určité šířce okna způsobila přetékání obsahu. Fixní velikost zajistila, že je *login* po celou dobu stále stejně široký.



Obrázek 14: Hlavička backoffice webu

Jak ale automaticky dopočítat zbylou šířku volného prostoru pro nadpis? Docílil jsem toho díky CSS3 funkci *calc()*. Ta je podporována většinou prohlížečů a nabízí řešení právě pro podobné případy. Jelikož je tedy *loginu* nastavena pevná šířka, stačilo ve stylech napsat, že šířka prvku *nadpis* bude sto procent šířky okna mínus šířka *loginu*. Starší verze Internet Exploreru tuto funkci neznají, a tak jsem pro ně nastavil, aby byl vzhled stejný jako na mobilním zařízení. To znamená že *login* i *nadpis* budou sto procent široké a každý na vlastním řádku.

```
width: calc(100% - 270px);
```

Při zjišťování, jak moc je daná funkce podporována, byl využíván web *Can I use* (www.caniuse.com), kde stačí zadat hledaný název a stránka vám zobrazí funkčnost v různých prohlížečích, známé problémy a jiné.

Další zajímavou situací k řešení bylo zajištění, aby byly oba prvky stejně vysoké i v situaci, když měl každý z nich jinak velký obsah. Nižší prvek by se vždy přizpůsobil vyššímu. I zde jsem si pomohl šikovnou funkcí, a to s názvem *flex*. Sama o sobě toho umí mnohem víc, ale mně stačilo pouze ono srovnání dvou prvků na stejnou výšku. Základem je, aby byly oba oddíly zaobaleny v dalším rodičovském oddílu, na který se poté aplikuje již zmíněná funkce.

```
header{
display: flex;
display: -webkit-flex;
```

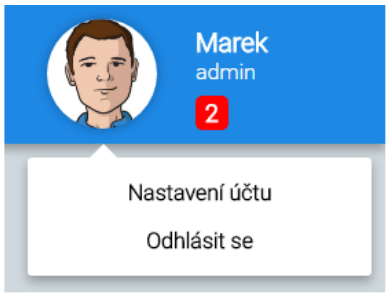
Zde bych rád zmínil jednu důležitou poznámku. Pokud použijete vlastnost *display* s hodnotou *flex*, je nutné, aby byly všechny vnější a vnitřní okraje (*margin*, *padding*) nastavené v jiných, než procentuálních hodnotách. Pokud tak neučiníte, může se v některých prohlížečích (Firefox, Edge) stát, že budou tyto údaje ignorovat. Samotná specifikace flexbox modulu doporučuje vyhnout se aplikování procentuálních hodnot *marginu* a *paddingu* na flex položky, protože mají odlišné chování v různých prohlížečích. Velikosti okrajů jsou totiž počítány na základě rozměrů flex prvku, na který jsou použity. [16, 17]

Nyní bych chtěl popsat dva grafické prvky, které nejsou až tak komplikované, ale spíše zajímavé a někdo může vysvětlení postupu ocenit. Prvním z nich je kruhový avatar umístěný vedle jména uživatele. Dokonalý tvar kruhu se s pomocí

CSS3 vlastnosti *border-radius* vytváří velice jednoduše. Stačí této vlastnosti nastavit takovou hodnotu jakou má prvek poloviční výšku, či šířku. Na tom která z nich nezáleží, protože aby byl kruh dokonalý, musí se hodnoty výšky a šířky rovnat. Alternativou může být hodnota *50 %*, ale taktéž zde platí dodržení rovnosti velikostních hodnot. Obrázek avatara se poté vloží tagem *(<img>)* a nastaví se mu hodnota šířky na *100 %*. Také je nutné nastavit přetékání oddílu na skryté (*overflow: hidden*), k zamezení přesahování hran obrázku ven.

```
.avatar{
display: inline-block;
width: 80px;
height: 80px;
border-radius: 40px;
background-image: ...
```

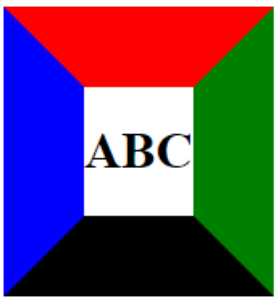
Druhým zajímavým prvkem je malá šipka u menu poukazující na to, že se volby vztahují k uživateli. Její vytvoření bylo poněkud komplikované, ale je zde krasně vidět, co vše se dá vytvořit pouze s pomocí stylů, bez použití obrázků, Javascriptu a podobně.



Obrázek 15: Volby uživatele

K celému divu jsem přiřadil selektor *::after* a tomu nastavil různé vlastnosti. *Left* a *bottom* nastavují pozici, v mém případě nahoře a vlevo, přímo pod avatarem. Tloušťka rámečku (*border-width*) ovlivňuje velikost celé šipky. Nejdůležitější vlastností je ale barva pozadí (*border-color*). Ta je zadána pomocí čtyř hodnot, z níž jsou tři průhledné (*transparent*) a zbylá určuje barvu šipky. Roli hraje i pořadí těchto hodnot. Při odlišné kombinaci by byla šipka otočena jiným směrem. Využívá se zde způsobu, jakým prohlížeč vykresluje rámečky objektů. Jednotlivé čáry na každé straně nejsou totiž obdélníky, nýbrž vzájemně se nepřekrývající rovnoramenné lichoběžníky (Obrázek 16). Pokud tedy prvek neobsahuje žádný obsah, pak se k sobě hrany natolik přiblíží, že každý z nich vytvoří trojúhelník

(Obrázek 17). Poté už stačí vybrat, kterým směrem má šipka směřovat a zbylé hrany zprůhledníme.



Obrázek 16: Ohraničení prvku

Obrázek 17: Ohraničení prázdného prvku

```
.loginMenu::after{
    content: "";
position: absolute;
    bottom: 100%;
     left: 18%;
border-width: 10px;
border-style: solid;
border-color: transparent transparent #FFFFFF transparent;
```

Vedle avatara si můžete všimnout malého čísla, indikující počet novinek, které se udály od posledního uživatelova přihlášení. Když žádné oznámení nepřibylo, je prvek šedivý. Pokud se ale něco událo, zbarví se prvek do červena, za účelem upoutání pozornosti uživatele. Vše se provádí za pomoci jednoduché javascriptové podmínky kontrolující číslo uvnitř prvku. Na základě toho pak pozmění příslušné nastylování.

Defaultně skryté modální okno s informacemi se pomocí Javascriptu zobrazí až po kliku uživatele na počet novinek. Toto okno je přesně horizontálně zarovnané na střed stránky. Docílil jsem toho prostřednictvím absolutního pozicování.

Důležité jsou zde vlastnosti *left*, *right*, *top* a *margin*. Vynecháním byť jednoho z nich, nebude zarovnání správně fungovat.

Odsazení zleva a zprava byla nastavena na hodnotu nula a vnější okraj je počítán automaticky. Aby prvek zůstal nahoře a neodsadil se ve stránce až úplně dolů, nastavil jsem také vlastnosti *top* nějakou hodnotu. Konkrétně pak vypočítanou díky JS funkci, která zjišťuje, o kolik pixelů je celá stránka posunuta (*scrollTop*). Tím se okno zobrazí vždy tam, kde se zrovna uživatel nachází. Nesmí se zde zapomenout i na hodnotu šířky celého prvku, která musí být nějak definovaná, jinak se prvek roztáhne přes celou šířku stránky. V mém případě je nastavená procentuálně tak, aby se přizpůsobila velikosti okna.

Existuje i jiný způsob dosažení stejného cíle a to čistě pomocí javascriptového kódu. Je ale dobré se snažit co nejvíce vyhnout používání JS a pokusit se využívat potenciálu kaskádových stylů, už jen z toho důvodu, že skriptování způsobuje pomalejší načítaní stránky a více zatěžuje systémové zdroje.

```
.modal{
position: absolute;
left: 0;
right: 0;
margin: auto;
width: 60%;
```

Nyní už se dostávám k hlavním ovládacím prvkům aplikace. Každému přihlášenému uživateli se tyto volby mění v závislosti na jeho právech. Menu se skládá s několika pod sebou seřazených tlačítek dost velkých na to, aby byly přehledné, jasně viditelné a snadno se ovládaly i na mobilních zařízeních. Samotný obsah každé volby je skryt do té doby, než ho uživatel bude potřebovat. Obsahy se totiž při načtení celé stránky nestáhnou, ale čekají na povel od uživatele a až poté jsou staženy asynchronně pomocí AJAXu. Tímto způsobem se i šetří data, což hlavně ocení lidé s mobilním či nějak slabším připojení k internetu.

Každý ze zmíněných obsahů má vytvořený svůj unikátní HTML soubor, takže se v případě potřeby jednoduše upraví. Všechny z nich sdílejí i stejné kaskádové styly, které jsou používány i pro hlavní stránku. Pokud má obsah další možnosti volby, jako jsou například různá tlačítka pro úpravy, zobrazí se taktéž asynchronně po vyžádání, ve vlastním modálním okně. Vše je řešeno použitím tříd a identifikátorů, umístěných uvnitř tagů daných tlačítek.

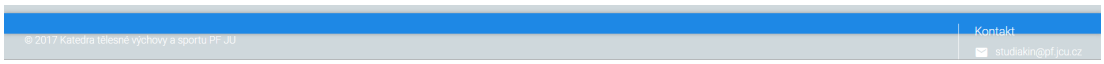
V javascriptovém souboru pracují dvě v základu stejné funkce pro načítání zmíněných obsahů. První se stará o vkládání do připravených oddílů a je volána

při kliku na jednu z voleb, které jsou v nabídce a je označena třídou *obsah*. Na základě identifikátoru se pak načte HTML soubor, který má totožný název. Aby funkce věděla kam vkládat, je k tomu určený oddíl označen stejným ID s přidaným písmenem „O" na konci. Druhá AJAX funkce pak pracuje na podobném základu s tím rozdílem, že zde bylo zapotřebí ji volat až když je nutné a má potřebné soubory, odkazující na ni, již vložené. Slouží totiž k vytváření modálních oken a je volána klikem na tlačítka, která jsou vložena díky první zmíněné funkci. Pokud se totiž pro zachycení uživatelského kliknutí myší na prvek použije klasický zápis (viz. první řádek kódu), musí být všechny prvky, u kterých chceme událost sledovat, existovat již na začátku při načtení stránky. Používá se tu tzv. přímá vazba, která ihned po načtení webu připojí *handler* na již existující prvky. U první funkce tomu tak je, nicméně u druhé sledujeme události u prvků vytvořených až později, když je první funkce sama vloží. Z toho důvodu je druhá zapsaná odlišně (viz. druhý řádek kódu) a prvek hledá až v potřebný moment, kdy je již vložen a tím pádem je nalezen.

```
$('.polozka').click(function()

$(document).on('click', '.novyModal', function()
```

Posledním prvkem, nacházejícím se na stránce zcela dole, je klasicky patička. Zde je uveden copyright a v případě potřeby také kontakt. Chtěl jsem, aby tyto dvě položky byly vedle sebe a každá úplně na kraji. To nebylo těžké zařídit. Přímo pro tyto potřeby existuje v CSS vlastnost obtékání (*float*), použitá již dříve. Když ale byla tato vlastnost aplikována na oba prvky, nastal problém s pozadím celého rodičovského oddílu. Ten netoleroval výšku těchto dvou divů a vykreslil pouze tenký pruh (Obrázek 18).



Obrázek 18: Špatné vykreslení výšky patičky

Proto zde bylo zapotřebí do patičky přidat další oddíl, kterému stačila nastavit pouze jediná vlastnost, a to *clear* s hodnotou *both*. Tato vlastnost určuje čekání na ukončení jiných obtékaných prvků. V té chvíli se patička roztáhne až na konec a pozadí tak překryje celou plochu, přesně jak je potřeba.

Další věcí, kterou jsem se zabýval, byly ikony. Již dříve v textu se zmiňuji, že byla použita sada ikon vytvořených po vzoru Material designu. Jejich implementace je velmi jednoduchá. Stačilo v dokumentu podle návodu připojit link, odkazující na daný font a poté kdekoliv, kde je potřeba, vložit tag (<*i*>) se speciální třídou. Text obsažený v tomto tagu pak definuje, jaká ikona se vykreslí. Ta

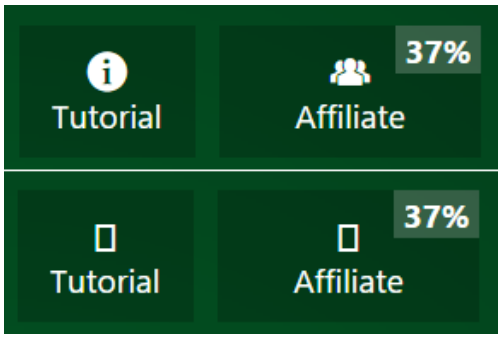
se poté bere jako kdyby to bylo písmeno a platí tak na ni nastavení aplikovatelná na běžný text. Například pouhá změna barvy je v tomto případě velmi jednoduchá, pomocí CSS stylů. Další výhodou metody je snadná použitelnost. Designér nemusí stahovat a ukládat každou ikonu zvlášť, stačí mu pouze vybrat si z on-line sady a připsat tag do HTML souboru.

Nicméně při vkládání těchto ikon jsem narazil na menší komplikaci. Při otevření webu v prohlížeči Internet Explorer verze 9 a starší se ikony špatně, nebo vůbec nezobrazily. Naštěstí stačila pouze menší úprava kódu, kterou doporučuje i samotný web. Uvnitř tagu (*<i>*) se místo názvu (např. *close*, pro ikonu křížku), vložil unikódový šestnáctkový zápis. Tím byla zajištěna zpětná kompatibilita.

```
STEP 2: Use Icon in Your Site
<!-- For modern browsers. -->
<i class="material-icons">close</i>
<!-- For IE9 or below. -->
<i class="material-icons">&#xE5CD;</i>
```

Obrázek 19: Metody vložení ikony

Proč tedy existují dvě metody zápisu kódu? Důvod je prostý. Pokud navrhujeme aplikaci pouze pro moderní prohlížeče, podporující tzv. ligatury, je dobré použít variantu s názvy, které pomáhají například lidem se zrakovým postižením, využívající čtečky. V případě, že má prohlížeč zakázáno stahování fontů, zobrazí se uživateli alespoň název. Při unikódovém zápisu je ikona nahrazena čtverečkem naznačujícím, že tento znak nebyl rozpoznán. [18]



Obrázek 20: Vykreslení ikon

Nicméně kromě ikonového fontu se používají i jiné metody. Jednou z takových je i vkládaní ikon pomocí SVG grafiky. Tato metoda má větší podporu v prohlížečích. Vkládání je klasické, jako normální obrázek s tím rozdílem, že oproti němu se jedná o vektorovou, nikoliv rastrovou grafiku. To umožní její libovolné zvětšování bez ztráty na kvalitě. SVG technologie byla pro podobné účely vytvořena, takže to není tak neobvyklé řešení jako ikonové fonty. Není třeba se

zde ani obávat nástroje pro blokování nepotřebného obsahu (reklamy, měřicí kódy, sociální pluginy apod.), které mohou blokovat i webová písma. [19]

Rozdílných názorů na toto téma je velké množství. Osobně si myslím, že není ze všech možností vkládání ikon pouze jediná ta správná a nejlepší. Záleží na mnoha ohledech, například jakou zpětnou kompatibilitu požadujete, jestli řešíte přístupnost, hlasové čtečky, rychlost načítání stránky a tak dále.

3.3 Implementace

Jak bylo již zmíněno, spolupracoval jsem na tomto projektu s kolegou. Ten vytvářel back-end, nebo-li tu část, která je v pozadí a normální uživatel ji nevidí, ani neovládá. Probíhají zde různé přenosy dat z formulářů, databáze a podobně. Z toho důvodu bylo třeba udržovat naši spolupráci velmi těsnou. V pravidelných setkáních se diskutovalo nad funkcionalitami systému a jeho podobě. Snažili jsme se o co největší optimalizaci a případně co nejjednodušší budoucí rozšiřitelnost o nové funkce. U každého důležitého prvku stránek má jeho třída, či identifikátor zvolen jednoznačný název, pro lepší přehlednost při další manipulaci. Byla vytvořena i testovací doména na serveru poskytující hosting zdarma, kam jsme postupně nahrávali verze svých souborů. Vše probíhalo bez větších problémů. Vždy jsme se pokoušeli najít co nejlepší řešení pro uživatele, aby byla aplikace intuitivní a přehledná.

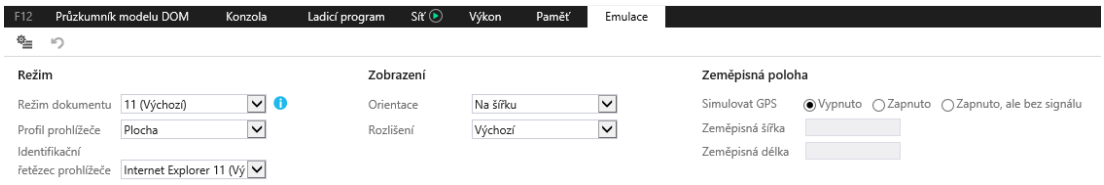
3.4 Testování

U každého projektu nesmí chybět jak průběžné, tak i finální testování celé aplikace. Moje práce nebyla výjimkou. Jelikož jsem se zabýval grafickým vzhledem, zaměřil jsem se především na ovladatelnost aplikace na různých typech zařízení. Snažil jsem se také o zpětnou kompatibilitu se staršími verzemi prohlížečů. Jako příklad mohu jmenovat Internet Explorer ve verzi 9, který potřeboval zvláštní pozornost. Některé prvky se v něm totiž nezobrazovaly správně, a tak bylo zapotřebí použít speciální stylování. Samozřejmostí byla i kontrola samotné validity kódu. Například zda jsou všechny párové tagy ukončené, obrázky mají svůj alternativní popisek (*alt*), web obsahuje dobře strukturované nadpisy, meta tagy v hlavičce a podobně.

3.4.1 Nástroje

Prvním z několika užitečných nástrojů, které jsem používal, byla webová stránka *Can* I *use* (www.caniuse.com). Zmiňoval jsem se o ní již v kapitole, kde se řeší tvorba backoffice. Díky ní lze zjistit, v kterých prohlížečích je daná funkce, nebo

vlastnost podporována, ještě než je použita v projektu. Dalším velmi používaným nástrojem byla funkce, dostupná v některých prohlížečích, nazývaná „prozkoumat", nebo také „zkontrolovat prvek". Ta vyvolává různé nabídky pro kontrolu kódu a je vybavena schopností ho v reálném čase upravovat a ihned zobrazit změnu. Nejlepší je ale možnost testování responzivního designu. Umožňuje libovolnou změnu velikosti okna a také zobrazuje různé druhy zařízení s jejich velikostmi obrazovek. Mohl jsem si tak ihned vyzkoušet, jak bude stránka vypadat. Internet Explorer jde ještě dál a v kartě „Emulace" nabízí přepnutí vykreslování tak, aby se prohlížeč choval jako jeho starší verze.



Obrázek 21: Nástroj zkontrolovat prvek v IE

Při testování aplikace se také snažím, aby se ve finále načetla pokud možno co nejrychleji a s tím souvisí i tzv. minifikace souborů, obzvláště pak těch kaskádových. Každá mezera, odřádkování apod. přidává souboru další bajty navíc a to způsobuje delší načítání stránky. Prohlížeč totiž nejdříve stahuje tyto typy souborů a až poté vykreslí samotný obsah.

K testu rychlosti načítání stránky, potažmo její optimalizace z hlediska výkonu, může posloužit web *PageSpeed Insights* (https://developers.google.com/speed/pagespeed/insights). Nabízí analýzu, která může pomoci určit, jaké prvky zpomalují načítání a také tipy pro jejich vylepšení. Tato stránka také nabízí odkaz na další, celkem jednoduchý, nástroj společnosti Google, který testuje použitelnost v mobilních zařízeních. Velmi podobný web má název *Resizer* (http://material.io/resizer/), opět testující responzivitu. Webové stránky si zde můžete libovolně prohlížet v jednotlivých, různě velkých oknech, simulujících odlišná zařízení. Na závěr ještě zmíním trochu pokročilejší testovací web *GTmetrix*, který zobrazuje detaily, jako přesný čas načtení celé stránky, její velikost, celkovou známku a podobně. Může při tom použít servery z různých regionů. Takovéto a jiné funkce jsou ale za měsíční poplatek, nicméně i tak má tato aplikace v základním neplaceném režimu co nabídnout.

Stabilitu a použitelnost celé aplikace prověří i následné používání samotnými uživateli. Díky jejich zpětné vazbě se může web případně upravit a zlepšit tak jeho uživatelskou přívětivost.

4 Závěr

V bakalářské práci jsem se zabýval tvorbou klientského rozhraní pro webovou aplikaci, sloužící ke správě a publikaci vědeckého časopisu Studia Kinanthropologica. Taktéž jsem vytvořil nový vzhled webu tohoto časopisu, který je vydáván katedrou tělesné výchovy a sportu.

Shrnuji při tom vše podstatné, popisuji problémy, na které jsem v průběhu práce narazil a vysvětluji jejich řešení. Tento text tak může posloužit jako základní příručka pro tvorbu webu. Nejdůležitější ale je, že bude aplikace fungovat jako nástroj pro rychlejší a snadnější správu časopisu a vyhoví tak požadavkům katedry.

Díky této práci jsem se naučil mnoho nového a získal tak cenné zkušenosti, které se mi budou jistě v budoucnu hodit.

Literatura a zdroje

[1] STANÍČEK, Petr. Dobrý designér to všechno ví!. 1. vyd. Kamenné Žehrovice: vydáno vlastním nákladem autora, 2016. ISBN 978-80-260-9427-2.

[2] SHARKIE, Craig a Andrew FISHER. Responzivní webdesign: okamžitě. 1. vyd. Brno: Computer Press, 2015. ISBN 978-80-251-4384-1

[3] Systém pro správu obsahu. Wikipedia [online]. [cit. 2016-08-09]. Dostupné z: https://cs.wikipedia.org/wiki/Syst%C3%A9m_pro_spr%C3%A1vu_obsahu

[4] Em vs Rem. Envatotuts [online]. [cit. 2016-08-10]. Dostupné z: http://webdesign.tutsplus.com/tutorials/comprehensive-guide-when-to-use-em-vs-rem--cms-23984

[5] Wireframe. Wikipedia [online]. [cit. 2017-03-13]. Dostupné z: https://cs.wikipedia.org/wiki/Wireframe

[6] Call to action (marketing). Wikipedia [online]. [cit. 2017-03-13]. Dostupné z: https://en.wikipedia.org/wiki/Call_to_action_(marketing)

[7] Material Design. Wikipedia [online]. [cit. 2017-03-13]. Dostupné z: https://en.wikipedia.org/wiki/Material_Design

[8] Material icons [online]. [cit. 2017-03-13]. Dostupné z: https://material.io/icons

[9] Introduction - Material design [online]. [cit. 2017-03-13]. Dostupné z: https://material.io/guidelines/material-design/introduction.html

[10] Elevation and shadows [online]. [cit. 2017-03-13]. Dostupné z: https://material.io/guidelines/material-design/elevation-shadows.html

[11] Material motion [online]. [cit. 2017-03-18]. Dostupné z: https://material.io/guidelines/motion/material-motion.html

[12] Sticky positioning [online]. [cit. 2017-03-18]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/CSS/position#Sticky_positioning

[13] Stick your landings! position: sticky lands in WebKit [online]. [cit. 2017-03-18]. Dostupné z: https://developers.google.com/web/updates/2012/08/Stick-your-landings-position-sticky-lands-in-WebKit

[14] JavaScript Operators [online]. [cit. 2017-03-18]. Dostupné z: https://www.w3schools.com/js/js_operators.asp

[15] Fighting the Space Between Inline Block Elements [online]. [cit. 2017-03-18]. Dostupné z: https://css-tricks.com/fighting-the-space-between-inline-block-elements/

[16] Flex Item Margins and Paddings [online]. [cit. 2017-03-19]. Dostupné z: https://drafts.csswg.org/css-flexbox/#item-margins

[17] CSS Flexible Box Layout Module Level 1 [online]. [cit. 2017-03-19]. Dostupné z: https://www.w3.org/TR/2015/WD-css-flexbox-1-20150514/#item-margins

[18] Why You Should Consider A Ligature Icon Font For Your Next Project [online]. [cit. 2017-03-20]. Dostupné z: http://thenewcode.com/620/Why-You-Should-Consider-A-Ligature-Icon-Font-For-Your-Next-Project

[19] Font ikony [online]. [cit. 2017-03-20]. Dostupné z: http://jecas.cz/font-ikony

[20] Google PageRank [online]. [cit. 2017-03-25]. Dostupné z: https://www.jakpsatweb.cz/seo/pagerank.html

[21] Img_viewport1.png [online]. [cit. 2017-04-02]. Dostupné z: http://www.w3schools.com/css/img_viewport1.png

[22] Img_viewport2.png [online]. [cit. 2017-04-02]. Dostupné z: http://www.w3schools.com/css/img_viewport2.png

[23] Ikonky.png [online]. [cit. 2017-04-02]. Dostupné z: http://jecas.cz/files/font-ikony/ikonky.png

Seznam obrázků

Přílohy

1. CD se zdrojovými kódy webových stránek a plné znění bakalářské práce v PDF

2. Weby: www.pf.jcu.cz/stru/katedry/tv/studia_kinantropologica/pages/index.html, www.pf.jcu.cz/stru/katedry/tv/studia_kinantropologica/system/backoffice