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

Bakalářská práce

# Mobilní dotyková verze webu informatické soutěže v HTML5

Vypracoval: Filip Čertík
Vedoucí práce: PaedDr. Petr Pexa, Ph.D.

České Budějovice 2015

## 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 - v úpravě vzniklé vypuštěním vyznačených částí archivovaných pedagogickou fakultou 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 1. dubna 2015

……
podpis

## Abstrakt

Cílem bakalářské práce je vypracovat mobilní dotykovou verzi veřejné části webu soutěže Bobřík informatiky. Student zjistí možnosti vytvoření mobilní verze webu, která by čerpala obsah z původních stránek soutěže. Navrhne vzhledové a funkční varianty mobilní verze webu včetně řešení touch funkcí.

Student vytvoří strukturu mobilní verze webu www.ibobr.cz., prozkoumá způsob čerpání dat pro články na mobilním webu a způsob komunikace s prostředím Joomla, případně jiný způsob aktualizace obsahu mobilní verze webu.

Součástí webu bude funkční cvičný test, vytvořený z dodaných úloh, který se bude skládat z několika variant úloh (test nebude načítat data z databáze, ale bude kontrolovat správná řešení).

## Klíčová slova

HTML5, mobilní dotyková verze, boostrap, responzivní design, bootstrap, telefony, tablety, displej, webdesign, js, css, html, media queries, joomla

## Abstract

Aim of this work is to develop a mobile version of touch the public site competition Informatics Beaver. Student finds the possibility of creating a mobile version of your site that could draw content from the original site of the competition. Suggests the visual and functional variants mobile version of site solutions including touch function.

Student creates a structure of a mobile version of the site www.ibobr.cz., To explore ways of drawing data for articles on the web and mobile way to communicate with the environment Joomla, or another way to update the contents of a mobile version of the site.

The website will be operational practice test, created from the supplied tasks, which will consist of several variants of tasks (test will not retrieve data from the database, but will check the correct solution).

## Keywords

HTML5, mobile touch version, bootstrap, responsive design, bootstrap, phones, tablets, display, webdesign, js, css, html, media queries, joomla

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH
**Fakulta pedagogická**
Akademický rok: **2012/2013**

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

| | |
|---|---|
| Jméno a příjmení: | **Filip ČERTÍK** |
| Osobní číslo: | **P11036** |
| Studijní program: | **B7507 Specializace v pedagogice** |
| Studijní obor: | **Informační technologie a e-learning** |
| Název tématu: | **Mobilní dotyková verze webu informatické soutěže v HTML5** |
| Zadávající katedra: | **Katedra informatiky** |

Zásady pro vypracování:

Cílem bakalářské práce je vypracovat mobilní dotykovou verzi veřejné části webu soutěže Bobřík informatiky. Student zjistí možnosti vytvoření mobilní verze webu, která by čerpala obsah z původních stránek soutěže. Navrhne vzhledové a funkční varianty mobilní verze webu včetně řešení touch funkcí.
Student vytvoří strukturu mobilní verze webu www.ibobr.cz., prozkoumá způsob čerpání dat pro články na mobilním webu a způsob komunikace s prostředím Joomla, případně jiný způsob aktualizace obsahu mobilní verze webu.
Součástí webu bude funkční cvičný test, vytvořený z dodaných úloh, který se bude skládat z několika variant úloh (test nebude načítat data z databáze, ale bude kontrolovat správná řešení). Jedním z cílů je posouzení možností a zmapování funkcí HTML5 při tvorbě takového testu.

| | |
|---|---|
| Rozsah grafických prací: | **CD ROM** |
| Rozsah pracovní zprávy: | **40** |
| Forma zpracování bakalářské práce: | **tištěná** |
| Seznam odborné literatury: | **viz příloha** |

| | |
|---|---|
| Vedoucí bakalářské práce: | **PaedDr. Petr Pexa, Ph.D.**<br>Katedra informatiky |
| Datum zadání bakalářské práce: | **16. dubna 2013** |
| Termín odevzdání bakalářské práce: | **30. dubna 2014** |

Mgr. Michal Vančura, Ph.D.
děkan

L.S.

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

V Českých Budějovicích dne 16. dubna 2013

## Poděkování

Děkuji vedoucímu mé bakalářské práce PaedDr. Petru Pexovi, Ph.D. za odborné vedení práce, věcné připomínky, dobré rady a vstřícnost při konzultacích a vypracovávání bakalářské práce.

# Obsah

# 1 Úvod

Úvodem bych chtěl říct, že téma oblasti IT (informatiky), konkrétně responzivní design, je oblast poměrně široká a hluboká. Zadáním této práce kromě praktického výsledku je zároveň i sepsání jeho obsahu v následujících kapitolách. Responzivní design v posledních letech zažívá růst a také jeho popularita roste více a více. Dříve v nepříliš častém používání této tehdy nové technologie na webech, se dnes stala samozřejmost ne-li povinnost. Největším zlomem v této oblasti se stal masivní nárůst různé drobnější elektroniky schopné využívat internet a zobrazovat nejrůznější webové stránky. A v tom nastal právě problém, jelikož většina telefonů a později tabletů, měla a stále má menší rozlišení, než současné počítače. To přineslo samozřejmě problémy, jelikož se webové stránky správně nezobrazovaly. Proto vznikl tzv.: responzivní design.

V mé bakalářské práci se podíváme na to, jak to celé funguje a proč to vzniklo. Zároveň představíme, jak nejlépe v současném trendu responzivní design používat. Konkrétně v open-source[18] redakčním systému Joomla[20], který jako takový na to v podstatě vliv nemá. Důležitější je, že se pohybujeme aktuálně v html[1] dokumentu v nejnovější verzi 5, tedy html5[2], která je vlastně předpokladem k responzivnímu webu v tzv.: full-responzivitě a námi používaného frameworku[23] bootstrap[24]. To je důležité, ale i splňuje aktuální náročné nároky dnešních webových aplikací. Ale o tom až později. Jen je třeba říci, že bootstrap je jedna z nejnovějších a jedna z nejosvědčenějších metod správně se zobrazujícího responzivního webu. Proto jsem si vybral právě tuto technologii pro realizaci této práce. To jsou zřejmě jasné důvody. Hlavním cílem mé bakalářské práce je vysvětlit čtenářům, jak to vzniklo, jak to funguje, osvědčené způsoby i samozřejmě, co nejlepší možné výsledky. Všechny reálně fungující výsledky jsou funkční v přepracované šabloně z původního webu ibobr.cz. Ke své práci budu používat především vlastní získané znalosti. Nebudu přímo vycházet z knižních publikací, ale uvedu zde několik citací z Wikipedie čerpané z internetu.

Celá šablona funguje v redakčním systému Joomla, který však na responzivní zobrazení v podstatě žádný vliv nemá. Více uvedu v kapitole, která se tomuto problému přímo věnuje. Žádné domněnky ani smyšlenky jsem nepoužil, jsou to, troufnu si tvrdit, čistá fakta. Samozřejmě ale z mého pohledu, dosavadních zkušeností a tedy subjektivního názoru. To je v podstatě vše na úvod.

Pro správné pochopení této práce jsou potřebné minimálně alespoň základní znalosti v oblasti informačních webových technologií, především znalost html značkovacího jazyka a kaskádové styly. Bude se možná hodit i znalost $UX_0$ designu a cit pro něj, taktéž základní znalost Javascriptu $(JS)_{5,}$ či spíše jeho knihovny $Jquery_6$.

## 1.1. Cíle práce

Sestavení, realizace praktického funkčního responzivního designu, upravené, přepracované šablony z původní webové stránky http://www.ibobr.cz. Nová šablona funguje na frameworku bootstrap. Konečným cílem je výsledek a jeho zhodnocení až v závěru. Cílem mé bakalářské práce je přiblížit responzivní design pro základní pochopení při četbě či studiu této bakalářské práce a přiblížit tuto problematiku co nejvíce lidem, respektive čtenářům pomoci správně pochopit tuto problematiku. Teoretická část se dá použít obecně a zároveň i stejným způsobem funguje upravená šablona.

## 1.2. Metoda práce

Pro sestavení mi pomohly především dosavadně dosažené dostatečné znalosti a zkušenosti, které jsem získal v průběhu let v responzivní problematice. Literatura a dokumentace jsou v tomto případě až na druhém místě, ale samozřejmě nepochybně přispěly k získání určitých znalostí. Ale nebudu na literaturu a dokumentaci přílišným způsobem odkazovat právě kvůli znalostem na odpovídající úrovni. Vystačím si maximálně s internetovými zdroji, což v IT (informatice) jako odvětví, je asi víc než na místě.

### 1.3. Výsledky

Výsledkem je funkční praktický příklad šablony správně se zobrazující v šířkách displeje od 320px responzivního zobrazení, které výchozí šablona neobsahovala. Web s touto šablonou pak tedy funguje správně i na telefonech a tabletech v nižších rozlišeních, které jsou pro neresponzivní web nemyslitelné.

# 2 Teoretická část

## 2.1. Co je responzivní webdesign?

Začátkem tohoto velkého tématu, které postupně probereme až k funkčnímu výsledku, bude potřebná definice, tak jako v každém správném příkladu.

Co to vlastně je responzivní design?

Responzivní web design (anglicky Responsive web design) je pojem, se kterým přišel americký programátor Ethan Marcotte ve stejnojmenném článku na blogu A LIST Apart.Jedná se o způsob stylování HTML dokumentu, které zaručí, že zobrazení stránky bude optimalizováno pro všechny druhy nejrůznějších zařízení (mobily, notebooky, netbooky, tablety atd.). Především díky vlastnosti Media Queries, která je zahrnuta ve specifikaci CSS3, lze rozpoznat vlastnosti zařízení, na kterém je stránka prohlížena a přizpůsobit tak samotnou stránku a její obsah.". [1]

Rozebereme si to postupně. Stylování html dokumentu znamená, že nastylujeme html atributy$_1$, pomocí kaskádových stylů (css$_3$) tak, aby se celý obsah webu zobrazoval zcela správně. Nás bude aktuálně zajímat nejpoužívanější a nejaktuálnější verze kaskádových stylů (css3$_4$).

Co jsou kaskádové styly?

Kaskádové styly (v anglickém originále Cascading Style Sheets se zkratkou CSS) je jazyk pro popis způsobu zobrazení elementů na stránkách napsaných v jazycích HTML, XHTML nebo XML. [2]

Tady je důležité si říci, že pro nás je skutečně důležitá verze css 3$_3$, a to prakticky výhradně. Protože navíc obsahuje funkci media queries, která je nezbytná pro správné zobrazení v různých šířkách displeje, jelikož umí detekovat rozlišení displeje. Toto vše detailněji probereme v kapitole věnující se media queries a css 3. Teď jsme si vysvětlili základní pojmy. Uvádět co je html jazyk nebudu, jelikož je to v základních znalostech požadovaných v úvodu pro pochopení. Jen si snad připomenout, že html je značkovací jazyk a zahrnuje veškerý obsah webu a jeho tzv.containery[2], to znamená, obsahové části layoutu$_3$ např. textu. Ten ovšem samotný není schopný se chovat responzivně bez definovaných css stylů a někdy

i javascriptu. Záleží na složitosti obsahu a všech vychytávkách, které chce mít na webu v responzivním zobrazení.

Co je JS, neboli Javascript? Je to objektově orientovaný skriptovací jazyk, který funguje na straně klienta.

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk, jehož autorem je Brendan Eich z tehdejší společnosti Netscape. Nyní se zpravidla používá jako interpretovaný programovací jazyk pro WWW stránky, často vkládaný přímo do HTML kódu stránky. Jsou jím obvykle ovládány různé interaktivní prvky GUI (tlačítka, textová políčka) nebo tvořeny animace a efekty obrázků. [3]

Není úplně podstatný při použití základních funkcí responzivity, protože v zjednodušené formě by si člověk vystačil jen s media queries, společně s css3 a samozřejmě html. Ale v praxi to tak příliš nefunguje, protože weby vyžadují složitější operace v responzivním zobrazení jako např. změna menu na vyjíždějící drop-down menu na jeden klik v odpovídajícím rozlišení, protože celé menu by se v tom rozlišení jednak moc nevešlo a jednak by zabralo hodně výšky, což je např. na mobilním zařízení dost nepraktické. To je asi vše v prvním tématu. I tohle všechno, co popisuji, uvidíme v naší šabloně.

## 2.2. Jak funguje responzivní web? (HTML5 + CSS3 +JS)


**Obrázek 1 - Technologie responzivního webdesignu [Zdroj: blogspot.com]**

V další kapitole se podíváme na to, jak funguje vlastně responzivní web. Takže jak jsme si již řekli na začátek, vše potřebné. Responzivní web stojí na třech základních stavebních kamenech. Základní stavební kameny pro aktuálně správně responzivní web je tedy již z nadpisu úplně jasně vidět HTML5+CSS3+JS.

V předchozí kapitole jsme si stručně řekli, jak to funguje a co to vlastně je. Ještě velmi důležitá věc, proč vlastně používat responzivní webdesign a proč nezobrazovat web tak jak je? Je to jednoduché, protože web se nám zobrazí pořád stejně nezávisle na rozlišení. A jelikož je velmi nepraktické používat posuvník a web nám ani nevyjde na celý displej, je to velmi neestetické v tom lepším případě. Většinou však absolutně nepoužitelné a v tomto případě se web nedá použít na přenosných zařízeních s nižším rozlišením (mobilní telefony, PDA, netbooky, tablety apod.). Popularita těchto přenosných zařízení s nižším rozlišením, jak uvidíme v grafu na konci, však stále rapidně stoupá.

Celá bakalářská práce se tedy zaměřuje na to, jak těmto problémům čelit, tedy vlastně hned jak nějakým rozumným způsobem tento problém instantně řešit. Pojďme si tedy vzít první stavební kámen a to je HTML5. Jak funguje HTML5 není třeba nějakým způsobem vysvětlovat. Jak jsem již zmínil, je to značkovací jazyk. Veškerý obsah se zobrazuje v jeho hlavních elementech. Struktura vypadá jako klasické html tak jak jej známe z verze 4. Ale samozřejmě se liší. Detailnější popis zde: http://www.w3.org/TR/html5-diff/. S tímto obsahem se pak dále pracuje v css a js. Upravuje se a mění se pomocí css a js. Html samo o sobě vůbec neřeší

responzivní zobrazení, jen v zjednodušené formě dodává obsah, se kterým se pracuje v nějakých jeho určených elementech.

Element je html tagem pro vysvětlení. Příklad 1.

```
  <!DOCTYPE html>
<html>
<head>
  <body>
  <!-- komentář -->
  <div id="content"><p>Můj text</p></div>
  </body></html>.
```

Rozdíly oproti verzi 4 jsou, že v html 5 je spousta nových funkcí, které i při responzivním designu ocení každý vývojář. Pojďme ale dále, na druhý stavební kámen a to je css3. Ten je dá se říci nejdůležitější jako takový pro responzivní design. Hlavně kvůli zmíněné funkci media queries, která umožňuje tzv. breakpoint[10], tzn. zlomový bod, v tomto případě rozlišení, při kterém se chová podle daného zápisu v každém breakpointu. Jak funguje media queries a breakpoint se podrobně dozvíme v další kapitole. Jak jsme si řekli, css jsou kaskádové styly. V první kapitole bylo řešeno, co to znamená a jaká je definice dle Wikipedie. Důležité je, že css nám dodává vlastně veškerý vzhled našemu layoutu html, který je jen obsahový a upravuje ho přesně dle našich představ.

Definice kaskádových stylů sestává z několika pravidel. Každé pravidlo obsahuje selektor a blok deklarací. Každý blok deklarací pak obsahuje deklarace oddělené středníky ; a každá deklarace sestává z identifikátoru vlastnosti, následuje dvojtečka : a hodnota vlastnosti. Nepovinně ještě může následovat označení !important, které zvýší sílu deklarace. [4]

To nám jasně říká, jak to funguje. CSS je vlastně takovým vzhledem html kódu, ve kterém se dají vzhledově upravovat jednotlivé html elementy, tak jak je zrovna potřeba.

Příklad 2 z naší šablony:

```
.banner img {
width: 100%;
}
```

Výše uvedený příklad nám ukazuje, že chceme selektovat (selektorem)[14] element div s id banner a zároveň to je taky element obrázku tedy img v tomto contentu[15]. A nastavíme mu hned vlastně šířku. Takže takhle to nějak může vypadat zjednodušeně. Asi je z toho zcela zřejmé, jak kaskádové styly vlastně fungují. To vše co popisuji, bude vidět v naší upravené šabloně z ibobr.cz, víc než stoprocentně. A teď se konečně podíváme na poslední základní kámen z naší trojice. A to je JS celým názvem Javascript. Co to Javascript je, jsme si již řekli. Teď je třeba říci a zároveň ještě připomenout, že to je jazyk fungující na straně klienta. Z toho vyplývá, že se projeví až načtením stránky v prohlížeči, oproti například jazyku PHP[9], který se zpracovává na straně serveru. Čistý Javascript se označuje za nativní.

V dnešní době již moc nativní JS používaný není. Nastupují tzv. Javascriptové kompilátory, nebo knihovny Javascriptu. Za zmínku z kompilátoru stojí AngularJS, Dart či CoffeeScript a NodeJS. Kompilátor je v podstatě samostatný jazyk s vlastní syntaxí, který se zpátky kompiluje do nativního Javascriptu, aby fungoval téměř všude. Výhoda je především psaní méně kódu a lepší jednodušší syntaxe[27].Ale to nás však nemusí zajímat. Nás zajímá knihovna Jquery. Proč? Protože ji využijeme u naší responzivní šablony. Konkrétně budeme používat verzi Jquery 2.1.0.

Najdeme ji v rootu[22] šablony zde /js/jquery-2.1.0.min.js. Používat budeme minimalistickou verzi. Jak nám značí předpona min.js. Co znamená minimalistická verze?

To znamená, že jsou tam odstraněny všechny prázdné mezery a znaky, abychom získali co nejmenší velikost v kb kvůli rychlosti načítání samotného scriptu, a aby byl co nejrychlejší.

Nejpoužívanější z Javascriptových knihoven je, jak jsem již řekl, Jquery. A na ni se budeme soustředit. Co je jquery? Zase oblíbená definice.

jQuery je javascriptová knihovna s širokou podporou prohlížečů, která klade důraz na interakci mezi JavaScriptem a HTML. Byla vydána Johnem Resigem v lednu 2006 na newyorském BarCampu. jQuery je svobodný a otevřený software pod licencí MIT.". [5]

Proč? Protože ji využijeme přímo u naší responzivní šablony jako knihovnu. I bootstrap, o kterém jsme ještě vůbec nemluvily, tedy front-end knihovna, které se budeme zvlášť věnovat, a na které stavíme celý responzivní web, také samozřejmě potřebuje jquery. Jelikož bootstrap je nepřímo závislý na této knihovně díky vlastním pluginům[17] napsaných v jquery.

A uvedeme si samozřejmě příklad jquery z naší šablony, vlastní funkce posuvník stránky nahoru.

Příklad 3 opět z naší šablony (vlastní skript):

```
"<script type="text/javascript"></script>
<script>
$( document ).ready(function() {
   if ( ($(window).height() + 100) < $(document).height() ) {
   $('#top-link-block').removeClass('hidden').affix({
         offset: {top:100}
   });
}
});
</script>".
```

Závěrem k JS. Je to asi nejmocnější nástroj v responzivním designu, ale hodí se ke složitějším věcem. A to co vlastně potřebujeme, tak např. ve front-end knihovně bootstrap máme řešeno.

Alespoň většinu. Je to nejlepší způsob, jak používat velmi kvalitní responzivní zobrazení.

Bootstrap již obsahuje vlastní jquery pluginy a css styly. Ale o bootstrapu více v kapitole tomu se věnující. Tři základní stavební kameny jsme probraly. Všechny tyto základní kameny spolu dohromady souvisí a dávají nám správně fungující responzivní design.

Na závěr této kapitoly si uvedeme malý příklad bez media queries, to se probere v další kapitole, jak tyto stavební kameny fungují na sebe vázaně. Např. máme nějaký div, to jest HTML5 základ.

Příklad 4:

```
<div id="mydiv">lorem ipsum</div>
```

Pak si nadefinujeme css, tak aby šířka byla sto procent. A to znamená, že se nám tento text bude automaticky přizpůsobovat na šířce dle aktuálního rozlišení.

Příklad 5:

```
div.mydiv {
width: 100%;
}
```

V Javascriptu by se pak dalo nastavit třeba šířku podle aktuální šířky daného contentu pouze v případě, že bychom nepoužívali 100 procentní šířku. Jinak by se to přizpůsobilo i bez JS. Ale to již nebudu uvádět, to je pro pokročilejší jedince. A máme hotovo, vysvětleno je zde jen to, jak je to propojeno. Možností je spousta.

### 2.3. Detekce rozlišení displeje (media queries)

Následuje téma z oblasti css3. Nová revoluční funkce media queries. Nejdřív něco o CSS3.

CSS3 (Cascading Style Sheets 3) je už třetí verzí kaskádových stylů CSS, a to již od roku 2005, kdy byl vývoj této technologie zahájen konsorciem W3C. Očekávané dokončení této technologie se předpokládá na rok 2015, ale již dnes je většina vlastností podporována nejběžnějšími webovými prohlížeči. [6]

Teď k Media queries. Opět si vezmeme přesnou definici. Jsou v ní obsaženy i pojmy, které jsem zmínil a krásně nám to sedí.

Media Queries je CSS3 modul umožňující adaptabilní vykreslování webových stránek podle různých činitelů jako rozlišení obrazovky, či velikost obrazovky (např. obrazovka smartphone vs. monitor u PC). Poprvé byly použity již v roce 2001 W3C a staly se doporučeným standardem v červnu 2012. Je to základní stavební kámen pro responzivní web design. Díky tomu lze jen jednoduše upravit styly a výrazně tak zlepšit uživatelům UX při prohlížení. [7]

A hned dále si uvedeme další citovanou definici a také, jak celá funkce vypadá.

Media Queries se považují za poslední úroveň responzivní web designu. Jsou to pravidla, díky kterým lze měnit stylování dokumentu v závislosti na šířce obrazovky zobrazovaného zařízení. Následující stylování (červené pozadí celého dokumentu) bude uplatněno pouze tehdy, pokud šířka prohlížeče na použitém zařízení bude v rozsahu od 660px do 780px." [8]

Funkční příklad v pořadí 6:

```
@media (max-width: 780px) and (min-width: 660px){
      body{
            background-color: red;
      }
}
```

Z příkladu jde jasně vidět, jak celá funkce vypadá a jak také funguje. Tyto stejné funkce používá naše responzivní šablona, jen jsou již zakomponovány v bootstrapu, v jeho css souborech. Funkce je vcelku velmi jednoduchá a logická. Lze ji použít i bez and bez použití druhé podmínky rozlišení. Takže stačí zadat jen max-width nebo min-width, tedy jeden parametr. Tyto podmínky tedy již chápeme, tak si můžeme uvědomit, že to jsou právě ony, ty breakpointy, o kterých jsem mluvil

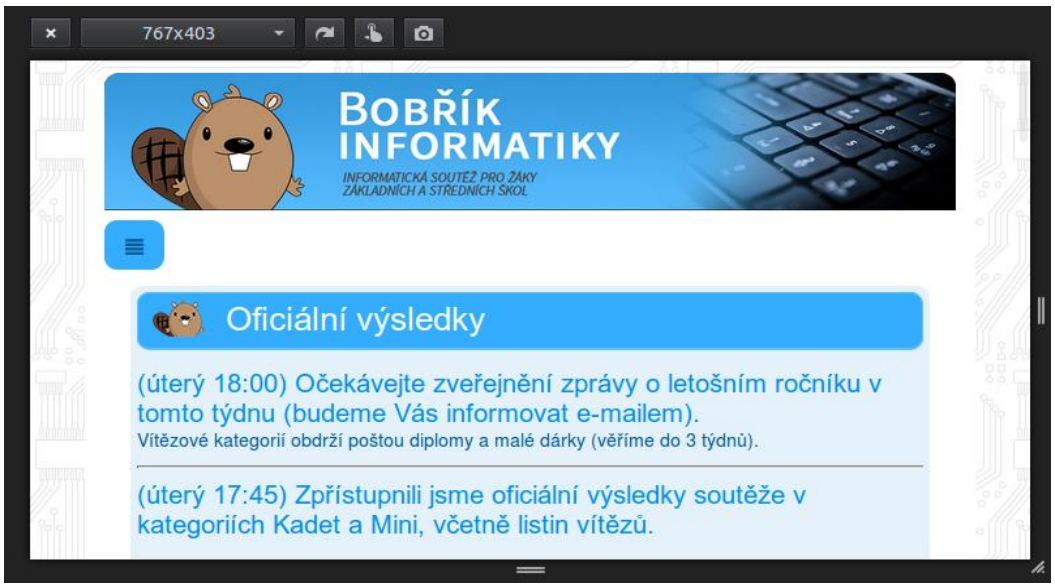
v některé z předchozích kapitol. Budeme vycházet z našeho příkladu z Wikipedie pro snadné pochopení.

A tyto hranice pixelů (breakpointy) v první definici v závorkách max-width: 780px znamená, že podmínka bude platit do 780px. 781px je breakpoint zlomový bod, který změní css styly dle aktuálně platných podmínek pro rozlišení od 781px. Pokud by se podmínky neobjevily, budou automaticky přiřazeny css styly pro zbytek rozlišení, které nejsou v media queries. To znamená pro základní, nebo jinak řečeno hlavní css styly, které jsou platné pro všechny rozlišení. Většinou se proto vytvoří obecný soubor s názvem style.css. To není ale pravidlem. A tak jsme vyřešili opět problém s tím, abychom byli schopni použít funkci media queries.

Bootstrap toto všechno sice řeší automaticky, ale je potřeba taky tomu rozumět. A kdybychom chtěli, tak si to kdykoliv dle libosti upravíme dle rozlišení, které chceme v naší šabloně v css stylech bootstrapu. Pokud nepoužijeme Bootstrap anebo jiný framework, ale o tom opět až v další kapitole této práce, pak bychom to museli sami takto tvořit.

A na závěr uvádím screeny z naší šablony na http://ibobr.cz / hraničních breakpointů. Aby bylo jasně vidět, kde se to láme, oba obrázky můžete porovnat, jak dochází ke změně rozlišení.

Zajímá nás jen šířka, aktuálně vidíme 767px.



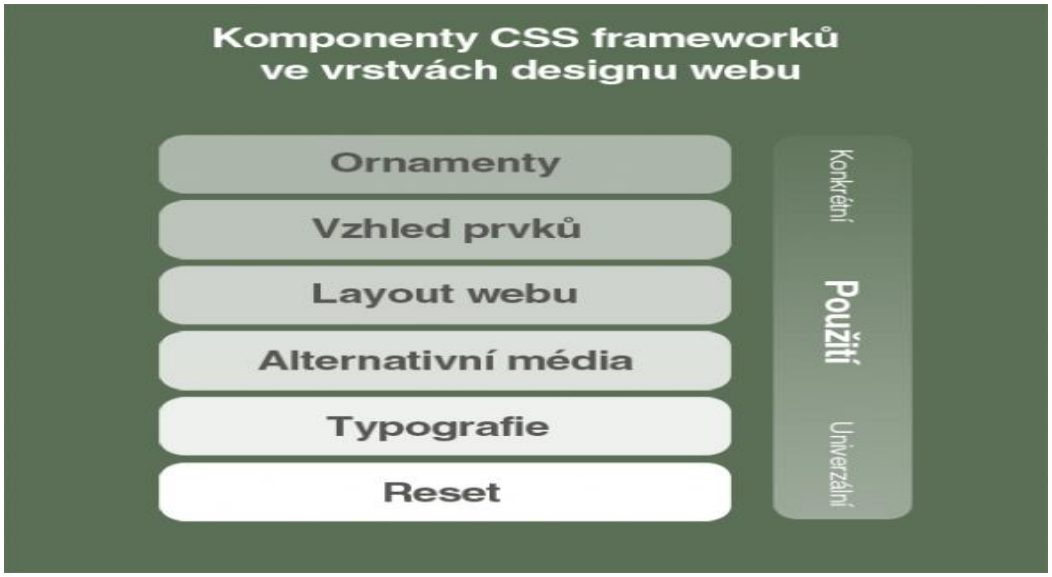

**Obrázek 2 - Screen1 [Zdroj: ibobr.cz]**

Zde na obrázku číslo 3 je šířka aktuálně 768px. Rozdíl jeden pixel a vidíme, že se již web změnil. Např. menu, a to právě z toho důvodu, že se překročil breakpoint a tím pádem vstoupily v platnost nové podmínky nadefinované ve funkci media queries s daným aktuálním rozlišením.


**Obrázek 3 - Screen2 [Zdroj: ibobr.cz]**

## 2.4. Pomocník front-end knihovna, komponenty (css Framework)

Toto téma je rozsáhlé. Budu se mu věnovat pouze v tom, co budeme v základním měřítku potřebovat v této práci a z čeho vycházíme. Nejdříve si řekneme, co je to vlastně front-end Framework a k čemu je dobrý, jak moc je potřebný a jaké problémy řeší. Prvně se podíváme na tento následující graf. Obrázek co front-end css Framework vlastně řeší, jaké aspekty problému pomocí jeho komponent. Jinými slovy co zahrnuje.




**Obrázek 4 - Komponenty CSS frameworků [Zdroj: zdrojak.cz]**

Z těchto komponent je, troufnu si říci, nejdůležitější Reset, což je velmi důležitá věc. Proč?

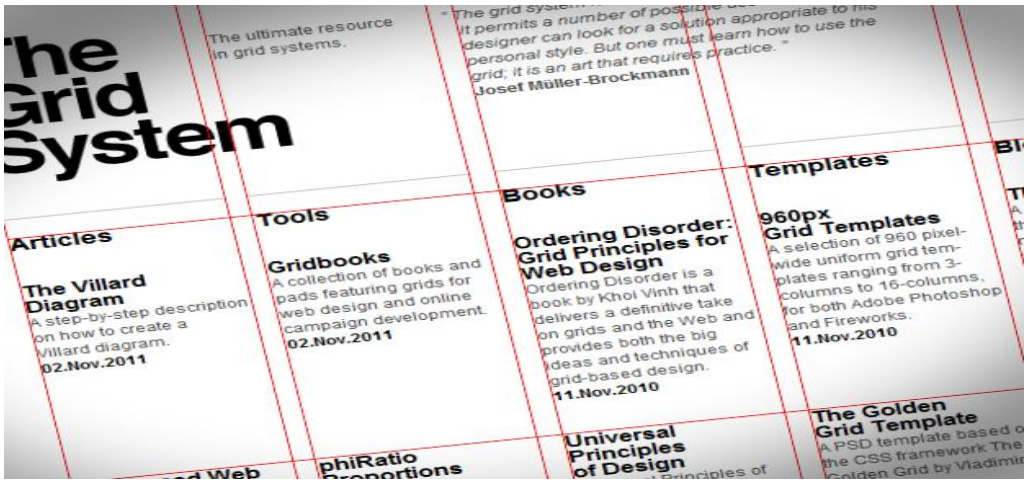
Protože resetuje, tedy vynuluje základní pravidla css, tak aby se všechny prohlížeče chovaly víceméně stejně, aby pro všechny začaly platit nová a pokud možno stejná pravidla. Což je pro nás důležité z několika důvodů. Hlavně aby se web zobrazil s co možná nejmenším rozdílem zobrazení css stylizací, které by se jinak zobrazovaly u některých elementů odlišně, tedy v jiných prohlížečích, nebo alespoň sjednocení v těch základních, jako např. Google Chrome, Firefox, Internet Explorer, Opera. Nejpoužívanější reset je reset.css je od Erica Meyera více podrobností o něm, zde:http://meyerweb.com/eric/tools/css/reset/.

Další komponentou je Typografie. Typografie nám po vyresetování css stylů dodává nový vzhled css prvkům písma.

Toto nebudu příliš nějak dál rozebírat. To stejné Alternativní média. Tam se řeší vzhled tisku, tím je myšleno to tzv. alternativní médium. To pro nás není nijak zvlášť podstatné.

Podstatné je další položka a to je Layout webu. Je to vlastně zjednodušeně rozvržení stránky. Zpravidla fixní anebo responzivní, jinak řečeno elastický layout16, v případě frameworků, fungujících na gridu. To nám nahrává další otázku. Co je to grid? Je to vlastně de facto rozvržení toho layoutu, jeho standardní rozlišení a počet sloupců v tomto daném layoutu. Třeba v našem bootstrapu, který používáme je k vidění příklad gridu v bootstrapu zde: http://getbootstrap.com/examples/grid/. Asi není víc co dodat.

Na obrázku, můžeme vidět, jak vypadá grid systém. Čáry jej přímo ohraničují. Tedy je to rozvržení layoutu.




**Obrázek 5 - Grid rozvržení webu [Zdroj: 99designs.com]**

Náš grid rozvržení si taky názorně ukážeme v praktické části této práce.

Pojďme na další položku Vzhled prvků. Zabývá se vzhledem prvků css např. formulářů. Případně nějakých html polí apod. Poslední položkou jsou Ornamenty. Tak to jsou, jak se dá z názvu předpokládat, sady různých tlačítek či jiných zvláštních elementů, které slouží k dekoraci. Tyto věci jsou obvykle zahrnuty již ve většině dnešních frameworků, samozřejmě ne všechny ve všech případech. Nás bude zajímat ale náš použitý framework bootstrap, kterému se budeme věnovat

podrobně hned v další kapitole. Kromě tohoto frameworku jsou ještě i jiné frameworky např. YAML, LESS FRAMEWORK, FOUNDATION….

Samozřejmě nejpoužívanějším komerčním frameworkem, a dle mě nejlepším řešením z aktuální nabídky, je náš zmíněný a použitý bootstrap. Jelikož jsem se zmínil o tom, že tyto frameworky jsou komerční, většina open-source[14] je volně ke stažení. Tak je také možnost si vytvořit framework vlastní. Ale u obou těchto variant jsou určitá úskalí.

Tedy výhody a nevýhody. Výhodou vlastního frameworku je bezpochyby to, že člověk si vytvoří přesně to chce bez nějakých věcí navíc. Z toho plyne ve většině případů rychlejší výkon a menší paměťové nároky na výkon serveru. Ale není to pravidlem. Záleží, jak šikovní jste.

Většinu lidí od vlastního frameworku odrazuje velká náročnost, debugging[19], složitost, časová náročnost. Zase pozitiva komerčního frameworku, jsou snadná dostupnost díky open-source, pokud tedy je, pak mnoho funkcí, všechno otestované, pravděpodobnost závažných chyb je minimální, díky velké testovatelnosti a šikovným vývojářům. Nevýhoda je častá neznalost návrhu frameworku jeho struktury, a jak jsem již uváděl, slabší výkon a paměťové nároky. Protože ty knihovny jsou komplexní, obsahují skoro vše, co by web-designér nebo kóder mohl potřebovat. Probrali jsme tedy front-end css framework a jeho úskalí.

A vyřešili další problém. Jestli používat css Framework nebo ne. Zvážení je na vás.

Já ho používám, ale ne vždy. Záleží na daném webu a vhodnosti jeho použití. Na závěr ještě slíbenou definici, co je to Framework. Teď jsme schopni to teprve pochopit.

Framework (aplikační rámec) je softwarová struktura, která slouží jako podpora při programování a vývoji a organizaci jiných softwarových projektů. Může obsahovat podpůrné programy, knihovny API7, podporu pro návrhové vzory nebo doporučené postupy při vývoji. [9]

## 2.5. Co je Twitter bootstrap ? (css Framework)

Probral jsem co je to framework, z čeho se skládá a jak funguje. Nás ale bude zajímat náš použitý framework a to je Twitter Bootstrap. Co je Twitter Bootstrap? Zjednodušeně front-end css framework.

Twitter Bootstrap je velmi jednoduchý a volně dostupný soubor nástrojů pro vytváření moderního webu a webových aplikací. Nabízí podporu nejrůznějších webových technologií HTML, CSS, JavaScript a mnoho prvků, které je možné snadno implementovat do své stránky. Pro použití Twitter Bootstrap jsou nutné základní znalosti HTML a CSS. Interaktivní prvky jako jsou tlačítka, boxy, menu a další kompletně nastavené a graficky zpracované elementy je možné vložit pouze pomocí HTML a CSS.". [10]

Tento framework vytvořili Mark Otto a Jacob Thornton z Twitteru. Stal se jednoznačně nejpopulárnějším front-end framworkem a není divu. Je velmi efektivní, relativně jednoduchý a rychlý. Je neuvěřitelně trendový a moderní. Kromě toho obsahuje řadu velmi zdařile zpracovaných komponent, o kterých jsme se bavili v předchozí kapitole. Zde je výčet všech komponent.

Dohromady poskytují komponenty a JavaScript pluginy následující elementy uživatelského prostředí:

- Button groups – Skupiny tlačítek
- Button dropdowns – Vysouvací tlačítka
- Navigational tabs, pills, and lists – Záložky, pilulky a seznamy pro navigaci
- Navbar
- Labels – štítky
- Badges – „odznáčky“
- Page headers and hero unit – hlavičky stránky a „hero unit“
- Thumbnails – náhledy
- Alerts – výstrahy
- Progress bars
- Modals
- Dropdowns – vysouvací menu
- Tooltips
- Popovers
- Accordion
- Carousel – posuvný slider
- Typeahea

Nebudu určitě rozebírat všechny komponenty, jelikož by to bylo mnoho stránek a asi by se to minulo účinkem a bylo zároveň kontraproduktivní. Stačí nám vědět, co všechno z bootstrapu získáváme. A dále si to již upravujeme alchymisticky, dle libosti. Tedy tak, jak to potřebujeme v aktuální naší šabloně z http://ibobr.cz /. Jak aktuálně bootstrap funguje na našem webu, si řekneme v následující kapitole. Na závěr je třeba říct, že bootstrap se skládá z viz. kapitola 2.2. Nyní budeme vědět,

že používá dva stavební kameny a to CSS + JS. Pak ještě obsahuje .svg soubory pro své fonty, to je celý bootstrap.

### 2.6. Responzivní šablona fungující na bootstru

Co je bootstrap již víme. Pojďme se ale podívat, jak funguje na naší šabloně. Jako první je třeba si stáhnout framework bootstrap, nejlépe zde: http://getbootstrap.com/.

Rozbalíme si archiv do složky, které si určíme, v naší šabloně to máme uloženo v root šablony/js/bootstrap. Poté provedeme načtení celého bootstrapu do naší šablony, také nezapomenout načítat i jquery knihovnu, jak jsem již v úvodu zmínil. O načítání si více řekneme v 1 kapitole praktické části. Po načtení musíme zkontrolovat, zda vše funguje jak má, zda se css styly bootstrapu načítají správně. Pokud ano máme hotovo.

Ale jen načítání, které začne přepisovat veškeré styly, které doposud fungovaly přímo ze šablony. Tady ale práce začíná. Je třeba vše upravit do gridu bootstrapu jeho elementů nebo upravit na vlastní. Všechno zkontrolovat, upravit na správná rozlišení layoutu. Ideálně do 320px responzivního zobrazení a upravovat to tak postupně, aby to fungovalo s frameworkem, tak jak my chceme, např. menu, aby se správně javascriptově skrylo, pokud dosáhnu určitého breakpointu, nic takového instantně nefunguje, ale ty javascriptové funkce v komponentách jsou již připraveny. Př. elementy contentu, na bootstrap vzor např. .col-md-4 .

Vycházíme opět z gridu bootstrapu zde http://getbootstrap.com/examples/grid/.

Rozhodně framework není něco, nějaká hotová instantní věc, jak se na první pohled může zdát. Něco instantního neexistuje, co byste dostávali zabalené v krabici a vy byste si ho jen vytáhli a používali. Takhle to úplně nefunguje, ač je to skvělý pomocník.

Hlavní výhoda je že většina základních důležitých funkcí k správnému responzivnímu zobrazení tam již jsou. Ale i tak je kolem toho spousta práce.

Vyplatí se to v určitých případech, protože je to kvalitní řešení. Je to kvalitní řešení otestované špičkami v oboru. Protože vytvořit takovýto důmyslný framework není úplně triviální. Je to časově poměrně náročné a ne každý je to schopný zvládnout.

A také pokud se pracuje v týmu, je to zvlášť nevýhodné, když má každý jiný Framework. Je to pak velmi nepraktické. Navíc člověk si může být téměř jist, že bootstrap je plně dotáhnutý framework.

Pokud by se objevil problém, komunita to prakticky ihned opraví. Takže po té, co si aplikujeme framework a upravíme podle toho, jak potřebujeme, pak je teprve hotovo. To je dlouhá cesta, co všechno se dělalo slovo od slova. Nebudu vše přímo popisovat do detailu, protože by to bylo na několik bakalářských prací. V této práci se probírají základy, jak co funguje a jak se vyřešily problémy v jednotlivých kapitolách a došlo k výsledku, aby vše mohlo fungovat bezchybně, a o to jde. Staré css styly, tedy konkrétně soubor style.css v root sablony/css/style.css se nahradil souborem styleb.css.

Je to nový css soubor, který kooperuje s naším bootstrapem. Starý style.css se již nepoužívá, jelikož tam bylo málo věcí potřebných k co nejlepší synchronizací s bootstrapem. A nedalo se použít. Pak dále tak stejně se souborem template.css, ve stejném umístění. Některé věci zůstaly, ale jiné se musely přizpůsobovat, nebo pozastavit (zakomentovat) či odstranit některé zápisy, které definovaly některé věci, které již ve spolupráci s naším frameworkem, máme vyřešeny. Nebo naopak tam něco chybělo. Celkově výchozí šablona z ibobr.cz, ale odpovídala čistě fixnímu starému způsobu realizace šablon webu, který není nějakým způsobem elastický. Spousta vlastních css stylů, jen těžko udržovatelných, zvlášť v dnešní době nových technologií. Tady čerpáme většinu z frameworku a z našich stylu kooperujících s frameworkem na stejné báze v gridu a elementy.

Přizpůsobíme se tak pohodlně každému zařízení do maximální možné šířky 320px, což se považuje za full-responzivní design. Podle breakpointů, jinak řečeno zmenšování displeje, pak určujeme celý vzhled tím, že měníme především css styly a dáváme pokyny případně javascriptu, pokud má něco změnit podle toho nějakou svoji nadefinovanou funkci. V javascriptu se nic extra zásadního neřešilo. Web nebyl ani javascriptově postavený. To co potřebujeme, čerpáme plně z frameworku.

Snad napsání nějaké funkce a opět kooperace s frameworkem, což jak jsem říkal, také není nic úplně triviálního. Nepoužíváme zde nějaké propracované vychytávky, tak není tedy ani nic extra potřeba. Máme další kapitolu vyřešenu a problém objasněn.

## 2.7. Správné responzivní zobrazení

Máme tady kapitolu, kde se budu plně věnovat správnému responzivnímu zobrazení. Co to vlastně je správně fungující responzivní zobrazení? Žádná obecná definice pro tohle příliš není. Sám bych to definoval jako zobrazení, při kterém se celý web zobrazuje zcela správně podle předem připraveného návrhu, ve všech různých rozlišeních v toleranci pro ni určenou. To znamená pro nás do 320px maximální možné šířky. Samozřejmostí je, že se nemůže zatím úplně jistě při rozdílných rozlišeních, různých OS prohlížečů, různých šířek, vše vždy zobrazovat na 100% jistě. Samozřejmě je to naším cílem. Dá se říci, že se k tomu přiblížíme na maximum.

Proč? Protože se stává, že i když použijeme i již zmíněný reset.css od Erica Meyera, tak stejně ne vždy nám pomůže stoprocentně. I když riziko minimalizuje na minimum. Někdy si zkrátka prohlížeč dělá co chce dle libovůle tvůrce. Jsou to sice detaily, ale někoho to i tak zarazí.

Nic není dokonalé ani v IT průmyslu. Pojďme si však říci, jak si projít celou strukturu webu a zjistit, zda je všechno v pořádku. Projdeme si každý hraniční breakpoint v praktické části na závěr. A zkontrolujeme, zda se vše zobrazuje tak jak má. Jestli se zmenšením rozlišení web celkově zmenšuje a celý layout se mění, ztenčuje a elasticky přizpůsobuje danému rozlišení, do daného minimálního rozlišení šířky. Dalším ukazatelem je dobrá čitelnost textu. Žádný obsah by neměl výrazně přesahovat svůj daný element.

Nepoužívat nikde fixní šířky, ale elastickou šířku zadávanou v procentech. Nejdůležitější je přizpůsobovat css styly všem daným breakpointům. Jinými slovy každý breakpoint by měl mít vlastní deklaraci a vlastní přizpůsobené css styly. Kromě některých univerzálních css stylů, které se automaticky díky dané elastičnosti těchto elementů přizpůsobí. Ale to je menšina elementů, alespoň by měla být.

V praktické části se budeme více věnovat názorným příkladům s detailním popisem. Zde si jen ukážeme rozdíly rozlišení zobrazení v naší šabloně u responzivního zobrazení.

Výsledky si můžeme prohlédnout na ilustračních obrázcích naší šablony.


**Obrázek 6 - Screen3 [Zdroj: ibobr.cz]**


**Obrázek 7 - Screen4 [Zdroj: ibobr.cz]**

## 2.8. Základní vazba na RS Joomla

V této kapitole probereme základní vazbu na RS Joomla[18].I když se v podstatě dá říci, že tam žádná přímá vazba jako taková, co se týká responzivního webdesignu není. Jelikož responzivní webdesign nemá nic moc společného s redakčním systém, kterým Joomla je. Přesto tam základní vazba je už proto, že pracujeme na tomto redakčním systému. Což je jádro celého systému, které musíme ať chceme, nebo ne respektovat. A řídit se podle něho.

A tak je dobré uvést, na jakém CMS[9], vlastně fungujeme. Co je vlastně CMS Joomla? Definice zjednodušeně

Joomla! je bezplatný open source CMS pro účely publikování informací na internetu a intranetu. Je napsána v jazyce PHP a využívá databázi MySQL. Joomla! podporuje caching, indexaci stránek, RSS, tisknutelné verze stránek, zobrazování novinek, blogy, diskusní fóra, hlasování, kalendář, vyhledávání v rámci web serveru, lokalizace a vícejazyčné verze. [11]

Teď, když víme co je Joomla, tak se pojďme podívat na tu vazbu, byť minimální. Hlavní vazba je asi základní kostra joomly, s kterou se musí počítat při tvorbě responzivního designu. S určitým stylem načítání css stylů a javascriptů, kterému se musíme přizpůsobit. A to tedy také znamená přizpůsobovat se jednotlivým blokům elementů v celém layoutu. Jinak, jelikož je Joomla psaná výhradně v php (až na pár javascriptů a css), nám jiným způsobem nepřekáží.

Myšleno ve tvorbě responzivního designu. Posledním aspektem, který bych ještě uvedl je, že způsob aktualizace obsahu, jak mobilního, tak i klasického se nijak nemění, jelikož přizpůsobujeme responzivní celý web s daným obsahem. Ve starších tzv. mobilních verzích, se používal trochu pozměněný přizpůsobený obsah. To však dnes již v podstatě neexistuje, nebo v malé míře. Tímto bych zakončil teoretickou část. A pustil se do části praktické. Vyřešili jsme spoustu problémů a nyní bude následovat minimum teorie a více praxe. Potřebujeme se orientovat kde co přesně najdeme v celé šabloně. Je to určitě užitečnější a to je můj subjektivní názor.

# 3 Praktická část upravené šablony ibobr.cz

## 3.1. Soubory, knihovny, javascript v bootstrapu

Začínáme praktickou část, kde nebudu zbytečně teoretizovat a přejdeme prakticky hned do praxe. Avšak vysvětlit a popsat to musím. Všechny teoretické znalosti jsme v tom naprostém základu probrali, takže se již nemusíme zdržovat. Vše pečlivě popíšu a samozřejmě vysvětlím. Znalosti máme, proto pojďme více pochopit praxi. První kapitola jsou soubory, knihovny, javascript v bootstrapu. Než si ukážeme a zjistíme, jaké soubory, knihovny a javascript máme k dispozici, musíme si bootstrap stáhnout. Poté nainstalovat a celkově uvést do provozuschopného stavu. Jdeme na to.

Již jsem to zmiňoval v teoretické části. Pro jistotu ještě jednou. Stáhneme si bootstrap na této adrese http://getbootstrap.com/. Pro okamžité stáhnutí využijte tento přímý link, který jsem zatím neuvedl https://github.com/twbs/bootstrap/releases/download/v3.3.1/bootstrap-3.3.1-dist.zip. Rozbalíme si archiv a všechny soubory do složky bootstrap. Pak nakopírujeme do šablony, v našem případě, to je celé umístění opět root/templates/ibobr_responzivni/js/bootstrap. Pojďme se podívat na strukturu naší složky bootstrap, která se skládá ze tří podsložek.

A to css, fonts, js. Podle zkratek lze očekávat, co asi bude v obsahu těchto složek. První máme složku obsahující css soubory. Ve druhé složce budeme mít soubory se speciálními fonty a ve třetím javascriptové soubory v bootstrapu. Nebudeme rozebírat jednotlivé soubory. Řekneme si jen, že ve složce s css jsou klasické css soubory a k němu minimalistické verze, což je obzvlášť výhodné a mapovací soubory s příponou .map, pro pre-processory28 nás nebudou vůbec zajímat.

Jelikož je nepoužíváme. Tak ani nemá smysl definovat, co to je pre-processor. Ve fonts máme fonty, ve formátu .svg, kde si můžeme nastavit hezké písmo z bootstrapu. Ale to pro nás taky není, nikterak důležité, používáme vlastní. Poslední složkou je js. Tam máme javascripty pro bootstrap. Opět taky minimalistickou verzi a soubor pro načítání rozšíření bootstrapu pro různé další přídavné pluginy.

V neposlední řadě, také potřebujeme načítat důležitou knihovnu jquery pro javascript, kterou budeme neodmyslitelně potřebovat, jelikož bootstrap je na ní postavený. A pojďme na praxi. Potřebujeme teď načítat složky css, font, js. Fonty nebudeme načítat, používáme výchozí v css. Máme tedy vlastní fonty. Dále načítání různých js knihoven a css stylů se zpravidla načítají v hlavičce uzavřených v tomto html tagu[28].

```
<head></head>
```

Jak vypadá v praxi načítání v naší nové šabloně. Zde v souboru příklad 7:

```
"<head>
    <!-- Set the viewport so this responsive site displays
correctly on mobile devices -->
    <meta name="viewport" content="width=device-width,
initial-scale=1">

   <jdoc:include type="head" />
    <!-- Include bootstrap CSS -->
  <link href="<?php echo $this->baseurl
?>/templates/ibobr_responzivni/js/bootstrap/css/bootstrap.min.
css" rel="stylesheet">
    <link href="<?php echo $this->baseurl
?>/templates/ibobr_responzivni/css/styleb.css"
rel="stylesheet">
```

```
  <!-- Include jQuery and bootstrap JS plugins -->
  <script src = "<? php echo $ this-> baseurl?> / templates / ibobr_responzivni / js / jquery-2.1.0.min.js"> </ script>
  <script src = "<? php echo $ this-> baseurl?> / templates / ibobr_responzivni / js / bootstrap / js / bootstrap.min.js"> </ script>
  </head>" .
```

Vidíme, zde ty kurzívou zvýrazněné linky, jsou právě ty, které se konkrétně načítají.

*<!-- Include bootstrap CSS -->* Jak je z názvu tohoto komentáře patrné, tak načítáme css soubory bootstrapu.

<!-- Include jQuery and bootstrap JS plugins -->Jak je z názvu tohoto komentáře patrné, tak načítáme js soubory bootstrapu.

Pokud jsme takto učinili, máme hotovo. Všechny css soubory a js knihovny by měly být načteny. Můžeme si to ověřit např. v konzoli vašeho prohlížeče, či zdrojovém kódu.

Když se podíváme na staré načítání v hlavičce, příklad 8:

```
"<head>
    <jdoc:include type="head" />
    <link rel="stylesheet" href="<?php echo $document->baseurl; ?>/templates/system/css/system.css" />    <link rel="stylesheet" href="<?php echo $document->baseurl; ?>/templates/system/css/general.css" />

    <!--[if lt IE 9]><script src="https://html5shiv.googlecode.com/svn/trunk/html5.js"></script><![endif]-->
```

```
    <link rel="stylesheet" href="<?php echo $templateUrl;
?>/css/template.css" media="screen">
    <!--[if lte IE 7]><link rel="stylesheet" href="<?php echo
$templateUrl; ?>/css/template.ie7.css" media="screen"
/><![endif]-->

  <link rel="shortcut icon" href="favicon.ico" type="image/x-
icon">
    <script>if ('undefined' != typeof jQuery)
document._artxJQueryBackup = jQuery;</script>
    <script src="<?php echo $templateUrl;
?>/jquery.js"></script>
    <script>jQuery.noConflict();</script>

    <script src="<?php echo $templateUrl;
?>/script.js"></script>
    <?php $view->includeInlineScripts() ?>
    <script>if (document._artxJQueryBackup) jQuery =
document._artxJQueryBackup;</script>
  </head>".
```

Tak vidíme daleko více zbytečného kódu, který již v podstatě nepotřebujeme. Vše řešíme přes bootstrap zcela kontinuálně, lépe a efektivněji.

### 3.2. Úprava a přepracování výchozí šablony, porovnání

Na úvod této kapitoly než začneme, ukážeme si náš načrtnutý grid, např. pro desktopové rozlišení 1280/640 pixelů. Čísla 1-4 jsou 4 sloupce gridu


**Obrázek 8 - Screen5 [Zdroj: ibobr.cz]**

Pojďme nyní dále. Když porovnáme šablonu původní jen pro fixní šířku a novou šablonu na bootstrapu s elastickou šířkou, je rozdíl velký. Struktura je prakticky dost hodně odlišná. Protože ta původní starší verze šablony ani neodpovídá novějším standardům, takže se to muselo upravit, tak aby nám to vyhovovalo.

Porovnáme tedy přímo soubory. Protože ale nelze nějakým rozumným způsobem

poukazovat na každý detail, tak si každý, kdo studuje tuto práci, může učinit sám, z příloh obou šablon, před a po. Když si projdeme např.: soubor index.php, tak vidíme změny, především ve změně div elementů, konkrétně jeho tříd.

Příklad 9 viz:

```
"<div class="col-md-9 content">  nebo <div class="col-md-3">".
```

Oproti:

```
"<div id="art-main">  nebo  <div class="art-sheet clearfix">".
```

Vidíme právě v nově upravené responzivní šabloně, používáme již grid systém z bootstrapu, takže se nám to krásně přizpůsobí při různých rozlišeních a můžeme

s tím krásně fungovat. Samozřejmě to ale také znamená změny všech css stylů a celkově i jejich uzpůsobení a přepsaní všech elementů do gridu.

Pak si napíšeme třeba nějaké vlastní css styly, případně do nového gridu bootstrapu, které budeme aktuálně potřebovat.

A v podstatě to máme hotové. Vždy však záleží na přesných okolnostech.

Pak si přidáváme věci, jaké potřebujeme, dle vlastní potřeby. Já jsem přidal tlačítko nahoru vlastním jquery skriptem. Téměř vše potřebné čerpáme z bootstrap javascriptu. Nedoporučuji však nic v něm upravovat, pokud nemá dotyčný dostatečné znalosti.

Měly by se všechny funkce používat pokud možno instantně. Tak to bylo navrženo a to velmi dobře. Na řešení úprav javascriptových funkcí takového rozsahu, jaký bootstrap vývojáři vytvořili, jsou potřeba velmi pokročilé znalosti knihovny jquery a nejen to. Uvedu tedy alespoň příklad, jak to funguje. Většina funkcí je poměrně těžká.

Vybral jsem tuto funkci. Je celá tak, jak je přímo v javascriptu.

Příklad 10:

```
"function transitionEnd() {
   var el = document.createElement('bootstrap')

   var transEndEventNames = {
     'WebkitTransition' : 'webkitTransitionEnd',
     'MozTransition'    : 'transitionend',
     'OTransition'      : 'oTransitionEnd otransitionend',
     'transition'       : 'transitionend'
   }
   for (var name in transEndEventNames) {
     if (el.style[name] !== undefined) {
       return { end: transEndEventNames[name] }
     }
   }

 return false // explicit for ie8 (  ._.)
 }

 // http://blog.alexmaccaw.com/css-transitions
 $.fn.emulateTransitionEnd = function (duration) {
   var called = false, $el = this
```

```
    $(this).one($.support.transition.end, function () { called
= true })
    var callback = function () { if (!called)
$($el).trigger($.support.transition.end) }
    setTimeout(callback, duration)
    return this
  }


  $(function () {
    $.support.transition = transitionEnd()
  })".
```

Je složité popsat, co tato funkce dělá laikovi. Je určena někomu se znalostmi na vysoké úrovni. Řeknu to zjednodušeně. Funkce, jak anglický název napovídá, nám zajišťuje jakýsi přechod. Jsou to přechody css. Tato funkce zajistí kompatibilitu přechodů pro všechny prohlížeče.

Jinak by se mohlo stát, že by to ve všech přechodech css stylů nefungovalo správně. Toto všechno nám zajistí framework. Psát si takovouto funkci, není absolutně nic jednoduchého, takže nám bootstrap pomůže a využijeme již něco otestovaného a spolehlivého. Dál už se javascriptům, nebudu věnovat. Patřilo by to do odvětví programování v javascriptu, v tomto případě programování v knihovně jquery.

### 3.3. Různé šířky (skládání webu dle rozlišení) media queries v praxi

Kapitola, kde využijeme znalosti z teoretické části z kapitoly 2.3. Tentokrát se více zaměříme na praxi, jelikož teorii máme zvládnutou. Podíváme se, jak v naší šabloně přímo media queries funguje.

Naše bootstrap struktura, pár příkladů:

Umístění root/templates/ibobr_responzivni/js/bootstrap/css/bootstrap.css.

Příklad 11:

```
"@media (min-width: 768px) {
 .lead {
   font-size: 21px;
 }
}
@media (min-width: 992px) {
 .container {
   width: 970px;

}
}

@media (min-width: 1200px) {
 .container {
   width: 1170px;
 }
}".
```

Tady vidíme, dá se říci částečné poloviční breakpointy. To znamená, že v této fázi fungují spíše, jako podmínky pro danou šířku. Kde se používá, jak jsme si v teorii řekli, pouze jeden parametr ze dvou.

Zde vidíme druhý parametr max-width v názorném příkladu.

Příklad: 12

```
"@media screen and (max-width: 767px) {
 .table-responsive {
   width: 100%;
   margin-bottom: 15px;
   overflow-y: hidden;
   -ms-overflow-style: -ms-autohiding-scrollbar;
   border: 1px solid #ddd;
 }
 .table-responsive > .table {
   margin-bottom: 0;
 }".
```

Pak zde uvidíme už celý breakpoint, který platí pro rozsah rozlišení daný oběma parametry a to min-width a max-width.

Příklad 13:

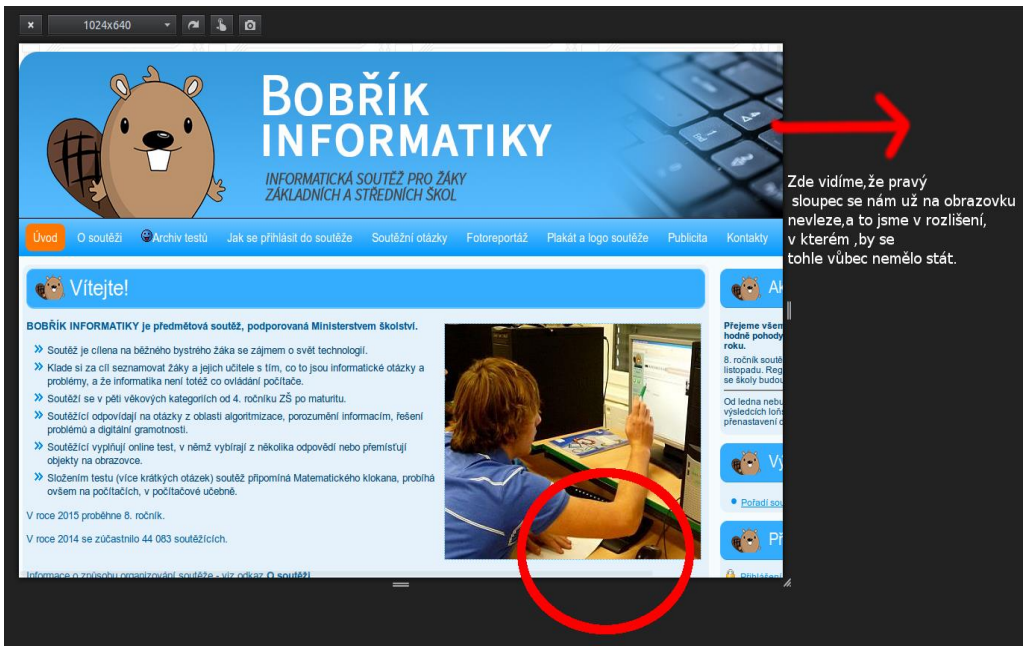
```
"@media (min-width: 768px) and (max-width: 991px) {
 .visible-sm-block {
   display: block !important;
 }
```

```
}
@media (min-width: 768px) and (max-width: 991px) {
 .visible-sm-inline {
   display: inline !important;
 }
}".
```

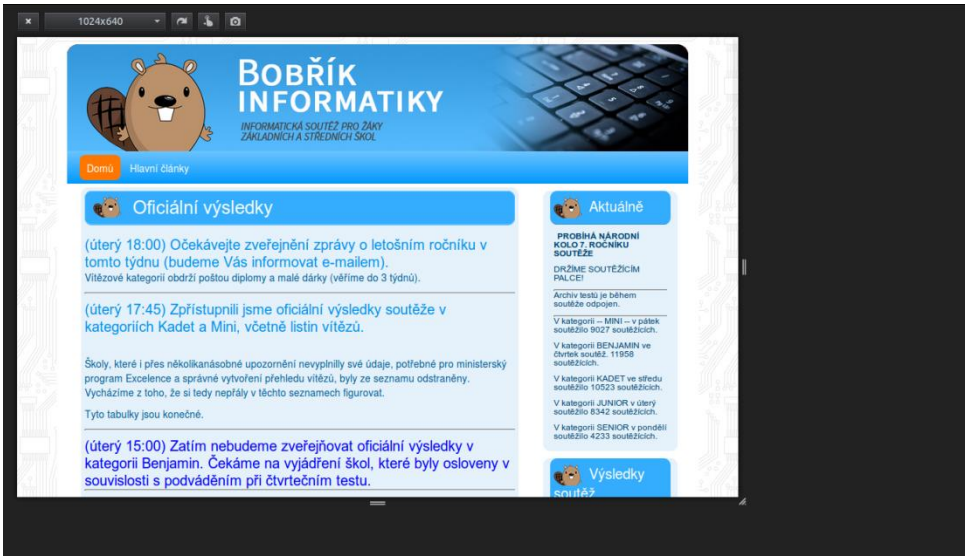
### 3.4. Kontrola správného zobrazení responzivního webu

Nyní přichází poslední kapitola, kde se bude řešit celková kontrola a porovnání toho, jak se nám vše zobrazuje a skládá správně. Podíváme se na rozlišení 1024/640 px. Zajímá nás vždy šířka, výška není nikdy důležitá při tvorbě čistě responzivního zobrazení, ale také se řeší. Například abychom nemuseli rolovat celý web, používáme automaticky posuvník, který si posléze ukážeme pod názvem funkce anglicky *back to top* (zpátky nahoru). Dále k tomu rozlišení. Je to standardní rozlišení, které je v dnešní době již menší. To nemění nic na tom, že to je stále rozlišení velmi často používané. Pojďme se na to podívat.

První příklad webu ibobr.cz a jeho neresponzivní šablony:




**Obrázek 9 - Screen6 [Zdroj: ibobr.cz]**




**Obrázek 10 - Screen7 [Zdroj: ibobr.cz]**

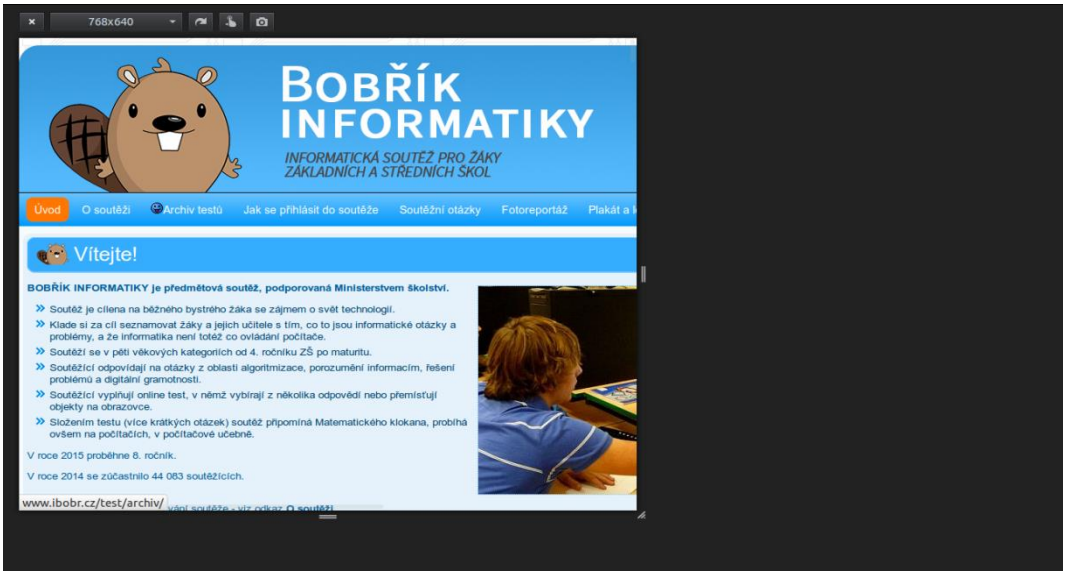
Na druhém příkladu stejného rozlišení vidíme, jak responzivní šablona je schopná reagovat a přizpůsobit se krásně aktuálnímu rozlišení 1024/640.

Vidíme, jak se nám krásně skládá naše nová šablona oproti staré. Vidíme i to, kolik je možné získat místa a ještě navíc grid layoutu je shodný pro výchozí rozlišení. Jenže to právě je správně určeno jen pro výchozí rozlišení. V responzivním webu ani celkově není možné zobrazovat pořád stejnou šířku a je to i logické. Je to jako když by si chtěl někdo na rozlišení 1024/768 monitoru zobrazit obrázek s rozlišením 1280/720 v jeho vlastním rozlišením. Jaký by byl výsledek? Jednoduše by také nebyl vidět celý tak, jako aktuální šablona z ibobr.cz.

Jediným řešením jak tohle obejít bez responzivního webu, je použít posuvník.

Vidíme to ostatně na obrázku z ibobr.cz. Dále porovnáváme při dalším zmenšování rozlišení. A to 768/640.Toto je takové střední zobrazení vhodné především pro tablety.

Zde na ibobr.cz vidíme, že web již nezobrazuje v této fázi vůbec pravý sloupec. Menu částečně tak, jako zbytek celého layoutu. Jakoby někdo web jednoduše úplně uřízl. Což je nepoužitelné, standardně v tomto rozlišení, maximálně opět použít posuvník. Je to vlastně jediná možnost, jak tuto stránku zobrazit. V našem druhém případě je to ale jinak díky elastické šířce. Podíváme se na to v následujícím obrázku, číslo 10.




**Obrázek 11- Screen8 [Zdroj: ibobr.cz]**


**Obrázek 12 - Screen9 [Zdroj: ibobr.cz]**

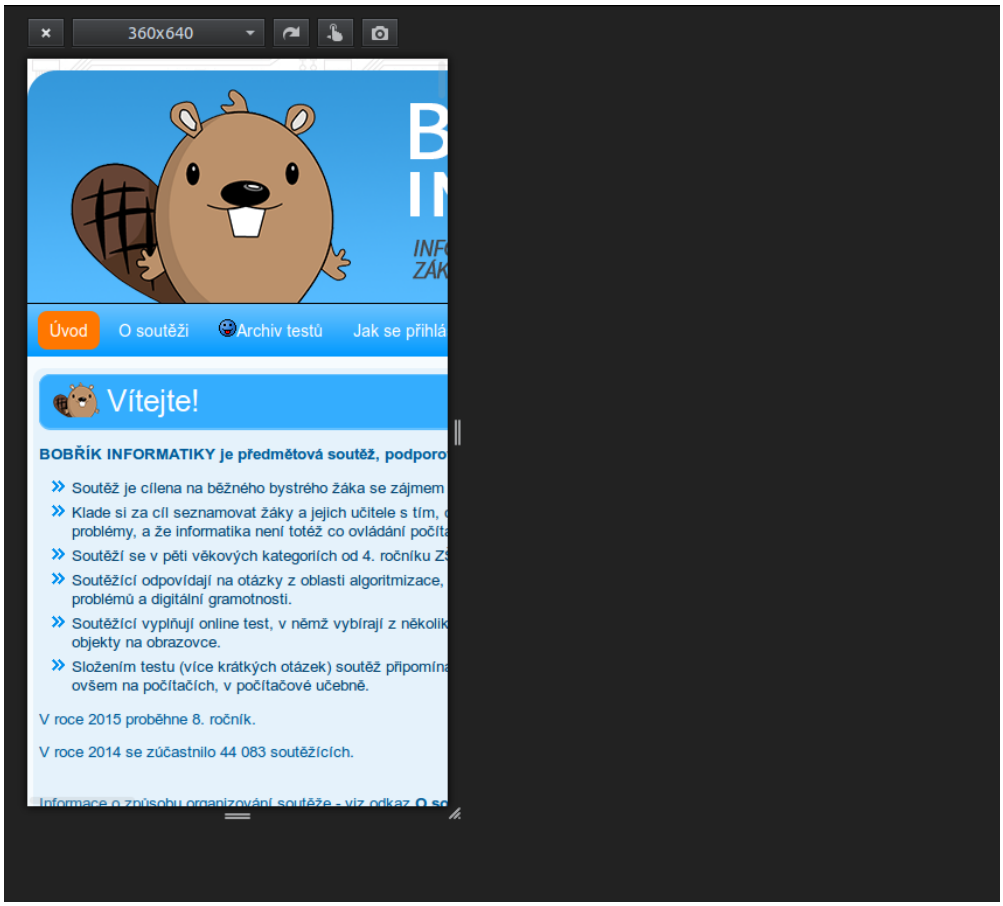
Na druhém obrázku vidíme, jak naše nová responzivní šablona to opět zvládá zcela bez problému, tak aby to fungovalo správně v tom daném rozlišení. Vidíme, že náš responzivní layout se přizpůsobil. Na rozdíl od rozlišení 1024/640 pixelů vidíme, že se nám již standardně nezobrazuje pravý sloupec kvůli právě snížení rozlišení. Responzivní šablona je schopna na základě tohoto vyhodnocení zařadit

správně sloupec dolů. Tedy vše vlastně do jednoho většího sloupce. A to je jasně nejlepší řešení. Protože nemůžeme např. sloupec, nebo jiný element už logicky se zmenšováním rozlišení zobrazovat tam, kde se zobrazoval původně.

Koukněme, jak vypadá dole po odrolování směrem dolů.

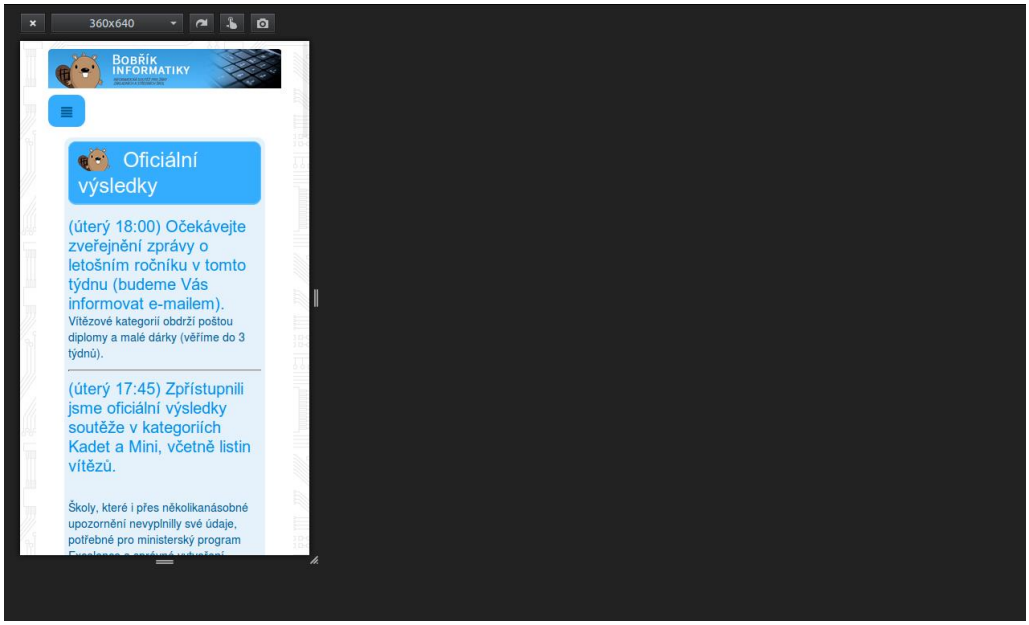

**Obrázek 13 - Screen10 [Zdroj: ibobr.cz]**

Vidíme, že máme všechno v jednom hlavním sloupci. Je to praktické v těchto nižších rozlišeních, protože např. na tabletu to uživatelé určitě ocení. Bez problému se to dá pak používat. Teď se podíváme na poslední rozlišení, které rozebereme. Je to rozlišení 360/640 pixelů. První opět z aktuální neresponzivní šablony ibobr.cz.



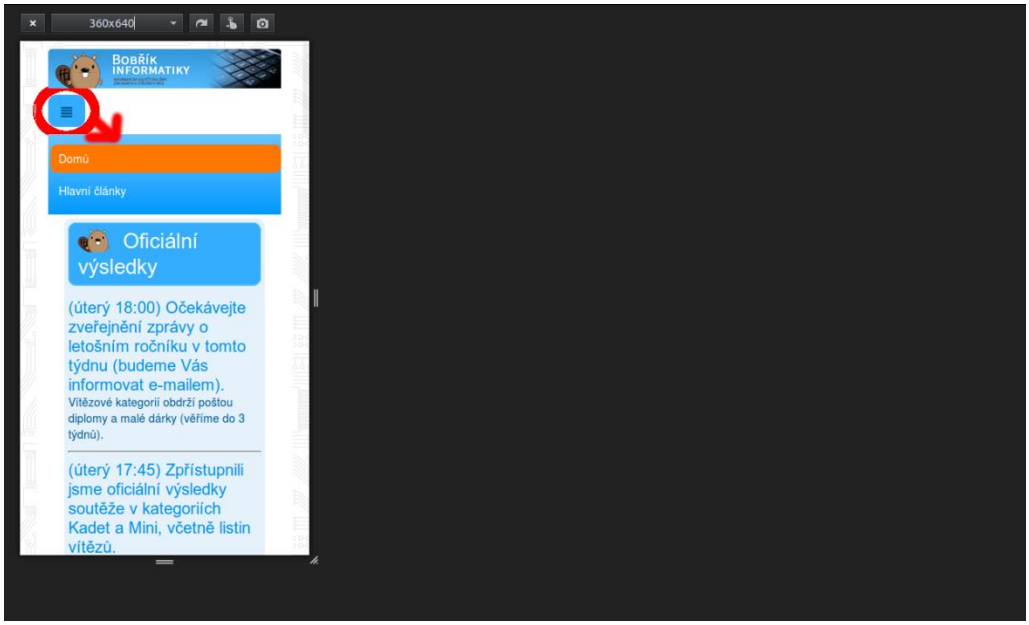

**Obrázek 14 - Screen11 [Zdroj: ibobr.cz]**

Vidíme, že již není v pořádku, co se týká v podstatě nic. Text je absolutně nečitelný, jelikož je vše oříznuto. To je právě fixní šířka. Web nereaguje elasticky a na telefonu s tímto rozlišením je to nepoužitelné. Jak jsem již uváděl, jediná cesta je pak využít posuvníku. Jiná možnost, jak si třeba přečíst text, nebo podobně, není možná. Na nové šabloně je všechno jinak. Pojďme se podívat jak.




**Obrázek 15 - Screen12 [Zdroj: ibobr.cz]**

Vidíme, že vše funguje jak má. Vše se zmenšeno v poměru, abychom to mohli na telefonu bez problému a prakticky použít. Vidíme, že i menu se již nezobrazuje standardně, ale funguje na jeden klik. Tak, aby nám to nezabralo příliš šířky a výšky. Je to logické a samozřejmě nejlepší řešení.



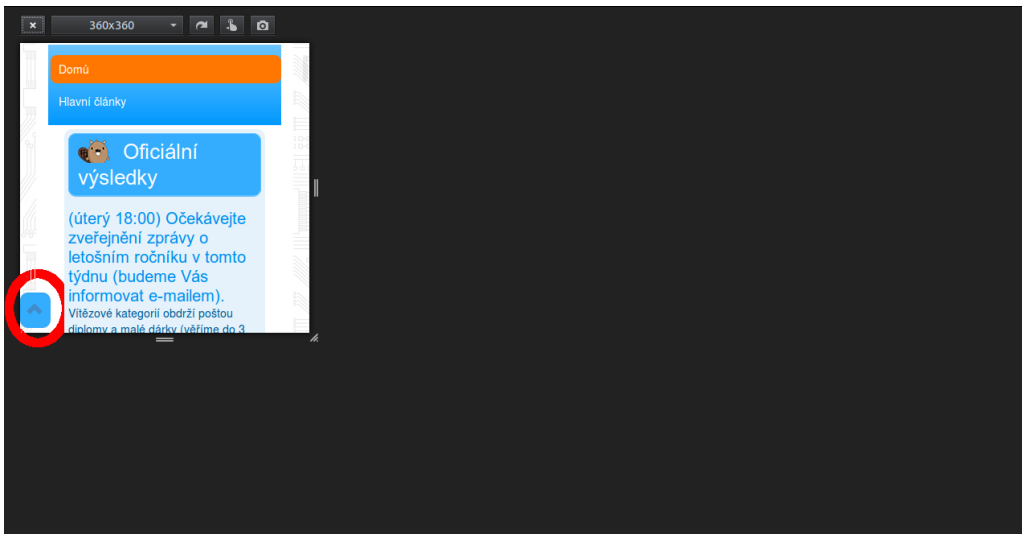

**Obrázek 16 - Screen13 [Zdroj: ibobr.cz]**

V dalším obrázku máme ukázáno nové rozkliknuté menu tak, jak to vlastně vypadá a funguje. Tuto funkce zajišťuje především javascript s návazností na změnu rozlišení css funkce media queries. Tedy změní se rozlišení na méně než 768 pixelů, to jest náš hraniční breakpoint. Toto jsme si vysvětlovali v teoretické části. Pak se tedy předá pokyn javascriptu v našem bootstrapu a menu se nastaví na css vlastnost (property)$_{21}$ display:none;. Dokud ovšem nerozkliknu menu.

Pak se nám zobrazí menu, ovšem upravené. Opět v jednom sloupci tak, aby to bylo co nejrychlejší a nejefektivnější. Vidíme znázorňující šipku u posledního obrázku výše, jak nám ukazuje, co se stane po kliknutí. To stejné platí samozřejmě i po odkliknutí. Menu se nám opět schová, ať nám zbytečně nezabírá prostor.

Jelikož čím nižší rozlišení je, tím musíme více přemýšlet u responzivního webdesignu o tom, abychom zabrali, co nejméně, šířky i výšky. Především šířky, jelikož na výšku tam máme alespoň funkci *back to top*. O tom jsem se již zmiňoval v tom smyslu, že výška není tak důležitá, jako šířka. Dále respektive to tlačítko, které jak anglický název napovídá je vlastně tlačítko, které při rolování webu dolů, většinou kolečkem myši, automaticky pomocí výškového posuvníku se nám po odrolování v naší šabloně, 100 pixelů výšky v našem případě, zobrazí na boku. Ještě podotknu, že toto tlačítko jsem vytvořil sám javascriptem, zmíněným v kapitole 2.2 uvedeným jako příklad.

Vypadá to pak takto:




**Obrázek 17 - Screen14 [Zdroj: ibobr.cz]**

Vidíme opět kroužek, který značí, kde to můžeme najít. Zhruba od této úrovně se nám toto tlačítko začne zobrazovat, protože jsme již odrolovali více než 100 px. Jak jsme si již řekli, proto se nám nezobrazí úplně nahoře. Nemělo by to ani jinak význam. I to se dá občas vidět na některých webech. Jaký je hlavní přínos tohoto tlačítka, je zřejmé. Po kliknutí nahoru se člověk chce pohodlně a rychle dostat nahoru. Bez toho, abychom museli zdlouhavě rolovat pomocí lišty výšku webu. Stačí bohatě kliknout na jedno tlačítko a máme to vyřešeno.

# 4 Cvičný test

Cvičný test se skládá z 5 soutěžních úloh, z kategorií Benjamin a Kadet. Je vytvořený za pomoci formuláře v HTML5 a nativního javascriptu přímo v kategorii Článků. Důležité je zde ukázat, že tento online-test je zakomponovaný přímo v redakčním systému Joomla. Za podpory html a povoleného javascriptu v TinyMCE[26], tzv. WYSIWIG editoru[25], lze vytvořený kód zavést přímo. Test je možné vytvořit tak, že u každé otázky bude na výběr ze čtyř odpovědí a jednoduchým zaškrtnutím se vybere ta správná. Nebo je i možnost vyplněný textového pole. Tento test se vytvoří za použití standardních HTML5 tagů pro vytvoření formuláře.

Vyhodnocování tohoto testu má ovšem na starosti javascript, který zde pro představu zveřejním.

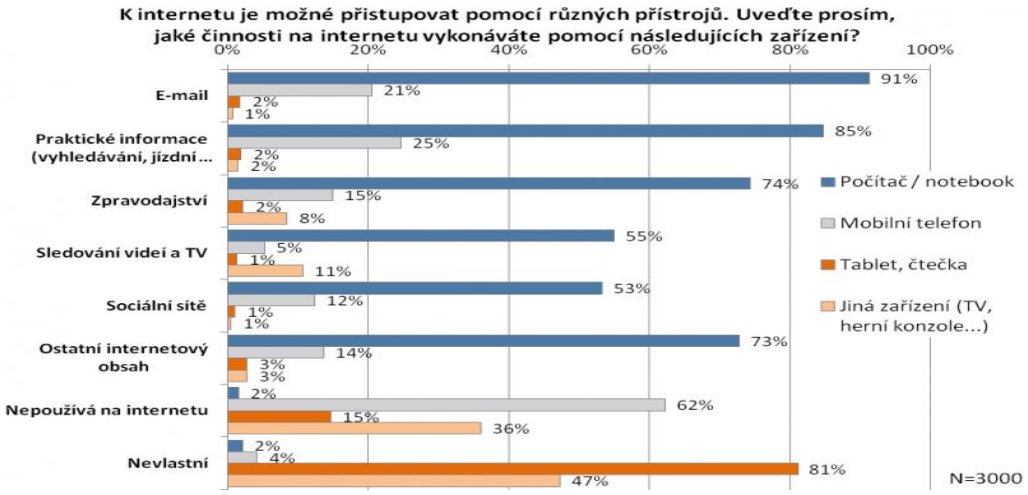
```
<![CDATA[
var odpovedi = new Array(0,0,0,0,0);


function vychozi() {
   for ( var i = 0; i < document.forms[0].elements.length;
i++ ) {
        document.forms[0].elements[i].checked = false;
   }
}
function odpoved(otazka,bodovani) {
   odpovedi[otazka-1] = bodovani;
}
function ukaz_hodnoceni() {
   var pocet_bodu = 0;
   for(var i=0; i<5; i++){
```

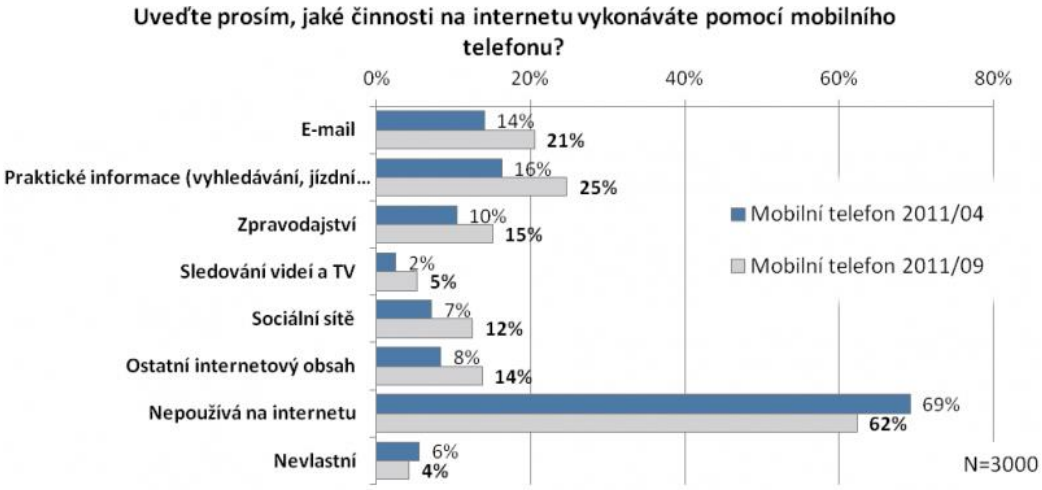
```
          pocet_bodu = pocet_bodu + parseInt(odpovedi[i]);
          if (odpovedi[i] == 0) {
               document.getElementById('o' + (i+1)).style.color
= 'red';
          }else {
               document.getElementById('o' + (i+1)).style.color
= 'black';
          }
          document.getElementById('dobre' + (i+1)).style.color
= 'green';
          document.getElementById('dobre' +
(i+1)).style.fontWeight = 'bolder';
     }
     var hlaska = "Bodový zisk v tomto testu je "+pocet_bodu+"
z 5.\n";
     if (pocet_bodu <5){
          hlaska = hlaska;
     }
     if (pocet_bodu <= 4){
          alert(hlaska);
     }
     else {
          alert(hlaska+" Gratulujeme!");
     }}
  // ]]
```

# 5 Grafy

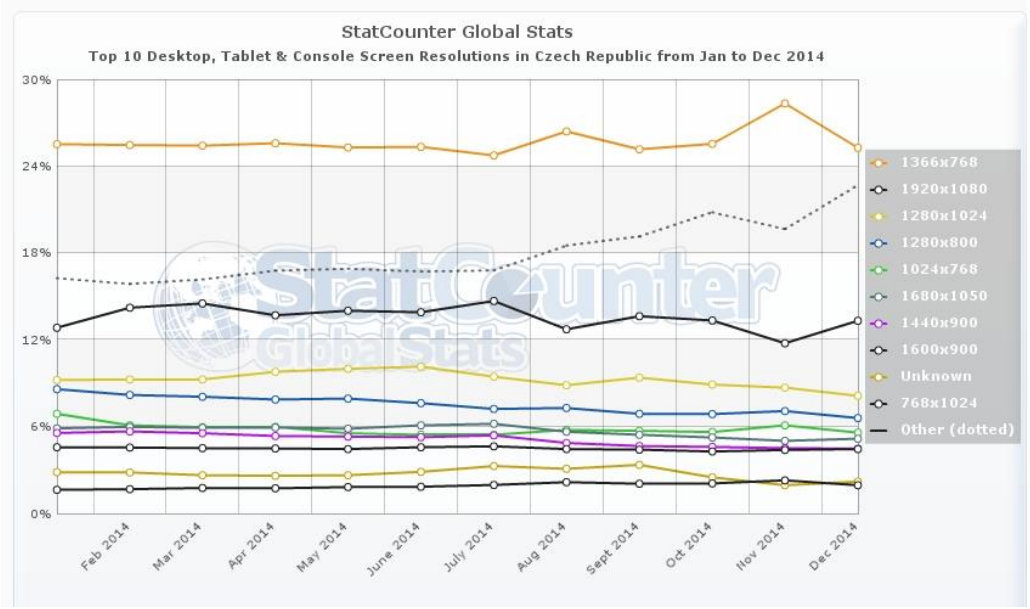
Obrázkové grafy, spíše pro zajímavost, ukazují, že naše téma je stále víc než aktuální. Jak jsem zmiňoval již v úvodu.




**Obrázek 18 - Statistika přístupu různých zařízení na web [Zdroj: netmonitor.cz]**




**Obrázek 19 - Statistika používání mobilních telefonů [Zdroj: netmonitor.cz]**




**Obrázek 20 - Statistika používaných rozlišení [Zdroj: netmonitor.cz]**

# 6 Závěr

## 4.1. Cíle práce a poučení

Hlavním cílem práce bylo, jak je již zmíněno v úvodu této bakalářské práce, vytvořit responzivní, jinak řečeno mobilní verzi webu ibobr.cz. Toto se úspěšně podařilo realizovat v teoretické a zároveň v praktické rovině. Tato má bakalářská práce objasňuje celý vznik responzivity a hlavně její funkčnost. Dále zde představuje základní stavební kameny responzivní problematiky a jak využíváme použitý framework bootstrap.

Teoretická a praktická část této práce objasňuje vše, co se týká responzivního webu. Co z toho plyne? Veškerá teorie je přímo zapsána v teoretické části. Zato praktická část se skládá také z přímo zapsané praktické části. Ovšem i s funkčním výsledkem hotové šablony, na kterou se velmi často nepřímo odkazuje za pomocí screenshotu. Přímo na této adrese http://ibobr.cz /. Zdrojové soubory jsou samozřejmě přiloženy k této bakalářské práci.

## 4.2. Výsledky celé práce a cvičný test (čerpání článku)

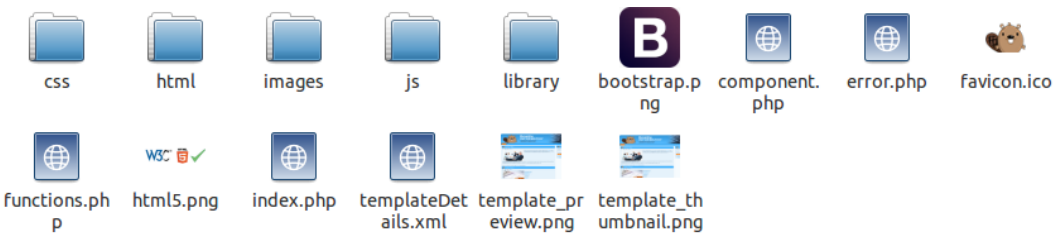
Výsledky celé práce jsou jednak vypracovaná a sepsaná bakalářská práce s jednotlivými kapitolami a přepracovaná šablona, tak i funkční test. Tomuto tématu jsem se nevěnoval, jelikož to není záležitost responzivního zobrazení, tedy téma této bakalářské práce.

Jen řeknu, že cvičný test je test vědomostní a skládá se z 5 otázek. Samozřejmě to není přímo vědomostní test, jelikož všechny odpovědi jsou již dopředu označeny zeleně. Tento test, tedy není soutěžní, ale jen poukazuje, jak funguje takový jednoduchý online test jen za pomoci nativního javascriptu a html5, jak bylo v zadání.

Důležité je ukázat, že tento online-test je zakomponovaný v redakčním systému Joomla, konkrétně přímo v článku. Za podpory html a povoleného javascriptu v TinyMCE[26], tzv. WYSIWIG editoru[25], lze vytvořený kód zavést přímo. Díky předpřipravené struktuře php v naší šabloně se obsah automaticky načte. To vše nám zcela usnadní Joomla. Ale co je důležitější, v responzivním webdesignu, který

jsme si již připravili, se nám automaticky načte do daného elementu, který již máme připravený a který nám podporuje responzivní zobrazení. Tím si ušetříme spoustu práce. Konkrétně v hlavním elementu gridu, třídy s názvem col-md-9 content. Což značí naše bootstrap elementy, které se právě díky tomuto označení o vše již postarají. Máme je již nadefinované předem a využíváme pak tento element na více místech pro různý obsah.

Veškeré soubory jsou přiloženy do společné složky s touto bakalářskou prací. Obsah složky root naší šablony je ukázán čistě pro ilustraci na obrázku zde:




**Obrázek 21- Screen15 [Zdroj: ibobr.cz]**

V přiložených souborech si celou strukturu můžeme projít detailně i s podsložkami.

Závěry si může každý udělat sám. Vše jsem vypracoval a vysvětlil co možná nejjednodušším způsobem, jak to jen bylo možné.

Popsal jsem základní jednoduché postupy ve 2 velkých kapitolách a jeho podkapitolách. Nebyly zde ale popsány určitě všechny skutečnosti, jelikož by to bylo na několik bakalářských prací a minulo by se to účinkem. Některé z těchto témat zasahují např. do oblastí spíše čistě programování a skriptování, což nebylo zadáním této práce. Z toho plyne, že nenastal žádný závažný problém.

Vše se podařilo dotáhnout úspěšně do konce. Vytvořila se nová šablona, která umí responzivní design a naprosto se přizpůsobuje displejům zařízení s nižším rozlišením, především telefonů a tabletů do maximální přípustné šířky 320 pixelů.

Nová šablona se jmenuje ibobr_responzivni. Tímto můžeme ukončit poslední kapitolu této práce. Ještě bude pár grafů používaných rozlišení a používání mobilních zařízení na závěr v příloze grafy. Ať máme taky k dispozici nějaká reálná data. Pak si již jen uvedeme citace z Wikipedie.

Děkuji za pozornost a za přečtení mé bakalářské práce. Snad vám pomohla lépe pochopit základní problematiku responzivního designu. Za všechny případné nedostatky, které se jistě vždy nějaké najdou, se předem upřímně omlouvám! Snažil jsem se určitě toto riziko minimalizovat.

# 7 Seznam využitých zdrojů a literatury


[1] Přispěvatelé Wikipedie, *Responzivní web design* [online], Wikipedie: Otevřená encyklopedie, c2015, Datum poslední revize 13. 01. 2015, 09:52 UTC, [citováno 28. 01. 2015] <http://cs.wikipedia.org/w/index.php?title=Responzivn%C3%AD_web_design&oldid=12152672>

[2] Přispěvatelé Wikipedie, *Kaskádové styly* [online], Wikipedie: Otevřená encyklopedie, c2014, Datum poslední revize 6. 10. 2014, 14:54 UTC, [citováno 28. 01. 2015] <http://cs.wikipedia.org/w/index.php?title=Kask%C3%A1dov%C3%A9_styly&oldid=11904468>

[3] Přispěvatelé Wikipedie, *JavaScript* [online], Wikipedie: Otevřená encyklopedie, c2014, Datum poslední revize 26. 08. 2014, 10:59 UTC, [citováno 28. 01. 2015] <http://cs.wikipedia.org/w/index.php?title=JavaScript&oldid=11776954>

[4] Přispěvatelé Wikipedie, *JQuery* [online], Wikipedie: Otevřená encyklopedie, c2014, Datum poslední revize 7. 08. 2014, 11:40 UTC, [citováno 28. 01. 2015] <http://cs.wikipedia.org/w/index.php?title=JQuery&oldid=11723755>

[5] Přispěvatelé Wikipedie, *CSS3* [online], Wikipedie: Otevřená encyklopedie, c2014, Datum poslední revize 14. 05. 2014, 16:23 UTC, [citováno 28. 01. 2015] <http://cs.wikipedia.org/w/index.php?title=CSS3&oldid=11459001>

[7] Přispěvatelé Wikipedie, *Media Queries* [online], Wikipedie: Otevřená encyklopedie, c2013, Datum poslední revize 26. 08. 2013, 07:39 UTC, [citováno 28. 01. 2015] <http://cs.wikipedia.org/w/index.php?title=Media_Queries&oldid=10669853>

[8] Přispěvatelé Wikipedie, *Responzivní web design* [online], Wikipedie: Otevřená encyklopedie, c2015, Datum poslední revize 13. 01. 2015, 09:52 UTC, [citováno 28. 01. 2015] <http://cs.wikipedia.org/w/index.php?title=Responzivn%C3%AD_web_design&oldid=12152672>

[9] Přispěvatelé Wikipedie, *Framework* [online], Wikipedie: Otevřená encyklopedie, c2014, Datum poslední revize 26. 08. 2014, 20:30 UTC, [citováno 28. 01. 2015] <http://cs.wikipedia.org/w/index.php?title=Framework&oldid=11780814>

[10] Přispěvatelé Wikipedie, *Twitter Bootstrap* [online], Wikipedie: Otevřená encyklopedie, c2014, Datum poslední revize 12. 11. 2014, 14:34 UTC, [citováno 28. 01. 2015] <http://cs.wikipedia.org/w/index.php?title=Twitter_Bootstrap&oldid=12002038>

[11] Přispěvatelé Wikipedie, *Joomla!* [online], Wikipedie: Otevřená encyklopedie, c2014, Datum poslední revize 13. 07. 2014, 14:30 UTC, [citováno 28. 01. 2015] <http://cs.wikipedia.org/w/index.php?title=Joomla!&oldid=11669183>

[12] ELIZABETH, Castro a Hyslop BRUCE. *HTML5 & CSS3 Visual QuickStart Guide*. ISBN 0321719611.

[13] *MALÝ, Martin. Mobilizujeme web v HTML5 [online]. [cit. 2013-03-10]. Dostupné z: http://www.zdrojak.cz/clanky/mobilizujeme-web-v-html5/*

[14] *Add Finger-Swipe Support to Webpages* [online]. [cit. 2013-03-10]. Dostupné z: http://padilicious.com/code/touchevents/

[15] *Making Websites Mobile Friendly* [online]. [cit. 2013-03-10]. Dostupné z: http://googlewebmastercentral.blogspot.cz/2011/02/making-websites-mobile-friendly.html

[16] VANÍČEK, J. Bobřík informatiky, výběr úloh z národních kol soutěže 2008 a 2009. Praha: Výzkumný ústav pedagogický, 2009. 32 s. ISBN 80-87000-26-7

[17] HRUŠECKÁ, A., PEKÁROVÁ, A., TOMCSÁNYI, P., TOMCSÁNYIOVÁ, M. Informatický bobor - nová súťaž v informačných technológiách pre žiakov základných a stredných škôl. [online] Portál *Česká škola*, 21. 4. 2008. Dostupné z www <http://www.ceskaskola.cz/ICTveskole/Ar.asp?ARI=104994&CAI=2129]

[18] *Informacinių technologijų konkursas „BEBRAS"* [online]. Vilnius: Matematikos ir informatikos insitutas [cit. 20. 11. 2008]. Dostupné z www <http://www.bebras.org>

[19] TOMCSÁNYI, P., VANÍČEK, J. Implementation of informatics contest Bebras in Czechia and Slovakia. In Mechlová, E. , Valchař, A. (eds.): Information and communication technology in education ´09. Ostrava: Ostravská univerzita, 2008. s. 214 - 218. ISBN 80-7368-459-4

# 8 Seznam obrázků

# 9 Seznam zkratek v (EN)

UX[0] - User experience

HTML[1] - HyperText Markup Language

HTML5[2] - HyperText Markup Language Version 5

CSS [3] - Cascading Style Sheets

CSS 3[4] - Cascading Style Sheets Version 3

JS[5] - Javascript (Objected oriented script language)

Jquery[6] - Javascript Library

API [7] - Application Programming Interface

*PHP* [8] - *Hypertext Preprocessor*(Programming language)

CMS[9] - Content Management System

# 10 Vysvětlivky odborné termíny

Breakpoint[10] - zlomový bod (media queries)

Media queries[11] - celá funkce většinou několika breakpointů v css

Html tag[12] – např.: 

Element[13] - html tag

Selektor[14] - aktuálně vybraný html element

Content[15] - obsah daného elementu, který uzavírá nějaký html blok html tagem

Layout[16] - celý html obsah elementů a jeho rozložení stránky všech contentů v px či procentech nejčastěji

Plugin[17] - zásuvný modul

Open-source[18] - volně šiřitelná licence, příklad: GPL

Debugging[19] - ladění kódu

Joomla[20] - open-source redakční systém php

Property[21] - vlastnost

Root[22] - hlavní adresářová složka

Framework[23] - zjednodušeně typ knihovny

Bootstrap[24] - front-end knihovna

WYSIWIG editor[25] typ html editorů

TinyMCE[26] - známy wysiwig editor, používaný např: v Joomla

Syntaxe[27] - způsob(stylopis) psaní daného programovacího jazyka, s jasně daným souborem pravidel

Pre-processor[28] - umožňuje vlastní syntaxi, odlišnou syntaxi od. css, liší se touto koncovkou, podobně jako javascript, se zpět kompiluje do css

# 11 Přílohy

- CD
  - ibobr_responzivni.zip
  - BP_Certik.pdf