

**Ročníková práce**

Tvorba vlastního frameworku

Jan, Prokop, C3

Konzultant: Lukáš Zelenka

Školní rok: 2015 / 2016

**Čestné prohlášení**

Prohlašuji, že jsem předloženou ročníkovou práci vypracoval zcela samostatně a uvádím v ní veškeré prameny, které jsem použil. Všechny převzaté informace jsem řádně ocitoval.

V Rakovníku, dne

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (podpis studenta).

# Anotace

Tématem mé ročníkové práce je Framework, zde budu rozebírat jeho vytvoření, jeho části a jeho využití v praxi.

Tato práce má sloužit jako návod a vysvětlení, co je to Framework, kde a jak ho použít a kdo Framework využije.

Seznámím vás s rozložením mého Frameworku, co obsahuje a jak to využít.

# Klíčová slova

Framework, Grid systém, CSS, HTML, Web,

Obsah

[1 Anotace 3](#_Toc442107216)

[2 Klíčová slova 3](#_Toc442107217)

[3 Úvod 5](#_Toc442107218)

[4 Metodika 5](#_Toc442107219)

[5 Co je to Framework 5](#_Toc442107220)

[5.1 Příklady frameworku 5](#_Toc442107221)

[6 Grid systém 6](#_Toc442107222)

[6.1 Pravidla grid sytému 6](#_Toc442107223)

[6.2 Možnosti grid systému 6](#_Toc442107224)

[6.3 Příklad grid sysému 6](#_Toc442107225)

[6.4 Odsazení sloupců v containeru 7](#_Toc442107226)

[6.5 Zarovnání sloupců v containeru 8](#_Toc442107227)

[7 Formuláře 8](#_Toc442107228)

[7.1 Formulář v řádce 9](#_Toc442107229)

[7.2 Vodorovný formulář 10](#_Toc442107230)

[7.3 Základní textové pole 11](#_Toc442107231)

[7.4 Textarea 11](#_Toc442107232)

[7.5 Zaškrtávací políčka a volba možností 11](#_Toc442107233)

[7.6 Zaškrtávací políčka a volba možností v řádce 12](#_Toc442107234)

[8 Tabulky 13](#_Toc442107235)

[8.1 Ohraničená tabulka 13](#_Toc442107236)

[8.2 Pomocné třídy u tabulek 13](#_Toc442107237)

# Úvod

V mé ročníkové práci se budu zabývat tvorbou a následným použitím vlastního frameworku.

Tato práce má sloužit jako následný manuál a vysvětlení různých prvků daného frameworku, aby měl uživatel používání tohoto frameworku co nejjednodušší a nemusel sám hledat, co daný element dělá a jaké jsou jeho možnosti.

# Metodika

Nejdříve jsem začal s tvorbou vlastního kódu CSS pomocí programu PSPad, kde jsem začal tvořit grid systém, který obsahuje containery a sloupce, kterým jsem musel předdefinovat jejich šířku a veškeré vlastnosti.

Po vytvoření celého grid systému jsem začal s tvorbou formulářů. Jednotlivým položkám formuláře byly nastaveny výchozí hodnoty, které si dále uživatel může upravit.

Následně jsem nastavil hodnoty tabulkám a obrázkům a tím dokončil celý kód.

Pro kód jsem použil své znalosti z předmětu web design a také jsem se inspiroval v již některých vytvořených frameworcích.

Data a informace pro vytvoření celé práce jsem hledal na různých webových stránkách a také za pomoci svého konzultanta Lukáše Zelenky, který mi poradil, kde naleznu ty nejlepší informace.

Tento postup jsem si vybral, protože pro mě bylo nejjednodušší vytvořit kód a poté pro něj napsat manuál a samotnou ročníkovou práci.

# Co je to Framework

Framework neboli aplikační rámec je softwarová struktura a šablona sloužící k usnadnění práce uživateli a zjednodušení práce při různých projektech.

Cílem frameworku je co nejvíce usnadnit uživateli svou práci, aby se mohl soustředit pouze na svou práci.

Někdy se můžeme setkat s názory, že použití frameworku je špatné a že často uživatel musí danž Framework důkladně nastudovat, tak to ale není. Při jednorázovém použití tomu tak může být, ale když daný Framework používáte opakovaně, je to opravdu veliké usnadnění.

## Příklady frameworku

J query

Bootstrap

Foundation Framework

Yahoo Pure CSS

A spousta dalších.

# Grid systém

Grid systém se využívá při tvorbě rozložení webových stránek pomocí sloupců a řádků.

Řádky mohou být umístěny do container-fluid (využívá plnou šíři stránky), nebo do container(využívá pevně danou šířku ze stránky).

Obsah by měl být umístěn správně do sloupců.

## Pravidla grid sytému

„Obsah by měl být umístěn uvnitř sloupce, a to pouze sloupce mohou být přímé potomky řádků“

„Řádky musí být umístěny do container-fluid (plná šířka), nebo do container(pevně daná šířka). Pro správné zarovnání a odsazení.“

„Předdefinované třídy, jako je .row a .col-4 jsou k dispozici pro rychlé vytváření rozvržení mřížky.“

„Sloupce jsou vytvořeny zadáním počtu 12 dostupných sloupců, které chcete span. Například, tři stejné sloupce by používal tříd .col-4.“

## Možnosti grid systému

## Příklad grid sysému

Použitím třídy .col-1 rozdělíme šířku stránky na dvanáct různých sloupců, přičemž musíme dvanáctkrát napsat tuto třídu.

**HTML:**

<div class=“row“>

<div class=“col-1“>.col-1</div>

<div class=“col-1“>.col-1</div>

<div class=“col-1“>.col-1</div>

<div class=“col-1“>.col-1</div>

<div class=“col-1“>.col-1</div>

<div class=“col-1“>.col-1</div>

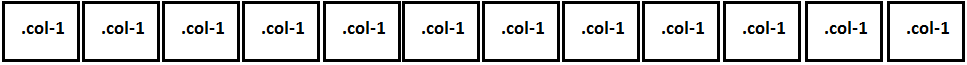
<div class=“col-1“>.col-1</div>

<div class=“col-1“>.col-1</div>

<div class=“col-1“>.col-1</div>

<div class=“col-1“>.col-1</div>

<div class=“col-1“>.col-1</div>

<div class=“col-1“>.col-1</div>

Pokud chceme stránku rozdělit pouze na dva sloupce různé šířky, použijeme k tomu jen dvě třídy a to : .col-9 + .col-3.

**HTML:**

<div class="container">

<div class="col-9">

<h1>.col-9</h1>

</div>

<div class="col-3">

<h1>.col-3</h1>

</div>

</div>



Kdybychom chtěli stránku rozdělit na dva sloupce stejné šířky použijeme k tomu dvě třídy a to : .col-6 a .col-6.

**HTML:**

<div class=“row“</div>

<div class=“col-6“>.col-6</div>

<div class=“col-6>.col-6</div>

</div>



## Odsazení sloupců v containeru

Pro odsazení sloupců v containeru používáme třídu .col-offset a číslo o kolik chceme daný sloupec odsadit. Příklad .col-offset-3 posune sloupec .col-3 o 3 sloupce do prava.

**HTML:**

<div class="container">

<div class="row">

<div class="col-3">.col-3</div>

<div class="col-3 col-offset-3">.col-3 col-offset-3</div>

</div>

</div>

****

Pro odsazení jednoho samotného sloupce přímo do středu použijeme následující kód. .col-6 způsobí že bude jeden sloupec a .col-offset-3 se postarají o odsazení z leva i z prava o 3 sloupce, tudíž náš sloupec je zarovnán na středu containeru.

**HTML:**

<div class="container">

<div class="row">

<div class="col-6 col-offset-3">.col-6 .col-offset-3</div>

</div>

</div>

## Zarovnání sloupců v containeru

Sloupce v containeru můžeme zarovnat dvěma způsoby, jedním z nich je, že sloupce posuneme do prava a k tomu použijeme .col-\*-pull-right. A druhý způsob je, že zarovnáme sloupce do leva a k tomu použijeme .col-\*-push-left.

**HTML:**

<div class="container">

<div class="row">

<div class="col-9 push-left">.col-9 push-left</div>

<div class="col-3 pull-right">.col-3 pull-right</div>

</div>

****</div>

# Formuláře

Formuláře a jejich základní funkce v mém frameworku jsou nastaveny na celou šířku stránky a samy se upravují podle šířky monitoru.

Ve formulářích nalezneme základní textová pole, která mohou mít různé typy, např.: text, heslo, textarea atd..

**Příklad formuláře:**

**HTML:**

<form action="submit.php" method="post">

<input required="required" pattern=".{1,} .{1,}" type="text" name="name" placeholder="Jméno Příjmení"><br>

<input required="required" type="email" name="email" placeholder="vas@email.cz"><br>

<input required="required" pattern=".{4,32}" type="text" name="subject" placeholder="Předmět"><br>

<textarea name="zprava" placeholder="Vaše zpráva ..."></textarea><br>

<input type="submit" class="odeslat pull-right" value="Odeslat">

</form>

## Formulář v řádce

Přidáním třídy .form-inline do našeho formuláře, zajistíme, že položky formuláře budou v jedné rovině.

Klasický formulář má na svých položkách nastavenou plnou šířku 100%, ale inline formuláře mají nastaveno auto, aby položky ve formuláři mohly být vedle sebe.

**HTML:**

<form class="form-inline">

<div>

<label for="Jméno">Jméno</label>

<input type="text" class="form-control" id="Jmeno2" placeholder="Vaše Jméno">

</div>

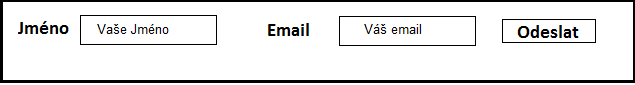
<div>

<label for="Email">Email</label>

<input type="email" class="form-control" id="exampleInputEmail2" placeholder="Váš email">

</div>

<button type="submit" class="btn btn-default">Odeslat</button>

****</form>

## Vodorovný formulář

Použitím předdefinovaného grid systému a použitím předdefinované třídy .form-horizontal, docílíme toho, že formulář bude vodorovný. Použitím třídy .form-group zajistíme, že každá položka je na nové řádce a nemusíme používat třídu .row.

**HTML:**

<form class="form-horizontal">

<div class="form-group">

<label for="inputEmail3" class="col-sm-2 control-label">Email</label>

<div class="col-10">

<input type="email" class="form-control" id="Zadatemail" placeholder="Email">

</div>

</div>

<div class="form-group">

<label for="zadej heslo">Password</label>

<div class="col-sm-10">

<input type="password" class="form-control" id="zadatheslo" placeholder="Heslo">

</div>

</div>

<div class="form-group">

<div>

<div class="checkbox">

<label>

<input type="checkbox"> Pamatovat si mě

</label>

</div>

</div>

</div>

<div class="form-group">

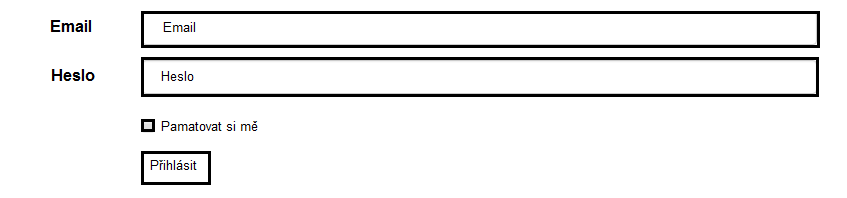
<div>

<button type="submit" class="btn btn-default">Přihlásit</button>

</div>

</div>

</form>



## Základní textové pole

<input type="text" class="form-control" placeholder="Zadejte text">

## Textarea

Položka formuláře, která umožňuje psát více textu na více řádků. Počet řádků se dá manuálně upravit podle potřeby uživatele.

<textarea class="form-control" rows="5"></textarea>

## Zaškrtávací políčka a volba možností

Zaškrtávací políčka slouží pro výběr z jedné nebo více možností. Pokud je u některých zaškrtávacích políček možnost zakázána je nutné přidat třídu .disabled.

**HTML:**

<div class="checkbox">

<label>

<input type="checkbox" value="">

První možnost

</label>

</div>

<div class="checkbox disabled">

<label>

<input type="checkbox" value="" disabled>

Druhá možnost je zablokována

</label>

</div>

<div class="radio">

<label>

<input type="radio" name="volbamoznosti" id="moznost1" value="moznost1" checked>

První možnost

</label>

</div>

<div class="radio">

<label>

<input type="radio" name="volbamoznosti" id="moznost2" value="moznost2">

Druhá možnost

</label>

</div>

<div class="radio disabled">

<label>

<input type="radio" name="volbamoznosti" id="moznost3" value="moznost3" disabled>

Třetí možnost je zablokována

</label>

</div>

## Zaškrtávací políčka a volba možností v řádce

Použitím třídy .checkbox-inline nebo .radio-inline zajistíme, aby zaškrtávací políčka a volba možností byla vedle sebe v řádce.

<label class="checkbox-inline">

<input type="checkbox" id="pole1" value="moznost1"> 1

</label>

<label class="checkbox-inline">

<input type="checkbox" id="pole2" value="moznost2"> 2

</label>

<label class="checkbox-inline">

<input type="checkbox" id="pole3" value="moznost3"> 3

</label>

<label class="radio-inline">

<input type="radio" name="volbavradce" id="volbavradce" value="moznost1"> 1

</label>

<label class="radio-inline">

<input type="radio" name="volbavradce" id="volbavradce2" value="moznost2"> 2

</label>

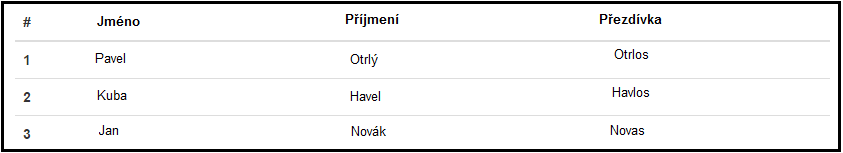
<label class="radio-inline">

<input type="radio" name="volbavradce" id="volbavradce3" value="moznost3"> 3

</label>

# Tabulky

Pro vytvoření tabulky v mém frameworku použijeme klasický tag <table>.

**Příklad klasické tabulky:**

## Ohraničená tabulka

Přidáním třídy .table-bordered docílíme toho, že se na naší tabulky přidá ohraničení ze všech stran.

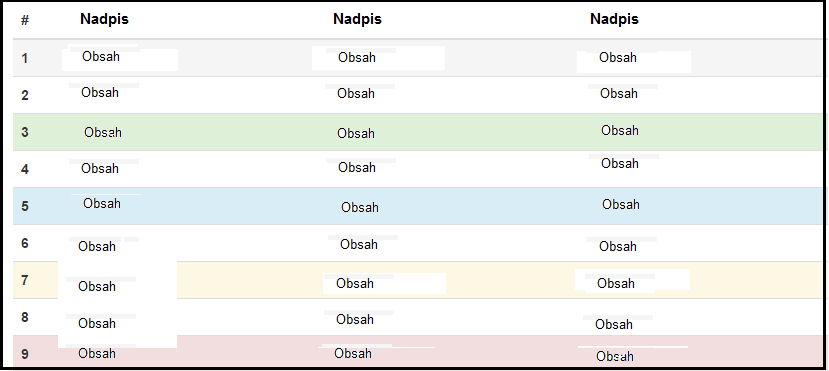
## Pomocné třídy u tabulek

**.active** – použije barvu na konkrétní řádek nebo na konkrétní buňku.

**.succes** – označí úspěšně splněnou buňku nebo řádek.

**.info** – označí nějakou informaci v buňce nebo řádku.

**.warning** – označí buňku nebo řádek s varováním, že je potřeba této buňce nebo řádku věnovat pozornost.

**.danger –** označí buňku nebo řádek, kde by mohlo vzniknout potencionální nebezpečí.

**HTML:**

*<!-- pro řádky -->*

<tr class="active">...</tr>

<tr class="success">...</tr>

<tr class="warning">...</tr>

<tr class="danger">...</tr>

<tr class="info">...</tr>

*<!-- Pro buňky (`td` nebo `th`) -->*

<tr>

<td class="active">...</td>

<td class="success">...</td>

<td class="warning">...</td>

<td class="danger">...</td>

<td class="info">...</td>

</tr>