**Prohlížeč Google Chrome**



Prosím, používejte společně s touto učebnicí prohlížeč Google Chrome. Je to proto, abychom vše viděli stejně.

Pokud Google Chrome nemáte, tak si ho stáhněte zde:

<https://www.google.com/intl/cs_CZ/chrome/>

Instalace trvá pár vteřin. Jen se stáhne soubor a ten otevřete. Nabídka už vás provede instalací.

**1. Nastavení projektu**

**Komentář**

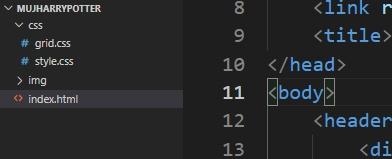
Stáhněte si k sobě do počítače obrázky: <https://drive.google.com/drive/folders/1EqvBVnxqwRlfqLIvaOsjvUFdwWM1CVMo?usp=sharing>

Soubor Grid.css si stáhněte zde (dole vpravo klikněte na Stáhnout pomalu zdarma):

<https://uloz.to/file/RyO9Rv02YeGA/grid-css>

Do složky CSS umístíme style.css (vytvoříme klasicky) a grid.css. Do složky img umístíme všechny stažené obrázky. Index.html vytvoříme klasicky.

Struktura bude vypadat nějak takto.



Do index.html vložíme základní strukturu. Nezapomenem napojit style.css, ale i grid.css. Nezapomínejme, že jsou umístěné ve složce css.

Do style.css vložíme základní kód

**Kód**

**HTML**

<!DOCTYPE html>

<html **lang="cs"**>

<head>

**<meta charset="UTF-8">**

**<link rel="stylesheet" href="css/grid.css">**

**<link rel="stylesheet" href="css/style.css">**

<title>Harry Potter</title>

</head>

<body>

</body>

</html>

**CSS**

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

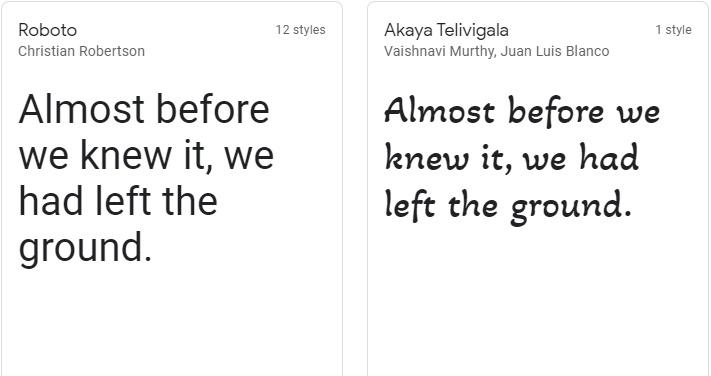
}

**2. Google fonty**

**Komentář**

Google fonty umožňují vložit do našich stránek libovolný font. Bude se načítat přímo z adresy fontu, takže nemusíme řešit, jestli návštěvník naší stránky má font v počítači nebo ne. Pojďme si ukázat příklad na fontu **Akaya Telivigala**, který normálně není podporovaný. My ho ale chceme na našich stránkách zobrazovat.

Na adrese <https://fonts.google.com/> najdeme tento font a rozklikneme.

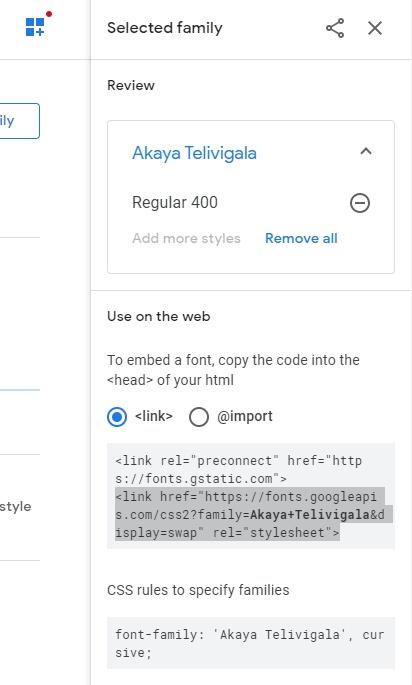


Tento font má jen jednu verzi - Regular 400. Tedy běžně “tučné” písmo. To znamená, že budeme moci používat pro tučnost jen font-weight: 400; (viz základní kurz). Některé fonty mají více tipů - nejčastěji od 100 do 900 (100 = tenké písmo, 900 = tučné písmo). Vždy si můžeme vybrat, které budeme na stránce chtít. My si zde musíme pomocí Select this tyle vybrat jen tuto jednu verzi.



Otevře se po výběru lišta, kterou si také můžeme otevřít pomocí ikonky tří čtverečků a pluska, kterou vidíme na následujícím obrázku vlevo nahoře.

Zde zkopírujeme pouze vyznačenou část, která začíná <link href=”https….



Tuto adresu vložíme do hlavičky naší webové stránky.

<!DOCTYPE html>

<html lang="cs">

<head>

<meta charset="UTF-8">

**<link href="https://fonts.googleapis.com/css2?family=Akaya+Telivigala&display=swap" rel="stylesheet">**

<link rel="stylesheet" href="css/grid.css">

<link rel="stylesheet" href="css/style.css">

<title>Harry Potter</title>

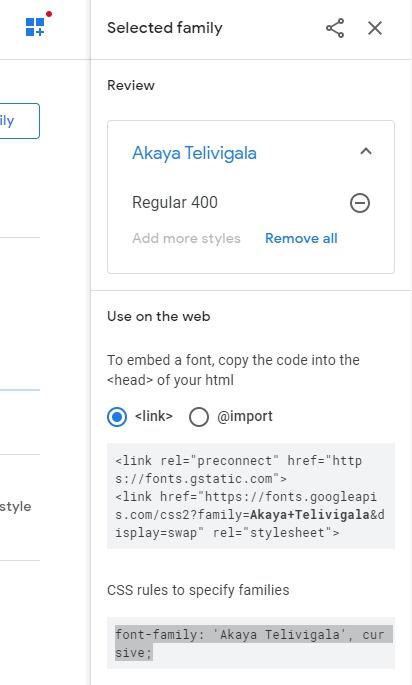
</head>

<body>

</body>

</html>

Nyní můžeme kdekoliv v kódu používat zápis zvýraznění na obrázku. Naše stránka rozeznává font Akaya Telivigala. Můžeme toto font-family nastavit jakémukoli prvku na naší stránce.



Například do HTML můžeme dát

<h1>Nadpis</h1>

Do CSS díky google fontům můžeme napsat tento zápis

h1 {

font-family: 'Akaya Telivigala', cursive;

}

A toto je výsledek. Takto se zobrazí stránka jakémukoliv návštěvníkovi, který na naše stránky přijde.



**Tento kód můžete z HTML a CSS odstranit. Sloužil pouze pro ukázku!!!**

**Úkol**

Z adresy google fontů <https://fonts.google.com/> vložte písmo Montserrat. Vložte jedno tenké, normální a tučné písmo. Vyzkoušejte vložené písmo na nadpisu h1, abyste se přesvědčili, že funguje.

**3. HTML hlavičky**

**Komentář**

Nyní si vytvoříme základní strukturu přímo v body. Bude to struktura hlavičky naší webové stránky.Bude mít část pro logo a část pro další prvky v hlavičce (text + tlačítko).

**Kód**

HTML - takto bude vypadat základní struktura hlavičky

<header class="header">

<div class="logo">

</div>

<div class="header-box">

</div>

</header>

Do části s třídou logo si vložíme obrázek loga.

<div class="logo">

**<img src="img/harrypotterlogo.png" alt="">**

</div>

Do elementu s klásou header-box vložíme nadpis h1, který bude mít dvě části, které obalíme do elementu span.

<div class="header-box">

**<h1 class="heading-main">**

**<span class="heading-main-head">Vítejte v Bradavicích</span>**

**<span class="heading-sub-head">Můžete vstoupit, i pokud jste mudlové</span>**

**</h1>**

</div>

Dále pod H1 vložíme odkaz, který zatím nebude nikam směřovat

<div class="header-box">

<h1 class="heading-main">

<span class="heading-main-head">Vítejte v Bradavicích</span>

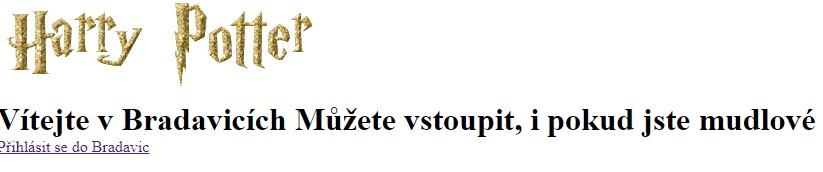
<span class="heading-sub-head">Můžete vstoupit, i pokud jste mudlové</span>

</h1>

**<a href="#" class="btn">Přihlásit se do Bradavic</a>**

</div>

**Jak kód vypadá na stránce**



**4. Základní styly v body**

**Komentář**

Ve style.css začneme úpravou body. Zadáme si libovolný font (Lato nemusíme importovat z google fontů, i když i mezi google fonty je Lato dostupné: <https://fonts.google.com/?query=Lato>). Dále vypneme tučnost a nastavíme velikost písma na 16 pixelů.

**Kód**

**CSS**

body {

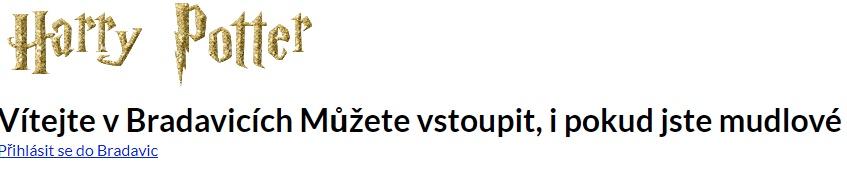
font-family: Lato, sans-serif;

font-weight: 400;

font-size: 16px;

}

**Jak kód vypadá na stránce**



**5. Obrázek na pozadí, jeho stylování a polygon**

**Komentář**

Vložíme do celé hlavičky obrázek na pozadí. Vkládá se pomocí cesty k obrázku a vlastnosti background: url(); Jako první ale musíme nastavit výšku hlavičce. Pokud někdo přijde na stránku, tak ji může vidět různě velikou podle velikosti zařízení (mobily, tablety, notebooky, počítače s monitorem atd.). Pokud chceme zadat univerzální šířku, že to má být vždy 95 % displaye daného zařízení, tak napíšeme 95vh (tedy 95 % viewport height).

**Kód**

**CSS**

.header {

height: 95vh;

background: url(../img/hogwarts-middle.jpg);

}

**Jak kód vypadá na stránce**



**Úkol**

Vytvořte si header a dejte mu výšku 95 % viewportu (height: 95vh) a do pozadí vložte obrázek přes background: url();

Obrázek na pozadí je hodně výrazný, tak ho překryjeme tmavou clonou.

**CSS**

.header {

height: 95vh;

background: **linear-gradient(rgba(0,0,0,0.8),**

**rgba(0,0,0,0.8)),**url(../img/hogwarts-middle.jpg);

}

**Jak kód vypadá na stránce**



Pokud máme obrázek na pozadí, tak doporučuji vždy napsat tyto tři vlastnosti. První zabraňuje opakování obrázku. Pokud obrázek vložíte, tak se automaticky opakuje po ose x i y. Jako kdyby chtěl vždy vyplnit veškerý prostor, který má k dispozici. Pomocí **background-repeat** tomu zabráníme.

Aby obrázek vhodně seděl do naší stránky, tak mu dáme **background-size: cover;**. To znamená, že prohlížeč se bude snažit vecpat co možná největší část obrázku do hlavičky. Jako kdyby se snažil, aby bylo z obrázku vidět co nejvíce.

Pokud prohlížeč budete zmenšovat a zvětšovat, tak je dobré obrázek uchytit za jednu ze stran. Ta bude jakoby fixní a obrázek za ní nebude zajíždět. My zvolíme **background-position: top;** Tedy horní hrana obrázku bude stabilní a fixní.

**CSS**

.header {

height: 95vh;

background: linear-gradient(rgba(0,0,0,0.8),

rgba(0,0,0,0.8)),url(../img/hogwarts-middle.jpg);

**background-repeat: no-repeat;**

**background-size: cover;**

**background-position: top;**

}

**Gradient - tipy a triky**

<https://cssgradient.io/>

**1. tip** - zadejte do gradientu více barev (např. blue, yellow, red)

.header {

height: 95vh;

**background: linear-gradient(blue, yellow, red);**

}



**2. tip** - otočíme gradient o 90 stupňů (můžeme ale o libovolnou hodnotu stupňů)

.header {

height: 95vh;

background: linear-gradient(**90deg**, blue, yellow, red);

}



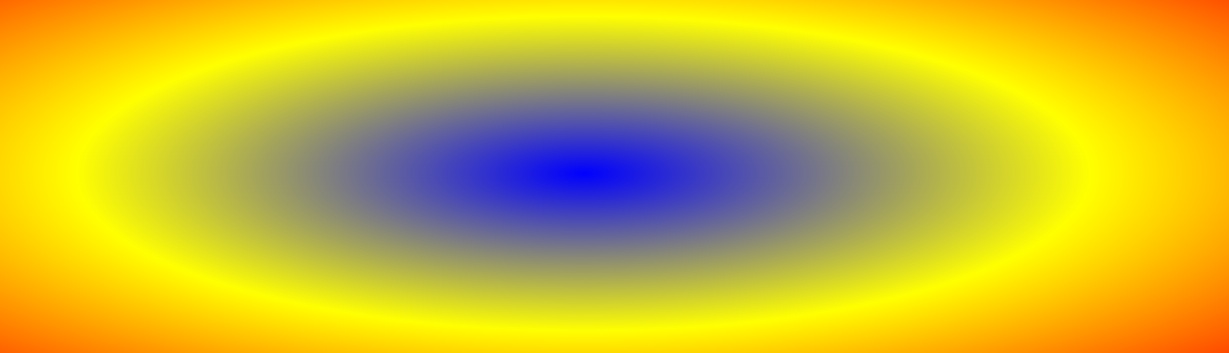
**3. tip** - kruhový gradient

.header {

height: 95vh;

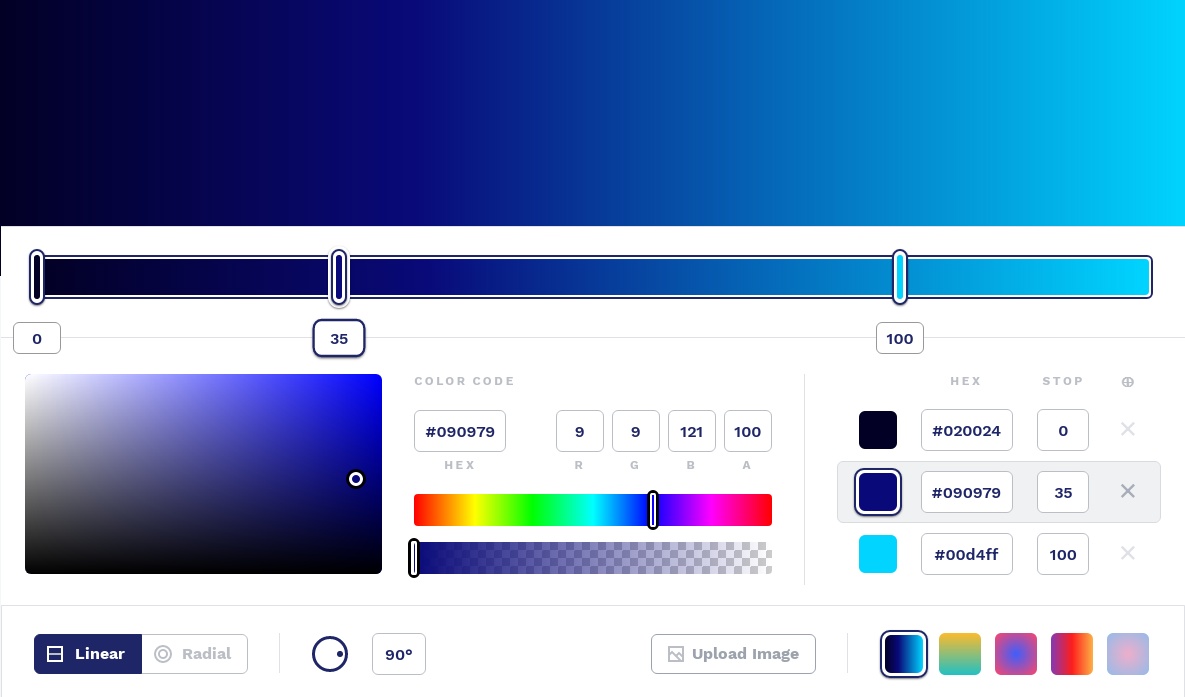
background: radial-gradient(blue, yellow, red);

}



4. tip - na gradienty použijeme nástroj, který vygeneruje kód podle našeho nastavení

<https://cssgradient.io/>

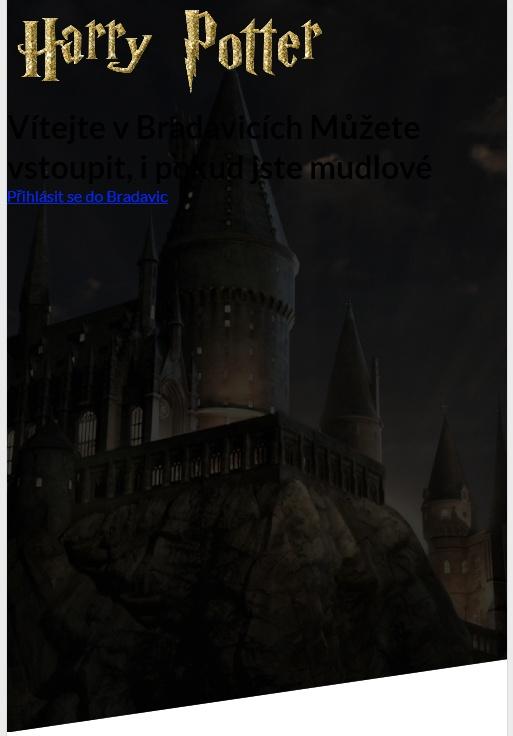


**Úkol**

Fotce na pozadí dejte linear-gradient libovolné barvy a přiřaďte trojici vlastností background-repeat, background-size a background-position.

**Jak kód vypadá na stránce**

Pokud prohlížeč zmenšíme, tak vidíme, že díky 3 uvedeným vlastnostem se pozadí přizpůsobuje. Přibližně takto by stránka vypadala na mobilu.



Co kdybychom chtěli dát třídě header jiný tvar než běžný obdélník? To můžeme pomocí clip-path. Nastavíme, že se bude jednat o mnohoúhelník a zadáme 4 souřadnice. Pro každý roh je jedna. Každý roh má souřadnici X a Y. První je vždy X a druhá je Y. Napište si toto do kódu a hrajete si s čísly v developerské konzoli.

**CSS**

.header {

height: 95vh;

background: linear-gradient(rgba(0,0,0,0.8),

rgba(0,0,0,0.8)),url(../img/hogwarts-middle.jpg);

background-repeat: no-repeat;

background-size: cover;

background-position: top;

**clip-path: polygon(0 0, 100% 0, 100% 90%, 0 100%);**

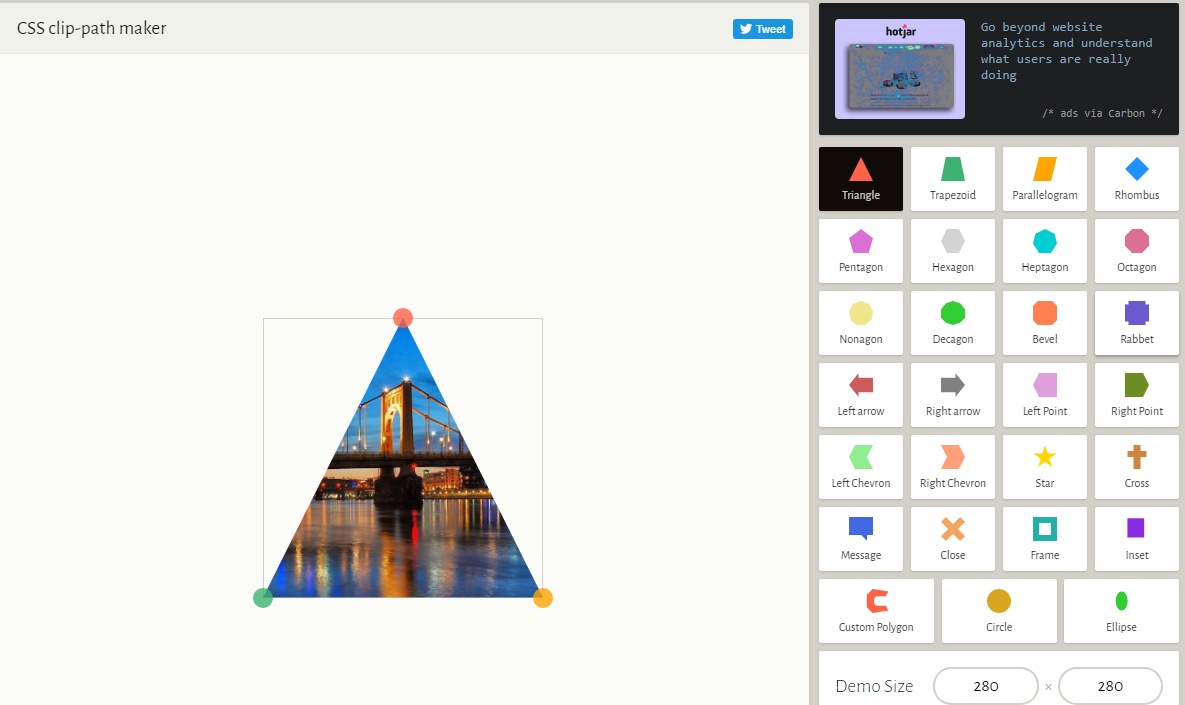
}

**Jak kód vypadá na stránce**



**Tip pro clip-pat**

<https://bennettfeely.com/clippy/>



**Úkol**

Pomocí clip-path si vytvořte podobné ustřižení jako vidíte na obrázku výše (dole ustřižený obrázek)

**6. Logo a jeho zarovnání**

**Komentář**

Pojďme umístit logo. Bude vlevo nahoře. Logo můžeme “vyrvat” z toku pomocí position: absolute; Odsadíme ho odshora 40 pixelů a zleva také 40 pixelů.

Nezapomínejte, že pokud je něco position: absolute, tak se to zarovnává vůči nadřazenému elementu, který je position: relative. Jinak se to zarovná vůči html. Abychom si to ohlídali, tak dáme celému headeru position: relative; aby se logo zarovnávalo vůči hlavičce, protože logo je v hlavičce umístěné a nechceme, aby “létalo” někde mimo.

**Kód**

**CSS**

.header {

height: 95vh;

background: linear-gradient(rgba(0,0,0,0.8),

rgba(0,0,0,0.8)),url(../img/hogwarts-middle.jpg);

background-repeat: no-repeat;

background-size: cover;

background-position: top;

clip-path: polygon(0 0, 100% 0, 100% 90%, 0 100%);

**position: relative;**

}

**.logo {**

**position: absolute;**

**top: 40px;**

**left: 40px;**

**}**

**Jak kód vypadá na stránce**



**7. Zarovnání obsahu hlavičky na střed**

**Komentář**

Kromě loga máme v hlavičce ještě část s třídou header-box. Její obsah bychom chtěli dostat na střed. Nejdříve vyjmeme obsah pomocí position absolute (vše se bude zarovnávat vůči třídě header, protože v předchozí části jsme jí dali position: relative;). Zarovnáme obsah 50 % odshora a 50 % zleva. Takže by obsah měl být uprostřed. Vzniká tu problém, že uprostřed je ve skutečnosti levý horní roh. Co tedy potřebujeme? Posunou obsah doleva o polovinu šířky obsahu a nahoru o polovinu výšky obsahu. Tím docílíme toho, že obsah bude přesně uprostřed. To nám pomůže udělat transform, které díky translate může pohybovat objekty na stránce.

Ještě zarovnáme text na střed pomocí text-align.

**Kód**

**CSS**

.header-box {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%,-50%);

text-align: center;

/\*color: #fff;\*/

}

**Jak kód vypadá na stránce (vy uvidíte nápis černý, já ho jen ukázkově dal bílý, abychom ho viděli lépe na obrázku; barvu upravíme až v další části)**



**Úkol**

Do vaší stránky (do header) vložte takovýto kód:

<div class="header-box">

<h1 class="heading-main">

<span class="heading-main-head">Vítejte v Bradavicích</span>

<span class="heading-sub-head">Vstupte mudlové</span>

</h1>

<a href="#" class="btn">Přihlásit se do Bradavic</a>

</div>

Vaším úkolem je zarovnat ho na střed pomocí výše uvedeného triku.

**8. Mohu to použít? (Can I use?)**

**Komentář**

Nyní se vrhneme na úpravu textů. Dáme textu zlatavou barvu pomocí color a všechna písma změníme na velká použitím text-transform.

Přidáme ještě obrys písmu. Stačí obrys tlustý 1 pixel a zvolíme černou barvu.

**Kód**

**CSS**

.heading-main {

color: #ae9959;

text-transform: uppercase;

-webkit-text-stroke-width: 1px;

-webkit-text-stroke-color: black;

}

Co je to webkit?

Představte si, že se vymyslí nová vlastnost v CSS. Například právě možnost text-stroke, který dává rámeček písmu. Protože je nová, tak musí projít testovací fází. Proto i prohlížeče ji zavádí postupně. Navíc to, že nyní má nějaký účel a funguje takto (tedy v době testovací fáze), neznamená, že v budoucnu nezačne fungovat jinak. Občas se tedy setkáte s tím, že je zapotřebí u vlastnosti říci, s jakým prohlížečem tato vlastnost komunikuje. Ne všechny prohlížeče ji musí podporovat.

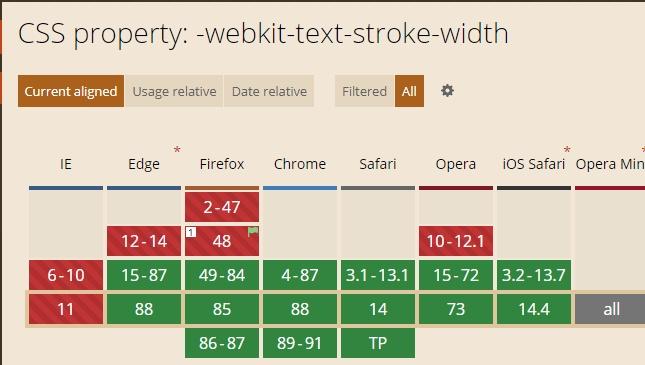
-webkit- = Chrome, Safari, nová verze Opery

-moz- = Firefox

-o- = starší verze opery

-ms- = Internet Explorer a Edge

Jak zjistíme, jak danou vlastnost zapsat? Stačí jít na stránku <https://caniuse.com/>. Zde zadáme danou vlasnost - např. zde <https://caniuse.com/?search=text-stroke-width>



Vidíme, že -webkit-text-stroke-width je podporováno většinou prohlížečů (jsou zelené). Tuto vlastnost nepodporuje např. IE = tedy Internet Explorer, ale ten byl později nahrazen Edge prohlížečem, který vlastnost podporuje - jak vidíme z obrázku. Všimněte si nahoře, že vlastnost musíte zapsat s prefixem -webkit-

**Jak kód vypadá na stránce**

Rámeček není moc vidět, protože je černý na černém pozadí, ale celkově na nápisu je díky tomu vidět lehký stín.



**Úkol**

Dříve jsme se seznámili s CSS vlastností clip-path. Použijete stránku caniuse.com, abyste zjistili, jestli nepotřebuje v některých prohlížečích prefix. Pokud ano, tak svůj kód upravte.

**9. Stylování textů v hlavičce**

**Komentář**

Pojďme si upravit obě dvě části nadpisu, které jsou obalené spany. Protože chceme mít oba dva na samostatném řádku, tak je dáme display: block; Budou z nich blokové elementy, které jsou nenasytné a zabírají celý řádek. Nezapomeňme, že spany jsou řádkové elementy, které se řadí za sebe - což jsme viděli do teď.

Pak už jen přidáme velikost textu (font-size), vypneme tučnost (font-weight) a navolíme mezery mezi písmeny (letter-spacing). Mysleme na to, že první část nadpisu je hlavní a zbytek už je “obyčejný”. Proto má hlavní část textu větší velikost textu než “obyčejná” část.

**Kód**

**CSS**

.heading-main-head {

display: block;

font-size: 50px;

font-weight: 400;

letter-spacing: 10px;

}

.heading-sub-head {

display: block;

font-size: 30px;

font-weight: 400;

letter-spacing: 5px;

text-transform: lowercase;

}

**Jak kód vypadá na stránce**



**10. Změna textu aby lépe vypadal**

**Komentář**

Občas grafika znamená i to, že budete muset upravit délku textu. Např. text pod hlavním nadpisem “můžete vstoupit, i pokud jste mudlové” by byl hezčí, kdyby byl kratší. Tak ho zkusme přepsat.

**Kód**

**HTML**

<span class="heading-sub-head">Vstupte do světa čar a kouzel</span>

**Jak kód vypadá na stránce**



**11. Stylování tlačítka, link, visited, active, hover**

**Komentář**

Nyní nastylujeme tlačítko. Budou to jen běžné styly. Jen pozor na první dva řádky níže. Pokud chceme tlačítko odsadit od textu nad ním pomocí margin-top, tak musíme tlačítko přepnout na inline-block. Proč? V základním kurzu jsme si říkali, že pokud je něco řádkový element (inline), tak má paddingy po všech stranách, ale marginy jen vlevo a pravo. A tady chceme margin nahoře. Takže změníme podstatu prvku pomocí **display: inline-block;**

Ostatní řádky už jsou jen dekorační - barva textu, mezera mezi písmeny, rámeček, zaoblení rámečku, barva pozadí, vše velkými písmeny, zrušení podtržení.

Každý odkaz má 4 stádia:

link = nenavštívený odkaz

visited = již navštívený odkaz

active = ve chvíli kliknutí na odkaz

hover = když se na odkaz najede myší

V CSS je zvykem definovat stejné vlastnosti pro link a visited. Pak můžeme mít stejné vlastnosti pro hover a active nebo, jako v našem případě, nemusíme (ale o tom až dále)

**Kód**

**CSS**

/\* Button in header \*/

.btn:link,

.btn:visited {

display: inline-block;

margin-top: 20px;

padding: 15px 40px;

color: #fff;

letter-spacing: 2px;

border: 2px solid #ae9959;

border-radius: 100px;

background-color: #37352c;

text-transform: uppercase;

text-decoration: none;

}

**Jak kód vypadá na stránce**



**Úkol**

Podobným způsobem nastylujete odkaz do podoby tlačítka v challenge složce. Zadejte stejné vlastnosti, ale hrajte si s jejich nastavení - měňte barvy, pixely atd. Cokoliv, co vás napadne.

**12. Elastické tlačítko a posun prvku**

**Komentář**

Pokud chceme vytvořit dojem elasticity tlačítka, tak ve chvíli najetí přisoudíme tlačítku posun o 4px nahoru (po ose y), dáme výraznější stín. Ve chvíli kliknutí posuneme tlačítko směrem dolů (ze 4 pixelů nahoru jen o 1 pixel nahoru = tlačítko se posune o 3px směrem dolů) a také uděláme menší stín při kliknutí.

**Kód**

**CSS**

.btn:hover {

transform: translateY(-4px);

box-shadow: 0 10px 20px #ae9959;

}

.btn:active {

transform: translateY(-1px);

box-shadow: 0 5px 10px #ae9959;

}

**Jak kód vypadá na stránce**

Ve chvíli najetí (hover)



Ve chvíli kliknutí (active)



**Úkol**

Nastylujte podobným způsobem své tlačítko v challenge složce.

**13. Animace hoveru a active pomocí transition**

**Komentář**

Pokud na tlačítko najedete (hover) nebo na něj klikáte (active), tak efekty jsou trhané, protože se okamžitě projeví. Pokud chceme, aby se něco animovalo = trvalo nějakou dobu a postupně se projevovalo, tak použijeme transition. Z výchozího stavu při najetí myši chceme, aby se posun i stín (tedy vše = all) postupně projevovaly (animovali) a to po dobu 0,2 sekundy (všimněte si zápisu tečka 2 s = nulu před desetinnou čárkou psát nemusíte; s znamená sekund). Musíme transition přidat k oběma, protože chceme animaci při hoveru i active.

**Kód**

**CSS**

.btn:hover {

transform: translateY(-4px);

box-shadow: 0 10px 20px #ae9959;

**transition: all .2s;**

}

.btn:active {

transform: translateY(-1px);

box-shadow: 0 5px 10px #ae9959;

**transition: all .2s;**

}

**Jak kód vypadá na stránce**

Zkuste na tlačítko nyní najet a měli byste vidět rozdíl

**Úkol**

Vytvořte efekt animace pro své tlačítko v challenge složce.

**Tip**

Pokud na tlačítko najedete, tak pohyb je animovaný. Pokud na tlačítko klikáte, tak pohyb je animovaný. Pokud z tlačítka ale jen odjedete myší, tak pohyb není animovaný a je opět trhaný. Jak to vyřešit? Kam ještě dát transition?

K obyčejnému stavu link a visited. Protože když odjíždíte ze stavu hover, tak se vše vrací zpět do stavu link a visited. Takže stačí zde zadat opět transition a pohyb tlačítka po odjetí myši bude animovaný.

.btn:link,

.btn:visited {

display: inline-block;

margin-top: 20px;

padding: 15px 40px;

color: #fff;

letter-spacing: 2px;

border: 2px solid #ae9959;

border-radius: 100px;

background-color: #37352c;

text-transform: uppercase;

text-decoration: none;

**transition: all .2s;**

}

**Úkol**

Aplikujte výše uvedený tip na své tlačítko v challenge složce. Porovnejte stav před a po přidání transition. Vidíte ten rozdíl?

**14. Animace pomocí keyframes**

**Komentář**

Nyní se seznámíme s keyframy. Ty nám umožní vytvářet animace. Předpokládejme, že chceme animovat nadpis H1. Budeme chtít, aby na začátku animace, kterou pojmenujeme fromTop, byl nadpis H1 posunutý o 20px nahoru oproti své výchozí pozici a byl průhledný. Na konci animaci budeme chtít, aby se zastavil na své výchozí pozici (tedy 0px) a abyl byl naopak zcela viditelný. To zapíšeme následovně. Napište si keyframe na konec CSS.

@keyframes fromTop{

0% {

opacity: 0;

transform: translateY(-20px);

}

100% {

opacity: 1;

transform: translateY(0px);

}

}

Nyní ještě musíme tuto animaci přidělit danému elementu. V našem případě h1. To uděláme tímto zápisem, který říká, aby u h1 proběhla animace s názvem fromTop po dobu 1 vteřiny a aby průběh byl lineární (aby animace probíhala stále stejně rychle).

.header-box h1 {

animation: fromTop 1s linear;

}

A nyní se podívejte na stránku a znovu jí načtěte. Uvidíte, jak se animace na H1 aplikovala. Jak se text pomalu objevuje a “přilétává” shora.



**Úkol**

Naanimujte text, který máte v hlavičce. Zkuste nejdříve tu samou variantu jako výše - přilétne shora.

**15. Pomocí keyframes animujeme tlačítko**

**Komentář**

Stejným způsobem nadefinujeme keyframe i pro tlačítko. Akorát budeme chytít, aby se tlačítko objevovalo zdola. Princip bude ale úplně stejný. Na začátku animace - v 0 % - bude tlačítko průhledné a posunuto po ose Y o 50 pixelů = dolů. Na konci animace - ve 100 % - bude tlačítko viditelné a na své původní pozici (0px). Opět si můžete napsat keyframe dolů do CSS a pak ho výše přiděláme odkazu, který představuje tlačítko.

**Kód**

**CSS**

@keyframes fromDown{

0% {

opacity: 0;

transform: translateY(+50px);

}

100% {

opacity: 1;

transform: translateY(0px);

}

}

Animaci přidělíme tlačítku

.header-box a {

animation: fromDown 1s linear;

}

**Jak kód vypadá na stránce**



**Úkol**Naanimujte libovolně tlačítko, které máte na stránce.

**Tip**

**Pokud chcete naanimovat nadpis tak, aby přilétal zprava, tak by to vypadalo takto**

@keyframes fromRight {

0% {

opacity: 0;

transform: translateX(+30px);

}

100% {

opacity: 1;

transform: translateX(0px);

}

}

**+ přiřadíme keyframe s názvem fromRight například hlavičce**

.heading-main {

color: #ae9959;

text-transform: uppercase;

-webkit-text-stroke-width: 0.5px;

-webkit-text-stroke-color: black;

**animation: fromRight 3s linear;**

}

**A pokud chcete animaci zleva, tak stačí:**

@keyframes fromLeft {

0% {

opacity: 0;

transform: translateX(-30px);

}

100% {

opacity: 1;

transform: translateX(0px);

}

}

**+ přiřadíme animaci fromLeft nějakému prvku (zde hlavičce)**

.heading-main {

color: #ae9959;

text-transform: uppercase;

-webkit-text-stroke-width: 0.5px;

-webkit-text-stroke-color: black;

animation: fromLeft 3s linear;

}

**Tip**

Zkusíme u elementu, kde je zadané animation, napsat ještě druhou časovou hodnotu. Např. tento zápis znamená, že animace má trvat 2 vteřiny, ale má začít až 3 vteřiny po načtení stránky.

animation: fromDown 2s 3s linear

Co se ale v takovém případě stane? Text nad tlačítkem má animaci, která se normálně provede. Jenže po celé 3 vteřiny bude tlačítko běžně vyditelné. Pak teprve zmizí a začne se objevovat. To není zrovna to, co bychom chtěli.



My bychom potřebovali, aby se pravidla v keyframes fromDown u 0% uplatnila okamžitě po načtení stránky (takže tlačítko by bylo neviditelné), počkalo se poté 3 vteřiny a pak se animace spustila. To dokážeme snadno. K samotné animaci přidáme ještě řádek animation-fill-mode: backwards; To způsobí, že se vše na element v 0% aplikuje, pak se čeká 3 vteřiny a poté se animace spustí.

animation: fromDown 2s 3s linear;

animation-fill-mode: backwards;

Podívejte se na stránku, že nám nyní vše skvěle funguje. S načasováním je zapotřebí si pohrát, aby vše vypadalo tak, jak má. Pokud vrchní text přilétá 2 vteřiny, tak by přilétnutí tlačítka mohlo být zpožděné jen o 2 vteřiny a ne o 3, jak máme v kódu. Můžeme popřípadě změnit na

animation: fromDown 2s 2s linear;

animation-fill-mode: backwards;

Ale to už je jen na nás

**16. Fotka v pozadí jede s námi**

**Komentář**

Populární je efekt, kdy při scrollování “jede” fotka společně s vámi dolů. Tento efekt nastavíme pomocí jednoho řádku v CSS. Nastavte a vyzkoušejte na stránce, jak vše funguje.

Přidejte si na stránku např. odstavce pod header, abyste měli více obsahu a mohli scrollovat.

**Kód**

**CSS**

.header {

height: 95vh;

background: linear-gradient(rgba(0,0,0,0.7), rgba(0,0,0,0.7)),url(../img/hogwarts-middle.jpg);

**background-attachment: fixed;**

background-repeat: no-repeat;

background-size: cover;

background-position: top;

-webkit-clip-path: polygon(0 0, 100% 0, 100% 90%, 0 100%);

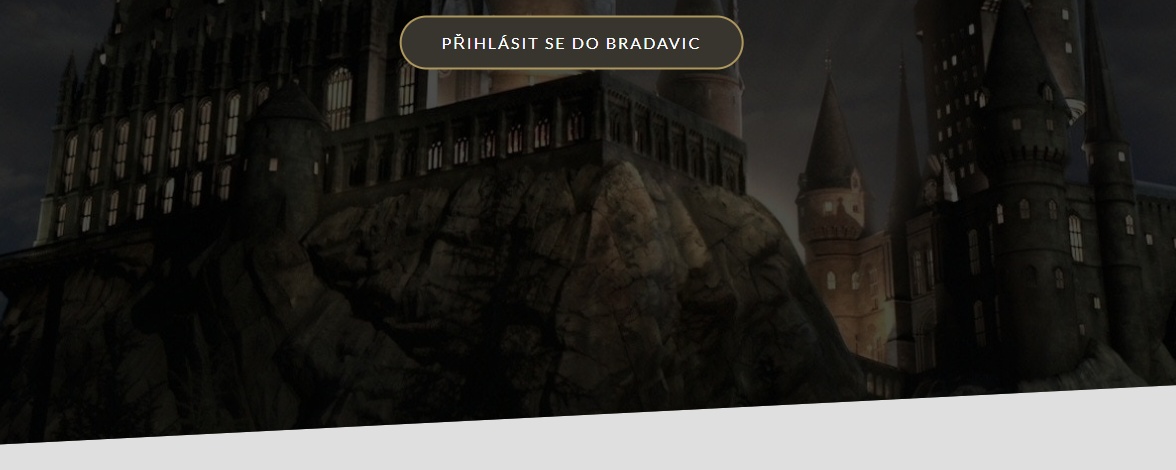
clip-path: polygon(0 0, 100% 0, 100% 90%, 0 100%);

position: relative;

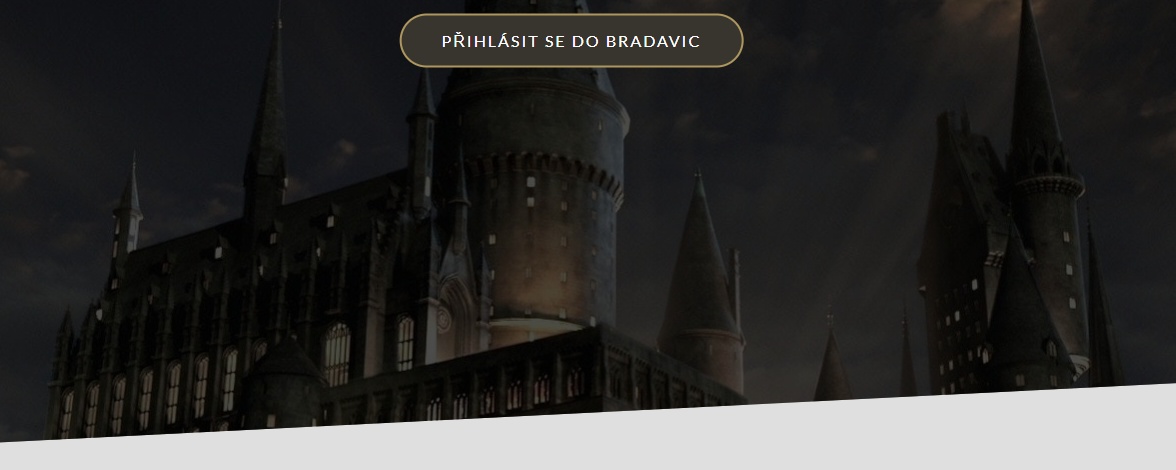
}

**Jak kód vypadá na stránce**

Bez této funkce při scrollování zůstává fotka na místě.



S touto funkcí při scrollování jeden fotka společně s námi dolů. Přidejte si na stránky nějaké odstavce pod hlavičku, abyste mohli scrollovat (poté odstavce zase smažte).



**17. Sekce s hrdiny**

**Komentář**

Nyní do sekce main vložíme první sekci, ve které ukážeme dva hrdiny. Budeme chtít vytvořit dva sloupečky vedle sebe, kdy v jednom bude fotka hrdiny a ve druhém text o něm. K tomu použijeme grid.

**Kód**

**HTML**

<main>

<!-- Heroe of Hogwarts -->

<section class="heroes">

**<div class="row">**

**<div class="col span-1-of-2">**

<img src="img/brumbal.jpg" alt="Albus Brumbál, ředitel v Bradavicích">

**</div>**

**<div class="about-hero col span-1-of-2">**

<h2>Albus Percival Wulfric Brian Brumbál</h2>

<p>Jméno Albus znamená v latině bílý, slovo dumbledore je archaický výraz pro čmeláka (stejně tak i starší české brumbál). I ostatní Brumbálova jména jsou symbolická, Percival je v legendě o svatém grálu statečný rytíř, který nakonec grál najde, Wulfric znamená „vládce nad vlky“, Brian v irštině znamená „vznešený“, rovněž jde o jméno hrdiny z irské mytologie.</p>

**</div>**

**</div>**

</section>

</main>

**Jak kód vypadá na stránce (obrázek je veliký, ale to nám nyní nevadí; dále ho upravíme)**



**18. Šedé pozadí sekci s hrdiny**

**Komentář**

Dáme sekci jemně šedé pozadí (background-color). A také odsadíme. Když dáme šedé pozadí, tak oříznutí vrchní sekce hlavičky způsobí, že mezi touto sekcí a hlavičkou bude vidět bílá část a to nechceme. Proto celý blok posuneme nahoru pomocí margin-top se zápornou hodnotou. Ale tím se nám posune nahoru i obrázek a text a začne mizet za hlavičkou. Proto jim dáme padding, abychom je posunuli níže a byly pěkně vidět.

**Kód**

**CSS**

.heroes {

background-color: rgb(224, 223, 223);

padding: 137px 0px;

margin-top: -120px;

}

**Jak kód vypadá na stránce**



**19. Zmenšení obrázku**

**Komentář**

Obrázek je hodně veliký, tak ho zmenšíme. Přepneme ho na blokový element (display), abychom si mohli hrát s jeho šířkou a upravovat jeho vnější okraje (marginy). Šířku jsme dali na 70%. Pokud by byla 100%, tak by obrázek šel až do okraje stránky a stránka by nebyla tak “vzdušná”. Ale můžete si to vyzkoušet. Ještě obrázku zaoblíme rohy (border-radius).

**Kód**

**CSS**

.heroes img {

display: block;

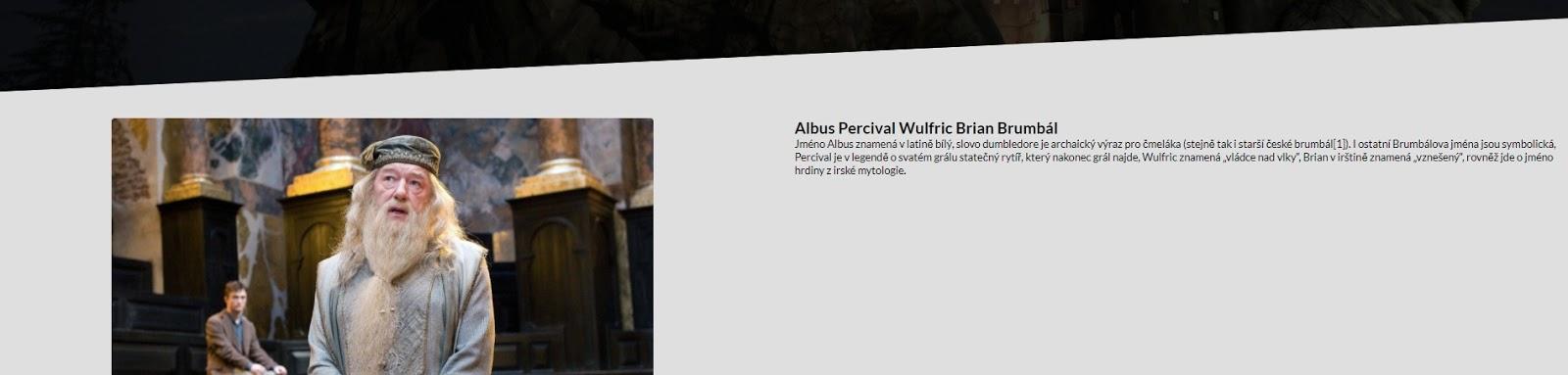
width: 70%;

margin: 0 auto;

border-radius: 5px;

}

**Jak kód vypadá na stránce**



**20. Upravujeme text**

**Komentář**

Ještě upravíme text. Zde nebude nic náročného. Upravíme velikost písma nadpisu a odstavce (font-size). Upravíme odsazení shora, aby žádné nebylo - nadpis bude zarovnán na úrovni obrázku. Ještě “provzdušníme” text a dáme mu výšku řádku (line-height). Také text vpravo odsadíme pomocí margin-right, aby nebyl tak “nalepený” na okraji. Text odstavce si dáme do boků pomocí text-align: justify;

**Kód**

**CSS**

.about-hero {

font-size: 23px;

padding-top: 0px;

}

.about-hero p {

margin-top: 10px;

font-size: 20px;

line-height: 57px;

margin-right: 40px;

text-align: justify;

}

**Jak kód vypadá na stránce**



**21. Gradient v pozadí textu**

Přidáme nadpisu gradient. To znamená přechod z jedné barvy do druhé. Nejdříve dáme nadpisu třídu gradientBackground.

Poté navolíme jako obrázek v pozadí linearní gradient. Zadáme dvě barvy a dáme, kolik % má každá barva obsahovat. Zkuste developerskou konzoli, abyste % měnili a uvidíte, jak se barvy hýbou.

Ještě tam máme údaj 90deg - to znamená 90 stupňů. K čemu to je? Pokud necháte gradient 0deg, tak je odshora dolů. My ho ale chceme zleva do prava, takže můžeme napsat 90deg. Místo 90deg můžete napsat **to right** a výsledek bude stejný.

**Kód**

**HTML**

<h2 **class="gradientBackground"**>Albus Percival Wulfric Brian Brumbál</h2>

**CSS**

.gradientBackground {

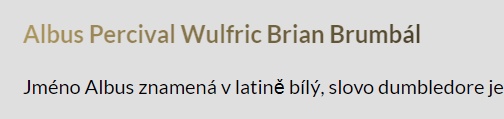
background-image: linear-gradient(90deg, rgba(174,153,89,1) 0%, rgba(0,0,0,1) 76%);

-webkit-background-clip: text;

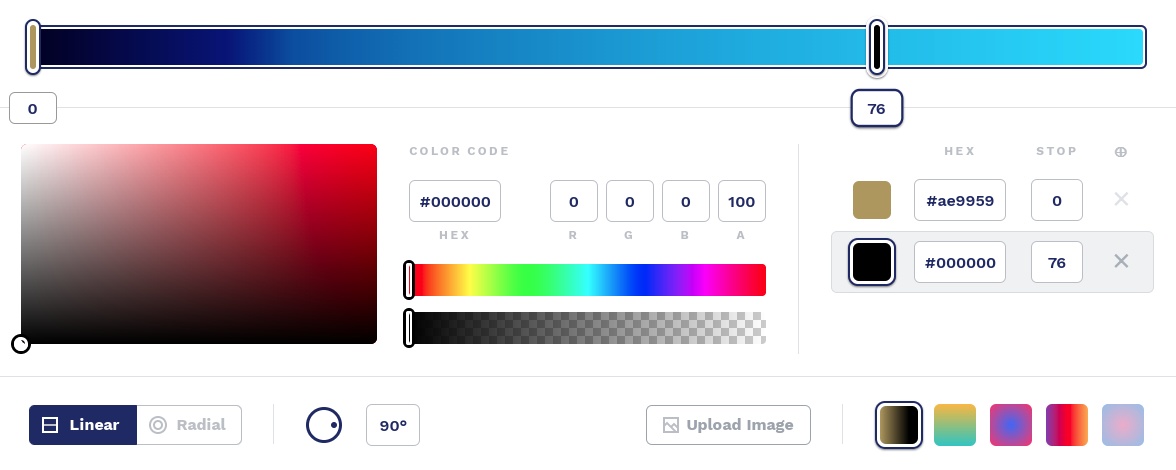
color: transparent;

}

**Jak kód vypadá na stránce**



Pokud se vám nechce gradient vymýšlet, tak použijte tento nástroj, který jsme si již dříve ukazovali <https://cssgradient.io/>

****

**22. Sekce s výčtem pomocí gridu**

**Komentář**

Vytvoříme si v main další sekci, která bude obsahovat výčet (ať už čehokoliv). Sekce bude mít dva řádky (row) a každý z nich bude rozdělený do tří sloupečků. K tomu využijeme grid. Doplníme si texty např. pomocí Lorem Ipsum.

**Kód**

**HTML**

<section class="**goals**">

<h2>Co se v Bradavicích naučíte</h2>

<div class="**row** **goals-content**">

<div class="**col span-1-of-3** **one-goal**">

<p>Officiis facere soluta aliquid? Velit, porro.</p>

</div>

<div class="**col span-1-of-3** **one-goal**">

<p>Officiis facere soluta aliquid? Velit, porro facere facere.</p>

</div>

<div class="**col span-1-of-3** **one-goal**">

<p>Officiis facere soluta aliquid? Velit, porro facere.</p>

</div>

</div>

<div class="**row** **goals-content**">

<div class="**col span-1-of-3** **one-goal**">

<p>Officiis facere soluta aliquid? Velit, porro.</p>

</div>

<div class="**col span-1-of-3** **one-goal**">

<p>Officiis facere soluta aliquid? Velit, porro.</p>

</div>

<div class="**col span-1-of-3** **one-goal**">

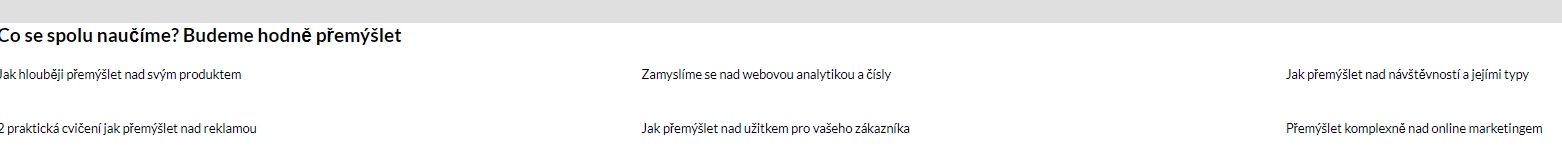
<p>Officiis facere soluta facere aliquid? facere Velit, facere porro.</p>

</div>

</div>

</section>

**Jak kód vypadá na stránce**



**23. Umístění sekce a stylování pozadí**

**Komentář**

Sekci upravíme ve stylu hlavičky. Dáme na pozadí obrázek (background) a ten přetáhneme tmavou vrstvou (linear-gradient). Pokud dáváte obrázek na pozadí, tak nezapomeňte dát vždy zarovnání obrázku a jeho umístění (background-size a background-position). Sekci “zařízneme” do tvaru polygonu (clip-path).

Sekci posuneme výše, aby se nám hezky napojovala na horní sekci (margin). To se nám ale obsah sekce skryje za horní sekci (bude jakoby pod ní), tak pomocí vnitřního okraje (padding) ji zase posuneme dolů.

**Kód**

**CSS**

/\* Goals section \*/

.goals {

background: linear-gradient(to right bottom, rgba(0,0,0,0.8),

rgba(0,0,0,0.8)), url(../img/hogwarts-middle.jpg);

background-attachment: fixed;

color: #ae9959;

background-size: cover;

background-position: top;

margin-top: -60px;

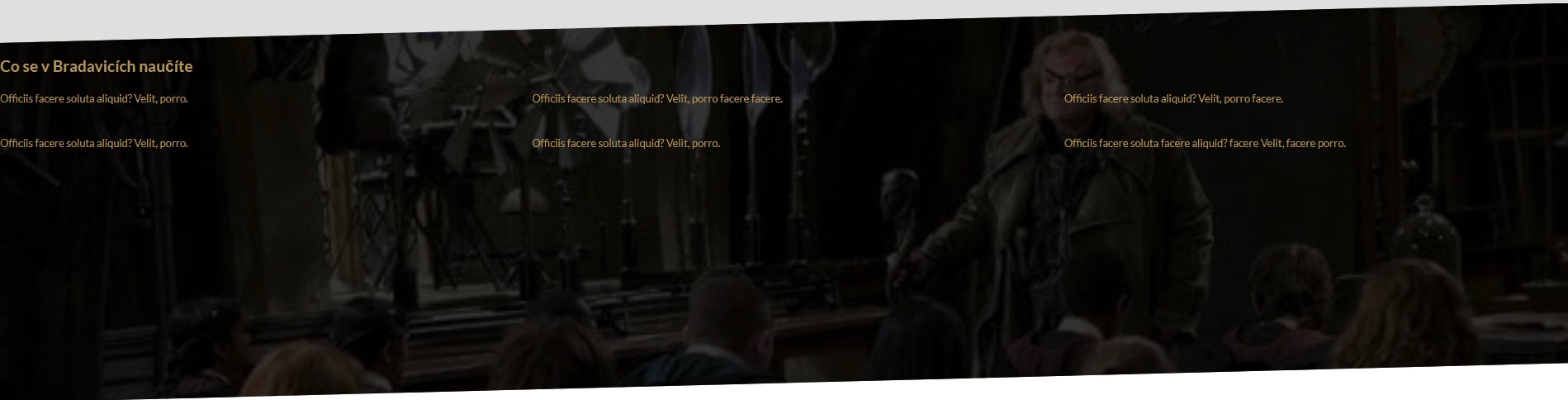
padding-top: 20px;

height: 605px;

clip-path: polygon(0 10%, 100% 0, 100% 90%, 0 100%);

}

**Jak kód vypadá na stránce**



**24. Úprava hlavního nadpisu a zarovnání obsahu na střed**

**Komentář**

Upravíme nadpis sekce běžnými úpravami. Obsah sekce zarovnáme na střed. Upravujeme div s třídou goals-content. Takže je to blokový element přes celý řádek. My mu jen přiřadíme šířku a tím vzniknou vnější okraj. Ten poté pomocí margin: 0 auto; rozpůlíme a dáme polovinu před div. Tím se div zarovná na střed. Přesněji řečeno se tím zarovnají na střed oba dva divy, protože máme dva řádky, které mají třídu goals-content.

**Kód**

**CSS**

.goals h2 {

margin-top: 90px;

text-align: center;

font-size: 40px;

letter-spacing: 5px;

}

.goals-content {

width: 1000px;

margin: 0 auto;

margin-top: 40px;

}

**Jak kód vypadá na stránce**



**25. Tvorba boxů s texty**

**Komentář**

Upravíme jednotlivé texty tak, aby z nich vznikly boxíky. Dáme jim rámeček (border) a vhodně obarvíme. Protože dáváme černou barvu pozadí (background-color), tak dáme každému boxu stín (box-shadow), který zvýrazní hranice boxíku.

Aby text uvnitř boxu byl nastředu, tak ho odsadíme shora pomocí margin-top.

**Kód**

**CSS**

.one-goal {

border: 2px solid rgba(0,0,0,0.8);

background-color: rgba(0,0,0,0.8);

box-shadow: 0 8px 20px #ae9959;;

font-size: 20px;

text-align: center;

padding: 5px;

margin-left: 30px;

border-radius: 10px;

width: 300px;

height: 100px;

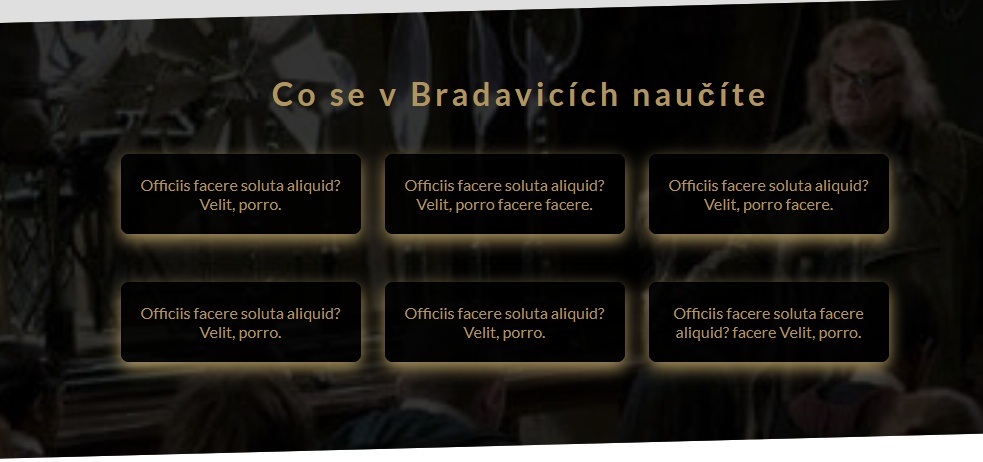
}

.one-goal p {

margin-top: 20px;

}

**Jak kód vypadá na stránce**



**26. Efekt po najetí myši a transition pro plynulost efektu**

**Komentář**

Pokud na box najedeme myší (hover), tak chceme, aby se box posunul směrem nahoru o 10 pixelů (tedy o 10 pixelů po ose y). Aby posun nebyl trhaný, tak pomocí transition řekneme, aby vlastnost transform trvala 0.3 sekundy a pohyb byl lineární - stále stejně rychlý po celou dobu 0.3 vteřin.

Zkuste si postupně odkomentovat řádky se stíny a sledujte, co se se stíny děje. Vždy musíte mít aktivní jen jeden řádek se stíny.

**Kód**

**CSS**

.one-goal p {

margin-top: 20px;

}

.one-goal:hover {

transform: translateY(-10px);

transition: all 0.3s linear;

/\* box-shadow: 0 0px 0px #ae9959; \*/

/\* box-shadow: 0 8px 20px rgb(169, 20, 20); \*/

}

**Jak kód vypadá na stránce (v první řadě uprostřed vidíme, jak se box posunul výše při najetí myší)**

