Úvod do HTML

<html>

<body>

<h1>Sem napíšte hlavný nadpis</h1>

<p>Sem napíšte odsek textu</p>

<h2>Sem napíšte podnadpis</h2>

</body>

</html>

<html> </html> - všetko medzi nimi je kód HTML

<body> </body> - čokoľvek medzi nimi sa zobrazuje v hlavnom okne prehliadača

<h1> </h1> - hlavný nadpis

<p> </p> - odsek textu

<h2> </h2> - podnadpis

Dôležité pojmy

element - pozostáva z úvodného tagu, záverečného tagu a obsahu medzi nimi

atribúty - dodatočné informácie o obsahu elementu, pozostáva z názvu (ďalšie informácie informácie o obsahu prvku (malé písmená)) a hodnoty (informácie nastavenia pre atribút (“ “)

<p lang=”sk”> odsek po slovensky<p>

lang - určenie jazyka

Užitočné Tagy

<a>

anchor

klikateľný odkaz

najviac používané atribúty:

href - adresa stránky/súboru na ktorý nás presmeruje po kliknutí na odkaz

target - hodnota "blank" otvorí novú kartu v prehliadači Acac

<a href=“https://www.alza.sk” target=“\_blank”>Odkaz na Alzu.</a>

<b>

bold

tučné písmo

<h1><b>Text</b></h1>

<i>

italic

kurzíva

<h1><i>Text</i></h1>

<hr/>

horizontal row

pridá horizontálnu čiaru

nepárový tag

<br/>

break row

presunutie do nového riadka

nepárový tag

<h1>Toto je celá <br/> veta.</h1>

<img/>

image

pridanie obrázka

nepárový tag

najviac používané atribúty:

src - cesta k obrázku

width - šírka obrázka v pixeloch

height - výška obrázka v pixeloch

alt - popis obrázka

<img src=”images/obrazok.png” width=”100” height=”100” alt=”Popis obrázka” />

HTML zoznamy

Usporiadané zoznamy

ordered lists

položky sú očíslované

napr. zoznam krokov pre recept, pracovný postup….

každá položka medzi hlavnými tagmi <ol></ol> je ešte umiestnená medzi tagy <li></li>

<ol>

<li>Narežeme cibuľu a zeleninu na malé kúsky.</li>

<li>Opražíme cibuľu na rozpálenom oleji.</li>

<li>Pridáme nakrájanú zeleninu a chvíľu ju opražíme.</li>

<li>Zalejeme horúcou vodou a pridáme korenie a soľ.</li>

<li>Varíme, kým zelenina nezmäkne a môžeme podávať.</li>

</ol>

Neusporiadané zoznamy

unordered lists

položky začínajú odrážkou

každá položka medzi hlavnými tagmi <ul></ul> je ešte umiestnená medzi tagy <li></li>

<ulč>

<li>Narežeme cibuľu a zeleninu na malé kúsky.</li>

<li>Opražíme cibuľu na rozpálenom oleji.</li>

<li>Pridáme nakrájanú zeleninu a chvíľu ju opražíme.</li>

<li>Zalejeme horúcou vodou a pridáme korenie a soľ.</li>

<li>Varíme, kým zelenina nezmäkne a môžeme podávať.</li>

</ul>

Zoznamy definícií

definition lists

skladajú sa zo sady výrazov spolu s definíciami

medzi tagmi <dl></dl> obvykle zadávame dvojicu tagov:

<dt></dt> sa používa na označenie definovaného pojmu (definition term)

<dd></dd> sa používa na zadanie definície (definition description)

<dl>

<dt>HTML</dt>

<dd>Vytváranie kostry webovej stránky</dd>

<dt>CSS</dt>

<dd>Vizuálne formátovanie webovej stránky</dd>

</dl>

Vnorené zoznamy

nested lists

<ul>

<li>Strešné okno</li>

<li>rozmer kolies

<ul>

<li>15 palcov</li>

<li>16 palcov</li>

</ul>

Ďalšie tagy a príkazy

Prepojenie na špecifickú časť stránky (napr. nadpis)

vytvoríme id

<h2 id=”uvod”>Úvod</h2>

vytvoríme prepojenie (umiestnime tlačidlo)

<a href=”#uvod”><button> Úvod</button></a>

Pridávanie komentárov k zdrojovému kódu (nezobrazuje sa na stránke)

<!--komentár-->

Prepojenie s emailovou službou

<p>Prosím<a href=”mailto:vas\_email”> kontaktujte nás</a>ak potrebujete viac informácií.</p>

Vytvorenie klikateľného obrázka

<a href=”subor.html”><img src=”image.jpg”></a>

Pridanie popisu obrázka

<figure>

<img src=”pes.jpg” alt=”fotografia\_psa”>

<figcaption> Na obrázku môžete vidieť psa. </figcaption>

</figure>

Pridanie GIF súboru na stránku

<h1>

<img src=”pes.gif” width=”400” height=”400”

alt=”obrazok\_psa”/>

</h1>

ako nadpis nemusíme použiť len text, ale aj obrázok

Vytvorenie tabuľky

nadpisy tabuľky sú umiestnené v prvku <thead>, ktorý definuje stĺpce

telo je umiestnené vo vnútri prvku <tbody>, určuje priame hodnoty a názvy riadkov

päta patrí do prvku <tfoot>, ukončuje tabuľku (súčet, strata…)

Table heading = <th> Table row = <tr> Table data = <td>

<table>

<thead>

<tr> <th>Dátum</th> <th>Príjem</th> <th>Výdavky</th> </tr>

</thead>

<tbody>

<tr> <th>8. január</th> <td>300</td> <td>50</td> </tr>

<tr> <th>2.február</th> <td>400</td> <td>80</td> </tr>

</tbody>

<tfoot>

<tr> <th>Medzisúčet</th> <td>700</td> <td>130</td> </tr>

</tfoot>

</table>

<td colspan=”2”>300</td>tento zápis znamená, koľko stĺpcov tabuľky položka zaberá (v našom prípade zaberá položka 300 presne 2 stĺpce).

<td rowspan=”2”>300</td>tento zápis znamená, koľko riadkov tabuľky položka zaberá (v našom prípade zaberá položka 300 presne 2 riadky>.

Vloženie formulárov

<form action=”http://www.example.com/page.php” method=”post”>

<fieldset>

<legend> Kontaktné údaje </legend>

<label> Meno a priezvisko <input type=”text” name=”kontaktne\_udaje” size=”30” maxlenght=”100”> </label> <br />

<label> E-mailová adresa: <input type=”email” name=”emailova\_adresa” size=”30” maxlenght=”100”> </label> <br />

</fieldset> <br />

</form action>

<fieldset>

<legend> Vaše hodnotenie </legend>

<p>

<label for=”info”>Ako ste sa o nás dozvedeli? </label>

<select name=”respondent” id=”info”>

<option value=”facebook”>Facebook</option>

<option value=”instagram”>Instagram</option>

</select>

</p>

<p>Páči sa vám dizajn našej stránky? <br />

<label> <input type="radio" name="hodnotenie" value="ano" /> Áno</label>

<label> <input type="radio" name="hodnotenie" value="nie" /> Nie</label>

</p>

<fieldset>

Zoskupenie prvkov stránky do jedného bloku

<div>

vytvárame k nemu CSS kód

atribúty

id

class

HTML:

<div class=”úvod”>

<h2>nadpis</h2>

<p>text</p>

</div>

CSS:

<style>

div.uvod {border: 7px outset green;

background-color: ivory;

text align: center;}

</style>

Zoskupenie jednotlivých prvkov v rámci riadku

<span>

podobný ako tag <div>, ale zoskupuje jednotlivé prvky v rámci riadku

HTML:

<p>

Pridáme <span class=”ingrediencie”>mrkvu</span>

<span class=”ingrediencie”>petržlen</span> a

<span class=”ingrediencie”>cibuľu</span> a všetko spolu zamiešame.

</p>

CSS:

<style>

span.”ingrediencie” {color: red}

</style>

Pridanie videa

<video src=”video.mp4” poster=”obrazok.jpg” width=”700” height=”600” controls>

</video>

Pridanie audia

<audio controls>

<source src=”resource/images/zvuk.mp3”>

</audio>

Úvod do CSS

Selektory

Univerzálny

{}

vzťahuje sa na všetky prvky stránky

Typový

h1, h2, h3 {}

vzťahuje sa na konkrétne prvky (napr. nadpisy)

CLASS

.dolezite {} (zameriava sa na akýkoľvek prvok, ktorého atribút class má hodnotu “dolezite”)

p.dolezite {} (zameriava sa iba na prvky <p>, ktorých atribút class má hodnotu “dolezite”)

vzťahuje sa na prvok, ktorého atribút class má hodnotu zhodujúcu sa s hodnotou zadanou za bodkou

ID

#uvod {} (zameriava sa na prvok, ktorého atribút ID má hodnotu “uvod”

p#uvod

zhoduje sa s prvkom, ktorého atribút ID má hodnotu, ktorá sa zhoduje s hodnotou zadanou za symbolom #

Selektor priameho potomka (dieťaťa)

li>a {} (zameriava sa na všetky prvky <a>, ktoré sú priamymi potomkami prvku <li> (ale nie

na iné prvky <a> na stránke)

zhoduje sa s prvkom, ktorý je priamym potomkom - dieťaťom iného prvku

Selektor potomka

p a {} (zameriava sa na akékoľvek prvky <a>, ktoré sa nachádzajú v prvku <p>, aj keď sú

medzi nimi ďalšie prvky)

zhoduje sa s prvkom, ktorý je potomkom iného určeného prvku (nie len priamym potomkom)

Všeobecný selektor súrodenca

zhoduje sa z prvkom, ktorý je súrodencom iného prvku, hoci to nemusí byť bezprostredne predchádzajúci prvok

ID VS CLASS

Rozdiel je v tom, že ID možno použiť na identifikáciu prvku len raz, zatiaľ čo rovnaký názov triedy (class) možno použiť na identifikáciu prvkov opakovane.

<div id=”horna\_cast”>

<h1>Autá</h1>

<p class=”uvod”>Časť stránky s autami.<p/>

<p class=”uvod”>Ďalšia časť stránky s autami.<p/>

</div>

Prepojenie HTML súboru s externým CSS súborom (umiestnenie nad tagom body)

Ak existujú 2 alebo viac pravidiel, ktoré sa vzťahujú na ten istý prvok, je dôležité pochopiť, ktoré bude mať prednosť.

Ak sú 2 selektory identické, prednosť má druhý z nich.

Ak je jeden sektor špecifickejší ako ostatné, konkrétnejšie pravidlo bude mať prednosť pred všeobecnejšími.

Napríklad:

h1 je špecifickejšie ako \*

p b je špecifickejšie ako p

Za akúkoľvek hodnotu niektorej vlastnosti je možné pridať kľúčové slovo !important (hodnota sa považuje za dôležitejšiu ako iné hodnoty, ktoré sa vzťahujú na ten istý prvok).

p b {

color: red !important

}

p b {

color: blue;

}

Pomocou kľúčového slova inheri dokážu podradené prvky deliť vlastnosti od nadradených

Vysvetlenie pojmov

Margin

vytvorenie priestoru okolo prvku smerom von od akýchkoľvek definovaných hraníc

Padding

používa sa na vytvorenie priestoru okolo prvku vo vnútri akýchkoľvek definovaných hraníc (border)

zhora, sprava, zdola, zľava

Section

obsah vnútri je zoskupený (patrí k 1 konkrétnej téme)

nevyužíva sa ako kontajner kvôli nastavovaniu rôznych štýlov na časti html kódu

DIV

nesprostredkováva žiadny špeciálny význam k obsahu, ktorý je v ňom umiestnený

kontajner na html elementy, na ktoré chceme aplikovať rôzne CSS štýly

Spôsob usporiadania veci v menu: serus kraken

<nav>

<ul>

<li class="zoznam"><a href="">Domov</a></li>

<li class="zoznam"><a href="">Modely</a></li>

<li class="zoznam"><a href="">Kontakt,</a></li>

</ul>

</nav>

Color

umožňuje určiť farbu textu vo vnútri prvku

Hodnoty rgb

rgb(100, 100, 90)

vyjadrujú farby podľa toho, koľko červenej, zelenej a modrej sa používa na ich vytvorenie

Hexadecimálne kódy

#ee3e80

šesťmiestne kódy, ktoré predstavujú množstvo červenej, zelenej a modrej vo farbe

Názvy farieb

DarkCyan

preddefinované názvy farieb, ktoré rozoznávajú prehliadače

Komentáre

/\*komentár\*/

klávesová skratka zakomentovanie a odkomentovanie ctr+K+U

\_comment: “css nedovoluje pouzit komentár v komentári - je ale možné použiť tento trik na komentovanie - vlastnosť \_comment sa ignoruje”

Background-color

predvolené nastavenie je biela farba

Základné parametre farieb

Hue - Odtieň

základná vlastnosť farby, podľa ktorej ju pomenúvavame

uvádza sa v stupňoch

Saturation - Sýtosť

označuje množstvo šedej farby

doplnková vlastnosť

Brightness - Jas

množstvo čiernej vo výslednej farbe

doplnková vlastnosť

Kontrast

je dôležité, aby bol text čitateľný - aby mala stránka dobrý kontrast

príliš nízky a vysoký kontrast nie je vhodný - zle sa číta

pri dlhých textoch sa odporúča použiť tmavosivý text na bielom pozadí alebo sivo-biele písmo na tmavom pozadí (pomáha zväčšiť medziriadkové medzery, či nastaviť písmo na tučné)

Nepriehľadnosť - Opacity

umožňuje špecifikovať nepriehľadnosť prvku a jeho podriadených prvkov

hodnota je číslo medzi 0,0 a 1,0 (0,5 je 50% nepriehľadnosť)

RGBA

umožňuje určiť farbu, rovnako ako pri hodnote RGB, ale pridáva štvrtú hodnotu alfa na označenie nepriehľadnosti

ovplyvní iba prvok, na ktorý sa použije, nie podradené prvky

background-color: rgba(255, 255, 0, 0.8);

HSL COLOR

zavedená ako alternatívny spôsob určenia farieb

hue, saturation, lightness

Hue - Odtieň

uvádza sa v stupňoch (0 - 360)

Saturation - Sýtosť

vyjadruje sa v percentách

množstvo šedej vo výsledej farbe

Lightness -Svetlosť

množstvo bielej alebo čiernej vo farbe

vyjadruje sa v percentách (0% - čierna, 100% - biela)

niekedy sa označuje aj ako svietivosť

Vlastnosť HSLA

umožňuje určiť farbu, rovnako ako pri hodnote HSL, ale pridáva štvrtú hodnotu alfa na označenie nepriehľadnosti

hodnota je číslo medzi 0,0 a 1,0 (0,5 je 50% nepriehľadnosť)

background-color: hsla(100, 100%, 50%, 0.5);

CSS VLASTNOSTI

Font-family

umožňuje určiť typ písma, ktorý sa použije pre text vo vnútri prvku, na ktorý sa vzťahuje pravidlo CSS

ak je názov viacslovný, treba dať dvojité úvodzovky

Napr.: font-family: Georgia,

"Times New Roman", serif

Existujú dva typy rodín písiem:

family-name - “times” “courier” “arial”

generic-family “serif” “sans-serif” “cursive” “fantasy”

Napr. font family: Georgia, “Times New Roman”, serif

!Na stránke nevyužívať viac ako tri fonty písma!

Font-size

-umožňuje určiť veľkosť písma

Najbežnejšie spôsoby určenia veľkosti písma:

Pixely

bežne sa používajú, pretože umožňujú webovým dizajnérom veľmi presnú kontrolu nad tým, koľko miesta zaberá ich text

Percentá

predvolená veľkosť textu v prehliadačoch je 16px, 75= je teda 12px a 200% 32px

em

historický odvodená od šírky písmena M

napr. font-size:1em; je zhodná s predvolenou veľkosťou fontu prehliadačov

Font-face

umožňuje použiť písmo aj keď nie je nainštalované v PC používateľa

Stránka s voľne dostupnými fontami: www.fontspace.com

@font-face {

font-family: 'mishella';

src: url(../font/Mishella-w1Pdw.otf);

}

p {

font-family: 'mishella';

font-size: 80px;

font-weight: bold;

}

Font-weight

vytvára tučný text

2 hodnoty, ktoré bežne nadobúda

normal - normálna hrúbka

bold - tučné písmo

Font-style

3 hodnoty, ktoré môže nadobúdať

normal - normálny štýl

italic - text sa zobrazí kurzívou (písma vytvorené priamo pre to)

oblique - text sa zošikmí

Text-transform

používa sa na nastavenie veľkých a malých písmen v texte

nadobúda hodnoty:

uppercase - text s veľkými písmenami

lowercase - text s malými písmenami

capitalize - prvé písmeno každého slova sa zobrazí ako veľké

h1 {

text transform: uppercase;

}

p:first-letter{

text-transform: capitalize;

}

veľké písmeno na začiatku vety

flex-basis

- zabezpečí 50% stlpcov, ktoré sme vytvorili

line-height

- riadkovanie

text-decoration

none-odstrani vsetky dekoracie aplikovane na text

underline - prida ciaru pod text

Overline - pridá ciaru a text

line-through - preciarknutie textu

line-height

vyska riadku

p{line-height: 1.4em}

letter-spacing

medzery medzi písmenami

p{ letter-spacing: 5em}

word-spacing

Medzery medzi slovami

p{word-spacing: 3em}

text-align

zarovnanie textu

left - text zarovnaný doľava

center - stred

right - doprava

justify - text sa roztiahne na celu sirku prvku

text-indent

odsadenie prvého riadku

p{text-indent: 20px}

transition

vytvorí prechod vzhľadom na vlastnost transform, ktorý bude trvat 0,5s

display:block

každý z prvkov zaberie celú šírku stránky

deafultne divko

display:inline

elementy nezaberajú celú šírku, len takú, akú potrebujú

span ju deafultne nadobúda

display:inline-block

kombinacia, ktora sa sprava podobne ako inline, akurat vieme nastaviť urcite parametre

deafultne pri obrázkoch

display:none

prestane daný element zobrazovať na stránke, ako keby v html jazyku neexistoval

display:flex

display:grid

vsetky elementy na stranke su v tvare boxu (obdlznik, stvorec)

cesta k boxu elementu sa delí na časti: obsah boxu (napr. text), padding boxu, border (orámovanie)

borderom konci element, margin je len prazdno okolo prvku

padding boxu zvacsuje samotny box, v ktorom sa nachadza obsah

border boxu obklopi priestor okolo nastaveneho paddingu o nastavenuhodnotu

Marginy jednotlivých elementov sa prelínajú a nezratavajú sa do kopy

Ak není padding (alebo border) obsah za zväčší

box-sizing: border-box; (css)

ak si do všeobecného selektora zadáme border-box tak vieme pomocou parametra height a weight nastaviť pri prvku konkrétny rozmer elementu margin a padding

text-shadow

xpx ypx rozmazané px farba; (text-shadow: 2px 2px 5px red;)

vytvára duplikát textu a dá ho pozadia

prispôsobuje sa textu (ak zväčším veľkosť textu, zväčší sa aj veľkosť tieňa)

p::first-line

prvý riadok v texte bude mať nejaký štýl

pseudo elementy slúžia nato, aby bližšie zadefinovali na ktorú časť elementu bude nastavený nejaký štýl

a:link

nastavuje link na odkaz ktorý nebol ešte zakliknuty používateľom

a:link {

color: green

}

a:visited

označuje čo už bolo navštívené užívateľom

a:visited {

color: green

}

a:hover

umožňuje nastaviť štýl prvku v momente, keď naň posuniem kurzor

a:hover {

color: green

}

a:active

umožňuje nastaviť štýl prvku v momente, keď na ňom držím stlačené ľavé tkačidlo myši

a:active {

color: green

}

a:focus

umožňuje nastaviť štýl prvku na ktorom nebolo naposledy kliknute

a:focus {

color: green

}

:: pseudonymy

pseudo elementy ::before, ::after v CSS umožňujú vložiť obsah na stránku bez toho, aby musel byť pridaný najprv HTML.

:before (alebo ::before) sa považuje za prvé dieťa prvku/elementu, na ktorý sa vzťahuje

:after (alebo ::after) sa považuje za posledné dieťa

Atribúty

používame na nastavenie štýlu na špecifický html element alebo ich vieme použiť pre vkladanie obsahu do časti content

v CSS zapisujeme [atribut]

Tlačidlo = rodič, má definovaný atribút s názvom.

parameter position nastavíme na rodičovskom elemente na hodnotu relative, aby sme vedeli na dieťati tohto elementu pomocou position absolute a dodatočných parametrov top, left a right nastaviť presné umiestnenie vzhľadom na rodiča

Transition

- vytvorí prechod vzhľadom na vlastnost transform, ktorý bude trvat 0,5s

Position

určuje typ metódy určovania polohy použitej pre prvok/element

sú umiestnené pomocou top, bottom, left, right vlastností, no position musí byť nastavená ako prvá

static (deafultné)

t, b, l, r nefunguje

umiestnenie prvku vzhľadom na normálny tok stránky

relative

prvok je možné pomocou t, b, l, r vychýliť od normálnej polohy

po presunutí sa žiaden ďalší obsah nenatlačí na vzniknuté miesto

fixed

prvok je umiestnený vzhľadom k zobrazovacej stránke (vľavo hore)

ostane stále na rovnakom mieste, aj keď je stránka skrolovaná

poloha sa nastaví pomocou t, b, l, r

absolute

a najbližšieho predka (vľavo hore v nadradenom prvku)

ak nie je nadradený prvok, tak je umiestnený vzhľadom k zobrazovacej stránke

prvok nastavený ako absolute, je vyňatý z normálneho toku sránky a môže prekrývať ostatné prvky

sticky

prvok je umiestnený na základed scrolovania používateľa

je na svojom mieste, kým nenarazí na hornú časť stránky, potom sa posúva spolu so skrolovaním

Flexbox

jednorozmerný spôsob rozloženia prvkov v CSS

vytvára hlavnú (horizontálna) a priečnu (vertikálna) os

rozdelený do 2 hlavných komponentov:

flexbox kontajner

nadradený prvok (rodič)

obsahuje všetky flexbox položky ako deti

flexibilne sa prispôsobuje zmenšovaniu a zväčšovaniu okna prehliadača

display:flex

flexbox položky

ak nešpecifikujeme ďalšie vlastnosti, rozložia sa na 1 riadok zľava doprava a zaberú len toľko miesta koľko porebujú (automaticky sa zmršztia, aby sa zmestili do flexbox kontajnera)

Rozloženie položiek na hlavnej osi flexbox kontajnera

.flex-polozka {

width: 15%;

}

.flex-kontajner {

display: flex;

justify-content: flex-start; (predvolené)

umiestňuje všetky položky na začiatok hlavnej osi (vľavo)

justify-content: flex-end;

umiestňuje všetky položky na koniec hlavnej osi (vpravo)

justify-content: center;

umiestňuje všetky položky do stredu hlavnej osi

justify-content: space-between;

priestor vo vnútri kontajnera je rovnomerne rozložený medzi každú flexbox položku (natlačené na pravú a ľavú stranu)

justify-content: space-around;

podobné správanie ako space-nbetween, no pridáva miesto aj zľava a sprava

justify-content: space-evenly;

podobné správanie ako space-around,ale veľkosť priestoru medzi vonkajšou stranou kontajnera a prvou/poslednou položkou je rovnaká ako priestor medzi prvkami (namiesto polovičnej veľkosti)

}

Rozloženie položiek na priečnej osi flexbox kontajnera

.flex-polozka {

width: 15%;

}

.flex-polozka nth-child(1) {

height: 60px;

}

.flex-polozka nth-child(2) {

height: 80px;

}

.flex-polozka nth-child(3) {

height: 100px;

}

pseudo trieda nth.child (číslo) vyberá položky flexboxu na základe ich poradia vo flexbox kontajneri a priradí im určité parametre

.flex-kontajner {

display: flex;

align-items: stretch;

roztiahne položky tak, aby vyplnili celú dĺžku krížovej osi, pokiaľ nemajú nastavenú špecifickú výšku (height:initial - ako keby výška nebola nastavená)

align-items: flex-start;

funguje rovnako ako v prípade justify-content, ale začína sa v hornej časti osi

align-items: flex-end;

funguje rovnako ako v prípade justify-content, ale začína sa v dolnej časti osi

Nastavenie orientácie hlavnej a priečnej osi

flex-direction: row; (predvolená orientácia flexbox kontajnera)

Hlavná os je horizontálna, zatiaľ čo priečna os je vertikálna. (To tiež znamená, že hlavná os začína vľavo, zatiaľ čo priečna os začína hore.)

.flex-kontajner {

display: flex;

flex-direction: row;

justify-content: flex-end;

align-items: center;

}

flex-direction: row-reverse;

hlavná os teraz začína na pravej strane, zatiaľ čo priečna os sa nebude meniť a stále začína hore

flex-direction: column;

orientácia osi sa úplne zmení - hlavná os je zvislá a priečna je vodorovná - pri použití justify-content budeme položky rozkladať vo vertikálnom smere a align-items v horizontálnom

flex-direction: column-reverse;

rovnaký ako column, ale obráti smer hlavnej osi, takže sa rozmiestnenie položiek začína v dolnej časti flexbox kontajnera

Nastavenie pre každú položku zvlášť

.flex-kontajner {

display: flex;

align.items: stretch;

}

.flex-polozka:nth-child(1) {

align-self: flex-start;

}

Jedna dôležitá vec, ktorú treba poznamenať je, že neexistuje spôsob, ako použiť justify-self na potomkovi (flexbox položka), pretože to môže iba rodič (flexbox kontajner) cez justify-content.

Skutočná sila flexboxu spočíva taktiež v jeho schopnosti meniť a prispôsobovať veľkosť položiek vo flexbox kontajneri navzájom medzi sebou a tiež okolím flexbox kontajnera.

To sa deje prostredníctvom 3 rôznych vlastností: flex-grow, flex-shrink a flex-base.

Flex-grow

definuje, koľko priestoru navyše môže flexbox položka zabrať na vyplnenie flexbox kontajnera. V predvolenom nastavení je táto vlastnosť nastavená na 0, čo znamená, že položka nedostane žiadne miesto navyše

.flex-polozka:nth-child(1) {

flex-grow: 2;

}