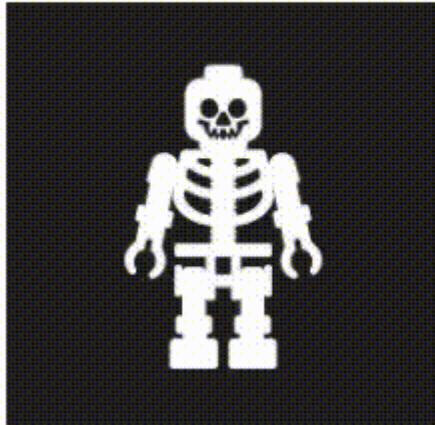# INTRO TO HTML/CSS

AGENDA

- HTML basic
    1. Doctype
    2. Main tags
    3. Semanitc tags
    4. HTML5 tags
    5. Forms
- CSS basic
    1. Cascade
    2. Inheritance
    3. Specificity
    4. and more
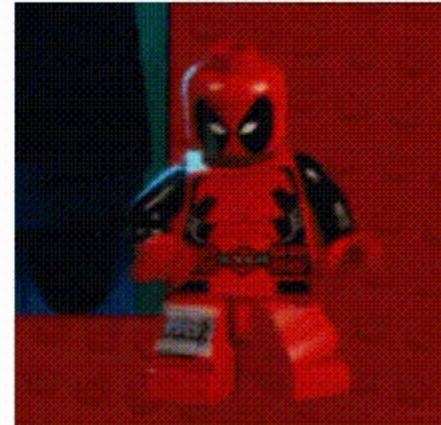- Layout
    1. Flex box
    2. Grid
- Bootstrap
- Responsive

| HTML | CSS | JavaScript |
| structure | presentation/appearance | dynamism/action |

HTML

DEFINITIONS

HTML - Hyper Text Markup Language

Tag - basic element of html structure <tagname>content goes here... </tagname>

Attribute - provides additional information about an element comes in name/value pairs like: name="value"

```html
<p data-attr="some-value"> Content</p>
```

BASIC HTML STRUCTURE

```html
<!DOCTYPE html>
<html>
	<head>
		<meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
		<link rel="stylesheet" href="style.css">
		<title>site header</title>
		<script src="some.file.launch.in.header"></script>
	</head>
	<body>
		...here live everything what you see on a page...
		<!-- and comment to -->
		<script src="some_script_could_live_here"></script>
	</body>
</html>
```

!DOCTYPE

```
<!DOCTYPE [Элемент верхнего уровня] [Публичность] "[Регистрация]//[Организация]//[Тип] [Имя]//[Язык]" "[URL]">
```

HTML4

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
```

HTML5

<!DOCTYPE html>

MAIN PAGE TAG

\<html\>root tag, shows browser were is page borders;\</html\>

\<head\>contains specific page info such as meta, link, scripts etc.;\</head\>

\<title\>page title, shows in browser tab; use for search engines;\</title\>

\<body\>contain all visible on page content;\</body\>

TAGS CATEGORY

category	example
Empty Tag	
 , <hr/> , <img/> , <link/> does not contain the closing tag
Container Tag	<main> </main>, <p> </p> has the opening and closing tag
Non formating content tags	<script></script> <style></style>
Foreign content tags	<ruby> </ruby> , <svg> </svg>

cheat sheet

<META> TAGS

used to provide additional information about page

name	**meta-name: description, keywords, author...**
http-equiv="***"	**HTTP Equivalent Info: title, etc.**
content="***"	**Information content**

Following is an example to set UTF-8 encoding

```
<meta http-equiv = "Content-Type" content = "text/html; charset = UTF-8" />
```

meta cheat-sheet

MORE HTML5 TAGS

<header> - header for a document or section

<main> - main content of a document

<aside> - content aside from the page content

<footer> - footer for a document or section

<nav> - navigation links

<article> - article in a document

<section> - section in a document

<svg> - container for SVG graphics

HTML5 media tags

HTML5 form tags

...more tags

HTML5 MEDIA TAGS

```html
<audio src="/link/to-audio.mp3" controls>
        <p>If you are reading this, it is because your browser does not support
         the audio element.</p>
</audio>
```

```html
<video src="/video/link-to-video.ogg" width="170" height="85" controls>
        <p>If you are reading this, it is because your browser does not
         support the HTML5 video element.</p>
</video>
```

```html
<source src="/music/video.wma" type="audio/x-ms-wma">
<source src="/music/video.mp3" type="audio/mpeg">
```

```html
<track src="/video/text_en.vtt" kind="subtitles" srclang="en" label="English">
<track src="/video/text_es.vtt" kind="subtitles" srclang="es" label="Spanish">
```

FORM


```html
<form action="onSubmit()" method="post" >

        <input type="email" placeholder="e-mail" required>
        <input type="password" placeholder="pass">
        <input type="text" id="">
        <input type="number" value=5 >

        <label for="yes">Yes</label>
        <input type="checkbox" name="yes" id="yes" >
        <label for="no">No</label>
        <input type="checkbox" name="no" id="no" >

        <input type="radio" name="radio" id="rad1" >
        <input type="radio" name="radio" id="rad2" >

        <input type="button" value="Submit">

</form>
```

form doc's

DIV VS SEMANTIC

Before

```html
<!DOCTYPE html>
<html>
  <head>
  <body id="index">
    <div id="wrapper">
      <div id="header">
      <div id="topNav">
      <div id="content">
        <div id="sidebar">
          <div id="subNav" class="aside">
          <div id="postListing" class="aside">
        </div>
        <div id="page">
        <div class="clear"> </div>
      </div>
      <div id="footer">
        <div id="footerNav">
        <div id="footerCopyright
      </div>
    </div>
    <script type="text/javascript">
  </body>
</html>
```

After

```html
<!DOCTYPE html>
<html>
  <head>
  <body id="index">
    <div id="wrapper">
      <header id="header">
      <nav id="topNav">
      <div id="content">
        <aside id="sidebar">
          <nav id="subNav" class="asid
          <section id="postListing" cl
        </aside>
        <article id="page">
        <div class="clear"> </div>
      </div>
      <footer id="footer">
        <nav id="footerNav">
        <small id="footerCopyright">
      </footer>
    </div>
    <script type="text/javascript">
  </body>
</html>
```

CSS

DEFINITIONS

CSS - Cascading Style Sheets

CSS rule - consists of a selector and a declaration block

Selector - points to the HTML element you want to style

Declaration block - one or more declarations separated by semicolons

```css
p {
    border: 1px solid black;
    color: red;
}
```

CSS Syntax Module Level 3

CSS PLUG & PLAY

you can add CSS inline

```
<style>
        div {
                    backgroung: purple;
                    color: pink;
        }
</style>
```

or normal

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

CASCADE

The cascading order (from lowest importance):

- User agent normal stylesheets
- User agent !important stylesheets
- User normal stylesheets
- Author normal stylesheets
- Author !important stylesheets
- User !important stylesheet

always use **<u>NORMALIZE</u>**

stylesheet with an **!important** declaration always has the highest priority

INHERITANCE

Inherited properties	Non-inherited properties
	background
	border
color	margin
font-family	padding
font-size	display
font-style	float
font-weight	width
text-align	height

Full property table

UNITS

Absolute	Relative
Pixels (px)	Percentages (%)
Centimeters (cm)	Font-sizes (em&rem)
Millimeters (mm)	Character-sizes (ex&ch)
Inches (in)	Viewport Dimensions (vw &vh)
Points (pt)	Viewport Max (vmax)
Picas (pc)	Viewport Min (vmin)

read more...

SELECTORS

Tag selector — `E`

`div`

`h1`

`img`

Class selector — `.class`

`.header`

`.item`

`.is-active`

ID selector — `#id`

`#header`

`#search-button`

`#dropdown`

Attribute selector — `[attr]`

`[disabled]`

`[role="dialog"]`

`[data-component*="popup"]`

Universal selector — `*`

`*`

Cascade selector — `E F`

`ul li`

`div a`

`header h1 .link`

Child selector — `E > F`

`ul > li`

`.menu > .item`

`footer > img`

General sibling selector — `E ~ F`

`input ~ label`

`span ~ em`

`.image ~ .link`

Adjacent sibling selector — `E + F`

`input + label`

`span + em`

`.image + .link`

PSEUDO CLASSES

Starts with single colon

Links

`:link`

`:active`

`:visited`

`:any-link`

Events

`:hover`

`:focus`

`:focus-within`

`:target`

`:active`

`:checked`

Forms

`:enabled`

`:disabled`

`:valid`

`:invalid`

`:indeterminate`

`:required`

Elements

`:first-child`

`:last-child`

`:first-of-type`

`:last-of-type`

`:nth-child(n)`

`:nth-last-child(n)`

Negation

`:not()`

General

`:root`

PSEUDO ELEMENTS

Starts with double colon

Generated content

`::before` & `::after`

```css
div::before {
        content: "before << ";
}
div::after {
        content: " >> after";
}
```

before << This is text paragraph >> after

Input placeholder

`::placeholder`

Enter some text

AT-RULE SELECTORS

`@charset`

`@media`

`@page`

`@keyframes`

`@supports`

[CSS Selectors Level 4 @ W3C](#)

[CSS Selectors @ KharkivCSS #2](#)

[HTML5 Elements and their styling @ KharkivCSS #3](#)

SPECIFICITY

element selector

Specificity: 0,0,1

class selector
attribute selector
pseudo-class

Specificity: 0,1,0 (10)

id selector

Specificity: 1,0,0

style attribute

Specificity: 1,0,0,0

Stuff & Nonsense

a
1 x element selector

Sith power: 0,0,1

p a
2 x element selectors

Sith power: 0,0,2

.foo
1 x class selector *

Sith power: 0,1,0

a.foo
1 x element selector
1 x class selector

Sith power: 0,1,1

p a.foo
2 x element selectors
1 x class selector

Sith power: 0,1,2

.foo .bar
2 x class selectors

Sith power: 0,2,0

p.foo a.bar
2 x element selectors
2 x class selectors

Sith power: 0,2,2

#foo
1 x id selector

Sith power: 1,0,0

a#foo
1 x element selector
1 x id selector

Sith power: 1,0,1

.foo a#bar
1 x element selector
1 x class selector
1 x id selector

Sith power: 1,1,1

.foo .foo #foo
2 x class selectors
1 x id selector

Sith power: 1,2,0

style
1 x style attribute

Sith power: 1,0,0,0

* Same specificity
class selector =
attribute attribute =
pseudo-classes

!important

BOX MODEL
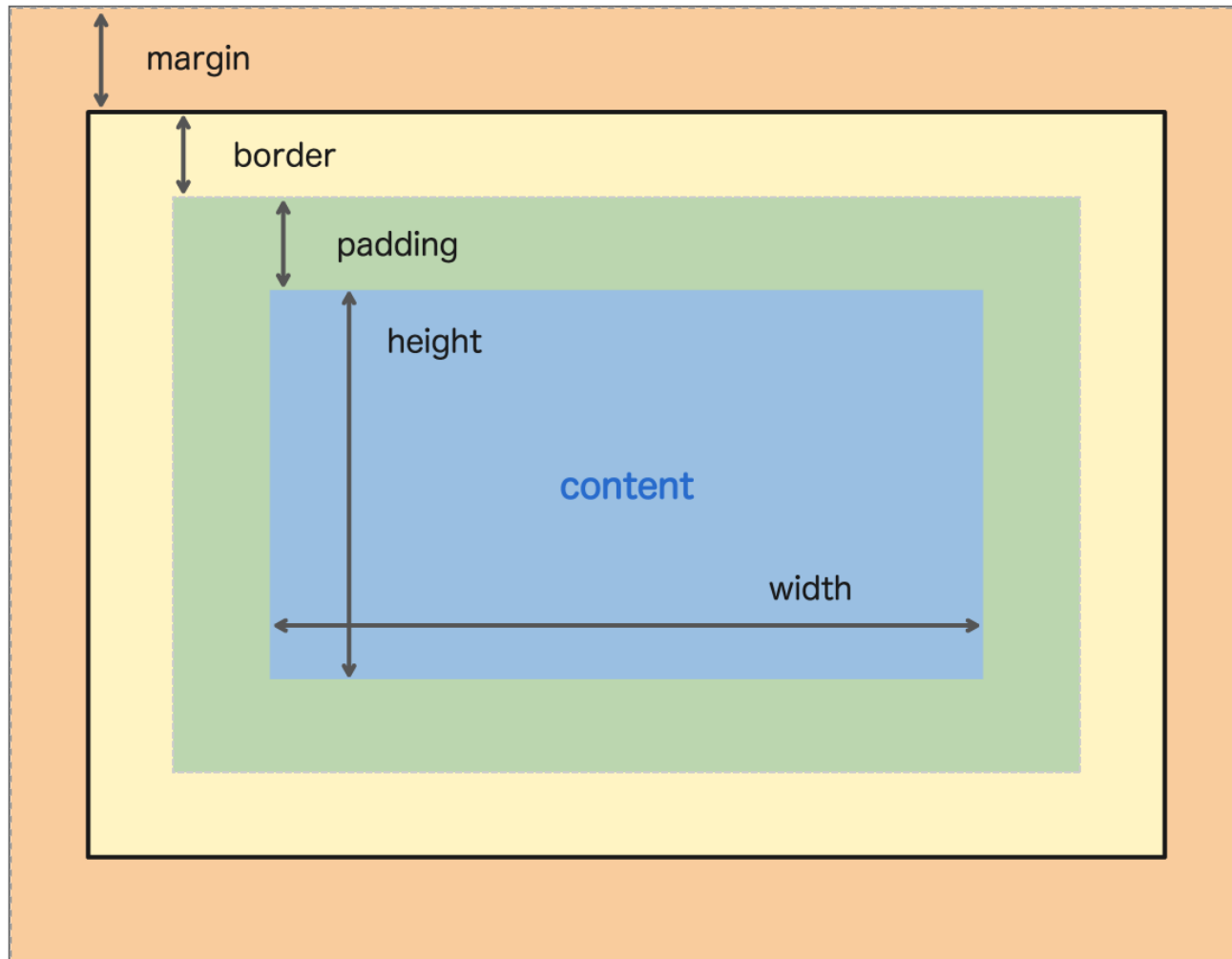
box-sizing
content-box vs border-box

HTML	CSS		Result

```
<div class="box border-box">
    block with box-model border-box
</div>

<div class="box content-box">
    block with box-model content-box
</div>
```

block with box-model border-box

block with box-model content-box

Resources   1×   0.5×   0.25×   Rerun

DISPLAY

rules that describe how HTML elements should be laid out and behave

- none
- inline
- block
- inline–block
- list–item
- table
- flex
- grid
- **<u>and more…</u>**

POSITION

- Static
- Relative
- Absolute
- Fixed
- Sticky

STATIC

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eos, quidem.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eos, quidem.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Tempore maiores pariatur exercitationem velit eligendi, molestias asperiores vero officia quod sed.

RELATIVE

```css
.position-relative {
    position: relative;
    left: 20px;
    top: 20px;
}
```

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eos, quidem.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eos, quidem.

Lorem ipsum dolor sit

amet, c   consectetur

ABSOLUTE

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eos, quidem.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eos, quidem.

Lorem ipsum dolor sit amet, adipisicing elit. Tempore consectetur maiores pariatur exercitationem velit eligendi, molestias asperiores vero officia quod sed.

FIXED

Result

```
.position-fixed {
    position: fixed;
    left: 20px;
    top: 20px;
}
```

consectetur m dolor sit amet, consectetur adipisicing elit. Eos, quidem.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eos, quidem.

Lorem ipsum dolor sit amet, adipisicing elit.

Temporo maiores periotur

Resources   1×   0.5×   0.25×   Rerun

FLOAT AND CLEAR

HTML	CSS	Result

```
<div>normal flow</div>
<div>normal flow</div>
<div>normal flow</div>
<div class="float">floated</div>
<div>normal flow</div>
<div>normal flow</div>
<div>normal flow</div>
<div>normal flow</div>
<div>normal flow</div>
```

normal flow

normal flow

normal flow

floated normal flow

normal flow

normal flow

Resources   1× 0.5× 0.25×   Rerun

Second
FirstThird

```html
<div class="float">First</div>
<div class="non-float">Second</div>
<div class="clear"></div>
<div class="after-float">Third</div>
```

Second

First

Third

CLEARFIX

```css
.clearfix::after {
        content: '';
        display: table;
        clear: both;
}
```

NAMING AND BEM

```
 1  nav ul {
 2  ──background-color: PaleVioletRed;
 3  ──list-style: none;
 4  ──padding: 0;
 5  ──width: 200px;
 6  ──border: 1px solid MediumVioletRed;
 7  }
 8
 9  nav ul li {
10  ──border-bottom: 1px solid MediumVioletRed;
11  ──padding: 5px;
12  }
13
14  nav ul li:last-child {
15  ──border-bottom: 0;
16  }
17
18  nav ul li a {
19  ──color: white;
20  ──text-decoration: none;
21  }
22
23  nav ul li a:hover {
24  ──text-decoration: underline;
25  }
```

```
 1  .site-nav {
 2  ──background-color: PaleVioletRed;
 3  ──list-style: none;
 4  ──padding: 0;
 5  ──width: 200px;
 6  ──border: 1px solid MediumVioletRed;
 7  }
 8
 9  .site-nav .site-nav-item {
10  ──border-bottom: 1px solid MediumVioletRed;
11  ──padding: 5px;
12  }
13
14  .site-nav .site-nav-item:last-child {
15  ──border-bottom: 0;
16  }
17
18  .site-nav .site-nav-item a {
19  ──color: white;
20  ──text-decoration: none;
21  }
22
23  .site-nav .site-nav-item a:hover {
24  ──text-decoration: underline;
25  }
```

<u>BEM naming</u>
<u>one more naming guideline</u>

LAYOUT

DEFINITION

Layout - General page blocks organization which defines structure of a page and positioning of its content.

FLEX BOX
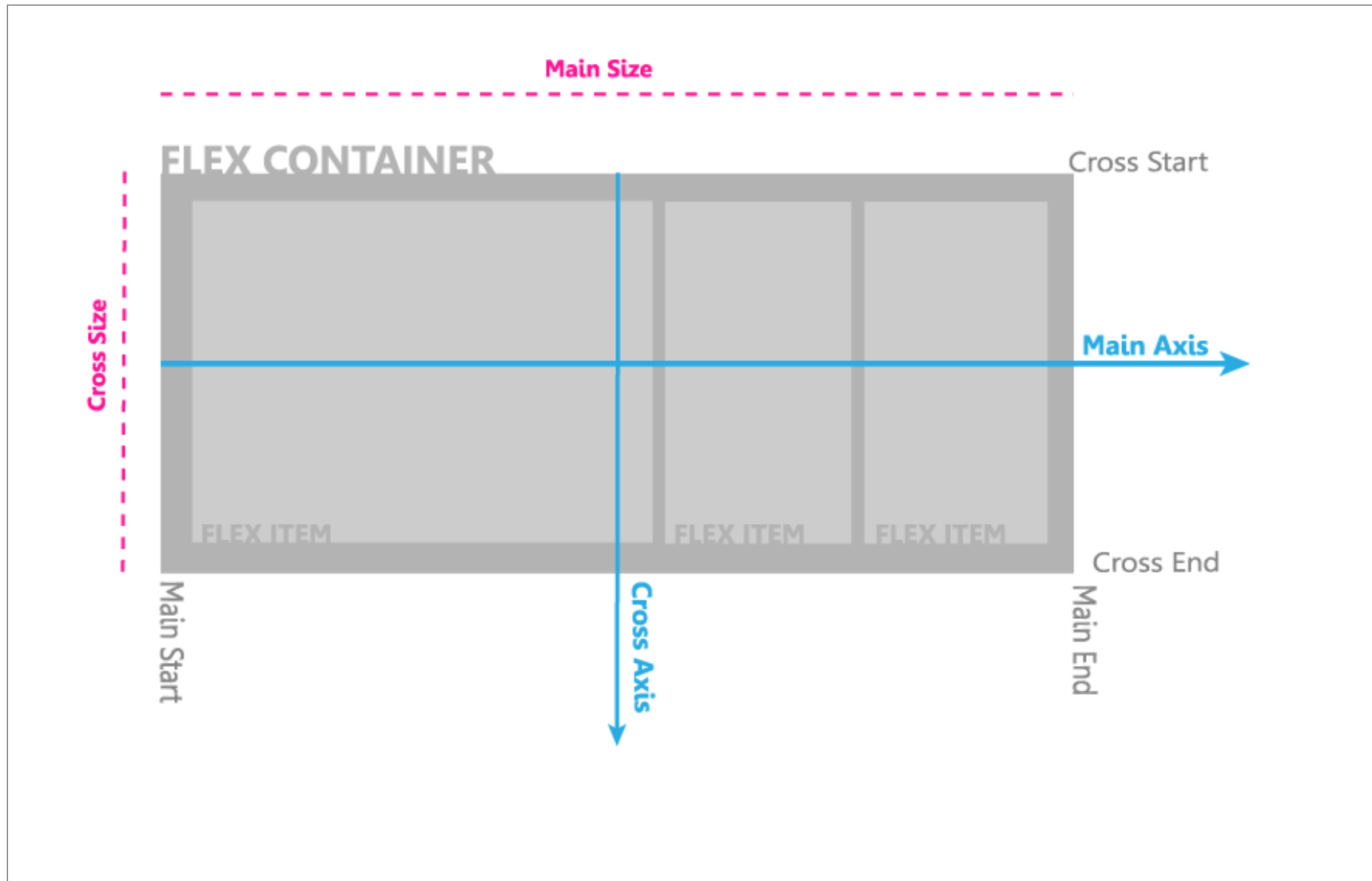
CSS Flexible Box Layout Module 📄 - CR

Method of positioning elements in horizontal or vertical stacks. Support includes all properties prefixed with `flex`, as well as `display: flex`, `display: inline-flex`, `align-content`, `align-items`, `align-self`, `justify-content` and `order`.

| Current aligned | Usage relative | Date relative | | Apply filters | Show all | ? |

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Blackberry Browser	Opera Mobile *	Chrome for Android	Firefox for Android
					10-11.5							
		[1] 2-21 ⊟	[1] 4-20 ⊟	[1] 3.1-6 ⊟	12.1	[1] 3.2-6.1 ⊟						
6-9		[3] 22-27	21-28	6.1-8	15-16	7-8.4		[1] 2.1-4.3 ⊟		12		
[2][4] 10 ⊟	12-17	28-62	29-69	9-11.1	17-55	9-11.4		4.4-4.4.4	[1] 7 ⊟	12.1		
[4] 11	18	63	70	12	56	12	all	67	10	46	70	63
		64-65	71-73	TP								

DEFINITIONS

- **Main axis** - main axis of a flex container is the primary axis along which flex items are laid out
- **Cross axis** - axis perpendicular to the main axis

FLEX-DIRECTION: ROW/ROW-REVERSE

HTML   CSS   Result

```css
.row {
  flex-direction: row;
}
.reverse {
  flex-direction: row-reverse;
}
```

```
1    2    3    4    5
```

```
5    4    3    2    1
```

Resources   1× 0.5× 0.25×   Rerun

FLEX-DIRECTION: COLUMN/COLUMN-REVERSE

```css
.column {
  flex-direction: column;
}
.reverse {
  flex-direction: column-reverse;
}
```

1
2
3
4
5

5
4
3
2

FLEX-WRAP

HTML   CSS   Result

```
.first {
  flex-wrap: nowrap;
}
.second {
  flex-wrap: wrap;
}
.third {
  flex-wrap: wrap-reverse;
}
```

```
1      2      3      4      5      6
```

```
1      2      3      4      5

6      7      8      9      10
```

Resources   1×   0.5×   0.25×   Rerun
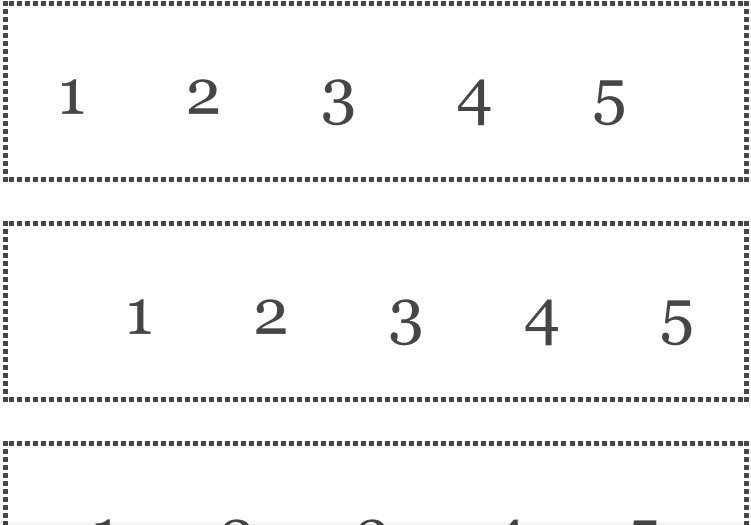
JUSTIFY-CONTENT

HTML | CSS | Result

```css
.first {
  justify-content: flex-start;
}
.second {
  justify-content: flex-end;
}
.third {
  justify-content: center;
}
.fourth {
  justify-content: space-between;
}
.fifth {
  justify-content: space-around;
}
```

```
1    2    3    4    5
```

```
1    2    3    4    5
```

```
1    2    3    4    5
```

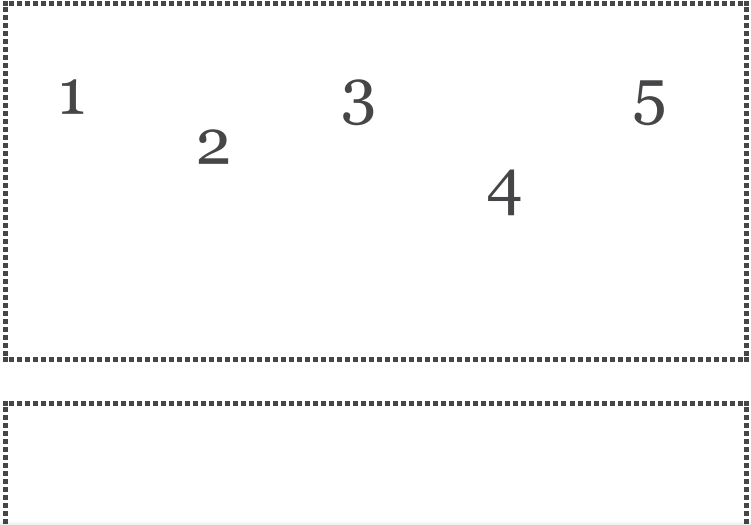Resources   1×   0.5×   0.25×   Rerun

ALIGN-ITEMS

   Result   EDIT ON

```css
.first {
  align-items: flex-start;
}
.second {
  align-items: flex-end;
}
.third {
  align-items: center;
}
.fourth {
  align-items: baseline;
}
.fifth {
  align-items: stretch;
}
```

1   3   5
   2   4

Resources   1×   0.5×   0.25×   Rerun

CUSTOM ORDER

HTML	CSS	Result

```css
.flex div:nth-child(1) {
  order: 4;
}

.flex div:nth-child(2) {
  order: 3;
}

.flex div:nth-child{3} {
  order: 5;
}

.flex div:nth-child(4) {
  order: 1;
}
```
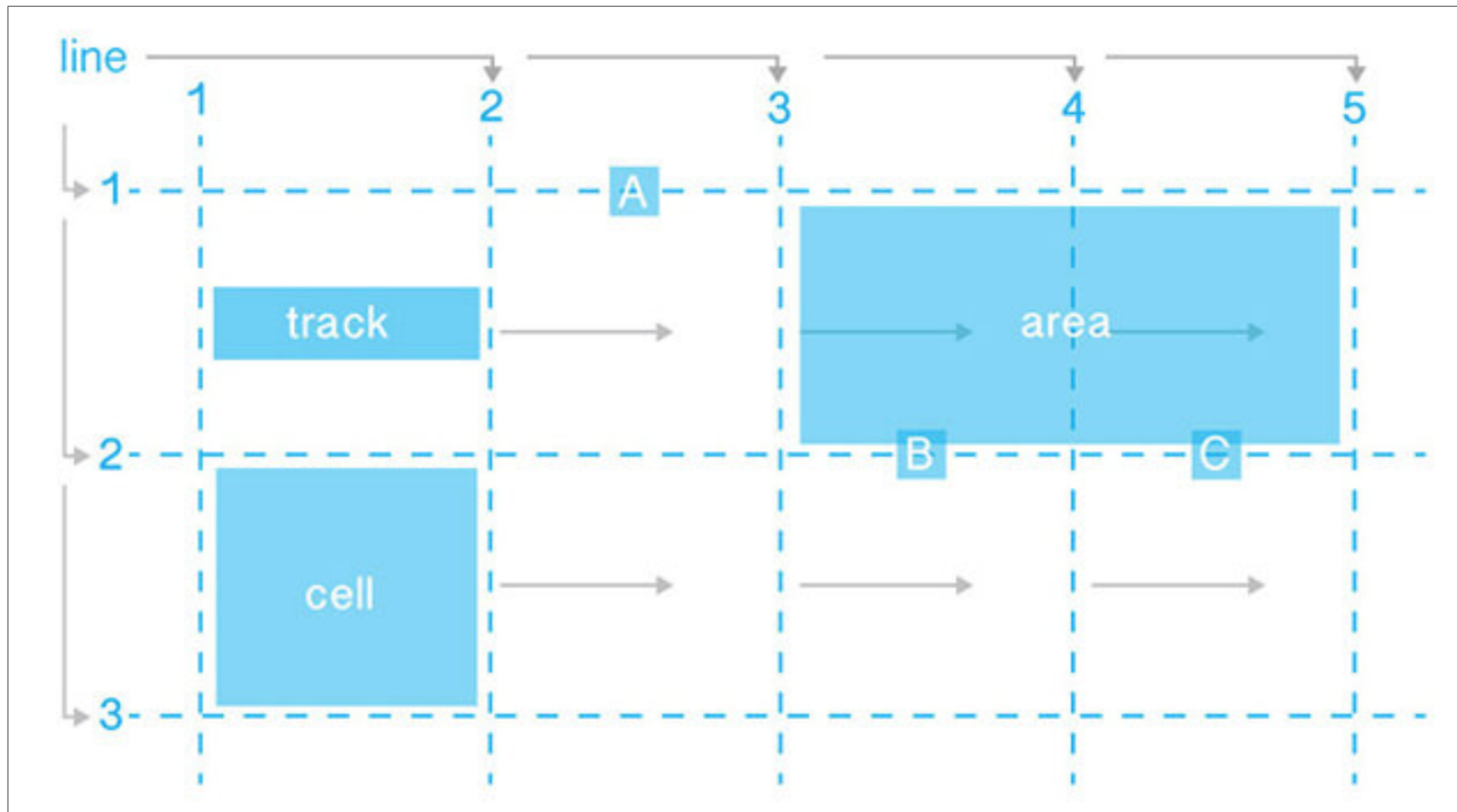
3   4   5   2   1

Resources   1×   0.5×   0.25×   Rerun

HELPFUL LINKS

- **Guide to flexbox**
- **playground**
- **and one more**
- **Flex Froggy**

GRID

CSS Grid Layout

IE	Edge	Firefox	Chrome	Safari	iOS Safari	Chrome for Android
9	15	59	66	10.1	10.3	
10	16	60	67	11	11.0-11.2	
11	17	61	68	11.1	11.3-11.4	67
	18	62	69	12	12	
		63	70	TP		

Полная поддержка   Частичная поддержка   С префиксом   Не поддерживается

Данные с сервиса caniuse.com

DEFINITIONS

- **Grid container** - element on which display: grid is applied
- **Grid item** - children (i.e. direct descendants) of the grid container
- **Grid line** - dividing lines that make up the structure of the grid
- **Grid track** - space between two adjacent grid lines
- **Grid cell** - space between two adjacent row and two adjacent column grid lines
- **Grid area** - total space surrounded by four grid lines
- **Fraction (fr)** - fraction of the free space of the grid container; calculated after any non-flexible items

DISPLAY: GRID & DISPLAY: INLINE-GRID

HTML   CSS   Result

```
<div class="container">
  <div class="grid">grid</div>

  <br/>
  <span>inline-element</span>
  <div class="inline-grid">inline-grid</div>
  <span>inline-element</span>
</div>
```

grid

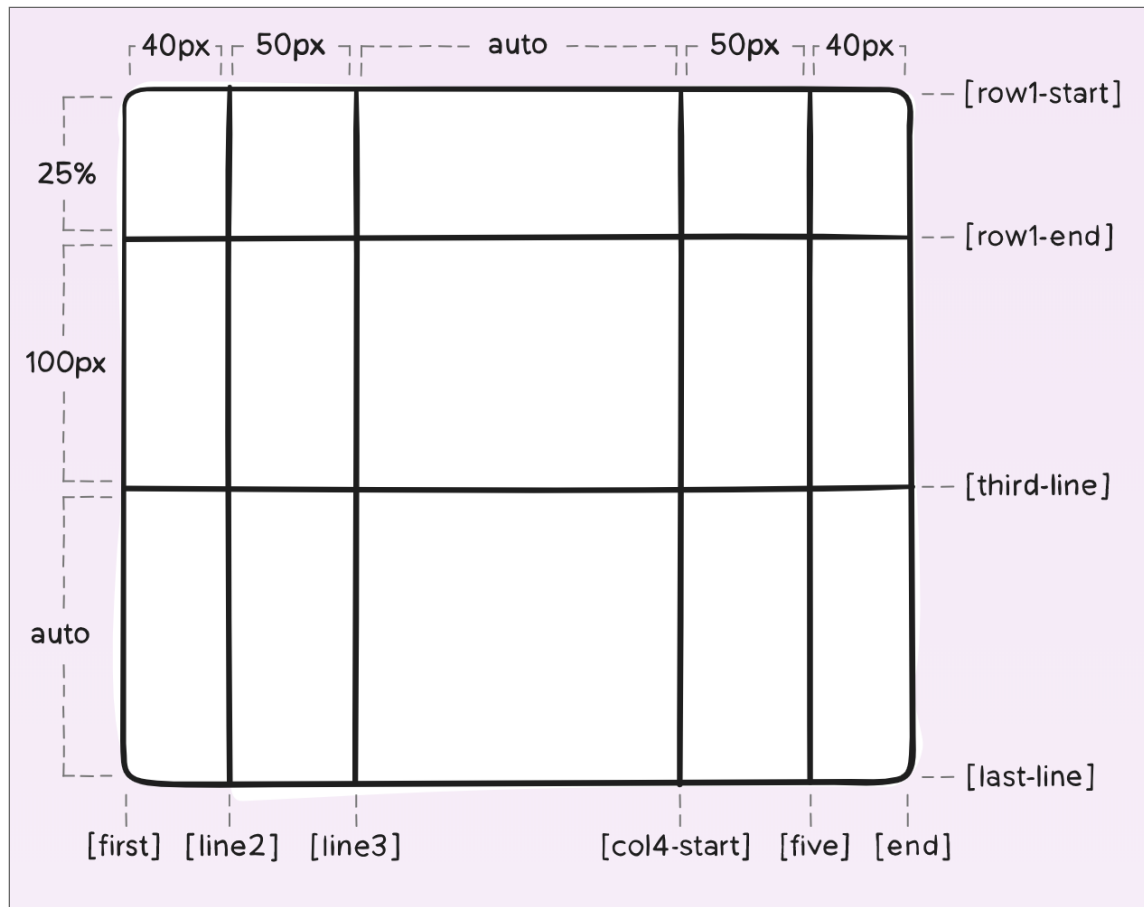inline-element   inline-grid   inline-

element

Resources   1×   0.5×   0.25×   Rerun

GRID-TEMPLATE-COLUMNS & GRID-TEMPLATE-ROWS

GRID-AREA & GRID-TEMPLATE-AREAS

HTML CSS   Result

```css
.container {
  display: grid;
  grid-template-areas:
    "header aside"
    "main aside"
    "main aside"
    "footer footer";
}
.header {
  grid-area: header;
}
.main {
  grid-area: main;
}
.aside {
  grid-area: aside;
}
.footer {
```

grid header

main content

Lorem, ipsum dolor sit amet consectetur adipisicing elit. Odit eos quas cum nostrum ipsam, nihil sit ut tenetur recusandae accusamus aut quasi explicabo quibusdam ex ullam minima architecto perspiciatis saepe.

aside content

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ratione pariatur quod asperiores assumenda, vero aliquid perspiciatis sequi repellat obcaecati rem.

grid footer

Resources   1×   0.5×   0.25×   Rerun

GRID-COLUMN-GAP + GRID-ROW-GAP = GRID-GAP

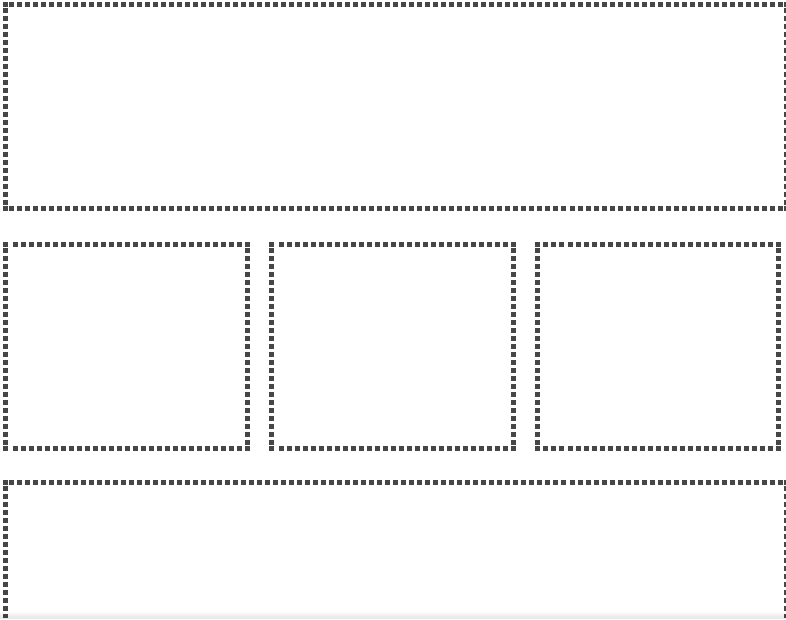HTML   CSS   Result

```css
.container {
  display: grid;
  grid-column-gap: 10px;
  grid-row-gap: 5%;
/*   grid-gap: 5% 10px; */
}
```

Resources   1×   0.5×   0.25×   Rerun

ALIGNING CONTENT ALONG ROW AXIS

justify-self & justify-items

```css
.container {
  display: grid;
/*  justify-items: start; */
/*  justify-items: end; */
/*  justify-items: center; */
/*  justify-items: stretch; */
}
.a, .b, .c {
/*  justify-self: end; */
}
.d, .e, .f {
/*  justify-self: start; */
}
.g, .h, .i {
/*  justify-self: center; */
}
.j, .k, .l {
/*  justify-self: stretch; */
}
```

Aligning Content Along Row Axis

A	B	C
D	E	F
G	H	I
J	K	L

Resources   1×   0.5×   0.25×   Rerun

ALIGNING CONTENT ALONG COLUMN AXIS

align-self & align-items

```css
.container {
  display: grid;
/*  align-items: start; */
/*  align-items: end; */
/*  align-items: center; */
/*  align-items: stratch; */
}
.a, .b, .c {
/*  align-self: start; */
}
.d, .e, .f {
/*  align-self: end; */
}
.g, .h, .i {
/*  align-self: center; */
}
```

Aligning Content Along Column Axis

A	B	C
D	E	F
G	H	I

THE ORDER PROPERTY IN GRID LAYOUT

```css
.b, .j {
  order: 2;
}
.a, .i {
  order: 3;
}
```

The order Property in Grid Layout

C	E	D	F
G	E	H	B
J	A	I	

ATTENTION!

First of all, Flexbox is about positioining and alignment element within container
Much more powerful than floats & tables, but for different use

HELPFUL LINKS

- **Guide**
- **one more guide**
- **playground**
- **Grid Garden**
- **css selectors cheatsheet**

BOOTSTRAP

HOW TO INSTALL

in head

```
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css
```

on the bottom of the body

```
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" ... ></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" ... ></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js" ... ></script>
```

click me

HOW TO USE

just add class to your markup

Example

RESPONSIVE

MAIN STEP TO RESPONSIVE

- Fluid grids
- Flexible images
- Media queries

FLUID GRIDS

- CSS Grid + Flex box
- **<u>Bootstrap</u>**
- **<u>Zurb Foundation</u>**
- **<u>Skeleton</u>**

FLEXIBLE IMAGES

HTML   CSS   Result

```css
.container {
  margin: 0 auto;
  width: 50%;
  border: 1px solid #552200;
}

.picture {
  width: 100%;
  height: auto;
```


Resources   1× 0.5× 0.25×   Rerun

HTML   CSS   Result

```css
.container {
  width: 50%;
  margin: 0 auto;
  border: 1px solid #449911;

}
.picture {
  max-width: 100%;
  height: auto;
}
```


Resources   1× 0.5× 0.25×   Rerun

HTML	Result

```
<picture>
<source
srcset="https://placeimg.com/500/500/animals"
media="(max-width: 600px)">
<source
srcset="https://placeimg.com/800/600/animals">
<img src="https://placeimg.com/800/600/animals"
alt="">
</picture>
```

Resources   1×   0.5×   0.25×   Rerun

HTML	CSS	Result

```
.container {
  width: 100%;
  height: 600px;
  background-image:
url(https://placeimg.com/400/300/nature);
}

@media only screen and (min-width: 600px) {
  .container {
    background-image: url(
```

Resources   1×   0.5×   0.25×   Rerun

MEDIA QUERY

target CSS rules based on screen size, device-orientation or display-density etc.

@media [feature] ([condition]) { CSS rule }

```css
div{
  width: 50%;
  height: 350px;
  margin: 0 auto;
  background-color: #444444;
}

@media screen and (min-width: 640px) {
  div {
    background-color: pink;
```

helpful link

SUMMARY

- Emmet make your life easy **(click me)**
- Semantic
- Normalize
- Clearable naming
- Absolute units for font, relative for size and margin/padding
- Use grid for layout and flex-box for content
- FFM

all link clickable, you just need to find them ;)

codepen examlpe also active: click on right-top corner (plz, don't save your changes)

GOOD LUCK!