



Internet Technologies

[GitHub - GDSE-62-64-Project-Zone/CSS-Basics](#)

[diagrams.net](#)

CSS

- Cascading Style Sheets (CSS)
- Styling Sheet Language එකක්
- Website වලට styles දැනෙන use කරයි.

Element වලට styles apply කරන්න පුලුවන් කිරීම,

- Inline styles

```
<h1 style="color: red"> Hello CSS </h1>
```

- Meta styles

```
<style>
  p{
    color: green;
  }
</style>
```

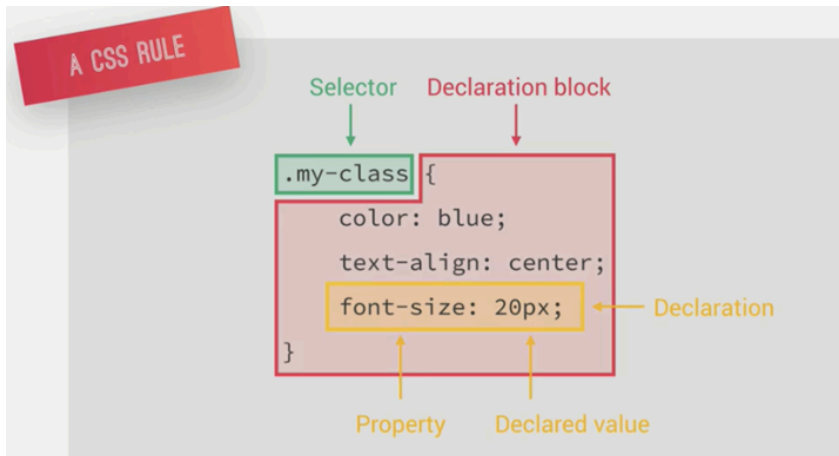
- External style

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

CSS Rule

- Style declarations groupe එකක්.

- Element වලට styles දාන්න Rules use කරයි.



- Declaration එකක නියමන්නේ property එකක් හා value එකක්.

property: value;

CSS Basic Selectors

→ 1. Class selector

- අපිට ඕන group වනෙ වනෙම එකකැරගේ select කරන්න

```
.selected{
  border: 1px solid red;
}
```

```
<h1 class="selected"> Hello CSS </h1>
```

→ 2. Id selector

- uniquely select කරන්න
- එක id එකක් , එක පාරයි use කරන්න සුලුවත්

```
#selectByID{
  color: blueviolet;
}
```

```
<p id="selectByID"> Lorem ipsum dolor </p>
```

→ 3. Universal selector

- ඔස්කොම එකපාර select කරන්න

```
*{  
  color: red;  
  font-family: 'Edu NSW ACT Foundation', cursive;  
}
```

→ 4. Type selector

- Element type එක අනුව select කරන්න
- ex : paragraph වලට විතරක් style එක apply කරන්න ↓

```
p{  
  color: black;  
  background-color: orange;  
}
```

→ 5. Attribute selector

- attribute එකකින් element select කරන්න
- ex : id attribute එක තියනෙ හැමතැනකම style එක apply කරනෙ ↓

```
[id]{  
  color: chartreuse;  
}
```

```
<p id="a"> Lorem ipsum dolor</p>  
<p id="b"> Lorem ipsum dolor</p>
```

- specially කියන්නන් පුලුවන්
- ex: දැන් "a" කියන id එකට විතරයි apply වන්නෙ

```
[id = "a"]{  
  color: chartreuse;  
}
```

→ Grouping selector ,

```
h1, h2 {  
  color: gold;  
}
```

```
#special, .selected{  
  background-color: orange;  
}
```

Pseudo classes (:)

:hover

Mouse එකින් point කළාම වෙනස්කම් apply වෙනවා

```
<h1> Hello </h1>  
<h1> Hello </h1>  
<h1> Hello </h1>
```

```
:hover{  
  color: red;  
}  
  
*{  
  color: blue;  
}
```

Hello

Hello

Hello

:nth-child(x)

ලමා parent ගෙ ඉඳින(x) වනේ child select කරයි.

```
<section>
  <h1> Hello </h1>
  <h1> Hello </h1>
  <p> Hello </p>
  <h1> Hello </h1>
  <h1> Hello </h1>
</section>
```

```
:nth-child(3) {
  color: red;
}
```

Hello
Hello
Hello
Hello
Hello

:first-child & :last-child

```
<section>
  <h1> Hello </h1>
  <h1> Hello </h1>
  <h1> Hello </h1>
  <h1> Hello </h1>
  <h1> Hello </h1>
  <h1> Hello </h1>
</section>
```

```
:first-child {
  color: red;
}

:last-child {
  color: green;
}

* {
  color: blue;
}
```

Hello
Hello
Hello
Hello
Hello
Hello

- parent element එකේ නැතිනම් last child අයුරෙන් බෑ.

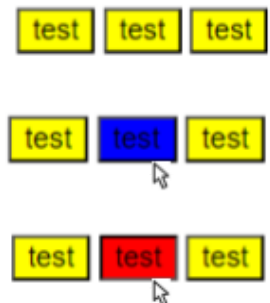
:focus & :active

```
<button type="button"> test </button>
<button type="button"> test </button>
<button type="button"> test </button>
```

```
button{
  background-color: yellow;
}

button:focus{
  background-color: blue;
}

button:active{
  background-color: red;
}
```



:visited

```
<a href="https://www.facebook.com" target="_blank"> FaceBook
</a>
```

```
<br>
<a href="https://www.google.lk" target="_blank"> Google </a>
<br>
<a href="https://www.amazon.com" target="_blank"> Amazon
</a>
```

```
a:visited{
  color: red;
}
a{
  color: blue;
}
```

FaceBook
Google → Google
Amazor → Amazon

- Visit කරපු site පෙන්වනේ. (history එකෙන්)

:before & :after

```
<h1>Hello</h1>
<h2>Hello</h2>
```

```
h1:before{
  content: "@ ";
}
h2:after{
  content: " #";
}
```

@ Hello

Hello #

Pseudo elements (::)

::first-letter

```
<h1>Hello CSS</h1>
```

```
h1::first-letter {
  color: red;
}
```

Hello CSS

::first-line

```
<p>Lorem ipsum dolor  
sit amet, consectetur  
adipiscing elit.  
Adipisci eligent</p>
```

```
p::first-line{  
    color: red;  
}
```

Lorem ipsum dolor sit
atque, aut dolores duc
quo quod rerum sint s

::selection

```
p::selection{  
    color: wheat;  
    background-color:  
black;  
}
```

e, aut dolores ducim

::marker

```
li::marker{  
    content: "@";  
    background-color:  
red;  
}
```

@item 1
@item 2
@item 3
@item 4

::placeholder

```
input::placeholder{
```

```
color: greenyellow;
}
```

text here

:: file-selector-button

```
<input type="file">
```

```
input::file-selector-button{
  color: black;
  background-color: lawngreen;
  border: 10px;
  border-radius: 10px;
}
```

Choose File No file chosen

CSS Combinators

Selector 2ක් හෝ කිහිපයක් combine කරලා styles apply කරන්න use කරයි.

01. Child Combinator
02. Descendant Combinator
03. General Siblings Combinator
04. Adjacent Combinator

01. Child Combinator (>)

```
selector1 > selector2 { style
properties }
```

- Direct child කනෙකේ නම් style එක apply වෙනෙය්.(Direct only)

```
<section>
  <h1>Hello</h1>
  <div>
    <h1>Hello</h1>
  </div>
</section>
<h1>Hello</h1>
```

```
section > h1 {
  color: red;
}
```

(section එකකට
Direct Child වුනු
හැම h1 එකක්ම)

Hello

Hello

Hello

02. Descendant Combinator (Space)

```
selector1 selector2 {
  /* property declarations */
}
```

- Parent ගෙ ඇතුළේ කොහේ හිටියත් style එක apply වෙනේ. (Direct & Indirect)

```
<section>
  <h1>Hello</h1>
  <div>
    <h1>Hello</h1>
  </div>
</section>
<h1>Hello</h1>
```

```
section h1 {
  color: red;
}
```

(section එකකට
Child වුනු හැම
h1 එකක්ම)

Hello

Hello

Hello

03. General Siblings Combinator (~)

```
former_element ~ target_element { style  
properties}
```

- Same root එකේ තියෙන ඕන.
- එක ලග තියෙන ඕන නෑ.

```

<section>
  <h4>Hello</h4>
  <p>inside section 01</p>
  <h4>Hello</h4>
  <h4>Hello</h4>
  <div>
    <h4>Hello</h4>
    <h4>Hello</h4>
    <p>inside section 01 > div</p>
    <h4>Hello</h4>
  </div>
</section>

```

```

h4~h4 {
  color: red;
}

```

(h4 න් පස්සෙ
Same root එකේ
තියෙන් හැම h4
ක්ම)

Hello

inside section 01

Hello

Hello

Hello

Hello

inside section 01 > div

Hello

04. Adjacent Combinator (+)

former_element + target_element { style
properties }

- Same root එකේ තියෙන ඕන.
- එක ලග තියෙන ඕන.

```

<section>
  <h4>Hello</h4>
  <p>inside section 01</p>
  <h4>Hello</h4>
  <h4>Hello</h4>
  <div>
    <h4>Hello</h4>
    <h4>Hello</h4>
    <p>inside section 01 > div</p>
    <h4>Hello</h4>
  </div>
</section>

```

```

h4+h4 {
  color: red;
}

```

(h4 න් පස්සෙ
Same root එකේ,
එක ලග
තියෙන් හැම h4 ක්ම
)

Hello

inside section 01

Hello

Hello

Hello

Hello

inside section 01 > div

Hello

Cascade

Process of combining different stylesheets and resolving conflicts between different css rules and declarations. When more than one rule applies to a certain element.

එක element එක target කරගෙන,
same property එකට, different selectors වලින්, different value එකක්
apply කරන්න යොමු, conflict එකක් ඇතිවෙන්නේ.
ඒ conflict එක විසඳන mechanism එක → cascade

```
*{  
    color: red;  
}  
h1{  
    color: green;  
}  
#mainHeader{  
    color: orange;  
}  
.mainHeader{  
    color: deeppink;  
}  
[id]{  
    color: blueviolet;  
}  
:first-child{  
    color: aqua;  
}
```

Specificity order

Specificity

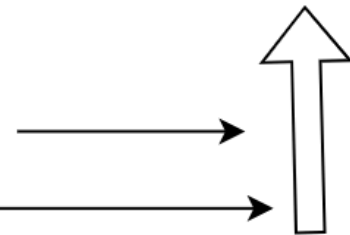
Source Order

01. Inline Styles = 1000

02. ID selector = 100

03. Class , Pseudo Class, Attribute Selector = 10

04. Type selector, Pseudo Element = 1



එකම priority එක නිසාම කට්ටි හමුවන source order එකට යනේ.
(priority එක → Stylesheet එකේ පහළ priority එක වැඩි)

https://www.w3schools.com/css/css_specificity.asp

```
/*1+100+10+1+100+10+10+1+100+10+1+100+10+1+100+10 = 565*/
```

```
body#mainBody.bodyClass>section#mainSection.mainClass:nth-child(2)>section#subSection.subClass>div#container.containerClass>h1#mainHeader.mainHeader{  
    color: yellow;  
}
```

!important

```
color: darkmagenta !important;
```

!important දැමීමෙන් specificity order එක බලපෑම නැ. ඒකට වැඩි priority එක දෙනේ.

Cascade resolving steps

1. !important
2. Specificity order

Specificity

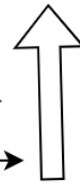
Source Order

01. Inline Styles = 1000

02. ID selector = 100

03. Class , Pseudo Class, Attribute Selector = 10 →

04. Type selector, Pseudo Element = 1 →



3. Source order

(file එකේ style apply කරපු order එක. යටින්ම දාපු style එක apply වන්නේ)

CSS values

CSS values තියන්න පුළුවන් විදි,

1. Integers (10)

2. Numbers (10.2)

3. Dimensions (10px 10em 10rem)

- Length (10px 10cm)

- Absolute (10px 10cm 10mm)

- Relative (10rem 10vh 10vw) - parent අනුව size වෙනස් වෙනවා

- Angle (10deg)

- Time (10s 10ms 10h)

- Resolution (10dpi)

4. Percentage (50%)

5. Keywords (red)

6. Functions (rgba())

Measurement units

Absolute measurement units

- **parent** මත බලපාන්නේ නෑ **size** එක

- px

- cm
- mm
- pc
- pt

Relative measurement units

- parent අනුව size වනෙයි වනෙය්

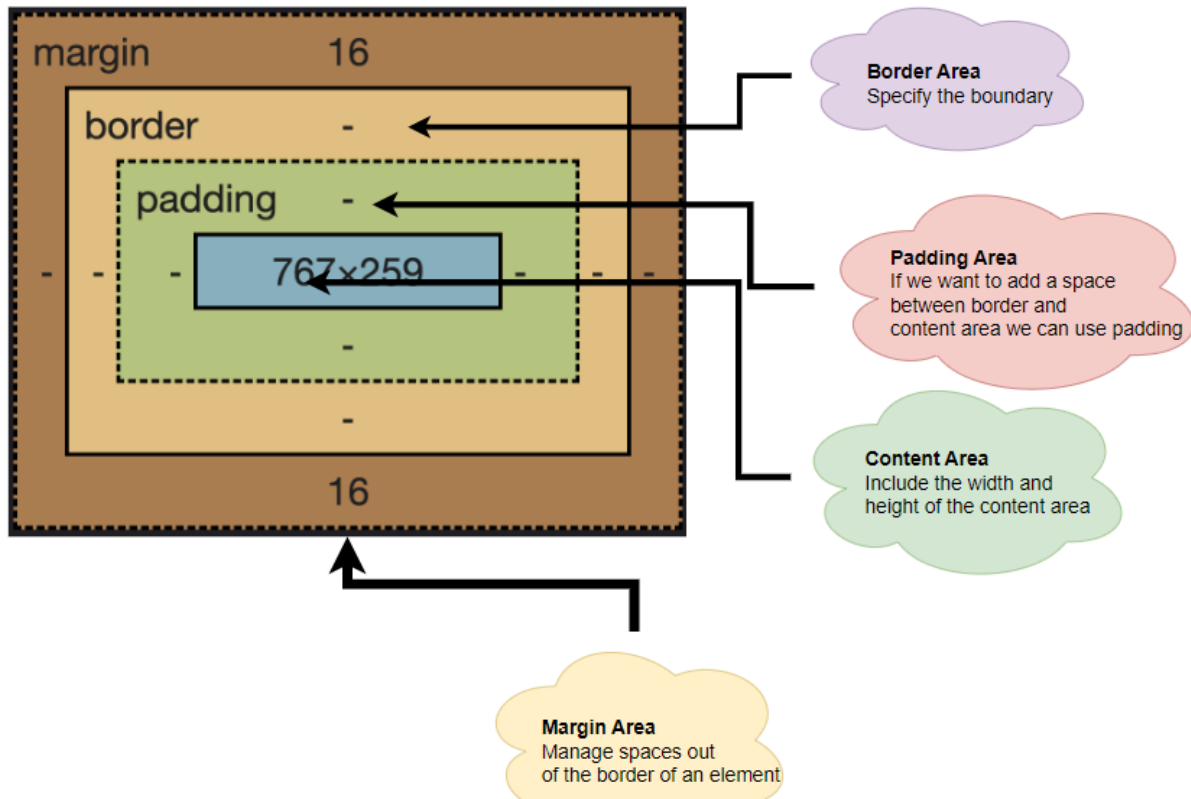
- em (nearest parent ගේ font size එකට relative)
- rem (root element එකේ font size එකට relative)
- vh (viewport height අනුව වනෙයි වනෙය්)
- vw (viewport width අනුව වනෙයි වනෙය්)
- fr (fragment , grid හද්දි row column බෙදෙන ඕන පරමාණ දේදි)

Box model

Elements වල structure එක, size එක, spacing, සහ borders calculated වී ඇති ආකාරය විස්තර කරන, web design වල මූලික සංකල්පයකි.

- Element එකක content area එකේ ඉඳන් margin area එක වනෙකම්, spaces manage කරන්න box model එක use කරයි.
- හැම element එකකටම box model එකක් තියනෙය්.

CSS Box Model



Content Area

Element එකේ content වල width & height

Border Area

Element එකේ Boundary එක

Padding Area

Content area එකයි Border එකයි අතර space එක

```
padding: 10px 20px 30px 40px;  
(top, right, bottom, left) (clockwise)
```

Margin Area

Border එකේ පිට space එක

CSS Positions

[CSS Positions](#) ← Example code

html elements අපිට ඔහු විදියට move කරන්න use කරයි.

- Static position ← default
- Relative position
- Absolute position
- Fix position
- Sticky position

position: static; ← default

position: relative; ← position එක වෙනස් වෙනේ, flow එකට බලපෑමක් නෙවේ.

position: absolute; ← flow එක වෙනස් වෙනේ. (parent කනෙකට සාපේක්ෂව)

- Parent ගෙ width & height ගැලපෙන්න ඔහු,
- position static නෙවේ තියෙන්න ඔහු.
එහෙම නැත්නම් parent විදියට ගන්නෙ නෑ.

position: fixed; ← viewport එකට සාපේක්ෂව පිහිටයි. (page එක scroll කරන එකම තැන ඉන්නෙ)

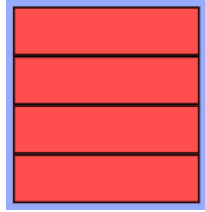
position: sticky; ← Scroll කරද්දි, එක් point එකක් දක්වා normal position එකෙ ඉන්නෙ, ටිකේම් point එකකට පස්සෙ viewport එකට සාපේක්ෂව පිහිටයි(එක තැන).

Inline & Block elements

[Block level & Inline elements with display:](#) ← Example code

HTML element වර්ග 2යි,

- Block level element
 - Height එක හැඳින්වෙ content එකෙ height එක අනුව, width එක හැඳින්වෙ ලගම ඉන්න parent ගෙ width එක අනුව
 - By default එකම line එකෙ තියෙන්නෙ නෑ (පහලට තියෙන්නෙ)



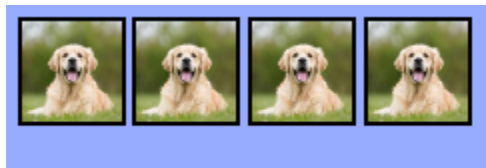
- ex: <div>, <p>, <h1>, <article>, <section>, <figure>, <footer>

- Inline elements

- Element එකේ width එකයි height එකයි අනුව height එකයි width එකයි හදාගන්නේ.

(width, height box model එකේ apply වන්නේ නෑ)

- By default එකම line එකේ තියන්නේ



- ex : , <button>, <a>, <code>, <cite>, , <input>

display: none; ← display එකේ අයිති කරන්න පුළුවන්

display : inline; ← inline කරන්න පුළුවන්

display : block; ← block කරන්න පුළුවන්

display: inline-block; ← width height properties තියාගෙන inline කරන්න පුළුවන් (inline, block mix එකක්)

normalize.css

- CSS rules set එකක් (normalize.css)
- Web Browser එකේ apply කරන default styles අයිති(reset/normalize) කරන්න use කරයි.

[🔗Normalize.css: Make browsers render all elements more consistently.](#)

z-index

Default → z-index : 0 ;

Flow එකේ උඩට / යටට ගනියන්න use කරයි.

අඩු values නම් flow එකේ යටට

වැඩි values නම් flow එකේ උඩට

transform



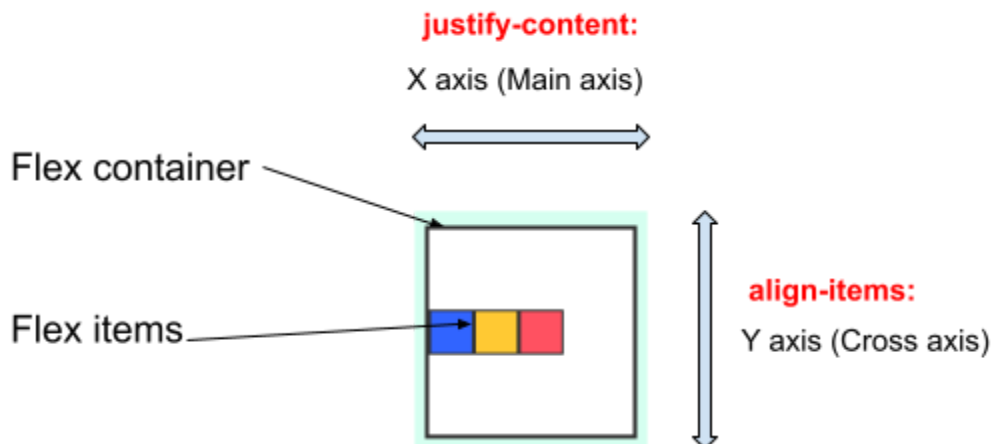
transform: rotate(44deg);

display : flex; (CSS Flexbox)

[CSS Flexbox](#) ← Example code

https://github.com/GDSE-62-64-Project-Zone/CSS-Basics/blob/master/32_Flex_Box_Advanced.html

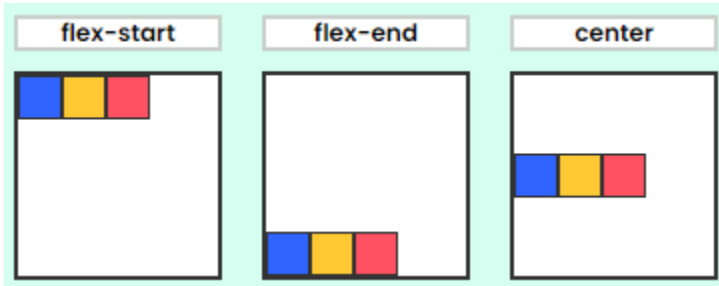
Flex is the value of css display . By using display flex in parent element, child elements automatically align like column or row with auto width and auto height



Flex Container Properties

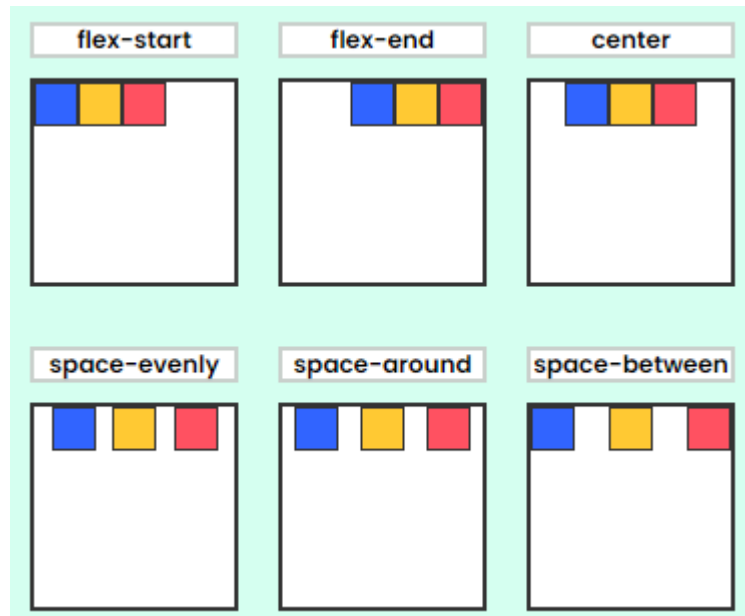
- Manage content according to y-axis (cross-axis)

```
align-items: center;
( flex-start | flex-end | center )
```



- Manage content according to x-axis (main-axis)

```
justify-content: space-around;
( flex-start | flex-end | center | space-evenly | space-around |
space-between)
```

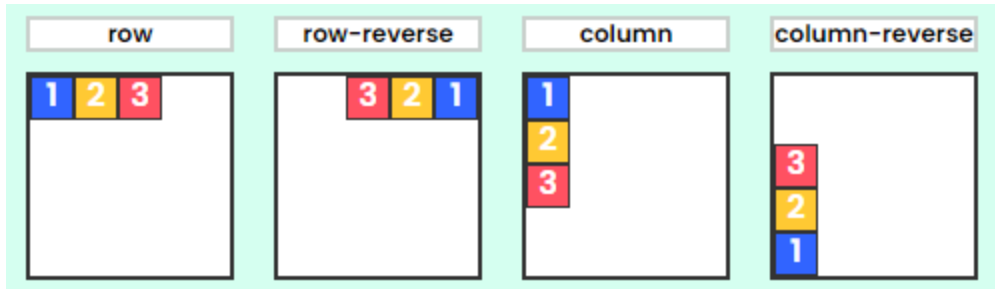


space-evenly (equal spaces)

space-around (place spaces around left and right flex items and then the rest of space between the center element)

space-between (remove the left and right flex items spaces and put all spaces between the center flex elements)

```
flex-direction: row;
( row | row-reverse | column | column-reverse )
```



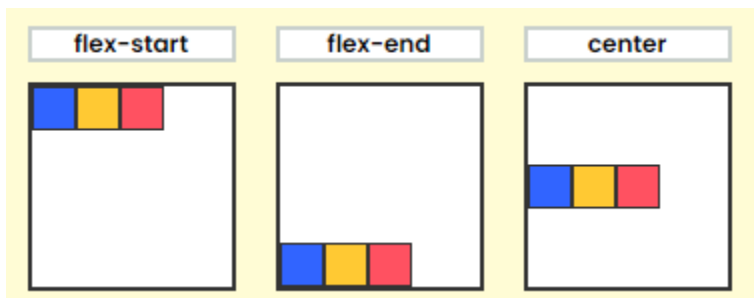
Flex Items Properties

- flex-grow:** ← unit less proportion value



```
div:nth-child(1){
  flex-grow: 0;
}
div:nth-child(2){
  flex-grow: 5;
}
div:nth-child(3){
  flex-grow: 1;
}
```

- align-self:** flex-start;
(flex-start | flex-end | center) container එකට සාපේක්ෂව items වල positions වෙනස් කරනවා.



- Flex container එකක් ඇතුළේ තියෙන ඒවාගේ width height container එකේ පිට යන දිශේ නෑ. (**default** flex-shrink: 1; නිසා)



- `flex-shrink: 0;` ← 0 කරාම flex container එකේ පිට යනේ



- `order:` ← flex items wala order eka wenas karanna



```
div:nth-child(1) {
  order: 3;
}

div:nth-child(2) {
  order: 1;
}

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

- `flex-wrap` (Container එකට Apply කරන්න) ← container එකේ එලියට පතින ඒවා ඇතුලට දාන්නේ. (`flex-shrink: 0;` වලො තියනේ ඕන item එක)



`display: grid;`

`grid-template-rows: 1fr 1fr 1fr 1fr 1fr;`

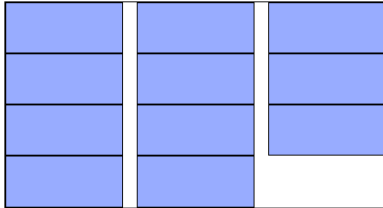
`grid-template-columns: 1fr 1fr 1fr`

- `fr` ← fragment (Relative measurement unit ပုံစံ)

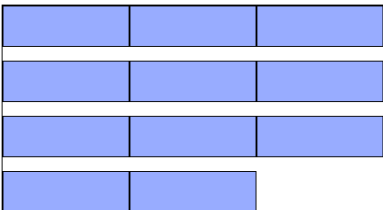
`grid-template-rows: repeat(5, 1fr);`

`grid-template-columns: repeat(6, 1fr);`

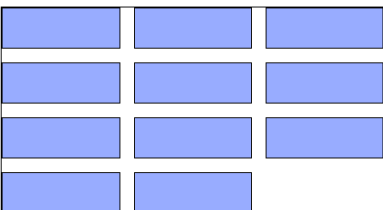
- `grid-column-gap: 10px;`



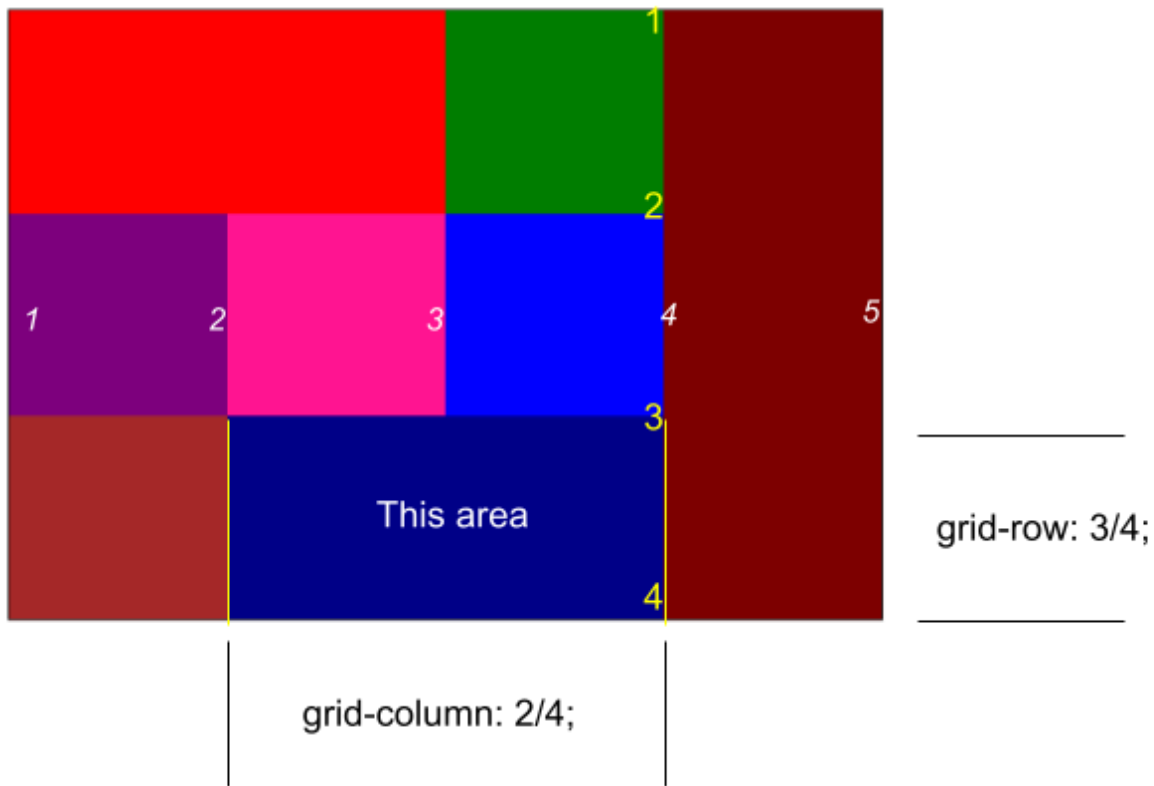
- `grid-row-gap: 10px;`



- `grid-gap: 10px;`



- `grid-column: 1/3;` (1 - start point, 3 - end point)



Transition

html element එකක, එක css state එකක ඉඳන් තව state එකකට smoothly change කරන්න පුළුවන් කිරීමයි.

```
div {
  height: 100px;
  width: 100px;
  background-color: #960707;

  transition: 2s;
}

div:hover {
  border-radius: 100%;
  left: 100px;
  transition: 2s;
}
```

transition-duration: 2s; or **transition: 2s;** ← transition එකට සහ
time එක
transition-property: left, border-radius; ← property 1ක කට විතරක්
transition apply කරන්න

Animation

Time duration එකක් ඇතුළත css rules set එකක් smoothly වෙනස් කරන්න පුළුවන්
කරමුණ. ඔබේ වාර ගානක් run කරන්න පුළුවන්.

keyframes

specified time duration එකක් ඇතුළත element එකකට apply වන්නේ ඕන style
හෝ animations sequence එකක් define කරන්න භාවිතා වේ.

ex:

- Two keyframes

```
section{
  width: 100px;
  height: 100px;
  background: yellow;
  position: relative;
  left: 0;

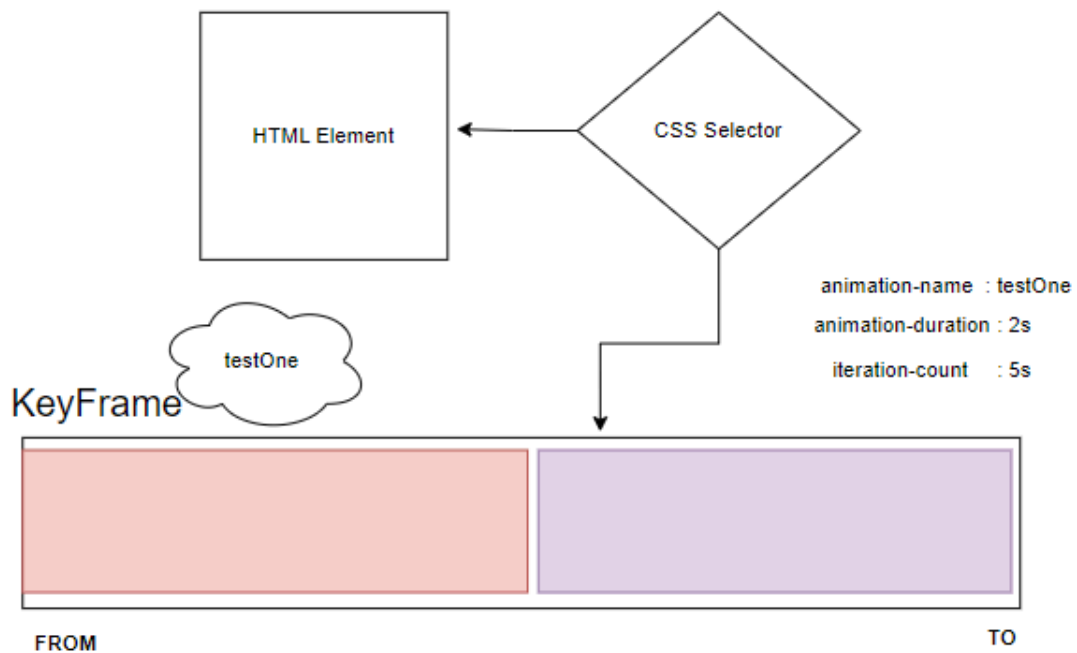
  animation:testOne;
  animation-duration: 2s;
  animation-iteration-count: 2;
}

@keyframes testOne {
  from{
    width: 100px;
    height: 100px;
    background: red;
  }
  to{
    width: 100px;
    height: 100px;
```

```

background: green;
border-radius: 100%;
left: 300px;
}
}

```



- More keyframes

```

@keyframes testOne {
  0% {
    left: 0;
    top: 0;
    background: brown;
  }
  25% {
    top: 0;
    left: 500px;
  }
}

```

```

50% {
  top: 400px;
  left: 500px;
}
75% {
  top: 400px;
  left: 0;
}
100% {
  left: 0;
  top: 0;
  background: green;
}
}

```

Media query

The @media rule, introduced in CSS2, made it possible to define different style rules for different media types.

Examples: You could have one set of style rules for computer screens, one for printers, one for handheld devices, one for television-type devices, and so on.

- Media query danna kalin daanna oon,
`<meta content="width=device-width initial-scale=1" name="viewport">`

```

@media (width: 1440px) {
  h1{
    color: red;
  }
}

```

Add wenne 1440px width ekata witarai

```
@media (min-width: 1440px) {
  h1{
    color: red;
  }
}
```

1440px ha eeta wadi wiata apply weno.
(apply wena min width eka 1440px, eeta wadi eewata kohomath apply weno)

```
@media (max-width: 1440px) {
  h1{
    color: red;
  }
}
```

1440px ha eeta adu eewata apply weno.
(apply wena max width eka 1440px, eeta adu eewata kohomath apply weno)

```
@media all {
  h1{
    color: red;
  }
}
```

Hama device ekatama apply weno

```
@media (max-width: 768px) and (min-width: 320px) {
  h1 {
    color: red;
  }
}
```

Deepu range ekata adalawa apply weno
(768px ha eeta adu, 320px ha eeta wadi device seratama apply weno)

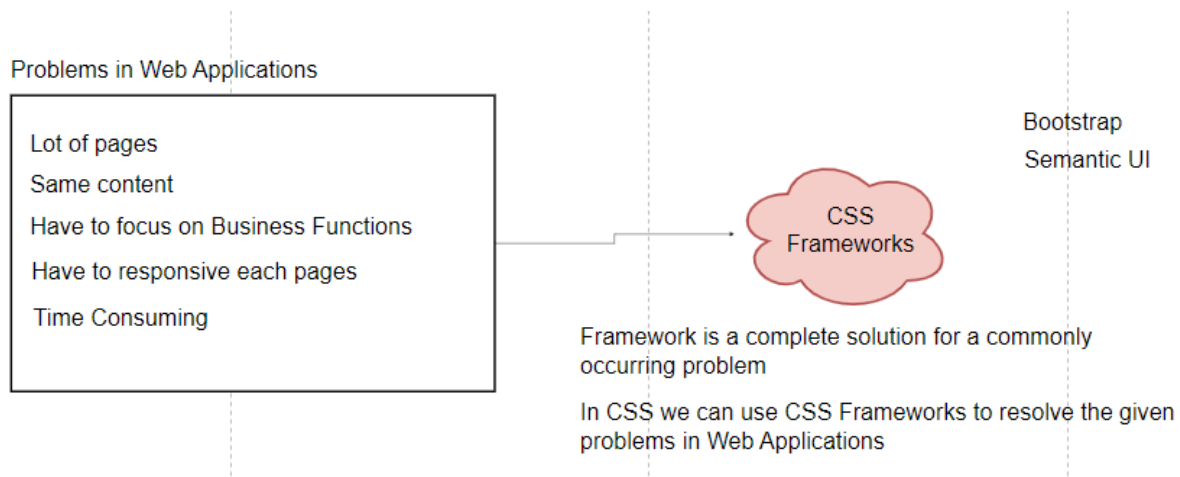
=====

Framework

Common Problem එකකට නියත complete solution එකකි.

CSS Framework

CSS වල complexity එක අඩු කරන්න නියත complete solution එකකි.



Bootstrap

CSS Framework ekak

<https://getbootstrap.com/docs/5.3/getting-started/introduction/>

`<section class="container">` ← depatten gap ekak eno

Responsive

<https://getbootstrap.com/docs/5.3/layout/grid/>

	xs <576px	sm ≥576px	md ≥768px	lg ≥992px	xl ≥1200px	xxl ≥1400px
Container max-width	None (auto)	540px	720px	960px	1140px	1320px
Class prefix	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-	.col-xxl-
# of columns	12					
Gutter width	1.5rem (.75rem on left and right)					
Custom gutters	Yes					
Nestable	Yes					
Column ordering	Yes					

.col eka podima eka

☹️ → pattata apply wenne

col-12 col-sm-12 col-md-4 col-lg-4 col-xxl-4

(extra small, small device waladi pahalata wateno row 2k widiyata,
anit eewage 4/12 space ekaka set weno)