Intro Of Web Site Design

Fundamental point in Web site creation

Present by Khin Maung Win

Before learning in web site creation

- Understand environment of business flow.
- Understand technology changing and accept it.
- Create clean and meaningful design.
- UI/UX rule apply on design.
- Follow basic writing flow base on google or bootstrap design.
- Understand information Architecture of system.

Important Technique / Tool in web site creation

Some important Technique show in below

Layout	Content	Forms	Components	Utilities
1. Container 2. Grid 3. Columns 4. Utilities	1. Typography 2. Images 3. Table	1. Forms Control 2. Select 3. Checks and Radio 4. validation	1. Button 2. Card 3. Carousel 4. Modal 5. Progress 6. Scrollspy	1. Flex 2. Overflow 3. Interactions

Grid, Column, Row, Container

Grid - is basic design structure of web site creation. Using Container , row , column to layout and align content. It is built with flexbox and fully responsive.

Column - is apply inside row and twelve column system using in one row.

Row - is horizontal structure and all content are apply with column.

Container - is like big box in website design and include rows, column, and other design and tools.

Flex Box

What is flexbox and what for?

-flexible Box that can be changing size depend on row and container design.

-flexbox is a css layout design constructed using the display:flex property

-easy way to control align and distribute space among items in a container.

-flexbox can be apply on web design and if we don’t know devices and screen sizes.

-prevent overflow design error in website.

Flexbox property

Basic usage with flexbox

- Create div -
- Inside container div
-
-
-
- Inside <head></head> - write
- <style></style> add and
- Call class name from body -> with .left-side{} , .right-side{} and .main-container{}

CSS simple

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Simple of FlexBox</title>
    <style type="text/css">
        .container {
            display: flex;
            width: 100%;
            height: 400px;
        }
        .left-side, .right-side {
            width: 50%;
            border: 1px sold black;
            padding:10px;

        }
        .main-container {
            display: block;
            border: 1px solid red;padding: 10px;
        }

    </style>
</head>
```

HTML simple

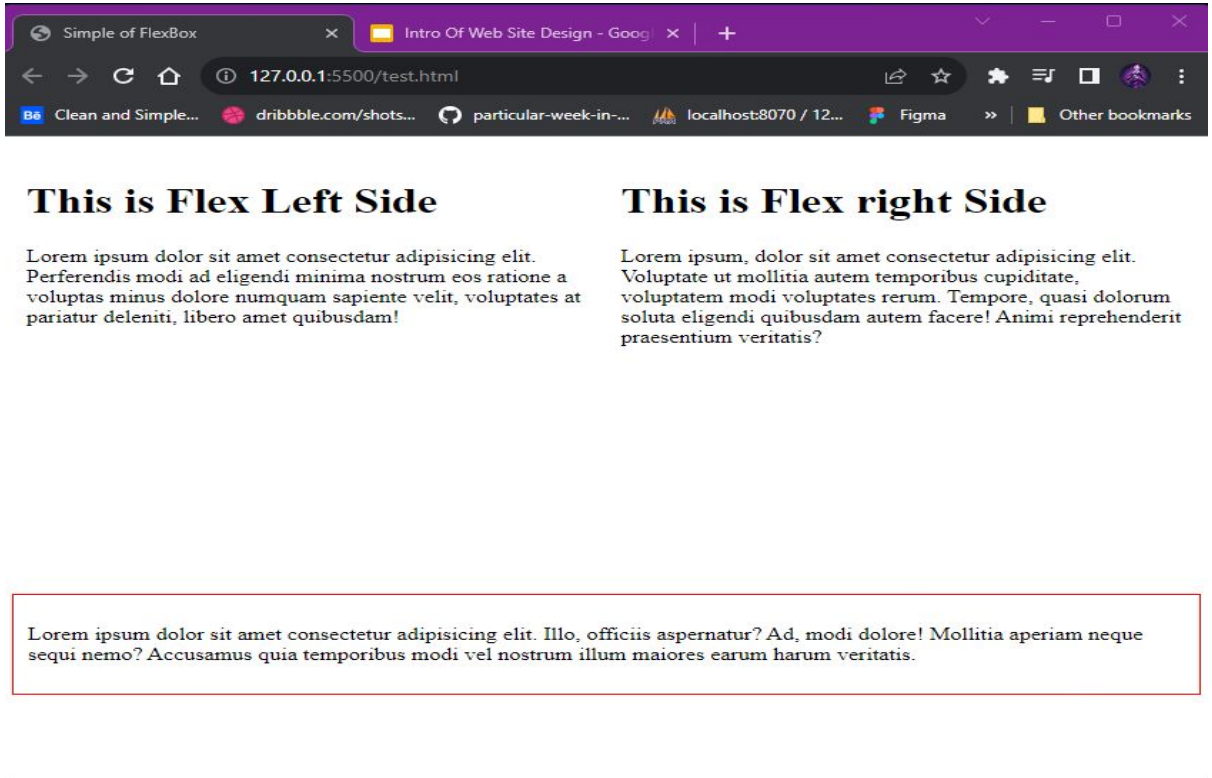
```
    </style>
</head>
<body>
    <div class="container">
        <div class="left-side">
            <h1>This is Flex Left Side</h1>
            <p>Lorem ipsum dolor sit amet consectetur adipisicing elit.
                Perferendis modi ad eligendi minima nostrum eos ratione a voluptas minus dolore numquam sapiente velit,
                voluptates at pariatur deleniti, libero amet quibusdam!
            </p>

        </div>
        <div class="right-side">
            <h1>This is Flex right Side</h1>
            <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit.
                Voluptate ut mollitia autem temporibus cupiditate, voluptatem modi voluptates rerum. Tempore, quasi dolorum soluta eligendi
                quibusdam autem facere! Animi reprehenderit praesentium veritatis?
            </p>
        </div>

    </div>
    <div class="main-container">
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Illo,
            officiis aspernatur? Ad, modi dolore! Mollitia aperiam neque sequi nemo?
            Accusamus quia temporibus
            modi vel nostrum illum maiores earum harum veritatis.</p>
    </div>

</body>
</html>
```

Result


Simple of FlexBox
Intro Of Web Site Design - Goog
127.0.0.1:5500/test.html
Clean and Simple...
dribbble.com/shots...
particular-week-in-...
localhost:8070 / 12...
Figma
Other bookmarks
This is Flex Left Side
Lorem ipsum dolor sit amet consectetur adipisicing elit. Perferendis modi ad eligendi minima nostrum eos ratione a voluptas minus dolore numquam sapiente velit, voluptates at pariatur deleniti, libero amet quibusdam!
This is Flex right Side
Lorem ipsum, dolor sit amet consectetur adipisicing elit. Voluptate ut mollitia autem temporibus cupiditate, voluptatem modi voluptates rerum. Tempore, quasi dolorum soluta eligendi quibusdam autem facere! Animi reprehenderit praesentium veritatis?
Lorem ipsum dolor sit amet consectetur adipisicing elit. Illo, officiis aspernatur? Ad, modi dolore! Mollitia aperiam neque sequi nemo? Accusamus quia temporibus modi vel nostrum illum maiores earum harum veritatis.

flex-direction

Flex-direction - direction of flex items are placed in the flex container

flex -direction can be defined horizontal row and vertical columns.

row- left to right

```
display:flex;
flex-direction: row;
```

row-reverse - right to left

```
display:flex;
flex-direction: row-reverse;
```

column - top to bottom

```
display:flex;
flex-direction: column;
```

column-reverse - bottom to top

```
display:flex;
flex-direction: column-reverse;
```

row test (1234)

1 2 3 4

row-reverse test (1234) change to (4321)

4 3 2 1

column test (1234) change to (1234) top to bottom

1
2
3
4

column test (1234) change to (4321) bottom to top

4
3
2
1

Flex-wrap

flex-warp - prevent overflow design in website and auto change to another line

nowrap(default) - all flex items will be on one line.

wrap - flex items will be change to multiple lines when space is not enough(top to bottom).

```
.row{
display: flex;
flex-wrap: wrap;
}
```

wrap-reverse - flex items will be change to multiple line from bottom to top.

```
.row{
display: flex;
flex-wrap: wrap-reverse;
}
```

flex-flow

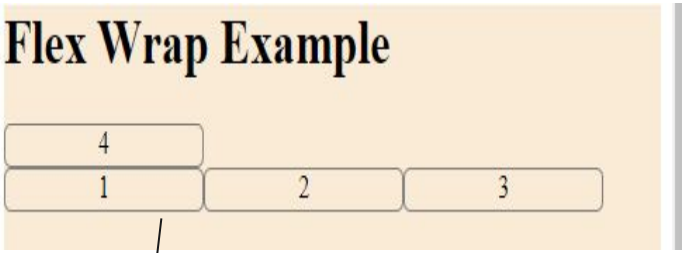
flex-flow - direction of flex and define with flex-wrap properties

```
.row{
display: flex;
flex-flow: column wrap;
}
```

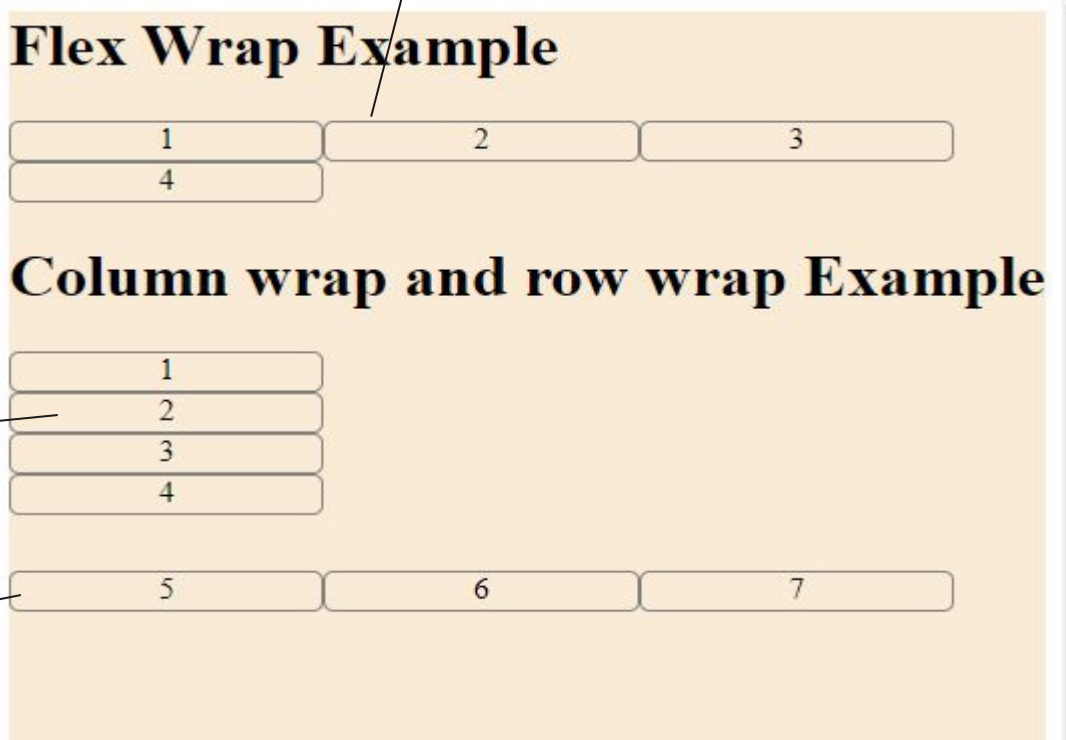
Row wrap

```
.row{
display: flex;
flex-flow: row wrap;
}
```

Flex-wrap example



wrap-reverse



wrap

column wrap

row wrap

justify-content

justify-content - control position of flex items alignment along main axis of flex container.

Flex-start -flex items start from start point of flex container.

Flex-end - flex items start from end point of flex container.

Center - all flex items stay in center of flex container.

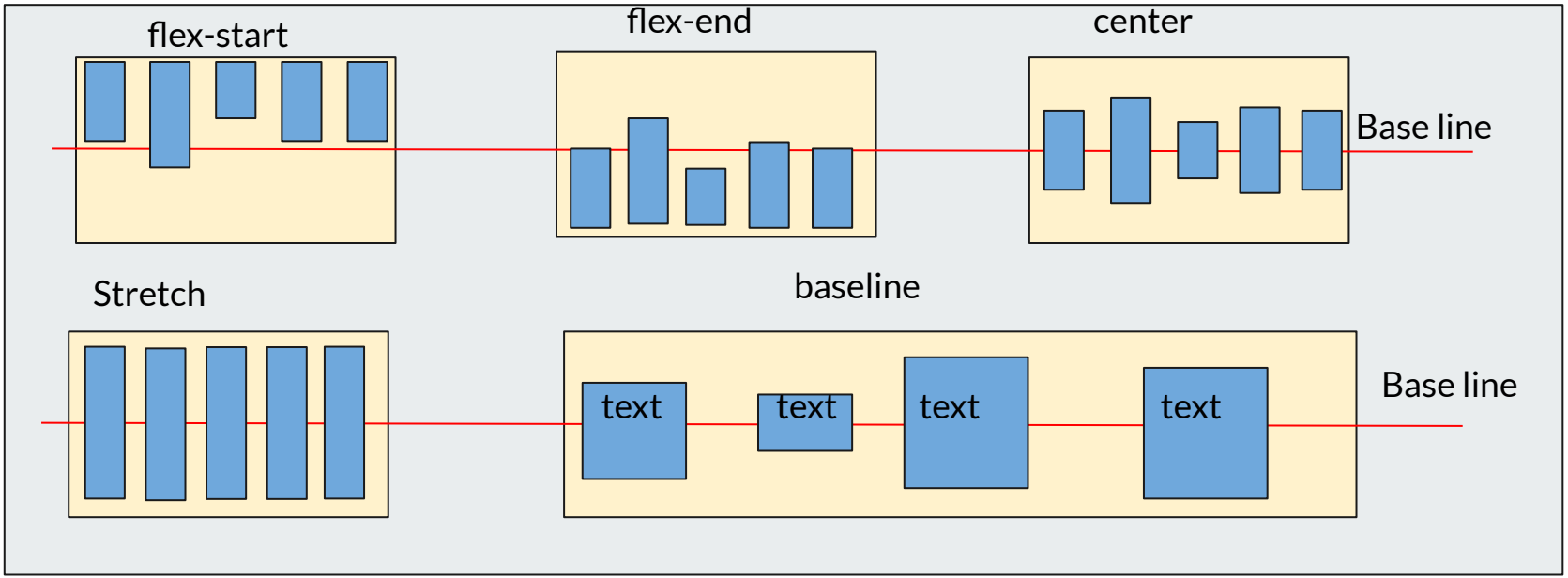
Space-between- create equal space between flex item that base on flex container space.

Space-around - items are evenly distributed in the line with equal space around then

Space-evenly - items are distributed so that spacing between any two items is equal.

Align-items

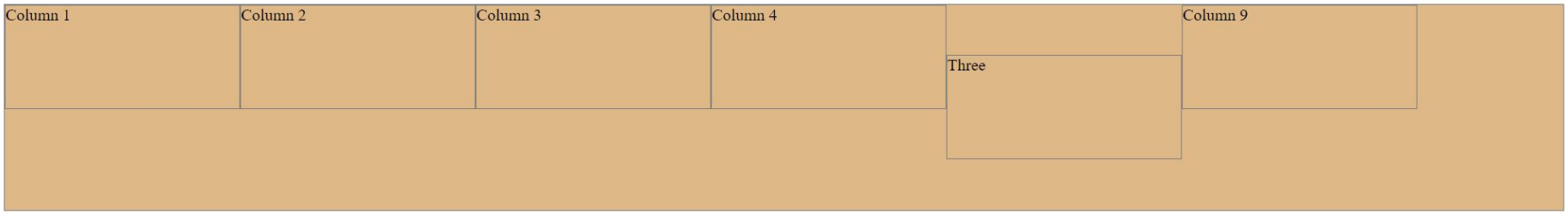
Align-items property is default arrangement of flex items along the cross-axis within the current line.



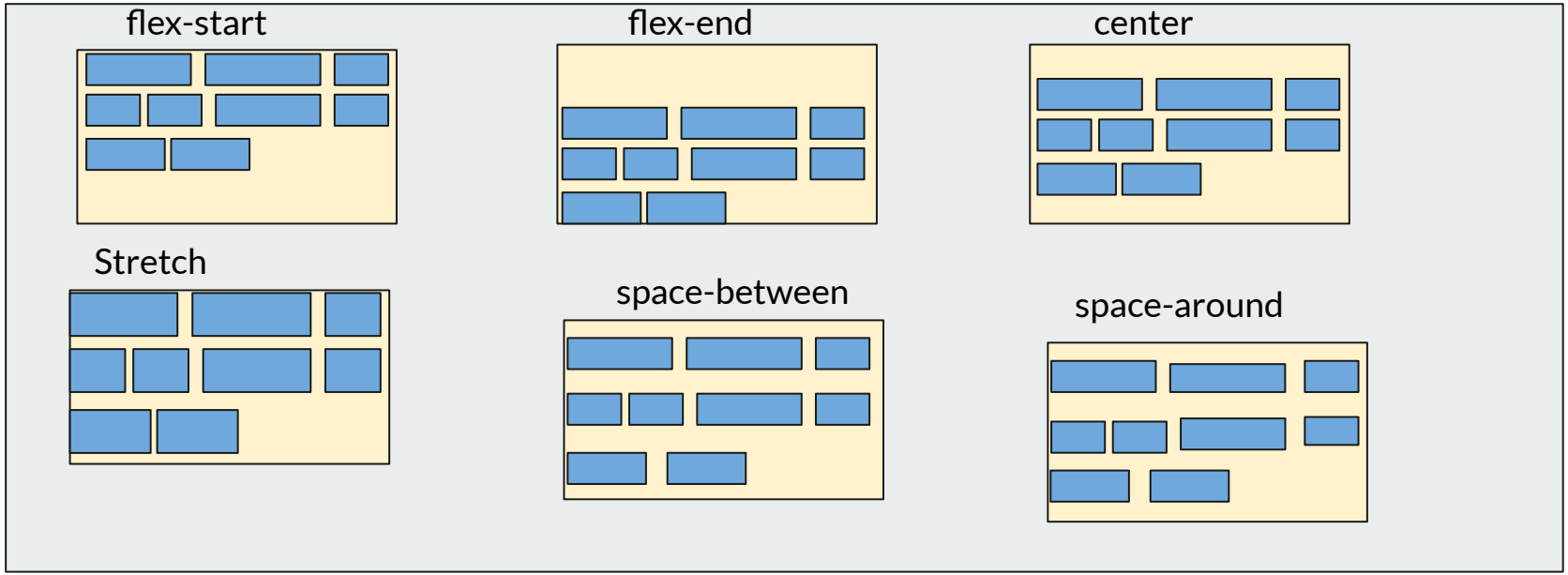
```css
.box {
  display: flex;
  align-items: flex-start;
  height: 200px;
  width: 100%;
  border:1px solid gray;
  background-color: burlywood;
}
```

```css
.box .selected{
    align-self: center;
}
.flex-item, .selected{
 border: 1px solid gray;
 height:100px;
 width: 15%;
}
```

Align-items Example



Align-content



Align Content Example


```
<style type="text/css">

    .flex-container{
        display: flex;
        background-color: antiquewhite;
        height: 200px;
        width: 100%;
        flex-flow: column wrap;
        /* align-content: flex-start;
        align-content: flex-end;
        align-content: center;
        align-content: SpacingCenter;
        align-content: space-around; */
        align-content: flex-end;

    }
    .flex-item{
        flex-basis: 20%;
        background-color: yellowgreen;
        width: 100px;
        margin:30px;
        text-align: center;
        letter-spacing: 1px;
    }
```

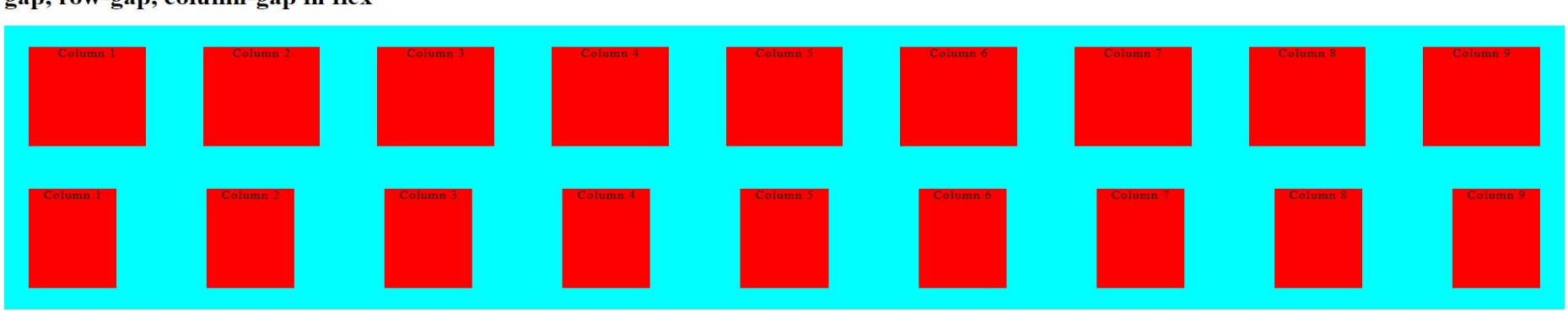
gap, row-gap, column-gap

gap - short configuring between rows and columns within a flex container

-simultaneous adjustment for the spacing between flex items in horizontal and vertical direction.

row-gap - vertical spacing between flex item and flex container row

column-gap - spacing between columns of flex items and flex container.



```css
<style type="text/css">
    .flex-container {
     display: flex;
     background-color: cyan;
     height: 200px;
     width: 100%;
     gap: 10px;
      gap:10px 20px;
     row-gap: 10px;
     column-gap:10px;
    }
    .flex{
        display: flex;
        background-color: cyan;
        height: 200px;
        width: 100%;
        gap:10px 20px;
        row-gap:30px;
        column-gap: 50px;

    }

    .flex-item{
     flex-basis: 20%;
     background-color: red;
     width: 100px;
     margin:30px;
     text-align: center;
     letter-spacing: 1px;
    }


</style>
```

```html
        <h1>gap, row-gap, column-gap in flex</h1>

        <div class="flex-container">
            <div class="flex-item">Column 1</div>
            <div class="flex-item">Column 2</div>
            <div class="flex-item">Column 3</div>
            <div class="flex-item">Column 4</div>
            <div class="flex-item">Column 5</div>
            <div class="flex-item">Column 6</div>
            <div class="flex-item">Column 7</div>
            <div class="flex-item">Column 8</div>
            <div class="flex-item">Column 9</div>
        </div>
        <div class="flex">
            <div class="flex-item">Column 1</div>
            <div class="flex-item">Column 2</div>
            <div class="flex-item">Column 3</div>
            <div class="flex-item">Column 4</div>
            <div class="flex-item">Column 5</div>
            <div class="flex-item">Column 6</div>
            <div class="flex-item">Column 7</div>
            <div class="flex-item">Column 8</div>
            <div class="flex-item">Column 9</div>
        </div>

</body>
```

Order property

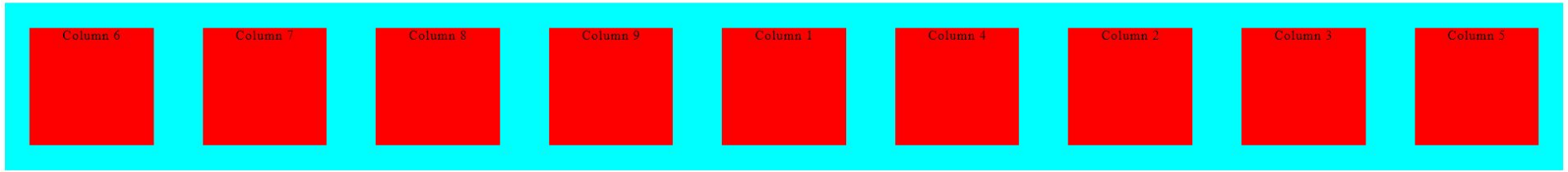
Order - reversing the visual display order of flex items.

Organizing items into specific groups.

Achieve both customized visual layouts and structured item grouping

.unorder list is first then order:1, order:2, order:3, order:4 and order:5

gap, row-gap, column-gap in flex



```css
<style type="text/css">
    .flex-container {
      display: flex;
      background-color: cyan;
      height: 200px;
      width: 100%;
      flex-direction: row;
    }
    .flex-container :nth-child(1){
        order: 1;
    }
    .flex-container :nth-child(2){
        order: 3;
    }
     .flex-container :nth-child(3){
         order: 4;
    }
    .flex-container :nth-child(4){
        order: 2;
    }
    .flex-container :nth-child(5){
        order: 5; /* high priority */
         /* :nth-child(5)  (5) is column posi
         /* 678914235 */
    }
```

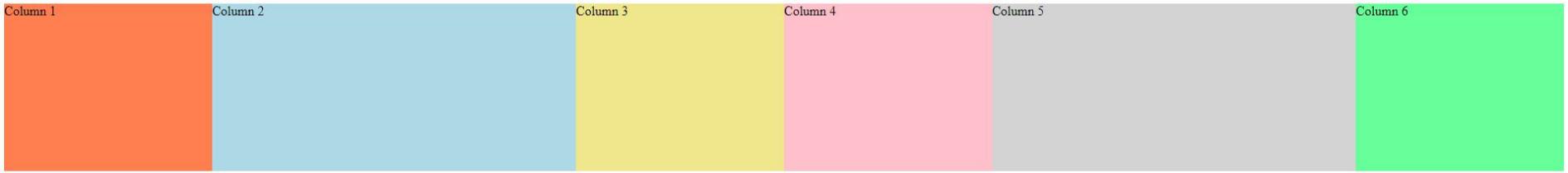
```html
<h1>gap, row-gap, column-gap in flex</h1>

<div class="flex-container">
    <div class="flex-item">Column 1</div>
    <div class="flex-item">Column 2</div>
    <div class="flex-item">Column 3</div>
    <div class="flex-item">Column 4</div>
    <div class="flex-item">Column 5</div>
    <div class="flex-item">Column 6</div>
    <div class="flex-item">Column 7</div>
    <div class="flex-item">Column 8</div>
    <div class="flex-item">Column 9</div>
</div>
```

flex-grow

Flex-grow - determines the extent to which an item can expend relation to other flexible items within the container.

Flex-grow Css



```css
<style type="text/css">
    .flex-container {
      display: flex;
      background-color: cyan;
      height: 200px;
      width: 100%;
      flex-direction: row;
    }
    .flex-container div:nth-of-type(1){
        flex-grow: 1;
        background-color: coral;
    }
    .flex-container div:nth-of-type(2){
        flex-grow: 2;
        background-color: lightblue;
    }
    .flex-container div:nth-of-type(3){
        flex-grow: 1;
        background-color: khaki;
    }
    .flex-container div:nth-of-type(4){
        flex-grow: 1;
        background-color: pink;
    }
    .flex-container div:nth-of-type(5){
        flex-grow: 2;
        background-color: lightgrey;
    }
    .flex-container div:nth-of-type(6){
        flex-grow: 1;
        background-color: #66ff99;
    }
</style>
```

```html
<h1>Flex-grow Css</h1>

<div class="flex-container">
    <div class="flex-item">Column 1</div>
    <div class="flex-item">Column 2</div>
    <div class="flex-item">Column 3</div>
    <div class="flex-item">Column 4</div>
    <div class="flex-item">Column 5</div>
    <div class="flex-item">Column 6</div>

</div>
```

flex-shrink

Flex-shrink - define how much a flexbox item should shrink if there's not enough space available

Apply with flex property

Flex: is shorthand property

Flex:110; is same like

Flex-grow:1;

Flex-shrink:1; larger number faster shrink than smaller number

Flex-basis:0;

```
#main div {
  flex-grow: 1;
  flex-shrink: 2;
  flex-basis:500px;
}

#main div:nth-of-type(3) {
  flex-shrink: 1;
}
#main div:nth-of-type(5) {
  flex-shrink: 2;
}
</style>
</head>
<body>

<h1>The flex-shrink Property</h1>

<div id="main">
  <div style="background-color: coral;"></div>
  <div style="background-color: lightblue;"></div>
  <div style="background-color: khaki;"><h4>This is shrink target</h4></div>
  <div style="background-color: pink;"></div>
  <div style="background-color: lightgrey;"><h4>This is shrink target</h4></div>
</div>
```

The flex-shrink Property

Thank you