Introduction to Responsive Web Design | Cheat Sheet

1. Bootstrap Grid System

Bootstrap Grid System is a collection of reusable code snippets to create responsive layouts. It is made up of **containers**, **rows**, and **columns**.

It uses a **12 column system** for layouting. We can create up to 12 columns across the page.

1.1 Container

The purpose of a container is to hold rows and columns.

Syntax:

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

Here, the **container** is a div element with the Bootstrap class name container.

1.2 Row

The purpose of a row is to wrap all the columns.

Syntax:

```
<div class="container">

  <div class="row"></div>

</div>
```

Here, the **row** is a div element with the Bootstrap class name row.

1.3 Column

We should place the columns inside a row and the content inside a column.

We can specify the number of columns our content should occupy in any device. The number of columns we specify should be a number in the range of 1 to 12.

Syntax:

```
<div class="container">

  <div class="row">

    <div class="col-12">

     I'm your content inside the grid!

    </div>

  </div>
```

</div>

Here, the column is a div element with the Bootstrap class name col-12.

Note:

If Bootstrap class name is col-12, it occupies the entire width available inside the row.

The Bootstrap class names **col-***indicates the number of columns you would like to use out of the possible 12 columns per row. For example**, col-1, col-5**, etc.

Bootstrap Grid System | Cheat Sheet

1. Column Wrapping

When we place more than 12 grid columns in a single row, the extra grid columns will wrap in a new line.

Try out the different combinations of Bootstrap class names like col-4, col-4, col-6, col-6 and col-6, col-4, col-6, col-4, etc.

2. The Layout at different Breakpoints

Bootstrap provides different **Bootstrap Grid Column class name prefixes** for Five Responsive Tiers (Responsive Breakpoints).

Device	Device Size (Width)	Class Name Prefix
Extra small devices	<576px	col-
Small devices	>=576px	col-sm-
Medium devices	>=768px	col-md-
Large devices	>=992px	col-lg-
Extra large devices	>=1200px	col-xl-

If we define the behaviour of the Bootstrap Grid Column in a particular device, similar behaviour is guaranteed in all devices with larger sizes.

Syntax:

<div class="container">

  <div class="row">

```
   <div class="col-6">

    <h1 class="heading">Taj Mahal</h1>

    <p>The Taj Mahal is on the southern bank of the river Yamuna.</p>

   </div>

  </div>

</div>
```

2.1 Class names in combination

We can use a combination of different Bootstrap class names for each Bootstrap Grid Column.

Try out the combinations of different Bootstrap class names like col-lg-4, col-lg-3, col-lg-8, etc.

Note:

Bootstrap follows **Mobile First Approach**.

First, design the Layout of a mobile version, and this will be adopted by devices with larger sizes.

Bootstrap Grid System | Part 2 | Cheat Sheet

1. CSS Box Properties

1.1 Margin

We can get spacing between the two HTML elements with the CSS Box property margin.

To get space only on one particular side, we use **Margin Variants**.

- margin-top

- margin-right

- margin-bottom

- margin-left

2. Bootstrap Spacing Utilities

2.1 Margin

CSS Margin property	Bootstrap class name
Margin	m-*
margin-top	mt-*
margin-right	mr-*
margin-bottom	mb-*
margin-left	ml-*

Note: The asterisk (*) symbol can be any number in the range of 0 to 5. For example, m-5, mr-2, mb-3, etc.

2.1.1 Margin Values

Size	Value
0	0
1	0.25 * spacer
2	0.5 * spacer
3	1 * spacer
4	1.5 * spacer
5	3 * spacer

Note: The **spacer** is a variable and has a value of 16 pixels by default.

For example,

- mb-3 = 1 * 16px = 16px

- m-5 = 3 * 16px = 48px

Note

Avoid using CSS margin-left and margin-right properties for Bootstrap Grid Columns. It disturbs the Bootstrap Grid System and gives unexpected results.

2.2 Padding

CSS Padding property	Bootstrap class name
Padding	p-*
padding-top	pt-*
padding-right	pr-*
padding-bottom	pb-*
padding-left	pl-*

Note: The asterisk (*) symbol can be any number in the range of 0 to 5. For example, p-3, pr-1, pb-5, etc.

2.2.1 Padding Values

Size	Value
0	0
1	0.25 * spacer
2	0.5 * spacer
3	1 * spacer
4	1.5 * spacer
5	3 * spacer

The **spacer** is a variable and has a value of 16 pixels by default.

For example,

- p-1 = 0.25 * 16px = 4px
- pt-4 = 1.5 * 16px = 24px

3. Bootstrap Background Color Utilities

You can use one of the below Bootstrap class names to apply a background color to an HTML element.

Example:

```
<div class="bg-primary p-2">

   <p>Primary</p>

  </div>

  <div class="bg-secondary p-2">

   <p>Secondary</p>

  </div>
```

Note:

In the above Code Playground, we used the bootstrap class name **p-2** for padding.

4. Developing Layouts for five Responsive Breakpoints

4.1 Color Palette

Example:

```
<h1 class="color-palette-heading">Color Palette</h1>

  <div class="container">

   <div class="row">

    <div class="col-12 col-md-6 col-lg-4 col-xl-3 mb-3">

     <div class="color-box bg-primary">

      <p class="color-name">Primary</p>

     </div>

    </div>

   </div>

</div>
```

Bootstrap Navbar | Cheat Sheet

1. Bootstrap Components

1.1 Navbar

A Navbar is a navigation header that is placed at the top of the page. With Bootstrap, a Navbar can extend or collapse, depending on the device size1.

1.1 HTML Nav element:

The HTML nav element is a container element similar to the HTML div element. We use the HTML nav element to add a Navbar to our website.

```
<nav class="navbar navbar-expand-lg navbar-light bg-light"></nav>
```

1.1.2 Nav Items inside Navbar:

Syntax:

```
<a class="nav-link active" href="#">

  Home

  <span class="sr-only">(current)</span>

</a>

<a class="nav-link" href="#">Features</a>

<a class="nav-link" href="#">Pricing</a>

<a class="nav-link disabled" href="#" tabindex="-1" aria-disabled="true">

  Disabled

</a>
```

1.1.3 Nav link

```
<a class="nav-link" href="#">Features</a>
```

2. HTML Elements

In general, HTML elements can be divided into two categories.

- Block-level Elements

- Inline Elements

2.1 Block-level Elements

These elements always start in a new line and take up the full width available. So, an HTML Block-level element occupies the entire horizontal space of its parent element.

For example, the HTML h1 element, p element, div element, etc.

Example:

```
<h1 class="heading">The seven wonders of the world</h1>

<p class="paragraph">The Taj Mahal is one of the seven wonders of the world</p>
```

2.2 Inline Elements

These elements do not start in a new line and only take up as much width as necessary.

For example, the HTML button element, img element, a element, etc.

Example1:

```
<img

  src="https://d1tgh8fmlzexmh.cloudfront.net/ccbp-static-website/mysore-palace2-img.png"

  class="image"

/>

<img

  src="https://d1tgh8fmlzexmh.cloudfront.net/ccbp-static-website/varanasi1-img.png"

  class="image"

/>
```

Example2:

```
<p class="paragraph">

  The <a class="link-text" href="https://en.wikipedia.org/wiki/Taj_Mahal">Taj Mahal</a>

  is one of the seven wonders of the world.

</p>
```

3. CSS Box properties

3.1 Margin

We can align HTML Block-level elements horizontally using CSS margin property.

Apart from values that are specified in pixels, it also accepts auto keyword as a value.

Note:

If we specify the CSS text-align property to the HTML Block-level element, it aligns the text or HTML Inline elements inside it.

3.1.1 Auto Value

The child element will be horizontally centred inside the HTML container element.

Syntax:

```
<div class="navbar-nav nav-items-center">
```

```
  <a class="nav-link active" href="#">

    Home

    <span class="sr-only">(current)</span>

  </a>

  <a class="nav-link" href="#">About Me</a>

  <a class="nav-link" href="#">Projects</a>

  <a class="nav-link" href="#">Testimonials</a>

</div>
```

CSS:

```
.nav-items-center {

  margin: auto;

}
```

3.1.2 Auto Value with Margin Variants:

Using auto as a value for the CSS margin-right property takes up all the available space, and the element gets aligned to the left.

Using auto as a value for the CSS margin-left property takes up all the available space, and the element gets aligned to the right.

Sytax:

```
<div class="navbar-nav nav-items-right">

  <a class="nav-link active" href="#">

    Home <span class="sr-only">(current)</span>

  </a>

  <a class="nav-link" href="#">About Me</a>

  <a class="nav-link" href="#">Projects</a>

  <a class="nav-link" href="#">Testimonials</a>

</div>
```

CSS:

```
.nav-items-right {
```

```
margin-left: auto;
}
```

4. Bootstrap Utilities

4.1 Margin

Apart from the numbers 0-5, the margin also has the below Bootstrap class names.

- m-auto
- ml-auto
- mr-auto

5. Step by Step Process to build a Navbar

Step-1: Adding Bootstrap Navbar Component

Step-2: Adding Logo

Step-3: Aligning Nav Items

Step-4: Changing Navbar Background color

Example:

```
<body>

<nav class="navbar navbar-expand-lg navbar-light bg-white">

    <a class="navbar-brand" href="#">

     <img src=https://d1tgh8fmlzexmh.cloudfront.net/ccbp-responsive-website/navbar-logo-img.png" class="navbar-image"  alt=""/>

    </a>

    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation">

     <span class="navbar-toggler-icon"></span>

    </button>

    <div class="collapse navbar-collapse" id="navbarNavAltMarkup">

     <div class="navbar-nav ml-auto">

      <a class="nav-link active" href="#">

       Home <span class="sr-only">(current)</span>
```

```html
        </a>

        <a class="nav-link" href="#">About Me</a>

        <a class="nav-link" href="#">Projects</a>

        <a class="nav-link" href="#">Testimonials</a>

      </div>

    </div>

  </nav>

</body>
```

CSS Selectors & Inheritance | Cheat Sheet

1. CSS Selectors

The CSS Selectors are used to select the HTML Elements that we want to style.

The different types of CSS Selectors are:

- Simple Selectors
- Class Selector
- ID Selector
- Type (tag name) Selector
- Attribute Selector
- Universal Selector
- Pseudo-class
- Compound Selectors
- Complex Selectors and many more.

1.1 Class Selector

The CSS Class Selector selects all the HTML elements that have a given CSS class selector as their class attribute value. It consists of a dot (.), followed by the class name of the HTML element.

Syntax:

<p class="paragraph">The population of India is around 138 crores.</p>

CSS:

.paragraph {

  color: blue;

}

Here, the CSS class selector is .paragraph. So, it selects all the HTML elements that have an HTML attribute name class, and it's value paragraph.

Note:

There can be **more than one** HTML element with the same class name in the HTML document.

1.2 ID Selector

The CSS ID selector selects an HTML element based on its ID attribute value. It consists of a hash (#), followed by the ID of the HTML element.

Syntax:

<p id="populationParagraph">The population of India is around 138 crores.</p>

CSS:

#populationParagraph {

  color: blue;

}

Here, the CSS ID selector is **#populationParagraph**. So, it selects the HTML element that has an HTML attribute name id and it's value **populationParagraph**.

Note:

There should be only one HTML element with a given ID in the entire HTML document. The HTML id attribute value doesn't need to have the prefix **section** as CCBP UI Kit is not used.

1.3 Type (tag name) Selector

The CSS Type Selector selects all the HTML elements based on their tag names (h1, p, div, etc.)

Syntax:

<p>The population of India is around 138 crores.</p>

CSS:

p {

  color: blue;

}

Here, the CSS Type selector is p. So, it selects all the HTML elements that have a tag name p.

2. Most fundamental concepts of CSS

In CSS, the styles that are applied to HTML elements depend on three fundamental concepts.

- **Inheritance**

- **Specificity**

- **Cascade**

2.1 CSS Inheritance

The mechanism through which the value of certain CSS properties is passed on from parent elements to child elements is called **Inheritance**.

2.1.1 Parent Element

If the HTML elements are placed inside the other HTML element, then the outer HTML element is called the parent element of the HTML elements inside it.

Example:

```
<div>

 <h1>The seven wonders of the world</h1>

 <p>

  The <a href="https://en.wikipedia.org/wiki/Taj_Mahal">Taj Mahal</a>

  is one of the seven wonders of the world.

 </p>

</div>
```

From the above Code Snippet, we can say:

- The HTML **div** element is the parent element of the HTML **h1** and **p** elements.
- The HTML **p** element is the parent element of the HTML **a** element.

2.1.2 Child Element

An HTML element that is directly inside the parent element is called the child element of that parent element.

Example:

```
<div>

 <h1>The seven wonders of the world</h1>

 <p>

  The <a href="https://en.wikipedia.org/wiki/Taj_Mahal">Taj Mahal</a>

  is one of the seven wonders of the world.

 </p>

</div>
```

From the above Code Snippet, we can say:

- The HTML **h1** and **p** elements are the child elements of the HTML **div** element.
- The HTML **a** element is the child element of the HTML **p** element.

CSS properties can be categorized into two types:

- Inherited properties

- Non-inherited properties

2.1.3 Inherited Properties

If the CSS properties applied to the parent element **are inherited** by the child elements, then they are called Inherited properties.

Some of the CSS Inherited Properties are:-

- Text related Properties
    - font-family
    - font-style
    - font-weight
    - text-align
- List related Properties
    - list-style-type
- color property and many more.

2.1.4 Non-inherited Properties

If the CSS properties applied to the parent element are not inherited by the child elements, then they are called Non-inherited properties.

Some of the CSS Non-inherited properties are:

CSS Box Properties

- width
- height
- margin
- padding
- border-style
- border-width
- border-color
- border-radius

CSS Background Properties

- background-image
- background-color
- background-size

text-decoration and many more.

CSS Specificity & Cascade | Cheat Sheet

1. Specificity

Specificity is how browsers decide which CSS property values are the **most relevant** to an HTML element and, therefore, will be applied.

The following list of CSS Selectors is in the lowest to highest order by specificity.

1. Type (tag name) Selector

2. Class Selector

3. ID Selector

1.1 Type Selector & Class Selector

A Class Selector is **more specific** compared to Type (tag name) Selector as it selects only the HTML elements that have a **specific class attribute value** in the HTML document.

Note:

It doesn't overwrite the entire CSS Ruleset but only overwrites the CSS properties that are the same.

1.2 Class Selector & ID Selector

An ID Selector is more specific when compared to a Class Selector as we provide a unique ID within the HTML document and it selects only a **single** HTML Element.

2. Inline Styles

The Inline styles are applied directly to an HTML element. They use the HTML style attribute, with CSS property values defined within it.

Syntax:

```
<tag style = "property1: value1; property2: value2; ...">Content</tag>
```

A HTML style attribute value can consist of one or more CSS property values.

Note:

Inline Styles have the highest specificity. They overwrite any other styles specified using CSS Selectors.

Using Inline Styles is not recommended because

- Inline Styles are not reusable.

- Writing HTML and CSS separately increases code readability.

3.CSS Cascade

The source order of CSS Rulesets matters. When two CSS Rulesets have equal specificity, the one that comes last in the CSS is applied.

Note:

The styles that apply to the HTML Elements are not determined by the **order the classes** defined in the HTML **class** attribute, but instead the order in which they appear in the CSS**.**

3.1 The !important exception

It is a special piece of CSS used to make a particular CSS property and value **the most specific thing**, irrespective of source order and specificity**.**

Example:

<h1 class="style-1">About India</h1>

CSS:

.style-1 {

 color: green;

}

h1 {

 color: orange !important;

}

The only way to override a **!important** property value is to include another **!important** property value. The added property value should either come later in the order or should be of higher specificity.

Note: Always look for a way to use specificity before even considering **!important**.Only use **!important** when you want to override foreign CSS (from external libraries, like Bootstrap).

Banner Section | Cheat Sheet

1. Bootstrap Navbar

A piece of code in the HTML body element in the below Code Playground is the Bootstrap Navbar without a list-based approach Code.

Syntax:

```
<body>

  <nav class="navbar navbar-expand-lg navbar-light bg-light">

    <a class="navbar-brand" href="#">Navbar</a>

    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation">

      <span class="navbar-toggler-icon"></span>

    </button>

    <div class="collapse navbar-collapse" id="navbarNavAltMarkup">

      <div class="navbar-nav">

        <a class="nav-link active" href="#">

          Home

          <span class="sr-only">(current)</span>

        </a>

        <a class="nav-link" href="#">Features</a>

        <a class="nav-link" href="#">Pricing</a>

        <a class="nav-link disabled" href="#" tabindex="-1" aria-disabled="true">Disabled</a>

      </div>

    </div>

  </nav>

</body>
```

2. Bootstrap Containers

2.1 Container

The Bootstrap class name container provides us default left and right spacings starting from smaller devices for a better user experience. It has one fixed width for each breakpoint in Bootstrap (extra small, small, medium, large, and extra large).

Device	Device Size (Width)	Container Max Width
Extra small devices	< 576px	100%
Small devices	>= 576px	540px
Medium devices	>= 768px	720px
Large devices	>= 992px	960px
Extra large devices	>= 1200px	1140px

Syntax:

```
<div class="container">
  <div class="row">
   <div class="col-12">
    <h1>Taj Mahal</h1>
    <p>
      The Taj Mahal is an ivory-white marble mausoleum on the southern
      bank of the river Yamuna in the Indian city of Agra.
    </p>
   </div>
  </div>
</div>
```

2.2 Fluid Container

The Bootstrap class name container-fluid is a full width container, spanning the entire width of the viewport.

If we don't need left and right spacings, we can use the Bootstrap class name container-fluid instead of container.

```
<div class="container-fluid">

  <div class="row">

   <div class="col-12">

    <h1>Taj Mahal</h1>

    <p>

      The Taj Mahal is an ivory-white marble mausoleum on the southern

      bank of the river Yamuna in the Indian city of Agra.

    </p>

   </div>

  </div>

</div>
```

3. CSS Colors

3.1 Transparent

The CSS **transparent** keyword represents a fully transparent color. This makes the background behind the colored HTML element completely visible.

For example, to set a transparent background color,

Syntax:

```
.custom-outline-button {

  background-color: transparent;

}
```

This allows you to set the background color of the HTML element to transparent so that any background HTML element will show through.

Bootstrap also provides you with a class name bg-transparent to set the background color to transparent.

Why Choose Us? Section | Cheat Sheet

1. Bootstrap Spacing Utilities

1.1 Padding

CSS Padding property	Bootstrap class name
padding	p-*
padding-top	pt-*
padding-right	pr-*
padding-bottom	pb-*
padding-left	pl-*

The asterisk (*) symbol can be any number in the range of 0 to 5. For example, p-3, pr-1, pb-5, etc.

1.1.1   Padding Values

Size	Value
0	0
1	0.25 * spacer
2	0.5 * spacer
3	1 * spacer
4	1.5 * spacer
5	3 * spacer

The **spacer** is a variable and has a value of 16 pixels by default.

For example,

p-1 = 0.25 * 16px = 4px

pt-4 = 1.5 * 16px = 24px

2. HTML Elements

2.1 HTML Span Element

The HTML span element is a generic inline container element which is mainly used for styling text in HTML Elements.

Syntax:

<p class="wcu-card-description">

  Food Coupons & Offers upto

  50% OFF

  and Exclusive Promo Codes on All Online Food Orders.

</p>

CSS:

.offers {

  color: #323f4b;

  font-style: italic;

  font-weight: 600;

}

Explore Menu Section | Cheat Sheet

1. CSS Units

1.1 Percentage

To define the size of a Child Element relative to its Parent Element, we can use Percentages.

2. Bootstrap Sizing Utilities

2.1 Percentage

You can use the below Bootstrap class names to specify the width of an HTML element in percentage.

CSS property and value	Bootstrap class name
width: 25%	w-25
width: 50%	w-50
width: 75%	w-75

CSS property and value	Bootstrap class name
width: 100%	w-100

Syntax:

```
<img

 src="https://d1tgh8fmlzexmh.cloudfront.net/ccbp-responsive-website/em-ginger-fried-img.png"

 class="menu-item-image w-100"

/>
```

Note:

By default, the height of the image automatically adjusts when we alter the width.

3. Bootstrap Icons

3.1 How to add the Bootstrap Icons

- Go to https://icons.getbootstrap.com in your Web browser. You will find many icons.
- Click on the icon you need. For the icon used in this section, click on arrow-right-short.
- Copy the HTML code and paste it.
- Change the HTML attributes width, height, and fill of the HTML svg element as you need.

A piece of the code in the HTML body element is the Bootstrap arrow-right-short icon code in the below Code Playground.

Note:

The HTML svg element is an HTML inline element. We can use it to add icons to our website.

4. Bootstrap Utilities

4.1 Shadow

To apply shadows to HTML elements, you can use the below Bootstrap class names.

Bootstrap class name
shadow-none
shadow-sm
shadow

Bootstrap class name
shadow-lg

Healthy Food, Delivery and Payment, Thanking Customers Sections | Cheat Sheet

1. Bootstrap Flex Utilities

1.1 Order

The Bootstrap Order class names are used to change the visual order of the flex items that appear inside the Flex Container.

For example**, order-1, order-2, order-3**, etc.

We can use any number in the range **of 0 to 12** for a bootstrap order class name.

These class names are responsive. So, you can set the order by breakpoint.

For example, **order-1, order-md-2, order-lg-3**, etc.

2. Bootstrap Display Utilities

We can hide and show HTML elements responsively for each screen size with the Display utilities.

We can hide HTML Elements using d-*-none class names, where * represents breakpoints (sm, md, lg, xl)

For example, d-none, d-sm-none, d-md-none, etc.

Syntax:

```
<div class="my-container">

  <p class="box">Box 1</p>

  <p class="box d-none">Box 2</p>

  <a href="" class="d-md-none">wikipedia</a>

</div>
```

Based on the type of HTML element, we can use d-*-inline and d-*-block class names to show HTML element.

For example, d-block, d-md-inline, d-lg-block, etc.

Follow Us Section and More Styling | Cheat Sheet

1. Adding Icons

There are a limited number of Icon choices in Bootstrap icons. Since we don't have the desired icons in Bootstrap Icons, we use Font Awesome Icons.

1.1 Font Awesome Icons

To use the Font Awesome Icons, you need to add the below Font Awesome Icons Kit Code in the HTML **head** element.

Syntax:

<script src="https://kit.fontawesome.com/20c5629a29.js" crossorigin="anonymous"></script>

Note:

The CSS Property border-radius allows you to add circular corners to an HTML element. We need to provide the same height and width to get circular corners else we will get elliptical corners.

3. Adding Links to the Sections

- Adding id to the section to which we want to navigate.

Syntax: <div class="wcu-section pt-5 pb-5" id="wcuSection">...</div>

- Providing id as href Attribute value to the Nav Item.

Syntax:

  Why Choose Us?

  (current)

4. Bootstrap Position Utilities

4.1 Fixed Top

The bootstrap class name fixed-top positions an HTML element at the top of the viewport irrespective of the scrolling.

Syntax: <nav class="navbar navbar-expand-lg navbar-light bg-white fixed-top">...</nav>

4.2 Fixed Bottom

The bootstrap class name fixed-bottom positions an HTML element at the bottom of the viewport irrespective of the scrolling.

Syntax: <nav class="navbar navbar-expand-lg navbar-light bg-white fixed-bottom">...</nav>

CSS Gradients & More Bootstrap Components | Cheat Sheet

1. Bootstrap Components

1.1 Modal

Try out changing the content and the styles of the Modal Header, Modal Body and Modal Footer in the Bootstrap Modal Code in the below Code Playground.

Syntax:

```html
<body>

  <button type="button" class="btn btn-primary" data-toggle="modal" data-target="#exampleModal">

    Launch demo modal

  </button>

  <div class="modal fade" id="exampleModal" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">

    <div class="modal-dialog">

     <div class="modal-content">

      <div class="modal-header">

       <h5 class="modal-title" id="exampleModalLabel">Modal title</h5>

       <button type="button" class="close" data-dismiss="modal" aria-label="Close">

         <span aria-hidden="true">&times;</span>

       </button>

      </div>

      <div class="modal-body">

       ...

      </div>

      <div class="modal-footer">

       <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>

       <button type="button" class="btn btn-primary">Save changes</button>
```

```
        </div>

      </div>

     </div>

    </div>

   </body>
```

3. Gradients

A special type of Background Image formed by the transition between two or more colors.

There are mainly two types of gradients:

- Linear Gradient
- Radial Gradient

3.1 Linear Gradient

To create the most basic type of gradient, all you need is to specify two colors. You must have at least two colors, but you can have as many as you want.

Syntax:

```
<div class="linear-gradient-background"></div>
```

CSS:

```
.linear-gradient-background {

 height: 100vh;

 background-image: linear-gradient(#2196f3, #f44336);

}
```

3.1.1 Changing Direction

By default, linear gradients run from top to bottom. You can change their transition by specifying a direction.

Direction	Description
to top	Colors transition (change) from bottom to top
to bottom	It is a default direction. Colors transition (change) from top to bottom

Direction	Description
to left	Colors transition (change) from right to left
to right	Colors transition (change) from left to right

CSS:

.linear-gradient-background {

  background-image: linear-gradient(to right, #2196f3, #f44336);

}

3.1.2 Using more than two colors

You don't have to limit yourself to two colors. You may use as many as you like! By default, colors are evenly spaced along the gradient.

CSS:

.linear-gradient-background-with-more-colors {

  height: 100vh;

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

}

3.2 Radial Gradient

Syntax:

<div class="radial-gradient-background"></div>

CSS:

.radial-gradient-background {

  height: 100vh;

  background-image: radial-gradient(#2196f3, #f44336);

}

3.2.1 Using more than two colors

You don't have to limit yourself to two colors. You may use as many as you like! By default, colors are evenly spaced along the gradient.

Syntax:

CSS:

.radial-gradient-background-with-more-colors {

  height: 100vh;

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

}