



9/6/2021


Learn CSS Media Queries by Building Three Projects - DEV Community

Learn CSS Media Queries



Learn CSS Media Queries by Building Three Projects

#css #tutorial #webdev #beginners

 joy.does.artworks Apr 26 Originally published at [freecodecamp.org](https://www.freecodecamp.org) · 15 min read

Today we're gonna learn how to use CSS Media queries to build responsive websites & practice by doing 3 projects. Let's go 🚀

Table of Contents -->

- [What are CSS Media Queries?](#)
- [Steps to follow](#)
- [The Syntax](#)
- [Practice Projects](#)
- [Conclusion](#)

Topics to discuss at a glance :

https://dev.to/artworks\_joy/learn-css-media-queries-by-building-three-projects-3a8b

1/44


9/6/2021

Learn CSS Media Queries by Building Three Projects - DEV Community

Original Post at [Free Code Camp](#)

What are CSS Media Queries?

Why use CSS Media Queries ?



CSS Media Queries allows us to create Responsive website across all screen sizes ranging from desktop to mobile screen. Therefore, It's a must to learn this topic.

Here's a demo of the magic of Media Queries 🖱

A

B

C

Mobile

A

B

C

D

E


F

G

H

I

Desktop



https://dev.to/artworks\_joy/learn-css-media-queries-by-building-three-projects-3a8b

3/44

9/6/2021

Learn CSS Media Queries by Building Three Projects - DEV Community


Topics To Discuss

#1 What are Media Queries?

#2 Steps to Follow


#3 The Syntax

#4 Practice Exercises



You can watch this tutorial on YouTube as well if you like:

Learn CSS Media Query - Build 3 Projects in 2021 🚀 ||...



https://dev.to/artworks\_joy/learn-css-media-queries-by-building-three-projects-3a8b

2/44


9/6/2021

Learn CSS Media Queries by Building Three Projects - DEV Community

We'll build that on project 2. Layout is called **Card Layout**. More Layout Examples [here!](#)

How to Set Up the Project

Let's Code Together



For this project, you need to know little bit of HTML, CSS and know how to work with VS code. Follow along with me ->

- Create a folder named "Project-1"
- Open VS Code
- Create **index.html**, **style.scss** & **main.js** file
- install Live Server & SASS Compiler
- Run Live Server & SASS Compiler

HTML

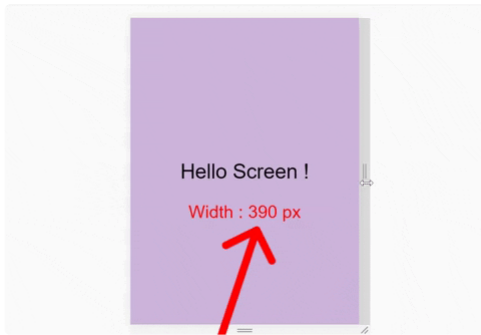
On HTML, Write this code inside the body tag 🖱

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

we also need to see the exact size of our window. Here's a demo of what I mean 🖱

https://dev.to/artworks\_joy/learn-css-media-queries-by-building-three-projects-3a8b

4/44



So, write this line below inside the html file, ->

```
<div id="size"></div>
```

## SCSS

We'll Use SCSS, not CSS. But..... what is SCSS?

# What is SCSS ?



SCSS is a pre-processor of CSS which is more powerful than regular CSS. Using SCSS we can ->

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

5/44

## JavaScript

We need to update our screen size inside our id every time we resize our window. So, write these codes in main.js file ->

```
// 'screen' is name of a function
window.onresize = screen;
window.onload = screen;

// Function named 'screen'

function screen() {
  Width = window.innerWidth;
  document.getElementById("size").innerHTML
    = "Width : " + Width + " px"
}
```

## Download the images for the project

# But WAIT.....



Responsive website also means **Responsive Images**. We're also going to make images responsive in this project. The images are on my [GitHub repository](#). Here's how to get them:

1. Visit and copy the link above
2. Go to [downgit](#) and paste the link you copied

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

7/44

1. Nest our selectors like a branch of a tree and better manage our code.
2. Store various values into variables
3. Use Mixins to stop code repetition & save time

And Much more !

On SCSS, we'll remove our default browser settings, change box-sizing, font-size & font-family, like this

```
*{
  margin : 0px;
  padding : 0px;
  box-sizing : border-box;

  body{
    font-size : 35px;
    font-family : sans-serif;
  }
}
```

**Don't forget** to set **height** of the **.container class**. Otherwise we'll fail to achieve our desired results ->

```
.container{
  height : 100vh;
}
```

Remember the additional id we wrote in HTML ? We'll style it & position it on our browser here ->

```
#size {
  position: absolute;

  // positioning screen size below our main text
  top : 60%;
  left: 50%;

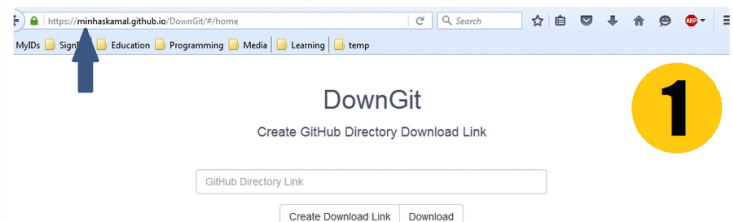
  transform: translateX(-50%);

  color : red;
  font-size : 35px;
}
```

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

6/44

3. Follow the steps in this video



go to [minhaskamal.github.io/DownGit](https://minhaskamal.github.io/DownGit)

And..... we're all set ! Let's Start Coding



# Grab Your Bubble Tea & Let's Start Coding



## The Syntax

The syntax of a Media Query

```
@media screen and (max-width: 768px){
  .container{
    //Your code's here
  }
}
```

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

8/44

```

}
}

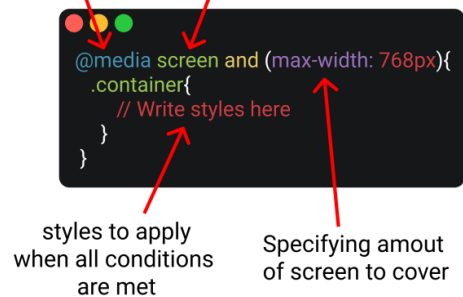
```

An illustrated Explanation ->

# The Syntax

declaration

Media Type



Let's divide the syntax into 4 section :-

1. Media Query Declaration
2. The Media Type
3. min-width & max-width Function
4. The Code itself

To understand all 4 section of the syntax, let's start our **First Project**

Now, come at the bottom, target the .container & .text classes. We'll also center our text like this👇

```

.container{
  //To place text at center

  display : grid;
  place-items : center;

  background-color : $color-1;
  height : 100vh;
}

.text{
  // keep it blank for now
}

```

So far so good !

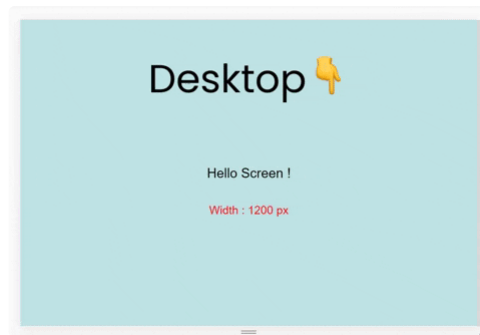
# Good Job !



## 1. Media query declaration Rule

Media Queries start with the @media declaration. Main purpose of writing this is to **tell the Browser** that we have specified a media query. On CSS, write like this 👇

```
@media
```



We'll Build this 🖱️ A small project where background-color changes on resizing the window by taking 1 small step at a time. Let's start !

## HTML

place some text inside our HTML, like this ->

```

<div class = "container">

  <div class = "text">
    Hello Screen !
  </div>
</div>

```

## SCSS

Now, we'll store 4 color codes inside variables like this 👇

```

$color-1 : #cdb4db ; // Mobile
$color-2 : #fff1e6 ; // Tablet
$color-3 : #52b788 ; // Laptop
$color-4 : #bee1e6 ; // Desktop

```

More colors at [coolers.co](https://coolers.co)

## 2. The Media Type

This is used to specify the nature of the device we're working with. The 4 values are ->

- all
- print
- screen
- speech

Purpose of every 4 value at a glance 👇

Value	Used For
all	all media type devices
print	Printers
screen	computer screens, tablets, smart-phones etc.
speech	screenreaders that "reads" the page out loud



We declare the **media type** after @media declaration. Like this 👇

```
@media screen
```

## FAQ : Why do we write The "and" operator?

# Why use the "and" operator ?



Let's say, we're placing an order at a restaurant, "A burger **and** a pizza". Notice that the 2 orders are separated by a **[and]**

Likewise, media type, min-width & max-width functions are basically conditions we are giving to the browser. We don't write "**and**" operator if we have 1 condition. Like this ->

```
@media screen {
  .container{
    // Your code here
  }
}
```

We write "**and**" operator if we have 2 conditions. Like this ->

```
@media screen and (max-width : 768px) {
  .container{
    // Your code here
  }
}
```

You can also skip the media type and work with just min-width & max-width. Like this ->

```
//Targeting screen sizes between 480px & 768px

@media (min-width : 480px) and (max-width : 768px) {
```

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

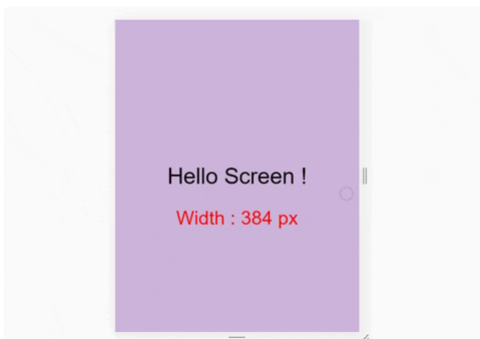
13/44

Here's a list of every device screen resolution on [CSS-Tricks](#).

## max-width :

Using this function, we are creating a boundary. This will work as long as we are **inside the boundary**. Here's a sample ↴

Our Boundary is 500px



notice how the light purple color gets **Disabled** when we hit above 500px.

To recreate this, write these on SCSS

```
.container{
  background-color: white ;
  height: 100vh;
  display: grid;
  place-items: center;
}
```

At the bottom, insert the media query, like this ↴

```
@media screen and (max-width : 500px){
  .container{
    background-color: $color-1;
  }
}
```

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

15/44

```
.container{
  // Your code here
}
```

If you have 3 conditions or more, you can use a **comma**, like this ->

```
//Targeting screen sizes between 480px & 768px

@media screen, (min-width : 480px) and (max-width : 768px) {
  .container{
    // Your code here
  }
}
```

## 3. min-width & max-width Function

Let's discuss the Most important component of a media query, Screen breakpoints.

To be honest, there's no such thing as a standard screen break-point guide due to countless screen sizes on the market. But, for our project, we'll follow [The Official Bootstrap 5](#) screen break-point values

## Bootstrap 5 Breakpoints

0px — 576px -> Small  
 577px — 768px -> Medium  
 769px — 992px -> Large  
 993px — 1200px -> Extra Large  
 1201px — 1400px -> Extra extra large



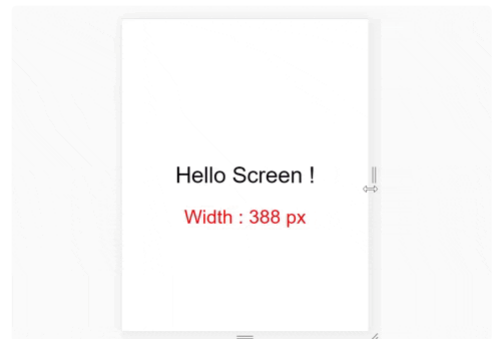
[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

14/44

## min-width :

we are also creating a boundary here. But, This will work if we go **outside the boundary**. Here's a sample ↴

Our Boundary is 500px



Notice how the light purple color gets **Enabled** after we hit above 500px width.

To recreate this, write these on SCSS

```
.container{
  background-color: white ;
  height: 100vh;
  display: grid;
  place-items: center;
}
```

At the bottom, insert the media query, like this ↴

```
@media screen and (min-width : 500px){
  .container{
    background-color: $color-1;
  }
}
```

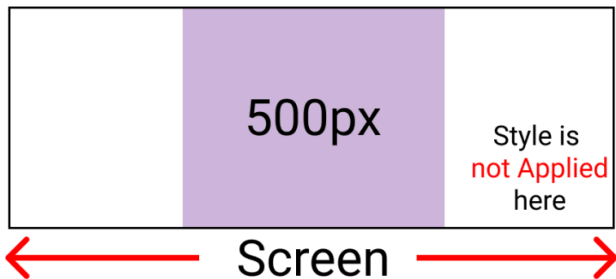
To sum it up. remember that

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

16/44

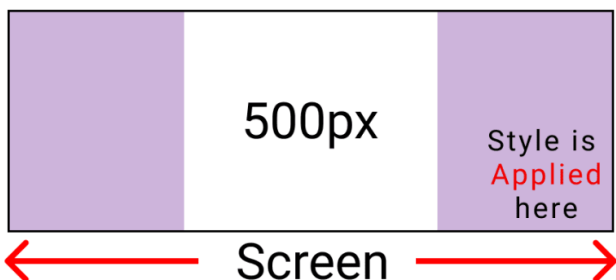
- **max-width** sets styles inside the set boundary

## Max-width



- **min-width** sets styles outside the set boundary

## Min-width



### The code itself :

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

17/44

```
.container {  
  background-color: white ;  
  height: 100vh;  
  display: grid;  
  place-items: center;  
}
```

We're all 50% done ! Now let's setup the 4 media queries 🐾

**But Wait...**

# Wait a Minute !



You need to follow a serial while writing the media queries. Start writing from the **largest display to the smallest display**.

### Desktop - 1200px

For the Desktop screen, write these on SCSS 🐾

```
// using variable here which is 🐾 1200px  
@media screen and (max-width: $desktop){  
  .container{  
    background-color: $color-4;  
  }  
}
```

The Result ->

Let's put our project-1 together !

We will have 4 screen breakpoints

- Mobile -> 576px
- Tablet -> 768px
- Laptop -> 992px
- Desktop -> 1200px

Yes, we are following the official [bootstrap 5](#) screen breakpoints. And each breakpoints will get these colors ->

## Break-Points

Mobile -> 576px

Tablet -> 768px

Laptop -> 992px

Desktop -> 1200px



For 4 device types, we will have 4 Media Queries. Before touching the 4 media queries, first, store the breakpoint values in variables. Like this 🐾

**Note :** Don't forget to put the \$ sign

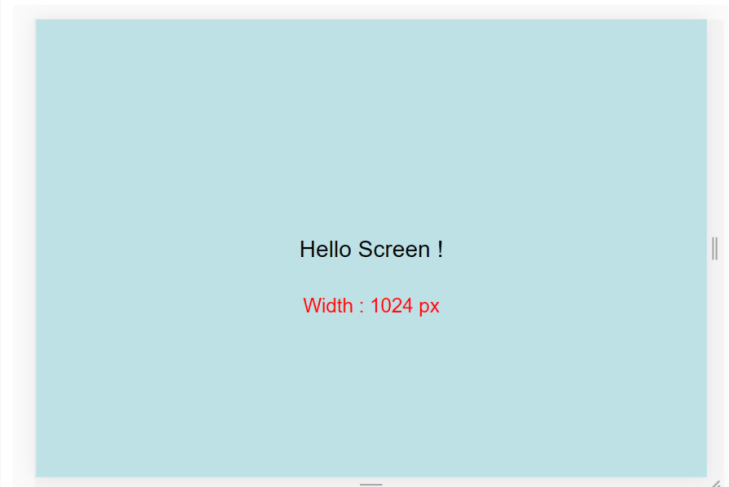
```
$mobile : 576px;  
$tablet : 768px;  
$laptop : 992px;  
$desktop : 1200px;
```

And our .container class should look like this 🐾

```
.container{
```

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

18/44

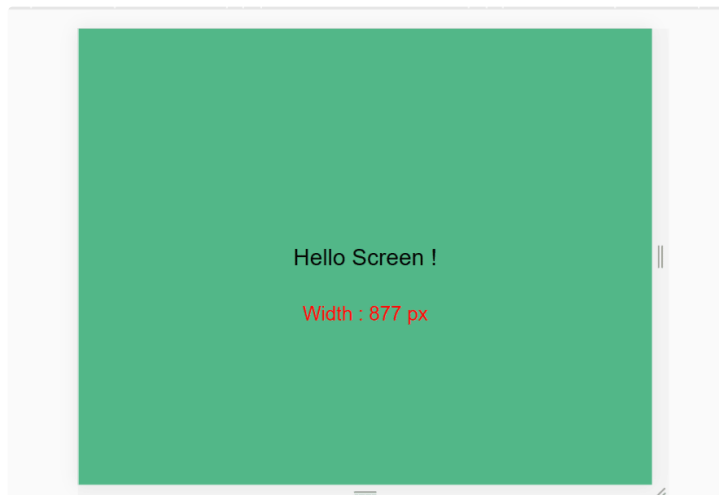


### Laptop - 992px

For the Laptop screen, write these on SCSS 🐾

```
// using variable here which is 🐾 992px  
@media screen and (max-width: $laptop){  
  .container{  
    background-color: $color-3;  
  }  
}
```

The Result ->

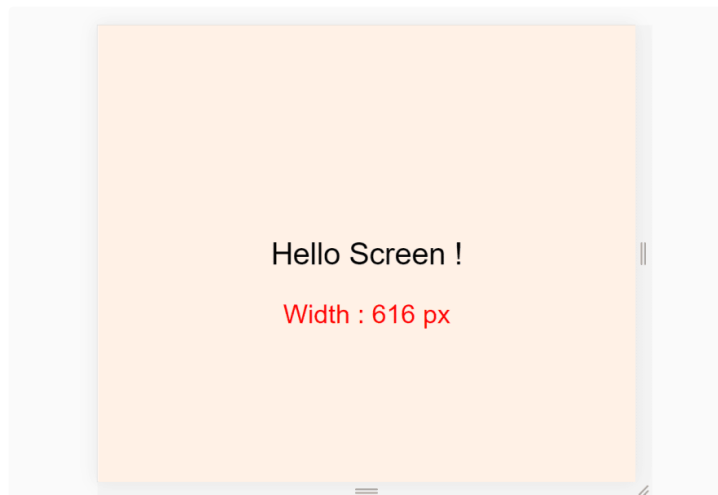


### Tablet - 768px

For the tablet screen, write these on SCSS [🔗](#)

```
// using variable here which is 🖱 768px
@media screen and (max-width: $tablet){
  .container{
    background-color: $color-2;
  }
}
```

The Result ->

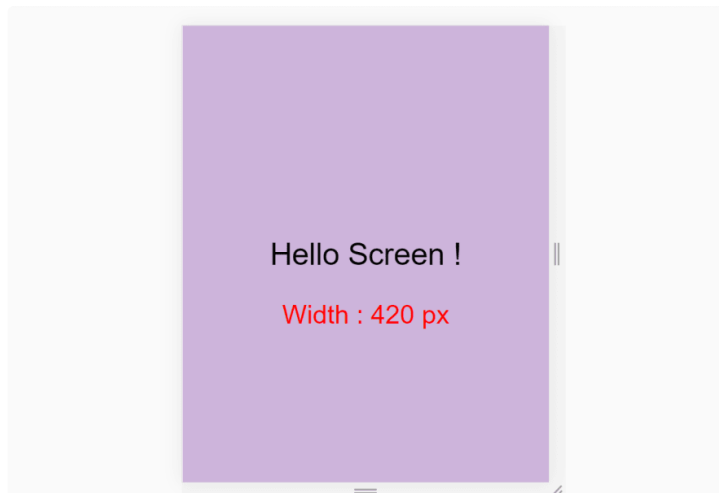


### Mobile - 576px

For the mobile screen, write these on SCSS [🔗](#)

```
// using variable here which is 🖱 576px
@media screen and (max-width : $mobile){
  .container{
    background-color: $color-1;
  }
}
```

The Result ->



### Take a Break

Congratulations for Completing Project 1 but, first, **take a break**. You deserve it !



# Take a Break

## Let's do some projects using CSS Media Queries

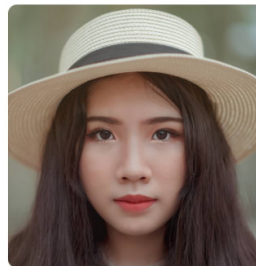
### Project-2 Responsive Portfolio

We'll build this, A small responsive Website [🔗](#)

#### Desktop View

Miya Ruma

[Home](#) [Portfolio](#) [Contact](#)



Hello 🙋

I'm **Miya Ruma**  
A Designer From  
Tokyo, Japan



#### Mobile View

9/6/2021

Learn CSS Media Queries by Building Three Projects - DEV Community

Miya Ruma



I'm **Miya Ruma**  
A Designer From  
Tokyo, Japan



Okay then, Let's start Coding ! First, let's work with the Desktop View by taking small baby steps

Before Starting

Create a folder named 'images' inside our 'Project-1' Folder. Place all the images you downloaded from my [GitHub Repository](#) inside the 'images' folder.

HTML

Step - 1

https://dev.to/artworks\_joy/learn-css-media-queries-by-building-three-projects-3a8b25/44

9/6/2021

Learn CSS Media Queries by Building Three Projects - DEV Community

<div class="header">

</div>

Step - 4

We'll place the social media icons inside the .footer div

<div class="footer">

<div class="footer\_\_instagram">



</div>

<div class="footer\_\_twitter">



</div>

<div class="footer\_\_dribbble">



</div>

<div class="footer\_\_behance">



</div>

</div>

SCSS

The **SCSS** Part !



https://dev.to/artworks\_joy/learn-css-media-queries-by-building-three-projects-3a8b27/44

9/6/2021

Learn CSS Media Queries by Building Three Projects - DEV Community

Step - 1

We'll create 3 sections for our website. Write these Codes inside HTML

<div class="container">

<div class="header"></div>

<div class="main"></div>

<div class="footer"></div>

</div>

Step - 2

We'll place the logo & menu items inside the .header div

<div class="header">

<div class="header\_\_logo">Miya Ruma</div>

<div class="header\_\_menu">

<div class="header\_\_menu-1"> Home </div>

<div class="header\_\_menu-2"> Portfolio </div>

<div class="header\_\_menu-3"> Contacts </div>

</div>

Step - 3

We'll place the image & text inside the .main div

<div class="main">

<div class="main\_\_image"></div>

<div class="main\_\_text">

<div class="main\_\_text-1">Hello 🐱</div>

<div class="main\_\_text-2">I'm <span>Miya Ruma</span></div>

<div class="main\_\_text-3">A Designer From</div>

<div class="main\_\_text-4">Tokyo, Japan</div>

https://dev.to/artworks\_joy/learn-css-media-queries-by-building-three-projects-3a8b26/44

9/6/2021

Learn CSS Media Queries by Building Three Projects - DEV Community

Step-1

Delete everything inside our SCSS & write these ->

\* {

// placing Margin to left & right

margin: 0px 5px;

padding: 0px;

box-sizing: border-box;

body {

font-family: sans-serif;

}

The result so far ->

Miya Ruma

Home

Portfolio

Contacts

Hello 🐱

I'm Miya Ruma

A Designer From

Tokyo, Japan



Step-2

Select All the classes we created in HTML on our stylesheet.

.container{}

.header{}

.main{}

.footer{}

https://dev.to/artworks\_joy/learn-css-media-queries-by-building-three-projects-3a8b28/44

### Step-3

Now select all the children of the parent classes.

```
.header{

  &__logo{}

  &__menu{}
}

.main{

  &__image{}

  &__text{}
}

.footer{

  [class ^="footer__"]{}

}
```

**Note :** &\_\_logo nested inside .header is shortcut of .header\_\_logo

### Step-4

Define the .container for desktop layout

```
.container{

  // Defining height
  height: 100vh;

  display: flex;

  flex-direction: column;
}
```

Apply display: flex; to .header & to the menu items so that it behaves like a row, not column

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

29/44

The Result ->

Miya Ruma Home Portfolio Contacts

Hello 🐼  
I'm Miya Ruma  
A Designer From  
Tokyo, Japan

### Step-5

Let's complete styling our .header section using flex-box properties & appropriate font-size

```
.header {
  // height
  height: 100%;

  display: flex;
  // Aligning logo & menu at center
  align-items: center;

  // space between logo & menu
  justify-content: space-between;

  &__logo {
    font-size: 4vw;
  }

  &__menu {
    display: flex;
    font-size: 2.5vw;

    // to put gap between menu items
    gap: 15px;
  }
}
```

The result ->

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

31/44

```
.header{
  display: flex;
  flex-direction: row;

  &__logo{}

  &__menu{
    display: flex;
    flex-direction: row;
  }
}
```

Divide each section & create borders to see what we are doing

```
.header{
  display: flex;

  // The border & height
  border: 2px solid red;
  height: 100%;

  // Other selectors are here
}

.main{

  //The border & height
  border: 2px solid black;
  height: 80%;

  // Other selectors are here
}

.footer{

  // Border & height
  border: 2px solid green;
  height: 10%;

  // Other selectors are here
}
```

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

30/44

Miya Ruma

Home Portfolio Contacts

Hello 🐼  
I'm Miya Ruma  
A Designer From  
Tokyo, Japan

### Step-6



Let's add the image inside .main section & create a partition for image & text.

```
.main {
  // image & text will act like a row
  display: flex;
  flex-direction: row;

  //The border & height
  border: 2px solid black;
  height: 80%;

  &__image {
    //Adding the image
    background-image: url("../images/Portrait.png");
    // will cover half of screen width
    width: 50%;
  }

  &__text {
```

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

32/44



```
// will cover half of screen width
width: 50%;

}
}
```

The ugly result so far, but don't lose hope !

## Miya Ruma



Hello 🙋  
I'm Miya Ruma  
A Designer From  
Tokyo, Japan

Width : 1280 px

### Step-7 - The image

Style the image to be responsive ->

```
.main{
  &__image{
    //make image fluid
    background-size: contain;

    // stop image repetition
    background-repeat: no-repeat;

    // position the image
    background-position: left center;
  }
}
```

The result so far ->

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

33/44

## Miya Ruma

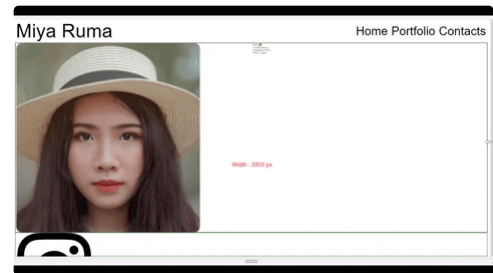
Home



Hello 🙋  
I'm Miya Ruma  
A Designer From  
Tokyo, Japan

Width : 1280 px

The image is responsive from **4k** till your **smart watch screen**. Don't believe me? Open chrome developer tools & test it yourself and see.



### Step-8 - The text

Let's style our text now. Bring it to the exact center

```
.main{
```

```
  &__text {
```

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

34/44

```
// will cover half of screen width
width: 50%;

display: flex;
flex-direction: column;

// To bring it at the center
justify-content: center;
align-items: center;
}

// To color The name
span{
  color: red;
}

}

.main{

  &__text{

// To add gaps between texts vertically
gap: 15px;

// font size for "hello"
&-1{
  font-size: 10vw;
}

// font size for other texts
&-2,&-3,&-4{
  font-size: 5vw;
}

}

}
```

The result ->

- Upto this point, you can remove all the borders we placed inside our header, main & footer classes

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

35/44

```
  &__text {
```

### Step-9 : The footer Section

First, resize the images like this ->

```
.footer{
  [class^="footer__"] {
    img {
      width: 5.3vw;
    }
  }
}
```

Then, position the images to our desired place, with some gap between the icons ->

```
.footer{
  display: flex;
  flex-direction: row;

  // To align icons along x-axis
  align-items: center;
  // placing image to the right side
  justify-content: flex-end;
  // Gap between icons
  gap: 20px;

  // margin to right side of icons
  margin-right: 10%;
}
```

The result, without the guides ->

### Step-10 : The mobile Layout

Almost there !

Create a media query at 650px mark & style the .header class like this ->

```
@media (max-width: 650px) {

  .header {

    // To place logo at center
```

[https://dev.to/artworks\\_joy/learn-css-media-queries-by-building-three-projects-3a8b](https://dev.to/artworks_joy/learn-css-media-queries-by-building-three-projects-3a8b)

36/44

```
justify-content: center;

    &__logo {
      font-size: 40px;
    }
  }
  //hiding the menu on mobile device
  &__menu {
    display: none;
  }
}
}
```

Step-11

Now, place the .main section at the exact center ->

```
@media (max-width: 650px){
  // styles of header section of step-10...

  // main section here
  .main {
    flex-direction: column;
    justify-content: center;
    align-items: center;
  }
}
```

step-12 :

Style the image & text for mobile layout. Like this ->

```
@media (max-width: 650px){

  .main {
    &__image {
      // Image size
      height: 200px;
      width: 200px;
      background-size: 100%;

      // To have rounded image
      border-radius: 100%;
      background-position: center;
    }

    // Styles for the text ->
```

```
&__text {
  width: 100%;

  &-1 {
    display: none;
  }
  &-2, &-3, &-4 {
    font-size: 30px;
  }
}
```

Step-13

The last step, Let's style the footer section for the mobile layout ->

```
@media (max-width: 650px){
  .footer {
    // placing icons along the X-axis
    justify-content: center;
    margin: 0px;

    [class*="__footer__"] {

      // Resizing images for mobile layout
      img {
        width: 45px;
        height: 45px;
      }
    }
  }
}
```

The result ->

Take a break

Good job so far ! Take a break 😊

Project-3 The Card Layout

In Project 3, We'll build this ->

Let's start !

SCSS

On your stylesheet, delete everything, but don't delete the styles of #size. And write these ->

```
* {
  margin: 0px;
  padding: 0px 10px;
  box-sizing: border-box;

  body {
    font-family: sans-serif;
    font-size: 55px;
  }
}

#size{
  position: absolute;
  // Positioning the text
  top: 60%;
  left: 50%;
  transform: translateX(-50%);
  // color & size of text
  color: red;
  font-size: 40px;
}
```

HTML

You're HTML should look like this inside the body tags 📄

```
<div class="container">
  // We'll place code here
</div>

// This will show our window width live
<div id="size"></div>
```

Now, create 3 classes with class names .row-\* like this 📄 inside .container

```
<div class="container">

  <div class="row-1">
  </div>

  <div class="row-2">
  </div>

  <div class="row-3">
  </div>
</div>

Each row will have 3 boxes with class names .box-* like this 📄
And yes, Insert Letters inside the boxes

<div class="container">

  <div class="row-1">
    <div class="box-1">A</div>
    <div class="box-2">B</div>
    <div class="box-3">C</div>
  </div>

  <div class="row-2">
    <div class="box-4">D</div>
    <div class="box-5">E</div>
    <div class="box-6">F</div>
  </div>

  <div class="row-3">
    <div class="box-7">G</div>
    <div class="box-8">H</div>
    <div class="box-9">I</div>
  </div>
</div>
```

We're done with the HTML part and the result should look like this 📄

SCSS

Follow these small baby steps one by one 📄

## Step-1

To select & style all the boxes & rows together, we do these on CSS

```
.container{
  // styles here
}

[class ^= "row-"]{
  // Styles applied on all rows
}

[class ^= "box-"]{
  // Styles applied on all boxes
}
```

## Step-2

Boxes should behave like a row. Write these ->

```
[class ^= "row-"]{
  display: flex;
  flex-direction: row;
}
```

The result

## Step-3

Expand the boxes across the width & height & place the letters at the center. Follow me ->

```
[class ^= "box-"]{

  background-color: #c4c4c4;
  border: 2px solid black;

  // Defining the size of the boxes
  width : (100%)/3;
  height: (100vh)/3;

  // Place letter at the center
  display: grid;
}
```

```
place-items: center;
}
```

The Result ->

## Step-4

create gap among the rows. Follow me ->

```
.container{
  display: flex;
  flex-direction: column;
  height: 100vh;

  // Creating gap between rows
  gap: 30px;
}
```

Now to create gap between Boxes ->

```
[class ^= "row-"]{
  display: flex;
  flex-direction: row;

  // Creating gap between boxes
  gap : 30px;
}
```

The Result ->

## Step-5 -> The mobile Layout

Create Media query which will be applied at 650px mark

```
@media (max-width: 650px){
  // We'll write code here
}
```

Change orientation of the boxes on the mobile screen from row to column, and stretch the boxes to 100% of the width ->

```
@media (max-width: 650px){
```

```
//Change orientation
[class ^= "row-"]{
  flex-direction: column;
}

// Change width of boxes
[class ^= "box-"]{
  width: 100%;
}
}
```

The Final Result ->

By the way, Project 2 is a part of [this article](#) of mine. If you're interested to learn & practice both about flexbox & media query, then go for it !

## Conclusion

Here's Your Medal For reading till the end

## Suggestions & Criticisms Are Highly Appreciated

- [YouTube / Joy Shaheb](#)
- [Twitter / JoyShaheb](#)
- [Instagram / JoyShaheb](#)

## Credits

- [CSS Tricks](#)
- [Portrait](#)
- [Images from Vecteesy](#)
- [Panda](#), & [Ice-cream](#)
- [Unicorn Pack](#) & [Kitty Avatar](#)
- [instagram](#), [Twitter](#), [Behance](#) and [Dribbble icons](#)

## Discussion (3)

Jean Pierre Chreim • Apr 27

Great Job man, I definetly needed it !

Temani Afif • Apr 26

better contact the support to get back your old account. If you own the email of the other account you can recover it easily (I guess ..)

joy.does.artworks • Apr 26

Let's hope for the best :")

[Code of Conduct](#) • [Report abuse](#)

joy.does.artworks

I lost password of my old account on Dev.to :/

JOINED  
Apr 26, 2021

Trending on DEV Community

What was your win this week?

#discuss #weeklyretro

Stop! Put Down That Ternary, Lines Are Free

#career #webdev #codenewbie #programming

September 2nd, 2021: What did you learn this week?

#weeklylearn #discuss #weeklyretro