**CSS3 Media Queries**

**What Is Media Query?**

* Media queries are used when we want to customize our website's presentation according to the user's screen size. With the help of media queries, you can display different markups based upon the device's general type(mobile, desktop, tablet).
* A media query is a logical operation. Whenever a media query becomes true, then the related CSS is applied to the target element.

**Syntax Of Media Query :**

A media query is composed of two things: media type and expression. A media query can contain one or more expressions.

@media media-type and (media-feature)

{

/\* CSS Rules to be applies\*/

}

**Example:**

@media screen and (max-width: 800px)

{

#contents{width:90%}

}

## Media Queries and Responsive Web Design

Media queries allow you to customize the presentation of your web pages for a specific range of devices like mobile phones, tablets, desktops, etc. without any change in markups. A media query consists of a media type and zero or more expressions that match the type and conditions of a particular media features such as device width or screen resolution.

Since media query is a logical expression it can be resolve to either true or false. The result of the query will be true if the media type specified in the media query matches the type of device the document is being displayed on, as well as all expressions in the media query are satisfied. When a media query is true, the related style sheet or style rules are applied to the target device. Here's a simple example of the media query for standard devices.

Example1:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>Example of CSS Media Queries</title>

<style>

/\* Smartphones (portrait and landscape) ---------- \*/

@media screen and (min-width: 320px) and (max-width: 480px){

body{

background: #7ce7e1;

}

}

/\* Smartphones (portrait) ---------- \*/

@media screen and (max-width: 320px){

body{

background: #ffd280;

}

}

/\* Smartphones (landscape) ---------- \*/

@media screen and (min-width: 321px){

body{

background: #9ddfbb;

}

}

/\* tablets, iPads (portrait and landscape) ---------- \*/

@media screen and (min-width: 768px) and (max-width: 1024px){

body{

background: #ffb497;

}

}

/\* tablets, iPads (portrait) ---------- \*/

@media screen and (min-width: 768px){

body{

background: #f0e68c;

}

}

/\* tablets, iPads (landscape) ---------- \*/

@media screen and (min-width: 1024px){

body{

background: #d6b3f4;

}

}

/\* Desktops and laptops ---------- \*/

@media screen and (min-width: 1224px){

body{

background: #d8ff9d;

}

}

/\* Large screens ---------- \*/

@media screen and (min-width: 1824px){

body{

background: #ffc0cb;

}

}

</style>

</head>

<body>

<h1>CSS Media Queries</h1>

<p>The background of the output area is different in different media or devices.</p>

<p><strong>Alternative:</strong> You can also see the effect of this media query by opening the output in a new window and resize it to different sizes.</p>

</body>

</html>

 Media queries are an excellent way to create responsive layouts. Using media queries you can customize your website differently for users browsing on devices like smart phones or tablets without changing the actual content of the page.

## Changing Column Width Based on Screen Size

You can use the CSS media query for changing the web page width and related elements to offer the best viewing experience for the user on different devices.

The following style rules will automatically change the width of the container element based on the screen or viewport size. For example, if the viewport width is less than 768 pixels it will cover the 100% of the viewport width, if it is greater than the 768 pixels but less than the 1024 pixels it will be 750 pixels wide, and so on.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>Example of Changing Element's Width Using CSS Media Queries</title>

<style>

.container {

margin: 0 auto;

background: #ceceff;

box-sizing: border-box;

}

/\* Mobile phones (portrait and landscape) ---------- \*/

@media screen and (max-width: 767px){

.container {

width: 100%;

padding: 5px 10px;

}

}

/\* Tablets and iPads (portrait and landscape) ---------- \*/

@media screen and (min-width: 768px) and (max-width: 1023px){

.container {

width: 750px;

padding: 5px 10px;

}

}

/\* Low resolution desktops and laptops ---------- \*/

@media screen and (min-width: 1024px) {

.container {

width: 980px;

padding: 5px 15px;

}

}

/\* High resolution desktops and laptops ---------- \*/

@media screen and (min-width: 1280px) {

.container {

width: 1200px;

padding: 5px 20px;

}

}

</style>

</head>

<body>

<div class="container">

<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, varius quam at, luctus dui. Mauris magna metus, dapibus nec turpis vel, semper malesuada ante. Vestibulum id metus ac nisl bibendum scelerisque non non purus. Suspendisse varius nibh non aliquet sagittis. In tincidunt orci sit amet elementum vestibulum. Vivamus fermentum in arcu in aliquam. Quisque aliquam porta odio in fringilla. Vivamus nisl leo, blandit at bibendum eu, tristique eget risus. Integer aliquet quam ut elit suscipit, id interdum neque porttitor. Integer faucibus ligula.</p>

<p>Quis quam ut magna consequat faucibus. Pellentesque eget nisi a mi suscipit tincidunt. Ut tempus dictum risus. Pellentesque viverra sagittis quam at mattis. Suspendisse potenti. Aliquam sit amet gravida nibh, facilisis gravida odio. Phasellus auctor velit at lacus blandit, commodo iaculis justo viverra. Etiam vitae est arcu. Mauris vel congue dolor. Aliquam eget mi mi. Fusce quam tortor, commodo ac dui quis, bibendum viverra erat. Maecenas mattis lectus enim, quis tincidunt dui molestie euismod. Curabitur et diam tristique, accumsan nunc eu, hendrerit tellus.</p>

</div>

</body>

</html>

**Note:** You can use the CSS3 [box-sizing](https://www.tutorialrepublic.com/css-tutorial/css3-box-sizing.php) property on the elements to create more intuitive and flexible layouts with much less effort.

## Changing Layouts Based on Screen Size

You can also use the CSS media query for making your multi-column website layout more adaptable and responsive for devices through little customization.

The following style rule will create a two column layout if the viewport size is greater than or equal to 768 pixels, but if less than that it'll be rendered as one column layout.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>Example of Making Responsive Layout Using CSS Media Queries</title>

<style>

.column {

width: 48%;

padding: 0 15px;

box-sizing: border-box;

background: #97ddff;

float: left;

}

.container .column:first-child{

margin-right: 4%;

}

@media screen and (max-width: 767px){

.column {

width: 100%;

padding: 5px 20px;

float: none;

}

.container .column:first-child{

margin-right: 0;

margin-bottom: 20px;

}

}

</style>

</head>

<body>

<div class="container">

<div class="column">

<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, varius quam at, luctus dui. Mauris magna metus, dapibus nec turpis vel, semper malesuada ante. Vestibulum id metus ac nisl bibendum scelerisque non non purus. Suspendisse varius nibh non aliquet sagittis. In tincidunt orci sit amet elementum vestibulum. Vivamus fermentum in arcu in aliquam. Quisque aliquam porta odio in fringilla. Vivamus nisl leo, blandit at bibendum eu, tristique eget risus. Integer aliquet quam ut elit suscipit, id interdum neque porttitor. Integer faucibus ligula.</p>

</div>

<div class="column">

<p>Quis quam ut magna consequat faucibus. Pellentesque eget nisi a mi suscipit tincidunt. Ut tempus dictum risus. Pellentesque viverra sagittis quam at mattis. Suspendisse potenti. Aliquam sit amet gravida nibh, facilisis gravida odio. Phasellus auctor velit at lacus blandit, commodo iaculis justo viverra. Etiam vitae est arcu. Mauris vel congue dolor. Aliquam eget mi mi. Fusce quam tortor, commodo ac dui quis, bibendum viverra erat. Maecenas mattis lectus enim, quis tincidunt dui molestie euismod. Curabitur et diam tristique, accumsan nunc eu, hendrerit tellus.</p>

</div>

</div>

</body>

</html>

# CSS Flex Responsive

<!DOCTYPE html>

<html>

<head>

<style>

\* {

box-sizing: border-box;

}

.flex-container {

display: flex;

flex-direction: row;

font-size: 30px;

text-align: center;

}

.flex-item-left {

background-color: #f1f1f1;

padding: 10px;

flex: 50%;

}

.flex-item-right {

background-color: dodgerblue;

padding: 10px;

flex: 50%;

}

/\* Responsive layout - makes a one column-layout instead of two-column layout \*/

@media (max-width: 800px) {

.flex-container {

flex-direction: column;

}

}

</style>

</head>

<body>

<h1>Responsive Flexbox</h1>

<p>The "flex-direction: row;" stacks the flex items horizontally (from left to right).</p>

<p>The "flex-direction: column;" stacks the flex items vertically (from top to bottom).</p>

<p><b>Resize the browser window to see that the direction changes when the

screen size is 800px wide or smaller.</b></p>

<div class="flex-container">

<div class="flex-item-left">1</div>

<div class="flex-item-right">2</div>

</div>

</body>

</html>

Another way is to change the percentage of the flex property of the flex items to create different layouts for different screen sizes. Note that we also have to include flex-wrap: wrap; on the flex container for this example to work:

<!DOCTYPE html>

<html>

<head>

<style>

\* {

box-sizing: border-box;

}

.flex-container {

display: flex;

flex-wrap: wrap;

font-size: 30px;

text-align: center;

}

.flex-item-left {

background-color: #f1f1f1;

padding: 10px;

flex: 50%;

}

.flex-item-right {

background-color: dodgerblue;

padding: 10px;

flex: 50%;

}

/\* Responsive layout - makes a one column-layout instead of a two-column layout \*/

@media (max-width: 800px) {

.flex-item-right, .flex-item-left {

flex: 100%;

}

}

</style>

</head>

<body>

<h1>Responsive Flexbox</h1>

<p>In this example, we change the percentage of flex to create different layouts for different screen sizes.</p>

<p><b>Resize the browser window to see that the direction changes when the

screen size is 800px wide or smaller.</b></p>

<div class="flex-container">

<div class="flex-item-left">1</div>

<div class="flex-item-right">2</div>

</div>

</body>

</html>