CSS Media Query

In CSS, media types allow you to target specific types of devices or media when applying styles.

The full syntax for a CSS media query allows you to apply styles conditionally based on the characteristics of the device or viewport. Here's a breakdown of the complete syntax, including different components you might use:

Basic Syntax

```
@media [not | only] media-type and (media-feature) {
  /* CSS rules go here */
}
```

Components of a Media Query

@media Keyword:

   - The @media rule initiates a media query and tells the browser to apply the enclosed CSS rules only if the conditions are met.

Media Types:

- screen: Targets screens (desktops, tablets, smartphones, etc.).
- all: Targets all devices.
- print: Targets print preview or printed documents.
- speech: Targets screen readers.

Screen

Purpose: Targets devices that have a screen, such as computers, tablets, and smartphones.

Use Case: This is the most common media type used in responsive web design to adapt the layout and styles for various screen sizes.

Example:

```
@media screen and (min-width: 768px) {
    body {
      background: black;
    }
  }
```

 Explanation: The above code will apply the specified styles only when the content is displayed on a screen device with a width of 768px or more.

Print

Purpose: Targets documents being printed or previewed in print mode.

Use Case: It allows you to create styles that are optimized for printing, such as setting page margins, hiding unnecessary elements like navigation menus, or changing colors to grayscale.

Example:

```css
@media print {
  body {
    font-size: 12pt;
    color: black;
    background: white;
  }

  .no-print {
    display: none;
  }
}
```

 Explanation: The above code will apply styles only when printing the document. The text size is adjusted, and all elements with the class .no-print will be hidden from the printed version.

All:

Purpose: Targets all media types, meaning it applies styles regardless of the device or medium being used.

 Use Case: This media type is used when you want a specific set of styles to be applied universally, no matter if the content is viewed on a screen, printed, or accessed in another way.

Example:

```css
@media all and (min-width: 763px){
  body {
    margin: 0;
    padding: 0;
  }
}
```

Explanation: The styles specified here will apply across all devices and media, ensuring consistency in how elements like margins and padding are rendered.

Speech

Purpose: Targets screen readers and other devices that "speak" the content.

Use Case: This media type is used for accessibility purposes to optimize the reading experience for users relying on assistive technology.

Example:

```
@media speech {
  body {
    voice-family: male;
    pitch: x-low;
    speak: literal-punctuation;
  }
}
```

Explanation: The above code defines how text should be read aloud by a screen reader, setting a specific voice and reading style.

Optional Modifiers:

- only: Applies the styles only if the entire media query matches (used to prevent older browsers from applying the styles).
- not: Negates whatever the rule is.

Media Features:

Media features are conditions based on device characteristics such as width, height, resolution, etc. They can be used with or without a value:

Width and Height:

- width, min-width, max-width
- height, min-height, max-height

Aspect Ratio:

- aspect-ratio, min-aspect-ratio, max-aspect-ratio

Resolution:

- resolution, min-resolution, max-resolution
- Orientation:
- orientation (values: portrait or landscape)

Others:

- color, monochrome, scan, grid

Examples of Media Query Syntax

Basic Example:

```css
@media screen and (min-width: 600px) {
   body {
     background-color: green;
   }
}
```

Explanation: Applies the background-color style when the screen width is 600px or more.

Using Multiple Media Features:

```css
@media screen and (min-width: 768px) and (orientation: landscape) {
   body {
     font-size: 18px;
   }
}
```

Explanation: The styles apply when the screen width is 768px or more and the device is in landscape orientation**.**

Using not Modifier:

```css
@media not screen and (min-width: 768px) {
   body {
     background-color: lightgreen;
   }
}
```

Explanation: The background color changes to light green for all devices except those with screens 768px wide or larger.

Using only Modifier:

```css
@media only screen and (max-width: 600px) {
   body {
     background-color: yellow;
   }
}
```

Explanation: The styles apply only to devices with a screen and a width of 600px or less, preventing older browsers that don't understand media queries from applying these styles.

Feature Without Media Type:

```css
@media (min-width: 1024px) {
    body {
      font-size: 20px;
    }
  }
```

Explanation: Applies the styles when the viewport is at least 1024px wide, regardless of the media type.

Nested Media Queries

Media queries can also be nested within CSS rules, allowing for more high level of control:

```css
.container {
  width: 100%;
  background-color: white;

  @media (min-width: 768px) {
    width: 50%;
    background-color: gray;
  }
}
```

In CSS media queries, min-width and max-width are used to apply styles based on the width of the viewport or screen. They help create responsive designs by targeting specific ranges of screen sizes.

min-width

Purpose: Targets devices or viewports "greater than or equal to" a specified width.

Behavior: Styles are applied if the viewport's width is "at least" the value specified by min-width.

Use Case:

Commonly used in a "mobile-first approach", where styles for smaller screens are defined first, and min-width queries add styles for larger screens.

Example: Applying styles for tablets and desktops.

```css
Example:
  @media screen and (min-width: 768px) {
    body {
      background-color: blue;
    }
  }
```

Explanation: The background color changes to light blue on devices with a viewport width of "768px or more" such as tablets and desktops.

max-width

- Purpose: Targets devices or viewports "less than or equal to" a specified width.

- Behavior: Styles are applied if the viewport's width is "at most" the value specified by max-width.

Use Case:

Commonly used in a "desktop-first approach", where styles for larger screens are defined first, and max-width queries adjust styles for smaller screens.

Example: Applying styles for mobile devices.

Example:

```css
@media screen and (max-width: 768px) {
  body {
    background-color: yellow;
  }
}
```

Explanation: The background color changes to light green on devices with a viewport width of "768px or less", such as smartphones and smaller tablets.

Key Differences:

Application Range:

- min-width: Applies styles "starting from" a certain width and up.
- max-width: Applies styles "up to and including" a certain width.

Usage in Responsive Design:

- min-width: Ideal for progressively enhancing the design for larger screens (mobile-first).
- max-width: Ideal for scaling down the design for smaller screens (desktop-first).

Example Comparison:

```css
Mobile-First Approach (Using min-width)
  /* Mobile styles (default) */
  body {
    font-color: red;
  }

  /* Tablet and up */
  @media screen and (min-width: 768px) {
    body {
      font-color: blue;
    }
```

```
    }
Desktop-First Approach (Using max-width):
  /* Desktop styles (default) */
  body {
    font-size: 16px;
  }

  /* Mobile and down */
  @media screen and (max-width: 768px) {
    body {
      font-size: 14px;
    }
  }
```

Conclusion:

min-width is used when you want to start applying styles at a certain width and continue applying them for larger viewports.

max-width is used when you want to apply styles up to a certain width, typically for smaller screens.

Both are crucial in creating responsive designs, allowing you to adapt your layout and styles to different screen sizes effectively.