educación IT
CODE YOUR FUTURE

Bootcamp Java Developer

Fase 2 - Java Web Developer

Módulo 14

Regla @media

El **`meta viewport`** tiene diferentes variantes y características.

La propiedad **`initial-scale` controla el zoom al ingresar** a la interfaz. De esa manera **evitamos los acercamientos o *zoom in* tan abruptos** que sucedían antes.

El valor máximo es 10 y el mínimo 0.1, siendo el valor por defecto 1.0 o 1.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Proyecto Final</title>
```

Viewport

Adaptabilidad

Esta regla me permite adaptar o destinar ciertos estilos para determinados medios o tamaños de pantalla.

Para esto, es importante **incluir el `meta viewport`, que indica cuál es el tamaño de área visible** desde el dispositivo que utiliza el usuario.

```
<!doctype html>
<title> Viewport </title>


<meta name="viewport" content="width=device-width; initial-scale=1.0; maximum-scale=1.0;
user-scalable=0;" />
```

@media

Permite desde el css, destinar estilos específicamente para ciertas situaciones:

```
@media (min-width: 300px){

body { background-color: red;}


}
```

Por ejemplo, en este caso, se pide que el fondo del documento, cambie a color rojo en pantallas con un tamaño de 300px hacia arriba, es decir más grandes.

También se pueden **fijar máximos:**

```
@media (max-width: 300px){

body { background-color: red;}


}
```

En la imagen, se ordena que se cambie a color rojo el fondo de nuestro documento en pantallas de 300px hacia abajo, es decir de menor tamaño.

También se puede agregar un mínimo escalable y un máximo, por ejemplo:

```
<meta name="viewport" content="initial-scale=1, maximum-scale=1">
```

- **`minimum-scale`:** controla la cantidad de *zoom-out* permitido en la página.
- **`maximum-scale`:** controla cuánto *zoom-in* es permitido. Se puede complementar con `minimum-scale` y así fijar un rango.

Una característica interesante es el **`width`**, o el **ancho** al que se va a **adaptar nuestra interfaz**.

El valor, si bien puede estar en **px**, es interesante que sepamos que el valor mayormente utilizado, **`device-width`**, es el equivalente al **`width`** de la pantalla, es decir, **se adapta a la pantalla** sin importar cuál sea su tamaño.

También existe el atributo **`height`** (altura), **con su correspondiente `device-height`**.

De esta forma, podemos comenzar a trabajar con nuestro css para que, cuando se detecte un cambio de tamaño de pantalla, se tomen los estilos pertinentes al caso.

```
<meta name="viewport" content="width=500, initial-scale=1" />
```

También, la regla `@media` permite **fijar rangos:**

```
@media (min-width: 700px) and (orientation: landscape){

body { background-color: red;}


}
```

O en el caso de la imagen, fijar **orientaciones como horizontal o apaisada**.

Los valores de orientation posibles son:

- **`landscape:`** el `width` del *viewport* es mayor que el `height`.
- **`portrait:`** el `height` del *viewport* es mayor que el `width`.

La **media** también se puede ubicar en el elemento **link** y no necesariamente en la **regla @media**:

```
<link rel="stylesheet" media="(max-width: 800px)" href="example.css" />
```

De esta manera entonces, podemos avanzar con nuestra adaptabilidad y setear reglas específicamente para determinado tamaño de pantalla o medio, por ejemplo, la impresión:

```
<link rel="stylesheet" media="print" href="impresion.css" >
```

También puede incluirse como visualizamos en la imagen debajo. Puede que esta forma resulte más práctica, dado que tendremos **todos los estilos, sean de impresión o no, en una misma hoja.**

```
@media print {

  p {
    color: blue;
  }

}
```

Revisión

- Repasar los conceptos de **`media`**.
- Implementar estas nuevas propiedades.
- Ver todos los videos y materiales necesarios antes de continuar.

¡Sigamos trabajando!