Pautas para escribir un CSS correcto

1. Posibles errores

> Las siguientes pautas evitan errores comunes

1. Comprueba que no existan propiedades duplicadas.
2. Asegurate que no existen reglas con bloques vacios.
3. Asegurate que la propiedad es una propiedad conocida.

2. Compatibilidad

> Las siguientes pautas de compatibilidad evitan problemas en los browsers y sus ajustes

1. Especifica la propiedad box-sizing cuando uses las siguientes combinaciones:

   a. width agrupado con las propiedades border, border-left, border-right, padding, padding-left, or padding-right.

   b. height agrupado con las propiedades border, border-top, border-bottom, padding, padding-top, or padding-bottom.

   El siguiente ejemplo muestra codigo incorrecto:

```
/* width and border /
.mybox {
border: 1px solid black;
width: 100px;
}
/ height and padding */
.mybox {
height: 100px;
padding: 10px;
}
```

2. Si la propiedad text-indent tiene valor negativo, en el mismo bloque tienes que especificar la propiedad direction con el valor ltr.
3. Cuando definas propiedades vendor-prefixed incluye la propiedad standard situándola en el último lugar.

   El siguiente código es incorrecto:

```
        /* missing standard property */
        .mybox {
            -moz-border-radius: 5px;
        }
        /* standard property should come after vendor-prefixed property */
        .mybox {
```

```
    border-radius: 5px;
    -webkit-border-radius: 5px;
}
```

El siguiente patron es considerado correcto:

```
/* both vendor-prefix and standard property */
.mybox {
    -moz-border-radius: 5px;
    border-radius: 5px;
}
```

4. Si quieres el mismo efecto en todos los browsers, recuerda especificar todas las propiedades vendor-prefixed para todos los browsers.

5. Si usas uno de los siguientes valores (para las propiedades color o background): rgba(), hsl(), or hsla(), asegurate que van precedidas por la propiedad color con el formato antiguo.

   El siguiente código es incorrecto:

```
/* missing fallback color */
.mybox {
    color: rgba(100, 200, 100, 0.5);
}
/* missing fallback color */
.mybox {
    background-color: hsla(100, 50%, 100%, 0.5);
}
/* missing fallback color */
.mybox {
    background: hsla(100, 50%, 100%, 0.5) url(foo.png);
}
/* fallback color should be before */
.mybox {
    background-color: hsl(100, 50%, 100%);
    background-color: green;
}
```

   El siguiente código es correcto:

```
`
/* fallback color before newer format */
.mybox {
    color: red;
    color: rgba(255, 0, 0, 0.5);
}
```

6. No precedas ninguna propiedad con un asterisco.

7. No precedas ninguna propiedad con un guión bajo.

8. Para compatibilidad con versiones antiguas de IE añade una almohadilla (#) despues del primer URL font.

   Este es un ejemplo correcto:

```
@font-face {
    font-family: 'MyFontFamily';
    src: url('myfont-webfont.eot?#iefix') format('embedded-opentype'),
        url('myfont-webfont.woff') format('woff'),
        url('myfont-webfont.ttf')  format('truetype'),
        url('myfont-webfont.svg#svgFontName') format('svg');
}
```

El siguiente es un ejemplo incorrecto:

```
@font-face {
    font-family: 'MyFontFamily';
    /* First web font is missing query string */
    src: url('myfont-webfont.eot') format('embedded-opentype'),
        url('myfont-webfont.woff') format('woff'),
        url('myfont-webfont.ttf')  format('truetype'),
        url('myfont-webfont.svg#svgFontName') format('svg');
}
```

9. Usa las propiedades adecuadas para los distintos valores de la propiedad display.

   Algunas propiedades del bloque son ignoradas debido al 'display' del elemento. Esto puede llevar a no saber como la regla puede funcionar. Las propiedades width, height, margin-top, margin-bottom, y float no tienen efecto para la propiedad display 'inline' porque elementos con este 'display' no tienen una box a la cual aplicar estilos. Otras reglas basadas en display son:

   - display: inline-block no uses float.
   - display: block no uses vertical-align.
   - display: table-* no uses margin (y todas sus variantes) o float.
   - display: inline no uses width, height, margin, margin-top, margin-bottom, and float.
   - display: inline-block used with float.

     Los siguientes patrones son incorrectos:

```
/* inline with height */
.mybox {
    display: inline;
    height: 25px;
}
/* inline-block with float */
.mybox {
    display: inline-block;
    float: left;
}
/* table-cell and margin */
.mybox {
    display: table-cell;
    margin: 10px;
}
```

3. Estilo

> Estas convenciones hacen tu código mas fácil de leer y mantener

1. Asegúrate de que todas las sentencias terminan en punto y coma ";" .
2. Asegúrate que las propiedades están indentadas dos espacios.
3. Deja un espacio detras de los dos puntos ":" al especificar el valor de la propiedad.
4. Asegurate que no hay espacios en blanco detras del caracter punto y coma ";".
5. Usa una linea para definir una propiedad y su valor.
6. Deja una linea en blanco entre dos bloques de regla.
7. Limita el ancho de las lineas a 80 caracteres, salvo para url y gradientes.
8. Comienza cada nueva seccion importante del proyecto con un titulo.

4. Semántica

1. El nombre de la clase debe ser aclaratorio del efecto que provoca la misma en el elemento.

5. Rendimiento

> Estas reglas mejoran el rendimiento de tu aplicacion. Aplícalas.

1. Utiliza clases en lugar de nombres de elementos para obtener un rendimiento de renderizado óptimo.
2. Evita el uso de selectores de atributos ([class ^ = "..."]) en componentes comunes. El rendimiento del navegador se ve afectado por estos.
3. Manten los selectores cortos y trata de limitar el número de elementos en cada selector a tres.
4. Usa clases de ámbito al padre más cercano sólo cuando sea necesario (por ejemplo, cuando no se utilizan clases prefijadas).
5. Si el valor de una propiedad es 0 no debe llevar unidades (px, em, %, etc).
6. Evita el uso de reglas que contengan el selector universal.
7. Usa shorthand y no definas todas sus propiedades.

   El siguiente ejemplo es incorrecto:

```
.mybox {
    margin-left: 10px;
    margin-right: 10px;
    margin-top: 20px;
    margin-bottom: 30px;
}
.mybox {
    padding-left: 10px;
    padding-right: 10px;
```

```
    padding-top: 20px;
    padding-bottom: 30px;
}
```

Este ejemplo es correcto:

```
/* only two margin properties*/
.mybox {
    margin-left: 10px;
    margin-right: 10px;
}
/* only two padding properties */
.mybox {
    padding-right: 10px;
    padding-top: 20px;
}
/* shortand */
.mybox {
    padding: 20px 10px 20px 10px;
}
```

8. Evita el uso de atributos no-cualificados como selectores clave.
   Puedes usarlo como parte de una regla, pero recuerda que no sea selector clave.
   Nota: selector clave es el que esta mas a la derecha de la regla.

   Este ejemplo es incorrecto:

```
[type=text] {
    color: red;
}
.selected [type=text] { // [type=text] es selector clave en esta regla
    color: red;
}
```

   Este ejemplo es correcto:

```
/* unqualified attribute selector is not key */
.selected [type=text] a { // a es el selector clave en esta regla
    color: red;
}
```

9. No escribas el nombre del elemento en el selector a menos que el nombre del elemtento provoque un comportamiento diferente.

   El siguiente ejemplo es incorrecto:

```
div.mybox {
    color: red;
}
.mybox li.active {
    background: red;
}
```

El siguiente ejemplo es correcto:

```
/* Two different elements in different rules with the same class */
li.active {
    color: red;
}
p.active {
    color: green;
}
```

6. Mantenibilidad y duplicación

> Estas pautas te ayudan a asegurar un código mantenible y entendible por otros.

1. Ordena las propiedades por orden alfabético dentro de los bloques.
2. El uso de !important, es una mala práctica y debe evitarse. Si necesitas priorizar una propiedad aumenta la especificidad de la regla. y usa !important como caso expecional solo en declaraciones específicas de CSS que sobrescriban CSS foráneo (de librerías externas como Bootstrap o normalize.css o material).
3. Evita el uso de ID como selectores
4. Limita el uso de floats a 10 por ficheros.
5. Limita las declaraciones de fonts a 4.

7. Accesibilidad

> Estas pautas están diseñadas para evitar posibles problemas de accesibilidad.

Las siguientes pautas aseguran que los usuarios keyboard-only tengan un indicador visual de 'focus':

1. Evita outline: none o outline: 0 en aquellos selectores de reglas que no contengan :focus.
2. Evita outline: none o outline: 0 en aquellas reglas cuyos selectores contengan :focus y la regla no contenga otra propiedad visual.

   Este ejemplo es incorrecto:

```
/* no :focus */
a {
    outline: none;
}
/* no :focus */
a {
    outline: 0;
}
/* :focus but missing a replacement treatment */
a:focus {
    outline: 0;
}
```

El siguiente es un ejemplo correcto:

```
/* :focus with outline: 0 and provides replacement treatment */
a:focus {
    border: 1px solid red;
    outline: 0;
}
```

8. OOCSS

> Estas pautas estan basadas en los principios de OOCSS.

1. No uses headings(h1-h6) en el ultimo lugar de la regla.

   Ejemplo incorrecto:

   ```
   /* qualified heading */
   .box h3 {
       font-weight: normal;
   }
   /* qualified heading */
   .item:hover h3 {
       font-weight: bold;
   }
   ```

   Este ejemplo esta considerado correcto:

   ```
   /* Not qualified */
   h3 {
       font-weight: normal;
   }
   ```

2. No dupliques las declaraciones de los elementos headings (h1-h6).