Cascada y Herencia

CSS:cascada y la herencia (orden de aplicación), dos mecanismos de CSS que definen las reglas que se aplican para decidir el valor final que toma una propiedad de un elemento concreto
de una página web.
En el cálculo se tiene que tener en cuenta el valor especificado, el valor calculado, el valor usado y finalmente el valor real.

inherit

es un valor permitido en todas las propiedades
Hace que el elemento al cual se aplica tome el valor calculado de la propiedad de su elemento padre.

Orden de Cascada

Cuando se usan varias hojas de estilo, puede haber un conflicto sobre cual controla a un selector en particular.
En estas situaciones, debe haber reglas para definir la regla de la hoja de estilo que prevalecerá.

00

	A	B	C	D
#id {color:red;}	0	1	0	0
ul li.important {color:green;}	0	0	1	2
ul li {color:blue;}	0	0	0	2
li.important {color:yellow;}	0	1	1	1
li {color:purple;}	0	0	0	1
style="color:pink"	1	0	0	0

Las hojas de estilo también pueden primar sobre hojas de estilo en conflicto basándose en su nivel de especificidad,
donde un estilo más específico siempre prevalecerá sobre uno menos específico.
Simplemente es cuestión de contar para calcular la especificidad de un selector.
a.Cuente el número de atributos ID en el selector.
b.Cuente el número de atributos CLASS en el selector.
c.Cuente el número nombres de etiquetas HTML en el selector.

Finalmente, escriba los tres números en orden exacto sin espacios ni comas para obtener un número de tres dígitos.

```
#id1         {xxx} /* a=1 b=0 c=0 --> especificidad = 100 */
UL UL LI.red {xxx} /* a=0 b=1 c=3 --> especificidad = 013 */
LI.red       {xxx} /* a=0 b=1 c=1 --> especificidad = 011 */
LI           {xxx} /* a=0 b=0 c=1 --> especificidad = 001 */
```

La lista final de números que corresponden a selectores determinará fácilmente la especificidad,
donde los números más altos priman sobre los más bajos. La siguiente es una lista de selectores ordenados por especificidad:
¿Cómo se calcula la especificidad?
La especificidad es una ponderación que se aplica a una declaración CSS determinada, determinada por el número de cada tipo de selector en el selector coincidente. Cuando varias declaraciones tienen la misma especificidad, la última declaración encontrada en el CSS se aplica al elemento. La especificidad solo se aplica cuando el mismo elemento es el objetivo de múltiples declaraciones.
Según las reglas CSS, los elementos directamente dirigidos siempre tendrán prioridad sobre las reglas que un elemento hereda de su antepasado.

Orden de especificación

Para hacerlo sencillo, cuando dos reglas tienen el mismo peso, prima la última regla especificada

Un ejemplo de especificidad con CSS
Con el siguiente código CSS:

```
p{
background: crimson;//Especificidad de 1 punto
}
.pc{
background: pink;//Especificidad de 10 puntos
}
p.pc{
background: maroon;//Especificidad de 11 puntos
}
#pid{
background: orange;//Especificidad de 100 puntos
}
p#pid{
background: red;//Especificidad de 101 puntos
}
p.pc#pid{
background:green;//Especificidad de 111 puntos
}
```

los estilos inline tendremos una especificidad puntuadao,
un estilo inline recibe una puntuación de 1000.

La última regla gana
usando exactamente los mismos selectores y haciendo uso de estilos diferentes.

```
1
h1 class="hello-header">Hello World!/h1

.hello-header {
  color: blue
}

.hello-header {
  color: red
}
```

Los selectores de IDs invalidan a los selectores de atributos

```
h1#hello-header {
  color: green
}

h1[id=hello-header] {
  color: red
}
```

Estos dos estilos se enfocan en el ID del elemento, pero la primera declaración lo hace usando el selector de ID (0100 puntos),
mientras que la segunda lo hace a través de un selector de atributos (0010 puntos).

Selectores universales
Los selectores universales se enfocan en cualquier elemento de la página en su totalidad. Este es un ejemplo de un selector universal:

```
* {
  color: green;
}
```

También pueden ser usados como selectores hijos (por ejemplo: div * {}).
Este tipo de selectores tiene una especificidad de 000

Las siguientes características determinarán el resultado de hojas de estilo que se contraponen.

! important

Puede establecerse una regla como importante al especificar ! important.
Un estilo designado como importante prevalecerá sobre estilos contradictorios de similar nivel.
Asimismo, ya que tanto el autor y el lector pueden especificar reglas importantes,
la regla del autor primará sobre la regla del lector en casos de importancia.
Una muestra del uso de la sentencia ! important:

```
BODY { background: url(bar.gif) white;
       background-repeat: repeat-x ! important}
```

Origen de las reglas (del autor frente a las del lector)
Como ya se mencionó, tanto los autores como los lectores tienen la capacidad de especificar hojas de estilo.
Cuando hay un conflicto entre reglas, las reglas del autor prevalecerán sobre las reglas del lector de similar peso.
Tanto las hojas de estilo del autor como las del lector primarán sobre las hojas de estilo incorporadas del navegador.

Los autores deberían ser cautelosos con el empleo de reglas ! important ya que primarán sobre las reglas ! important del lector.
Un usuario puede, por ejemplo, necesitar de grandes tamaños de fuentes o colores especiales debido a problemas de visión,
y tal usuario querrá declarar que ciertas reglas de estilo sean ! important, ya que algunos estilos son vitales para que pueda leer una página.
Cualquier regla ! important prevalecerá sobre las reglas normales, por lo que se aconseja a los autores usar reglas normales casi exclusivamente
para asegurar que los usuarios con necesidades especiales de estilos puedan leer la página.

Reglas de estilo CSS de usuario y de autor

Veamos qué son las reglas de estilo de usuario, configuradas opcionalmente por cada usuario en su navegador,
y las reglas de estilo de autor, que define el desarrollador de cada web.

- Las reglas de estilo de usuario (user stylesheet rules) las define cada persona en su navegador, a modo de configuración global, para todas las páginas que visita.

- Las reglas de estilo de autor (author stylesheet rules) son las que definen los autores de las páginas, es decir, los diseñadores o desarrolladores de cada una de las páginas que visitamos.

  Por decirlo de otra forma, las hojas de estilo de autor son las CSS

Las reglas de estilo de usuario son menos importantes para el navegador, de cara al orden de precedencia o prioridad.

Es decir, en caso que en las reglas de estilo de usuario y de autor se defina una misma propiedad de estilos CSS,

la que se tiene en cuenta es la regla de estilo de autor, osea, lo que haya configurado el diseñador de la pá
Sin embargo, esto se puede cambiar puntualmente si el usuario lo desea.

important! se puede utilizar en las hojas de estilo de usuario, para que cada persona pueda definir para su propio navegador si lo desea,

un estilo CSS por defecto que se tenga en cuenta en todas las web que visitemos. Este caso ya se explicó en el artículo Reglas de estilo CSS

de usuario y de autor.

Uso y efecto de la declaración !important

Para utilizar !important en una regla de estilo, siempre se coloca en la parte del valor del atributo, antes del punto y coma ";"

```
body{ font-family: verdana, arial !important; }
```

La excepción !important
Cuando se utiliza una regla en una declaración de estilo, esta declaración reemplaza cualquier otra declaración.
Aunque técnicamente no tiene nada que ver con la especificidad, interactúa directamente con ella.
El uso es una mala práctica y debe evitarse porque dificulta la depuración al interrumpir la cascada natural en las hojas de estilo.
Cuando se aplican dos declaraciones en conflicto con la regla al mismo elemento, se aplicará la declaración con una mayor especificidad.
important!important!important,!important

Nivel de importacncia	origen	!important
1	user agent	
2	user agent	!important
3	user	
4	autor	
5	autor	!important
6	user	!important

Reglas-at

Una **regla-at** es una declaración CSS que comienza con el símbolo arroba, '@', seguido por un identificador, e incluye todo el contenido hasta el siguiente punto y coma, ';' o el siguiente bloque CSS, lo que sea primero.

Hay varias reglas-at designadas por sus identificadores, cada una con sintaxis distinta:

- @charset — Define el conjunto de caracteres usado por la hoja de estilos.
- @charset "UTF-8";

- @import — Indica al motor de CSS que incluya una hoja de estilos externa.
- @import url; /* Representa la ubicación del recurso a importar. La url puede ser absoluta o relativa. */
  @import url list-of-media-queries;
  @import url("fineprint.css") print;
  @import 'custom.css';

  Es una lista separada por comas de consultas de medios (media query)
  que condicionan la aplicación de las reglas CSS definidas en el enlace url.

  La declaración !important asociada al estilo de un elemento ignora la especificidad del resto de estilos aplicados a ese elemento y aplica los estilos marcados con dicha declaració
  .

- @namespace — Indica al motor de CSS que todo el contenido usa como prefijo un espacio de nombres XML.

- @namespace url(XML-namespace-URL); @namespace "XML-namespace-URL"

- ***Reglas-at anidadas*** — Un subconjunto de declaraciones anidadas, que pueden ser usadas como declaraciones de estilos, así como grupos de reglas condicionadas internas:

  - @media — Un grupo de reglas condicional que aplicará su contenido si el dispositivo cumple los criterios de las condiciones definidas usando un *media query*.

  - @media <media-query-list> {
      <group-rule-body>
    }

  - @supports — Un grupo de reglas condicional que aplicará su contenido si el navegador cumple los criterios de la condición dada.

  - @document — Un grupo de reglas condicionadas que aplicará su contenido si el documento donde se aplica la hoja de estilos cumple los criterios de la condición dada. *(diferida al Nivel 4 de la Especificación CSS)*

  - @document url("https://www.example.com/") { h1 {color: green; }
    }

  - @page — Describe los cambios en la disposición de la página que serán aplicados al imprimir el documento.

  - @page <page-selector-list> {<page-body> }

  - @font-face — Describe la configuración de fuentes externas que se descargarán.

  - ```
    @font-face {font-family: "Bitstream Vera Serif Bold";
    src: url("http://developer.mozilla.org/@api/deki/files/2934/=VeraSeBd.ttf");
    }
    ```

  - @keyframes — Describe la configuración de pasos intermedios en una secuencia de animación CSS.

  -
    ```

```
    Reglas de @
    (Reglas-At o at-rules)

    Son declaraciones CSS que encapsulan un grupo de reglas, pero no están directamente relacionadas a elementos HTML/XML. Controlan la forma en que se aplican los estilos extendiendo la capacidad de CSS y cada una tiene su propia sintaxis.

    Reglas generales

    Se aplican a todo el CSS

    @charset Define el conjunto de caracteres usado en el CSS.
    @import Permite incluir otro CSS.
    @namespace Establece el XML a ser usado.
    Reglas anidadas

    Pueden estar como declaración de estilo o como condicionales.

    @media Establece reglas condicionales de acuerdo a las características del dispositivos de salida (tamaño de pantalla, tipo de impresora, dispositivos braille, etc.)

    all

    Aplicable a todos los dispositivos.

    print

    Destinado material paginado y para documentos visibles en pantalla en modo de vista previa para impresión.

    screen

    Destinado a principalmente a pantallas de computadora a color.

        speech

    Destinado a sintetizadores de voz. Nota: CSS2 tenía un tipo de medio similar llamado 'aural' para este propósito.
    V

    @supports Establece reglas condicionales según el navegador utilizado.
    @document Restringe las reglas contenidas según la URL. @document url("https://www.example.com/") { h1 {   color: green; }
}
    @page Restringe las reglas contenidas para cuando se quiere imprimir (en impresora) el documento.
    @font-face Incluye fuentes (tipográficas) externas.
    @keyframes Controla los pasos intermedios en una secuencia de animación.
    @viewport Restringe las reglas según el tamaño y orientación de la ventana, especialmente útiles en celulares.
    @counter-style Define estilos de contador específicos.sólo implementada en Gekko Motor de renderizado
    @font-feature-values, @swash, @ornaments, @annotation, @stylistic, @styleset y @character-variant Define nombre



@keyframes identifier {
    0% { top: 0; left: 0; }
    30% { top: 50px; }
    68%, 72% { left: 50px; }
    100% { top: 100px; left: 100%; }
    }
```

- @viewport — Describe los aspectos del viewport para dispositivos de pantalla pequeña. *(actualmente en Borrador)*

```
@viewport {
   <group-rule-body>
  }

  @viewport {
    min-width: 640px;
    max-width: 800px;
  }
@viewport {
    zoom: 0.75;
    min-zoom: 0.5;
    max-zoom: 0.9;
  }
@viewport {
    orientation: landscape;
  }
```

- @counter-style — Define estilos de contador específicos que no son parte de los conjuntos de estilos predeterminados. *(en estado de Recomendación Candidata, pero sólo implementada en Gekko al momento de esta publicación)*

```
@counter-style circled-alpha {
    system: fixed;
    symbols: Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ Ⓕ Ⓖ Ⓗ Ⓘ Ⓙ Ⓚ Ⓛ Ⓜ Ⓝ Ⓞ Ⓟ Ⓠ Ⓡ Ⓢ Ⓣ Ⓤ Ⓥ Ⓦ Ⓧ Ⓨ Ⓩ;
    suffix: " ";
  }
```

- @font-feature-values (junto con @swash, @ornaments, @annotation, @stylistic, @styleset y @character-variant)

  Define nombres comunes para la propiedad font-variant-alternates.

- pero sólo implementada en Gekko al momento de esta publicación)

```
@font-feature-values Font One { /* Cómo activar nice-style en Font One */
    @styleset {
      nice-style: 12;
    }
  }

  @font-feature-values Font Two { /* Cómo activar nice-style en Font Two */
    @styleset {
      nice-style: 4;
    }
  }
```

Grupos de Reglas Condicionales

Así como los valores de las propiedades, cada regla-at tiene sintaxis propia.

Sin embargo, muchas de esas reglas pueden ser agrupadas en categorías especiales, llamadas **conditional group rules**.

Estas declaraciones comparten sintáxis común y cada una puede incluir *nested statements*—ya sean *conjuntos de reglas* o *reglas-at anidadas*.

Además, pueden transmitir un significado semántico común—todas incluyen algun tipo de condición, que devuelve un resultado que puede ser **verdadero** o **falso**.

Si el valor de la condición resulta **verdadero**, todas las declaraciones del grupo serán aplicadas.

Los grupos de reglas condicionales están definidos en CSS Conditionals Level 3 y son:

- @media,

- @supports,
- @document. *(diferida al Nivel 4 de la Especificación CSS)*

Como cada grupo de condición puede incluir también declaraciones anidadas, puede haber un número de anidaciones ilimitado.