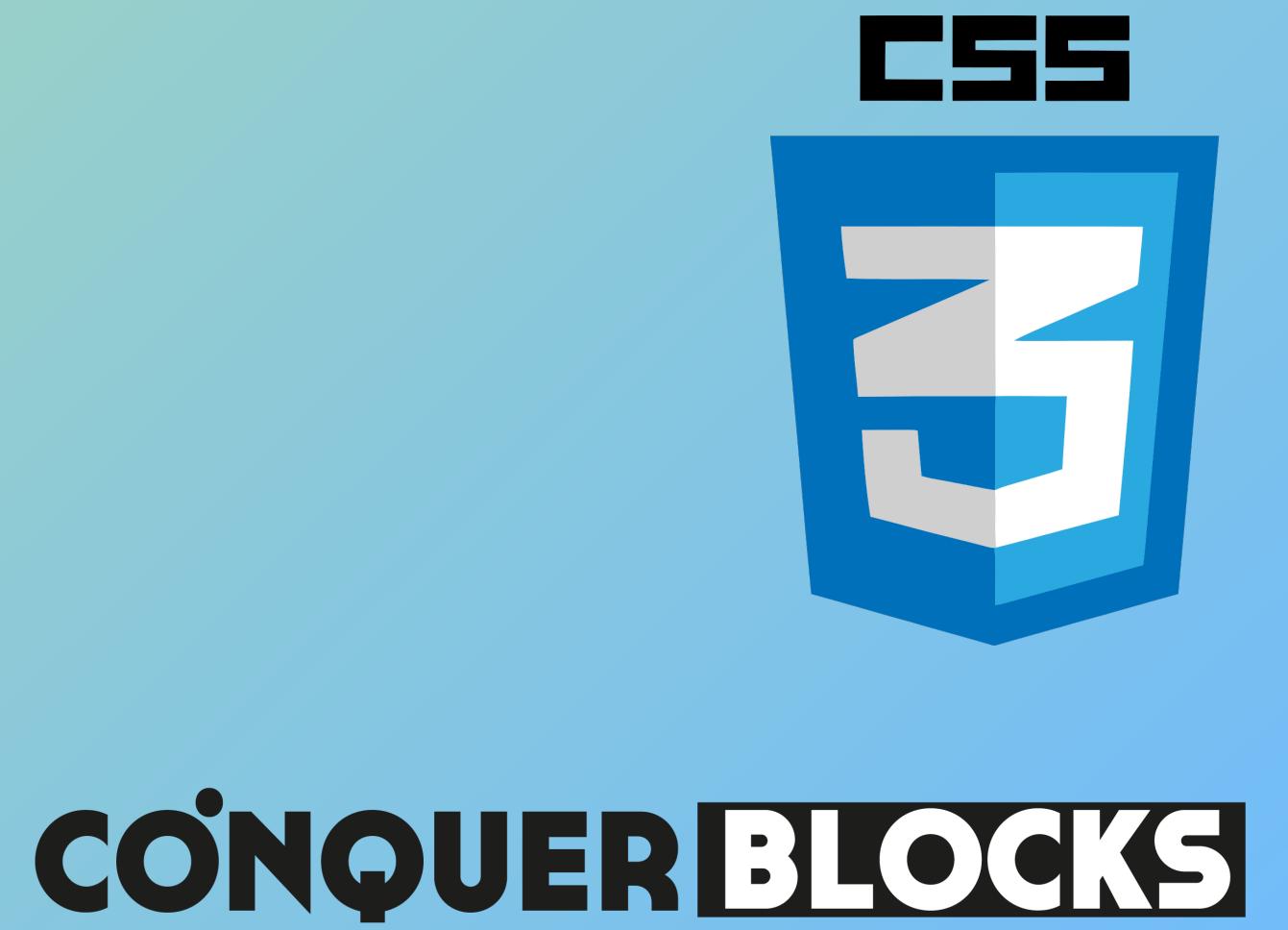


# {css}

Clase 07



# Clase anterior

**Clase anterior**

**Cascada y prioridad**

**Selector descendente: doble, triple, etc...**

---

**Selector de hijo**

---

**Selector de hermano adyacente**

---

**Selector de hermanos**

---

**Selector múltiple**

---

**La importancia de un espacio y un punto**

---

**Especificidad**

---

# Clase 07

<índice>

## Clase 7: Selectores III

Pseudoclases

---

Pseudoclases de hijos

---

Pseudoclases de hijos según tipo de elemento

---

Pseudoclases de interacción

---

Pseudoclases de ubicación

---

Pseudoclases de formularios (Selectores IV)

---

Pseudoelementos (Selectores IV)

---

# Pseudoclasses

# Pseudoclases

Las pseudoclases se utilizan para hacer referencia a elementos HTML que tengan un cierto comportamiento concreto

# Pseudoclases

Volvamos a recordar el esquema general de sintaxis de CSS, donde ahora añadiremos las pseudoclases, que se definen añadiendo **dos puntos** antes del nombre de la pseudoclase concreta, de la siguiente forma

# Pseudoclases

```
selector #id .clase [atributo] :pseudoclase  
propiedad : valor ;  
propiedad : valor  
}
```

# Pseudoclases

De esta forma, podremos seleccionar elementos que en principio parecen iguales, pero tienen diferentes características de comportamiento

# Pseudoclases

Para entenderlo bien, vamos a ver las categorías o tipos de pseudoclases y ver cuales se encuentran entre ellas.

# Pseudoclases de hijos

# Pseudoclases de hijos

Lo usaremos para poder seleccionar cualquier hijo de cualquier elemento sea del tipo que sea

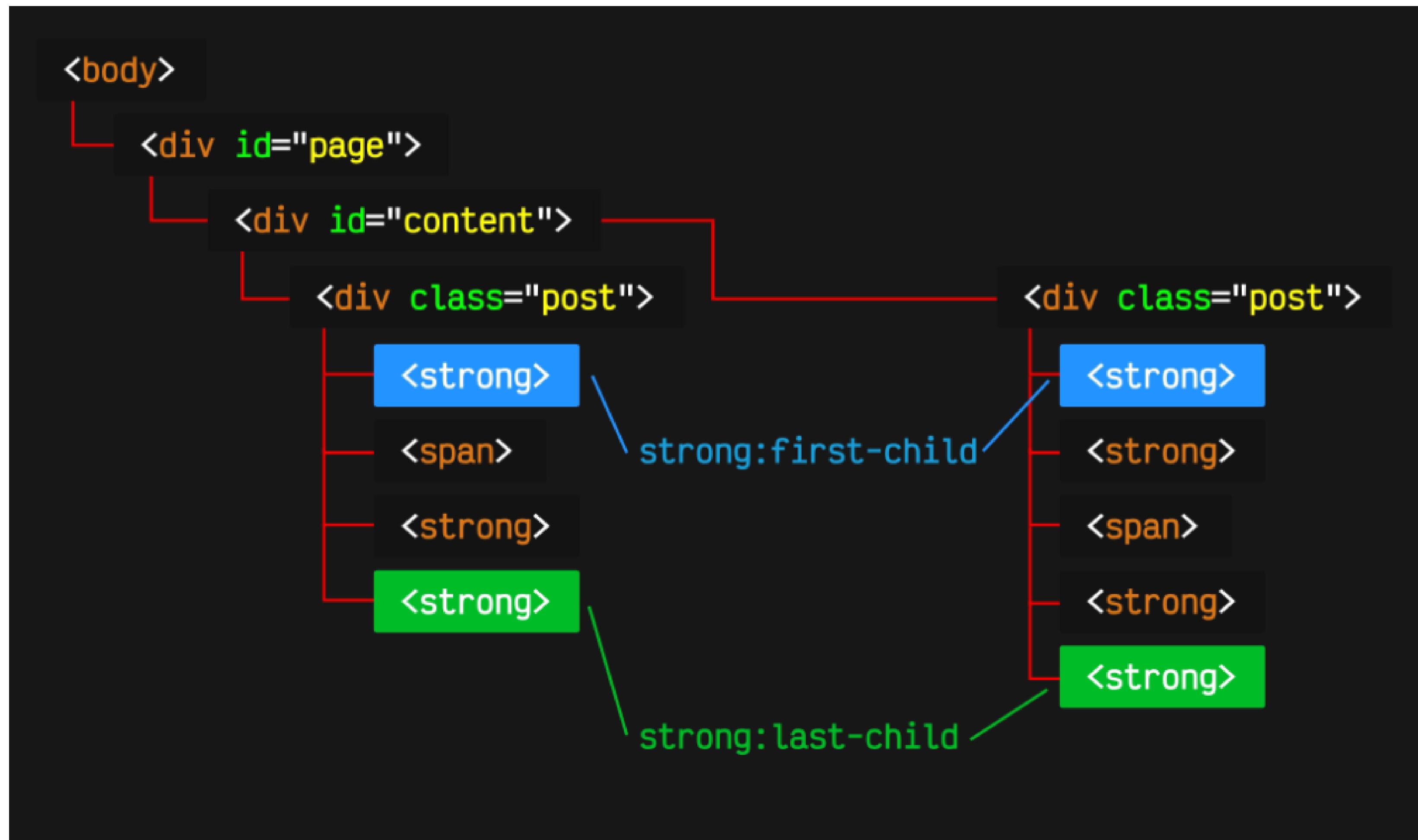
# Pseudoclases de hijos

- :first-child
- :last-child

# Pseudoclases de hijos

```
strong:first-child {  
    background-color: cyan;  
}  
  
strong:last-child {  
    background-color: green;  
}
```

# Pseudoclases de hijos



# Pseudoclases de hijos

- :nth-child(n)
- :nth-last-child(n)
- :nth-child(2n)
- :nth-child(2n+1)
- :nth-child(odd): impares
- :nth-child(even): pares

# Pseudoclases de hijos

- La pseudoclase :nth-child(A) permite especificar el elemento hijo deseado, simplemente estableciendo su número en el parámetro A. No obstante, hay que tener en cuenta que el parámetro A no es sólo un número, sino que es posible escribir ciertas expresiones:

# Pseudoclases de hijos

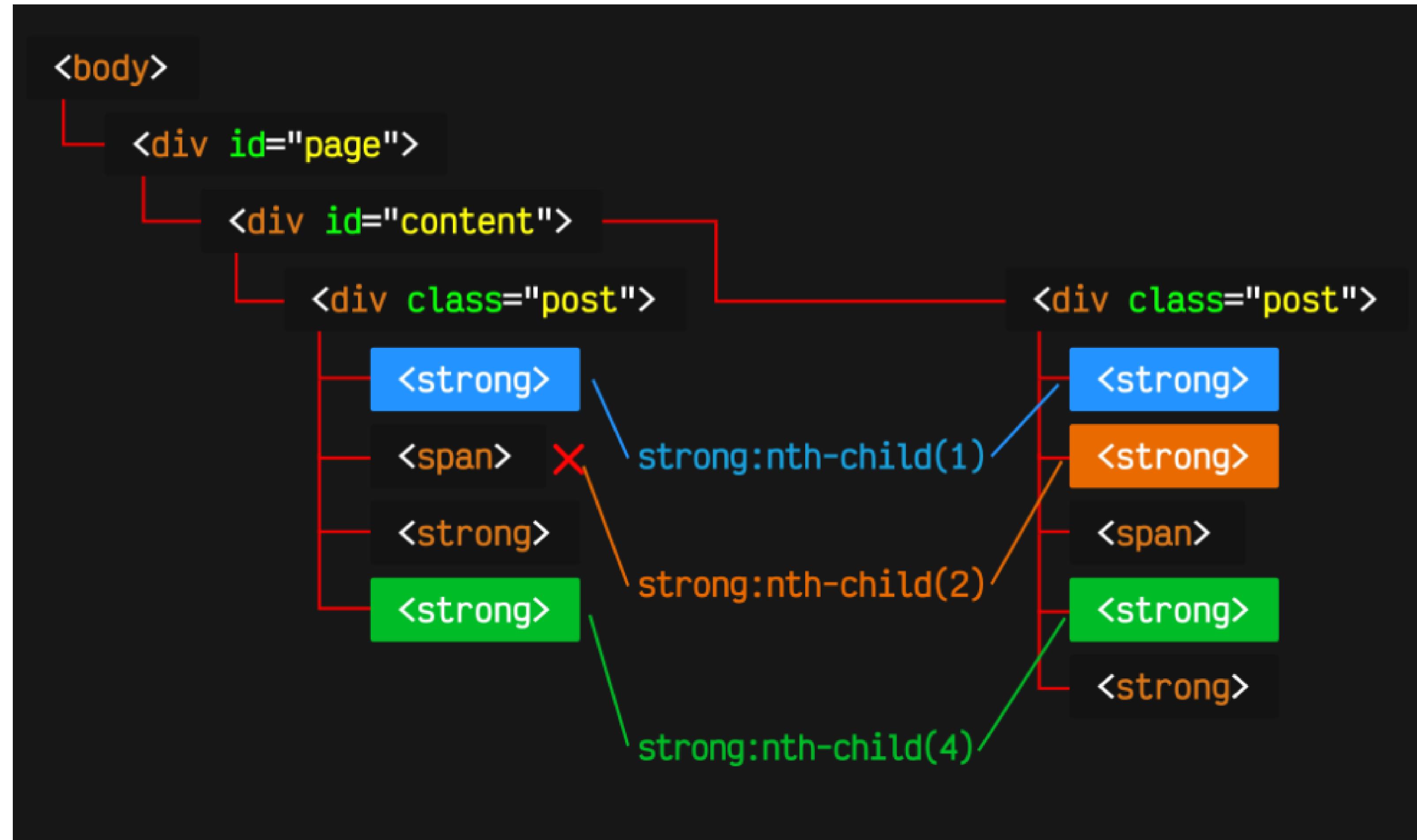
Selector	Equivalente a...	Significado
<code>strong:nth-child(1)</code>	<code>strong:first-child {}</code>	Primer elemento hijo, que además es un <code>&lt;strong&gt;</code>
<code>strong:nth-child(2)</code>		Segundo elemento hijo, que además es un <code>&lt;strong&gt;</code>
<code>strong:nth-child(3)</code>		Tercer elemento hijo, que además es un <code>&lt;strong&gt;</code>
<code>strong:nth-child(n)</code>	<code>strong</code>	Todos los elementos hijos que son <code>&lt;strong&gt;</code>
<code>strong:nth-child(2n)</code>		Todos los elementos hijos pares <code>&lt;strong&gt;</code>
<code>strong:nth-child(2n-1)</code>		Todos los elementos hijos impares <code>&lt;strong&gt;</code>

# Pseudoclases de hijos

La pseudoclase funcional :nth-last-child(A) funciona de forma muy similar a la anterior, permitiendo también indicarle un parámetro A donde especificar una expresión o número para indicar el hijo concreto.

**La diferencia respecto a la anterior, es que comenzamos a contar desde el final**

# Pseudoclases de hijos



# Pseudoclases de hijos

DEMO

# Pseudoclases de hijos según tipo de elemento

# Pseudoclases de hijos

En los casos anteriores, seleccionamos elementos independientemente de que tipo de elemento sea. Simplemente, hacemos caso a la posición donde está ubicado. Y en algún caso, si no coincide la posición con el tipo de elemento especificado en el selector, simplemente no lo selecciona.

# Selector de hijo

Una forma de actuar, quizás, más predecible para nosotros, es que queramos hacer referencia sólo a elementos del mismo tipo, ignorando el resto. Para ello, utilizaremos los selectores siguientes, análogos a los que ya hemos visto, pero haciendo referencia sólo a elementos del mismo tipo

# Selector de hijo

- :first-of-type
- :last-of-type

# Selector de hijo

Por ejemplo, la pseudoclase :first-of-type es la análoga a :first-child, sólo que tendrá en cuenta sólo elementos de su mismo tipo.

Observa el siguiente ejemplo donde no sólo tenemos <div>, sino que también tenemos un

<p>

# Selector de hijo

```
<div class="container">
  <div class="element">Element 1</div>
  <div class="element">Element 2</div>
  <p class="element">Element 3</p>
  <div class="element">Element 4</div>
</div>

<style>
  .container div:first-of-type {
    /* Selecciona "Element 1" */
  }

  .container p:first-of-type {
    /* Selecciona "Element 3" */
  }

  .container :first-of-type {
    /* Selecciona los dos anteriores */
  }
</style>
```

# Selector de hijo

- :nth-of-type(n)
- :nth-last-of-type(n)

# Selector de hijo

La pseudoclase :nth-of-type(A) es la análoga a :nth-child(A). Se trata de una pseudoclase funcional que admite pasar parámetros, donde le podemos indicar un número (o cierta expresión) para ser mucho más específicos a la hora de seleccionar elementos del mismo tipo.

# Selector de hijo

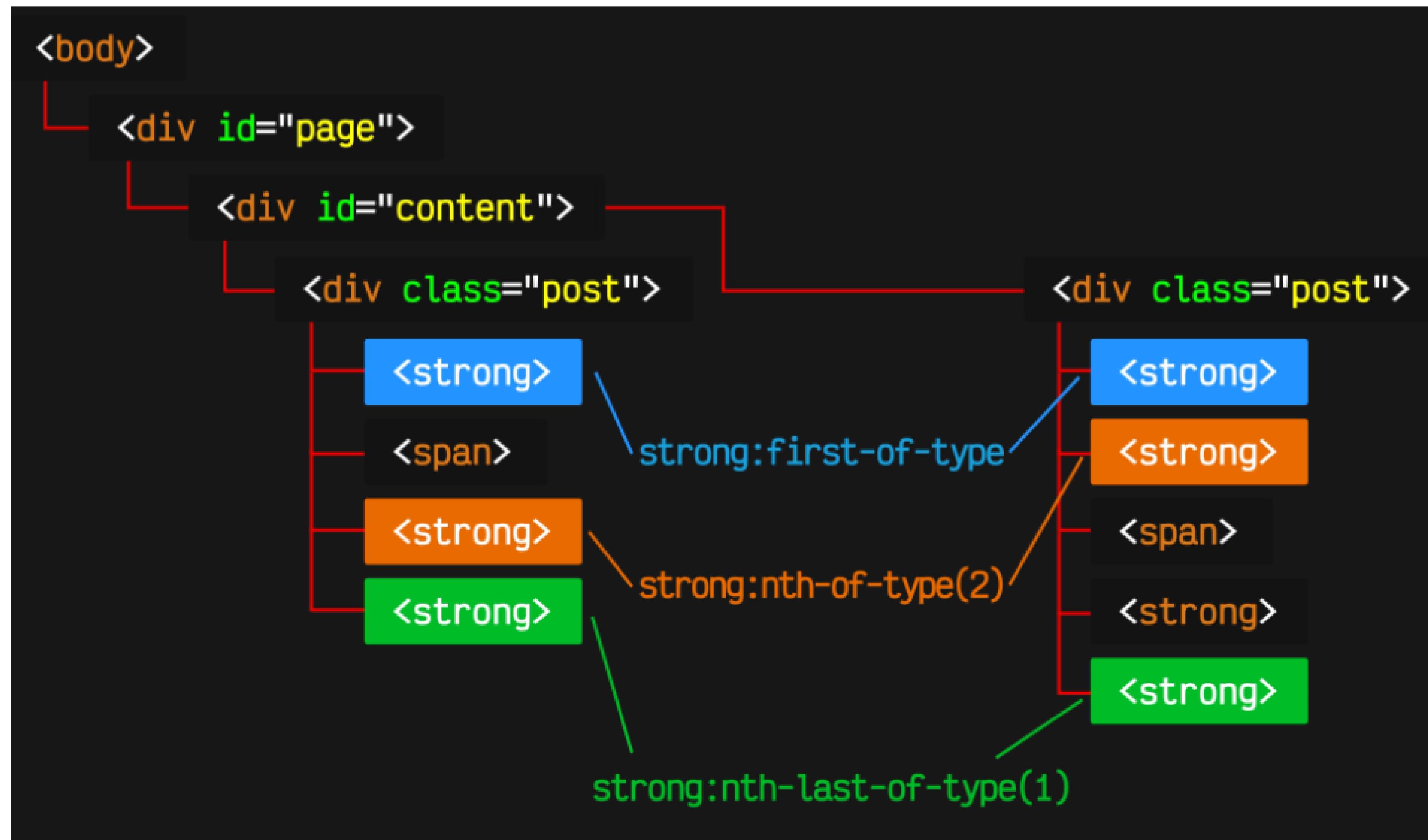
```
<div class="container">
  <div class="element">Element 1</div>
  <div class="element">Element 2</div>
  <p class="element">Element 3</p>
  <div class="element">Element 4</div>
</div>

<style>
  .container :nth-of-type(2) {
    /* Seleccionamos sólo el "Element 2", ya que no hay un segundo <p> */
  }
</style>
```

# Selector de hijo

La pseudoclase :nth-last-of-type(A) es la análoga a :nth-last-child(A). Veamos un nuevo ejemplo sobre el ejercicio anterior, utilizando ahora estas últimas pseudoclases que hemos visto

# Selector de hijo



# Selector de hijo

En este gráfico, se puede ver como `strong:nth-of-type(2)` selecciona el segundo elemento `strong` en ambos casos, a pesar de que en el primero ocupa la tercera posición. En este caso se selecciona porque es el segundo elemento de su mismo tipo (`<strong>`).

Por otro lado, `strong:nth-last-of-type(1)` hace una selección de forma inversa, empezando por el último elemento, por lo que elige el último elemento.

# Selector de hijo

- `:only-child`: hijo único de cualquier tipo
- `:only-of-type`: hijo único de ese tipo concreto
- `:empty`: elemento sin hijos

# Selector de hijo

DEMO

# Pseudoclases de interacción

# Pseudoclases de interacción

Las pseudoclases de interacción se pueden utilizar en cualquier elemento, aunque lo más frecuente es usarlo en elementos interactivos como enlaces, botones o similares, y pueden seleccionar elementos cuando ocurre una cierta interacción por parte del usuario en ellos.

¿Qué tipo de interacción es esa?

# Pseudoclases de interacción

Pseudoclase	Descripción
:hover	Selecciona el elemento si el usuario pasa el ratón sobre dicho elemento.
:active	Selecciona el elemento si el usuario se encuentra pulsando dicho elemento.
:focus	Selecciona el elemento cuando tiene el foco (está en primer plano).
:focus-within	Selecciona el elemento si uno de sus miembros hijos ha ganado el foco.
:focus-visible	Selecciona el elemento cuando tiene el foco sólo de forma visible ( <code>TAB</code> , por ejemplo).

# Pseudoclases de interacción

## hover

La primera de ellas, :hover, es muy útil e interesante, ya que permite aplicar estilos a un elemento justo cuando el usuario pasa el ratón sobre él. Es una de las pseudoclases más utilizadas

# Pseudoclases de interacción

```
/* Usuario mueve el ratón sobre un enlace */
a:hover {
    background-color: cyan;
    padding: 2px
}

/* Usuario mueve el ratón sobre un div y resalta todos los enlaces que contiene */
div:hover a {
    background-color: steelblue;
    color: white;
}
```

# Pseudoclases de interacción

active

Por otro lado, la segunda pseudoclase, :active, permite resaltar los elementos que se encuentran activos, o lo que es lo mismo, elementos que están siendo pulsados en ese instante con el ratón por el usuario

# Pseudoclases de interacción

```
a:active {  
    border: 2px solid #FF0000;  
    padding: 2px  
}
```

# Pseudoclases de interacción

## focus

Cuando estamos posicionados en un elemento, se dice que ese elemento tiene el foco, mientras que al pulsar TAB y saltar a otro, solemos decir que pierde el foco. También es posible ganar o perder el foco pulsando con el ratón en un elemento.

# Pseudoclases de interacción

```
/* El campo ha ganado el foco */
input:focus {
    border: 2px dotted #444
}
```

# Pseudoclases de interacción

## focus-within

La pseudoclase :focus-within permite darle estilo no sólo al elemento que tiene el foco, sino también a los elementos contenedores relacionados con el elemento que gana el foco.

# Pseudoclases de interacción

```
<style>
  form :focus-within {
    background: yellow;
  }
</style>

<form>
  <label>Name: <input type="text"></label>
  <label>Email: <input type="text"></label>
</form>
```

# Pseudoclases de interacción

DEMO

# Pseudoselectores de ubicación

# Pseudoselectores de ubicación

Existen algunas pseudoclases orientadas a los enlaces o hipervínculos. En este caso, permiten cambiar los estilos dependiendo del comportamiento del enlace

# Pseudoselectores de ubicación

Si pensamos otras opciones en el ejemplo anterior, es posible que necesitemos ser menos específicos y en lugar de querer seleccionar los elementos hermanos que sean adyacentes, queramos seleccionar todos los hermanos en general, sin necesidad de que sean adyacentes. Esto se puede conseguir con el selector hermano general, simbolizado con el carácter ~

# Pseudoselectores de ubicación

- :link
- :visited

# Pseudoselectores de ubicación

## link

Permite seleccionar enlaces a páginas que aún no han sido visitadas por el navegador del usuario, lo que puede ser interesante para personalizar el color de este tipo de enlaces. Por defecto, estos enlaces sin visitar suelen ser de color azul.

# Pseudoselectores de ubicación

## visited

También tenemos la pseudoclase :visited, que se utiliza para seleccionar y dar estilo a los enlaces que hayan sido visitados previamente en el navegador del usuario. Por defecto, estos enlaces suelen ser de color violeta.

# Pseudoselectores de ubicación

DEMO

# Ejercicios

# Ejercicios Selectores II

1. El primer elemento `<a>` hijo de `<p>`
2. Selecciona los checkbox que estén marcados, hijos de un `<form>`
3. Selecciona el último `<p>` que sea descendiente de `<article>`
4. Seleccionar el segundo hijo de un elemento section
5. Seleccionar los hijos pares de un elemento ol
6. Seleccionar el antepenúltimo li de un ul de longitud indefinida
7. El segundo `<a>` de dentro de un `<p>`
8. Los button que sean hijos únicos
9. Los span que sean primeros hijos de un p
10. Penúltimo hijo de body

# <Despedida>

Email

**bienvenidosaez@gmail.com**

Instagram

**@bienvenidosaez**

Youtube

**youtube.com/bienvenidosaez**

**CONQUERBLOCKS**