|  |  |
| --- | --- |
| **Cách nhúng CSS vào HTML** | \* Có 3 cách nhúng  - **inline** : sử dụng thuộc tính style của các thẻ HTML để định nghĩa các quy tắc CSS trực tiếp trong thẻ đó. Các quy tắc CSS được đặt trong thuộc tính style sẽ được áp dụng chỉ cho thẻ HTML đó.    - **internal** : bằng cách sử dụng thẻ <style> trong phần <head> của tài liệu HTML. Với cách này, các quy tắc CSS được định nghĩa trong tài liệu HTML của bạn và chỉ áp dụng cho tài liệu đó.    - **External** : sử dụng thẻ index( <link> ) để nhúng 1 file CSS từ bên ngoài vào |
| **Mức độ ưu tiên trong các cách nhúng** | Inline > internal hoặc external. Với 2 cách internal và external thì cách nào được sử dụng gần thẻ hơn thì sẽ ưu tiên cách đó |
| **Selector ()** | Selector trong CSS được sử dụng để xác định phần tử HTML nào sẽ áp dụng các quy tắc CSS được định nghĩa trong tài liệu CSS. Các loại selector trong CSS bao gồm:  1/ Selector phần tử (Element Selector): Chọn tất cả các phần tử HTML có tên thẻ giống với tên selector.    2/ Selector ID: Chọn phần tử HTML có thuộc tính ID giống với tên selector.    3/ Selector class: Chọn tất cả các phần tử HTML có thuộc tính class giống với tên selector.    4/ Selector thuộc tính (Attribute Selector): Chọn tất cả các phần tử HTML có thuộc tính giống với tên selector.    5/ Selector kết hợp (Combining Selectors): Kết hợp nhiều loại selector để chọn ra các phần tử HTML cụ thể.    6/ Selector Pseudo-classes: Chọn phần tử HTML dựa trên trạng thái của nó.    7/ Selector Pseudo-elements: Chọn các phần tử HTML dựa trên vị trí của nó trong tài liệu HTML. |
| **Độ ưu tiên của selector** | #id > . class > tag  Ngoài ra, có một số trường hợp đặc biệt khi các selector có cùng độ ưu tiên. Ví dụ như:  - Nếu có nhiều hơn một selector trực tiếp được áp dụng cho cùng một phần tử, selector cuối cùng sẽ được áp dụng.  - Nếu có nhiều hơn một selector ID được áp dụng cho cùng một phần tử, selector có ID cuối cùng sẽ được áp dụng.  - Nếu các selector còn lại có cùng độ ưu tiên và được áp dụng cho cùng một phần tử, thì selector được khai báo sau cùng sẽ được áp dụng. |
| **Box Model** | CSS là một cách để xác định cách mà các phần tử HTML được định vị và định dạng trong tài liệu. Box Model là một khái niệm quan trọng trong CSS, nó mô tả cách mà mỗi phần tử HTML được xây dựng như một hộp, với nội dung bên trong và các thuộc tính về độ rộng, chiều cao, đường viền và khoảng cách được xác định.  - Box Model bao gồm 4 phần chính:  - **Content** (Nội dung): Phần bên trong của hộp, chứa nội dung thực sự của phần tử, ví dụ như văn bản, hình ảnh, hoặc các phần tử con.  - **Padding** (Lề): Khoảng cách giữa nội dung và đường viền, nó có thể được đặt bằng một giá trị duy nhất hoặc bằng các giá trị riêng biệt cho mỗi cạnh (top, right, bottom, left).  - **Border** (Đường viền): Một đường viền bao quanh phần tử, được định dạng bằng cách sử dụng các thuộc tính như độ rộng, màu sắc và kiểu đường.  - **Margin** (Khoảng cách): Khoảng cách giữa đường viền và các phần tử khác, nó cũng có thể được đặt bằng một giá trị duy nhất hoặc bằng các giá trị riêng biệt cho mỗi cạnh (top, right, bottom, left).  -> margin sẽ lấy khoảng cách từ border của phần tử này đến border của phần tử khác mà không quan tâm đến margin của phần tử khác |
| **Position** | Trong CSS, thuộc tính position được sử dụng để xác định vị trí của một phần tử HTML trong tài liệu. Có 5 giá trị khác nhau cho thuộc tính position, bao gồm:  - **static**:Khi sử dụng giá trị này, phần tử sẽ được định vị theo vị trí bình thường, không bị ảnh hưởng bởi các thuộc tính top, bottom, left, right và z-index. Phần tử sử dụng giá trị static sẽ nằm trong luồng bình thường của trang, tức là các phần tử khác có thể chồng lên phần tử này  .- **relative**: Định vị phần tử dựa trên vị trí của nó so với vị trí ban đầu của nó. Các thuộc tính top, right, bottom, left được sử dụng để định vị phần tử tương đối với vị trí ban đầu của nó.  - **absolute**: Định vị phần tử dựa trên vị trí của phần tử cha gần nhất có thuộc tính position khác static. Các thuộc tính top, right, bottom, left được sử dụng để định vị phần tử tuyệt đối trong phạm vi phần tử cha.  - **fixed**: Tương tự như giá trị absolute, nhưng phần tử được định vị tuyệt đối đối với vùng xem, và không bị ảnh hưởng bởi cuộn trang.  - **sticky**: Phần tử được định vị dựa trên vị trí của nó trong phạm vi phần tử cha, nhưng vẫn có thể di chuyển khi phần tử cha được cuộn. Khi phần tử tiếp cận đến giới hạn của phần tử cha, nó sẽ bắt đầu dính vào vị trí và không di chuyển nữa. |
| **Display** | Display trong CSS dùng để xác định loại phần tử HTML sẽ được hiển thị như thế nào trên trang web. Display định nghĩa các giá trị sau:  - **block**: Hiển thị phần tử HTML là một khối block rộng đầy trang. Khối block thường sẽ đặt trên một dòng mới và chiếm hết chiều rộng của cha bao nó. Ví dụ: div, p, h1, h2, h3, h4, h5, h6, ul, ol, li.  - inline: Hiển thị phần tử HTML là một phần của câu lệnh, trên cùng một dòng với các phần tử khác. Nó sẽ không tạo ra một khối đầy đủ như khối block. Ví dụ: a, span, img, strong, em.  - inline-block: Hiển thị phần tử HTML như một khối block, nhưng vẫn giữ lại khả năng của phần tử inline, tức là hiển thị trên cùng một dòng với các phần tử khác như phần tử inline. Ví dụ: button, input, textarea.  - none: Ẩn phần tử HTML khỏi trang web. Nó không chiếm bất kỳ không gian nào trên trang. Ví dụ: display: none. |
|  |  |