*HTML-CSS*

1. **Display** : là một trong những thuộc tính CSS quan trọng nhất, xác định cách 1 phần tử HTML được hiện thị trên trang web. Một số thể loại phổ biến:

* Display : flex; được sử dụng để tạo ra một container (p.tử cha) Flexbox. Flexbox là một mô hình bố cục mạnh mẽ và linh hoạt, cho phép chúng ta dễ dàng sắp xếp, căn chỉnh và phân phối không gian trong container.
* Flex-direction: Xác định hướng mà các mục flexbox được sắp xếp (row, row-reverse, column, column-reverse)
* Justify-content: Căn chỉnh các mục flexbox dọc theo trục chính (flex-start, flex-end, center: được sắp xếp ở giữa trục chính, space-between, space-around, space-evenly)
* Align-items: Căn chỉnh dọc theo trục chéo, trục vuông góc với trục chính (flex-start, flex-end, center: canh giữa các mục dọc theo trục chéo, stretch (mặc định), baseline)
* Flex-wrap: Xác định xem các mục flexbox có nên được bọc (được xuống dòng) sang nhiều dòng hay không (nowrap, wrap, wrap-reverse)
* Flex-grow, flex-shrink, flex-basis: Kiểm soát kích thước và tính linh hoạt cảu các mục flexbox
* Display: grid/ inline-grid ; được sử dụng để tạo ra một hệ thống lưới (grid) 2 chiều, hữu ích khi bạn muốn tạo ra các bố cục phức tạp và có cấu trúc rõ ràng.
* Grid-template-columns: dùng để định nghĩa các cột trong một Grid Container, cho phép xác định kích thước của ô theo cột và số lượng các cột(ô)
* Grid-template-row: dùng để định nghĩa các hàng trong một Grid Container, cho phép xác định kích thước của ô theo hàng và số lượng các hàng(ô)
* Column-gap: khoảng cách giữa các cột
* Row-gap: khoảng cách giữa các hàng
* Grid-column/row: định nghĩa một vị trí và kích thước của 1 ô trong 1 grid container theo chiều ngang (cột)/ dọc(hàng)
* Display: inline-block; là kết hợp giữa *display block* và *display inline,* nhưng sẽ không chiếm hết chiều ngang của vùng và vẫn sử dụng được các thuộc tính như: width, height, margin, padding, border
* Display: table, table-row, table-cell, teble-column, table-column-group, table-header-group, table-footer-group, table-caption; Các phần tử được này được sử dụng để tạo bảng

1. **Box-shadow, text-shadow** : hiệu ứng đổ bóng cho cho phần tử, với cấu trúc

* box-shadow: offset-x offset-y blur-radius spread-radius color;
* offset-x: Giá trị dương đẩy bóng sang phải, giá trị âm đẩy bóng sang trái.
* offset-y: Giá trị dương đẩy bóng xuống dưới, giá trị âm đẩy bóng lên trên.
* blur-radius: Khoảng cách mờ của bóng, giá trị càng lớn thì bóng càng mờ.
* spread-radius: Kích thước của vùng bóng, giá trị dương sẽ làm bóng lớn hơn, giá trị âm sẽ làm bóng nhỏ hơn.
* color: Màu sắc của bóng.

1. **Position** : được sử dụng để định vị vị trí hiện thị của các phần tử thẻ HTML. Nó xác định cách phần tử được đặt trong luồng văn bản và có thể được sử dụng để tạo các hiệu ứng như popup, menu, hoặc các phần tử được định vị cố định. Có 5 giá trị chính của position:

* Position: static ; là giá trị mặc định. Phần tử được đặt ở vị trí tự nhiên của nó, không thể chỉnh sửa
* Position: relative ; được định vị tương đối so với vị trí bình thường của nó. Có thể sử dụng top, right, bottom, left để di chuyển phần tử, vẫn giữ vị trí trong luồng văn bản, có thể được chồng chéo bằng thuộc tính z-index
* Top: giúp căn chỉnh phần tử từ trên xuống dưới nếu vị trí > 0 và ngược lại
* Bottom: cho phép căn chỉnh phần tử từ dưới lên trên nếu giá trị > 0 và ngược lại
* Right: giúp căn chỉnh phần tử qua phải nếu > 0 và ngược lại
* Left: cho phép căn chỉnh phần tử qua phải nếu giá trị > 0 và ngược lại
* Position: absolute ; được định vị tuyệt đối so với phần tử cha gần nhất có position khác static. Nếu không có phần tử cha, nó sẽ được định vị tương đối so với body, được tách khỏi luồng văn bản
* Position: fixed ; được định vị tuyệt đối so với viewport (cửa sổ trình duyệt). Vị trí của phần tử sẽ không thay đổi khi người dùng cuộn trang, được tách khỏi luồng văn bản
* Position: sticky ; kết hợp tính năng của relative và fixed. Ban đầu được định vị như relative, sau đó khi người dùng cuộn đến một điểm nhất định, nó sẽ chuyển sang fixed

1. **Overflow**: dùng để xử lý nội dung của một phần tử khi nó vượt quá kích thước của phần tử đó, một số giá trị chính:

* Overflow: visible ; nội dung vượt quá kích thước của phần tử sẽ hiện thị bên ngoài phần tử
* Overflow: hidden ; nội dung vượt quá kích thước sẽ bị ẩn
* Overflow: scroll ; sẽ xuất hiện thanh cuộn để người dùng có thể xem toàn bộ nội dung ngay khi nó vượt qua kích thước phần tử
* Overflow: auto ; trình duyệt sẽ tự động thêm thanh cuộn nếu nội dung vượt quá kích thước của phần tử
* overflow-x/y : xử lý nội dung vượt quá chiều ngang/ dọc của phần tử

1. **Object-fit** : dùng để xác định cách một phần tử <img> hoặc <video> sẽ được hiện thị trong khu vực được chỉ định bởi chiều rộng và chiều cao của nó, một số giá trị chính:

* Object-fit: contain ; giữ nguyên tỷ lệ khung hình của đối tượng và chứa toàn bộ đối tượng trong khu vực chỉ định
* Object-fit: cover ; giữ nguyên tỷ lệ khung hình của đối tượng và cắt bớt các phần tử để vừa khít với khu vực chỉ định
* Object-fit: fill ; kéo dãn đối tượng để vừa khít với khu vực chỉ định
* Object-fit: none ; không thay đổi kích thước của đối tượng, hiện thị toàn bộ trong khu vực được chỉ định
* Object-fit: scale-down ; giảm kích thước phần tử nếu cần thiết, nhưng không bao giờ tăng kích thước của nó

1. **Transition** : là một tính năng cho phép bạn định nghĩa sự chuyển tiếp giữa các trạng thái CSS khác nhau của một phần tử

* Transition: transform 0.3s ease-in-out ; hiệu ứng chuyển tiếp trong 0.3s, bắt đầu chậm, tăng tốc dần và cuối cùng chậm lại
* Transform: scale(1.2) ; khi di chuyển chuột vào phần tử, nó sẽ phóng to lên 120% trong 0.3s