Contents

[Node JS: 1](#_Toc160120618)

[Developer Tools: 1](#_Toc160120619)

[Markup Language – Ngôn Ngữ Đánh Dấu: 4](#_Toc160120620)

[Mongo DB: 6](#_Toc160120621)

[SQL – Structured Query Language: 8](#_Toc160120622)

[JS: 16](#_Toc160120623)

[Development – Phát Triển: 17](#_Toc160120624)

[Storage – Lưu Trữ Dữ liệu: 20](#_Toc160120625)

Node JS:

1. Node JS Dùng Để Làm Gì?

* Chạy File JS không cần sử dụng trình duyệt Web

1. Cách Cài Đặt Node JS Phiên Bản Mới Nhất Cho Windows 7 32 Bit?

* https://nodejs.org/download/release/v17.9.1/
* Click vào “node-v17.9.1-win-x86.zip” + chờ nó tải + giải nén tại chỗ + di chuyển cái File vừa bung ra vào thư mục “C:\Program Files” và đổi tên thành “nodejs” + xóa cái File nén vừa tải + thêm đường dẫn “C:\Program Files\nodejs” vào biến hệ thống
* Cách này là tải Node JS bản tương thích lâu dài cho nhiều hệ điều hành

1. Cách Gỡ Cài Đặt Hoàn Toàn Node JS Ra Khỏi Máy?

* Xóa thư mục “nodejs” trong thư mục “C:\Program Files” và xóa đường dẫn “C:\Program Files\nodejs” khỏi biến hệ thống

1. Show Mô Tả Của Lệnh Node?

node –h

1. Show Phiên Bản Của Node?

node –v

1. Chạy 1 File JS?

node <Tên File JS

Developer Tools:

1. Cách Hoạt Động?

* Để mở cửa sổ Developer Tools, gọi tắt là cửa sổ phát triển, phải chuột vào chỗ bất kì trong trang Web đang hiện ở 1 Tab + chọn “Inspect”, khi này 1 cửa sổ phát triển liên kết với Tab vừa chọn sẽ hiện lên
* Khi đóng Tab, thì cửa sổ phát triển cũng đóng theo
* Tab cập nhật gì, thì cửa sổ phát triển cũng cập nhật theo, và ngược lại, cửa sổ phát triển cập nhật gì, thì Tab cũng cập nhật theo
* Để chủ động đóng cửa sổ phát triển, Click dấu X góc phải trên của nó
* Mặc định cửa sổ phát triển sẽ nằm bên phải và tích hợp vào Tab liên kết, để thay đổi vị trí của nó, Click dấu 3 chấm dọc góc phải trên + tại mục “Dock side” chọn biểu tượng “Dock to right” để đặt bên phải, “Dock to left” để đặt bên trái, “Dock to bottom” để đặt bên dưới và “Undock to separate window” để tách nó thành 1 cửa sổ riêng không tích hợp nữa, vẫn liên kết với Tab
* Cửa sổ phát triển bao gồm 3 phần chính, thanh trên cùng chứa các Tab, phần ngay bên dưới sẽ khác nhau với mỗi Tab, và phần dưới cùng có thể có hoặc không, là Tab “Console” mở sẵn, nó sẽ ẩn đi khi bạn vào Tab “Console” chính, và hiện ra ở những Tab khác, để ẩn hoặc hiện nó, nhấn “Esc”

1. Tab “Elements”?

* Cửa sổ phát triển sẽ có 2 chế độ, 1 là chế độ chọn phần tử, 2 là chế độ không chọn phần tử, mặc định ở chế độ không chọn phần tử
* Để thay đổi chế độ, Click vào biểu tượng mũi tên góc trái trên = nhấn “Ctrl” + “Shift” + “C”
* Tab “Elements” bao gồm 2 phần, phần trên chứa cấu trúc DOM thời điểm hiện tại của Tab liên kết với cửa số phát triển, trình bày theo mã HTML, gọi tắt là cửa sổ DOM, phần bên dưới chứa đầy đủ thông tin của Node được chọn, gọi tắt là cửa sổ Node được chọn
* Khi bạn đang trong chế độ chọn phần tử, và tiến hành Click chuột tại chỗ nào đó trong trang Web, thì cửa sổ phát triển sẽ nhảy ngay sang Tab “Elements” và phần tử bạn Click vào tương ứng trong cửa sổ DOM sẽ được chọn, hoặc bạn chọn trực tiếp trong cửa sổ DOM cũng được
* Cửa sổ Node được chọn bao gồm 3 phần, thanh trên cùng chứa đường đi từ Node gốc, là Node HTML, tới Node được chọn, nghĩa là danh sách các Node từ Node HTML tới Node được chọn, bạn có thể Click vào để chọn 1 trong số chúng, thanh bên dưới chứa các Tab, gọi là thanh thuộc tính Node được chọn, toàn bộ phần bên dưới sẽ khác nhau với các Tab khác nhau
* Các Tab tại thanh thuộc tính Node được chọn bao gồm
* Tab “Styles”, gồm 5 phần, thanh trên cùng gọi là thanh công cụ kiểu cách, phần ngay bên dưới là toàn bộ kiểu cách mà bạn đã chỉ định trực tiếp cho Node được chọn thông qua chỉ định thuộc tính style, bạn có thể chỉnh sửa phần này, trước mỗi dòng ở đây có thêm ô để bạn Tick hoặc Untick, nếu Untick thì dòng tương ứng trở nên vô dụng, phần ở giữa tương tự, nhưng là kiểu cách bạn đặt trong Tag Style hoặc trong File CSS bên ngoài, bạn có thể chỉnh sửa phần này, phần dưới in nghiêng có đề “user agent stylesheet” chứa toàn bộ kiểu cách mặc định của Node được chọn, bạn không thể chỉnh sửa phần này, ví dụ kiểu cách mặc định của Tag đoạn văn sẽ khác Tag đề mục, và giả sử xóa hết kiểu cách mặc định này đi, thì nó chả khác gì Node văn bản mặc định, phần cuối cùng là bản vẽ Node này khi kết xuất, bao gồm Padding, Border, Margin

1. Tab “Console”?

* Gồm 2 phần, thanh trên cùng gọi là thanh công cụ Console, phần bên dưới là nơi bạn có thể chạy lệnh JS trực tiếp, nó giống như Terminal, những gì được in ra sẽ nằm ở đây, gọi là Console
* Bạn chỉ có thể nhập lệnh lên Console và nhấn Enter để chạy chỉ khi luồng JS của trang Web đang không chạy
* Tất cả những gì trong Tab “Console” sẽ bị xóa khi Tab liên kết với cửa sổ phát triển chuyển hướng sang trang Web khác
* Để chủ động xóa tất cả nội dung trong Tab “Console”, Click biểu tượng “Clear console” bên trái thanh công cụ Console = nhấn “Ctrl” + “L”
* Ngoài ra, để ẩn hiện cửa sổ cài đặt Console, Click biểu tượng bánh răng bên phải cùng thanh công cụ Console
* Để ngăn chặn việc xóa nội dung trong Tab “Console” khi chuyển hướng trang, Tick vào ô “Preserve log” tại cửa sổ cài đặt Console, và Untick để trở lại mặc định
* Nếu khi in ra Console dòng tiếp theo, mà nội dung giống dòng ngay trước, thì không in ra dòng mới, đồng thời cũng sẽ hiển thị số lần lặp lại

1. Tab “Sources”?

* Bao gồm 3 phần, phần góc trái trên là cửa sổ Navigator, phần góc phải trên là cửa sổ Edit File, phần bên dưới là cửa sổ Debug
* Cửa sổ Navigator bao gồm 2 phần, thanh trên cùng chứa các Tab là thanh Navigator, phần bên dưới sẽ khác nhau với mỗi Tab
* Cửa sổ Edit File chứa các Tab giống VS Code là các File bạn mở, bạn có thể chỉnh sửa và lưu nó, không thể chỉnh sửa nếu là File HTML
* Tab “Page” tại thanh Navigator sẽ chứa cấu trúc thư mục của trang Web, giống trong VS Code, thư mục to nhất đại diện cho trang Web tên là “top”, các thư mục con ngay bên dưới là các tên miền, bạn có thể Click vào các File bên trong để mở nó trong cửa sổ Edit File, ở đây bao gồm tất cả các File mà trang Web đang sử dụng, bao gồm cả các File lưu ở các trang Web khác

1. Tab “Network”?

* Có 4 phần, thanh trên cùng, gọi tắt là thanh công cụ mạng, phần ngay bên dưới, gọi là cửa sổ đồ thị mạng, phần rộng hơn bên dưới, là cửa sổ File tải lên, và thanh dưới cùng, gọi là thanh thống kê mạng
* Tab “Network” ban đầu khi bạn mở cửa sổ phát triển sẽ trống, và kể từ lúc này nó bắt đầu hóng dữ liệu mà máy chủ gửi về trang Web ở máy bạn, cũng như những dữ liệu bạn gửi cho máy chủ, …, để tạm ngưng việc hóng, Click biểu tượng màu đỏ bên trái cùng ở thanh công cụ mạng = nhấn “Ctrl” + “E”, khi này nó sẽ thành biểu tượng vòng tròn xám, nhấn lại nó nữa để bắt đầu hóng lại, nhưng đồng thời cũng xóa tất cả những gì đã hóng
* Tất cả những gì trong Tab “Network” cũng sẽ bị xóa khi Tab liên kết với cửa sổ phát triển chuyển hướng sang trang Web khác, để ngăn chặn việc xóa nội dung trong Tab “Network” khi chuyển hướng trang hoặc khi hóng lại, Tick vào ô “Preserve log” tại thanh công cụ mạng, và Untick để trở lại mặc định
* Để chủ động xóa tất cả nội dung trong Tab “Network”, Click biểu tượng “Clear network log” bên trái thanh công cụ mạng = nhấn “Ctrl” + “L”
* Để ý ngay dưới thanh công cụ mạng, sẽ có 1 thanh khác, gọi là thanh lọc, thanh này mặc định sẽ hiện ra, để ẩn hiện nó, Click biểu tượng “Filter” ở bên trái thanh công cụ mạng

1. Mô Phỏng Điện Thoại?

* Tại cửa sổ phát triển, Click vào biểu tượng “Toggle device toolbar” = nhấn “Ctrl” + “Shift” + “M” để thay đổi giữa 2 chế độ là chế độ PC và chế độ điện thoại, mặc định ở chế độ PC
* Ở chế độ điện thoại, trang Web của bạn mô phỏng như ở điện thoại, gồm 3 phần chính, thanh trên cùng là thanh công cụ điện thoại, thanh bên dưới là thanh kích thước điện thoại, phần dưới cùng là trang Web của bạn trên điện thoại, gọi là cửa sổ điện thoại
* Khi đóng cửa sổ phát triển thì trang Web nếu đang ở chế độ điện thoại, sẽ trở lại chế độ PC
* Có 2 loại điện thoại, 1 là loại thích ứng, 2 là loại điện thoại thông thường, để chọn loại điện thoại, Click vào “Dimensions:” bên trái thanh công cụ điện thoại + chọn “Responsive” để chọn loại thích ứng hoặc chọn các tên điện thoại bên dưới để chọn loại thông thường
* Ở giữa thanh công cụ điện thoại sẽ hiện kích thước điện thoại theo cú pháp chiều rộng x chiều cao, không thể thay đổi các kích thước này nếu là loại điện thoại thông thường, và có thể thay đổi với loại thích ứng
* Bạn cũng có thể Click vào thanh kích thước điện thoại để chuyển ngay sang loại thích ứng với chiều rộng là kích thước bạn chọn, ngoài ra, loại thích ứng còn cho phép bạn thay đổi kích thước điện thoại trực tiếp bằng cách kéo thả biên điện thoại

1. Mã Nguồn Trang Web?

* Để xem đoạn mã HTML gốc mà máy chủ gửi cho bạn khi bạn gửi yêu cầu tới trang Web, phải chuột vào chỗ bất kì trong trang + chọn “View page source” = nhấn “Ctrl” + “U”, khi này 1 Tab mới sẽ mở ngay bên cạnh Tab của trang Web, Tab này hiện mã HTML bạn muốn

Markup Language – Ngôn Ngữ Đánh Dấu:

1. DOM (Document Object Model)?

* Mỗi File được viết bởi ngôn ngữ đánh dấu được coi như 1 Rooted Tree, mỗi Node sẽ có giá trị và thuộc tính riêng
* Để xem cấu trúc DOM của 1 File HTML, vào Link
* <https://software.hixie.ch/utilities/js/live-dom-viewer/>
* Dán đoạn HTML vào mục “Markup to test”, khi này tại mục “DOM view” sẽ hiện cấu trúc DOM
* Node gốc sẽ là Node HTML, nó có 2 Node con là Node Head và Node Body, trong Node Body là các Node chứa nội dung trang Web
* Các Leaf Node có thể là Node văn bản, Node hình ảnh, …

1. Phần Mở Rộng File HTML?

* “.html”

1. Cách Hoạt Động Của Ngôn Ngữ Đánh Dấu Trong Trình Duyệt?

* Khi trình duyệt gửi yêu cầu tới máy chủ nội dung File đánh dấu, máy chủ trước tiên sẽ gửi về nội dung của File này đã được mã hóa thành Byte Code, nghĩa là bây giờ máy ta sẽ nhận được 1 chuỗi nhị phân, sau đó nó sẽ gửi Header ứng với File này, dựa vào Header, ta biết chuỗi này được mã hóa bởi cơ chế mã hóa nào đó, ví dụ UTF 8, khi này trình duyệt của ta sẽ sử dụng cơ chế giải mã hóa tương ứng, trong trường hợp này là UTF 8, để biến chuỗi nhị phân nhận được thành các kí tự văn bản như trong File đánh dấu ban đầu, đồng thời trong Header cũng có nói là nó thuộc kiểu ngôn ngữ đánh dấu nào, ví dụ HTML, hay SVG, …, thuộc phiên bản bao nhiêu, … và dựa vào đó để hiểu nội dung File này nó muốn làm gì, từ đó kết xuất nội dung trang Web ra trình duyệt, lưu ý là ta có thể gửi yêu cầu tới chính máy của ta, chỉ cần dán đường dẫn tới File đánh dấu trong máy ta lên trình duyệt

1. Khai Báo Phiên Bản Ngôn Ngữ Đánh Dấu Của File Đánh Dấu?

* Phần mở rộng File đã cho biết ngôn ngữ đánh dấu là gì
* Đặt dòng sau vào đầu File, nếu không đặt thì tự động hiểu là phiên bản cũ

<!DOCTYPE <Phiên Bản>>

* Ví dụ

<!DOCTYPE html>

* Dòng trên chỉ định ta dùng phiên bản HTML 5, là cái mới nhất

1. Cấu Trúc 1 File HTML?

* Dòng đầu là dòng khai báo ngôn ngữ đánh dấu
* Tiếp theo phải tạo Node HTML, Node Head và Node Body, nếu không tạo thì khi chạy trên trình duyệt sẽ tự động thêm

<html>

<head>

</head>

<body>

</body>

</html>

* Các dấu cách hoặc dấu xuống dòng liên tiếp, đều quy về = 1 dấu cách, nếu chúng nằm giữa 2 kí tự khác cách và xuống dòng
* Ví dụ

<body> a b </body>

* Phần Header do máy chủ gửi sẽ 1 phần dựa vào nội dung trong Node Head
* Trong Node Head, ta có thể tạo các Node sau
* Để đặt tiêu đề cho Tab, thêm Node tiêu đề

<title><Tiêu Đề></title>

* Khi này, Node Head sẽ có thêm Node con là Node tiêu đề, Node tiêu đề có Node con là Node văn bản với nội dung là <Tiêu Đề>
* Trong Node Head, ta có thể tạo ra các Node con là các Node Meta chỉ định thông tin Header
* Để chỉ định kiểu mã hóa ra Byte Code

<meta charset = <Kiểu Mã Hóa>>

* Ví dụ

<meta charset = "UTF-8">

* Để chỉ định tên người tạo ra Web

<meta name = "author" content = <Tên>>

* Ví dụ

<meta name = "author" content = "Huan">

* Để chỉ định từ khóa để Google tìm kiếm

<meta

name = "keywords"

content = "khabanh, daubuoi, cac"

>

* Để chỉ định mô tả ngắn cho trang Web

<meta name = "description" content = <Mô Tả>>

* Trình duyệt sẽ đọc từng Tag trong File HTML, và biến nó thành 1 Node trong DOM, khi gặp phải Tag Script, nó sẽ chuyển sang chạy hết Code JS được đặt trong Tag này tại chỗ, nghĩa là không gửi yêu cầu tới File JS bên ngoài, còn nếu Tag Script có chỉ định Link File JS bên ngoài, thì nó sẽ gửi yêu cầu để nhận nội dung của File này và tiến hành chạy nó, trong trường hợp này không được đặt Code JS trong Tag Script, trong trường hợp dùng File bên ngoài, bạn được phép chỉ định thêm việc chạy nó sau khi DOM được tạo xong xuôi hoặc chạy ngay lập tức, nếu chỉ định chạy sau khi DOM được tạo xong xuôi, thì File JS này sẽ được đưa vào hàng đợi để chạy sau, song song với việc tạo DOM, trình duyệt cũng tạo luôn CSSOM (CSS Object Model) có cấu trúc Rooted Tree như DOM, nhưng chứa thông tin về kiểu cách của các loại Node trong DOM, khi gặp Tag Link có thuộc tính rel = "stylesheet" chỉ định Link tới File CSS bên ngoài hoặc Tag Style với mã CSS bên trong, thì trình duyệt sẽ dựa vào đó mà thêm thắt vào CSSOM, chỉ khi cả DOM và CSSOM được tạo xong xuôi, chúng sẽ liên kết với nhau để tạo Render Tree, trình duyệt sẽ dựa vào đây mà kết xuất nội dung ra màn hình, tuy nhiên, nếu bạn chỉ định thuộc tính style cho Tag trực tiếp, thì nó sẽ ghi đè các thuộc tính trong CSSOM cho Tag này, khi mã JS của bạn làm thay đổi cấu trúc DOM và CSSOM, thì Render Tree thay đổi, do đó khi đó trình duyệt cũng sẽ cập nhật lại những thứ thay đổi trên trang Web
* Để tạo 1 Tag Script với Code JS bên trong

<script>

<Code JS Ở Đây>

</script>

* Để tạo 1 Tag Script sử dụng File JS bên ngoài

<script src = <Đường Dẫn Tới File JS Có Phần Mở Rộng>></script>

* <Đường Dẫn Tới File JS Có Phần Mở Rộng> có thể là đường dẫn tương đối với địa chỉ trên thanh tìm kiếm, hoặc Link tới File JS Online
* Để đưa File JS vào hàng đợi, khi nào DOM và CSSOM tạo xong thì mới chạy, thêm thuộc tính defer

<script src = <Đường Dẫn Tới File JS Có Phần Mở Rộng> defer></script>

* Để chỉ định File JS hoặc đoạn Code JS trong Tag Script là 1 Module, tức là thông báo cho trình duyệt biết đoạn Code này sẽ sử dụng các từ khóa import, export, …, để nó biết cách xử lí, ta thêm thuộc tính type = "module", các Module sẽ luôn chỉ được chạy sau khi DOM và CSSOM đã tạo xong xuôi, đồng thời, nếu có nhiều Tag Script với Module có tên giống nhau, nó chỉ gửi yêu cầu 1 lần
* Ví dụ

<script type = "module" src = "foo.js"></script>

<script type = "module" src = "foo.js"></script>

* Khi này, trình duyệt chỉ gửi yêu cầu 1 lần duy nhất tới File “foo.js”
* Để tạo 1 Tag Style với mã CSS bên trong

<style>

<Mã CSS Ở Đây>

</style>

* Để chỉ định File CSS ở bên ngoài để trình duyệt gửi yêu cầu tới

<link rel = "stylesheet" href = <Đường Dẫn Tới File CSS Có Phần Mở Rộng>>

* Ví dụ chỉ định thuộc tính style trực tiếp cho 1 Tag bằng mã CSS đặt trong 1 String

<h1 style = "color: red;"> ggnore </h1>

1. Comment HTML?

<!--<Đoạn Mã HTML Bị Comment>-->

1. Thuộc Tính Của Tag?

* Bạn có thể tạo thuộc tính bất kì với tên và giá trị bất kì cho nó trong Tag mà không có lỗi gì xảy ra

HTML Tag:

1. Tag Không Làm Gì Cả?

* Khi bạn chỉ muốn bao văn bản vào 1 Tag, nhưng không muốn tạo kiểu cách mặc định cho nó, dùng Tag Span
* Ví dụ

<span>abc</span>

1. Inline, Block, Inline Block?

* 1 Node trong DOM chỉ có thể thuộc 1 trong 3 kiểu là Inline, Block hoặc Inline Block, mỗi Node sẽ tương ứng 1 Box trên trang Web, 1 Box có 4 vùng từ thấp lên cao là Content Box, Padding Box, Border Box và Margin Box
* Mặc định Node HTML thuộc kiểu Block, chiều rộng của nó = chiều rộng màn hình, nếu nội dung trang mà vượt qua chiều rộng màn hình, thì sẽ có thanh cuộn ngang, chiều rộng Node HTML vẫn = chiều rộng màn hình
* Node Body thuộc kiểu Block với Margin 8 px 4 phía, do đó nó nằm gọn trong Node HTML, và do đó nằm gọn trong màn hình
* Xét 1 Node A bất kì thuộc kiểu Block, ta chỉ chú tâm đến Content Box của nó, tạm gọi chiều rộng và chiều cao Content Box này là W và H, ta định nghĩa cách bố trí nội dung theo đệ quy sau
* Coi nguyên trang Web chỉ là 1 chuỗi các kí tự và ảnh, mỗi kí tự và ảnh sẽ có kiểu cách riêng áp lên nó
* Mỗi kí tự văn bản tách thành 1 Node đơn lẻ, gọi là Node kí tự, Node kí tự thuộc kiểu Inline, Base Line của Node kí tự được xác định theo cơ chế sau, ví dụ xét chuỗi “AyooÁe”, thì Base Line của mỗi kí tự trong đây là đường thẳng chạm đít “A”, “o”, “e”, Bottom Line của mỗi kí tự trong đây là đường thẳng đi qua đầu cuống kí tự “y” và song song Base Line, Top Line của mỗi kí tự trong đây là đường thẳng đi qua điểm cao nhất của dấu sắc trong “Á” và song song Base Line
* Node kí tự chỉ có thể có Margin trái hoặc phải, Margin trên và dưới đều vô tác dụng, Bottom Edge của mỗi Node kí tự là đường thẳng cách Middle Line, là gạch ngang kí tự “e”, một khoảng = thuộc tính CSS line-height của kí tự này / 2, và nằm bên dưới, Top Edge tương tự nhưng nằm bên trên, lưu ý ở đây có thể có sai số
* Các Node ảnh đều là Node Inline Block, Content Box = cái ảnh, Base Line của nó = cạnh dưới, Bottom Edge = cạnh dưới Margin Box, Top Edge = cạnh trên Margin Box
* A chỉ có thể có loại Node con là Inline, Block và Inline Block, trong Block sẽ chỉ chứa Inline, trong Inline Block sẽ chứa cả Block và Inline, và trong Block con của Inline Block sẽ chỉ chứa Inline, bây giờ ta sẽ xét từng trường hợp
* Trường hợp 1, ta đang đặt các Node Inline cũng như Inline Block vào A, các Node con Inline và Inline Block sẽ đặt lần lượt từ trái qua phải, Margin trái thằng đầu tiên sẽ chạm cạnh trái của Content Box của A, đặt sao cho Margin phải thằng bên trái chạm Margin trái thằng bên phải, cho đến khi nào gặp phải Node con Block thì đưa Node con Block này xuống dòng dưới, bắt đầu dòng mới và căn chỉnh lại dòng cũ, hoặc gặp phải Node kí tự là dấu cách và đang tràn biên cạnh phải Content Box của A thì cũng xuống dòng, khi gặp Node Break thì cũng xuống dòng ngay lập tức, căn chỉnh dòng như sau, Base Line của các Node phải thẳng 1 hàng ngang, Top Edge cao nhất phải chạm Bottom Edge thấp nhất của dòng bên trên, nếu là dòng đầu tiên thì phải chạm cạnh trên Content Box của A
* Trường hợp 2, Node con Block có tổng chiều rộng Content Box của nó + Margin + Padding + Border mặc định sẽ bằng chiều rộng Content Box của A, tuy nhiên ta có thể thay đổi chiều rộng này, Top Edge của Node con Block = cạnh trên Margin Box của nó, Bottom Edge = cạnh dưới Margin Box, tất cả phần tử Inline nằm trong nó được sắp xếp như khi sắp các Node Inline vào A, nếu không chỉ định chiều cao cho Content Box của Node con Block này thì tự động = khoảng cách từ Bottom Edge thấp nhất của dòng thấp nhất đến Top Edge cao nhất của dòng cao nhất trong Node con Block này, sau khi kết xuất Node con Block này thì sẽ xuống dòng, Base Line của Node Block = Base Line của dòng dưới cùng trong nó
* Nếu 2 dòng Node Block liên tiếp, thì Margin ở chỗ tiếp xúc của 2 thằng sẽ dung hợp, tức là lấy Margin của thằng có Margin lớn hơn để ngăn cách Border
* Node con Inline Block , có thể chỉnh mọi thứ như chiều rộng, chiều cao Content Box của nó, cũng Margin, Padding, Border sẽ có tác dụng cả 4 hướng, các Node con trong nó được sắp xếp như khi sắp xếp các Node con Inline và Node con Block vào A, Base Line của Node con Inline Block = Base Line của dòng cuối cùng trong nó, nếu ko chỉ định chiều rộng cho Content Box của Node con Inline Block thì = tổng chiều dài của dòng dài nhất bao gồm cả Margin, Padding, Border, cách bố trí thì đã nêu trong phần Inline, chỉ có điều Bottom Edge của nó = cạnh dưới Margin Box và Top Edge = cạnh trên Margin Box như đã nói ở phần ảnh

1. Tag Div?

<div>

<Nội Dung>

</div>

* = Span, nhưng thêm thuộc tính display = block, nghĩa là tạo Node Block

1. Tag Đề Mục?

<h<Số>>

<Đề Mục>

</h<Số>>

* <Số> phải là số nguyên từ 1 tới 6
* Các thuộc tính mặc định bao gồm display = block, margin-inline-start và end = 0, font-weight = bold và các thuộc tính sau

|  |  |  |
| --- | --- | --- |
| <Số> | font-size | margin-block-start và end |
| 1 | 2 em | 0.67 em |
| 2 | 1.5 em | 0.83 em |
| 3 | 1.17 em | 1 em |
| 4 | 1 em | 1.33 em |
| 5 | 0.83 em | 1.67 em |
| 6 | 0.67 em | 2.33 em |

1. Tag Đoạn Văn?

<p>

<Nội Dung>

</p>

* Các thuộc tính mặc định

|  |  |
| --- | --- |
| Thuộc tính | Giá trị |
| display | block |
| margin-block-start | 1em |
| margin-block-end | 1em |

1. Tag Ảnh?

<img src = <Link Ảnh> alt = <Văn Bản Thay Thế>>

* Lệnh trên sẽ gửi yêu cầu tới <Link Ảnh>, khi không nhận được ảnh sẽ hiện

1 kí tự ảnh + dấu cách vô hình + <Văn Bản Thay Thế>, và hoạt động y chang Node Inline, khi này nó có các thuộc tính mặc định sau

|  |  |
| --- | --- |
| Thuộc tính | Giá trị |
| overflow | clip |
| overflow-clip-margin | content-box |

* Nếu nhận được ảnh, thì hoạt động như Node Inline Block, với vùng Content Box luôn chứa khít ảnh của bạn, mặc định kích thước Content Box = kích thước ảnh, bạn có thể chỉ định nó thành Node Block cũng được, nhưng nếu chỉnh nó thành Inline thì sẽ bị đổi thành Inline Block

1. Tag Liên Kết?

* Để tạo 1 Link tới trang Web khác

<a href = <Link>><Nội Dung></a>

* <Nội Dung> sẽ được hiển thị với kiểu cách, Click vào nó sẽ chuyển tới <Link>
* Nếu <Link> không phải Link tới trang Web khác, mà là 1 chuỗi nào đó, thì chuỗi đó sẽ được thêm vào sau địa chỉ trên thanh Search
* Các thuộc tính CSS mặc định

|  |  |
| --- | --- |
| Thuộc tính | Giá trị |
| color | #0000EE |
| text-decoration-line | underline |
| cursor | pointer |

* Nếu không chỉ định thuộc tính href cho Tag a, thì nó chả khác mẹ gì Tag span
* Khi nhấn Link, không có Tab mới tạo ra mà Tab hiện tại sẽ bị chuyển hướng

1. Tag List Item?

<li>

<Nội Dung Item>

</li>

* Tag này chỉ có thuộc tính CSS display = list-item, do đó nó sẽ có lớp giả marker, và kí tự đánh dấu mặc định là chấm đen

1. Tag Danh Sách Không Thứ Tự?

<ul>

<Danh Sách>

</ul>

* Các thuộc tính CSS mặc định

|  |  |
| --- | --- |
| Thuộc tính | Giá trị |
| display | block |
| list-style-type | disc |
| margin-block-start | 1 em |
| margin-block-end | 1 em |
| padding-inline-start | 40 px |

1. Tag Danh Sách Có Thứ Tự?

<ol>

<Danh Sách>

</ol>

* Giống y chang Tag ul nhưng thuộc tính CSS list-style-type = decimal

1. Tag Bảng?

<table>

<Nội Dung>

</table>

* Tag này sẽ có thuộc tính CSS mặc định display = table, tuy nhiên nếu ta không dùng Tag table mà chẳng hạn dùng span, hay bất cứ Tag nào khác với display = table, nó đều vô dụng, coi nó hoạt động như Block
* Các thuộc tính CSS mặc định

|  |  |
| --- | --- |
| Thuộc tính | Giá trị |
| border-collapse | separate |
| text-indent | 0 px |
| box-sizing | border-box |
| border-spacing | 2 px |
| border-color | gray |

* Trong <Nội Dung>, sẽ có 3 phần, Header, Body và Footer, nếu ta không tạo thì nó cũng tự động tạo
* Header

<thead>

<Nội Dung Header>

</thead>

* Body

<tbody>

<Nội Dung Body>

</tbody>

* Footer

<tfoot>

<Nội Dung Footer>

</tfoot>

* Để tạo 1 hàng chứa các ô đặt vào 1 trong 3 phần này

<tr>

<Các Ô>

</tr>

* Mỗi ô sẽ đặt trong 1 Tag th hoặc td, th in đậm căn giữa, td không in đậm căn trái
* Ví dụ

<tr>

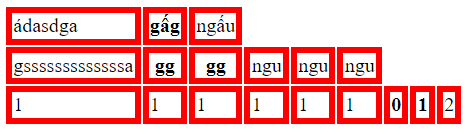
<th>foo</th>

<td>bar</td>

<th>bob</th>

</tr>

* Vì số ô ở các dòng có thể khác nhau, quy ước độ rộng cột thứ k = chiều rộng của ô có chiều rộng lớn nhất trong cột thứ k
* Minh họa



1. Tag Input?

<input name = <Tên> type = <Loại> value = <Nội Dung Hiển Thị>>

* Tag trên hoạt động như Node Inline Block, tuy nhiên các thuộc tính CSS mặc định khá phức tạp nên ta cũng không nên tạo kiểu lại cho nó làm gì
* Ta có

|  |  |
| --- | --- |
| <Loại> |  |
| "checkbox" | Ô vuông bo góc dùng để Tick hoặc Untick |
| "radio" | Vòng tròn dùng để Tick, một khi đã Tick không thể Untick, trong 1 File HTML, nếu có nhiều Tag input với type = radio, và cùng giá trị name, thì chỉ có tối đa 1 vòng tròn được Tick, Tick vòng khác, thì vòng trước Untick |
| "text" | Khung chữ nhật bo góc, Click chuột vào để nhập văn bản, Click chuột ra để không nhập nữa, văn bản nhập không bị xóa |
| "submit" | Nút bấm, Click chuột vào để bấm, chữ trên nút bấm = <Nội Dung Hiển Thị>, mặc định là “Submit” |

1. Tag Button?

<button>

<Nội Dung>

</button>

* Tag trên hoạt động như Node Inline Block, <Nội Dung> sẽ được bao trong 1 nút bấm, Click chuột lên nó để bấm

1. Tag Label?

<label>

<Nội Dung>

</label>

* Chả khác mẹ gì Tag span, ngoại trừ việc có thêm thuộc tính CSS cursor = default

1. Thẻ Form?

<form action = <Link Submit>>

<Nội Dung>

</form>

* Các thuộc tính CSS mặc định

|  |  |
| --- | --- |
| Thuộc tính | Giá trị |
| display | Block |
| margin-top | 0 px |

* Bất cứ khi nào 1 Tag input với thuộc tính type = "submit" nằm trong <Nội Dung> bị bấm vào, thì Tab hiện tại sẽ lập tức chuyển hướng tới <Link Submit>, nếu không chỉ định thì chuyển hướng tới chính Tab hiện tại

1. Thuộc Tính Lang?

* Thêm thuộc tính lang = <Ngôn Ngữ> vào 1 Tag để 1 số trình duyệt khi truy cập trang Web của bạn, thì nếu <Ngôn Ngữ> khác xa tiếng Anh, như tiếng Việt, nó sẽ hỏi có muốn dịch nội dung trong Tag này từ tiếng Việt sang tiếng Anh không, mặc định <Ngôn Ngữ> = tiếng Anh
* Ta có

|  |  |
| --- | --- |
| <Ngôn Ngữ> |  |
| "vi" | Tiếng Việt |
| "en" | Tiếng Anh |

* Ví dụ

<h1 lang = "vi">

foo

</h1>

1. Favicon?

* Là biểu tượng nhỏ nằm bên trái tiêu đề Tab
* Để thêm Favicon

<link rel = "shortcut icon" href = "<Đường Dẫn Tới Favicon>">

* Ví dụ

<link rel = "shortcut icon" href = "foo.png">

CSS:

1. Thuộc Tính Kế Thừa?

* Có 1 số thuộc tính CSS nếu bạn không chỉ định ở Node con thì nó sẽ có giá trị = thuộc tính CSS đó ở Node cha
* Các thuộc tính CSS còn lại không có tính kế thừa nếu bạn không chỉ định giá trị cho thuộc tính CSS đó, thì nó sẽ được gán giá trị mặc định đã định sẵn
* Để trả về giá trị đã định sẵn, dùng initial
* Ví dụ

color: initial;

* Với thuộc tính không kế thừa, dùng từ khóa inherit để kế thừa
* Ví dụ

border: inherit;

1. Selector?

* Để áp kiểu cách lên Node trong DOM, cần phải xác định đó là những Node nào, việc làm này thông qua Selector, cú pháp CSS như sau

<Selector> {

<Các Thuộc Tính Và Giá Trị>

}

* <Các Thuộc Tính Và Giá Trị> có cú pháp như sau

<Tên Thuộc Tính> : <Giá Trị>;

* <Selector> sẽ là thứ chỉ định Node được áp kiểu, ta có bảng giá trị ưu tiên sau

|  |  |
| --- | --- |
| <Selector> | Giá trị ưu tiên |
| \* | 0-0-0-0 |
| <Tên Tag> | 0-0-0-1 |
| .<Tên Class> | 0-0-1-0 |
| #<ID> | 0-1-0-0 |

* .<Tên Class> sẽ chọn tất cả các Node có thuộc tính class = <Tên Class>, lưu 1 Node có thể thuộc nhiều Class, ngăn cách nhau bởi dấu cách, ví dụ class = "foo bar" thì vừa thuộc Class foo vừa thuộc Class bar
* #<ID> sẽ chọn tất cả các Node có thuộc tính id = <ID>
* \* sẽ chọn tất cả các Node trong DOM
* Giá trị ưu tiên được hiểu như sau, mỗi số trong chùm 4 không có giới hạn, ví dụ 11-100-99-4 vẫn hợp lệ, và khi so sánh, thì so từ trái sang từng số một, ví dụ 99-4-9-5 sẽ lớn hơn 98-4-9-5 và cũng lớn hơn 99-4-8-5
* Giá trị ưu tiên của 1 Selector = tổng giá trị ưu tiên của các Selector đơn lẻ hợp thành nó
* Ví dụ 1

h1 {

width: 100px;

}

* Thì thuộc tính CSS width của tất cả Tag h1 đều trở thành 100px, nhưng chỉ trong trường hợp việc chỉ định thuộc tính CSS width trong đoạn CSS này có độ ưu tiên cao nhất, độ ưu tiên hiện tại của nó = 0-0-0-1
* Ví dụ 2

.foo {

height: 100px;

}

* Tất cả các Node có thuộc tính class = "foo" đều được áp thuộc tính CSS height = 100px, chỉ trong trường hợp đoạn CSS này có độ ưu tiên cao nhất, độ ưu tiên hiện tại của nó = 0-0-1-0
* Ví dụ 3

#bar {

background-color: red;

}

* Tất cả các Node có thuộc tính id = "bar" đều được áp thuộc tính CSS background-color = red, chỉ trong trường hợp đoạn CSS này có độ ưu tiên cao nhất, độ ưu tiên hiện tại của nó = 0-1-0-0
* Ví dụ 4, kết hợp nhiều Selector, phải viết sát nhau, tên Tag nếu có phải đặt ở đầu, có thể lặp Selector để tăng độ ưu tiên

h1.foo#bar.foo.foo#bar {

color: blue;

}

* Tất cả các Tag h1 với id = "bar" và class = "foo" đều được áp thuộc tính CSS color = blue, chỉ trong trường hợp đoạn CSS này có độ ưu tiên cao nhất, độ ưu tiên hiện tại của nó = 0-0-0-1 +(0-0-1-0) \* 3 + (0-1-0-0) \* 2 = 0-2-3-1
* Nếu chỉ định cùng 1 thuộc tính CSS trong 2 Selector khác nhau cùng độ ưu tiên, thì cái viết sau sẽ được áp
* Khi bạn chỉ định trực tiếp, ví dụ width = 100px trong thuộc tính style của Tag nào đấy, thì khi này độ ưu tiên của nó = 1-0-0-0
* Các thuộc tính CSS mặc định đã tích hợp sẵn vào Tag có độ ưu tiên = 0-0-0-0, do đó bạn có thể ghi đè chúng bất cứ lúc nào
* Nếu bạn muốn việc chỉ định giá trị cho thuộc tính CSS nào đó có độ ưu tiên tối cao, tức là độ ưu tiên = vô tận, thì thêm !impotant
* Ví dụ

h1 {

color: blue !important;

}

* Tất cả các Tag h1 đều được áp thuộc tính CSS color = blue, nó không thể bị ghi đè
* Tuy nhiên, nếu bạn thêm !important ở cả 2 nơi, thì nó sẽ chọn nơi có độ ưu tiên lơn hơn khi bỏ !important, nếu bằng nhau thì chọn cái viết sau
* Các thuộc tính kế thừa có độ ưu tiên = 0-0-0-0
* Trong VS Code, bạn có thể xem độ ưu tiên của 1 Selector bằng cách di chuột lên nó

1. Combinator?

* Là phép kết hợp các Selector
* Để chỉ chọn những Node là Node con cháu của Node tổ tiên nào đó

<Node Tổ Tiên> <Node Con Cháu>

* Ví dụ

div p {  
 color: green;

}

* Khi này chỉ có những Tag p là con cháu của Tag div mới được áp thuộc tính CSS color = green, độ ưu tiên = 0-0-0-1 + 0-0-0-1 = 0-0-0-2
* Để chỉ chọn những Node là Node con của Node cha nào đó

<Node Cha>><Node Con>

* Ví dụ

div>p {  
 color: green;

}

* Khi này chỉ những Tag p là con trực tiếp của Tag div mới được áp thuộc tính CSS color = green, độ ưu tiên = 0-0-0-1 + 0-0-0-1 = 0-0-0-2
* Để chỉ chọn những Node là Node anh em và đứng ngay sau Node nào đó

<Node Trước>+<Node Sau>

* Ví dụ

div+p {

color: green;

}

* Khi này chỉ những Tag p là Tag anh em của Tag div và phải đứng ngay sau Tag div mới được áp thuộc tính CSS color = green, độ ưu tiên = 0-0-0-1 + 0-0-0-1 = 0-0-0-2
* Để chỉ chọn những Node là Node anh em và đứng sau Node nào đó, không cần đứng ngay sau

<Node Trước>~<Node Sau>

* Ví dụ

div~p {

color: green;

}

* Khi này chỉ những Tag p là Tag anh em của Tag div và đứng sau Tag div mới được áp thuộc tính CSS color = green, độ ưu tiên = 0-0-0-1 + 0-0-0-1 = 0-0-0-2
* Tương tự như tên Tag, thông thường, bạn có thể kết hợp với Class và ID
* Ví dụ

h1~span#foo.bar {

color: green;

}

* Khi này chỉ những Tag span có id = "foo" và class = "bar" là Tag anh em của Tag h1 và đứng sau Tag h1 mới được áp thuộc tính CSS color = green, độ ưu tiên = 0-0-0-1 + 0-0-0-1 + 0-1-0-0 + 0-0-1-0 = 0-1-1-2

1. Font Size?

* Mặc định Node HTML có thuộc tính CSS “font-size: 16px;”, tất cả các thuộc tính CSS liên quan tới Font đều có tính kế thừa
* Đơn vị “em” nếu dùng trên thuộc tính CSS font-size của Node A thì 1 em = font-size của Node cha của A, nếu dùng trên các thuộc tính CSS khác font-size như width, height, … của A thì 1 em = font-size của A

1. Màu Thập Lục Phân?

* Thêm dấu # đằng trước
* Ví dụ

#0000EE

1. Màu Chữ?

color: <Màu>;

1. Gạch Chữ?

text-decoration-line: <Gạch Ở Đâu>;

* Ta có

|  |  |
| --- | --- |
| <Gạch Ở Đâu> |  |
| none | Mặc định, không gạch gì hết |
| overline | Hiện Top Line lệch 1 tí của kí tự |
| underline | Hiện Base Line xuống dưới 1 tí của kí tự, nếu kí tự ví dụ như g, nó có cuống ở phía dưới thì có thể che con mẹ nó underline |
| line-through | Hiện Middle Line lệch 1 tí của kí tự |

* Có thể hiện cùng lúc nhiều gạch, ví dụ

text-decoration-line: overline line-through;

* Để chỉ định kiểu gạch

text-decoration-style: <Kiểu Gạch>;

* Ta có

|  |  |
| --- | --- |
| <Kiểu Gạch> |  |
| solid | Mặc định, gạch thẳng |
| dashed | Gạch đứt quãng |

* Để chỉ định màu gạch

text-decoration-color: <Màu>;

* Để chỉ định độ dày gạch

text-decoration-thickness: <Độ Dày>;

* Có thể viết gộp 4 thuộc tính trên lại

text-decoration: <Các Thuộc Tính>;

* Ví dụ

text-decoration: overline underline dashed 10px red;

1. Viết In Đậm?

* Thêm thuộc tính CSS “font-weight: bold;”

1. Viết Số Kiểu Cách?

font-variant-numeric: <Kiểu Cách>;

* Thuộc tính CSS này có tính kế thừa
* Áp dụng cho toàn bộ kí tự số trong Node
* Ta có

|  |  |
| --- | --- |
| <Kiểu Cách> |  |
| normal | Mặc định, như bạn hay thấy |
| oldstyle-nums | Viết các số với độ cao chênh lệch nhau |
| tabular-nums | Chiều rộng mỗi số = nhau |

1. Chuyển Sang In Hoa Hoặc In Thường?

text-transform: <Kiểu In>;

* Thuộc tính CSS này có tính kế thừa
* Ta có

|  |  |
| --- | --- |
| <Kiểu In> |  |
| none | Mặc định, không làm gì hết |
| uppercase | Tất cả kí tự đều in hoa |
| lowercase | Tất cả kí tự đều in thường |
| capitalize | Biến kí tự đầu tiên mỗi từ viết hoa |

1. Viết Chữ Từ Trên Xuống?

* Thêm thuộc tính CSS “writing-mode: vertical-lr”
* Xét Node có thuộc tính CSS này, quay Box của nó ngược chiều kim đồng hồ góc 90 độ, được điểm nhìn A, gọi điểm nhìn cũ là B, viết chữ từ trái sang phải, hết dòng thì lên dòng bên trên, bây giờ giới hạn không còn là chiều rộng mà là chiều cao màn hình, đồng thời kí tự và ảnh trong mỗi dòng sẽ được căn giữa dòng, lưu ý ảnh không bị quay ở điểm nhìn B
* Base Line so với viết theo kiểu ngang là kí tự bên phải cùng ở điểm nhìn A
* Tương tự “writing-mode: vertical-rl”, quay Box 90 độ ngược chiều kim đồng hồ, viết từ trái sang, hết dòng thì xuống dưới
* Ta có chỉ định thuộc tính CSS direction để xác định chiều viết, điểm nhìn trái phải sẽ bị đảo ngược

direction: <Chiều>;

* Thuộc tính CSS này có tính kế thừa
* Ta có

|  |  |
| --- | --- |
| <Chiều> |  |
| ltr | Mặc định, đéo có gì để nói |
| rtl | Chưa đặt dòng vào Content Box vội, xét mỗi dòng, lặp qua từng Node con, nếu bắt gặp Node Inline Block thì đẩy nó sang bên phải Content Box, nếu gặp Node Inline, thì đẩy 1 lúc hết phần còn lại |

* Theo điểm nhìn A, margin-block-start = margin-top, margin-block-end =

margin-bottom, margin-inline-start = margin-left, margin-inline-end = margin-right, tương tự cho cả padding

1. Hàm Trả Về Giá Trị Thuộc Tính Tag?

attr(<Tên Thuộc Tính>)

* Lệnh trên sẽ trả về String, do đó nó chỉ có thể áp dụng cho thuộc tính CSS content của lớp giả before và after
* Ví dụ

h1::after {

content: attr(daucac);

}

* Khi này content = giá trị của thuộc tính daucac của Tag h1 được chọn

1. Đặt Biến?

* Biến được khai báo ở Node cha, sẽ có thể dùng lại ở Node con cháu
* Nếu 1 biến được khai báo 2 lần, thì lần nào Selector có độ ưu tiên cao hơn thì chọn, nếu cùng độ ưu tiên thì lấy thằng viết sau
* Cú pháp

--<Tên Biến> : <Giá Trị>;

* Để trả về giá trị của 1 biến

var(--<Tên Biến>)

* Để nếu biến không tồn tại thì lấy giá trị mặc định nào đó

var(--<Tên Biến>, <Giá Trị Mặc Định>)

* Ví dụ

.foo {

--daucac: red;

--daulon: 10px;

}

.bar {

color: var(--daucac);

width: var(--daulon, 8px);

}

1. Padding?

* Để chỉ định Padding cho 1 phía

padding-<Phía>: <Kích Thước>;

* <Phía> phải là top, left, bottom hoặc right
* Ví dụ

padding-top: 10px;

* Để chỉ định Padding cho nhiều phía cùng lúc

padding: <Các Kích Thước>;

* Ta có

|  |  |
| --- | --- |
| Số kích thước trong <Các Kích Thước> | Áp dụng phía nào |
| 1 | Trái phải trên dưới |
| 2 | Trên dưới, trái phải |
| 3 | Trên, trái phải, dưới |
| 4 | Trên, phải, dưới, trái |

* Ví dụ

padding: 10px 20px 30px;

* Khi này Padding trái = phải = 20px, Padding trên = 10px, Padding dưới = 30px

1. Margin?

* Cú pháp y chang Padding, thay mỗi chữ “padding” thành “margin”

1. Border?

* Border phân ra 4 phía, trên, dưới, trái, phải, bạn có thể tạo kiểu cách riêng cho mỗi phía
* Để chỉ định kích thước Border cho 1 phía

border-<Phía>-width: <Kích Thước>;

* Để chỉ định kích thước Border cho nhiều phía cùng lúc

border-width: <Các Kích Thước>;

* Để chỉ định kiểu Border cho 1 phía

border-<Phía>-style: <Kiểu Border>;

* Ta có

|  |  |
| --- | --- |
| <Kiểu Border> | Kết xuất |
| solid | Đường thẳng bình thường, nếu kích thước Border của phía nào không được chỉ định thì mặc định = 2px, nếu không chỉ định màu cho Border thì mặc định màu đen |
| hidden | Đây là kiểu mặc định nếu không chỉ định kiểu Border, tương đương việc không có Border nào tồn tại, tương đương kích thước = 0 bất chấp |

* Để chỉ định kiểu Border cho nhiều phía cùng lúc

border-style: <Các Kiểu Border>;

* Để chỉ định màu Border cho 1 phía

border-<Phía>-color: <Màu>;

* Để chỉ định màu Border cho nhiều phía cùng lúc

border-color: <Các Màu>;

* Để ý thấy cú pháp nhiều phía tương tự Padding
* Để chỉ định cùng lúc nhiều thuộc tính CSS cho 1 phía

border-<Phía>: <Các Thuộc Tính>;

* <Các Thuộc Tính> bao gồm kiểu Border, kích thước, và màu
* Ví dụ

border-top: solid red 10px;

* Để chỉ định cùng lúc nhiều thuộc tính CSS chung cho 4 phía

border: <Các Thuộc Tính>;

* Ví dụ

border: solid red 10px;

1. Kích Thước Box?

* Dựa vào thuộc tính CSS box-sizing mà 2 thuộc tính sau áp cho Content Box, Padding Box, hay Border Box

width: <Kích Thước>;

* Sau khi đã xác định được width, thì bạn có thể Clip nó lại bằng

min-width: <Chiều Rộng Tối Thiểu>;

max-width: <Chiều Rộng Tối Đa>;

* Để chỉ định chiều cao Content Box

height: <Kích Thước>;

* Sau khi đã xác định được height, bạn có thể Clip nó lại bằng

min-height: <Chiều Cao Tối Thiểu>;

max-height: <Chiều Cao Tối Đa>;

* Mặc định áp cho Content Box
* Để chuyển sang áp cho Box khác

box-sizing: <Box Khác>;

* Ví dụ

box-sizing: border-box;

* Khi này width và height chỉ định chiều rộng và chiều cao Border Box
* Lưu ý khi này nếu giảm width và height làm sao cho phần Content Box bị nén về hư vô, thì có giảm xuống nữa vẫn không thay đổi gì

1. Màu Nền?

* Để chỉ định màu nền, lưu ý màu nền sẽ phủ kín Content Box, Padding Box hay Border Box tùy thuộc vào thuộc tính CSS background-clip

background-color: <Màu>;

* Mặc định phủ Border Box, và bị tất cả những thứ như Border, chữ đè lên trên, để chỉ phủ Box khác như Padding hay Content Box

background-clip: <Box Khác>;

* Ví dụ

background-clip: content-box;

1. Ảnh Nền?

* Để chỉ định ảnh nền, lưu ý ảnh nền sẽ phủ kín Content Box, Padding Box hay Border Box tùy thuộc vào thuộc tính CSS background-clip

background-image: url(<Đường Dẫn Tới Ảnh>);

* Lệnh trên sẽ gửi yêu cầu tới <Đường Dẫn Tới Ảnh>
* Ví dụ

background-image: url("foo.png");

* Ngoài ra, bạn có thể chỉ định ảnh là 1 dải màu

linear-gradient(<Góc>, <Từ Màu>, <Đến Màu>)

* <Góc> = 0 tương ứng chiều mũi tên hướng từ dưới lên trên, tăng <Góc> thì quay cùng chiều kim đồng hồ
* Dải màu sẽ từ gốc của mũi tên với màu = <Từ Màu>, lần theo hướng mũi tên mà chuyển dần thành <Đến Màu>
* Các đơn vị <Góc>

|  |  |
| --- | --- |
| deg | Độ |

* Dải màu coi như 1 ảnh nền, kích thước ảnh này mặc định = kích thước Box ứng với thuộc tính CSS background-clip
* Ví dụ

background-image: linear-gradient(123deg, red, yellow);

* Bạn có thể chỉ định nhiều ảnh 1 lúc, ảnh viết trước thì hiện ở phía trên, một số thuộc tính sẽ áp dụng chung cho các ảnh, ví dụ

background-image:

url("foo.png"),

url("bar.png"),

linear-gradient(123deg, red, yellow)

;

* Khi này, ta có thể chỉ định nhiều giá trị cho thuộc tính CSS background-clip, mỗi giá trị sẽ ứng với 1 ảnh riêng, nghĩa là mỗi ảnh sẽ có vùng chứa khác nhau, thuộc tính CSS background-color sẽ lấy Box đầu
* Ví dụ

background-clip: content-box, border-box;

* Gốc tọa độ để đặt ảnh sẽ tùy thuộc vào thuộc tính background-origin, mặc định là Padding Box, do đó khi đặt ảnh, góc trái trên ảnh sẽ khít với góc trái trên Padding Box

background-origin: <Box Khác>;

* Phần ảnh vượt quá Box sẽ bị ẩn, tất cả mọi thứ sẽ đè lên ảnh nền trừ màu nền, mặc định ảnh sẽ lặp lại theo cả chiều ngang và dọc, lặp theo cả chiều âm, để chỉ định nó chỉ lặp lại theo 1 phương hoặc không lặp lại theo chiều nào

background-repeat: <Chiều Lặp>;

* Ta có

|  |  |
| --- | --- |
| <Chiều Lặp> | Cách lặp |
| repeat | Đây là mặc định, lặp theo cả phương ngang lẫn dọc |
| repeat-x | Chỉ lặp theo phương ngang |
| repeat-y | Chỉ lặp theo phương dọc |
| no-repeat | Không lặp |

* Mặc định ảnh nền có chiều rộng và ngang = kích thước thực của nó, để điều chỉnh lại

background-size: <Chiều Rộng Mới> <Chiều Cao Mới>;

* Ví dụ

background-size: 10px 20px;

* Để đảm bảo tỉ lệ ngang / dọc như kích thước thực, dùng từ khóa auto và chỉ chỉ định 1 kích thước
* Ví dụ

background-size: 100px auto;

* Khi này chiều rộng ảnh = 100px, chiều cao sẽ tự xác định để tỉ lệ = tỉ lệ thực
* Để điều chỉnh lại tự động, giữ nguyên tỉ lệ ngang / dọc của các ảnh nền, thu nhỏ chúng lại cho đến khi tất cả ảnh nền đều không có phần nào bị vượt biên khỏi Box ứng với thuộc tính CSS background-clip

background-size: contain;

* Để điều chỉnh lại tự động, giữ nguyên tỉ lệ ngang / dọc của các ảnh nền, phóng to chúng cho đến khi mỗi ảnh nền đều đủ sức bao kín Box ứng với thuộc tính CSS background-clip

background-size: cover;

* Để Offset ảnh nền khỏi gốc tọa độ

background-position: <Hoành Độ> <Tung Độ>;

* Lưu ý trục tung hướng xuống dưới
* Ví dụ

background-position: 10px 20px;

* Để tự động căn lề ảnh so với Box ứng với thuộc tính CSS background-clip

background-position: <Các Căn Lề>;

* Ta có

|  |  |
| --- | --- |
| <Căn Lề> |  |
| center | Căn giữa |
| left | Căn trái |
| right | Căn phải |
| bottom | Căn dưới |
| top | Căn trên |

* Ví dụ

background-position: 10px center;

* Là ta đang Offset trái 10px và căn giữa ảnh theo chiều dọc

1. Tính Toán Với Các Đơn Vị Khác Nhau?

* Dùng hàm sau

calc(<Biểu Thức>)

* Ví dụ

width: calc(1em + 4px);

1. Lớp Giả?

* Loại lớp giả 1

<Selector>:<Trigger> {

<Mã CSS>

}

* <Mã CSS> sẽ được áp lên các Node chỉ định bởi <Selector> khi và chỉ khi hành động <Trigger> được thực thi trên các Node này, và sau khi hành động này kết thúc, các kiểu cách đã áp sẽ bị xóa
* Ta có

|  |  |
| --- | --- |
| <Trigger> | Khi nào kích hoạt |
| hover | Khi bạn di chuột lên trên Node |
| active | Khi bạn nhấn giữ chuột trái vào Node |
| first-child | Khi Node được chọn là Node con đầu tiên của Node cha của nó |
| last-child | Khi Node được chọn là Node con cuối cùng của Node cha của nó, lưu ý có vài trường hợp Tag Script vô tình được chèn ở cuối Node Body, do đó Tag Script mới là Node con cuối cùng của Node Body |
| nth-last-child(<Index>) | Khi Node được chọn là Node con thứ <Index> từ dưới lên của Node cha của nó, <Index> = 1 tương đương last-child |

* Ví dụ

h1:hover {

color: red;

}

* Loại lớp giả 2 tương tự

|  |  |
| --- | --- |
| <Trigger> | Áp vào đâu |
| :first-letter | Kí tự đầu tiên |
| :first-line | Dòng đầu tiên, tức là từ kí tự đầu tiên tới trước kí tự xuống dòng |
|  |  |

* Ví dụ

h1::first-letter {  
 color: red;

}

* Loại lớp giả 3 đặc biệt

|  |  |
| --- | --- |
| <Trigger> | Cơ chế |
| :before | Tạo 1 Node giả trước Node con đầu tiên, mặc định như 1 Node văn bản rỗng |
| :after | Tạo 1 Node giả ngay sau Node con cuối cùng, mặc định như 1 Node văn bản rỗng |
| :selection | Nếu kí tự bị bôi đen thì áp kiểu không thì thôi |
|  |  |

* Ví dụ

h1::selection {  
 color: blue;

}

* Độ ưu tiên của lớp giả = độ ưu tiên của Class, ví dụ h1::selection = độ ưu tiên của h1 + độ ưu tiên của selection = 0-0-0-1 + 0-0-1-0 = 0-0-1-1

1. Vị trí?

* Mặc định mỗi Node có thuộc tính CSS “position: static;”, khi này, thuộc tính CSS top, left, bottom, right sẽ vô tác dụng, nghĩa Node sẽ luôn đứng im tại chỗ

position: <Kiểu Vị Trí>;

* ta có

|  |  |
| --- | --- |
| <Kiểu Vị Trí> |  |
| static | Mặc định, không thể Offset |
| relative | Gốc tọa độ là vị trí mặc định, tức là vị trí khi position = static, có thể dịch sang phải bằng left, dịch sang trái bằng right, dịch xuống dưới = top và dịch lên trên bằng bottom, nếu chỉ định left thì right vô dụng, nếu chỉ định top, thì bottom vô dụng, khi Offset, bản thể thật sự của nó vẫn đứng yên, tức là phần nó chiếm chỗ vẫn như cũ, chỉ có phần kết xuất ra là bị Offset |
| absolute | Gốc tọa độ là góc trái trên Padding Box của Node tổ tiên gần nhất có thuộc tính CSS position khác static, nếu không tìm thấy thì lấy Node HTML, tọa độ (0, 0) thì góc trái trên của Border Box của Node này sẽ = gốc tọa độ = góc trái trên của Padding Box của Node tổ tiên đã nói, tuy nhiên điều này chỉ áp dụng khi bạn chỉ sử dụng left và top, nếu bạn sử dụng right và bottom, thì gốc tọa độ sẽ là góc phải dưới của Padding Box của Node tổ tiên đã nói, góc phải dưới Border Box của Node này sẽ = gốc tọa độ này khi tọa độ = (0, 0), cơ chế dịch tương đồng với left và right, chỉ đảo chiều lại, nếu cùng chỉ định left, right và width, thì right bị lơ, nếu cùng chỉ định left và right nhưng không chỉ định width, thì width sẽ tự điều chỉnh, tương tự cho top và bottom và height, còn một điều lưu ý nữa là Node này sẽ bị bứt ra khỏi trang Web, nghĩa là phần không gian nó chiếm khi position = static sẽ được trả tự do, và Node này sẽ trong trạng thái lơ lửng |
| fixed | Gốc tọa độ là góc trái trên màn hình, bứt Node này ra khỏi trang Web cho nó nổi tự do, phần không gian nó chiếm cũng bị trả lại, coi màn hình là Node tổ tiên của nó, cơ chế top, left, bottom, right y chang như absolute |

1. Xử Lí Tràn Biên?

* Giả sử nội dung bị tràn ra khỏi Padding Box của Node nào đó, bạn muốn ẩn nó đi và tạo thanh cuộn, xử lí cả tràn ngang và tràn dọc

overflow: <Cách Xử Lí>;

* Ta có

|  |  |
| --- | --- |
| <Cách Xử Lí> |  |
| visible | Mặc định, không xử lí, hiện cả nội dung tràn |
| hidden | ẩn nội dung tràn khỏi Padding Box, kể cả phần ảnh bị tràn |
| scroll | = hidden, nhưng có thêm 2 thanh cuộn, 1 cái nằm bên trái cạnh phải của Padding Box, 1 cái nằm bên trên cạnh dưới của Padding Box, luôn hiện dù tràn hay không tràn |
| auto | = scroll, nhưng chỉ hiện thanh cuộn khi tràn |

* Để chỉ xử lí tràn ngang hoặc tràn dọc

overflow-<Phương>: <Cách Xử Lí>;

* <Phương> = x là tràn ngang, = y là tràn dọc, nếu chỉ chỉ định 1 phương thì phương còn lại mặc định = auto
* Nếu chỉ định theo cách này thì bạn không được phép chỉ định 1 phương visible và phương còn lại không visible, để có thể làm được điều này, dùng clip, nó giống hidden nhưng khi chỉ định lên 1 phương, thì phương còn lại được phép visible
* Ví dụ

overflow-x: hidden;

overflow-y: visible;

* Phương x hidden, phương y auto

overflow-x: clip;

overflow-y: visible;

* Phương x hidden, phương y visible
* Để hiện thêm 1 phần ngoài rìa Padding Box

overflow-clip-margin: <Kích Thước>;

* Lệnh này chỉ có tác dụng khi cả 2 phương đều dùng clip
* <Kích Thước> có thể là content-box, border-box, mặc định là 0px kể từ Padding Box
* Ví dụ

overflow: clip;

overflow-clip-margin: 20px;

* Nghĩa là sẽ hiện trong vùng Padding Box đã mở rộng thêm 20px ra 4 phía

1. Xử Lí Con Trỏ Chuột Khi Di Lên Content Box Của Node?

cursor: <Kiểu Con Trỏ>;

* Ta có

|  |  |
| --- | --- |
| <Kiểu Con Trỏ> |  |
| auto | Mặc định, hình con trỏ văn bản chữ I nếu di lên chữ, hình mũi tên nếu di lên những thứ khác |
| pointer | Luôn là hình bàn tay |
| default | Luôn là hình mũi tên |

1. Căn Chữ?

text-align: <Kiểu Căn>;

* Thuộc tính CSS này có tính kế thừa, chỉ có tác dụng trên Node Inline Block hoặc Node Block
* Ta có

|  |  |
| --- | --- |
| <Kiểu Căn> |  |
| left | Mặc định, đéo làm gì |
| right | Căn phải tất cả các Node con Inline và Inline Block, nghĩa là gạt chúng sang bên phải |
| center | Căn giữa tất cả các Node con Inline và Inline Block |
| justify | Khi có từ 2 dòng chỉ chứa Node con Inline hoặc Inline Block trở lên, các Node này sẽ có khoảng trống cách nhau ra để Node đầu mỗi dòng chạm cạnh trái Content Box Node cha, và Node cuối mỗi dòng chạm cạnh phải Content Box Node cha |
| start | = left khi thuộc tính CSS direction = ltr và = right khi thuộc tính CSS direction = rtl |
| end | = left khi thuộc tính CSS direction = rtl và = right khi thuộc tính CSS direction = ltr |

* Lưu ý tất cả kiểu căn chữ ở đây đều phụ thuộc vào điểm nhìn được xác định bởi thuộc tính CSS writing-mode
* Để chỉnh sửa căn dòng cuối cùng

text-align-last: <Kiểu Căn>;

* Để thụt đầu dòng dòng đầu tiên, có thể thụt lùi, mặc định không thụt

text-indent: <Độ Thụt>;

1. Danh Sách?

display: list-item;

* Khi 1 Node có thuộc tính CSS trên, nó sẽ được xem là Node Block, đồng thời được tạo thêm 1 lớp giả marker, đứng trước cả lớp giả before, lớp giả này có các thuộc tính CSS mặc định sau

|  |  |
| --- | --- |
| Thuộc tính | Giá trị |
| font-variant-numeric | tabular-nums |
| text-indent | 0px !important |
| text-align | start !important |
| text-align-last | start !important |

* Để tạo kiểu cho lớp giả này

<Tên Tag>::marker {

<Mã CSS>

}

* Trong <Mã CSS>, để xác định kí tự đánh dấu

content: "<Kí Tự>";

* Nếu <Kí Tự> gồm nhiều kí tự, nó lấy kí tự cuối cùng
* Ví dụ

span::marker {

content: "g";

}

* Để chỉ định kí tự đánh dấu từ Node cha cho Node con, đặt lệnh sau vào Node cha, lưu ý bên phải kí tự đánh dấu sẽ có Margin ngắn

list-style-type: <Kí Tự>;

* Thuộc tính CSS này có tính kế thừa và sẽ bị thuộc tính CSS content của lớp marker của Node con ghi đè
* Ta có

|  |  |
| --- | --- |
| <Kí Tự> |  |
| disc | Mặc định, chấm đen, |
| circle | Đường tròn |
| decimal | Đánh số từ 1. trở đi, bắt đầu từ Node có thuộc tính CSS này, lan truyền xuống phía dưới bằng DFS, gặp Node có thuộc tính CSS display = list-item thì đánh số, gặp Node con cũng có thuộc tính decimal thì tạo 1 Process con, DFS bên trong Node này, bắt đầu từ 1. |

1. Định Dạng Bảng?

* Các thuộc tính sau chỉ sử dụng được với Tag table

border-collapse: <Hợp Nhất Không>;

* Thuộc tính này có tính kế thừa
* Ta có

|  |  |
| --- | --- |
| <Hợp Nhất Không> |  |
| seperate | Các ô trong bảng sẽ có Border cho riêng nó |
| collapse | 2 ô liền nhau trong bảng sẽ dùng chung Border ở chỗ tiếp xúc |

* Để Border của các ô cách nhau 1 khoảng, mặc định sát nhau, chỉ áp dụng khi border-collapse = separate

border-spacing: <Khoảng Cách>;

1. Flex Box?

display: flex;

* Node Flex = Node Block, và khác khá nhiều thứ, nó có 2 trục là trục chính và trục phụ, mặc định trục chính hướng từ trái sang phải, trục phụ từ trên xuống dưới
* Để thay đổi chiều trục chính và trục phụ

flex-direction: <Chiều>;

* Ta có

|  |  |  |
| --- | --- | --- |
| <Chiều> | Chiều trục chính | Chiều trục phụ |
| row | Trái sang phải | Trên xuống dưới |
| row-reverse | Phải sang trái | Dưới lên trên |
| column | Trên xuống dưới | Trái sang phải |
| column-reverse | Dưới lên trên | Phải sang trái |

* Lưu ý chiều trục phụ ở đây đang xét trong trường hợp thuộc tính CSS flex-wrap = no-wrap là cái mặc định hoặc wrap, nếu = wrap-reverse thì tất cả chiều trục phụ bảng trên sẽ bị đổi ngược chiều
* Các bước để kết xuất 1 Node Flex
* Bước 1, khởi tạo, tất cả Node con đéo quan tâm thuộc tính CSS display, đều hành xử như Inline Block, nghĩa là có thể chỉ định chiều rộng, chiều cao, …
* Bước 2, xếp các Node con dọc theo chiều trục chính lần lượt, sao cho cạnh phía gốc trục phụ, ví dụ trục phụ từ trái sang phải, thì xét cạnh phía trái, của Margin Box của các Node này thẳng hàng với nhau, và chỉ có 1 hàng duy nhất, có thu hẹp màn hình kiểu gì cũng 1 hàng
* Bước 3
* Trường hợp trục chính nằm ngang, các Node con thu hẹp tới chiều rộng tối thiểu, chiều rộng này được xác định như sau, lưu ý đây là chiều rộng của Content Box
* Thuộc tính CSS flex-basis của Node con sẽ ghi đè thuộc tính CSS width của nó
* Nếu Node con đã chỉ định chiều rộng, và chiều rộng này < chiều rộng của từ có ít kí tự nhất trong nó, thì đây là chiều rộng tối thiểu và cũng là chiều rộng tối đa
* Nếu Node con đã chỉ định chiều rộng = K, và chiều rộng này > chiều rộng của toàn bộ từ trong nó, thì chiều rộng tối thiểu = chiều rộng từ có nhiều kí tự nhất, và chiều rộng tối đa = K
* Nếu Node con không khai báo chiều rộng, chiều rộng tối thiểu = chiều rộng của từ có nhiều kí tự nhất và chiều rộng tối đa = dòng có nhiều kí tự nhất
* Mỗi Node con có thuộc tính CSS flex-shrink mặc định = 1, nếu nó dần tới 0, thì chiều rộng tối thiểu sẽ tăng dần tới chiều rộng tối đa, nghĩa là nếu = 0, thì chiều rộng tối thiểu = chiều rộng tối đa, > 1 thì chiều rộng tối thiểu không đổi
* Trường hợp trục chính nằm dọc, thì cũng như trên, thay từ “rộng” thành “cao”, “width” thành “height”, chỉ khác là chiều cao tối thiểu = chiều cao đủ đế chứa Full dòng, chiều cao tối đa = chiều cao đã chỉ định, nếu không chỉ định thì = chiều cao tối thiểu
* Bước 4, nếu flex-wrap khác wrap, thì nếu chiều rộng Node Flex không đủ chứa toàn bộ Node con khi chúng ở trạng thái tối đa kích thước, thì các Node sẽ xuống dòng, chiều xuống dòng = chiều trục phụ
* Bước 5, dãn các Node con, các Node con từ kích thước tối thiểu sẽ dãn tới kích thước tối đa, nếu không đủ chỗ thì ngừng dãn, cái nào mà chênh lệch giữa tối đa và tối thiểu lớn hơn thì dãn nhanh hơn, đồng thời cái nào có flex-shrink lớn hơn sẽ tăng chậm hơn
* Bước 6, các Node con mặc định có thuộc tính CSS flex-grow = 0
* Xét phần không gian còn lại của Node Flex sau khi các Node con dãn ra tối đa, cho nó = 1, flex-grow chính là tỉ lệ phần trăm không gian này mà Node con có thể dãn ra để chiếm nữa, ví dụ flex-grow = 0.3, thì sẽ dãn ra để chiếm 30% phần không gian còn lại, nếu tổng flex-grow của các Node con > 1, thì sẽ được chuẩn hóa
* Ta có thể viết chung 3 thuộc tính CSS flex-grow, flex-shrink, flex-basis của mỗi Node con trên 1 dòng bằng thuộc tính CSS flex
* Flex: <Flex Grow> <Flex Shrink> <Flex Basis>;
* Nếu chỉ chỉ định 2 giá trị
* Giá trị đầu tiên sẽ là <Flex Grow>
* Giá trị thứ 2 nếu là 1 số không có đơn, vị, thì nó sẽ là <Flex Shrink> và flex-basis tự động = 0, ngược lại là <Flex Basis>, và flex-shrink tự động = 1
* Nếu chỉ định 1 giá trị
* Nếu là số không có đơn vị, nó sẽ là <Flex Grow>, flex-shrink = 1, flex-basis = 0
* Ngược lại, nó sẽ là <Flex Basis>, flex-grow = flex-shrink = 1
* Ta cũng có thể viết chung 2 thuộc tính CSS flex-direction, flex-wrap trong 1 dòng

flex-flow: <Flex Direction> <Flex Wrap>;

* Ta có thể tạo Node Inline Flex, giống Inline Block + Flex, Base Line là Base Line của kí tự đầu tiên của Node con đầu tiên trong nó, nếu không có kí tự thì là cạnh dưới Padding Box của Node con đầu tiên

display: inline-flex;

1. Căn Chỉnh Trong Node Flex?

* Để tạo khoảng không giữa các Node con trong Node Flex theo phương trục chính

justify-content: <Cách Chia>;

* Ta có

|  |  |
| --- | --- |
| <Cách Chia> |  |
| space-between | Lấy hết phần không gian còn lại trong Node Flex, chia đều và đặt vào khe giữa mỗi 2 Node con kề nhau |
| space-around | Giống space-between, nhưng trước Node con đầu tiên và đằng sau Node con cuối cùng sẽ có khoảng không = nửa khoảng không giữa khe của 2 Node con liền kề |
| space-evenly | Giống space-around, nhưng khoảng không trước Node con đầu tiên và sau Node con cuối cùng = khoảng không giữa khe của 2 Node con liền kề |

* Ta có thể thay đổi thứ tự các Node con, ví dụ biến Node con đầu tiên thành Node con thứ 2, …

order: <Index>;

* Ví dụ để biến Node con thứ 3 thành Node con thứ 5

order: 5;

Mongo DB:

1. Mongo DB Dùng Để Làm Gì?

* Làm hệ quản trị cơ sở dữ liệu

1. Truy Cập Tài Khoản Mongo DB Cloud?

* Cái này dùng để lưu dữ liệu trên máy chủ của nó không cần lưu trên máy của mình
* <https://cloud.mongodb.com>
* Đăng nhập = tài khoản Google

1. Cách Sử Dụng Mongo DB Cloud?

* Dưới chữ “Atlas” ở góc trái trên là tên dự án hiện tại

1. Cách Tải Mongo DB Cho Windows 7 32 Bit?

* Cái này dùng để tạo cơ sở dữ liệu trên chính máy tính của bạn luôn, máy của bạn sẽ vừa là Server máy chủ, vừa là máy khách
* <https://fastdl.mongodb.org/win32/mongodb-win32-i386-3.3.22-signed.msi>
* Mở File vừa tải + nhấn Next liên tục nhớ đồng ý điều khoản + nhấn Complete + Click Install + chờ cài đặt + Click Finish + xóa File vừa tải
* Sau khi làm xong, trong thư mục “C:\Program Files” sẽ xuất hiện thư mục “MongoDB”
* Tạo thêm 1 thư mục ở 1 nơi nào đó, đây sẽ là nơi chứa cơ sở dữ liệu, ví dụ tạo thư mục “C:\data\db”
* Thêm vào biến hệ thống Path đường dẫn

“C:\Program Files\MongoDB\Server\3.2\bin”, chứa File lệnh “mongod.exe” và “mongo.exe”

1. Cách Gỡ Cài Đặt Hoàn Toàn Mongo DB Ra Khỏi Máy?

* Gỡ cài đặt

1. Cách Khởi Chạy Server Cục Bộ?

* Lúc mới cài, nhập câu lệnh sau vào CMD

mongod --storageEngine=mmapv1 --dbpath <Đường Dẫn Đến Cơ Sở Dữ Liệu>

* Port 27017 của Local Host, hay địa chỉ 127.0.0.1:27017 sẽ bắt đầu lắng nghe
* Ví dụ

mongod --storageEngine=mmapv1 --dbpath C:\data\db

* Kể từ sau trở đi, muốn khởi chạy lại Server 1 lần nữa chỉ cần nhập

mongod

* Vì nó đã lưu <Đường Dẫn Đến Cơ Sở Dữ Liệu>
* Nếu muốn thay đổi đường dẫn đến cơ sở dữ liệu, thì nhập như lúc mới cài

1. Cách Hiện Tên Tất Cả Cơ Sở Dữ Liệu Hiện Có?

* Sau khi khởi chạy Server cục bộ, nhập câu lệnh sau vào CMD

mongo

* Chờ kết nối, sau đó nhập

show dbs

* Nếu muốn coi tên bằng cách thủ công, thì mở thư mục chứa cơ sở dữ liệu, các File mà có phần mở rộng là “.0” chính là các cơ sở dữ liệu, tên của File cũng chính là tên của cơ sở dữ liệu

1. Cấu Trúc Cơ Sở Dữ Liệu Trong Mongo DB?

* Trong 1 dự án, sẽ có nhiều cơ sở dữ liệu, dữ liệu của mỗi cơ sở dữ liệu sẽ được lưu trong 1 File có phần mở rộng BSON
* Mỗi cơ sở dữ liệu lại chứa nhiều bảng
* Mỗi bảng chứa nhiều bản ghi
* Mỗi bản ghi bao gồm ID, tên, ngày tạo, thuộc tính khác, …

1. Show Tên Các Bảng Trong 1 Cơ Sở Dữ Liệu?

* Sau khi khởi chạy Server cục bộ, nhập câu lệnh sau vào CMD

mongo

* Chờ kết nối, sau đó nhập

use <Tên Cơ Sở Dữ Liệu>

show collections

SQL – Structured Query Language:

1. MySQL?

* Giống Mongo DB, là hệ quản trị cơ sở dữ liệu, dùng ngôn ngữ SQL

1. Cài Đặt MySQL?

* Vào Link
* <https://dev.mysql.com/downloads/mysql/>
* Click “Download” ứng với cái MSI Installer + Click “No thanks, just start my download.” + chờ tải + chạy File vừa tải + Click “Next” + đồng ý điều khoản + Click “Next” + Click “Typical” + Click “Install” + chờ cài đặt + Click “Finish” + Click “Next” 3 lần + tại mục “MySQL Root Password” điền mật khẩu đăng nhập, gọi là Root Password, khi này tên người dùng là root + tại mục “Repeat Password” điền lại mật khẩu + Click “Next” + Untick “Start the MySQL Server at System Startup” + Click “Next” 3 lần + Click “Execute” + Click “Next” + Click “Finish”
* Khi này trong thư mục “C:\Program Files” sẽ xuất hiện thư mục “MySQL”, trong đây chứa 1 thư mục có tên là phiên bản MySQL bạn đã cài đặt, và có thể chứa nhiều thư mục ứng với nhiều phiên bản khác nhau
* Đồng thời trong thư mục “C:\ProgramData” cũng sẽ xuất hiện thư mục “MySQL”, trong đây có chứa các thư mục với tên là các phiên bản MySQL, bên trong mỗi thư mục sẽ chứa các File và thư mục liên quan đến Database
* Trong thư mục “C:\ProgramData\Microsoft\Windows\Start Menu\Programs” cũng xuất hiện thư mục “MySQL”, bên trong là 1 hoặc nhiều thư mục có tên là các phiên bản bạn đã tải + bên trong mỗi thư mục này sẽ chứa File EXE “Command Line Client”, giúp bạn giao tiếp với MySQL
* Bạn cần thêm đường dẫn “C:\Program Files\MySQL\MySQL Server 8.3\bin” vào biến môi trường Path để có thể dùng lệnh thông qua CMD

1. Command Line Client (CLC)?

* Ban đầu, nó sẽ bắt bạn nhập mật khẩu, khi này bạn sẽ nhập Root Password, sau đó nó sẽ kết nối với Server MySQL đang chạy trên máy, nếu Server chưa chạy thì không có gì xảy ra
* Để bắt đầu chạy Server MySQL, vào ứng dụng Services (giống như khi vào Settings) + cuộn xuống cho đến khi thấy tên MySQL + chọn nó + Click “Start the service” để chạy hoặc “Stop the service” để đóng Server, Server sẽ được chạy ở Port 3306
* Sau khi CLC đã kết nối với Server MySQL, bạn có thể bắt đầu sử dụng lệnh của ngôn ngữ SQL

1. PopSQL?

* Là giao diện thân thiện với người dùng, dùng để giao tiếp với Server MySQL chạy trên máy bạn, ngon hơn CLC

1. Cài Đặt PopSQL?

* Vào Link
* <https://popsql.com/download>
* Click “Download for Windows” + chờ tải + chạy File vừa tải + thoát
* Khi này, trong thư mục

“C:\Users\<Tên Người Dùng Hiện Tại>\AppData\Local\Programs” sẽ xuất hiện thư mục “@popsqldesktop”, trong đây lại chứa File EXE “PopSQL” là phần mềm bạn muốn chạy

* Sau đó, chạy phần mềm PopSQL lên, đăng nhập bằng tài khoản Google + nó chuyển hướng đến trang đăng ký + điền thông tin + xác nhận Email + mở lại phần mềm PopSQL rồi đăng nhập bằng tài khoản Google hồi nãy + điền tên công ty + điền số người trong công ty + Click “Create organization” + Click “Skip this” + chọn 1 vài thông tin cá nhấn + Click “Next” + chọn mục tiêu làm việc hôm nay + Click “Next” + Click “Start querying” + đóng phần mềm
* Như vậy là xong bước cài đặt, từ giờ chỉ cần mở phần mềm và làm việc

1. Cấu Trúc Database Trong MySQL?

* 1 Port sẽ chứa nhiều Database, mỗi Database là 1 thư mục, và mỗi cái lại chứa nhiều bảng, mỗi bảng là 1 File có phần mở rộng là “.ibd”
* Nếu bạn lưu Database trong máy, ứng với localhost và Port 3306, thì các Database ứng với Port này được lưu trong thư mục “C:\ProgramData\MySQL\<Phiên Bản MySQL> \Data”

1. Cách Dùng PopSQL?

* Cửa sổ PopSQL bao gồm thanh tác vụ trên cùng và thanh công cụ bên trái, khi Click vào 1 biểu tượng ở thanh công cụ thì cửa sổ tương ứng với công cụ đó sẽ hiện trồi ra hoặc rút vào
* Quá tình làm việc với PopSQL sẽ giống như chỉ làm trên 1 dự án duy nhất, bạn không cần nhấn lưu File hay gì hết, tất cả tự động lưu và mỗi khi bạn mở PopSQL lên thì sẽ tiếp tục với phần việc dở dang trước đó
* Phần 1, cài đặt
* Để vào cài đặt
* Vào Tab “File” + chọn “Preferences…” = nhấn “Ctrl” + “,”
* Để thoát cài đặt, Click “Preferences” to đùng
* Phần 2, danh sách có thể kết nối, nằm trong phần cài đặt
* Để vào danh sách có thể kết nối, vào thẻ “Connections”
* Để thêm 1 Port vào danh sách có thể kết nối, lưu ý bản PopSQL Free chỉ được phép có tối đa 2 Port trong danh sách
* Click “New connection” + chọn “MySQL” + tại mục “Connection name” điền tên sẽ hiển thị ở danh sách có thể kết nối, tạm gọi là tên kết nối + tại mục “Hostname/Port” lần lượt điền tên miền và Port lưu trữ các Database, nếu cục bộ thì điền localhost và Port 3306 + tại mục “Database” điền tên Database muốn kết nối + tại mục “Username/Password” điền tên người dùng và mật khẩu, tên người dùng và mật khẩu này phải có quyền truy cập vào các Database tại Port, ví dụ root và matkhau123 + tại mục “Connection Type” gạt sang phải để chỉ định Port nằm trên máy mình + Click “Save”
* Phần 3, bố cục, vào Tab “Schemas” ở thanh công cụ
* Ở đây sẽ hiển thị bố cục tổng thể của 1 Database nào đó
* Bước 1, chọn tên kết nối ứng với Port chứa Database muốn hiện bố cục
* Bước 2, chọn Database muốn hiện bố cục
* Bước 3, xem bố cục ở bên dưới
* Bạn có thể đồng bộ lại cấu trúc Database bằng cách Click biểu tượng 2 mũi tên tròn bên phải tên kết nối, khi này PopSQL sẽ cập nhật các File và thư mục trong Port tương ứng vào trong PopSQL
* Phần 4, mẫu hỏi, nằm ở Tab “Queries” ở thanh công cụ
* Mẫu hỏi tương đương 1 File, chạy File này sẽ trích xuất dữ liệu trong Database nào đó và hiển thị lên màn hình PopSQL, phần bên trái là Code, bên phải là màn hình thông báo kết quả chạy lệnh
* Dự án bao gồm 3 thư mục to nhất là “Team queries”, các File nằm ở đây sẽ được dùng chung bởi tất cả thành viên trong công ty, “My queries”, chỉ có người dùng hiện tại mới truy cập được, “Drafts”, các bản nháp
* Để tạo 1 File mẫu hỏi trong 1 thư mục, Click dấu cộng bên phải thư mục đó + chọn “New query” để tạo 1 File mẫu hỏi, hoặc chọn “New folder” để tạo 1 thư mục con + đặt tên File hoặc thư mục
* Để chỉnh sửa 1 File mẫu hỏi, Click vào File đó, khi này Tab ứng với nó sẽ được chèn vào thanh trên cùng, Click dấu x trên Tab để đóng File tương ứng
* Để chạy 1 File mẫu hỏi trên 1 Database nào đó
* Bước 1, vào Tab của nó
* Bước 2, chọn tên kết nối ứng với Port chứa Database đích, nằm ở góc phải trên
* Bước 3, chọn Database đích trong Port này, nằm ở góc phải trên
* Bước 4, Click mũi tên chỉ xuống bên phải nút “Run” + Tick “Run all statements” nếu muốn chạy nguyên File mẫu hỏi, Untick nó để chỉ chạy câu lệnh ở vị trí con trỏ văn bản hoặc nguyên đoạn Code đang được bôi đen
* Bước 5, Click nút “Run”, Database sẽ bị thay đổi trực tiếp
* Phần 5, tạo File Database thủ công
* Không được tạo hay thay đổi File Database thủ công, khi này bạn sẽ không được phép chạy Server MySQL, tất cả phải thông qua lệnh

1. Cú Pháp SQL?

* Không phân biệt hoa thường
* Kết thúc lệnh bằng chấm phẩy
* String được đặt trong cặp dấu nháy đơn hoặc kép

1. Tạo 1 Database Mới?

create database <Tên Database>;

* Khi này trong Port sẽ xuất hiện thư mục rỗng có tên là <Tên Database> nếu chưa tồn tại, và báo lỗi nếu đã tồn tại
* Ví dụ

create database foo;

1. Các Kiểu Dữ Liệu?

varchar(<Số Kí Tự Tối Đa>)

* <Số Kí Tự Tối Đa> phải là số nguyên trong đoạn từ 0 đến 16383
* Giá trị hợp lệ là String với số kí tự không vượt quá <Số Kí Tự Tối Đa>

int

* Giá trị hợp lệ là số nguyên có dấu 32 Bit, nằm trong đoạn từ –2147483648 tới 2147483647
* Giá trị mặc định cho 1 ô trong bảng là null

1. Tạo 1 Bảng Rỗng?

create table <Tên Bảng>(<Các Cột>);

* <Các Cột> ngăn cách nhau bởi dấu phẩy, mỗi cột sẽ có cú pháp

<Tên Cột> <Kiểu Dữ Liệu> <Hiệu Ứng>

* <Hiệu Ứng> có thể có hoặc không
* Đồng thời ta có thể chỉnh định các cột nào sẽ làm khóa chính bằng cách đặt lệnh sau vào bất kì chỗ nào trong <Các Cột>

primary key(<Tên Các Cột Làm Khóa Chính>)

* Các cột làm khóa chính không được phép chứa dù chỉ 2 tổ hợp giá trị giống nhau, ví dụ cột A và B làm khóa chính, thì không được phép có 2 bản ghi mà thuộc tính A của chúng đều = 2 và B của chúng đều = 5, nhưng sẽ hợp lệ khi 1 thằng có A = 2, B = 5, 1 thằng có A = 3, B = 5
* Các cột làm khóa chính sẽ phân ra khóa cấp 1, cấp 2, … dựa theo thứ tự ghi trong <Tên Các Cột Làm Khóa Chính>
* Mỗi khi chèn 1 bản ghi vào bảng, thì nó sẽ được đặt ở vị trí sao cho khóa cấp 1 tăng dần từ trên xuống, nếu 2 giá trị bằng nhau, thì sắp xếp tăng dần theo khóa cấp 2, nếu 2 giá trị trong khóa cấp 2 bằng nhau, thì tiếp tục xét đến khóa cấp 3, …
* Ví dụ

primary key(bar, alice, john)

* Thì bar là khóa cấp 1, alice cấp 2, john cấp 3
* Bạn cũng có thể đặt cụm từ primary key ở cột làm khóa chính, khi này chỉ được phép đặt ở 1 cột, nghĩa là khóa chính chỉ gồm 1 cột
* Danh sách <Hiệu Ứng>

|  |  |
| --- | --- |
| not null | Báo lỗi khi chèn thêm 1 bản ghi mà thiếu giá trị của cột này  Nếu bạn đã tạo bảng rồi, và nó có vài bản ghi rồi, mà bạn lại tạo cột mới có hiệu ứng này, thì giá trị cột này thay vì là null sẽ = 0 nếu kiểu số nguyên và = "" nếu là String |
| unique | Báo lỗi khi chèn thêm 1 bản ghi mà giá trị của cột này bị lặp  Cho phép giá trị null có thể bị lặp |
| default <Giá Trị> | Khi chèn thêm 1 bản ghi mà không chỉ định giá trị của cột này thì mặc định nó có giá trị là <Giá Trị> |
| auto\_increment | Cột có hiệu ứng này phải thuộc khóa chính, và là khóa cấp 1, có kiểu dữ liệu số nguyên  Khi chèn thêm 1 bản ghi mới mà không có chỉ định giá trị cho cột này, thì giá trị của nó tự động = giá trị lớn nhất hiện tại của cột này + 1, nếu đang là giá trị tối đa của kiểu dữ liệu tương ứng thì không tăng  Cho 1 bảng đã có vài bản ghi, nếu bạn chèn thêm cột có hiệu ứng này vào thì cột đó sẽ tự động đánh số tăng dần từ 1 |

* Khi này trong thư mục Database hiện tại sẽ xuất hiện File có tên là

“<Tên Bảng>.ibd” nếu chưa tồn tại, và báo lỗi nếu đã tồn tại

* Ví dụ tạo bảng foo, có 3 cột là bar, alice và john, trong đó bar và alice làm khóa chính, bar là cột số thứ tự, john là cột có giá trị mặc định 4

create table foo(

bar int auto\_increment,

primary key(bar, alice),

alice varchar(100),

john int default 4

);

* Ví dụ khóa chính chỉ gồm 1 cột là bar

create table foo(

bar int auto\_increment primary key,

alice varchar(100),

john int default 4

);

* Để hiện thị thông tin về toàn bộ cột của bảng, như tên cột, kiểu dữ liệu, giá trị mặc định, …

describe <Tên Bảng>;

* Để xóa 1 bảng ra khỏi thư mục Database

drop table <Tên Bảng>;

* Nếu không tồn tại bảng để xóa thì báo lỗi

1. Thêm 1 Cột Mới Vào Bảng?

alter table <Tên Bảng> add <Thông Tin Cột Mới>;

* Bản chất là đặt nguyên cái <Thông Tin Cột Mới> vào cuối phần cấu hình khi bạn create table
* Ví dụ

create table foo(

alice int,

john varchar(10),

primary key(alice)

);

alter table foo add bar int not null;

* Tương đương

create table foo(

alice int,

john varchar(10),

primary key(alice),

bar int not null

);

1. Xóa 1 Cột Trong Bảng?

alter table <Tên Bảng> drop column <Tên Cột Muốn Xóa>;

* Ví dụ

alter table foo drop column bar;

1. Thêm 1 Bản Ghi Vào Bảng?

insert into <Tên Bảng> values(<Tất Cả Các Giá Trị Của Các Cột>);

* Ví dụ

insert into foo values(20, "daubuoi", 4);

* Để thêm 1 bản ghi mới nhưng chỉ xác định giá trị cho 1 số cột thay vì toàn bộ

insert into <Tên Bảng>(<Tên Các Cột>) values(<Giá Trị Các Cột Tương Ứng>);

* Ví dụ

insert into foo(bar, alice) values(4, "daubuoi");

1. Trích Xuất Các Cột Và Bản Ghi Trong Bảng Thỏa Mãn Điều Kiện Gì Đó Rồi In Ra Màn Hình?

* Để trích xuất 1 số cột

select <Tên Các Cột> from <Tên Bảng>;

* Các cột sẽ sắp đúng thứ tự trong <Tên Các Cột>
* Ví dụ

select alice, bar from foo;

* Nếu <Tên Các Cột> là \*, thì tương đương trích xuất tất cả cột, hay toàn bộ bảng
* Ví dụ

select \* from foo;

* Bạn cũng có thể đổi tên cột khi trích xuất ra bằng cú pháp

<Tên Cột Cũ> as <Tên Cột Mới>

* Ví dụ

select alice as kenny, bar as bob from foo;

* Để trích xuất 1 số bản ghi từ trích xuất cột

select <Tên Các Cột> from <Tên Bảng> where <Điều Kiện>;

* Ví dụ

select bar, alice from foo where john = 5 and bob <> 9;

* Nghia là trích xuất cột bar và alice trong bảng foo, từ 2 cột này, trích những hàng mà có ô john = 5 và ô bob khác 9
* Để trích xuất sau đó sắp xếp các cột theo thứ tự nào đó

select <Tên Các Cột> from <Tên Bảng> where <Điều Kiện>

order by <Chiều Sắp Xếp Các Cột>;

* <Chiều Sắp Xếp Các Cột> là 1 dãy các phần tử có cú pháp sau

<Tên Cột> <Tăng Hay Giảm Dần>

* <Tăng Hay Giảm Dần> chỉ có thể là asc hoặc desc, asc là tăng dần, desc là giảm dần
* Nếu không chỉ định <Tăng Hay Giảm Dần> thì nó mặc định là asc
* Thứ tự ưu tiên sắp xếp theo đúng thứ tự cột ghi trong <Chiều Sắp Xếp Các Cột>
* Ví dụ

select bar, alice from foo order by john desc, alice;

* Nghĩa là sắp xếp lại bảng foo thành 1 bảng sao cho cột john của nó giảm dần, nếu 2 giá trị bằng nhau thì xét đến cột alice tăng dần, sau đó trích xuất cột bar và alice từ bảng đã sắp xếp này
* Để chỉ lấy N cột trích xuất đầu tiên

select <Tên Các Cột> from <Tên Bảng> where <Điều Kiện>

order by <Chiều Sắp Xếp Các Cột> limit <N>;

* Ví dụ

select bar, alice from foo order by alice limit 10;

* Tức là trích xuất cột bar và alice từ bảng foo, sau đó sắp xếp tăng dần theo alice rồi cắt lấy 10 bản ghi đầu tiên
* Để kết quả trích xuất không có 2 bản ghi nào có tổ hợp giá trị hoàn toàn giống nhau thì thêm distinct vào trước <Tên Các Cột>
* Ví dụ

select distinct bar, alice, john from foo;

* Nghĩa là giả sử trích xuất được các tổ hợp giá trị là (4, 5, 6), (4, 5, 6), (4, 5, 7) thì sẽ chỉ còn (4, 5, 6), (4, 5, 7)

1. Cập Nhật Lại Giá Trị Cho 1 Số Ô Thỏa Mãn Điều Kiện Nào Đó Trong Bảng?

update <Tên Bảng> set <Các Lệnh Gán Lại Giá Trị> where <Điều Kiện>;

* ví dụ

update foo set bar = 4, alice = "haha" where john = 6 or bob > 8;

* Lệnh trên nghĩa là xét trong bảng foo, hàng nào có ô ứng với cột john có giá trị = 6 hoặc ô ứng với cột bob có giá trị > 8 thì ô ứng với cột bar của hàng đó sẽ bị chuyển thành 4 và ô ứng với cột alice sẽ bị chuyển thành "haha"
* Nếu không có vế where, thì tương đương tất cả các ô trong cột sẽ bị thay đổi giá trị
* Ví dụ

update foo set bar = 4, alice = "haha";

* Tất cả các ô thuộc cột bar sẽ có giá trị = 4 và các ô thuộc cột alice sẽ có giá trị là "haha"

1. Xóa 1 Số Bản Ghi Thỏa Mãn Điều Kiện Nào Đó Trong Bảng?

delete from <Tên Bảng> where <Điều Kiện>;

* Ví dụ

delete from foo where bar = 6 and alice >= 8;

* Nghĩa là xóa tất cả bản ghi trong bảng foo nếu bản ghi đó có ô ứng với cột bar mang giá trị 6 và ô ứng với cột alice mang giá trị >= 8
* Nếu không có vế where thì xóa hết mẹ bản ghi trong bảng
* Ví dụ

delete from foo;

1. Toán Tử?

|  |  |
| --- | --- |
| = | Bằng |
| <> | Khác |
| >= | Lớn hơn hoặc bằng |
| A in (B, C, D, …) | Tương đương biểu thức  A = B or A = C or A = D or … |
| A like B | Tương đương biểu thức  A = B  Trong đó B phải là 1 String Regex, A tự động được ép kiểu sang String |
| A in (B) | B là trích xuất gồm duy nhất 1 cột  Trả về True nếu chỉ cần A có giá trị = 1 ô trong B, nếu không thì False  Ví dụ  foo in (select bar from bob) |
| A = (B) | B là trích xuất gồm duy nhất 1 ô  Trả về True nếu A có giá trị = ô này, nếu không thì False  Ví dụ  foo = (select count(bar) from bob) |

* Regex

|  |  |
| --- | --- |
| % | Đại diện cho chuỗi kí tự hoặc số bất kì với độ dài bất kì |
| \_ | Đại diện cho 1 kí tự bất kì |
|  |  |
|  |  |

1. Thống Kê?

* Để trích xuất ra vài cột rồi áp dụng từng hàm lên cột

select <Các Hàm Cột> from <Tên Bảng>;

* <Các Hàm Cột> là 1 dãy các cú pháp sau, ngăn cách bởi dấu phẩy

<Hàm>(<Cột>)

* Các hàm

|  |  |
| --- | --- |
| count | Đếm số phần tử khác null trong cột |
| avg | Tính giá trị trung bình của các phần tử khác null trong cột |
| sum | Tính tổng các phần tử khác null trong cột |

* Kết quả trả về sẽ là 1 bảng gồm duy nhất 1 hàng chứa kết quả tính
* Ví dụ

select avg(bar), sum(alice) from foo;

* Nghĩa là trả về 1 bảng gồm 1 hàng, hàng này gồm 2 ô, ô đầu tiên chứa giá trị trung bình của cột bar và ô thứ 2 chứa tổng các giá trị trong cột alice
* Để chia nhóm xong mới áp hàm lên từng nhóm

select <Các Hàm Cột Tính> from <Tên Bảng> group by <Tên Các Cột Nhóm>;

* Trước tiên các bản ghi trong bảng sẽ được sắp xếp lại theo tổ hợp cột nhóm, sao cho tổ hợp giá trị ở các cột nhóm bằng nhau thì xếp sát nhau, bản ghi tạo ra sau thì nằm dưới
* Ví dụ
* Bảng ban đầu

|  |  |  |
| --- | --- | --- |
| bar | alice | john |
| 1 | 2 | 10 |
| 8 | 4 | 2 |
| 1 | 2 | 3 |
| 1 | 2 | 4 |
| 5 | 2 | 5 |
| 8 | 4 | 6 |
| 5 | 2 | 7 |
| 8 | 9 | 88 |

* Bảng sau khi sắp theo tổ hợp cột nhóm là cột bar và cột alice

|  |  |  |
| --- | --- | --- |
| bar | alice | john |
| 1 | 2 | 10 |
| 1 | 2 | 3 |
| 1 | 2 | 4 |
| 8 | 4 | 2 |
| 8 | 4 | 6 |
| 5 | 2 | 5 |
| 5 | 2 | 7 |
| 8 | 9 | 88 |

* Mỗi màu là 1 nhóm, sau đó áp dụng hàm lên từng nhóm và rút gọn, giả sử ta dùng hàm sum, cột tính là john
* Kết quả

|  |  |  |
| --- | --- | --- |
| bar | alice | john |
| 1 | 2 | 17 |
| 8 | 4 | 8 |
| 5 | 2 | 12 |
| 8 | 9 | 88 |

* Ở <Các Hàm Cột Tính>, ta có thể chèn thêm <Tên Các Cột Nhóm> để hiển thị cả các cột nhóm, như ở ví dụ trên ta hiển thị luôn cả cột bar và cột alice trong phần kết quả
* Ví dụ

select bar, alice, sum(john), avg(bob) from foo group by bar, alice;

1. Hợp Nhiều Trích Xuất?

* Ví dụ ta trích xuất được bảng A, và trích xuất được bảng B, khi này ta muốn hợp A và B lại theo chiều dọc

<Trích Xuất A> union <Trích Xuất B>;

* <Trích Xuất A> và <Trích Xuất B> phải có cùng số cột và các cột tương ứng phải cùng kiểu dữ liệu, tên cột sau khi hợp nhất sẽ lấy theo tên cột của <Trích xuất A>
* Cơ chế hợp như sau
* Đầu tiên <Trích xuất A> sẽ tự nhóm các bản ghi của nó lại với nhau, cơ chế nhóm tương tự group by với các cột nhóm là toàn bộ nhóm của <Trích Xuất A>, được C
* <Trích Xuất B> cũng tự nhóm như <Trích Xuất A>, được D
* Sau đó lặp qua tất cả bản ghi của D, nếu bản ghi đó có tổ hợp giá trị chưa có trong C thì chèn vào dưới C
* Cuối cùng trả về C
* Ta có thể nhóm nhiều trích xuất cùng lúc
* Ví dụ

select bar, alice from foo where alice = 5

union

select john, bob from fool

union

select john, shit from fool where john > 4;

JS:

1. JIT (Just In Time Compiler)?

* Là trình biên dịch, sao cho nó biên dịch lệnh nào trong File nguồn ra Byte Code là chạy Byte Code đó luôn, nghĩa là thay vì biên dịch nguyên File ra 1 File Byte Code, rồi mới chạy File Byte Code này, thì nó sẽ không tạo ra File Byte Code nào hết
* JS có thể chạy trực tiếp trên trình duyệt, do trình duyệt đã tích hợp sẵn JIT để biên dịch và chạy thẳng JS, tuy nhiên, nó sẽ có 1 bước là quét Full mã để thực hiện Hoisting, rồi mới chạy
* Nếu muốn chạy File JS ở ngoài trình biên dịch, thì cần sử dụng Node JS, vì nó đã tích hợp sẵn JS Engine

1. Cú Pháp?

* Viết hoa khác viết thường
* Có hoặc không có dấu chấm phẩy cuối lệnh không quan trọng

1. Đường Dẫn?

* Nếu là đường dẫn đến File JS thì không cần phần mở rộng “.js”

1. Dấu Nháy?

* Dùng để bọc String, dùng dấu nháy đơn hay kép đều được, nhưng nếu muốn String chứa dấu nháy đơn, thì bọc trong dấu nháy kép và ngược lại, hoặc dùng kí tự thoát

1. Cách Mượn Hàm, Hằng, … Của File Khác?

* Ở File chứa thứ cần mượn thì thêm dòng Code này dưới cùng

module.exports = <Bất Kể Thứ Gì>

* <Bất Kể Thứ Gì> có thể là mảng, hằng, hàm, …
* Ví dụ

module.exports = ham\_tinh\_tong

* Ở File mượn thì nếu muốn sử dụng <Bất Kể Thứ Gì> do File chứa thứ cần mượn Export ra, thì sử dụng hàm

require(<Đường Dẫn Đến File Chứa Thứ Cần Mượn>)

* Hàm này sẽ trả về <Bất Kể Thứ Gì>
* Ví dụ

const ham\_tinh\_tong\_doi\_ten = require(“./file\_chua\_ham”)

* Chú ý nếu sử dụng cú pháp sau

require(<Tên Thư Viện>)

* Ví dụ

require(“express”)

* Thì máy sẽ vào thư mục có tên “node\_modules”, tìm thư mục có tên

<Tên Thư Viện> và trả về thứ được Export ra trong File “index.js” ở thư mục này

1. Trả Về Kiểu Dữ Liệu Của 1 Giá Trị?

typeof <Giá Trị>

* Ví dụ

foo = typeof bar

* ta có

|  |  |
| --- | --- |
| bar | 15 |
| foo | "number" |

1. Đổi Kiểu Dữ Liệu Từ Number Sang String?

String(<Number>)

1. Interrupt?

* Khi mã JS đang chạy dưới nền 1 trang Web, bạn không thể giao tiếp với trang Web, nghĩa là nó giống bị treo
* Mỗi Tab Web sẽ có 1 luồng JS riêng, và có 1 Interupt Stack để lưu các hàm Interrupt đang chờ để chạy
* Khi 1 Interrupt được gọi, thì luồng JS sẽ tạm ngừng để chạy nó, và 1 lúc chỉ có 1 Interrupt được chạy
* Các hàm thông báo sau đều là Interrupt

alert(<Giá Trị>)

* Lệnh trên sẽ xuất ra 1 cửa sổ nhỏ phía trên màn hình, chứa <Giá Trị>, và nút “OK”, nhấn nút này để trở về luồng thực thi hiện tại

confirm(<Giá Trị>)

* Giống alert, nhưng có thêm nút “Cancel”

prompt(<Giá Trị>)

* Giống confirm, nhưng có thêm ô để bạn nhập văn bản vào

1. In Ra Console?

console.log(<Các Giá Trị>)

* Giống y chang hàm print trong Python

1. Comment?

* Giống C++

1. Khai Báo Biến?

* Có 3 loại biến, biến toàn cục, biến cục bộ và hằng
* Cả 3 loại trên khi khai báo đều được Hoisting khi biên dịch, nghĩa là nó sẽ tách phần khai báo và phần gán giá trị khởi đầu ra, và đưa phần khai báo lên đầu Scope tương ứng
* Với biến toàn cục, phần khai báo sẽ được đưa lên đầu chương trình, nếu nó đặt ngoài hàm, nếu đặt trong hàm, thì nó sẽ được đưa lên đầu hàm, và gán giá trị undefined cho nó
* Với biến cục bộ, phần khai báo sẽ đưa lên đầu Scope chứa nó, nghĩa là ngay sau dấu {, đồng thời gán giá trị lỗi cho nó, nghĩa là khi truy xuất đến nó mà chưa có giá trị khởi tạo, chương trình sẽ lỗi
* Với hằng, tương tự biến cục bộ
* Ví dụ

{

var foo = 5

console.log(bar)

let bar = 7

}

* Tương đương

foo = undefined

{

bar = error

foo = 5

console.log(bar) // khúc này lỗi

bar = 7

}

1. Khai Báo Hàm?

* Khai báo hàm cũng được Hoisiting, nghĩa là sẽ tách riêng hàm nguyên mẫu và phần thực thi hàm ra, đưa hàm nguyên mẫu lên đầu chương trình
* Để tạo 1 hàm

function <Tên Hàm>(<Các Tham Số>){

<Làm Gì Đó>

}

* Ví dụ

function foo(bar, alice){

console.log(bar + alice)

}

* Để tạo 1 hàm mũi tên, giống hàm Lambda trong Python

(<Các Tham Số>) => {

<Làm Gì Đó>

}

* Nếu <Làm Gì Đó> chỉ có 1 lệnh, thì có thể bỏ dấu ngoặc nhọn, nếu lệnh này là return, thì có thể bỏ từ khóa return, tương tự <Các Tham Số> nếu chỉ có 1 tham số, có thể bỏ dấu ngoặc tròn
* Ví dụ

let foo = x => x + 2

let bar = foo(4)

* Ta có

|  |  |
| --- | --- |
| bar | 6 |

1. Tạo Interrupt Sau 1 Khoảng Thời Gian?

setTimeout(<Hàm Interrupt>, <Số Mili Giây>)

* <Hàm Interrupt> sẽ được chạy sau <Số Mi Giây> mili giây kể từ lệnh này, các lệnh tiếp theo trong thời gian chờ vẫn chạy bình thường, khi hết thời gian chờ, nếu có 1 hàm Interrupt khác đang chạy, thì <Hàm Interrupt> sẽ được đặt vào Interrupt Stack, cho đến khi hàm Interrupt kia chạy xong thì các Interrupt trong Interrupt Stack sẽ được ưu tiên chạy
* Ví dụ

setTimeout(() => console.log(4), 3000)

* Để gọi <Hàm Interrupt> sau mỗi vài giây

setInterval(<Hàm Interrupt>, <Chu Kỳ Mili Giây>)

* ví dụ

setInterval(() => console.log(4), 3000)

1. Toán Tử?

* Giống C++, chỉ thêm toán tử chia lấy dư là % và lũy thừa là \*\*

1. Câu Điều Kiện?

* Giống C++

1. Object?

* Để tạo 1 Object, tạo giống như khi tạo Dictionary ở Python, tuy nhiên Key phải là tên giống tên biến
* Ví dụ

let foo = {

bar: 4,

foo: 5,

alice: "ggnore",

haha: x => x + 2

}

* Để truy cập 1 thuộc tính hoặc phương thức, dùng chấm chấm như C++, hoặc

<Tên Object>[<Thuộc Tính>]

* Ví dụ

foo["bar"] = 9

Sandbox:

1. Sandbox?

* Là nơi thực thi Code mà không sợ ảnh hưởng tới môi trường xung quanh, nghĩa là mã độc sẽ vô tác dụng

1. WASM (Web Assembly)?

* Giả sử bạn muốn chạy Code C++ trên trình duyệt, thì khi này, đoạn Code bạn nhập vào sẽ được gửi đến máy chủ, sau đó nó biên dịch thành mã WASM và được lại trình duyệt của bạn để chạy, nghĩa là nó khá giống JS, nhưng ngôn ngữ bậc thấp hơn

Development – Phát Triển:

1. Khởi Tạo Web Server Cục Bộ?

* Tạo 1 thư mục, đặt tên tùy ý, đây là thư mục dự án, mở VS Code lên tại thư mục này, tạo 1 thư mục mới, đặt tên tùy ý, đây là thư mục chứa phần Back End, chuyển Terminal về thư mục này
* Nhập dòng lệnh sau vào Terminal

npm install express

* Câu lệnh này sẽ tải thư viện Express có chức năng tạo Web Server, đồng thời cũng tạo thư mục “node\_modules” nếu chưa có để chứa các thư viện và File “package–lock.json” chứa thông tin phụ thuộc của các thư viện
* Tạo 1 File JS, đặt tên tùy ý, rồi nhập đoạn Code sau

const app = require('express')()

app.listen(<Số Port>)

app.get('/', (req, res) => res.send('Hello World!'))

* Rồi chạy File này sử dụng Node
* Vào trình duyệt Web nhập “localhost:<Số Port>” để truy cập vào trang Web vừa tạo
* Terminal sẽ duy trì trang Web này, nếu đóng thì xóa Server
* Web Server này 1 khi đã chạy là cố định, có thay đổi File nào đi nữa thì phải chạy lại mới thấy sự thay đổi

1. Cách Tự Động Khởi Chạy Lại Web Server Cục Bộ Khi Thay Đổi File?

* Nhập dòng lệnh sau vào Terminal

npm init

* Terminal sẽ hỏi bạn, cứ trả lời theo ý thích
* Sau khi trả lời xong, 1 File “package.json” sẽ xuất hiện, File này có tác dụng tạo các câu lệnh ngắn, lưu trữ thông tin dự án, xác định thư viện nào sẽ được sử dụng và File nào sẽ được chạy đầu tiên khi triển khai lên Server thật sự, File đầu tiên được chạy được xác định bởi “main”
* Nhập tiếp

npm install nodemon ––save–dev

* Câu lệnh này sẽ tải thư viện Nodemon có tác dụng tự khởi chạy lại Web Server cục bộ khi lưu thay đổi File, đưa nó vào thư mục “node\_modules” và “devDependencies” của “package.json” để khi triển khai dự án thì thư viện này không được sử dụng
* Trong thư mục “.bin” ở “node\_modules” cũng xuất hiện thêm nhiều File mới, trong đó có “nodemon.cmd”
* Sử dụng lệnh nodemon vừa tải để khởi chạy Web Server

nodemon .\node\_modules\.bin\nodemon <Tên File JS>

* Câu lệnh này cơ bản giống với

node <Tên File JS>

* Nhưng mỗi khi có bất kì sự thay đổi nào trong thư mục dự án thì sẽ tự động khởi chạy lại Web Server

1. Cách Thu Ngắn Các Câu Lệnh Terminal?

* Vào File “package.json”, thêm 1 ánh xạ tên lệnh với dòng lệnh thực sự ở thẻ “scripts”
* Ví dụ

“foo”: “nodemon bar.js”

* Muốn chạy câu lệnh “nodemon bar.js” thì bây giờ chỉ cần nhập

npm foo

* Khi nhập câu lệnh này, npm sẽ rà soát xem “foo” có phải là 1 câu lệnh thuộc về npm hay không, nếu không, nó sẽ vào tệp “package.json” cùng cấp với CMD hiện tại, tìm thẻ “scripts”, trong thẻ “scripts”, nó sẽ tìm ánh xạ nào tên “foo”, nếu không thì nó sẽ báo lỗi, nếu có thì nó sẽ chạy câu lệnh ứng với “foo”, nếu câu lệnh này không tồn tại, nó sẽ vào thư mục “.bin” của “node\_modules” để tìm, nếu có thì nó sẽ chạy lệnh này, nếu không thì nó sẽ báo lỗi

1. Xử Lí Khi Máy Khách Gửi Yêu Cầu Với Phương Thức Get Đến 1 Tuyến Đường?

app.get(<Tuyến Đường>, (req, res) => <Code Xử Lí>)

* Câu lệnh này sẽ làm cho mỗi khi máy khách truy cập vào <Tuyến Đường> thì sẽ được nhận được HTML, CSS, … do <Code Xử Lí> trả về
* Ví dụ
* Bạn muốn mỗi khi máy khách nhập lên thanh tìm kiếm tên miền trang Web của bạn + tuyến đường “/foo” sẽ nhận lại giao diện có chữ “Hello Bar”

app.get(‘/foo’, (req, res) => res.send(‘Hello Bar’))

1. Cách Kết Nối Với Server Cục Bộ Hoặc Cloud Mongo DB?

* Nhập dòng lệnh sau vào Terminal

npm install mongoose@6.0.15

* Câu lệnh này sẽ tải thư viện Mongoose, có tác dụng kết nối với Mongo DB và thực hiện các tác vụ thêm, sửa, xóa, … các bản ghi trong cơ sở dữ liệu
* Để kết nối, dùng đoạn Code sau

const mongoose = require(‘mongoose’)

mongoose.connect(

<URI Của Cơ Sở Dữ Liệu>,

{useNewUrlParser: true, useUnifiedTopology: true}

)

.then(() => <Code Xử Lí Khi Kết Nối Thành Công>)

.catch((error) => <Code Xử Lí Khi Nhận Được Lỗi>)

* useNewUrlParser: true và useUnifiedTopology: true dùng để chỉ định sử dụng trình phân tích URL mới khi kết nối với cơ sở dữ liệu và phương thức kết nối mới, nói là mới vì nó chỉ áp dụng từ phiên bản Mongo DB 3.1 trở đi
* <URI Của Cơ Sở Dữ Liệu> được bắt đầu bằng “mongodb”, sau đó là địa chỉ Server cùng với số Port và cuối cùng là tên của cơ sở dữ liệu nằm trong Server, ví dụ

‘mongodb://localhost:27017/local’

1. Cách Thêm, Sửa, Xóa 1 Bản Ghi Vào Cơ Sở Dữ Liệu?

* Sau khi tải Mongoose và kết nối với cơ sở dữ liệu, nhập

const <Biến Bảng> = mongoose.model(<Tên Bảng>, <Cấu Trúc Bảng>)

* Câu lệnh này sẽ tạo ra 1 bảng để làm chuẩn cho chúng ta tạo bản ghi
* <Biến Bảng> tên là gì cũng được
* <Tên Bảng> là tên bảng trong cơ sở dữ liệu
* <Cấu Trúc Bảng> quy định các trường và kiểu dữ liệu mỗi trường trong bảng
* Ví dụ

const schema = new mongoose.Schema({

name: String,

email: String,

age: Number

})

const foo = mongoose.model("bar", schema);

* Muốn thêm 1 bản ghi vào 1 bảng của cơ sở dữ liệu hiện tại đang được kết nối

const <Biến Bản Ghi> = new <Tên Biến Bảng>({

<Điền Thông Tin Theo Cấu Trúc Bảng>

})

<Biến Bản Ghi>

.save()

.then(() => <Code Xử Lí Khi Thêm Bản Ghi Thành Công>)

.catch((error) => <Code Xử Lí Khi Thêm Bản Ghi Không Thành Công>)

* <Biến Bản Ghi> tên gì cũng được
* Nếu trong cơ sở dữ liệu không có bảng có tên là <Tên Bảng> thì khi lưu nó sẽ tạo ra 1 bảng mới
* Ví dụ

const barfoo = new foo({

name: “FooBarFoo”,

email: “BarFoo@gmail.com”,

age: “125”

})

barfoo

.save()

.then(() => console.log(‘Success’))

.catch((error) => console.log(error))

1. Thủ Tục Trước Khi Làm Front End?

* Từ thư mục Back End, chuyển Terminal lên thư mục dự án
* Nhập dòng lệnh sau vào Terminal

npx create-react-app <Tên Thư Mục Chứa Front End>

* Lệnh này sẽ tạo thư mục mới chứa Front End, trong đó có thư mục “node\_modules” chứa thư viện, “public” chứa các ảnh công cộng, “src” chứa các File Code chính, File “.gitignore”, “README.md”, “package-lock.json”,

“package-json” chứa thông tin các thư viện Front End

* Còn muốn khởi tạo phiên bản lược bỏ 1 số File không cần thiết thì Copy thư mục

“E:\Ultimate Code\React Init” vào trong thư mục dự án rồi đổi tên thành tên tùy ý

Storage – Lưu Trữ Dữ liệu:

1. Truy Cập Storage Của Google Chrome = Dev Tools?

* Mở Chrome + vào trang Web muốn xem Storage mà nó lưu trên máy bạn + nhấn “Ctrl” + “Shift” + “I” để mở Dev Tools + vào Tab “Application” + xem mục “Storage”

1. Cách Hoạt Động Của Local Storage?

* Mỗi 1 tên miền sẽ chỉ truy cập được dữ liệu ứng với tên miền đó trong Local Storage
* Dữ liệu trong Local Storage sẽ không thay đổi cho đến khi bạn cố tình sử dụng Dev Tools để chỉnh sửa hoặc nếu dữ liệu này thuộc về tên miền A, thì Code trong A có thể thay đổi nó
* Tất cả các File HTML cục bộ trên máy tính của bạn sẽ dùng chung 1 Local Storage, với miền “file:///”

1. Local Storage Của Google Chrome Được Lưu Ở Đâu Trên Máy Tính?

* “C:\Users\<Tên Người Dùng>\AppData\Local\Google\Chrome\User Data\Default\Local Storage”

1. Cách Hoạt Động Của Session Storage?

* Y chang Local Storage chỉ khác là dữ liệu sẽ bị xóa sạch nếu đóng Tab, và cùng 1 trang Web nhưng 2 Tab khác nhau sẽ có Session Storage khác nhau

1. Session Storage Của Google Chrome Được Lưu Ở Đâu Trên Máy Tính?

* “C:\Users\<Tên Người Dùng>\AppData\Local\Google\Chrome\User Data\Default\Session Storage”

1. Cách Hoạt Động Của Cookies?

* Y chang Local Storage chỉ khác là dữ liệu sẽ bị xóa sau 1 khoảng thời gian do ta đặt, ngay cả khi ta đóng trình duyệt thì nó vẫn duy trì, gọi là Persistent Cookies, còn nếu không đặt, thì nó sẽ bị xóa giống như Session Storage, gọi là Session Cookies
* Mỗi lần máy bạn gửi 1 yêu cầu lên 1 tên miền, thì Cookies ứng với tên miền đó trong máy bạn cũng được gửi theo

1. Cookies Của Google Chrome Được Lưu Ở Đâu Trên Máy Tính?

* “C:\Users\<Tên Người Dùng>\AppData\Local\Google\Chrome\User Data\Default\Network”