# Tìm hiểu về Less

## Less là gì

Less là từ viết tắt của Learner Style Sheets. Less là ngôn ngữ tiền xử lý động có thế được biên dịch thành css và có thể chạy ở phía client hoặc phía server

Less là một css preprocessor, nó dùng để khắc phục các nhược điểm của css, giúp đoạn code ngắn gọn và dễ quản lý.

## Cài đặt trên client side

|  |  |
| --- | --- |
| |  | | --- | | <link rel=”stylesheet/les” type=”text/css” href=”styles.less”>  <script src = “less.js” type = “text/javascript”></script> | |

## Variables

Cú pháp :

@variable\_name: value;

Việc khai báo biến và sử lại biến đó sẽ dễ dàng cho việc thay đổi code.

Xét hai ví dụ sau:

|  |  |  |  |
| --- | --- | --- | --- |
| VD1   |  | | --- | | p{  color: #444;  font-size : 20px;  }  button {  color: #444;  font-size : 20px;  background : #fff;  }  .sidebar {  color: #444;  font-size : 20px;  background : #222;  } | | VD2:   |  | | --- | | @set-color : #444;  @set-font-size: 20px;  p{  color: @set-color;  font-size : @set-font-size;  }  button {  color: @set-color;  font-size : @set-font-size;  background : #fff;  }  .sidebar {  color: @set-color;  font-size : @set-font-size;  background : #222;  } | |

Trong **VD1**, tất cả các màu và kích thước chữ ở trong thẻ p, button và sidebar đều đồng bộ. Vậy khi muốn thay đổi thuộc tính ta sẽ phải thay thủ công ở tất cả mọi nơi.

Còn ở **VD2**, ta khai báo biến ***@set-color*** và ***@set-font-size rồi*** lưu giá trị cho nó, mỗi khi cần thay đổi thì ta chỉ cần thay đổi ở vị trí khai báo.

## Mixins

Ta có thể lồng các class, id này vào class, id khác

Xét hai ví dụ sau:

|  |  |  |  |
| --- | --- | --- | --- |
| VD1   |  | | --- | | .button-1 {  width : 80px;  height: 30px;  background: red;  }  .button-2 {  width : 80px;  height: 30px;  background: red;  color : white;  } | | VD2:   |  | | --- | | .button-1 (@width : 80px) {  witdth : @width;  height: 30px;  background: red;  }  .button-2 {  .button-1(100px);  color : white;  } | |

Ở **VD1**, ta phải thiết lập thuộc tính cho cả hai button một cách độc lập, mặc dù 2 button có rất nhiều điểm tương đồng. Thay vì copy lại đoạn thuộc tính giống nhau thì ta có thể gọi thẳng ***button-1*** vào trong ***button-2*** (**VD2**), cùng lúc có thể thay đổi các giá trị như ý muốn nhờ việc khai báo biến như trên.

## Netsing & scope

Xét hai ví dụ sau:

|  |  |  |  |
| --- | --- | --- | --- |
| VD1   |  | | --- | | main{  background : yellow;  }  main p {  color: red;  }  main article {  background: blue;  }  main article p {  color :white;  } | | VD2:   |  | | --- | | main {  background : yellow;  p{  color: red;  }  article{  background: blue;  p{  color :white;  }  }  } | |

Ở **VD1**, mỗi lần xét thuộc tính ta phải chỉ định cụ thể các thẻ lồng nhau. Còn **VD2** ta chì cần chỉ ra thẻ cha lớn nhất rồi xét lần lượt thuộc tính cho các thẻ con.

## Operators

Xét hai ví dụ sau:

|  |  |  |  |
| --- | --- | --- | --- |
| VD1   |  | | --- | | p{  font-size : 20px;  }  article {  font-size: 30px;  } | | VD2:   |  | | --- | | @base-size : 20px;  p{  font-size : @base-size;  }  article {  font-size: @base-size +10px;  } | |

Ở **VD1**, ***font-size*** của thẻ ***<p>*** và ***<article>*** cách nhau ***10px***, vậy mỗi lần thay đổi giá trị sẽ phải thay đổi cả hai. Còn **VD2**, việc dùng ***operator +*** và ***variable*** sẽ giúp ta dễ dàng thay đổi giá trị thuộc tính.