Module 5

Style Sheets

Module Overview

In this module, you will learn about:

- Style Sheets
- Selectors in CSS
- Properties and Values
- Inheritance and Cascades in CSS

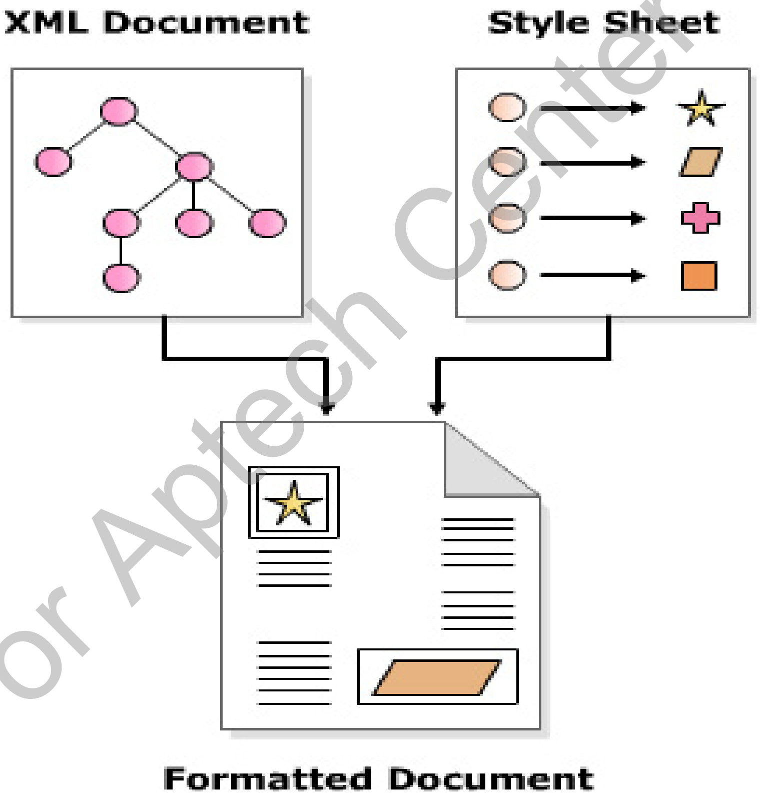
Lesson 1 – Style Sheets

In this first lesson, **Style Sheets**, you will learn to:

- Define and describe style sheets.
- Define and describe cascading style sheets (CSS).
- Explain how to implement styles with CSS.

Need of Style Sheets

- They are a set of rules that describe the appearance of data in an XML document.



Various Style Sheets

Cascading Style Sheet (CSS)

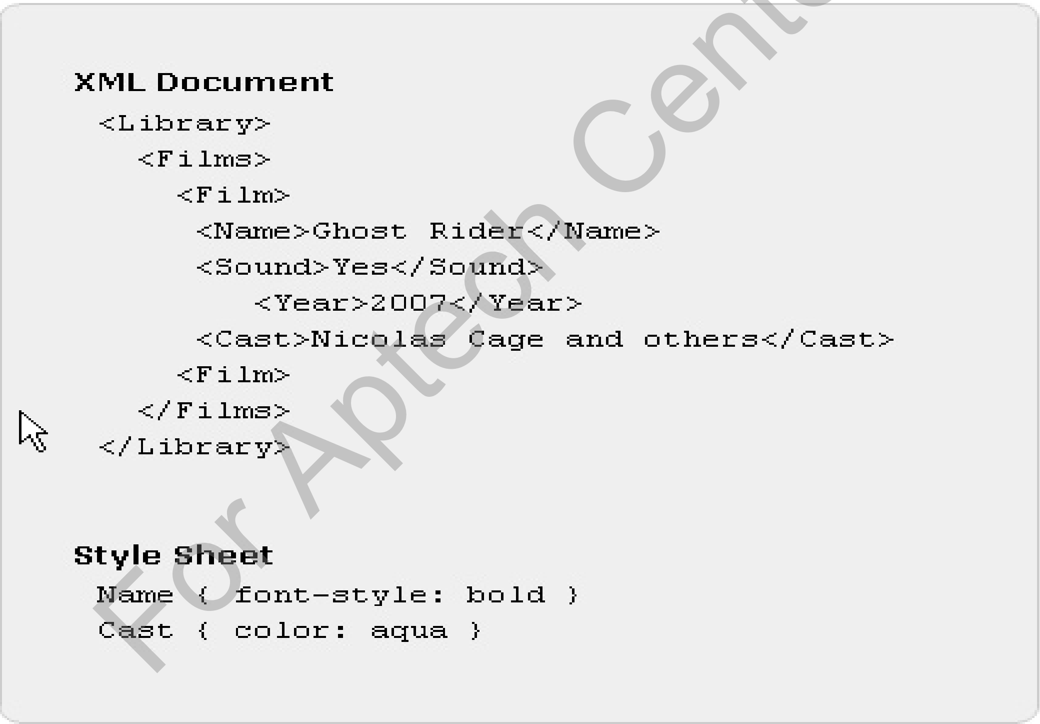
- It allows you to control the appearance of data in HTML and XML documents.

Extensible Style Sheet (XSL)

- It is used to define the appearance of data contained only in XML documents.

Cascading Style Sheets

- It comprises a set of rules for various elements in a document.
- Each rule defines how the data enclosed in the element should be displayed.
- The term cascading is derived from this ability to mix and match rules.



XML Document

```
<Library>
  <Films>
    <Film>
     <Name>Ghost Rider</Name>
     <Sound>Yes</Sound>
        <Year>2007</Year>
     <Cast>Nicolas Cage and others</Cast>
    <Film>
  </Films>
</Library>
```

Style Sheet

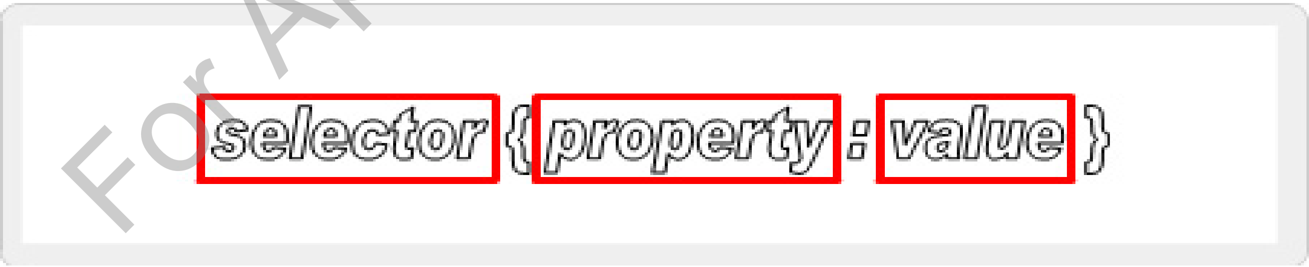
```
Name { font-style: bold }
Cast { color: aqua }
```

Benefits of CSS

- Any style or presentation changes to data can be achieved faster as changes are to be implemented in one place.
- Device independence can be achieved by defining different style sheets for different device.
- Reduction in document code as the presentation information is stored in a different file and can be reused.

Style Rules

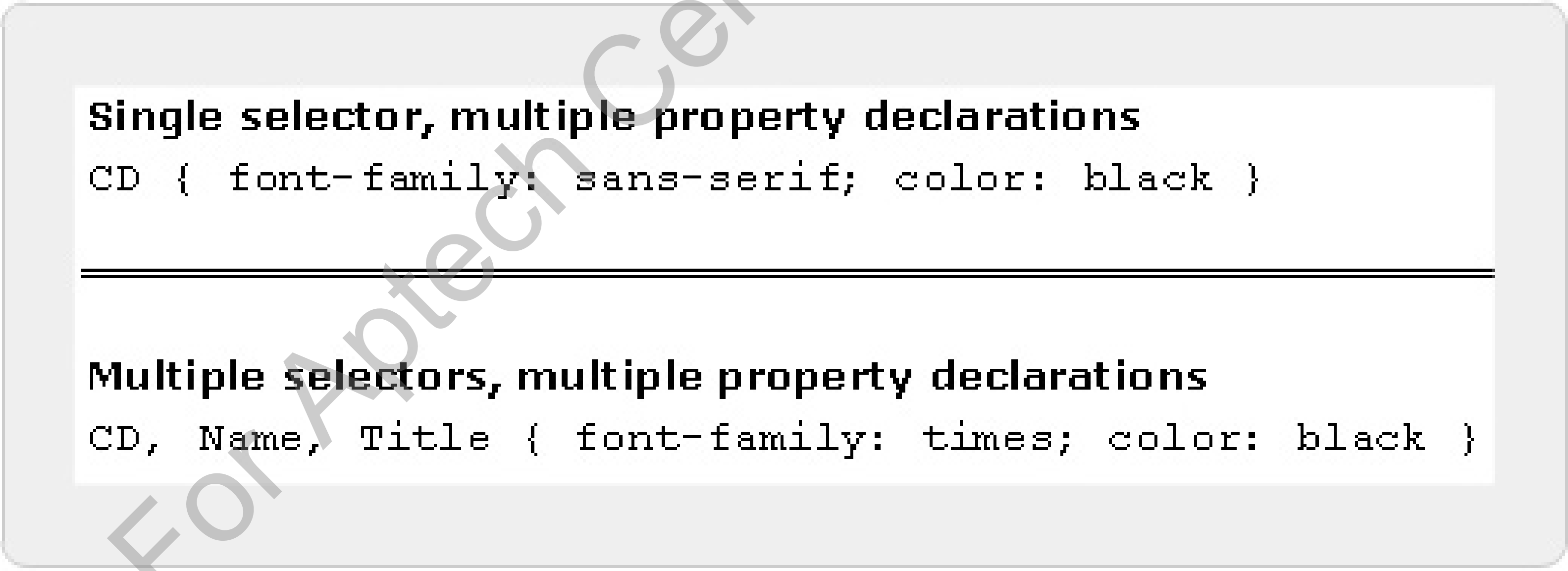
- They define how the content enclosed in an element will be displayed.
- These rules are applicable to all child elements within an element.
  - **Selector**
  - **Property**
  - **Value**



```
selector { property : value }
```

Ways of Writing Style rules

- A single selector can have more than one property-value pairs associated with it.
- A collection of one or more property-value pairs can be associated with more than one selector.



Single selector, multiple property declarations

```
CD { font-family: sans-serif; color: black }
```

Multiple selectors, multiple property declarations

```
CD, Name, Title { font-family: times; color: black }
```

External Style Sheets

- It should appear in the prolog section of an XML document, that is, it should appear before any tag of an element.
- One XML document can have more than one style sheet processing instructions, each linked to one .css file.

Syntax

```
<?xml-stylesheet href=”headers.css” type=”text/css”?>
```

where,

xml-stylesheet is the processing instruction.
url is the URL of a .css file; the .css file can on a local system or anywhere on the Internet.
type=”text/css” is optional; however if a browser does not support CSS, it informs the browser that it does not have to download a .css file.

Code Snippet

```
<?xml-stylesheet href=”url” [type=”text/css”]?>
```

Lesson 2 – Selectors in CSS

In this second lesson, **Selectors in CSS**, you will learn to:

- Identify simple selectors in CSS.
- State the use of universal selector in CSS.
- Describe ID selectors.

Simple Selectors

- It comprises an element name followed by one or more property declarations.
- It match every occurrence of the element in a document.

Code Snippet

```
/* Simple selector */
CD { color: black }
/* Single element, multiple property declarations */
CD { color: white; background-color: blue }
/* Multiple elements, multiple property declarations */
CD, Name, Title { color: white; background-color: blue }
```

Universal Selector

- It comprises an asterisk followed by property declarations.
- It matches all the elements in a document.

Code Snippet

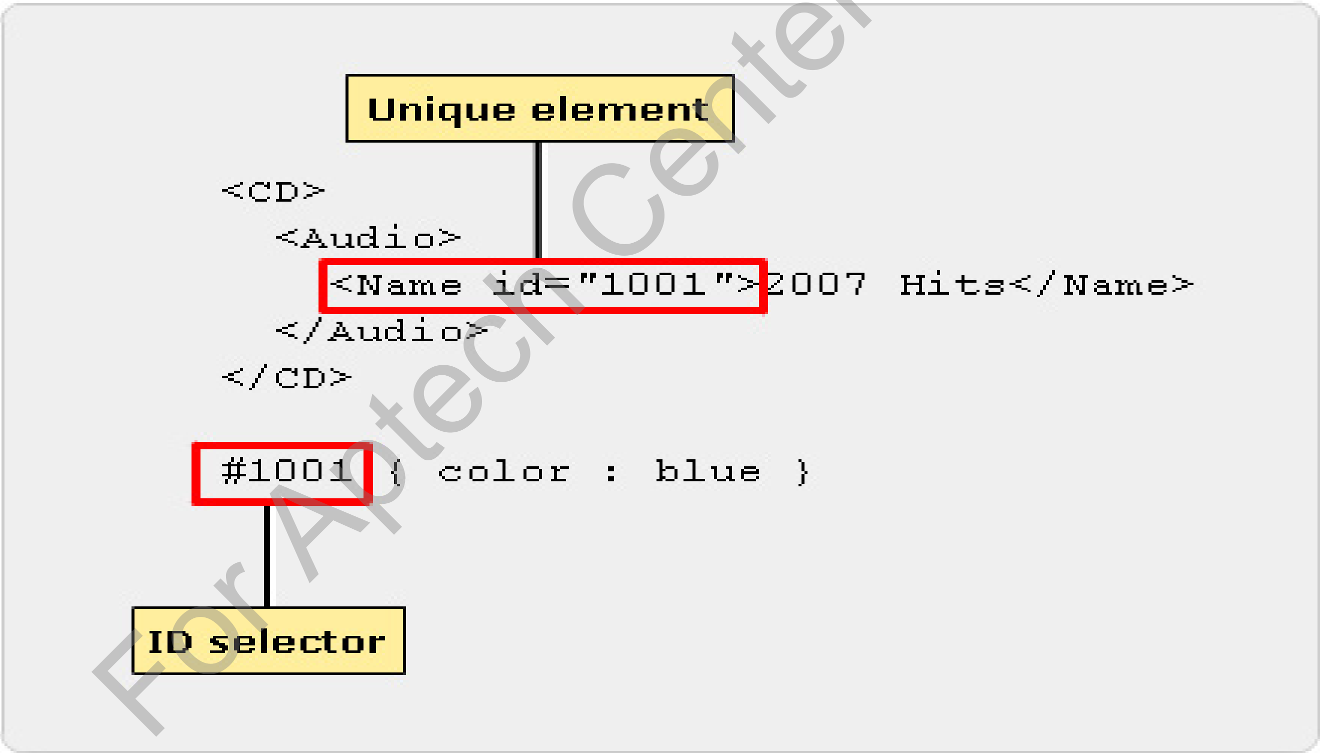
```
* { color : blue }
```

Displays the content of all elements in a document in blue.

ID Selector 1-2

- It comprises a hash (#) symbol, immediately followed by an attribute’s value followed by property declarations.

- It is used to define styles for unique elements in a document.



```
<CD>
  <Audio>
    <Name id="1001">2007 Hits</Name>
  </Audio>
</CD>

#1001 { color : blue }
```

ID Selector 2-2

Syntax

```
#attribute_value { property_declarations }
```

Code Snippet

```
#1001 { color : blue }
```

Displays the content of an element in blue if its id attribute's value equals 1001.

Lesson 3 – Properties and Values

In this third lesson, **Properties and Values**, you will learn to:

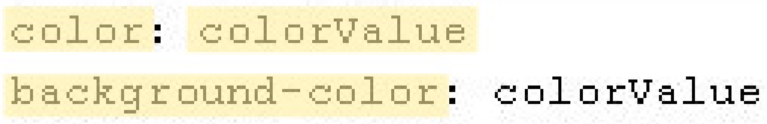
- State and describe how to use color properties.
- Describe the font property.
- Describe the other properties such as margins, borders, padding.
- Explain briefly about positioning and alignment.

Color Properties

Color Names	RGB Percentages	RGB Values	Hexadecimal Values
aqua	rgb(0%,65%,65)	rgb(0,160,160)	#00a0a0
black	rgb(0%,0%,0%)	rgb(0,0,0)	#000000
blue	rgb(0%,32%,100)	rgb(0,80,255)	#0050ff
gray	rgb(65%,65%,65%)	rgb(160,160,160)	#a0a0a0
green	rgb(0%,100%,0%)	rgb(0,255,0)	#00ff00
lime	rgb(0%,65%,0%)	rgb(0,160,0)	#00a000
maroon	rgb(70%,0%,32%)	rgb(176,0,80)	#b00050
navy	rgb(0%,0%,65%)	rgb(0,0,160)	#0000a0
olive	rgb(65%,65%,0%)	rgb(160,160,0)	#a0a000
purple	rgb(65%,0%,65%)	rgb(160,0,160)	#a000a0
red	rgb(100%,0%,32%)	rgb(255,0,80)	#ff0050
silver	rgb(90%,90%,90%)	rgb(225,225,255)	#d0d0d0
teal	rgb(0%,65%,100%)	rgb(0,160,255)	#00a0ff
white	rgb(100%,100%,100%)	rgb(255,255,255)	#ffffff
yellow	rgb(100%,100%,0%)	rgb(255,255,0)	#ffff00

Setting Color Properties 1-3

Syntax



```
color: colorValue
background-color: colorValue
```

where,

`color`

Property to set the foreground color of text in an element.

`colorValue`

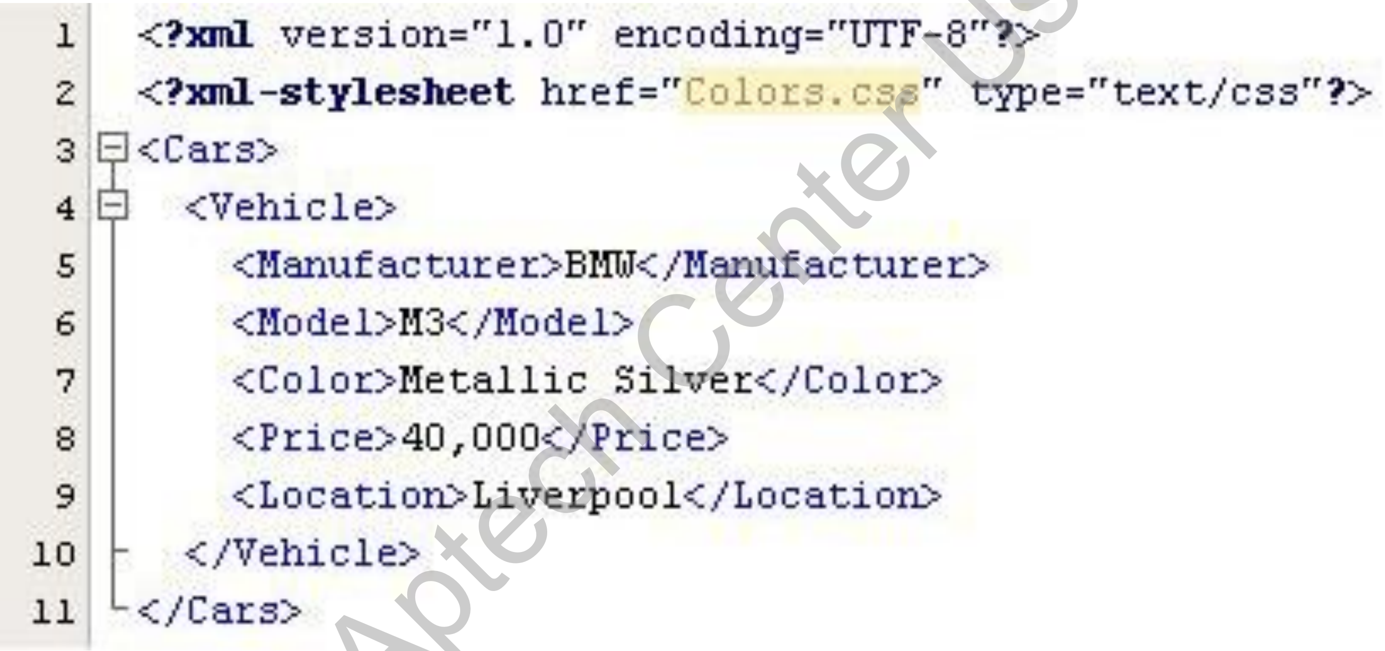
colorValue can take up any value from the CSS color table.

`background-color`

Property to set the background color of text in an element.

Setting Color Properties 2-3

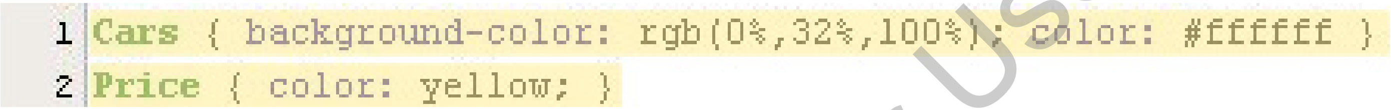
Code Snippet



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="Colors.css" type="text/css"?>
<Cars>
  <Vehicle>
    <Manufacturer>BMW</Manufacturer>
    <Model>M3</Model>
    <Color>Metallic Silver</Color>
    <Price>40,000</Price>
    <Location>Liverpool</Location>
  </Vehicle>
</Cars>
```

Setting Color Properties 3-3

Style Sheet



```
Cars { background-color: rgb(0%,32%,100%); color: #ffffff }
Price { color: yellow; }
```

where,

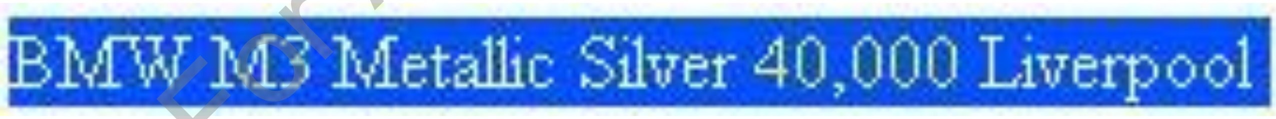
`Cars { background-color:rgb(0%,32%,100%);color: #ffffff }`

Causes the text enclosed in Cars element to be displayed in white color with a background color of blue.

`Price{color:yellow;}`

Causes the text enclosed in Price element to be displayed in yellow color.

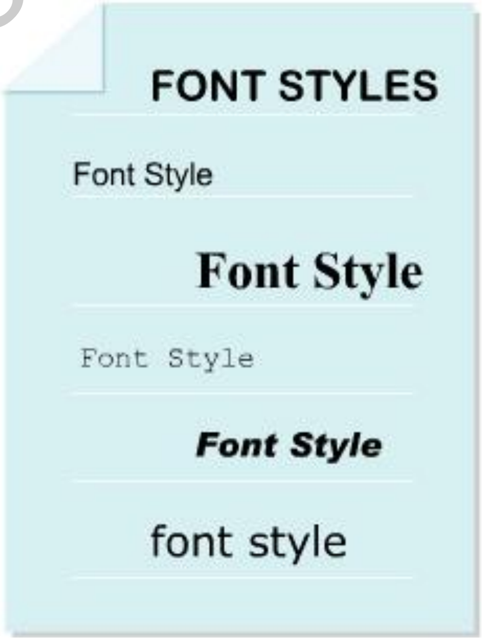
Output



```
BMW M3 Metallic Silver 40,000 Liverpool
```

Font Properties

Property Name	Description
`font-family`	To specify the font family
`font-size`	To specify the size of font
`font-style`	To specify the style of font
`font-weight`	To specify the weight of font



FONT STYLES

Font Style

Font Style

Font Style

Font Style

font style

Font-family Property 1-2

Syntax

```
font-family : "font-family name(s)"
```

where,

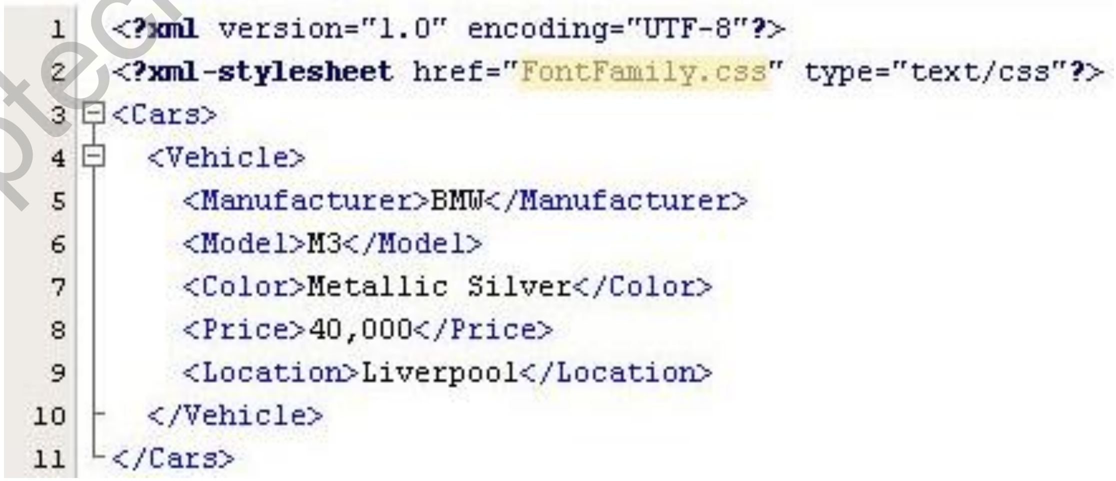
`font-family`

Property to specify the font-family to be used.

`font-family name(s)`

Comma separated list of font-family names such as serif, san-serif, monospace, cursive, and fantasy. The list should start with the most specific font in which you want to display the data and end with the most generic font.

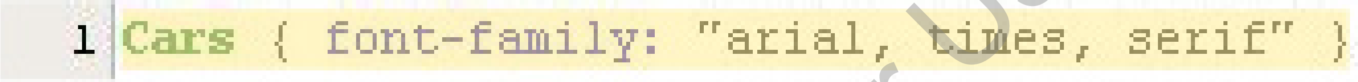
Code Snippet



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="FontFamily.css" type="text/css"?>
<Cars>
  <Vehicle>
    <Manufacturer>BMW</Manufacturer>
    <Model>M3</Model>
    <Color>Metallic Silver</Color>
    <Price>40,000</Price>
    <Location>Liverpool</Location>
  </Vehicle>
</Cars>
```

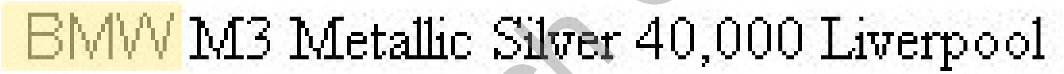
Font-family Property 2-2

Style Sheet



```
Cars { font-family: "arial, times, serif" }
```

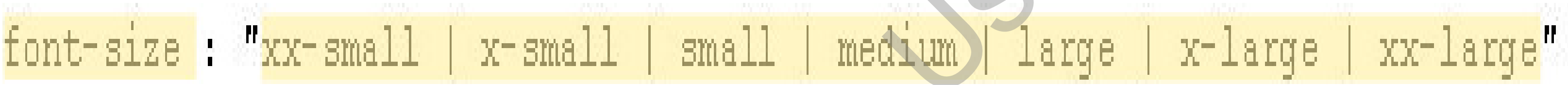
Output



```
BMW M3 Metallic Silver 40,000 Liverpool
```

Font-size Property 1-3

Syntax



```
font-size : "xx-small | x-small | small | medium | large | x-large | xx-large"
```

where,

`font-size`

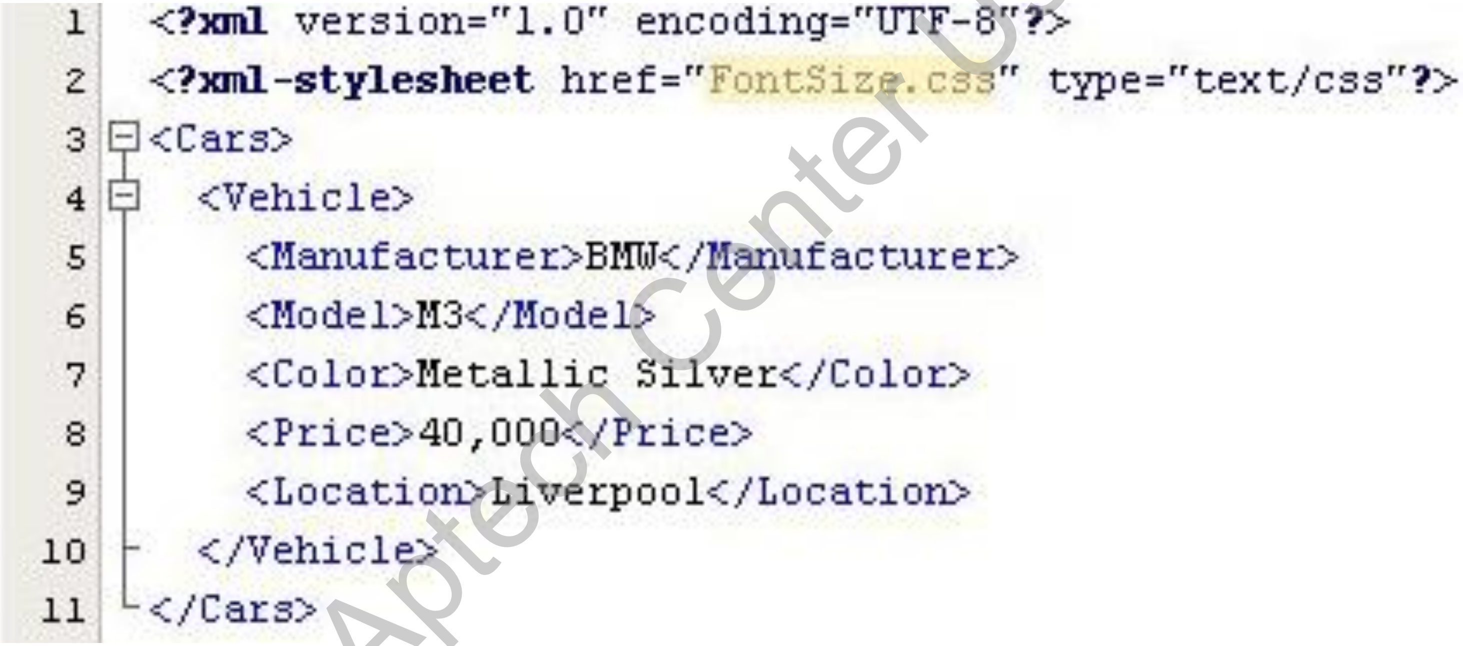
Property to specify the size of font.

`xx-small|x-small|small|medium large|x-large|xx-large`

One of various values that can be assigned to the property font-size.

Font-size Property 2-3

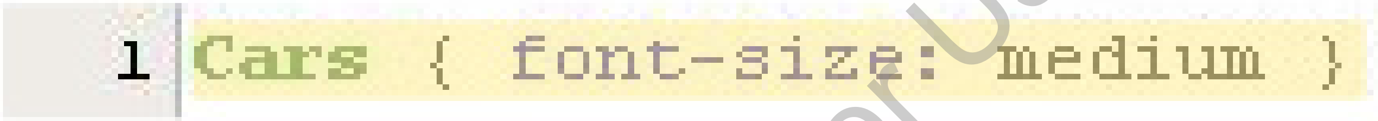
Code Snippet



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="FontSize.css" type="text/css"?>
<Cars>
  <Vehicle>
    <Manufacturer>BMW</Manufacturer>
    <Model>M3</Model>
    <Color>Metallic Silver</Color>
    <Price>40,000</Price>
    <Location>Liverpool</Location>
  </Vehicle>
</Cars>
```

Font-size Property 3-3

Style Sheet



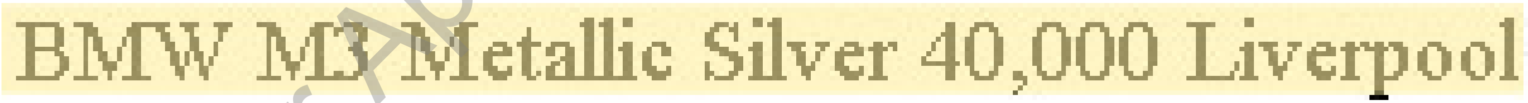
```
Cars { font-size: medium }
```

where,

`Cars{font-size:medium }`

All the text enclosed in Cars element and its child elements will be displayed with medium font size.

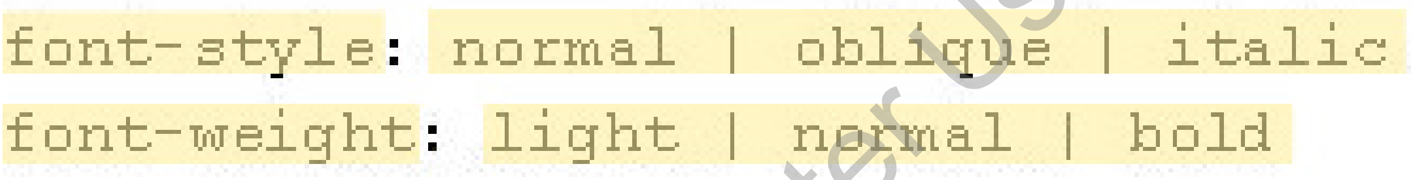
Output



BMW M3 Metallic Silver 40,000 Liverpool

Font Style and Weight Properties 1-3

Syntax



```
font-style: normal | oblique | italic
font-weight: light | normal | bold
```

where,

`font-style`

Property to specify the style of text in an element.

`normal|oblique|italic`

One of the values that can be assigned to font- style property.

`font-weight`

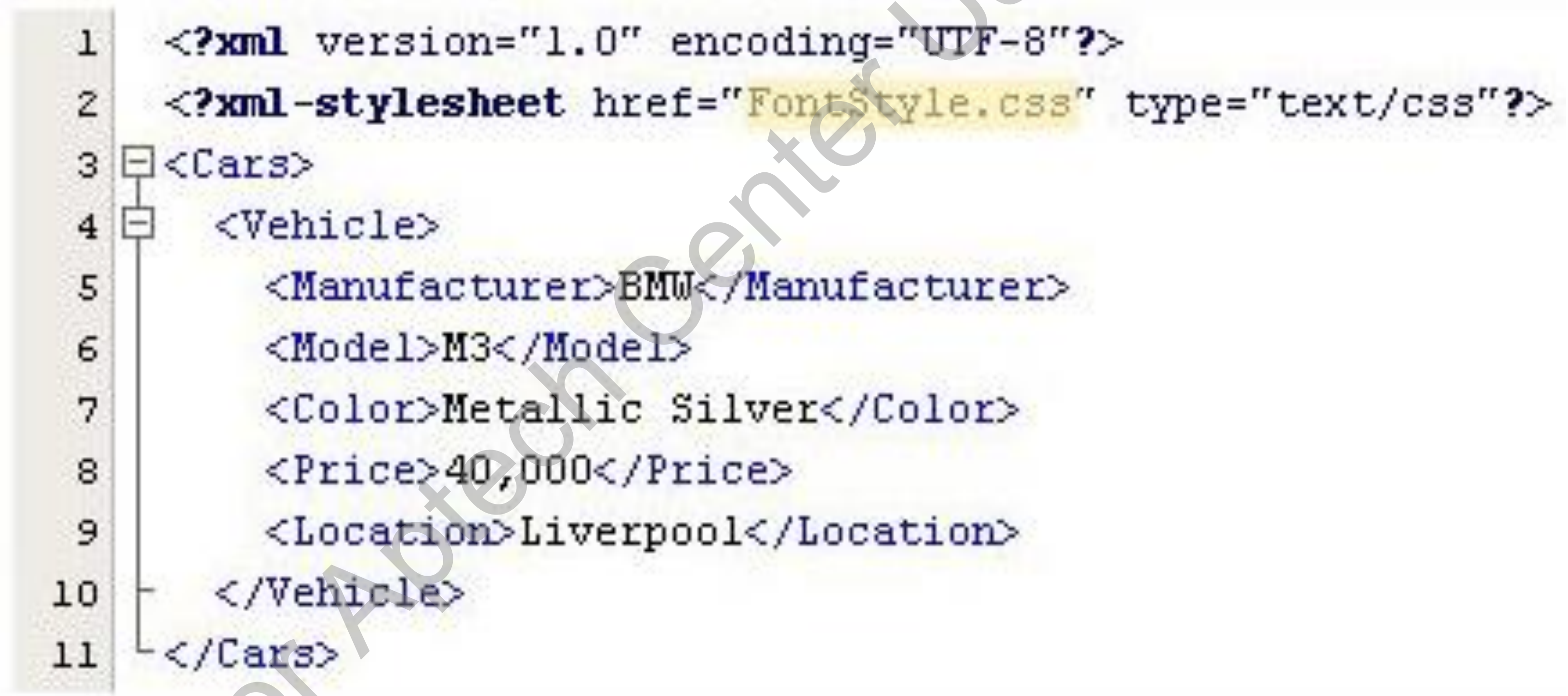
Property to specify the weight style of the text   in an element.

`light | normal | bold`

One of the values that can be assigned to font- weight property.

Font Style and Weight Properties 2-3

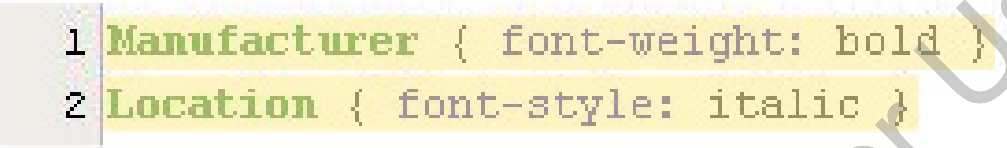
Code Snippet



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="FontStyle.css" type="text/css"?>
<Cars>
  <Vehicle>
    <Manufacturer>BMW</Manufacturer>
    <Model>M3</Model>
    <Color>Metallic Silver</Color>
    <Price>40,000</Price>
    <Location>Liverpool</Location>
  </Vehicle>
</Cars>
```

Font Style and Weight Properties 3-3

Style Sheet



```
Manufacturer { font-weight: bold }
Location { font-style: italic }
```

where,

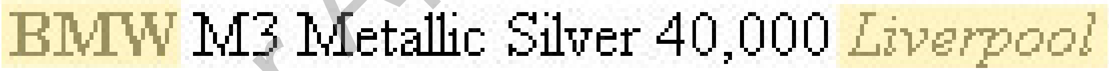
`Manufacturer { font-weight: bold }`

The text enclosed in Manufacturer element will be displayed in bold.

`Location {font-style: italic}`

The text enclosed in Location element will be displayed in italics.

Output



BMW M3 Metallic Silver 40,000 *Liverpool*

Margins in CSS 1-3

Syntax

```
margin-left | margin-right | margin-top | margin-bottom =
marginValue
```

where,

`margin-left | margin-right | margin-top | margin- bottom`

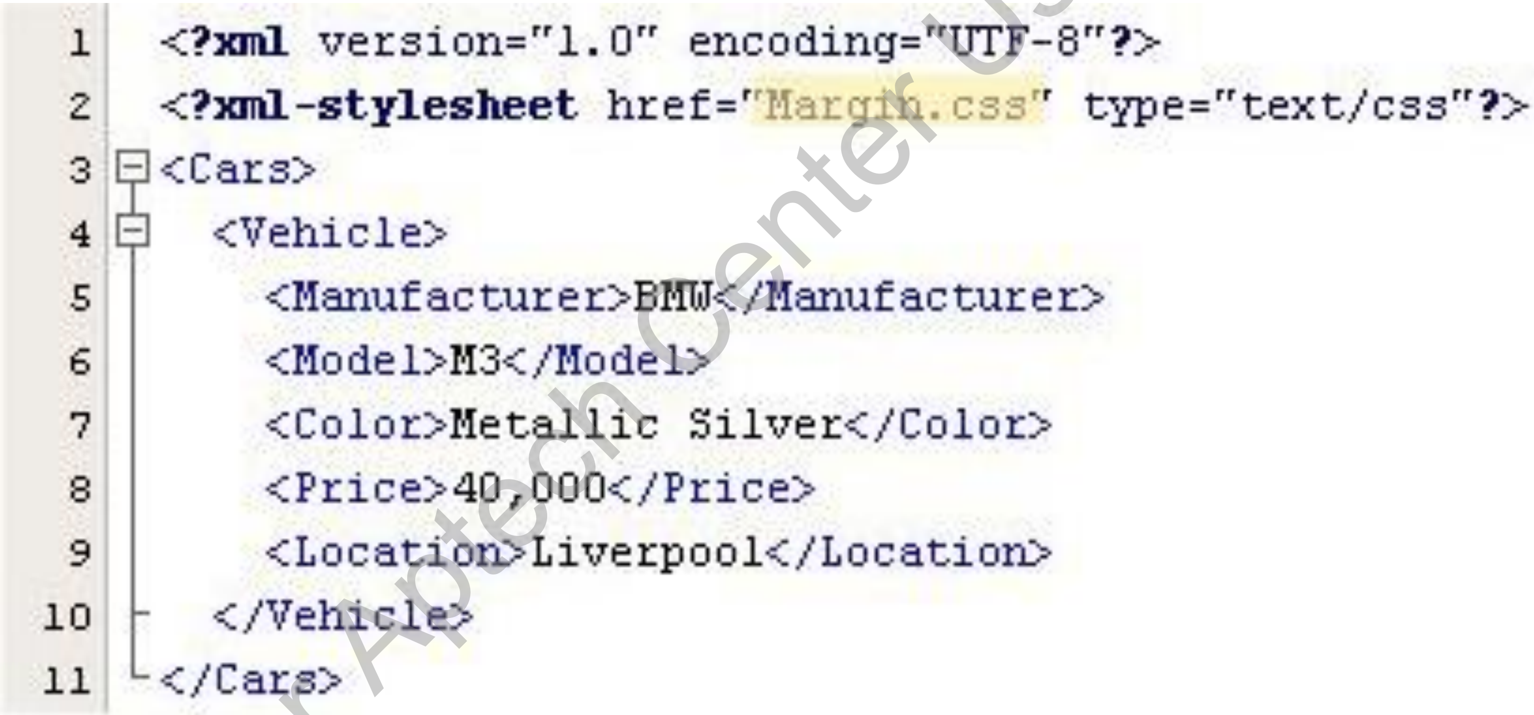
The various margin properties to set left, right, top and bottom margins.

`marginValue`

The value to be assigned to one of the margin properties.This value can be a fixed value or a percentage.

Margins in CSS 2-3

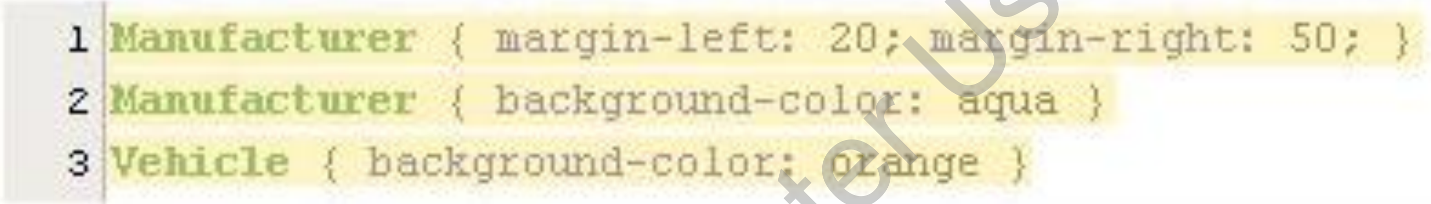
Code Snippet



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="Margin.css" type="text/css"?>
<Cars>
  <Vehicle>
    <Manufacturer>BMW</Manufacturer>
    <Model>M3</Model>
    <Color>Metallic Silver</Color>
    <Price>40,000</Price>
    <Location>Liverpool</Location>
  </Vehicle>
</Cars>
```

Margins in CSS 3-3

Style Sheet



```
Manufacturer { margin-left: 20; margin-right: 50; }
Manufacturer { background-color: aqua }
Vehicle { background-color: orange }
```

where,

`Manufacturer {margin-left:20;margin-right:50;}`

Inserts a space of 20 pixels to the left and a space of 50 pixels to the right of text enclosed in element Manufacturer.

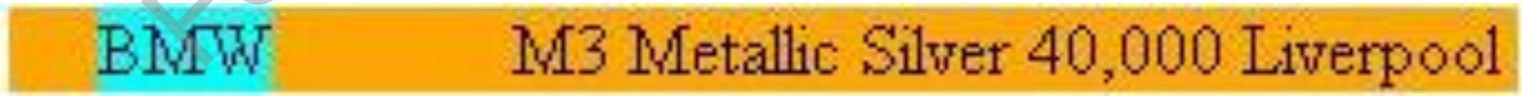
`Manufacturer {background-color: aqua}`

Sets the background of text enclosed in element Manufacturer to aqua.

`Vehicle {background-color: orange}`

Sets the background of text enclosed in element Vehicle to orange.

Output



```
BMW          M3 Metallic Silver 40,000 Liverpool
```

Border Properties in CSS 1-3

Syntax

```
border : border_width border_style border_color
```

where,

`border`: Property to set the border of the box surrounding an element’s data.

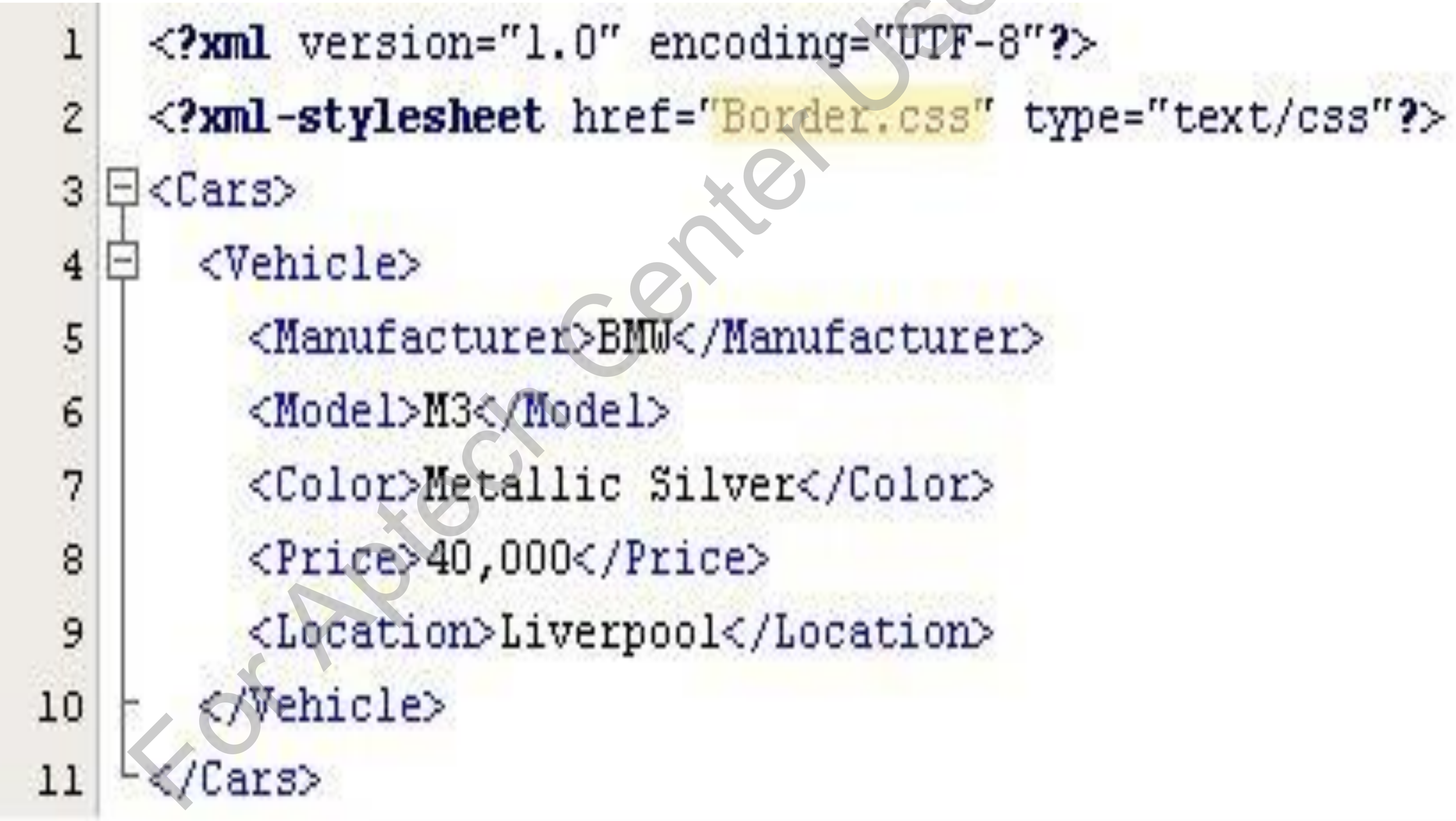
`border_width`: Specifies the thickness of the border. Possible values are thin, medium and thick.

`border_style`: Specifies the style of the border. Possible values are solid, dashed, dotted, groove, ridge, double, inset, outset.

`border_color`: Specifies the color of the border. All values that are applicable to CSS color property are allowed.

Border Properties in CSS 2-3

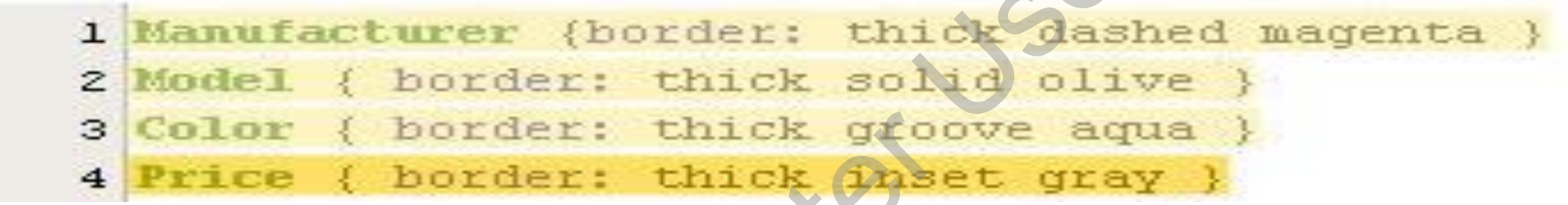
Code Snippet



```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="Border.css" type="text/css"?>
<Cars>
  <Vehicle>
    <Manufacturer>BMW</Manufacturer>
    <Model>M3</Model>
    <Color>Metallic Silver</Color>
    <Price>40,000</Price>
    <Location>Liverpool</Location>
  </Vehicle>
</Cars>
```

Border Properties in CSS 3-3

Style Sheet



```
Manufacturer {border: thick dashed magenta }
Model { border: thick solid olive }
Color { border: thick groove aqua }
Price { border: thick inset gray }
```

where,

`Manufacturer {border: thick dashed magenta }`

Displays a thick and dashed magenta border around the content of Manufacturer element.

`Model { border: thick solid olive }`

Displays a thick and solid olive border around the content of Model element.

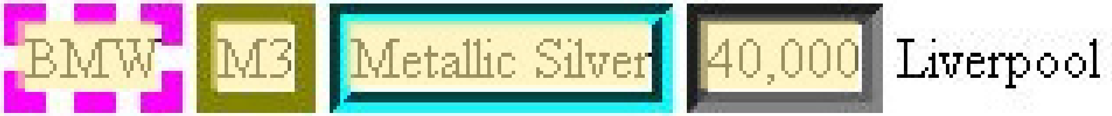
`Color {border: thick groove aqua}`

Displays a thick and groove aqua border around the content of Color element.

`Price {border: thick inset gray}`

Displays a thick and inset gray border around the content of Price element.

Output



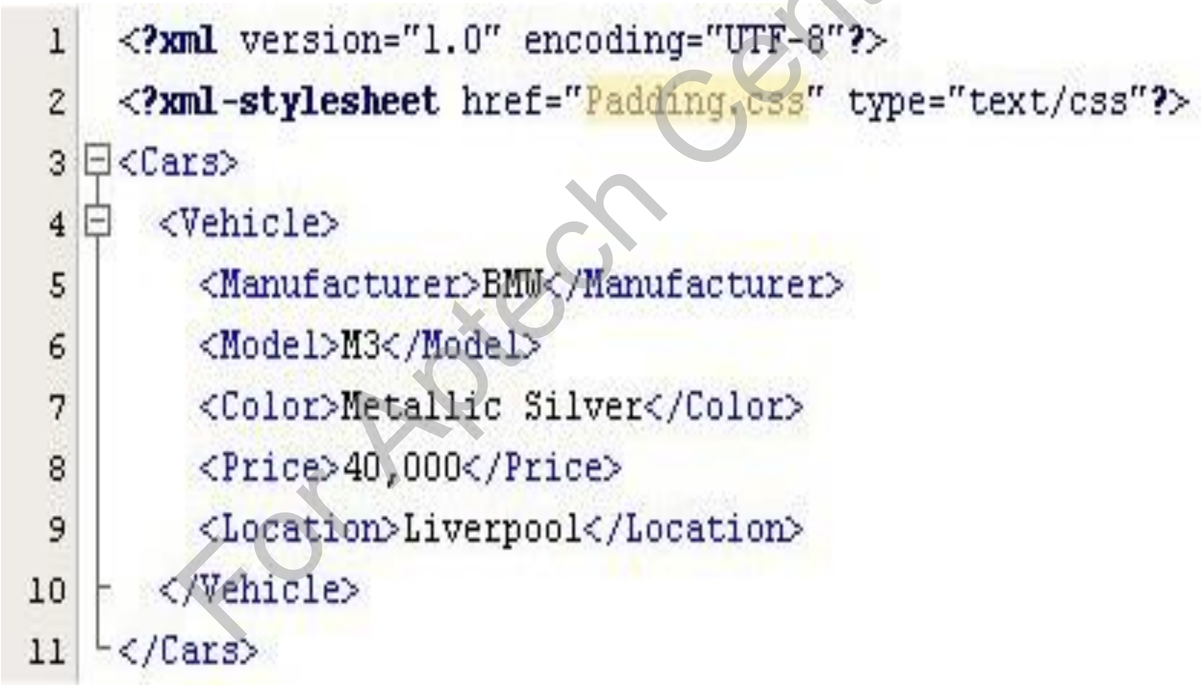
```
BMW M3 Metallic Silver 40,000 Liverpool
```

Padding Properties in CSS 1-2

Syntax

```
padding : padding_width
```

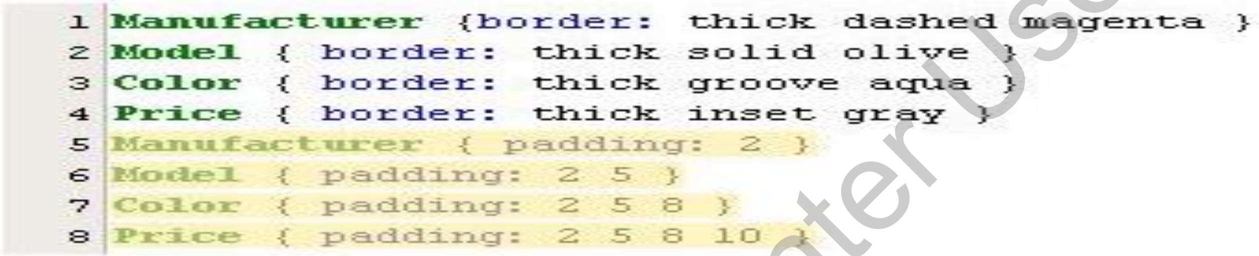
Code Snippet



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="Padding.css" type="text/css"?>
<Cars>
  <Vehicle>
    <Manufacturer>BMW</Manufacturer>
    <Model>M3</Model>
    <Color>Metallic Silver</Color>
    <Price>40,000</Price>
    <Location>Liverpool</Location>
  </Vehicle>
</Cars>
```

Padding Properties in CSS 2-2

Style Sheet



```
Manufacturer {border: thick dashed magenta }
Model { border: thick solid olive }
Color { border: thick groove aqua }
Price { border: thick inset gray }
Manufacturer { padding: 2 }
Model { padding: 2 5 }
Color { padding: 2 5 8 }
Price { padding: 2 5 8 10 }
```

where,

`Manufacturer { padding: 2 }`

Inserts padding of 2 pixels between the four borders and text of Manufacturer element. The four borders applicable are top, right, bottom, and left.

`Model { padding: 2 5 }`

Inserts padding between the borders and text of Model element. The value 2 is applied to top and bottom borders and 5 is applied to left and right borders.

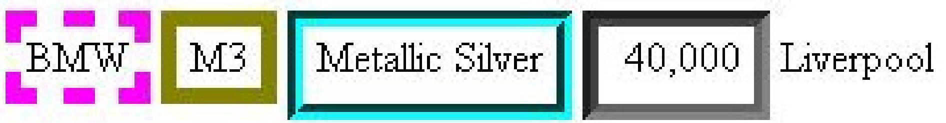
`Color { padding: 2 5 8 }`

Inserts padding between the borders and text of Color element. The value 2 is applied to top border, 5 to left and right borders, and value 8 to bottom border.

`Price { padding: 2 5 8 10 }`

Inserts padding between the borders and text of Price element. The value 2 is applied to top border, 5 to right border, 8 to bottom border, and 10 to left border.

Output



BMW M3 Metallic Silver 40,000 Liverpool

CSS Units

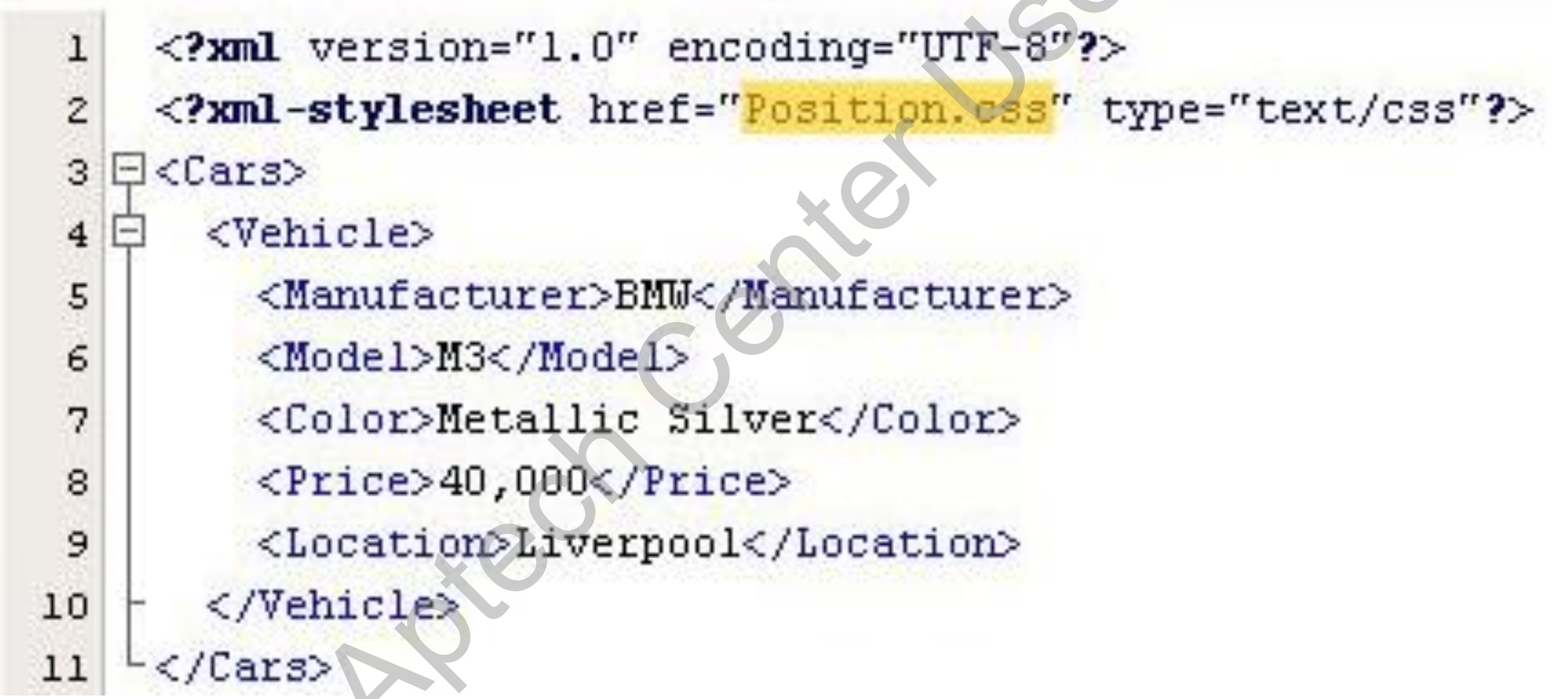
Unit Type	Unit Designator
Relative	em – defines the height of element’s font. ex – defines the x-height of element’s font. px – defines the pixel relative to display device. % - percentage.
Absolute	in – inches cm – centimeters mm – millimeters pt – 1/72 inch pc – 12 pt

Position Properties

Property	Description	Possible Values
position	Property to place an element in a static, relative, absolute or fixed position	static, fixed, relative or absolute
top	Property specifying how far the top edge of an element is above/below the top edge of the parent element	auto, integer or floating point values adhering to CSS length units
left	Property specifying how far the left edge of an element is to the right/left of the left edge of the parent element	auto, integer or floating point values adhering to CSS length units
bottom	Property specifying how far the bottom edge of an element is above/below the bottom edge of the parent element	auto, integer or floating point values adhering to CSS length units
right	Property specifying how far the right edge of an element is to the left/right of the right edge of the parent element	auto, integer or floating point values adhering to CSS length units

Position Properties Example 1-3

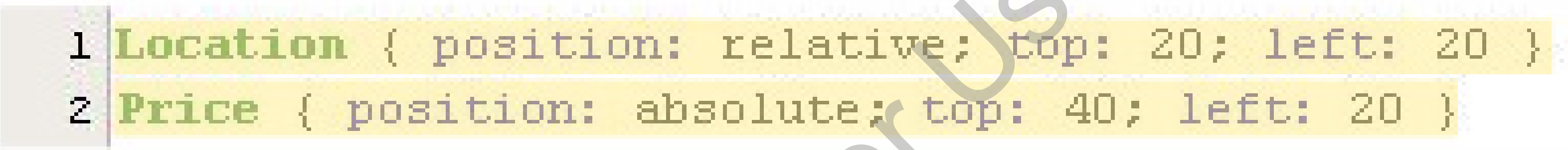
Code Snippet



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="Position.css" type="text/css"?>
<Cars>
  <Vehicle>
    <Manufacturer>BMW</Manufacturer>
    <Model>M3</Model>
    <Color>Metallic Silver</Color>
    <Price>40,000</Price>
    <Location>Liverpool</Location>
  </Vehicle>
</Cars>
```

Position Properties Example 2-3

Style Sheet



```
Location { position: relative; top: 20; left: 20 }
Price { position: absolute; top: 40; left: 20 }
```

where,

`Location {position:relative;top:20;left:20}`

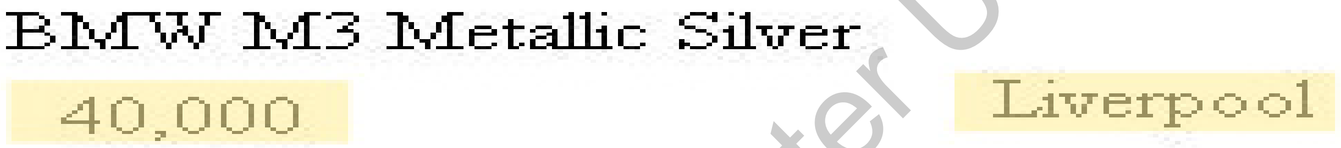
Positions the text of Location element relative to the previous element. However, inside the box, the content is placed at 20 pixels from top and 20 pixels from left.

`Price {position:absolute;top:40;left:20}`

Positions the text of Price element at an absolute location of 40 pixels from the top and 20 pixels from the left.

Position Properties Example 3-3

Output



BMW M3 Metallic Silver

40,000

Liverpool

where,

`40,000`

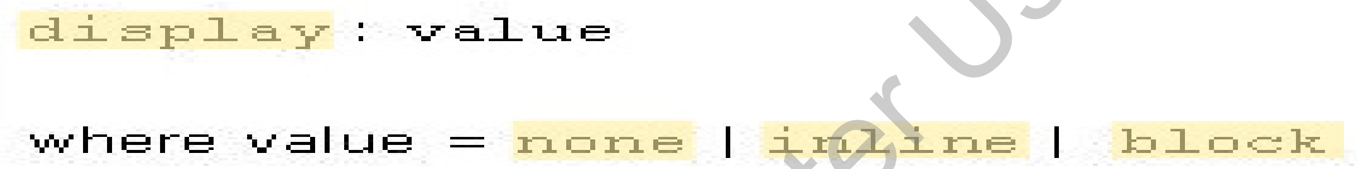
Text displayed using absolute positioning

`Liverpool`

Text displayed using relative positioning

Display Property 1-3

Syntax



```
display : value

where value = none | inline | block
```

where,

display

Property to specify how the element is to be rendered.

none

No rendering is applied to the element.

inline

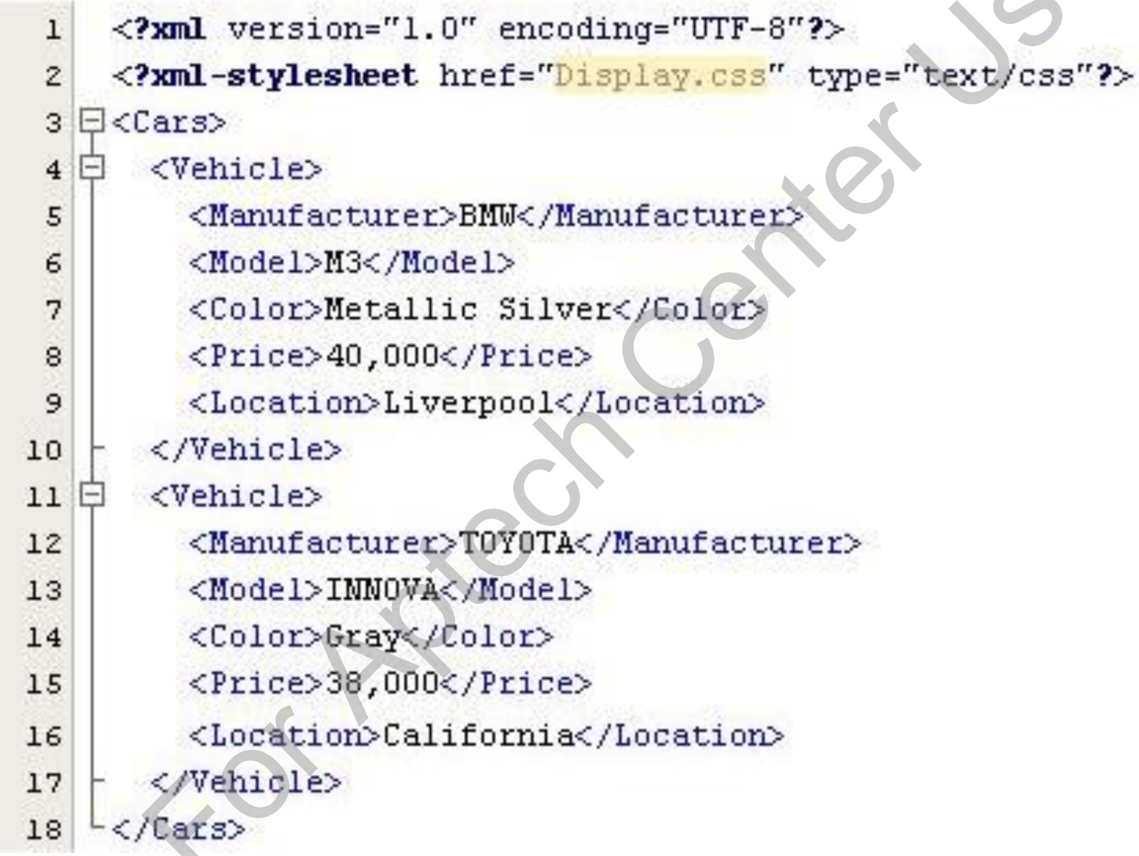
Displays the element text as in line.This is the default value if no value is specified.

Block

Displays the element text on a new line in a block of its own.

Display Property 2-3

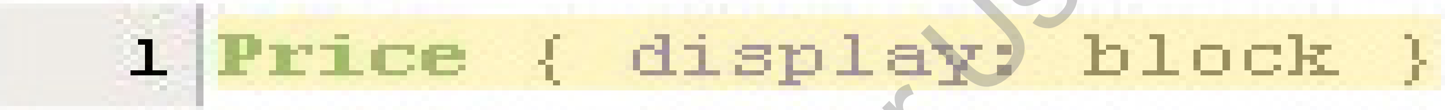
Code Snippet



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="Display.css" type="text/css"?>
<Cars>
  <Vehicle>
    <Manufacturer>BMW</Manufacturer>
    <Model>M3</Model>
    <Color>Metallic Silver</Color>
    <Price>40,000</Price>
    <Location>Liverpool</Location>
  </Vehicle>
  <Vehicle>
    <Manufacturer>TOYOTA</Manufacturer>
    <Model>INNOVA</Model>
    <Color>Gray</Color>
    <Price>38,000</Price>
    <Location>California</Location>
  </Vehicle>
</Cars>
```

Display Property 3-3

Style Sheet



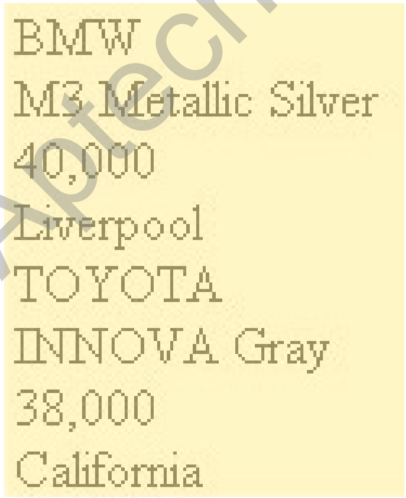
```
Price { display: block }
```

where,

`Price { display: block }`

Inserts a new line before and after the text of element `Price`.

Output



```
BMW
M3 Metallic Silver
40,000
Liverpool
TOYOTA
INNOVA Gray
38,000
California
```

Text Alignment and Indentation 1-3

Syntax

```
text-align : alignment_value

text-indent : value
```

where,

`text-align`

Property to align the text in a block.

`alignment_value`

Can be one of the following: left(default), right, center and justify.

`text-indent`

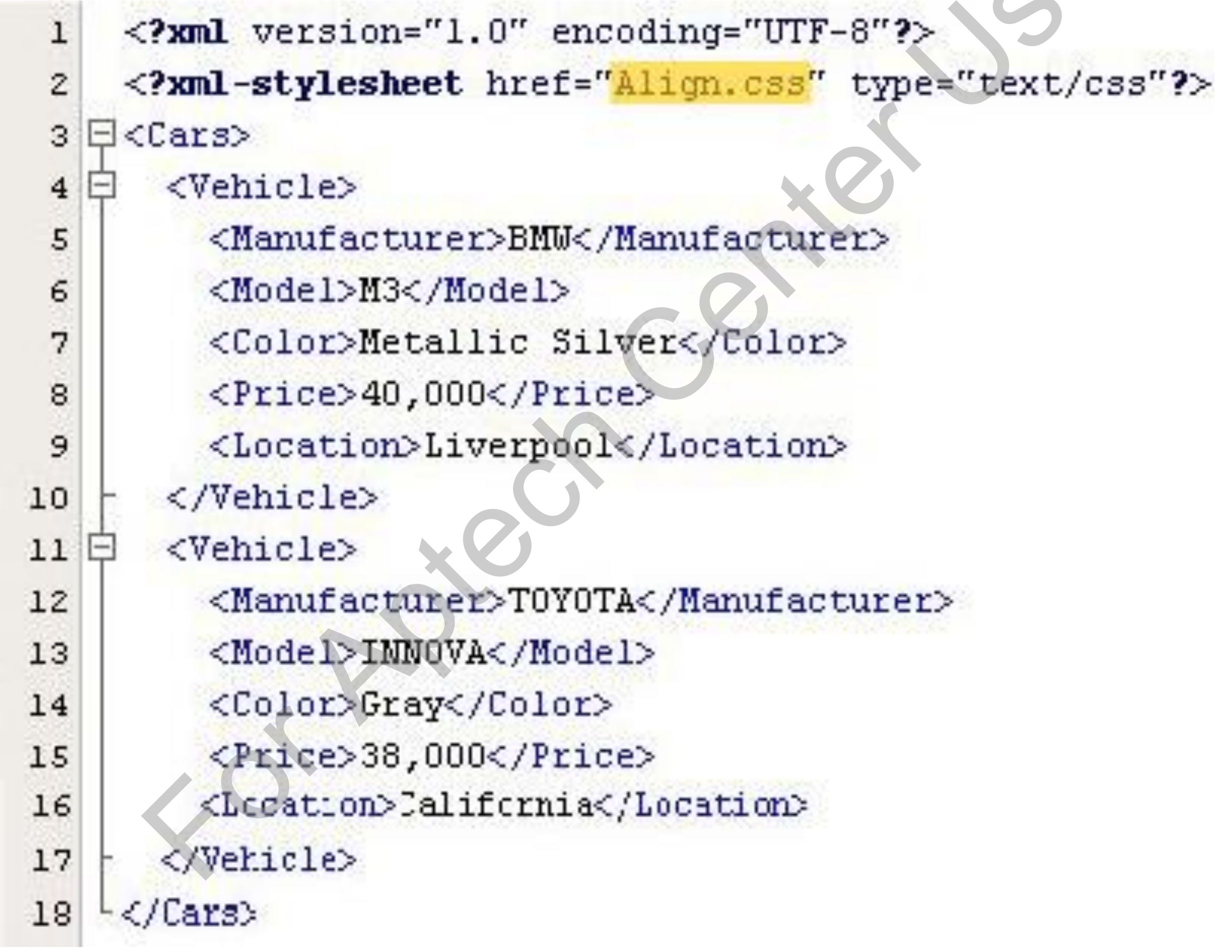
Property to indent the text in a block.

`value`

Floating point value followed by absolute units designators or relative units designators; or an integer value followed by percentage (%) symbol.

Text Alignment and Indentation 2-3

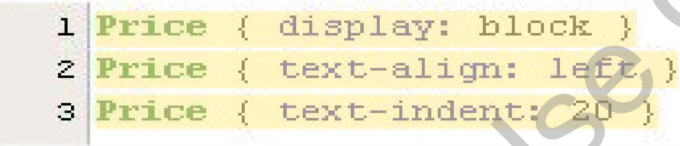
Code Snippet



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="Align.css" type="text/css"?>
<Cars>
  <Vehicle>
    <Manufacturer>BMW</Manufacturer>
    <Model>M3</Model>
    <Color>Metallic Silver</Color>
    <Price>40,000</Price>
    <Location>Liverpool</Location>
  </Vehicle>
  <Vehicle>
    <Manufacturer>TOYOTA</Manufacturer>
    <Model>INNOVA</Model>
    <Color>Gray</Color>
    <Price>38,000</Price>
    <Location>California</Location>
  </Vehicle>
</Cars>
```

Text Alignment and Indentation 3-3

Style Sheet



```
Price { display: block }
Price { text-align: left }
Price { text-indent: 20 }
```

where,

`Price {display: block}`

Inserts a new line before and after the text of element Price and displays the text in a separate block.

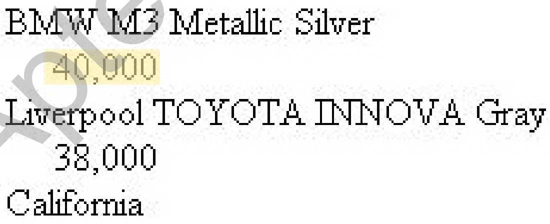
`Price {text-align: left}`

Left aligns the text of Price element.

`Price {text-indent: 20}`

Indents the text of Price element at 20 pixels.

Output



```
BMW M3 Metallic Silver
   40,000
Liverpool TOYOTA INNOVA Gray
   38,000
California
```

where,

`40,000`

The text in element Price is display on a separate row and left indented at 20 pixels.

Lesson 4 - Inheritance and Cascades in CSS

In this last lesson, **Inheritance and Cascades in CSS**, you will learn to:

- Define the process of cascading.
- Explain inheritance.

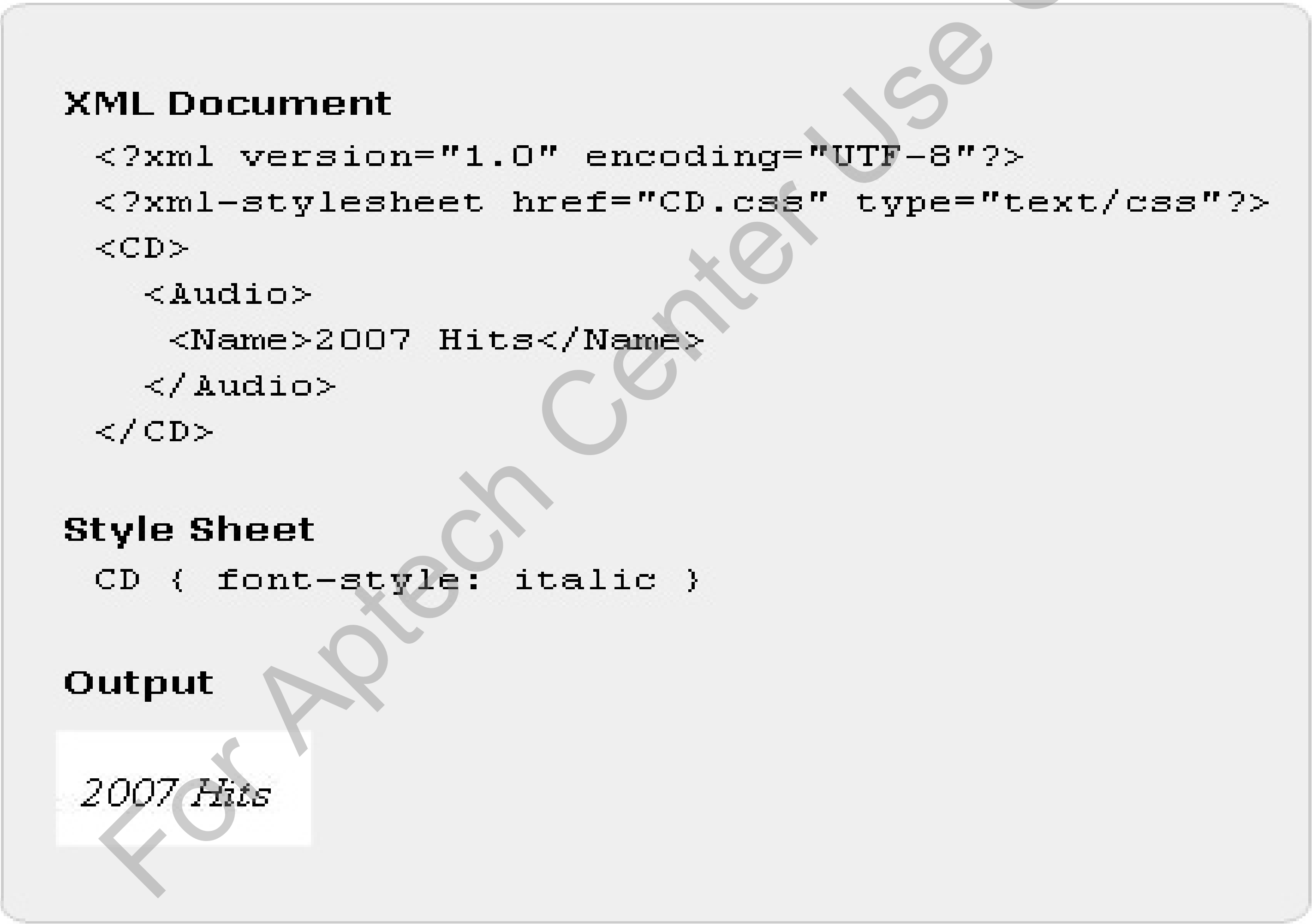
Cascading in CSS

- All declarations that apply to an element and property are grouped.
- Declarations are sorted by weight and origin.
- Declarations are sorted by specificity of selector.
- Declarations are sorted by the order specified.

Inheritance in CSS 1-2

- Inheritance is the ability of one entity to acquire the characteristics of another entity.
- In CSS, the child elements inherit the styles rules defined for parent element.
- However, this is applicable only if the child has no explicit style rule defined for it.
- Otherwise, the style rule defined for child element overrides the style rule defined for parent element.

Inheritance in CSS 2-2



XML Document

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="CD.css" type="text/css"?>
<CD>
  <Audio>
   <Name>2007 Hits</Name>
  </Audio>
</CD>
```

Style Sheet

```
CD { font-style: italic }
```

Output

2007 Hits

Summary

- **Style Sheets**
  - Style Sheets are a set of rules that define the appearance of data.
  - These rules are written in a file with the extension `.css`.
  - The `.css` file is associated with an XML document using the xml-processing instruction.
- **Selectors in CSS**
  - Selectors define the elements to which the styles will be applied.
  - The various types of selectors are simple, universal and ID selectors.
- **Properties and Values**
  - CSS provides several properties to define the appearance of data.
  - Some of these properties are color, background-color, position, padding, font, text- align to name a few.
- **Inheritance and Cascades in CSS**
  - In CSS, a child element inherits the styles rules applied to its ancestor element.
  - Also there are several sources of style sheets, hence CSS follows W3C defined cascading order when applying style rules to elements.