

# Design and CSS

Cascading Style Sheets





Design and CSS

# Objectives

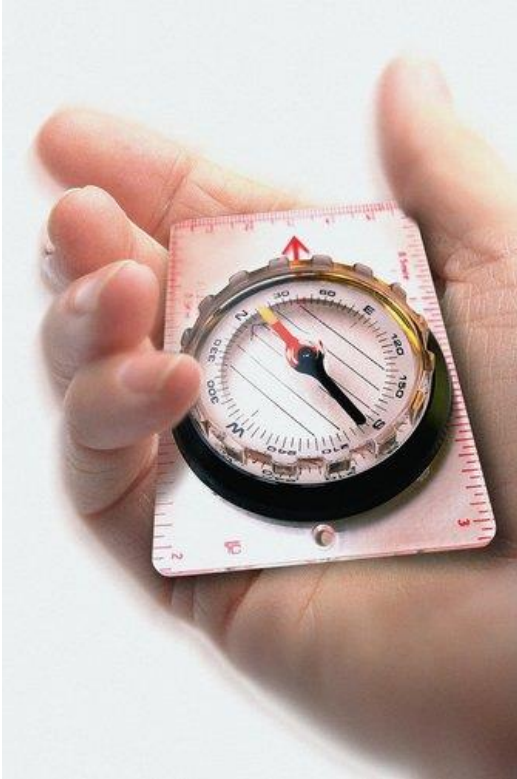
By completing this course you will be able to:

- Define what is CSS
- Use CSS in your websites



Design and CSS

# Course plan



- Introduction to CSS
- How to CSS
- Float & Position
- Display property
- Pseudo-Classes & Elements
- Directives
- Selector's Precedence

Design and CSS

# INTRODUCTION





# Before diving in...

- HTML, CSS and JavaScript have different tasks:
  - HTML:
    - Create the page structure
    - Define the content (pictures, text...)
  - CSS:
    - Makes your page beautiful
  - JavaScript:
    - Create interactions with user (buttons, validation, ...)
    - Create interactions with browser (redirect, set data, ...)
    - Look like any other functional language (loops, variables, ...)



# Before diving in...

- Combination of these makes the Web





# An analogy

- HTML:

YAMAHA  
MX49



# An analogy

- CSS:







# An analogy

- JavaScript:





# Definition

- CSS: Cascading Style Sheet
- Language really different from (x)HTML
- Allows to design a HTML document
- Currently CSS 3



# Advantages

- Centralizes and factories the layout elements in an external file
- Separates design and content
- Simplifies maintenance of Web pages
- Reduces page size



# Possibilities

- Define document layout in acting on elements like:
  - Background (color and/or image)
  - Font style (color, weight, family...)
  - Borders and margins
- Properties can be set on:
  - Element type (<h1>, <p>, <ul>, <a>, ...)
  - Element class (class attribute)
  - Identified element (id attribute)
  - Etc...



# Warning about CSS

Syntax is very easy to understand.

You need to focus on **vocabulary**, because CSS has many properties with even more values.

Your CSS level will mostly rely on practice and experiments, because it's a **descriptive language**. No variables, no loops, no conditional operator.

Design and CSS

# HOW TO CSS



# Splitting up the page

```
<div>...</div>
```

- Defines a section (division) in the document
  - *Block* tag
  - Used to separate each important part
  - Facilitates styling with CSS
  - No special behavior

## Splitting up the page

...

```
<div id="banner">
```

```
  
```

```
</div>
```

**Banner**

```
<nav>
```

```
<ul>
```

```
  <li><a href="home.html">Home</a></li>
```

```
  <li><a href="forum.html">Forum</a></li>
```

```
</ul>
```

```
</nav>
```

**Menu**

```
<header>
```

```
  <h1>Welcome on my Website</h1>
```

```
</header>
```

**Main content**

```
<footer>
```

```
  Copyright WebDev &#169; 2012
```

```
</footer>
```

**Footer**

...





How to CSS

# Without CSS



- [Home](#)
- [Forum](#)

## Welcome on my Website

Copyright WebDev © 2012



Programming in CSS

# Display of the previous page with CSS





# Where to place CSS?

- HTML Style / “inline” style

```
<span style="color: red;">Red text... </span>
```

Red text...

- Discouraged method
  - Hard to maintain
  - Targets only one tag



# Where to place CSS?

- Internal style sheet

```
<head>  
  <style type="text/css">  
    p { color: blue; }  
  </style>  
</head>
```

- OK to use, but:
  - CSS code usable in only one page
  - Increase page size and length



# Where to place CSS?

- Extern style sheet

```
<!-- Place this code in the head tag -->
<link href="main.css" media="all" rel="stylesheet"
      type="text/css" />
```

- Best way to place CSS
  - Usable on many pages
  - Easier to maintain
  - Separation of concerns





# Syntax

- selector {  
    property: value;  
    ...  
}

```
p {  
    background-color: white;  
    color: black;  
    font-family: Verdana, "Sans serif";  
}
```



# Syntax

- Selectors can be grouped

```
h1 { background-color: white; color: red; }  
p, h1 { font-family: Verdana, "Sans serif"; }
```

- Style are applied in the definition order
- In case of conflicts, the last property “overrides” the previous ones



# Comments

- Syntax: `/* ... */`
  - It's not possible to use `//`

```
/* Document font family */  
body { font-family: Verdana, "Sans serif"; }
```





# Selectors

- CSS selectors overview

Selector	Targets	Example
<b>*</b>	All elements	<code>* { ... }</code>
<b>E</b>	All <b>E</b> elements	<code>div { ... }</code>
<b>E F</b>	All elements downward <b>F</b> of <b>E</b>	<code>div p { ... }</code>
<b>E &gt; F</b>	All elements <b>F</b> son of <b>E</b>	<code>div &gt; p { ... }</code>
<b>#myId</b>	The element with id « myId »	<code>#banner { ... }</code>
<b>E#myId</b>	The <b>E</b> element with id « myId »	<code>img#logo { ... }</code>



# Selectors

- CSS selectors overview

Selector	Targets
<b>.myClass</b>	All elements having the class « myClass »
<b>E.myClass</b>	All E elements having the class « myClass »



# Attribute *class*

- Apply a style on a set of defined tags
  - This CSS code:

```
h1.section { background-color: red; }
```

- Can be associated with any h1 tag:

```
<h1 class="section">My Title</h1>
```



# Attribute *id*

- Apply a style on **only one and unique** tag
  - This CSS code:

```
#MyObjectId { background-color: red; }
```

- Can be associated with any tag:

```
<div id="MyObjectId">My Text</div>
```



# Colors

- Can be defined in several ways

Method	Restriction	Example
<b>colorname</b>	Already CSS build-in colors	red
<b>rgb(r, g, b)</b>	$0 \leq (r, g, b) \leq 255$	rgb(255, 0, 0)
<b>rgb(r%, g%, b%)</b>	$0 \leq (r, g, b) \leq 100$	rgb(100%, 0, 0)
<b>#rrggbb</b>	$00 \leq (rr, gg, bb) \leq FF$	#FF0000
<b>#rgb</b>	$0 \leq (r, g, b) \leq F$	#F00



# Size unit

- Available measure units:
  - %
  - in, cm, mm
  - em, px
  - pt, pc
  - px

```
body { font-size: 1.2em; }
```



# Attributes: Text

- Important text properties:

Property	Description
<b>text-align</b>	Align to left, center or right
<b>text-decoration</b>	<u>Underline</u> text
<b>text-indent</b>	Move text to right or left
<b>text-transform</b>	<b>U</b> ppercase, <b>l</b> owercase, <b>C</b> apitalize <b>E</b> very <b>S</b> ingle <b>W</b> ord
<b>color</b>	What we just saw about colors
<b>line-height</b>	Set height of each text line
<b>letter-spacing</b>	Set some space between each letter



# Attributes: Text

- Important text properties:

Property	Description
<b>font-size</b>	Set size of text
<b>font-family</b>	Define the used fonts. Can be several for a fallback
<b>font-weight</b>	Set text to <b>bold</b>
<b>font-style</b>	Set text to <i>italic</i>
<b>text-shadow</b>	Create some shadows on texts





# Attributes: Text

```
p {  
  color: orange;  
  text-align: center;  
  text-decoration: underline;  
  text-transform: uppercase;  
  letter-spacing: 10px;  
  font-style: italic;  
  font-weight: bold;  
  line-height: 1.4em;  
  font-family: "Serif";  
  font-size: 1.2em;  
}
```

CSS  
IS NOT  
COUNTER STRIKE SOURCE !



How to CSS

# Attribute: Background

- background-color
- background-image
- background-repeat
- background-attachment
- background-position



# Attribute: Background

```
body {  
  background-color: black;  
  background-image: url('kaleidoscope.jpg');  
  background-repeat: repeat-x;  
  background-attachment: fixed;  
  background-position: 0% 50%;  
}
```





# Attribute: *margin*

Specify outside margins

– *margin-position*:

- margin-top
- margin-right
- margin-bottom
- margin-left





# Attribute: *margin*

```
p { margin: 40px; }
```

- Without style:

SUPINFO est l'une des rares écoles d'informatique françaises à vocation professionnalisante qui soit à la fois reconnue par l'Etat par décret du 10 janvier 1972, qui délivre un titre homologué par l'Etat par décret au niveau I (BAC+5, Master, Ingénieur).

SUPINFO travaille régulièrement sur la baisse des frais de scolarité qui passent pour 2006 à 5990 euros tout compris quelle que soit l'année d'étude, en baisse pour la 3ème année consécutive. En plus de cette volonté d'accessibilité financière des études, SUPINFO est fière de constater que, la sélection se faisant à l'admission dans l'école et pas en cours d'études, plus de 90% des étudiants qui sont admis en première année du cycle préparatoire, réussissent leur cursus complet, 5 ans plus tard.

- With style:

SUPINFO est l'une des rares écoles d'informatique françaises à vocation professionnalisante qui soit à la fois reconnue par l'Etat par décret du 10 janvier 1972, qui délivre un titre homologué par l'Etat par décret au niveau I (BAC+5, Master, Ingénieur).

SUPINFO travaille régulièrement sur la baisse des frais de scolarité qui passent pour 2006 à 5990 euros tout compris quelle que soit l'année d'étude, en baisse pour la 3ème année consécutive. En plus de cette volonté d'accessibilité financière des études, SUPINFO est fière de constater que, la sélection se faisant à l'admission dans l'école et pas en cours d'études, plus de 90% des étudiants qui sont admis en première année du cycle préparatoire, réussissent leur cursus complet, 5 ans plus tard.

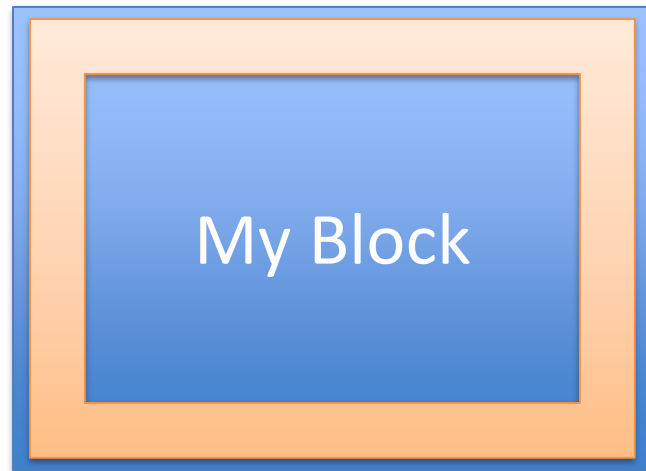


# Attribute: *padding*

Specify inside margins

— *padding-position*:

- padding-top
- padding-right
- padding-bottom
- padding-left





# Attribute: *padding*

```
th { padding: 10px 40px 10px 40px; }  
td { padding: 20px; text-align: right; }
```

- Without style:

Nom du jeu	Nombre de cartes en main
Texas Hold'em	2
Omaha	4

- With style:

Nom du jeu	Nombre de cartes en main
Texas Hold'em	2
Omaha	4



# Attribute: *border*

- Usable on any element
  - border-width
  - border-style
  - border-color
  - border-collapse
  - border-*position*
    - border-top
    - border-bottom
    - border-left
    - border-right





## Attribute: *border*

```
table {  
    border: 5px solid green;  
    border-collapse: collapse;  
}  
  
th {  
    padding: 10px 40px 10px 40px;  
    border-bottom: 3px dotted red;  
    border-left: 1px solid blue;  
}  
  
td {  
    padding: 20px;  
    text-align: right;  
    border-left: 1px solid blue;  
    border-bottom: 1px solid blue;  
}
```

Nom du jeu	Nombre de cartes en main
Texas Hold'em	2
Omaha	4



# Attributes: *list*

- list-style-type
- list-style-position
- list-style-image

```
ul {  
    list-style-type: disc;  
    list-style-position: outside;  
    list-style-image: url('orange.gif');  
}
```



Pomme



Banane



Pêche



Mirabelle



# Debug in your browser

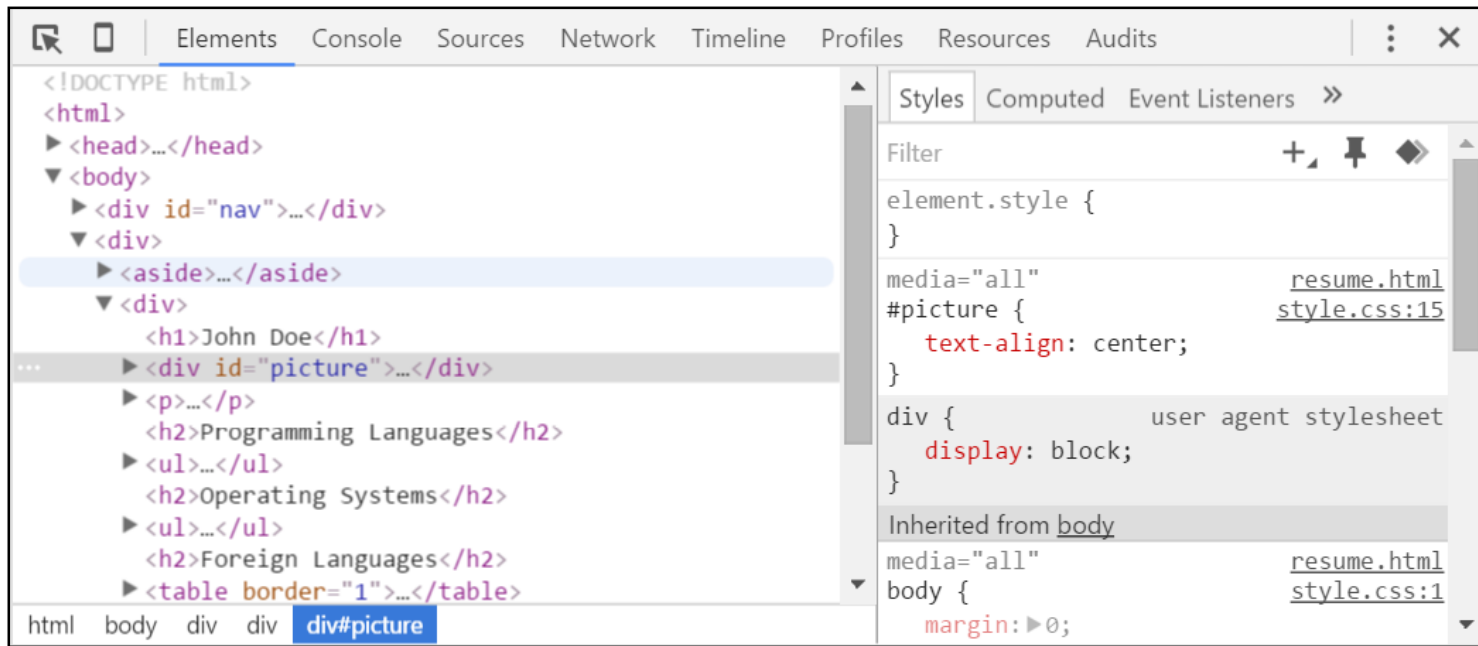
Before we start off, how to debug your CSS!

- You can use your favorite browser to debug
  - Right click on any element and select « Inspect element »
  - Any CSS rule can be edited
  - You can even add CSS in your browser
- Be careful, if you refresh, all modification will be lost 😊



# Debug in your browser

- Example in Chrome:



Design and CSS

# FLOAT & POSITION



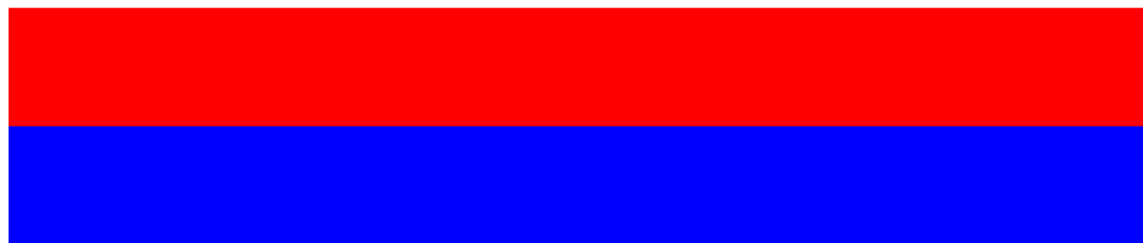


# Understand the flow

- Simple structure:

```
<div>  
  <p>Lorem ipsum dolor sit amet</p>  
  <div style="height: 50px; background: red;"></div>  
  <div style="height: 50px; background: blue;"></div>  
</div>
```

Lorem ipsum dolor sit amet



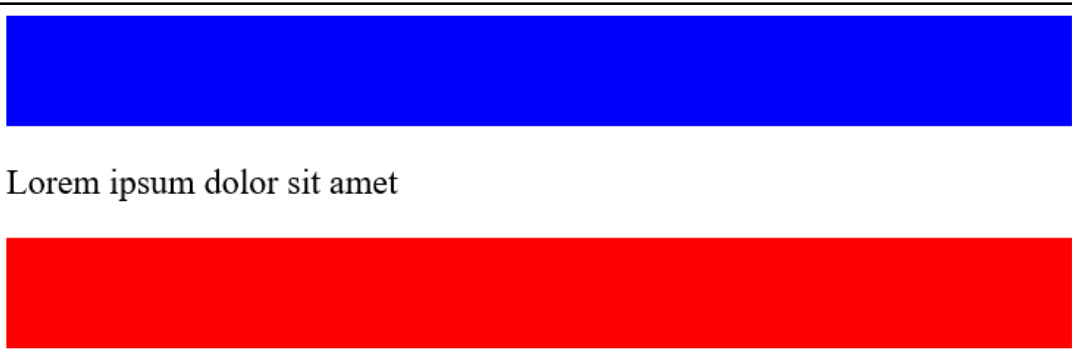


# Understand the flow

- The flow is **predictable**
  - If we change the order...

```
<div>  
  <div style="height: 50px; background: blue;"></div>  
  <p>Lorem ipsum dolor sit amet</p>  
  <div style="height: 50px; background: red;"></div>  
</div>
```

- It changes!

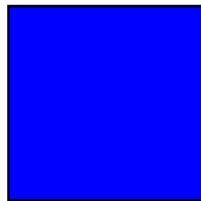




# Examples – Normal

```
<div>
  <div class="block"></div>
  <p>...</p>
  <p>...</p>
</div>
```

```
.block {
  height: 98px; width: 98px;
  background: blue;
  border: 1px black solid;
}
```



Lorem ipsum dolor sit amet, consectetur molestie orci. Vivamus cursus metus q metus dapibus. Nullam maximus preti Proin laoreet dictum consectetur. Sed r gravida vel. Vivamus in consectetur ni convallis.

Praesent scelerisque magna non conse porttitor leo ultrices rutrum eget eget e nec bibendum dui commodo sit amet. l ultricies dapibus. Aenean cursus vitae vitae sapien bibendum, pellentesque vi arcu. Pellentesque maximus volutpat p hendrerit nulla et, volutpat magna. Pha in, posuere bibendum ante. Morbi curs

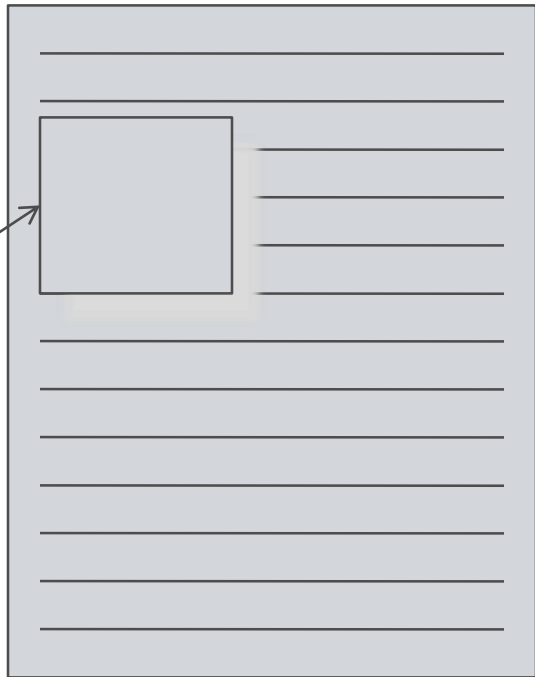




# Attribute: *float*

- Determines floating element
- Possible values:
  - left
  - right
  - none
  - inherit

Floating block



```
.menu { float: left; }
```



# Examples – Float

```
<div>
  <div class="block"></div>
  <p>...</p>
  <p>...</p>
</div>
```

```
.block {
  height: 98px; width: 98px;
  background: blue;
  border: 1px black solid;
  float: right;
}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin quis molestie orci. Vivamus cursus metus quis augue euismod, eu efficitur metus dapibus. Nullam maximus pretium purus, nec feugiat nisl feugiat et. Proin laoreet dictum consectetur. Sed maximus mollis leo, sed efficitur erat gravida vel. Vivamus in consectetur nisl. Donec ornare ullamcorper convallis.



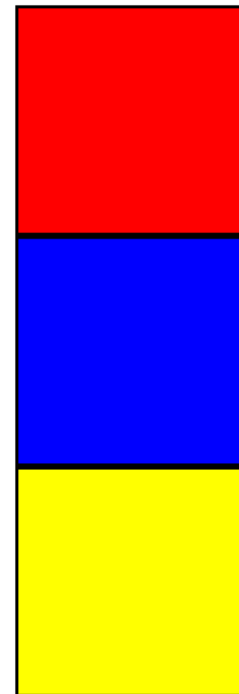
Praesent scelerisque magna non consequat semper. Duis non neque porttitor leo ultrices rutrum eget eget elit. Quisque suscipit euismod velit, nec bibendum dui commodo sit amet. Nunc facilisis risus eu massa ultricies dapibus. Aenean cursus vitae odio id facilisis. Duis luctus erat vitae sapien bibendum, pellentesque viverra eros tempor. Etiam sed dictum arcu. Pellentesque maximus volutpat placerat. Nulla a risus egestas. hendrerit nulla



# Examples – Multi blocks

```
<div class="block red"></div>
<div class="block blue"></div>
<div class="block yellow"></div>
```

```
.block {
  height: 98px; width: 98px;
  border: 1px black solid;
}
.red { background: red; }
.blue { background: blue; }
.yellow { background: yellow; }
```

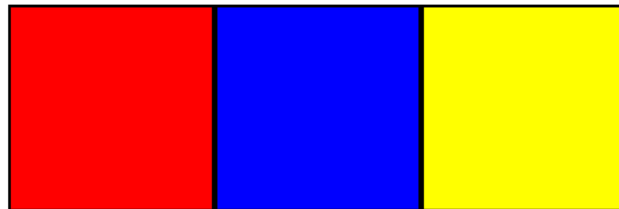




# Examples – Multi blocks float left

```
<div class="block red"></div>
<div class="block blue"></div>
<div class="block yellow"></div>
```

```
.block {
  height: 98px; width: 98px;
  border: 1px black solid;
  float: left;
}
.red { background: red; }
.blue { background: blue; }
.yellow { background: yellow; }
```

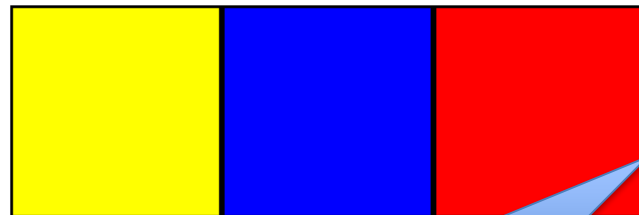




# Examples – Multi blocks float right

```
<div class="block red"></div>  
<div class="block blue"></div>  
<div class="block yellow"></div>
```

```
.block {  
  height: 98px; width: 98px;  
  border: 1px black solid;  
  float: right;  
}  
.red { background: red; }  
.blue { background: blue; }  
.yellow { background: yellow; }
```



Browser right border



Why are  
colors “reversed”?



## Attribute: *position*

- Move an element out of the flow
- Possible values:
  - static: element is at his normal position
  - relative: element position is relative to its normal position
  - absolute: element position is relative to parent position origin
  - fixed: like absolute but the element follows user scrolling

```
.menu { position: fixed; }
```



# Examples – Without position

static

```
.static {  
  position: static;  
}
```

CSS

```
<div class="static">
```

`static` est la valeur par défaut de tous les éléments. Un élément avec `position: static;` n'est positionné d'aucune manière spéciale.

Un élément `static` est dit *non positionné* et un élément avec une propriété `position` ayant une valeur autre que `static` est dit *positionné*.



# Examples – Relative position

## relative

```
.relative1 {  
  position: relative;  
}  
.relative2 {  
  position: relative;  
  top: -20px;  
  left: 20px;  
  background-color: white;  
  width: 500px;  
}
```

CSS

```
<div class="relative1">
```

relative se comporte de la même façon que static sauf si vous ajoutez quelques propriétés en plus.

```
<div class="relative2">
```

Ajouter les propriétés `top`, `right`, `bottom` et `left` à un élément positionné en relative va le placer ailleurs que sa position normale. Le reste du contenu ne sera pas ajusté pour prendre la place dans l'espace laissé par l'élément.

```
</div>
```





# Examples – Absolute position

`<div class="relative">`

Cet élément est positionné en relative. S'il était positionné en `position: static;` son élément enfant positionné en absolute s'échapperait et serait positionné de manière relative au corps du document.

`<div class="absolute">`

Cet élément est positionné en absolute. Sa position est relative à son parent.

`</div>`

`</div>`

```
.relative {  
  position: relative;  
  width: 600px;  
  height: 400px;  
}  
.absolute {  
  position: absolute;  
  top: 120px;  
  right: 0;  
  width: 300px;  
  height: 200px;  
}
```

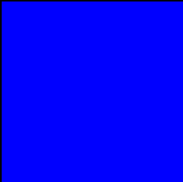
CSS



# Examples – Position

```
<div>
  <div class="block"></div>
  <p>...</p>
  <p>...</p>
</div>
```

```
.block {
  height: 98px; width: 98px;
  background: blue;
  border: 1px black solid;
  position: absolute;
}
```



olor sit amet, consectetur  
Proin quis molestie orci.  
s metus quis augue euismod, eu  
dapibus. Nullam maximus  
nec feugiat nisl feugiat et.  
Prom laoreet dictum consectetur. Sed  
maximus mollis leo, sed efficitur erat gravida  
vel. Vivamus in consectetur nisl. Donec  
ornare ullamcorper convallis.

Praesent scelerisque magna non consequat  
semper. Duis non neque porttitor leo ultrices  
rutrum eget eget elit. Quisque suscipit  
euismod velit, nec bibendum dui commodo sit  
amet. Nunc facilisis risus eu massa ultricies  
dapibus. Aenean cursus vitae odio id facilisis.  
Duis luctus erat vitae sapien bibendum,  
pellentesque viverra eros tempor. Etiam sed  
dictum arcu. Pellentesque maximus volutpat  
placemat. Nulla a risus egestas, hendrerit nulla  
et, volutpat magna. Phasellus nibh tellus,  
suscipit in quam in, posuere bibendum ante.  
Morbi cursus blandit vulputate.



# Examples – Fixed position

```
.fixed {  
  position: fixed;  
  bottom: 0;  
  right: 0;  
  width: 200px;  
  background-color: white;  
}
```

CSS

```
<div class="fixed">
```

Bonjour ! Ne faites pas  
attention à moi pour  
l'instant.

```
</div>
```

Design and CSS

# DISPLAY PROPERTY





# What is « display »?

- Remember block and inline tags in HTML?
  - Block tags: `<p>`, `<div>`, `<hn>`, ...
  - Inline tags: `<span>`, `<em>`, `<strong>`, ...
- Note this is default behavior, it can be overridden



# What is « display »?

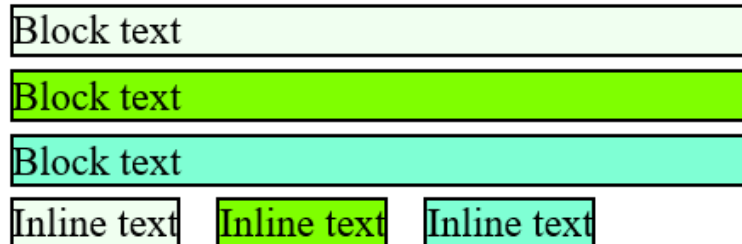
- Difference between block and inline:
  - A block tag will take all possible width
  - An inline tag will take the smallest possible width
  - **You can't set an inline tag height and width**



# Overriden span display example

```
<span class="block one">Block text</span>
<span class="block two">Block text</span>
<span class="block thr">Block text</span>
<span class="one">Inline text</span>
<span class="two">Inline text</span>
<span class="thr">Inline text</span>
```

```
span { border: 1px black solid;
        margin: 5px; }
.block { display: block; }
.one { background: honeydew; }
.two { background: chartreuse; }
.thr { background: aquamarine; }
```





# Many display values

- Most used display values:

Display value	Description	Example tags
<b>Block</b>	For every container element	div, p, ol, ul, hr...
<b>Inline</b>	For every content element	span, strong, em...
<b>Inline-Block</b>	Inline, access to block properties*	img, button, textarea...
<b>None</b>	The element is hidden	script, style
<b>Table-cell</b>	Acts like a table-cell	td
<b>List-item</b>	Acts like a list-item	li





# INLINE / INLINE-BLOCK / BLOCK

**display: inline**

Ach, was muß man oft von bösen  
Kindern hören oder lesen!  
Wie zum Beispiel hier von diesen  
Welche Max und Moritz hießen.  
Die, anstatt durch weise Lehren  
Sich zum Guten zu bekehren,  
Oftmals noch darüber lachten  
Und sich heimlich lustig machten.

**display: block**

Ach, was muß man oft von bösen  
Kindern hören oder lesen!  
Wie zum Beispiel hier von diesen  
Welche  
Max und Moritz  
hießen.  
Die, anstatt durch weise Lehren  
Sich zum Guten zu bekehren,  
Oftmals noch darüber lachten  
Und sich heimlich lustig machten.

**display: inline-block**

Ach, was muß man oft von bösen  
Kindern hören oder lesen!  
Wie zum Beispiel hier von diesen  
Welche Max und Moritz hießen.

Die, anstatt durch weise Lehren  
Sich zum Guten zu bekehren,  
Oftmals noch darüber lachten  
Und sich heimlich lustig machten.

Difference of supported styles as summary:

- **inline:** only `margin-left`, `margin-right`, `padding-left`, `padding-right`
- **inline-block:** `margin`, `padding`, `height`, `width`

Design and CSS

# **PSEUDO-CLASSES & ELEMENTS**



# Target something else

- Pseudo-classes/elements target specific cases:
  - When you hover an element
  - When you click on an element
  - When an element is odd/even son of its parent
  - When a link has already been clicked
  - ...
- Global syntax : **element:pseudoclass**



# Pseudo-classes

- Pseudo-classes target external factors
  - Can be in relation with the element position in HTML
  - Can be in relation with the client behavior
- Global syntax : **element:pseudoclass**



# Pseudo-classes

- CSS pseudo-classes overview

Selector	Targets
<b>E:link</b>	All <b>E</b> elements when it's a link
<b>E:visited</b>	All <b>E</b> link elements when the user already navigated to
<b>E:hover</b>	All <b>E</b> elements hovered by the user's cursor
<b>E:active</b>	The <b>E</b> element whose clicked by the user right now
<b>E:nth-child</b>	When an element is <b><i>n</i></b> <sup>th</sup> child

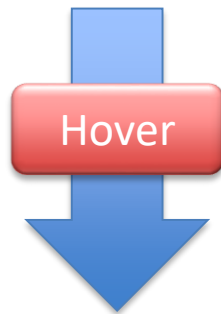


# Pseudo-classes example

```
<a href="page2.html">My second page</a>
```

```
a:link, a:visited, a:active {  
    text-decoration: none;  
    color: red;  
}  
a:hover {  
    text-shadow: 1px 1px 5px black;  
    color: black;  
}
```

My second page



My second page



# Pseudo-classes example

```
<table>
  <tr> <th>Col1</th>  <th>Col2</th>  </tr>
  <tr> <td>ValA1</td> <td>ValA2</td> </tr>
  <tr> <td>ValB1</td> <td>ValB2</td> </tr>
  <tr> <td>ValC1</td> <td>ValC2</td> </tr>
</table>
```

Col1	Col2
ValA1	ValA2
ValB1	ValB2
ValC1	ValC2
ValD1	ValD2

```
table, td { border: 1px black solid;
             border-collapse: collapse; }
th, td    { padding: 5px; }
tr:nth-child(odd) td { background: #CCC; }
tr:nth-child(even) td { background: #888; }
```



# Pseudo-elements

- Pseudo-classes style parts of the document
  - First letter of a paragraph
  - Style before/after an element
- Global syntax : **element::pseudoelement**





# Pseudo-elements

- CSS pseudo-classes overview

Selector	Targets
<b>E::first-letter</b>	First letter of all <b>E</b> elements
<b>E::first-line</b>	First line of all <b>E</b> elements
<b>E::before</b>	A "virtual" generated element before every <b>E</b> elements
<b>E::after</b>	A "virtual" generated element after every <b>E</b> elements



# Pseudo-elements example

```
<p>Lorem ipsum...</p>
```

```
p::first-letter {  
  font-size: 24px;  
  color: whitesmoke;  
  text-shadow: 0px 0px 2px black,  
              1px 1px 2px black;  
  letter-spacing: 2px;  
}
```

Lorem ipsum  
ultricies felis. S  
Nunc sagittis, d  
tellus enim sed  
lacinia ac mass  
porttitor eu, ten  
congue Morbi



# Pseudo-elements example

```
<p>Lorem ipsum...</p>
```

```
p::first-line {  
  font-size: 24px;  
  color: whitesmoke;  
  text-shadow: 0px 0px 2px black,  
              1px 1px 2px black;  
  letter-spacing: 2px;  
}
```

Lorem i  
adipiscing elit  
volutpat vitae  
tortor gravida  
magna quis co  
ligula tellus, v



# Pseudo-elements example

`<h1>Ace of Spades</h1>`

♠ Ace of Spades ♦

```
h1 { color: grey; }
```

```
h1::before { content: '\2660';  
              margin-right: 5px; }
```

```
h1::after { content: '\2666';  
            margin-left: 5px;  
            color: red; }
```

Design and CSS

# DIRECTIVES





# Inclusion

- Instead of a second `<link/>` tag, you can also include another stylesheet with **@import** directive

```
@import "style.css"
```

```
@import "http://awebiste.com/style/s5.css"
```

```
@import url("http://awebiste.com/style/s5.css")
```



# Supports

- Attribute media of the <link> tag
- Allows the support of printers, Braille, etc.

```
@media print {  
  p {  
    font-family: serif;  
    font-size: 12pt;  
  }  
}
```

- Complete supported list:  
<http://www.w3.org/TR/CSS2/media.html>



# @media Use case

- You know, when you want to print a web page
  - Or save it as a PDF file...
  - ...which is perfect for a resume
- Remove unnecessary style

```
@media print {  
  #navigation { display: none; }  
  body { background: none; }  
  table { background: none; }  
}
```





# @media print example

- After a Ctrl+P (Print) shortcut...
  - Magic operates!

