

# Talk about CSS

Abraham Kuri Vargas  
@kurrenn

---



# Tabla de contenidos

---

01 Fundamentos

02 Herencia y especificidad

03 Box Model

04 Mejores prácticas

05 Pseudo selectores

06 ¿Dónde aprendo más?

.3-style-ways {  
}



# 3 style ways

- \* Estilo inline
- \* En etiqueta <head>
- \* Externa <link>



# 3 style ways

## \* Estilo inline

```
<h1 style="color: #98C7D4;">Hello World!</h1>
```



# 3 style ways

\* En etiqueta <head>

```
<head>
  <style>
    h1 {
      color: #98C7D4;
    }
  </style>
</head>
```



# 3 style ways

## \* External <link>

```
<head>  
  <title>Untitled</title>  
  <link rel="stylesheet" href="/css/master.css" type="text/css">  
</head>
```



```
.selectores {  
}
```



# Selectores

\* Por elemento

```
<h1 class="welcome" id="header">Hello World!</h1>
```




```
h1 {  
  color: #ABA4AC;  
  margin-bottom: 8px;  
}
```

# Selectores

\* Por clase

```
<h1 class="welcome" id="header">Hello World!</h1>
```




```
.welcome {  
  color: #ABA4AC;  
  margin-bottom: 8px;  
}
```

# Selectores

\* Por ID

```
<h1 class="welcome" id="header">Hello World!</h1>
```

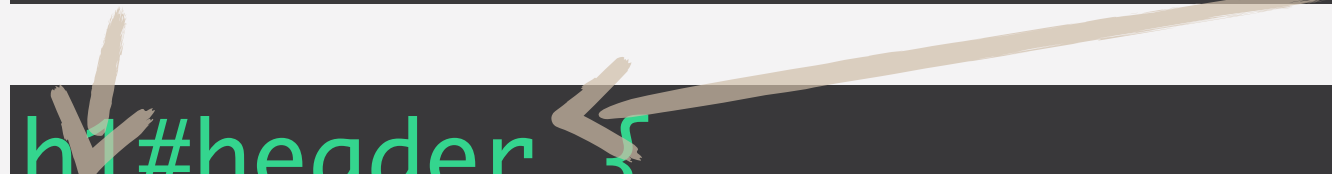


```
#header {  
  color: #ABA4AC;  
  margin-bottom: 8px;  
}
```

# Selectores

## \* Compuesto

```
<h1 class="welcome" id="header">Hello World!</h1>
```



A diagram with two arrows pointing to the CSS selectors. One arrow points from the 'h1' in the HTML code to the 'h1' in the first CSS selector. The other arrow points from the 'id="header"' in the HTML code to the '#header' in the first CSS selector.

```
h1#header {  
  color: #ABA4AC;  
  margin-bottom: 8px;  
}
```

```
h1#welcome {  
  color: #ABA4AC;  
  margin-bottom: 8px;  
}
```

```
.en-cascada {  
}
```

# Orden en cascada

\* La posición del código en el documento es importante.

```
.welcome {  
  color:#ABA4AC;  
  margin-bottom: 8px;  
}
```

```
.welcome {  
  color:#FFFFFF;  
}
```



*La segunda definición sustituye a la primera*

# Orden en cascada

Las propiedades se combinan cuando no chocan entre si

```
.welcome {  
  color:#ABA4AC;  
  margin-bottom: 8px;  
}
```

```
.welcome {  
  width:100px;  
}
```



```
.welcome {  
  color:#ABA4AC;  
  margin-bottom: 8px;  
  width:100px;  
}
```

# Tabla de contenidos

---

01 Fundamentos

02 Herencia y especificidad

03 Box Model

04 Mejores prácticas

05 Pseudo selectores

06 ¿Dónde aprendo más?



```
.herencia {  
}
```

# Herencia

\* Los hijos heredan todos los estilos de su padre

```
<div class="intro">  
  <h1>Hello World!</h1>  
</div>
```

```
.intro {  
  color:#ABA4AC;  
}
```



*El h1 hereda el color de su padre*

# Herencia

\* Los selectores pueden anidarse para tener otro estilo

```
<div class="intro">  
  <h1>Hello World!</h1>  
</div>
```

```
.intro {  
  color:#ABA4AC;  
}  
.intro h1 {  
  color:#FFF;  
}
```

*El color del h1 será blanco*



.especificidad {  
}

# Especificidad

\* Como manejar la especificidad

```
<div class="intro" id="content" >  
  <h1>Hello World!</h1>  
</div>
```

```
#content {  
  color:#ABA4AC;  
}  
.intro {  
  color:#FFF;  
}
```

*Inline*



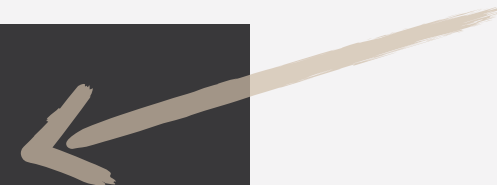
*Selector por ID*



*Selector por clase*



*Selector por elemento*



# Tabla de contenidos

---

01 Fundamentos

02 Herencia y especificidad

03 Box Model

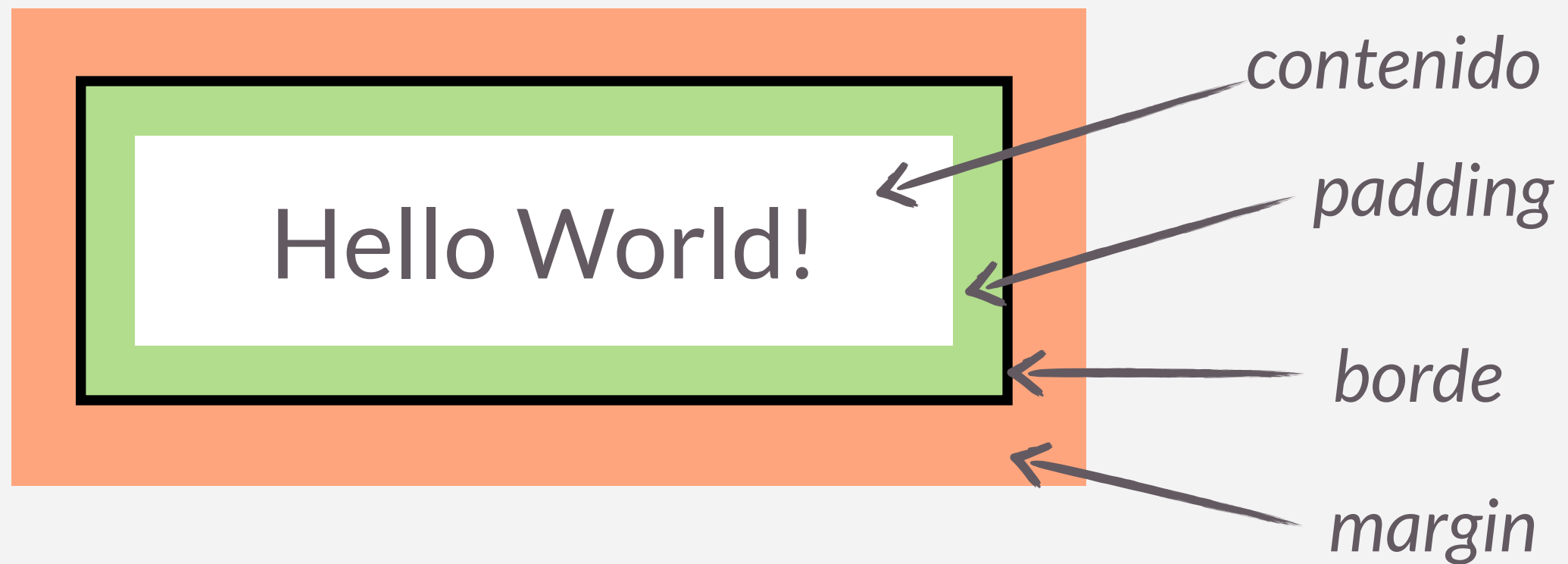
04 Mejores prácticas

05 Pseudo selectores

06 ¿Dónde aprendo más?

# Box Model

\* Es la representación de bloques en CSS



# Box Model

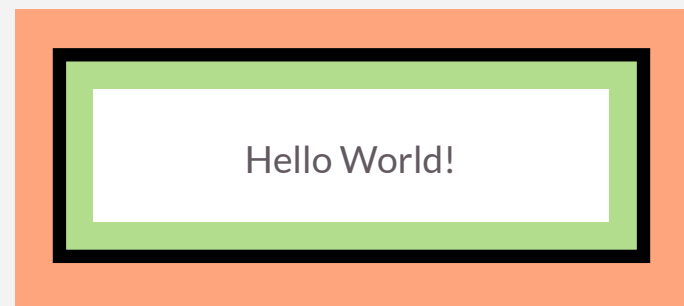
\* A simple example

```
#content {  
  border:5px solid #CCC;  
  padding-left:10px;  
  padding-right:5px;  
  width:100px;  
}
```

100px    *width*  
+ 15px   *padding*  
10px    *border*  

---

125px   *ancho total*



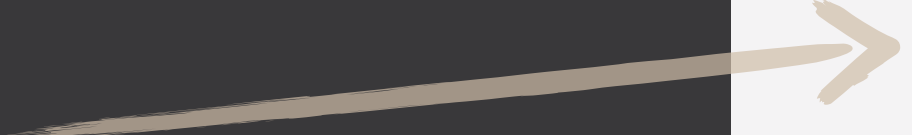


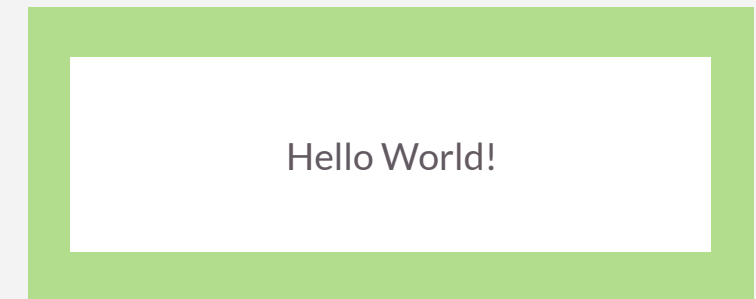
```
.padding {  
}
```

# Padding

\* Es espacio generado hacia el interior del elemento

```
#content {  
  padding-left:10px;  
  padding-right:5px;  
  padding-bottom:10px;  
  padding-top:5px;  
  width:100px;  
}
```


$$\begin{array}{rcl} & + & \begin{array}{ll} 100\text{px} & \text{width} \\ 15\text{px} & \text{padding} \end{array} \\ \hline 115\text{px} & \text{ancho total} & \end{array}$$

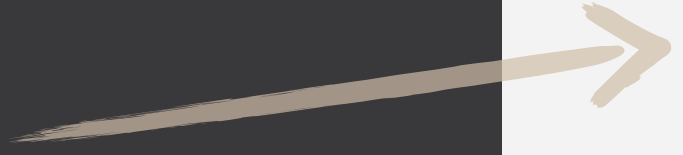


```
.border {  
}
```

# Border

\* Es un borde del contenedor

```
#content {  
  border: 5px solid #CCC;  
  width: 100px;  
}
```


$$\begin{array}{rcl} 100\text{px} & \text{width} & \\ + 10\text{px} & \text{border} & \\ \hline 115\text{px} & \text{ancho total} & \end{array}$$

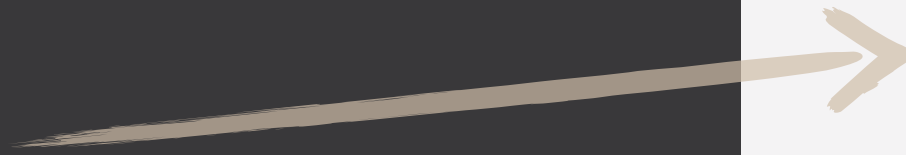
Hello World!

```
.margin {  
}
```

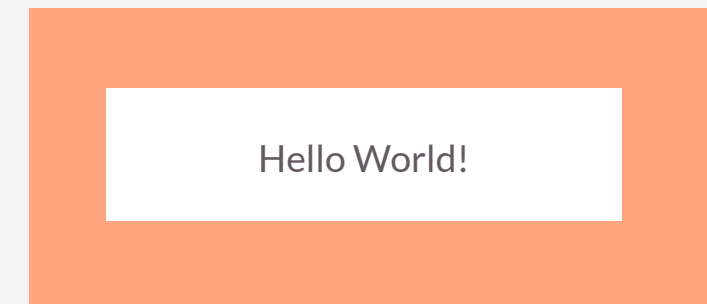
# Margin

\* Es un espacio que se genera hacia afuera del contenedor

```
#content {  
  margin-left:10px;  
  margin-right:5px;  
  width:100px;  
}
```



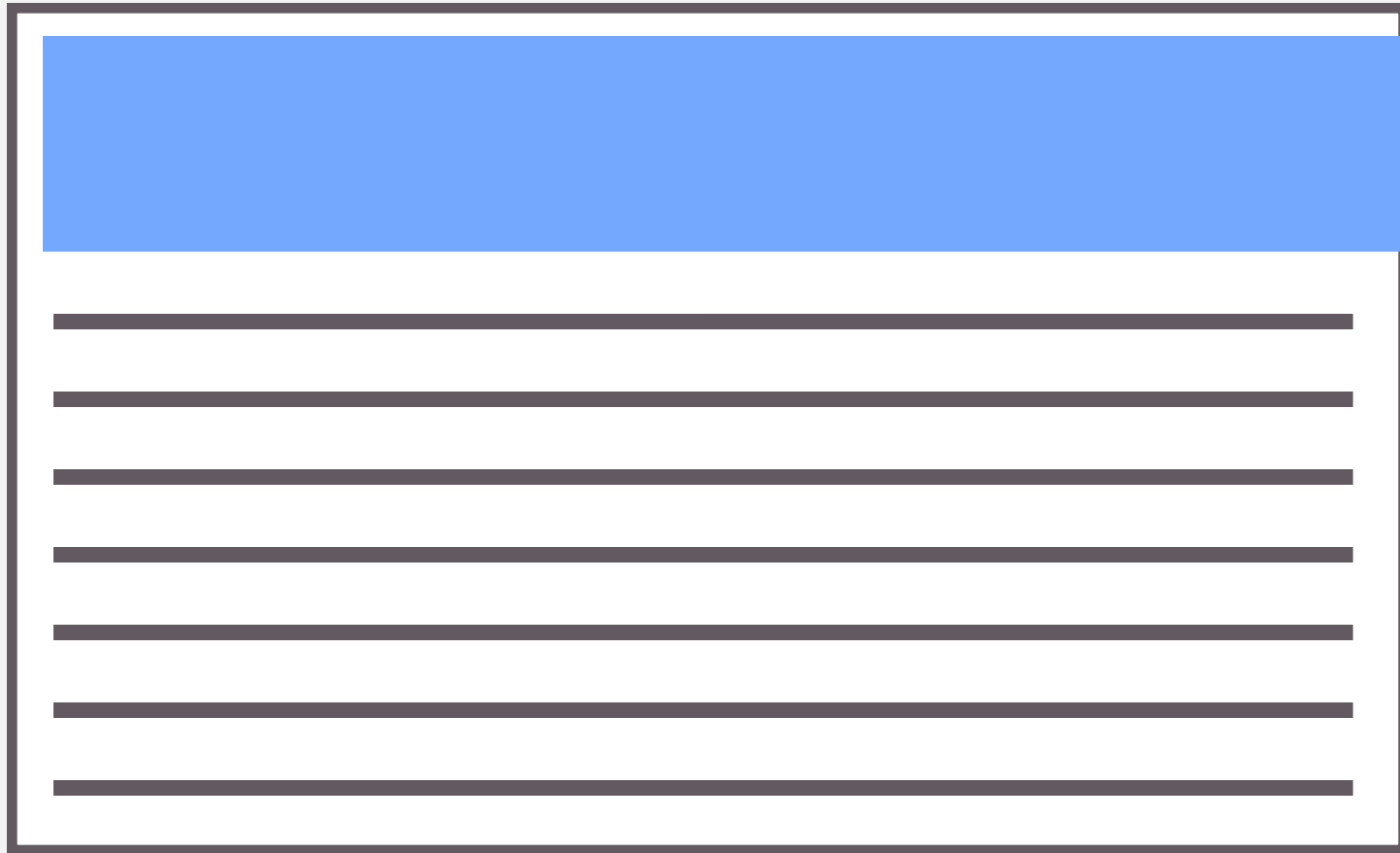
$$\begin{array}{rcl} 100\text{px} & \text{width} & \\ + 15\text{px} & \text{margin} & \\ \hline 115\text{px} & \text{ancho total} & \end{array}$$



```
.overflow {  
}
```

# Overflow

\* visible / auto / hidden / scroll

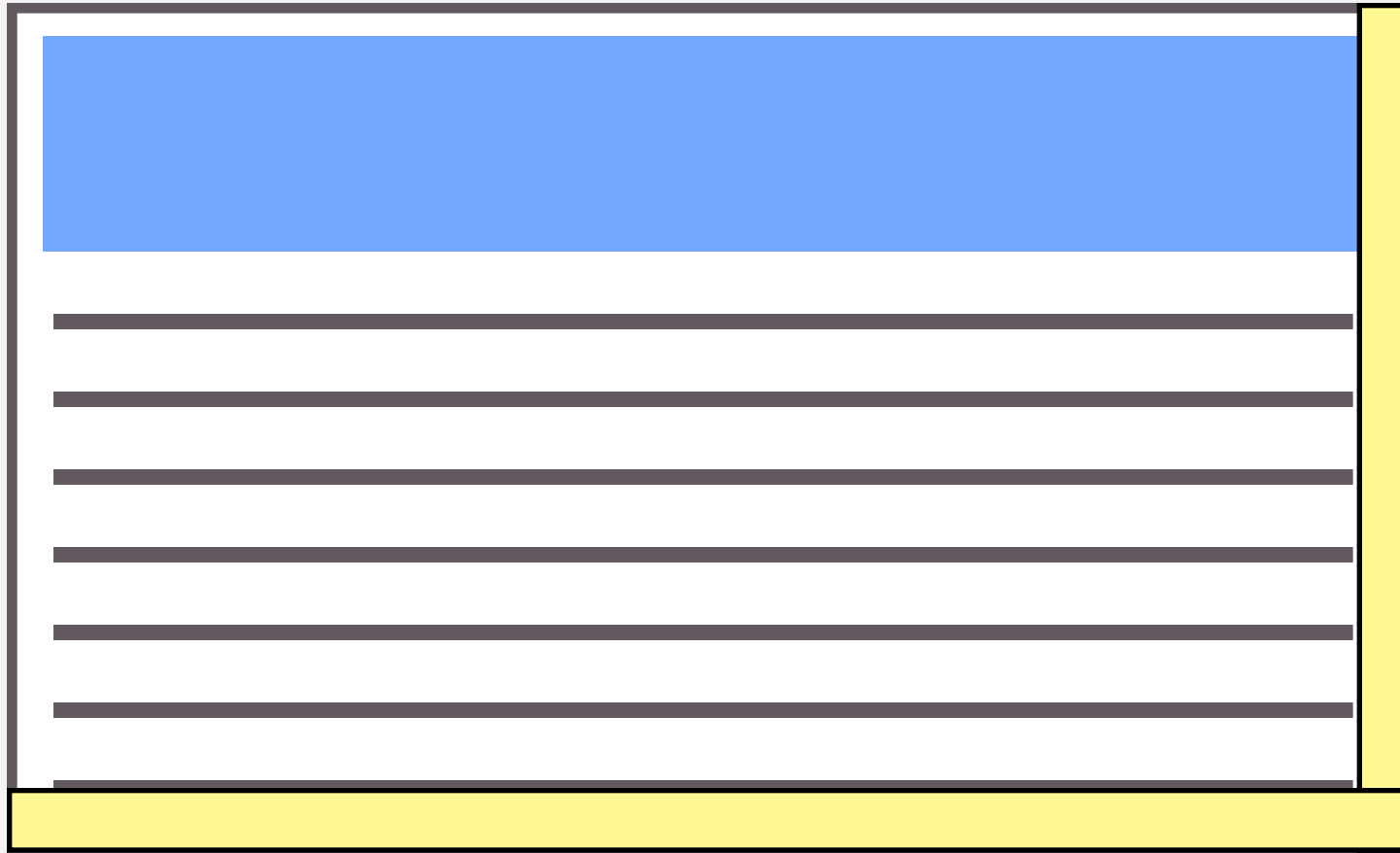


Visible - Es el valor predeterminado, lo que permite al contenido extenderse mas alla de sus limites.



# Overflow

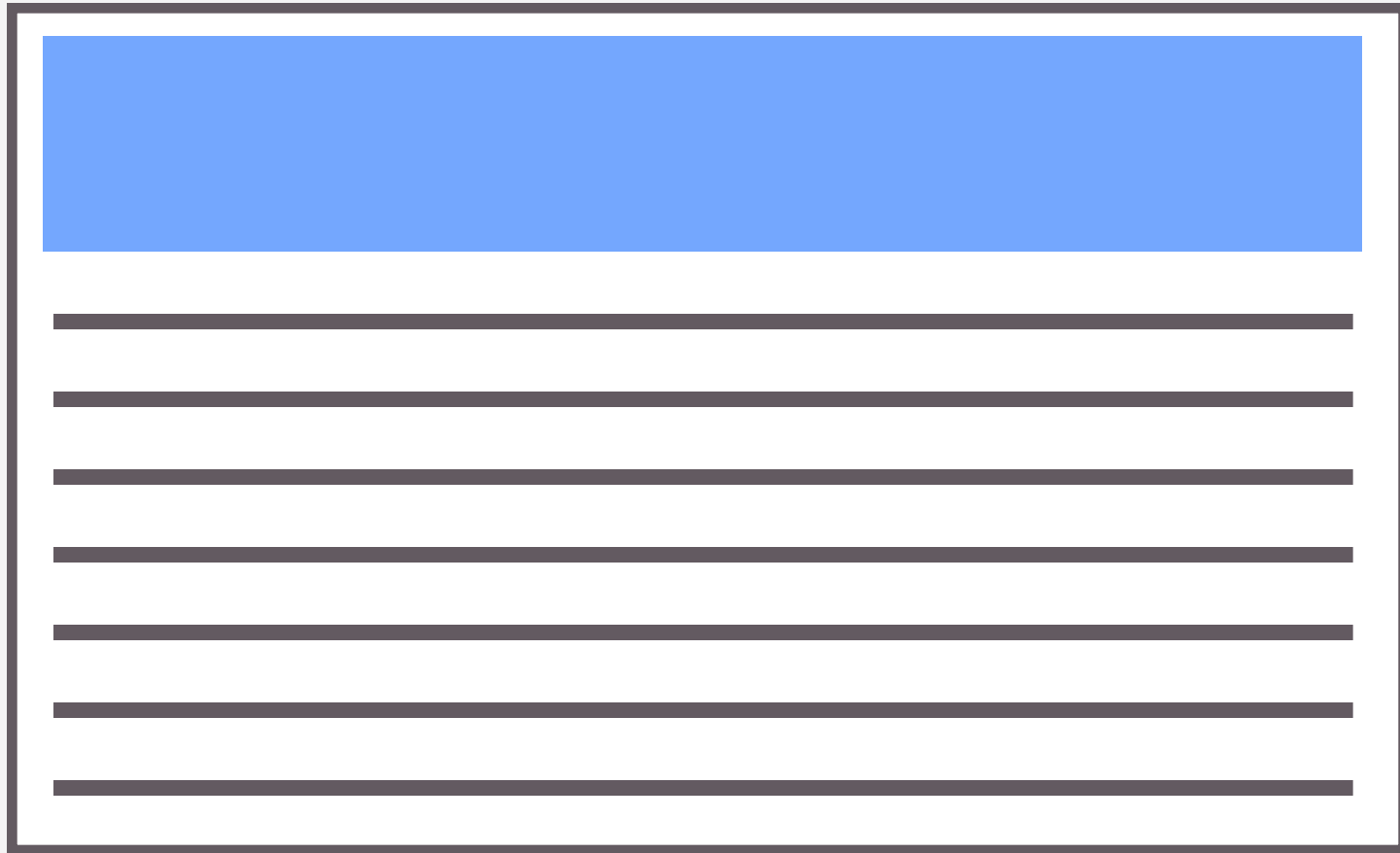
\* visible / auto / hidden / scroll



auto - Agrega barras de arrastre o scroll cuando es necesario.

# Overflow

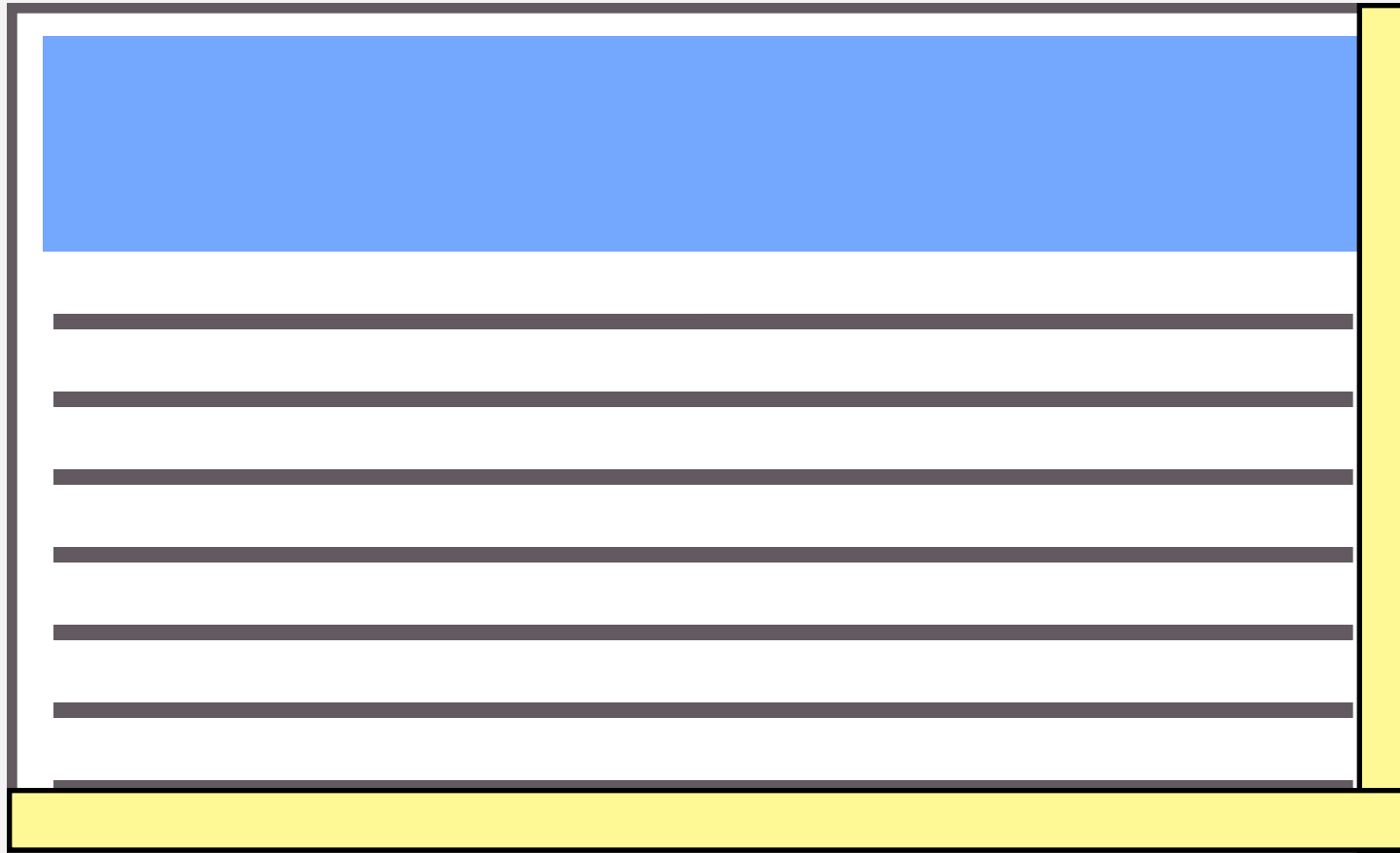
\* visible / auto / **hidden** / scroll



hidden - esconde el contenido que sale de los limites.

# Overflow

\* visible / auto / hidden / scroll



scroll - agrega barras de arrastre, aunque no sea necesario.

# Tabla de contenidos

---

01 Fundamentos

02 Herencia y especificidad

03 Box Model

04 Mejores prácticas

05 Pseudo selectores


06 ¿Dónde aprendo más?

```
.dry {  
}
```

# Don't Repeat Yourself

scroll - agrega barras de arrastre, aunque no sea necesario.

\* A philosophy to write less code which may be duplicated.



```
h1 {
  font-family: 'Helvetica', sans-serif;
}

p {
  font-family: 'Helvetica', sans-serif;
}

.welcome {
  font-family: 'Helvetica', sans-serif;
}
```



```
html, body {
  font-family: 'Helvetica', sans-serif;
}
```

```
<body>
  <h1>Welcome</h1>
  <p>Hello World</p>
  <div class="welcome">
    This is a small
    text
  </div>
</body>
```

$$\{$$
  
$$\}$$

# Display

\* Es una propiedad para manipular como se despliega un elemento (block, inline, inline-block, none)

```
.welcome {  
  display:block;  
}
```

Convierte el elemento en un tipo bloque

```
.welcome {  
  display:inline;  
}
```

Convierte el elemento en un tipo inline

```
.welcome {  
  display:inline-block;  
}
```

Convierte el elemento en un tipo inline con propiedades de bloque

```
.welcome {  
  display:none;  
}
```

Desaparece el elemento



```
.centering {  
}
```


# Centering

## \* Texto en bloques

```
span {  
  display: block;  
  text-align: center;  
  width: 100%;  
}
```

```
p {  
  text-align: center;  
}
```

```
h1 {  
  text-align: center;  
}
```



El texto es centrado en elementos de tipo bloque, ya que los inline al no ocupar el 100% de la pantalla y solo del contenido en si, no es posible centrarlos como tal.

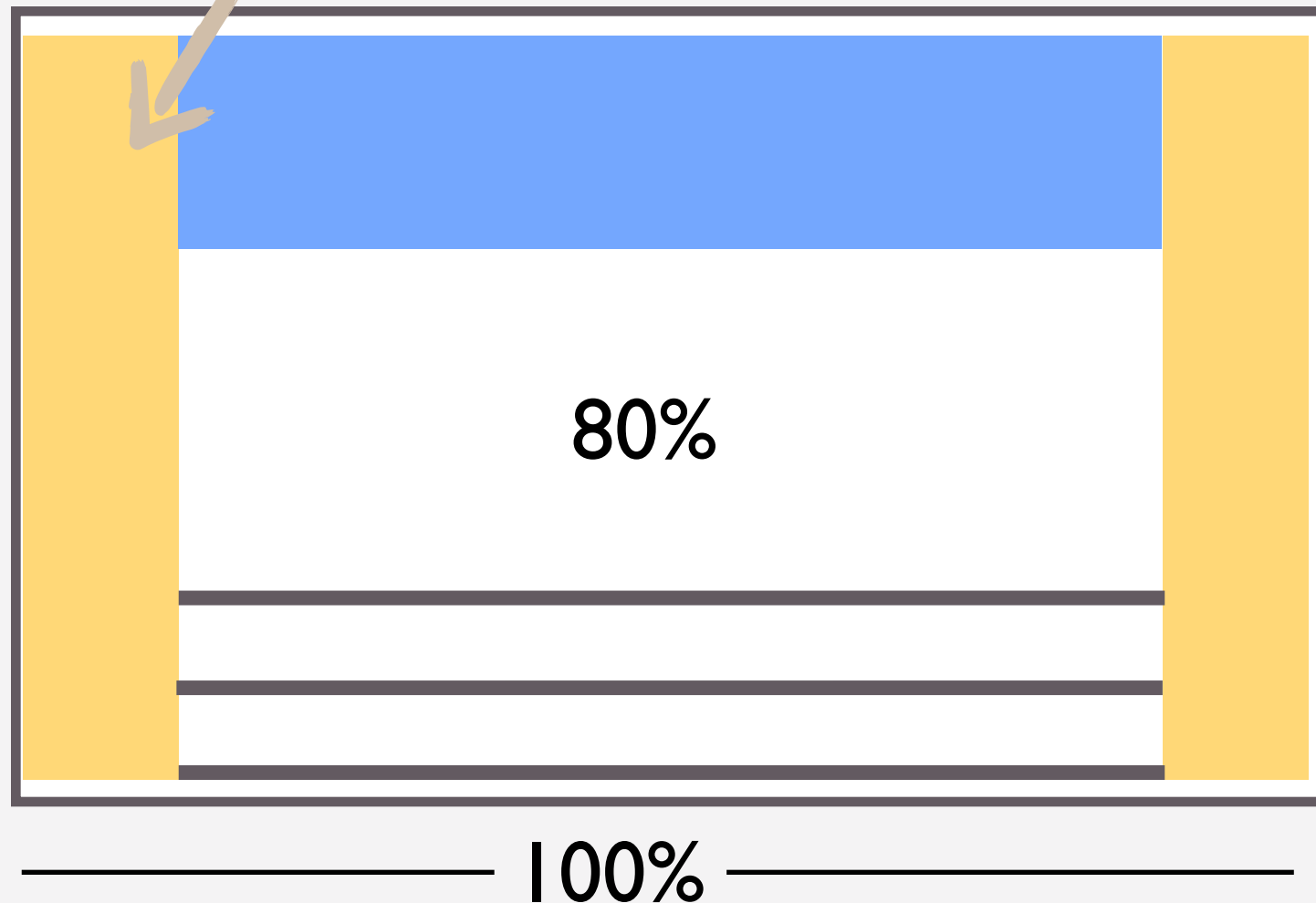
# Centering

## \* Centrar bloques

```
.main-content {  
  width:80%;  
  margin:0 auto;  
}
```

Un “shortcut” para especificar el margen

```
.main-content {  
  width:80%;  
  margin-top:0;  
  margin-right:auto;  
  margin-bottom:0;  
  margin-left:auto;  
}
```



Gracias a que esta definido el ancho del contenedor, el margen derecho e izquierdo en auto centran dicho elemento sin importar el tamaño de la pantalla

```
.normalize {  
}
```

# normalize.css

## A modern, HTML5-ready alternative to CSS resets

Normalize.css makes browsers render all elements more consistently and in line with modern standards. It precisely targets only the styles that need normalizing.

**Download v2.1.3**


IE 8+, Firefox 4+, Safari 5+, Opera, Chrome

[See the CHANGELOG](#)

```
bower install --save normalize-css
```

```
component install necolas/normalize.css
```



 [Tweet](#)

 [Follow @necolas](#)

[Read more about normalize.css »](#)

# Tabla de contenidos

---

01 Fundamentos

02 Especificidad y herencia

03 Box Model

04 Mejores prácticas

05 Pseudo selectores

06 ¿Dónde aprendo más?

```
:hover {  
}
```

```
:visited {  
}
```



```
:first-letter {  
}
```

```
:after {  
}
```

```
:before {  
}
```

```
:first-child {  
}
```

# Tabla de contenidos

---

01 Fundamentos

02 Especificidad y herencia

03 Box Model

04 Mejores prácticas

05 Pseudo selectores

06 ¿Dónde aprendo más?



e.g. Responsive design

## CODING

CSS

HTML

JavaScript

Techniques

## DESIGN

Web Design

Typography

Inspiration

Business

## MOBILE

Responsive

iPhone &amp; iPad

Android

Design Patterns

# Taming Advanced CSS Selectors

By [Inayaili de Leon](#) August 17th, 2009 CSS 98 Comments

CSS is **one of the most powerful tools** that is available to web designers (if not the most powerful). With it we can completely transform the look of a website in just a couple of minutes, and without even having to touch the markup. But despite the fact that we are all well aware of its usefulness, CSS selectors are still not used to their full potential and we sometimes have the tendency to litter our HTML with excessive and unnecessary classes and ids, divs and spans.

The best way to avoid these plagues spreading in your markup and keep it clean and semantic, is by using more complex CSS selectors, ones that can target specific elements without the need of a class or an id, and by doing that **keep our code and our stylesheets flexible**.

## CSS Specificity



## Smashing Job Board

[Senior UX Designer - New Relic - \(San Francisco\) - FullTime](#)

Are you excited about designing slick and interactive experiences for customers who want to make data-driven decisions? Do you find yourself browsing FlowingData?

[Front-End Web Developer - Ridgeway - \(Oxfordshire\) - FullTime](#)

We are currently seeking a front-end developer to join our team, developing responsive mobile applications to be part of our development team, developing first-class user experiences.

[View more job openings...](#)



**SHARE THIS PAGE**

f Like 82k

**CSS Basic**

CSS HOME

CSS Introduction

CSS Syntax

CSS Id &amp; Class

CSS How To

**CSS Styling**

Styling Backgrounds

Styling Text

Styling Fonts

Styling Links

Styling Lists

Styling Tables

**CSS Box Model**

CSS Box Model

CSS Border

CSS Outline

CSS Margin

CSS Padding

# CSS Tutorial

« W3Schools Home

Next Chapter »



Save a lot of work with CSS!

In our CSS tutorial you will learn how to use CSS to control the style and layout of multiple Web pages all at once.

## Examples in Each Chapter

This CSS tutorial contains hundreds of CSS examples.

With our online editor, you can edit the CSS, and click on a button to view the result.

### CSS Example

```
body
{
background-color:#d0e4fe;
}
h1
{
color:orange;
text-align:center;
}
```

**WEB HOSTING****WEB BUILDING****STATISTICS**

Browser Statistics

OS Statistics

Display Statistics





## CSS Selectors

A

- ::after
- :active
- Adjacent sibling
- Attribute

B

- ::before

C

- :checked
- Class

## CSS Properties

A

- align-content
- align-items
- align-self
- animation
- appearance

B

- backface-visibility
- background
- border
- border-collapse
- border-image
- border-radius

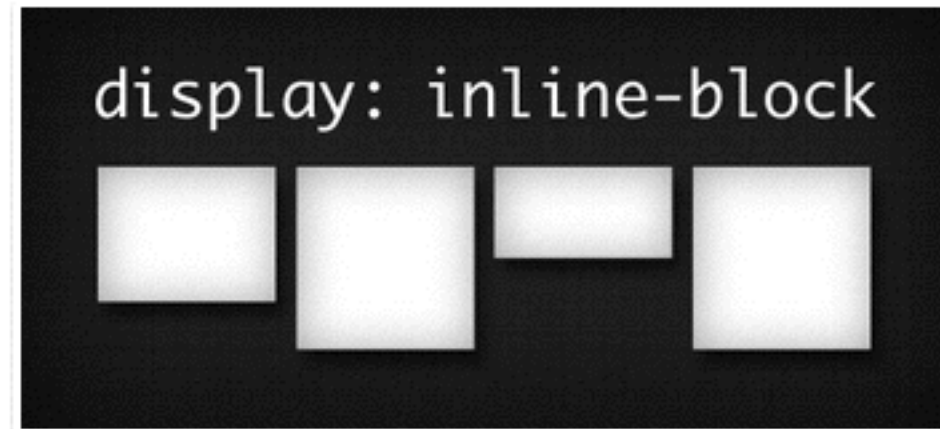


# What's the Deal With Display: Inline-Block?

by **Joshua Johnson** on 29th February 2012 with **63 Comments**

We've been using floats for layout pretty much since we left tables behind. It's a quirky solution that can often cause troubles, but if you know what you're doing, it works.

One interesting alternative to floats that people are turning to more and more lately is to set the *display* value of an element to *inline-block*. What does this do exactly? How is it like a float? How is it different? Let's dive in and see what we can discover.



## The Display Property

Subscribe



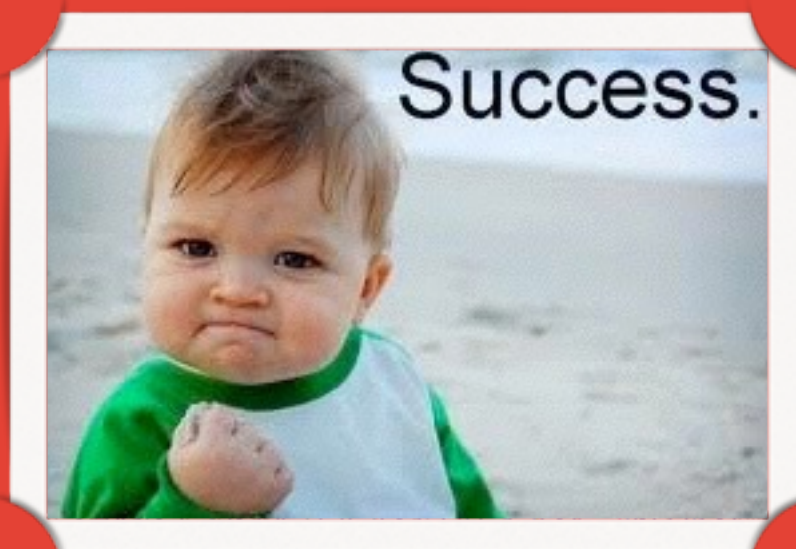
Membership

Join the community as a Design Shack member, and you'll be able to:

- ✓ Feature your designs in the gallery
- ✓ Save your favourite designs into a collection
- ✓ Receive our weekly newsletter for designers
- ✓ Use keyboard navigation
- ✓ Gain early access to new features

Become a Member

Talk about CSS  
¡Gracias!



Abraham Kuri Vargas  
@kurrenn

