CSS vendor prefixes

https://www.lifewire.com/css-vendor-prefixes-3466867

prefix	browser
-webkit-	Chrome, Safari, Android, iOs
-moz-	Firefox
-o-	Opera
-ms-	Microsoft Internet Explorer

`https://codepen.io/paqui-molina/pen/PyyZgV`

Styles

Columns

Multiple columns

https://www.w3schools.com/css/css3_multiple_columns.asp

CSS lists

https://www.w3schools.com/css/css_list.asp

CSS Table

Tables are used to arrange data in tabular format - rows and columns of cells. You can put a variety of data like text, images, forms, links and even other tables in your table.

You may hear experienced Web developers decrying the use of tables, giving the impression that tables should be avoided at all costs. It might seem off-putting at first, but they're not really talking about using tables for tabular information. In earlier days, when layout options were limited, many developers resorted to tables as a means of layout. There's really no need to do that anymore, there are plenty of layout capabilities in CSS3. You really don't want to use tables that way, but there absolutely nothing wrong with using them for their intended purpose --making tables.

Separating content and style

Tables are **semantically incorrect** for layout because they represent presentation and not content. It puts presentation data in your content making your **HTML larger**. The user must download this unnecessary presentation data for every page they visit.

Accessibility: tables are not very screen reader friendly. Using tables for layout will clutter your HTML making it harder for assistive technology to parse your Web page.
Redesigns are harder when your HTML is cluttered with presentational code that should go into CSS. To change the layout of the page, you shouldn't be editing your content. Instead, you should just have to make CSS related changes.
Using CSS (one or two external stylesheets for your whole Web site) is easier to maintain consistency among pages.

Tables were not intended as a layout tool, so it is best to stick to them only for tabular data.

Table elements:

Type	Element
	<table>
	<caption>
Row groups	<thead>, <tfoot>,<tbody>
Col groups	<colgroup>,<col>
Table row	<tr>
Table cells	<th>,<td>

We will use these elements to build our table

Standards CSS for some elements:

table { display: table; border-collapse: separate; border-spacing: 2px; border-color: gray; }

tr { display: table-row; vertical-align: inherit; border-color: inherit;}

td { display: table-cell; vertical-align: inherit; }

th { display: table-cell; vertical-align: inherit; font-weight: bold; text-align: center; }

https://codepen.io/paqui-molina/pen/yLLvvgj

https://codepen.io/paqui-molina/pen/eYYVajb

The <colgroup> and <col> tags

https://codepen.io/paqui-molina/pen/MWWQdxz

<colgroup>

This tag allows you to group columns in a table. Grouping columns is useful if you want to specify properties for a group of columns like applying styles to the whole column instead of repeating it for each cell.

Attribute:

- `span` - takes a positive integer value. It specifies the number of columns you want your colgroup to span (cover). The colgroup element shares its attributes like style and width with all the columns it spans. Essentially it allows a single cell to stretch to cover multiple columns on a particular row.

<col>

Used within `<colgroup>`, the `<col>` tag specifies the column property for each column within a colgroup. The only element a `<colgroup>` can contain is `<col>`.

Attribute:

- `span` - takes a positive integer value. It specifies the number of columns you want the col element to span (cover).

Consider the table above we created using `<tr>`, `<th>` and `<td>`. Let's say I want the 'name' column to be in green and the 'age' column to be orange. You need to use the `<colgroup>` and `<col>` tags to achieve styling effects specific to a column.

The thead, tbody and tfoot tags

Similar to an HTML document, a table in HTML can be split into header, body and footer. We use these three tags - `<thead>`, `<tbody>` and `<tfoot>` - to specify parts of a table.

https://codepen.io/paqui-molina/pen/yLLvvgj

https://www.w3schools.com/html/html_tables.asp
https://developer.mozilla.org/en-US/docs/Web/HTML/Element/table

Styles in your table

https://www.w3schools.com/css/css_table.asp

//w3css_tables
https://www.w3schools.com/w3css/w3css_tables.asp

Formularios HTML5

http://www.mclibre.org/consultar/htmlcss/html/html-formularios.html
https://www.w3schools.com/html/html_forms.asp
https://www.html.am/tags/html-output-tag.cfm

http://desarrolloweb.dlsi.ua.es/cursos/2012/nuevos-estandares-desarrollo-sitios-web/html5-formularios

Style forms

https://www.w3schools.com/css/css_form.asp

https://codepen.io/paqui-molina/pen/VwjdPYj?editors=1100
https://codepen.io/paqui-molina/pen/zYBaomb

https://codepen.io/paqui-molina/pen/NOewOd

Align forms without tables:

- legend
- fieldset
- label

pseudoclasess to forms

Focus: when the input has the focus
Invalid: while the content of the field is not valid

Valid: while the content of the field is valid

Required: When the attribute required has been used

Optional: the opposite, podemos seleccionar todos los campos que no tengan aplicado el atributo "required" mediante la pseudo-clase "optional"

In-range: dates or number field. White the value is between the range, min and max.

Out-of-range: When the value is out of the range

we can mix pseudoclass how we want

```
/* the input has the focus, it's required and, for the moment, it's no valid */

input:focus:required:invalid {
      background-color:red;
}
```

CSS3 fonts

http://www.mclibre.org/consultar/amaya/css/css-fuentes-web.html#

https://fonts.google.com/
https://developers.google.com/fonts/docs/getting_started

https://codepen.io/paqui-molina/pen/JjjvowL
https://codepen.io/paqui-molina/pen/GRRdJgV
https://codepen.io/paqui-molina/pen/WNNJvoo

Box model

https://www.w3schools.com/css/css_boxmodel.asp
https://developer.mozilla.org/es/docs/Learn/CSS/Introduction_to_CSS/Modelo_cajas
https://developer.mozilla.org/es/docs/Learn/CSS/Introduction_to_CSS/Modelo_cajas#Aprendizaje_activo_trabajando_con_cajas

box model: https://codepen.io/paqui-molina/pen/aRrVzO
Tipos de cajas: https://codepen.io/paqui-molina/pen/OBYdOW

Outline: https://www.w3schools.com/css/css_outline.asp
https://codepen.io/paqui-molina/pen/RemdMx

Display: https://www.w3schools.com/css/css_display_visibility.asp
Overflow: https://www.w3schools.com/css/css_overflow.asp
https://codepen.io/paqui-molina/pen/OBYrXG

https://devcode.la/tutoriales/modelo-caja-css/

colors and transparency

https://www.w3schools.com/css/css_colors.asp
http://w3.unpocodetodo.info/css3/hsl.php

Backgrounds

https://www.w3schools.com/css/css_background.asp

Multiple backgrounds

https://www.w3schools.com/css/css3_backgrounds.asp
https://developer.mozilla.org/es/docs/Web/CSS/background-size

lists that don't look like it

DropDown

https://www.w3schools.com/css/css_dropdowns.asp

https://www.w3schools.com/cssref/pr_class_visibility.asp

Shadows:

https://www.w3schools.com/css/css3_shadows.asp

Gradients

https://www.w3schools.com/css/css3_gradients.asp

Repetición de gradientes
https://www.w3schools.com/cssref/func_repeating-linear-gradient.asp

Gradientes radiales
https://developer.mozilla.org/es/docs/Web/CSS/radial-gradient

Transitions

https://www.w3schools.com/css/css3_transitions.asp