Adding CSS to a Page

The best way to store CSS styles for a website is in external .css files, and linking them into the various web pages.

You can also store CSS per-page within `<style type="text/css">` tags in the `<head>...</head>` portion of the webpage.

You can inline CSS using the **style** attribute of a tag, but it is generally not good practice.

CSS format

In CSS, you style an element by identifying it by its **element type** (e.x., p, a, div, etc.), its **class**, or its **id**. Whichever type you use, you specify the identifier first, then enclose the styling rules within curly braces { }.

`p { text-indent: 30px; }`	Styling an Element
`.story { text-indent: 30px; }`	Styling an Element with the given Class
`#chapter-1 { font-style: italic; }`	Styling an Element with the given ID

Setting up rules for elements

There are various ways to specify rules for one or many elements, with the option of being more or less specific as needed.
CSS is based on specificity to choose what rule an element obtains, as it can get multiple styles from multiple rules, and rules can be overridden. The more specific your identifier is, the more likely its styles are to get priority.

For example, we could have a broad class:

`.center { }`

But, we can't center *text* and center *divs* in the same manner! So what would you put as the styles for the .center class?

`p.center { text-align: center; }`	Any paragraph tags with the class "center" use the text-align style to center the text.
`div.center { margin: 0 auto; }`	Any div tags with the class "center" use the margin style to center the container.

So, you can reuse the same class name but give specific styles based on the element type it belongs to.

You can do the same with an ID...

```
p#chapter-1 { text-align: center; }
```

Here, this style only applies to a paragraph with the ID “chapter-1”.

You can give multiple elements the same style by separating them with a comma:

h1, h2 { font-weight: bold; }	h1 and h2 tags will be bold now
a, a:active, a:visited { color: #ff0000; }	A standard link, a link with the psuedoclass active, and a link with the pseudoclass visited all will be the same color.

One element can also contain multiple classes:

```
<div class=”column col-1”></div>
<div class=”column col-2”></div>
```

There, this div will obtain styling from both the **column** class and the **col-1** class. If they have colliding styles, it's hard to tell what will win out, but it might get separate styles from each...

```
.column { float: left; } /* All columns will float to the left */

/* Each specific column has its own styles, too */
.col-1 { width: 25%; }
.col-2 { width: 10%; }
```

You might want to style the same sort of element or same class in different areas of the page. For example, a **news** container might style the **h2** tag differently from the **photo-gallery** container. Here is where specificity comes in handy. You can style h2 tags in separate ways, based on who its parents are.

```
nav h2              { font-size: 30px; }
.news h2            { text-align: center; font-size: 13px; }
.photo-gallery h2   { color: #aaaaaa; font-style: italic; }
```

This can be as nested as it needs to be:

```
header { background: #aaaaaa; }
header nav { clear: both; }
header nav ul li { float: left; }
```

```
.main-content { background: #ffffff; }
.main-content ul li { list-style-type: none; }
```

Remember that you can use **elements, classes,** and **IDs**!

Layout nuances

Having your page flow in more than just a vertical format can be tricky until you're familiar with the tricks.

Layouts with Floats

You can stack containers next to each other, or on either side of a parent container, with float. Float can also be used to have an image only take up a portion of a region and allow inline (text) items to flow next to it.

```
img { float: left; }
```

Note that for this application, you do not need to have a clearfix. The non-floating elements have had their position altered, and this is desired here.

Lorem ipsum dolor sit amet, conse
lectus. Sed sit amet ipsum mauris.
nec consectetur ante hendrerit. Do
libero egestas mattis sit amet vitae
porta lorem lacinia consectetur. D
non tortor. Lorem ipsum dolor sit
ut gravida lorem. Ut turpis felis, p
Pellentesque auctor nisi id magna consequat sagitt
pharetra tincidunt feugiat nisl imperdiet. Ut conva
Donec sed odio eros. Donec viverra mi quis quam
sociis natoque penatibus et magnis dis parturient
accumsan ultricies. Mauris vitae nisi at sem facilis

If you're floating multiple items together, you will want to either:

- Add the “clearfix” or “cf” class to the parent element (you need to add the clearfix CSS to your CSS file)
- Add an element immediately after the last floating item, whose clear style is set to “clear:both;”

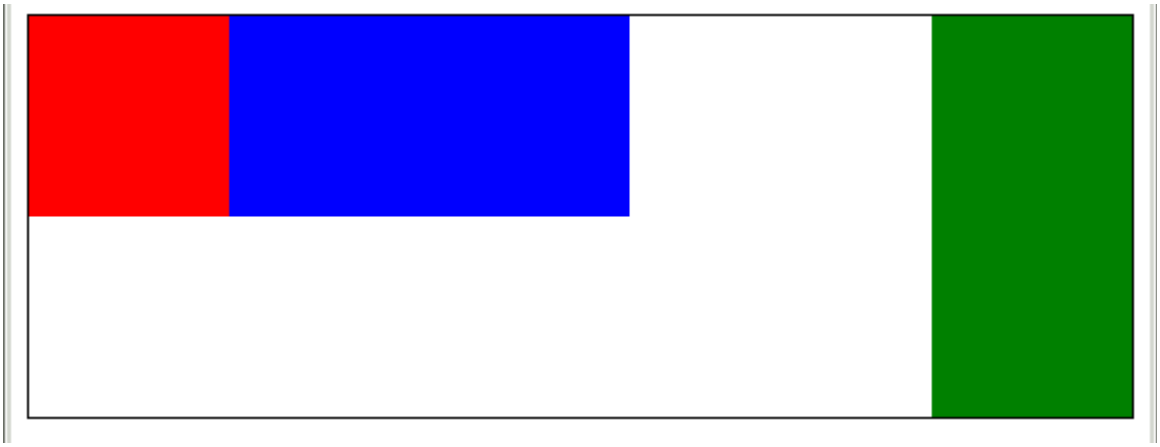
You can make elements float left or right, and they will stack against each other against either side of their parent container.

```
.container { border: solid 1px #000000; }

.div1 { width: 100px; height: 100px; background: red; float: left; }
.div2 { width: 100px; height: 200px; background: green; float: right; }
.div3 { width: 200px; height: 100px; background: blue; float: left; }

.clear { clear: both; }
```

```
<div class="container">
	<div class="div1"></div>
	<div class="div2"></div>
	<div class="div3"></div>
	<div class="clear"></div>
</div>
```



However, a clearfix is required if you are doing this. There are two ways you can fix the float error.

How do you know when you have a float problem on your hands? Look out for parent containers not showing up anymore – generally, their height has been reduced to 0px and therefore you cannot see it. Also look for elements after the floating items having their positions changed, or your page just looking twisted in general.

Positioning

Be familiar with how Relative, Absolute, Fixed, and Static positions work. What is their positioning based on?