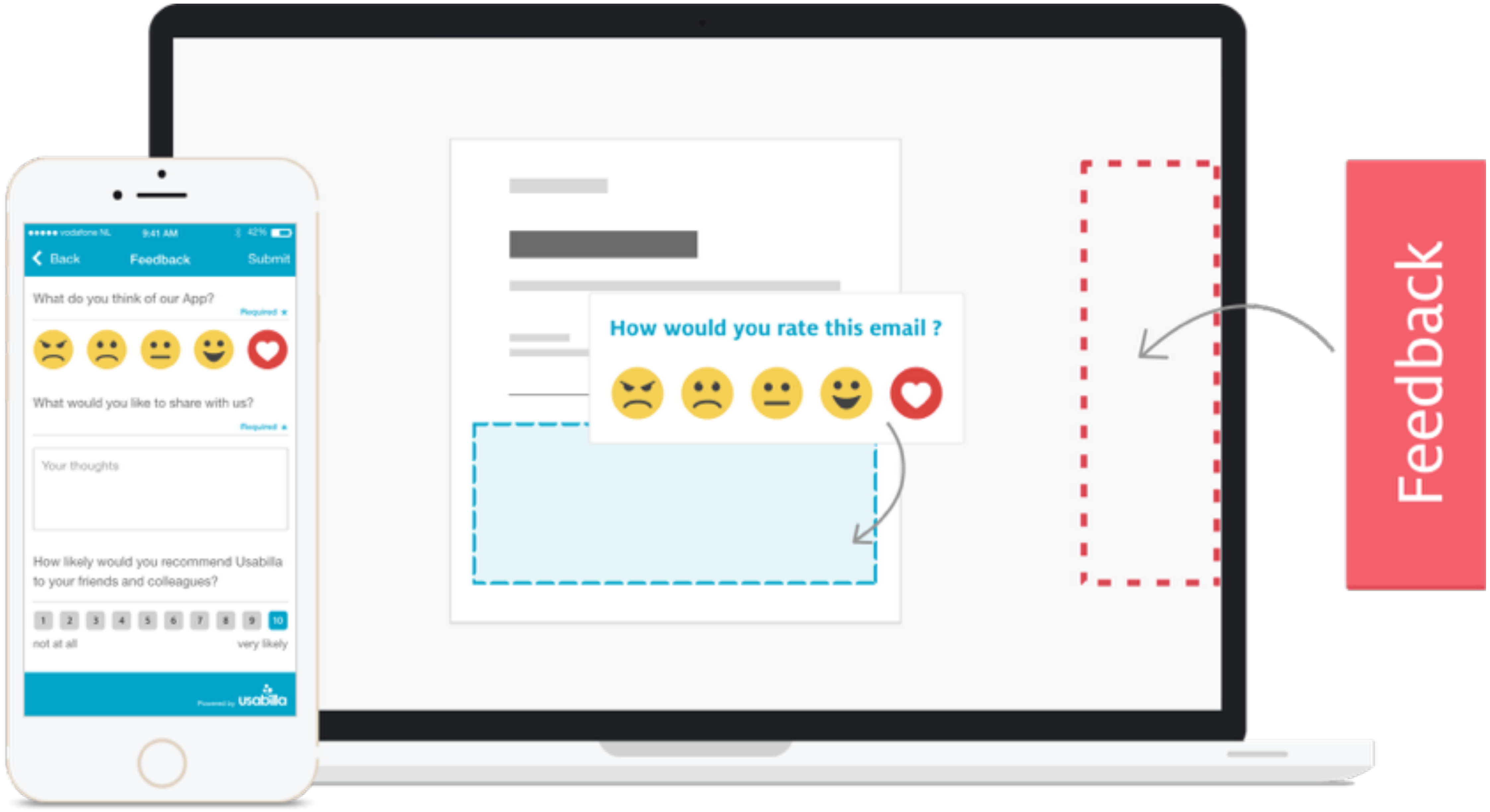
Intro in HTML & CSS

Codaisseur

About Me


●●●●● vodafone NL 9:41 AM 42%

Back Feedback Submit

What do you think of our App?

Required *

What would you like to share with us?

Required *

Your thoughts

How likely would you recommend Usabilla to your friends and colleagues?

1 2 3 4 5 6 7 8 9 10

not at all very likely

Powered by usabilla

How would you rate this email ?

usabilla

Program

- HTML & CSS basics
- Build a personal website
- Demo’s (ship it)

Timetable

09.30 - 11.00 HTML Basics

11.00 - 12.30 Begin CSS

12.30 - 13.30 Lunch

13.30 - 16.00 Second part CSS

16.00 - 16.30 CSS frameworks

17.00 - 18.00 Demo’s

HTML

What is HTML?

- Hypertext Markup Language
- The language of the web
- A document rendered by the browser
- Tags and Attributes

Tags

Basic html file with tags

index.html

```
<!doctype html>
<html>
  <head>
    <title>BeerLover</title>
  </head>
  <body>
    <h1>Beers I love</h1>
  </body>
</html>
```

Tags

most tags need to be opened and closed

index.html

```
<!doctype html>
<html> ←——— opening tag
  <head>
    <title>BeerLover</title>
  </head>
  <body>
    <h1>Beers I Love</h1>
  </body>
</html> ←——— closing tag
```

Tags

Some exceptions:

```
<img src="beer.jpg">
```

```
<link type="stylesheet/css" href="style.css">
```

```
<br>
```

Attributes

Tags can have attributes

index.html

```
<html>
  <head>
    <title>BeerLover</title>
  </head>
  <body>
    <h1>Beers I Love</h1>
    <img src="amstelmalt.jpg">
  </body>
</html>
```

the src attribute
of the img tag

Attributes

tag can have multiple attributes

```
<img src="amstelmalt.jpg" alt="picture of the delicious
Amstel Malt beer" class="beer_picture">
```

Nesting

Introducing the ul tag

index.html

```
  …
  <body>
    <h1>Beers I Love</h1>
    <ul>
      <li>Amstel Malt</li>
      <li>Hoegaarden 0%</li>
      <li>Erdinger</li>
    </ul>
  </body>
</html>
```

Nesting

index.html

has tag(s) inside

nested inside of ul

```
…
<body>
  <h1>Beers I Love</h1>
  <ul>
    <li>Amstel Malt</li>
    <li>Hoegaarden 0%</li>
    <li>Erdinger</li>
  </ul>
</body>
</html>
```

Nesting

If one element encapsulates other element(s) it can be referred to as the parent of the encapsulated elements.

index.html

parent of li

child of ul

```
  …
  <body>
    <h1>Beers I Love</h1>
    <ul>
      <li>Amstel Malt</li>
      <li>Hoegaarden 0%</li>
      <li>Erdinger</li>
    </ul>
  </body>
</html>
```

Nesting

So in this case the li tags are children of the ul tag

Question: is ul a parent or a child?

index.html

```
…
<body>
  <h1>Beers I Love</h1>
  <ul>
    <li>Amstel Malt</li>
    <li>Hoegaarden 0%</li>
    <li>Erdinger</li>
  </ul>
</body>
</html>
```

Nesting

Important: Use indentation for your the code to maintain an overview of the parent/child relations

```
<!doctype html>
<html>
<head><title>Bad indentation example
</title></head>
<body>
<h1>Why this is bad</h1>
<ul>
  <li>Bad readability</li>
    <li>Messy</li>
<li>No overview</li>
  </ul>
</body>
</html>
```

Bad

```
<!doctype html>
<html>
  <head>
  <title>Bad indentation example</title>
  </head>
  <body>
    <h1>Why this is good</h1>
    <ul>
      <li>Good readability</li>
      <li>Neat</li>
      <li>Overview</li>
    </ul>
  </body>
</html>
```

Good

Basic structure

An .html file consist of an html element with head and body as its children.

index.html

```
<!doctype html>
<html>
  <head>
    <title>BeerLover</title>
  </head>
  <body>
    <h1>Beers I love</h1>
  </body>
</html>
```

head - contains document info etc
not visible on the screen

body - contains page content
visible elements on the screen

Assignment

Setup a basic html page containing at least one header tag, one image an ordered and an unordered list

Resources
http://gethtml.at
http://www.w3schools.com

CSS

CSS

(yay)

Cascading Style Sheets

1. Selectors
2. Classes & ID’s
3. Fonts & Colors
4. Box model
5. Positioning
6. Display options

Selectors

CSS selectors are used to target the elements on the page that you want to style.

```
<html>
  <head>
    <style>
      h1 {
        color: purple;
      }
    </style>
  </head>
  <body>
    <h1>The Internet Cat Strikes Again</h1>
  </body>
</html>
```

css selector targeting the h1 tag

Selectors

Multiple style rules can apply to the selected element

```
<html>
  <head>
    <style>
      h1 {
        color: purple;
        background: yellow;
        font-size: 32px;
        font-family: ‘Comic Sans’;
      }
    </style>
  </head>
  <body>
    <h1>The Internet Cat Strikes Again</h1>
  </body>
</html>
```

css selector targeting the h1 tag

Selectors

Question: What if there are multiple elements with the same tag name?

```
<html>
  <head>
    <style>
      p { font-family: ‘Comic Sans’; }
    </style>
  </head>
  <body>
    <h1>The Internet Cat Strikes Again</h1>
    <p>Meeeeooooooow!</p>
    <p>Made with love by the internet.</p>
  </body>
</html>
```

Classes & ID’s

Introducing classes and ID’s

```
<html>
  <head>
    <style>
      .cat-quote { font-family: ‘Comic Sans’; }
      #footer-message { font-family: ‘Arial’; }
    </style>
  </head>
  <body>
    <h1>The Internet Cat Strikes Again</h1>
    <p class=“cat-quote”>Meeeeoooooow!</p>
    <p id=“footer-message”>Made with love by the internet.</p>
  </body>
</html>
```

Classes & ID’s

Introducing classes and ID’s

```
<html>
  <head>
    <style>
      .cat-quote { font-family: ‘Comic Sans’; }
      #footer-message { font-family: ‘Arial’; }
    </style>
  </head>
  <body>
    <h1>The Internet Cat Strikes Again</h1>
    <p class=“cat-quote”>Meeeeoooooow!</p>
    <p id=“footer-message”>Made with love by the internet.</p>
  </body>
</html>
```

selector for the element with cat-quote class

selector for the element with footer-message ID

Classes & ID’s

Classes are targeted by using the . notation and ID’s by #

.css

```
#garfield { border: 5px solid black; }
.cat-quote {
  font-family: ‘Comic Sans’;
  font-style: italic;
  font-weight: bold;
  font-size: 60px;
}
.cat-image {
  width: 100px;
  height: 100px;
}
```

.html

```
<div id=“garfield”>
  <h1>Quotes from Garfield</h1>
  <img class=“cat-image” src=“cat1.jpg”>
  <p class=“cat-quote”>Meoww?</p>
  <img class=“cat-image” src=“cat2.jpg”>
  <p class=“cat-quote”>Puurrrrrrr.</p>
</div>
```

Classes & ID’s

Applicable to multiple elements
Multiple elements can be styled with the same set of style rules.

Classes vs ID’s
Typically the use of ID’s are discouraged and not recommended. Use classes instead.

Naming
Make use of semantic naming like “nav-bar” or “beers-list” to make the html easy to read. For more info check out the BEM methodology for naming classes.

Assignment

Add styling to your page using classes and ID's.

Resources
http://adamschwartz.co/magic-of-css/
http://www.colorzilla.com/

Fonts

Custom fonts

Fonts can be added using the @font-face CSS selector. In the @font-face you'll declare the name, properties and the source for the font that you want to use. Make sure you do it at the top of your styles.

```
@font-face {
  font-family: ‘Brandon Grotesque’;
  src: url(‘fonts/brandon_grotesque.ttf’) format(‘ttf’);
  font-weight: normal;
  font-style: normal;
}

h1 { font-family: ‘Brandon Grotesque’ }
```

Fonts

Google fonts

Google fonts can be added by adding the link tag with to your index.html. It's linking to an external stylesheet with all the @font-face selectors that you need to start using the font.

```
…
<head>
  <title></title>
  <link href='https://fonts.googleapis.com/css?family=Roboto:400,700' rel='stylesheet'
  type=‘text/css’>
  <link href="style.css" rel="stylesheet">
</head>
…
```

https://www.google.com/fonts

Colors

In CSS colors can be applied using different type of values:

Names

```
body { background-color: yellow; }
```

Hex decimal

```
body { background-color: #FFFF00; }
```

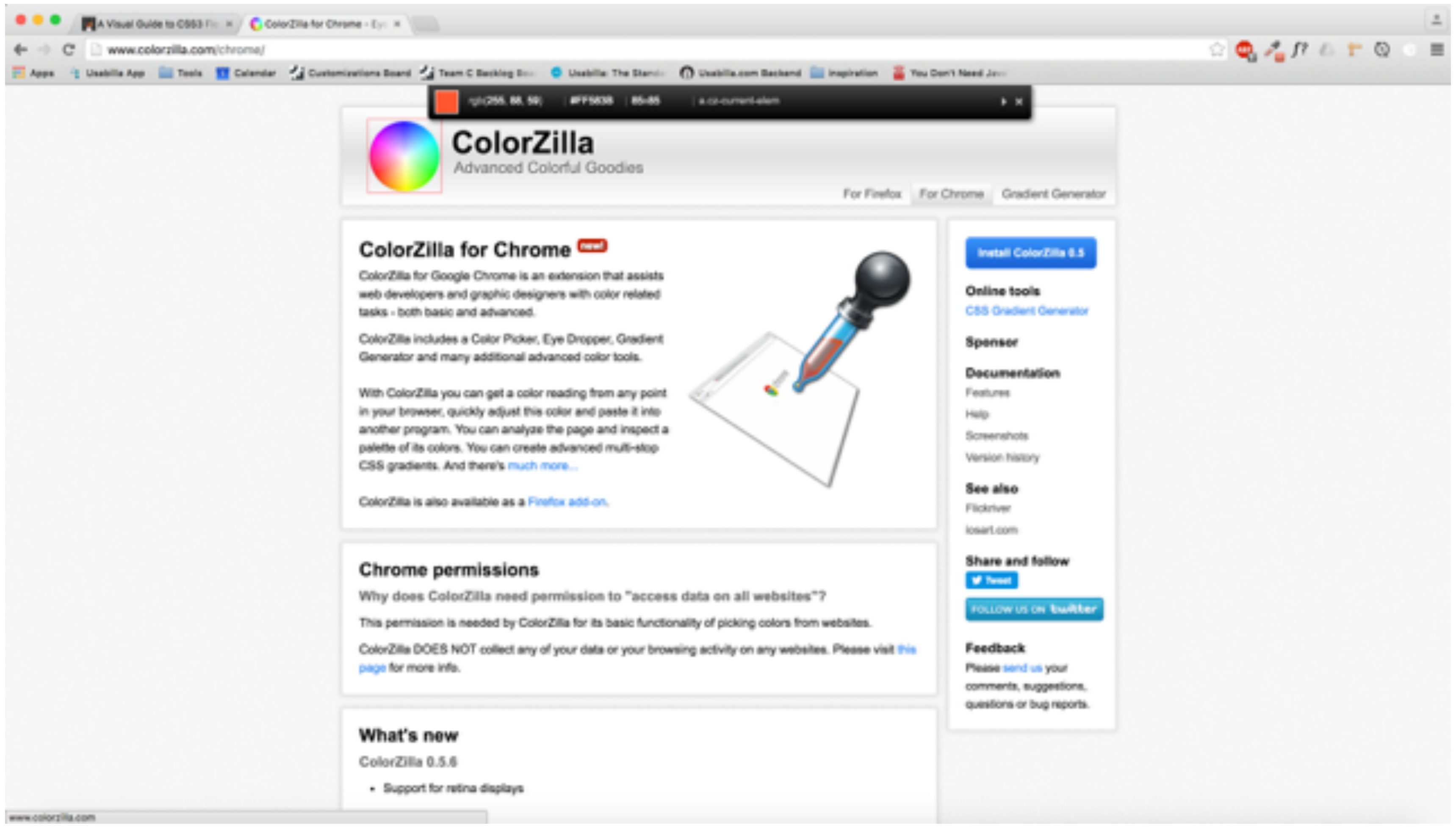
RGB

```
body { background-color: rgb(255,255,0) }
```

*There is also RGBA which also let's you set the opacity of the color

Colors

To get a more specific color use it’s better to use the Hex value or rgb. Useful tool to check out is the **color picker** from Colorzilla. With this chrome plugin you can pick colors that you like from any website.



ColorZilla
Advanced Colorful Goodies

ColorZilla for Chrome new!

ColorZilla for Google Chrome is an extension that assists web developers and graphic designers with color related tasks - both basic and advanced.

ColorZilla includes a Color Picker, Eye Dropper, Gradient Generator and many additional advanced color tools.

With ColorZilla you can get a color reading from any point in your browser, quickly adjust this color and paste it into another program. You can analyze the page and inspect a palette of its colors. You can create advanced multi-stop CSS gradients. And there's much more...

ColorZilla is also available as a Firefox add-on.

Chrome permissions

Why does ColorZilla need permission to "access data on all websites"?

This permission is needed by ColorZilla for its basic functionality of picking colors from websites.

ColorZilla DOES NOT collect any of your data or your browsing activity on any websites. Please visit this page for more info.

What's new

ColorZilla 0.5.6

- Support for retina displays

Online tools
CSS Gradient Generator

Sponsor

Documentation
Features
Help
Screenshots
Version history

See also
Flickriver
iosart.com

Share and follow

Feedback
Please send us your comments, suggestions, questions or bug reports.

Assignment

Apply a custom font and make use of different color throughout your page.

Resources
http://www.colorzilla.com/

Time
to Organise
our Files

Move our styles out of our index.html by pasting them in a separate file called style.css

old index.html

```
<!doctype html>
  <head>
    <title></title>
    <style>
      h1 {
        font-size: 16px;
        font-weight: bold;
      }
      p {
        color: blue;
      }
    </style>
  </head>
  <body>
    <h1>Hello, world.</h1>
    <p>The world loves you.</p>
  </body>
</html>
```

new index.html

```
<!doctype html>
  <head>
    <title></title>
    <link rel="stylesheet/css" href="style.css">
  </head>
  <body>
    <h1>Hello, world.</h1>
    <p>The world loves you.</p>
  </body>
</html>
```

style.css

```
h1 {
  font-size: 16px;
  font-weight: bold;
}
p {
  color: blue;
}
```

Box model

Think of every element in your browser to be a box.

Try it out:

Add the following code to the top of your .css file

```
* { border: 1px solid red; }
```

Box model

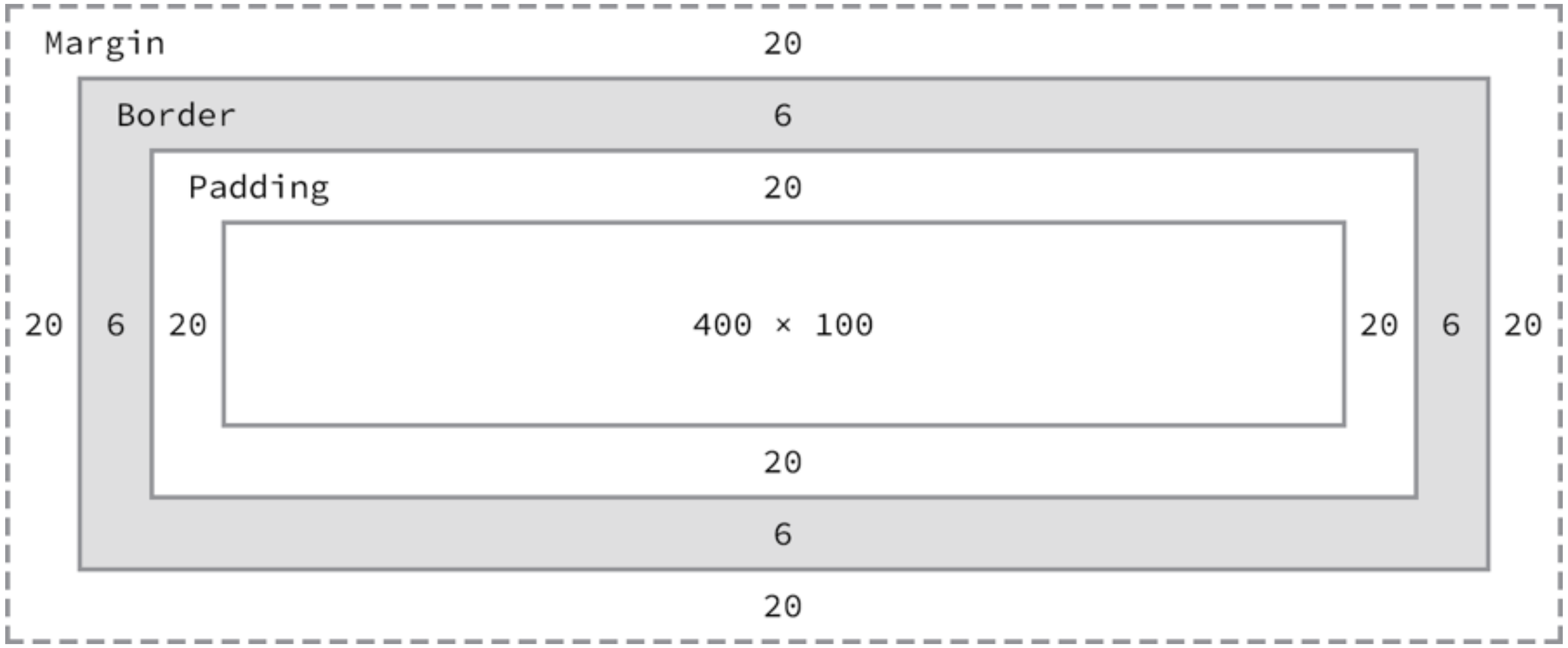
In order to manipulate the space between the boxes we can use margins and paddings.

Padding (inner)
space between the border of the box and its content.

Margin (outer)
space between the border of the box and other boxes

Box model

Box model visualized:



Normalize / CSS

Because different browsers have different default styles for html elements, developers started to use an extra stylesheet to ‘reset’ or ‘normalize’ their page before applying their main styles.

This is done by linking normalize.css **above** your style.css in your .html file

```
…
<head>
  <title></title>
  <link rel=“stylesheet” href=“normalize.css”>
  <link rel=“stylesheet” href=“style.css”>
</head>
…
```

Assignment

Experiment with margin and paddings to create more space between the elements. Add normalize.css to your file.

Positioning

There are 4 different positioning properties for every element in the page. If an element is positioned it responds to left, top, right, bottom properties.

1. **Static**
2. **Absolute**
3. **Relative**
4. **Fixed**

Positioning

1. Static
Default positioning, renders as normal
Does not respond to left, right, top or bottom properties.

2. Absolute
Element is removed from its original position and positioned relative of the first positioned parent.

3. Relative
Element is positioned relative from its original default position.

4. Fixed
Element is removed from its original position and positioned relative to the viewport.

*check out positioning folder in lesson materials for demo’s

Display property

1. **Block**
2. **Inline**
3. **Inline-block**
4. **Flex**

*check out display folder in lesson materials for demo’s

Display property

1. Block
Element will try to take up the full width of the parent element and start on a new line.

2. Inline
Element will stay on the same line and only take up the necessary space the content needs.

3. Inline-block
Similar to inline, but the inside of the element will behave like a block.

4. Flex
New in CSS3 - developed for complex layouts.

*check out display folder in lesson materials for demo's

Display property

1. Block
Element will try to take up the full width of the parent element and start on a new line.

2. Inline
Element will stay on the same line and only take up the necessary space the content needs.

3. Inline-block
Similar to inline, but the inside of the element will behave like a block.

4. Flex
New in CSS3 - developed for complex layouts.

*check out display folder in lesson materials for demo’s

Assignment

Use display and position properties to make your layout more interesting. Like adding multiple columns and a navigation bar.

Resources
http://adamschwartz.co/magic-of-css/
http://flexboxin5.com/

CSS Frameworks

Frameworks are essentially CSS libraries with prewritten classes that you can apply to your html. It can speed up the development process and are very handy for prototyping.

The most popular framework is **Bootstrap**. Originally developed by Twitter.

Benefits

- Fast prototyping
- Grid system
- Easy to use

Disadvantages

- Masks the CSS rules
- Plain look and feel
- Not customized

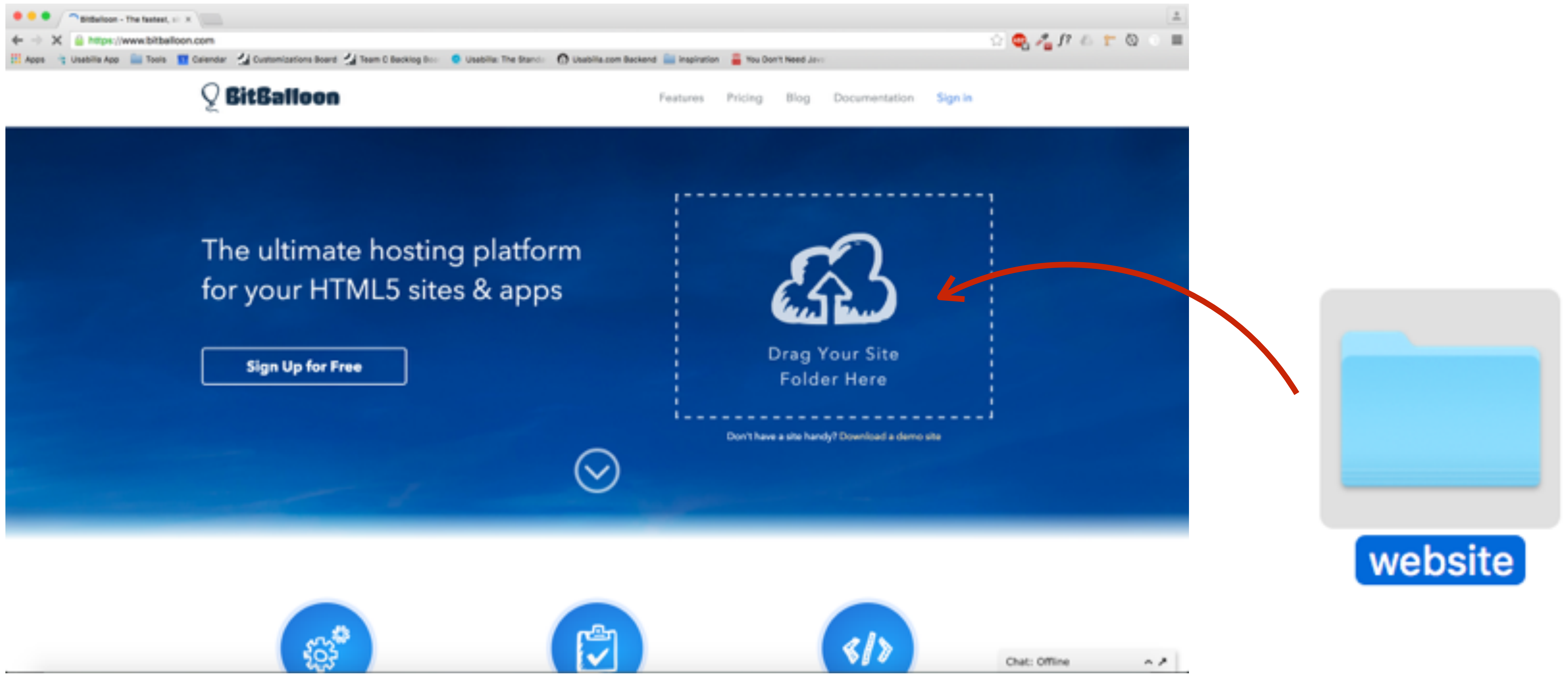
(extra) Assignment

Implement a CSS framework in your current project.

*check out frameworks folder in lesson materials for demo's

Deploy your site

For deployment we'll use a service called bitballoon.com. It's very easy to use: Grab the entire folder where your index.html is in and place it on the landing zone. After processing you'll be given a link - that's where your website can be visited.


BitBalloon
Features
Pricing
Blog
Documentation
Sign in
The ultimate hosting platform for your HTML5 sites & apps
Sign Up for Free
Drag Your Site Folder Here
Don't have a site handy? Download a demo site
Chat: Offline
website