

Chapter

# 7

## Web Development

### **CHAPTER AUTHORS**

Abhinay Anand

Kalpita Jain

Kshitiz Shankar

Ho Winston Haryoho

## CONTENTS

1	Introduction .....	4
2	Big Picture.....	4
3	Libraries.....	5
4	Web Frameworks.....	6
5	Html 5 .....	9
5.1	Introduction .....	9
5.2	Features.....	10
5.2.1	Multimedia and Graphics .....	10
5.2.2	Performance .....	13
5.2.3	Security .....	17
6	CSS → CSS3 .....	18
6.1	Borders.....	18
6.1.1	Rounded Borders.....	18
6.1.2	Gradients .....	19
6.1.3	Box Shadows.....	19
6.1.4	Border Images.....	20
6.2	Text Effects.....	20
6.2.1	Text Shadow .....	20
6.2.2	Web Fonts.....	20
6.3	Columns.....	21
6.3.1	Multiple columns using CSS3.....	21
6.4	Transformation .....	22
6.4.1	Skew .....	22
6.4.2	Rotate .....	22
6.4.3	Translate .....	23
7	Syntactically Awesome Stylesheets (Sass).....	23
7.1	Nesting.....	24
7.2	Variables .....	24
7.3	Mixins.....	25
7.4	Selector Inheritance .....	25
8	Conclusion.....	26
9	RESOURCES.....	27



## 1 INTRODUCTION

*As of 2011, there are 582 million registered websites.*

*The World Wide Web has over 2 billion users worldwide.*

*As of 2008, total revenue generated over the web was a whopping \$133.7 billion*

Those are pretty huge numbers aren't they? The above figures quite effectively capture the immensity of the World Wide Web. It also speaks volumes about the incredible opportunities that exist in the field of web development. Over the years, Web Development has been used to provide many applications, right from providing information to multimedia, e-commerce, social networking and online gaming. This diversity and versatility of the web makes developing web applications exciting and fun.

Web development is a broad term for the work involved in developing a web site for the Internet (World Wide Web) or an intranet (a private network). This can include web design, web content development, client liaison, client-side/server-side scripting, web server and network security configuration, and e-commerce development. However, among web professionals, "web development" usually refers to the main non-design aspects of building web sites: writing markup and coding. Web development can range from developing the simplest static single page of plain text to the most complex web-based internet applications, electronic businesses, or social network services.

## 2 BIG PICTURE

When we start learning anything, we are always curious if we need to have some background knowledge before we get into learning the topic. There are several similar cases when people try to gain knowledge in web development. They search on forums, blogs and several websites to find a source that can place all the 'buzzwords' and 'fads' they have been hearing into a complete overview of Web Development.

The entire web largely can be divided into two broad parts, Front end that includes all that you see on the website and Back end that includes the components that generate what you see. Front end includes your browser such as Google Chrome, Firefox, Safari, Internet Explorer, Opera etc. that shows you a web page. Web browsers do not understand Java, C# or PHP. They can only understand HTML. All the code that you write using different programming languages ultimately generates some HTML that the browser shows to you. Hence, HTML is the first thing you need to know when you are learning web development.

Apart from HTML, your browser also understands CSS and Javascript. CSS is used to style the web pages that we see. It provides all the colors, layouts and fonts to the otherwise dull and boring web page. Javascript is a client-side scripting language that helps to enhance the user interface (UI) and is useful in creating dynamic websites. Some of the other common buzzwords that you would hear in the front end technology includes JQuery, HTML5, CSS3, AJAX that are all built on top of Javascript and traditional HTML and CSS.

Now the next question that arises is that who provides the content to these browsers. This certainly is provided by some of the components in the Back end. You might have noticed some website URLs that contain .aspx, .jsp or .php. Whenever we type a URL in the browser's address bar, it forwards that request to a Web Server in Internet. A Web Server exclusively handles all the HTTP requests from the browser. It analyses the request and forwards it to the relevant application service. After a Web Server gets any reply from the Application Server, it generates

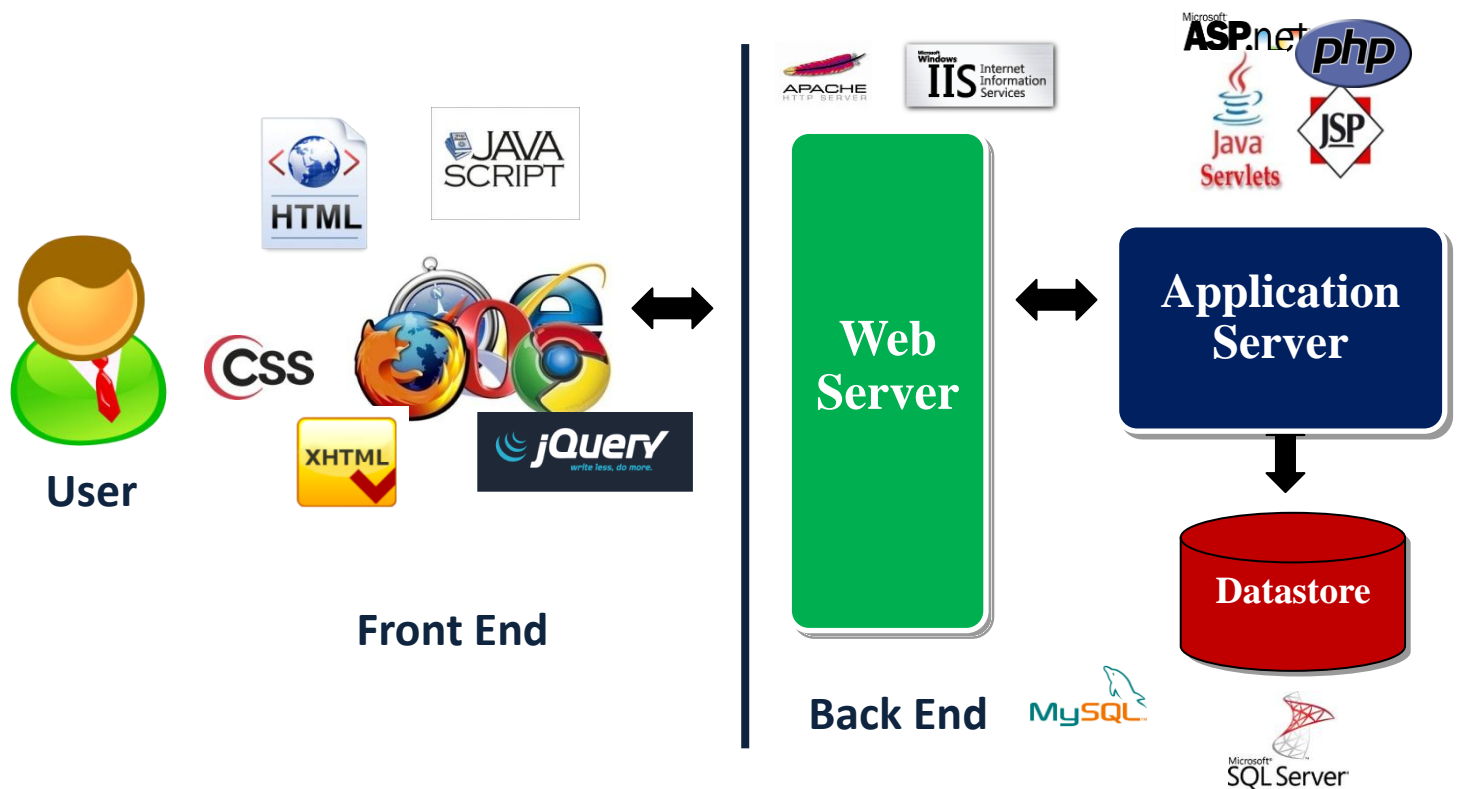
a HTTP response in HTML that it sends to your browser. All our website files also reside in the web server including image files and videos.

Application server provides business logic to the client application programs. In other words, it basically contains all the methods that client applications can call. The information travelling between Application Server and its client is not restricted to simple display mark-ups; instead it has the program logic. Application Server provides software applications with services such as security, data services, transaction support and load balancing. IBM Websphere, Jetty, Apache TomEE, Windows Server AppFabric, Zend and Zope are some of the examples of Application Servers.

Another common Back end component is the database that is used for storing data. It is accessed whenever the application needs to retrieve or store some data relevant to the user. MySQL and Microsoft SQL Server are examples of databases.

As we have discussed, we can divide the web applications into two broad parts that are front end and the back end. The back end is further made up of the application and the storage tiers. The figure below showcases the broad structure of a web application.

Figure 1



### 3 LIBRARIES

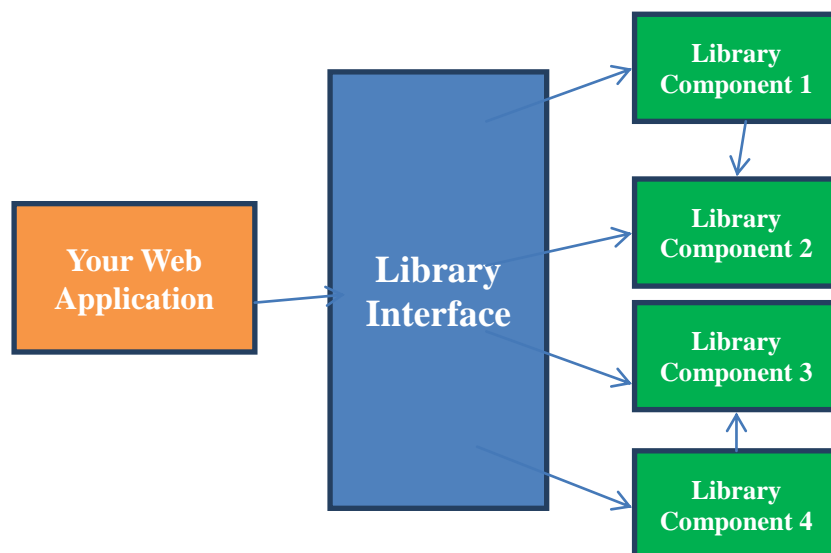
Given all the technologies and tools that are available in the market for both the front-end and back-end components, it would appear that building a web application would consist of huge amount of coding and programming in JavaScript, HTML, or server side scripting language.

However as web developers, we do not have to rely on manual coding for our websites. Instead we can easily use and integrate 'libraries' into our web application. Libraries are a set of resources that are used to fulfill certain functionalities and perform specific task. They include pre-written code and subroutines. Just like libraries in high level programming languages, libraries allow us to reuse an existing resource and to quickly implement a commonly used functionality.

There are a huge number of libraries for web development that are available in the market, with most of them being under GNU General Public License, which means that they are essentially free to use for the web developers. An example for library for the front end components will be JQuery, which is a JavaScript Library. By downloading and integrating the relevant library into our web application, a certain feature can be added quickly into our web application. Given the nature of JavaScript, most of the JQuery libraries cater to appearance or layout, and other front end implementation. Some notable examples will be LavaLamp, JQuery libraries to allow floating buttons and tabs. Another example will be jqZoom, a library that allows the user to zoom in on an image by just hovering the mouse over that particular image. There are still many other libraries for other purposes such as photo slideshow, transition effects, color picker, and so on.

Apart from JQuery, there are many more libraries available for server side scripting languages. As these libraries cater more to the back-end components, the features they provide are generally more diverse. Some example will be MagpieRSS, a PHP library used to parse a RSS (Really Simple Syndication). Another example of libraries will be pChart, which is a PHP library to build charts.

Figure 2



The figure above demonstrates the interaction that takes place when libraries are used by your application. The web developer will only be calling and accessing the library interface. He is abstracted away from the internal implementation of that particular library. Thus this allows the developer to easily add functionalities to his application, implementing a modular approach to web development.

## 4 WEB FRAMEWORKS

While libraries allow us to implement specific functionalities quickly and easily, there are some limitations in only using libraries for large scale web applications. While adding functionalities with libraries is easy, it will soon make the web application huge and disorganized. At the same time, there are various features and common components that can be easily implemented using web framework, without the need to download a library for every single component.

Web framework is a tool that acts as a foundation to create web applications. Contrary to library that generally only perform a specific functionality or provide a specific feature, each web frameworks provide a large number of features and core components under one roof. The main difference lies in the fact that web frameworks calls the code that we write whereas in the case of libraries, our code calls those.

### **User authentication and Session Management**

Management of user accounts is generally already offered by web framework, eliminating the need for additional implementation. Additionally, with the user authentication, frameworks can automatically check and require the authentication of the user for a given page. This increases security of the website by ensuring only relevant users can access certain pages.

Additionally, web frameworks also offer common interface to disk or database storage to ensure simplified way to implement caching, which is very useful to improve the responsiveness of the web application.

### **Data Persistence**

In dynamic web application, most information is generated from persistent data before being displayed to the user. The management of persistent data is done using several ways, such as by creating a core object structure, so that all data will have a consistent basic interface. Additionally, a consistent API will be made available to quickly access data from different storage systems. Some web frameworks also provide automatic storage and retrieval of data objects.

### **Administrative Interface**

The administrative interface allows the administrator to change the content of the site, manage users and permissions. Web framework does this by generating a navigation structure. Web frameworks also provide common interface elements to form fields. It also allows the web developers to automatically generate edit and list pages from persistent data structure.

Apart from the core components, web framework also allows web developers to quickly download and implement specific functionalities very quickly. The exact implementation for this differs among web framework. One such implementation is the usage of modules in Drupal. Drupal is a Content Management System (CMS) and Content Management Framework (CMF) written in PHP. The standard release of Drupal consists of basic features common to CMS. Users can then download community contributed add-ons, more commonly known as modules. Each module provides specific functionality and you can easily add or remove any modules instantly and it will be reflected on the page immediately. The figure on the left below show an example of adding and removing module through a checklist.

Figure 3

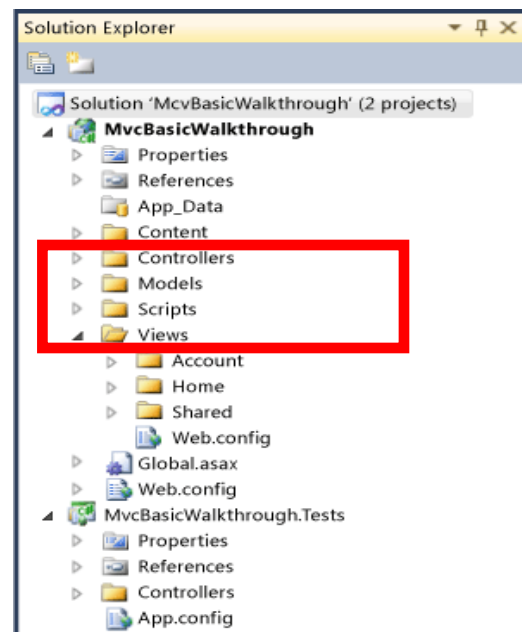
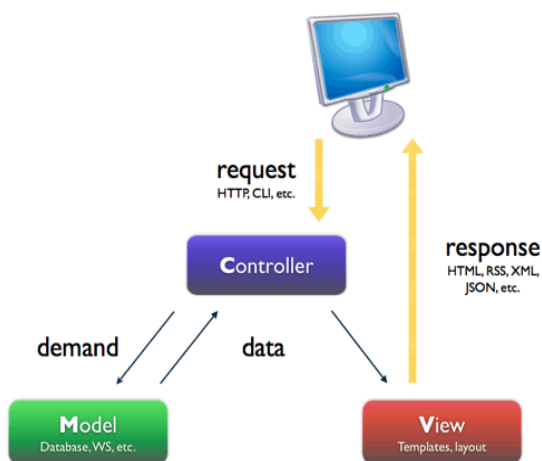
Enabled	Name	Version	Description
<input checked="" type="checkbox"/>	AdSense	5.x-1.6	Display Google AdSense Depends on: Profile (enabled)
<input checked="" type="checkbox"/>	Block Theme	5.x-1.1	Provides a configuration c
<input checked="" type="checkbox"/>	Date API	5.x-1.6	Makes the date api availa Required by: Calendar (enab
<input type="checkbox"/>	Google Analytics	5.x-1.3	Adds Google Analytics jav pages.
<input checked="" type="checkbox"/>	Pathauto	5.x-2.0	Provides a mechanism for for the content they mana Depends on: Path (enabled)
<input type="checkbox"/>	Photo Shuffler		A block module that displa
<input checked="" type="checkbox"/>	Theme Settings API	5.x-2.7	Allows themes to add opti Provides a shared API for



## Design Patterns

All web frameworks are based on certain design pattern, and we are going to focus on the most commonly used design pattern of web application framework which is the Model-View Controller (MVC). The MVC pattern introduces abstraction by isolating the domain logic from the user interface, allowing for separation of concerns between them.

Figure 13.4



The figure above shows exactly how MVC model will be implemented in a web application developed using a web framework. The initial code of the framework handles the initial request from the browser such as the HTTP request and passes it to the application *controller*. The *controller* then performs the appropriate actions by interacting with the application's *model* such as the database. The *controller* would then pass the response to the *view* to build the actual content of the response. The response would then be passed back to the client.



Another example of structure would be the presence of web page themes. With web frameworks, web developer can quickly integrate any web page themes and start building up on the content of the web application.



There are large databases of themes available for any web framework, catered for various different purpose and topics. By using web page themes, the web developer can minimize the amount of work spent on coding in HTML or CSS. Some web templates also have JQuery functionalities already implemented. The web developer only has to customize the web application or add additional functionalities as required.

## Maintenance

Another excellent advantage of web framework is that it allows great maintenance for the existing web application. First of all it allows great extensibility, in the sense that it makes it extremely easy for any web application to incorporate additional functionalities. We have found out how to do this exactly in Drupal, where a checklist is enough to implement specific functionalities.

Another way that web framework improves maintenance is by the presence of continuous bug tracking and bug fixing most commonly found in the framework. Due to the widespread use and the dedicated community support found in most web frameworks, most bugs can be easily tracked by the user and reported. Other users or developers can then fix the bug. The figure below shows a typical bug tracker for a given Drupal module. It lists out the issue, priority, as well as the status of the bug. The developer of that module or authorized user can then fix the problem, and all websites that uses that particular module will also be updated. This greatly simplifies the maintenance effort required.

Figure 13.5

Issues for Translation Management

Create a new issue · Advanced search · Statistics · Subscribe

Search for

Status

Priority

Category

Version

- Open issues -

- Any -

- Any -

- Any -

Component

- Any -

Search

Summary	Status	Priority	Category	Version	Component	Replies	Last updated
don't try to send notifications if disabled <span>new</span>	needs review	normal	bug reports	6.x-1.x-dev	Code	0	9 hours 43 min
discard ICL_DASHBOARD_DISPLAY macro in favor of configurable setting <span>new</span>	needs review	normal	feature requests	6.x-1.x-dev	Code	0	9 hours 51 min
Empty fields break Google Translate <span>new</span>	active	normal	bug reports	6.x-1.2	Code	0	1 day 22 hours
Taxonomy tags double translation	active	normal	bug reports	6.x-1.x-dev	Code	0	1 day 23 hours
Better error reporting for the Google Translate module <span>new</span>	patch (to be ported)	normal	bug reports	6.x-1.2	Code	0	2 days 5 hours
Undefined constants	postponed (maintainer needs more info)	normal	bug reports	6.x-1.1	Code	2	2 days 2 hours
Upon pressing Translate selected documents on q=admin/content/translation-management/dashboard	postponed (maintainer needs more info)	normal	bug reports	6.x-1.1	Code	2	2 days 2 hours
Remove references to "icl_core" module	fixed	minor	bug reports	6.x-1.1	Code	3	3 days 2 hours
Google Translate support is buggy <span>updated</span>	fixed	normal	bug reports	6.x-1.1	Code	2 <span>new</span>	3 days 2 hours

## 5 HTML 5

### 5.1 Introduction

With the introduction of HTML 5, a new set of features have been introduced that increase the functionalities, performance, speed and the overall experience of web applications. All these benefits can be obtained while at the same time maintaining the cross-browser compatibility feature that HTML is known for. However, in order to do that, we have to first learn what HTML 5 is.

HTML 5 itself is not a single independent technology. Instead it is a collection of features that are based on technologies such as HTML mark-up language, the new CSS3, DOM (Document

Object Model), and various new JavaScript APIs. However for our discussion on HTML 5 in this chapter, we will limit ourselves to some of the new features that can be found in the fifth version of the HTML mark-up language.

HTML 5 is being developed and maintained by the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG). HTML 5 has been in development since 2006 with various new features still undergoing development. While HTML 5 is not yet fully supported by any web browser, major web browsers have progressively added additional features of HTML 5 into their latest versions.

## 5.2 Features

### 5.2.1 Multimedia and Graphics

With HTML 5, it is easier than ever for web developer to integrate multimedia capabilities on their web application. Several features, such as playing video or audio file, can now be done using HTML code directly, instead of relying on external plugins such as Adobe Flash. There are also some new features not available previously such as Canvas, that extend the multimedia capability of website even further. In this section, we are going to cover the features mentioned above, namely Video and Audio tag, and Canvas tag.

#### Video

Before HTML5, HTML does not provide any native multimedia support to play video file on the application. Instead, web application has to rely on external plugin such as Adobe Flash in order to play those video files. However this may pose some problems as different browsers may rely on different plugins, and the users have to obtain all the different plugins to play the video on the different browser. Certain plugin, such as Flash is also not supported in some system, namely iOS.

Even though around 98% of today's websites use flash to play video files, there are some inherent weaknesses associated with Flash. Flash has long been known to be relatively slow and inefficient, especially when playing large video. Furthermore, Flash takes a lot of computational resource and power, a problem most commonly found in mobile devices and older computers. On some occasions, Flash also tends to crash the user's browser when attempting to play the multimedia file.

With the issue on hand, HTML 5 has introduced the <video> tag that is aimed to alleviate this problem. The <video> tag or element is a new element which will specify a standard way to embed a video file on a given web page.

A typical <video> element can be configured as below. Here as can be seen, by using the <video> tag, the user can quickly embed any video file of supported format in his website. There are three main video formats supported namely MP4, WebM, and Ogg. The actual file format supported depends on the browser being used to display the video. In the code below, there are two different file with different format being embed. This is done to ensure that all websites will be able to display the video.

Figure 13.6

```
<video width="480" height="360" controls="controls">
  <source src="movie.mp4" type="video/mp4" />
  <source src="movie.ogg" type="video/ogg" />
</video>
```



Apart from displaying the video, "controls" attribute will also provide a standard video player. Not only that, but the video element will also have functions such as `play()`, `pause()`, `load()` methods, together with different properties. With these methods and properties, web developer can create their own player that will incorporate additional features, such as increasing the player size on the fly.

Figure 13.7



## Audio

Similar to playing video files, at the moment web applications have to rely on external plugins in order to embed any audio file. However, HTML 5 has introduced a new element called `<audio>` that will allow the web application to play audio files of different format, namely MP3, Wav, and Ogg audio format.

The actual syntax and output of the `<audio>` element is incredibly similar to `<video>` element. Web developers will be able to obtain a standard player by default, and they can use the various methods and properties of the `<audio>` element to generate their own player or to add additional functionalities.

```
<audio controls="controls">
  <source src="song.ogg" type="audio/ogg" />
  <source src="song.mp3" type="audio/mpeg" />
</audio>
```

A standard HTML5 audio player interface showing a progress bar and a volume icon. The time displayed is 0:10.

As a conclusion, with HTML 5, web developers no longer have to rely on external plugins to embed multimedia content on their website. This also ensures better cross-browser compatibility as users no longer have to rely on different plugins for different browsers in order to play the multimedia content.

At the moment, many websites have been trying out these new features of HTML 5 to stream their multimedia content. Youtube, the largest video streaming website in the world has introduced an opt-in trial where users can use HTML5 player instead of Flash player to play most of the videos available in Youtube. This shows the actual commercial application of these features of HTML 5.

## Canvas

Canvas element is a new element introduced by HTML 5 that allows for real-time rendering of graphics. The rendering itself is dynamic and rely on script, most commonly Javascript. Canvas was first introduced by Apple in 2004, and it was used in its Safari browser. It was then adopted by Opera in 2006 before being introduced as a new element HTML 5.

Canvas itself acts a container or drawable region in which its dimension can be specified with the *height* and *width* attribute. JavaScript will then be used to generate graphics by using drawing functions that are defined for the canvas element, allowing website to dynamically generate graphics.

```
<canvas id="myCanvas" width="200" height="100"></canvas>
```

JavaScript will then obtain the <canvas> element using its id, and a context object will be created using the getContext function.

```
var c=document.getElementById("myCanvas");  
  
var ctx=c.getContext("2d");
```

The resulting object is a built-in HTML 5 object, with methods already defined that will allow the web developers to draw images, shapes, characters, and so on.

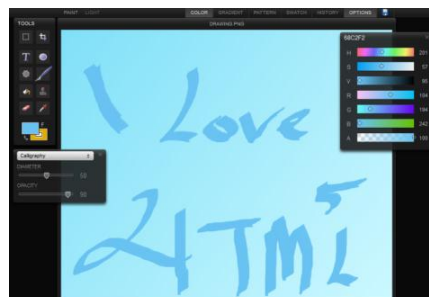
```
var grd = ctx.createLinearGradient(0,0,175,50);  
grd.addColorStop(0, "#FF0000");  
grd.addColorStop(1, "#00FF00");  
ctx.fillStyle = grd;  
ctx.fillRect(0,0,175,50);
```

The first line in the code specifies that we want to create a gradient background with dimension 175\*50. The second and the third line will specify the range of colours to be included within the two ends of the gradient. The last line will generate a new rectangle in the canvas with the same dimension as the image, which is 175\*50. Lastly the fillStyle will be used to ensure that the rectangle will be filled with the gradient background.



This is the final output. The bigger box refer to the canvas element, while the coloured rectangle refers to the rectangle we defined above.

The possibility of drawing graphics on the fly greatly increase the possibilities not only on rendering graphics, but also changing the way the user interact with the web applications.



The picture above depicts a web drawing application. This website relies exclusively on HTML 5 and JavaScript, without any external plugins. Here the users are able to draw graphics and the resulting image will be rendered real time dynamically. This shows the potential application of canvas element introduced by HTML 5.

### 5.2.2 Performance

HTML 5 introduces several new technologies and APIs that aim to increase the performance and responsiveness of web applications. It also aims to create a seamless experience for the user by ensuring that users can still access the web applications even in the presence of intermittent network connection. In this section, we will cover two new technologies, namely Offline API and Web Workers.

#### Offline API

At present, more users are accessing the web applications while in motion such as via mobile phones, laptops, or tablets. In these situations, users often have to rely on intermittent network connection. Bandwidth reduction or sudden loss of connection may often interrupt their connection. With Offline API, users are now able to continually access the web application, even when they are disconnected or unable to reach the server.

When a user is connected to a web server, the user is able to access all the files and resources required to load the website. These resources may include the HTML page, CSS files, JavaScript files, image files, and so on. When a user loses connection and is no longer able to reach the server, the access to those files are interrupted and the browser is unable to display and load the web application. However with HTML 5's Offline API, browsers are able to download some of the files and store them locally. As a result, the user is still able to access the website even without access to the web server. This creates a seamless experience for the user.

In order to do that, the web developer has to first use a Cache Manifest on their web page. Each cache manifest is a file that tells the browser what are the resources and files required to be downloaded and store locally. These files are divided into three main categories and they are also known as namespaces.

The three main categories are:

- **CACHE Category** – This is the default category and if nothing is stated then all the files listed are assumed to be under this category.
- **NETWORK category** – This category lists files that are not to be cached, or not to be downloaded.
- **FALLBACK category**- This lists files with offline alternatives. Hence there will be two different versions for a given file, an online version and an offline version.

In the case of HTML 5 web games such as Angry Birds, the Cache Manifest will include all the files required to play the game, even when the user is offline. All these files will be listed under CACHE.

If the said game also includes a multiplayer functionality, then this feature is not needed in the offline version as the user will not be connected to the internet anyway. Hence the relevant multiplayer file (Multiplayer.php) does not have to be downloaded, and it will be listed under NETWORK in the Cache Manifest.

As the multiplayer feature is disabled in the offline version, the JavaScript file that activates the multiplayer feature in the game will have to be swapped with another JavaScript file that informs the user that the multiplayer feature has been disabled. In our example, the original JavaScript file multiplayer.js will be swapped with its offline counterpart, which is fallback\_multiplayer.js. multiplayer.js and fallback\_multiplayer.js are both listed under FALLBACK in the Cache Manifest.

This is how the Cache Manifest will look like:

CACHE	FALLBACK	NETWORK
Game.css	Multiplayer.js	Multiplayer.php
Game.js	Fallback_multiplayer.js	
Game_image.png		
Game.html		

This file is saved as *game.manifest*. Now we need to add a reference to the manifest in the actual html file. This is done by adding the manifest attribute to the <html> element. When the user first encounters the homepage of the game which is webpage game.html, their browser will download all the files listed in game.manifest.

```
<!DOCTYPE HTML>
<html manifest= "game.manifest" >
<head> <title> Cool Game</title>
</head>
<body>
</body>
</html>
```

## Web Workers

These are API specifications that create background JavaScript threads in order to process CPU intensive tasks. The JavaScript code that runs in the browser is executed using a single thread. This is not an efficient method as some CPU intensive piece of JavaScript can render the page very slowly or even make it unresponsive. Web Workers solve this problem by creating several JavaScript threads that execute independent of each other. This prevents one CPU intensive piece of JavaScript from interfering with the UI code.

### *Creating a simple web worker*

In this example we will create a simple Web Worker that will 'Add' or 'Multiply' two numbers. The JavaScript code below does the actual arithmetic calculation. This Worker is stored in the 'arithmetic.js' file.

```
/* arithmetic.js */

function addNumbers(x,y) {
    return x + y;
}
```

```

function mulNumbers(x,y) {
    return x*y;
}

/*
Add a event listener to the worker, this will
be called when the worker receives a message
from the main page.
*/
this.onmessage = function (event) {
    var data = event.data;

    switch(data.op) {
        case 'mult':
            postMessage(mulNumbers(data.x, data.y));
            break;
        case 'add':
            postMessage(addNumbers(data.x, data.y));
            break;
        default:
            postMessage("Wrong operation specified");
    }
};

```

Now that we have our Worker file ready, we need to call it from our main file. The following line will create a new Worker thread and run the code stored in 'arithmetic.js'.

```

/* Create a new worker */
arithmeticWorker = new Worker("arithmetic.js");

```

After creating workers, we need to send and receive messages from them. For that we need to add an event handler to the main calling code.

```

/*
Add a event listener to the worker, this will
be called whenever the worker posts any message.
*/
arithmeticWorker.onmessage = function (event) {
    document.getElementById("output").value = event.data;
};

```

Now whenever a Worker posts a message the 'onmessage' event is fired and the code within it is executed. The data passed by the Worker is stored in the *event.data* container. We have seen how to receive a message from the Worker but we still do not know how we post one. Posting a message to the Worker is very simple.



```
/* Message sent by the main page to the Worker. */
arithmeticWorker.postMessage('Main says...');
```

And below is the code for the main file

```
<!DOCTYPE html>
<body>
<input type="text" id="x" value="2" />
<br />
<input type="text" id="y" value="3" />
<br />
<input type="text" id="output" />
<br />
<input type="button" id="multButton" value="Multiply" />
<input type="button" id="addButton" value="Add" />

<script>

/* Check if Web Workers are supported */
function getWebWorkerSupport() {
    return (typeof(Worker) !== "undefined") ? true:false;
}
if(getWebWorkerSupport() == true)
{
    var x,y,message;

    /* Create a new worker */
    arithmeticWorker = new Worker("arithmetic.js");

    /*
        Add a event listener to the worker, this will
        be called when the worker posts a message.
    */
    arithmeticWorker.onmessage = function (event) {
        document.getElementById("output").value = event.data;
    };

    /* Register events for buttons */
    document.getElementById("multButton").onclick = function() {
        /* Get the values to do operation on */
        x = parseFloat(document.getElementById("x").value);
        y = parseFloat(document.getElementById("y").value);
        message = {
            'op' : 'mult',
            'x' : x,
            'y' : y
        };
        arithmeticWorker.postMessage(message);
    }

    document.getElementById("addButton").onclick = function() {
        /* Get the values to do operation on */
        x = parseFloat(document.getElementById("x").value);
        y = parseFloat(document.getElementById("y").value);
        message = {
            'op' : 'add',
            'x' : x,
            'y' : y
        };
        arithmeticWorker.postMessage(message);
    }
}
```



```

    }
}
</script>
</body>
</html>

```

The thread doesn't terminate by itself after the main page starts it. The calling page has to explicitly ask the Worker to terminate. This may become necessary because creating each new Worker consumes precious browser resources, which you will need to reclaim once the Workers task is no longer required.

```

/* Terminate the Worker */
arithmeticWorker.terminate();

```

### 5.2.3 Security

#### Sandbox

Third Party Content such as advertisements, blog comments, and widgets, are very commonly found in today's websites. While they serve some purposes, and in some cases, enhance the user's experience, they also pose a serious security threat to both the website owner, and its users. This is because the inner working of this third party content is not always visible to the web developer, or the third party content may have its own security vulnerabilities. Hence threats such as phishing or information disclosure are a risk that has to be addressed by the web developer.

With Sandbox, HTML introduces new tools that will allow the web developers to restrict the access of these third party content. This is done by first placing the said content into an iframe. The developer can then specify the sandbox attribute on the iframe in order to apply a set of basic security restrictions:

```
<iframe src="untrusted.html" sandbox></iframe>
```

The **sandbox** attribute, when specified, enables a set of extra restrictions on any content hosted by the iframe. Some of the possible values for the sandbox attribute are:

- allow-scripts
- allow-forms
- allow-same-origin
- allow-top-navigation
- ms-allow-popups

When the attribute is set in default, the content is treated as being from a unique origin. This means that cookie access and local content access will be disabled. This can be overridden by specifying *allow-same-origin* in the sandbox attribute.

On default mode, all forms, popups, and scripts executions will also be disabled, and web developers can re-enable the activations of these features by specifying the *allow-forms*, *ms-allow-popups*, or *allow-scripts* into the sandbox attribute. The *allow-top-navigation* keyword on the other hand, allows the content to navigate its top-level browsing context

For example, consider the code below-

```
<iframe src="untrusted.html" sandbox="allow-scripts allow-forms"></iframe>
```

This particular iframe will be allowed to execute scripts and forms.

Hence using these allow tags, developers can create their own custom rules depending on the application's need.

To highlight the usefulness of Sandboxing, consider a website which hosts dynamic advertisement.

- The page which contains the advertisement may come with a script which tries to steal critical information from cookies. With sandbox, cookie access can be disabled.
- The advertisements can contain forms resembling the native website which can result in information disclosure. With sandbox, form submission can be disabled.
- Advertisements may contain scripts which can voluntarily or involuntarily land the user on a harmful malicious page. With sandbox, page redirection can be blocked.

By using Sandbox, web developers will be able to make their web application more secure , with little to no effort on his part.

## 6 CSS → CSS3

CSS3 has revolutionized the way websites are beautified. Effects which were painful to implement or even impossible to achieve in CSS2, have now been made possible using just a few lines of code in CSS3. This helps give web applications a very elegant, expressive and flexible design. Being the new generation of CSS after CSS2, CSS3 brings a host of new features. For instance, visual enhancements like rounded corners, gradients and shadows are some of the new effects that can be easily achieved using this new technology. Furthermore, one can also use the newly introduced transformation capabilities of CSS3 to achieve full blown animation. This is a completely novel innovation that CSS3 has introduced which opens up whole new avenues of styling options in a web application.

Unlike the previous versions of CSS, CSS3 is being released in modules and not as one big rollout. This allows specifications for new features to be completed and approved more quickly because segments are developed and approved in chunks. This is useful because it allows browser vendors to support whichever sections of the specification that are useful for their requirements. Additionally, CSS3 is completely backward compatible. Thus, developers do not have to rewrite their pre-existing CSS2 code for all their websites.

Thus, all in all, CSS3 equips the developer with a whole lot of easy-to-use tools which help in creating more beautiful web applications with minimal effort on part of the developer. Let us have a look at some of these new features.

### 6.1 Borders

CSS3 takes borders to a new level with the ability to use gradients, rounded corners, shadows and border images. We are going to look at each of these in a bit more detail, using examples where possible.

#### 6.1.1 Rounded Borders

Achieving rounded borders using current CSS coding can be tricky – there are numerous methods available, but none are extremely straight forward. Creating individual images for each border is often needed in addition. Using CSS3, creating a rounded border is incredibly easy. It can be applied to each corner or individual corners, and the width/colour are easily altered. The CSS code is:

```
.border_rounded {  
    background-color: #ddccb5;
```

```
border: 2px solid #897048;  
padding: 10px;  
width: 310px;  
}
```

This is an example of a box with rounded borders

### 6.1.2 Gradients

Gradient borders can look effective if used correctly. This code is a little more complex, requiring you to define each colour of the gradient. The CSS code is:

```
.border_gradient {  
  
border: 8px solid #000;  
  
border-bottom-colors: #897048 #917953 #a18a66 #b6a488 #c5b59b #d4c5ae #e2d6c4  
#eae1d2;  
  
border-top-colors: #897048 #917953 #a18a66 #b6a488 #c5b59b #d4c5ae #e2d6c4 #eae1d2;  
  
border-left-colors: #897048 #917953 #a18a66 #b6a488 #c5b59b #d4c5ae #e2d6c4 #eae1d2;  
  
border-right-colors: #897048 #917953 #a18a66 #b6a488 #c5b59b #d4c5ae #e2d6c4 #eae1d2;  
  
padding: 5px 5px 5px 15px;  
  
width: 300px;  
  
}
```

This is an example of a box with gradient border

### 6.1.3 Box Shadows

Adding a shadow to an element is difficult at present – it is a good way to differentiate a certain area, but as with any effect, it should be used sparingly. The CSS code is:

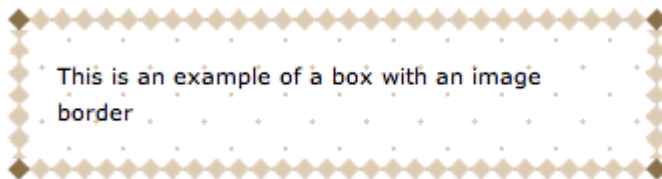
```
.border_shadow {  
  
padding: 5px 5px 5px 15px;  
  
width: 300px;  
  
}
```

This is an example of a box with a drop shadow

#### 6.1.4 Border Images

Border images are one of the most useful additions – I’m really looking forward to discovering how people choose to use them. CSS has the ability to repeat, or stretch a border image as you choose. The CSS code is similar to the following (it varies between browsers at present):

```
.border_image {  
  
border-image: url(border.png) 27 27 27 27 round round;  
  
}
```



## 6.2 Text Effects

CSS is already reasonably versatile in the way in which text can be displayed, but still constricts design in quite a few areas. CSS3 goes some way towards removing those limitations.

### 6.2.1 Text Shadow

CSS3 makes it very easy to provide a shadow drop to text elements, something which was not possible in the earlier versions of CSS.

```
.text_shadow {  
  
color: #897048;  
  
background-color: #fff;  
  
text-shadow: 2px 2px 2px #ddccb5;  
  
font-size: 15px;  
  
}
```

This is an example of text with a shadow applied

### 6.2.2 Web Fonts

Whilst these are not classed as a ‘text effect’, we are going to cover them here briefly. A Web Font is simply a way to use any font in your page, being downloaded automatically by the browser. This will be a revolutionary change to web design using CSS3, which previously has been limited to 10-15 widely supported fonts.

```
@font-face {
```

```
font-family: 'Name of the new font';

src: url('http://www.designshack.net/fonts/font.ttf');

}
```



## 6.3 Columns

Multiple columns are a major facet of laying out text – newspapers have used them for decades. So important are they that it is amazing that the current way to achieve a multi column layout is one of the most complex techniques for a new designer to grasp.

CSS3 introduces a new module known, appropriately, as multi-column layout. It allows you to specify how many columns text should be split down into and how they should appear.

### 6.3.1 Multiple columns using CSS3

There are four properties which relate to the multiple column layout in CSS3, allowing you to set the number of columns, width, gap between each column and a border between each *column-count*, *column-width*, *column-gap*, *column-rule*.

```
.multiplecolumns {
  column-width: 130px;
  column-gap: 20px;
  column-rule: 1px solid #ddccb5;
}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean egestas blandit ipsum. Morbi nulla metus, luctus et, ullamcorper sit amet, commodo quis, nisl. Ut blandit lacus nec nibh. Phasellus eleifend enim et risus. Nam condimentum. Praesent euismod auctor dui.

### Heading

Nunc ut leo vel magna adipiscing tempor. Donec pretium, ligula et hendrerit faucibus, sem velit accumsan tortor, sodales tempor est ligula non velit. Nulla sagittis, odio quis porta nonummy, mauris arcu gravida odio, quis aliquam lacus elit

non libero. Proin aliquam augue accumsan augue. Quisque ut eros at erat ultrices sodales. Nunc vitae ipsum. Mauris in elit in dolor imperdiet interdum. Vivamus egestas sagittis justo. Sed lorem. Sed vel neque in ipsum gravida nonummy. Nulla tempor blandit elit. Nullam a nibh. Nam quis diam non ligula pharetra sagittis. Maecenas rhoncus est vel tortor. Fusce in sem. Mauris in risus id lorem volutpat elementum. Pellentesque adipiscing laoreet ligula. Suspendisse erat. Donec porta auctor lacus. Vestibulum cursus, orci eget mollis ullamcorper, enim massa elementum dui, sed consequat nibh nisi eu tellus.

## 6.4 Transformation

A very exciting feature of CSS3 is its Transformation capabilities. This is something completely novel and thus gives the developer exciting new ways to make their website more beautiful. Transformation functions work on <div> elements and can achieve a variety of effects.

### 6.4.1 Skew

We can use this property to skew a <div> element by a certain angle. In the below example, the Google Logo has been skewed by an angle of 35 degrees.



```
#skew{  
    Transform: skew(35deg);  
}
```

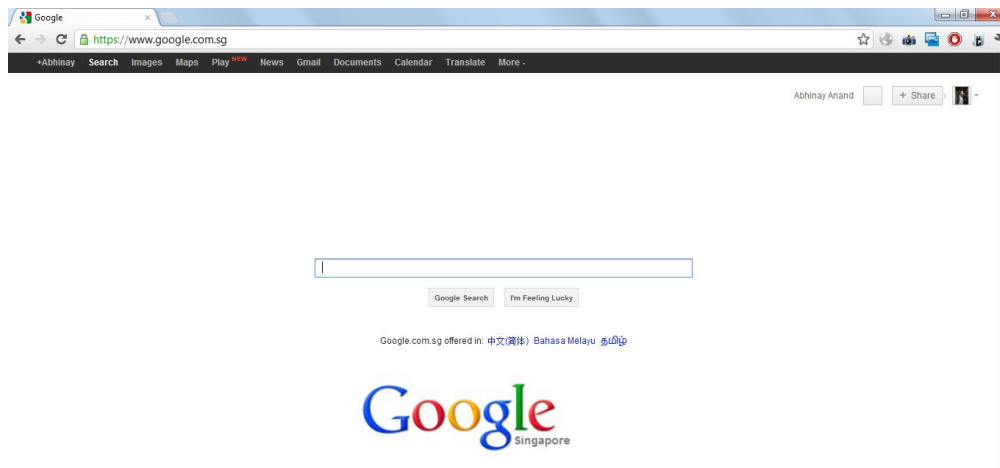
### 6.4.2 Rotate

We can use this property to rotate a <div> element by a certain angle. In the below example, the Google Logo has been rotated by an angle of 35 degrees.



```
#rotate{
    Transform: rotate(35deg);
}
```

### 6.4.3 Translate



We can use this property to move a <div> element from one point to another. In the below example, the Google Logo has been moved down by 100px.

```
#translate{
    Transform: translate(100px); }
```

## 7 SYNTACTICALLY AWESOME STYLESHEETS (SASS)

Sass is a CSS meta-language. It makes CSS full of fun again. It is basically an extension of CSS3 with added nested rules, variables, mixins and selector inheritance. It is a scripting language that is interpreted into CSS. Valid CSS is valid SASS with the same semantics. Various text editors have different level of support for SASS. All the major IDEs support it. Moreover, SASS also supports integration with the Firefox extension Firebug.

Sass has two syntaxes. The original syntax is the indented syntax that uses indentation to separate code blocks and newline characters to separate rules. The newer syntax, SCSS (Sassy CSS) uses block formatting like that of CSS with the use of curly brackets. These two syntazes are given the extensions .sass and .scss respectively.

To get started with Sass, we need to first install Ruby. Once we have Ruby, we can install Sass by running the command:

```
gem install sass
```

Then, open a new file called style.scss and add:

```
/* style.scss */
#navbar {
  width: 80%;
  height: 23px;
}
```

To translate this .scss to .css file, run:

```
sass --watch style.scss:style.css
```

From now on, whenever you make any changes to .scss file, .css file gets updated automatically. Now let's look at what additional features Sass provides:

## 7.1 Nesting

While writing CSS, instead of writing the style separately for #navbar ul and #navbar li, Sass allows the nesting of child selectors within the parent selectors. For example,

```
/* style.scss */

#navbar {
  width: 80%;
  height: 23px;

  ul { list-style-type: none; }
  li {
    float: left;
    a { font-weight: bold; }
  }
}
```

```
/* style.css */

#navbar {
  width: 80%;
  height: 23px; }
#navbar ul {
  list-style-type: none; }
#navbar li {
  float: left; }
#navbar li a {
  font-weight: bold; }
```

It is also possible to nest properties so that you do not have to repeat things like border-right all the time. So,

```
/* style.scss */

.fakeshadow {
  border: {
    style: solid;
    left: {
      width: 4px;
      color: #888;
    }
    right: {
      width: 2px;
      color: #ccc;
    }
  }
}
```

```
/* style.css */

.fakeshadow {
  border-style: solid;
  border-left-width: 4px;
  border-left-color: #888;
  border-right-width: 2px;
  border-right-color: #ccc; }
```

## 7.2 Variables

Sass also allows you to declare variables that can be used throughout the stylesheet. Variables begin with \$ and accepts all the values that a CSS property does, such as colors, numbers (with units), or text. For example,



```
/* style.scss */

$main-color: #ce4dd6;
$style: solid;

#navbar {
  border-bottom: {
    color: $main-color;
    style: $style;
  }
}

a {
  color: $main-color;
  a:hover { border-bottom: $style 1px; }
}
```

```
/* style.css */

#navbar {
  border-bottom-color: #ce4dd6;
  border-bottom-style: solid; }

a {
  color: #ce4dd6; }
a:hover {
  border-bottom: solid 1px; }
```

Moreover, Sass also supports the standard math operations (+, -, \*, /, and %) on numbers and functions like lightness, hue and saturation for colors. Also, variables can be used to insert into property names or selectors similar to #define in high level programming languages.

### 7.3 Mixins

Mixins are the most powerful Sass features. They allow the re-use of styles – properties or even selectors – without having to copy or paste them or move them into a non-semantic class.

Mixins are defined using the “@mixin” directive, which takes a block of styles that can then be included in another selector using the “@include” directive. Also, it is possible to pass arguments to the mixins. For example,

```
/* style.scss */

@mixin rounded($side, $radius: 10px) {
  border-#{ $side }-radius: $radius;
  -moz-border-radius-#{ $side }: $radius;
  -webkit-border-#{ $side }-radius: $radius;
}

#navbar li { @include rounded(top); }
#footer { @include rounded(top, 5px); }
#sidebar { @include rounded(left, 8px); }
```

```
/* style.css */

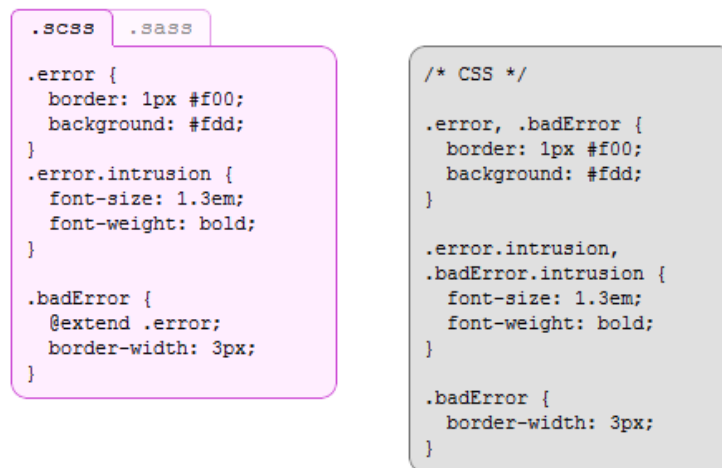
#navbar li {
  border-top-radius: 10px;
  -moz-border-radius-top: 10px;
  -webkit-border-top-radius: 10px; }

#footer {
  border-top-radius: 5px;
  -moz-border-radius-top: 5px;
  -webkit-border-top-radius: 5px; }

#sidebar {
  border-left-radius: 8px;
  -moz-border-radius-left: 8px;
  -webkit-border-left-radius: 8px; }
```

### 7.4 Selector Inheritance

Sass can tell one selector to inherit all the styles of another without duplicating the CSS properties. For example,



The image shows two code snippets side-by-side. The left snippet is in a pink box with tabs for '.scss' and '.sass', showing SCSS code. The right snippet is in a grey box showing the equivalent SASS code. The SCSS code uses curly braces for all rules and includes the @extend directive. The SASS code uses curly braces for rules but uses a comma to separate selectors and does not use @extend.

```
.scss
.error {
  border: 1px #f00;
  background: #fdd;
}
.error.intrusion {
  font-size: 1.3em;
  font-weight: bold;
}

.badError {
  @extend .error;
  border-width: 3px;
}
```

```
/* CSS */
.error, .badError {
  border: 1px #f00;
  background: #fdd;
}

.error.intrusion,
.badError.intrusion {
  font-size: 1.3em;
  font-weight: bold;
}

.badError {
  border-width: 3px;
}
```

## 8 CONCLUSION

In this chapter, we have introduced the general concepts in Web development. We have discussed the complete overview of Web development, with its respective components. We have also introduced some of the tools that web developers can use to simplify the web development process, namely Web Frameworks and Libraries.

In the preceding section, we also discussed in greater detail the three new technologies in Web Development that are HTML 5, CSS 3, and SASS. We looked at the capabilities of these new features, and some exciting possibilities that can be obtained using these new technologies.

With these in mind we would like encourage you to explore more about these technologies and use them in their next Web Development project, because these new technologies will be the future of Web Development.

## 9 RESOURCES

### HTML5

<http://www.pageresource.com/html5/offline-api/>

Canvas Tutorial: [https://developer.mozilla.org/en/Canvas\\_tutorial](https://developer.mozilla.org/en/Canvas_tutorial)

Wikipedia, HTML5 Canvas: [http://en.wikipedia.org/wiki/Canvas\\_element](http://en.wikipedia.org/wiki/Canvas_element)

HTML5 Rocks: <http://www.html5rocks.com/>

Introducing HTML5 Web Workers: <http://www.codediesel.com/javascript/introducing-html5-web-workers/>

W3 Schools: <http://www.w3schools.com/>

### CSS3

Step by step tutorial to learn all about CSS3: <http://www.w3schools.com/css3/default.asp>

A collection of impressive CSS3 demos: <http://webdesignerwall.com/trends/47-amazing-css3-animation-demos>

Book: Hogan, Brian P., “HTML5 and CSS3”, “Develop with Tomorrow’s Standards Today”

### SASS

SASS, ‘Style with attitude’: <http://sass-lang.com/>

Book: Catlin Hampton, Catlin Michael Lintorn, “Pragmatic Guide to SASS”