Web Development

Scott Buchanan

First Edition, 2012

ISBN 978-81-323-4262-5

Published by:
White Word Publications
4735/22 Prakashdeep Bldg,
Ansari Road, Darya Ganj,
Delhi - 110002
Email: info@wtbooks.com

Table of Contents

Chapter 1

Web Development

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.

For larger organizations and businesses, web development teams can consist of hundreds of people (web developers). Smaller organizations may only require a single permanent or contracting webmaster, or secondary assignment to related job positions such as a graphic designer and/or information systems technician. Web development may be a collaborative effort between departments rather than the domain of a designated department.

Web development as an industry

Since the mid-1990s, web development has been one of the fastest growing industries in the world. In 1995 there were fewer than 1,000 web development companies in the United States, but by 2005 there were over 30,000 such companies in the U.S. alone. The growth of this industry is being pushed by large businesses wishing to sell products and services to their customers and to automate business workflow.

In addition, cost of Web site development and hosting has dropped dramatically during this time. Instead of costing ten thousands of dollars, as was the case for early websites, one can now develop a simple web site for less than a thousand dollars, depending on the complexity and amount of content. Smaller Web site development companies are now able to make web design accessible to both smaller companies and individuals further fueling the growth of the web development industry. As far as web development tools and platforms are concerned, there are many systems available to the public free of charge to aid in development. A popular example is the LAMP (Linux, Apache, MySQL, PHP) stack, which is usually distributed free of charge. This fact alone has manifested into many people around the globe setting up new Web sites daily and thus contributing to increase in web development popularity. Another contributing factor has been the rise of easy to use WYSIWYG web development software, most prominently Adobe Dreamweaver, Netbeans, WebDev, or Microsoft Expression Studio, Adobe Flex. Using such software, virtually anyone can develop a Web page in a matter of minutes. Knowledge of HyperText Markup Language (HTML), or other programming languages is not required, but recommended for professional results.

The next generation of web development tools uses the strong growth in LAMP, Java Platform, Enterprise Edition technologies and Microsoft .NET technologies to provide the Web as a way to run applications online. Web developers now help to deliver applications as Web services which were traditionally only available as applications on a desk based computer.

Instead of running executable code on a local computer, users are interacting with online applications to create new content. This has created new methods in communication and allowed for many opportunities to decentralize information and media distribution. Users are now able to interact with applications from many locations, instead of being tied to a specific workstation for their application environment.

Examples of dramatic transformation in communication and commerce led by web development include e-commerce. Online auction sites such as eBay have changed the way consumers consume and purchase goods and services. Online resellers such as Amazon.com and Buy.com (among many, many others) have transformed the shopping and bargain hunting experience for many consumers. Another good example of transformative communication led by web development is the blog. Web applications such as WordPress and Movable Type have created easily implemented blog environments for individual Web sites. Open source content management systems such as Joomla!, Drupal, XOOPS, and TYPO3 and enterprise content management systems such as Alfresco have extended web development into new modes of interaction and communication.

In addition, web development has moved to a new phase of Internet communication. Computer web sites are no longer simply tools for work or commerce but used most for communication. Websites such as Facebook and Twitter provide users a platform to freely communicate. This new form of web communication is also changing e-commerce through the number of hits and online advertisement.

Typical Areas

Web Development can be split into many areas and a typical and basic web development hierarchy might consist of:

Client Side Coding

- Ajax Asynchronous JavaScript provides new methods of using JavaScript, and other languages to improve the user experience.
- Flash Adobe Flash Player is an ubiquitous browser plugin ready for RIAs. Flex 2 is also deployed to the Flash Player (version 9+).
- JavaScript Formally called ECMAScript, JavaScript is a ubiquitous client side platform for creating and delivering rich Web applications that can also run across a wide variety of devices.
- Microsoft Silverlight Microsoft's browser plugin that enables animation, vector graphics and high-definition video playback, programmed using XAML and .NET programming languages.
- REAL Studio Web Edition is a rapid application development environment for the web. The language is object oriented and is similar to both VB and Java. Applications are uniquely compiled to binary code.
- HTML5 and CSS3 Latest HTML proposed standard combined with the latest proposed standard for CSS natively supports much of the client-side functionality provided by other frameworks such as Flash and Silverlight

Server Side Coding

- ASP (Microsoft proprietary)
- CSP, Server-Side ANSI C
- ColdFusion (Adobe proprietary, formerly Macromedia, formerly Allaire)
- CGI and/or Perl (open source)
- Groovy (programming language) Grails (framework)
- Java, e.g. Java EE or WebObjects
- Lotus Domino
- PHP (open source)
- Python, e.g. Django (web framework) (open source)
- REAL Studio Web Edition
- Ruby, e.g. Ruby on Rails (open source)
- Smalltalk e.g. Seaside, AIDA/Web
- SSJS Server-Side JavaScript, e.g. Aptana Jaxer, Mozilla Rhino
- Websphere (IBM proprietary)
- .NET (Microsoft proprietary)

The World Wide Web has become a major delivery platform for web development a variety of complex and sophisticated enterprise applications in several domains. In addition to their inherent multifaceted functionality, these web applications exhibit complex behavior and place some unique demands on their usability, performance, security and ability to grow and evolve. However, a vast majority of these applications continue to be developed in an ad-hoc way, contributing to problems of usability, maintainability, quality and reliability.(1)(2) While web development can benefit from established practices from other related disciplines, it has certain distinguishing characteristics that demand special considerations. In recent years of web development there have been some developments towards addressing these problems and requirements. As an emerging discipline, web engineering actively promotes systematic, disciplined and quantifiable approaches towards successful development of high-quality, ubiquitously usable web-based systems and applications.(3)(4) In particular, web engineering focuses on the methodologies, techniques and tools that are the foundation of web application development and which support their design, development, evolution, and evaluation. Web application development has certain characteristics that make it different from traditional software, information system, or computer application development.

Web engineering is multidisciplinary and encompasses contributions from diverse areas: systems analysis and design, software engineering, hypermedia/hypertext engineering, requirements engineering, human-computer interaction, user interface, information engineering, information indexing and retrieval, testing, modelling and simulation, project management, and graphic design and presentation. Web engineering is neither a clone, nor a subset of software engineering, although both involve programming and software development. While web engineering uses software engineering principles, web development encompasses new approaches, methodologies, tools, techniques, and guidelines to meet the unique requirements for web-based applications.

Client Side + Server Side

- Google Web Toolkit provides tools to create and maintain complex JavaScript front-end applications in Java.
- Pyjamas is a tool and framework for developing Ajax applications and Rich Internet Applications in python.
- Tersus is a platform for the development of rich web applications by visually defining user interface, client side behavior and server side processing. (open source)

However lesser known languages like Ruby and Python are often paired with database servers other than MySQL (the M in LAMP). Below are example of other databases currently in wide use on the web. For instance some developers prefer a LAPR(Linux/Apache/PostgreSQL/Ruby on Rails) setup for development.

Database Technology

- Apache Derby
- DB2 (IBM proprietary)
- Firebird
- Microsoft SQL Server
- MySQL
- Oracle
- PostgreSQL
- SQLite
- Sybase

In practice, many web developers will also have **interdisciplinary** skills / roles, including:

- Graphic design / web design
- Information architecture and copywriting/copyediting with web usability, accessibility and search engine optimization in mind
- Project management, QA and other aspects common to IT development in general

The above list is a simple website development hierarchy and can be extended to include all client side and server side aspects. It is still important to remember that web development is generally split up into client side coding, covering aspects such as the layout and design, and server side coding, which covers the website's functionality and back end systems.

Looking at these items from an "umbrella approach", client side coding such as XHTML is executed and stored on a local client (in a web browser) whereas server side code is not available to a client and is executed on a web server which generates the appropriate XHTML which is then sent to the client. The nature of client side coding allows you to alter the HTML on a local client and refresh the pages with updated content (locally), web designers must bear in mind the importance and relevance to security with their server side scripts. If a server side script accepts content from a locally modified client side script, the web development of that page is poorly sanitized with relation to security.

Security Considerations

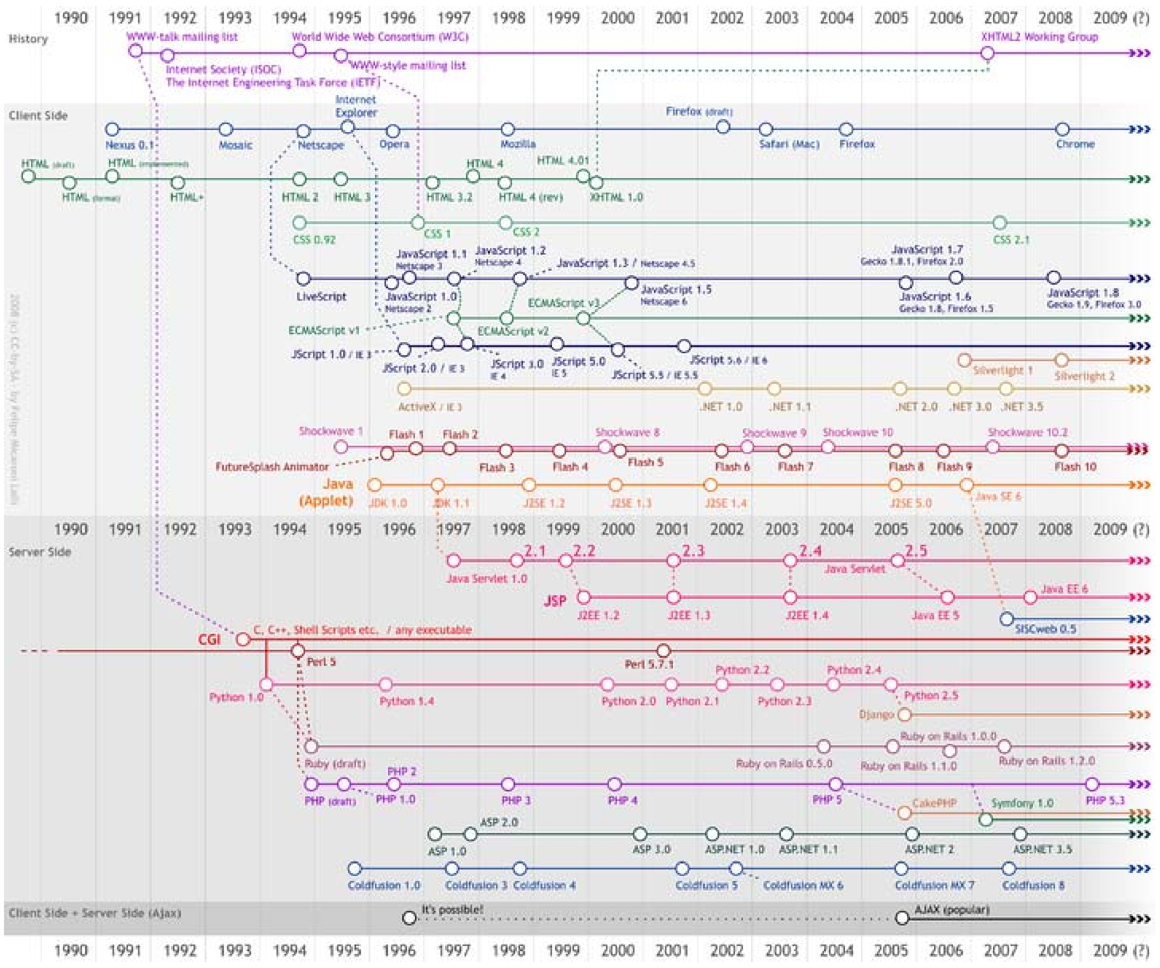
Web development takes into account many security considerations, such as data entry error checking through forms, filtering output, and encryption. Malicious practices such as SQL injection can be executed by users with ill intent yet with only primitive knowledge of web development as a whole. Scripts can be exploited to grant unauthorized access to malicious users trying to collect information such as email addresses, passwords and protected content like credit card numbers.

Some of this is dependent on the server environment (most commonly Apache or Microsoft IIS) on which the scripting language, such as PHP, Ruby, Python, Perl or ASP is running, and therefore is not necessarily down to the web developer themselves to maintain. However, stringent testing of web applications before public release is encouraged to prevent such exploits from occurring.

Keeping a web server safe from intrusion is often called *Server Port Hardening*. Many technologies come into play when keeping information on the internet safe when it is transmitted from one location to another. For instance Secure Socket Layer Encryption (SSL) Certificates are issued by certificate authorities to help prevent internet fraud. Many developers often employ different forms of encryption when transmitting and storing sensitive information. A basic understanding of information technology security concerns is often part of a web developer's knowledge.

Because new security holes are found in web applications even after testing and launch, security patch updates are frequent for widely used applications. It is often the job of web developers to keep applications up to date as security patches are released and new security concerns are discovered.

Timeline



Chapter 2

Ajax (Programming) and Web Syndication

Ajax (programming)

Ajax is a group of interrelated web development methods used on the client-side to create interactive web applications. With Ajax, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behavior of the existing page. Data is usually retrieved using the *XMLHttpRequest* object. Despite the name, the use of XML is not needed, and the requests need not be asynchronous.

Like DHTML and LAMP, Ajax is not one technology, but a group of technologies. Ajax uses a combination of HTML and CSS to mark up and style information. The DOM is accessed with JavaScript to dynamically display, and to allow the user to interact with the information presented. JavaScript and the XMLHttpRequest object provide a method for exchanging data asynchronously between browser and server to avoid full page reloads.

History

In the 1990s, most web sites were based on complete HTML pages; each user action required that the page be re-loaded from the server (or a new page loaded). This process is inefficient, as reflected by the user experience: all page content disappears then reappears, etc. Each time a page is reloaded due to a partial change, all of the content must be re-sent instead of only the changed information. This can place additional load on the server and use excessive bandwidth.

Asynchronous loading of content first became practical when Java applets were introduced in the first version of the Java language in 1995. These allow compiled client-side code to load data asynchronously from the web server after a web page is loaded. In 1996, Internet Explorer introduced the iframe element to HTML, which also enabled asynchronous loading. In 1999, Microsoft created the XMLHTTP ActiveX control in Internet Explorer 5, which was later adopted by Mozilla, Safari, Opera and other browsers as the XMLHttpRequest JavaScript object. Microsoft has adopted the native XMLHttpRequest model as of Internet Explorer 7, though the ActiveX version is still supported. The utility of background HTTP requests to the server and asynchronous web technologies remained fairly obscure until it started appearing in full scale online applications such as Outlook Web Access (2000) and Oddpost (2002), and later, Google made a wide deployment of Ajax with Gmail (2004) and Google Maps (2005).

The term *Ajax* was coined on February 18, 2005 by Jesse James Garrett in an article entitled *Ajax: A New Approach to Web Applications*.

On April 5, 2006 the World Wide Web Consortium (W3C) released the first draft specification for the XMLHttpRequest object in an attempt to create an official web standard.

Technologies

The term *Ajax* has come to represent a broad group of web technologies that can be used to implement a web application that communicates with a server in the background, without interfering with the current state of the page. In the chapter that coined the term Ajax, Jesse James Garrett explained that the following technologies are incorporated:

- HTML or XHTML and CSS for presentation
- the Document Object Model (DOM) for dynamic display of and interaction with data
- XML for the interchange of data, and XSLT for its manipulation
- the XMLHttpRequest object for asynchronous communication
- JavaScript to bring these technologies together

Since then, however, there have been a number of developments in the technologies used in an Ajax application, and the definition of the term Ajax. In particular, it has been noted that JavaScript is not the only client-side scripting language that can be used for implementing an Ajax application; other languages such as VBScript are also capable of the required functionality. (However, JavaScript is the most popular language for Ajax programming due to its inclusion in and compatibility with the majority of modern web browsers.) Also, XML is not required for data interchange and therefore XSLT is not required for the manipulation of data. JavaScript Object Notation (JSON) is often used as an alternative format for data interchange, although other formats such as preformatted HTML or plain text can also be used.

Drawbacks

- Pages dynamically created using successive Ajax requests do not automatically register themselves with the browser's history engine, so clicking the browser's "back" button may not return the browser to an earlier state of the Ajax-enabled page, but may instead return to the last full page visited before it. Workarounds include the use of invisible iframes to trigger changes in the browser's history and changing the URL fragment identifier (the part of a URL after the '#') when Ajax is run and monitoring it for changes.

- Dynamic web page updates also make it difficult to bookmark a particular state of the application. Solutions to this problem exist, many of which use the URL fragment identifier (the part of a URL after the '#') to keep track of, and allow returning to, the application in a given state.
- Depending on the nature of the Ajax application, dynamic page updates may interfere disruptively with user interactions, especially if working on an unstable internet connection. For instance, editing a search field may trigger a query to the server for search completions, but the user may not know that a search completion popup is forthcoming, and if the internet connection is slow, the popup list may show up at an inconvenient time, when the user has already proceeded to do something else.
- Because most web crawlers do not execute JavaScript code, publicly indexable web applications should provide an alternative means of accessing the content that would normally be retrieved with Ajax, thereby allowing search engines to index it.
- Any user whose browser does not support JavaScript or XMLHttpRequest, or simply has this functionality disabled, will not be able to properly use pages which depend on Ajax. Similarly, devices such as mobile phones, PDAs, and screen readers may not have support for the required technologies. Screen readers that are able to use Ajax may still not be able to properly read the dynamically generated content. The only way to let the user carry out functionality is to fall back to non-JavaScript methods. This can be achieved by making sure links and forms can be resolved properly and do not rely solely on Ajax.
- The same origin policy prevents some Ajax techniques from being used across domains, although the W3C has a draft of the XMLHttpRequest object that would enable this functionality. Methods exist to sidestep this security feature by using a special Cross Domain Communications channel embedded as an iframe within a page, or by the use of JSONP.

- Ajax-powered interfaces may dramatically increase the number of user-generated requests to web servers and their back-ends (databases, or other). This can lead to longer response times and/or additional hardware needs.
- The asynchronous, callback-style of programming required can lead to complex code that is hard to maintain or debug.

- Ajax cannot easily be read by screen-reading technologies, such as JAWS, without hints built into the corresponding HTML based on WAI-ARIA standards. Screen-reading technologies are used by individuals with an impairment that hinders or prevents the ability to read the content on a screen.

Web syndication

Common web feed icon

Web syndication is a form of syndication in which website material is made available to multiple other sites. Most commonly, *web syndication* refers to making web feeds available from a site in order to provide other people with a summary or update of the website's recently added content (for example, the latest news or forum posts). The term can also be used to describe other kinds of licensing website content so that other websites can use it.

Motivation

Syndication benefits both the websites providing information and the websites displaying it. For the receiving site, content syndication is an effective way of adding greater depth and immediacy of information to its pages, making it more attractive to users. For the transmitting site, syndication drives exposure across numerous online platforms. This generates new traffic for the transmitting site — making syndication a relatively cheap, free and easy form of advertisement. However, as Search Engine Optimization has become an increasingly important topic among website owners and online marketers, content syndication has become a highly effective strategy for link building. Links embedded within the syndicated content are typically optimized around anchor terms that will point an optimized link back to the website that the content author is trying to promote. These links tell the algorithms of the search engines that the website being linked to is an authority for the keyword that is being used as the anchor text.

The prevalence of web syndication is also of note to online marketers, since web surfers are becoming increasingly wary of providing personal information for marketing materials (such as signing up for a newsletter) and expect the ability to subscribe to a feed instead. Although the format could be anything transported over HTTP, such as HTML or JavaScript, it is more commonly XML. The two main families of web syndication formats are RSS and Atom.

History

The basic idea of restructuring information about web sites goes back to as early as 1995, when Ramanathan V. Guha and others in Apple Computer's Advanced Technology Group developed the Meta Content Framework (MCF).

Large scale web syndication of content started in 1999 when Studio One Networks produced and distributed the first series of syndicated programs designed to be distributed on the Internet for its sponsor American Honda. Nowadays, many different types of content are syndicated on the Internet. Millions of online publishers, including newspapers, commercial websites and blogs, now publish their latest news headlines, product offers or blog postings in standard format news feed.

Web syndication as a commercial model

In addition to freely distributed material, some broadcasters and others use similar methods for the controlled placement of proprietary content on multiple partnering Internet destinations. In addition to web feeds, such commercial syndicators may use other methods to distribute their content such as Reuters, Associated Press, and All Headline News.

Such commercial web syndication borrows its business models from syndication in other media, such as Print, radio and television. Primarily, syndication arose in those other media so that content creators could reach a wider audience. In the case of radio, the United States Federal government proposed a syndicate in 1924 so that the nation's executives could quickly and efficiently reach the entire population. In the case of television, it is often said that "Syndication is where the real money is." Additionally, syndication accounts for the bulk of TV programming.

Commercial web syndication can be categorized in three ways:

- by **business models**
- by **types of content**
- by **methods for selecting distribution partners**

Commercial web syndication involves partnerships between content producers and distribution outlets. There are different structures of partnership agreements. One such structure is licensing content, in which distribution partners pay a fee to the content creators for the right to publish the content. Another structure is ad-supported content, in which publishers share revenues derived from advertising on syndicated content with that content's producer. A third structure is free, or barter syndication, in which no currency changes hands between publishers and content producers. This requires the content producers to generate revenue from another source, such as embedded advertising or subscriptions. Alternatively, they could distribute content without remuneration. Typically, those who create and distribute content for free are promotional entities, vanity publishers or government entities.

Types of content syndicated include RSS or Atom Feeds and full content. With RSS feeds, headlines, summaries, and sometimes a modified version of the original full content is displayed on users' feed readers. With full content, the entire content—which might be text, audio, video, applications/widgets or user-generated content—appears unaltered on the publisher's site.

There are two methods for selecting distribution partners. The content creator can hand-pick syndication partners based on specific criteria, such as the size or quality of their audiences. Alternatively, the content creator can allow publisher sites or users to "opt in" to carrying the content through an automated system. Some of these automated "content marketplace" systems involve careful screening of potential publishers by the content creator to ensure that the material does not end up in an inappropriate environment.

Just as syndication is a source of profit for TV producers and radio producers, it also functions to maximize profit for Internet content producers. As the Internet has increased in size it has become increasingly difficult for content producers to aggregate a sufficiently large audience to support the creation of high-quality content. Syndication enables content creators to amortize the cost of producing content by licensing it across multiple publishers or by maximizing distribution of advertising-supported content. However, a potential drawback for content creators is that they can lose control over the presentation of their content when they syndicate it to other parties.

Distribution partners benefit by receiving content either at a discounted price, or for free. One potential drawback for publishers, however, is that because the content is duplicated at other publisher sites, they cannot have an "exclusive" on the content.

For users, the fact that syndication enables the production and maintenance of content allows them to find and consume content on the Internet. One potential drawback for them is that they may run into duplicate content, which could be an annoyance.

Web syndication and e-commerce

Similar to syndication of proprietary content, web syndication has been used to distribute product content (feature descriptions, images, specifications etc.). Given that manufacturers are regarded as authorities and that most sales do not happen on manufacturer Web sites, best-in-class manufacturers take their best content and enable other retailers or dealers to publish the information on their sites. By syndicating content, a manufacturer is more likely to pass consistent and often comprehensive information to channel partners. Such web syndication has been shown to generate an increase in sales.

Web syndication has been increasingly used as a way to syndicate online news content to websites, too as part of Search Engine Optimisation techniques. Adding news content lets websites target specific keywords that could cost a fortune in PPC campaigns absolutely free. Also, by the very nature of writing about a topic - for example the UK housing market - you naturally cover other keyword variations in the same page, eg. house prices, mortgages, first-time buyers - all in one page: all keywords that someone interested in the

UK housing market could type into a search engine also. Done on a daily basis, over time, news content will improve your search engine ranking considerably.

Chapter 3

Web Application Framework

A **web application framework** is a software framework that is designed to support the development of dynamic websites, web applications and web services. The framework aims to alleviate the overhead associated with common activities performed in Web development. For example, many frameworks provide libraries for database access, templating frameworks and session management, and they often promote code reuse.

History

As the design of the World Wide Web was not inherently dynamic, early hypertext consisted of hand-coded HTML that was published on web servers. Any modifications to published pages needed to be performed by the pages' author. To provide a dynamic web page that reflected user inputs, the Common Gateway Interface (CGI) standard was introduced for interfacing external applications with web servers. CGI could adversely affect server load, though, since each request had to start a separate process.

Programmers wanted tighter integration with the web server to enable high traffic web applications. The Apache HTTP Server, for example, supports modules that can extend the web server with arbitrary code executions (such as mod perl) or forward specific requests to a web server that can handle dynamic content (such as mod jk). Some web servers (such as Apache Tomcat) were specifically designed to handle dynamic content by executing code written in some languages, such as Java.

Around the same time, new languages were being developed specifically for use in the web, such as ColdFusion, PHP and Active Server Pages.

While the vast majority of languages available to programmers to use in creating dynamic web pages have libraries to help with common tasks, web applications often require specific libraries that are useful in web applications, such as creating HTML (for example, JavaServer Faces).

Eventually, mature, "full stack" frameworks appeared, that often gathered multiple libraries useful for web development into a single cohesive software stack for web developers to use. Examples of this include JavaEE (Servlets), WebObjects, OpenACS, Catalyst, Ruby on Rails, Django, and Zend Framework.

Architectures

Most web application frameworks are based on the MVC pattern. From an architecture perspective, there are generally five major types: request-based, component-based, hybrid, meta, and RIA-based.

Model view controller (MVC)

Many frameworks follow the Model View Controller (MVC) architectural pattern to separate the data model with business rules from user interface. This is generally considered a good practice as it modularizes code, promotes code reuse, and allows multiple interfaces to be applied.

Push-based vs. Pull-based

Most MVC frameworks follow a push-based architecture. These frameworks use actions that do the required processing, and then "push" the data to the view layer to render the results. Struts, Django, Ruby on Rails and Spring MVC are good examples of this architecture. An alternative to this is pull-based architecture, sometimes also called "component-based". These frameworks start with the view layer, which can then "pull" results from multiple controllers as needed. In this architecture, multiple controllers can be involved with a single view. Struts2, Lift, Tapestry, JBoss Seam, Wicket and Stripes are examples of pull-based architectures.

Content Management Systems

Some projects that have historically been termed content management systems have begun to take on the roles of higher-layer web application frameworks. For instance, Drupal's structure provides a minimal *core* whose function is extended through *modules* that provide functions generally associated with web application frameworks. However, it is debatable whether "management of content" is the primary value of such systems, especially when some, like SilverStripe, provide an object-oriented MVC framework. Add-on *modules* now enable these systems to function as full fledged applications beyond the scope of content management. They may provide functional APIs, functional frameworks, coding standards, and many of the functions traditionally associated with *Web application frameworks*.

Features

Web template system

Dynamic web pages usually consist of a static part (HTML) and a dynamic part, which is code that generates HTML. The code that generates the HTML can do this based on variables in a template, or on code. The text to be generated can come from a database, thereby making it possible to dramatically reduce the number of pages in a site.

Consider the example of a real estate agent with 500 houses for sale. In a static web site, the agent would have to create 500 pages in order to make the information available. In a dynamic website, the agent would simply connect the dynamic page to a database table of 500 records.

In a template, variables from the programming language can be inserted without using code, thereby losing the requirement of programming knowledge to make updates to the pages in a web site. A syntax is made available to distinguish between HTML and variables. E.g. in JSP the <c:out> tag is used to output variables, and in Smarty, {$variable} is used.

Many template engines do support limited logic tags, like IF and FOREACH. These are to be used only for decisions that need to be made for the presentation layer, in order to keep a clean separation from the business logic layer, or the M(odel) in the MVC pattern.

Caching

Web caching is the caching of web documents in order to reduce bandwidth usage, server load, and perceived "lag". A web cache stores copies of documents passing through it; subsequent requests may be satisfied from the cache if certain conditions are met. Some application frameworks provide mechanisms for caching documents and bypassing various stages of the page's preparation, such as database access or template interpretation.

Security

Some web application frameworks come with authentication and authorization frameworks, that enable the web server to identify the users of the application, and restrict access to functions based on some defined criteria. Drupal is one example that provides role-based access to pages, and provides a web-based interface for creating users and assigning them roles.

Database access and mapping

Many web application frameworks create a unified API to a database backend, enabling web applications to work with a variety of databases with no code changes, and allowing programmers to work with higher-level concepts. For higher performance, database connections should be pooled as e.g. AOLserver does. Additionally, some object-oriented frameworks contain mapping tools to provide Object-Relational Mapping, which will map objects to tuples.

Other features web application frameworks may provide include transactional support and database migration tools.

URL mapping

A framework's URL mapping facility is the mechanism by which the framework interprets URLs. Some frameworks, such as Drupal and Django, match the provided URL against pre-determined patterns using regular expressions, while some others use URL Rewriting to translate the provided URL into one that the underlying engine will recognize. Another technique is that of graph traversal such as used by Zope, where a URL is decomposed in steps that traverse an object graph (of models and views).

A URL mapping system that uses pattern matching or URL rewriting allows more "friendly" URLs to be used, increasing the simplicity of the site and allowing for better indexing by search engines. For example, a URL that ends with "/page.cgi?cat=science&topic=physics" could be changed to simply "/page/science/physics". This makes the URL easier to read and provides search engines with better information about the structural layout of the site. A graph traversal approach also tends to result in the creation of friendly URLs. A shorter URL such as "/page/science" tends to exist by default as that is simply a shorter form of the longer traversal to "/page/science/physics".

Ajax

Ajax, shorthand for "*Asynchronous JavaScript and XML*", is a web development technique for creating interactive web applications. The intent is to make web pages feel more responsive by exchanging small amounts of data with the server behind the scenes, so that the entire web page does not have to be reloaded each time the user requests a change. This is intended to increase the web page's interactivity, speed, and usability.

Due to the complexity of Ajax programming in Javascript, there are numerous Ajax frameworks that exclusively deal with Ajax support. Some Ajax frameworks are even embedded as a part of larger frameworks. For example, the Prototype JavaScript Framework is included in Ruby on Rails.

With the increased interest in developing "Web 2.0" Rich Media Applications, the complexity of programming directly in Ajax and Javascript has become so apparent that compiler technology has stepped in, to allow developers to code in high-level languages such as Java, Python and Ruby. The first of these compilers was Morfik followed by Google Web Toolkit, with ports to Python and Ruby in the form of Pyjamas and RubyJS following some time after. These compilers and their associated widget set libraries make the development of Rich Media Ajax Applications much more akin to that of developing Desktop applications.

Automatic configuration

Some frameworks minimize web application configuration through the use of introspection and/or following known conventions. For example, many Java frameworks use Hibernate as a persistence layer, which can generate a database schema at runtime capable of persisting the necessary information. This allows the application designer to design business objects without needing to explicitly define a database schema. Frameworks such as Ruby on Rails can also work in reverse, that is, define properties of model objects at runtime based on a database schema.

Web services

Some frameworks provide tools for creating and providing web services. These utilities may offer similar tools as the rest of the web application.

Chapter 4

Web Analytics

Web analytics is the measurement, collection, analysis and reporting of internet data for purposes of understanding and optimizing web usage.

Web analytics is not just a tool for measuring website traffic but can be used as a tool for business research and market research. Web analytics applications can also help companies measure the results of traditional print advertising campaigns. It helps one to estimate how the traffic to the website changed after the launch of a new advertising campaign. Web analytics provides data on the number of visitors, page views, etc. to gauge the traffic and popularity trends which helps doing the market research.

There are two categories of web analytics; *off-site* and *on-site* web analytics.

Off-site web analytics refers to web measurement and analysis regardless of whether you own or maintain a website. It includes the measurement of a website's *potential* audience (opportunity), share of voice (visibility), and buzz (comments) that is happening on the Internet as a whole.

On-site web analytics measure a visitor's journey once *on your website*. This includes its drivers and conversions; for example, which landing pages encourage people to make a purchase. On-site web analytics measures the performance of your website in a commercial context. This data is typically compared against key performance indicators for performance, and used to improve a web site or marketing campaign's audience response.

Historically, web analytics has referred to on-site visitor measurement. However in recent years this has blurred, mainly because vendors are producing tools that span both categories.

On-site web analytics technologies

Many different vendors provide on-site web analytics software and services. There are two main technological approaches to collecting the data. The first method, *logfile analysis*, reads the logfiles in which the web server records all its transactions. The second method, *page tagging*, uses JavaScript on each page to notify a third-party server when a page is rendered by a web browser. Both collect data that can be processed to produce web traffic reports.

In addition other data sources may also be added to augment the data. For example; e-mail response rates, direct mail campaign data, sales and lead information, user performance data such as click heat mapping, or other custom metrics as needed.

Web server logfile analysis

Web servers record some of their transactions in a logfile. It was soon realized that these logfiles could be read by a program to provide data on the popularity of the website. Thus arose web log analysis software.

In the early 1990s, web site statistics consisted primarily of counting the number of client requests (or *hits*) made to the web server. This was a reasonable method initially, since each web site often consisted of a single HTML file. However, with the introduction of images in HTML, and web sites that spanned multiple HTML files, this count became less useful. The first true commercial Log Analyzer was released by IPRO in 1994 .

Two units of measure were introduced in the mid 1990s to gauge more accurately the amount of human activity on web servers. These were *page views* and *visits* (or *sessions*). A *page view* was defined as a request made to the web server for a page, as opposed to a graphic, while a *visit* was defined as a sequence of requests from a uniquely identified client that expired after a certain amount of inactivity, usually 30 minutes. The page views and visits are still commonly displayed metrics, but are now considered rather rudimentary.

The emergence of search engine spiders and robots in the late 1990s, along with web proxies and dynamically assigned IP addresses for large companies and ISPs, made it more difficult to identify unique human visitors to a website. Log analyzers responded by tracking visits by cookies, and by ignoring requests from known spiders.

The extensive use of web caches also presented a problem for logfile analysis. If a person revisits a page, the second request will often be retrieved from the browser's cache, and so no request will be received by the web server. This means that the person's path through the site is lost. Caching can be defeated by configuring the web server, but this can result in degraded performance for the visitor to the website.

Page tagging

Concerns about the accuracy of logfile analysis in the presence of caching, and the desire to be able to perform web analytics as an outsourced service, led to the second data collection method, page tagging or 'Web bugs'.

In the mid 1990s, Web counters were commonly seen — these were images included in a web page that showed the number of times the image had been requested, which was an estimate of the number of visits to that page. In the late 1990s this concept evolved to include a small invisible image instead of a visible one, and, by using JavaScript, to pass along with the image request certain information about the page and the visitor. This information can then be processed remotely by a web analytics company, and extensive statistics generated.

The web analytics service also manages the process of assigning a cookie to the user, which can uniquely identify them during their visit and in subsequent visits. Cookie acceptance rates vary significantly between web sites and may affect the quality of data collected and reported.

Collecting web site data using a third-party data collection server (or even an in-house data collection server) requires an additional DNS look-up by the user's computer to determine the IP address of the collection server. On occasion, delays in completing a successful or failed DNS look-ups may result in data not being collected.

With the increasing popularity of Ajax-based solutions, an alternative to the use of an invisible image, is to implement a call back to the server from the rendered page. In this case, when the page is rendered on the web browser, a piece of Ajax code would call back to the server and pass information about the client that can then be aggregated by a web analytics company. This is in some ways flawed by browser restrictions on the servers which can be contacted with XmlHttpRequest objects.

Logfile analysis vs page tagging

Both logfile analysis programs and page tagging solutions are readily available to companies that wish to perform web analytics. In some cases, the same web analytics company will offer both approaches. The question then arises of which method a company should choose. There are advantages and disadvantages to each approach.

Advantages of logfile analysis

The main advantages of logfile analysis over page tagging are as follows:

- The web server normally already produces logfiles, so the raw data is already available. To collect data via page tagging requires changes to the website.
- The data is on the company's own servers, and is in a standard, rather than a proprietary, format. This makes it easy for a company to switch programs later, use several different programs, and analyze historical data with a new program.
- Page tagging solutions can involve vendor lock-in, but this problem can be avoided by using some type of include to embed the tracking code on each page from one file. It can then easily be changed for all pages simply by changing it in this one file, thus making it possible to easily add and remove analytics platforms without changing each page individually.

- Logfiles contain information on visits from search engine spiders. Although these should not be reported as part of the human activity, it is useful information for search engine optimization.
- Logfiles require no additional DNS Lookups. Thus there are no external server calls which can slow page load speeds, or result in uncounted page views.
- The web server reliably records every transaction it makes. Page tagging may not be able to record all transactions. Reasons include:
  - Page tagging relies on the visitors' browsers co-operating, which a certain proportion may not do (for example, if JavaScript is disabled, or a hosts file prohibits requests to certain servers).
  - Tags may be omitted from pages either by oversight or between bouts of additional page tagging.
  - It may not be possible to include tags in all pages. Examples include static content such as PDFs or application-generated dynamic pages where re-engineering the application to include tags is not an option.

Advantages of page tagging

The main advantages of page tagging over logfile analysis are as follows.

- Counting is activated by opening the page, not requesting it from the server. If a page is cached, it will not be counted by the server. Cached pages can account for up to one-third of all pageviews. Not counting cached pages seriously skews many site metrics. It is for this reason server-based log analysis is not considered suitable for analysis of human activity on websites.
- Data is gathered via a component ("tag") in the page, usually written in JavaScript, though Java can be used, and increasingly Flash is used. JQuery and AJAX can also be used in conjunction with a server-side scripting language (such as PHP) to manipulate and (usually) store it in a database, basically enabling complete control over the how data is represented.
- It is easier to add additional information to the tag, which can then be collected by the remote server. For example, information about the visitors' screen sizes, or the price of the goods they purchased, can be added in this way. With logfile analysis, information not normally collected by the web server can only be recorded by modifying the URL.
- Page tagging can report on events which do not involve a request to the web server, such as interactions within Flash movies, partial form completion, mouse events such as onClick, onMouseOver, onFocus, onBlur etc.
- The page tagging service manages the process of assigning cookies to visitors; with logfile analysis, the server has to be configured to do this.
- Page tagging is available to companies who do not have access to their own web servers.
- Lately page tagging has become a standard in web analytics.

Economic factors

Logfile analysis is almost always performed in-house. Page tagging can be performed in-house, but it is more often provided as a third-party service. The economic difference between these two models can also be a consideration for a company deciding which to purchase.

- Logfile analysis typically involves a one-off software purchase; however, some vendors are introducing maximum annual page views with additional costs to process additional information. In addition to commercial offerings, several open-source logfile analysis tools are available free of charge.
- For Logfile analysis you have to store and archive your own data, which often grows very large quickly. Although the cost of hardware to do this is minimal, the overhead for an IT department can be considerable. For example, if you run out of disk space your database may start over-writing old entries which can often be irreparable.
- For Logfile analysis you need to maintain the software, including updates and security patches.
- Complex page tagging vendors charge a monthly fee based on volume i.e. number of pageviews per month collected.

Which solution is cheaper to implement depends on the amount of technical expertise within the company, the vendor chosen, the amount of activity seen on the web sites, the depth and type of information sought, and the number of distinct web sites needing statistics.

Regardless of the vendor solution or data collection method employed, the cost of web visitor analysis and interpretation should also be included. That is, the cost of turning raw data into actionable information. This can be from the use of third party consultants, the hiring of an experienced web analyst, or the training of a suitable in-house person. A cost-benefit analysis can then be performed. For example, what revenue increase or cost savings can be gained by analysing the web visitor data?

Hybrid methods

Some companies are now producing programs that collect data through both logfiles and page tagging. By using a hybrid method, they aim to produce more accurate statistics than either method on its own. The first Hybrid solution was produced in 1998 by Rufus Evison, who then spun the product out to create a company based upon the increased accuracy of hybrid methods.

Visitors Geolocation

With IP geolocation, it is possible to track visitors location. Using IP geolocation database or API, visitors can be geolocated to city, region or country level.

IP Intelligence, or Internet Protocol (IP) Intelligence, is a technology that maps the Internet and catalogues IP addresses by parameters such as geographic location (country, region, state, city and postcode), connection type, Internet Service Provider (ISP), proxy information, and more. The first generation of IP Intelligence was referred to as geotargeting or geolocation technology. This information is used by businesses for online audience segmentation in applications such online advertising, behavioral targeting, content localization (or website localization), digital rights management, personalization, online fraud detection, geographic rights management, localized search, enhanced analytics, global traffic management, and content distribution.

Click analytics

Click analytics is a special type of web analytics that gives special attention to clicks (Point-and-click).

Commonly, click analytics focuses on on-site analytics. An editor of a web site uses click analytics to determine the performance of his or her particular site, with regards to where the users of the site are clicking.

Also, click analytics may happen real-time or "unreal"-time, depending on the type of information sought. Typically, front-page editors on high-traffic news media sites will want to monitor their pages in real-time, to optimize the content. Editors, designers or other types of stakeholders may analyze clicks on a wider time frame to aid them assess performance of writers, design elements or advertisements etc.

Data about clicks may be gathered in at least two ways. Ideally, a click is "logged" when it occurs, and this method requires some functionality that picks up relevant information when the event occurs. Alternatively, one may institute the assumption that a page view is a result of a click, and therefore log a simulated click that lead to that page view.

Customer lifecycle analytics

Customer lifecycle analytics is a visitor-centric approach to measuring that falls under the umbrella of lifecycle marketing. Page views, clicks and other events (such as API calls, access to third-party services, etc.) are all tied to an individual visitor instead of being stored as separate data points. Customer lifecycle analytics attempts to connect all the data points into a marketing funnel that can offer insights into visitor behavior and website optimization.

Other methods

Other methods of data collection are sometimes used. Packet sniffing collects data by sniffing the network traffic passing between the web server and the outside world. Packet sniffing involves no changes to the web pages or web servers. Integrating web analytics into the web server software itself is also possible. Both these methods claim to provide better real-time data than other methods.

Key definitions

There are no globally agreed definitions within web analytics as the industry bodies have been trying to agree definitions that are useful and definitive for some time. The main bodies who have had input in this area have been JICWEBS (The Joint Industry Committee for Web Standards in the UK and Ireland), ABCe (Audit Bureau of Circulations electronic, UK and Europe), The WAA (Web Analytics Association, US) and to a lesser extent the IAB (Interactive Advertising Bureau). This does not prevent the following list from being a useful guide, suffering only slightly from ambiguity. Both the WAA and the ABCe provide more definitive lists for those who are declaring their statistics using the metrics defined by either.

- **Hit** - A request for a file from the web server. Available only in log analysis. The number of hits received by a website is frequently cited to assert its popularity, but this number is extremely misleading and dramatically over-estimates popularity. A single web-page typically consists of multiple (often dozens) of discrete files, each of which is counted as a hit as the page is downloaded, so the number of hits is really an arbitrary number more reflective of the complexity of individual pages on the website than the website's actual popularity. The total number of visitors or page views provides a more realistic and accurate assessment of popularity.
- **Page view** - A request for a file whose type is defined as a page in log analysis. An occurrence of the script being run in page tagging. In log analysis, a single page view may generate multiple hits as all the resources required to view the page (images, .js and .css files) are also requested from the web server.
- **Visit / Session** - A visit is defined as a series of page requests from the same uniquely identified client with a time of no more than 30 minutes between each page request. A session is defined as a series of page requests from the same uniquely identified client with a time of no more than 30 minutes and no requests for pages from other domains intervening between page requests. In other words, a session ends when someone goes to another site, or 30 minutes elapse between pageviews, whichever comes first. A visit ends only after a 30 minute time delay. If someone leaves a site, then returns within 30 minutes, this will count as one visit but two sessions. In practice, most systems ignore sessions and many analysts use both terms for visits. Because time between pageviews is critical to the definition of visits and sessions, a single page view does not constitute a visit or a session (it is a "bounce").
- **First Visit / First Session** - A visit from a visitor who has not made any previous visits.
- **Visitor / Unique Visitor / Unique User** - The uniquely identified client generating requests on the web server (log analysis) or viewing pages (page tagging) within a defined time period (i.e. day, week or month). A Unique Visitor counts once within the timescale. A visitor can make multiple visits. Identification is made to the visitor's computer, not the person, usually via cookie and/or IP+User Agent. Thus the same person visiting from two different computers will count as two Unique Visitors. Increasingly visitors are uniquely identified by

Flash LSO's (Local Shared Object), which are less susceptible to privacy enforcement.

- **Repeat Visitor** - A visitor that has made at least one previous visit. The period between the last and current visit is called visitor recency and is measured in days.
- **New Visitor** - A visitor that has not made any previous visits. This definition creates a certain amount of confusion, and is sometimes substituted with analysis of first visits.
- **Impression** - An impression is each time an advertisement loads on a user's screen. Anytime you see a banner, that is an impression.
- **Singletons** - The number of visits where only a single page is viewed. While not a useful metric in and of itself the number of singletons is indicative of various forms of Click fraud as well as being used to calculate bounce rate and in some cases to identify automatons bots.
- **Bounce Rate** - The percentage of visits where the visitor enters and exits at the same page without visiting any other pages on the site in between.
- **% Exit** - The percentage of users who exit from a page.
- **Visibility time** - The time a single page (or a blog, Ad Banner...) is viewed.
- **Session Duration** - Average amount of time that visitors spend on the site each time they visit. This metric can be complicated by the fact that analytics programs can not measure the length of the final page view.
- **Page View Duration / Time on Page** - Average amount of time that visitors spend on each page of the site. As with Session Duration, this metric is complicated by the fact that analytics programs can not measure the length of the final page view unless they record a page close event, such as onUnload().
- **Active Time / Engagement Time** - Average amount of time that visitors spend actually interacting with content on a web page, based on mouse moves, clicks, hovers and scrolls. Unlike Session Duration and Page View Duration / Time on Page, this metric **can** accurately measure the length of engagement in the final page view.
- **Page Depth / Page Views per Session** - Page Depth is the average number of page views a visitor consumes before ending their session. It is calculated by dividing total number of page views by total number of sessions and is also called Page Views per Session or PV/Session.
- **Frequency / Session per Unique** - Frequency measures how often visitors come to a website. It is calculated by dividing the total number of sessions (or visits) by the total number of unique visitors. Sometimes it is used to measure the loyalty of your audience.
- **Click path** - the sequence of hyperlinks one or more website visitors follows on a given site.
- **Click** - "refers to a single instance of a user following a hyperlink from one page in a site to another". Some use click analytics to analyze their web sites.
- **Site Overlay** is a technique in which graphical statistics are shown besides each link on the web page. These statistics represent the percentage of clicks on each link.

Common sources of confusion in web analytics

The hotel problem

The hotel problem is generally the first problem encountered by a user of web analytics. The term was first coined by Rufus Evison explaining the problem at one of the Emetrics Summits and has now gained popularity as a simple expression of the problem and its resolution.

The problem is that the unique visitors for each day in a month do not add up to the same total as the unique visitors for that month. This appears to an inexperienced user to be a problem in whatever analytics software they are using. In fact it is a simple property of the metric definitions.

The way to picture the situation is by imagining a hotel. The hotel has two rooms (Room A and Room B).

	Day 1	Day 2	Day 3	Total
Room A	John	John	Jane	2 Unique Users
Room B	Mark	Jane	Mark	2 Unique Users
Total	2	2	2	?

As the table shows, the hotel has two unique users each day over three days. The sum of the totals with respect to the days is therefore six.

During the period each room has had two unique users. The sum of the totals with respect to the rooms is therefore four.

Actually only three visitors have been in the hotel over this period. The problem is that a person who stays in a room for two nights will get counted twice if you count them once on each day, but is only counted once if you are looking at the total for the period. Any software for web analytics will sum these correctly for whatever time period, thus leading to the problem when a user tries to compare the totals.

New visitors + Repeat visitors unequal to total visitors

Another common misconception in web analytics is that the sum of the new visitors and the repeat visitors ought to be the total number of visitors. Again this becomes clear if the visitors are viewed as individuals on a small scale, but still causes a large number of complaints that analytics software cannot be working because of a failure to understand the metrics.

Here the culprit is the metric of a new visitor. There is really no such thing as a new visitor when you are considering a web site from an ongoing perspective. If a visitor makes their first visit on a given day and then returns to the web site on the same day they are both a new visitor and a repeat visitor for that day. So if we look at them as an individual which are they? The answer has to be both, so the definition of the metric is at fault.

A new visitor is not an individual; it is a fact of the web measurement. For this reason it is easiest to conceptualize the same facet as a first visit (or first session). This resolves the conflict and so removes the confusion. Nobody expects the number of first visits to add to the number of repeat visitors to give the total number of visitors. The metric will have the same number as the new visitors, but it is clearer that it will not add in this fashion.

On the day in question there was a first visit made by our chosen individual. There was also a repeat visit made by the same individual. The number of first visits and the number of repeat visits will add up to the total number of visits for that day.

Web analytics methods

Problems with cookies

Historically, vendors of page-tagging analytics solutions have used third-party cookies sent from the vendor's domain instead of the domain of the website being browsed. Third-party cookies can handle visitors who cross multiple unrelated domains within the company's site, since the cookie is always handled by the vendor's servers.

However, third-party cookies in principle allow tracking an individual user across the sites of different companies, allowing the analytics vendor to collate the user's activity on sites where he provided personal information with his activity on other sites where he thought he was anonymous. Although web analytics companies deny doing this, other companies such as companies supplying banner ads have done so. Privacy concerns about cookies have therefore led a noticeable minority of users to block or delete third-party cookies. In 2005, some reports showed that about 28% of Internet users blocked third-party cookies and 22% deleted them at least once a month.

Most vendors of page tagging solutions have now moved to provide at least the option of using first-party cookies (cookies assigned from the client subdomain).

Another problem is cookie deletion. When web analytics depend on cookies to identify unique visitors, the statistics are dependent on a persistent cookie to hold a unique visitor ID. When users delete cookies, they usually delete both first- and third-party cookies. If this is done between interactions with the site, the user will appear as a first-time visitor at their next interaction point. Without a persistent and unique visitor id, conversions, click-stream analysis, and other metrics dependent on the activities of a unique visitor over time, cannot be accurate.

Cookies are used because IP addresses are not always unique to users and may be shared by large groups or proxies. Other methods of uniquely identifying a user are technically challenging and would limit the trackable audience or would be considered suspicious. Cookies are the selected option because they reach the lowest common denominator without using technologies regarded as spyware.

Unique landing pages vs referrals for campaign tracking

Tracking the amount of activity generated through advertising relationships with external web sites through the referrals reports available in most web analytics packages is significantly less accurate than using unique landing pages.

Secure analytics (metering) methods

All the methods described above (and some other methods not mentioned here, like sampling) have the central problem of being vulnerable to manipulation (both inflation and deflation). This means these methods are imprecise and insecure (in any reasonable model of security). This issue has been addressed in a number of papers , but to-date the solutions suggested in these papers remain theoretic, possibly due to lack of interest from the engineering community, or because of financial gain the current situation provides to the owners of big websites.

Chapter 5

Web Colors

Web colors are colors used in designing web pages, and the methods for describing and specifying those colors. Hexadecimal color codes begin with a hash (#).

Authors of web pages have a variety of options available for specifying colors for elements of web documents. Colors may be specified as an RGB triplet in hexadecimal format (a *hex triplet*); they may also be specified according to their common English names in some cases. Often a color tool or other graphics software is used to generate color values.

The first versions of Mosaic and Netscape Navigator used the X11 color names as the basis for their color lists, as both started as X Window System applications.

Web colors have an unambiguous colorimetric definition, sRGB, which relates the chromaticities of a particular phosphor set, a given transfer curve, adaptive whitepoint, and viewing conditions. These have been chosen to be similar to many real-world monitors and viewing conditions, so that even without color management rendering is fairly close to the specified values. However, user agents vary in the fidelity with which they represent the specified colors. More advanced user agents use color management to provide better color fidelity; this is particularly important for Web-to-print applications.

Hex triplet

A **hex triplet** is a six-digit, three-byte hexadecimal number used in HTML, CSS, SVG, and other computing applications, to represent colors. The bytes represent the red, green and blue components of the color. One byte represents a number in the range 00 to FF (in hexadecimal notation), or 0 to 255 in decimal notation. This represents the least (0) to the most (255) intensity of each of the color components. The hex triplet is formed by concatenating three bytes in hexadecimal notation, in the following order:

Byte 1: red value (color type red)
Byte 2: green value (color type green)
Byte 3: blue value (color type blue)

For example, consider the color where the red/green/blue values are decimal numbers: red=36, green=104, blue=160 (a greyish-blue color). The decimal numbers 36, 104 and

160 are equivalent to the hexadecimal numbers 24, 68 and A0 respectively. The hex triplet is obtained by concatenating the 6 hexadecimal digits together, 2468A0 in this example.

Note that if any one of the three color values is less than 16 (decimal) or 10 (hex), it must be represented with a leading zero so that the triplet always has exactly six digits. For example, the decimal triplet 4, 8, 16 would be represented by the hex digits 04, 08, 10, forming the hex triplet 040810.

The number of colors that can be represented by this system is

$$256 \times 256 \times 256 = 16,777,216$$

An abbreviated, three (hexadecimal) digit form is sometimes used. Expanding this form to the six-digit form is as simple as doubling each digit: 09C becomes 0099CC. This allows each color value to cover its full range from 00 to FF. The three-digit form is described in the CSS specification, not in HTML. As a result, the three digit form in an attribute other than "style" is not interpreted as a valid color in some browsers.

Converting RGB to hexadecimal

RGB values are usually given in the 0–255 range; if they are in the 0–1 range, the values are multiplied by 255 before conversion. This number divided by 16 (integer division; ignoring any remainder) gives us the first hexadecimal digit (between 0 and F, where the letters A to F represent the numbers 10 to 15). The remainder gives us the second hexadecimal digit. For instance the RGB value 201 divides into 12 groups of 16, thus the first digit is C. A remainder of 9 gives us the hexadecimal number C9. This process is repeated for each of the three color values.

Conversion between number bases is a common feature of calculators, including both hand-held models and the software calculators bundled with most modern operating systems. Web-based tools specifically for converting color values are also available.

HTML color names

The HTML 4.01 specification defines sixteen named colors, as follows (names are defined in this context to be case-insensitive):

CSS1 / HTML3–4 / VGA color names

Name	Hex triplet	Red	Green	Blue	Hue	Satur	Light	Satur	Value	alias; CGA Number — Name
White	#FFFFFF	100%	100%	100%	0°	0%	100%	0%	100%	15 — white

Silver	#C0C0C0	75%	75%	75%	0°	0%	75%	0%	75%	7 — light gray
Gray	#808080	50%	50%	50%	0°	0%	50%	0%	50%	8 — dark gray
Black	#000000	0%	0%	0%	0°	0%	0%	0%	0%	0 — black
Red	#FF0000	100%	0%	0%	0°	100%	50%	100%	100%	12 — high red
Maroon	#800000	50%	0%	0%	0°	100%	25%	100%	50%	4 — low red
Yellow	#FFFF00	100%	100%	0%	60°	100%	50%	100%	100%	14 — yellow
Olive	#808000	50%	50%	0%	60°	100%	25%	100%	50%	6 — brown
Lime	#00FF00	0%	100%	0%	120°	100%	50%	100%	100%	green; 10 — high green
Green	#008000	0%	50%	0%	120°	100%	25%	100%	50%	2 — low green
Aqua	#00FFFF	0%	100%	100%	180°	100%	50%	100%	100%	cyan; 11 — high cyan
Teal	#008080	0%	50%	50%	180°	100%	25%	100%	50%	3 — low cyan
Blue	#0000FF	0%	0%	100%	240°	100%	50%	100%	100%	9 — high blue
Navy	#000080	0%	0%	50%	240°	100%	25%	100%	50%	1 — low blue
Fuchsia	#FF00FF	100%	0%	100%	300°	100%	50%	100%	100%	magenta; 13 — high magenta
Purple	#800080	50%	0%	50%	300°	100%	25%	100%	50%	5 — low magenta

These 16 were labelled as sRGB and included in the HTML 3.0 specification, which noted they were "the standard 16 colors supported with the Windows VGA palette."

X11 color names

In addition, a number of colors are defined by web browsers. A particular browser may not recognize all of these colors, but as of 2005 all modern general-use browsers support the full list. Many of these colors are from the list of X11 color names distributed with the X Window System. These colors were standardized by SVG 1.0, and are accepted by SVG Full user agents. They are not part of SVG Tiny.

The list of colors actually shipped with the X11 product varies between implementations, and clashes with certain of the HTML names such as green. Furthermore, X11 colors are defined as simple RGB (hence, no particular color space), rather than sRGB. This means that the list of colors found in X11 (e.g. in /usr/lib/X11/rgb.txt) should not directly be used to choose colors for the web.

The list of web "X11 colors" from the CSS3 specification, along with their hexadecimal and decimal equivalents, is shown below, compare the alphabetical lists in the W3C standards. Note that this includes the common synonyms: aqua (HTML4/CSS 1.0 standard name) and cyan (common sRGB name), magenta (common sRGB name) and fuchsia (HTML4/CSS 1.0 standard name), gray (HTML4/CSS 1.0 standard name) and grey.

HTML name	**Hex code**			**Decimal code**		
	R	**G**	**B**	**R**	**G**	**B**
Red colors						
IndianRed	CD	5C	5C	205	92	92
LightCoral	F0	80	80	240	128	128
Salmon	FA	80	72	250	128	114
DarkSalmon	E9	96	7A	233	150	122
LightSalmon	FF	A0	7A	255	160	122
Crimson	DC	14	3C	220	20	60
Red	FF	00	00	255	0	0
FireBrick	B2	22	22	178	34	34
DarkRed	8B	00	00	139	0	0
Pink colors						
Pink	FF	C0	CB	255	192	203
LightPink	FF	B6	C1	255	182	193
HotPink	FF	69	B4	255	105	180
DeepPink	FF	14	93	255	20	147
MediumVioletRed	C7	15	85	199	21	133
PaleVioletRed	DB	70	93	219	112	147
Orange colors						
LightSalmon	FF	A0	7A	255	160	122
Coral	FF	7F	50	255	127	80
Tomato	FF	63	47	255	99	71
OrangeRed	FF	45	00	255	69	0
DarkOrange	FF	8C	00	255	140	0
Orange	FF	A5	00	255	165	0
Yellow colors						
Gold	FF	D7	00	255	215	0
Yellow	FF	FF	00	255	255	0
LightYellow	FF	FF	E0	255	255	224
LemonChiffon	FF	FA	CD	255	250	205
LightGoldenrodYellow	FA	FA	D2	250	250	210
PapayaWhip	FF	EF	D5	255	239	213
Moccasin	FF	E4	B5	255	228	181
PeachPuff	FF	DA	B9	255	218	185

PaleGoldenrod	EE	E8	AA	238	232	170
Khaki	F0	E6	8C	240	230	140
DarkKhaki	BD	B7	6B	189	183	107
Purple colors						
Lavender	E6	E6	FA	230	230	250
Thistle	D8	BF	D8	216	191	216
Plum	DD	A0	DD	221	160	221
Violet	EE	82	EE	238	130	238
Orchid	DA	70	D6	218	112	214
Fuchsia	FF	00	FF	255	0	255
Magenta	FF	00	FF	255	0	255
MediumOrchid	BA	55	D3	186	85	211
MediumPurple	93	70	DB	147	112	219
Amethyst	99	66	CC	153	102	204
BlueViolet	8A	2B	E2	138	43	226
DarkViolet	94	00	D3	148	0	211
DarkOrchid	99	32	CC	153	50	204
DarkMagenta	8B	00	8B	139	0	139
Purple	80	00	80	128	0	128
Indigo	4B	00	82	75	0	130
SlateBlue	6A	5A	CD	106	90	205
DarkSlateBlue	48	3D	8B	72	61	139
MediumSlateBlue	7B	68	EE	123	104	238

HTML name	Hex code			Decimal code		
	R	G	B	R	G	B
Green colors						
GreenYellow	AD	FF	2F	173	255	47
Chartreuse	7F	FF	00	127	255	0
LawnGreen	7C	FC	00	124	252	0
Lime	00	FF	00	0	255	0
LimeGreen	32	CD	32	50	205	50
PaleGreen	98	FB	98	152	251	152
LightGreen	90	EE	90	144	238	144
MediumSpringGreen	00	FA	9A	0	250	154
SpringGreen	00	FF	7F	0	255	127
MediumSeaGreen	3C	B3	71	60	179	113
SeaGreen	2E	8B	57	46	139	87
ForestGreen	22	8B	22	34	139	34
Green	00	80	00	0	128	0
DarkGreen	00	64	00	0	100	0
YellowGreen	9A	CD	32	154	205	50
OliveDrab	6B	8E	23	107	142	35
Olive	80	80	00	128	128	0
DarkOliveGreen	55	6B	2F	85	107	47
MediumAquamarine	66	CD	AA	102	205	170
DarkSeaGreen	8F	BC	8F	143	188	143

LightSeaGreen	20	B2	AA	32	178	170
DarkCyan	00	8B	8B	0	139	139
Teal	00	80	80	0	128	128
Blue/Cyan colors						
Aqua	00	FF	FF	0	255	255
Cyan	00	FF	FF	0	255	255
LightCyan	E0	FF	FF	224	255	255
PaleTurquoise	AF	EE	EE	175	238	238
Aquamarine	7F	FF	D4	127	255	212
Turquoise	40	E0	D0	64	224	208
MediumTurquoise	48	D1	CC	72	209	204
DarkTurquoise	00	CE	D1	0	206	209
CadetBlue	5F	9E	A0	95	158	160
SteelBlue	46	82	B4	70	130	180
LightSteelBlue	B0	C4	DE	176	196	222
PowderBlue	B0	E0	E6	176	224	230
LightBlue	AD	D8	E6	173	216	230
SkyBlue	87	CE	EB	135	206	235
LightSkyBlue	87	CE	FA	135	206	250
DeepSkyBlue	00	BF	FF	0	191	255
DodgerBlue	1E	90	FF	30	144	255
CornflowerBlue	64	95	ED	100	149	237
MediumSlateBlue	7B	68	EE	123	104	238
RoyalBlue	41	69	E1	65	105	225
Blue	00	00	FF	0	0	255
MediumBlue	00	00	CD	0	0	205
DarkBlue	00	00	8B	0	0	139
Navy	00	00	80	0	0	128
MidnightBlue	19	19	70	25	25	112

HTML name	Hex code			Decimal code		
	R	G	B	R	G	B
Brown colors						
Cornsilk	FF	F8	DC	255	248	220
BlanchedAlmond	FF	EB	CD	255	235	205
Bisque	FF	E4	C4	255	228	196
NavajoWhite	FF	DE	AD	255	222	173

Wheat	F5	DE	B3	245	222	179
BurlyWood	DE	B8	87	222	184	135
Tan	D2	B4	8C	210	180	140
RosyBrown	BC	8F	8F	188	143	143
SandyBrown	F4	A4	60	244	164	96
Goldenrod	DA	A5	20	218	165	32
DarkGoldenrod	B8	86	0B	184	134	11
Peru	CD	85	3F	205	133	63
Chocolate	D2	69	1E	210	105	30
SaddleBrown	8B	45	13	139	69	19
Sienna	A0	52	2D	160	82	45
Brown	A5	2A	2A	165	42	42
Maroon	80	00	00	128	0	0
White colors						
White	FF	FF	FF	255	255	255
Snow	FF	FA	FA	255	250	250
Honeydew	F0	FF	F0	240	255	240
MintCream	F5	FF	FA	245	255	250
Azure	F0	FF	FF	240	255	255
AliceBlue	F0	F8	FF	240	248	255
GhostWhite	F8	F8	FF	248	248	255
WhiteSmoke	F5	F5	F5	245	245	245
Seashell	FF	F5	EE	255	245	238
Beige	F5	F5	DC	245	245	220
OldLace	FD	F5	E6	253	245	230
FloralWhite	FF	FA	F0	255	250	240
Ivory	FF	FF	F0	255	255	240
AntiqueWhite	FA	EB	D7	250	235	215
Linen	FA	F0	E6	250	240	230
LavenderBlush	FF	F0	F5	255	240	245
MistyRose	FF	E4	E1	255	228	225

Gray colors						
Gainsboro	DC	DC	DC	220	220	220
LightGrey	D3	D3	D3	211	211	211
Silver	C0	C0	C0	192	192	192
DarkGray	A9	A9	A9	169	169	169
Gray	80	80	80	128	128	128
DimGray	69	69	69	105	105	105
LightSlateGray	77	88	99	119	136	153
SlateGray	70	80	90	112	128	144
DarkSlateGray	2F	4F	4F	47	79	79
Black	00	00	00	0	0	0

Web-safe colors

At one time many computer displays were only capable of displaying 256 colors. These may be dictated by the hardware or changeable by a "color table". When a color is found (e.g., in an image) that is not one available, a different one has to be used. This can be either using the closest color (fast) or dithering (slow, looks better).

There were various attempts to make a "standard" color palette. A set of colors was needed that could be shown without dithering on 256-color displays; the number 216 was chosen partly because computer operating systems customarily reserved sixteen to twenty colors for their own use; it was also selected because it allows exactly six shades each of red, green, and blue ($6 \times 6 \times 6 = 216$).

The list of colors is often presented as if it has special properties that render them immune to dithering. In fact, on 256-color displays applications can set a palette of any selection of colors that they choose, dithering the rest. These colors were chosen specifically because they matched the palettes selected by the then leading browser applications. Fortunately, there were not radically different palettes in use in different popular browsers.

"Web-safe" colors had a flaw in that, on systems such as X11 where the palette is shared between applications, smaller color cubes (5×5×5 or 4×4×4) were often allocated by browsers—thus, the "web safe" colors would actually dither on such systems. Better results were obtained by providing an image with a larger range of colors and allowing the browser to quantize the color space if needed, rather than suffer the quality loss of a double quantization.

As of 2007, personal computers typically have at least 16-bit color and usually 24-bit (TrueColor). Even mobile devices have at least 16-bit color, driven by the inclusion of cameras on cellphones. The use of "web-safe" colors has fallen into practical disuse, but persisted in culture.

The "web-safe" colors do not all have standard names, but each can be specified by an RGB triplet: each component (red, green, and blue) takes one of the six values from the following table (out of the 256 possible values available for each component in full 24-bit color).

6 shades of each color

key	hex	decimal
0	00	0
3	33	51
6	66	102
9	99	153
C or (12)	CC	204
F or (15)	FF	255

The following table shows all of the "web-safe" colors, underlining the *really-safe* colors. (One shortcoming of the web-safe palette is its poor selection of light background colors.) The intensities at the low end of the range, especially the two darkest, are often hard to distinguish.

Color table

In the table below, each color code listed is a short-hand for the RGB value; for example, code 609 is equivalent to RGB code 102-0-153 or HEX code #660099.

Web-Safe Colors					
000	300	600	900	C00	*F00*
003	303	603	903	C03	*F03*
006	306	606	906	C06	F06
009	309	609	909	C09	F09
00C	30C	60C	90C	C0C	F0C
00F	30F	60F	90F	C0F	*F0F*
030	330	630	930	C30	F30
033	333	633	933	C33	F33
036	336	636	936	C36	F36

039	339	639	939	C39	F39
03C	33C	63C	93C	C3C	F3C
03F	33F	63F	93F	C3F	F3F
060	360	660	960	C60	F60
063	363	663	963	C63	F63
066	366	666	966	C66	F66
069	369	669	969	C69	F69
06C	36C	66C	96C	C6C	F6C
06F	36F	66F	96F	C6F	F6F
090	390	690	990	C90	F90
093	393	693	993	C93	F93
096	396	696	996	C96	F96
099	399	699	999	C99	F99
09C	39C	69C	99C	C9C	F9C
09F	39F	69F	99F	C9F	F9F
0C0	3C0	6C0	9C0	CC0	FC0
0C3	3C3	6C3	9C3	CC3	FC3
0C6	3C6	6C6	9C6	CC6	FC6
0C9	3C9	6C9	9C9	CC9	FC9
0CC	3CC	6CC	9CC	CCC	FCC
0CF	3CF	6CF	9CF	CCF	FCF
0F0	3F0	*6F0*	9F0	CF0	*FF0*
0F3	*3F3*	*6F3*	9F3	CF3	*FF3*
0F6	*3F6*	6F6	9F6	*CF6*	*FF6*
0F9	3F9	6F9	9F9	CF9	FF9
0FC	*3FC*	6FC	9FC	CFC	FFC
0FF	*3FF*	*6FF*	9FF	CFF	*FFF*

Safest web colors

Designers were often encouraged to stick to these 216 "web-safe" colors in their websites; however, 8-bit color displays were much more common when the 216-color palette was developed than they are now. David Lehn and Hadley Stern have since discovered that only 22 of the 216 colors in the web-safe palette are reliably displayed without inconsistent remapping on 16-bit computer displays. They called these 22 colors the "really safe" palette; it consists mainly of shades of green and yellow, as can be seen in the table above, where the "really safe" colors are underlined.

CSS colors

The Cascading Style Sheets language defines the same number of named colors as the HTML 4 spec, namely the 16 listed previously. Additionally, CSS 2.1 adds the 'orange' color name to the list:

Colors added in CSS 2.1

Name	Hex triplet	Red	Green	Blue	Hue	Satur	Light	Satur	Value	Alias
orange	#FFA500	100%	65%	0%	39°	100%	50%	—%	—%	

CSS 2, SVG and CSS 2.1 also allow web authors to use so-called *system colors*, which are color names whose values are taken from the operating system. This enables web authors to style their content in line with the operating system of the user agent. As of early 2004, it appears that the CSS3 color module will once again drop these values, marking them deprecated, but this may change.

The developing CSS3 specification will also introduce HSL color space values to style sheets:

```
/* RGB model */
p { color: #F00 } /* #rgb */
p { color: #FF0000 } /* #rrggbb */
p { color: rgb(255, 0, 0) } /* integer range 0 - 255 */
p { color: rgb(100%, 0%, 0%) } /* float range 0.0% - 100.0% */
/* RGB with alpha channel, added to CSS3 */
p { color: rgba(255, 0, 0, 0.5) } /* 0.5 opacity, semi-transparent */
/* HSL model, added to CSS3 */
p { color: hsl(0, 100%, 50%) } /* red */
p { color: hsl(120, 100%, 50%) } /* green */
p { color: hsl(120, 100%, 25%) } /* dark green */
p { color: hsl(120, 100%, 75%) } /* light green */
p { color: hsl(120, 50%, 50%) } /* pastel green */
/* HSL model with alpha channel */
p { color: hsla(120, 100%, 50%, 1) } /* green */
p { color: hsla(120, 100%, 50%, 0.5) } /* semi-transparent green */
p { color: hsla(120, 100%, 50%, 0.1) } /* very transparent green */
```

Accessibility

Some browsers and devices do not support colors. For these and blind and colorblind users, Web content depending on colors can be unusable or difficult to use. Both foreground and background color should be modified to avoid **black on black** effects. Similarly, most browsers show links as shades of blue by default; therefore, dark background colors, such as blue or navy, do not display well for such links.

Chapter 6

Comet (Programming)

Comet is a web application model in which a long-held HTTP request allows a web server to push data to a browser, without the browser explicitly requesting it. *Comet* is an umbrella term, encompassing multiple techniques for achieving this interaction. All these methods rely on features included by default in browsers, such as JavaScript, rather than on non-default plugins. The Comet approach differs from the original model of the web, in which a browser requests a complete web page at a time.

The use of Comet techniques in web development predates the use of the word *Comet* as a neologism for the collective techniques. Comet is known by several other names, including *Ajax Push*, *Reverse Ajax*, *Two-way-web*, *HTTP Streaming*, and *HTTP server push* among others.

Implementations

Comet applications attempt to eliminate the limitations of the page-by-page web model and traditional polling by offering real-time interaction, using a persistent or long-lasting HTTP connection between the server and the client. Since browsers and proxies are not designed with server events in mind, several techniques to achieve this have been developed, each with different benefits and drawbacks. The biggest hurdle is the HTTP 1.1 specification, which states that a browser should not have more than two simultaneous connections with a web server. However, holding one connection open for real-time events has a negative impact on browser usability: the browser may be blocked from sending a new request while waiting for the results of a previous request, e.g., a series of images. This can be worked around by creating a distinct hostname for real-time information, which is an alias for the same physical server.

Specific methods of implementing Comet fall into two major categories: streaming and long polling.

Streaming

An application using streaming Comet opens a single persistent connection from the client browser to the server for all Comet events. These events are incrementally handled and interpreted on the client side every time the server sends a new event, with neither side closing the connection.

Specific techniques for accomplishing streaming Comet include the following:

Hidden iframe

A basic technique for dynamic web application is to use a hidden iframe HTML element (an *inline frame*, which allows a website to embed one HTML document inside another). This invisible iframe is sent as a chunked block, which implicitly declares it as infinitely long (sometimes called "forever frame"). As events occur, the iframe is gradually filled with `script` tags, containing JavaScript to be executed in the browser. Because browsers render HTML pages incrementally, each `script` tag is executed as it is received.

One benefit of the iframe method is that it works in every common browser. Two downsides of this technique are the lack of a reliable error handling method, and the impossibility of tracking the state of the request calling process.

XMLHttpRequest

The XMLHttpRequest (XHR) object, the main tool used by Ajax applications for browser–server communication, can also be pressed into service for server–browser Comet messaging, in a few different ways.

In 1995, Netscape Navigator added a feature called "server push", which allowed servers to send new versions of an image or HTML page to that browser, as part of a multipart HTTP response, using the content type `multipart/x-mixed-replace`. Since 2004, Gecko-based browsers such as Firefox accept multipart responses to XHR, which can therefore be used as a streaming Comet transport. On the server side, each message is encoded as a separate portion of the multipart response, and on the client, the callback function provided to the XHR `onreadystatechange` function will be called as each message arrives. This functionality is only included in Gecko-based browsers, though there is discussion of adding it to Webkit.

Instead of creating a multipart response, and depending on the browser to transparently parse each event, it is also possible to generate a custom data format for an XHR response, and parse out each event using browser-side JavaScript, relying only on the browser firing the `onreadystatechange` callback each time it receives new data.

Ajax with long polling

None of the above streaming transports works across all modern browsers without negative side-effects in any. This forces Comet developers to implement several complex streaming transports, switching between them depending on the browser. Consequently many Comet applications use long polling, which is easier to implement on the browser side, and works, at minimum, in every browser that supports XHR. As the name suggests, long polling requires the client to poll the server for an event (or set of events). The browser makes an Ajax-style request to the server, which is kept open until the server has new data to send to the browser, which is sent to the browser in a complete response. The browser initiates a new long polling request in order to obtain subsequent events.

Specific technologies for accomplishing long-polling include the following:

XMLHttpRequest long polling

For the most part, XMLHttpRequest long polling works like any standard use of XHR. The browser makes an asynchronous request of the server, which may wait for data to be available before responding. The response can contain encoded data (typically XML or JSON) or Javascript to be executed by the client. At the end of the processing of the response, the browser creates and sends another XHR, to await the next event. Thus the browser always keeps a request outstanding with the server, to be answered as each event occurs.

Script tag long polling

While any Comet transport can be made to work across subdomains, none of the above transports can be used across different second-level domains (SLDs), due to browser security policies designed to prevent cross-site scripting attacks. That is, if the main web page is served from one SLD, and the Comet server is located at another SLD, Comet events cannot be used to modify the HTML and DOM of the main page, using those transports. This problem can be side-stepped by creating a proxy server in front of one or both sources, making them appear to originate from the same domain. However, this is often undesirable for complexity or performance reasons.

Unlike iframes or XMLHttpRequest objects, `script` tags can be pointed at any URI, and JavaScript code in the response will be executed in the current HTML document. This creates a potential security risk for both servers involved, though the risk to the data provider (in our case, the Comet server) can be avoided using JSONP.

A long-polling Comet transport can be created by dynamically creating `script` elements, and setting their source to the location of the Comet server, which then sends back JavaScript (or JSONP) with some event as its payload. Each time the script request is completed, the browser opens a new one, just as in the XHR long polling case. This method has the advantage of being cross-browser while still allowing cross-domain implementations.

History

Early Java applets

The ability to embed Java applets into browsers (starting with Netscape 2.0 in March 1996) made real-time communications possible, using a raw TCP socket to communicate between the browser and the server. This socket can remain open as long as the browser is at the document hosting the applet. Event notifications can be sent in any format — text or binary — and decoded by the applet.

First Comet applications

In March 2006, software engineer Alex Russell coined the term Comet in a post on his personal blog. The new term was a play on Ajax (Ajax and Comet both being household cleaners, common in the USA).

In 2006, while not really using the term Comet, some applications exposed those techniques to a wider audience: Meebo's multi-protocol web-based chat application enables users to connect to AOL, Yahoo, and Microsoft chat platforms through the browser; Google added web-based chat to Gmail; JotSpot, a startup since acquired by Google, built Comet-based real-time collaborative document editing. New Comet companies and enterprise solutions were created, such as the Java-based ICEfaces JSF framework (although they prefer the term "*Ajax Push*"). Others that had previously used Java-applet based transports switched instead to pure-JavaScript implementations.

Alternatives

Browser-native technologies are inherent in the term Comet. Attempts to improve non-polling HTTP communication have come from multiple sides:

- The HTML 5 draft specification produced by the Web Hypertext Application Technology Working Group (WHATWG) specifies so called server-sent events, it offers a new HTML element, `event-source` and a new data format called DOM event stream. Experimental implementation of this feature was introduced in Opera 9.
- The HTML 5 WebSocket API working draft specifies a method for creating a persistent connection with a server and receiving messages via an `onmessage` callback.
- The Bayeux protocol by the Dojo Foundation. It leaves browser-specific transports in place, and defines a higher-level protocol for communication between browser and server, with the aim of allowing re-use of client-side JavaScript code with multiple Comet servers, and allowing the same Comet server to communicate with multiple client-side JavaScript implementations. Bayeux is based on a publish/subscribe model, so servers supporting Bayeux have publish/subscribe built-in.
- The BOSH protocol by the XMPP standards foundation. It emulates a bidirectional stream between browser and server by using two synchronous HTTP connections.
- The JSONRequest object, proposed by Douglas Crockford, would be an alternative to the XHR object.
- Use of plugins, such as Java applets or the proprietary Adobe Flash (Java BlazeDS is a server plugin which streams events to Flash applications). These

have the advantage of working identically across all browsers with the appropriate plugin installed and need not rely on HTTP connections, but the disadvantage of requiring the plugin to be installed

- Google announced a new Channel API for Google App Engine, implementing a Comet-like API with the help of a client javascript library on the browser. But it could be later replaced by HTML5 websockets.

Chapter 7

Mashup (Web Application Hybrid)

In Web development, a **mashup** is a Web page or application that uses and combines data, presentation or functionality from two or more sources to create new services. The term implies easy, fast integration, frequently using open APIs and data sources to produce enriched results that were not necessarily the original reason for producing the raw source data.

The main characteristics of the mashup are combination, visualization, and aggregation. It is important to make existing data more useful, moreover for personal and professional use.

To be able to permanently access the data of other services, mashups are generally client applications or hosted online. Since 2010, two major mashup vendors have added support for hosted deployment based on Cloud computing solutions; that are Internet-based computing, whereby shared resources, software, and information are provided to computers and other devices on demand, like the electricity grid.

In the past years, more and more Web applications have published APIs that enable software developers to easily integrate data and functions instead of building them by themselves. Mashups can be considered to have an active role in the evolution of social software and Web 2.0. Mashup composition tools are usually simple enough to be used by end-users. They generally do not require programming skills and rather support visual wiring of GUI widgets, services and components together. Therefore, these tools contribute to a new vision of the Web, where users are able to contribute.

The term *mashup* is also used to describe a *remix* of digital data.

History

The history of mashup can be backtracked by first understanding the broader context of the history of the Web. For Web 1.0 business model companies stored consumer data on portals and updated them regularly. They controlled all the consumer data, and the consumer had to use their products and services to get the information.

With the advent of Web 2.0 a new proposition was created, using Web standards that were commonly and widely adopted across traditional competitors and unlocked the consumer data. At the same time, mashups emerged allowing mixing and matching competitor's API to create new services. The term isn't formally defined by any standard-setting body.

The first mashups used mapping services or photo services to combine these services with data of any kind and therefore create visualizations of the data. In the beginning, most mashups were consumer-based, but recently the mashup is to be seen as an interesting concept useful also to enterprises. Business mashups can combine existing internal data with external services to create new views on the data.

Mashups are in the ascending. As a statistic from Programmable Web found out in 2009 that three new mashups have been registered every single day for the last two years.

Types of mashups

There are many types of mashups, such as business mashups, consumer mashups, and data mashups. The most common type of mashup is the consumer mashup, aimed at the general public.

- *Business* (or *enterprise*) *mashups* generally define applications that combine their own resources, application and data, with other external Web services. They focus data into a single presentation and allow for collaborative action among businesses and developers. This works well for an agile development project, which requires collaboration between the developers and customer (or customer proxy, typically a product manager) for defining and implementing the business requirements. Enterprise mashups are secure, visually rich Web applications that expose actionable information from diverse internal and external information sources.

- *Consumer mashups* combines different data types. It combines data from multiple public sources in the browser and organizes it through a simple browser user interface.

- *Data mashups*, opposite to the consumer mashups, combine similar types of media and information from multiple sources into a single representation. The combination of all these resources create a new and distinct Web service that was not originally provided by either source.

By API Type

Mashups can also be categorized by the basic API type they use but any of these can be combined with each other or embedded into other applications.

Data types

- Indexed data (documents, weblogs, images, videos, shopping articles, jobs ...) used by Metasearch engines
- Cartographic and geographic data: Geolocation software, Geovisualization
- Feeds, podcasts: News aggregators

Functions

- Data converters: Language translators, Speech processing, URL shorteners...
- Communication: E-mail, Instant messaging, notification...
- Visual data rendering: Information visualization, diagrams
- Security related: electronic payment systems, ID identification...
- Editors

Mashups versus portals

Mashups and portals are both content aggregation technologies. Portals are an older technology designed as an extension to traditional dynamic Web applications, in which the process of converting data content into marked-up Web pages is split into two phases: generation of markup "fragments" and aggregation of the fragments into pages. Each markup fragment is generated by a "portlet", and the portal combines them into a single Web page. Portlets may be hosted locally on the portal server or remotely on a separate server.

Portal technology defines a complete event model covering reads and updates. A request for an aggregate page on a portal is translated into individual read operations on all the portlets that form the page ("`render`" operations on local, JSR 168 portlets or "`getMarkup`" operations on remote, WSRP portlets). If a submit button is pressed on any portlet on a portal page, it is translated into an update operation on that portlet alone (`processAction` on a local portlet or `performBlockingInteraction` on a remote, WSRP portlet). The update is then immediately followed by a read on *all* portlets on the page.

Portal technology is about server-side, presentation-tier aggregation. It cannot be used to drive more robust forms of application integration such as two-phase commit.

Mashups differ from portals in the following respects:

	Portal	**Mashup**
Classification	Older technology, extension to traditional Web server model using well-defined approach	Using newer, loosely defined "Web 2.0" techniques

Philosophy/Approach	Approaches aggregation by splitting role of Web server into two phases: markup generation and aggregation of markup fragments	Uses APIs provided by different content sites to aggregate and reuse the content in another way
Content dependencies	Aggregates presentation-oriented markup fragments (HTML, WML, VoiceXML, etc.)	Can operate on pure XML content and also on presentation-oriented content (e.g., HTML)
Location dependencies	Traditionally, content aggregation takes place on the server	Content aggregation can take place either on the server or on the client
Aggregation style	"Salad bar" style: Aggregated content is presented 'side-by-side' without overlaps	"Melting Pot" style - Individual content may be combined in any manner, resulting in arbitrarily structured hybrid content
Event model	Read and update event models are defined through a specific portlet API	CRUD operations are based on REST architectural principles, but no formal API exists
Relevant standards	Portlet behavior is governed by standards JSR 168, JSR 286 and WSRP, although portal page layout and portal functionality are undefined and vendor-specific	Base standards are XML interchanged as REST or Web Services. RSS and Atom are commonly used. More specific mashup standards such as EMML are emerging.

The portal model has been around longer and has had greater investment and product research. Portal technology is therefore more standardized and mature. Over time, increasing maturity and standardization of mashup technology will likely make it more popular than portal technology because it is more closely associated with Web 2.0 and lately Service-oriented Architectures (SOA). New versions of portal products are expected to eventually add mashup support while still supporting legacy portlet applications. Mashup technologies, in contrast, are not expected to provide support for portal standards.

Business mashups

Mashup use is expanding in the business environment. Business mashups are useful for integrating business and data services, as business mashups technologies provide the ability to develop new integrated services quickly, to combine internal services with external or personalized information, and to make these services tangible to the business user through user-friendly Web browser interfaces.

Business mashups differ from consumer mashups in the level of integration with business computing environments, security and access control features, governance, and the sophistication of the programming tools (mashup editors) used. Another difference between business mashups and consumer mashups is a growing trend of using Business mashups in commercial software as a service (SaaS) offering.

Many of the providers of Business Mashups technologies have added SOA features.

Architectural aspects of mashups

The architecture of a mashup is divided into three layers:

- Presentation / User interaction: this is the user interface of mashups. The technologies used are HTML/XHTML, CSS, Javascript, Asynchronous Javascript and Xml (Ajax).

- Web Services: the products functionality can be accessed using the API services. The technologies used are XMLHTTPRequest, XML-RPC, JSON-RPC, SOAP, REST.

- Data: Handling the data like sending, storing and receiving. The technologies used are XML, JSON, KML.

Architecturally, there are two styles of mashups: Web-based and server-based. Whereas Web-based mashups typically use the user's Web browser to combine and reformat the data, server-based mashups analyze and reformat the data on a remote server and transmit the data to the user's browser in its final form.

Mashups appear to be a variation of a Facade pattern. That is: a software engineering design pattern that provides a simplified interface to a larger body of code (in this case the code to aggregate the different feeds with different APIs).

Mashups can be used with software provided as a service (SaaS).

After several years of standards development, mainstream businesses are starting to adopt Service-oriented Architectures (SOA) to integrate disparate data by making them available as discrete Web services. Web services provide open, standardized protocols to provide a unified means of accessing information from a diverse set of platforms (operating systems, programming languages, applications). These Web services can be reused to provide completely new services and applications within and across organizations, providing business flexibility.

Chapter 8

Microformat

A **microformat** (sometimes abbreviated **µF**) is a web-based approach to semantic markup which seeks to re-use existing HTML/XHTML tags to convey metadata and other attributes in web pages and other contexts that support (X)HTML, such as RSS. This approach allows software to process information intended for end-users (such as contact information, geographic coordinates, calendar events, and the like) automatically.

Although the content of web pages is technically already capable of "automated processing", and has been since the inception of the web, such processing is difficult because the traditional markup tags used to display information on the web do not describe what the information means. Microformats can bridge this gap by attaching semantics, and thereby obviate other, more complicated, methods of automated processing, such as natural language processing or screen scraping. The use, adoption and processing of microformats enables data items to be indexed, searched for, saved or cross-referenced, so that information can be reused or combined.

As of 2010 microformats allow the encoding and extraction of events, contact information, social relationships and so on. More are being developed.

Background

Microformats emerged as part of a grassroots movement to make recognizable data items (such as events, contact details or geographical locations) capable of automated processing by software, as well as directly readable by end-users. Link-based microformats emerged first. These include vote links that express opinions of the linked page, which search engines can tally into instant polls.

As the microformats community grew, CommerceNet, a nonprofit organization that promotes electronic commerce on the Internet, helped sponsor and promote the technology and support the microformats community in various ways. CommerceNet also helped co-found the Microformats.org community site.

Neither CommerceNet nor Microformats.org operates as a standards body. The microformats community functions through an open mailing list, and Internet relay chat (IRC) channel. Most of the existing microformats were created at the Microformats.org and the associated mailing list, by a process of gathering examples of web publishing behaviour, then codifying it. Some other microformats (such as rel=nofollow and unAPI) have been proposed, or developed, elsewhere.

Plain Old Semantic HTML (POSH)

The phrase "plain old semantic HTML" has been found online as early as 1998, but the coinage of the acronym **POSH** used in connection with microformats occurred in April 2007 on the microformats irc channel. Semantic HTML focuses on the use of tags and attributes for semantic rather than presentational purposes. Its proponents discourage the use of tables for layout and the use of the `<b>` or `<br />` tags since these tags are purely presentational.

Technical overview

XHTML and HTML standards allow for the embedding and encoding of semantics within the attributes of markup tags. Microformats take advantage of these standards by indicating the presence of metadata using the following attributes:

- `class`
- `rel`
- `rev` (in one case, otherwise deprecated in microformats)

For example, in the text "The birds roosted at 52.48, -1.89" is a pair of numbers which may be understood, from their context, to be a set of geographic coordinates. With wrapping in spans (or other HTML elements) with specific class names (in this case `geo`, `latitude` and `longitude`, all part of the geo microformat specification):

```
The birds roosted at
   <span class="geo">
     <span class="latitude">52.48</span>,
     <span class="longitude">-1.89</span>
   </span>
```

software agents can recognize exactly what each value represents and can then perform a variety of tasks such as indexing, locating it on a map and exporting it to a GPS device.

In-context examples

Specific microformats

Several microformats have been developed to enable semantic markup of particular types of information.

- hAtom – for marking up Atom feeds from within standard HTML
- hCalendar – for events
- hCard – for contact information; includes:
    - adr – for postal addresses
    - geo – for geographical coordinates (latitude, longitude)
- hMedia - for audio/video content
- hNews - for news content
- hProduct – for products
- hRecipe - for recipes and foodstuffs.
- hResume – for resumes or CVs
- hReview – for reviews
- rel-directory – for distributed directory creation and inclusion
- rel-enclosure – for multimedia attachments to web pages
- rel-license – specification of copyright license
- rel-nofollow, an attempt to discourage third-party content spam (e.g. spam in blogs)
- rel-tag – for decentralized tagging (Folksonomy)
- xFolk – for tagged links
- XHTML Friends Network (XFN) – for social relationships
- XOXO – for lists and outlines

Microformats under development

Among the many proposed microformats, the following are undergoing active development:

- hAudio – for audio files and references to released recordings
- citation – for citing references
- currency – for amounts of money
- figure – for associating captions with images
- geo extensions – for places on Mars, the Moon, and other such bodies; for altitude; and for collections of waypoints marking routes or boundaries
- measure – For physical quantities, structured data-values

Uses of microformats

Using microformats within HTML code provides additional formatting and semantic data that applications can use. For example, applications such as web crawlers can collect data about on-line resources, or desktop applications such as e-mail clients or scheduling software can compile details. The use of microformats can also facilitate "mash ups" such as exporting all of the geographical locations on a web page into (for example) Google Maps to visualize them spatially.

Several browser extensions, such as Operator for Firefox and Oomph for Internet Explorer, provide the ability to detect microformats within an HTML document. When hCard or hCalendar are involved, such browser extensions allow to export them into formats compatible with contact management and calendar utilities, such as Microsoft Outlook. When dealing with geographical coordinates, they allow to send the location to maps applications such as Google Maps. Yahoo! Query Language can be used to extract microformats from web pages. On 12 May 2009, Google announced that they would be parsing the hCard, hReview and hProduct microformats, and using them to populate search result pages. They have since extended this to use hCalendar for events and hRecipe for cookery recipes.

Microsoft expressed a desire to incorporate Microformats into upcoming projects; as have other software companies.

Alex Faaborg summarizes the arguments for putting the responsibility for microformat user interfaces in the web browser rather than making more complicated HTML:

- Only the web browser knows what applications are accessible to the user and what the user's preferences are
- It lowers the barrier to entry for web site developers if they only need to do the markup and not handle "appearance" or "action" issues
- Retains backwards compatibility with web browsers that don't support microformats
- The web browser presents a single point of entry from the web to the user's computer, which simplifies security issues

Evaluation of microformats

Various commentators have offered review and discussion on the design principles and practical aspects of microformats. Additionally, microformats have been compared to other approaches that seek to serve the same or similar purpose. From time to time, there is criticism of a single, or all, microformats. Documented efforts to advocate both the spread and use of microformats are known to exist as well. Opera Software CTO and CSS creator Håkon Wium Lie said in 2005 "We will also see a bunch of microformats being developed, and that’s how the semantic web will be built, I believe." However, as of August 2008, Toby Inkster, author of the "Swignition" (formerly "Cognition") microformat parsing service pointed out that no new microformat specifications had been published for over three years.

Design principles

Computer scientist and entrepreneur, Rohit Khare stated that *reduce, reuse, and recycle* is "shorthand for several design principles" that motivated the development and practices behind microformats.[:71-72] These aspects can be summarized as follows:

- Reduce: favor the simplest solutions and focus attention on specific problems;

- Reuse: work from experience and favor examples of current practice;
- Recycle: encourage modularity and the ability to embed, valid XHTML can be reused in blog posts, RSS feeds, and anywhere else you can access the web.

Accessibility

Because some microformats make use of title attribute of HTML's `abbr` element to conceal machine-readable data (particularly date-times and geographical coordinates) in the "abbr design pattern", the plain text content of the element is inaccessible to those screen readers that expand abbreviations. In June 2008, the BBC announced that it would be dropping use of microformats using the `abbr` design pattern because of accessibility concerns.

Comparison with alternative approaches

Microformats are not the only solution for providing "more intelligent data" on the web. Alternative approaches exist and are under development as well. For example, the use of XML markup and standards of the Semantic Web are cited as alternative approaches. Some contrast these with microformats in that they do not necessarily coincide with the design principles of "reduce, reuse, and recycle", at least not to the same extent.

One advocate of microformats, Tantek Çelik, characterized a problem with alternative approaches:

> “ Here's a new language we want you to learn, and now you need to output these additional files on your server. It's a hassle. (Microformats) lower the barrier to entry. ”

For some applications the use of other approaches may be valid. If one wishes to use microformat-style embedding but the type of data one wishes to embed does not map to an existing microformat, one can use RDFa to embed arbitrary vocabularies into HTML, for example: embedding domain-specific scientific data on the Web like zoological or chemical data where no microformat for such data exists. Furthermore, standards such as W3C's GRDDL allow microformats to be converted into data compatible with the Semantic Web.

Another advocate of microformats, Ryan King, put the compatibility of microformats with other approaches this way:

> “ Microformats provide an easy way for many people to contribute semantic data to the web. With GRDDL all of that data is made available for RDF Semantic Web tools. Microformats and GRDDL can work together to build a better web.

Chapter 9

Push Technology and Style Sheet (Web Development)

Push technology

Push technology, **server push**, describes a style of Internet-based communication where the request for a given transaction is initiated by the publisher or central server. It is contrasted with pull technology, where the request for the transmission of information is initiated by the receiver or client.

General use

Push services are often based on information preferences expressed in advance. This is called a publish/subscribe model. A client might "subscribe" to various information "channels". Whenever new content is available on one of those channels, the server would push that information out to the user.

Synchronous conferencing and instant messaging are typical examples of push services. Chat messages and sometimes files are pushed to the user as soon as they are received by the messaging service. Both decentralised peer-to-peer programs (such as WASTE) and centralised programs (such as IRC or XMPP) allow pushing files, which means the sender initiates the data transfer rather than the recipient.

Email is also a push system: the SMTP protocol on which it is based is a push protocol. However, the last step —from mail server to desktop computer— typically uses a pull protocol like POP3 or IMAP. Modern e-mail clients make this step seem instantaneous by repeatedly polling the mail server, frequently checking it for new mail. The IMAP protocol includes the IDLE command, which allows the server to tell the client when new messages arrive. The original BlackBerry was the first popular example of push technology for email in a wireless context.

Another popular type of Internet push technology was PointCast Network, which gained popularity in the 1990s. It delivered news and stock market data. Both Netscape and Microsoft integrated it into their software at the height of the browser wars, but it later faded away and was replaced in the 2000s with RSS (a pull technology).

Other uses are push enabled web applications including market data distribution (stock tickers), online chat/messaging systems (webchat), auctions, online betting and gaming, sport results, monitoring consoles and sensor network monitoring.

Technologies

HTTP server push

HTTP server push (also known as HTTP streaming) is a mechanism for sending data from a web server to a web browser. HTTP server push can be achieved through several mechanisms.

Generally the web server does not terminate a connection after response data has been served to a client. The web server leaves the connection open such that if an event is received, it can immediately be sent to one or multiple clients. Otherwise the data would have to be queued until the client's next request is received. Most web servers offer this functionality via CGI (e.g. Non-Parsed Headers scripts on Apache).

Another mechanism is related to a special MIME type called `multipart/x-mixed-replace`, which was introduced by Netscape in 1995. Web browsers would interpret this as a document changing whenever the server felt like pushing a new version to the client. It is still supported by Firefox, Opera and Safari today, but ignored by Internet Explorer. It can be applied to HTML documents, but also for streaming images in webcam applications.

The WHATWG Web Applications 1.0 proposal included a mechanism to push content to the client. On September 1, 2006, the Opera web browser implemented this new experimental technology in a feature called "Server-Sent Events." It is now being standardized as part of HTML5. Another related part of HTML5 is the WebSockets API, which allows a web server and client to communicate over a full-duplex TCP connection.

Pushlet

In this technique, the server takes advantage of persistent HTTP connections and leaves the response perpetually "open" (i.e. it never terminates the response), effectively fooling the browser into continuing in "loading" mode after the initial page load would normally be complete. The server then periodically sends snippets of javascript to update the content of the page, thereby achieving push capability. By using this technique the client doesn't need Java applets or other plug-ins to keep an open connection to the server. The clients will be automatically notified by new events, pushed by the server. One serious drawback to this method, however, is the lack of control the server has over the browser timing out. A page refresh is always necessary if a timeout occurs on the browser end.

Long polling

Long polling is a variation of the traditional polling technique and allows emulation of an information push from a server to a client. With long polling, the client requests information from the server in a similar way to a normal poll. However, if the server does not have any information available for the client, instead of sending an empty response, the server holds the request and waits for some information to be available. Once the information becomes available (or after a suitable timeout), a complete response is sent to the client. The client will normally then immediately re-request information from the server, so that the server will almost always have an available waiting request that it can use to deliver data in response to an event. In a web/AJAX context, long polling is also known as Comet programming.

Long polling is itself not a push technology, but can be used under circumstances where a real push is not possible.

Flash XMLSocket relays

This technique, used by Cbox and other chat applications, makes use of the XMLSocket object in a single-pixel Adobe Flash movie. Under the control of JavaScript, the client establishes a TCP connection to a unidirectional relay on the server. The relay server does not read anything from this socket; instead it immediately sends the client a unique identifier. Next, the client makes an HTTP request to the web server, including with it this identifier. The web application can then push messages addressed to the client to a local interface of the relay server, which relays them over the Flash socket. The advantage of this approach is that it appreciates the natural read-write asymmetry that is typical of many web applications, including chat, and as a consequence it offers high efficiency. Since it does not accept data on outgoing sockets, the relay server does not need to poll outgoing TCP connections *at all*, making it possible to hold open tens of thousands of concurrent connections. In this model, the limit to scale is the TCP stack of the underlying server operating system.

Other technologies

The term Comet has been used to describe push technologies applied to Ajax web applications. It is used as an umbrella term for a combination of web technologies such as HTTP server push and long polling (see above).

XMPP is often used for push applications as well, especially the PubSub extensions. Apple uses this technology for its Mobile Me push support.

BOSH is a long-lived HTTP technique used in XMPP, but that can be used on the web. The specification (XEP-0124: Bidirectional-streams Over Synchronous HTTP (BOSH)) reads: *This specification defines a transport protocol that emulates the semantics of a long-lived, bidirectional TCP connection between two entities (such as a client and a server) by efficiently using multiple synchronous HTTP request/response pairs without requiring the use of frequent polling or chunked responses.*

Style sheet (web development)

Web **style sheets** are a form of separation of presentation and content for web design in which the markup (i.e., HTML or XHTML) of a webpage contains the page's semantic content and structure, but does not define its visual layout (style). Instead, the style is defined in an external stylesheet file using a style sheet language such as CSS or XSL. This design approach is identified as a "separation" because it largely supersedes the antecedent methodology in which a page's markup defined both style and structure.

The philosophy underlying this methodology is a specific case of separation of concerns.

Benefits

Separation of style and content has many benefits, but has only become practical in recent years due to improvements in popular web browsers' CSS implementations.

Speed

Overall, users experience of a site utilising style sheets will generally be quicker than sites that don't use the technology. 'Overall' as the first page will probably load more slowly – because the style sheet AND the content will need to be transferred. Subsequent pages will load faster because no style information will need to be downloaded – the CSS file will already be in the browser's cache.

Maintainability

Holding all the presentation styles in one file significantly reduces maintenance time and reduces the chance of human errors, thereby improving presentation consistency. For example, the font color associated with a type of text element may be specified — and therefore easily modified — throughout an entire website simply by changing one short string of characters in a single file. The alternate approach, using styles embedded in each individual page, would require a cumbersome, time consuming, and error-prone edit of every file.

Accessibility

Sites that use CSS with either XHTML or HTML are easier to tweak so that they appear extremely similar in different browsers (Internet Explorer, Mozilla Firefox, Opera, Safari, etc.).

Sites using CSS "degrade gracefully" in browsers unable to display graphical content, such as Lynx, or those so very old that they cannot use CSS. Browsers ignore CSS that they do not understand, such as CSS 3 statements. This enables a wide variety of user agents to be able to access the content of a site even if they cannot render the stylesheet or are not designed with graphical capability in mind. For example, a browser using a refreshable braille display for output could disregard layout information entirely, and the user would still have access to all page content.

Customization

If a page's layout information is all stored externally, a user can decide to disable the layout information entirely, leaving the site's bare content still in a readable form. Site authors may also offer multiple stylesheets, which can be used to completely change the appearance of the site without altering any of its content.

Most modern web browsers also allow the user to define their own stylesheet, which can include rules that override the author's layout rules. This allows users, for example, to bold every hyperlink on every page they visit.

Consistency

Because the semantic file contains only the meanings an author intends to convey, the styling of the various elements of the document's content is very consistent. For example, headings, emphasized text, lists and mathematical expressions all receive consistently applied style properties from the external stylesheet. Authors need not concern themselves with the style properties at the time of composition. These presentational details can be deferred until the moment of presentation.

Portability

The deferment of presentational details until the time of presentation means that a document can be easily re-purposed for an entirely different presentation medium with merely the application of a new stylesheet already prepared for the new medium and consistent with elemental or structural vocabulary of the semantic document. A carefully authored document for a web page can easily be printed to a hard-bound volume complete with headers and footers, page numbers and a generated table of contents simply by applying a new stylesheet.

Practical disadvantages today

Currently specifications (for example, XHTML, XSL, CSS) and software tools implementing these specification are only reaching the early stages of maturity. So there are some practical issues facing authors who seek to embrace this method of separating content and style.

Narrow adoption without the parsing and generation tools

While the style specifications are quite mature and still maturing, the software tools have been slow to adapt. Most of the major web development tools still embrace a mixed presentation-content model. So authors and designers looking for GUI based tools for their work find it difficult to follow the semantic web method. In addition to GUI tools, shared repositories for generalized stylesheets would probably aid adoption of these methods.

Chapter 10

Web Workers and WebSockets

Web Workers

Web Workers defines an API for running scripts, basically JavaScript, in the background independently of any user interface scripts. This allows for long-running scripts that are not interrupted by scripts that respond to clicks or other user interactions, and allows long tasks to be executed without yielding to keep the page responsive.

Web workers are relatively heavy-weight, and are not intended to be used in large numbers as they could hog system resources. Browser implementation of Web Workers are different.

Generally, web workers are expected to be long-lived, have a high start-up performance cost, and a high per-instance memory cost.

Overview

Web workers allows for concurrent execution of the browser threads and one or more JavaScript threads running in the background. The browser which follows a single thread of execution will have to wait on JavaScript programs to finish executing before proceeding and this may take significant time which the programmer may like to hide from the user. It allows for the browser to continue with normal operation while running in the background. Web worker specification is a separate specification from HTML5 specification but can be used with HTML5.

There are two types of web workers : **Dedicated worker** and **Shared worker**

When web workers run in the background, they do not have direct access to the DOM but communicate with the document by message passing. It allows for a multi-threaded execution of JavaScript programs.

Features

Web workers interact with the document via message passing. The following code loads a JavaScript file

```
  var worker = new Worker("worker_script.js");
```

To send message to the worker, the `postMessage` method of the worker object is used as shown below

```
  worker.postMessage("Hello World!");
```

The method `onmessage` is used to retrieve information from the worker

```
  worker.onmessage = function(event) {
              alert("received message " + event.data);
              doSomething();
        }

        function doSomething() {
                //do work
                postMessage("Work done!");
        }

        worker.close
```

Once a worker is terminated, it goes out of scope and a new worker has to be created if needed.

Example

The simplest use of web workers is for performing a computationally expensive task without interrupting the user interface.

In this example, the main document spawns a web worker to compute prime numbers, and progressively displays the most recently found prime number.

The main page is as follows:

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Worker example: One-core computation</title>
 </head>
 <body>
  <p>The highest prime number discovered so far is: <output
id="result"></output></p>
  <script type="text/javascript">
   var worker = new Worker('worker.js');
   worker.onmessage = function (event) {
     document.getElementById('result').textContent = event.data;
   };
  </script>
 </body>
</html>
```

The `Worker()` constructor call creates a web worker and returns a `Worker` object representing that web worker, which is used to communicate with the web worker. That object's `onmessage` event handler allows the code to receive messages from the web worker.

The web worker itself is as follows:

```
var n = 1;
search: while (true) {
  n += 1;
  for (var i = 2; i <= Math.sqrt(n); i += 1)
    if (n % i == 0)
     continue search;
  // found a prime!
  postMessage(n);
}
```

To send a message back to the page, the `postMessage()` method is used to post a message when a prime is found.

Support

If the browser supports web workers, a worker property will be available on the global window object . The window property will be undefined if the browser does not support it.

The following example code checks for web worker support on a browser

```
  function detect_web_worker() {
                return !!window.Worker;
        }
```

Third party libraries for JavaScript such as Modernizr can also be used to detect browser functionality for web workers. Web workers is currently supported by Safari, Chrome, Opera and Mozilla Firefox . Internet Explorer, iOS 4.2, and Android 2.3 do not support this.

WebSockets

WebSocket is a technology providing for bi-directional, full-duplex communications channels, over a single Transmission Control Protocol (TCP) socket. It is designed to be implemented in web browsers and web servers but it can be used by any client or server application. The WebSocket API is being standardized by the W3C and the WebSocket protocol is being standardized by the IETF. Since ordinary TCP connections to ports other than 80 are frequently blocked by administrators outside of home environments it can be used as a way to overcome these restrictions and provide similar functionality with some additional protocol overhead while multiplexing several WebSocket services over a single TCP port.

For the client side, WebSocket was to be implemented in Firefox 4, Google Chrome 4, Opera 11 and Safari 5, as well as the mobile version of Safari in iOS 4.2. However, although present, support is now disabled by default in Firefox and Opera, due to concerns over security vulnerabilities.

WebSocket Protocol Handshake

To establish a WebSocket connection, the client sends a WebSocket handshake request, and the server sends a WebSocket handshake response, as shown in the following example:

Proxy traversal

WebSocket protocol client implementations try to detect if the user agent is configured to use a proxy when connecting to destination host and port and, if it is, uses HTTP CONNECT method to set up a persistent tunnel.

While the WebSocket protocol itself is unaware of proxy servers and firewalls, it features an HTTP-compatible handshake so that HTTP servers can share their default HTTP and HTTPS ports (80 and 443) with a WebSocket gateway or server. The WebSocket protocol defines a ws:// and wss:// prefix to indicate a WebSocket and a WebSocket Secure connection, respectively. Both schemes use an HTTP upgrade mechanism to upgrade to the WebSocket protocol. Some proxy servers are harmless and work fine with WebSocket; others will prevent WebSocket from working correctly, causing the connection to fail. In some cases additional proxy server configuration may be required, and certain proxy servers may need to be upgraded to support WebSocket.

If unencrypted WebSocket traffic flows through an explicit or a transparent proxy server on its way to the WebSocket server, then, whether or not the proxy server behaves as it should, the connection is almost certainly bound to fail today (as WebSocket become more mainstream, proxy servers may become WebSocket aware). Therefore, unencrypted WebSocket connections should be used only in the simplest topologies.

If an encrypted WebSocket connection is used, then the use of Transport Layer Security (TLS) in the WebSocket Secure connection ensures that an HTTP CONNECT command is issued when the browser is configured to use an explicit proxy server. This sets up a tunnel, which provides low-level end-to-end TCP communication through the HTTP proxy, between the WebSocket Secure client and the WebSocket server. In the case of transparent proxy servers, the browser is unaware of the proxy server, so no HTTP CONNECT is sent. However, since the wire traffic is encrypted, intermediate transparent proxy servers may simply allow the encrypted traffic through, so there is a much better chance that the WebSocket connection will succeed if WebSocket Secure is used. Using encryption is not free of resource cost, but often provides the highest success rate.

A recent update to the draft (version 76) broke compatibility with reverse-proxies and gateways because the 8 bytes of data the client must send after the headers is not advertised in a Content-Length header, so the intermediates won't forward that data until the handshake completes. And since the handshake needs those 8 bytes to complete, the handshake never completes and deadlocks. In current state of affairs, it's not advisable to modify such intermediate components to support this non-standard HTTP behaviour because doing so would render the components vulnerable to HTTP smuggling attacks, since an attacker would just have to pretend trying to upgrade to the WebSocket protocol in a request to be able to send more data than the target plain HTTP server can parse, possibly bypassing some mandatory security filtering. It is not known if this recent breakage will be worked around in a new draft or not.

URL scheme

The WebSocket protocol specification defines two new URI schemes, **ws:** and **wss:**, for unencrypted and encrypted connections. Apart from the scheme name, the rest of the URI components are defined to use URI generic syntax.

Browsers supporting WebSocket

- Chrome 4
- Safari 5 (includes iOS 4.2)

The following browsers originally supported WebSocket, but have since disabled the protocol by default. Chrome also plans to disable the WebSocket if actual exploit code appears before the protocol is revised.

- Firefox 4
- Opera 11

Currently, two browsers still support the outdated draft-ietf-hybi-thewebsocketprotocol-00. It has been disabled in Firefox and Opera due to security issues.

Microsoft supports the draft-ietf-hybi-thewebsocketprotocol-04 in Internet Explorer through a prototype HTML5 Labs.

Implementation status

Protocol	Internet Explorer	Mozilla Firefox	Google Chrome	Safari	Opera	NetFront
draft-hixie-thewebsocketprotocol-75			4	5.0.0		
draft-hixie-thewebsocketprotocol-76 **draft-ietf-hybi-thewebsocketprotocol-00**		4.0 RC (DISABLED)	6	5.0.1	11.00 (DISABLED)	
draft-ietf-hybi-thewebsocketprotocol-06	HTML5 Labs	dev				

Chapter 11

DBFree

DBFree is a free software package for Microsoft Windows (x32/x64) that includes a web server and an interpreter (Maxscript) intended for developing web applications based on the xBase language dialect and conventions.

- Web applications built with DBFree are built around dynamic web pages (identified by the **.msp** extension, that is automatically recognized by the interpreter, and parsed line-by-line in search of xBase code embedded between **<%** and **%>** markers).
- Anytime valid xBase code is found is processed by the interpreter, and the results of its logic (in form of plain text) are embedded in the exact position where the code was found.
- This realizes true *server-side scripting* because the original code is never passed back to browser, and all the calculations are done on the web server host.
- HTML, Javascript and xBase code can be merged seamlessly. Because the interpretation is taken *before* to pass the page to browser, xBase code can be used to feed Javascript functions with live data, in response to different contexts or conditions.
- Database support is built-in into the interpreter, thus no other software is needed to feed data to web pages; the default drivers provided with installation package give access to standard DBF tables (level 3) used by dBase III.

Other commercial drivers are available, including Clipper, FoxPro, visual dBase and ODBC, but not included. Tables in use by the web application can be accessed via internet from remote clients using web browser concurrently with users using other software capable to manage the DBF file format on the LAN side, like the latest versions of dBase Plus.

- Web pages can also be written using dBase syntax, although in both cases (Clipper 87 or dBase syntax) only commands that make sense in the specific environment of a web page are recognized.Commands include logic branching with IF..ENDIF, looping with DO..WHILE, FOR..NEXT and DO CASE..ENDCASE. MaxScript commands and functions give complete access to host's file system with SET ALTERNATE or memoread() and memowrite()

functions. The webserver (being the preconfigured Xitami or the optional Apache) can be accessed programmatically modifying their initialization file and launching batch files with appropriate links.
- DBFree is derived from Maxsis commercial product DBMax and maintained by volunteers sponsored by the original producers.

Programming in MaxScript

- All Maxscript instructions must be contained between <% and %> tags. Pages to be interpreted must have the .msp extension.
- All pages are to be written in plain text using a text editor (or specialized web authoring tools).

A simple *hello world* application:

```
<html>

<body>

Hello World! Time is <% ? time()%>

</body>

</html>
```

Another example: a FOR..NEXT loop:

```
<html><body>
<p>These are the days of the week:<br>
<%
declare aDays(7)  //-- declaring an array of 7 elements
for iii = 1 to alen(aDays)
   aDays[iii] := cdow(iii)  //-- assigning name of the day (string) to
array element
   ? aDays[iii] + "<br>"    //-- printing list to web page
next
%></p>
</body></html>
```

Another example: using a DBF table:

```
<html><body>
<p>Listing content of table CUSTOMERS.DBF:<hr>
<%
use CUSTOMERS index CUSTOMERS_BY_NAME key CUST_NAME
go top
do while not eof()
   ? recno() | CUST_NAME | CUST_ADDR | CITY | STATE html
   skip
enddo
close CUSTOMERS
%>
```

```
<br>
</body></html>
```

MaxScript's DBFree implementation

DBFree offers a triple choice to programming with MaxScript:

- Standard style
- Ajax mode
- MaxObjects.

All these techniques can easily coexist in the same application: must be noticed that the standard CGI style can be considered a first step to Ajax mode, that is its natural evolution.

Standard Style

The standard CGI programming style is typical of plain MaxScript code, offering maximum backward compatibility with Clipper and Xbase existing code. This programming approach consists into dividing all application logic into single tasks, where almost each operation is demanded to a specific pair of pages, one containing the HTML code for collecting user inputs (with a web form) and the sibling containing the MaxScript code for processing the data submitted by the form.

Main advantages of CGI-style programming:

- Does not depend from external libraries, so pages load faster
- Web forms are presented into static HTML pages relieving the CPU workload
- Code is clearer to read while all links are hardcoded into the pages
- Development is simple and can be done just using a browser with editing tools (like Firefox) because the code for each task is clearly isolated

Some notably disadvantages:

- The web applications tends to have lot of pages (thus becomes more complex to maintain)
- Development can be very tedious because of the higher number of pages to code and the exponential complexity of the resulting web site
- Without libraries most of MaxScript advanced functions won't be available
- Modularity (reusability of code) is greatly reduced while most of the code refer to fixed locations on host
- Every action requires the page to be reloaded (and this can be annoying for the user), even if in most cases using Javascript along with MaxScript can mitigate this inconvenience

Ajax mode

Javascript can be of great help into moving part of workload from server to user's CPU. It can also add lot of interactivity to web pages, especially if the user's browser supports Ajax asynchronous operations.

Using Ajax most of the reloads typical of Standard Style programming can be avoided: more over, performances are greatly enhanced, because of caching mechanisms of modern webservers, with only a minor impact on server's CPU workload. Incorporating Ajax consists mainly into redesigning the user interface so to have a clear division inside the page from static elements and dynamics ones.

Static element (that can be either HTML forms or HTML links) remain untouched inside the page, while dynamics elements (like

elements) and not the page itself become the target of MaxScript calculations, and are going to be updated "in place" without reloading the page.

With this programming style there is a single static web page (hosting all the necessary user interface) that drives the action, calling several active pages, each one providing only the results of data extraction or manipulation done in the background.

Main advantages of Ajax mode programming:

- Results of data extractions can be shown while user is typing
- HTML grids can be made interactive, with "in place editing"

Some notable disadvantages:

- A good understanding of Javascript is required
- Debugging can be very difficult because it is not always clear what a certain page does (and how).
- FrontPage and other visual development tools get "fouled" by this technique (you will have a lot in "code view" not "design view")
- Ajax is a feature of the browser, not of the server. Anyway today almost every computer browser around supports Ajax.

Adopting Ajax with MaxScript should be accompanied with adoption of Json notation.

MaxObjects

MaxObjects are active pages written in MaxScript just like all other active pages we've seen so far, with some notably exceptions:

- MaxObjects expects parameters
- MaxObjects are recursive (they repeatedly call themselves)

- MaxObjects can respond to the caller performing specific actions (usually MaxScript calculations)

MaxObjects can be considered the web counterpart of the "objects" of an Object Oriented Language.

MaxObjects are standard MaxScript pages with specific structure that permits recursive calls of the page itself. Even if MaxObjects can be built with plain MaxScript classic code, DBFree offers specific libraries to facilitate this task. In theory a complete web application can be implemented with a single, large MaxObject: this simple web application can consists in a web made of a single web page that is loaded at the beginning without parameters and that stands waiting for user inputs to call itself after submission to react at the inputs.

MaxObjects' main advantages:

- MaxObjects can be implemented so to be shared between different applications through the use of parameterization
- There is no need to know the internal mechanisms of a MaxObject to make use of it: it is enough to provide the necessary documentation of necessary parameters
- MaxObjects can include thousands of lines of code without any impact on performances: only the branch of code relevant to parameters passed is executed
- MaxObjects may call other MaxObjects in a circular way taking advantage of webserver caching mechanisms
- Reusability of code: pieces of code can be easily moved from an object to another (providing they all use the same libraries and headers)

DBFree and mobiles

MaxScript is well suited for developing data-centric web application for mobiles. Writing such applications is considerably simplified by the extreme compactness of the pages for mobiles when unburdened from HTML beautifying code. Anyway must be noticed that while developing for mobile the Ajax-style is generally not supported.

Chapter 12

Web Accessibility

Web accessibility refers to the inclusive practice of making websites usable by people of all abilities and disabilities. When sites are correctly designed, developed and edited, all users can have equal access to information and functionality. For example, when a site is coded with semantically meaningful HTML, with textual equivalents provided for images and with links named meaningfully, this helps blind users using text-to-speech software and/or text-to-Braille hardware. When text and images are large and/or enlargable, it is easier for users with poor sight to read and understand the content. When links are underlined (or otherwise differentiated) as well as coloured, this ensures that color blind users will be able to notice them. When clickable links and areas are large, this helps users who cannot control a mouse with precision. When pages are coded so that users can navigate by means of the keyboard alone, or a single switch access device alone, this helps users who cannot use a mouse or even a standard keyboard. When videos are closed captioned or a sign language version is available, deaf and hard of hearing users can understand the video. When flashing effects are avoided or made optional, users prone to seizures caused by these effects are not put at risk. And when content is written in plain language and illustrated with instructional diagrams and animations, users with dyslexia and learning difficulties are better able to understand the content. When sites are correctly built and maintained, all of these users can be accommodated while not impacting on the usability of the site for non-disabled users.

The needs that Web accessibility aims to address include:

- **Visual:** Visual impairments including blindness, various common types of low vision and poor eyesight, various types of color blindness;
- **Motor/Mobility:** e.g. difficulty or inability to use the hands, including tremors, muscle slowness, loss of fine muscle control, etc., due to conditions such as Parkinson's Disease, muscular dystrophy, cerebral palsy, stroke;
- **Auditory:** Deafness or hearing impairments, including individuals who are hard of hearing;
- **Seizures:** Photoepileptic seizures caused by visual strobe or flashing effects.
- **Cognitive/Intellectual:** Developmental disabilities, learning disabilities (dyslexia, dyscalculia, etc.), and cognitive disabilities of various origins, affecting

memory, attention, developmental "maturity," problem-solving and logic skills, etc.

Assistive technologies used for web browsing

Individuals living with a disability use assistive technologies such as the following to enable and assist web browsing:

- Screen reader software, which can read out, using synthesized speech, either selected elements of what is being displayed on the monitor (helpful for users with reading or learning difficulties), or which can read out everything that is happening on the computer (used by blind and vision impaired users).
- Braille terminals, consisting of a Refreshable Braille display which renders text as Braille characters (usually by means of raising pegs through holes in a flat surface) and either a QWERTY or Braille keyboard.
- Screen magnification software, which enlarges what is displayed on the computer monitor, making it easier to read for vision impaired users.
- Speech recognition software that can accept spoken commands to the computer, or turn dictation into grammatically correct text - useful for those who have difficulty using a mouse or a keyboard.
- Keyboard overlays, which can make typing easier and more accurate for those who have motor control difficulties.

Guidelines on accessible web design

Web Content Accessibility Guidelines

In 1999 the Web Accessibility Initiative, a project by the World Wide Web Consortium (W3C), published the Web Content Accessibility Guidelines WCAG 1.0. In recent years, these have been widely accepted as the definitive guidelines on how to create accessible websites.

On 11 December 2008, the WAI released the WCAG 2.0 as a Recommendation. WCAG 2.0 aims to be up to date and more technology neutral.

Criticism of WAI guidelines

For a general criticism of the W3C process, read Putting the user at the heart of the W3C process. There was a formal objection to WCAG's original claim that WCAG 2.0 will address requirements for people with learning disabilities and cognitive limitations headed by Lisa Seeman and signed by 40 organisations and people. In articles such as WCAG 2.0: The new W3C guidelines evaluated, To Hell with WCAG 2.0 and Testability Costs Too Much, the WAI has been criticised for allowing WCAG 1.0 to get increasingly out of step with today's technologies and techniques for creating and consuming web content, for the slow pace of development of WCAG 2.0, for making the new guidelines difficult to navigate and understand, and other argued failings.

Other guidelines

Canada

Canada has the Common Look and Feel Standards requiring federal government internet websites to meet Web Content Accessibility Guidelines (WCAG) 1.0 Checkpoints Priorities 1 and 2 (Double A conformance level). The standards have existed since 2000 and were updated in 2007.

Philippines

As part of the Web Accessibility Initiatives in the Philippines, the government through the National Council for the Welfare of Disabled Persons (NCWDP) board approved the recommendation of forming an adhoc or core group of webmasters that will help in the implementation of the Biwako Millennium Framework set by the UNESCAP.

The Philippines was also the place where the Interregional Seminar and Regional Demonstration Workshop on Accessible Information and Communications Technologies (ICT) to Persons with Disabilities was held where eleven countries from Asia - Pacific were represented. The Manila Accessible Information and Communications Technologies Design Recommendations was drafted and adopted in 2003.

Spain

In Spain, UNE 139803 is the norm entrusted to regulate web accessibility. This standard is based on Web Content Accessibility Guidelines 1.0.

Sweden

In Sweden, Verva, the Swedish Administrative Development Agency is responsible for a set of guidelines for Swedish public sector web sites. Through the guidelines, Web accessibility is presented as an integral part of the overall development process and not as a separate issue.

The Swedish guidelines contain criteria which cover the entire lifecycle of a website; from its conception to the publication of live web content. These criteria address several areas which should be considered, including:

- accessibility
- usability
- web standards
- privacy issues
- information architecture
- developing content for the web
- Content Management Systems (CMS) / authoring tools selection.
- development of web content for mobile devices.

An English translation was released in April 2008: Swedish National Guidelines for Public Sector Websites

The translation is based on the latest version of Guidelines which was released in 2006.

United Kingdom

In December 2010, the BSI (British Standards Institute) released the standard BS 8878:2010 Web accessibility. Code of practice. This standard effectively supersedes PAS 78 (pub. 2006). PAS 78, produced by the The Disability Rights Commission and British Standards Institution, provided guidance to organisations in how to go about commissioning an accessible website from a design agency. It describes what is expected from websites to comply with the UK Disability Discrimination Act 1995 (DDA), making websites accessible to and usable by disabled people.

BS 8878:2010 Web accessibility - Code of Practice. The standard has been designed to introduce non-technical professionals to improved accessibility, usability and user experience for disabled and older people. It will be especially beneficial to anyone new to this subject as it gives guidance on process, rather than on technical and design issues. BS 8878 is consistent with the Equality Act 2010 and is referenced in the UK government's e-Accessibility Action Plan as the basis of updated advice on developing accessible online services. It includes recommendations for:

- Involving disabled people in the development process and using automated tools to assist with accessibility testing
- The management of the guidance and process for upholding existing accessibility guidelines and specifications.

BS 8878 is intended for anyone responsible for the policies covering web product creation within their organization, and governance against those policies (e.g. Chief Executive Officers, Managing Directors, Headteachers, ICT managers). It would also assist:

- People responsible for promoting and supporting equality and inclusion initiatives within an organization (e.g. Human Resource (HR) managers or those responsible for Corporate Social Responsibility - CSR).
- Procurement managers (e.g. those responsible for procuring web products or the tools to create them such as content production systems or virtual learning environments).
- Web production teams (e.g. product owners, project managers, technical architects and web developers, designers, usability and accessibility engineers, test engineers).
- People with responsibility for creating or shaping online content (e.g. website editors, marketing managers, web content authors).
- People who create web production, testing or validation tools.

- People who write and deliver training courses in web production, design or coding.

Other audiences that might also be interested in this British Standard include:

- Assistive technology creators, vendors and trainers who need insights into how their technologies impact on the production of accessible web products.
- Those disabled and older people whose web accessibility needs the Standard aims to support and present.

Japan

Web Content Accessibility Guidelines in Japan was established in 2004 as JIS (Japanese Industrial Standards) X 8341-3. JIS X 8341-3 will be revised within 2009 by adopting WCAG 2.0. New version will have the same 4 principles, 12 guidelines, and 61 success criteria as WCAG 2.0 has.

Essential Components of Web Accessibility

In order for the web to be accessible, 7 components must be included:

1. the content on Web pages must be natural information (text, images, and sound),
2. Web browsers and media players,
3. assistive technologies,
4. users' knowledge and experience using the Web,
5. developers,
6. authoring tools
7. evaluation tools

These components interact with each other to create an environment that is accessible to people with disabilities.

Web **developers** usually use **authoring tools** and evaluation tools to create Web **content**. **People** ("**users**") use Web **browsers**, **media players**, **assistive technologies** or other "**user agents**" to get and interact with the **content**."

Guidelines for Different Components

Authoring Tool Accessibility Guidelines (ATAG)

- ATAG contains 28 checkpoints that provide guidance on:
  - producing accessible output that meets standards and guidelines
  - promoting the content author for accessibility-related information
  - providing ways of checking and correcting inaccessible content
  - integrating accessibility in the overall look and feel
  - making the authoring tool itself accessible to people with disabilities

Web Content Accessibility Guidelines (WCAG)

- WCAG 1.0: 14 guidelines that are general principles of accessible design
- WCAG 2.0: 12 principal guidelines

User Agent Accessibility Guidelines (UAAG)

- UAAG contains a comprehensive set of checkpoints that cover:
    - access to all content
    - user control over how content is rendered
    - user control over the user interface
    - standard programming interfaces

Legally required web accessibility

A growing number of countries around the world have introduced legislation which either directly addresses the need for websites and other forms of communication to be accessible to people with disabilities, or which addresses the more general requirement for people with disabilities not to be discriminated against.

Australia

In 2000, an Australian blind man won a court case against the Sydney Organizing Committee of the Olympic Games (SOCOG). This was the first successful case under Disability Discrimination Act 1992 because SOCOG had failed to make their official website, Sydney Olympic Games, adequately accessible to blind users. The Human Rights and Equal Opportunity Commission (HREOC) also published World Wide Web Access: Disability Discrimination Act Advisory Notes. All Governments in Australia also have policies and guidelines that require accessible public websites; Vision Australia maintain a complete list of Australian web accessibility policies.

Ireland

In Ireland, the Disability Act 2005 was supplemented with the National Disability Authority's Code of Practice on Accessible Public Services in July 2006. It is a practical guide to help all Government Departments and nearly 500 public bodies to comply with their obligations under the Disability Act 2005.

United Kingdom

In the UK, the Disability Discrimination Act 1995 (DDA) does not refer explicitly to website accessibility, but makes it illegal to discriminate against people with disabilities. The DDA applies to anyone providing a service; public, private and voluntary sectors. The Code of Practice: Rights of Access - Goods, Facilities, Services and Premises document published by the government's Disability Rights Commission to accompany the Act does refer explicitly to websites as one of the "services to the public" which should be considered covered by the Act.

Website accessibility audits

A growing number of organizations, companies and consultants offer *website accessibility audits*. These audits, a type of system testing, identify accessibility problems that exist within a website, and provide advice and guidance on the steps that need to be taken to correct these problems.

A range of methods are used to audit websites for accessibility:

- Automated tools are available which can identify some of the problems that are present.
- Expert technical reviewers, knowledgeable in web design technologies and accessibility, can review a representative selection of pages and provide detailed feedback and advice based on their findings.
- User testing, usually overseen by technical experts, involves setting tasks for ordinary users to carry out on the website, and reviewing the problems these users encounter as they try to carry out the tasks.

Each of these methods has its strengths and weaknesses:

- Automated tools can process many pages in a relatively short length of time, but can only identify some of the accessibility problems that might be present in the website.
- Technical expert review will identify many of the problems that exist, but the process is time consuming, and many websites are too large to make it possible for a person to review every page.
- User testing combines elements of usability and accessibility testing, and is valuable for identifying problems that might otherwise be overlooked, but needs to be used knowledgeably to avoid the risk of basing design decisions on one user's preferences.

Ideally, a combination of methods should be used to assess the accessibility of a website.

Accessible Web applications and WAI-ARIA

For a Web page to be accessible all important semantics about the page's functionality must be available so that assistive technology can understand and process the content and adapt it for the user. However as content becomes more and more complex, the standard HTML tags and attributes become inadequate in providing semantic reliably. Modern Web applications often apply scripts to elements to control their functionality and to enable them to act as a control or other dynamic component. These custom components or widgets do not provide a way to convey semantic information to the user agent. WAI-ARIA (Accessible Rich Internet Applications) is a specification published by the World

Wide Web Consortium that specifies how to increase the accessibility of dynamic content and user interface components developed with Ajax, HTML, JavaScript and related technologies. ARIA enables accessibility by enabling the author to provide all the semantics to fully describe its supported behaviour. It also allows each element can expose its current states and properties and its relationships between other elements. Accessibility problems with the focus and tab index are also corrected.

Chapter 13

Adobe Flash

Adobe Flash (formerly **SmartSketch FutureSplash**, **FutureSplash Animator** and **Macromedia Flash**) is a multimedia platform used to add animation, video, and interactivity to web pages. Flash is frequently used for advertisements and games. More recently, it has been positioned as a tool for "Rich Internet Applications" ("RIAs").

Flash manipulates vector and raster graphics to provide animation of text, drawings, and still images. It supports bidirectional streaming of audio and video, and it can capture user input via mouse, keyboard, microphone, and camera. Flash contains an object-oriented language called ActionScript.

Flash content may be displayed on various computer systems and devices, using Adobe Flash Player, which is available free of charge for common web browsers, some mobile phones and a few other electronic devices (using Flash Lite).

Some users feel that Flash enriches their web experience, while others find the extensive use of Flash animation, particularly in advertising, intrusive and annoying, giving rise to a cottage industry that specializes in blocking Flash content. Flash has also been criticized for adversely affecting the usability of web pages.

History

Originally developed by Macromedia, Flash was introduced in 1996, and is currently developed and distributed by Adobe Systems, as the result of their 2005 purchase of the company. The precursor to the Flash application was SmartSketch, a drawing application for pen computers running the PenPoint OS developed by Jonathan Gay, who began working on it in college and extended the idea for Silicon Beach Software and its successors. When PenPoint failed in the marketplace, SmartSketch was ported to Microsoft Windows and Mac OS. With the Internet becoming more popular, SmartSketch was re-released as FutureSplash, a vector-based web animation in competition with Macromedia Shockwave. In 1995, SmartSketch was further modified with frame-by-frame animation features and re-released as FutureSplash Animator on multiple platforms. The product was offered to Adobe and used by Microsoft in its early work with the Internet (MSN). In 1996, FutureSplash was acquired by Macromedia and released as **Flash,** contracting "Future" and "Splash".

Recent developments

Adobe Labs (previously called *Macromedia Labs*) is a source for news and pre-release versions of emerging products and technologies from Adobe. Most innovations, such as Flash 10, Flex 3, and ActionScript 3.0 have all been discussed and/or trialled on the site.

One area Adobe is focusing on (as of February 2009) is the deployment of Rich Internet Applications (RIAs). To this end, they released Adobe Integrated Runtime (AIR), a cross-platform runtime environment which can be used to build, using Adobe Flash, rich Internet applications that can be deployed as desktop applications. It surpassed 100 million installations worldwide in February 2009. Flash is installed silently when Acrobat Reader is installed.

Two additional components designed for large-scale implementation have been proposed by Adobe for future releases of Flash: first, the option to require an ad to be played in full before the main video piece is played; and second, the integration of digital rights management (DRM) capabilities. This way Adobe can give companies the option to link an advertisement with content and make sure that both are played and remain unchanged.

Flash Player for smart phones is available to handset manufacturers at the end of 2009.

Open Screen Project

On May 1, 2008 Adobe announced *Open Screen Project*, which hopes to provide a consistent application interface across devices such as personal computers, mobile devices and consumer electronics. When the project was announced, several goals were outlined: the abolition of licensing fees for Adobe Flash Player and Adobe Integrated Runtime, the removal of restrictions on the use of the Shockwave Flash (SWF) and Flash Video (FLV) file formats, the publishing of application programming interfaces for porting Flash to new devices and the publishing of The Flash Cast protocol and Action Message Format (AMF), which let Flash applications receive information from remote databases.

As of February 2009, the specifications removing the restrictions on the use of SWF and FLV/F4V specs have been published. The Flash Cast protocol—now known as the Mobile Content Delivery Protocol—and AMF protocols have also been made available, with AMF available as an open source implementation, BlazeDS. Work on the device porting layers is in the early stages. Adobe intends to remove the licensing fees for Flash Player and Adobe AIR for devices at their release for the Open Screen Project.

The list of mobile device providers who have joined the project includes Palm, Motorola and Nokia, who, together with Adobe, have announced a $10 million Open Screen Project fund.

Format

Flash files are in the *SWF* format, traditionally called "**S**hock**W**ave **F**lash" movies, "Flash movies," or "Flash applications", usually have a .swf file extension, and may be used in the form of a web page plug-in, strictly "played" in a standalone Flash Player, or incorporated into a self-executing Projector movie (with the .exe extension in Microsoft Windows). Flash Video files have a .flv file extension and are either used from within .swf files or played through a flv-aware player, such as VLC, or QuickTime and Windows Media Player with external codecs added.

The use of vector graphics combined with program code allows Flash files to be smaller — and thus for streams to use less bandwidth — than the corresponding bitmaps or video clips. For content in a single format (such as just text, video, or audio), other alternatives may provide better performance and consume less CPU power than the corresponding Flash movie, for example when using transparency or making large screen updates such as photographic or text fades.

In addition to a vector-rendering engine, the Flash Player includes a virtual machine called the ActionScript Virtual Machine (AVM) for scripting interactivity at run-time, support for video, MP3-based audio, and bitmap graphics. As of Flash Player 8, it offers two video codecs: On2 Technologies VP6 and Sorenson Spark, and run-time support for JPEG, Progressive JPEG, PNG, and GIF. In the next version, Flash is slated to use a just-in-time compiler for the ActionScript engine.

Flash Player is a browser plugin, and cannot run within a usual e-mail client, such as Outlook. Instead, a link must open a browser window. A Gmail labs feature allows playback of YouTube videos linked in emails.

Flash Video

Virtually all browser plugins for video are free of charge and cross-platform, including Adobe's offering of Flash Video, which was first introduced with Flash version 6. Flash Video has been a popular choice for websites due to the large installed user base and programmability of Flash. In 2010, Apple publicly criticized Adobe Flash, including its implementation of video playback for not taking advantage of hardware acceleration, one reason Flash is not to be found on Apple's mobile devices. Soon after Apple's criticism, Adobe demoed and released a beta version of Flash 10.1, which takes advantage of GPU hardware acceleration even on a Mac. Flash 10.2 beta, released December 2010, adds hardware acceleration for the whole video rendering pipeline.

Flash Audio

Flash Audio is most commonly encoded in MP3 or AAC (Advanced Audio Coding) however it does also support ADPCM, Nellymoser (Nellymoser Asao Codec) and Speex audio codecs. Flash allows sample rates of 11, 22 and 44.1 kHz. It does not support 48 kHz audio sample rate which is the standard TV, DVD sample rate.

On August 20, 2007, Adobe announced on its blog that with Update 3 of Flash Player 9, Flash Video will also support some parts of the MPEG-4 international standards. Specifically, Flash Player will have support for video compressed in H.264 (MPEG-4 Part 10), audio compressed using AAC (MPEG-4 Part 3), the F4V, MP4 (MPEG-4 Part 14), M4V, M4A, 3GP and MOV multimedia container formats, 3GPP Timed Text specification (MPEG-4 Part 17) which is a standardized subtitle format and partial parsing support for the 'ilst' atom which is the ID3 equivalent iTunes uses to store metadata. MPEG-4 Part 2 and H.263 will not be supported in F4V file format. Adobe also announced that it will be gradually moving away from the FLV format to the standard ISO base media file format (MPEG-4 Part 12) owing to functional limits with the FLV structure when streaming H.264. The final release of the Flash Player supporting some parts of MPEG-4 standards had become available in Fall 2007.

Adobe Flash Player 10.1 does not support acoustic echo cancellation, unlike the VoIP offerings of Skype and Google Voice, making this and earlier versions of Flash less suitable for group calling or meetings. Flash Player 10.3 Beta incorporates acoustic echo cancellation.

Proprietary restrictions

The proprietary nature of Flash has been a concern to advocates of open standards and free software. Its widespread use has, according to such observers, harmed the otherwise open nature of the World Wide Web. A response may be seen in Adobe's Open Screen Project: Adobe's restrictions on the use of the SWF/FLV specifications have been lifted.

Representing open standards, inventor of CSS and co-author of HTML5, Håkon Wium Lie explained in a Google tech talk entitled "the <video> element" the proposal of Theora as the format for HTML5 video:

I believe very strongly, that we need to agree on some kind of baseline video format if [the video element] is going to succeed. Flash is today the baseline format on the web. The problem with Flash is that it's not an open standard.

Disclosure

In October 1998, Macromedia disclosed the Flash Version 3 Specification to the world on its website. It did this in response to many new and often semi-open formats competing with SWF, such as Xara's Flare and Sharp's Extended Vector Animation formats. Several developers quickly created a C library for producing SWF. In February 1999, the company introduced MorphInk 99, the first third-party program to create SWF files. Macromedia also hired Middlesoft to create a freely available developers' kit for the SWF file format versions 3 to 5.

Macromedia made the Flash Files specifications for versions 6 and later available only under a non-disclosure agreement, but they are widely available from various sites.

In April 2006, the Flash SWF file format specification was released with details on the then newest version format (Flash 8). Although still lacking specific information on the incorporated video compression formats (On2, Sorenson Spark, etc.), this new documentation covered all the new features offered in Flash v8 including new ActionScript commands, expressive filter controls, and so on. The file format specification document is offered only to developers who agree to a license agreement that permits them to use the specifications only to develop programs that can export to the Flash file format. The license forbids the use of the specifications to create programs that can be used for playback of Flash files. The Flash 9 specification was made available under similar restrictions.

In June 2009, Adobe launched the Open Screen Project (Adobe link), which made the SWF specification available without restrictions. Previously, developers could not use the specification for making SWF-compatible players, but only for making SWF-exporting authoring software. The specification still omits information on codecs such as Sorenson Spark, however.

Authoring tools

Adobe Flash Professional

The Adobe Flash Professional multimedia authoring program is used to create content for the Adobe Engagement Platform, such as web applications, games and movies, and content for mobile phones and other embedded devices.

History

Adobe Flash Professional is the successor of a software product known as **FutureSplash Animator**, a vector graphics and vector animations program released in May 1996. FutureSplash Animator was developed by FutureWave Software, a small software company whose first product, SmartSketch, was a vector-based drawing program for pen-based computers. In 1995, the company decided to add animation capabilities to their product and to create a vector-based animation platform for World Wide Web; hence FutureSplash Animator was created. Initially, the only way to deploy such animations on the web was through the use of Java platform; however, the Java platform was later replaced with the Netscape's plug-in architecture. The FutureSplash animation technology was used on several notable websites such as MSN, the official *The Simpsons* website and *Disney Daily Blast* of The Walt Disney Company.

In December 1996, Macromedia bought FutureWave and so re-branded and released FutureSplash Animator as *Macromedia Flash* v1.0. In 2005, Adobe Systems acquired Macromedia; subsequently, in 2007, *Adobe Flash CS3 Professional*, the next version of Macromedia Flash was released.

Release	Year	Description
FutureSplash Animator	1996	Initial version of Flash with basic editing tools and a timeline
Macromedia Flash 1	1996	A re-branded version of the FutureSplash Animator
Macromedia Flash 2	1997	Released with Flash Player 2, new features included: the object library
Macromedia Flash 3	1998	Released with Flash Player 3, new features included: the movieclip element, JavaScript plug-in integration, transparency and an external stand alone player
Macromedia Flash 4	1999	Released with Flash Player 4, new features included: internal variables, an input field, advanced ActionScript, and streaming MP3
Macromedia Flash 5	2000	Released with Flash Player 5, new features included: ActionScript 1.0 (based on ECMAScript, making it very similar to JavaScript in syntax), XML support, Smartclips (the precursor to components in Flash), HTML text formatting added for dynamic text
Macromedia Flash MX(6)	2002	Released with Flash Player 6, new features included: a video codec (Sorenson Spark), Unicode, v1 UI Components, compression, ActionScript vector drawing API
Macromedia Flash MX 2004(7)	2003	Released with Flash Player 7, new features included: Actionscript 2.0 (which enabled an object-oriented programming model for Flash, although it lacked the Script assist function of other versions, meaning Actionscript could only be typed out manually), behaviors, extensibility layer (JSAPI), alias text support, timeline effects. Macromedia Flash MX Professional 2004 included all Flash MX 2004 features, plus: Screens (forms for non-linear state-based development and slides for

		organizing content in a linear slide format like PowerPoint), web services integration, video import wizard, Media Playback components (which encapsulate a complete MP3 and/or FLV player in a component that may be placed in an SWF), Data components (DataSet, XMLConnector, WebServicesConnector, XUpdateResolver, etc.) and data binding APIs, the Project Panel, v2 UI components, and Transition class libraries.
Macromedia Flash 8	2005	Macromedia Flash Basic 8, a less feature-rich version of the Flash authoring tool targeted at new users who only want to do basic drawing, animation and interactivity. Released with Flash Player 8, this version of the product has limited support for video and advanced graphical and animation effects. Macromedia Flash Professional 8 added features focused on expressiveness, quality, video, and mobile authoring. New features included Filters and blend modes, easing control for animation, enhanced stroke properties (caps and joins), object-based drawing mode, run-time bitmap caching, FlashType advanced anti-aliasing for text, On2 VP6 advanced video codec, support for alpha transparency in video, a stand-alone encoder and advanced video importer, cue point support in FLV files, an advanced video playback component, and an interactive mobile device emulator.

Adobe Flash CS3(9) Professional	2007	Flash CS3 is the first version of Flash released under the Adobe name. CS3 features full support for ActionScript 3.0, allows entire applications to be converted into ActionScript, adds better integration with other Adobe products such as Adobe Photoshop, and also provides better Vector drawing behavior, becoming more like Adobe Illustrator and Adobe Fireworks.
Adobe Flash CS4(10) Professional	2008	Contains inverse kinematics (bones), basic 3D object manipulation, object-based animation, a text engine, and further expansions to ActionScript 3.0. CS4 allows the developer to create animations with many features absent in previous versions.
Adobe Flash Professional CS5(10.1)	2010	Flash CS5 was released on April 12, 2010 and launched for trialling and normal buying on April 30, 2010. Flash CS5 Professional includes support for publishing iPhone applications. However, on April 8, 2010 Apple changed the terms of its Developer License to effectively ban the use of the Flash-to-iPhone compiler and on April 20, 2010 Adobe announced that they will be making no additional investments in targeting the iPhone and iPad in Flash CS5. Other features of Flash CS5 are a new text engine (TLF), further improvement to inverse kinematics, and the Code Snippets panel.

Third-party tools

Open Source projects like Ajax Animator and the (now defunct) UIRA aim to create a Flash development environment, complete with a graphical user environment.

Alternatively, programs such as swfmill, SWFTools, and MTASC provide tools to create SWF files, but do so by compiling text, actionscript or XML files into Flash animations. It is also possible to create SWF files programmatically using the Ming library, which has interfaces for C, PHP, C++, Perl, Python, and Ruby. haXe is an open source, high-level object-oriented programming language geared towards web-content creation that can compile Flash files.

Many shareware developers produced Flash creation tools and sold them for under US$50 between 2000 and 2002. In 2003 competition and the emergence of free Flash creation tools had driven many third-party Flash-creation tool-makers out of the market, allowing the remaining developers to raise their prices, although many of the products still cost less than US$100 and support ActionScript. As for open source tools, KToon can edit vectors and generate SWF, but its interface is very different from Macromedia's. Another, more recent example of a Flash creation tool is SWiSH Max made by an ex-employee of Macromedia. Toon Boom Technologies also sells a traditional animation tool, based on Flash.

In addition, several programs create .swf-compliant files as output from their programs. Among the most famous of these are Screencast tools, which leverage the ability to do lossless compression and playback of captured screen content in order to produce demos, tutorials, or software simulations of programs. These programs are typically designed for use by non-programmers, and create Flash content quickly and easily, but cannot actually edit the underlying Flash code (i.e. the tweening and transforms, etc.) Screencam is perhaps the oldest screencasting authoring tool to adopt Flash as the preferred output format, having been developed since the mid-90s. The fact that screencasting programs have adopted Flash as the preferred output is testament to Flash's presence as a ubiquitous cross-platform animation file format.

Other tools are focused on creating specific types of Flash content. Anime Studio is a 2D animation software package specialized for character animation which creates SWF files. Express Animator is similarly aimed specifically at animators. Question Writer publishes its quizzes to Flash file format.

Users who are not programmers or web designers will also find on-line tools that allow them to build full Flash-based websites. One of the oldest services available (1998) is FlashToGo. Such companies provide a wide variety of pre-built models (templates) associated to a Content Management System that empowers users to easily build, edit and publish their websites. Other sites, that allows for greater customization and design flexibility are Wix.com and CirclePad.

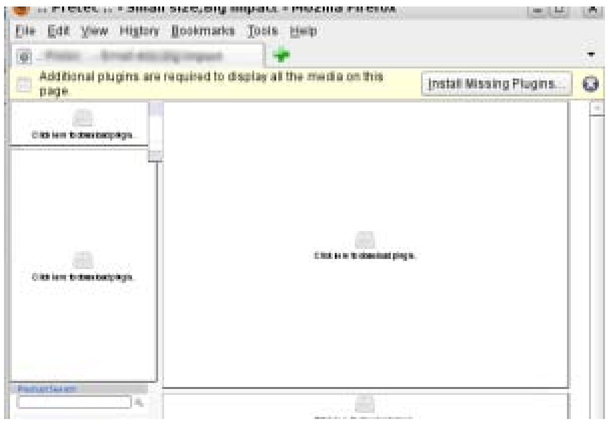
Adobe wrote a software package called Adobe LiveMotion, designed to create interactive animation content and export it to a variety of formats, including SWF. LiveMotion went through two major releases, but failed to gain any notable user base.

In February 2003, Macromedia purchased Presedia, which had developed a Flash authoring tool that automatically converted PowerPoint files into Flash. Macromedia subsequently released the new product as Breeze, which included many new enhancements. In addition, (as of version 2) Apple's Keynote presentation software also allows users to create interactive presentations and export to SWF.

User experience

Flash as a format has become widespread on the desktop market; one estimate is that 95% of PCs have it, while Adobe claims that 98 percent of U.S. web users and 99.3 percent of all Internet desktop users have installed the Flash Player, with 92 to 95% (depending on region) having the latest version. Numbers vary depending on the detection scheme and research demographics.

The Adobe Flash Player exists for a variety of systems and devices: Windows, Mac OS 9/X, Linux, Solaris, HP-UX, Pocket PC/Windows CE, OS/2, QNX, Android, Symbian, Palm OS, BeOS, and IRIX, although the performance is typically best on Windows.


File Edit View History Bookmarks Tools Help
Additional plugins are required to display all the media on this page.
Install Missing Plugins...
Click here to download plugin.
Click here to download plugin.
Click here to download plugin.

Some websites rely on Flash so heavily that they are totally unusable without this plugin

Among mobile devices, Flash has less penetration because Apple does not bundle or allow third-party runtimes on its iPhone, which accounts for more than 60% of global smartphone web traffic, or the iPod touch, which makes up more than 95% of "mobile Internet device" traffic. This hurts Adobe's ability to market Flash as a ubiquitous mobile platform. However, Flash is enabled on competing mobile platforms, including the version 2.2 Android while other O.S.s such as Symbian and Palm have versions coming.. On Mar 8, 2011, Techradar reported that Adobe provides an experimental server side tool (Wallaby) to convert Flash programs (as far as possible) to HTML5 code, thus allowing iOS devices to display the content.

Downloading Flash is blocked in countries that are under U.S sanctions (such as Syria & Sudan). Users in these countries are blocked (by Adobe) from downloading Flash plug-ins for both Internet Explorer and Firefox browsers.

Flash content is usually embedded using the <object> html tag, or the nonstandard <embed> tag. Software that does not support either of these tags, and users who cannot or will not install a plugin, will see the replacement text if this is supplied by the web page.

Accessibility

Using Flash tends to break conventions associated with normal HTML pages. Selecting text, scrolling, form control and right-clicking act differently than with a regular HTML webpage. Many such interface unexpectancies are fixable by the designer. Usability expert Jakob Nielsen published an Alertbox in 2000 entitled, *Flash: 99% Bad* which listed issues like these. Some problems have been improved upon since Nielsen's complaints:

- Text size can be controlled using full page zoom, found in many modern browsers.
- It has been possible for authors to include alternative text in Flash since Flash Player 6. This accessibility feature is compatible only with certain screen readers and only under Windows.

Performance

- Any Flash player has to be able to animate on top of video renderings, which makes hardware accelerated video rendering at least not as straightforward as with a purpose-built multimedia player. Therefore, even when only displaying video, Flash players are more resource-intensive than dedicated video player software.
- Comparisons have shown Adobe Flash Player to perform better on Windows than Mac OS X and Linux with the same hardware. However, the 10.1 update significantly improved performance for Mac OS X.

Flash blocking in web browsers

Some web browsers default to not play Flash content before the user clicks on it, e.g. Konqueror, K-Meleon. Equivalent "Flash blocker" extensions also exist for many popular browsers: Firefox has NoScript and Flashblock, and Opera versions since 10.5 feature native Flash blocking. Opera Turbo requires the user to click to play Flash content. Internet Explorer has Foxie, which contains a number of features, one of them also named Flashblock. WebKit-based browsers under Mac OS X, such as Apple's Safari, have ClickToFlash.

Flash client security

Flash's security record has caused several security experts to recommend to either not install Flash or to block it. The US-CERT recommends to block Flash using NoScript. Charlie Miller recommended "not to install Flash" at the computer security conference CanSecWest. As of October 31, 2010, The Flash Player has over 100 CVE entries, 65 of which have been ranked with a high severity (leading to arbitrary code execution), and 40 ranked medium. In February 2010, Adobe officially apologized for not fixing a known vulnerability for over 1 year. In June 2010 Adobe announced a "critical vulnerability" in recent versions, saying there are reports that this vulnerability is being actively exploited in the wild against both Adobe Flash Player, and Adobe Reader and Acrobat. Later, in October 2010, Adobe announced another critical vulnerability, this time also affecting Android-based mobile devices. Android users have been recommended to disable Flash or make it only on demand.

Symantec's Internet Security Threat Report states that a remote code execution in Adobe Reader and Flash Player was the second most attacked vulnerability in 2009. The same report also recommends to employ browser add-ons wherever possible to disable Adobe Flash Player when visiting untrusted sites. McAfee predicted that Adobe software, especially Reader and Flash, would be primary target for attacks in 2010. Adobe applications had become, at least at some point, the most popular client-software targets for attackers during the last quarter of 2009.

Local Shared Objects (“Flash cookies”)

Like the HTTP cookie, a flash cookie (also known as a “Local Shared Object”) can be used to save application data. Flash cookies are not shared across domains. An August 2009 study by the Social Science Research Network found that 50% of websites using Flash were also employing flash cookies, yet privacy policies rarely disclosed them, and user controls for privacy preferences were lacking. Most browsers' cache and history suppress or delete functions do not affect Flash Player's writing Local Shared Objects to its own cache, and the user community is much less aware of the existence and function of Flash cookies than HTTP cookies. Thus, users having deleted HTTP cookies and purged browser history files and caches may believe that they have purged all tracking data from their computers when in fact Flash browsing history remains. Adobe's own Flash Website Storage Settings panel, a submenu of Adobe's Flash Settings Manager web application, and other editors and toolkits can manage settings for and delete Flash Local Shared Objects.

64-bit support

Adobe's 64-bit Flash player is available as a preview3 release ("Square"), which was released in September 2010. The "Square" preview is available for Windows, Mac and Linux. This new version can be downloaded at the Adobe lab site.

The key new capabilities in the Flash Player "Square" preview are:

- 64-bit support — Native support for 64-bit operating systems and 64-bit web browsers on Linux, Mac OS, and Windows. (Hulu and Amazon which depends on RTMPE are not currently functioning because there are some 64-bit libs that need to be integrated into the branch"Adobe Forums: Flash Player "Square": 64-bit".)

- Internet Explorer 9 hardware accelerated rendering support — Enhanced support for Internet Explorer 9 Beta. It takes advantage of hardware accelerated graphics in Internet Explorer 9 Beta, utilizing hardware rendering surfaces to improve graphics performance and enable seamless composition.

The first experimental release of 64-bit builds of Adobe Flash Player was for the Linux platform, on November 11, 2008.

The project was closed temporarily on June 15, 2010, while Adobe was preparing for the preview release on September 15, 2010.

The official 32-bit player is still distributed in 64-bit Linux distributions e.g. Ubuntu, openSUSE, of which some users have reported problems with the 32-bit player on some websites. Affected users can install the 64-bit player manually or through a special repository.

Adobe expects to provide 64-bit versions of its Flash Player for Windows, Macintosh and Linux with an upcoming major release of Adobe Flash Player.

Alternatives to Flash

HTML5

HTML 5 is gaining ground as a competitor to Flash: the canvas element assists animation, and text can be more easily synchronized with audio and video element timeupdate events. In one example of this, Scribd, a 50 million user a month document sharing website, announced in May 2010 that after three years of investment in Flash, it is changing from that platform to the HTML5 standard. YouTube introduced HTML5 support in January 2010, and on January 11, 2011, the Google Chromium Project announced on their blog that support for closed codecs (particularly H.264) would be removed from future releases of Chrome. The Chromium announcement specifically mentioned that this was an effort to increase the use of license-free HTML5 and the `video` element, and drive web-wide adoption of the open-source codecs VP8 and Theora.

Microsoft Silverlight

In recent years, Microsoft Silverlight has emerged as a potential competitor to Flash. While not yet as prevalent on websites as Flash, Silverlight has been used to provide video streaming for many high profile events, including the 2008 Summer Olympics in Beijing, the 2010 Winter Olympics in Vancouver, and the 2008 conventions for both major political parties in the United States. Silverlight is also used by Netflix for its instant video streaming service.

Java

Java applets are used both to create interactive visualisations and to present video, three dimensional objects and other media. Java applets are more appropriate for complex visualizations that require significant programming effort in high level language or communications between applet and originating server. Sun's new JavaFX is considered as another competitor for Rich Internet Applications.

Other open alternatives

There are equivalent open standards for many simple uses of Flash. Most notably the SVG and SMIL file formats, the *canvas*, *audio* and *video* HTML elements, and the JavaScript programming language. More complex use cases can be achieved by combining these.

The W3C's SVG and SMIL standards are seen as the nearest equivalents of Flash. Opera has supported SVG since version 8 and Safari has since version 3, and Firefox's built-in support for SVG continues to grow. Adobe formerly developed and distributed the 'Adobe SVG Viewer' client plug-in for Internet Explorer, but discontinued support and distribution on January 1, 2009. This was in a time when Adobe went from competing with Macromedia's Flash to owning the technology itself.

UIRA was a free software project that intended to become a complete replacement for Adobe Flash. The project collapsed in mid 2007, though people are now discussing reviving or continuing it, and a few other projects like Ajax Animator still exist.

Third-party players

Since Flash files do not depend on an open standard such as SVG, this reduces the incentive for non-commercial software to support the format, although there are several third party tools which use and generate the SWF file format. Flash Player cannot ship as part of a pure open source, or completely free operating system, as its distribution is bound to the Macromedia Licensing Program and subject to approval.

There is, as of late 2008, no complete free software replacement which offers all the functionality of the latest version of Adobe Flash Player.

Presenting the free software movement, Richard Stallman stated in a speech in October 2004 that:

The use of Flash in websites is a major problem for our community.

Stallman's argument then was that no free players were comparatively good enough. As of February 2010, Gnash and Swfdec have seen limited success in competing with Adobe's player. Many important and popular websites require users to have a Flash player, sometimes with no fallback for non-Flash web users. Therefore, the lack of a good free Flash player is arguably an obstacle to enjoying the web with free software, and the aforementioned ubiquity of Flash makes the problem very evident for anyone who tries. The continual high ranking of Gnash on the Free Software Foundation's list of high priority projects might indicate the severity of the problem, as judged by the free software community.

Gnash is an active project that aims to create a free player and browser plugin for the Adobe Flash file format and so provide a free alternative to the Adobe Flash Player under the GNU General Public License. Despite potential patent worries because of the proprietary nature of the files involved, Gnash supports most SWF v7 features and some SWF v8 and v9. Gnash runs on Windows, Linux and other operating systems on 32-bit, 64-bit and other architectures.

Swfdec is another open-source flash player available for Linux, FreeBSD and OpenBSD.

Lightspark is a new implementation aiming to create a more modern and fast player. Besides hardware-accelerated rendering, it exploits multithreading and JIT compilation. It supports only the new ActionScript 3 VM introduced in Flash 9.

Scaleform GFx is a commercial alternative Flash player that features full hardware acceleration using the GPU and has high conformance with both Flash 10 ActionScript 3 and Flash 8 AS2. Scaleform GFx is licensed as a game middleware solution and used by many PC and console 3D games for user interfaces, HUDs, mini games, and video playback.

rtmpdump is an open source software implementation of an RTMP client, Flash's own streaming protocol. rtmpdump was removed from Sourceforge on request by Adobe. As a result, flvstreamer was forked from rtmpdump, removing all cryptography (i.e. support for RTMPE and SWF verification).