

1

Web 2.0

This book deals with Web 2.0 topics. But then again, maybe it doesn't. You see, the problem is that no one really knows what Web 2.0 is. In this chapter, I'll present some thoughts on the matter and then you can decide what it really means to be Web 2.0. Being Web 2.0 or not might not really matter in the end. What you will learn from this book are some cool techniques to take the Web to the next level.

What Is Web 2.0?

Web 2.0 as a term was first coined during a brainstorming session at a conference on Internet technology. The breakout session was the easy part. Web 2.0 as a tangible entity has taken quite a while longer to manifest itself.

In the old days, six years ago (back in Web 1.0), Web sites were well-insulated entities that executed entirely within the browser and well within their own sphere of influence. If you wanted to build a Web site, you thought only of yourself. Users were important, but no one would dare venture so far as to suggest that you let the users tell you what they wanted, or worse still, write the requirements.

Web 2.0 challenges this line of thinking on just about every level. Web 2.0, like art, has no real definition, but you'll know it when you see it. Web 2.0 companies take a different approach to almost every aspect of their business.

Traditional Thinking

A typical approach to starting a business circa 2000 would dictate that you figure out the business you want to be in, conduct a series of brainstorming sessions, document requirements, get some funding, build out your site, and sit back and wait for the money to roll in.

Chapter 1

This “forklift” approach demanded a huge investment up front and, of course, venture capitalists were quite happy to oblige. There was also a large amount of effort before the product ever hit the market.

Many companies got their start this way, and huge amounts of venture capital were blown through before dollar one was ever earned. This continued for a number of years (the “salad days,” as they say), and then the roof caved in.

For a time, investment ceased, and the general opinion was that the Internet held no more promise as a business platform. This opinion is slowly being changed.

In recent years, the entrepreneurial spirit of the Internet has been rekindled, and smart companies are taking small, gradual steps into new and unconventional business directions. The result is what’s commonly referred to as Web 2.0.

There are many elements that contribute to this phenomenon. The remainder of this chapter describes these elements, many of which are explained in detail in subsequent chapters.

Folksonomies

What are folksonomies? The term *folksonomy* refers to the process whereby a group of people collaborate to organize information using an impromptu vocabulary. This can happen anywhere, and it can be either implicit or explicit.

A common example in the corporate world is team-building exercises, whereby a group of individuals rearrange flash cards on the floor or stickies on a white-board. By getting a large group of people’s input, you have a higher probability of getting an appropriate classification of the information in question.

How Do Folksonomies Apply to Web 2.0?

In the Web 2.0 world, there is a huge amount of information and it’s updated constantly. It would be naive to think that any one company could categorize that information so accurately that the classification would make sense to everyone. Rather, a collaborative approach makes more sense in this instance.

Who better to categorize data than the people closest to it? You’ve probably participated in folksonomies already without even realizing it.

Amazon now allows users to tag products with keywords. These are words of the customer’s choosing and can be completely arbitrary. Over time, this will evolve into its own folksonomy whereby the *users* are adding value for other users simply by using the Amazon site.

This is essentially a self-defining taxonomy, as illustrated in Figure 1-1.

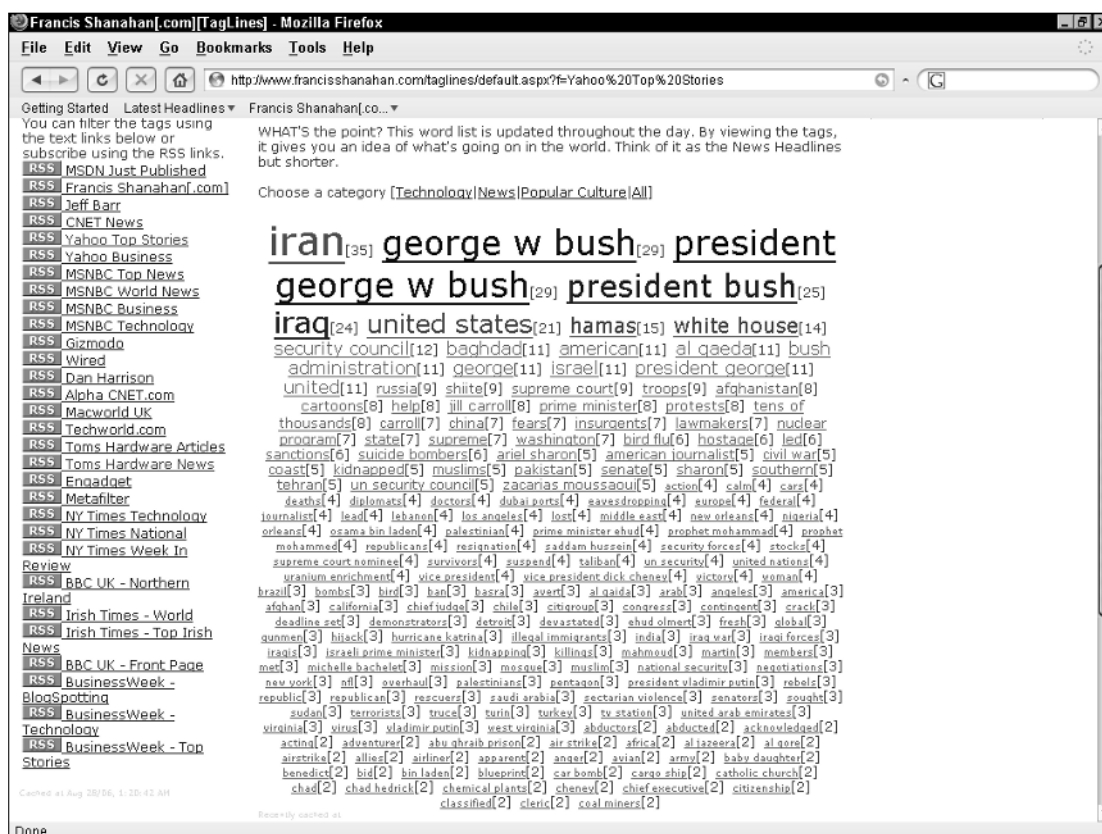


Figure 1-1: TagLines — an automated folksonomy tool

Software as a Service

The Web 1.0 paradigm consisted of a human sitting behind a browser. Sites were not meant to be machine-readable or manipulated. Web 2.0 is about exposing a rich functionality set and much more data. The data is generally accessible to both humans and machines, leading to more automation and derived applications than ever before.

In the “old days,” a company would create a service and then package it up in a Web site and publish it out on the Internet. The only way to access that functionality was through the pre-packaged mechanism implemented by the company, usually its Web site.

In the Web 2.0 world, companies are seeing more and more value in offering functionality in reusable and interoperable channels such as Web services. These channels are then handed over to the user for them to do with as they see fit. In essence, a Web 2.0 approach puts more trust in the user than ever before.

Data Is King

The mantra “content is king” has been rewritten as “data is king.” Allowing the users to consume data makes it possible to define an entirely new business model and functionality other than those that were originally intended.

Independent developers are now capable of delivering applications that would be impossible without a large team of resources. For example, developers have combined Google Maps with numerous other sources of information to build new and useful applications. It would be very difficult for an individual developer to gather satellite images of the world, or a street map of the entire United States. Yet, Google gives this data away for free.

By making the data available, the *idea* of how to apply this data is what becomes important. This is as it should be. The developer is no longer concerned with basic plumbing and is freed up to think in broader terms of overall application features.

Convergence

At present, applications are diverging from the desktop and being accessed from various devices. The next logical step will be a convergence whereby these various access channels become integrated. My personal guess is that this convergence will be centered around the television.

A personal media center is basically a television hooked up to a computer. You can view and record television without the use of tapes. You can view an enhanced programming guide with links out to Internet content. You can view RSS and news headlines on this PC viewing them as television, and so on and so on.

By integrating additional devices into the media center, new usage scenarios are enabled. For example, imagine taking a phone call by pausing live television and broadcasting the audio over your 5.1 stereo system. The television display might show caller ID, combined with a photo of the caller. Finally, the entire call could be recorded for your records, synched with your MP3 player, and published to a personal blog using RSS.

That’s just one scenario, and there will be many others. In this book, you learn the techniques necessary to implement such scenarios.

Iterative Development

Rather than use a “forklift” approach to getting functionality out the door, Web 2.0 companies tend to operate in very short cycles of design, develop, launch, get feedback, repeat. This means time-to-market is reduced. Companies purposefully leave features out to achieve shorter cycle times. Rather than guess

at what the users want, it's better to launch a small subset of functionality and then take real-world users' feedback. The feedback is then used to drive feature definition in subsequent cycles.

This constant loop of development and product releases is commonly referred to as *perpetual beta*. Rather than delivering a finished product, the application is never complete. It's constantly being iterated on and refined.

By shipping functionality early and more often, a company's projections and estimates become more accurate. The gap between users' needs and a project's requirements is lessened. The value to the users is that they get an application that doesn't attempt to solve *all* their problems, but just some (one or two) of their problems extremely well.

Rich Browser Experience

The browser is the traditional interface to the Internet. Dealing with an application in a browser (versus a rich client installed on the desktop) has typically meant reduced functionality and/or reduced productivity. Pages need time to load. Data is static in the browser until it is refreshed. Advanced controls such as "draggable" grids are not available, and so on and so on.

The arrival of faster Internet access, improved JavaScript support, and the proliferation of Web services have made a rich browser experience possible.

Asynchronous JavaScript and XML (Ajax) is usually top-of-mind whenever anyone mentions Web 2.0 or rich browser experiences. Ajax provides the ability to communicate asynchronously with a Web server while a page is being viewed in a browser. Using Ajax, you can perform partial page updates so that data is kept fresh, even if the user doesn't refresh the page.

You will implement an Ajax powered application in Chapter 10. Ajax is not the only game in town, however, when it comes to enabling a rich experience in the browser.

JavaScript Object Notation (JSON), when combined with some clever JavaScript, can also be used to dynamically update a page's content. Chapter 12 describes how this works and walks you through building a JSON-powered application.

Regardless of what rich Internet application (RIA) features a site might implement, the experience is ultimately more important than the technology.

Multiple Delivery Channels

Most applications start out as Web application-accessible via a standard browser on a desktop or laptop. With the advent of broadly available Internet access via WiFi (and, to some degree, Bluetooth) application functionality is now being delivered more and more through cellular telephones and wireless devices such as the Pocket PC or Palm.

In Chapter 14, you'll build a mobile interface into Amazon's database accessible from a BlackBerry device or a cell phone.

Social Networking

We've all been to dinner parties where you attend despite knowing only a single other person. Through that person, you get introduced to someone else, and, before you know it, you're leaving the party with a series of new business contacts.

The same thing is happening on the Web, except on the Internet, you really don't need to know anyone. Sites such as MySpace, YouTube, and Flickr (Figure 1-2) allow users to create their own personalized areas free of charge and publish content to the same. This was possible in the Web 1.0, too, but not to the same level of sophistication. Interfaces have become far more accessible, and having a personal Web site is no longer the domain of the programmer or advanced user. Literally everyone can enjoy the experience of having an online persona.

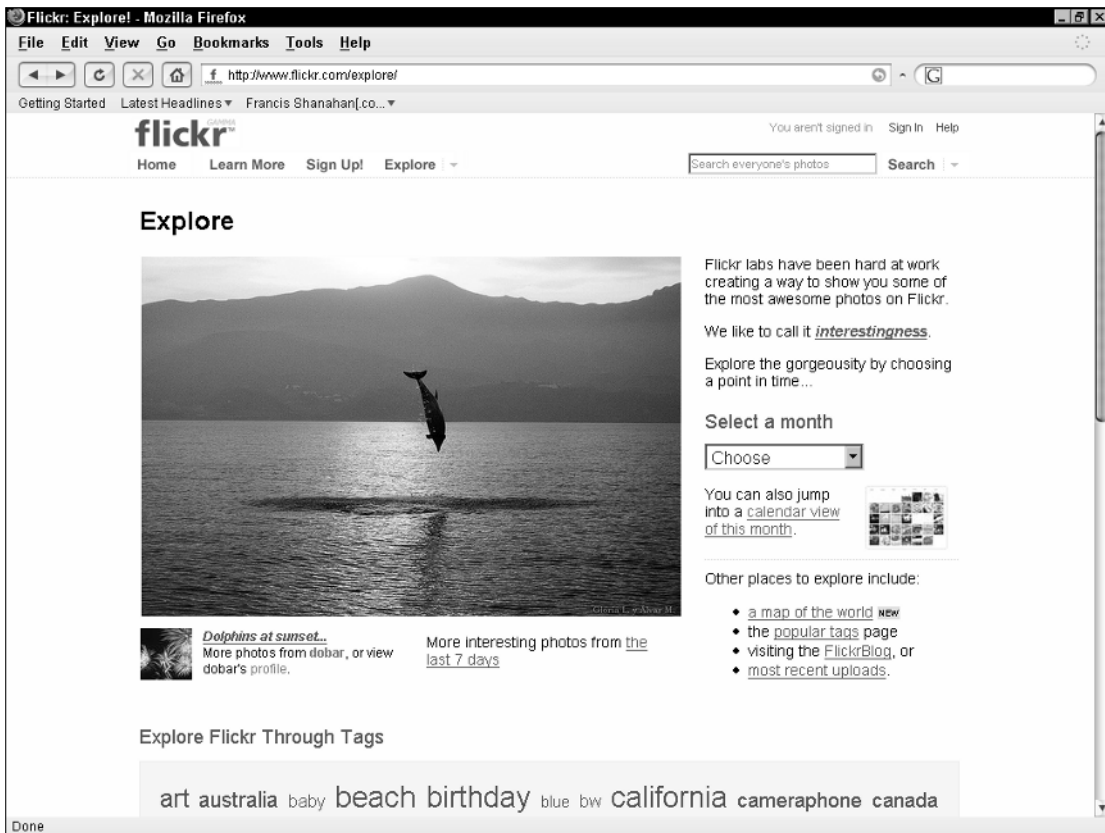


Figure 1-2: Flickr provides a forum for collaboration and social networking

Cross-pollination is also a factor. There is something fundamentally liberating about reviewing or critiquing published content. The psychological effect of publishing one's thoughts in a blog is heightened when a reader comments or links back to that blog.

Bloggers link to other bloggers, and reputations are born. Slowly, a community of real people forms around a given topic. In this manner, the leap is made from a set of technologies such as Hypertext Markup Language (HTML), Hypertext Transfer Protocol (HTTP), eXtensible Markup Language (XML), and so on, to a living, breathing community that enhances people's lives through a sense of belonging.

The Rise of the Individual Developer

Conventional thinking would indicate that to build an application of any great significance, you need a lot of people. Web 2.0 thinking exposes this as incorrect.

Web 2.0 affirms that you can develop a better application, faster, with a handful of developers who know what they are doing. Conversely, nine mothers cannot make a baby in one month.

For example, in the gaming industry, some of the most sophisticated three-dimensional games ever produced have been written soup-to-nuts by just a handful of developers. Doom started the three-dimensional first-person shooter genre, essentially launching an industry, yet it was coded by a small handful of developers.

The most popular mashups and remixes follow a similar pattern. For example, Google Maps (which kick-started the rich browser application movement) was developed by a very small company in approximately two weeks. Productivity and, consequently, business impact is so high because developers have the tools they need on hand, and because, at the end of the day, the developers know what they're doing.

All of the samples in this book have been developed by a single developer, yours truly. These applications are purposefully kept simple so that you can take them and run with them without having to understand a lot of code.

Chapter 16 provides explanations of some more complex applications. These, too, were developed by a single developer, but these applications really try to push the boundaries of what's possible in a Web 2.0 mode of thinking.

Amazon and Web 2.0

Amazon as a company has adopted the Web 2.0 line of thinking with open arms. Having weathered the dot-com bubble, it has its sights fixed firmly on the future, and is making great strides toward defining the direction of its company and, to some degree, the Internet itself.

Amazon and the Consumer

The Amazon consumer Web site offers a wide range of individual features to the customer that, when combined, provide a highly interactive experience. For this reason, Amazon customers tend to be repeat customers, even when prices are cheaper on competitor sites.

Amazon allows customers to review products, tag and categorize them, rate products, and even rate product reviews by other customers.

Over time, a customer can build a reputation as a reviewer. This promotes a sense of ownership on the site. Customers can even publish photos they've taken of products they've bought. This all serves the buyers by providing additional data points that factor into their purchasing decisions.

It doesn't stop there. An Amazon customer can build arbitrary lists of items that they recommend as a group. For example, you could define an "Xbox essential accessories" list that might include all the miscellaneous items such as game controllers, games, cables, and so on, a buyer would need in addition to the core Xbox console.

This is all explicit content contribution. But Amazon also watches its customers' actions and garners information from their patterns. If a customer looks at one product but ends up purchasing another, that's a data point that can potentially influence a buying decision for the next customer. Amazon gathers this data and builds a product ranking, along with auxiliary information that ultimately gets published along with the core product details.

Amazon and the Developer

For the developer, the Web 2.0 thinking is self-evident. Amazon has maintained a Web developers program for a number of years. Amazon provides comprehensive access to its repository of functionality and data. This access is provided free of charge, so developers have a very low barrier of entry. This attitude of embracing developers rather than restricting access is typical of the forward-thinking Web 2.0 culture.

Probably the biggest contributing factor to the success of Amazon's developer program is its ability to build a community around the services. The best services in the world would not be widely adopted were it not for a supporting community, both internal and external to Amazon. Tools are provided such as online documentation and an active developer's forum that encourages developers (both new and seasoned alike) to get their hands dirty. Amazon has recognized that the community around any Web service platform is just as important, if not more so, than the platform itself (Figure 1-3).

With the recent additions of Amazon's Mechanical Turk, Simple Storage Solution, and Elastic Computing Cloud, Amazon has widened its offering far beyond the scope of the original product database. Rather than stick to the core business model of online e-tailor, Amazon is reacting to a need and redefining its business model and business environment as it goes. Again, this adaptability is indicative of the level to which Amazon's leadership has adopted the spirit of Web 2.0.

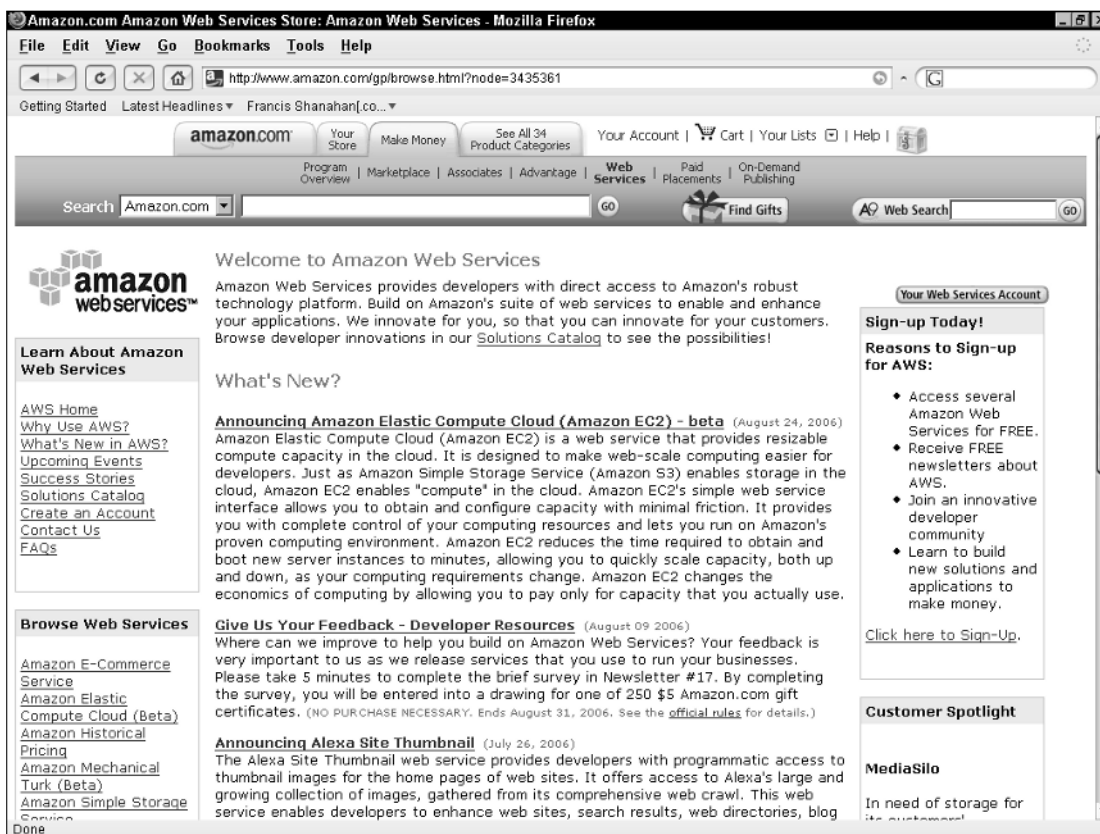


Figure 1-3: The Amazon Developer Community landing page

Summary

The Web is alive and well and more compelling than ever. I once sat in a conference room where one CEO likened the creation of the Internet to the discovery of fire. Now, even with my best propeller hat firmly in place, I think fire was a little more significant. Nevertheless, there is a wealth of opportunity on the Web for developers and business people alike.

This book teaches you everything you need to know to begin leveraging that opportunity with minimal investment.

In the upcoming chapters, you learn about the Amazon Web service platform in detail, along with other offerings from a cast of supporting characters. You learn ways to recombine these offerings into new applications.

In Chapter 2, you learn some fundamentals about how to work with the Amazon E-Commerce Service (ECS), as well as the other services available through the Amazon Developer's Platform.

