Get into Ajax

Everyone agrees that the World Wide Web is taking off. Everywhere you look, the talk is all about going online. Even cell phones are online these days. The online revolution is here, and it is permanent. No matter what ultimate shape it takes, the Web is going to be around a long, long time. People are addicted to it — online browsing, shopping, gaming — you name it, it is on the Web.

The online revolution has been a long time in the making, and many people were not convinced that it was going to last, but there is no doubt today. Many millions of users are online at any given time, and the online community is international in scope and size. With that many potential users just waiting to work with your Web site, you naturally want to give them the best experience possible. It turns out that in some areas, Web browsing leaves a lot to be desired.

Whenever you want to perform some action in a browser, such as update a table or select a new color, you must send the entire page back to the server online to which your browser is connected, and then the server sends the page back. That means that the page in front of you blinks, the figures and text in it move around as the page is rebuilt while you watch, and then the new results are presented.

Downloads Take Time

Not only does it take time to download an entire page from the Web server each time you click a button or perform some action in the page, it is also annoying. The whole display flickers and your location in the page resets back to the top. You must scroll back to where you were working with the page and find what you were doing again. As the page reloads from the server, you must watch as the page redisplays in front of you, meaning that text and images can squirm around the screen until ending up in their final positions.

Enter Ajax

Ajax allows you to fetch data from the server behind the scenes without a full-page refresh. You can communicate with the server and fetch text or call online code, all while the browser keeps displaying the current page. When you fetch the data you want from the server, you can use dynamic HTML techniques (such as writing to a `<div>` element) to update the part of the text or a table in a Web page that you want to change.

The Desktop Feel

That feels very little like an application that is loaded onto your computer, such as Microsoft Excel or Word. When you perform an action with that kind of application, you can see the results immediately in the application, and the whole screen does not flicker as the page reloads. If you type a word into Microsoft Word, for example, the word appears at the location of the cursor. The entire page does not reload each time you type a word, and the cursor does not reset to the top of the page.

These days, browsers can display many of the controls (those interactive elements you use: buttons, list boxes, scroll bars, and so on) that you see in desktop applications. However, there has been a continual problem with the feel of Web applications that display in browsers. They feel like Web applications, not desktop applications. It would be great if you could create online applications that do not need to refresh the entire display each time they need to grab some new data from the server.

Improve User Experience

Imagine how that improves the user's experience — instead of clicking a button to update the display and then watching as the entire display is reset from scratch, they see just the text or data they were working with change. That is just what happens with Ajax — you can communicate with the server behind the scenes, and the user never need know about it. Ajax represents another step along the way of making online programs look and feel like programs that run directly on the user's computer.

Origin of Ajax

Adaptive Path's Jesse James Garrett coined the term Ajax. The idea was that Web designers became envious of desktop programmers, who did not have to put up with page refreshes and extra round trips to the server to collect data. Desktop applications have a solid feel that Web applications have problems duplicating, because you have to rely on the Send button so often along with the delay that entails. However, Ajax is working to close the gap between desktop and Web applications.

What Is Ajax?

Ajax stands for Asynchronous JavaScript + XML. It is about a basic shift in the way Web applications work. Ajax itself is not a single technology, it is a group of technologies working together to bring a rich experience to the user. Ajax is made of HTML and XHTML, Cascading Style Sheets (CSS) and dynamic HTML, as well as auxiliary technologies like XML and XSLT. JavaScript holds everything together. Ajax is bringing the kind of dynamic application you see on the desktop to the Web, and not only are users the better off for it, Web programmers are as well.

How Ajax Works

Ajax works by using JavaScript and the `XMLHttpRequest` object that most modern browsers support to communicate with the server without having to refresh the Web page currently displayed. You write code in a Web page that includes JavaScript, which will fetch data from the server. That data can be in XML or text format, and it is downloaded into an object supported by the browser, an `XMLHttpRequest` object. This object not only handles the download behind the scenes, but also stores the data fetched from the server. After the data is downloaded, you can use JavaScript to retrieve that data from the `XMLHttpRequest` object and display or otherwise use that data.

Because the download happens behind the scenes, the application the user is working with does not have to come to a screeching halt while data is fetched from the server. What makes Ajax *asynchronous* is the data fetching that goes on behind the scenes without making everything stop and wait. A synchronous interact is waiting for data to be fetched before proceeding, as in a standard, non-Ajax Web application.

Ajax Beginnings

The ability to use Ajax has been available since 1998. However, Ajax did not really catch on until early 2005 when some very popular Web applications started using it, such as Google Suggest and Google Maps. Jesse James Garrett wrote an article on Ajax, bringing the term and the technology into the spotlight. Since this new release, Ajax has burst onto the popular computing stage and is being used in more and more places.

Using an Ajax-Based Application

You can see the difference in Ajax applications very clearly. As you know, Ajax's specialty is fetching data from Web servers behind the scenes. Ajax applications are based on JavaScript, and they run in Web browsers.

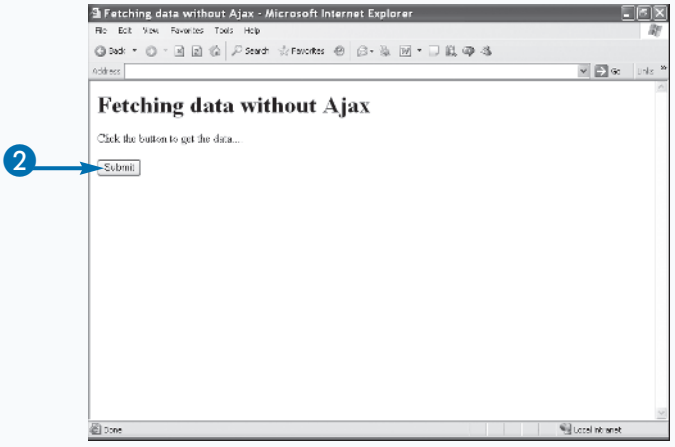
With standard Web applications, you click a button — usually a Submit button — and your data is sent to a Web server. Typically, that data is sent to a program on the Web server, and that program then creates a new Web page, which the server sends back to you. Because all the data handling is performed on the server, the page you see in the browser must be fully refreshed to display any changes. In other words, the browser just displays data; it takes no real part in handling or working with that data.

When you use Ajax, the browser becomes an active participant. That happens because you add JavaScript to your Web pages so those pages can work with the browser's `XMLHttpRequest` object to communicate with the server, download data, and display or work with that data as needed. JavaScript is a substantial language, and you can create powerful applications using it.

Ajax represents a revolution in Web design. There is nothing else like it, and it is going to become standard in Web applications. Using Ajax, you can download data behind the scenes from the Web server, before the user needs it. Then, when you want to, you can display that data as needed. This is a very useful way of operating, and it is much more in line with the desktop way of doing things — you can display data in most cases without waiting for a page refresh in the browser, which users find annoying.

Using an Ajax-Based Application

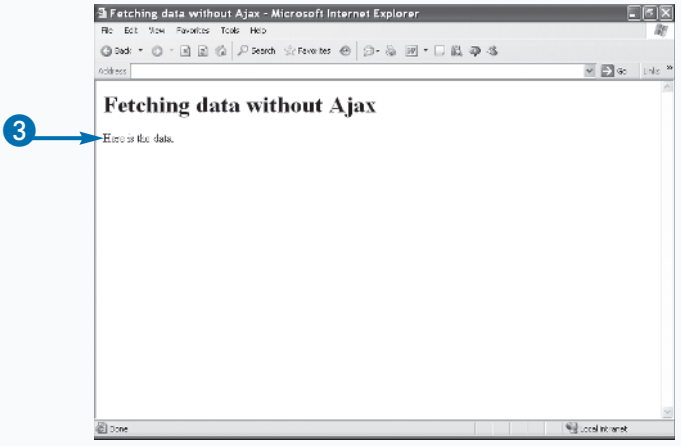
1. Open a non-Ajax Web application in your browser.
2. Click Submit to fetch data from the server.



Wait as the new page is entirely downloaded from the server.

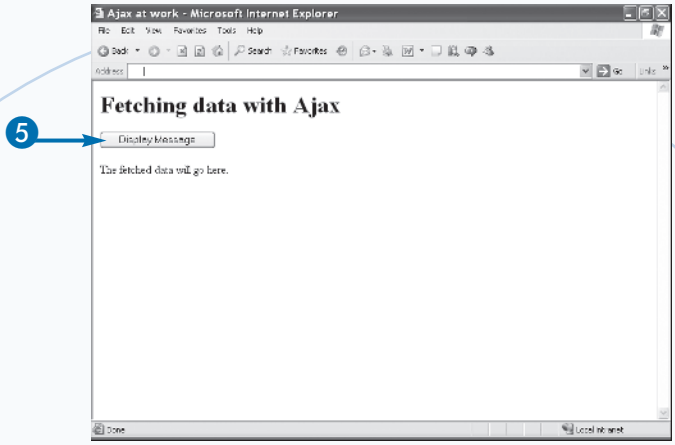
3. Read the fetched data in the newly displayed Web page.

In this example, it is the text Here is the data.

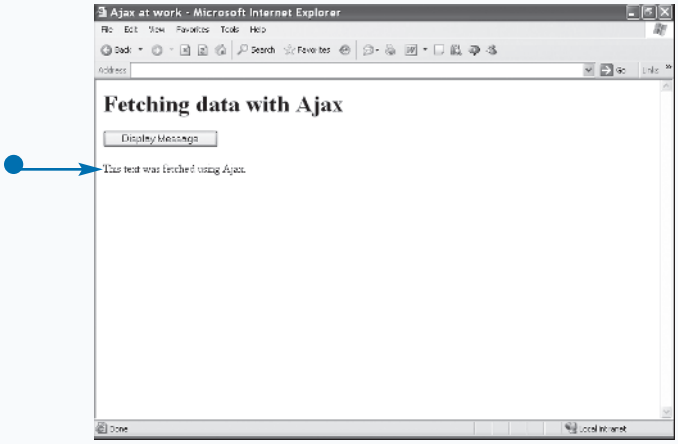


Note: The entire Web page has to be refreshed in this case.

4. Open an Ajax-enabled application in your browser.
5. Click Display Message.



- The results appear in the current Web page — with no page refresh needed.



Extra

It is important to bear in mind that Ajax relies on JavaScript to do what it does. All browsers today, such as Internet Explorer, Firefox, Mozilla, Safari, and Opera, support JavaScript. Internet Explorer is the most common browser today, but browsers based on Mozilla, including Firefox and Firebird, are gaining ground. All of these browsers already support JavaScript.

Some browsers may have JavaScript turned off. Users sometimes turn off JavaScript support in their browsers for security reasons. One of the most annoying capabilities of JavaScript is that it can be used to display pop-up ads, and many people turn off JavaScript for that reason. Sometimes, JavaScript is used to perform legal, but very annoying things, such as opening dozens of new windows in a way that effectively jams the computer and freezes it. JavaScript can even be used in illegal ways when browser bugs are exploited and hackers can grab control of the computer. In Chapter 2, you learn how to handle situations where the user has turned off JavaScript.

Run a Live Search

You can find many live-search Ajax-enabled applications online. These applications take advantage of the capabilities of Ajax to fetch data as you search for matches to specific keywords.

For example, when you perform a search on Google, you type your search term, and then click Google Search to perform your search. Your search term is sent to Google's server. The programs on that server search for matches to your search term among the billions of archived Web pages on the server. If a match is found, it is sent back to your browser. Without Ajax, however, an entirely new page appears in front of you in the browser when your search is complete.

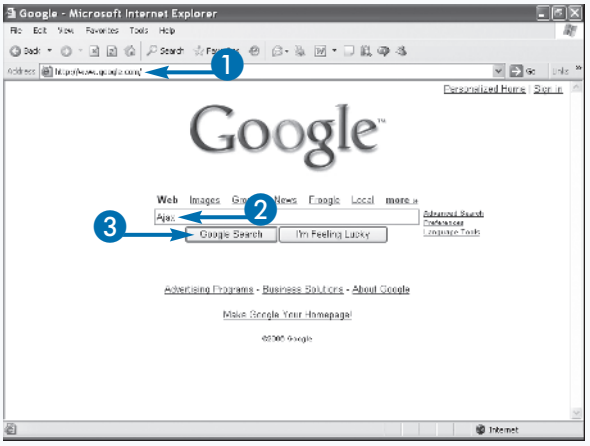
Because Ajax specializes in fetching data from the server behind the scenes, Google can use Ajax to help speed up searches. That is the purpose of a new Google page, Google Suggest. As you type your search term, the Web page can send what you have already typed to the Google server, which can then suggest matches to your term. That can save you a great deal of time — if Google has already found what you are searching for, all you have to do is click the suggested word instead of typing the full word, and Google performs the search for you. That is very handy.

That is a good indication of the power of Ajax — you can use Ajax to fetch data behind the scenes without having to wait for a page refresh. In this case, the Web page fetches data from Google without making the user wait. Users like that kind of utility because it helps speed the Web interaction.

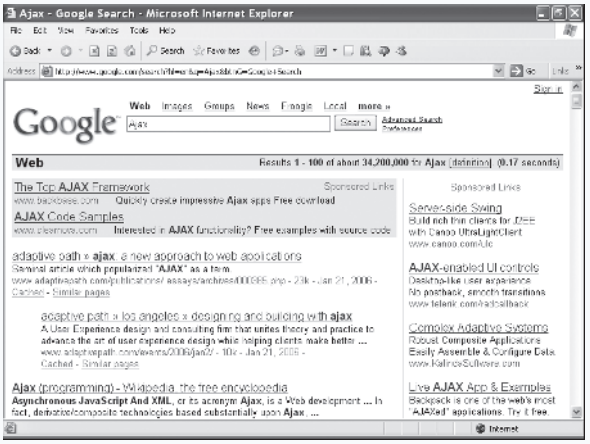
Run a Live Search

1. Navigate to the non-Ajax-enabled Google Web site, www.google.com.
2. Type a term for which to search.

   This example uses the term Ajax.
3. Click Google Search.



The pages that Google finds that match your search appear.

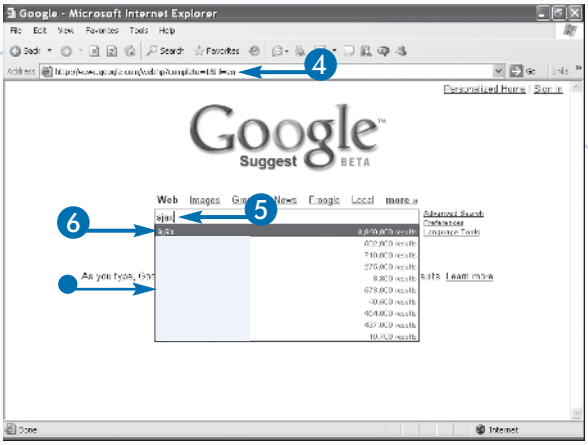


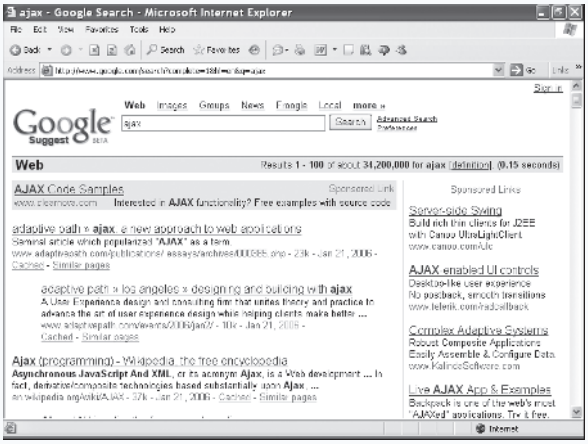
4. Navigate to the Ajax-enabled Google site at www.google.com/webhp?complete=1&hl=en.
5. Type **ajax** into the search term field.
6. Click ajax from the drop-down list that appears.

   This page uses Ajax to get those suggested search terms from the Google server.

   - The number of matches to the various selected terms appears, making it easier to select the term you want.

   Google finds pages that match your search.





Extra

Google Suggest can save you time when you type search terms simply because when you type the first few letters of your term, the whole term may appear and you can click it, saving you the trouble of typing the rest of the term. The terms that Google suggests to you are not based on your previous searches; they are based on whichever matching terms provide the most hits, which are the most frequently searched for, and so on.

You can make Google Suggest your home page if you really like it (thousands of people use the main Google page as their home page). In Internet Explorer, you can do this by selecting Tools, then Internet Options. On the Internet Options menu, look for the Home Page section, find the Address: text field, and enter www.google.com/webhp?complete=1&hl=en. Click OK. In Firefox, you can find this menu by choosing Tools➔Options. In Mozilla, you can find it by choosing Edit➔Preferences. In Opera, you can find it by choosing Edit➔Options.

Autocomplete What You Type

You can also use Ajax to *autocomplete* your data entry, which means that when you start typing a word, the autocomplete capability in an Ajax-enabled Web page can offer suggestions to complete the word. For example, if you type Aj, a Web page that supports autocomplete may suggest Ajax as the full term. Clicking Ajax in the drop-down list in which it appears replaces Aj with the full term, Ajax.

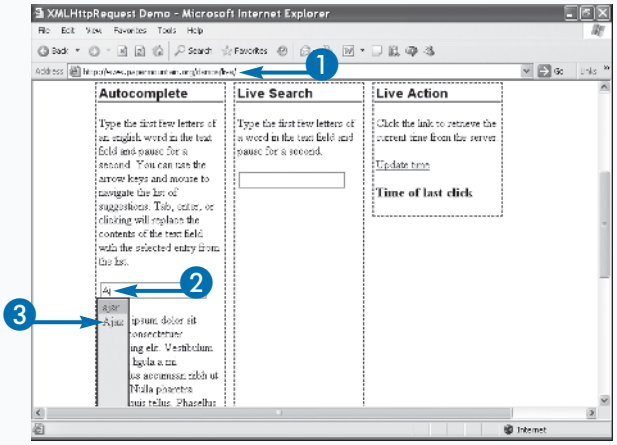
In other words, autocomplete is much like the live search you just saw in the previous task at Google Suggest. However, autocomplete is different, and the two terms are different in Ajax usage. While live searches offer suggestions specifically for search terms, autocomplete offers suggestions for any words you enter. Selecting an autocomplete term from the offered list completes the word you have been typing; it does not perform a search. That is useful for applications of all kinds, besides search applications. If you keep a store of words online, you can offer the user a list of possible completions for the word or term the user is currently typing.

You can find an Ajax autocomplete example at www.papermountain.org/demos/live/. There are three Ajax demonstrations worth looking at: an autocomplete example, a live search example, and a Live Action example. The live search example searches http://dictionary.reference.com instead of Google. The Live Action is the term www.papermountain.org uses when data is fetched using Ajax and displayed in a Web page without a page refresh. (See the section "Using an Ajax-Based Application.") You are going to see how to create all three of these Ajax applications in this book.

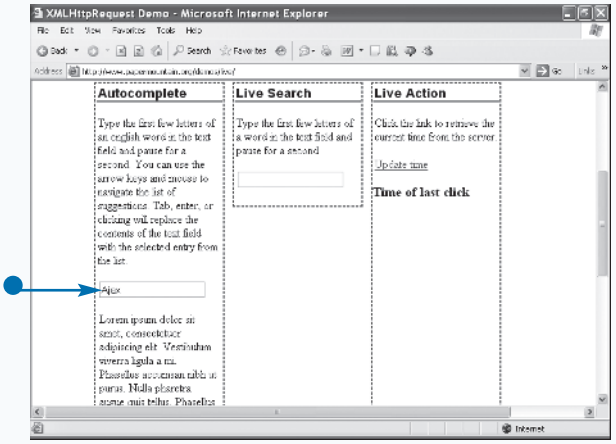
Autocomplete What You Type

1. Navigate to www.papermountain.org/demos/live/.
2. Type **Aj** in the text field in the Autocomplete section.

   The demo displays a number of autocomplete suggestions that are retrieved from the server.
3. Click the Ajax autocomplete suggestion.



- The letters you typed are replaced by the word Ajax.

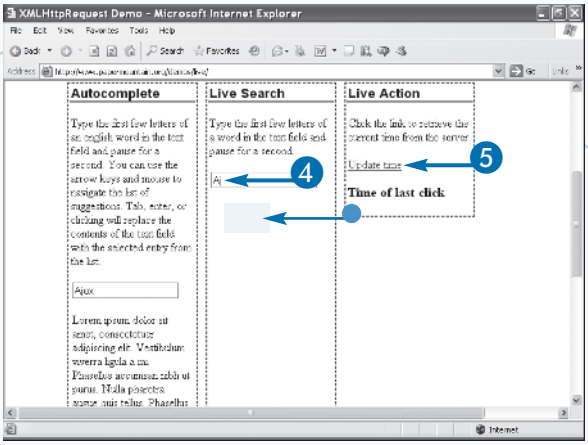


4 Type **Aj** in the Live Search section's text field.

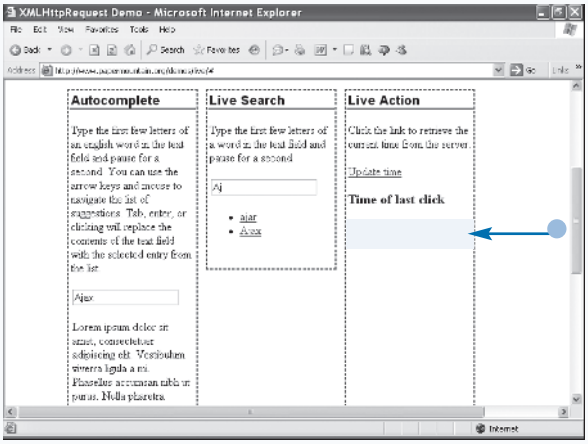
- The application displays possible search terms.

If you select a suggested search term, that term's definition from http://dictionary.reference.com opens in your browser.

5 Click the Update Time link in the Live Action section of the page.



- Using Ajax techniques, the application displays the time without refreshing the page.



Extra

The www.papermountain.org site was created by Leslie and Cassondra Hensley. Their Ajax demonstration page at www.papermountain.org/demos/live/ is all about what you can do with Ajax and the XMLHttpRequest object using JavaScript, showing how you can update Web pages without resorting to page refreshes in the browser. If you want to look at the actual JavaScript that supports this demonstration, you can find it at http://cvs.sourceforge.net/viewcvs.py/lakeshore/lakeshore/resources/liveUpdater.js?view=markup. The Hensleys say you can adapt it and use it as you like (and if you have suggestions or make improvements, you can e-mail them at hensleyl@papermountain.org). The server-side code responsible for finding search terms and offering suggestions can be found online at www.papermountain.org/demos/live/live-demos-source.html. That code is written in the Ruby language, however, which you may not be familiar with (server-side programming in this book uses a small amount of PHP). The Hensleys discuss these examples and more about Ajax at www.papermountain.org/blog/index.cgi/Tech/Javascript/PoorMansAjax.txl and www.papermountain.org/blog/index.cgi/Tech/Ruby/AutoCompleteXMLHttpRequest.txl.

Modify Web Pages without Page Refreshes

You can use Ajax to modify Web pages on the fly as the user watches. When you fetch data from the server behind the scenes and store that data, you want to display the fetched data in the Web page itself. There are many different ways of doing that, most of which involve some form of dynamic HTML or Cascading Style Sheets (CSS). These methods of displaying data are very powerful, because they let you display new data without needing a page refresh in the browser.

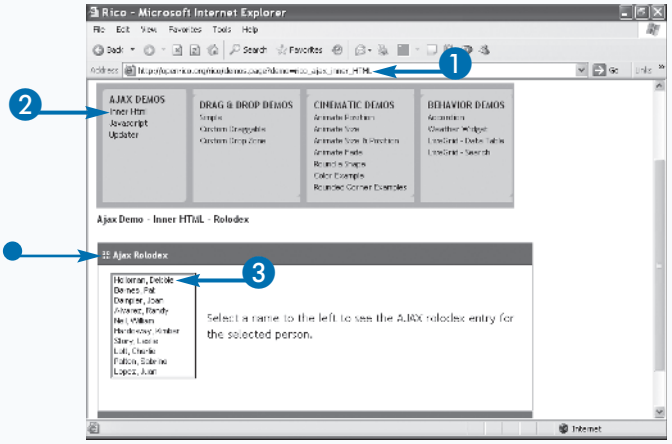
Dynamic HTML lets you update the elements in a Web page dynamically. In the early days of Web browsers, all the elements (such as the HTML elements `<h1>`, `<p>`, and so on) in the displayed pages could not be altered after they were displayed. Years ago, however, dynamic HTML became the standard, and using dynamic HTML enables you to alter the contents of any HTML element in the page. That is great when you are using Ajax, because when you grab some data from the server, you can display that data by assigning it to the contents of an element. You see this technique at work frequently in this book.

The other technique you use to display data in a Web page without a page refresh is CSS. CSS allows you to create and display pop-up windows like those used in the previous task and in the Google Suggest task to display suggestions for words from the server. That is also a very powerful technique, and it gives Web applications the feel of desktop applications.

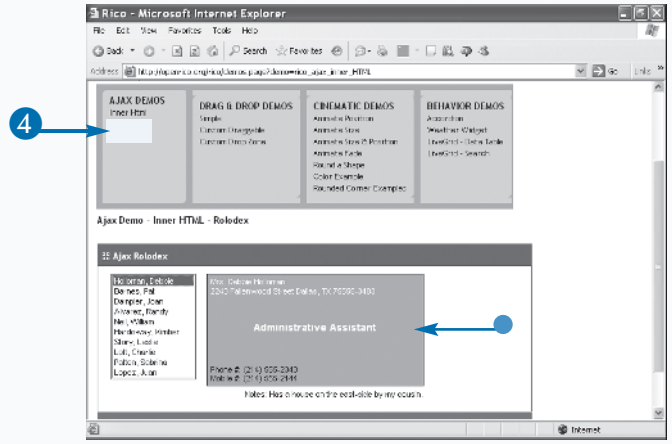
You see both dynamic HTML and CSS used with Ajax, which lets you update a page in the browser with data you get from the server. A site that uses both dynamic HTML and CSS together with Ajax in a series of demonstrations is http://openrico.org/rico/demos.page.

Modify Web Pages without Page Refreshes

1. Navigate to http://openrico.org/rico/demos.page.
2. Click the Inner Html demo link.
   - The Ajax Rolodex example opens.
3. Click Holloman, Debbie.

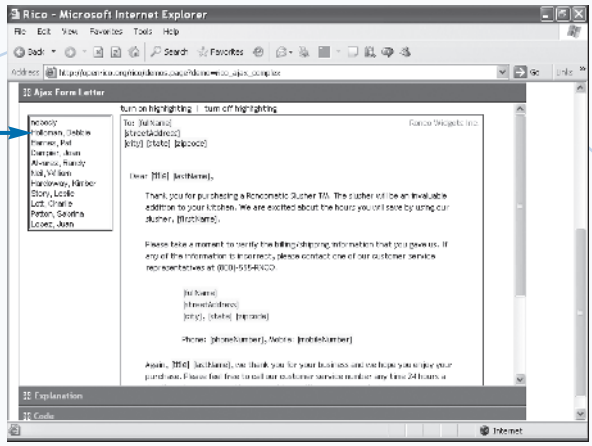


   - Using CSS, the application displays what looks like a business card in the Web page.
4. Click the JavaScript Updater link.

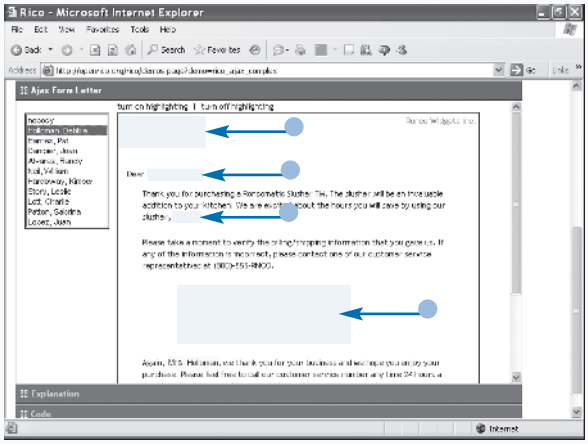


An Ajax Form Letter appears with default entries for the person's name, address, and so on.

5 Click Holloman, Debbie.



- Using dynamic HTML, the blank items in the letter are updated with data fetched from the server for the person you selected.



Extra

You can do the same kinds of things that you see in this Rico demonstration. It is easier than you might expect, because these examples are part of the ajaxEngine Ajax framework that Rico offers.

There is a considerable amount of programming involved in writing Ajax applications — and even more if you want to produce the kind of effects you see in the Rico demonstrations. Ajax frameworks are full libraries of code that handle much of the programming for you. When you use an Ajax framework, all you have to do is install it and customize the code as you want it.

Some Ajax frameworks are written in pure JavaScript, and you use them in the browser only. Those are the easiest of the frameworks to use. There are also a number of Ajax frameworks that you can use on the server that are written in a server-side language, most often PHP. The server-side frameworks actually write JavaScript code to do what you want and send that code to the browser. You see both kinds of Ajax frameworks in this book.

Drag and Drop with Ajax

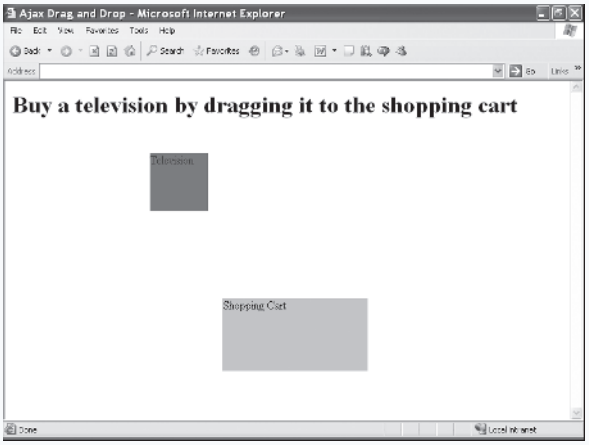
You can use Ajax to simplify hundreds of standard tasks that users perform every day online. For example, consider the process of buying something online and then checking out. The shopping process involves moving from page to page. You look at an item on one Web page, and then move to another item's page, all of which involves a great deal of navigation using the browser, which means many page refreshes. The screen is continually being refreshed.

Finally, when you select an item, the shopping cart page appears, showing the item you added to the cart. The check-out process also involves three or more page refreshes. First, you open the checkout page, then you move to a page that gets your personal data, then another page pops up that asks about shipping, and then you get a confirmation page. This process can become very tiresome to the user, who is tired of watching the browser window clear and then having to wait for the elements in the page to reappear.

The whole process involves many page refreshes, and that is one of the unpleasant aspects of shopping online — you continually see pages flash and flicker. Ajax specializes in avoiding page refreshes, so it is a natural technology to use here. Every aspect of the online shopping experience can be improved with Ajax — for example, instead of moving from page to page to see new items, the new items can just appear in the current page. When you want to check out, everything can be handled on the same page. Even adding items to the shopping cart benefits from using Ajax. Instead of clicking Add to Cart and then viewing the shopping cart page, you can simply drag an item onto the shopping cart icon.

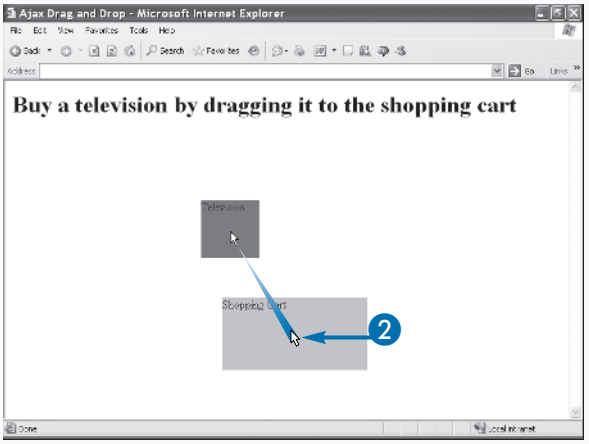
Drag and Drop with Ajax

1. Navigate to a shopping page where you see an item that you want to buy.



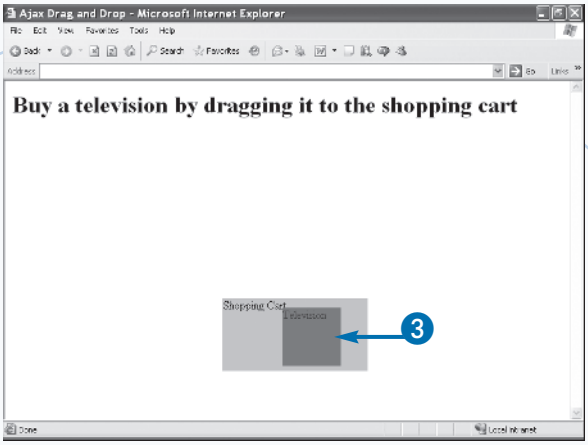
2. Click and drag the television to the shopping cart.

   You no longer have to use a separate shopping cart page to add the item to the shopping cart.



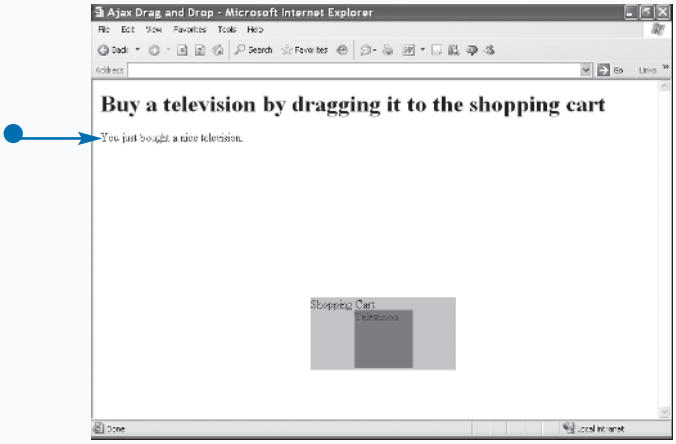
3 Drop the television into the shopping cart and release the mouse.

The Ajax application contacts the server behind the scenes to inform the server that you added an item to the cart.



- The server sends confirmation, which appears in the upper-left corner of the Web page.

You added an item to the shopping cart without leaving the current shopping page.



Extra

This way of using Ajax can revolutionize processes like online shopping. Using Ajax can make the entire online shopping experience just like a desktop application, because it appears that you never left the current page. Not only can the process of adding items to a shopping cart work as in this example, but it is particularly important to realize that the check-out process can be simplified as well. Surveys have shown that users find the check-out process, which involves moving between at least three different Web pages, particularly annoying. After you type your credit card number, for example, and you wait for the next page, there is always that thought that the server will lose your information and the transaction may fail, which is the kind of thing that makes the online shopping experience feel so fragile and error-prone. When everything takes place on the same page, however, it makes the experience feel much more solid and trustworthy.

Get Instant Login Feedback

You can also use Ajax to make the online login process much easier. Logging in to a site is another one of those online tasks that can benefit from using Ajax to avoid page refreshes.

For example, say that you are dealing with a Web site to which you must supply a username and password before you can gain entry. Suppose that you misspell your username. You are directed to a new page that explains the problem with a message something like, "No such user." You probably must click the Back button in your browser, go back to the login page, and retype both your username and password. Once again, you face several screen refreshes to get through the process, and that can be very annoying as well as time-consuming — which is something you would like to avoid if possible.

With Ajax, it is a different story. You type your username and password into an Ajax-enabled login page. Using Ajax techniques, the login page sends the username and password to the server without making you go to a new page. If your username and password are correct, you are logged in without having to leave the current page.

On the other hand, if you make a mistake in the login process, you get immediate notification and can retype your username or password — all without having to leave the login page. Using Ajax, the login page sends your username and password to the server. If they are incorrect, you are immediately notified. You can correct what you typed at once, without having to get a page refresh that takes you to a login error page. You can see an example at www.jamesdam.com/ajax_login/login.html.

Getting Instant Login Feedback

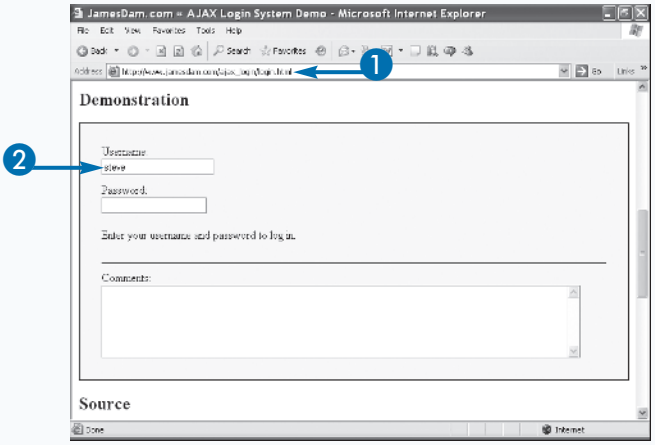
1. Navigate to www.jamesdam.com/ajax_login/login.html.

   The login page appears.

   A correct username is user1, and the matching password is pass1.

2. Type an incorrect name in the Username text field.

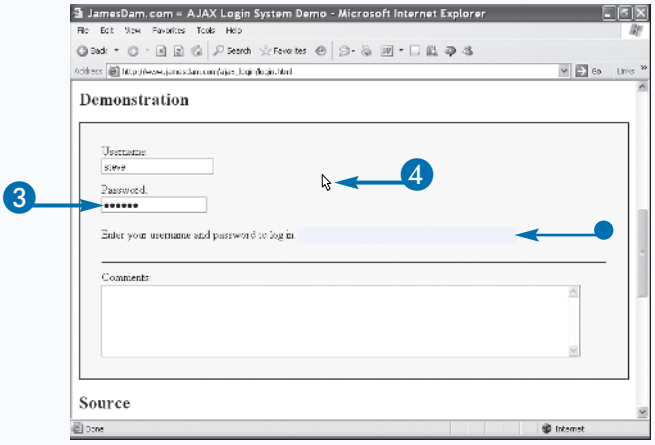
   This example uses steve.



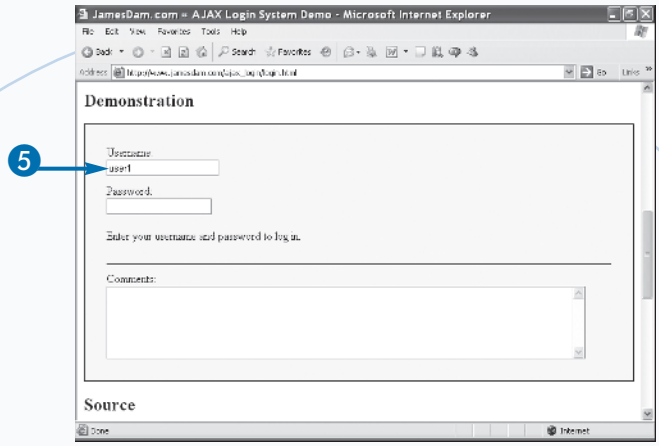
3. Type an incorrect password in the Password field.

4. Click anywhere outside the Username or Password text fields.

   - The login page displays the text Invalid username and password combination.

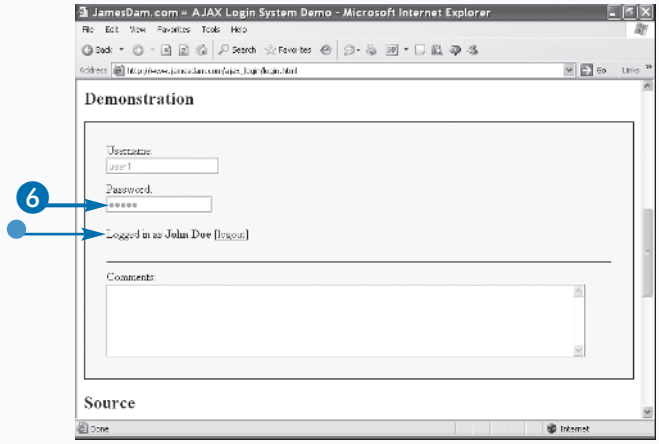


5 Type **user1**, the correct username, in the Username text field.



6 Type **pass1**, the correct password, in the Password text field.

- The login page indicates that you are logged in.



Extra

One of the problems with an Ajax-enabled login page is that you must make sure that you send username and password information to the server in a secure way. Do not send that information freely over the Internet without any security, because it can be easily read. To create an Ajax-enabled login page, provide some password security.

The login page in this example, www.jamesdam.com/ajax_login/login.html, uses the MD5 encryption algorithm to send its username and password data to the server. There are links in the page at www.jamesdam.com/ajax_login/login.html that show you the code on how to do this. On the other hand, as the text in this page itself notes, the MD5 encryption algorithm is not secure, which means it is not the best way to send username and password information. If you want to make sure your data is sent securely, use a strong encryption algorithm such as SHA-1.

Create Rich Displays with Ajax

You can create rich displays with Ajax. When the server returns information, you can use it to add elements to the current data you display to the user. For example, your site may display a generic map of an area so that users can orient themselves. Users may be interested in a particular location on your map. You can simply leave the map as it is and assume users can figure out where the location is.

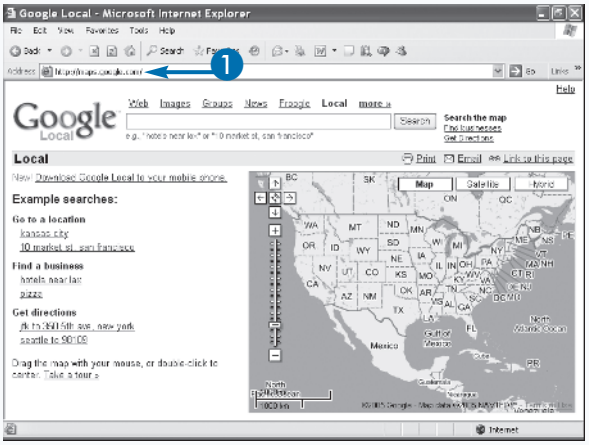
On the other hand, you can use Ajax to make it easier to find a specific location. When users ask about a particular location, for example, your page can send the request to the server using Ajax techniques. Your page can use those techniques to retrieve information from the server to help the user find the exact location on the map.

In other words, you display a map of an area, and you know the location in the map that the user is interested in (the server may send you the map x and y coordinates, in pixels, of the location that the user is inquiring about). Using Cascading Style Sheet (CSS) techniques, you can move a marker, such as an image of an arrow or a pushpin, to mark the exact location in the map in which the user is interested.

Such an application already exists at Google Maps, which you can find at http://maps.google.com/. This application uses Ajax to position markers and text to point out the location on a map the user is interested in. As you would expect, you do not need a page refresh in order to make it work; all you have to do is use it, and Ajax will handle all the details for you.

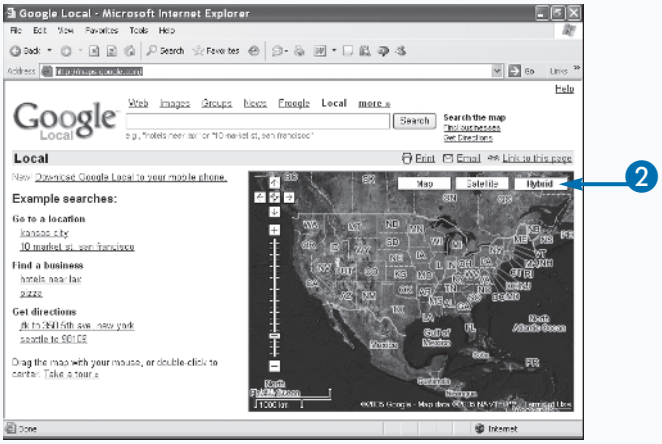
Create Rich Displays with Ajax

1. Navigate to Google Maps at http://maps.google.com/.



2. Click Hybrid.

   Google Maps displays not just maps, but also actual satellite images of the area as the background.



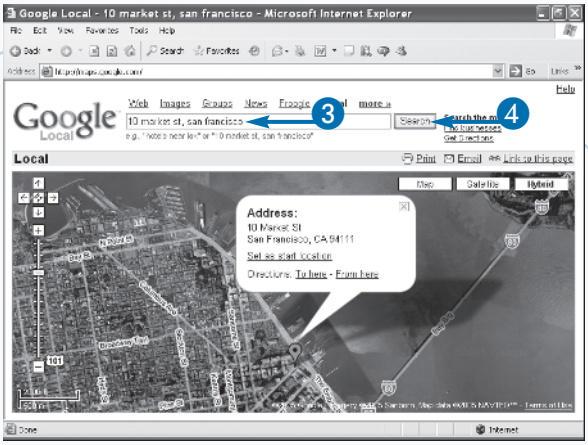
3. Type a location you are interested in into the search box.

   In this example, 10 Market St, San Francisco is used.

4. Click Search.

   Google Maps displays the location.

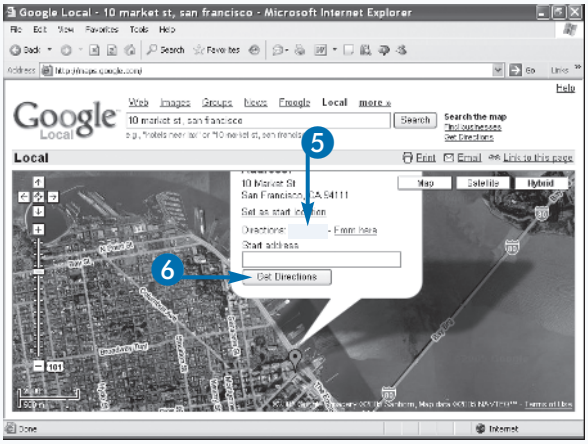
   The position of the marker and the text information superimposed on the map are fetched using Ajax, and the page is not refreshed.



5. Click the To here link in the text bubble if you want more directions, and type the location you are starting from.

6. Click Get Directions.

   Google Maps displays directions from your current location to the location pinpointed on the map.



Extra

Getting directions from Google Maps also involves using Ajax. Typing the location you want to start from in the text field in the text bubble and clicking Get Directions sends the information to the Google servers using Ajax — the page is not refreshed.

Google calculates how you can travel from the location you type to the location pinpointed on the map by using its large database of streets and distances. The server then sends that information back to the Web page in your browser, which displays the driving directions using a panel of text inside the Web page. In that way, Google Maps uses Ajax again to avoid making the user deal with page refreshes where the map would jump around in the browser view.

Look at this application at http://maps.google.com/. It is a very good introduction to using Ajax programming techniques.

Create Games with Ajax

You can create online games with Ajax. Online games that interact with the server are easy to create with Ajax. If you wanted to create an online game, your choices were limited to creating your own software that communicates over the Internet, because browsers were not up to the task.

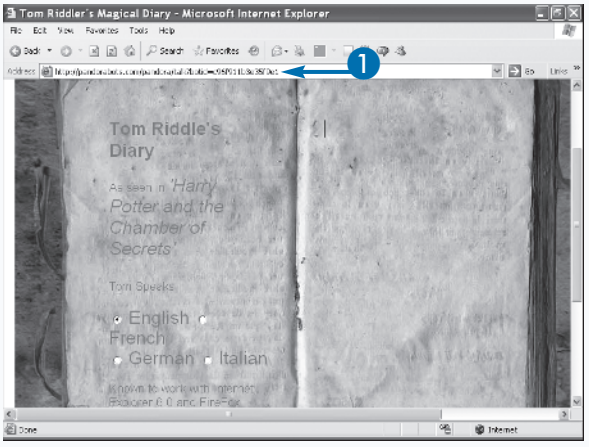
In the old days, if you relied on using a browser to create a game, you ran up against the same problem discussed throughout this chapter — page refreshes. When you made a move, the entire page refreshed as your new move was sent to the server, and the server sent back a whole new page. That was bad because the entire page jumped and reset its position back to the top left, which meant that you had to scroll down to your earlier position to continue the game. Rapidly interactive games, as when you had to avoid rocket ships or torpedoes, were impossible because of those page refreshes. Slower games, like chess, were barely possible. Creating interactive graphical games required that you create your own software, which the user had to download and install in order to run the entire game using that software (no Internet connection necessary), or use the software to communicate with the server itself.

Ajax has improved the situation dramatically. You can use dynamic HTML and CSS to move elements around in the browser window, and you can use Ajax to communicate with the server to find out what should be happening in the game. One such example is at http://pandorabots.com/pandora/talk?botid=c96f911b3e35f9e1, where Tom Riddle's diary reads what you write and answers you — all without a page refresh.

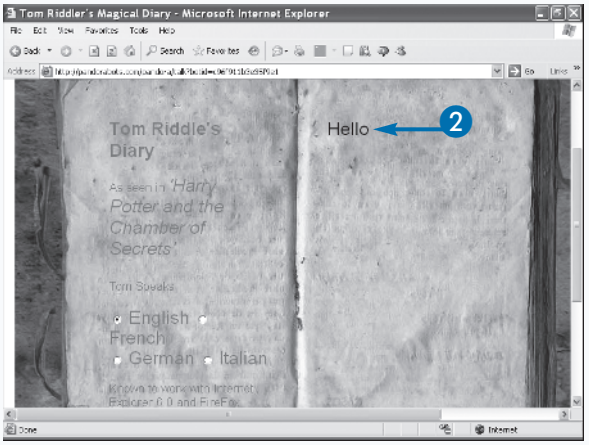
Create Games with Ajax

1. Navigate to Tom Riddle's diary at http://pandorabots.com/pandora/talk?botid=c96f911b3e35f9e1.

   A cursor appears, waiting for you to type something.

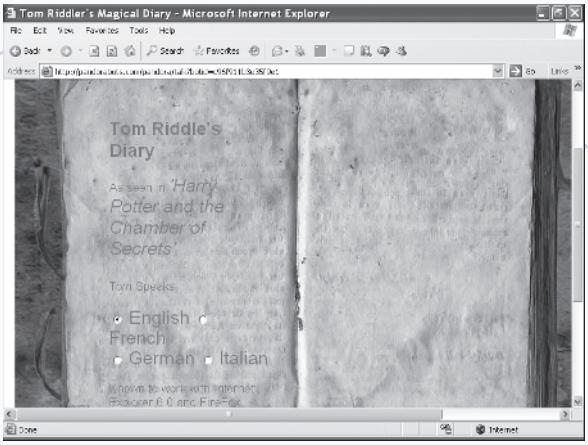


2. Type **Hello** at the cursor.

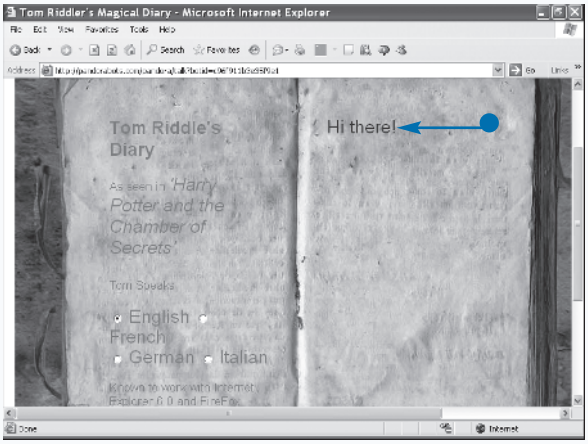


Tom Riddle's diary reads what you write and deletes it.

The cursor also disappears because the diary is not asking for input at this time.



- The diary answers with a response that it types as you watch.



Extra

Tom Riddle's diary is an impressive game, and its answers are usually surprisingly accurate. It is a good example of a game that would be difficult to write purely in JavaScript because the database for the answers is so large that to download it into the browser would be difficult. Moreover, you can communicate with the diary in English, German, French, or Italian. Imagine downloading all that into a browser. That points out one of the useful aspects of Ajax games online: You can store a huge database of game moves on the server and access that database using Ajax. That is useful when your game needs such a database, as when you have a chess game that relies on a library of standard opening moves from the great games of the past.

There are also other Ajax games available today. For example, The Fonz and Ajax at http://mrspeaker.webeisteddfod.com/2005/04/17/the-fonz-and-ajax/ is a word game that lets you move around using typed commands. In addition, there is a Monopoly-like game available at http://llor.nu/ as well.

Chat Online Using Ajax

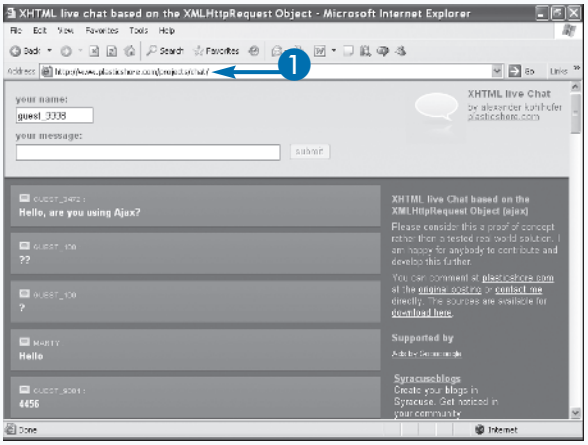
You can use Ajax's ability to let browsers display and update Web pages without page refreshes in many different kinds of applications. Any online application where you need to communicate with the server behind the scenes is a good candidate for Ajax. Here is another example: online chat applications. Such applications let you type text that is seen by everyone who is logged on at the time. Whatever you type can be seen by others; whatever others type can be seen by you.

The reason this is a good candidate for Ajax techniques is that you need to communicate with the server in order to download what other people are typing and to upload what you have typed. This is how it works: You type your message and click a button to upload that text to the server. The server displays your text in the chat application's Web page. When others view that page in their browsers, it automatically updates itself every few seconds using Ajax so that they can see what other people have been typing. Ajax is perfect here, because you need to communicate with the server while avoiding screen refreshes.

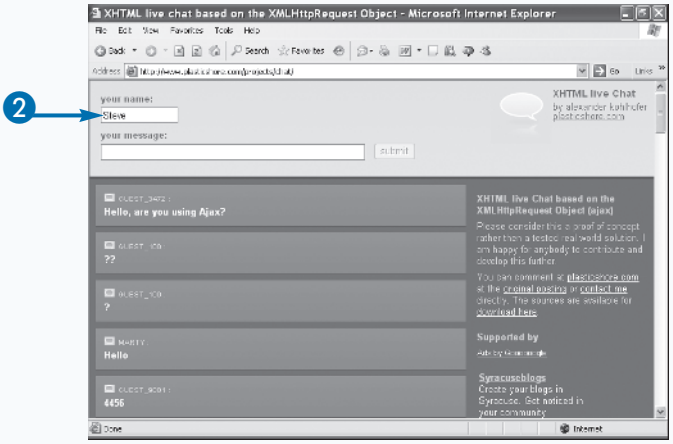
There is an example showing how an Ajax chat works at www.plasticshore.com/projects/chat/. To use this chat application, type your name and your comments, and click Submit. Your comments are displayed in the main part of the page. Anyone else viewing the page can also see your comments and can reply to you with their own answers to your comments. This is a perfect example of Ajax at work. You do not need any screen refreshes to execute this example, just pure Ajax code that automatically updates the screen.

Chat Online Using Ajax

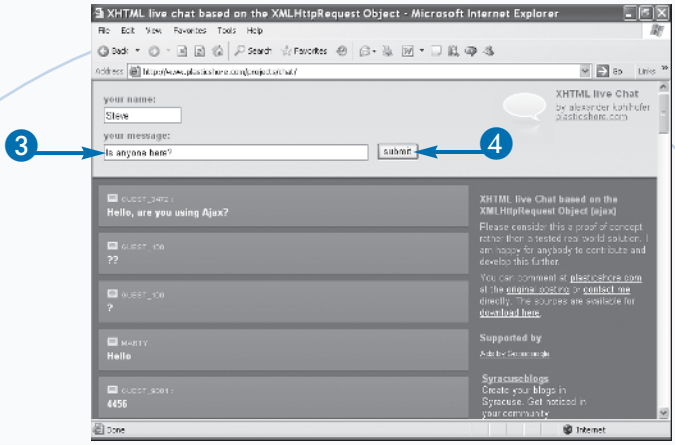
1. Navigate to www.plasticshore.com/projects/chat/.



2. Type your name in the your name text field.

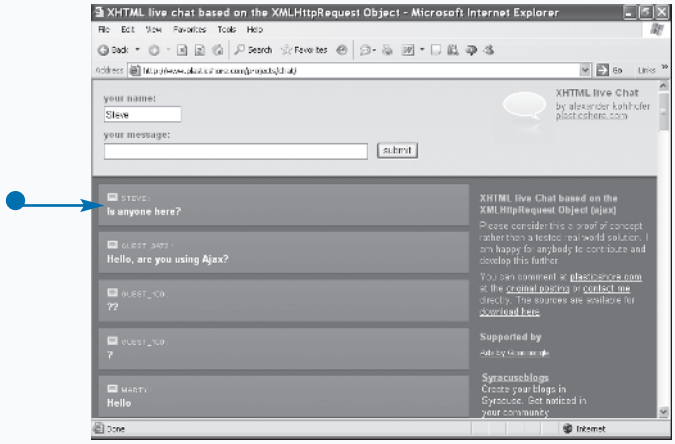


3. Type your comments in your message text field.
4. Click Submit.



- Your comments appear in the chat application.

You and everyone else viewing that page can see the comments.



Extra

This kind of instant communication using Ajax is a very exciting technology that allows you to write tools such as instant messenger-type programs using nothing more than Web browsers. There are all kinds of possibilities here. Using this kind of technology can enable a person to communicate instantly with someone who has posted something in a Web page. For example, if someone publishes a blog (a Web log of commentary), he or she might include a comment button that allows you to communicate with him or her directly. Alternatively, if someone lists an item in an online auction and you have questions about the item (such as the whether a DVD player supports both 4:3 and 16:9 screen ratios), you could contact that person immediately, and get a response in the same Web page. You can also use this technique to broadcast live events, where everyone looking at your Web page can see it instantly updated, without page refreshes, with the latest news of a sporting event, for example.

Download Images with Ajax

You can also use Ajax to download images from the server and display them. You can also use Ajax to download text or XML from the server.

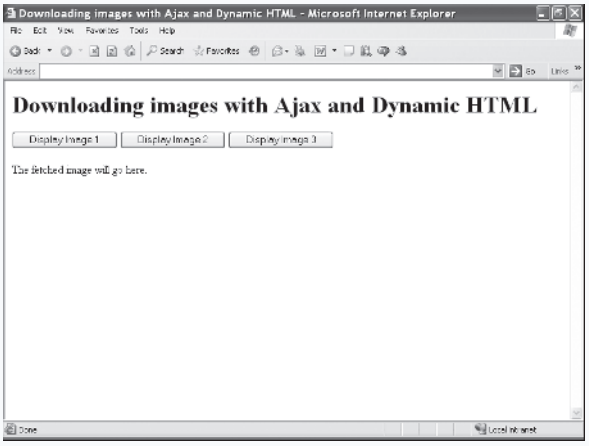
You cannot download images directly using Ajax, but if you are clever, you can see how to use Ajax together with dynamic HTML to download images without refreshing the Web page. Normally, when you display an image in a Web page, you use an `<img>` element this way: `<img src="image1.jpg">`.

If the server wants you to display a different image to match some new data, you can download the name (or URL if that new image is not in the same directory as the Web page itself on the server) of the new image file using Ajax — say it is image2.jpg. Now that you have the name of the new image file you can display it using dynamic HTML by rewriting the contents of the `<img>` element so that the `src` HTML attribute, which gives the name of the image you want to display, is assigned the name of the new image like this: `<img src="image2.jpg">`.

When the browser sees that the image it is supposed to display has been changed, it downloads the new image and displays it. You just used Ajax to download the name of the new image, or its URL, to display and relied on dynamic HTML to make the browser display that new image. The effect is the same. You have been able to fetch those images behind the scenes without a page refresh, and it looks as if you have used Ajax to download images.

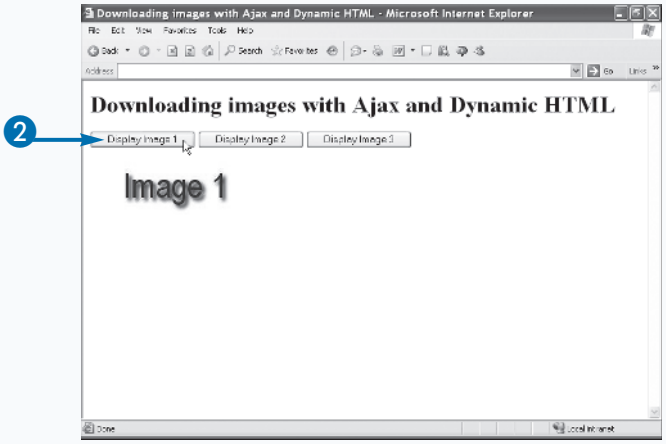
Download Images with Ajax

1. Open a Web page that lets you swap images.



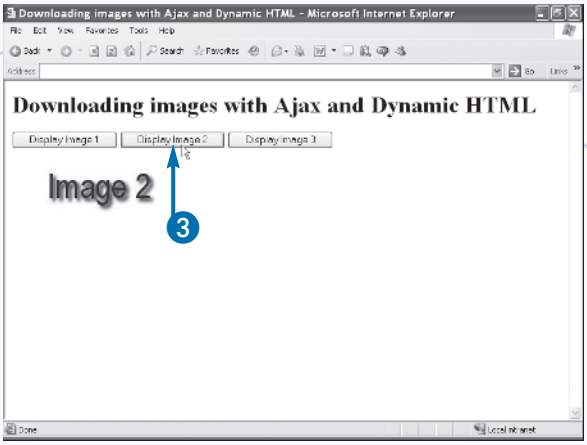
2. Click Display Image 1.

   Image 1 is downloaded and displayed using a combination of Ajax and dynamic HTML.



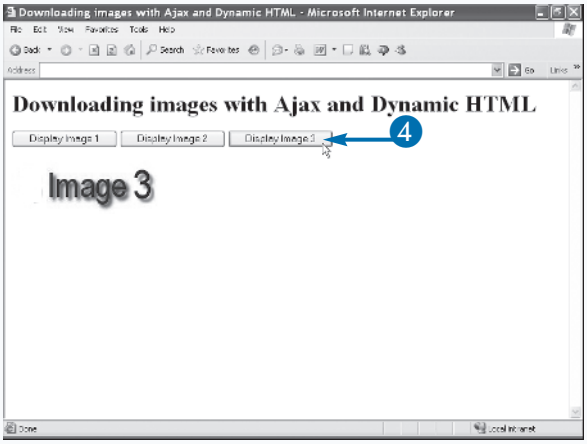
3 Click Display Image 2.

Image 2 is downloaded and displayed.



4 Click Display Image 3.

Image 3 is downloaded and displayed.



Extra

This technique of using the browser to do the downloading is an interesting one. While Ajax itself is restricted to downloading text or XML data, as you see in this example, you can use Ajax to force the browser to download image data that is in binary format, not text format. You can extend this technique to any kind of binary file, not just images, as long as the browser knows what to do with such files. For example, you can download podcast MP3 files and have the browser play them (or automatically launch software that can play them). You can do the same with video files like AVI or MPEG, and the browser can play them (or launch software that can play them). The same goes for other binary formats, such as Adobe PDF files. Overall, this is a very handy technique to remember when working with Ajax if you want to work with other data formats besides text or XML.