Introduction

Overview

A tremendous amount of information is available through the Internet: today's news, the location of an expected package, the score of last night's game, or the current stock price of your company. Open your favorite browser, and all of this information is only a mouse click away. Nearly any piece of current information can be found online; you have only to discover it.

Most of the information content of the Internet is both produced and consumed by human users. As a result, web pages are generally structured to be inviting to human visitors. But is this the only use for the Web? Are human users the only visitors a website is likely to accommodate?

Actually, a whole new class of web user is developing. These users are computer programs that have the ability to access the Web in much the same way as a human user with a browser does. There are many names for these kinds of programs, and these names reflect many of the specialized tasks assigned to them. *Spiders, bots, aggregators, agents,* and *intelligent agents* are all common terms for web-savvy computer programs. As you read through this book, we will examine how to create each of these Internet programs. We will examine the differences between them as well as see what the benefits for each are. Figure I.1 shows the hierarchy of these programs.

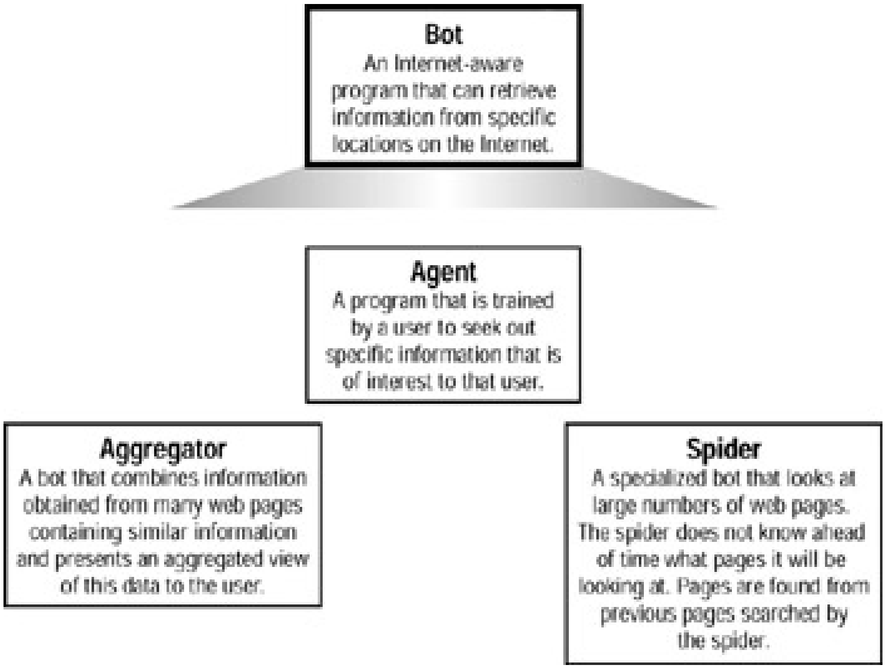
Bot
An Internet-aware program that can retrieve information from specific locations on the Internet.
Agent
A program that is trained by a user to seek out specific information that is of interest to that user.
Aggregator
A bot that combines information obtained from many web pages containing similar information and presents an aggregated view of this data to the user.
Spider
A specialized bot that looks at large numbers of web pages. The spider does not know ahead of time what pages it will be looking at. Pages are found from previous pages searched by the spider.

Figure I.1: Bots, spiders, aggregators, and agents

What Is a Bot?

Bots are the simplest form of Internet-aware programs, and they derive their name from the term *robot*. A robot is a device that can carry out repetitive tasks. A software-based robot, or bot, works in the same way. Much like a robot on an assembly line that will weld the same fitting over and over, a bot is often programmed to perform the same task repetitively.

Any program that can reach out to the Internet and pull back data can be called a bot; spiders, agents, aggregators, and intelligent agents are all specialized bots. In some ways, bots are similar to the macros computer programs, such as Microsoft Word, give users the ability to record. These macros allow the user to replay a sequence of commands to accomplish common repetitive tasks. A bot is essentially nothing more than a macro that was designed to retrieve one or more web pages and extract relevant information from them.

Many examples of bots are used on the Internet. For instance, *search engines* will often use bots to check their lists of sites and remove sites that no longer exist. Financial software will go out and retrieve balances and stock quotes. Desktop utilities will check Hotmail or Yahoo! Mail accounts and display an icon when the user has mail.

In the February 2001 issue of *Windows Developer's Journal,* I published a very simple library that could be used to build bots. I received numerous letters from readers telling me of the interesting uses they had found for my bot foundation. One such use caught my eye: A father wanted to buy a very popular and recently released video game console for his son's birthday. As part of a promotion, the manufacturer would place several of these game consoles into public Internet auction sites as single bid items. The first person that saw the posting got the game console. The father wrote a bot, based on my published code, that would troll the auction site waiting for new consoles. The instant the bot saw a new game console for sale, it would spring into action and secure his bid. The plan worked and his son got a game console. The father was so delighted he wrote to tell me of his unique use for my bot. I was even invited to stop by for a game if I was ever in Maryland.

This story brings up an important topic that arises when you are working with bots. Is it legal to use them? You will find that some sites may take specific steps to curtail bot usage, for example, some stock quote sites will not display the data if they detect a bot. Other sites may specifically forbid the use of bots in their terms of service or licensing agreement. Some sites may even use both of these methods, in case a bot programmer ignores the terms of service. But, for the most part, sites that do not allow bot access are in the minority. The ethical and legal usage of bots is discussed in more detail in Chapter 12, "Using Bots Conscientiously."

Warning

As the author of a spider, bot, or aggregator, you must ensure that it is legal to obtain the data that your bot seeks, and if you are still in doubt after conducting such a study, you should ask the site owner or an attorney.

What Is a Spider?

Spiders derive their name from their insect counterparts: spiders spin and then travel large complex webs, moving from one strand to another. Much like the insect spider, a computerized spider moves from one part of the World Wide Web to another.

A spider is a specialized bot that is designed to seek out other sites based on the content found in a known site. A spider works by starting at a single web page (or sometimes several). This web page is then scanned for references to other pages. The spider then visits those web pages and repeats the process, continuing it indefinitely. The spider will not stop until it has exhausted its supply of new references to additional web pages. The reason this process is not infinite is because a spider is typically given a specific site to which it should constrain its search. Without such a constraint, it is unlikely that the spider would ever complete its task. A spider not constrained to one site would not stop until it had visited every site on the World Wide Web.

The Internet search engine represents the earliest use of a spider. Search engines enable the user to enter several keywords to specify a website search. To facilitate this search, the search engine must travel from site to site trying to match the keywords. Some of the earliest search engines would actually traverse the Web while the user waited, but this quickly became impractical because there are simply too many websites to visit. Because of this, large databases are kept to cross-reference websites to keywords. Search engine companies, such as Google, use spiders to traverse the Web in order to build and maintain these large databases.

Another common use for spiders is *website mapping*. A spider can scan the homepage of a website, and from that page, it can scan the site and get a list of all files that the site uses. Having a spider traverse your own website may also be helpful because such an exploration can reveal information about its structure. For instance, the spider can scan for broken links or even track spelling errors.

What Are Agents and Intelligent Agents?

Merriam-Webster's Collegiate Dictionary defines an agent as "a person acting or doing business for another." For example, a literary agent is someone who handles many of the business transactions with publishers on behalf of an author. Similarly, a computerized agent can access websites and handle business for a particular user, such as an agent selling an investment position in response to some other event. Other more common uses for agents include "computerized research assistants." Such an agent knows the types of news stories that its master is interested in. As stories that meet these interests cross the wire, the agent can clip them for its master.

Agents have a tremendous amount of potential, yet they have not achieved widespread use. This is because in order to create truly powerful and generalized agents, you must have a level of artificial intelligence (AI) programming that is not currently available.

There is a distinction between an intelligent agent and a regular agent. A *nonintelligent agent* is nothing more than a bot that is preprogrammed with information unique to its master user. Most news-clipping agents are nonintelligent agents, and they work in this way: their master user programs them with a series of keywords and the news source they are to scan.

An *intelligent agent* is a bot that is programmed to use AI to more easily adapt to the needs of its master user. If such an agent is used to clip articles, the master user can train the agent by letting it know which articles were useful and which were not. Using AI *pattern recognition* algorithms, the agent can then attempt to recognize future articles that are closer to what the master user desires.

Note	This book specifically deals with spiders, bots, and aggregators—the bots that deal directly with web pages. Intelligent agents are programs that can make decisions based on a user's training, and therefore they are more of an AI topic than a web programming topic. Because

this book deals mainly with the types of bots directly tied to web browsing, intelligent agents will not be covered.

What Are Aggregators?

Aggregation is the process of creating a compound object from several smaller ones. Computerized aggregation does the same thing. Internet users often have several similar accounts. For instance, the average user may have several bank accounts, frequent flyer plans, and 401k plans. All of these accounts are likely held with different institutions, and each is also secured with different user ID/password information.

Aggregators allow the user to view all of this information in one concise statement. An *aggregator* is a bot that is designed to log into several user accounts and retrieve similar information. In general, the distinction between a bot and an aggregator can be understood by the following example: if a program were designed to go out and retrieve one specific bank account, it would be considered a bot; if the same program were extended to retrieve account information from several bank accounts, this program would be considered an aggregator.

Many examples of aggregators exist today. Financial software, such as Intuit's Quicken and Microsoft Money, can be used to present aggregated views of a user's financial and credit accounts. Certain e-mail scanning software can tell you if messages are waiting in any of several online mailboxes.

Note

Yodlee (http://www.yodlee.com/) is a website that specializes in aggregation. Using Yodlee, users can view one concise view of all of their accounts. The thing about Yodlee that makes it unique is that it can aggregate a diverse range of account types.

The Java Programming Language

The Java programming language was chosen as the computer language on which to focus this book because it is ideally suited to Internet programming. Many programming techniques, which other languages must use as third party extensions, are inherently part of the Java programming language. Java provides a rich set of classes to be used by the Internet programmer.

Java is not the only language for which this book could have been written because the bot techniques presented in this book are universal and transcend the Java programming language; the techniques revealed here could also be applied to C++, Visual Basic, Delphi, or other object-orientated programming languages. In addition, some programming languages have the ability to use Java classes. The Bot package provided in this book could easily be used with such a language.

This book assumes that you are generally familiar with the Java programming language, but it doesn't require you to have expert knowledge in the Java language. This book does not assume anything beyond basic Java programming. For instance, you aren't required to have any knowledge of sockets or HTTP. You should, however, already be familiar with how to compile and execute Java programs on your computer platform. Given this, a good Java reference, such as *Java 2 Complete* (Sybex, 1999), would make an ideal counterpart to this book.

This book was written using Sun’s JDK 1.3 (JS2SE edition). Every example, as well as the core package, contains build script files for both Windows and UNIX. The JDK is not the only way to compile the files, however. Many companies produce products, called *integrated development environments (IDEs)*, that provide a graphical environment in which to create and execute Java code.

You do not need an IDE in order to use this book. However, this book does provide all the necessary project files that you could use with WebGain’s VisualCafé. The source code is compatible with any IDE that supports JDK1.3. Once a project file is set up, other IDEs such as Forte, JBuilder, and CodeWarrior could also be supported. Microsoft Visual J++ only supports up to version 1.1 of Java and, as a result, it will have some problems running code from this book. It is unclear, as of the writing of this book, if Microsoft intends to continue to support and extend J++.

Wrap Up

As a reader, I have always found that the books that are the most useful are those that teach a new technology and then provide a complete library of routines that demonstrate this new technology. This way I have a working toolbox to rapidly launch me into the technology in question. Then, as my use of the new technology deepens, I gradually learn the underlying techniques that the book seeks to teach. That is the structure of this book. You, the reader, are provided with two key things:

- A reusable bot, spider, and aggregator package that can be used in any Java or JSP project (hereafter referred to as the *Bot package*). This package is found on the companion CD.
- Each chapter contains examples of how to use the Bot package. These examples are also contained on the companion CD.

Complete source code to the Bot package is included on the companion CD. Additionally, the chapters provide an in-depth explanation of how the Bot package works.