CHAPTER 1

Introducing IDEAScript

You've just completed the same analysis for the fiftieth time and wonder if there isn't an easier way to get the job done. Yes, using IDEA is fast and easy, but there must be a way to make things faster still. Of course, you can try to find some way to improve your own efficiency or try to perform the analysis a few less times, but there are limitations to that approach and they often require additional work on your part. Why not have someone else, or more importantly, something else, do the work for you? That's what scripting is all about. This book tells you everything you need to know in order to make the computer work for you, rather than you work for it. In this chapter, you discover just how much benefit you can obtain by spending a few hours learning to tell the computer what to do. As you go through this chapter, you learn how to:

- Understand how automation can make working with IDEA easier.
- Consider the ways in which you can use automation.
- Decide which forms of automation to pursue first.
- Determine how your skills can help you use automation best.

Understanding Automation

The computer community will use all kinds of technical terms you don't understand to describe scripting. In fact, the word scripting itself sounds foreign and technical. What this book really describes is automation, and you use automation every day. When you go to the gas station and fill your car with gas, you're using automation. After all, you don't have to pump the gas from the storage tank yourself—you let the gas pump do the work. When you go to the store, the cashier uses a cash register and scanner to total the amount of money you owe for food—no one uses pen and paper any longer. The cashier is employing yet another kind of automation. You get home and click a button—the garage door opens. The garage door is yet more automation. In fact, it won't take long for you to find automation everywhere in your life. Why not automate IDEA as well?

All forms of automation rely on some kind of control. When you pump gas, you press buttons or tell the gas pump to begin pumping in some other way. Controls inside the gas pump automatically stop the flow of gas. At the store, the act of clicking a few keys on the cash register and scanning the items provides control over the adding process. The garage door opens when you click a button on its control. Likewise, *scripting* is a form of control over IDEA that you exercise using special words and phrases. As you can see, scripting isn't anything new—you've already been exercising control over things all your life.

Scripting is a little more complex than pumping gas, scanning groceries, or opening a garage door, but it also does a lot more for you. The complexity comes in the form of a procedure you must write. Of course, you've already been doing that task for a long time too. Any time you have someone stay at your house to water the plants or ask a coworker to perform a task, you write a procedure for them—you tell them what you want them to do and when to do it. Automating tasks in IDEA is no different. You use control words to write a procedure that IDEA performs for you. This procedure is called a *macro* and you use macros to tell IDEA how to automate tasks for you. The following sections describe the benefits of automating tasks in IDEA in more detail.

How Does Automation Benefit You?

The main reason you're reading this book is to gain a new skill that benefits you in some way. After all, why bother to learn something that isn't going to help you in some way? The following list outlines the benefits you should consider as you read this book:

- You can perform work faster.
- The results you obtain will contain fewer errors.
- Any analysis is performed more consistently.
- Your work becomes more interesting because you can focus on unique tasks.
- You don't have to remember how to perform complex procedures because the procedure is contained in the macro.
- It's easy to justify actions you take based on the consistency of your macros.

Tip

There are many ways in which learning to script will benefit you that this book can't cover. For example, if you know how to script and none of the other people in your organization do, you'll likely find that your job security is greater and you'll receive promotions more often. Many people are afraid of scripting, but you're brave enough to give it a try. You'll find that scripting is actually quite easy and straightforward as the book progresses.

How Does Automation Benefit Others?

Believe it or not, your new skill will also benefit others. When you know how to create macros, you become an important asset to others who don't know how to perform this task or simply want to benefit from what you've learned. The following list outlines the benefits others will receive from your macros:

- People can use your macros to obtain the same benefits you obtain.
- Your organization can perform analysis in a consistent fashion, making the analysis easier for everyone to understand.
- The reports and other output you generate will make it easier to see trends.
- It's possible to create a *workflow* (a standard method of performing a task) for the entire organization.
- A single employee absence won't mean that work stops.

Best Practices for Using Automation

Given the benefits of automation, you may be tempted to use automation all the time. However, automation isn't always the answer; you must use some discretion in employing automation. For example, you wouldn't want a completely automated plane—having a pilot is important for safety reasons. The following list provides best practices you should follow when considering automation:

- Always choose tasks that you'll repeat. The more often you need to repeat a task, the better a candidate it is for scripting.
- Always choose well-defined tasks. In order to write a procedure, you must understand the task completely.
- Always plan your macros carefully and completely so that the procedure works as you expect it should. The planning process begins when you separate tasks that will automate well from those that won't.
- Whenever possible, create macros that everyone in your organization can use, rather than focus on macros for personal needs. When everyone benefits, the time you use to write the macro is paid off faster.
- Whenever possible, write down the procedure you use and then test the procedure carefully. This act is no different from any other automation you use. For example, you'd expect that a cashier would receive training that relies on written and tested procedures.
- Avoid writing macros that are too complicated for your current skill level. Discover scripting a step at a time. As this chapter progresses, you'll learn tricks you can use to avoid getting in over your head.
- Never assume that the macro you write for your machine will work on another machine until you test it on that machine. Just as a procedure for one cash register may not work on another, you can't assume your macro will work on every machine. As the book progresses, you'll discover methods for testing your macros to ensure they work as anticipated.

Understanding How You Use Macros

Macros, the written procedures used for scripting, can perform all kinds of tasks. In fact, the number of tasks you can perform is literally limited only by your imagination. Some people have written games and done all kinds of other interesting things with macros. Of course, most people use macros for more practical purposes. The following sections describe some of the common tasks you can perform with macros.

Interacting with Databases

The task you perform most often in IDEA is interacting with a database of some sort. From your perspective, you're performing a data analysis. However, from the perspective of the IDEA application, you're manipulating data found in databases. Whatever the perspective, being able to get the data you need is important, and it's often repetitious. Macros that help you get the information you need are probably the singular most important kind of macro that you can write.

Starting with Chapter 7, you begin working with databases and discover that no matter the source, databases often have similar needs and requirements when writing a macro. Chapter 8 shows you how to interact with databases, while Chapter 9 describes the model IDEAScript uses to work with databases. When working with IDEAScript, you can access two kinds of databases:

- Internal IDEA databases
- External databases such as SQL Server

This book helps you work with both kinds of databases. The external database information starts in Chapter 14 and you see some advanced techniques in Chapter 17. In fact, you'll find that you can access data in all its forms, even an Excel document (see Chapter 16) or a text file (see Chapter 15) on your hard drive. Procedures that might seem complex when you perform them by hand suddenly become easy and fast when you use a macro to perform them.

Customizing the IDEA Interface

You can attach (bind) your macros to the IDEA interface. By adding buttons that access your macros, you can customize the IDEA interface to meet your specific needs. Your macros, in essence, become part of the IDEA application and make using IDEA easier. Chapter 4 tells you how to add your macros to the IDEA application interface.

Performing Calculations

Analysis normally includes performing comparisons and employing equations to calculate specific values. Of course, you want to be sure you perform the right comparisons and obtain the correct calculations. Fortunately, your computer is far faster and significantly more accurate in both comparisons and calculations, so this is one area that you really should let the computer take care of for you. Chapter 6 tells you how to perform comparisons, while Chapter 10 addresses math requirements.

Designing New Application Features

Binding macros to the IDEA interface isn't the only kind of customization you can do. In addition, you can create your own interface elements as dialog boxes. You can use dialog boxes for two purposes:

- To output information to the user.
- To obtain input from the user.

You can do everything from telling the user of your macro the status of a calculation to asking the user for the name of the database they want to use for analysis. Chapter 12 shows you how to create interactive dialog boxes that let you do amazing things.

Importing and Exporting Data

Most businesses need to exchange data in some form or another. Depending on the kind of analysis you perform, you might have to obtain data from many different businesses. Unfortunately, businesses use different applications and different methodologies to store information. Trying to remember all of the methods used to access this information can prove daunting. Fortunately, macros make fast work of importing and exporting data as needed. Chapter 14 describes the resources that IDEAScript provides for importing and exporting data.

Data comes in many forms. Normally, you'll work with databases, but that isn't always the case. Besides the database chapters in this book, you can discover how to interact directly with files in Chapter 15 and Excel in Chapter 16.

Controlling Other Applications

One of the most useful ways to use macros is to control other applications. It's inconvenient and time wasting to have to interact with more than one application at a time. If you can perform at least part of that work by using a macro, you save time and can maintain a focus on IDEA.

Nothing limits the kinds of applications you can control. If you want to start a copy of Excel and use it to create a chart or graph, you can do so with the information found in Chapter 18. Chapter 19 shows how to control other applications, such as Word, to create reports.

Warning

Never execute macros unless you know what task the macro performs and that the macro is safe to use. Executing a macro that you don't know about can cause damage to your data or produce unreliable results. In addition, macros can cause significant problems on your machine, such as installing a virus. The macros that are completely safe are the ones you understand and obtain from a reliable source.

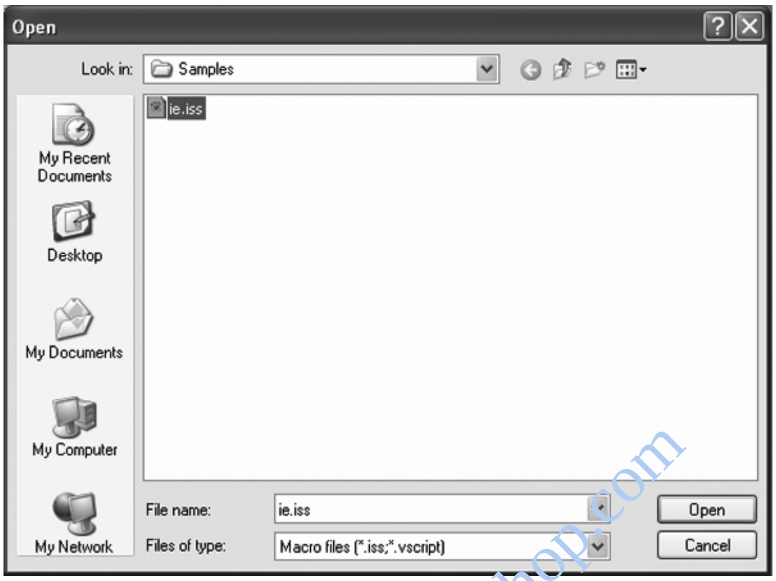


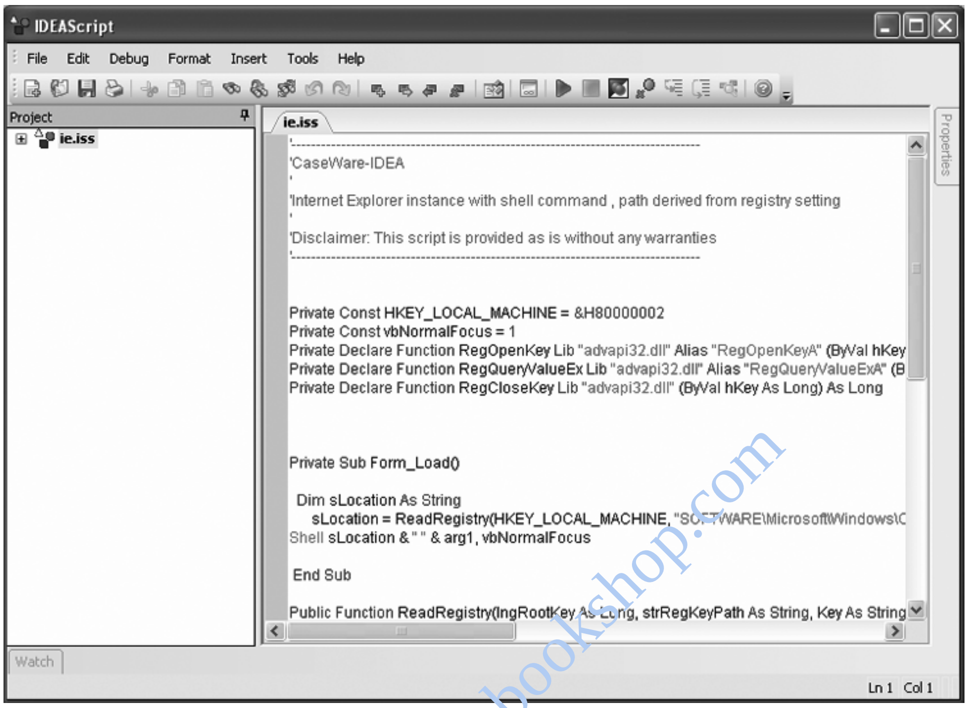
FIGURE 1.1 The Open Dialog Box Lets You Choose a Macro File on Your Hard Drive

You're probably thinking that controlling another application sounds too hard, but it really isn't. The first exercise in this book is to execute a macro that starts a copy of Internet Explorer. This macro actually comes with your copy of IDEA. Just follow these steps.

1. Open IDEA and select **Tools** > **Macros** > **Open**. This command displays the **Open** dialog box shown in Figure 1.1. IDEA automatically selects the `Samples` folder for you, which contains a macro named `ie.iss`. If you don't see the `Samples` folder, you can find it by locating `\My Documents\IDEA\Samples` in the **Look in** field.
2. Select `ie.iss` and click **Open**. IDEA opens the **IDEAScript Editor** shown in Figure 1.2. Don't worry about the editor for now; Chapter 2 tells you how to work with it. The text you see in the right **Editor** pane is a macro and we're going to execute it.
3. Click **Run Script**. The **Run Script** button is the blue right-pointing arrow on the toolbar. You can also press **F5**. IDEA executes the macro and opens a copy of Internet Explorer for you. Congratulations! You just executed your first macro.
4. Close Internet Explorer. Select **File** > **Exit** in the **IDEAScript Editor**.

Having Things Your Way

For many people, the idea of scripting can become overwhelming. At first, you can't quite accept that you can actually write macros, but then, once you get used to the idea, all kinds of macro ideas start coming to mind. It's nice to have things your way. Once



```
'-----------------------------------------------------------------------------------
'CaseWare-IDEA
'
'Internet Explorer instance with shell command , path derived from registry setting
'
'Disclaimer: This script is provided as is without any warranties
'-----------------------------------------------------------------------------------


Private Const HKEY_LOCAL_MACHINE = &H80000002
Private Const vbNormalFocus = 1
Private Declare Function RegOpenKey Lib "advapi32.dll" Alias "RegOpenKeyA" (ByVal hKey
Private Declare Function RegQueryValueEx Lib "advapi32.dll" Alias "RegQueryValueExA" (B
Private Declare Function RegCloseKey Lib "advapi32.dll" (ByVal hKey As Long) As Long



Private Sub Form_Load()

 Dim sLocation As String
   sLocation = ReadRegistry(HKEY_LOCAL_MACHINE, "SOFTWARE\Microsoft\Windows\C
Shell sLocation & " " & arg1, vbNormalFocus

 End Sub

Public Function ReadRegistry(lngRootKey As Long, strRegKeyPath As String, Key As String
```

FIGURE 1.2 The IDEAScript Editor Lets You Write Macros That You Can Save to Disk

you discover the full capability of scripting, you'll find that you can do a host of things that you didn't think were possible in the past.

Of course, that list of ideas can become a burden, too. Make sure you write your ideas down because it's all too easy to forget something you want to do. Even if you don't know how to write the macro today, write it down. As you work with IDEAScript, your skills will improve and today's impossible task will become quite possible tomorrow. Make sure you prioritize your list. Use these criteria for prioritization:

- **Skill Level:** Your skill level determines the macros you can create today.
- **Existing Knowledge:** Macros that play to knowledge you already possess, say a math macro if you're already a math expert, should be a priority.
- **Pressing Need:** Personal or organizational needs can act as a great motivator to finish the macro. Some people try for a short time and then give up—finishing the macro is the only way to build the knowledge you need.
- **Interest:** Some projects are definitely more interesting than others. It's more likely that you'll finish a macro that interests you, so be sure to tackle these macros first.

Considering Your Skills

Even though macro writing is like many other things you've already done and is basically writing a procedure for IDEA to follow, it's still a skill. As you write more macros, you'll

Chapter = c01 Date: Feb 17, 2011 Time: 10:13 am

learn more about the IDEAScript language and be able to create procedures that are more complex than those you create at first. The best way to learn scripting is to start slowly and discover new commands one at a time until you become proficient.

Some of the skills you already possess will help as you discover IDEAScript. For example, if you already have solid math skills, you'll find that writing macros that perform math tasks is significantly easier. You may even want to focus on math-related macros when you first begin scripting. Some people already know quite a bit about databases, so working on database-related macros is easier. Don't force yourself to start out with something too difficult—ease into scripting.

You should create some goals for yourself based on the scripting needs you discover as you work with IDEA. Put these goals on your To Do list—the same as you would anything else you want to learn. When you have a little additional time or you're waiting for another task to complete, take some time to learn a new IDEAScript command and then begin employing it in your macros. It won't be long before you'll be writing complex macros without any trouble at all.

No one's asking you to memorize anything. The purpose of this book is to act as your memory. As you work through the book, you'll discover that IDEA provides a number of other useful aids to make writing macros easier. Writing macros should be something you do to improve your work experience, not ruin your mood.

Summary

This chapter has started you on the road to a new kind of experience—scripting. The most important idea to take from this chapter is that anyone can write a macro as long as they fully understand the task at hand. While not every task is suitable for automation, many tasks are and you should make full use of this capability in IDEA to reduce your workload. The macros you create help both you and everyone else in your organization, so writing good macros is essential.

One of the most important aspects of using automation is to employ it correctly. Of course, only you can decide when automation applies. Before you go to the next chapter, consider a few places in which automation will help you and your organization. Using the information in this chapter, write down the pros and cons of using automation for the tasks you define. Present your list and reasoning to other people and see if they agree that automation is the right choice for the tasks you list. This exercise will save you considerable time trying to automate tasks that you really shouldn't automate.

Now that you have a list of tasks you want to automate, Chapter 2 takes the next step and begins to show how to create macros. Of course, your first macros will be very simple. You want to make scripting fun and easy to perform, so these initial steps are important. The macro in Chapter 2 is functional and you can even show it off to your friends. However, you'll create significantly more interesting macros as the book progresses.