1

Learning to Provide Modern Solutions

Over the course of this book, you will learn to enhance your existing applications to modernize the output of the system. To do this, we'll take advantage of the modern capabilities of ILE RPG that you can incorporate into your existing code. The Integrated Language Environment (ILE) enables you to write code in a much more modular way, which means you can write code in small, reusable components that are better encapsulated and easier to maintain.

Providing Solutions to Meet Today's Industry Standards

When you look at your Original Program Model (OPM) programs and think about why these programs were developed, the goal was likely to provide comprehensive software that would withstand the test of time and produce solid, reliable results as the backbone for business-critical operations. If you were to focus solely on this aspect, you might not see the need to even bother using ILE. After all, you have a programming language that works, it has been around for a very long time, and it is easy to understand. So, let's consider some of the limitations

of this way of programming and define the reasons why modernization of the code is necessary.

The first thing that comes to mind when discussing modernization is aesthetics. When you have a system that provides indisputable data that is necessary for business processes, you may consider the ability to create "modern" reports in Microsoft Excel or Adobe PDF form as being an unnecessary enhancement. After all, the data is being generated, the information is being audited, reporting is being performed, and the company is generating revenue. How important is it to make the reports look better? Well, there are other factors beyond appearance that you need to consider when evaluating the need for spreadsheets and PDF files.

Portability

Portability is a big one. To address this issue, your initial thought might be to generate ASCII text files. If you've been working with RPG for a few years, there's a good chance you've already tried this approach.

However, if you delimit the ASCII data using a special character, you must make the user aware of which delimiter you're using, and the user must process the data properly. This means you need specifications that identify the fields and how they should be separated. In the end, the data is likely to be imported into an Excel spreadsheet anyway.

When you provide an ASCII text file, you may have different formats for multiple records in the file. For example, you may have a header that displays a title and possibly user information that does not match the layout of the data, requiring you to identify when record formats change.

There is also the issue of delivering the file to the users. Will you use a data transfer that must be defined for each user to download the correct physical file member? Will you use File Transfer Protocol (FTP) and require a user name and password?

If you e-mail the text file, its carriage returns and line feeds might be altered during conversion through MIME format. Will you zip the file to avoid this issue, or will your e-mail application embed the text into the e-mail message and force the user to copy and paste the data from there? You could make the data available using a Web server, but if you're going to go that far, you might as well use HTML.

The bottom line is that unless you're dealing with another programmer or a power user, you will likely encounter users who resist using ASCII text files, especially given the other options available today.

PDF files are built specifically to overcome these issues. With PDF files, all that users need is a reader, and they can view any PDF file without any of the preceding concerns. PDF readers are typically free and work on most operating systems. In fact, one is probably already installed with your OS.

Even though PDFs are portable, they can be difficult to modify, which can be a good thing or a bad thing depending on your intentions.

Microsoft Excel files offer the same portability as PDF files. A free viewer available from Microsoft lets users view these files in read-only mode. And with the availability of several open-source options, you can easily create and change such files using software that is supported on most operating systems.

Usability and Standardization

Another reason to provide electronic documents is usability. When you furnish data in an Excel spreadsheet, users can easily search and sort the data, creating ad hoc reports and performing "what if" scenarios. They can also create graphical representations of the data using charts and graphs.

Last, Excel and PDF are two of the most commonly used types of data. Providing data in these standard formats makes the work of your users much easier to use and share with others.

Evolving Your RPG Development Skills

In the following chapters, we'll explore the evolution of RPG from OPM to ILE and discuss how to exploit the new capabilities of the IBM i platform. Our discussion is structured in a logical direction, first building on your existing RPG coding skills by explaining the new features available in ILE and the new concepts involved with scoping and modular coding practices.

We'll start by converting some sample OPM source to an ILE-compatible format and implementing the use of the ILE compilers. With ILE, there are some new concepts to cover that involve the binding of components and the differences between an OPM program and an ILE program. The primary objective of the book is to integrate RPG with Java, so we'll focus our efforts on capabilities that directly lead us to this goal.

Once we have the code in an ILE format, we'll begin to discuss procedures, service programs, and activation groups and why you may want to use them. For the purposes of this book, we'll look at how you can access Java objects and methods.

We'll discuss the definitions of objects and methods thoroughly in upcoming chapters. But on the simplest conceptual level, you can think of an object as being similar to a program that is called and put onto the call stack. It will be initiated to be contained in memory and will have variables initialized to specific values based on their data types.

Java methods are similar to the subroutines that are available to a mainline program. Each subroutine has a specified name and function designed to make the subroutine a reusable resource to the mainline program. Java methods are actually more like procedures because they have clearly defined parameters — one of the new ILE RPG capabilities we'll discuss. So, learning to use this new feature will also help you to understand Java methods when we get to that point.

Developing Reusable Code

In our journey toward understanding the use of Java classes and methods from RPG, we will make every effort to encapsulate the Java interface into reusable code. When we do this, I'll go into a detailed analysis of how to interface with the Java environment, and I'll provide some RPG-friendly code that you can use in the future. I call these pieces of code "RPG wrappers" because we'll be wrapping RPG code around the Java classes and methods.

You can make the RPG wrappers available to the program you're working with and then use pure RPG code to access the Java functionality. This technique not only makes the code easier for you to use in the future; you can also use it to perform all the exception handling, so you won't have to repeat the same code.

This approach also lends more stability to your code because as you reuse the code, it will be exposed to more applications that employ it in different scenarios, flushing out any problems. This, in turn, will cause you to make the necessary changes to the code, which will then become more stable for the application in question and all the others that use it. When you start the next project, you'll have some solid building blocks with which to begin.

Once you have proven, reusable code components, you also will be able to reduce development time for subsequent projects and lessen maintenance support time because you'll be using a smaller percentage of first-time code. So, you win all around with the use of reusable code.

Another benefit to wrapping Java classes and methods in RPG is that you can easily share the new Java capabilities with your fellow RPG programmers in a form that they will be able to use immediately.

Integrating RPG with Java

Enhancing your RPG development skills is a great side effect of the primary objective of this book, which is to exploit the capabilities of Java from within RPG. When we get to that point, we'll explore all the new concepts you need to

understand about Java to get it to work properly. It would be nice if you could just plug in a Java method, as you can an application programming interface (API). But there are other things you need to think about when dealing with the Java Virtual Machine, such as starting and stopping it. And because two totally different environments are communicating with each other, you need to take some extra measures to handle some of the ugliness of reclaiming memory resources.

Some of these topics may sound unappealing to you, and I agree. But fortunately, both IBM and Sun Microsystems have provided some useful code segments for dealing with these types of matters. So, what we'll do is create some handy procedures that will take care of most of these details for us, and we'll have to do minimal coding to support these issues.

It took me quite a while to get through all these issues when I first started. There were little pieces of information all over the place, and I had to discover the solutions as I stumbled through one frustrating setback after another. I hope to provide you with a much smoother transition into the world of Java than I had.

I don't mean to sound discouraging. If anything, I intend just the opposite. My goal is to help you to easily overcome those obstacles and get you into production mode relatively quickly. As with the introduction of any new programming language, you will have to go through the growing pains of the initial learning curve. But a practical programming guide dramatically accelerates your ability to provide results quickly. And here we are!

This book introduces you to the Java components necessary to provide the results you're looking for. It is not intended to teach you Java, but it does introduce you to Java and how to use specific parts of open-source technology to provide the product of your hard work, which in this case will be Excel spreadsheets and PDF files that you can distribute using e-mail from RPG.

Creating Electronic Documents

This is where the fun stuff starts to happen. You will learn a new skill in a real-world environment that will actually produce tangible results that are clearly visible to your users. Most of the time, new programming skills give you a greater knowledge base and better mechanics for your code development. But the latter sections of this book will give you the capability to hand over to your users a shiny new product that you know they will totally enjoy! This, in turn, will add much value to your software in the eyes of your users.

If your objective is to provide reports that can be sorted, graphed, charted, and manipulated, you can convert your greenbar reports into an Excel spreadsheet. We'll do this using the Horrible Spreadsheet Format (HSSF) capability of the POI open-source project.

If your intention is to provide documents that can be distributed easily to users and customers in a portable format that anyone on any operating system can use, you may want to convert your greenbar reports, billing invoices, or labels and bar codes to the PDF format. We'll use the iText open-source project to provide these capabilities.

We'll accomplish these objectives by first defining the technical terms and concepts of the components that make up an electronic document and relating them to the Java classes and methods available in POI and iText. Then, we'll convert the methods and parameters into an RPG-friendly format and build wrappers for them. After that, you can start using them from within RPG. Sounds simple enough!

Distributing Electronic Documents Using E-mail

Once you have a program that generates electronic documents, you'll need to decide how you're going to get the end results to your users. My first step was to put the electronic documents onto a publicly accessible network drive. This solution was okay when I first got started with a few users. But my user base quickly grew, and certain reports needed to be visible only to specific users, and it just

got out of control from there. Users were starting to archive their reports on the network drive and calling the IT support line about files they were seeking. As the number of files grew, the amount of time required to perform backups grew as well, requiring more administration than I had bargained for.

So, I soon decided to accommodate these requirements using an automated e-mail system. This approach allowed the distribution lists to be changed to send only to particular users, gave users the option to save or delete the files, and did not require users to have a network drive mapped. It also made it possible to easily send the reports to customers who didn't have access to the network.

When evaluating the options to provide my IBM i system with e-mail capability, I found JavaMail to be the most self-contained and flexible solution. Using JavaMail, we will build an e-mail client that simply creates the e-mail and relays it through your current e-mail server — no additional servers or configuration required. We'll start with a simple text e-mail, specifying the to, from, and subject, and then advance to formatted e-mails containing HTML and attachments. You can even embed the images right into the e-mail, if desired.

Using JavaMail for your e-mail client frees you from being tied to the operating system. Once you understand the JavaMail API and you start using Java, you'll be able to easily rewrite your RPG program into pure Java and use the exact same API to provide the same capability to any other server on the network. This freedom and knowledge will also help you as a programmer to begin to integrate with other areas of technology in your company and will give you greater flexibility, not only in your programming options but as a valued member of the programming team.

And that's just the beginning! In addition to giving you the isolated capabilities of the service programs we'll discuss, I'll walk you through the process of identifying and implementing the classes and methods. So, once you get your RPG programs to start generating electronic documents and sending e-mails, you can further explore other options that are available and expand your service programs to implement more and more features. I hope this will be the beginning of a whole new level of programming using Java for you. Let's get started.