THE EXPERT'S VOICE® IN DATABASES

SECOND EDITION

Pro Oracle Application Express 4

MAKE APPLICATION EXPRESS "DANCE" WITH UNDER-THE-HOOD TECHNIQUES FOR CUSTOMIZING, SECURING, LOCALIZING, AND DEPLOYING YOUR APPLICATIONS

Tim Fox, John Scott, and Scott Spendolini

Foreword by Thomas Kyte

Apress®

Pro Oracle Application Express 4

Second Edition

Tim Fox
John Edward Scott
Scott Spendolini

Apress®

Pro Oracle Application Express 4, Second Edition

ISBN-13 (pbk): 978-1-4302-3494-4

ISBN-13 (electronic): 978-1-4302-3495-1

President and Publisher: Paul Manning
Lead Editor: Jonathan Gennick
Technical Reviewer: Alex Fatkulin
Editorial Board: Steve Anglin, Mark Beckner, Ewan Buckingham, Gary Cornell, Jonathan Gennick, Jonathan Hassell, Michelle Lowman, James Markham, Matthew Moodie, Jeff Olson, Jeffrey Pepper, Frank Pohlmann, Douglas Pundick, Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh
Coordinating Editor: Jessica Belanger
Copy Editor: Kim Burton-Weisman
Compositor: Bytheway Publishing Services
Indexer: BIM Indexing & Proofreading Services
Cover Designer: Anna Ishchenko

Distributed to the book trade worldwide by Springer Science+Business Media, LLC., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail `orders-ny@springer-sbm.com`, or visit `www.springeronline.com`.

For information on translations, please e-mail `rights@apress.com`, or visit `www.apress.com`.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales–eBook Licensing web page at `www.apress.com/bulk-sales`.

I would like to dedicate this effort to my father, Louis Fox,
for introducing me to software development by bringing home a TRS-80 when I was 14.
—Tim Fox

Contents at a Glance

Contents

About the Authors

■ **Tim Fox** has been using Oracle since version 5 in 1989. At that time, the Oracle Forms version was 1.x and procedural code had not yet arrived. Since that time, Tim has been constantly involved in projects where some version of an Oracle database or development tool was being used. Tim has worked in corporate IT for the State of Indiana and The Associates, spent ten years as a manager at Andersen Consulting / Accenture, and is currently the Director of Development at Enkitec, a Dallas-based Oracle services provider.

While at Enkitec, Tim has been an instrumental figure in development projects across dozens of companies and has greatly assisted in the uptake of Oracle Application Express. As a result, Tim is a frequent contributor to the Dallas Oracle User Group (DOUG), Oracle-sponsored Tech Days, and other regional user groups and conferences. In 2010, Tim became heavily involved in implementation and training related to Oracle's Exadata database machine. Given his background in APEX development and his access to Enkitec's Exadata machine, he has APEX 4 running directly on Exadata.

■ **John Edward Scott** has been using Oracle since version 7 (around 1993), and has used pretty much every release since then. He has had the good fortune to work on a wide range of projects for a varied group of clients. He was lucky enough to start working with Oracle Application Express when it was first publicly released, and has worked with it nearly every day since (and loves it). John is an Oracle ACE Director and was named Application Express Developer of the Year 2006 by Oracle Magazine. He is also the cofounder of ApexEvangelists (http://www.apex-evangelists.com), a company that specializes in providing training, development, and consulting specifically for the Oracle Application Express product. You can contact John at john.scott@apex-evangelists.com.

■ **Scott Spendolini** is president & co-founder of Sumneva™, a world-class Oracle® Application Express (APEX) consulting, training, and solutions firm founded in 2010. He has assisted a number of clients from various verticals with their Oracle APEX development and training needs. Spendolini has presented at a number of Oracle-related conferences, including Oracle OpenWorld, ODTUG, and IOUG, and is a regular contributor to the Oracle APEX Forums on OTN. Spendolini is also the host & conference chair of APEXposed, an APEX-specific annual conference. He is a recent recipient of the Oracle Ace Director designation and is also a co-author of the book Pro Oracle Application Express. In 2009, Spendolini along with ODTUG was presented with the Oracle Innovation Award for his work on ODTUG's public web site, odtug.com. Spendolini is also an Oracle Certified Oracle Application Express developer.

Previous to co-founding Sumneva, Spendolini founded and ran Sumner Technologies from 2005 through 2009, which also focused on Oracle APEX consulting, training, and solutions. Before that, he was employed by Oracle Corporation for almost 10 years, the last three of which he was a Senior Product Manager for Oracle APEX. He holds a dual bachelors degree from Syracuse University in Management Information Systems and Telecommunications Management and currently resides in Ashburn, Virginia with his wife and two children.

About the Technical Reviewer

■ **Alex Fatkulin** is master of the full range of Oracle technologies. This mastery has been essential in addressing some of the greatest challenges his customers have met. Alex draws on years of experience working with some of the world's largest telco companies, where he was involved with almost everything related to Oracle databases, from data modeling to architecting high availability solutions and resolving performance issues of extremely large production sites.

With companies that rely on technologies such as RAC, Oracle Streams, and Data Guard, Alex's intimate knowledge of client, data, and software gives them robust and dependable production systems that allow them to run their businesses smoothly and stay ahead of their competitors.

Alex holds a Bachelor of Computer Sciences degree from Far Eastern National University in Vladivostok, Russia.

Acknowledgments

Working on a project like this is not a singular effort. In my case, my family had to give me time to work on a daily basis, which wasn't easy for Reese and Ryan, my four- and seven-year-old daughters. They didn't really understand what I was doing (they're not APEX developers … yet), but they knew that I was "working on the book" almost every night. My wife, Thanh, probably paid the highest price as I tried only to work after the kids were tucked in for the night. She was very supportive of this project and fell asleep next to me on the couch more times than I can count. For all of that, I thank my family more than they know.

During the course of writing this book, I learned a great deal more than I had anticipated, but I did not do it alone. The team that is Enkitec (my day job) is a group of people I like to call "home run hitters," many of whom are currently working on critical APEX projects. You'll see a few of their first names in this book, but I'd like to thank our core APEX team formally now: Brian Hill, David Little, and Toby Marks. I feel very lucky to be associated with a group of really smart people who genuinely enjoy what they do.

I ultimately have Kerry Osborne to thank for getting me into this project in the first place. He and another Enkitec colleague, Randy Johnson, were embarking on writing the *Pro Oracle Exadata* book and he encouraged me to join in the fun of technical writing. I actually thought at the beginning that I would be able to work on this project during business hours, but that didn't happen. It didn't happen for Kerry and Randy either so now we share stories of marathon writing sessions and comparing amusing comments from the editors.

Speaking of the editors, without their experience and guidance, this would be a very different book. Jonathan Gennick and Jessica Belanger of Apress worked diligently with me to ensure that I had all the resources I needed and that I stayed on schedule. Alex Fatkulin was also instrumental as a technical reviewer.

Since this book is a revision of the original *Pro Oracle Application Express*, I have to give the original authors their due. John Edward Scott and Scott Spendolini are obviously APEX experts in the truest sense of the word. The body of APEX knowledge that they created in the first edition is still relevant today and is a must for any serious APEX developer. Many of the concepts they describe apply to any Oracle-based development whether APEX, Java, or otherwise.

As for APEX in general, I continue to be amazed by the completeness and the professional feel of the development interface and applications created with it. APEX 4.0 takes application look and feel to a higher level than ever before and I find it truly amazing how easy APEX makes it is to create fully functional, professional applications. In my day job, I always offer APEX as the development language of choice for web-based applications. I explain it to my customers like this: in the time it takes to describe how a form should work, I can have it built in APEX. Development productivity can't be matched by any other environment (in my opinion). Maybe the best thing about APEX is that it's still 100% free.

—Tim Fox

Foreword

I consider myself a pragmatic person—one who uses the right tools for a job and employs the most straightforward and easy way to accomplish a task. To that end, I've been a great supporter and fan of Oracle's Application Express (APEX) from before the day it was introduced. I say "before the day" because I've had the honor and pleasure of using APEX long before it was released to the public at large. My web site, `http://asktom.oracle.com`, is one of the first ever built with the software that was to become known as APEX.

APEX is one of the most pragmatic database development tools I know of. It does one thing and one thing well: it rapidly implements fully functional *database* applications—applications that are used to predominantly access, display, and modify information stored in the database (you know, the *important* applications out there). It facilitates using the database and its feature set to the fullest, allowing you to implement some rather complex applications with as little work (code) as possible. It is possible to build extremely scalable applications with a huge user base (`http://shop.oracle.com`, for example, is built with APEX). It is possible to build extremely functional applications, with seriously powerful user interfaces (APEX itself is written in APEX, as proof of this). It is easy to build applications rapidly; for example, the current version of `http://asktom.oracle.com` was developed in a matter of days by two developers—in their spare time; it was not a full-time job.

While it all sounds wonderful and easy so far, APEX is a rather sophisticated tool with many bits of functionality and a large degree of control over how the generated application will look and feel. To fully utilize the power of APEX, you need to have a guide and a mentor to show you how to do so, very much akin to what I do with people regarding the Oracle database.

This book, *Pro Oracle Application Express 4*, is that guide. The authors, Tim Fox, John Scott, and Scott Spendolini are those mentors. The book walks you through the steps you need to understand after you've installed and started using APEX, to go beyond the sample applications. Covering diverse topics such as using the database features to full advantage (one of my favorite topics), to SQL injection attacks (what they are and how to avoid them in APEX), to printing, you'll find many real-world issues you will be faced with explained, demystified, and solved in this book.

This second edition of the book has been enhanced to cover some of the very latest additions to APEX. One of the most powerful new features is the ability to create your own components using a new plug-in architecture. You can create new components, and you can share those components in standardized ways. Chapter 12 goes into more detail on this exciting aspect of APEX development.

Other new features covered in this edition include Websheets and Dynamic Actions. Websheets provide a friendly and accessible way for end users to create their own applications. Dynamic Actions represents APEX 4.0's new support for Ajax, helping you to create highly interactive applications that run in the browser without constantly needing to refresh and load new pages.

Chapter 4, "Data Security," covers a wide breadth of topics about securing your database application, which is an even greater concern now than when the first edition was published. There is a section on URL injection issues that discusses what they are, how they are exploited, why you should care about them, and how to protect yourself from them. There is a section on session state protection that follows the same format: what it is, how it is exploited, why you should care, and how to protect

yourself. The same mentoring occurs with data-level access, where the authors introduce how to use Virtual Private Database, a core database feature (not really an APEX feature) to protect your data from unauthorized access. Lastly, a critical application feature, auditing, is discussed in depth using the same "what it is, why it is, why you should care, and then how to do it" approach. While some of the content in this chapter is not specific to APEX, it is needed to give you a holistic view to building database applications, which is what this book is about.

This book covers not just the nitty-gritty details of building a secure application, but it also covers all you need to know to build database applications with APEX. When they are finished with security, the authors move on to other necessary topics, such as how to perform screen layout and application screen navigation, how to integrate reports and charts, how to integrate web services—enabling you to perform application integration—in an APEX environment, and much more.

If you are an APEX developer just starting out, or an APEX developer with experience under your belt and want to learn more about the environment you are using, this book is for you. It describes from start to finish how to build secure, functional, scalable applications using the APEX application development environment.

—Thomas Kyte
http://asktom.oracle.com

Introduction

If you're new to APEX development, prepare yourself for one of the most productive tools available for Oracle application development. The speed with which you can create fully functional, secure, web-based applications is truly astounding, especially when compared to using a tool like Java. If you are used to writing procedural or object-oriented code, you may find APEX's declarative development methodology a little challenging at first, but the learning curve is not significant. When you consider that you only need an Oracle database to get started, the barriers to beginning APEX development are minimal.

The first edition of this book, *Pro Oracle Application Express*, was written in 2008 using APEX 3.2. In 2010, Oracle released APEX 4.0, which included a significant number of new features along with a major upgrade of the development environment's user interface. Along with the much improved look and feel, Oracle included several Web 2.0 features like Dynamic Actions, Plug-ins, and RESTFul web service support. Prior to APEX 4.0, you could implement the newest web features, but it usually required custom coding. With APEX 4.0, you can use the familiar declarative development techniques to add an Ajax feature, for example. The inclusion of these significant new features obviated the need for a revision of the original book.

In general, APEX 4.0 lets you create professional looking, feature-laden applications with less effort than ever before.

CHAPTER 1

Development Best Practices

Oracle Application Express (APEX) makes it extremely easy to quickly prototype and develop a web application. However, as a software developer, you're probably aware that speed of development is only one of a number of criteria that will contribute to the perceived success (or failure) of your project.

Of course, perceptions about a project's success can vary. For example, the people who encounter a typical project might include developers, testers, managers, production support, and end users. The developers may feel that the project was a success because development went quickly, Production support might feel like the project was a failure because no one has a clear strategy for performing application upgrades. The end users may dread using the application because it runs incredibly slowly. Clearly, for the project to be considered a success, you need to satisfy the expectations of all these people (or as many as you reasonably can). Ideally, you should strive for an application that has the following characteristics:

- Easy to develop
- Easy to deploy and upgrade
- Easy to maintain and debug
- Enjoyable for end users to use
- Fast enough for the users' requirements
- Stable from the end users' perspective
- Secure enough to protect your data from unauthorized access

You should never end up feeling like developing, deploying, maintaining, or (even worse) using the application is seen as a chore. Each of these areas can often benefit from the adoption of some best practices to ensure that everyone involved sees your application as a success.

Chapter 1 is the best place to introduce best-practice techniques, since they should form the foundation of every significant development project you undertake. You can certainly create applications without using any of the techniques we'll discuss, but adopting techniques like these will make your job as a developer easier, and your applications will be considerably more successful.

APEX Installation Decisions

This book will not cover the details of installing APEX version 4.0, since that information is already bundled with the product, as well as discussed in detail in several online resources. It is worth mentioning, however, that APEX 4.0 supports another connection type via the Oracle Application Express Listener. The APEX Listener is a Java-based web server that is certified to run under Web Logic, Tomcat, OC4J with Oracle WebLogic Server, OC4J, and Oracle Glassfish.

Indeed, many people enjoy using APEX without bothering with installation, either because someone else has installed it for them or they are using a hosted environment (such as the public Oracle `apex.oracle.com` site or one of the commercial providers such as Shellprompt). Others use Oracle Database Express Edition (XE), a free edition of the database that includes a preinstalled version of APEX.

However, if you are installing APEX, one important decision is which tablespace to use for the product. The installer usually defaults to installing APEX into the `SYSAUX` tablespace, and if you're using the 11*g* version of the database, the `SYSAUX` tablespace will be selected by default. I highly recommend that instead of using `SYSAUX` you create a dedicated tablespace that you'll use specifically for the APEX database objects and metadata. By using a dedicated tablespace, you can gain a far greater degree of control and flexibility over the administration of the APEX environment. For example, should it become necessary to recover the tablespace using point-in-time recovery from an Oracle Recovery Manager (RMAN) backup, you'll be confident that you haven't affected any other database components, like AWR (which may not be the case if you install into `SYSAUX`).

Installing into a separate, dedicated tablespace also allows the database administrator (DBA) to make decisions about where that dedicated tablespace should be stored on disk (to reduce contention), control the storage growth of the tablespace, and perhaps also take advantage of advanced Oracle features, such as transportable tablespaces to quickly move the tablespace to another database instance.

Application Development Considerations

The decisions related to how to create and organize your application within the APEX and database environment will greatly affect how easily you'll be able to deploy and migrate your application later on. By structuring your development environment in a logical and organized way, you'll encounter far fewer problems when your application needs to be deployed or updated to your live environment.

Users and Administrators

When APEX is installed, an Application Express instance administrator is created. You can connect to APEX as this instance administrator in two ways:

- Connect to one of the following URL's:
- PL/SQL Plugin: `http://server:port/pls/apex/apex_admin`
- APEX Listener or EPG: `http://server:port/apex/apex_admin`

Use the username `ADMIN` and the password you used when you installed the product.

- Connect to the same URL you'd use to log into any workspace, such as

  `http://server:port/pls/apex/apex_login`, and use `INTERNAL` as the workspace and `ADMIN` as the username with the password you used when you installed the product.

Including the instance administrator, four different types of users exist for APEX:

Application Express instance administrator: This is the user you'll use to administer the APEX installation. The instance administrator can connect only to the `INTERNAL` workspace to perform administration tasks, such as creating workspaces and users, monitoring activity, and managing the APEX service. Instance administrators can't create any applications themselves; they must create workspaces and other users for applications to be created. The instance administrator can create workspace administrators, developers, and built-in users for any of the workspaces.

Workspace administrator: A workspace administrator is responsible for the administration of a particular workspace. As a workspace administrator, you are able to create developers and users for that workspace, and create applications. Workspace administrators can also log into any application within the same workspace that uses APEX account credentials.

Application developer: Application developers are created within a particular workspace by workspace administrators. They can create and maintain an application within that workspace. They can't log in to other workspaces, but they are able to log in to any application within the same workspace that uses APEX account credentials.

Application user: Application users can take two forms. They can be created and managed within the APEX environment and are then known as *built-in users* (or *cookie users*). Alternatively, they can be created and managed outside the APEX environment; for example, they could be stored within a database table or as part of a Lightweight Directory Access Protocol (LDAP) directory. Built-in users are able to log in to any application within the same workspace that uses APEX account credentials.

For small projects with a single developer, it is quite possible to perform all application development as the workspace administrator. However, for any development that uses two or more developers, it's best to create a specific developer account for each physical developer, since this lets you use features such as page-locking, as well as track changes to the application at the developer level.

Although the workspace administrator could be one of the physical developers, a better idea is to create a developer account to use for development. Use the workspace administrator account only when necessary to perform administration duties.

Workspaces and Schemas

When you create an application in APEX, you must select a schema as the default parsing schema. In other words, if you built a report that issued a query like this:

```
select empno, ename from emp;
```

the query would use the `emp` table in the schema you selected as the parsing schema when you created your application. If you wanted to access an object in a different schema, you could prefix the object name with the schema name, like this:

```
select empno, ename from payroll.emp;
```

Because the application will be executed using the parsing schema (and its privileges), accessing objects in other schemas requires that the appropriate privileges are granted to the parsing schema. Objects in other schemas can also be accessed via synonyms or a view, which effectively hides the schema and enables you to reference the object without needing to specify the schema name yourself.

Choosing a Parsing Schema

The schemas assigned to the workspace you are currently logged into define the choice of schemas that can be used as the parsing schema. When you create a workspace (as an APEX administrator), you must specify whether to use an existing schema or create a new one, as shown in Figure 1-1. If no other schemas are assigned to the workspace, you will be able to select only this schema as the parsing schema when you create your application.

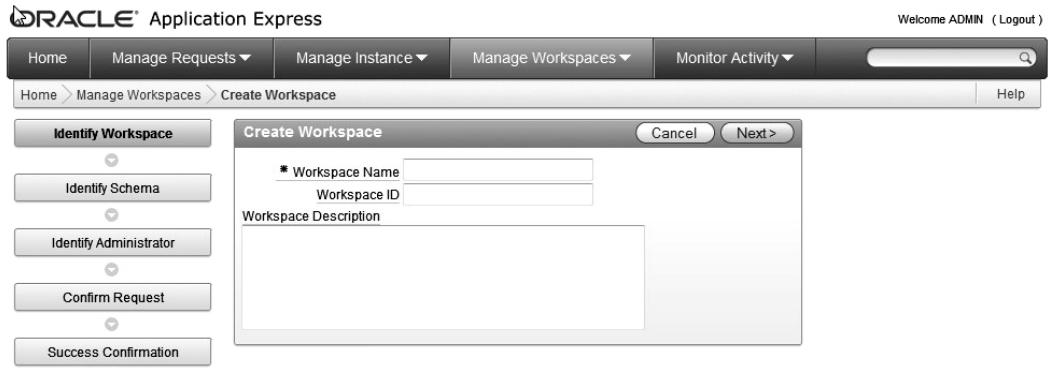

Figure 1-1. *Creating a new workspace*

This means that if you already have an existing schema with a lot of objects you'd like to access, you can select that schema. Then, any applications that are created within the schema will be able to access those schema objects directly. This way, you can create an application in APEX that provides a front end to existing data very quickly.

Although you can select only a single schema during the provisioning of the workspace, extra schemas can be assigned to the schema later on. After these additional schemas have been assigned to the workspace, they are available to workspace developers to use as the default parsing schema when they create an application within that workspace.

If you choose to create a new schema during the provisioning of the workspace, a new tablespace and corresponding datafile will be created for that schema automatically. The disadvantage is that the tablespace and datafile will have nondescriptive names, such as `APEX_1400423609989676` and `APEX_1400423609989676.dbf`. Moreover, if you later decide to remove the workspace, that tablespace and datafile will not be deleted. If you regularly provision and delete a lot of workspaces, you can end up with

many tablespaces and datafiles cluttering up your disk (and perhaps being unnecessarily included in your backups).

For small developments or evaluation, it may be fine to create a new schema through the APEX wizard. However, from a maintenance point of view, this approach often increases the difficulty in correlating schemas, tablespaces, datafiles, and workspaces because of the nondescriptive names. While this may not be a primary concern to you as a developer, it can be critical to how quickly the DBA is able to restore your schema from a backup if necessary.

Generally, for larger developments, if you are not using an existing schema, you may find it beneficial to manually create the tablespace and schema yourself, using a tool such as Enterprise Manager. For example, you can create a tablespace called `APEXDEMO`, which has a single datafile named `APEXDEMO01.dbf` that's allowed to grow to 2GB. You can then create a user `APEXDEMO` that has the `APEXDEMO` tablespace as its default tablespace. Figure 1-2 shows how the schema would look after being created in Enterprise Manager.

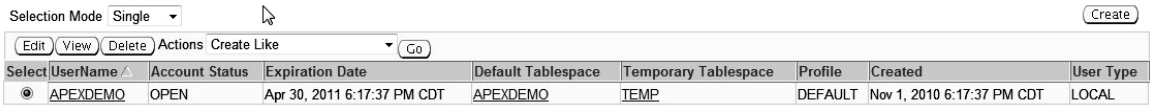

Figure 1-2. *Creating a schema in Enterprise Manager*

You could now create a workspace named `APEXDEMO` and select the `APEXDEMO` schema you just created in Enterprise Manager, as shown in Figure 1-3. This naming scheme ties together your workspace with the underlying schema and related tablespace and datafiles. If you should accidentally drop some tables (forgetting for the moment about the recycle bin in Oracle Database 10*g* / 11*g*), you can use RMAN to recover them easily, since their schema and tablespace will be obvious.

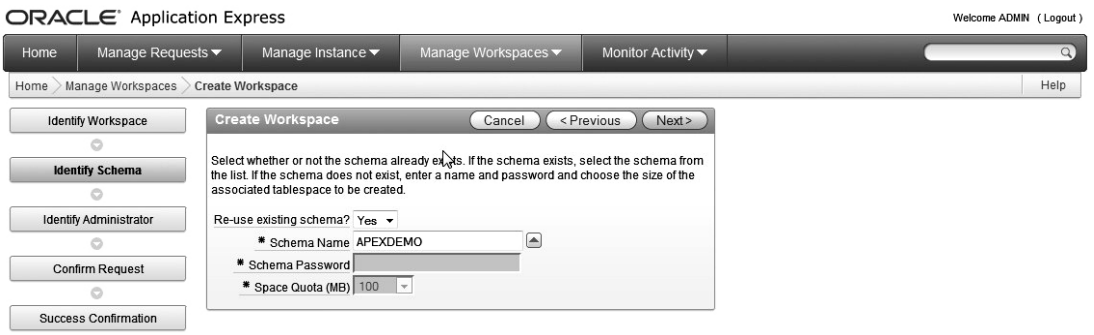

Figure 1-3. *Creating a workspace using an existing schema*

Although the APEX administrator can view reports that show which schemas and tablespaces particular workspaces are using, adopting a sensible naming convention makes it easier to get this information. For example, the DBA could look at a tablespace called `APEXDEMO` and be able to understand the purpose of that tablespace, which would not be clear from a generic tablespace name like `APEX_1400423609989676`.

■ **Note** Naming and coding standards can be extremely subjective. For example, some people may prefer to name the tablespace as `APEXDEMO_TS` while others prefer `TS_APEXDEMO`. If you already have an existing policy that details how you should name database objects, it makes sense to adopt that same policy for your development with APEX. If you do not currently have a policy in place, you should consider adopting one. The standards policy you use should be detailed enough to aid you in your work, but not so draconian that it actually hinders you.

Once the workspace is provisioned, additional schemas can be assigned to it. For example, you can create an `APEXDEMO_TEST` schema in Enterprise Manager, log in as the Application Express instance administrator, choose Manage Workspaces↗Manage Workspace to Schema Assignments, and select that schema, as shown in Figure 1-4.

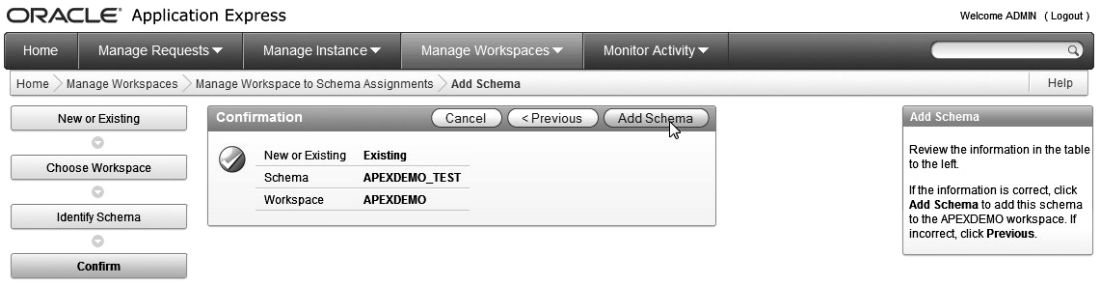

Figure 1-4. *Adding a schema to a workspace*

Controlling Access to New Schemas

Now the workspace administrator can specify which application developers can use the new schema (or, indeed, any of the assigned schemas). Figure 1-5 shows an example of a new developer account being created. By default, the developer will be able to access both schemas (`APEXDEMO` and `APEXDEMO_TEST`), since both schemas have been assigned to this workspace.

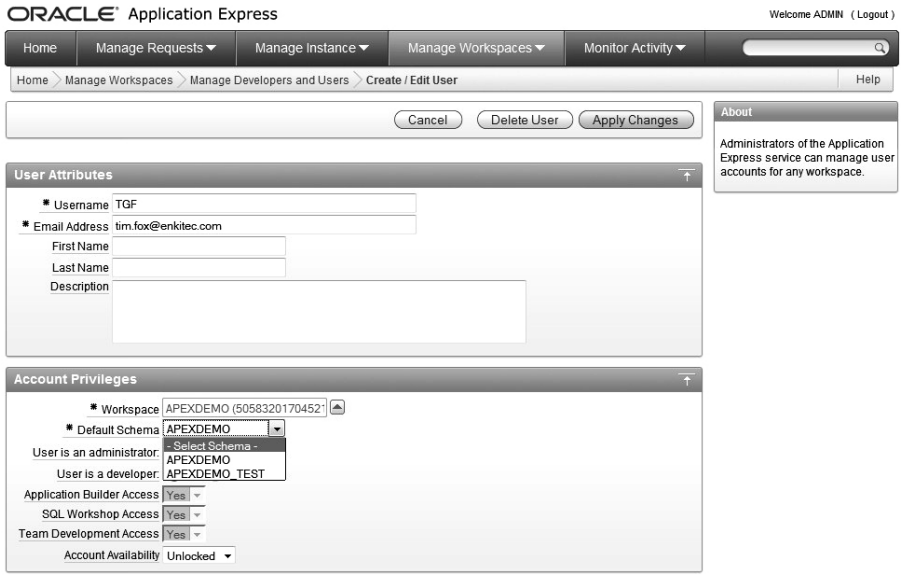

Figure 1-5. *Creating a new developer with access to all assigned schemas*

When this developer logs into the workspace, he or she can see the list of schemas that are available by clicking on the Administration↗Manage Instance menus, as shown in Figure 1-6. Any application this developer creates can use any of the available schemas as its default parsing schema, as shown in Figure 1-7.

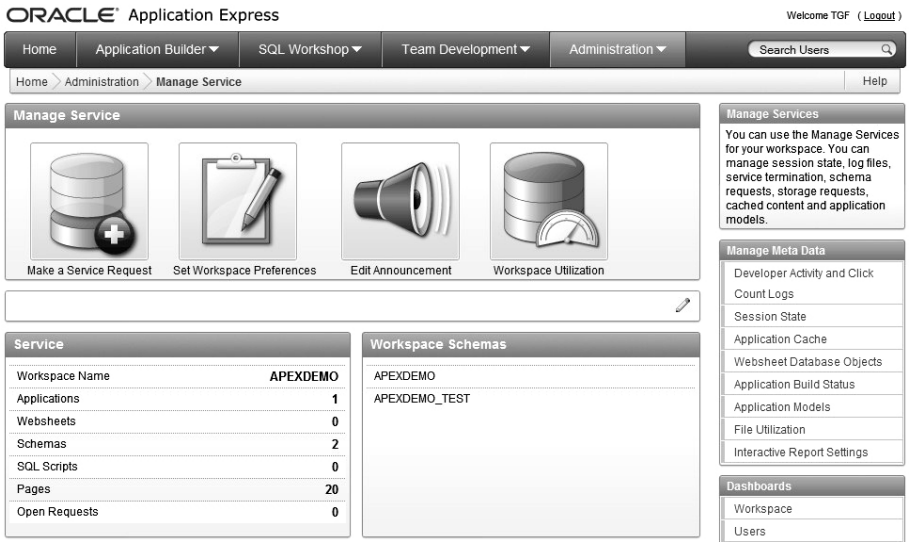

Figure 1-6. *Schemas available to the developer*

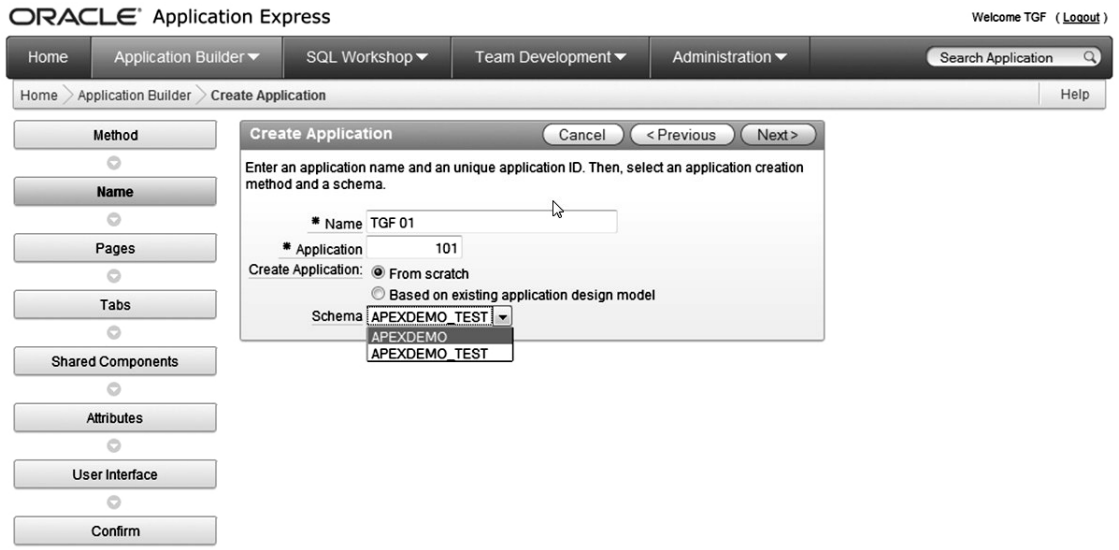

Figure 1-7. *Selecting a default parsing schema for an application*