
DATALINE SOFTWARE

Getting Started with i5



Suite 6 • Clarence House • 30-31 North Street • Brighton • BN1 1EB UK

Phone 01273 324939

enquiries@dataline.co.uk • www.dataline.co.uk

Contents

1. An introduction to i5	5
What is i5 ?	5
Application organisation	5
What is in this guide	6
2. Installing and running i5	7
Pre-Installation Considerations	7
Software and hardware requirements	8
Installation	10
Starting i5	17
Running i5: Some key issues	18
Using i5 with databases	19
3. Components & features of i5	20
Main components	20
i5 Studio	21
Page properties	24
Wizards	25
SQL wizards	28
Component Assistant	31
Other Features in i5 Studio	32
i5 Runtime	35
i5 Object Engine	35
Fitting it together – the underlying architecture	36
4. Building a single-table i5 book	38
Building a single section book	38
Previewing Books	43
Behind the scenes	44
Summary	46
5. Building a multi-section i5 book	47
Creating a new multi-section book	47
Formatting pages	48
More page formatting	51
Modifying SQL statements	52
Even more page modifications	53
Testing the book	54
Formatting the browse table	54
Formatting the detail page	58
6. Using page links	60
Single field value in a page link	60
Testing the page link	62
Creating a list box search page link	63
7. Child tables and groups	66
Creating a child table in a detail page	66

Editing and saving data in a child table	69
Using groups	73
Summary	75
8. Frame Rendering	76
Creating a basic frame layout	76
Putting pages into frames	78
Linking to pages in different frames	79
Formatting frames	81
9. Case Study - i5demo	84
Before we start	84
i5demo site overview	85
Product searching and viewing	86
The Wish List	91
Conclusion	93

Trademarks

i5 is a registered trademark of Dataline Software Limited (Dataline).

Centura, SQLBase, Quasar, SQLTalk and Team Developer are trademarks of Centura Software Corporation and may be registered in the United States of America and/or other countries. SQLWindows is a registered trademark used and licensed by Centura Software Corporation.

Adobe is a trademark of Adobe Systems Incorporated.

Apache Web Server is a trademark of Apache Group. This product includes software developed by the Apache Group for use in the Apache HTTP server project, <http://www.apache.org>.

“IT IS EXPRESSLY AGREED THAT DATALINE SHALL NOT BE IN ANY WAY RESPONSIBLE FOR THE COMMERCIAL SUCCESS OF THE APACHE WEB SERVER PRODUCT. DATALINE OFFERS THE APACHE WEB SERVER PRODUCT “AS IS” WITHOUT WARRANTIES OF ANY KIND , AND DATALINE SHALL HAVE NO LIABILITY OR RESPONSIBILITY FOR DAMAGES OR LOSS RESULTING FROM OR IN ANY WAY CONNECTED WITH THE APACHE WEB SERVER PRODUCT.”

Java is a trademark of Sun Microsystems Inc.

C#, Microsoft, Internet Explorer, Internet Information Server, SQL Server, Visual Basic , Visual Studio, Visual Studio .NET, Win32, Windows, Windows NT, Windows XP and Windows Vista are registered trademarks or trademarks of Microsoft Corporation in the USA and/or other countries.

All other products or service names mentioned herein are trademarks or registered trademarks of their respective owners.

Copyright

Copyright © 2009 by Dataline Software Limited. All rights reserved. July 2009

1. An introduction to i5

What is i5 ?

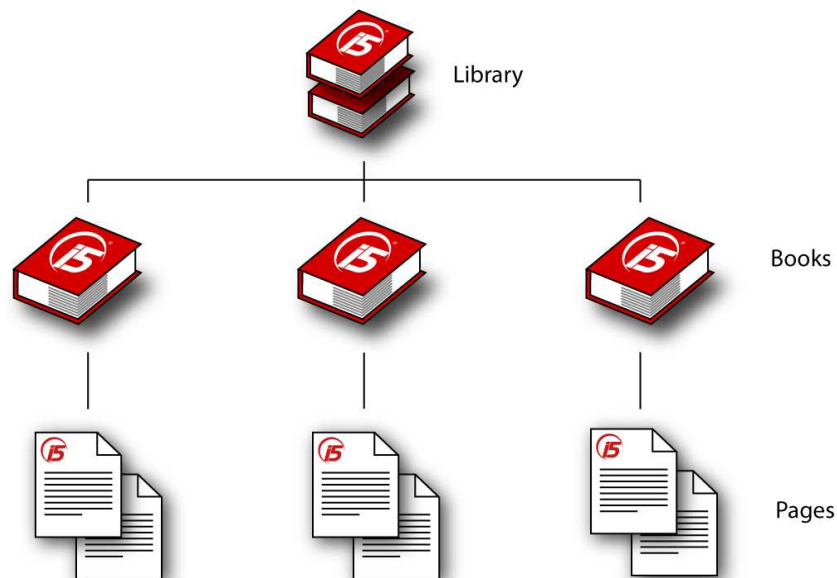
i5 is a rapid development e-business tool for creating robust, data-driven web sites simply and quickly. It is an extension of the net.db rapid development tool first released by Dataline Software in 1998 and marks a considerable upgrade and refinement of the functionality and potential associated with it. Having been through several interim versions, i5 3.5 features full compatibility with Microsoft's .Net framework.

From its inception, i5 was designed with five inter-related software development and deployment needs to the fore:

- The *interoperability* of an object engine which works on a number of platforms and with a range of database management systems;
- The ease of *interfacing* deriving from the use of XML as a markup language to package and distribute data
- A simple mechanism for the *implementation* of business rules encoded in business logic and incorporated into applications by a Business Logic Server;
- The capacity to maximise the *integration* possibilities that a wealth of application components in a Component Developer Kit can offer;
- The *immediacy* of solution deployment using a Component Library.

Application organisation

At its simplest, an i5 application provides read-only access to data stored in a relational database, including images, charts, and data. In its more advanced guises, an i5 application allows the user to create interactive web pages where end-users can query, view, create, update, and graph data through a web browser. Applications are organised as *books*, made up of *pages* that are held in *libraries*:



i5 terminology thus reflects that of print-based publishing. In the world of i5, the following terms are the most important ones used and are repeatedly used in this guide and its companions:

Library	This is a collection of i5 books associated with a single database. These books are categorised according to whether they are in development, are in a phase of testing, or have been released. They are displayed in the i5 Designer when the user logs on to that database.
Book	This is a collection of HTML pages and is the basic unit for an i5 application. i5 books are organized into sections. Books are stored in the i5 directory as obf files. In i5, the obf suffix denotes an Object Book File.
Section	This is a collection of HTML pages normally associated with a database table although there are occasions when this is not the case. Typically, although again not always, a section will include a Search, a Browse and a Detail page.
Page	A page is a collection of fields and components such as data fields, list & combo boxes, page links, graphs, HTML. Typically the data fields and components on a page correspond to particular sets of information contained in tables in a database. Indeed, the tools provided as part of the i5 Studio application assume this relationship as the default.

What is in this guide

As its name implies this guide has been designed to get users up and running with i5 as quickly as possible, and the aim is to give you a sense of how quick and easy i5 is to use. It covers everything from what hardware and software you need to run i5, how to install it, how it all fits together and what you can do with it. You may find this surprising, but you will not have to write any computer code.

If you are already familiar with the basics of website design (what HTML is, how hyperlinks work and so on) this should be enough to get you building some fairly complex and dynamic data-driven websites.

After going through hardware and software prerequisites, the installation process and the basic features of i5, the guide will show in more detail how to build single- and multi-table web sites using i5 and how its various components work. To round off, you will be given a tour through the demonstration website which comes with i5.

Our aim is to provide you with not just an introduction to i5 but to instill an idea of its technical possibilities.

If you think the above seems rather basic and below your abilities, or if you have finished working through this guide and wish to know more, there is a companion *Developer's Guide* available from the Dataline website. This moves on to introduce some of the more advanced features of i5 and in particular, the art of writing and including business logic to add sophistication to your applications.

2. Installing and running i5

This chapter runs through what has to be done to get i5 installed and up and running. The hardware and software requirements for i5 are defined and the installation process explained. Finally, the key issues to do with actually running i5 are examined.

The i5 server components do not need to be installed on the same machine as the i5 web components. If more than one web server is available, i5's web components can be installed on each of them and configured to use the same i5 Object Engine.

If you decide to install i5 as a Windows service, a valid (server) username and password will be required during installation. Also, if you intend to install i5 as a service, it is recommended that you install MSDE (or SQL Server) similarly.

Pre-Installation Considerations

Before installing i5 Studio there are a number of hardware and software requirements that need to be fulfilled in order to ensure that it will install and run correctly on your equipment. By current standards, these are minimal, partly because of the design ethos behind i5 - that of maximising its interoperability - and partly as a result of the efficiency of the i5 architecture itself. For a comprehensive development environment, it has a comparatively small footprint.

Operating system	Windows NT4, 2000, XP , Vista, 7, Windows Server 2003, 2008
Web browser	i5 is compatible with web browsers on any platform. Microsoft Internet Explorer 3.0.2 or higher, Netscape Navigator 4.72 or higher are recommended, as are Firefox, Safari and Chrome. The only special requirements that i5 makes on browsers is that the browser supports JavaScript (which is used to validate data displayed on a page) and preferably that the browser supports frames. Dataline test i5 Studio applications on all generally available browsers.
Web server	i5 requires a standard web server, such as Microsoft Internet Information Server 3.0 or higher, Netscape FastTrack 2.0 or higher, or Apache Web Server 1.3.2. The web server must support CGI (Common Gateway Interface) or ISAPI. i5 works equally well with all of these web servers so it probably doesn't make too much difference which one you use. i5 will install Apache for you if you ask it to but you should remember that as a rule of thumb it is not a great idea to run more than one web server on your machine.
Database server	Microsoft SQL Server 6.x. or higher via ODBC 3.70.06.23 or above. Setup gives you the option of installing MSDE automatically during the i5 installation process. MSDE is Microsoft's embedded database and provides a subset of the functionality compared to the standard SQL Server product. Access (Office 97) via Microsoft ODBC 4.00.35.13.00 or above. SQLBase 6.1.1 or higher. Oracle 7.x or higher via ODBC 3.10.00.00 or above. Pervasive 7.x or higher via Pervasive ODBC 2.53.03.00 or above (configured to use Scalable SQL Mode).

	Sybase System 11 or higher via InterSolv ODBC 3.10.00.00 or above. MySQL 3.x or higher via MySQL ODBC 2.50 or above. IBM DB2 7.x or higher via IBM DB2 ODBC 7.01.00.40 or above.
--	--

The requirements information presented here will, if met, be sufficient to allow you to install i5 on your system. The *i5 Developer's Guide* can provide you with more detailed technical information about the support that i5 Studio provides for particular database management systems as well as specific information for running i5 in large multi-user setups.

Software and hardware requirements

Software

Before you install i5, please note the following software requirements:

You should ensure that the SQL.INI file, which stores the configuration for details for your database server is set up with the appropriate entries and keyword values.

If you are unsure about the configuration of your database server, consult your system administrator.

Hardware

Before you install i5, please note the following hardware requirements:

Memory	512 MB minimum RAM. A minimum of 1 GB RAM is recommended for deployment
Storage	50 to 100 MB disk space depending upon software configuration

Directory organisation

i5 Studio requires a particular organisation of the directories which contain the components that it uses. These components include:

.EXE files	These actually do the bulk of the work required to allow the user to build and run applications and are stored in the main i5 directory
.OBF files	These contain the data for specific i5 books and are stored in the \bin subdirectory. These can be copied, moved, and stored as any standard text file but for a book to be rendered by i5, the relevant .obf file must be in the \bin subdirectory;
HTML files	These contain colour and graphic information which i5 uses as style templates to define the rendered output. These files are stored in the \include subdirectory (see below <i>HTML and GIF files</i>)
Other files	There are various other kinds of files that i5 uses. Perhaps the most important are the .dll files that are held in the \lib directory. These contain the business logic that are used for various parts of i5's operation. We will look at business logic in the <i>i5 Developer's Guide</i> if getting started with i5 whets your appetite sufficiently.

The relationship between the different i5 directories/sub-directories is as shown in the diagram below. The structure is created automatically when i5 Studio is installed. It is not a good idea to change the structure of the directories as this can confuse the software. Where you choose to

install the software, however, is a slightly less constrained issue. The default home directory for i5 is `C:\dataline\i5`. If you use the default installation option, all software will be installed in this directory, following the structure illustrated here.

If you choose to install i5 in a directory other than `C:\dataline\i5`, be sure to install future server software products into that directory. This prevents the installation of multiple copies of common files (which can cause version mismatch errors).

Configuring your web server

The web server is an important component in the proper functioning of i5 as it is the device which actually serves up the HTML-encoded pages which make up an i5 book. The way i5 is set up on installation is partly dependent on the web server that you are using, so when you run the installation program a prompt will ask you to indicate which web server you are using and will then install i5 accordingly.

CGI directory

i5 includes two CGI (Common Gateway Interface) programs:

- Page Designer (`i5studio.exe`) and
- Page Viewer (`i5.exe`).

CGI is simply a set of rules which define how a web server talks to another piece of software installed on the same machine. CGI is critically important for many dynamic web applications because it is effectively what allows a web server to interact with users and it does this by using *environment variables* to pass information to the server. When installing i5, you must specify a directory that contains only CGI programs.

Web Servers

The only web servers that the i5 installer will detect are IIS and Apache. For both IIS and Apache it will identify all the available executable directories and allow the user to pick from a drop down list. If the directory that the user believes to be correct is not in the list it can be typed in manually. Once the directory path has been established the next screen will display the web address path that is associated e.g. `C:\inetpub\scripts` is usually associated with `/scripts/`. If no associated path is found the default is `\scripts\` but the user can type anything in and it will be created (IIS and Apache only). In the case of other web servers manual entry is all that is available.

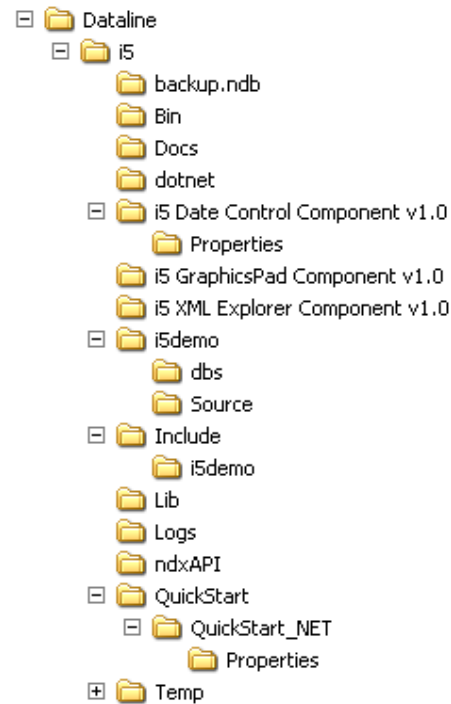
```
C:\InetPub\scripts
```

HTML & GIF files

i5 uses a number of HTML and GIF files. These files are stored in different directories depending on the web server you are running. The i5 installer automatically installs the image files in the correct directory.

i5 HTML files

If you want to customize i5 with your own HTML files, they must be located in:



`\Installation-directory\include`

to appear in the HTML template file and HTML footer file fields on the Page Properties formatting page.

Image files used within HTML scripts do not need to be stored in a particular location, since they are called by their relative path within those scripts.

Upgrading from previous versions

If you are upgrading from net.db, it is recommended that you backup the bin directory (the directory containing your .obf book files) prior to installing i5 over the top of a previous version or net.db.

Installation

Installation Options

To install *i5 Studio* you need to run the *i5* Installer program. You can do this simply by locating and running the executable file you have downloaded (setup.exe) and double-click on it to run.

When the installer first starts running, a *Welcome dialog* will be displayed. How the installation process will run from this point on depends on both your installation requirements and the operating system onto which you are installing i5.

There are two basic types of setup, namely:

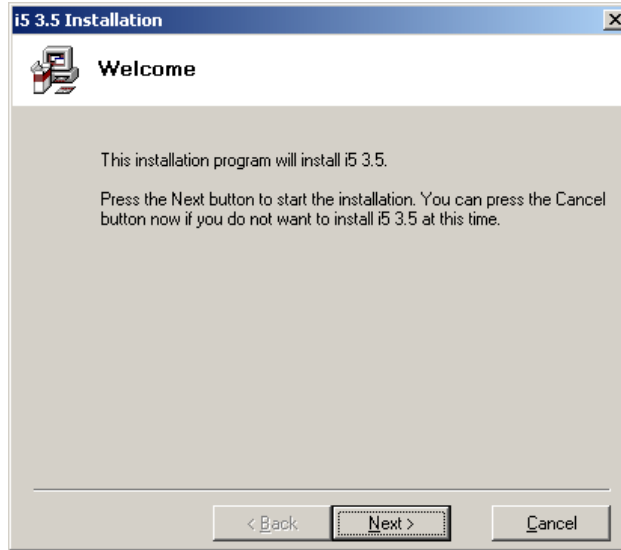
- Express Setup** Select this option if you are a new user and want to install all of the i5 components. The setup checks your system and detects if an existing web server is in operation. If no web server is detected on your machine, the Installer program will provide the option of installing the Apache Web Server. It will also automatically install MSDE, a free, easy-to-use, lightweight, and embeddable version of Microsoft's SQL Server 2005 RDBS.
- Custom Setup** Select this option if you wish to customize your installation, particularly if you already have a web server installed and/or are only interested in installing specific i5 components. This setup will give you the option of installing MSDE if you do not have an existing SQL database installed. It will also provide you with the option of installing the Apache Web Server.

Let's walk through the processes involved in installing i5. We'll look at Express installation on a step-by-step basis..

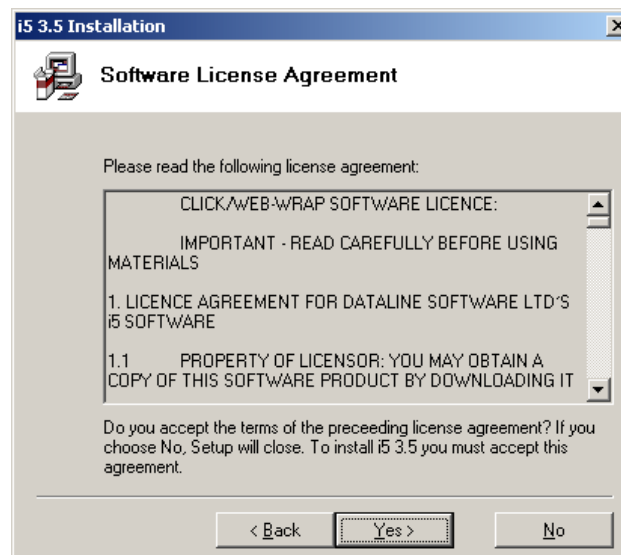
Express workstation setup

The following steps are typical for the Express setup:

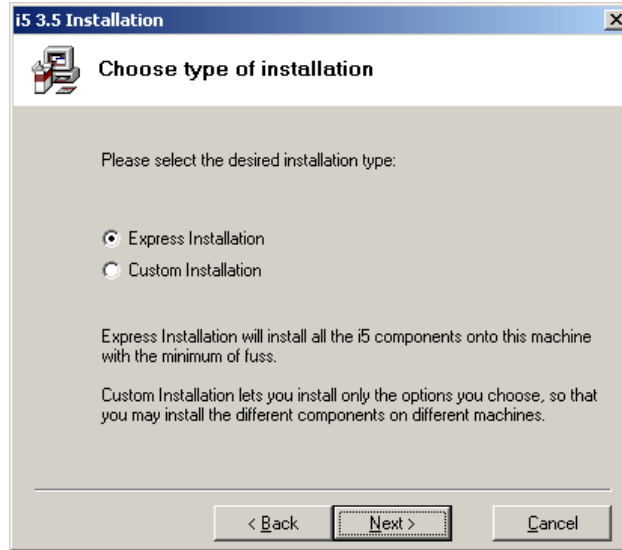
1. The Welcome dialog displays basic information. Click Next to continue.



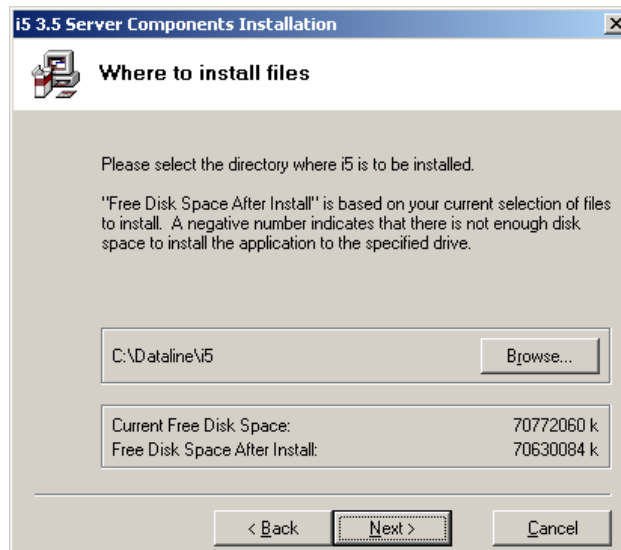
2. The Software License Agreement dialog is displayed. Click Yes to indicate acceptance of the license agreement and click Next to continue.



3. The Select Installation Type is displayed. Select Express and click Next.



4. The installer will recommend a directory for you to install i5 in. By default this is C:\Dataline\i5. The installer will also tell you how much of the free memory space on your computer the installation will take up



5. You will then be prompted to enter your user name and company details. The license key ID will automatically be displayed. Click Next to continue.

i5 3.5 Installation

User information

Please enter your information below.

User Name

Company

If you have a License Key, please enter it below. Otherwise, accepting the default License Key will result in a five-user i5 installation

You may enter a new License Key at any time, by using the i5 Server Components Configuration Utility.

License Key

< Back Next > Cancel

- The Server Port dialogue is displayed and you will be asked to provide an installation name. The default port number is 6670 but if this is already being used, the number of the next higher free port will be displayed. Alternatively, you can select a port if required. Click Next to continue.

i5 3.5 Server Components Installation

Server configuration

Please specify a unique name for this installation of i5. This could be the name of the customer or department who owns this installation (SALES, ENGINEERING, etc.). Please choose carefully; you'll not be able to change this later.

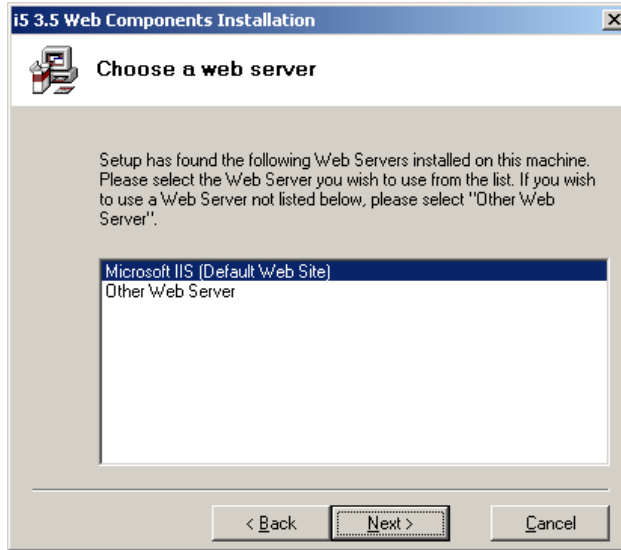
Installation Name:

Please enter the port number that you want the i5 Server to listen on. It is recommended that you use the default suggested below.

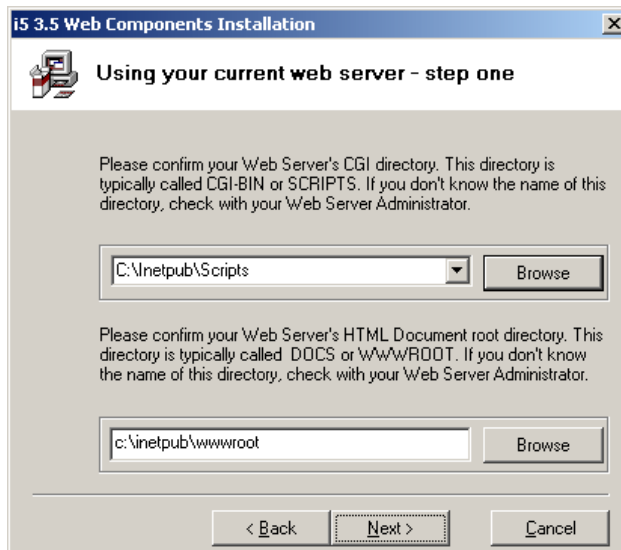
Server Port Number:

< Back Next > Cancel

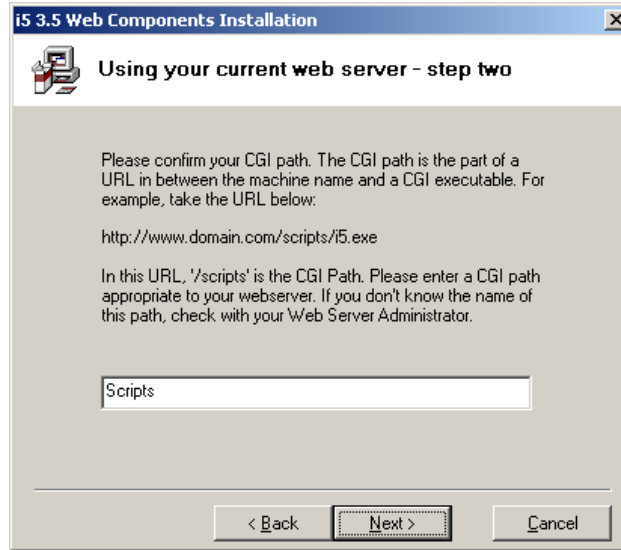
- At this point, the installer will check to see if you have a web server/s installed on your machine. If you do it will invite you to select the web server that you wish i5 to use from the list. If it doesn't find one, it will offer to install Apache for you. There will be some supplementary steps in the installation process if this is the case.



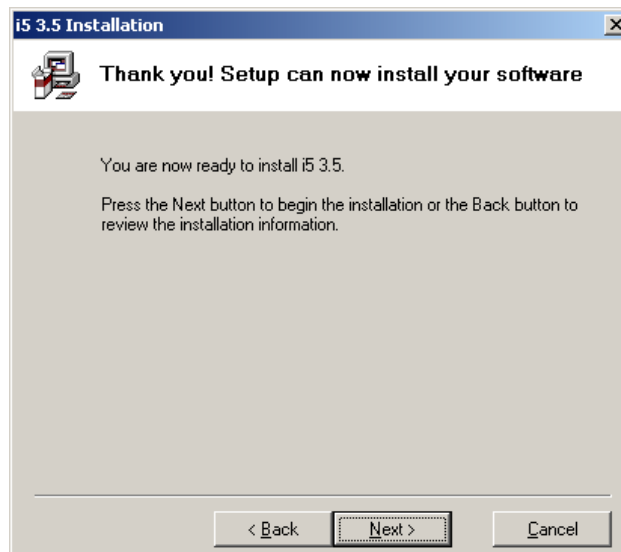
8. Once you have selected your web server, you will need to tell i5 where to put the CGI scripts and the HTML documents that it uses. If you have a CGI directory or a root HTML document directory set up it will ask you to confirm that you want to use them. If you don't, you can, at this point tell i5 to use some other directory or directories.



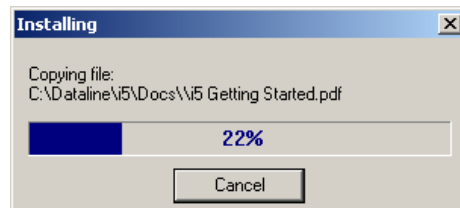
9. You will then need to confirm the CGI path for your web server. Typically this is called something like Scripts, as in the screen shot below.



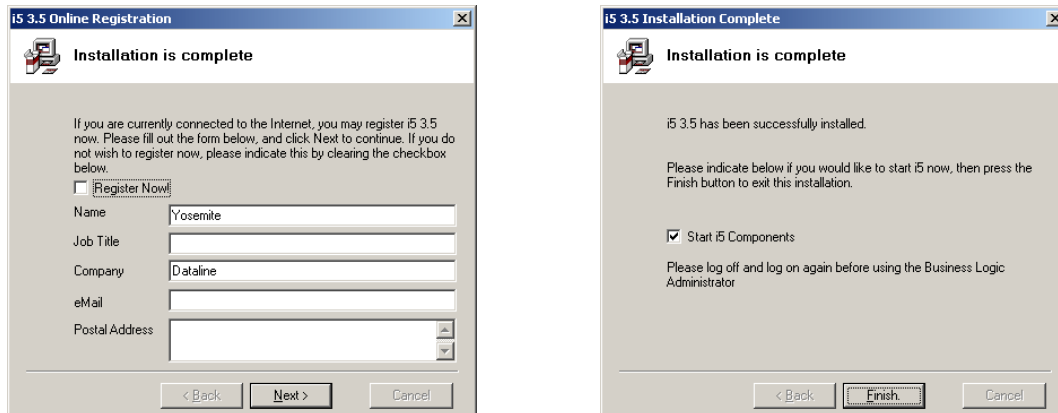
10. Assuming you have got this far without any hitches, i5 will now tell that it is ready to begin the installation process. Assuming that you do wish to go on with the installation, you should click 'Next' in order to complete the process.



11. At this point a loading bar will appear on the screen indicating how the installation is progressing. It doesn't take long – typically around five minutes or less.



12. Finally, if installation has been successful, you will be asked to register your details and then you will be taken to the 'Welcome to i5' start page:



Remember that the text on the installation dialogs and the installation steps will vary according to your selections and your machine's configuration.

On startup the following processes occur:

- Apache Web Server (if selected during the installation) is started;
- The MSDE database is started;
- The i5 Object Engine is started;
- The i5 Business Logic Server is started;
- Your browser opens to the Welcome to i5 page.

Custom workstation setup and troubleshooting

The installation process detailed above is for the Express installation. For various reasons, you may not wish to install every component of i5 on your machine. For example, you might have i5 installed on a server but wish to allow developers to have their own business logic servers running (particularly helpful when you have a team working on a project and each needs to be able to debug their code locally). Under these circumstances, the Custom installation process will allow you to select which i5 components you wish to install. We will not walk you through the process of performing a Custom installation here. It is pretty straightforward, and in all likelihood, if you are performing such an installation, the chances are you have already got i5 running somewhere else in your business, so you will be familiar with the process.

The installation process is pretty straightforward in general, but things can and sometimes do go wrong. If for any reason the installer cannot install i5, you will get an error message telling you that this is the case. Web server configuration issues in particular can be a problem. You must ensure that your web server has CGI enabled. For example, the Apache Web Server does not, although the Apache install as done by i5 will configure it. The typical way of CGI-enabling Apache is to use a `ScriptAlias` directive in the `httpd.conf` file. For more information refer to the documentation on the Apache website: <http://apache.org>. Users often forget to install IIS when they are doing afresh install of a Windows operating system, and this will cause the installer to fail.

If you are having problems, the best bet is to check on the i5 section of Dataline's bulletin board, which can be found at: <http://www.dataline.co.uk/forum/YaBB.pl>

Post-installation configuration utilities

There are three utilities that allow for post-installation configuration. These utilities will allow you to configure the i5 web server and Service settings, permission settings for designers using i5 and timeout settings. Details of how to use these utilities are provided in the *i5 Developer's Guide*. We're assuming here that you're happy with the default settings.

Starting i5

Once i5 is installed and configured to your satisfaction, you'll need to complete the following to start it running: Start your web server, the i5 Object Engine, and your database server, then select the [Getting Started with i5](#) option from your i5 Program Group on your Programs menu. The i5 main page will appear in your browser, assuming there are no problems.

It doesn't matter what order you do all these things in, although you won't be able to do anything from the *Getting Started with i5* screen if you haven't done everything else. Note also the following points:

- If you want to start the Object Engine in console mode so that you can review its configuration, you should issue the i5 console command from the command line. Alternatively, you can install it and run it as a service (this option is restricted to certain operating system setups - see below).
- By default, i5 installs the MSDE database, which is configured by the installer to start automatically when required. If you are not using a SQL Server database, you will need to ensure that your ODBC connectivity is configured correctly.
- i5 relies on the database for username and password authentication. As an additional layer of security, it also requires that the username you use to design pages with i5 is defined to the Page Designer.

Running i5: Some key issues

Starting & stopping the i5 service

Once the installation process is complete, i5 is set to run as a console application. The startup mode is set to manual when the Windows operating system initializes. Running i5 in console mode makes you responsible for starting and stopping the software (and you will be prompted to this effect each time you start up your machine, unless you elect to dismiss the alert). However, you may find it more convenient to run i5 as a service - if you are running i5 on Windows NT, 2000, XP, Vista or Server 200x you can do this by following the notes in the *i5 Developer's Guide* about running the Service Configuration utility. You can also do this from a DOS command prompt (see *Reinstalling i5 as a Service* below).

If i5 is installed as a service, you can administer the i5 Object Engine using the Windows Services module in the Control Panel. To start and stop the i5 when it is running as a service, select i5 from the Services list and click Stop (or Start).

Removing the i5 service

If for any reason you decide that you prefer the control that running i5 from the console offers, it is very easy to remove i5 as a service. To do so from a DOS command prompt, navigate to the i5 directory and type:

```
>i5server.exe remove
```

The following messages appear:

```
>Stopping i5.  
>i5 stopped.  
>i5 removed.
```

If you previously stopped the i5 service using the Services module of Control Panel, you will only receive the `>i5 removed` message. The service is now stopped and is de-registered with the Service Control Manager.

Reinstalling i5 as a service

If you have removed the i5 service and want to reinstall it manually you need to register the service with your computer's Service Control Manager. To do this, go to the DOS prompt, navigate to the i5 directory and type:

```
>i5server.exe install
```

To complete the installation, go to the Services module of Control Panel and start the service.

Running i5 as a console application

To run i5 as a console application under Windows, navigate to the i5 directory from the DOS prompt and type

```
>i5server.exe console
```

i5 is now running as a console application, with the same functionality as the service version. Diagnostic information is displayed when running i5 as a console application.

Uninstalling i5

Remove components by using the Control Panel Add/Remove Programs applet.

Security Issues

To run i5, you must have: i5 design-time access as specified by the Connectivity Administrator and stored in the registry, and 'connect' privileges to the database you are using.

The i5 Page Designer requires read (SELECT) access to system tables - minimally those which contain information about the following items:

- User tables
- Columns - both the attributes and the data within the column
- Indexes
- Primary and foreign keys
- System stored procedures which access these other objects

To maintain security between the browser and web server, use browsers and web servers that support industry-standard security mechanisms such as Secure HTTP (HTTPS) and Secure Sockets Layer (SSL). You should also use firewalls and third-party products to protect your data.

i5 registry keys

When i5 starts, it checks the i5 registry keys for configuration information. These registry keys are populated during installation. You do not need to edit the majority of the registry keys. Typically, the only two keys you need to change are `designerUsers` and `timeOut`. To edit these key settings, use the i5 Server Configuration utility as described in the *Developer's Guide*.

Using i5 with databases

About MSDE

MSDE is Microsoft's embedded database and provides a subset of the functionality supplied with the standard SQL Server products. If you do not have a previous installation of SQL Server, the i5 Custom Setup program gives you the option of installing MSDE. The Express Setup automatically installs MSDE if it does not detect an installed SQL Server. For the purposes of configuration, documentation, and performance in relation to i5, MSDE is essentially the same as SQL Server.

ODBC connected databases

The Page Designer requires that you specify a database name, username, and password for logging onto a database. Not all databases require that you have usernames and passwords for them. For example, some databases allow you to specify a username without requiring a password. If you wish to use i5 with a username that doesn't have a password specified, you will need to create a database password it. Or, of course, log on with a username that already has a password specified.

Some databases don't require usernames or passwords at all. In this case, you can enter anything you like in the username and password fields for the Page Designer.

In all cases, the usernames you use have to be defined to the i5 Page Designer. To do this, use the i5 Server Components Configuration utility as described in the *i5 Developer's Guide*.

3. Components & features of i5

Having got to this point in our *i5 Getting Started* guide, you are now ready for a tour of i5 that will give you an insight into what it can do - and more importantly what you can do with it. In this chapter we are going to take a look at some of the most important working features of i5. In particular, we are going to look at i5 Studio - the key point of entry for anybody planning to do any work with i5. When we've had a look at the key elements of the system, we will take a closer look at some aspects of its underlying architecture.

You don't really need to know anything about the underlying architecture of i5 to be able to work with it effectively. However, it's a good idea to have at least a basic understanding of how the different components fit together because in the long run this knowledge will assist you when you move on to building more advanced web applications.

Main components

The four most significant components of i5 from the developer's point of view are the:

- i5 Studio
- i5 Runtime
- i5 Object Engine
- i5 Business Logic Server

The operation of the first three of these is covered in this manual. The i5 Business Logic Server is dealt with in detail in the *i5 Developer's Guide*.

There are a number of ways that you can access the different components of i5. If you are running the components as a service, by far the easiest, at least initially, is to navigate using the Windows Start menu: Start > Programs > i5 > Documentation > Welcome to i5 v3.5

However, for the sake of completeness, we will assume here that you have not configured the i5 components to run as a service. This means that you will have to ensure manually that the various components of i5 are actually running before you can begin to do any development work.

Additional configuration notes

If you want to start the Object Engine in console mode so that you can review its configuration, you should run the `i5 console` command from a DOS command line. Alternatively, you can install it and run it as a service.

- By default, i5 installs the embedded database MSDE, which is configured by the installer to start automatically when required.
- If the Installer detected a local copy of SQL Server, you will need to ensure that it's correctly configured to start with i5 or it will be necessary to start it yourself.
- If you are using a database other than SQL Server/MSDE you need to ensure that your ODBC connectivity is configured correctly as described in Chapter 2.
- i5 relies on the database for username and password authentication. As an additional layer of security, it also requires that the username you use to design pages with i5 is defined in i5 Studio.

The opening i5 screen

Once you have selected the [Welcome to i5](#) option, the i5 Studio Getting Started screen will appear in your default web browser.



This page provides access to the key i5 components and other useful features.

Logging-on & server components configuration

Before you can access any of the core features of i5 you will need to log on. This is very easy provided you have the appropriate credentials – i.e. a valid user/password combination.

Credentials can be set using the Server Components Configuration tool, which, once again, can be accessed via your computer's main Windows Programs menu: Start > Programs > i5 > Server > Server Components > Configuration.

i5 Studio

i5 Studio is effectively a 'development environment' from within which you can create the pages that will together make up the book/s that forms the basis for an i5 web application. It is the focal point for all design time processes.

Logging-in

Select [Launch i5 Studio](#) from the Getting Started screen above. This will take you to the i5 log-in page which requires a log-in to a named database. Logging on will require the name of the database (as specified by the ODBC connection handle for this database – ask your database administrator if you are not sure about this), a legitimate user name and password.

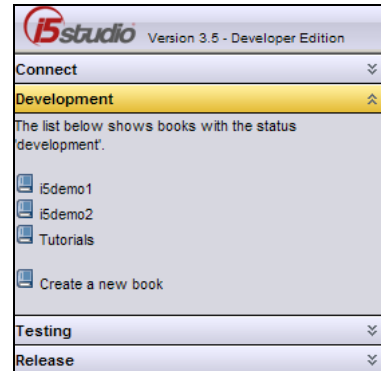
So, to log on to the *i5demo* database (which is installed by default) by entering the following values into the log-in box data fields and then clicking [Connect](#):

Database: i5demo
User Name: sa
Password: sa

A successful log-in will take you into *i5 Studio*

In the top left hand frame of the screen there are a number of tabbed options.

Connect	Clicking on this the tab returns users to the log-in page, where they can log-in to another database.
Development	Books that are currently under development are held and can be viewed.
Testing	Books that have been developed and are now undergoing testing prior to release are held and can be viewed.
Release	Books that have been deployed – i.e. gone live.



Development tab

The Development tab is the default option when i5 Studio is started and appears in its expanded state. Initially, there will not be any books under this tab but typically you would be presented with a series of named book icons as illustrated in the screenshot above.

When any of the books or the Create New Book option is clicked three items will appear in the left hand frame of the browser screen:

- A Toolbar in the upper part of the frame;
- The Tab view, immediately below the toolbar;
- And below that the navigation tool referred to here as the Explorer view (or, more typically, just the Explorer):

Toolbar

The toolbar displays a series of options for key development operations:



Quit - clicking on this will return you to the login page.



Open Book - clicking on this will open the book you have been working on in the Designer in i5 Runtime (ie it will run your application).



Wizards - clicking on this will display the wizards tab in the top left hand frame.



Business Logic Administrator Clicking on this button will open up the BLA in a separate browser. The BLA is discussed in detail in the i5 Developer's Guide.



Help - clicking on this will open the i5 Help for the homepage in the main right hand frame of the i5 Studio.



Show/Hide Preview - this option allows you to effectively "switch off" the previewing of pages in the right hand frame of the browser. The Show/Hide Preview button is situated n the far right of the toolbar.

Tab view

The Tab view of an i5 book displays a series of menu tabs which provide access to all the Book level properties available to you for any particular component of an i5 book at design time. These tabs are listed and described below:

- Properties** Used for setting the basic properties of a book - such as its name or its author. The status of a book can also be set – development, testing or released – from this tab and will cause the book to appear under a different book listing *Tab* view at login. This is the default tab.
- Formatting** Used to determine the overall formatting options for a book – the default text font or the default HTML template.
- Colors** The basic palette of colours to be used in a book.
- Advanced** Enables the setting of such things as the data-source to which the book is connected, whether the book will use cookies, etc.
- Templates** Allows the user to select and set which HTML template is to be used throughout a book.
- Components** As its name suggests, this tab is used when new components are to be added to a book.
- Comments** Provides a shared repository of comments from developers working together on a project. Very useful when working in a team environment on large books.



Explorer view

Beneath the Tab view, the i5 Explorer is displayed. The Explorer view provides a hierarchical representation of the components making up an i5 book. Clicking on any of one these components will trigger two responses:

1. The component's Properties tab is revealed in the Tab view;
2. The component (if it is a page), the default page (if the component is a section), - or the page on which the component is located (if the component is an object on a page) - is displayed in the main, right hand frame.

The Explorer view in i5 Studio displays a user interface with the familiar branching tree structure. The branches/leaves of the tree will display the component parts of the application. When an i5 library is opened the Tab view shows the books that have been created and when one of these is opened - by clicking on the title of the book - the Explorer view is populated with a list of that book's sections.

Clicking on the section node plus sign (+) in the Explorer view expands the node, displaying its child components.

The example Explorer view displayed in the screen shot above shows a book with one section expanded, this section in turn showing the key page types for any i5 application - *Search*, *Browse* and *Detail* pages. i5 offers you the option of creating such pages automatically whenever you elect to create a book section keyed into a particular table or set of tables in a database.

You are not absolutely obliged to have Search, Browse and Detail pages in every section that you create in an i5 book, but if you use the Book Wizard to create sections, it *will* add all three automatically. We will look at all of these section components in more detail in due course but the Search, Browse and Detail pages are used, respectively (and briefly) to:

- Insert the search criteria to be used for searching the database table any specific book section is based on;
- Display, in tabular format, the records that match the search criteria set in the search page above;
- Display the details of any particular record selected from the browse table above. This page also enables the user to modify an existing record in the database or to insert a new one.

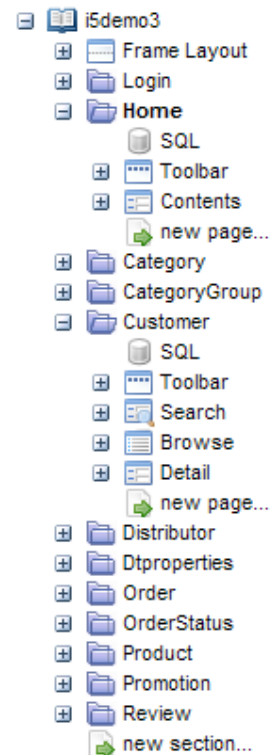
Each of these pages constitutes a view of the database table on which the section is based. As the above suggests, each view allows the user to perform a different action using the information in that page. Typically you would use a Search page to find records in a database table, a Browse page to look through the set of records returned from that search, and a Detail page to look at the detailed record selected using either search or browse. The difference between a Search page and a Browse page is that in the former you are simply provided with a set of possible search criteria to find one or more records, with the latter, you are given basic details of all the records returned from a search.

The Toolbar is important here because it contains a number of components that allow the user to navigate between the different parts of your book, to search for, update and/or delete records in a database.

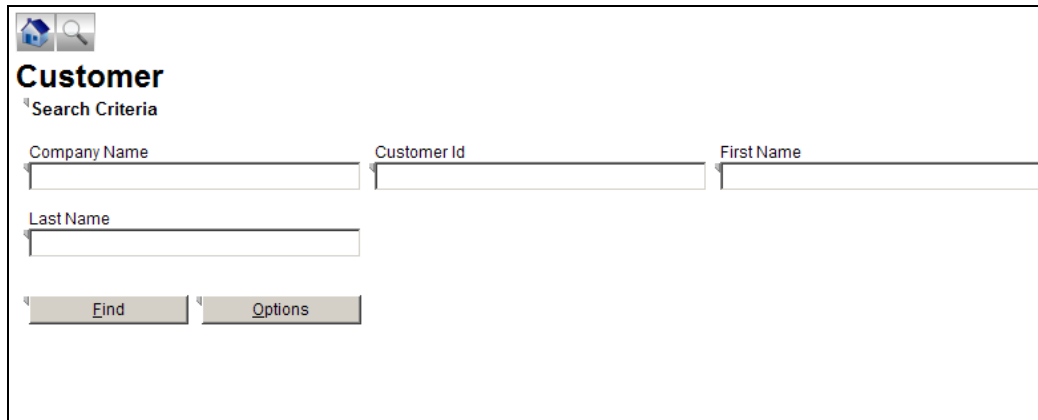
Page properties

When an i5 page is created it has a number of editable properties, and it is these that allow the user to customise many aspects of both its appearance and behaviour.

The rendering of page properties within a book can be viewed in the main frame of the Designer and edited using the Tab view of i5 Studio by clicking on the name of the page in the Explorer



view. Selecting a page in this way will automatically display its properties tab and the actual page itself in the right hand frame of the Designer. For example, clicking on the Customer Search page in the Explorer view shown above will display the page in the main frame.

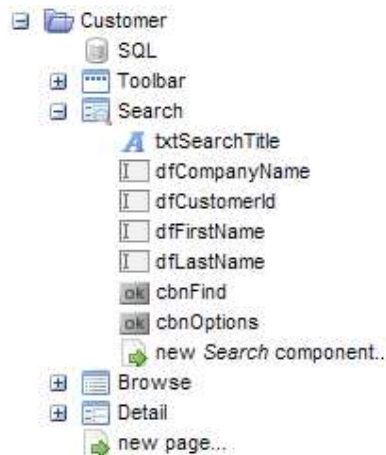


Expand the Search page by clicking the **+** adjacent to it in the Explorer view. The components that can be seen in the main frame – which make up that page – are now displayed in the Explorer view, namely:

- The page title - *txtSearchTitle*
- Four editable data fields – *dfCompanyName*, *dfCustomerId*, *dfFirstName*, *dfLastName*
- Two control buttons – *cbnFind* and *cbnOptions*

Notice also the 'new Search component' option. This is a link which, as the name suggests, allows the developer to select a new component to add to the page in question.

If you click on the small black arrow at the top left hand corner of any component displayed on the main page (in the right hand frame of the browser), that component's **Properties** tab will be displayed in the Tab view. Note too that when the mouse cursor is positioned over the top left hand corner of a component, its name and position - expressed in pixel distance from the top left hand corner of the page – are also displayed.



Wizards

Let's now look at some of the tools provided within i5 Studio, starting with its collection of wizards.

These wizards enable users to accomplish regular web application development tasks quickly and with minimal fuss. Each takes the user through a series of dialog boxes and can, for instance, model an entire data source or add and link i5 pages using just a few mouse clicks.

A number of wizards can be accessed by clicking the **Wizards** button on the main toolbar, the **Book**, **Page**, **Graph** and **Import** wizards.

Additionally, a series of Wizards can be accessed by clicking on the [SQL](#) icon within the Explorer view at section level: the [Join](#), [Query](#) and [Sort](#) wizards. These are designed to simplify the writing and editing of SQL queries used in an application.

Note that the Book, Page, Graph and Import wizards can also be accessed from the [New Section](#) menu item located in the Explorer view of i5 Studio:



Book wizard

The book wizard makes it possible for the user to build an i5 book for a selected data source with little more input than a single mouse click. The wizard exploits the referential integrity typical of a well designed database so as to build a separate section for each table in the database and provide links between pages in the resulting book. This wizard is the default option in the Tab view when the Wizards tool is selected.

Using the Book wizard is very simple. Select an HTML background template if required, and then click [Go](#) to build your book.

A timer appears and gives an indication of the wizard's progress and when the process is complete a report is displayed in the Tab view. This shows the sections that have been added to the book and the joins made between tables. The basic rules which the wizard uses in constructing a book are as follows:

- Sections** A section is created for every table in the database, providing the table has a primary key or, failing that, a unique index that can be used instead.
- Fields** Fields are added within the Search page, Browse page, and Detail page for each column in the table. Fields are defined as either Primary keys or search fields.
- Primary keys** All fields the data-source identifies as primary keys, or those fields that are constituent parts of a unique index, are added as non-editable primary key fields. All primary key fields and a maximum of five other columns of type char() or varchar() with length less than 20, are added as editable search fields.
 Primary key fields are added to the section first, followed by the remaining fields sorted by length as reported by the data source. Any remaining fields are added as editable fields.
- Page links** Page links are added to the pages in the books using referential integrity information. These page links allow users to navigate through related tables.

i5 supports the use of compound primary keys – see the i5 Developer's Guide for more details.

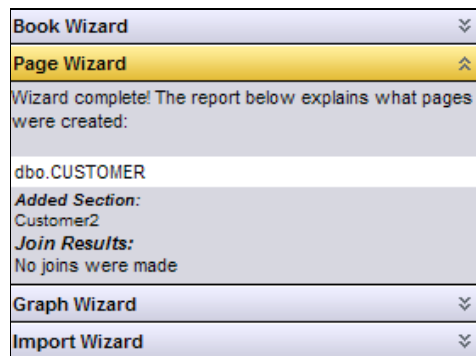
Page wizard

The rules which the Page wizard follows for generating pages of fields and the links between them are the same as for the Book wizard but it is essentially a more flexible version of the latter.

Rather than generating sections for every table in a database, it prompts the user to select the database tables – in i5 referred to as data-sources - from which to generate book sections. Minimally, the Page wizard allows the user to build a single section based upon a single database table.

To generate more than one section at a time, it is simply a matter of holding down the Ctrl key whilst selecting the multiple data-source tables required. Once the selection of tables has been made, click the Go button to start the process of generating the sections. As before, every section built using this wizard will comprise a Search page, a Browse page, a Detail page and a Toolbar.

Using the Page wizard, an i5 book can be built from scratch using only a subset of the tables available. It can also be used to add one or more wizard-generated sections to an existing book.



Graph wizard

The Graph wizard creates data-aware graphs with built-in hyperlinks to detail pages containing the data from which a graph has been constructed. There is no limit to the number of graphs which can be added to an i5 book and each one can be mapped either to a single database table or to multiple tables, providing that these tables are joined together first.

The Graph wizard does, however, require more user input than the wizards previously described and the dialog will prompt - in turn - for the following information:

- Name** This identifies the graph using a unique name and must contain alpha-numeric characters only. i5 does not actually display the name of the graph, it serves instead as a reference when defining a link from a push button on another page.
- Title** This identifies the title of the graph within a book's Explorer.
- Data-source** This identifies the database table i5 is using to retrieve data.
- Type** This specifies the type of graph to be generated - supported types are: 3D Bar, 2D Bar, 3D Line, 2D Line, 3D Area, 2D Pie Chart, 3D Pie Chart.

The wizard dialog will further prompt for the following information – firstly for bar or line graphs:

- X-Axis labels taken from** This specifies the database column to use to label the X-axis.

Y-Axis values taken from This specifies the database column to use to label the Y-axis.

Or the following information in the case of a pie chart:

Labels taken from This specifies the database column to determine the number of segments of pie and their labels.

Values taken from This specifies which database column should be used to set the size of the pie segments.

When the wizard has been provided with the required information a click on the Finish button will prompt the wizard to build the graph.

The i5 wizard generates the graph images in either PNG or JPEG format depending upon the client browser.

Import wizard

The Import wizard allows users to import sections from other i5 books into the book that they are currently working on providing that the source book (or books) and the destination book are themselves based on a common datasource (ie database). This can be a very useful shortcut for combining work from one application with another.

To import a section from another book, the Import wizard basically needs to know two things and prompts the user to supply these pieces of information - in order:

- The name of the book from which you wish to import a section
- The name of the section you wish to import from that book

Once the import has taken place, i5 will provide you with a report to tell you if any page links in the book you are importing into have been broken.

If the book section being imported has a name that is identical with that of an existing section in the receiving book, the wizard dialog prompts the user to choose between overwriting the existing section and copying into a new section (leaving the existing one in place).

If the latter option is selected, the imported section will be automatically numbered to indicate that it is the second of two identically named sections.

SQL wizards

Although you don't need to be an expert in SQL to use i5, a basic understanding of SQL queries will make the going much easier when using and fine-tuning the SQL wizards.

SQL is an extremely powerful language, but it's not too difficult to get to grips with and if you wish to become familiar with its syntax, it is worth looking at one of the many available books and online tutorials that cover this subject. The *w3schools* web-based tutorial below is particularly clear and a good starting point: if you are tackling SQL for the first time: <http://www.w3schools.com/sql/default.asp>

The i5 SQL wizards, all of which are accessed by clicking the SQL icon within any section in an Explorer view, are designed to make it easier to define the kinds of operations the application is to carry out on the underlying database.

Join wizard

As its name implies, the Join wizard performs SQL Join operations on tables within a database. A join is what SQL databases use to link secondary database tables to a primary table so that fields from more than one table can be addressed by the same query. In i5 joins are used to allow you to display data from more than one table on the same page.

In order to join two tables, they must have one column in common as without this there is no consistent criterion on which to base the join between them.

The Join wizard prompts for the following information:

Join to table	Identifies the database table to which a join is to be made.
Aliased to	Identifies the table by an alias name, instead of its table name.
From column	Specifies the database column from the primary table which i5 uses to match the result set.
To column	Specifies the database column from the secondary table (the table joined to) used to match the result set.
Outer join	If selected, this allows the non-participating rows of one of the joined tables to be included in the result set. These rows are termed non-participating because they contain key values that are not referenced by any rows of the other table.

Different database management systems have differing policies regarding joins. Some, for example, allow only one outer join per query, and others can be configured to conform to the Oracle outer join standard or the ANSI standard. Data is returned differently depending upon the configuration.

The SQL properties of any section in a book incorporate an *Advanced* tab. From this tab you can set the properties of any joins that you might wish to make on tables that you want to use in that book section. When using joins, the Join Criteria box can be used to provide join criteria manually and the following rules apply in this instance:

- The value you enter in the Join Criteria box must take the form of a SQL ‘where’ clause, minus the keyword ‘where’;
- None of the columns that comprise the join criteria need to have fields based upon them in the page;
- If the tables do not have a column in common, the join cannot be reliably attempted

Let’s look at what you would have to do if you wanted to join the Customer and Order tables from the i5demo database. Why would you want to do this? Well let’s just imagine that you wanted to allow a user to examine the orders for a specific company recorded within the i5demo database. The database makes this very easy for you to do as it provides a ‘foreign key’ to the Customer database table in the Order table.

Let’s assume you have a Customer section in the book you are working on. If you open up the SQL properties of that section and navigate to the Join wizard, a series of dialog boxes will ask you:

- If you wish to create a new join. You do, so click on Next
- To select the table which you wish to join to the Customer table. Select *dbo.order* from the Table drop down list box and then click Next
- How you want to join the tables together.

You need to have a basic understanding of the SQL *Join* concept to appreciate exactly what this last point means. But briefly, when you join two tables you need to find some criterion by which to join them. This means finding a data item – or column - which is common to both. In this case it will be the *Customer_ID* key value. Select *Customer_id* in both the From and To dropdown list boxes. Select Inner join as the Join type - this specifies that only the records in both tables which have the common value will be selected. As it happens the *Customer_ID* value is required for all order records – which makes sense, as orders have to belong to someone, but isn't always true that all records in all joined tables share the joining values. This is why SQL allows you to specify other types of joins if you need to. Save your changes.

At this point in our guide, how you use joined tables in practice might not be clear, but don't worry. The point of this section is only to show you how to join tables, not to use the tables thus joined. Using joins will become clearer later on.

Query wizard

The Query wizard allows you to build 'canned' queries which can be used instead of, or to augment, your search page. Canned queries effectively allow the user to specify the 'where' clause of an SQL Select statement that is then used to retrieve data to populate the pages of i5 book sections.

To build a query using the Query wizard, you should

1. Click on the SQL icon in the relevant section in the Explorer view and select the Query wizard tab
2. Specify: Data-source column e.g. Price from a Product table.

Operator e.g. Is equal to, less than, more than etc.

Value (in this case numeric).

Note that you are not restricted to matching records on only one data-source column, because

3. Clicking More you can add other data-source columns, with other operators and values. This allows you a relatively fine measure of control over how you filter the results of your query.

If you make an error, clicking the Less button will remove a row from your query.

Depending on your data-source, it may prove necessary to wrap quotes around any string (alpha-numeric) values that you use.

Once written, a canned query will filter the result set available to you to search on for that section. For example, if you had set a query on a Product section to filter results such that only products with a price of less than \$10 were selected, then any products in that table in the database with a price of more than \$10 will be filtered out of the result set which i5 uses to populate its Product pages.

You can easily verify that this is the case by adding a query to the Product section of the book that you created at the start of this chapter, filtering results on Product.Price and experimenting with the operator and price values - open up the Product.Browse page to check how the results change with a variation in the query values.

Sort wizard

This wizard is used simply to define the ordering of the results of searches made against the data-source. The wizard allows the definition of as many pairings of a data-source column and an ascending or descending order as you require, and the order in which they are applied will be dependent on the order in which they appear in your list.

For example, given a table called *Customers*, containing the details of customers in a database, you could put the Surname/Ascending and FirstName/Ascending pairings into a sorting clause. Any

results returned would then be ordered firstly by the customer's surname and then by their first name, so *Johnson, Lucy* would come before *Johnson, Magic*. In other words, the Sort wizard is the equivalent of an SQL 'Order By' clause to any result sets it retrieves from the database.

Using it is easy. To add a pairing to your sorting clause simply click [More](#) and then enter the appropriate values. To remove a pairing, click [Less](#). Clicking [Save Changes](#) does exactly that.

Advanced tab

As you use the different SQL wizards to build up a query for a book section, the information which you supply to each one of these wizards forms the basis of the statements in the different text fields of the Advanced tab. You can verify how i5 has constructed your SQL queries by checking the different clauses in the text fields of the Advanced tab. It is also possible to fine tune queries here as well.

Component Assistant

In addition to the wizards and the tools available from the navigation toolbar, i5 also provides a [Component Assistant](#) to make it easy to add a range of different objects to i5 books. The Component Assistant is context aware in the sense that the range of components it will allow you to create is dependent upon the place in your application you are trying to create them from. For example, at the book level you can add things like Sections; at the Section level you can add things like Search, Browse and Detail pages and at the Page level you can add a whole range of components: those that come as standard with i5 and those 'plug-in' components that you can make yourself (we discuss the development of plug-ins in the *i5 Developer's Guide*).

Data-field	A datafield is used to display and capture data and for this reason can be defined as editable or read-only
Browse table field	As its name suggests, a browse table field is a field which appears in a browse table (on a browse page). Generally, such a field has a read-only status because it is used to display some element of a result set which would then be edited from a detail page
List box	List boxes are typically used in search and detail pages and are populated with a list of data taken from the datasource to allow the user to set search criteria or to edit information on a detail page.
Check box	Check boxes are typically used when there are just two options – Yes and No are a good example.
Radio button group	Radio buttons are typically used when there are three or more – although not too many – options. Yes, No and Don't know is a good example.
Control button	These are used on all types of pages and for many functions/reasons. On a detail page, for example, control buttons are used to create new records, save records or delete records.
Image	These can be placed on any page of an application.
Page links	These are used on any page in order to navigate within a i5 book.
HTML	This can be inserted in any page.

Child table	This can be inserted into a detail page.
Child graph	As above, this can be inserted into a detail page (master/detail use)
Child report	This appears as a button or hyperlink and is typically used to generate a report that is automatically exported into, say, an Excel spreadsheet

The above list is not definitive as some of the more sophisticated components - the [Multi-Table Update](#) component is an example - have been omitted as they will not be required at this stage.

The function and use of some of the components listed above is intuitively obvious, but for others it is less so. How these components can be used will be explored in more detail further on in this guide.

There are two ways of accessing the Component Assistant:

Tab view	Books, Sections, Login pages, Content pages, Toolbars, Search, Browse and Detail pages and Child Tables all have a <i>Components</i> tab which when clicked will take you to the <i>Component Assistant</i> .
Explorer view	Alternatively, the <i>Component Assistant</i> appears in the Explorer under each of the above components, preceded by the <i>New Component</i> icon.

Other Features in i5 Studio

Before we launch into an in-depth demonstration of book-building using i5 Studio, we need to take a look at some of the other key features which allow you to control the appearance and behaviour of i5 applications that you might build.

Colours

i5 allows you to customize colours for objects, including text, non-editable field backgrounds, column header, and row header. To edit the default colour settings for an i5 book, select the Colours tab from the Tab view of the book with which you are working. This will display a set of eight pre-defined colours. These colours are:

<i>Default Text Color</i>	<i>Default Field Background Color</i>
<i>Default Table Background Color</i>	<i>Default Column Header Background Color</i>
<i>Default Primary Background Color</i>	<i>Default Secondary Background Color</i>
<i>Default Title Color</i>	<i>Default Frame Border Color</i>

To modify any of these default colours, click the Colour Palette icon in the colour column next to the item you want to edit. This will display the properties for that particular colour, and it allows you to modify the colour in one of two ways:

1. Enter an RGB value in the [RGB Value](#) field and enter a description
2. Using the [colour picker](#), you can click on the palette to select a colour

You can also modify the description of the colour – the text which appears in the [Description](#) field will appear in the drop-down menus of page, field and graph formatting properties.

i5 will also allow you to add new colours of your own to the list of default colours – any new colour which you add will also appear on the drop down menus of the formatting tabs.

Formatting date & numeric values

i5 uses data type picture clauses – sometimes called *tokens* - to format dates and numeric types. A data type picture clause is basically a rule or set of rules that determines how these types of data in a database will be rendered on screen. Take dates, for example. A date can be rendered in a number of ways - dd/mm/yyyy, yyyy/mm/dd, mm/dd/yyyy and so on. A picture clause simply determines which of these formats will be used.

i5 supports a variety of date and numeric formats and they can be edited by expanding the detail page of a section, clicking on the relevant field name to display its field properties, selecting the Advanced tab and entering a picture clause in the Data Format field.



i5 allows the following *numeric* tokens to be used:

0.00	is rendered as	4.87
\$0000	is rendered as	\$2386
£0.000	is rendered as	£4.867

The following are allowable *date* tokens:

yy or yyyy	are used to define year rendering	99 or 1999
MM or MMM	used to define month rendering	11 or Nov
dd	used to describe	Day date
hh	used to describe	Hours number
mm	used to describe	Minutes number
ss	used to describe	Seconds number

Each of these tokens is optional, and i5 allows you to choose the delimiter of your choice. It then displays the data on the page in the exact format you provide for the field in question. So, the picture clause dd/yyyy/MMM-hh:mm:ss in the above screenshot renders a date as 09/1974/SEP-21:00.00

Search options

Search options were mentioned very briefly in the description above of the *Component Assistant*. In fact, i5 allows you to specify search options on a search page in three ways. Firstly, you can use the SQL wizards, and secondly you can hand write your SQL search criteria by going directly to the SQL component's Advanced tab for the section on which you want the search to operate. Finally, you can add an *Options* control button from the *Component Assistant* for the search page in question. In the latter case, i5 supports exact match and case sensitive searches.

Exact match

This specifies that a database query only returns a row if it exactly matches the specified entries on the search page. If the Exact Match option is left unchecked, then the database finds all rows where the relevant column begins with the specified text.

The i5 default is to leave Exact Match unchecked. In this case, if the user specifies *Ann* as her search string, i5 returns all the rows that contain this string at their beginning – e.g. the names *Ann*

Lane, Annie, Anne and *Ann*. If Exact Match is checked, then i5 only returns rows that contain precisely the string *Ann*.

Technically, exact match corresponds to the = predicate in SQL. If not used, the SQL predicate is LIKE, with left-anchored wildcards. In MSDE/SQLExpress/SQL Server, for example, the relevant SQL query fragment is as follows:

```
(...) AND FIRST_NAME LIKE 'Ann%'
```

The % in this context means 'any alphanumeric character'

Case sensitive

The case sensitive search option requires fields in the database to match exactly the fields that are typed into the search criteria. For example, *Ann* will not match *ANN* if the Case Sensitive option is checked. By default, i5 searches are case insensitive.

If Case Sensitive is checked, then i5 makes the search text uppercase and issues the SQL query with the relevant database column converted to upper case. Once again, in the case of SQLBase, the relevant query fragment is as follows:

```
(...) and @UPPER( FIRST_NAME) = "ANN"
```

By default, the i5 exact match and case sensitive options are not checked. So this would be the relevant query fragment in SQLBase:

```
(...) and @UPPER( FIRST_NAME) LIKE "ANN%"
```

To search for the string *Ann* anywhere within a column, specify the search text as *%Ann*. i5 adds the trailing wildcard and makes it uppercase as needed. In this case, i5 displays rows containing names such as *Mary-Anne* as well as names that simply begin with *Ann*.

Users should consult the documentation specific to their databases for more information about wildcard based searching and upper case conversion.

Page layout

i5 provides two ways for the application designer to lay components out on the pages of a book - flow-based and absolute layout.

Flow-based This determines the position of components on the basis of two properties - where a component sits in the list of components for a page - and how the i5 position property has been set - to the right of the previous field, below the previous field or to the left margin of the page.

Absolute i5 also provides a more flexible and precise layout option - absolute layout, which uses pixel values to determine the position of components on i5 pages. Altering the position of a component using absolute layout is a matter of using a mouse to drag a component to and drop it on the desired position on the page.

Layout options are set at page level. To alter the page layout option you must: open a book section in the Explorer, click on the Properties tab for the page whose layout method you wish to change. The Layout Method list box allows you to choose between flow-based and absolute methods (i5 defaults to absolute). We will look at page layout in more detail later in this manual.

i5 Runtime

i5 Runtime is a runtime component that enables you to interact with databases from within a browser using pages created with i5 Studio. It can be accessed either from the Windows Start Menu, by navigating to Start > Programs > i5 > Documentation > Welcome to i5 or by clicking the [Open Book](#) arrow icon on the main i5 Studio toolbar.

If the former option is selected the user is taken to a page displaying an Explorer of the books that are currently held in the [Development](#), [Release](#) and [Testing](#) areas. The latter access option can only be selected if the user already has a book open in i5 Studio.

Whether you select the book you wish to view from the Explorer or open it directly from within i5 Studio, a second browser window is launched on which a login page is rendered. A successful login will take you to that book's default section, which may or may not be that book's index. If it is you are provided with a hyperlinked list of Sections for that particular book. Clicking on any Section name will take you to the Search page for that section, allowing you to interact with the database to which that book is connected.

i5 Runtime Toolbar

When the i5 Explorer was first discussed, it was mentioned that under each section there was a Toolbar element. This toolbar has a default position (unless it has been purposely moved of course) when a book is opened up from i5 Runtime. The standard buttons that appear on the toolbar are as follows:



Index This returns the user to the i5 Runtime index page.



Search This returns the user to the i5 Runtime search page.



Save This saves changes made to a record.



Delete This deletes any selected record

Not all of the i5 Runtime buttons appear on every i5 page. For example, it doesn't make sense to put a *Save* button on the toolbar for a *Search* page as the requirement for it would be, to say the least, unlikely.

You don't have to view the pages that you have designed using *i5 Runtime* – you can view and interact with them in the *i5 Studio*. However, only the *i5 Runtime* will give you an accurate idea of what your pages will look like to a user at runtime in a browser.

i5 Object Engine

Although the i5 Object Engine is not a component of i5 that you need to have any direct interaction with, it is important because it handles the connection and persistence of multiple users to either single or multiple databases. As the word intimates, persistence means that a user connection to a database persists over time. Database connection handling is designed to operate behind the scenes, meaning that the work of the Object Engine is invisible to the user, and because you don't need to see it, it is designed to be invisible.

The Object Engine validates requests to connect to a database - i.e. it verifies that the connection is being made by a valid user - and once a connection has been validated, any subsequent requests

from the client device making the request - browser, cell-phone or handheld PDA - are routed transparently to a database and, where appropriate, to COM/COM+ or .NET business logic.

The Object Engine is designed with the technical qualities of high database connectivity bandwidth and the high scalability and redundancy of COM/COM+ or .NET logic in mind. Database connectivity bandwidth determines how many client requests to a database/set of databases can be handled at any moment in time and this is managed by the Object Engine by passing database requests to secondary 'i5sql' processes. The level of scalability and redundancy for business logic determines how easily an application can be scaled up to deal with more users or a greater volume of traffic and the reliability of the application. The more redundancy an application has, the more reliable it will be - and to achieve this, the Object Engine passes requests for business logic to registered Business Logic Servers. Business logic servers are out of the scope of the discussion of this chapter, but more can be found about them in the *i5 Developer's Guide*.

Another technical way of putting all this is to say that whilst the i5 Object Engine delegates responsibility for database access and code invocation to secondary mechanisms, it retains the responsibility for session management and content presentation.

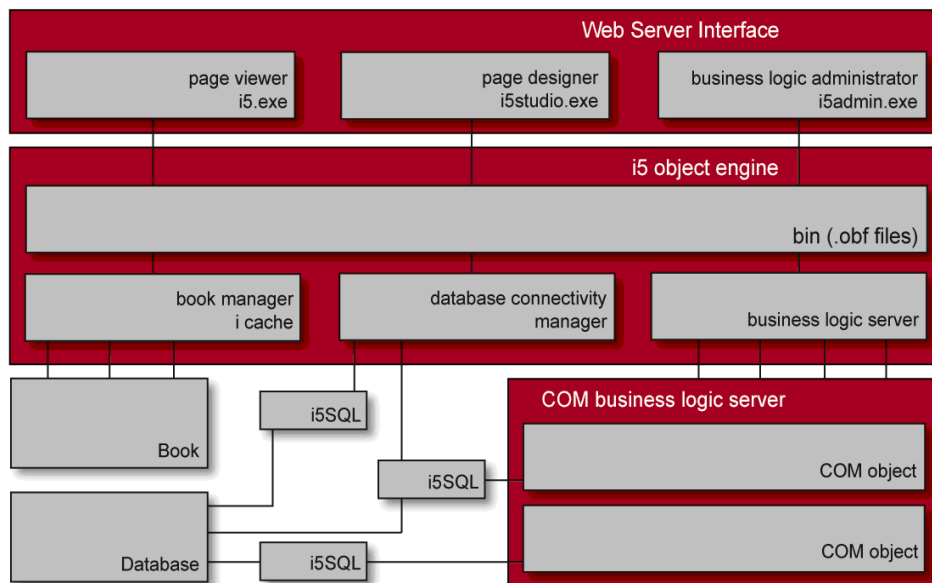
Shutting down the i5 Object Engine

When you installed i5 you will have opted to run the i5 Object Engine as either a service or in console mode. If i5 is running on Windows 95/98 or on Windows NT/2K/XP/Vista/7 as a console application, select the Object Engine window and press Control-C or Control-Break.

If i5 is running on Windows NT/2K/XP/Vista/7 as a service, you must close it down using the Services applet of the Control Panel.

Fitting it together – the underlying architecture

The diagram below illustrates the basic architecture of i5:



The components making up the web server interface layer are the components with which you will have most interaction. The components in the object engine layer do all the critical work holding together the different bits of the system, translating your design-time interactions into code, turning code into readable, useable web pages, shuttling data to and from databases.

The business logic server serves up code written in a proprietary programming language to allow i5 to produce web pages dynamically. The component processes that the business logic server

generates - called *i5SQL* processes - allow a piece of software like the business logic server to get data from a database at a distance.

The Book component has been mentioned earlier – it is that part of an i5 application that contains all the crucial information about how to render pages created at design time. The database component should be self-explanatory.

The diagram omits the both the web server and the database server which we discussed in the last chapter, but if you want mentally to put them in the picture, the web server is between the interface and object engine layers and the database server between the database and the i5SQL processes.

Hopefully, this chapter has provided a good overview of key aspects of i5 - the architecture, the Object Engine server, and the i5 Studio. In addition, our brief description of the range of semi-automated tools that i5 provides - the wizards, the Component Assistant, colour customization and searching – will, we hope, have given you an indication of what help you have at your disposal to make building web applications easier.

In the following chapters these features will be explored in more detail by means of a series of walkthrough tutorials which, starting at a very basic level, will gradually become more complex, ending with a discussion of a case study website.

4. Building a single-table i5 book

Having explained some of the key components of i5, it is now time to learn how to use them, and in this chapter we will walk through a series of tutorials where you web-enable a database and then exploit i5's query and search capabilities. These tutorials, you will find, are based on the built-in point-and-click wizards as described in the last chapter, and coding skills are not a pre-requisite..

Before starting i5, check that your database server & the i5 Object Engine are running.

Building a single section book

Our aim in this tutorial is simply to create a one section (table) book using the Customer table from the i5demo database.

Logging on and creating a new book

The first thing you need to do to create a new i5 book is to log on to i5 Studio. You have done this already, but here's a reminder of what you need to do:

- Select Programs > i5 > Documentation > Welcome to i5 from the Windows Start menu to display the contents page
- Click [i5 Studio](#) to display the Connect dialog:
- Enter the following values:

Database:	i5demo
User Name:	sa
Password:	sa
- Click [Connect](#)

This will display the library of i5demo books (ie i5 books that use i5demo as their database) and a link for creating a new book. By clicking on the link [Create a New Book](#) the basic outline of a new book appears in the Explorer. In the Tab view, the Book Properties tab will open up. It is a good idea, particularly if you are going to be doing a lot of development to be sure to record intelligent information about the book you are working on in the fields provided. We suggest that you add the following information:

Book Title:	Single Table Tutorial
Version:	1.0

Clicking [Save Changes](#) as you might expect, will save that information in the book file and thus the book title in the Explorer view will change from *Untitled* to *Single Table Tutorial*.

Adding a section to a book

You now have a book which contains a login page and a contents section but nothing else. It's time to put a section with some actual data in it into the book.

Let's use the Page Wizard to create a new section for viewing and updating the Customer table automatically:

- Click the Wizards button to display the Wizards page
- Select the Page tab and the wizard will guide you through the process of adding sections to your book
- Select dbo.customer from the Tables list box.

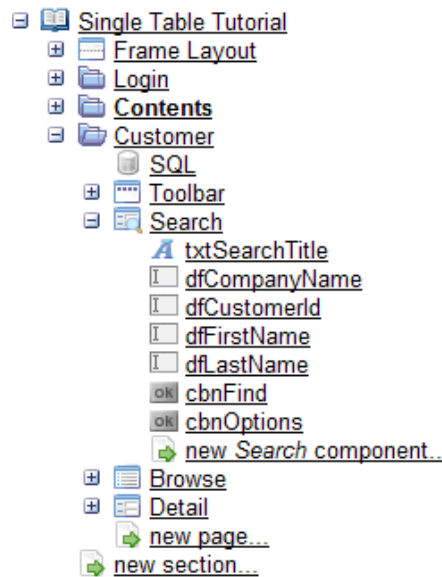
- Scroll down the page and click Go.

The Page wizard now creates a section for the Customer table and adds a data-field for each column in the table. The wizard's results appear in a table in the Tab view. Note also that a Customer section now appears in the Explorer. If you click on the Customer section node in the Explorer view to expand it, you will see that folders are displayed for Search, Browse, and Detail pages. These folders contain the fields and other components displayed on each page. If you then click on the Search page + node to expand it, you will see the fields that have been automatically added to the Search page by the wizard. You should see something like the file hierarchy shown in the screenshot below.

Changes to a field's appearance or position, or its deletion, are made via the properties page. You can access this either by clicking on the field's name in the Explorer view, or by clicking on the arrow button adjacent to the field's name on the preview page.

It is often quite likely that you will not need all of the data fields that the wizard creates automatically from the database table. This is the case with the Detail page here. Removing such fields is simple. If you open up the Customer Detail page and then click on the dfZip field to display its property page, you can delete the field by clicking the [Delete](#) button and then [Confirm](#) to delete the field.

If you repeat these steps for the following fields on Customer Detail page: dfCardNumber, dfCardExpiry, dfFax and dfUrl you will start to have a page which looks a bit less cluttered.



Formatting the section

Without user intervention, i5 will format the pages that it has built for the Customer section automatically. Fields corresponding to columns in a database table are rendered in alphabetical order, and the size of such fields varies according to the properties of the columns in the database. As you can imagine, this may not always result in an optimal layout of items on a page. The modification of the appearance of a page is easily accomplished via the properties options displayed in the Tab view of i5 Studio for each constructed page.

The [Properties](#) page in the Tab view allows you to edit a number of the parameters controlling the appearance of a page, including the position of data fields and other objects on a page. In particular it allows you to select one of two methods for rendering components on an i5 page: *Absolute* and *Flow-based* layout, as already mentioned. Let's have a quick look at flow-based layout.

Changing the layout method

Before you can change the position of any components on an i5 page you will need to set the layout method for the page as a whole. As indicated in the previous chapter this is achieved by selecting the layout method option from the Properties tab of the page in question.

Changing field positions

Flow-based layout determines the position of an object on an i5 page by considering which other objects on the page are contiguous to it and how it relates to those contiguous objects. i5 provides a deceptively simple set of criteria to let you determine the relative position of any one component, although how it will determine the actual spacing between objects is decided by i5 at runtime.

Let's try changing the position of some fields on the Customer Detail page using flow-based layout method.

- In the Explorer view, select the Customer Detail page.
- In the right hand frame click the arrow symbol next to the field whose position you want to change. This will display the field properties page.
- Click the Position tab. This displays information about the field's position on the Detail page. In the screenshot here, for dfAddress, you can see that in order of appearance running down the page, dfAddress is situated between dfAccountCode and dfCardExpiry. You can also see that it will be rendered next to the left margin of the page.

The screenshot shows the 'Properties' window for the 'dfAddress' field. The 'Position' tab is active, displaying the following settings:

- Move to where:** Between dfAccountCode and dfCardExpiry
- Alignment:** Right of Previous Field
- Tab Index:** (empty)
- X Coordinate:** 270
- Y Coordinate:** 20

Buttons for 'Save' and 'Undo' are visible at the bottom of the tab. Below the 'Position' tab, the 'Advanced' and 'Comments' tabs are also visible.

Changing the value in the Move to Where field will change the relative position of a field in comparison to other page objects. Changing the Alignment value allows you to have an object appear next to the left margin, to the right of the previous object on the page, or below the previous object on the page. The best way to understanding how object positioning works is to experiment with it a little yourself. Don't forget to click the Save button whenever you make changes to an object's properties.

Using the guidance above, edit the Customer Detail page so that you are left with the following objects on the page: dfCustomerName, dfAccountCode, dfFirstName, dfLastName, dfAddress, dfPhone, dfEmail, dfCustomerId and dfRowId..

Arrange these fields in two parallel columns, in the order given above. We suggest that you be sure to align dfPhone to the left of dfAddress, and dfEmail below dfPhone. See if you can figure out how to hide dfCustomerId (hint: look at the field's Properties tab).

The page should now look a bit like the screenshot below, which makes a bit more sense than before. If it doesn't, don't worry too much – have a play around with the position properties for the fields on the page until you've got something you like and you are happy with how flow-based layout works:

The screenshot shows a web form titled "Customer" with a toolbar at the top containing icons for home, search, save, and delete. The form contains the following fields:

- Company Name
- Account Code
- First Name
- Last Name
- Address (a larger text area with a vertical scrollbar)
- Phone
- Email

Changing field labels

All page components in i5 can have a title property, which must be set by the developer when adding components to a page manually, but which are set automatically by i5 when creating pages using a wizard. The titles that i5 creates automatically are not always the most sensible for the data items in question (they are based on column names in the database, which one would normally expect to make sense to a developer but not necessarily to an end user). You can change a field's title by accessing that field's Field Title property from its Properties tab. For example, using the Customer Detail page, click the arrow symbol next to Phone field to expose its properties, and then click on the Properties tab, change the Field Title property from Phone to Telephone Number and click Save.

Adding HTML template files

A selection of ready-made HTML templates is supplied with i5 to allow users to add colour and style to their books. For example, if you wanted to change the overall appearance of the Company Search page by using the stored templates and footers, you would need to: click on the + on the Customer node in the Explorer view to expand the section, then click on Search to expose the properties of the Search page and select the Formatting tab. You can then select the HTML Template of your choice from the dropdown listbox selection of tasteful, pre-programmed options, and click Save button at the bottom of the screen to save your changes.



You are not restricted in your use of templates to those supplied with i5. Users with some knowledge of HTML will not find it difficult to create their own. The templates provided with i5, such as the FreshGreen template shown here, can also be customised if required.

There are typically two components to a template - the images included in it and the HTML file which the browser requires. The latter is located in the `Dataline\i5\include\` subdirectory, whilst the former are located in the root directory of your web server, in a subdirectory given the name of the template.

So, for the template 'FreshGreen' on an installation of i5 using IIS, there is something like:
`C:\Inetpub\wwwroot\i5\Templates\FreshGreen\`

Hiding the delete button

If you select and view the Customer Detail page you will see that the Delete button has been made part of the page toolbar by the Page wizard. To prevent your users from deleting records from the Customer section, you can remove this button from the toolbar and de-activate the ability of the page to delete records. To do this you will first need to click on the Customer node in the Explorer to expand it and display its components; you should then click on the SQL node to display its properties and un-tick the Allow Deletes check-box and click Save. Having done this you should then open up the Toolbar in the Customer section, click on `cbnDelete` and tick the Invisible checkbox.

Summary

OK, so by now, if you've followed everything here properly, you will have built a functional i5 book using only minimal keyboard input. You should have a better idea of how to create a book section with search, browse and detail pages, and how to use the page properties to manipulate the appearance of the pages. Provided that you have indeed done everything properly, you should now be ready to move from design time in i5 Studio to runtime using i5 Runtime. This will allow you to interact with the i5demo database from within your browser using the sections you created in i5 Studio.

Previewing Books

In the process of book development it is, of course, a good idea, to preview your work before you publish it to the world. You will want to verify the layout and appearance of individual pages in your browser (and, in all likelihood, in a number of different browsers, given the subtle differences in the way that browsers can and do render html). To preview your new *Simple Book*, click on the arrow icon in the toolbar at the top of the left hand frame of the Designer. This will take you to the login page that i5 created automatically for you.

The login

Enter *sa* in both the Username and Password fields and click the Login button. A page index for Tutorial 1 will be displayed, showing a section named Customer

Note that although you are using the same credentials as before you are now logging-in to an i5 book - usernames and passwords for i5 Studio and for i5 books do not necessarily correspond. After logging in you should be taken to the Contents page, which provides links to 'data aware' sections in your book.

Click Customer to go to the Customer Search page.

The search page

Type the letter P - upper or lower case - in the Last Name field and click the Find button. The letter P acts here as a search string and it will select all the records in the Customer table in the database which have a contact whose surname begins with P. Click the Find button to display the browse page.

The browse page

The results will be displayed in a Browse page. Search results are tabulated - i.e. each row on the page represents a matching record in the Customer table and displays a number of data items. Here we show the Browse page after clicking the Find button with no specific search criteria. As you can see, it has returned all the customers listed in the database.

Details	Company Name	Customer Id	First Name	Last Name	Orders
	Whitehouse Enterprises	0	Bill	Clinton	Orders
	Boppin Burgers	1	Elvis	Presley	Orders
	Universal Domination Inc	2	Darth	Vader	Orders
	Bedrock Furnishings	3	Betty	Rubble	Orders
	Weekend Dance Infections	4	John	Travolta	Orders
	Russia First Productions	6	Vladimir	Putin	Orders

Page 1 of 1

Click the Detail icon on the left of the *Boppin Burgers* row in the browse table.

The detail page

Data for the Boppin Burgers is displayed on a Detail page, as shown below (actually, we've cheated a little: if you want to see the strawberries you will have to set the [HTML Template](#) formatting property to point to the Strawberry.htm template).

Account Code	Address	Card Expiry
97815561519	36th Way, box 97017, Redmond, Washington	
Card Number	Company Name	Customer Id
	Boppin Burgers	1
Email	Fax	First Name
me@elvis.com		Elvis
Last Name	Phone	Rowid
Presley	1556151985	00000000000013EE
Title	Url	Zip
Mr	www.boppinburgers.com	

Orders

Template supplied with i5® | Dataline Software Ltd

You might want to try modifying some of the details of the customer record that you have chosen. If you followed the exercises above, you should find that whilst clicking on the icon of the diskette in the toolbar in the top left hand corner of the page saves any changes that you make to the record, you are not able to delete the record from the database because there is no delete button available for you to do this with. Obviously, if you go back into the Designer you can reverse the changes that you made to disallow record deletion.

This simple, one-table tutorial book is clearly not very sophisticated and we don't pretend it is, but it does serve to illustrate some of the mechanisms behind any i5 application. In particular it illustrates the relationship – the connections - between the Search, Browse and Detail pages which are crucial in any data-driven i5 application (aka book).

Behind the scenes

Before going any further, it might be worthwhile trying to understand a little bit of what happens behind the scenes when your book is rendered as a set of webpages.

When the Customer hyperlink on the index page of your one-table book is clicked, a request is sent to the i5 Object Engine for an i5 Search page.

The i5 Object Engine looks up details - via the Book Manager - for this page in the appropriate book (.obf) file using the information it receives from the browser. This information is used when it retrieves the relevant data to create an HTML stream which is sent back to your browser to render the Customer search page.

When the Find button is clicked - after entering the letter A into the Customer Name data field - the data is returned to the i5 Object Engine as part of a SQL 'submit' event - a concept discussed in more detail in the *i5 Developer's Guide*.

Understanding that this new data was submitted as a query on a search page, i5 then formulates this information as a SQL query which is sent - via its *Connectivity Manager* and as an *i5sql* process - to the database. The results of this query are then returned to the i5 Object Engine.

Whilst all this is going on, the book file is being searched for information regarding a possible Browse page. The information retrieved is then merged with some of the results of the database query, formatted as HTML and sent back to your browser. Intelligently, i5 understands that results for one column of the data returned should be formatted as an HTML hyperlink so that when the user then clicks on the Name of one of the company records displayed on the browse page it can return to the results previously retrieved from the database, find the relevant data associated with that company and then return that to the browser, formulated as HTML according to the instructions held in the book file.

Actually, the process is quite a bit more complicated than this, but it's enough to explain what's happening. To summarise the key points:

- The information on how to render a book is kept in a book file, and possibly in a style sheet stored in the /Dataline directory;
- The information to be rendered as a book is - for the most part - contained in a database;
- The browser on your machine does all the rendering work;
- The i5 Object Engine processes all the information received, whether from database, book file, browser or Business Logic Server.

Primary keys

One of the cleverer aspects of i5 is the way in which it searches the database to which the user is connected. It does this by taking the information entered as search criteria on a search page and uses this to create a SQL select statement. Whilst such select statements can vary considerably in complexity, i5 uses the SQL concept of a primary key - the table column that uniquely identifies a row or record in a table - to help it return data properly. When the Customer table is searched at run-time, i5 uses the primary key to retrieve a record from the database.

This means that in order to select data for the Customer record in i5, at least one of the fields on the Customer detail page must have its primary key property set. This allows i5 to identify the record and in this particular case, the *CustomerID* field is the primary key. When a record is selected from the browse page table the page link takes the CustomerId value across to the detail page and uses it to select the appropriate record from the database for display.

Page object properties

It is easy to determine which field or fields on a page are acting as the primary key for the record or records displayed. Inspecting the properties of an individual field in the Tab View of the Designer will reveal whether or not that particular field has its Primary Key property set. On the Properties tab of any field on a page there is a Primary Key checkbox. If this is ticked, it indicates that the database column that this field refers to is being considered by i5 to be the primary key for that page. You can have more than one field act as primary key, in which case you have a compound key. You will note also that there is an Update Key checkbox, that i5 uses when updating records. However, we will be covering that later in this guide.

Summary

We have tried in this chapter to show you the basics of building simple i5 pages connected to a datasource. In addition to showing some of the parameters that you have at your disposal for formatting pages, we've also tried to explain a little bit of how these pages work, the mechanisms that they use for finding records and so on. We will admit that the results are rather basic. The important thing is that you come away with a sense of a) how easy it is to build serviceable pages very quickly and b) how the core page types in i5, the Search, Browse and Detail pages interact with each other. In the next chapter we will look at developing something a bit more sophisticated.

5. Building a multi-section i5 book

In the previous chapter we looked at the absolute basics of building an i5 book, namely the process whereby a section, containing data from the columns of one database table, is created in a book. Sections are the basic building blocks of i5 books and - as we have seen already - by default i5 will create a data-field for each column in the table to which the section corresponds.

In addition to looking at the process of creating a section in an i5 book, we also looked very briefly at some basic formatting issues - specifically, the way that the position of data fields on an i5 page can be changed using absolute and flow-based layout.

In reality a one-section, one-table i5 book is not likely to be much use to anyone, and neither does it demonstrate the versatility of i5, so in this chapter we will begin to address the problem of building multi-section books with more sophisticated formatting, adding other components such as list boxes, radio buttons and so on.

The chapter is structured into three parts. In the first, we look at the process of adding sections to an i5 book, and in the second, we look at some basic formatting issues - deleting unwanted fields, adding list boxes and so on. In the third and final part of the chapter, we look at some more advanced formatting issues.

As with the previous chapter, the emphasis is on exploring what can be done with i5 by simple point and click, and computer programming skills are not a prerequisite.

Creating a new multi-section book

To extend our understanding of how to use i5 we will create a new book. To do this, follow exactly the same steps for creating a book as you did in the previous chapter, being sure to log on to the *i5demo* database.

So that you know where you are and what you are working with, be sure to call the book that you create something both pertinent and memorable. We suggest you enter the following values into the relevant fields on the book Properties page:

Book Title:	Multi-Table Tutorial
Version:	1.0

Adding pages to the book

By far the easiest way to add data-aware pages to an i5 book, of course, is to use the Book wizard. You may recall that the Book wizard gives you the option of selecting a template for the formatting of pages at the outset, and when run creates a section for every table in the i5demo database, and a field for each column of each table. A progress bar will appear in the Tab view and the phrase *Wizard running* will appear on the status bar at the bottom of your browser while i5 is busy.

The Book wizard results are displayed in a table format on the i5 Wizards page - see right. The table on this page also specifies the joins that have been made for each page - if, indeed, any have. The Explorer view of your site is also populated with the newly created sections.

Book Wizard	
Wizard complete! The report below explains what pages were created:	
dbo.CATEGORY	Added Section: Category Join Results: Joining to table dbo.CATEGORY_GROUP (aliased to 'a'), from column CATEGORY_GROUP_ID to column CATEGORY_GROUP_ID
dbo.CATEGORY_GROUP	Added Section: CategoryGroup Join Results: No joins were made
dbo.CUSTOMER	Added Section: Customer Join Results: No joins were made

Formatting pages

At this point, the new book that you have created holds a section for all of the tables in the *i5demo* database. It is worth noting at this point that i5 will only create sections for tables that have primary keys associated with them. Note also how tables which are joined by foreign keys have joins created between them. This 'greedy' strategy of the Book wizard also ensures that each section holds a data-field for each column in the table for which it is a section. Whilst it is unlikely in any real-world situation that you would ever build a website which requires you to display all the columns from all the tables in a database, it is far easier and far quicker to prune off the data fields that you don't need than it is to manually add each one that you do want to your book.

Deleting unwanted fields

As we did in the last chapter, delete the following fields from the Customer Search page of the book that you have just created: *dfFirstName*, *dfLastName* and *dfCompanyName* (we will explain why we are deleting the latter shortly). Then open up the Customer Detail page and format as per the single-table book tutorial you followed in the last chapter

Component Prefixes

If you have a look through some of the pages that i5 has created for you, you will notice that different objects on any one page have different prefixes. i5 wizards automatically adds these prefixes to field (df), push button (pb) and control button (cbn) names. These tokens provide a naming convention to identify the component type, something which is of particular value when working with a large book displaying a large number of page components.

Changing field positions & alignment

Having removed some data fields, we can now change the position of some of those that remain. As we looked at flow-based layout in the last chapter, let's use absolute layout here. As absolute layout is the default layout setting for i5 pages you need make no changes in order to work in this mode.

Let's do some work on the Customer Detail page. If you position your mouse over the top left-hand corner of any object on the page, the pointer icon will turn to a four-directional arrow. If you click and hold down the left mouse button, you can drag the object you have selected around the page. Cross-hairs will appear on the page to help you reposition the field, as in the screenshot below

After dragging the data field to a new position, releasing the mouse button will send the new x/y co-ordinates of the data-field back to the i5 server, which will update the page layout details accordingly. You can also fine-tune absolute layout by editing the pixel values of the x and y co-ordinates of the upper left hand corner of any component. These values are displayed on its [Position](#) tab and so can be edited manually.

Repeat the process of re-positioning for each data-field on the page until you have a layout with which you are happy.

Changing field width

i5 provides a default width for the data fields it uses when creating a page, but this is not always appropriate for the data with which the fields are populated. In the case of the Customer detail page, which we have been working with, for example, it is evident that the dfZip field is far wider than is actually necessary for displaying a zip code. To change the width of this field: click the arrow button adjacent to the dfZip field - or click on dfZip in the Explorer - to display the field's properties, then click on the Formatting tab to display the formatting properties for the field. You can then change the width of the field by changing the value in the Field Width field (from, say, 250 to 80) and then clicking Save.

Adding a list box

To save key strokes, to minimize error and generally to make interactions with book pages more efficient, it is a good idea to use list boxes for selecting data to update tables with or, indeed, to set a search criterion for database search. In the previous chapter, we showed you how to search for a company on the Customer search page using the first letter of the customer name. Here we will illustrate how to select a company record by using a list box to display a list of all available customers.

- Click on the [New Search Component](#) on the Customer Search page to display the Component Assistant, and select [ListBox](#) from the list of available components
- Give the new ListBox the [Name](#) `lbCustomerName` and the [Title](#) `Customer Name`, select the option [Automatic – Content comes from Datasource](#) from the Automatic or Manual Content list box

and change the value in the Datasource Column list box to `dbo.customer.company_name` to associate the list box with that table column.

- Click Next and change the Datasource Table list box to `dbo.customer`. This tells i5 that the contents of the list box are to be retrieved from the Customer table
- Click Next and change the Value to be Stored list box to `customer.company_id`. This will define the value that is to be written into the database column. Now change the Value to be Displayed list box to `dbo.customer.company_name` so as to define the value that is to be displayed in the list box
- To finish off, clicking Next for the last time will display the properties defining the list box you have just set. You can now verify that these are what you want. If they are, click *Finish* and you are done. If not click Back to go back and change what needs changing.

The `lbCustomerName` list box property page will be displayed on the Customer Search screen, and you should be able to change field values from interactively from within this screen if required. Note also that the Explorer view has been updated to include the `lbCustomerName` list box.

You may wish to reposition the list box – probably to the location of the deleted `dfCustomerName` data field. If this is the case, drag and drop it to the appropriate position.

When you insert a list box into a detail page the wizard requests the default value for the primary key. This is not required when the component is being added to a search page as above. You can find out more about primary keys later in this tutorial.

The Customer section search page should now look something like this (we're using the `WhiteOnRed.htm` template here):

Changing field text colour and width

The `dfCustomerId` field is the primary key for the `dbo.customer` table, and to make it stand out on the Customer search page, we will change its title text colour to red. Formatting text colour is very simple in i5, and in this instance you should open up the properties tabs of the Customer ID field (click on the arrow in the top left hand corner of the object in the main window of the Designer), then click on the Formatting tab of the object in the Explorer and change the value in the Title Color list box from Default text colour to Red. Whilst you are there, change the Field Width value from 250 to 80. We don't need a field that wide to display a primary key. Don't forget to save your changes.

Adding HTML template and footer files

You will know from your use of the Book wizard that you can select a template to customize the look and feel of your web pages when you first run the wizard. However, it is not always the case that you either know in advance exactly what you want your web pages to look like or whether you want all the pages to look the same. For this reason, i5 allows you to change HTML header and footer files on a *per page* basis if that is required.

Template and footer file properties are accessed from the Formatting tab of a page – there is a list box for the header and a list box for the footer and both are populated by i5 with the template files located in `C:\DataLine\i5\Include`.

To change, for example, the template and footer files for the Customer Search page you would open up the Customer Search page in the Explorer so as to access its Formatting tab and then select new template and footer files from the HTML Template File and HTML Footer File list boxes, saving your changes as you go.

We've provided a number of more or less tasteful templates that you can use. There's no guarantee that our taste or design requirements will match yours and if you wish to add templates of your own, then obviously there is nothing to stop you. If you want to make a template of your own, it is not difficult to use an existing template as a starting point and adapt as you see fit. Please note though that if your template requires an image, these will need to go into a directory that your webserver can locate. If you are using i5 with IIS, this directory will be something like: `C:\Inetpub\wwwroot\i5\templates\TemplateNameHere`.

More page formatting

There are a range of other page formatting options available to you in i5. We explained previously how unwanted fields and list boxes which have been put on a page automatically by either the page or the book wizard can be deleted. In an earlier chapter we pointed out that it was possible – and indeed, sometimes, necessary – to modify the SQL statements that your pages use. Let's look at these again and also try adding some other components – a radio button set for example.

To do this we will work on the *Order* section of our basic book. The aim of this exercise is ultimately to produce a page which is easier to read and to work with – achieving this result is really dependent on a good knowledge of how to optimize user interfaces, which typically involves a trade off between what the structure of the data that you work with will allow and what you would like to achieve in terms of elegant page layout and design.

Deleting unwanted fields & list boxes

The first thing to do is to delete those data fields on the pages in the *Order* section which are superfluous to requirements. These should be reasonably easy to identify. Select and expand the Order section in the Explorer view to display its pages and expand its search page. Delete the *lbCustomerId*, *lbProductId* and *lbStatusId* list boxes in the search page.

Adding & positioning a list box

Now let's add a Customer Name list box to the Order search page. You should be able to do this without too much bother, as we have already done this in an earlier section of this chapter.

Whether you work in flow-based or absolute positioning on a page, i5 will add any new component below those that already exist on the page. Assuming that you are working using the default absolute positioning, you will now need to drag the new Customer Name list box to an appropriate place on the page. You might change your mind about this later – but for the time being this doesn't really matter.

Adding radio buttons

The i5demo database contains a 'lookup' table for setting the status of an order. There are five types of order status – unconfirmed, confirmed, back order, shipped and cancelled. By default this information will be displayed as a numerical value in a data field – as you can verify if you take a look at the Order Detail page. This is OK, but a simple graphic representation of an order's status is preferable from an end user's point of view. By using a radio button group we can provide one button for each of the five possible payment statuses.

Let's open up the Order Search page to add a Radio Button group:

- Click on the [RadioButton](#) option to create a RadioButton, giving it the Name 'rbStatus' and the Title 'Order Status'.
- Select the Automatic option from the Automatic or Manual Content list box and then select `dbo.order.status_id` as its Datasource Column and ensure that there is no default value for the Primary Key field (we do not wish the status of an order to be the primary key when searching for details of an order).
- Click Next and in succession select `dbo.order_status` as the Datasource Table, `dbo.order_status.status_id` as the Value to be Stored and `dbo.order_status.description` as the Value to be Displayed

When you are finished, the properties of the new radio button group will be displayed and the search page should look something a bit like this:

The screenshot shows a web application interface for searching orders. At the top left, there are icons for home and search. The main heading is "Order". Below it, there is a "Search Criteria" section with a dropdown menu set to "All" and an input field for "Order Id". To the right of the search criteria is a radio button group labeled "Order Status" with five options: "Back Order", "Cancelled", "Confirmed", "Shipped", and "Unconfirmed". At the bottom of the search criteria section are two buttons: "Find" and "Options". The footer of the page reads "Template supplied with i5® | Dataline Software Ltd".

Modifying SQL statements

Now let's do the same on the Order Detail page. As noted previously, it already has a standard data field for displaying the status, `dfStatusId`, so let's delete that and add a radio button group in exactly the same way as we did on the Order Search page. This should improve the usability of the page, as we now have a textual description for each of the different order statuses (the key value, `status_id`, is not very informative for a human).

When you add a radio button group, i5 makes a distinction between the database column your buttons are keyed to in the database and where it extracts the information to build the radio buttons. In the example we have been looking at, what i5 is doing effectively is creating a join between the Order table and the Order Status table, using the column specified in the Datasource

Column property to make the join. This allows you to display more information than is held in the (correctly normalised) Order table.

You can inspect the SQL that i5 uses when it creates a radio button group. This information is held in the Radio Button Contents field on the object's Properties tab:

```
SQL: SELECT dbo.ORDER_STATUS.STATUS_ID,
        dbo.ORDER_STATUS.DESCRPTION FROM dbo.ORDER_STATUS ORDER BY
        2
```

It is worth noting that you can modify this SQL by hand. So, if you wanted to display the numerical value for the order status id in addition to the description, you could rewrite the statement as follows:

```
SQL: SELECT dbo.ORDER_STATUS.STATUS_ID, convert(varchar(2),
        dbo.ORDER_STATUS.STATUS_ID) + ' ' +
        dbo.ORDER_STATUS.DESCRPTION FROM dbo.ORDER_STATUS ORDER BY
        2
```

What we've done here is simply say that our SQL should retrieve the Status ID value twice over, but in the second instance we have said that we want to prepend it to the Description string. To do this, we have to convert it into a character string itself (it is stored in the database as a numerical value). Microsoft database systems use the Convert() or Cast() functions to do this. We have used Convert(). Don't worry too much if you don't follow the details of the SQL. You will get the hang of it eventually.

The new radio button group has now been placed on the page beneath the existing objects, and the Order Detail page should now look like this:

The screenshot shows a web form titled "Order" with a toolbar at the top containing icons for home, search, print, and delete. The form contains several input fields arranged in a grid:

- Comments (text area)
- Complete Date (text field)
- Cost (text field)
- Customer Id (text field)
- Order Code (text field)
- Order Date (text field)
- Order Id (text field)
- Product Id (text field)
- Quantity (text field)
- Rowid (text field)
- Status (radio button group):
 - 0 Unconfirmed
 - 1 Confirmed
 - 2 Back Order
 - 3 Shipped
 - 4 Cancelled

At the bottom right of the form, there is a small text label: "Template supplied with i5® | Dataline Software Ltd".

Even more page modifications

Changing field positions

Let's go back and change the positions of the fields on the detail page. You might still have some quibbles about the pertinence of the order that i5 has given to the fields on the page and you will probably find that for the purposes of fine tuning it is easier to set the final X-Y co-ordinates for the data fields from the Formatting tab of each field (assuming you are using absolute positioning).

Single record search result sets

When a search is made the result set is displayed in a browse page in a tabular, non-editable format, and from that result table a single record is selected for editing in the detail page. We mentioned in passing in the previous chapter that in the case where a result set comprises a single record it is possible to send the user automatically to the detail page for display and editing without the need to display the browse page.

For a customer search it is inevitable that our search will only ever return a single record - remember that our Customer Name list box is populated with all and only the company names stored in the i5demo database. It would be pointless to require the user to go to the trouble of having to select the single record from a browse page every time a search is made for a company. Fortunately, i5 provides you with a setting to bypass the Browse page if only one record is returned in the Search query.

Open up the Company section in the Explorer and navigate to the Advanced tab of the Company.Browse page. By ticking the Go to detail page if only showing one row check box on the Advanced tab you tell i5 that if your company search only returns one record, then the user should be sent directly to the Detail page for that company

Now do exactly the same thing for the Order browse page. Obviously, in the case of orders it is far more likely that a search will return more than one row in the result set, necessitating a browse page. However, in case we do have customers who don't do a great deal of business with us, let's just make sure we force the redirect from the Order search page to the Order detail page.

Setting date formats

The default date format in i5 is yyyy-MM-dd. As noted in Chapter 3 above, however, other formats can be specified if required. You might want to change the format for the date fields on the *Order* detail page, again to improve legibility. To do this you will need to navigate to the Advanced tab of a date data field and enter the requisite date format - let's use dd/MM/yyyy - in the Date Format field and save your changes. Try it for the two date fields on the Order Detail page (dfOrderDate and dfCompleteDate)

Testing the book

Now that we have finished formatting the pages of our book, it is time to check that everything works properly. Open up the i5 Runtime and after logging-in, navigate to the Customer section.

From the Customer Name list box, enter Clinton in the Customer Surname field and click on Find. You should be taken directly to the detail page for Bill Clinton's company (Whitehouse Enterprises). If you are not and are taken to a browse page instead, you have not enforced the page redirect mechanism discussed earlier in the chapter.

On the Bill Clinton detail page, click the Orders button at the bottom of the page. There is only one order in the database for Bill Clinton, so provided you have again enforced the page redirect, you will be taken straight to the detail page for that order.

Formatting the browse table

You will probably have noticed by now that although the Book and Page wizards take a lot of the effort out of building books and sections in books, they can be rather limited in how they lay data fields and other components out on a page.

Both these wizards are greedy (and a bit indiscriminate) in that they put all the fields they can on to a page and insert them in database order. The net result of this is that it is usually necessary to spend time formatting and laying out the pages being built in a more intuitive, user-friendly way -

the part of the development process that requires your ability for creating an appropriate user interface.

In this section we are going to continue with the formatting theme by looking at some of the more advanced formatting options that are available to you for designing browse tables. These options include the display of photographs, multiple data records on a single row and column alignment in relation to other columns. We will take a look at these features now.

Creating a products section

For the sake of practice, let's start our exercise in browse table formatting by creating a new book. You know how to do this, but if you've forgotten, click on the Home icon on the main toolbar if you are already working in i5 Studio and then select the Development tab and click Create New Book. Call your new book something relevant, like Browse Tutorial and then click Save.

To create a new section in your new book, click on the Wizard icon on the main toolbar, select the Page Wizard from the Tab view, select dbo.product as the table for which you wish to create a section and then click Go.

Modifying the browse table

The browse table generated by the Page wizard does not show the most interesting bits of the *Product* data and the formatting is a little dull, so let's modify the columns which are on the table.

The first thing we will do is make a page link which allows you to drill down to the Product detail page containing an image of the product. We can do this very easily, as the database keeps an image of each of the products in its Product table. What we want to do is put a clickable thumbnail image in the Browse table. You can do this by a) Telling i5 that the pbDetail page link on the Product Detail page uses an image taken from the database and b.) Telling i5 what sort of image the image held in the database actually is. You can do the former by setting the Data Source property of pbDetail to dbo.product.picture and the latter by setting the Images are of Type property to JPG Image (this property is located on the Formatting tab of the page link component).

If you preview the Detail page at this point, you will see that the page link image is too small to be of any real value, so it would be a good idea to change the dimensions of the image, which you can do by navigating to the Formatting tab of pbDetail and setting the Width property to 40 and the Height property to 44.

Now let's remove some of the other columns on the table – delete dfCategoryId and dfDistributorId – and hide the dfProductId field, by going to its Formatting tab and clicking the Invisible check box









We are hiding rather than deleting the dfProductId data field for a good reason. The page link button - or rather, image – that pbDetail uses needs to know which product to select when the user clicks on it to go to the Product Detail page. The Product ID column in the database provides it with this information.

We will look at how a page link works in more detail in the next chapter.

At this point the Product browse page displays a set of images, a product description and a distributor name. Let's add a price. Navigate to the Product Browse page in the Explorer, click on New Browse Component and select a Browse Table Field from the Tab view. Name the field something sensible, like dfPrice and make its Data Source product.price, uncheck the editable box, click Next and then Finish.

Making a field on a Browse page non-editable is preferable for two reasons. Firstly, it prevents users from accidentally modifying this data, and secondly, an uneditable data-field has more formatting options than an editable one.

Product

Details	Description	Distributor Name	Price
	Eau de Toilette	Ralph Lauren	32.450000000000003
	Shower Gel	Ralph Lauren	11.99
	Shower Gel	Ralph Lauren	6.1900000000000004
	After Shave	Ralph Lauren	24.5
	Eau de Toilette	Gianni Versace	25.0
	Eau de Toilette	Giorgio Armani	29.0
	Eau de Toilette	Giorgio Armani	30.0
	Shower Gel	Gianni Versace	9.0

Page 1 of 6

Let's format this page a little better. First, let's move the Distributor Name and Price fields so that they display underneath the product description. Doing this is easy: go first to the `dfDistributorName` field Position tab and set its **Alignment** value to **Below Previous Field**. Now do exactly the same thing for the `dfPrice` field. That should look a bit better.

We've still got the problem that the formatting of the price data is a very uneven. It would be nice if we could format it so that we show the currency it is in, and only displaying values to two decimal points. If all we wanted to do was the latter, it would be very easy: you would go to the Data Format field of the Advanced properties tab and set the value to 0.00.

However, that is not going to display our currency. We can do a little formatting of our own by modifying the SQL statement that `i5` uses to retrieve data from the database.

Open the Advanced properties tab of `dfCurrency`. The Datasource Column field currently reads as follows:

```
dbo.PRODUCT.PRICE
```

Change this so that it reads like as follows:

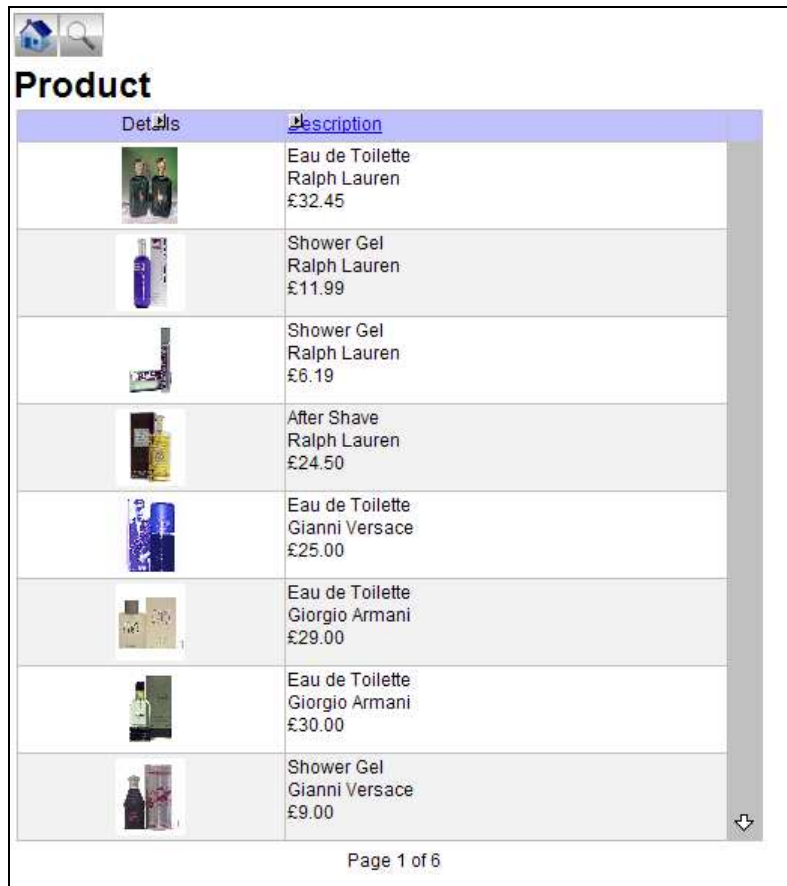
```
'£' + cast(cast(dbo.PRODUCT.PRICE as dec(4,2)) as varchar)
```









What we have done here is exploit a function built in to SQL Server, which allows us to take a column in a database that is formatted as one thing (a floating point number) and *cast* it as something else. In fact, we've done that twice: first we have cast it as a decimal using the line `cast....as dec(4,2)`. The numbers in parentheses simply tell the database that you can have a maximum of 4 digits to the left of the decimal point and two to the right. That takes care of the numerical format required for a currency. However, we have then used the cast function again

to turn our newly formatted decimal into a string that we can then prepend with a pound sign (which is in inverted commas because it is a string).

This data formatting trick depends on functions that are specific to SQL Server. If you are using some other DBMS you may need to talk to your database administrator to find out if there are functions in your DBMS equivalent to cast() and the concatenation operator +.





























The net result of this is that those oddly formatted numbers in the dfPrice datafield will now be returned as strings that make sense as prices in pounds sterling. If you preview the page now, it should look something like this:



Details	Description
	Eau de Toilette Ralph Lauren £32.45
	Shower Gel Ralph Lauren £11.99
	Shower Gel Ralph Lauren £6.19
	After Shave Ralph Lauren £24.50
	Eau de Toilette Gianni Versace £25.00
	Eau de Toilette Giorgio Armani £29.00
	Eau de Toilette Giorgio Armani £30.00
	Shower Gel Gianni Versace £9.00

Page 1 of 6

Finally, let's get rid of the column headers, and put a few more products on each row of the results table. You can do this by going to the Formatting tab of the Browse table, checking the Suppress Column Headers box, setting the Number of Records Per Row value to 3 and maybe setting the Rows value to something like 8. With a bit of luck, the table should now look something like this (which is much better than before):

Product			
	Eau de Toilette Ralph Lauren £32.45		Shower Gel Ralph Lauren £11.99
	Shower Gel Ralph Lauren £6.19		Eau de Toilette Gianni Versace £25.00
	After Shave Ralph Lauren £24.50		Eau de Toilette Giorgio Armani £29.00
	Eau de Toilette Giorgio Armani £30.00		Shower Gel Gianni Versace £9.00
	Eau de Toilette Gianni Versace £20.99		Deodorant Ralph Lauren £3.40
	Deodorant Ralph Lauren £3.40		Eau de Toilette Calvin Klein £23.76
	After Shave Gianni Versace £25.36		Deodorant Giorgio Armani £4.20
	Deodorant Giorgio Armani £4.20		After Shave Calvin Klein £23.55
	Eau de Toilette Calvin Klein £19.99		After Shave Gianni Versace £23.89
	Eau de Toilette Calvin Klein £22.50		Eau de Toilette Jean Paul Gaultier £32.44
	Deodorant Elizabeth Taylor £3.00		Eau de Toilette Calvin Klein £22.50
	Eau de Toilette Giorgio Armani £25.99		Foam Bath Elizabeth Taylor £13.40
	Foam Bath Elizabeth Taylor £13.40		Eau de Toilette Gianni Versace £26.98
	Deodorant Calvin Klein £5.59		Deodorant Giorgio Armani £5.40

Page 1 of 2

Formatting the detail page

Modifying the detail page

If you click through from the newly reformatted Browse page you will find that the Detail page for any selected product doesn't really work properly – the format for the price field is poor, the image doesn't display and a number of other fields are inappropriately sized and/or superfluous to requirements.

To correct this situation, let's start by deleting some fields: remove *dfAuthors*, *dfReleased*, *dfSubtitle* and *dfVersion*. Let's hide some other fields: *dfProductId* and *dfRowId*. We also need to replace some fields: *lbDistributorId* needs to become a datafield rather than a listbox, as does *lbCategoryId* and *dfPicture* needs to become an image. So, delete these three components and create a datafield that takes *dbo.distributor.name* as its datasource column, another that takes *dbo.category.description* as its datasource and an image that takes *dbo.product.picture* as its datasource. When you created the Product section, i5 will have added in joins to the database tables *category* and *distributor*, and will have given these tables aliases. The database column *dbo.category.description* will appear as *a.description* and the column *dbo.distributor.name* will appear as *b.name*. Make sure too that the new image field (call it *imgPicture*) is set to display JPEGs – you can set this value using the Images are of type field on the image's Formatting tab.

We also need to do some formatting of fields. The product title doesn't need a four line datafield, so let's change the Number of lines in field property to 1 (via the Formatting tab of the field *dfTitle*) and change its Height to match that of the other single line datafields – 22 pixels. We also need to set the Data Format of the *dfPrice* field. We won't use the formatting trick that we looked

at earlier here – it will make editing the price value a little tricky. But so we know what we are dealing with, let's just change the title of the field to read 'Price (£)' so that we know what we currency we are working in.

You will also need to experiment with the position of these components on the page. Using absolute positioning, see if you can end up with a page that looks something like this:

The screenshot shows a web application interface for a product page. At the top left, there are navigation icons: a home icon, a magnifying glass, a document, and a trash can. Below these icons is the heading "Product".

The form contains the following fields:

- Title:** A text input field containing "Contradiction".
- Product Code:** A text input field containing "2000113".
- Category:** A text input field containing "Eau de Toilette".
- Description:** A text area containing "Light, spicy, and romantic, Contradiction for men by Calvin Klein is a blend of juniper, bergamot and lavender." The text area has a vertical scrollbar on the right side.
- Stock:** A text input field containing "89".
- Price (£):** A text input field containing "23.76".

In the center of the page, there is an image of the product packaging. The image shows a clear glass bottle of "Contradiction for men" by Calvin Klein next to its black box. The box has "Contradiction for men" and "Calvin Klein" printed on it, along with "3.4 FL OZ" and "100 ml" at the bottom.

If you've got this far, congratulations – you've learned to create a data-driven multi-table website. To put it another way, you've learned how to web-enable an SQL database and, by using a range of i5 components and the extensive formatting properties that the i5 Studio provides, how to develop intuitive user-friendly interfaces.

We can't pretend that the tutorials that you have followed in this chapter will make you into web application gurus or into SQL overlords but we hope at least to have convinced you that i5 can do a great deal for you. With practice, you will now be able to construct data-driven web applications, making productive use of many of i5's facilities.

The next step is to explore another important feature of i5: page links.

6. Using page links

As i5 is a browser-based product designed for building browser-based applications it will come as no surprise that page links – whether in the form of text, buttons or images - are an intrinsic part of the way that it works. The reality is that navigation in i5 applications, as with any browser-based system is page-link driven.

i5 page links accomplish two important functions - navigation, as noted, and data transfer between pages. We have observed the operation of both these functions already, since they are crucial to the process of moving between the Search, Browse and Detail pages which the i5 Book and Page wizards create. These wizards generate page links automatically for accomplishing such tasks, but there may well be occasions when a page link is needed for some other purpose, such as linking to pages created outside i5, or more likely when you are working with business logic.

This chapter focuses on the mechanics of page links - how to construct them and how to use them whilst the *i5 Developer's Guide* provides, for the more adventurous, a detailed description of the use and creation of business logic. More specifically, we will be showing you how to use page links with single and multiple field values and how to use list boxes with a search page.

Single field value in a page link

For the purpose of clarity, it is a good idea to create a new book – why not call it *Page Links?* - for the tutorials that are to follow. If you can't remember how to do this, refer back to the tutorials and discussions in the previous two chapters.

Create a section for Page Links, using the Component Assistant - call it Pages perhaps - but in this case set the Datasource Table field to None.

By leaving the value of this field empty - or null – you indicate to i5 that it does not need to be data aware. This means that i5 will not create SQL statements to retrieve data from the tables in the connected database. Consequently, i5 will not create search, browse and detail pages for the new section. This does not, however, prevent you from creating pages that can be utilized for practically any purpose.

For this tutorial we will create two blank detail pages to demonstrate how page links work.

Creating new pages

To create two pages in the new Pages section, expand the section and click on the New Page link to display the Component Assistant. Select Detail Page from the list of component options to create a new detail page, call the page *Page1* and give it the title Page One. Click Next, leave the Datasource Column as it is, displaying None, click Next again and then Finish

The new page is displayed in the Explorer and its properties will appear in the Tab view. Rather than repeating the same process to create a second page, you can clone – or copy - the one that has just been made. Cloning is a useful feature of i5 – particularly when the item that you wish to clone would otherwise take a great deal of time to create. In this case it actually wouldn't take long to create a new blank page, but you can imagine how long it might take to create a page with a number of bespoke components and a hand-crafted layout.

To clone the page, navigate to the Properties tab of Page1, click the Clone button. Clear the message telling that the object has been cloned successfully – you will see that a new page, *Page12* has appeared in the Explore. Navigate to the Properties tab of Page12 to display the page's properties and change its Name to *Page2* and its Title to Page Two (and don't forget to save your changes).

Adding data-fields

It's now time to add a data-field and a page link to each page to allow the capture of a data value from the first page and to carry it across to the other.

To add data-fields to Page1 and Page2, navigate to and select the Datafield item on the Component Assistant list for Page1 (you can get there via the New Component link on Page1 in the Explorer. Give the new datafield that you create the Name dfField1 and the Title Page One and click Next, retain the No Default Value option, click Next and then Finish. Do exactly the same thing for Page2.

Of course, you could actually have waited to clone Page1 until you had created the page link on it – that would have saved you the bother of doing the same operation twice, as you have done here.

The 'df' prefix is not obligatory that we are using here is not obligatory, but it does clearly identify the component as a data-field.

Adding page links

Adding a page link component to a page is as easy as adding a datafield, although you do need to know one or two other things – where you are going to use the page link to navigate *to* and what data, if any, you want to transport across the link.

To create a page link here you will first need to navigate to the Component Assistant for Page1 and select the Page Link item. Call the new page link pbPage2

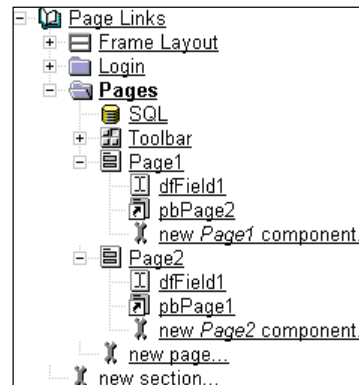
In the Link Text field - this is the button or hyperlink text that will be displayed on the page - enter Goto Page Two, select Pages.Page2 from the Go to Page list box, render it as Push button, and click Next.

In the next page ignore the Add Link button, click Next and then Finish. An onscreen report will summarize what you have elected to do and a push button should now be visible on Page1.

Repeat steps 1 to 4 identically for Page2, but call your page link pbPage1, make the link text Goto Page One and select Pages.Page1 from the Go To Page list box.

In the next step of the dialog, click Add Link. At this point i5 scans the datafields that are available on the pages that are being linked and prompts you to select which field on Page Two you wish to take a value from and which field on Page One you wish to take it to. You may also add more values to carry across from one page to the other – in situations where more than one datafield is available - by clicking More. In this instance there is only one data field on each page, so this option is not available. Choose :dfField1 and :dfField1, click Next and Finish.

If you've done everything correctly, your tree of components in the expanded view of the Explorer should look like the screen shot on the right here.



The operation of a page link may be defined using SQL Select and Insert modes. If you are creating a page link and wish to carry with it a data-field value to match a value in the target page data table, the select mode ensures that if there is a match the existing record is displayed in the target detail page. Insert mode is for the situation where the data carried in the page link is the basis of a new record, and as such is inserted into the target detail page as a new record.

Note that the 'pb' prefix, like the 'df' prefix for datafields, is not obligatory but identifies the component as a push button or page link.

Testing the page link

As suggested in the previous chapter, it is good practice to test your work as it is developed. This is especially so when working with page links as it is very easy, particularly when there is more than one data-field on a page, to inadvertently take values from the wrong source data-field and/or to carry them over to the wrong target data-field. Although there is no risk of that happening in the present example, testing your work at an early stage is a good habit to acquire.

To verify that your page link works correctly and data is transferred from Page2 to Page1 only, try typing something into Field One on Page1 and clicking the Go to Page Two button. Field One on Page2 should be empty. Now type something into the empty datafield on Page2 and clicking the Go to Page One button to return to Page1. You should find that the data is carried across to Page1 but not in the opposite direction.

Multiple field values in a page link

The ability to carry over multiple field values in a page link is an important feature in i5. As you can imagine, whilst the ability to carry one data item is useful, in applications which handle large amounts of information it is far more likely that multiple items will need to be manipulated. Here we will look at carrying invoice data from a browse page over to a detail page using page linking.

Using the *Page Links* book that you created previously, the first thing that we need to do is to add some data-fields to *Page 1* in the *Pages* section of that book and then reposition the page link that is already on the page. Following the steps that you took earlier, add two more datafields to *Page1* – *dfField2* and *dfField3* (use the clone function if you think it will be easier).

If you are working in the Absolute Layout mode, at this point *dfField3* will have been created but it won't be completely visible, because it is sitting directly 'underneath' *dfField2* from which it has been cloned. You will need to drag *dfField2* aside to make *dfField3* visible. If you are working in Flow-based mode, this problem won't occur.

Regardless of which mode you are working in *pbPage2* - the Page Link object - is definitely in the wrong position. It is better if this object appears at the bottom of the page after the three data fields, so at this point you should move them around accordingly so that *Page1* eventually looks a bit like the image here.

The screenshot shows a window titled "Page One" containing three data fields labeled "Field One", "Field Two", and "Field Three", each with an empty input box. Below the fields is a button labeled "Go to Page Two".

We will now create a browse page in a new section. It will use the Product table from the *i5demo* database as its data-source. This browse page will supply the values that we are going to carry in our page link.

The first step is to create a new section, so navigate to the New Section item in the Explorer and select the Section option from the Component Assistant list. Call the new section Products, select *dbo.product* from the Datasource Table list box, click Next and Finish.

The second step is to create a browse page in the new section, which you can do by expanding the Products section in the Explorer view, and selecting the Browse Page option from the Component Assistant list. Give the new page the Name 'Browse' and the Title 'Products'. You will be prompted to select a series of columns from the Product table in the database. Using the Ctrl button, select the following columns – price, product_code, product_id, stock and title, and then click Next. (You may find it easier to select all columns and then delete the columns that you don't need once the page has been created). Finally, you should select *product_id* from the Primary Key list box, click Next and Finish.

Custom links on a browse page

Browse pages come with a ready-made page link which will carry over the primary key for the product record in the *Product* table. We don't want to use this page link as we wish to define a number of data items for the page link.

First we must remove the page link: expand the *Products* section and the *Browse* page in the Explorer, click on *pbDetail*, click the Delete button on its properties page and then confirm the delete.

Next we need to add a new page link to link to *Page 1* carrying the data that we want. On the *Products* section *Browse* page, add a page link component called *pbPage1*, with a Title of 'Go to Page 1' and select *Pages.Page1* from the Go to Page list box. Choose the push button option from the Render As listbox and create the three following links using the Add Link option:

```

:dfProductCode to :dfField1
:dfPrice        to :dfField2
:dfTitle        to :dfField3
  
```

After you have saved your changes, you should find that the Products Browse page now looks a bit like this:

Products					
Price	Product Code	Product Id	Stock	Title	Link
32.45	2000102	2	65	Polo	Go to Page One
11.99	2000103	3	19	Polo Sport	Go to Page One
6.19	20001040	4	43	Polo Sport Extreme	Go to Page One
24.50	20001050	5	98	Safari	Go to Page One
25.00	20001070	7	78	Versus	Go to Page One
29.00	20001080	8	29	Acqua di Gio for men	Go to Page One
30.00	20001090	9	52	Armani for men	Go to Page One
9.00	2000110	10	92	Black Jeans for men	Go to Page One
20.99	2000111	11	87	Blue Jeans for men	Go to Page One
3.40	2000112	12	832	Chaps for men	Go to Page One

Page 1 of 5

From the Explorer select the *Product Browse* page and click on the *Go to Page 1* button at the end of one of the product rows. If you've done everything correctly, *Page 1* should display with three of the product values in place. If it doesn't, you should go back and correct any mistake you may have made.

Creating a list box search page link

To further illustrate the flexibility of page links we will now use a list box to select the value that we wish to carry across via page link. The list box will be populated with data from the *Distributor* table in the *i5demo* database and will allow us to make the link a search page link.

The first thing we need to do is add a new detail page to the Pages section. You probably know how to do this by now but just in case, navigate to the New Page option and select Detail Page from the Component Assistant list and call your new page *Page3* (with Title of *Page Three*).

Page Three

Distributor

None

Having created a new page, let's add a list box to it, so select ListBox from the Page3 Component Assistant, give your list box the Name lbDistributor, the Title Distributor and select Automatic from the Content list box. After clicking Next, select dbo.distributor from the Datasource list box, then make distributor.distributor_id the Value to be Stored and distributor.distributor_name the Value to be Displayed, click Next and then Finish.

You might decide at this point that displaying the name of the distributor by itself is not enough. This can be easily corrected. The SQL that is used to populate the list box with information from the database can be inspected on the Advanced properties tab of the list box. It should look like this:

```
SQL: SELECT DISTRIBUTOR.DISTRIBUTOR_ID, DISTRIBUTOR.DISTRIBUTOR_NAME
FROM DBO.DISTRIBUTOR ORDER BY 2
```

Let's modify the SQL statement so that it returns both the distributor's name and account code – that will be a bit more informative. To do this we are going to use the concatenation operator of our DBMS (which in this instance is Microsoft's SQL Server) so as to return both the distributor name and account code in one string:

```
SQL:SELECT DISTRIBUTOR.DISTRIBUTOR_ID, (DISTRIBUTOR.ACCOUNT_CODE + '
' + DISTRIBUTOR.DISTRIBUTOR_NAME) FROM DBO.DISTRIBUTOR ORDER BY 2
```

It is worth noting here that not every DBMS uses the plus sign (+) as its operator for string concatenation. You should check the documentation for your DBMS or with your database administrator if you are not sure of the exact syntax for string concatenation.

Having created our page link, let's create a new section to hold the page that will display distributor information. This is the page to which we will be carrying a value using a page link.

- Create a new section called Distributors, using the table dbo.distributor as its Datasource Table.
- Add a new page, a Detail page, with the Name Detail and the Title Distributor Details to the section, selecting distributor_id, distributor_name, account_code and address from the Datasource column list box and then selecting distributor_id to be the primary key for the page
- Add a PageLink component to the Detail page, with the Name pbPage3, the Link Text Go to Page 3, Pages.Page3 as the Go to Page, and set it to Render As a Textual Hyperlink.
- We aren't going to take any values back to Page3, so let's set the value of the Action on Link Page field to None and click Finish.

For navigation purposes - so that we can shuttle back to Page 3 if required - we will now add a page link to the *Detail* page in the *Distributors* section:

- Expand the Detail page in the Explorer view, click on New Component and select Page Link from the list.
- Name the link pbPage3, enter its Link Text as Goto Page 3, select Pages.Page3 from the Go to Page list box, render the link as Textual Hyperlink and click Next.
- We aren't going to take any values back to Page Three, so set the value of Action on Link Page field to None and Finish.

The screenshot shows a form titled "Distributor Details". It has four input fields: "Account Code", "Address", "Distributor Id", and "Distributor Name". Below the "Distributor Id" and "Distributor Name" fields, there is a blue hyperlink that says "Go back to Page Three".

Finally, and to round off the tutorial, let's add a page link to Page3 so that we can carry the data value that is selected from its list box over to the Detail page. This will ensure that the correct distributor information is displayed when a user makes the selection from the list box.

- Create a new Page Link component from the New Component list of Page3, naming it pbDistributor, giving it the Link Text Show Details, selecting Distributor.Detail as the Target Page. Render the link as a Textual Hyperlink
- After clicking Next, set the Take value from property to :lbDistributor and the Take value to property to :dfDistributorId. Click Next and Finish.

Testing

The routine should be familiar by now. From the Distributor.Detail page, click on the Goto Page 3 button, select a name from the Distributor list box and then click the Show Details page link. If everything is working properly, you will be taken to the Detail page displaying information about your selected distributors. It's worth trying this a few times with different distributors just to check that it works as it should. It's worth noting here that in order to select the data for the distributor record, i5 requires that at least one of the fields on the Detail page to be set as the primary key.

If it isn't working properly, you need to retrace your steps and check that you followed the guide properly. If it does work – and there's no reason why it shouldn't – you have now learned the nuts and bolts of page linking in i5.

In the next chapter of this Guide we are going to take a look at child tables and groups. These are slightly more sophisticated components to use on i5 pages and for that reason require a more detailed discussion.

7. Child tables and groups

The child table object can be used to extend i5's capacities to display, update and create data beyond the basic search, browse and detail page capabilities which have already been examined.

For example, in much the same way as a browse page is used to display a range of records returned from a database, the child table object on an i5 detail page can be utilised to display a more detailed set of data for each master record. A good example might be the orders that a customer places for a set of goods. Provided with a Customer Detail page, for example, it would be useful to display further details of the orders that a customer has placed, in the form of a scrollable table within the same page.

This chapter explores the features of i5 child tables and explains how a child table can be embedded within a detail page. In the first instance we will look at how to display records using a child table. Later in the chapter we will change tack and look at how we can edit and save data in a child table. As this will require the use of the i5 [Multi-Table Update](#) component, we will have to explain some basic aspects of the way that databases keep track of records using keys. The discussion is not particularly complex, but it is necessary in order to understand how the Multi-Table Update component works. We will then move over to a discussion of group components, of which the Multi-Table Update is but one example.

Creating a child table in a detail page

In the first part of this tutorial chapter, we will do three things:

- Put a child table component onto a detail page and specify the database columns we wish to use to display data;
- Implement some filtering on that table so that the orders it displays are always only those for the customer in question;
- Ensure that data items are displayed in the correct order.

However, before doing any of this, you will need to create a Customer section in the book that you are using for your tutorials. Use the Page wizard to do this and be sure to select the Customer table from the Tables list box when doing so. If you can't remember how to create a section using the Page wizard, refer back to one of the previous chapters in this guide.

Once you have created a section based on the Customer table, you need to edit the associated detail page, as the data fields will appear on it in an arbitrary order. We suggest that you spend some time re-arranging the fields on the page until they appear in an order that makes more sense to you. You may also decide that some of the fields are not really necessary – perhaps the dfCardNumber, dfCardExpiry, dfFax, dfFirstName, dfLastName, dfTitle and dfZip fields. If so, you should remove them. Whilst you are at it, set the dfRowstamp and dfCustomerId fields to Invisible (via their respective Formatting tabs).

The formatting of the page should now be a little more orderly and ready for you to add a child table.

Creating a child table

Having prepared the detail page in our Customer section, we can now add the child table to it and format it so that it displays details of the orders placed by specific customers.

- Navigate to the Component Assistant for your new detail page and select the Child Table item, make the Name of the child table tblOrders and its Title Orders, and set its Datasource to dbo.Order;
- Using the Control key on your keyboard, select the following as Datasource Columns for the child table: order_id, cost, order_code and order_date, complete_date, quantity, then click Next and set the Primary Key to order_id, click Next and Finish;

You will probably want to do some formatting on the table – making it wide enough to display all the data that you want to show, for example. If you go to the Formatting tab of tblOrders you can set the Width of the table to something more appropriate, say 550 pixels.

Customer

Account Code: 29026147345

Company Name: Weekend Dance Infections

Address: 77 Sunset Blvd, Beverly Hills

Email: johnt@sat-night-fever.com

Phone:

Url: www.sat-night-fever.com

Orders

Complete Date	Cost	Order Code	Order Date	Order Id	Quantity
	30.00	5437658765	23/01/2000	1	2
16/10/1999	23.99	543743	11/07/1999	2	1
	20.99	43654843	22/01/2000	3	1
	3.40	6548745	24/01/2000	4	1
24/01/2000	50.72	3265437	22/12/1999	5	2
10/01/2000	29.00	1231	09/01/2000	6	1
20/01/2000	26.98	86367	13/01/2000	7	1
03/01/2000	40.20	853213	19/12/1999	8	3
30/12/1999	20.99	35467889	11/12/1999	9	1
03/01/2000	15.00	785643	27/10/1999	10	5

If your page looks something like the above then all is well. However, there is a problem with the table at the moment – it is actually displaying all orders for all companies, although the detail page itself is displaying data associated with a specific company. We therefore need to edit our settings so that only those orders associated with the specific customer in question are displayed

Making the orders match the customer

To select the orders we need to tell the database to filter the results of the selection of data that i5 asks it to perform. We can do this using the i5 [Query Wizard](#). You can navigate to the Query Wizard by selecting tblOrders from within the Explorer and then clicking on the SQL icon under the tblOrders 'node'.

After clicking More in the Query Wizard, the dialogue will prompt you to provide a column name and criterion for searching on that column. Complete the dialogue by setting the Column field to order.customer_id, the Operator to '=' and the Value field to :dfCustomerId (note the colon prefix here). Save your changes.

The colon prefix is important here: it serves to indicate to i5 that you are using a 'bind variable', a variable that acts as a placeholder in an SQL statement. It is a placeholder in the sense that the value which occupies the place is determined dynamically at runtime rather than being set statically in advance. Use of a bind variable allows i5 to set the value of dfCustomerId when the SQL statement is actually executed. Both the child table and the detail page itself contain a customer id field, and this ensures that the value of the bind variable dfCustomerId in the child table will be determined as identical to that of the variable dfCustomerId on the page.

Run a search for a customer from the Customer Search page or select a customer from the result set in the Customer browse page. If you select a customer now you should find that the child table on the Customer Detail page is displaying only the items associated with that order.

Displaying the orders in...order

When you previewed your work you may have noticed that although our child table displays orders for the correct customer it does so in an unsorted order. This is not really a problem when there are so few records, as is the case with the customers here, but it can become an issue with larger result sets.

To order the items numerically is very easy and involves adding an 'Order By' clause to the SQL statement which selects the results to display. You can do this using the [Sort Wizard](#) accessible from the SQL tab of the tblOrders component : open up the Sort Wizard, click More and set the Column value to order.order_code and the Order value to Ascending

Now if you select a customer from the Customer browse page, its detail page will be displayed with the child table listing - in order - the orders associated with that customer.

Previewing your work will probably suggest to you that the information that we have put into our child table is not fantastically helpful. To flesh it out, let's make it a bit more useful by grabbing the description of the status of the order from the Order_Status table in the database. Displaying the order status will supplement the information that we have extracted from the Order table.

From the components tab of the tblOrders child table add a new ListBox. We want to use dbo.Order.Status_Id as its Datasource Column, drawing the listbox data from the dbo.Order_Status table, selecting dbo.Order_Status.description as the column to display, and dbo.Order_Status.status_id as the value to save. Set its Title to something like 'Status'. The page should now look something like this:

Customer

Account Code: 97815561519

Company Name: Boppin Burgers

Address: 36th Way, box 97017, Redmond, Washington

Email: me@elvis.com

Phone: 1556151985

Url: www.boppinburgers.com

Orders

Complete Date	Cost	Order Code	Order Date	Order Id	Quantity	Description
	30.00	5437658765	23/01/2000	1	2	Confirmed
03/01/2000	40.20	853213	19/12/1999	8	3	Confirmed
	90.00	43648893	11/01/2000	16	3	Back Order

Editing and saving data in a child table

i5 allows data on a page to be saved to a database by using HTML forms to pass the information to be saved back to the database. The *Save* button which is added automatically to detail pages created using the Page wizard allows this task to be accomplished.

A limitation of this approach to saving data is that it will only allow data to be saved to one table – the table from which the section is constructed. The i5 Multi-Table Update component overcomes this limitation by replacing the standard i5 Save button so that in addition to 'form' data, the contents of any child table or Groups on the form can also be saved or deleted at the same time.

The Multi Table Update button is a pre-programmed i5 plug-in component. It is actually written using business logic, and is part of the i5 Quick Start component toolkit. Because it is built in business logic, it won't actually appear on the New Components list in the Designer unless the i5 Business Logic Server is running.

We will now look at what we need to do to allow both the contents of the Customer form and the Order child table to be edited and saved using Multi-Table Update. To help you understand how this works we will be explaining how the Multi-Table Update component uses keys in order to identify which records in which tables are to be updated. Once we've done that, we will take a look at how to allow rows on a child table to be deleted and how to add new ones.

Saving changes to the table

Before we add a Multi-Table update button to the Customer Detail page, we need to remove the Save and Delete buttons from the standard i5 toolbar (these, you will remember, are added automatically whenever a detail page is created). Once we have done that we can add the button, set an update key and then make the data-fields that we wish to be able to update in the child table editable.

In order to remove the Save and Delete buttons, locate the Customer section, expand the Toolbar, click on the Properties tab of the button cbnUpdate and click Delete. Then do the same for the

button `cbnDelete`. And to add a Multi-Table Update button, expand the Customer Detail page in the Explorer and click on New Detail Component and select `qsMultiTableUpdate`.

Click the Position tab of `qsMultiTableUpdate` and select the position `Before dfInvoiceNo`

In order to be able to save changes both to the table and to the form (i.e. page) on which it sits, you will need to check both the 'Update Form' and the 'Allow Updates' boxes on the Properties tab of the Multi-Table Update button. Provided you have done everything correctly, you should find that you are able to save changes to data on the Customer form (in just the same way as you would with a standard i5 Save button).

In a multi-user environment it is important that an application is able to keep track of the changes which are being made to data. If two or more users are working independently on the same pages in an application each must be made aware of the changes that the other is making.

To detect changes made to data by other users i5 uses an Update Key and the Multi-Table Update component will, by default, flag the `dfRowid` field on the form as key to check for when updating. On a child table, on the other hand, it is necessary to set the update key flag manually. We will deal with this shortly. The use of an update key means that users can track any changes made to a record in a database because the data in the column which is used for this purpose changes each time the record is changed.

At this point we are not yet able to update the child table which we have placed on the Customer detail page. Let's have a look at how we can enable our `tblOrders` child table so that we can update its Quantity, Price and Status columns. We first need to make the fields `dfQuantity`, `dfItem` and `lbStatus` editable – the Editable setting is accessed from the Properties tab of each of these fields on `tblOrders`. After you have ticked the Editable checkbox for each of these fields, you might want to change the width of each field to say 40 pixels each. The Width setting is accessed from the Formatting tab of each field.

Specifying the primary & update key columns on the table

In order for the data in the child table to be saved properly, we need to identify a *primary key* for the table. In database terms, a primary key is an attribute which uniquely defines a record in a set of database records. The Multi-Table Update component requires the setting of a primary key so that it uniquely identifies each row in the table of data it is saving. In order to track changes made to data in the child table by other users, we also need to set an update key (see the discussion above).

In any well-designed database, tables will have their primary keys set when the database is designed, and i5 wizards should be able to detect which database column/s form the primary key in any given instance. Typically though, i5 requires that you set the primary key on any child table yourself. In the case of `tblOrders` you can do this by clicking on the field `dfOrderId` and on its Properties tab, if it is not already ticked, tick the PrimaryKey check box and then click Save.

Now let's add a column to our child table which we can use as an update key: select the Datafield item from the Component Assistant of `tblOrders`, give the field the Name `dfRowId`, leave the Title value blank and set its Datasource Column to `dbo.order.rowid`. On the Properties tab of the new `dfRowId` tick the UpdateKey property and click Save.

Making the `dfRowId` datafield our Update Key column tells the Multi Table Update button to monitor the contents of this column to determine if the row has been changed by another user. We can demonstrate this point by editing some order records in the orders table for a specific customer and then clicking the Multi Table Update button. Notice how the data in the `dfRowId` column of the updated row/s in the table changes when you do this.

The Multi Table Update button saves changes to the form as well as to the data in the table. You can use the Multi Table Update button to update as many child tables or groups as you like at the same time. This is done by simply providing each table with a PrimaryKey column and an UpdateKey column. All database updates are performed within one database transaction, and if any errors occur, then all changes will be rolled back.

In many instances, the primary and update key columns would be hidden from the end user. We are leaving them visible for the moment as they are useful for explaining how the Insert functions of the Multi Table Update button work.

Deleting rows in a child table

As well as updating the data in the table, the Multi-Table Update button can be used to insert new records and delete unwanted ones. Deleting records is easy.

Firstly, we need to allow the Multi-Table Update button to delete records. You can do this simply by opening up the Properties tab of `qsMultiTableUpdate` and ticking the Allow Deletes check box.

Secondly, we have to find a way of informing the Multi-Table Update button what records in a child table it must delete - as we don't want the button to remove everything in the table indiscriminately. We can do this by adding a tick box column to the child table to flag the records that we wish to remove:

- Select the `CheckBox` component from the `tblOrders` Component Assistant, call it `cbDelete`, set its Title to Delete and leave its Datasource Column set to `[None]`;
- Set the property When column contains this value, uncheck the CheckBox to 'Y' and leave the property When column contains this value, uncheck the CheckBox empty, click Next and Finish;
- Navigate to the Properties tab of the `qsMultiTableUpdate` component and set the property Delete Rows Using Delete Flag to 'cbDelete' and the Delete Rows Where Delete Flag Equals to 'Y', saving all your changes.

At this point you should verify that the Multi-Table Update button will allow you to delete records from the `tblOrders` child table. If it doesn't, navigate to the Advanced tab of your multi table update button and check that the Data Type is set to Plain Text. If it is and the control still isn't working, go through the properties of the button one by one, checking that they are set to the values specified above.

Adding rows to a child table

As mentioned above, the Multi-Table Update button does have the ability to insert new records into the database and will do so automatically during the save operation - i.e. when the user clicks the button. A new record will be inserted into the database on any row in the child table where the UpdateKey column contains no value, the assumption - which is effectively coded into the component - being that a row with no `UpdateKey` value must be a new row.

Firstly, we need to allow the Multi Table Update button to do inserts. We do this by ticking the Allow Inserts check box on the Properties tab of `qsMultiTableUpdate`.

Secondly, we need to add a new blank row to the child table. `i5` provides the TableInsert button for this purpose. When the `TableInsert` button is clicked it will add a row to the child table with which it is associated. To add a `TableInsert` button, navigate to the Component Assistant for the Customer Detail page, select the `TableInsert` component from the list and (in this instance), set the Name of Table for Insert property to `tblOrders` - this will tell `i5` which component it is to add new rows for. Make sure that the `TableInsert` button actually works, by navigating to the Customer Detail page and inserting a new row.

You will notice, at this point, that without some further adjustments, we can't actually tell the database what product we want to create an order for. There is a 'product_id' foreign key on the `dbo.Order` table but we cannot select a product when creating a new order. To do this, we need to add a listbox to the table, which we can populate with product titles from the `dbo.Product` table. To do this Expand the Explorer view of the child table `tblOrders` from the detail page of the Customer section. Click on the New Component button and add a listbox - called something like `lbProduct` - to the table, setting its Datasource Column to `dbo.Order.product_id`. Take the contents of the

listbox from the table `dbo.Product` and use `dbo.Product .product_id` as the value to store and `dbo.Product.title` as the value to display. You will need to ensure that the listbox is editable for it to be of any use when inserting new records. Check that the listbox works ok by previewing the page in the normal way.

You will notice that the values that are displayed in the listbox are not especially helpful – there appears to be replication of product titles. To correct this problem, we need to display more information in the listbox. There is a small problem though. The available columns in the `dbo.Product` database table are not really suitable for displaying in a listbox – the 'Description' column contains product descriptions that are too lengthy to be of value. What we can do, though is to make a join with the `dbo.Category` table, using the `Category_Id` value, which is common to `dbo.Product` and `dbo.Category`. We can then add in a category description with the product title in the listbox so we have more meaningful information to make our selection (remember that as we have told i5 that it is the `Product_Id` that we want to store when using the listbox, adding further columns in the value to be displayed is not a problem).

Change the text in the List Contents property of `lbProduct` to read as follows:

```
SQL:select dbo.PRODUCT.PRODUCT_ID, dbo.PRODUCT.TITLE + ' ' +
dbo.CATEGORY.DESCRPTION from dbo.PRODUCT inner join dbo.CATEGORY on
dbo.PRODUCT.CATEGORY_ID = dbo.CATEGORY.CATEGORY_ID order by 2
```

This revised text tells the database to give us back the `Product_ID` and then a composite string made up of the title of the product and the category that that product belongs to. The "+ ' ' +" tells the database to intercalate a space between the product title and the category description.

When you preview the page this time you should find that the information displayed in the listbox is considerably more helpful. There is still some overlap, because even with the category we still don't have quite enough information to disambiguate between listbox entries. However, you should have got the idea about what it is we are demonstrating here. You might want to try figuring out which other database column you need to add to the product title and category description to individuate terms in the listbox completely.

Generating a Primary Key

So far, then, we have described how a new row is added to a child table, but the Multi-Table Update component is not quite ready to add a new record to the database yet. This is because a primary key for the record that is being created is not being generated. You can see that when a new row is added its primary key value field remains blank. It is not good practice to insert a record with no primary key value into a database because the database has no means of uniquely identifying such a record.

For this reason then it is important to be clear about how the primary key for a new record is to be generated. Some database server systems – SQL Server, MSDE and Access for example - have the ability to generate their own unique primary keys on inserting new records. Columns in the database table can be assigned as being an AutoNumber data type. This is not the case with all relational database management systems however, and in such situations the Multi Table Update button will generate the INSERT statement, but we will need to provide the primary key value.

The `i5demo` database that comes as standard with `i5`, and which we are using here will insert a primary key automatically, using Microsoft's SQL Server *Identity Insert* functionality. However, you do need to tell `i5` that this is what should happen: you can do this by changing the datatype of the primary key field on the table (`dfOrderId`) to `AutoNumber` (you do this from the field's Advanced tab).

For the sake of argument, and just to explain what you might do in a situation where your DBMS does not allow for autonumbering, let us just look briefly at one fairly rough and ready way in which we can generate primary keys for inserting records.

Probably the easiest way to generate primary keys is to select the highest current primary key value and add one to it. To do this is very straightforward. Given a hypothetical child table *tblMyChildTable* (populated with data from the hypothetical database table *dbo.MyTable*) with a primary key *dfMyPrimaryKey* (populated from the column *MyPrimary_PK*) all you would need to do is open up the Properties tab of the field you have set as primary key on your child table and add the following text to its Default Value field:

```
SQL: SELECT MAX(MYPRIMARY_PK) + 1 FROM MYTABLE
```

(Obviously 'MyPrimary_PK and MyTable are dummy names here – you would need to enter the appropriate names from your database). What this bit of SQL does is select the key value for the last record in your database table and adds 1 to it – but it only does so when there is no primary key, which would be the case for a new record.

We don't need to use this way of generating a primary key here though, because we are assuming that you are using MSDE or SQL Server as your DBMS. To finish off this part of the tutorial, let's tidy up the page a little. By default the Multi-Table Update and TableInsert buttons have dimensions of 360 pixels in width and 200 pixels in height, which may spoil the formatting of your page somewhat. You can change these default dimensions from the Formatting tab of each component.

Hide the fields *dfOrderId*, *dfCustomerId* and *dfRowId* by clicking on the Formatting tab of the field properties, and ticking the Invisible check box

On the Position tab of the Table Insert button *qsTableInsert*, set the position to be after the *qsMultiTableUpdate* and set the Alignment to the right of the previous field

On the Formatting tab of the detail page set the Component Spacing property to 0. This will allow the Multi Table Update and the TableInsert buttons to abut each other.

Using groups

i5 uses the concept of a *group* of i5 components. A group, like a form, contains components such as fields, check boxes and child tables. A *group* can also be data-aware in that it can select data into its data-fields automatically, using an SQL query. The *i5 group* acts as the basic container for creating *plug-in components* - developer defined components which can be re-used across i5 pages and applications. *Plug-in components* are discussed in more detail in the *i5 Developer's Guide*.

Having looked at Multi-Table Update and TableInsert buttons, we now turn our attention to creating a basic data-aware group to display information about the distributor of the products that are associated with orders on the Customer detail screen that was built in the previous section of this chapter.

Before we can demonstrate the use of a group, we need to make some small modifications to our book. First we need to add another section, this time for product and make a few other changes.

1. Using the Page Wizard, create a section based on the database table *dbo.Product*. When you have done this, expand the section and delete the Search and Browse pages. Be sure to set the field *dfProductId* as the primary key on the Detail page and add a datafield, with no datasource column, called *dfCustomerId*. You might want to format this new detail page a bit, as it will be a bit of a mess at first.
2. Add a Page Link to the table, calling it something like *pbProduct*. Render it as a textual hyperlink and from the Advanced tab of the page link component, set its Datasource Column to *dbo.product.title*.
3. We now need to tell *pbProduct* what it has to link to and what values to carry across to its target page, so from the Link properties tab of *pbProduct*, set the target page to the Product detail page you have just created (it will be something like *Product.Detail*). We want to take values across from the fields *dfProductId* and *dfCustomerId* to the fields with the same names

in the Product.Detail page. (If you can't remember how to do this, refer back to the previous chapter on Page Links).

4. Save your changes (you were doing this all along, weren't you?!)

The changes that we have made should allow you to link directly from an entry in the child table tblOrders on the customer detail page directly to a page of detailed product information: our page link carries over a product_id value, which is then used to select a record on the target page. So far, so good. Of course, one problem that we now have is how to get back to the source page once we have looked at the details of a product. At the moment we can't. However, that situation is easily resolved – by adding a page link to the Product.Detail page that is targetted from the Customer page. If you do that now, you will probably realise pretty quickly why we had to a) add a dfCustomerId field to the Product.Detail page and also why b) we had to carry it across in our pbProduct page link. If you haven't realised, try to figure out how on earth you might get back to a Customer detail page without a primary key to do so.

Once you've added your 'back' button to the Product.Detail page, be sure to test that the links work – often it's a good idea to keep datafields that hold primary keys or hold values that are passed across from one page to another visible when you are developing – this makes it easier to understand what the problems are when things go wrong.

Adding a group to a detail page

We are now ready to add a group to our page. The process is exactly the same as it is for adding any other component to any other container object.

- Select the Group item from the Product Detail page New Component list, call it grpDistributor and set its title to Distributor. Click Next and Finish;
- Add four new datafields to the grpDistributor via the latter's New Component list. Call them dfName, dfAddress, dfEmail and dfPhone. Position them on the left margin of the group with dfName coming first and everything else aligning underneath it. Note that you can only use flow-based layout for the formatting of groups;
- We aren't using field titles here so if we check the Suppress Field Titles box on the group Formatting tab, we can ensure a smaller vertical distance between components in the group.

If you preview the Product page, you will see that the grpDistributor group is displayed at the foot of the page. You might want to experiment with the dimensions a little to ensure an orderly layout of objects on the page. More importantly, though, the group is currently an empty component, so we need to go through the process of populating the data-fields in the group with data from the database. We want the group to select its data from the table dbo.distributor in the database.

- On the SQL tab of grpDistributor, set the Primary Table to dbo.distributor;
- On the Properties tabs of dfName and dfAddress, set their respective Datasource Column properties to dbo.distributor.distributor_name, and dbo.distributor.address.

We're going to do a little kludge on the Datasource Column settings for dfPhone and dfEmail. By using the string concatenation operator '+' we can incorporate a little text into these two fields.

- Go to the Advanced tab of dfPhone and enter the following text into the Datasource Column field: 'Tel No.: ' + dbo.distributor.phone. Now on the Advanced tab of dfEmail, enter the text 'Email: ' + dbo.distributor.email. into the Datasource Column field. Be sure to save your changes as you go along.

Finally we need to tell i5 to only select data from the distributor table for the specific product that we are looking at. We can do this by setting a bind variable in the Advanced tab of the group's SQL properties. Do this by adding the following text to the SQL 'Where' statement field on the Advanced tab:

```
dbo.DISTRIBUTOR.distributor_id=:lbDistributorId
```

If you select a company, view its orders and then click through to an associated produced page, you should find that you get related distributor information appearing in the group we have just created. You should also find that you can click back to the Company detail page, with its associated orders. There are some minor complications – the table insert and multi-table update buttons won't work properly now that you have put the product page link into the child table on the Company detail page. There is a good reason for this. However we won't look at that here. You will need to have a read of the sister guide to this, the *i5 Developer's Guide* to understand that. For the time being you could simply make the Multi-Table Update and Table Insert buttons invisible on the page, so you aren't tempted to try and use them.

Summary

In this chapter we've taken a look at i5 groups and child tables. These are both powerful tools for the i5 application developer. Both extend its capabilities quite considerably by making it very easy to display information from different tables on the same page without compromising the simplicity of the default i5 page-building - one table per page - behaviour. The Multi-Table Update button in particular is of great value because of the way that it then allows you to save data updates to a number of tables simultaneously. If you are interested in finding out about some of the more advanced features of groups and child tables you should consult the *i5 Developer's Guide*, which takes a much closer look at how these components work.

8. Frame Rendering

Technically speaking, frames are simply a way to describe a presentation by dividing a browser window into two or more rectangular areas. From the i5 point of view, a frame is an area of the browser window which contains an i5 page and as i5 can split the browser window into multiple frames, it is possible to display more than one page in the browser at any one time. Each frame contains its own i5 page and collectively each page is part of the same i5 web application. In this chapter the aim is to follow the processes of designing frame layout and tying it together with the i5 pages in the application.

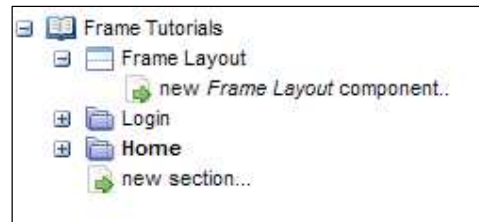
Creating a basic frame layout

Used intelligently, frames can be the key to organizing an application in such a way as to make it easier for a user to navigate. For example, there can be a table of contents in a frame on the left-hand side of the browser screen while the right-hand side is devoted to displaying the main content, the data for which changes each time the user clicks on a new topic in the table of contents. Frames also allow the right-hand side to change independently. Our first exercise is going to be to create such a layout.

The first thing you should do is create a new book. You can do this yourself now so go ahead and name it something meaningful to the context such as Frame Tutorials.

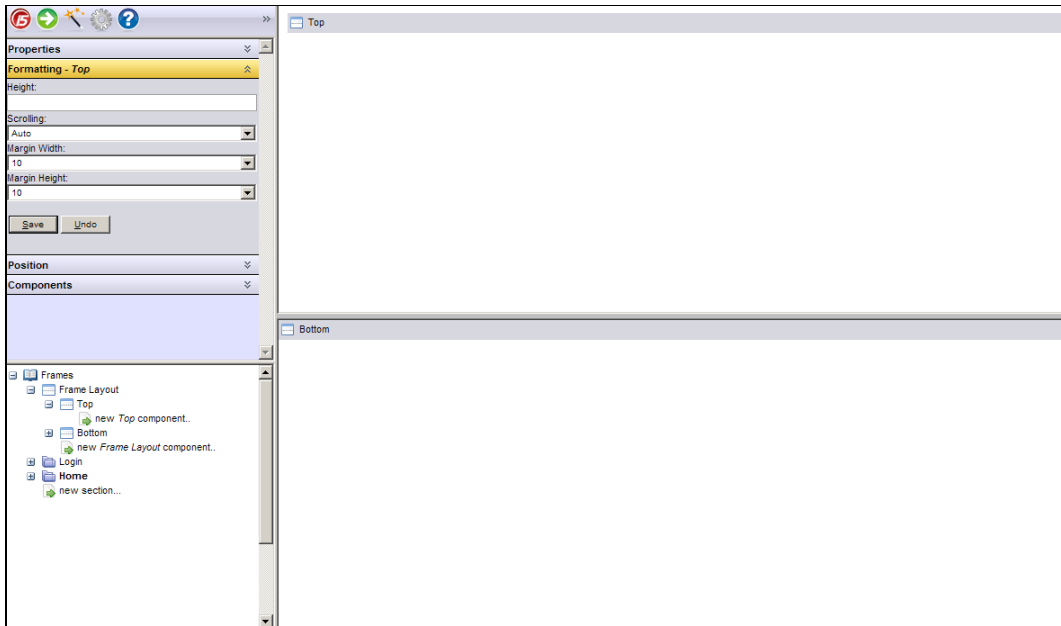
In the Explorer view you will notice how the book you have just created - like all the others - has its own Frame Layout section. This will contain any frame objects that you create. Clicking on the [Frame Layout](#) hyperlink in the Explorer view at the moment, however, will do absolutely nothing, because there are no frames in the current book.

Geometrically it should be clear that if you are going to add frames to an application you have to have a minimum of two frames.



To add some frames to a book, simply navigate in the Explorer to the Frame Layout section, click on the New Frame Layout component option and select Frame from the Component Assistant. Having done that, leave the Split property as horizontal, name the two frames Top and Bottom, click Next and then Finish.

Two new frames will have been created and have split the browser window through the middle, as in the screenshot below. If you resize the browser window, you will see that the split is always centred in the middle of the window.



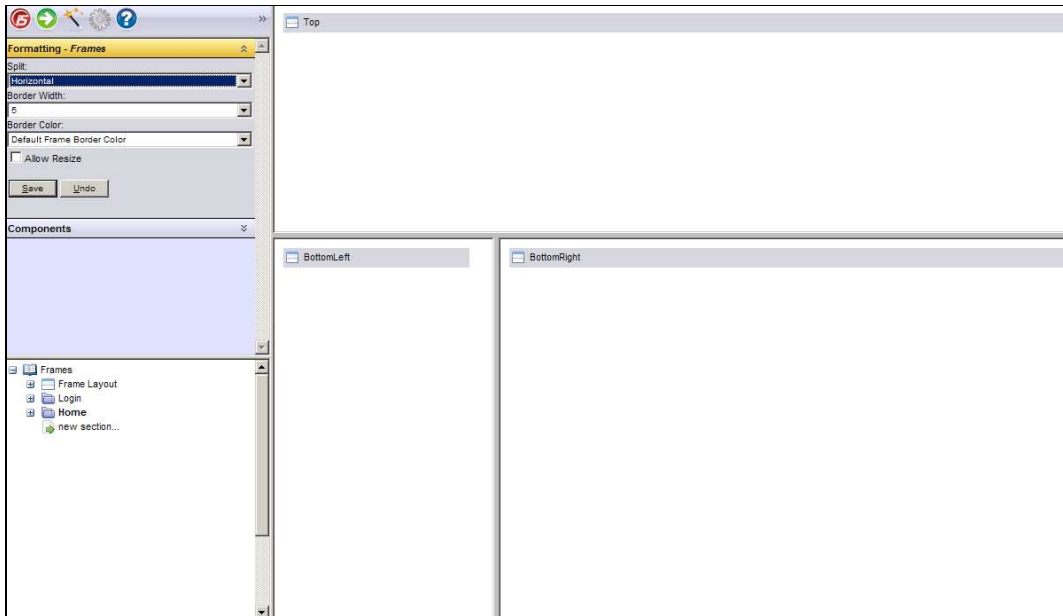
The problem with this, of course, is that you may not want both your frames to resize themselves automatically – particularly if one of your frames contains something – some graphics, say – which needs to remain in the same size ratio to the frame no matter how big or small the browser window is made. This problem can be solved very easily by fixing the height (in pixels) of a frame. If the frames split the screen vertically it would of course be the width of the frame that we would be fixing.

To fix the height of a frame - using the top frame as an example, click the Formatting property tab for the Top frame. Currently the Height property – which sets the height of the frame - is blank. This means that by default the frame will automatically be sized at 50% of the window. If you change the Height to 150 and click Save, you will find that the top frame in your frameset will now remain 150 pixels high, no matter how much you resize the window.

Clearly, being able to divide your browser window into two frames can make your life a lot easier. Better still though, because any frame can itself be subdivided, there is no limit to the number of frames you can create inside a browser (although it is obvious that there are practical and aesthetic limits as to how many frames you might want to deploy within a browser window).

Let's look at a practical use of sub-divided frames.

By splitting the bottom of our two frames vertically we can easily create a structure in which the bottom left hand frame could be used to display a table of contents, whilst the bottom right hand frame displays the main content. To do this, expand the Bottom frame component in the Explorer and select the Frame item from the Component Assistant to add two new frames. Call the two new frames Bottom Left and Bottom Right, change the Split to Vertical, click Next and then Finish. Navigate to the Formatting tab of the BottomLeft frame and set the frame's Width property to 250 and Save your changes. You should end up with a screen that looks something like the one below.



Putting pages into frames

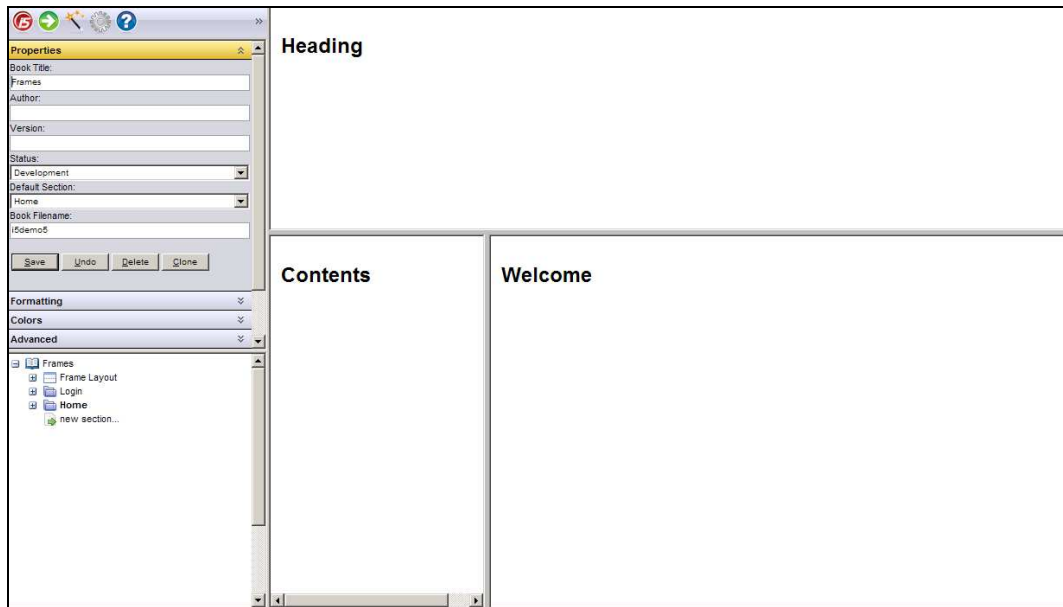
We have created some frames, and we now need to put some i5 pages into them. When doing this, it is important to be absolutely clear as to which pages go where. The mechanism for locating specific pages in specific frames is very simple – i5 allows the setting of default pages for any frames in an application at the section level. Let's see how this is done. We can begin by adding some more pages to the Home section of the book you are doing your frames tutorial in. Let's have a detail page called pgWelcome, with a title of Welcome Page. It is possible to set the Frame in which this page is to appear when you create the page. However, we will leave the Frame Name setting for pgWelcome as Default for the moment.

If you now click on the Home section in the Explorer you will see that the Welcome page automatically appears in the top frame. Unless you change the default frame formatting option of the section, i5 will always put the first page it finds into the first frame it finds - these will always be the first Frame and Page nodes for the Explorer view of the section in question.

To change from the default setting and display the welcome page in the bottom right hand frame of the book, you should click on the Formatting tab of the Home section and change the Default Page setting for BottomRight to pgWelcome. Whilst you are at it, change the Default Page setting for BottomLeft to Content.

All we are missing now is a heading page to display in the top frame as a title, so let's add a Detail page to the Home section, calling it pgSiteHeading and setting its title to Site Heading. Leave the Frame as Default. Click Next then Finish. Navigate to the Formatting property tab of the section Home, changing the Default Page for Top to pgSiteHeading then Save the changes.

The Home section should now look something like this:



With a little imagination you should be able to see how the three frame structure we have created can effectively become the home page - or more accurately, the home *view* - for an i5 application. By setting the default pages for each frame we can be sure that however we might choose to update pages in separate frames, the default presentation is always of a *title*, a *menu* and a *welcome* page.

If you wish to display pages from other sections of a book, you don't have to recreate them in the new section as the New Page Component Assistant allows you to add a reference to a pre-existing page, which you can then assign as a Default Page for a frame.

Linking to pages in different frames

i5 works on the basis that any specific page will always appear in the same frame, so in the example, we have developed above, the Contents Menu will always appear in the bottom left frame, and the Welcome Page in the bottom right. This philosophy makes development with frames easy because you don't have to worry about accidentally displaying the Contents Menu in the Site Heading if you link to the Contents Menu from the Site Heading.

Every section in an i5 application that uses frames allows you to set the default page for each frame. Once the default for each frame has been set, what do we do with other pages, those we may not want to display immediately? For example, it is unlikely that a search page is required in a home view, but it certainly needs to be placed somewhere.

i5 solves this problem by providing each page with a Frame property, and setting this informs the page which frame it displays in. The Frame property for a page can be accessed from the Tab view Position property tab. For a verification of this, click on the Position tab for the page pgWelcome and you will see that the Frame Name property for the page is already set to Bottom Right.

To illustrate the use of this i5 page feature, we will add a couple of content pages to our book and some page link menu items. Exploiting the structure of our frameset, the aim is to have the content pages appear in the main frame, and the menu items to appear on the menu in the bottom right-hand frame.

Let's start by adding a section to the book using the Page Wizard. It doesn't really matter which database table you build your section from – let's say, for example, that you add a Products section.

Adding a section in this way has the advantage that i5 will automatically create a page link to its Search page in the default page of the Contents section.

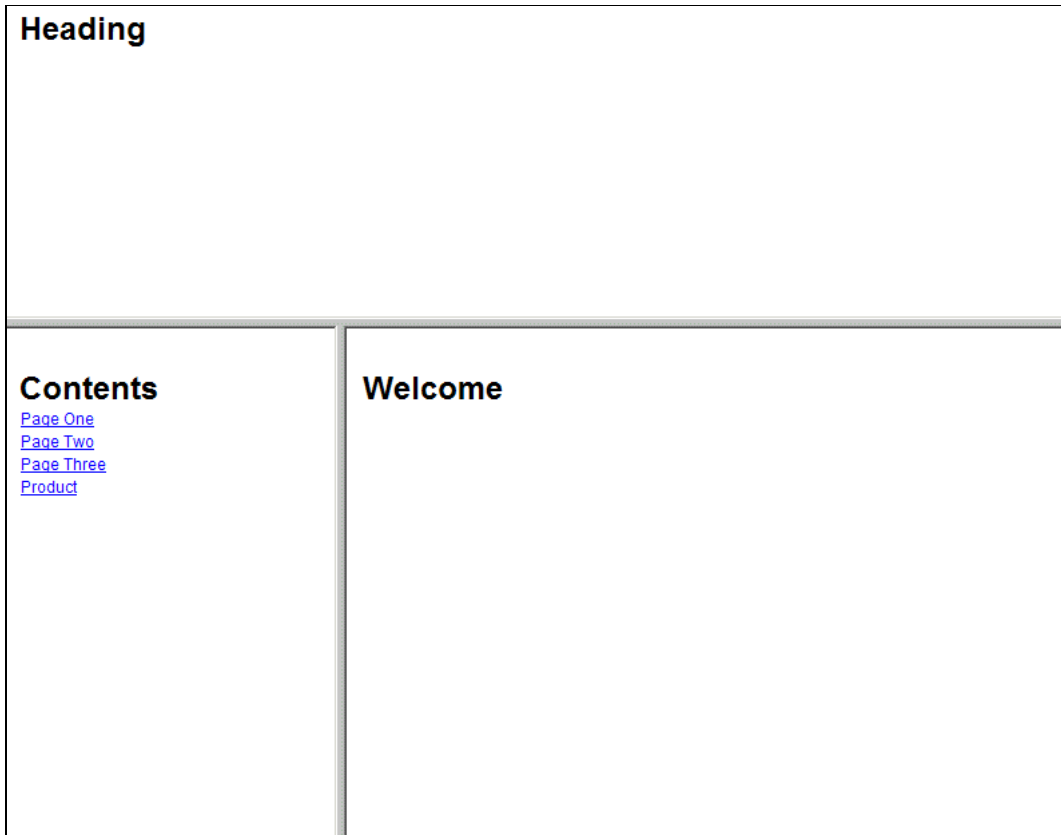
Once you've created the section, you need to assign pages to frames. At the outset, and following the simple 'rule' specified above, previewing the section will show that the search page displays in the first frame that it comes across – the top frame. Likewise, clicking on the 'Product' link in the Content page in the Home section will display the search page in the bottom left hand frame. We want the search page to display in the bottom right hand frame and we want to be sure that the top frame and the bottom left hand frame continue to display the page heading and the contents pages respectively.

You can set pages to frames by defining the Frame Name setting on each page's Position tab. Doing so will ensure that that page always displays in the frame you wish it to appear in. At the section level, you can also define the default pages for each frame. Whilst navigating to a new page in one frame will not necessarily result in all frames refreshing, it is a good idea to set the default page for each section so as to avoid unpredictable behaviour when a user accidentally or deliberately refreshes individual frames.

In our example, set the Frame Name property for the Search Browse and Detail pages of the Products section to the BottomLeft frame. From the Section Formatting tab, set the default pages for each frame so that the top frame displays the Heading page, the bottom left hand frame the Contents page and the bottom right hand frame the Products Search page.

Previewing the book you should now find that a consistent behaviour is displayed as you navigate through pages. You could easily add to this – creating say a 'Contacts' page, for example, or a link back to our 'Welcome' page.

It is worth noting that the default Content section that i5 creates can be added to – you can add links to pages that have not been created automatically by the Page Wizard. This makes it easy to create pages which may not necessarily be 'data-aware' (that is, linked to a database table) or which require complex business logic.



Formatting frames

We now know how to create frames, how to set default pages for the frameset of a book so that we can create a home view, and how to make sure that individual i5 pages display in the frame of choice. There remain two issues to cover:

- How to format frames so they display in the way we wish them to;
- How to resolve the issue of what happens when an i5 application is to be displayed in a browser that does not support frames? This is a situation which pertains in particular to some hand-held devices and earlier browsers.

Formatting

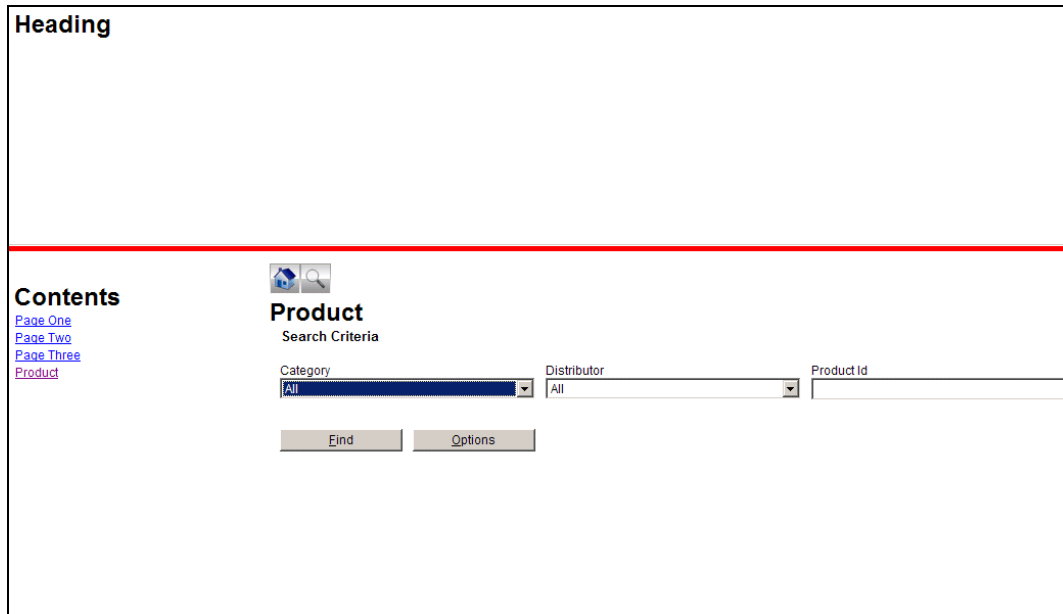
The list of elements and attributes in an HTML frameset is a short one and as a result, formatting options are somewhat restricted. i5 allows you to set the width of any frame border that you use, including the possibility of turning it off by setting its width to zero. It will also allow you to specify the colour to use in the rendering of a frame border and the margin on the interior of the frame (this determines the gap between the border of the frame and the components it contains).

Frame borders are adjustable on all frames which contain children and their settings affect the borders of all immediate child frames. In the current example, the settings for the border that exists between the frames BottomLeft and BottomRight are found on the parent frame Bottom.

To turn off the border between the frames BottomLeft and BottomRight and to change the colour of the border between the frames Top and Bottom to red, you would need to:

- Navigate to the Formatting tab of the frame Bottom and change the Border Width to 0, and Save your changes

- Click on the Formatting tab of the frame Frame Layout and change the Border Color to Red, remembering to Save the changes.
- Change the Margin Width and Margin Height settings to 10 pixels on the Top, BottomLeft and BottomRight frames. Your browser window should now look something like the screenshot below (we've added a Product section to the book and set its pages to display in the bottom right hand frame).



Frame free browsers

The standard approach given by mainstream web development tools to the support of frames is a simple one - either you use frames, or you don't! As a result, many website designers have to make a decision either to build frame-free sites or to build sites that will not work on primitive browsers. It is possible for a really dedicated designer to implement 'back-out' strategies, but these tend to be laborious to create and are still mainly browser specific.

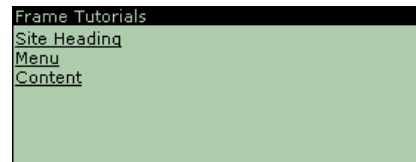
The support for frames in i5 allows the website designer to benefit from the flexibility of frame layout where the client browser supports frames. Nearly all HTML browsers support frames in one form or another, and i5 will render the correct frame layout code for those browsers. Where i5 finds a browser which does not support frames, it will automatically implement a presentation policy that allows the site to continue to work on frame free browsers. As mentioned previously, browsers which do not support frames are typically those on mobile phones and handheld devices.

To cope with the problem of dealing with a browser which does not support frames, i5 adopts a 'back-out' policy – but one that does not make the unrealistic aim of emulating a framed site in the frame-free browser. What i5 does, instead, is simply to treat the application as if it didn't use frames at all and then to ensure that it is always the most important page that is kept in view in the browser at all times. To do this, i5 ensures that all the pages in the application are rendered in the main browser window and that clicking a link to select a new page in an application will result in the new page replacing the displayed page rather than having it appear in a different frame.

Where there are a number of pages which would be displayed simultaneously in the framed version of the application - as with our *Home* example from the previous section where we had a site heading, a menu page and a welcome page - i5 will present the user with a list of the pages which it would like to render. The user then simply selects the page of interest and continues accordingly.

An example will make this a little clearer.

As the user enters the application, i5 displays a list of the available pages:



The user clicks on the link [Home Menu](#) and is presented with the main menu



The user clicks on the link [About Us](#) and is presented with the About Us content page:



As this example suggests, for the list to be meaningful to the end user, it is important that the pages be given intelligible titles. Failing to do this will result in i5 generating lists of pages which, for the end user, make little or no sense given the context.

9. Case Study - i5demo

The i5demo site is a demonstration of how to build a simple application using i5. It has been designed as a starting point for any i5 developer who wants to build a basic product site. By swapping the graphics and the products, it is easy to adapt the i5demo architecture and core functionality to re-purpose and re-brand it.

More significantly however, i5demo serves as a useful introductory example to how you can use business logic to extend and refine the functionality of data-driven websites developed using i5studio. In this sense the i5demo case study points towards the sister manual the *i5 Developer's Guide*, away from the world of point and click and into the world of computer programming.

There is an awful lot that you can accomplish with i5 without having to write a single line of code. However, if you really want to benefit from this chapter's discussion of how i5demo works, you will need to look at some code. You don't actually have to write any code here: we are only going to show you example of how you can begin to enhance your website using run time criteria. We're saving the delights of business logic programming for the Developer's Guide.

Before we start

The database storing all the products, reviews and promotion records, along with the C# source code required to power the website, comes with i5 when you install it, so nothing extra is required to view the website itself. However, if you do wish to examine the source code – and we strongly recommend that you do – you will need to have a copy of Microsoft's Visual Studio 2005 installed on your computer.

You will also need to know a little bit about business logic. We haven't covered business logic in this Guide at all, so you should put this Guide down and take a look at the first chapter of the *i5 Developer's Guide* ("Getting Started with Business Logic"). Reading this won't make you an expert but it will give you an idea of the basic mechanics of adding business logic to i5 pages. A more general willingness and ability to read through and understand computer code would also be helpful.

Starting the Business Logic Server

If you've done everything we've asked you to do so far, you should be more or less ready to get started. Because i5demo uses business logic, you will, in the first instance, need to start your business logic server. If you've got this running as a service, no problem – it will be running already. If not, you will need to start it from the Windows Start Menu (go to: Programs > i5 > Server > i5 Business Logic Server (console mode)).

Viewing the i5 book

Next, you will want to inspect the pages of the i5demo book in i5 Studio. To do this, you will need to open up i5 Studio and then log in to i5demo. We've helpfully called the database for this application i5demo, and the user name/password combination is sa/sa. On successful login you should then select the i5demo – The Essential Essence book from the list of i5 books within "Release".

Viewing the site

Viewing the book is just like viewing any of the work you have produced using i5 so far. You should click on the Open Book icon on the main toolbar in i5 Studio in order to call up i5 Runtime. Doing this will present you with the following screen:

The screenshot shows a website titled "TheEssentialEssence" with a light blue background and decorative sunburst graphics. The navigation menu includes "Home", "Search", and "My Wish List". A welcome message states: "Welcome to the Essential Essence! This site was created as a demonstration of how to build a simple application using i5. It was designed as a starting point for any i5 developer who wishes to build a product review site by simply swapping the graphics and the products. In this case the graphics and product database has been created for the fictitious perfumes and toiletries company - TheEssentialEssence.co.uk".

The main content area features a product detail for "Versace by Gianni Versace". It includes a breadcrumb trail: "For Women > Eau de Toilette > Gianni Versace". The product image shows a yellow perfume bottle. The description reads: "A delicate floral fragrance, perfect for evening use has been created through blends of bergamot, ylang-ylang and amber." The product is rated with five stars by one user, and the recommended retail price is £26.78.

Below the product details, there are two sections: "Site Facilities" and "Site Components".

Site Facilities

- Advanced product searching
- Wish list
- Customer review and rating facility
- Promotion listings

Site Components

- i5 studio using the "i5demo" book
- .NET Business Logic
- i5demo database

If it doesn't, something is wrong.

Notice that the graphics are for a company called TheEssentialEssence.com, a fictitious perfume review site.

Viewing the Business Logic code

You will need to be familiar with Microsoft's Visual Studio development environment to view the Business Logic code. To do this:

1. Start Visual Studio 2005 and open the project i5demo.csproj (it is located in the i5demo\Source subdirectory under your installation of i5).
2. Go to the View menu and select Solution Explorer. Amongst others you will see two files, ProductDetail.cs and WishList.cs.
3. Double click on ProductDetail.cs to view the Business Logic code.

The ProductDetail file contains the business logic for the Product Detail page in the Product section of the i5demo book and the WishList file contains the business logic for the Detail page in the WishList section of the i5demo book.

i5demo site overview

Let's begin by taking a look at the overall organisation of the i5demo website. By today's standards, i5demo is not a very complicated website at all. However, with a little imagination, it is not difficult

to appreciate that the kind of functionality on which it is based (i.e. what the site does and/or will allow the end user to do), is pretty much core to any online product listing – including the giants such as Amazon. Of course, we’re not comparing ourselves to Amazon, we are simply saying that there are underlying patterns of functionality which our modest little effort and some commercial websites do share in common.

Areas of site

The i5demo application has been organized into three main areas, each corresponding to an equivalently named section in the i5 book containing all the pages that are necessary for use in that area of the website.

Home area	This contains all the general purpose pages of the site. Currently only the home page is contained in this area.
Product area	This holds all the pages and Business Logic used for customer product browsing.
Wish List area	This area contains the page for users to review the products they have added to their “Wish List” and the ability to remove products or clear the list.

Product searching and viewing

This section describes how the searching and displaying of products is implemented in i5demo. All products are held in the PRODUCT table, however the searching and displaying of products will also include data related to the product category (from the CATEGORY table), and the product brand (from the DISTRIBUTOR table).

Product Searching

There are three types of product search available

- By product type
- By brand
- By price

The i5demo product types include body lotion, eau de toilette, and foam bath. This product type drop down list allows the user to search specifically for one type of product or to leave this filter as “All product types”.

Searching by Brand of product, as you might expect, allows a customer to choose from a list of product brands. When a product brand is chosen, all products with this brand will be displayed in the search results.

Price searching is implemented through a minimum and maximum price. If only a minimum price is entered, the results will include all products which exceed this price. Likewise, if only a maximum price is entered, then the results will contain only products which cost less than that entered.

Product searching in i5demo is accomplished by some filter fields and a result product group. A page link is used to pass customer input from the search pane to the results pane, where the search is actually processed.

Let’s have a more detailed look at the mechanics of this process:

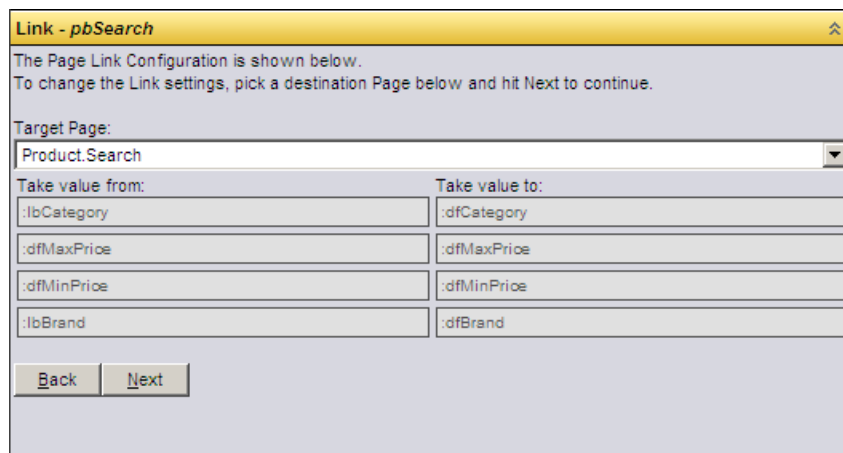
In i5 Studio, open the Product section, then expand and preview Search:



All the objects on the Search screen have been created in i5 Studio and no Business Logic has been used.

If you expand the group grpProductSearch you will see the data entry items which are used for filtering the search results: lbCategory; lbBrand; dfMaxPrice; dfMinPrice; pbSearch. The pagelink pbSearch is used to refresh the page passing the values from these objects to the results group. This is done through the use of some page level objects which are accessible to the group.

All this can be verified by looking at the Link property of pbSearch:



Now to look at the results group, expand grpProductList.

When you first access the page no default search criteria are defined, so all products are available in the result list. If you try selecting “Elizabeth Taylor” from the brands drop down list and clicking on Search, you will see that only the products by Elizabeth Taylor are displayed.

Clicking on the picture or the title for one of these products will take you to Product.Detail where further information is displayed and Business Logic is used.

Product Viewing

The viewing of products in i5demo is handled by a standard i5 detail page, Product.Detail, which displays information about the selected product, including a photograph of it, also held in the database.



The page is constructed of five child containers, each with a very distinct purpose.

- grpProduct, displays the information about the product, including its title, image, average user rating and a link to add the product to the wish list.
- grpTabs, contains the links to appear as tabs for viewing existing reviews, adding a new review, and seeing any promotions the product is currently part of. These links are controlled using javascript to show and hide the other page content depending on which of the links is clicked.
- The product reviews are displayed within the group grpReviews which has a primary datasource of dbo.REVIEW. A simple bind set in the QueryWizard on the SQL properties for this group limits the result set to only showing reviews for the current project.

- New reviews are added through the grpNewReview component, which contains user entry fields for name, review and product score. This group is not data aware as the insertion of new reviews is handled by Business Logic.
- The last container is a child table, ctPromotions, which selects all promotions for this product within the current date range.

As already mentioned, this page does use Business Logic. Its purpose is to allow a user either to write a review for the product, or to add it to their wish list.

In Visual Studio (and assuming that you have located and opened up the i5demo C# project), open the ProductDetail.cs file and locate the class ProductReview. This class contains just one function, OnCreate. This function is called by i5 every time grpReviews is about to be drawn by i5.

Here are the first few lines of code,

```
protected override void OnCreate(Container group)
{
    base.OnCreate(group);

    // Check if any reviews have been recorded for this product
    if (group.RowCount < 1)
    {
        group.RowCount = 1;
        group.Field("dfText").StrValue = "There are no reviews. Be
the first person to review this product...";
        group.Text("txtLine").Hidden = true;
    }
}
```

This section of the code is really very simple: it checks the number of rows within the group to see if it contains anything and then sets the number of rows and value of the dfText field accordingly.

Also within the ProductDetail.cs file is a class called ProductDetail. This class contains two functions; OnCreate and OnSubmit. The OnCreate function checks for any errors which may have been thrown during the OnSubmit event and presents them to the user in the txtError object:

```
// Check for any exceptions which may have been thrown when adding a
review
if (page.Session.PageVar("Exception") != null)
{
    Exception ex = (Exception)page.Session.PageVar("Exception");
    page.Text("txtError").StrValue = ex.Message;
    page.Session.PageVar("Exception", null);
}
}
```

The remainder of the code in the OnCreate function is to inform the user of whether or not the product currently selected is in their wish list or not. The wish list information is stored within a cookie, so the code uses the IndexOf() function to determine whether the cookie contains the current product id. If it finds the id within the cookie value then it hides the link to add the product to the wish list and shows a “This product is in your wish list!” message instead:

```
if(page.Session.Cookie("i5demo").IndexOf(page.Field("dfProductId").Str
rValue) >= 0)
{
    // Product is in the wish list, so lets let the user know
    page.Group("grpProduct").PageLink("pbWishList").Hidden = true;
    page.Group("grpProduct").Text("dfWishList").Hidden = false;
}
}
```

The OnSubmit function within the ProductDetail class is a little more complicated. Here we see how a product is added to a wish list, and how a new review is submitted to the database.

The first few lines of the code are to add a product to a wish list, which is set in a browser cookie. The first check it makes is to determine whether the link clicked was instructing the Business Logic to add the product to the wish list. This is done by comparing our link name "grpProduct.pbWishList" with the name of the link pressed within the i5 API. Only if this is true then is the product added:

```
if (page.LinkPressed == "grpProduct.pbWishList")
{
    // Add this item to my wish list
    string wishList = page.Session.Cookie("i5demo");
    if (wishList != "") wishList = wishList + ",";
    wishList = wishList + page.Field("dfProductId").NumValue;
    page.Session.Cookie("i5demo", wishList);
}
```

To add the product to the wish list, the current wish list content is read from the cookie. If the cookie already contains values, then a separating comma is added to the end of the value. Then the id of the product is retrieved from the dfProductId field on the page and added to the end of the value. Lastly, the updated cookie value is written back to the cookie. The product has now been added to the wish list!

Another if statement follows this, determining if the user clicked on the Submit button to add a new review.

```
if (page.LinkPressed == "grpNewReview.pbSubmit")
```

Following this, the new review is inserted into the i5demo database using the Container.Session.Database object. First the SQL statement is constructed using the values retrieved from the i5 page. The statement is then prepared using the .Prepare() method and the data passed in using .StrBind(). If successful, the .Execute() method is called and then finally the .Commit().

```
SQL conn = page.Session.Database.Connect();
try
{
    using (Container grp = page.Group("grpNewReview"))
    {
        string name = grp.Field("dfName").StrValue;
        string email = grp.Field("dfEmail").StrValue;
        string command = "INSERT INTO [dbo].[REVIEW] " +
            "([PRODUCT_ID],[TEXT]," + (name != "" ? "[NAME]," : "") +
            "[RATING]) VALUES " +
            "(:id, :text, " + (name != "" ? ":name," : "") + " :rating)";
        if (!conn.Prepare(command)) throw new Exception("Error in
SQL");
        if (!conn.StrBind("id", page.Field("dfProductId").StrValue))
        throw new Exception("Error creating bind");
        if (!conn.StrBind("text", grp.Field("dfText").StrValue))
        throw new Exception("Error creating bind");
        if (name != "") if (!conn.StrBind("name", name)) throw new
Exception("Error creating bind");
        if (!conn.StrBind("rating",
grp.RadioGroup("rgRating").StrValue)) throw new Exception("Error
creating bind");
    }
    if (!conn.Execute()) throw new Exception("Error executing SQL");
    conn.Commit();
}
```

A try/catch is used to respond to any errors which are thrown here, by executing the `.Rollback()` method. Finally our connection handle is freed by calling `.Disconnect()` on it.

```
catch (Exception ex)
{
    page.Session.PageVar("Exception", ex);
    conn.Rollback();
}
finally
{
    conn.Disconnect();
}
```

For further information about the issues discussed here, please refer to the chapter on ‘Databases, Connectivity and Child Containers’ in the *i5 Developer’s Guide*

The Wish List

Logically there are three aspects to this functionality: identifying a user and the content of their wish list, adding a product to a wish list and then removing them.

How a user is identified

As we have already mentioned, i5demo uses browser cookies to track the content of a wish list. This enables many users to have independent wish lists, and also means that the application “remembers” the content of a user’s wish list from one day to the next, as long as they use the same computer and do not clear their cookies.

Viewing the wish list

The page which is used for displaying the content of the current wish list is called Detail and is located within the WishList section. If you expand and preview this page in i5 Studio, you will see it contains a message to the user as well as a child table called `tblWishList`.

The Business Logic associated with this page is located in the file `WishList.cs`. The file contains just one class with two functions: an `OnCreate` function to populate the child table with the correct content, and an `OnSubmit` which handles the removal of items from the wish list.

The `OnCreate` method is responsible for constructing the SQL used to populate the child table “`tblWishList`”. It begins by retrieving the content from the “i5demo” cookie and converting all URL encoded commas back into comma characters.

Then the cookie string value is split into an array, identifying a comma as a delimiter. This array is then used to construct a properly formatted comma separated string list which can be used within the SQL select statement.

```
// Undo URL encoding if occurred
page.Session.Cookie("i5demo",
page.Session.Cookie("i5demo").Replace("%2C", ","));

// separate out the comma separated list into an array
char[] separators = {','};
string[] items = page.Session.Cookie("i5demo").Split(separators);

// Loop through the array reconstructing a new array acceptable for
// use by the child table
string inList = "";
foreach (string item in items)if (item != "") inList = inList + item
```

```

+ ",";

// Strip off the final comma
if (inList.Length > 0) inList = inList.Substring(0, inList.Length -
1);

// Store the new tidy list back in the cookie and in the page
page.Session.Cookie("i5demo", inList);
page.Field("dfWishList").StrValue = inList;

```

Once this is complete, the comma separated string is used to construct the WHERE clause of the child table SQL. This is done with the use of an IN statement.

If the comma separated string is found to have no content then the wish list table is hidden from the user.

```

if (inList != "")
{
    // Set where clause of child table to use our array
    page.Table("tblWishList").Query = "PRODUCT_ID IN (" + inList +
    ")";
    page.Text("txtWelcome").StrValue = "Your wish list is shown
below. To add more items to your wish list, please search for
products. To remove an item from your list, use the links provided.";
}
else
{
    // Wish list empty, so hide table and remove all button.
    page.Table("tblWishList").Hidden = true;
    page.PageLink("pbRemoveAll").Hidden = true;
}

```

Removing items from the wish list

Also from this screen, the user may remove a single item from their wish list or empty the list completely. This is done either by pressing the “Remove from my wish list” link next to a particular product or by clicking the “Clear wish list” button at the bottom of the screen.

Business Logic code in the Submit event determines which link has been pressed and performs the appropriate action accordingly.

If the user requests that just one item is removed from the wish list, the business logic begins by identifying which product should be removed. It does this by retrieving the value from the hidden field dfProductId in the record that was clicked. This record is the default row context in the table.

Once the id has been removed, the Replace() function is used to replace the string value of the product id with an empty string, thus removing it from the list.

```

if (page.Session.LinkPressed == "tblWishList.pbRemove")
{
    // Remove item from wish list
    string item =
page.Table("tblWishList").Field("dfProductId").StrValue;
    page.Session.Cookie("i5demo",
page.Session.Cookie("i5demo").Replace(item, ""));
}

```

Alternatively if the “Clear wish list” is clicked then the business logic can simply clear out the entire content of the cookie, and therefore deleting all products in one go.

```
if (page.Session.LinkPressed == "pbRemoveAll")
{
    // Remove all items from wish list
    page.Session.Cookie("i5demo", "");
}
```

Conclusion

In this whistle-stop tour of the code which we used to build the i5demo website, we have looked – albeit in a rather cursory way – at the core functionality of a product listing website. In the grand scheme of things, of course, i5demo is not the most complicated of websites, but as we said at the start of this chapter, its relative simplicity should not belie the generic quality of the functionality on which it is based. If you have never looked at computer code before, then it is possible that some of the discussion above will not make that much sense at first. However, if you have read this guide from the start we hope that the leap from pointing and clicking in i5 Studio to the programmatic creation of i5 pages using business logic is not too great. In any case, how the code accomplishes the tasks required of it will become clearer if you a) play with the site and b) re-read the code. We are not claiming here that on reading this chapter you will be able to build an online product listing website from scratch – that wasn’t the aim of the exercise at all. What we have tried to do here is to de-mystify the functioning of a dynamic data-driven website and show you how business logic works in conjunction with the kind of i5 functionality you have learned to build using i5 Studio. We’ve also tried to whet your appetite for building complex websites using business logic: if the refinements and complexity which computer programming can introduce interests you, we suggest that you now move on to the *i5 Developer’s Guide* for some more advanced tutorials about working with i5.