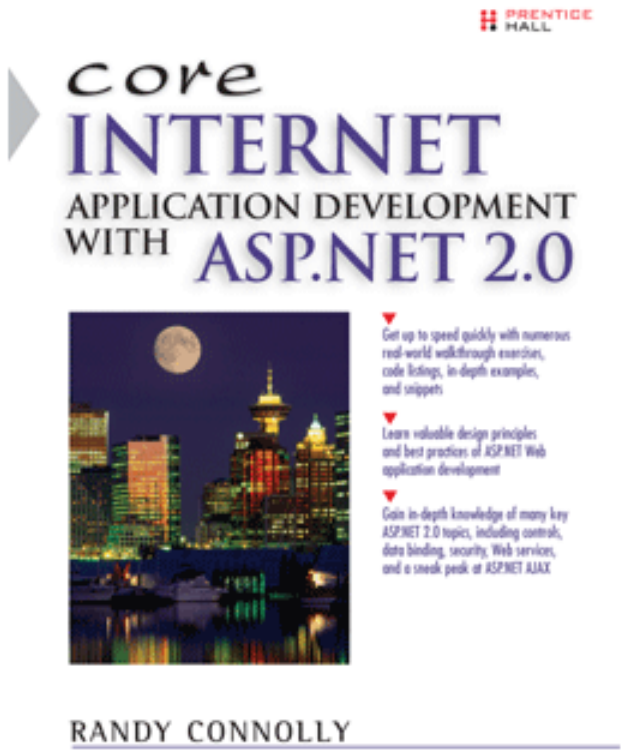
Chapter 8
Data Binding and Representation

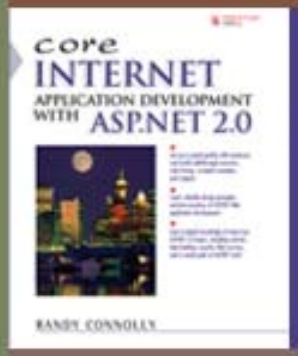
It is a capital mistake to theorize before one has data.

Sir Arthur Conan Doyle, *The Adventures of Sherlock Holmes*, "Scandal in Bohemia"



Overview

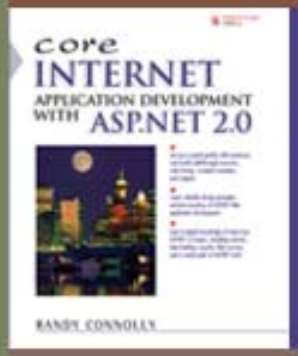
- This chapter examines how data can be represented with the .NET Framework and how this data can be displayed using data binding in ASP.NET.
- It covers:
  - Data binding in ASP.NET
  - Arrays
  - Collection classes
  - Generics
  - DataTable and DataSet



CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

Data Binding



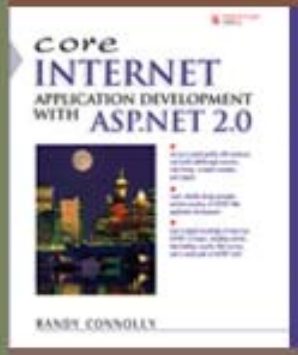
CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

- Almost every Web application displays data from some type of data source.
  - i.e., a typical Web page reads data from some external source, usually a database or an XML file, and presents it to the user.
- Thus the developers of ASP.NET added a feature called **data binding** that facilitates displaying data from a data source in a control.

Data Binding

- Data binding refers to the process of dynamically assigning a value or values to a control at either design time or runtime and having the control automatically display the value or values.
- That is, you only need tell the control where it can find its data, and the control handles the process of displaying it.



CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

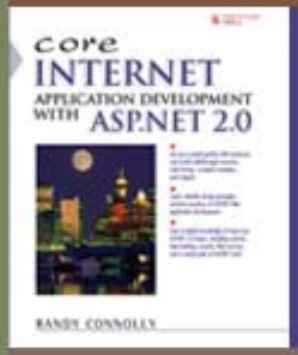
Prentice Hall, 2007
www.randyconnolly.com/core

Data Binding

- Data binding a control to a data source can be achieved in one of two ways:
  - Setting the `DataSource` property of the control and then calling the `DataBind` method of the control.
    - E.g.,

```
aControl.DataSource = someDataSource;
aControl.DataBind();
```

  - Setting the `DataSourceId` property of the control to a data source control.
    - The second approach was introduced in version 2.0 of ASP.NET and will be examined in the next chapter.

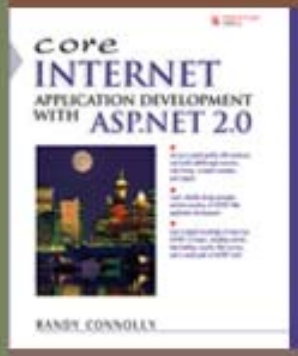


CORE INTERNET APPLICATION DEVELOPMENT WITH ASP.NET 2.0

Prentice Hall, 2007

www.randyconnolly.com/core

What Can Be a Data Source?

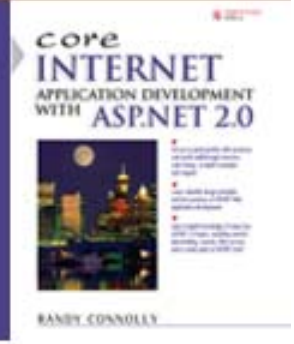


CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

- The object that is to be assigned to the `DataSource` property of a bindable Web server control must be one that implements, either directly or indirectly, the `IEnumerable` or the `IListSource` interface.
- These are:
  - Arrays
  - In-memory .NET collections classes, such as List and Dictionary
  - ADO.NET `DbDataReader` objects
  - ADO.NET `DataView`, `DataTable`, or `DataSet` objects
  - Any custom class you create, which also implements the `IEnumerable` interface

Data Binding an Array

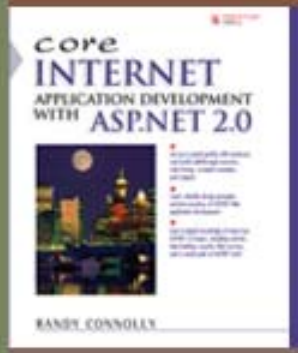



```
<asp:DropDownList id="drpSample" runat="server" />
```

```
private void Page_Load(object sender, System.EventArgs e)
{
   string[] names = new string[3] { "Austen","Dante","Goethe" };
   drpSample.DataSource = names;
   drpSample.DataBind();
}
```

Using Collections

- .NET Framework provides a number of **collection classes** that can also be used for data binding
  - in fact, an array is just a special type of collection class.
- A collection class is used to group related objects together.

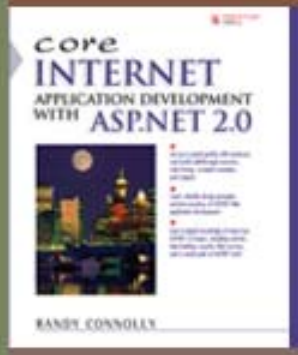


CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

Collection Classes

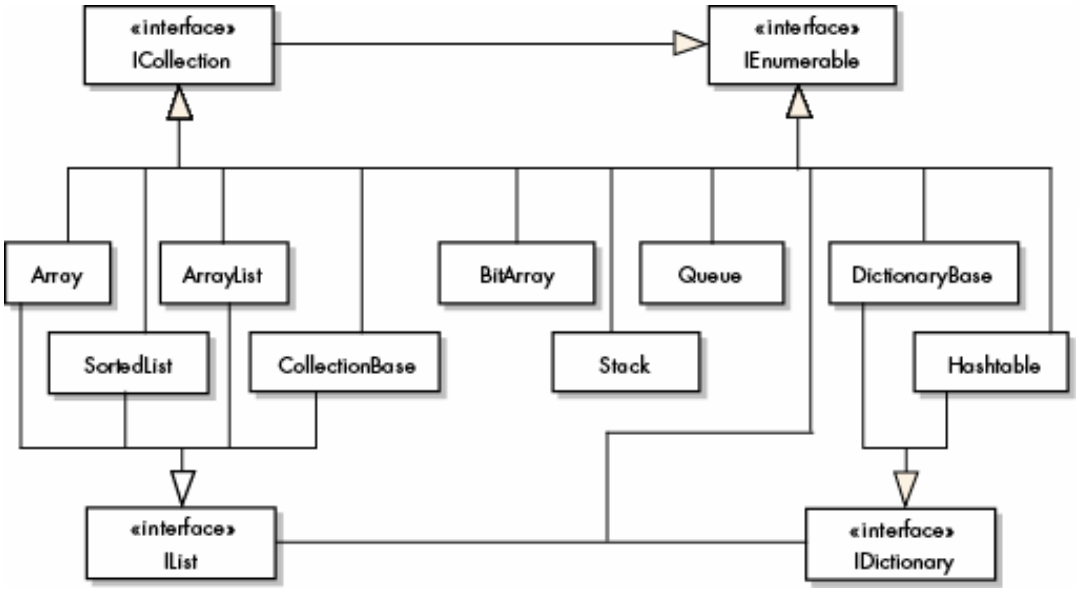
- There are a number of different types of collection classes.
- There are queues, stacks, sorted and unsorted lists, and dictionaries.
- Some of these collections can hold any type of object, whereas others are strongly typed.
- They provide members for adding, removing, and retrieving data elements.
- All collections provide a mechanism that makes it easy to iterate through its elements.
- Collections expand automatically as necessary.
- There are also special **generic** versions of the collection classes that allow you to create strongly typed collections.



CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

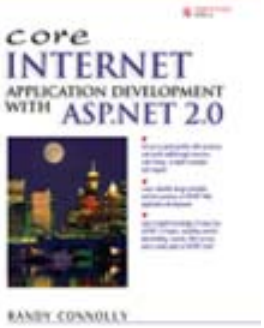
Collection Interfaces



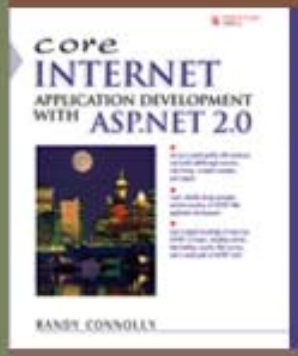
CORE INTERNET APPLICATION DEVELOPMENT WITH ASP.NET 2.0

Prentice Hall, 2007

www.randyconnolly.com/core



Using the Collections

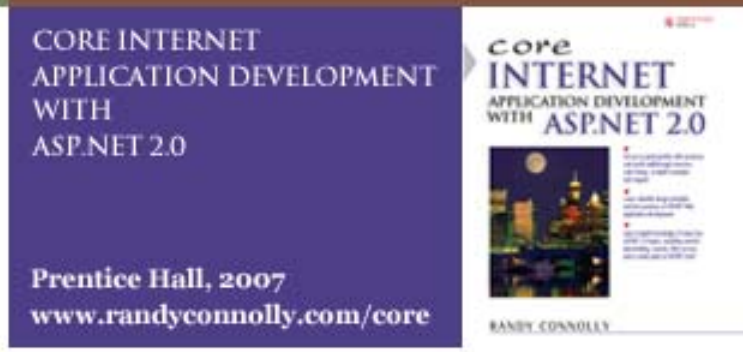


CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

- Perhaps the two most commonly used by programmers are the `ArrayList` and `Hashtable` collections.
- The `ArrayList` collection is ideal when:
  - data needs to be principally accessed sequentially (that is, accessed from beginning to end)
  - speed of searching for an element is not too vital.
- The `Hashtable` collection, by contrast, is optimized for quick nonsequential retrieval of elements via a key.

Using an ArrayList

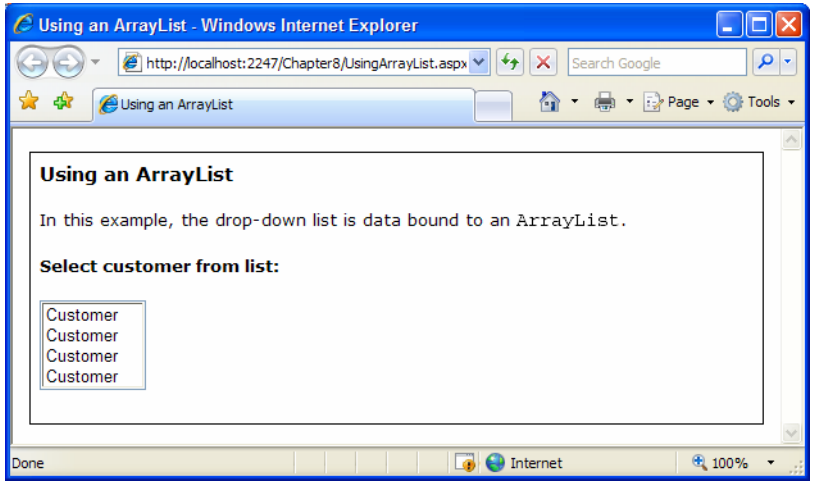


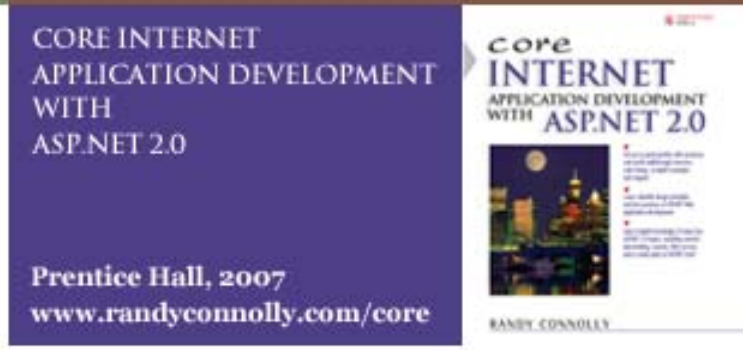

```
ArrayList myList = new ArrayList();

…

Customer c1 = new Customer( … );

myList.Add( c1 );
```

```
int index = …

Customer c = (Customer)myList[index];
```

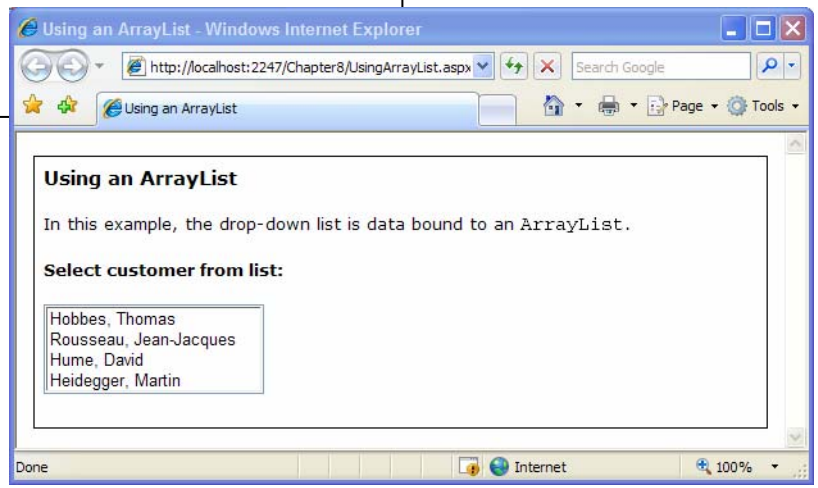
```
// Create sample customer objects
Customer c1 = new Customer("334", "Thomas", "Hobbes", "123-4567");
Customer c2 = new Customer("123", "Jean-Jacques", "Rosseau", "456-1267");
Customer c3 = new Customer("085", "David", "Hume", "564-7823");

// Create and populate collection
ArrayList myList = new ArrayList();
myList.Add(c1);
myList.Add(c2);
myList.Add(c3);

// Data bind collection to control
lboxCustomers.DataSource = myList;
lboxCustomers.DataBind();
```

Using an ArrayList



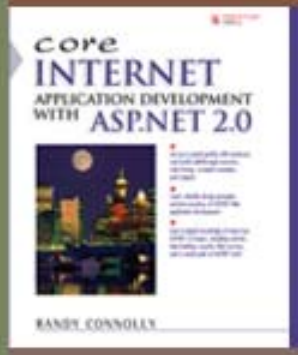



```
<asp:ListBox id="lboxCustomers" runat="server"
    DataTextField="Name"
    DataValueField="Id" />
```



Problems with Standard Collection Classes

- Because all .NET classes have the `Object` class as their ultimate ancestor, this means that collections such as `ArrayList` and `Hashtable` can store and retrieve any .NET object.
  - Requires casting to appropriate type
- Thus there are some drawbacks:
  - there is a potential for a casting exception if the data in the collection is not of the expected class.
  - There is a performance penalty with the casting operation.
- In version 2.0 of the Framework there are now generic collections as a solution to these problems.

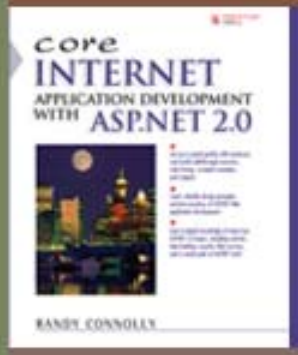


CORE INTERNET APPLICATION DEVELOPMENT WITH ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

Generics

- Generics let you define a type-safe collection without maintenance or performance drawbacks.
  - They shift the burden of type-safety away from the developer and give it instead to the compiler.
  - This reduces the possibility of runtime errors and generally makes for cleaner code.
- refers to the new capability in version 2.0 programming languages to support the use of placeholders rather than explicit type specifications when defining classes, structures, interfaces, or methods.
- You can create your own generics as well as use any of the generic collection classes contained within the `System.Collections.Generics` namespace



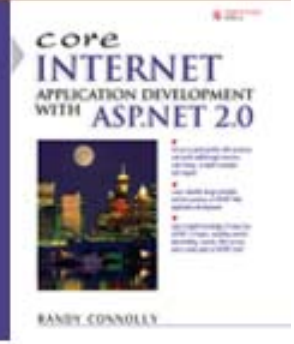
CORE INTERNET APPLICATION DEVELOPMENT WITH ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

Using Generics

CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core



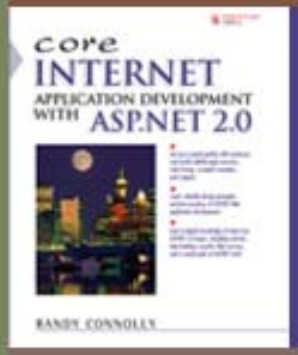

```
List<int> numbers = new List<int>();


List<Customer> customers = new List<Customer>();
```

```
numbers.Add(47);


customers.Add( new Customer("085","David","Hume","564-7823") );
```

```
int n = numbers[0];


Customer c = customers[0];
```

Dictionary Collections

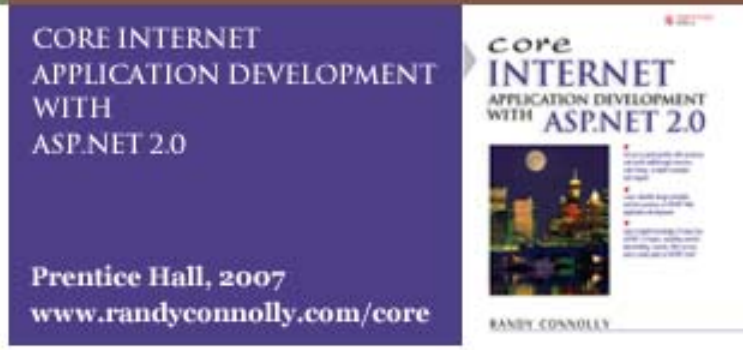
- With an `ArrayList` or `List` collection, you retrieve an individual element using its ordinal position within the collection.
- There are times, however, when it is not convenient to retrieve an element by its position in the collection.
  - e.g., if you had a collection of `Customer` objects, it might be much more useful to retrieve a specific customer by using the customer name.
  - You can do so by using a dictionary-style collection such as `Dictionary` or `SortedDictionary` (or the older, nongeneric `Hashtable`).



CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

Using a Dictionary

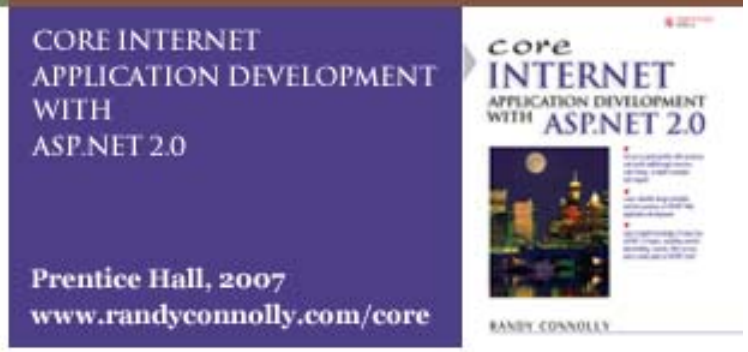



```
Dictionary<string, Customer> dict = new Dictionary<string, Customer>();


String id = "085";
Customer c = new Customer(id, "David", "Hume", "564-7823");
dict.Add(id, c);
…
Customer c = dict[key];
```

Using a Dictionary

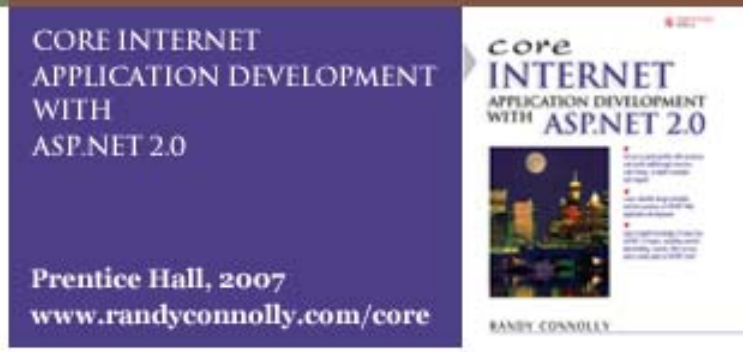

```
List<Customer> myList = new List<Customer>();
…
foreach (Customer c in myList)
{
   if (c.Id == valueToFind) {
     // Do something with this customer
   }
}
```

```
Dictionary<string, Customer> dict = new
    Dictionary<string, Customer>();

Customer c = dict[valueToFind];
if (c != null) {
  // Do something with this customer
}
```



Iterating a Dictionary

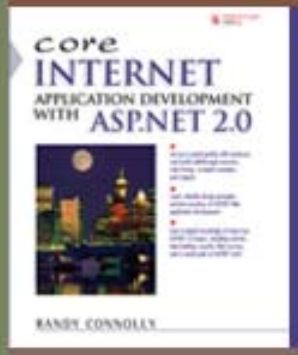

```
Dictionary<string, Customer> dict =
   new Dictionary<string, Customer>();

// This does NOT work
foreach (Customer c in dict)
```




```
Dictionary<string, Customer> dict =
   new Dictionary<string, Customer>();

// This does work
foreach (Customer c in dict.Values)
{
   labMsg.Text += c.Id + "," + c.LastName + "<br/>";
}
```

DataSet

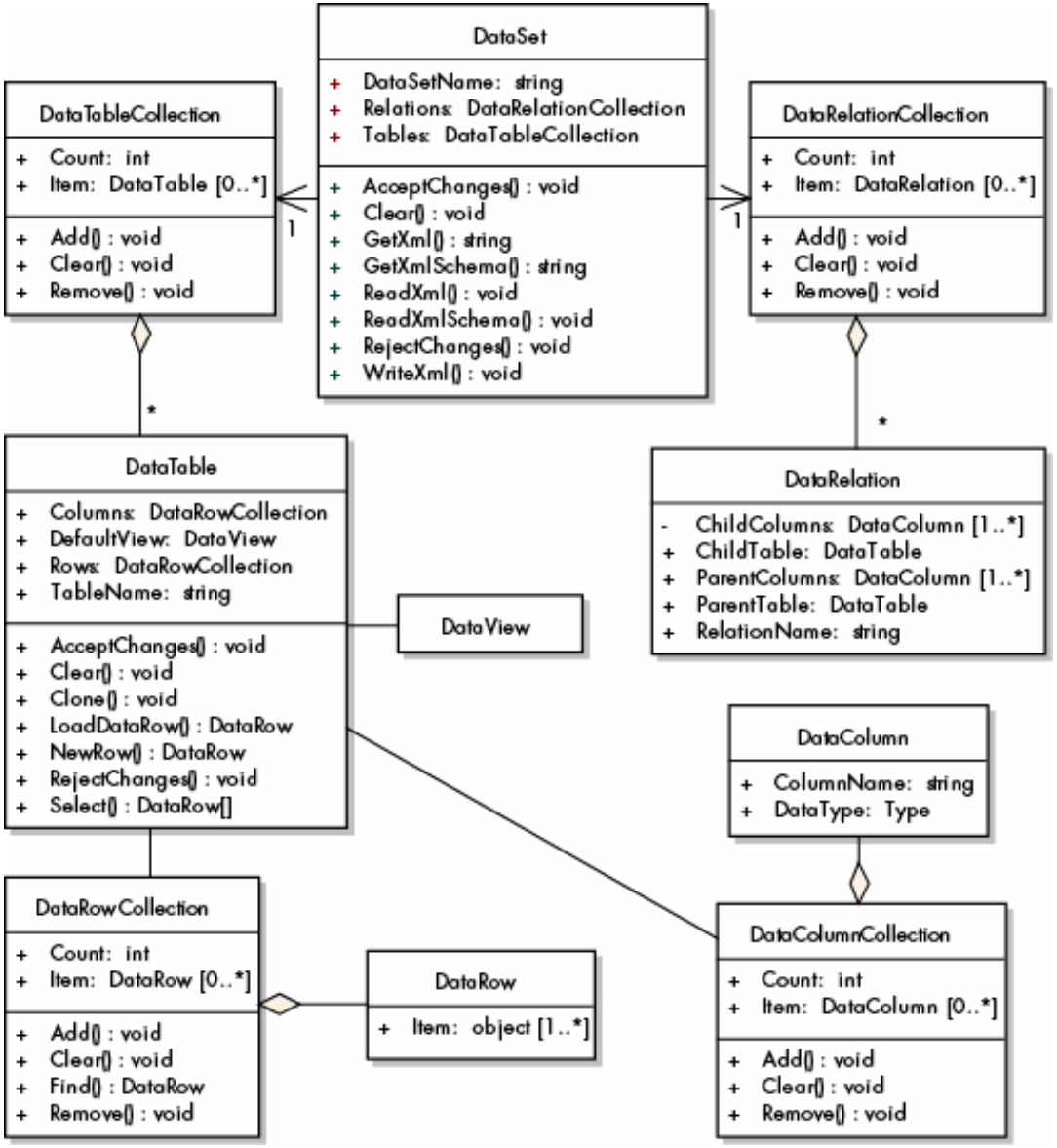
- The `DataSet` is a very rich and complete in-memory data container that mirrors the organization and some of the functionality of a DBMS.
- That is, a `DataSet` is
  - an in-memory data holder that can store not only data, but also its relational structure,
  - can perform a variety of useful operations, such as
    - sorting and filtering data,
    - populating itself from XML,
    - exporting its data and schema to XML.



CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

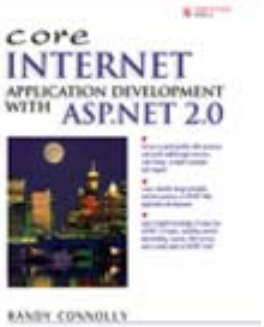
DataSet



CORE INTERNET APPLICATION DEVELOPMENT WITH ASP.NET 2.0

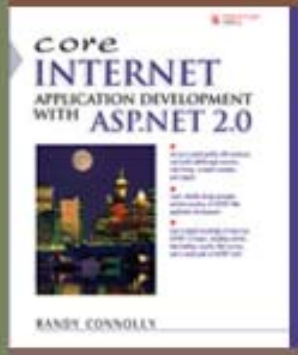
Prentice Hall, 2007

www.randyconnolly.com/core



Populating a DataSet

- There are three ways of populating a `DataSet` with data.
  - Fill it from a database using the appropriate data provider's `DataAdapter` class.
    - Most common approach.
    - Covered in next chapter.
  - Programmatically add `DataTable` objects to the `DataSet`.
  - Use the `ReadXml` method of the `DataSet` to populate data from an XML file or stream.

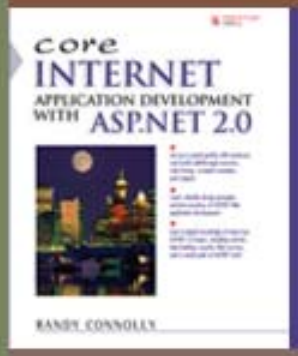


CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

DataTable

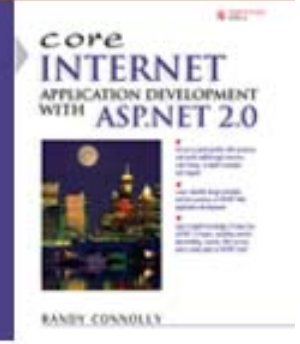
- The heart of the `DataSet` is its collection of `DataTable` objects.
- The `DataTable` class represents a single table of in-memory data, and can be used outside of the `DataSet`.
- Because,
  - the `DataSet` is such a complex class,
  - with Web applications you often need to only retrieve the data from a single table or query,
- you often may want to work just with the `DataTable` as a general purpose data container and ignore its containing `DataSet`.



CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

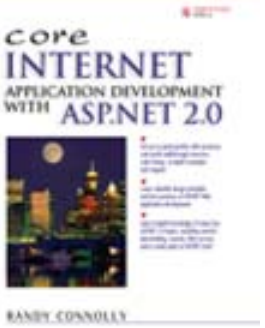
Defining a DataTable




```
DataTable table = new DataTable();
DataColumn firstNameCol = new DataColumn("FirstName", typeof(string));
table.Columns.Add(firstNameCol);
```

```
DataColumn idCol = new DataColumn();

idCol.ColumnName = "Id";
idCol.DataType = typeof(Int32);
idCol.AllowDBNull = false;
idCol.Unique = true;

table.Columns.Add(idCol);
```

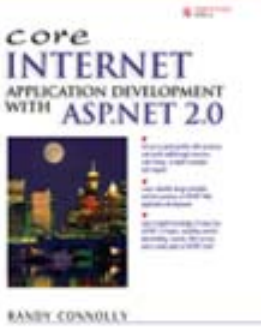
Filling the DataTable




```
DataRow r1 = table.NewRow();

r1[0] = 1;
r1[1] = "Thomas";
r1[2] = "Hobbes";
r1[3] = "123-4567";

table.Rows.Add(r1);
```

```
DataRow r2 = table.NewRow();

r2["Id"] = 2;
r2["FirstName"] = "David";
r2["LastName"] = "Hume";
r2["Phone"] = "564-7823";

table.Rows.Add(r2);
```

Using a DataSet



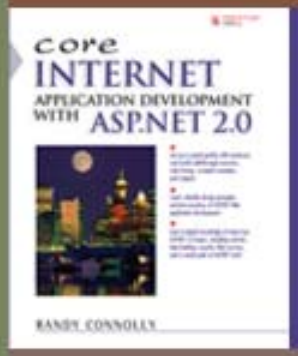

```
DataSet ds = new DataSet();
DataTable dt1 = new DataTable();
DataTable dt2 = new DataTable();
…
ds.Tables.Add( dt1 );
ds.Tables.Add( dt2 );
```

```
DataTable t1 = ds.Tables[0];
DataTable t2 = ds.Tables[1];
```

```
ds.Tables[0].TableName = "Cust";
…
DataTable dt = ds.Tables["Cust"];
```

```
DataRow dr = ds.Tables[0].Rows[0];

string s1 = (string)dr[1];
string s2 = (string)dr["Phone"];
string s3 = (string)ds.Tables[0].Rows[0]["Phone"];
```

Typed DataSets



CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

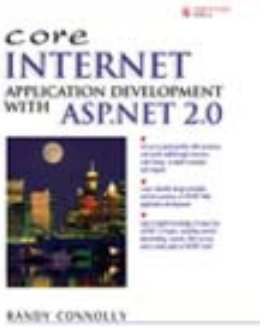
- There is a way to achieve cleaner, type-safe code using the `DataSet` by using what is known as a typed `DataSet`.
- A typed `DataSet` is not a class within the .NET Framework.
  - Rather, it refers to a series of classes, the main of which inherits from `DataSet`, that is generated by a special tool (usually you do this within Visual Studio).
  - These generated classes expose the data within a `DataSet` in a type-safe manner and allow for a more domain-oriented naming convention.

```
SampleTypedDataSet myDataSet = new SampleTypedDataSet();

…

double price = myDataSet.Products[0].Price;
```

XML Integration

CORE INTERNET APPLICATION DEVELOPMENT WITH ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

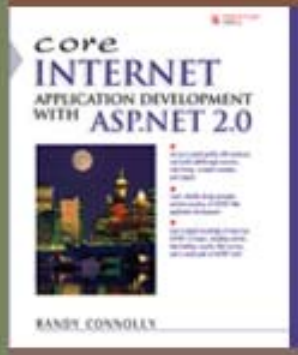



```
try
{
   DataSet ds = new DataSet();

   ds.ReadXml( Server.MapPath("~/App_Data/somefile.xml") );

   // Use the data set
}
catch (IOException)
{
   // Handle the error
}
```

```
ds.WriteXml("output.xml");
```

Choosing a Data Container

- Because almost every ASP.NET application needs to work with data in some form or another, it is important to have some sense of the relative merits of the different ways in which you can store data programmatically.

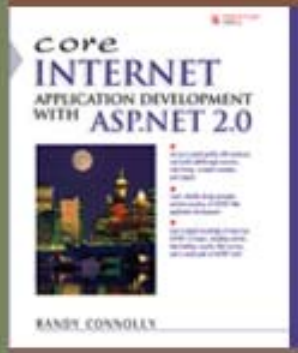


CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

Internal Data

- For most applications, data is external to the application.
  - That is, data is stored in some format, such as a database, an XML file, or a Web service, that is external to the application.
- This external data also needs to be read in and manipulated and possibly stored internally within the application.

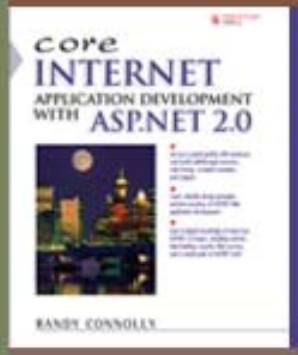


CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

Storing Internal Data

- In this chapter, we have worked with the four main possibilities:
  - .NET collections,
  - custom collections,
  - the DataSet (or just the DataTable),
  - the typed DataSet.

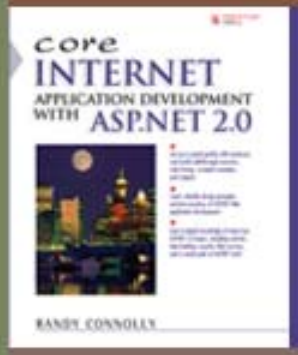


CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

.NET Collections as Containers

- When used in conjunction with custom classes that represent entities in the business or application domain, the .NET collections, especially the generic versions, can be a particularly easy yet powerful way to organize the data used in your application.
  - They allow you to model the problem space of the application using the language of the problem domain, rather than the language of .NET.

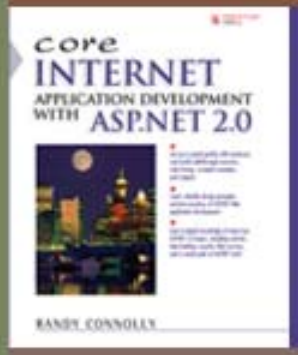


CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

.NET Collections as Containers

- The main drawback for these .NET collections is that these collections are general purpose.
  - That is, the functionality they provide is limited to the general behaviors of a collection, such as adding elements, retrieving elements, and removing elements.
  - You may have to write the code yourself for other behaviors, such as filling these collections from an external data source, searching for elements, sorting elements, or persisting the values back to the data source.

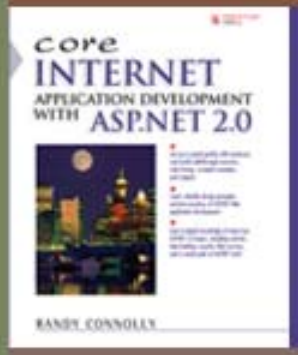


CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

Custom Collections as Containers

- Custom collections along with custom entities allow you to create code that closely models the problem domain.
  - This tends to make the code more meaningful, and as some have argued creates a more maintainable and adaptable application.
  - This author prefers this approach, and in Chapter 11 we will explore it further.
- The disadvantage to this approach is the necessity for implementing the extra functionality yourself.
- As well, not every programmer is comfortable with the domain-focused, object-oriented development paradigm.
  - there are a variety of third-party code-generation or object-relational mapping tools that can help the developer that wants to use this domain-oriented approach.

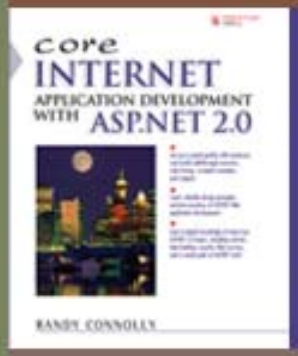


CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

DataSets as Containers

- Has a great deal of built-in functionality
  - However, the code necessary to manipulate individual items inside it is neither the most attractive, the most efficient, nor the most maintainable code possible.
  - The great power of the DataSet is often overkill for many scenarios, since most ASP.NET pages do not need a disconnected data container that caches multiple changes to the data.
  - If all we need is temporary storage for some data for the life of a request, all the extra memory used by the DataSet and its ancillary classes will have an effect on performance for no corresponding gain.
    - In fact, when retrieving data from a database, the DataSet can be several magnitudes of speed slower than other alternatives.

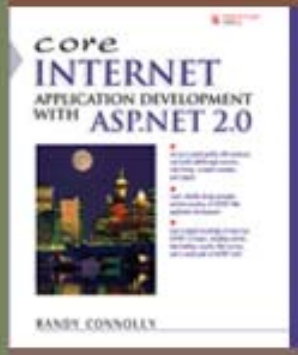


CORE INTERNET APPLICATION DEVELOPMENT WITH ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core

Typed DataSets as Containers

- A typed DataSet has all the advantages of the untyped DataSet.
- As well, the typed DataSet provides a much simpler and easier object model than does the regular untyped DataSet.
- However,
  - they impose a database-centric approach to your application.
  - it can not contain any behaviors, such as business rules.
  - the schema needs to be refreshed whenever the underlying data structure changes.



CORE INTERNET
APPLICATION DEVELOPMENT
WITH
ASP.NET 2.0

Prentice Hall, 2007
www.randyconnolly.com/core