Creating Collections of Entity Objects

What is an Entity Class

An Entity class has properties and typically no methods. An entity class is generally used to hold a single row of data from a table. So, if you have a Category table with the fields *CategoryId*, *CategoryName* and *Description*, you will create a Category class with properties of the same name. For example:

```
C#
public class Category
{
  public int CategoryId { get; set; }
  public string CategoryName { get; set; }
  public string Description { get; set; }
}


Visual Basic
Public Class Category
  Public Property CategoryId() As Integer
  Public Property CategoryName() As String
  Public Property Description() As String
End Class
```

You would then create a collection class to hold 1 or more entity classes. So if your Category table had 100 rows and you read all 100 using a DataSet/DataTable, then you would end up with 100 Category classes in a collection. Below is a definition of a collection class using the Generic List<> class.

```
C#
public class Categories : List<Category>
{
}


Visual Basic
Public Class Categories
  Inherits List(Of Category)
End Class
```

There are many reasons for building an entity class and a collection class. Using an Entity class allows you to serialize this object and send across the web to any other application that needs a structure of the data. When you use an Entity class you get IntelliSense on the properties as opposed to a DataTable where you have to remember the name of the column in the collection. An Entity class also strongly types the data to what it was in the database. When the data is placed into a DataTable, each column goes in as the data type of *object* and comes back out as *object*. That means each time you access it you must convert it into the correct data type. There are many more reasons as well, but these are the main ones.

Reading Data into a Collection

Let’s use the Categories table in the Northwind database to create a collection of entity classes. You will use a DataTable to read all of the rows in the table, then move each row of data into a new **Category** class.

```
C#
private List<Category> GetAllCategories()
{
  List<Category> ret = new List<Category>();
  Category cat;
  SqlDataAdapter da;
  DataTable dt = new DataTable();

  da = new SqlDataAdapter("SELECT CategoryId, CategoryName,
                           Description FROM Categories",
            AppConfig.ConnectionString);

  da.Fill(dt);

  foreach (DataRow dr in dt.Rows)
  {
    cat = new Category();

    cat.CategoryId = Convert.ToInt32(dr["CategoryId"]);
    cat.CategoryName = Convert.ToString(dr["CategoryName"]);
    cat.Description = Convert.ToString(dr["Description"]);

    ret.Add(cat);
  }

  return ret;
}


Visual Basic
Private Function GetAllCategories() As List(Of Category)
  Dim ret As New List(Of Category)()
  Dim cat As Category
  Dim da As SqlDataAdapter
  Dim dt As New DataTable()

  da = New SqlDataAdapter("SELECT CategoryId, CategoryName,
                          Description FROM Categories",
           AppConfig.ConnectionString)

  da.Fill(dt)

  For Each dr As DataRow In dt.Rows
    cat = New Category()

    cat.CategoryId = Convert.ToInt32(dr("CategoryId"))
    cat.CategoryName = Convert.ToString(dr("CategoryName"))
    cat.Description = Convert.ToString(dr("Description"))

    ret.Add(cat)
  Next

  Return ret
End Function
```

Summary

In this chapter you learned how to create an entity class and a collection of entity classes. You should try to use entity classes in your programming instead of data tables as you get more flexibility with classes than you do with loosely-typed objects such as data tables.