Module03

Developing ASP.NET MVC 4 Models

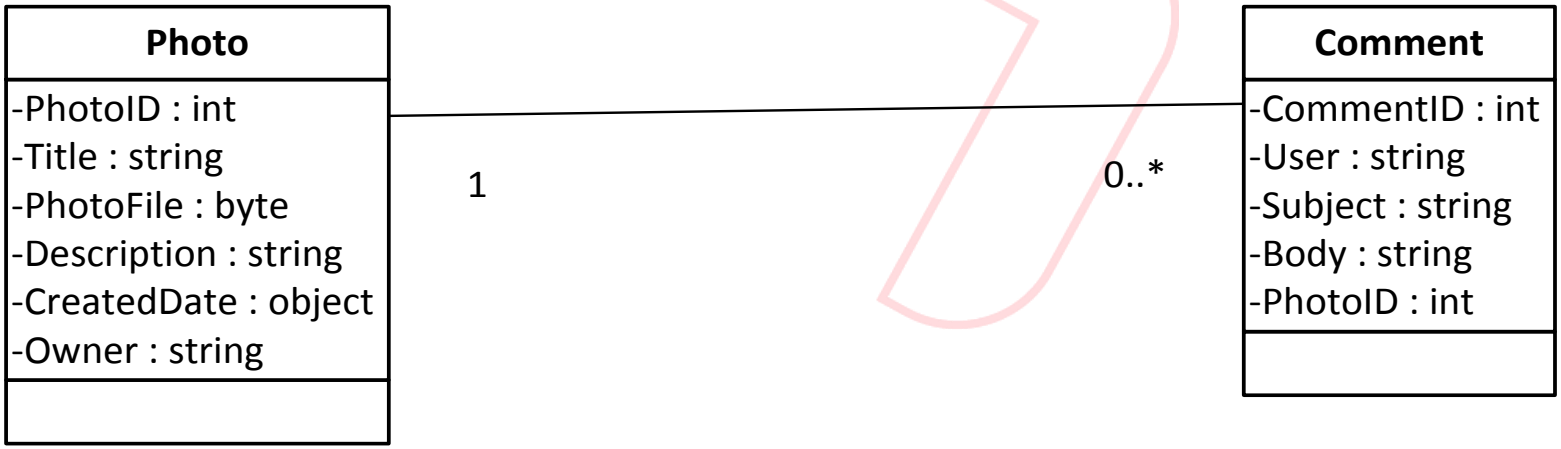
Module Overview

- Creating MVC Models
- Working with Data

Lesson 1: Creating MVC Models

- Developing Models
- Using Display and Edit Data Annotations on Properties
- Validating User Input with Data Annotations
- What Are Model Binders?
- Model Extensibility
- Demonstration: How to Add a Model

Developing Models




```
public class Photo
{
   public int PhotoID { get; set; }
   public string Title { get; set; }
   public byte[] PhotoFile { get; set; }
   public string Description { get; set; }
   public DateTime CreatedDate { get; set; }
   public string Owner { get; set; }
   public virtual ICollection<Comment>
      Comments { get; set; }
}
```

Developing Models (Continued)

```
public class Comment {
        public int CommentID { get; set; }
        public int PhotoID { get; set; }
        public string UserName { get; set; }
        public string Subject { get; set; }
        public string Body { get; set; }
        public virtual Photo Photo { get; set; }
}
```

```
Comment newComment = new Comment();
newComment.UserName = User.Identity.Name;
newComment.Subject = "This is an example comment";
return View("Display", newComment);
```

Using Display and Edit Data Annotations on Properties

```
public class Photo
{
    public int PhotoID { get; set; }
    public string Title { get; set; }
    [DisplayName("Picture")]
    public byte[] PhotoFile { get; set; }
    [DataType(DataType.MultilineText)]
    public string Description { get; set; }
    [DataType(DataType.DateTime)]
    [DisplayName("Created Date")]
    [DisplayFormat(DataFormatString = "{0:MM/dd/yy}",
      ApplyFormatInEditMode = true)]
    public DateTime CreatedDate { get; set; }
    public string UserName { get; set; }
    public virtual ICollection<Comment>
      Comments { get; set; }
}
```

Validating User Input with Data Annotations

```
public class Person
{
   public int PersonID { get; set; }

   [Required(ErrorMessage="Please enter a name.")]
   public string Name { get; set; }

   [Range(0, 400)]
   public int Height { get; set; }

   [Required]
   [RegularExpression(".+\\@.+\\..+")]
   public string EmailAddress { get; set; }
}
```

What Are Model Binders?

- The Default Controller Action Invoker uses model binders to determine how parameters are passed to actions
- The Default Model Binder passes parameters by using the following logic:
  - The binder examines the definition of the action that it must pass parameters to
  - The binder searches for values in the request that can be passed as parameters

Model Extensibility

- Custom validation data annotations can be used to indicate to MVC how to validate the data a user enters in a form or passes in query strings
- There are four built-in validation attributes:
  - Required
  - Range
  - StringLength
  - RegularExpression
- A custom model binder ensures that it identifies parameters in a request and passes all of them to the right parameters on the action

A Custom Validation Data Annotation

```
[AttributeUsage(AttributeTargets.Field)]
public class LargerThanValidationAttribute : ValidationAttribute
{
   public int MinimumValue { get; set; }
   public LargerThanValidationAttribute (int minimum) {
      MinimumValue = minimum;
   }
   public override Boolean IsValid (Object value) {
      var valueToCompare = (int)value;
      if (valueToCompare > MinimumValue) { return true; }
      else { return false; }
   }
}
```

A Custom Model Binder

```
public class CarModelBinder : IModelBinder
{
   public object BindModel (ControllerContext controllerContext,
            ModelBindingContext bindingContext)
   {
      string color =
            controllerContext.HttpContext.Request.Form["color"];
      string brand =
            controllerContext.HttpContext.Request.Form["brand"];
      Car newCar = new Car();
      newCar.color = color;
      newCar.brand = brand;
      return newCar;
   }
}
```

Demonstration: How to Add a Model

In this demonstration, you will see how to:

1. Create a new ASP.NET MVC 4 web application.
2. Add a new model class.
3. Add properties to a model class.
4. Add data annotations to a model class.
5. Add and use a custom validator data annotation.

Lesson 2: Working with Data

- Connecting to a Database
- The Entity Framework
- Using an Entity Framework Context
- Using LINQ to Entities
- Demonstration: How to Use Entity Framework Code
- Data Access in Models and Repositories

Connecting to a Database

- ADO.NET supports a wide range of databases by using different data providers
- Cloud Databases can be used for web applications that are hosted in the cloud
- To connect an MVC web application to a database:
  - Add a reference to the **System.Data** namespace
  - Add a connection string to the **web.config** file

Example Connection Strings

SQL Express:

```
<connectionStrings><add name="PhotoSharingDB"
   connectionString="Data Source=.\SQLEXPRESS;Initial
    Catalog=PhotoSharingDB;Integrated Security=SSPI"
    providerName="System.Data.SqlClient"/>
</connectionStrings>
```

Windows Azure SQL Database:

```
<connectionStrings><add name="PhotoSharingDB" connectionString=
"Server=tcp:example.database.windows.net,1433;
Database=PhotoSharingDB;User ID=Admin@example;
Password=Pa$$w0rd;Trusted_Connection=False;
Encrypt=True;Connection Timeout=30;
PersistSecurityInfo=true"
  providerName="System.Data.SqlClient"/>
</connectionStrings>
```

The Entity Framework

- Types of Entity Framework Workflows
  - Database First
  - Model First
  - Code First

- Adding an Entity Framework Context

```
public class PhotoSharingDB : DbContext
{
    public DbSet<Photo> Photos { get; set; }
    public DbSet<Comment> Comments { get; set; }
}
```

Using an Entity Framework Context

Using the Entity Framework involves:

- Using the Context in Controllers
  - After defining the Entity Framework context and model classes, you can use them in MVC controllers to pass data to views for display
- Using Initializers to Populate Databases:
  - If you are using the code-first or model-first workflow, Entity Framework creates the database the first time you run the application and access data

Using an Entity Framework Context in Controllers

```
public class PhotoController : Controller
{
    private PhotoSharingDB db = new PhotoSharingDB();
    public ActionResult Index()
    {
        return View("Index", db.Photos.ToList());
    }
    public ActionResult Details(int id = 0)
    {
        Photo photo = db.Photos.Find(id);
        if (photo == null) return HttpNotFound();
        return View("Details", photo);
    }
}
```

Using LINQ to Entities

- LINQ to Entities is the version of LINQ that works with Entity Framework
- Sample LINQ Query:

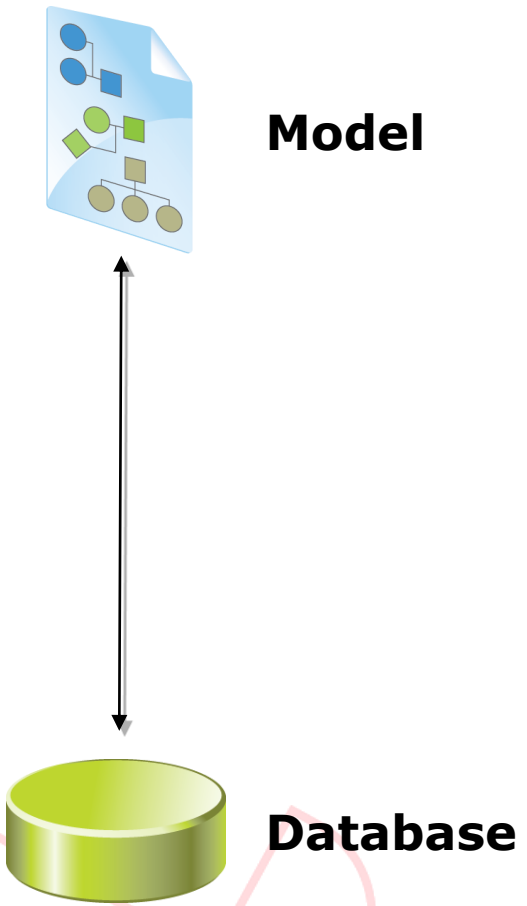
```
photos = (from p in context.Photos
          orderby p.CreatedDate descending
          select p).Take(number).ToList();
```

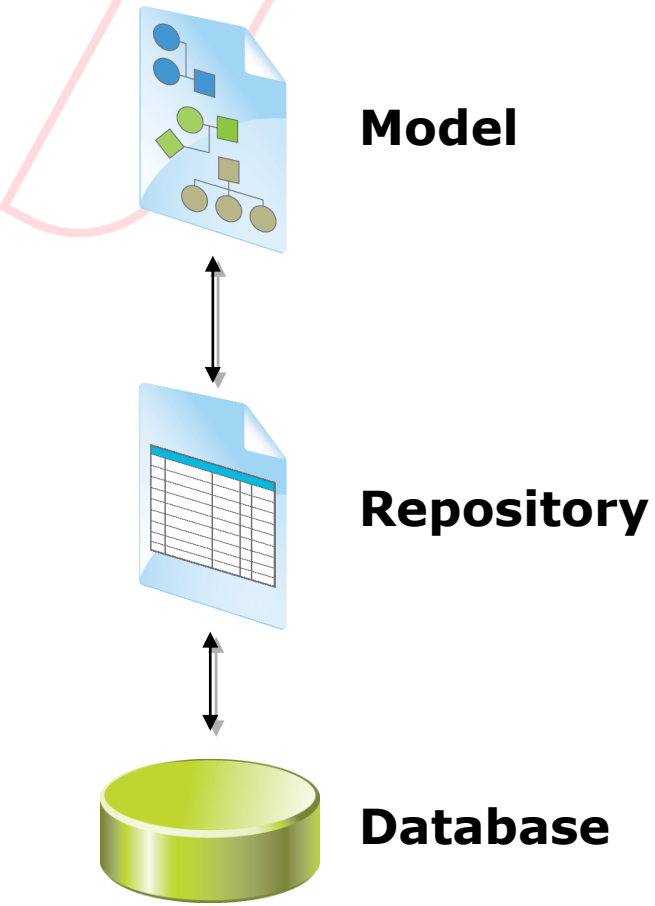
Demonstration: How to Use Entity Framework Code

In this demonstration, you will see how to:

1. Add a connection string in the Web.config file.
2. Install Entity Framework in your project.
3. Add an Entity Framework context to the model.
4. Add an Entity Framework Initializer.
5. Override the **Seed** method.
6. Build the web application.

Data Access in Models and Repositories





Lab: Developing ASP.NET MVC 4 Models

- Exercise 1: Creating an MVC Project and Adding a Model
- Exercise 2: Adding Properties to MVC Models
- Exercise 3: Using Data Annotations in MVC Models
- Exercise 4: Creating a New Windows Azure SQL Database
- Exercise 5: Testing the Model and Database

Estimated Time: 30 minutes

Lab Scenario

You are planning to create and code an MVC model that implements your plan for photos and comments in the Adventure Works photo sharing application. The model must store data in a Windows Azure SQL database and include properties that describe photos, comments, and their content. The model must enable the application to store uploaded photos, edit their properties, and delete them in response to user requests.

Lab Review

- You are building a site that collects information from customers for their accounts. You want to ensure that customers enter a valid email address in the Email property. How would you do this?
- You have been asked to create an intranet site that publishes a customer database, created by the sales department, to all employees within your company. How would you create the model with Entity Framework?

Module Review and Takeaways

- Review Question(s)
- Real-world Issues and Scenarios
- Tools
- Best Practice
- Common Issues and Troubleshooting Tips