In this video we will dsicuss using UpdateModel() function. [Please watch Part 14](http://csharp-video-tutorials.blogspot.com/2013/05/part-14-mapping-aspnet-request-data-to.html), before proceeding.

Change the implementation of **"Create"** action method that is decorated with **[HTTPPost]** aatribute in **"EmployeeController"** as shown below.  
[HttpPost]  
public ActionResult Create(Employee employee)  
{  
    if (ModelState.IsValid)  
    {  
        EmployeeBusinessLayer employeeBusinessLayer =  
            new EmployeeBusinessLayer();  
  
        employeeBusinessLayer.AddEmmployee(employee);  
        return RedirectToAction("Index");  
    }  
    return View();  
}

**Please note:**  
**1.** Model state is being checked using **IsValid**boolean property of **ModelState** object. We will discuss **ModelState** in a later video session.  
**2.** Instead of passing the individual properties of **"Employee"** object as parameters to the **"Create"** action method, we are now passing the **"Employee"** object itself.  
**3.** The **"Employee"** object is then handed over to **AddEmployee**() method of **"EmployeeBusinessLayer"** class, which takes the responsibility of saving the **"Employee"** object to the database table.  
**4.** Upon saving the employee, the user is then redirected to the **"Index"** action method.  
**5.** If there are any "Model" validation errors, **ModelState.IsValid** returns false. In this case, we stay on the same create view, which gives the opportunity to correct the errors and resubmit the page.  
  
**The above method can be rewritten as shown below.**  
[HttpPost]  
public ActionResult Create()  
{  
    if (ModelState.IsValid)  
    {  
        EmployeeBusinessLayer employeeBusinessLayer =  
            new EmployeeBusinessLayer();  
  
        Employee employee = new Employee();  
        UpdateModel<Employee>(employee);  
  
        employeeBusinessLayer.AddEmmployee(employee);  
        return RedirectToAction("Index");  
    }  
    return View();  
}  
  
**When you make this change**, you get a compilation error stating - Type 'MVCDemo.Controllers.EmployeeController' already defines a member called 'Create' with the same parameter types.Our intention here is to overload the **"Create"** controller action method based on **"HttpGet"** and **"HttpPost"**. To fix this error, use **"ActionName"** attribute as shown below.  
[HttpGet]  
[ActionName("Create")]  
public ActionResult Create\_Get()  
{  
    return View();  
}  
  
[HttpPost]  
[ActionName("Create")]  
public ActionResult Create\_Post()  
{  
    if (ModelState.IsValid)  
    {  
        EmployeeBusinessLayer employeeBusinessLayer =  
            new EmployeeBusinessLayer();  
  
        Employee employee = new Employee();  
        UpdateModel<Employee>(employee);  
  
        employeeBusinessLayer.AddEmmployee(employee);  
        return RedirectToAction("Index");  
    }  
    return View();  
}  
  
**Please Note:**  
**1.** We have changed the names of **"Create"** action methods to **"Create\_Get"** and **"Create\_Post"** depending on the actions they respond to.  
**2.** **"ActionName"** is specified as **"Create"** for both of these methods. So, if a **"GET"**request is made to the **"URL - http://localhost/MVCDemo/Employee/Create"** then **"Create\_Get()"** controller action method is invoked. On the other hand if a **"POST"**request is made to the same URL, then **"Create\_Post()"** controller action method is invoked.

**3.** Instead of passing **"Employee"** object as a parameter to **"Create\_Post()"** action method, we are creating an instance of an "Employee" object with in the function, and updating it using **"UpdateModel()"** function. **"UpdateModel()"** function inspects all the **HttpRequest** inputs such as posted Form data, QueryString, Cookies and Server variables and populate the employee object.  
  
When you run the application, you may get an intermittent error stating - Adding the specified count to the semaphore would cause it to exceed its maximum count. To fix this error, either   
**1.** Restart IIS   
OR  
**2.** Disable connection pooling in the connection string of your web.config file