**HiddenInput attribute** is useful when you want to render a property using **input type=hidden**. This attribute is extremely useful, when you don't want the user to see or edit the property, but you need to post the property value to the server when the form is submitted, so the correct record can be updated. **HiddenInput** is present in **System.Web.Mvc** namespace.

**ReadOnly**attribute is present in **System.ComponentModel** namespace. As the name suggests, this attribute is used to make a property readonly. Please note that, we will still be able to change the property value on the view, but once we post the form the model binder will respect the readonly attribute and will not move the value to the property.  You can also, make property of a class **readonly** simply, by removing the SET accessor.

**Changes to Employee.cs file used in the demo**. Notice that Id property is decorated with HiddenInput attribute, and EmailAddress is decorated with ReadOnly attribute.  
public class EmployeeMetadata  
{  
    // Id property is hidden and cannot be changed  
    [HiddenInput(DisplayValue=false)]  
    public int Id { get; set; }  
  
    // EmailAddress is read only  
    [ReadOnly(true)]  
    [DataType(DataType.EmailAddress)]  
    public string EmailAddress { get; set; }  
          
    [ScaffoldColumn(true)]  
    [DataType(DataType.Currency)]  
    public int? Salary { get; set; }  
  
    [DataType(DataType.Url)]  
    [UIHint("OpenInNewWindow")]  
    public string PersonalWebSite { get; set; }  
          
    [DisplayAttribute(Name= "Full Name")]  
    public string FullName { get; set; }  
  
    [DisplayFormat(DataFormatString="{0:d}")]  
    public DateTime? HireDate { get; set; }  
  
    [DisplayFormat(NullDisplayText="Gender not specified")]  
    public string Gender { get; set; }  
}  
  
**Changes to HomeController.cs file**  
public ActionResult Edit(int id)  
{  
    SampleDBContext db = new SampleDBContext();  
    Employee employee = db.Employees.Single(x => x.Id == id);  
              
    return View(employee);  
}  
  
[HttpPost]  
public ActionResult Edit(Employee employee)  
{  
    if (ModelState.IsValid)  
    {  
        SampleDBContext db = new SampleDBContext();  
        Employee employeeFromDB = db.Employees.Single(x => x.Id == employee.Id);  
  
        // Populate all the properties except EmailAddrees  
        employeeFromDB.FullName = employee.FullName;  
        employeeFromDB.Gender = employee.Gender;  
        employeeFromDB.Age = employee.Age;  
        employeeFromDB.HireDate = employee.HireDate;  
        employeeFromDB.Salary = employee.Salary;  
        employeeFromDB.PersonalWebSite = employee.PersonalWebSite;  
  
        db.ObjectStateManager.ChangeObjectState(employeeFromDB, System.Data.EntityState.Modified);  
        db.SaveChanges();  
        return RedirectToAction("Details", new { id = employee.Id });  
    }  
    return View(employee);  
}  
  
**Edit.cshtml view**  
@model MVCDemo.Models.Employee  
@{  
    ViewBag.Title = "Edit";  
}  
  
<div style="font-family:Arial">  
  
@using (Html.BeginForm())  
{  
    @Html.EditorForModel()  
    <br />  
    <br />  
    <input type="submit" value="Save" />  
}  
</div>