**Steps to Dependency Injection – Example 3  
  
Alter the Application**

using System.Web.Mvc;

using Lab6.Models;

namespace Lab6.Controllers

{

public class HomeController : Controller

{

StoreItem item = new StoreItem { Name = "Sofa", Category = "Furniture",   
 Price = 935.95M, Discountable = true };

* Remove the original statement that created   
  the object.
* Add a declaration for the object at the top of the class. Use an interface.
* **Add a constructor** that accepts the object as a parameter.

public ActionResult Index()

{

IPriceCalculator calc = new BlackFridaySalePrice();

if (item.Discountable == true)

{

decimal totalValue = calc.CalcSalesPrice(item.Price);

return View(totalValue);

}

else

{

return View(item.Price);

}

}

}

}  
**Set Up Ninject**

* Install Ninject.MVC5 using NuGet. **Use version 3.2.1!**
* Add a folder called Infrastructure and create in there a class called NinjectDependencyResolver.cs. Copy code from Canvas and paste in code. Add the bindings in AddBindings().   
  kernel.Bind< >().To< >();

System.Web.Mvc.DependencyResolver.SetResolver(new   
 *YourProjectNameHere*.Infrastructure.NinjectDependencyResolver(kernel));

* In the RegisterServices method in NinjectWebCommon.cs, tell the application where to find the resolver.