

Copyright © Todos los Derechos Reservados - Cibertec Perú SAC

Tipo : Guía de Laboratorio

Capítulo : Aplicando técnicas en una aplicación ASP.NET MVC

Duración : 45 minutos

I. OBJETIVO

Implementa inyección de dependencia con SimpleInjector.

II. REQUISITOS

Los siguientes elementos de software son necesarios para la realización del laboratorio:

- Windows 10 (como mínimo Windows 8)
- Visual Studio 2017 (como mínimo Visual Studio 2015)

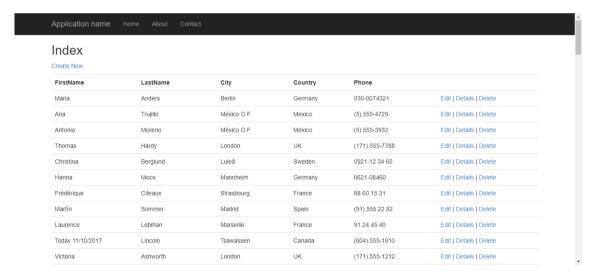
III. EJECUCIÓN DEL LABORATORIO

- Ejercicio Nº 5.1: Implementa inyección de dependencia con SimpleInjector.
- 1. Abrir la solución del módulo anterior.
- 2. Instalar el siguiente paquete nuget en el proyecto Cibertec.Mvc:
 - Install-Package SimpleInjector.Integration.Web.Mvc
- En la carpeta App_Start crea una nueva clase llamada "DIConfig" con el siguiente código:

```
using System.Web.Mvc;
using SimpleInjector;
using SimpleInjector.Integration.Web;
using SimpleInjector.Integration.Web.Mvc;
using System.Reflection;
using Cibertec.UnitOfWork;
using Cibertec.Repositories.Dapper.Northwind;
using System.Configuration;
namespace Cibertec.Mvc
{
    public class DIConfig
        public static void ConfigureInjector()
            var container = new Container();
            container.Options.DefaultScopedLifestyle = new
WebRequestLifestyle();
            container.Register<IUnitOfWork>(()=> new
NorthwindUnitOfWork(ConfigurationManager.ConnectionStrings["NorthwindConne
ction"].ToString()));
container.RegisterMvcControllers(Assembly.GetExecutingAssembly());
            container.Verify();
```

```
DependencyResolver.SetResolver(new
  SimpleInjectorDependencyResolver(container));
       }
   }
4. Refactorizamos el constructor del controlador Customer de la siguiente manera:
  using Cibertec.UnitOfWork;
  using System.Web.Mvc;
  namespace Cibertec.Mvc.Controllers
   {
       public class CustomerController : Controller
           private readonly IUnitOfWork _unit;
           public CustomerController(IUnitOfWork unit)
                _unit = unit;
           public ActionResult Index()
               return View(_unit.Customers.GetList());
           }
       }
   }
```

5. Ejecutamos la aplicación y el resultado es:



IV. EVALUACIÓN

1. ¿Cuáles son las carpetas principales en una aplicación MVC?

Las carpetas principales son: Controllers, Views, Models y wwwroot

2. ¿Qué pasa si al nombre del controlador no se le coloca el prefijo Controller?

Por convención todos los archivos que van a ser controladores deben tener el prefijo Controller al final del nombre para que puedan ser reconocidos por Asp.NET como tales.

3. ¿Cuál es el tipo de datos de los métodos del controller?

El tipo base de todos los controladores es l'Action Result.

4. ¿Qué es una vista Layout?

Es la vista maestra, en ella se establece la estructura base (html) de todas las vistas que tendrá la aplicación web. Algunas vistas pueden no utilizar la vista base.

5. Los modelos, ¿pueden ser librería de clases?

Sí, los modelos son un concepto que involucra entidades, acceso a datos, reglas de negocio y pueden ser implementadas en librería de clases (dll).

6. ¿Qué son las vistas parciales?

Al igual que las vistas normales, estas permiten ser reutilizadas en ciertas partes de la aplicación.