28/02/22

**EJERCICIO**:

- Sobre el proyecto de referencia "Ssid.Arquitectura", identifica métodos y/o clases que sirvan de ejemplo de los diferentes principios SOLID, así como de los principios DRY (Don't Repeat Yourself) y KISS (Keep It Simple). Dichos ejemplos pueden ser tanto de casos donde se esté rompiendo el principio en cuestión (e.g. se está haciendo copy/paste en lugar de crear un método, de forma que no se respeta el principio DRY), como de casos en que sí se esté aplicando correctamente.

A continuación pasamos a definir ejemplos de cada tipo:

1. SOLID

* Singles class, open/closed:

SSID.Ejemplo.Entities.Utils.Results: Clase plantilla bien construida

public static Result<TValue> Ok<TValue>()

{

              return new Result<TValue>();

}

public static Result<TValue> Ok<TValue>(TValue value)

{

              var result = new Result<TValue>(value);

              return result;

}

public static Result<TValue> Fail<TValue>(string errorMessage)

{

              var result = new Result<TValue>();

              result.AddErrorMessage(errorMessage);

              return result;

}

SSID.Ejemplo.Entities.Results:

    public class Result<T> : Result

    {

        protected T Value;

        //Crea el objeto con exito por defecto

        public Result(T valor)

        {

            Value = valor;

        }

        public Result()

        {

        }

        public void AddValue(T value)

        {

            Value = value;

        }

        public T GetValue()

        {

            return Value;

        }

        public override bool Equals(object obj)

        {

            if (obj == null || GetType() != obj.GetType())

            {

                return false;

            }

            var other = (Result<T>)obj;

            return (ReferenceEquals(other.GetValue(), null) || (GetValue().Equals(other.GetValue())))

                && Success == other.Success

                && ErrorMessages.SequenceEqual(other.ErrorMessages)

                && WarningMessages.SequenceEqual(other.WarningMessages)

                && InfoMessages.SequenceEqual(other.InfoMessages);

        }

        public override int GetHashCode()

        {

            return GetValue().GetHashCode();

        }

        public static implicit operator bool(Result<T> value)

        {

            // assuming, that 1 is true;

            // somehow this method should deal with value == null case

            return value.Success;

        }

    }

* Inyección de dependencias:

public class ProveedorRepository : IProveedorRepository

{

public const string GET\_PROVEEDORES = "Get\_Proveedores";

private readonly IDataBaseManager \_dataBaseManager;

private readonly IProveedorMapper \_proveedorMapper;

public ProveedorRepository(IDataBaseManager dataBaseManager, IProveedorMapper proveedorMapper)

{

\_dataBaseManager = dataBaseManager;

\_proveedorMapper = proveedorMapper;

1. DRY / KISS

* **UserManager.cs:** repetición código

context.Response.Cookies.Add(new HttpCookie(FormsAuthentication.FormsCookieName)

{

Path = FormsAuthentication.FormsCookiePath,

Value = "",

Domain = (!String.IsNullOrEmpty(FormsAuthentication.CookieDomain))

? Convert.ToString(FormsAuthentication.CookieDomain)

: context.Request.Url.GetBaseDomain(),

HttpOnly = true,

Expires = DateTime.Now.AddYears(-1)

});

* **ProductoRepository.cs:** repetición código

using (var dataReader = \_dataBaseManager.ExecuteReader(command))

{

while (dataReader.Read())

{

var producto = \_productoMapper.MapearProducto(dataReader);

listaProductos.Add(producto);

}

}

* **SubidaBajadaFicherosService.cs:** comentario innecesario

catch (Exception ex1)

{

//Escribimos log de error

\_logProvider.AddError(Layer.Bll, "Msg\_Error\_Eliminar\_Documento", ex1);

}

* **ObjectDumper.cs:** complejidad cognitive (demasiado if dentro de otro…), difícil de seguir…

private string DumpElement(object element, bool isTopOfTree = false)

{

if (\_currentIndent > \_depth)

{

return null;

}

if (element == null || element is string)

{

Write(FormatValue(element));

}

else

{

var valueTypeElement = element as ValueType;

if (valueTypeElement != null)

{

WriteValueTypeElement(valueTypeElement);

}

else

{

var enumerableElement = element as IEnumerable;

if (enumerableElement != null)

{

WriteEnumerableElement(enumerableElement);

}

else

{

//Hack: Al recorrer recursivamente las propiedades del DataReader por reflexion acaba su lectura

//Por tanto, cualquier clase que implemente la interfaz no se lee

var dataReaderElement = element as IDataReader;

if (dataReaderElement != null)

{

WriteDataReaderElement(dataReaderElement);

return \_stringBuilder.ToString();

}

WritePropertyElement(element);

}

}

}

return isTopOfTree ? \_stringBuilder.ToString() : null;

}

* **UnityInterceptionExtensions.cs:** demasiados parámetros

public static IUnityContainer RegisterTypesWithCheck(this IUnityContainer container, IEnumerable<Type> typeList,

Func<Type, IEnumerable<Type>> getFromTypes, Func<Type, string> getName = null,

Func<Type, ITypeLifetimeManager> getLifeTimeManager = null,

Func<Type, IEnumerable<InjectionMember>> getInjectionMembers = null, bool overwriteExistingMappings = false)

* **UnityInterceptionExtensions.cs:** sobra el “== **true**”

**if** (resultadoRecursivo == **true**)

* **UnityConfig.cs:**

APP\_Start: Ssid.Ejemplo.Internet.UnityConfig: Repetición de valor texto appSettings en varios puntos de la aplicación

private static void ConfigurarParametrosEnvioCorreo(BoostrapInitParameters parameters)

        {

            parameters.EnvioCorreosBinding = new CustomBinding("EnvioCorreosCustom");

            //Parámetros necesarios para el envío de correos

            parameters.WsEnvioCorreosAddress = TryGetAppSetting("WSEnvioCorreos\_Address");

            var envioCorreoCredentials = new ClientCredentials();

            envioCorreoCredentials.UserName.UserName = TryGetAppSetting("WSEnvioCorreos\_UserName");

            envioCorreoCredentials.UserName.Password = TryGetAppSetting("WSEnvioCorreos\_UserPassword");

            parameters.EnvioCorreosCredentials = envioCorreoCredentials;

            var envioCorreosConfiguration = new EnvioCorreosConfiguration()

            {

                Servidor = TryGetAppSetting("WSEnvioCorreos\_Servidor"),

                CuentaOrigenCorreo = TryGetAppSetting("WSEnvioCorreos\_CuentaOrigenCorreo"),

                CuentaOrigenUsuario = TryGetAppSetting("WSEnvioCorreos\_CuentaOrigenUsuario"),

                CuentaOrigenPassword = TryGetAppSetting("WSEnvioCorreos\_CuentaOrigenPassword"),

                CodigoAplicacion = ConfiguracionUtils.TryGetAppSetting<int>("WSEnvioCorreos\_CodigoAplicacion")

            };

            parameters.EnvioCorreosConfiguration = envioCorreosConfiguration;

            parameters.EstaEnvioCorreosPruebasActivado = ConfiguracionUtils.TryGetAppSetting<bool>("EnvioCorreos\_EnvioPruebas");

            var listaDestinatarios = new string[0];

            var destinatariosPruebas = ConfiguracionUtils.TryGetAppSetting<string>("EnvioCorreos\_DestinatariosPruebas");

            if (destinatariosPruebas != null)

            {

                listaDestinatarios = destinatariosPruebas.Split(';');

            }

            parameters.EnvioCorreosDestinatariosPruebas = listaDestinatarios;

        }

* **Líneas duplicadas:**

