# Sistema de espacios de nombres (Namespaces)

Los Namespaces son utilizados para crear un sistema destinado organizar la forma de exponer la funcionalidad de un aplicativo. Dicha funcionalidad se expresa en clases .NET.

**Sistemas jerárquico**

Para sistemas grandes donde intervienen numerosos desarrolladores y donde los Dominios del sistema están bien separados se podría aplicar un patrón de nombrado jerárquico es decir definir un sistema de nombres comenzando del mayor nivel de abstracción hasta llegar a niveles más específicos.

A continuación se visualiza la regla general para nombrar *Namespaces*:

**<Compañía>.<[Sistema]>.<Extremo>.<Módulo>.<Componente>**

A continuación citamos algunos ejemplos:

**Compañía**: Planet

**Sistema**: Crm

|  |
| --- |
|  |
|  |
| Planet.Front.Win32.UserControls |
| Planet.Front.Web.UserControls |
|  |
| Planet.Crm.Front.GestionCustomersInternos |
| Planet.Crm.Common |
| Planet.Crm.Common.Customers.ISVC |
| Planet.Crm.Common.Customer.BE |
| Planet.Crm.Back.Customer.BC |
| Planet.Crm.Back.Customer.DAC |
| Planet.Crm.Back.Customer.SVC |

De esta manera resultará más fácil poder ordenar y ubicar los archivos por solución y realizar deploys sin cometer error alguno.

Supongamos e que estamos trabajando con el módulo de Locations pero nuestra solución tiene un pos build que copia el ultimo release en un repositorio común junto con los demás módulos del sistemas. Con esta nomenclatura sería más sencillo poder ordenar y obtener un assembly secuencial de otro.

Planet.Crm.Back.Customers.BC

Planet.Crm.Back.Customers.DAC

Planet.Crm.Back.Customers.SVC

Planet.Crm.Back.Locations.BC

Planet.Crm.Back.Locations.DAC

Planet.Crm.Back.Locations.SVC

De lo contrario tendríamos el siguiente orden, donde es más complicado la selección y/o ubicación rápida de los archivos, más aún si tenemos varios dominios de desarrollo:

Planet.Crm.Back.BC.Customers

Planet.Crm.Back.BC.Locations

Planet.Crm.Back.DAC.Customers

Planet.Crm.Back.DAC.Locations

Planet.Crm.Back.SVC.Customers

Planet.Crm.Back.SVC.Locations

# Proyectos y soluciones Vs Namespaces

Una solución de Visual Studio.NET está conformada por uno o más proyectos, uno o más sitios web y carpetas de solución, cuyos nombres tienen que mantener cierta coherencia.

Podemos diferenciar dos grandes grupos de tipos de soluciones que se desarrollan en una fábrica de software:

* **Componentes utilitarios de uso general:**

**<[Tecnologíao o Compañia]>.Tools.NombreUtilitario**

|  |  |  |
| --- | --- | --- |
| Tipo de componente | Default Namespace | Nombre de proyecto |
| ErrorViewer | Planet.Tools.ErrorViewer | Planet.Tools.ErrorViewer |
| CodeGenerator | Fwk.Tools.CodeGenerator | Fwk.Tools.CodeGenerator |
| ConfigurationMannager | Fwk.Tools.ConfigurationMannager | Fwk.Tools.ConfigurationMannager |
|  |  |  |

* **Componentes específicos de la aplicación:**
* Back-End: **<Sistema>.Back.Production.<Componente>**

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo de componente | Sufijo | Default Namespace | Nombre de proyecto |
| BusinessServices | SVC | Crm.Back.Production | Crm.Back.Production.SVC |
| BusinessComponents | BC | Crm.Back.Production | Crm.Back.Production.BC |
| DataAccessComponents | DAC | Crm.Back.Production | Crm.Back.Production.DAC |
| BusinessEntities | BE | Crm.Common.Production | Crm.Common.Production.BE |
| InterfaseServices | ISVC | Crm.Common.Production | Crm.Common.Production.ISVC |
|  |  |  |  |

* Front-End **<Sistema>.Front.Production.<Componente>**

|  |  |  |
| --- | --- | --- |
| Tipo de componente | Default Namespace | Nombre de proyecto |
| View | **Crm.Front.GestionarStock** | Crm.Front.GestionarStock |
| Library | **Crm.Front.GestionarStock.Library** | Crm.Front.GestionarStock.Library |
|  |  |  |
| Common | Crm.Front.Common | Crm.Front.Common |
| Common User controls | Crm.Front.Common.UserControls | Crm.Front.Common |
|  |  |  |

* Common (Componte de acceso tanto de Frontend como desde Back)

**<Sistema>.Common.<Componente>**

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo de componente | Sufijo | Default Namespace | Nombre de proyecto |
| Common | Common | Crm.Common | Crm.Common |
| Enumeraciones | Enums | Crm.Common.Enums | Crm.Common |
| Constantes | Const | Crm.Common.Const | Crm.Common |
|  |  |  |  |

# 

Patrón de nombrado Compacto

En muchos casos la aplicación o sistema a desarrollar no es muy grande o bien no está siendo construida por demasiados desarrolladores. En tal caso es posible que existan pocas soluciones y proyectos que separen los diferentes contextos de componentes.

En tal caso podríamos diseñar un modelos de namenspase de menor complejidad que el anterior de una forma más compacta.

En el modelo compacto los namespace y los proyectos de .net no tienen una relación 1 a 1. Si no que dentro de un proyecto pueden convivir más de un namespase.

**Ejemplo:**

Supongamos una empresa de salud con los siguientes datos del sistema

Compañía: Family Care

Nombre de sistema: Health

Tipo de sistema : Historia clínica y carga de pacientes

|  |  |  |
| --- | --- | --- |
| Componente | Default Namespace | Nombre de proyecto |
| FrmABMPatient | **Health.Front** | Health.Front |
| Common | Health.Front | Health.Front |
| User controls | Health.Front | Health.Front |

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo de componente | Sufijo | Default Namespace | Nombre de proyecto |
| Common | Common | Crm.Common | Health.Common |
| Enumeraciones | Enums | Crm.Common.Enums | Health.Common |
| Constantes | Const | Crm.Common.Const | Health.Common |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo de componente | Default Namespace | | Nombre de proyecto (dll) |
| BusinessServices SVC | | Health.Back.Svc  O bien  Health.Back | Health.SVC |
| BusinessComponents BC | | *El negocio lo absorbe la SVC* | |
| DataAccessComponents DAC | | Health.Back.Dac  O bien  Health.Back | Health.DAC |
| BusinessEntities BE | | Health.BE  Health.BE.Enums  Health.Back.BE *(para clases Entity Fwk)* | Health.BE |
| InterfaseServices ISVC | | Health.Isvc.[servicename] | Health.SVC |
|  |  | |  |