



Introducción al Framework .NET y C#

Nicolás M. Paez

Algoritmos y Programacion 3

npaez@fi.uba.ar



Agenda

- La plataforma .NET
- .NET Framework
- El lenguaje C#
- Herramientas de desarrollo
- Por donde empezar



La plataforma .NET



¿Qué es .NET?

- La combinación de:
 - Framework.NET
 - .NET Enterprise Servers
 - Web Services
 - Ambiente Integrado de desarrollo



.NET Enterprise Servers

- Windows 2003 Server
- SQL Server
- Sharepoint Portal Server
- Content Server
- Exchange Server
- Internet Information Server
- Biztalk Server
- Commerce Server



Web Services

- Proveen una forma estándar de comunicación entre aplicaciones de múltiples plataformas
- Utilizan internet
- Estan basados en estándares abiertos
 - XML
 - SOAP
 - WSDL
 - HTTP



Ambiente Integrado de Desarrollo

- Desarrollo en cualquier lenguaje del .NET Framework
- Múltiples Editores
 - Lenguajes
 - Datos (XML, esquemas)
 - Pantallas (HTML, WinForms)
 - Recursos (Gráficos, archivos varios)
- Manejo del software de base
 - Bases de datos
 - Sistema operativo (procesos, threads, event log, servicios)
 - Otros servicios como colas de MSMQ



Componentes Fundamentales

ASP.NET, Web Services

ADO.NET y XML

.NET Framework

Sistema Operativo

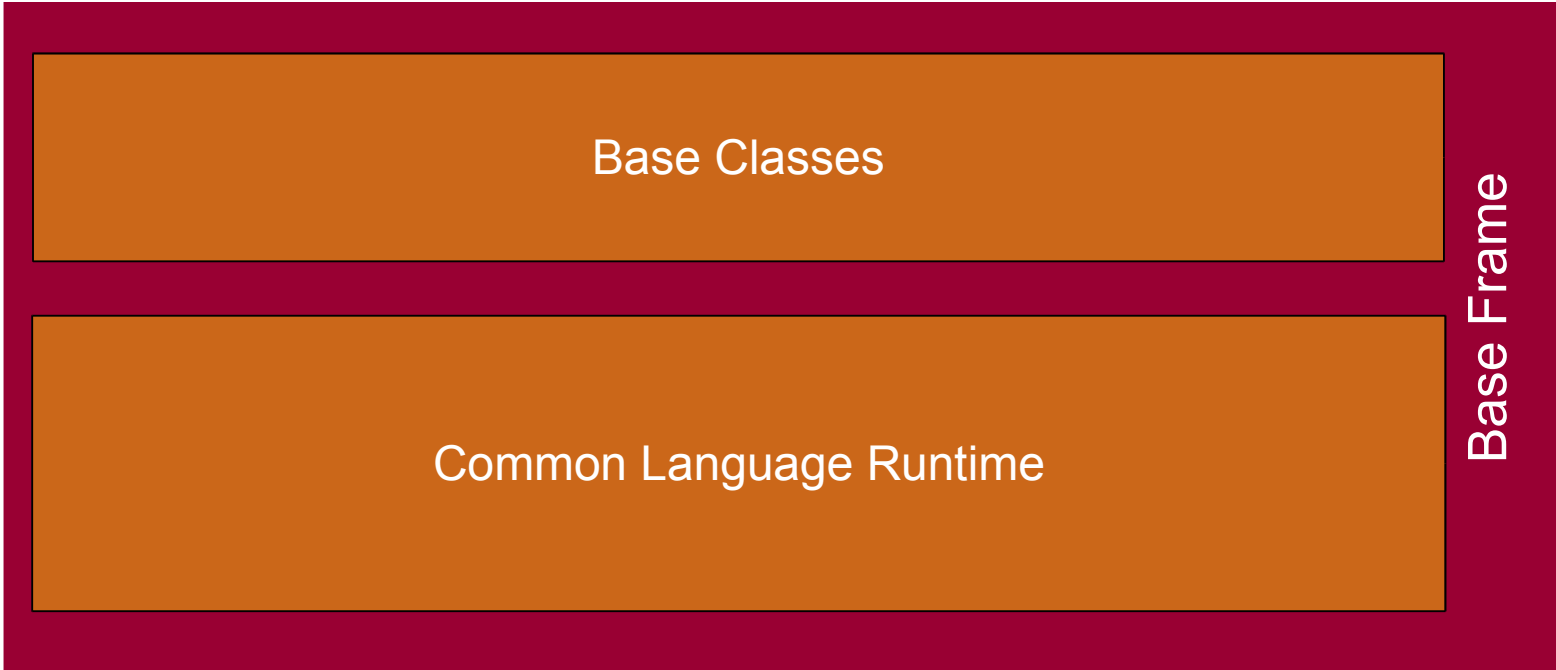
Visual Studio.NET



.NET Framework

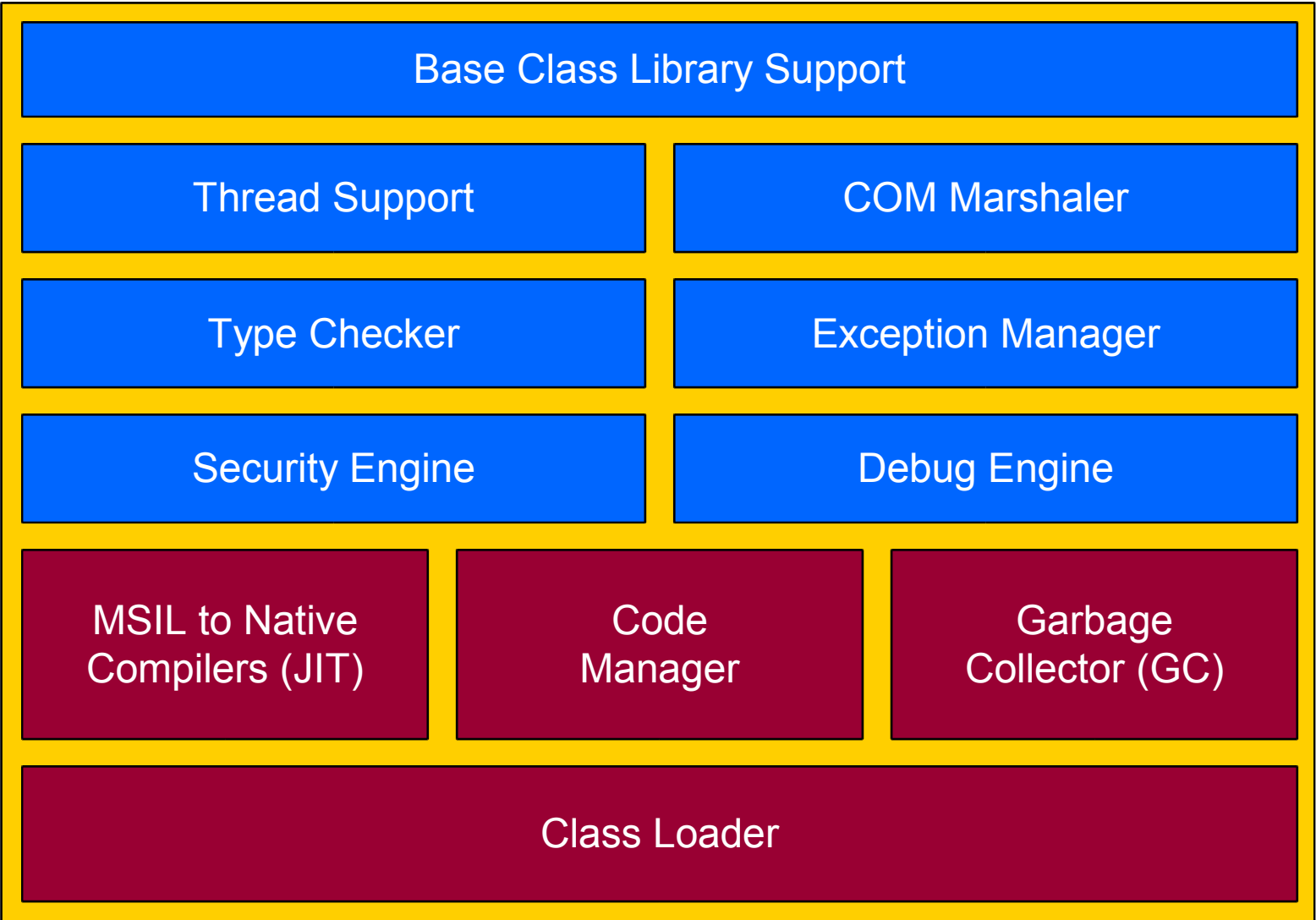


Framework .NET





Common Language Runtime





Common Language Runtime

- Ambiente de ejecución de .NET
 - Las aplicaciones corren dentro del CLR
 - Basado en la especificación CLI
 - Common Language Infrastructure
 - Especificación abierta, ECMA-335
- Brinda servicios básicos a los ejecutables
 - Ejecución (threading, gestión de errores)
 - Gestión de memoria
 - Seguridad
 - Diagnóstico (debugging, tracing)
 - RTTI

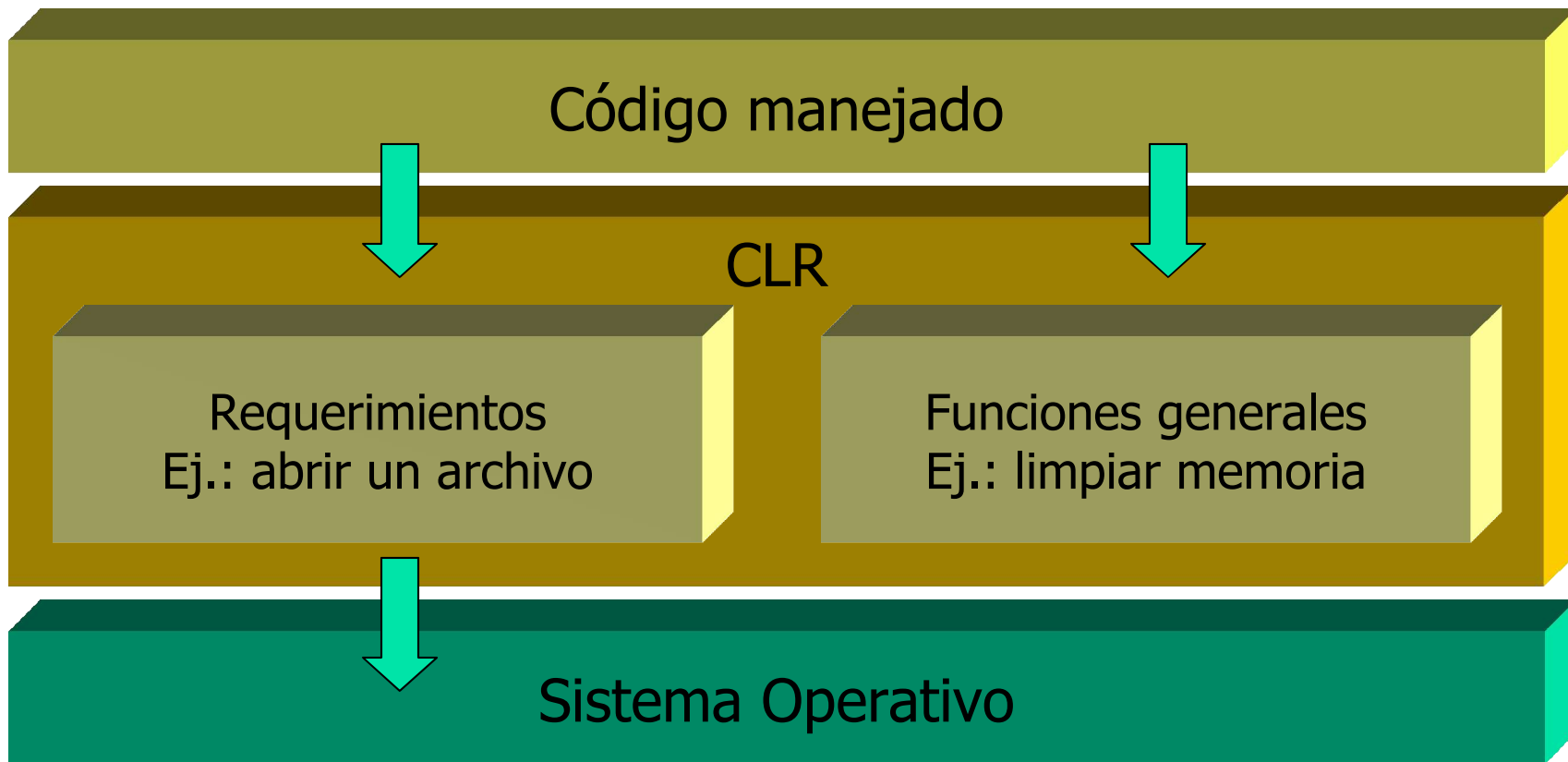


Common Language Runtime

- Puntos importantes
 - Herencia entre lenguajes
 - Sistema común de tipos
 - Compiladores Just-in-time (JIT)
 - Compilan código intermedio (MSIL) en código nativo
 - Garbage collector
 - Seguridad
 - Manejo de Excepciones entre lenguajes
 - Depuración entre lenguajes
 - Multi-Threading entre lenguajes
 - Objetos distribuidos
 - Diagnóstico, tracing y profiling



Código manejado



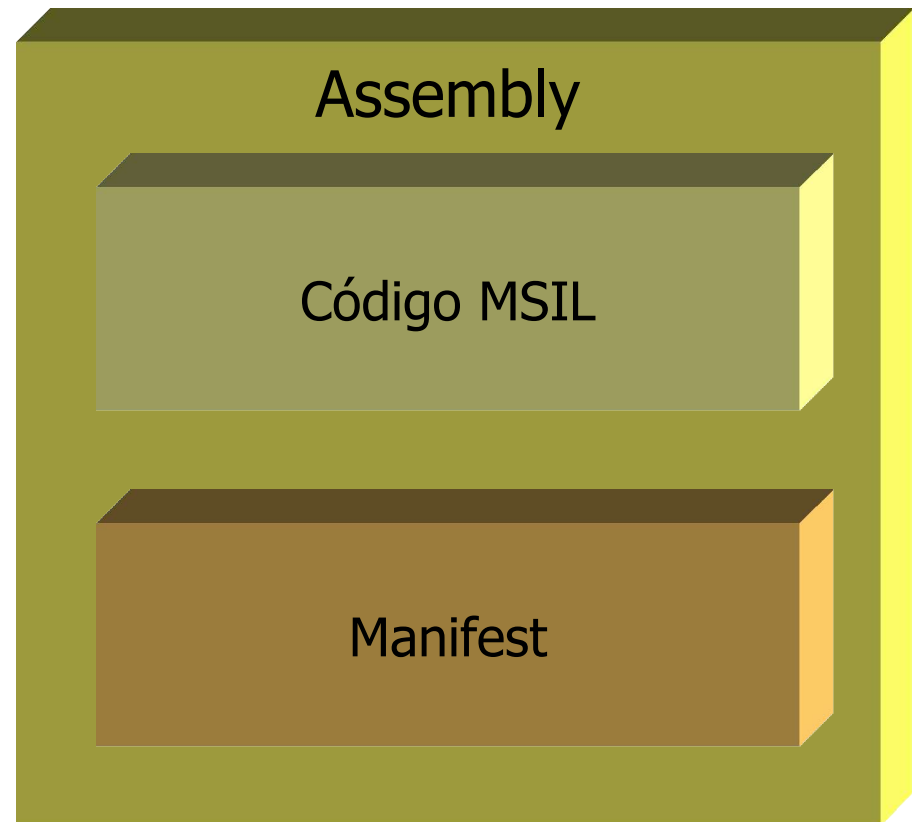


Microsoft Intermediate Language

- El código fuente es compilado a MSIL
 - Código intermedio (no nativo)
 - Modelo de VM
- MSIL tiene características especiales
 - Verificable
 - No depende de una plataforma en particular
 - Incluye construcciones de alto nivel
 - Soporte para objetos
 - Herramientas en el .NET Framework SDK
 - ILADM: IL Assembler
 - ILDASM: IL Disassembler

Ensamblados (Assemblies)

- EXE o DLL
- Simples o múltiples
- Manifest
 - Nombre
 - Versión
 - Cultura
 - Nombre fuerte
 - Archivos
 - Tipos
 - Referencias
- GAC





Sistema Unificado de Tipos

- CTS – Common Type System
 - Establece las reglas para la descripción y manipulación de tipos de datos
- Reglas básicas de los tipos:
 - Value types
 - Reference type
- Establece la base para el soporte de interoperabilidad inter-lenguaje



Interoperabilidad Inter-lenguaje

- En .NET se pueden combinar lenguajes
 - Para consumir y ser llamado por otras clases
 - Para heredar y ser heredado por otras clases
- Soporte multi-lenguaje
 - Basado en el sistema unificado de tipos
 - Soportado por el CLR y los compiladores
 - Las clases deben ser CLS-Compliant
- Interoperabilidad
 - El mismo debugger puede usarse para todos los lenguajes
 - Mecanismo uniforme de gestión de errores



Lenguajes .NET

- Ada
- APL
- Visual Basic.NET
- C#
- Managed C++
- J#
- COBOL
- Component Pascal (Queensland Univ of Tech)
- ECMAScript (JScript)
- Eiffel (Monash University)
- Haskell (Utrecht University)
- Icc (MS Research Redmond)
- Mondrian (Utrecht)
- ML (MS Research Cambridge)
- Mercury (Melbourne U.)
- Oberon (Zurich University)
- Oz (Univ of Saarlandes)
- Perl
- Python
- Scheme (Northwestern U.)
- S#



Interoperabilidad

- COM
 - Los componentes COM se pueden usar como clases .NET
 - Las clases .NET se pueden exportar como componentes COM
 - Todos los servicios de COM+ están disponibles
- P/Invoke
 - Para llamar a funciones nativas
 - Sistema operativo
 - DLLs nativas (no .NET)

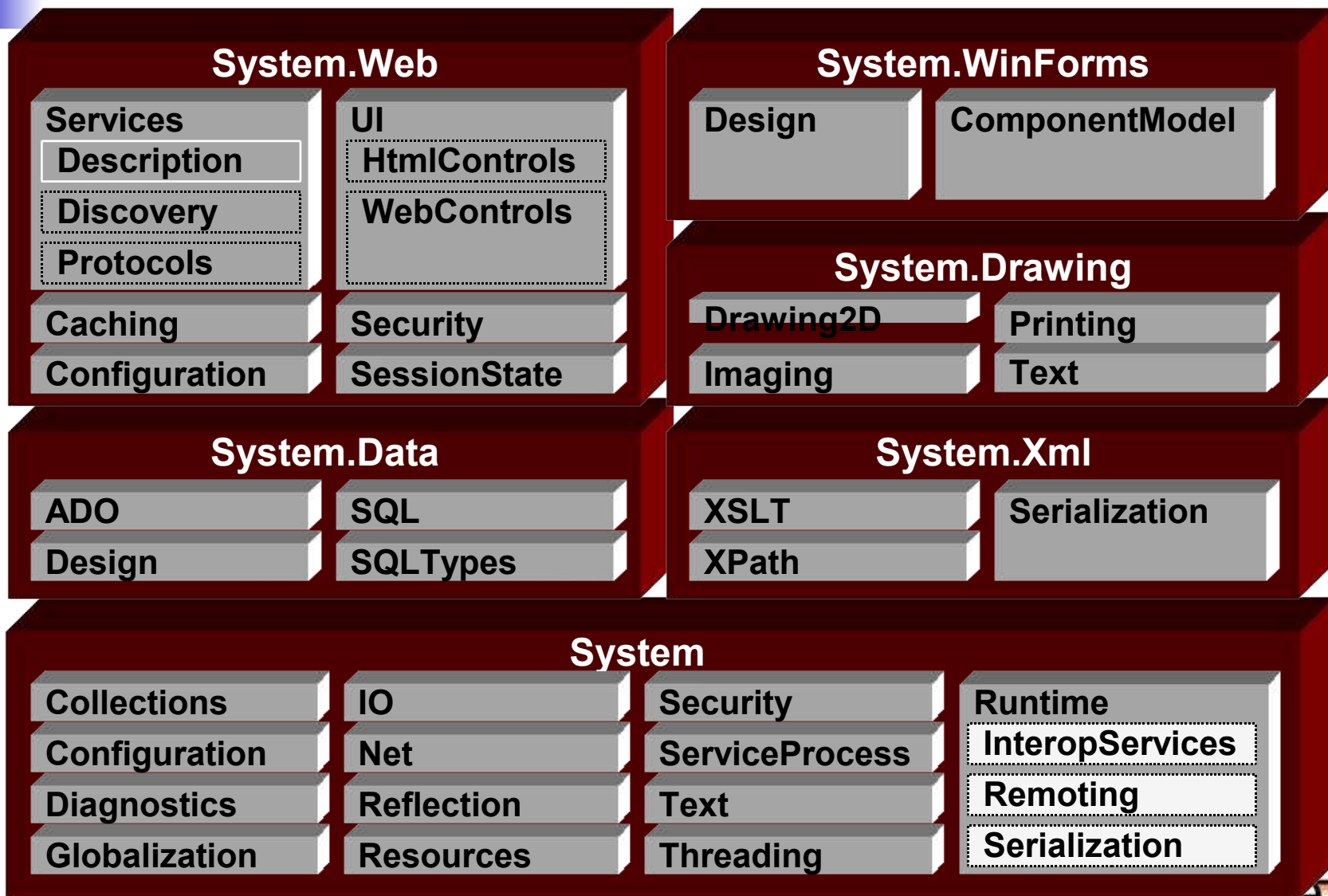


Base Class Library

- WinForms
 - Contrucción de aplicaciones Windows
- WebForms
 - Contrucción de aplicaciones Web
- Clases base
 - Estructuras de datos
 - Bases de datos
 - Manipulación de documentos XML
 - Soporte de I/O (disco, redes, etc.)
 - Sistema de base
 - threading, sincronización, ...
 - Servicios de COM+ y Message Queuing con MSMQ



Librería de clases





Ventajas en .NET

- Entorno unificado
 - Tanto para desarrollo y ejecución
 - Windows, Web, Servicios, WebServices, ...
- Independencia real del lenguaje
- Integración con las últimas tecnologías
 - XML
 - WebServices (SOAP, WSDL, etc.)
- Framework de aplicación
 - Amplia biblioteca de clases base
 - Abstracción de los servicios del sistema



El lenguaje C#



C#: generalidades

- $(C++)_+ = C+++ = C\#$
- Sintaxis tipo C
- Orientado a objetos
- De propósito general
- Soporte de componentes
- Complementamente integrado a .NET
- Estándar
 - Aceptado por la ECMA

Estructura general

- Basado en bloques
 - Al estilo C/C++
 - Bloque=Sentencia o Grupo delimitado por “{” ”}”
 - Las sentencias terminan en “;”
- No hay funciones libres (fuera de clases)
- Sensible a mayúsculas
- El espacio en blanco no se considera

Sentencias

- `if(<bool expr>) {...} else {...}`
- `switch(<var>) { case <const>: ...; }`
 - Soporta Strings
- `while(<bool expr>) {...}`
- `for(<init>; <bool test>; <mod>) {...}`
- `do {...} while(<bool expr>);`
- try-catch-throw-finally

Sentencias 2



- `foreach`
 - Iteración de contenedores y arreglos
- `using`
 - Define el ámbito de validez de un objeto
- `lock(<object>) {...}`
 - Sincronización (sección crítica)
- `checked {...} / unchecked {...}`
 - Protege contra integer overflows
 - Puede trabajar como operador "checked (expresión)"
- `fixed`
 - Para fijar objetos en "áreas inseguras"

Operadores

- Algebraicos: + - * / %
- Lógicos: & | ^ ! ~ && || true false
- Concatenación (de strings): +
- Unarios: ++ --
- Desplazamiento: << >>
- Relacionales: == != < > <= >=
- Asignación: = += -= *= /= %= &= |= ^= <<= >>=
- Acceso a miembros: .
- Índice: []
- Condicional: :?
- Información de tipos: is sizeof typeof
- Type casting: ()
- Creación de objetos: new
- Indirección y direcciones: * -> [] &

Directivas de pre-compilación

- Controlan el proceso de compilación
 - “Pre-procesador” por razones históricas
 - No hay pre-procesador en C#
 - Definición de símbolos
 - #define, #undef
 - Compilación condicional
 - #if, #elif, #else, #endif
 - Números de línea
 - #line
 - Errores
 - #error, #warning
 - Regiones
 - #region, #endregion



Sistema de Tipos

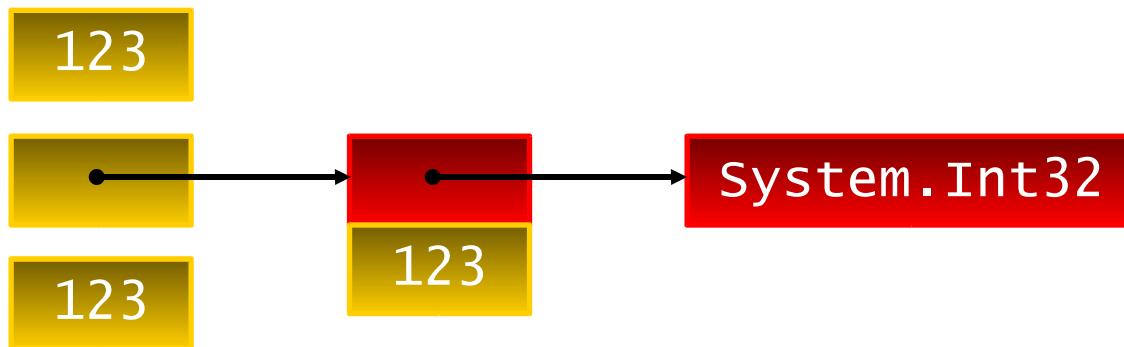
- Basados en el CTS de .NET
- Tipos por valor
 - Int, float, char, structs, enums, punteros, etc
 - Residen en la pila
 - Pasaje como parámetros: ref, out
- Tipos por referencia
 - objetos
 - Residen en el heap
 - Heredan de object
- Boxing y unboxing

Boxing & Unboxing



- Boxing
 - Crea un objeto de "soporte" (box) y copia el valor en él
- Unboxing
 - Verifica el tipo de datos y extrae el valor

```
int i = 123;  
object o = i;  
int j = (int)o;
```





Espacio de nombres

- Todo tipo pertenece a un espacio de nombres
- Pueden anidarse
- Permiten evitar el choque de nombres
- Se declaran con namespace
- Se referencian con using
- Similares a los paquetes de java, pero mapeo a directorios



Tipos Abstractos de datos

- Clases
 - Object es la clase base por omisión
 - Herencia simple
 - Pueden denifirse como: sealed ó abstract
- Interfaces
 - Implementación múltiple
 - Pueden definir propiedades
- Structs
 - Son tipos por valor
 - Pueden tener métodos
 - Pueden implementar interfaces



Modificadores de visibilidad

- Public
- Private
- Protected
- Internal



Metodos 1

- No pueden estar sueltos
- Devuelven un valor
- Reciben cero o más parametros
 - Parámetros variables con param
 - Parámetros de entrada-salida con ref
 - Parámetros de salida con out

Métodos 2

- Pueden ser virtuales
 - Hay que indicarlo explícitamente "virtual"
 - Al redefinirlos hay que usar el "override"
 - "new" permite cortar la cadena virtual
- Pueden ser abstractos "abstract"
- Pueden ser de clase "static"
- Puede evitarse su redefinición "sealed"



Métodos 3: sobrecarga operadores

- Se puede definir el significado de operadores para instancias de clases y estructuras
- Tipos de operadores
 - Binarios (+, -, /, etc.)
 - Unarios (!, ~, ++, --, etc.)
 - Conversión (typecasting) explícita e implícita



Propiedades

- Permiten el acceso a los atributos privados de una instancia
- Pueden ser de lectura, escritura o ambos
- Equivalentes a los getters/setters

```
private int entero;
public int MiEntero
{
    get { return entero; }
    set { entero = value; }
}
```

Delegates

- Punteros a funciones, pero tipados y OO
- Son la base para la implementación de eventos
- y.....mejor vemos un ejemplo...



Eventos

- No son exclusivos de IU
- Basados en publicación-suscripción
- Se disparan llamando a una función
- Vemos un ejemplo....



Anotaciones (atributos)

- Permiten anotar el código en forma declarativa
 - Se aplican a clases, métodos, variables, propiedades, etc.
- Basadas en el soporte de metadata de .NET
- Extensibilidad sin modificar el lenguaje
- Ampliamente utilizados por .NET Framework
- Definibles por el usuario
- Las clases de Reflection se usan para explotarlos



.NET 2.0

- Clases parciales
- Tipos genericos
- Propiedades de lectura/escritura con distinta visibilidad



Herramientas de desarrollo



Entornos de desarrollo



- Microsoft Visual Studio
- Microsoft Web Matrix
- Borland C# Builder
- Borland Delphi .NET
- SharpDevelop
- MonoDevelop
- Snippet Compiler
- C# for Eclipse



Utilitarios

- NDoc
 - A partir de los comentarios de en el código genera documentación en múltiples formatos
- NUnit
 - Framework de testing unitario
- Nant
 - Herramienta para hacer builds
- FXCop
 - Analizador de código



Frameworks

- ORMs
 - Nhibernate, OBJ.NET, Retina.NET
- Frameworks de aplicación
 - MBI, Spring.NET, Castle .NET, EDAF
- PAG Enterprise Libraries



Por donde empezar



Como empezar: Paso 1

- Elegir framework y SDK
 - Microsoft .NET Framework
 - Rotor
 - Mono
 - DotGNU
- Elegir un entorno de desarrollo
 - SnippetCompiler
 - SharpDevelop
 - MonoDevelop
 - Notepad, Vi, Emacs



Como empezar: Paso 2

- Elegir un lenguaje
- Programar, Googling, Entender
- Programa desarrollador 5 estrellas
- Terrarium
- Imagine CUP



Como empezar: Paso 3

- Conocer las distintas tecnologías
 - Windows Forms
 - ADO.NET
 - XML
 - Direct X
 - Compact Framework
 - ASP.NET
 - Web Services

Sitios de referencia

- Comunidad .NET Framework
 - <http://www.gotdotnet.com>
- ASP.NET sitio oficial
 - <http://www.asp.net>
- Proyecto Mono
 - <http://www.mono-project.com>
- DotGNU Project
 - <http://www.dotgnu.org>
- Universidad .NET
 - <http://www.microsoft.com/spanish/msdn/latam/academicalliance/>



¿Preguntas?





Muchas gracias por su participación

Consultas, dudas y sugerencias
npaez@fi.uba.ar