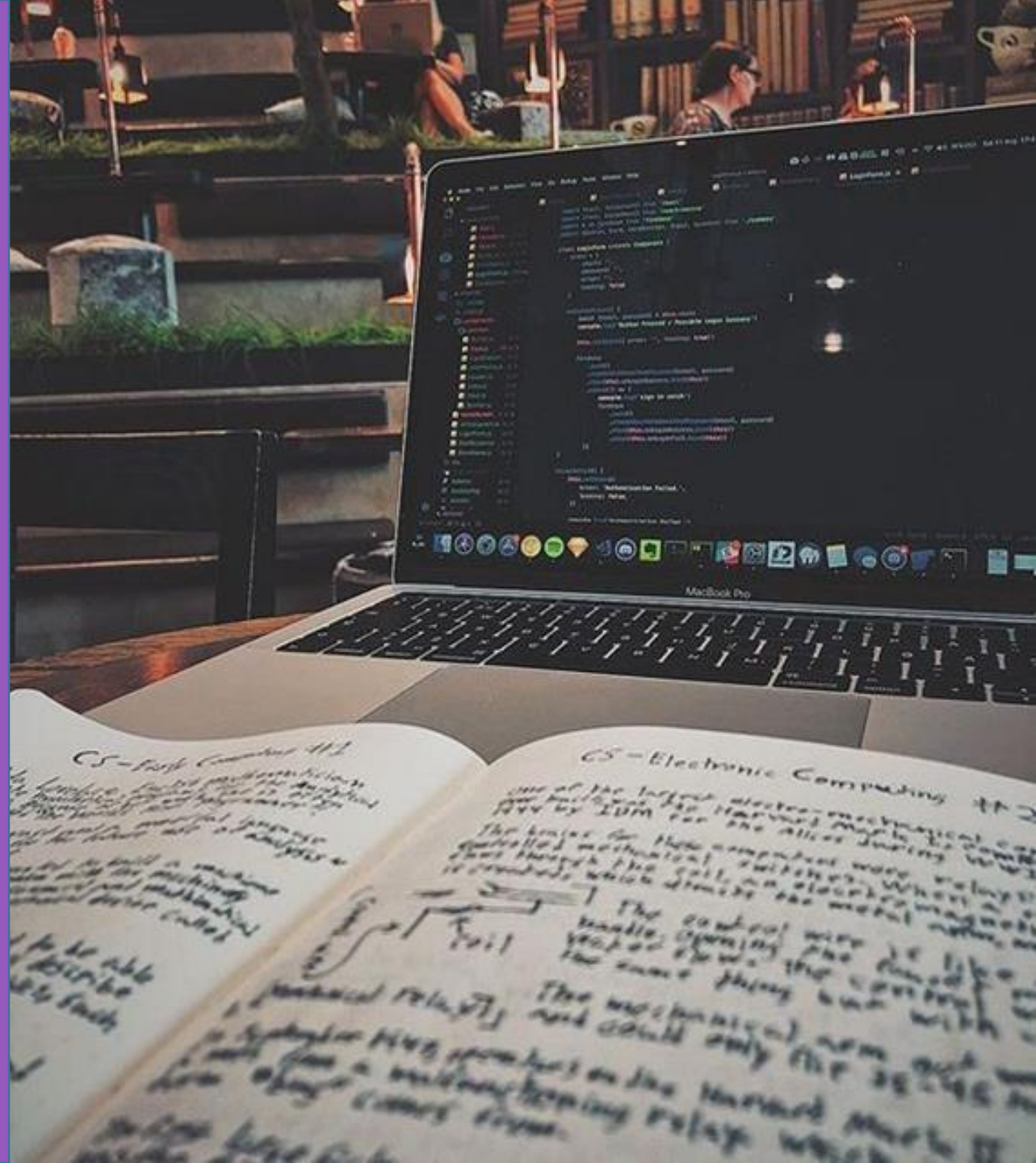
Clase Nro 5

- Introducción a c#
- El Lenguaje c#, sus características y posibilidades
- SDK .Net
- Lenguajes interpretado y lenguajes compilados
- Linea de comandos CLI
- Tipos de datos en C#
- Estructuras de control
- Compilación de proyectos .Net
- Proyectos en C#
- Práctica




I'm your father
C++
C#
NOOOOOOOO!!!

Algunos datos de interés

C# (pronunciado *sí sharp* en inglés) es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA (ECMA-334) e ISO (ISO/IEC 23270)

El lenguaje de programación C# fue creado por el danés Anders Hejlsberg que diseño también los lenguajes Turbo Pascal y Delphi

Es un lenguaje en constante evolución. Hoy contamos con la versión 12.0

Características de C#

- **Sintaxis sencilla**. La sintaxis de C# es muy similar a Java, lo que simplifica al desarrollador a la hora de escribir código.
- **Fuertemente Tipado**
- **lenguaje orientado a objetos**, así que obliga que todos los métodos y propiedades estén dentro de una clase.
- **Multi paradigma**
- **Programación Multihilo**.
- **Sistema de tipos unificado**. Todos los tipos de datos sencillos de C# derivan de una clase común llamada **System.Object**.
- **Múltiples formas de organización en proyectos y tecnologías.** Desde espacios de nombres hasta proyectos y soluciones.
- **Mejora en la gestión de memoria dinámica.** Uso de garbage collector.
- **Tratamiento de errores**. Cualquier lenguaje de programación moderno utiliza las excepciones para controlar los posibles errores en el código.
- **Manejo sencillo de colecciones de datos con LINQ**

¿Qué puedo programar con C#?

- **Proyectos de consola** - (Windows, Linux y MAC) con Net Core
- **Desarrollo de escritorio** – Windows Form, WPF, UWP (win 10 y 8.1), GTK# (Linux)
- **Desarrollo de WEB** – ASP.Net (Windows) o ASP Net Core (Windows, Linux, MAC)
- **Desarrollo multiplataforma de aplicaciones móviles** (Android, IOS, Windows, MAC).
- **Desarrollo de Video Juegos** – Monogame, Godot, Unity3D entre otros.

  (PC, XBOX, Playstation, Nintendo Switch, etc)
- **IOT** - Microframework

Evolución del lenguaje C#

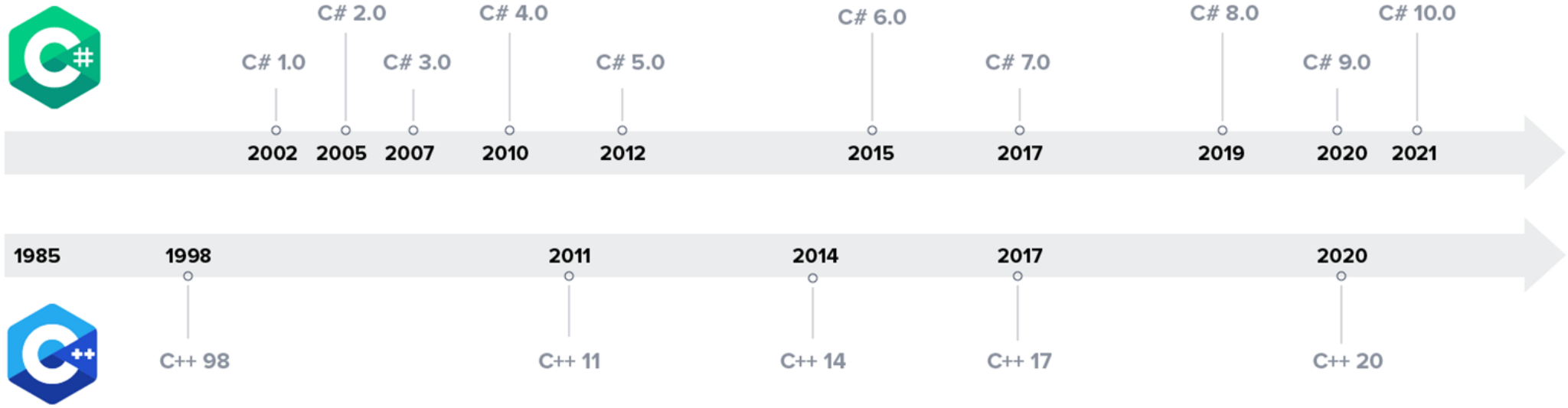


Imagen: https://www.toptal.com/c-sharp/c-sharp-vs-c-plus-plus

Historia: https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-version-history

Tipos de Lenguajes

Lenguaje Interpretado	Lenguaje Compilado
El ciclo de desarrollo (el tiempo entre el momento en que escribes el código se lo pruebas) **es más rápido en un lenguaje interpretado**.	**Un lenguaje compilado está optimizado para el momento de la ejecución**, aunque esto signifique una carga adicional para el programador
Un lenguaje interpretado está optimizado para hacerle la vida más fácil al programador, aunque eso signifique una carga adicional para la máquina.	Un lenguaje compilado está optimizado será traducido a código máquina y por tanto dependerá de **la ARQ**. Para la que fue traducida.

Un lenguaje puede ser un script o un lenguaje de programación, dependiendo del entorno en el que se utilizan.

Un script requiere un intérprete, mientras que un programa requiere un compilador.

Actividad: Buscar la definición de:

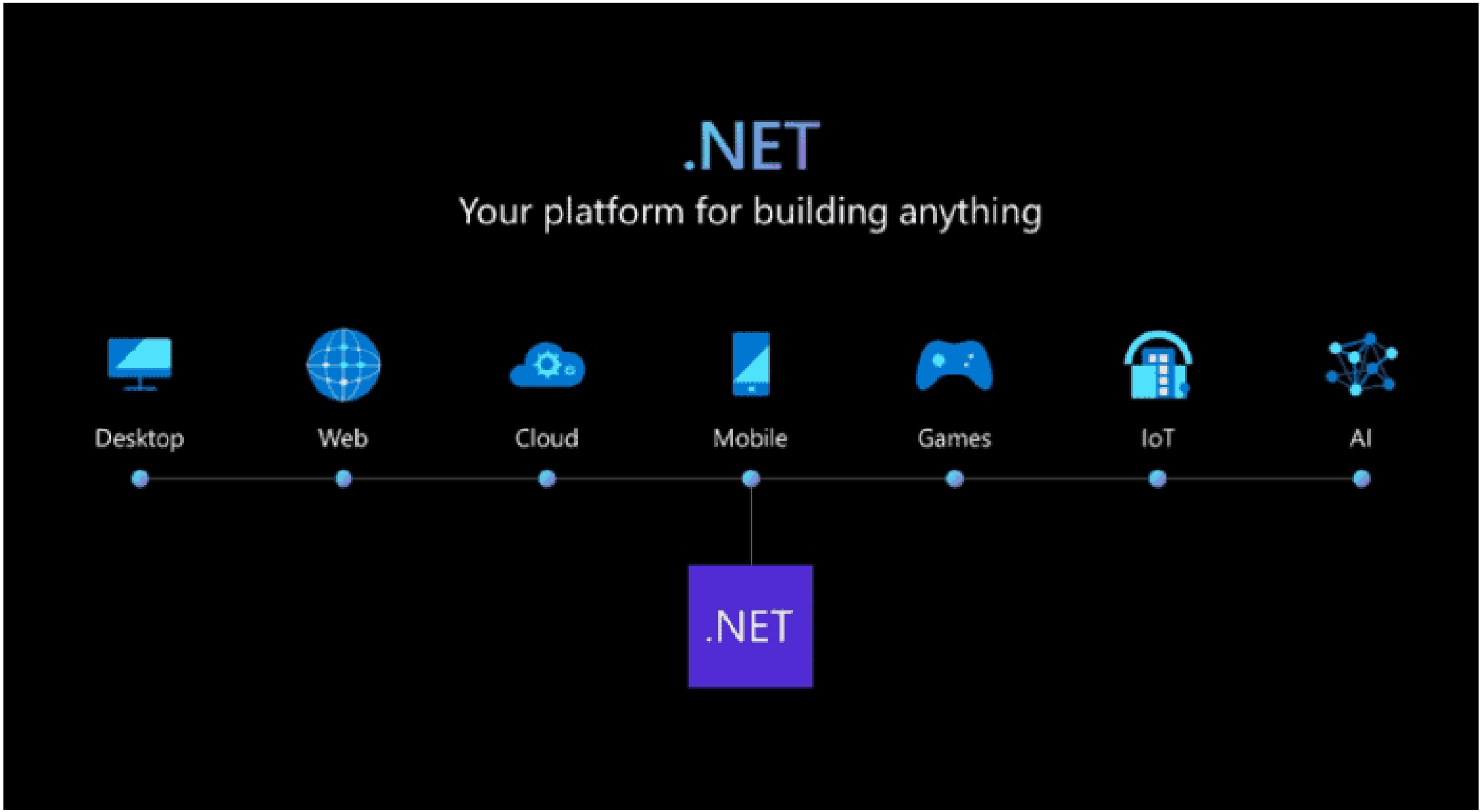
- Librería
- Una API
- Un SDK
- Un Framework

Definiciones

- Una **biblioteca o librería (library)** es un conjunto de elementos (funciones, clases, tipos predefinidos, constantes, variables globales, macros, etc) que es posible utilizar en un programa para facilitar la implementación de ese programa.

- **API** viene del inglés "Application programming interface" que significa "Interfaz para programación de aplicaciones". Una API proporciona una interfaz para que los programas accedan a las funcionalidades proporcionadas por una biblioteca o un sistema, permitiendo que los programas utilicen estas funciones sin necesidad de conocer los detalles de implementación subyacentes.

- Un **SDK** y **framework** son una colección de librerías, APIs, documentación, utilidades y/o código de ejemplo que le ayudan a desarrollar un software. La diferencia está en cómo ofrecen esta solución. Mientras que un sdk ofrece herramientas para usarla como queramos un Frameworks requieren que se inserte el código en ellos y ellos utilizan ese código en el momento adecuado.


.NET
Your platform for building anything
Desktop
Web
Cloud
Mobile
Games
IoT
AI
.NET

Algunos datos de interés

.NET foundation

.IDE

IDEs (Entorno de desarrollo Integrado) de trabajo

MonoDevelop

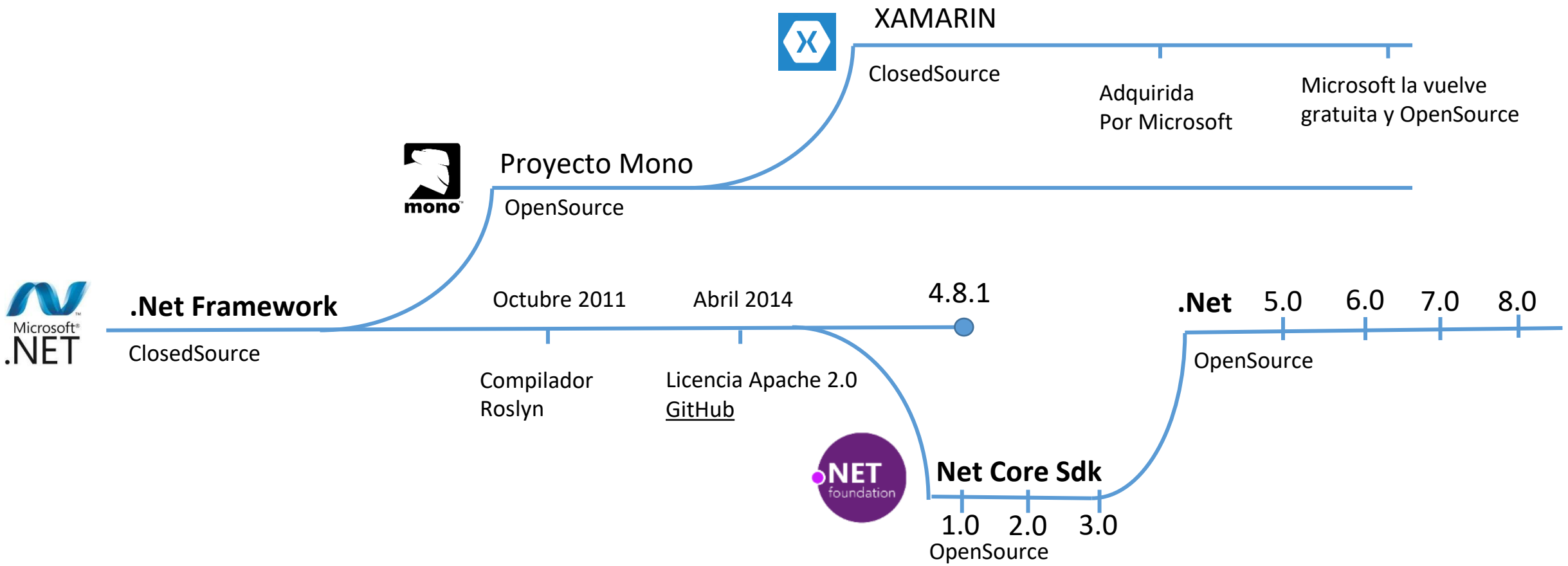
RIDER

VS CODE

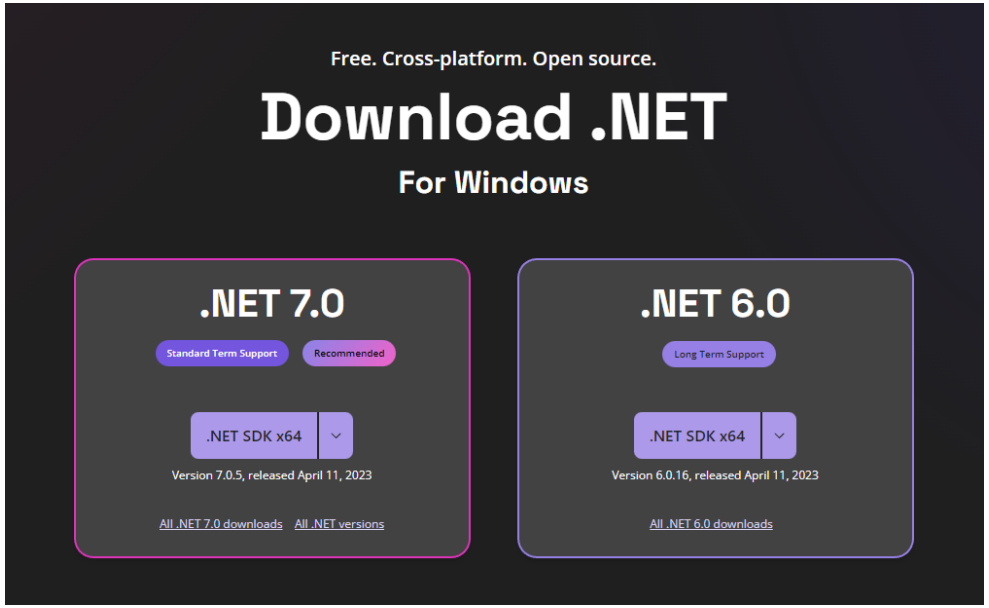
Visual Studio

.Net (sdk)

Bifurcaciones de .Net



Instalación del sdk .Net



EN WINDOWS Y MAC: https://dotnet.microsoft.com/en-us/download

EN LINUX: https://learn.microsoft.com/es-es/dotnet/core/install/linux-ubuntu-2204

.Net Cli (Linea de comandos)

https://learn.microsoft.com/en-us/dotnet/core/tools/

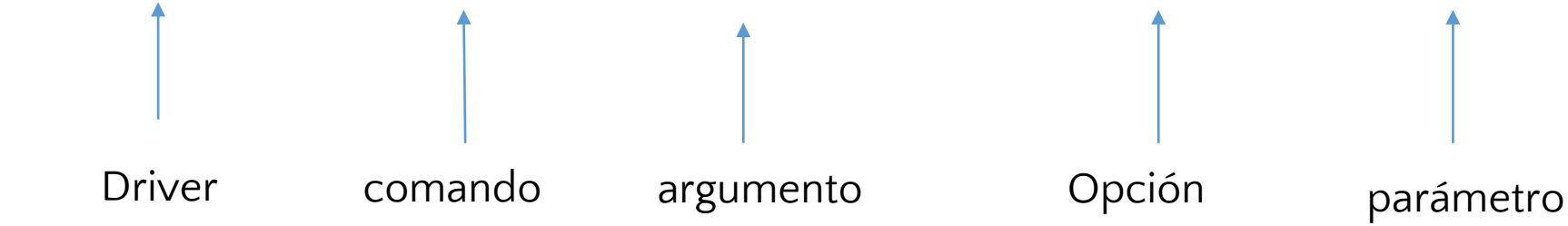
.Net Cli

La estructura de los commandos de CLI consisten en: El driver ("dotnet"), El commando, y la posibilidad de argumentos y opciones.

dotnet <comando> <argumentos> < argumentos> <opciones>

Comando de
Ejemplo:

> **dotnet** new console –Framework net8.0



.Net Cli

Algunos commandos útiles en .Net

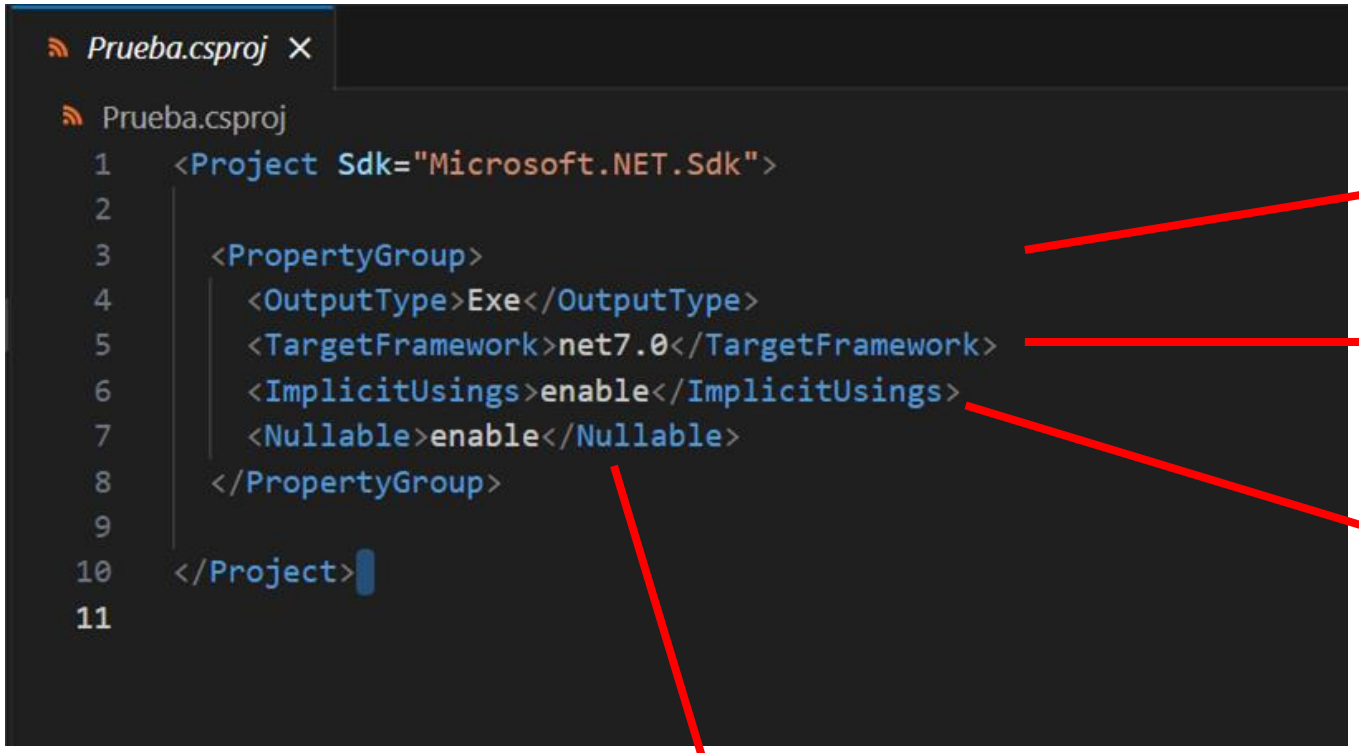
NET 7	
dotnet --version	Lista las versión del framework
dotnet --info	Muestra información variada del entorno y el sdk instalado
dotnet --list-sdks	Muestra los SDKs Instalados
dotnet new list	Lista los templates instalados para crear diferentes tipos de proyectos.
dotnet new console	Crea un nuevo proyecto de consola
dotnet new classlib -o nombre	Crea un nuevo proyecto de tipo librería de clases
dotnet run	Corre un proyecto si no se especifica el Path corre el proyecto local
dotnet build ruta	Se compila un proyecto en un path particular (se puede usar “./nombreApp.exe”
dotnet clean	Eliminar los archivos de la última copilación

Partes del SDK .net

- Tiempo de ejecución de .NET Core: GitHub.com/dotnet/coreclr
- Bibliotecas de .NET: GitHub.com/dotnet/corefx
- Herramientas e interfaz de la línea de comandos: GitHub.com/dotnet/cli
- Compilador Roslyn (C# y Visual Basic) y herramientas de lenguaje para Visual Studio: GitHub.com/dotnet/roslyn
- Documentación de .NET Core: GitHub.com/dotnet/core-docs

Comenzando con C#

Estructura de un proyecto de consola



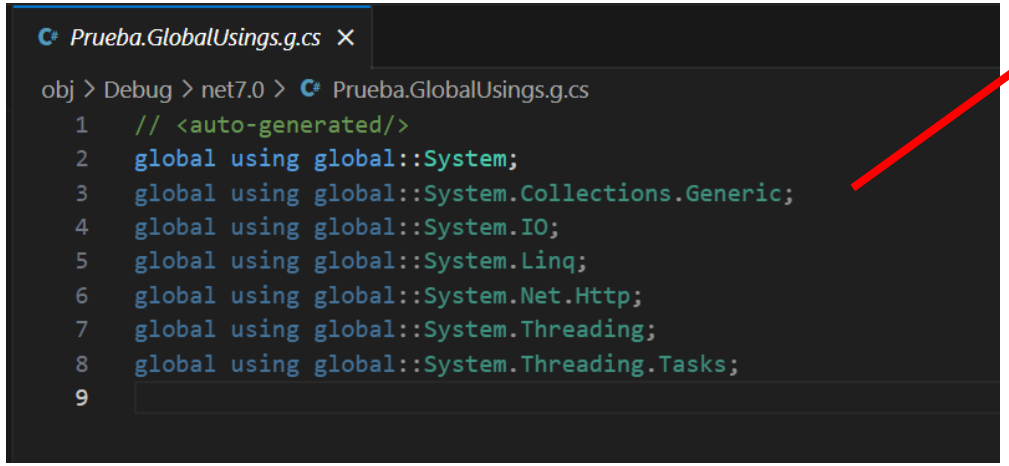
Tipo de salida

Framework Seleccionado

Carga un conjunto de librerías por defecto.
En caso de no encontrarse estas Librerías en las que necesitamos Debemos cargalas usando la palabra Reservada **using**

Cambia la forma en la que el Compilador se comporta con el El uso de null en los tipo referencia. Esto se presenta como una Advertencia verde en nuestro código

Estructura de un proyecto de consola



El archivo **Prueba.GlobalUsings.g** se genera de forma automática incluyendo ciertas librerías por defecto.

Podemos cambiar las libreías que se incluyen desde el archivo **.cproj** o incluso deshabilitar esta opción lo que hará que debamos incluir las dependencias a mano.

Ruta donde se genera esté archivo:
/obj/{reléase/debug}/ Prueba.GlobalUsings.g

Estructura de un proyecto de consola

```
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>net7.0</TargetFramework>
    <ImplicitUsings>enable</ImplicitUsings>
    <Nullable>disable</Nullable>
  </PropertyGroup>

  <ItemGroup>
      <Using Remove="System">
      <Using Include="System.Data">
  <ItemGroup>

</Project>
```

El archivo **Prueba.GlobalUsings.g** se regenera cada vez qué ejecutamos el código. Por lo tanto, si queremos modificar las librerías que se incluyen por defecto Podemos usar las especificaciones marcadas para Incluir o remover las librerías.

Si queremos deshabilitar este comportamiento podemos marcar como ***disable*** la opción ImplicityUsing.

Tipos en lenguaje C#

El sistema de tipos de C# contiene las siguientes categorías:

- Tipos de valor
- Tipos de referencia
- Tipos de puntero

Tipos en lenguaje C#

Los tipos de valor y los tipos de referencia son las dos categorías principales de tipos de C#.

Una variable de un tipo de valor contiene una instancia del tipo. En el caso de las variables de tipo valor pasar un argumento a un ables de tipo de valor, se copian las instancias de tipo correspondientes.

una variable de un tipo de referencia, que contiene una referencia a una instancia del tipo. De forma predeterminada, al asignar, método o devolver el resultado de un método, se copian los valores de variable.

Tipos valor

Un tipo de valor es un tipo de estructura o un tipo de enumeración. C# proporciona un conjunto de tipos de struct predefinidos denominados tipos simples. Los **tipos simples** se identifican mediante palabras reservadas.

Tipo de valor	Categoría	Sufijo de tipo
bool	Booleano	
byte	Sin signo, numérico, entero	
char	Sin signo, numérico, entero	
decimal	Numérico, punto flotante	M o m
double	Numérico, punto flotante	D o d
enum	Enumeración	
float	Numérico, punto flotante	F o f
int	Con signo, numérico, entero	
long	Con signo, numérico, entero	L o l
sbyte	Con signo, numérico, entero	
short	Con signo, numérico, entero	
struct	Estructura definida por el usuario	
uint	Sin signo, numérico, entero	U o u
ulong	Sin signo, numérico, entero	UL, Ul, uL, ul, LU, Lu, lU o lu
ushort	Sin signo, numérico, entero	

Mostrando algo por pantalla

El objeto Console

Representa los flujos de entrada, salida y error estándar para las aplicaciones de consola. Esta clase no puede heredarse.

```
Console.Read(); // lee un valor int

Console.ReadLine(); // lee una cadena de texto

Console.ReadKey(); // espera que el usuario presione una tecla

Console.WriteLine(“Hola"); // Muestra un mensaje por pantalla
```

Palabras clave y palabras contextuales de C#

Las palabras clave son identificadores reservados predefinidos que tienen un significado especial para el compilador.

Las palabras clave contextuales se usan para proporcionar un significado específico en el código, pero no son una palabra reservada en C#.

https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/

Probando un poco la sitaxis

Mostrar:

- Plugins visual studio para C#
- CLI de visual studio para crear proyectos.
- Sintaxis básica (tipos primitivos)
- El tipo String en C# - necesita ser nulleable
- Sintax suggar -> var (como funciona)
- Struct (?)
- Nullable types hay que agregar -> ?
- Null y formas de preguntar por null

DateTime

El DateTime tipo de valor representa fechas y horas con valores que van desde las 00:00:00 (medianoche), 1 de enero de 0001 Anno Domini (Era común) hasta las 11:59:59 p.m., 31 de diciembre de 9999 A.D. (D.C) en el calendario gregoriano.

Creando un nuevo objeto de tipo DateTime

```
DateTime Fecha = new DateTime(2023, 15, 5, 8, 30, 52);

DateTime Hoy = DateTime.Now; // tomo la fecha de hoy
```

DateTime

Inicialización de un objeto DateTime

Puede asignar un valor inicial a un nuevo DateTime valor de muchas maneras diferentes:

- Llamar a un constructor, ya sea uno en el que especifique argumentos para los valores o use el constructor sin parámetros implícito.
- Asignar un DateTime objeto al valor devuelto de una propiedad o método.
- Analizar un DateTime valor a partir de su representación de cadena.

Contenido muy útil

https://learn.microsoft.com/es-es/dotnet/api/system.datetime?view=net-7.0

Tipos de valor

El tipo ***struct*** es un tipo de valor que normalmente se usa para encapsular pequeños grupos de variables relacionadas:

```
public struct Coords
{
        public int x, y;
        public Coords(int p1, int p2)
        {
                x = p1;
                y = p2;
        }
 }
```

Tipos de referencia

Las palabras clave siguientes se usan para declarar tipos de referencia:

- class
- interface
- delegate

Tipos de referencia

El tipo String

C# usa la palabra clave string para representar el tipo de datos de cadenas de texto. La palabra clave string es un alias para el tipo System.String. Por lo tanto, string y String son equivalentes.

Una cadena es un objeto de tipo String cuyo valor es texto. Internamente, el texto se almacena como una colección secuencial de solo lectura de objetos Char. No hay ningún carácter que finalice en NULL al final de una cadena de C#; por lo tanto, la cadena de C# puede contener cualquier número de caracteres nulos insertados ("\0"). La propiedad Length de una cadena representa el número de objetos Char que contiene, no el número de caracteres Unicode.

Contenido muy útil

Cadenas: Guía de programación de C# con listas

Tipos de referencia

Conversión de string a tipo valor

Para determinar si una cadena es una representación válida de un tipo numérico especificado, use el método estático TryParse implementado por todos los tipos numéricos primitivos y también por tipos como DateTime y IPAddress.

```
int i = 0;
string s = "108";
bool resultado = int.TryParse(s, out i);  // i = 108 (numérico).
```

Estructuras de control de flujo

Estructuras de decisión

```
if (condición)
{
   instrucciones
}
else
{
   instrucciones
}
```

```
switch (expresion)
{
case constante 1:
            instruciones

case constante 2:
            instrucciones

default;
            instrucciones
break;
}
```

Estructuras de control de flujo

Ciclos iterativos

```
Do
{
        instrucciones
}
while(condicion);

while(condicion)
{
        condiciones
}


for (<inicialización>; <condición>; <modificación>)
{
        <instrucciones>
}
```

Estructuras de control de flujo

Navegar colecciones

```
foreach (<tipoElemento> <elemento> in <colección>)
{
        <instrucciones>
}
```

De lo anterior podemos ver:

- Primero se construye con MSBuild el lenguaje que hayamos programado, ya sea una solución, un proyecto de C#, un proyecto de Visual Basic, un proyecto de F#, etc.

- El mismo pasará al compilador de .NET, que actualmente es Roslyn, o en su defecto el de F# si fuera el caso.

- Después se obtiene un ejecutable (.EXE), un DLL si es una librería o lo que tengamos especificado a la hora de compilar.

- Cuando ejecutamos este código vamos a obtener el lenguaje intermedio (IL), que es un lenguaje, que se parece mucho a una especie de ensamblador, que crea .NET para tener una identificación del código que hemos aplicado ya en alto nivel. Se podría asemejar al Bytecode de Java.

- Finalmente se convierte a código nativo, es decir, en código ensamblador, y el mismo pasa a código binario.

- Entre medio de toda esa máquina virtual vamos a tener, como en muchos lenguajes de alto nivel, un garbage collector, es decir, no nos vamos a tener que preocupar en la liberación de memoria ni en que algo que se haya quedado colgado.

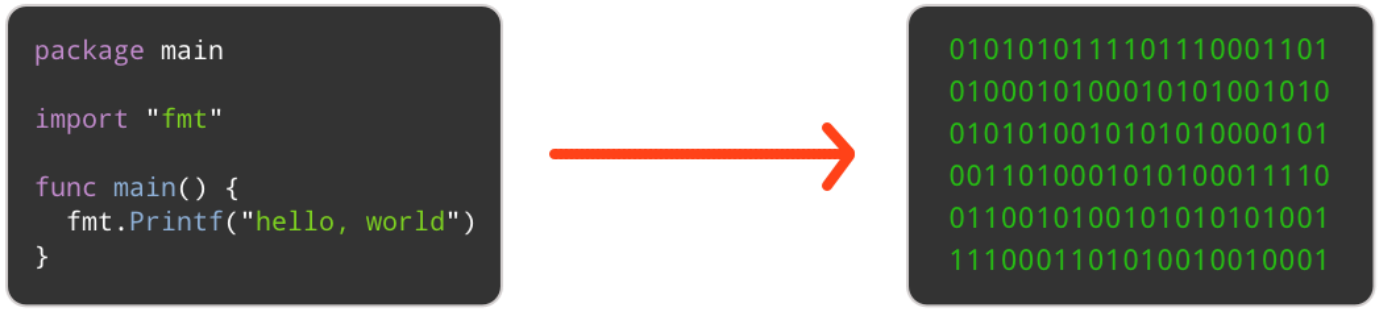
Compilando C#

Como compila C

(requiere un código distinto dependiendo de cada Arquitectura)

Lenguajes compilados

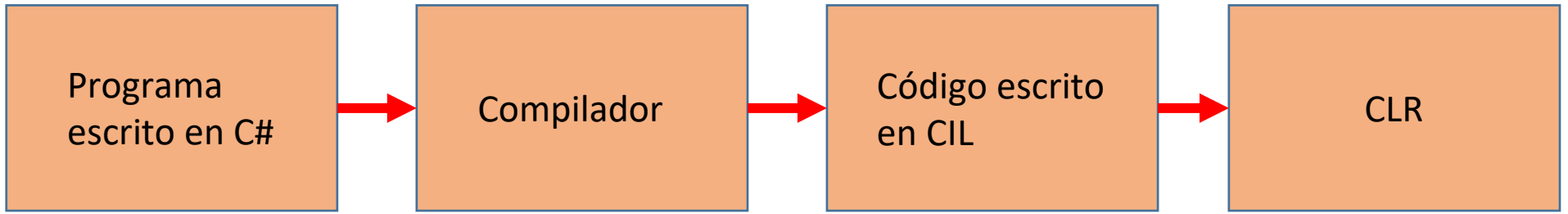
Ejemplos de Lenguajes compilados: C, C++, Go y Rust, entre



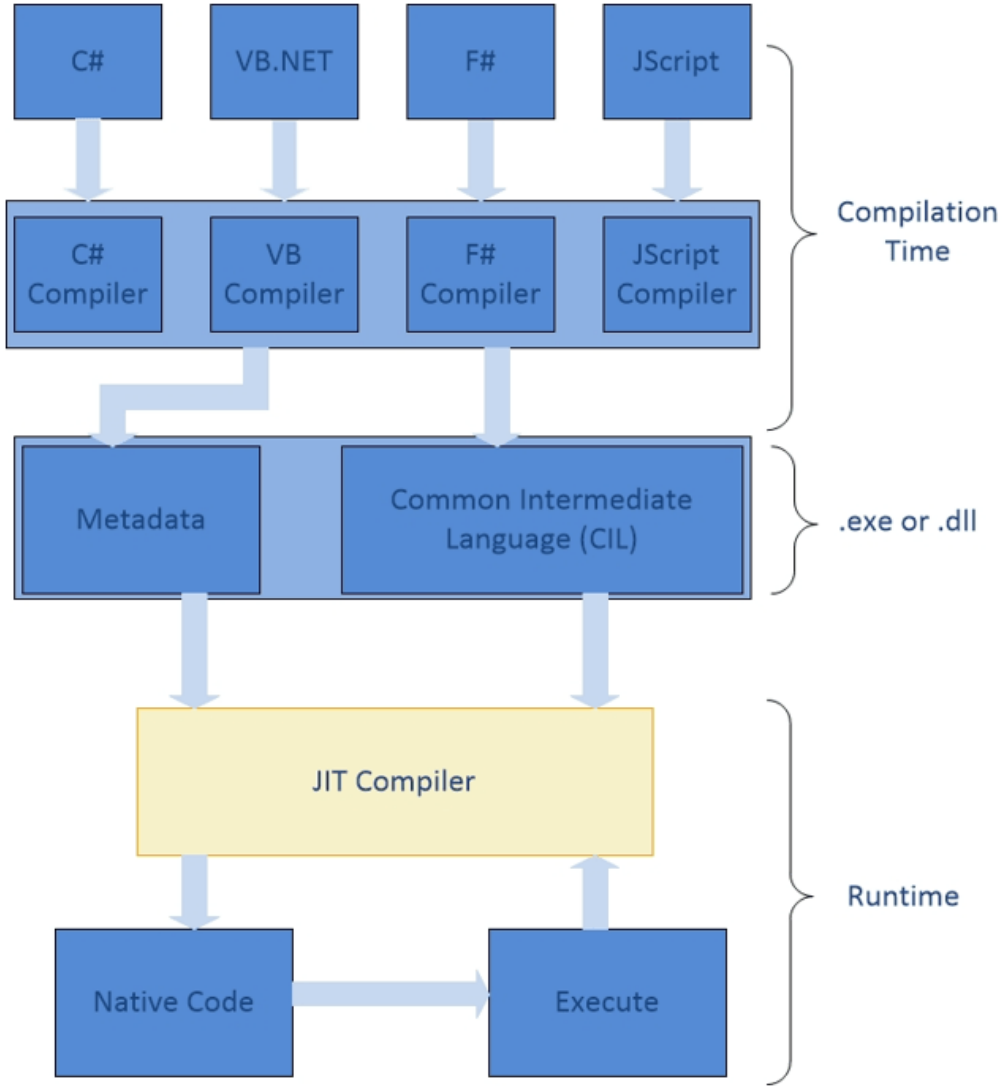
Lenguaje de alto nivel que entiende el programador

Lenguaje de máquina que entiende el procesador

Como compila .Net

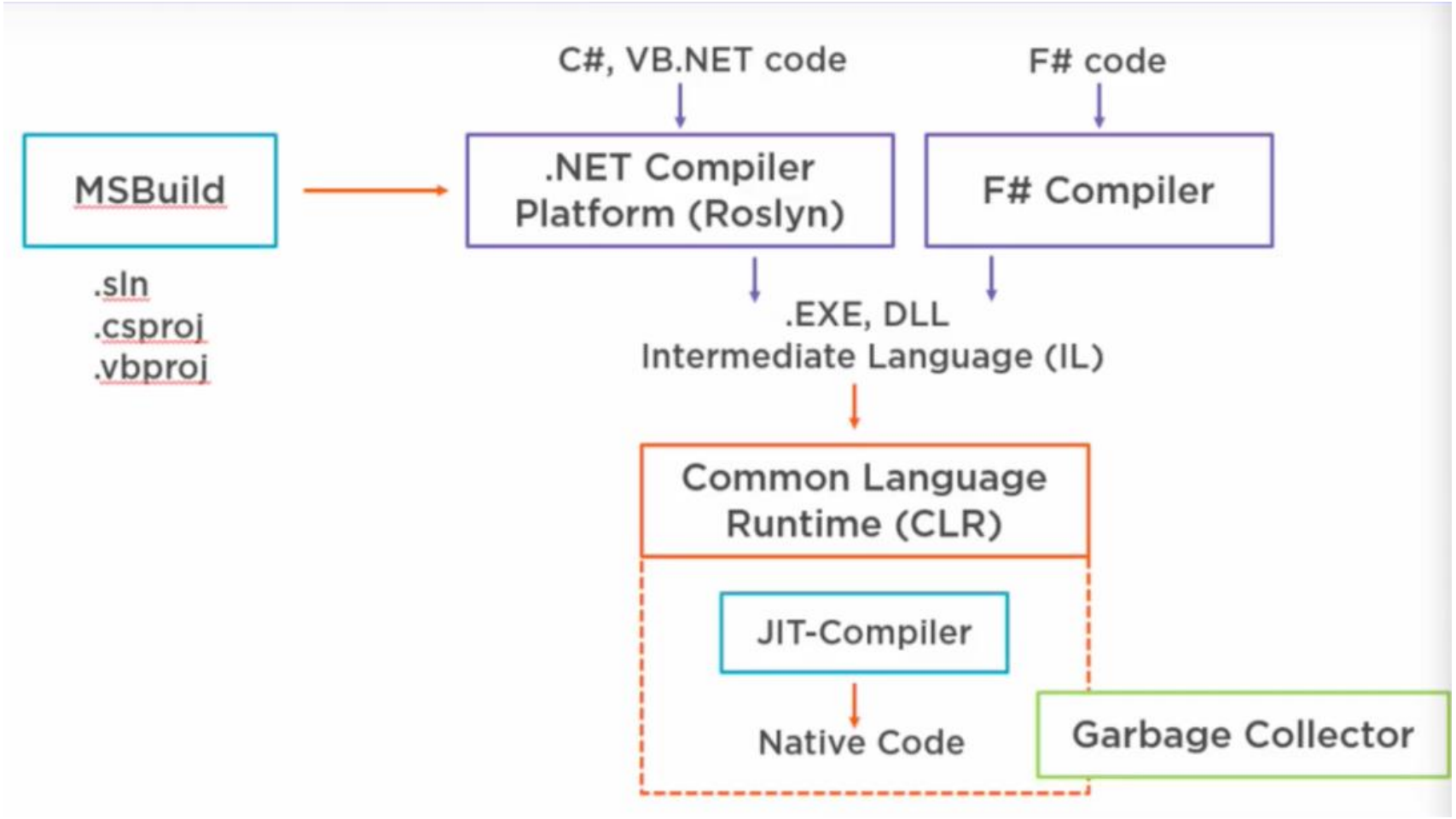


Rosly
n



Fuente: https://es.wikipedia.org/wiki/Common_Language_Runtime

Como compila .Net



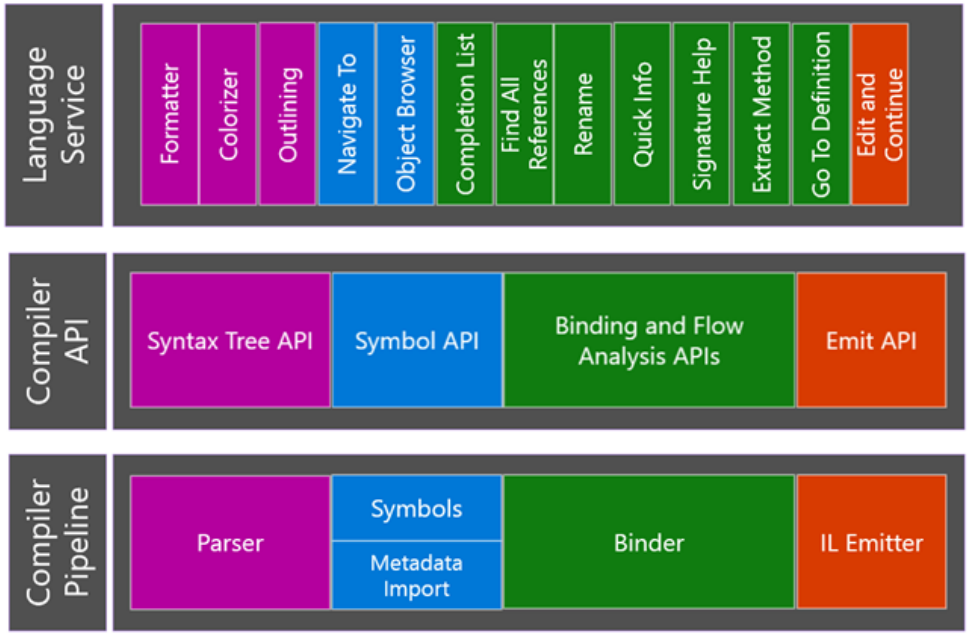
Rosly
n

Es el nuevo compilador de C# y Visual Basic

Cuenta con una serie de servicios de compilador para mejorar la interacción con las herramientas de desarrollo. El compilador es ahora el encargado de:

- realizar la comprobación de sintaxis,
- completar el código
- de realizar los refactors
- muchas cosas más, tanto dentro de Visual Studio como a través de la línea de comando.

El código está publicado bajo licencia Apache 2.0, una de las licencias más abiertas que hay



Roslyn

Algunas funcionalidades:

- Refactorings
- Renombrado inteligente de variables, en vivo desde el editor
- Nuevos refactorings como Introduce Local y Inline Temporary Variable
- Avisos (lightbulbs) para saber dónde podemos mejorar nuestro código
- Y muchas otras pequeñas mejoras como: coloreado de sintaxis en vista rápida, gestión más inteligente de nombres de variables en el Intellisense, etc...

¿Cómo se estructura una aplicación con C#?

- Los principales conceptos organizativos en C# son ***programas***, ***espacios de nombres***, ***tipos***, ***miembros*** y ***ensamblados***.

- Los programas de C# constan de **uno o más archivos de origen**. Los programas declaran *tipos*, que contienen miembros y pueden organizarse en espacios de nombres.

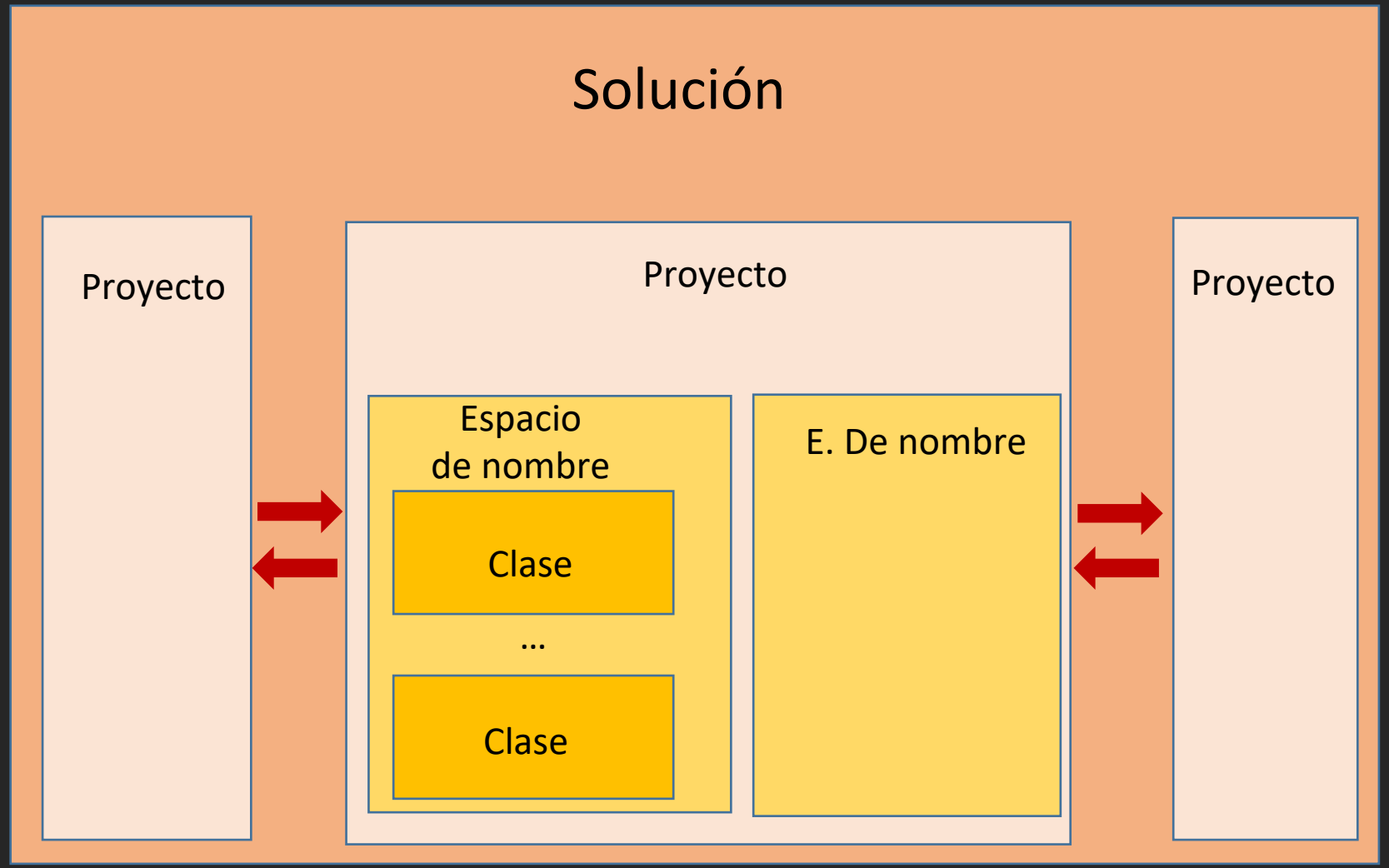
- Las clases e interfaces son ejemplos de *tipos*.

- Los campos, los métodos, las propiedades y los eventos son ejemplos de miembros. Cuando se compilan programas de C#, se empaquetan físicamente en ensamblados.

- Normalmente, los ensamblados tienen la extensión de archivo .exe o .dll de acuerdo a si es una **Aplicación** o **biblioteca de clases.**

Fuente: https://docs.microsoft.com/es-es/dotnet/csharp/tour-of-csharp/program-structure

¿Cómo se estructura una aplicación de C#?



Agrupador de proyectos
Su estructura define la tecnología.

Agrupador de proyectos
Su estructura define la tecnología.

Agrupador de espacios de nombre
Los usaremos como capas (layer) en nuestro sistema.

namespace
Agrupador de clases

Clase
Unidad mas pequeña

Bibliografía

https://webmiguel.com/blog/que-es-un-framework/

Fuente: https://es.stackoverflow.com/questions/23239/que-diferencia-existe-entre-api-biblioteca-y-framework

https://learn.microsoft.com/en-us/dotnet/standard/glossary

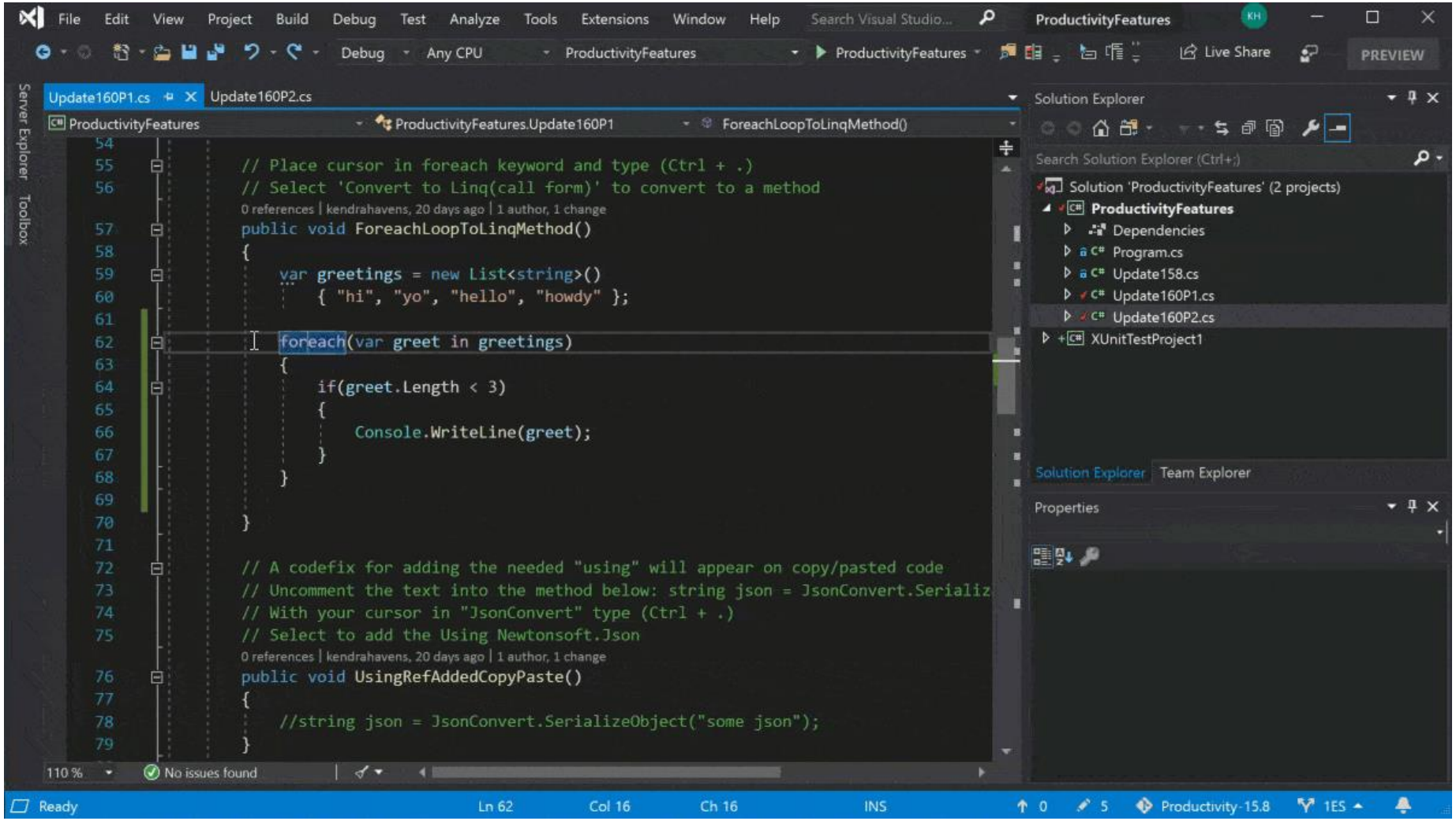
.NET Framework version history - Wikipedia

https://devblogs.microsoft.com/cesardelatorre/net-core-1-0-net-framework-xamarin-the-whatand-when-to-use-it/

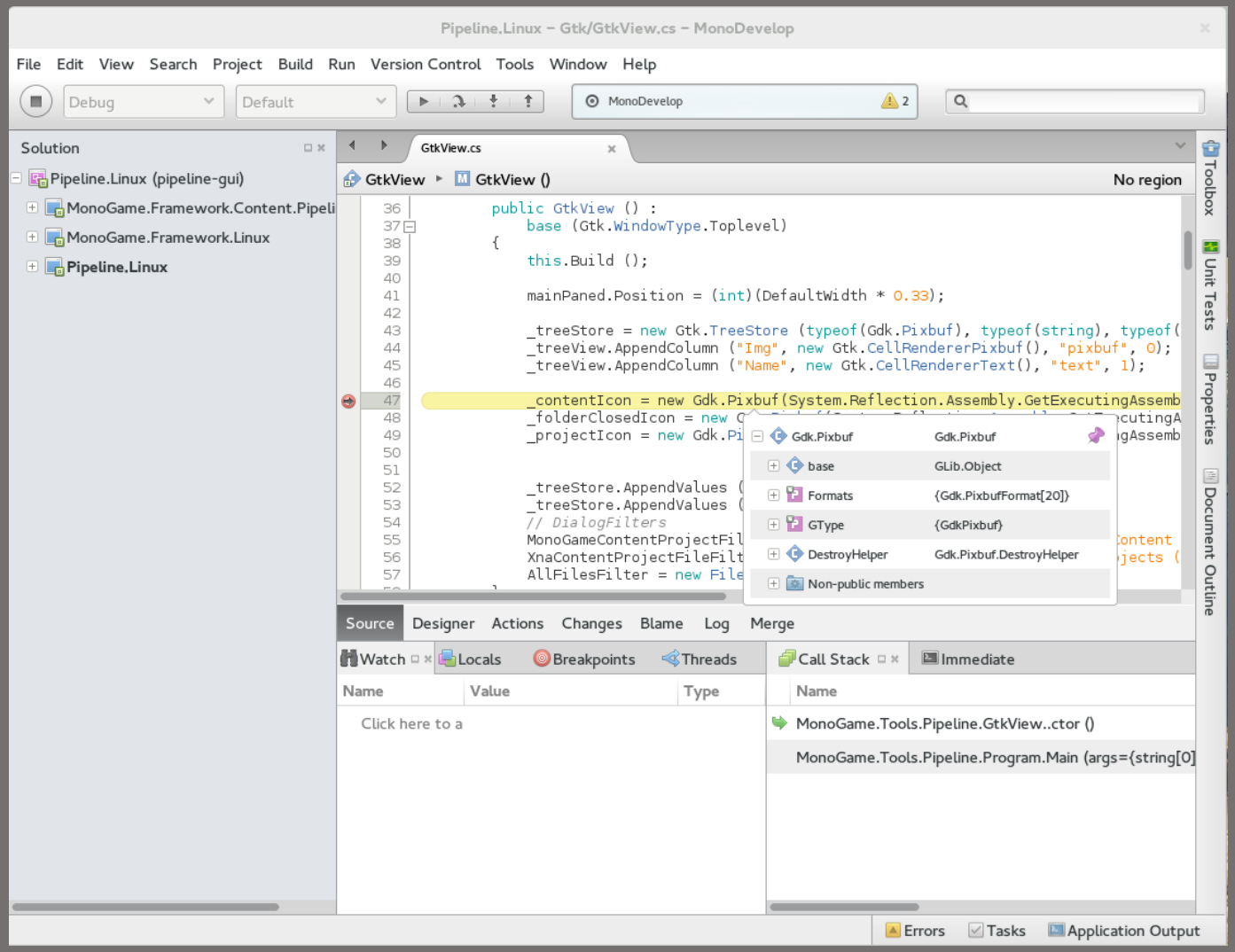
Cómo garantizar el soporte a largo plazo de tu aplicación .NET | campusMVP.es

Proyecto de consola

Basta de chachara, Manos a la obra…



MonoDevelop



Funciona en:

- Windows
- Mac
- Linux

Permite desarrollar para:

- Linux GTK# Apps
- Windows

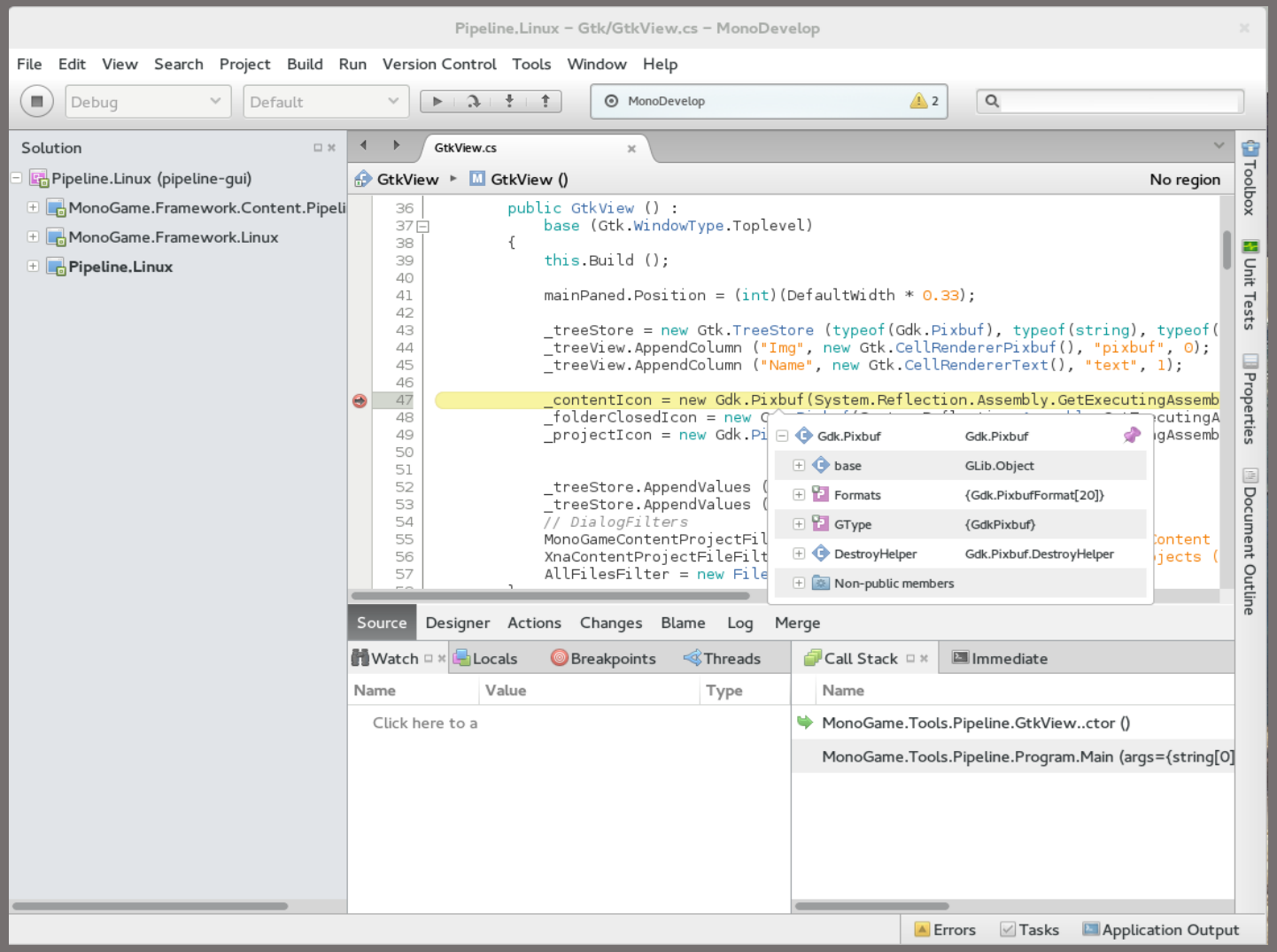
Algunas características:

- Varios lenguajes
- Core de MonoDevelop está licenciado en LGPLv2

Descarga

http://www.monodevelop.com/

VS CODE



Funciona en

- Windows
- Mac
- Linux

Permite desarrollar Para

- IOS
- ANDROID
- Windows
- Mac

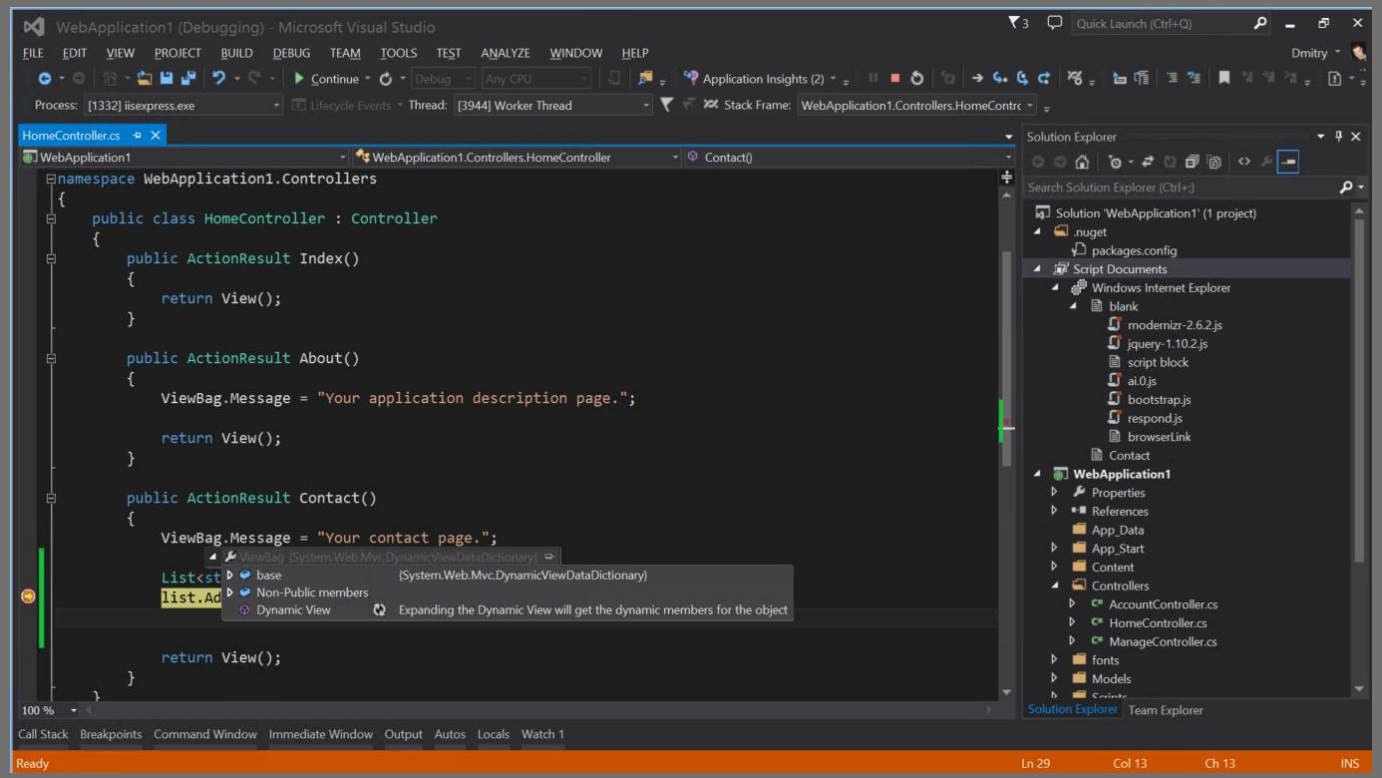
Algunas características

- Permite desarrollar en 30 lenguajes distíntos, entre ellos, C#, Java, Phyton, node.js.
- Proyecto Open source
- Gran cantidad de extensiones y corre de forma nativa los comandos de Git.

Descarga

https://code.visualstudio.com/

Visual Studio



Funciona en

- Windows

Permite desarrollar Para

- IOS
- ANDROID
- Windows universal app
- Mac

Descarga

https://www.visualstudio.com/