

UT.1-Repaso de POO e introducción al C#

Contenidos

En esta Unidad de Trabajo vamos a tratar los siguientes puntos:

- Configuración del Editor de C# en Unity. *Diapositivas 3-4.*
- Creación y estructura de Script en Unity. *Diapositivas 5-17*
- Documentación de Unity sobre librerías propias. MonoBehaviour. *Diapositivas 18-30*
- Criterios de evaluación. *Diapositiva 1*

Material Adicional a esta presentación:

• Vídeos:

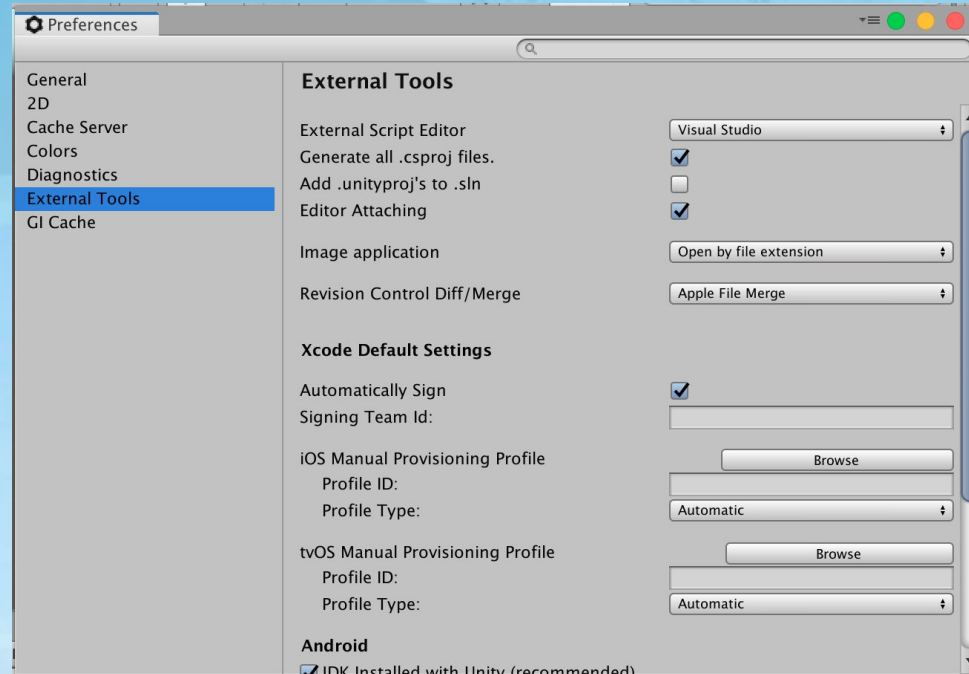
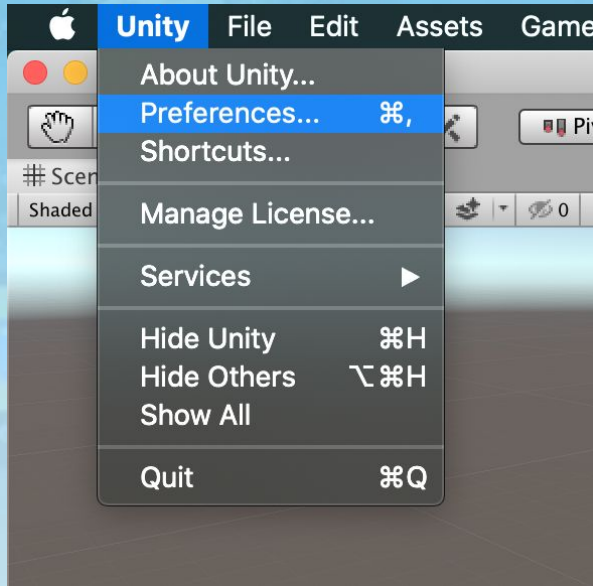
- UT1-ConfiguracionEditor.
- UT1-CreaciónEstructuraScripts.
- UT1-DocumentacionUnity.

• Prácticas:

- Práctica_UT1_RepasoPOO

Configuración del Editor de C# en Unity.

Para configurar nuestro Editor de Código, vamos a entrar en “External Tools”, siguiendo los pasos que muestran las imágenes.



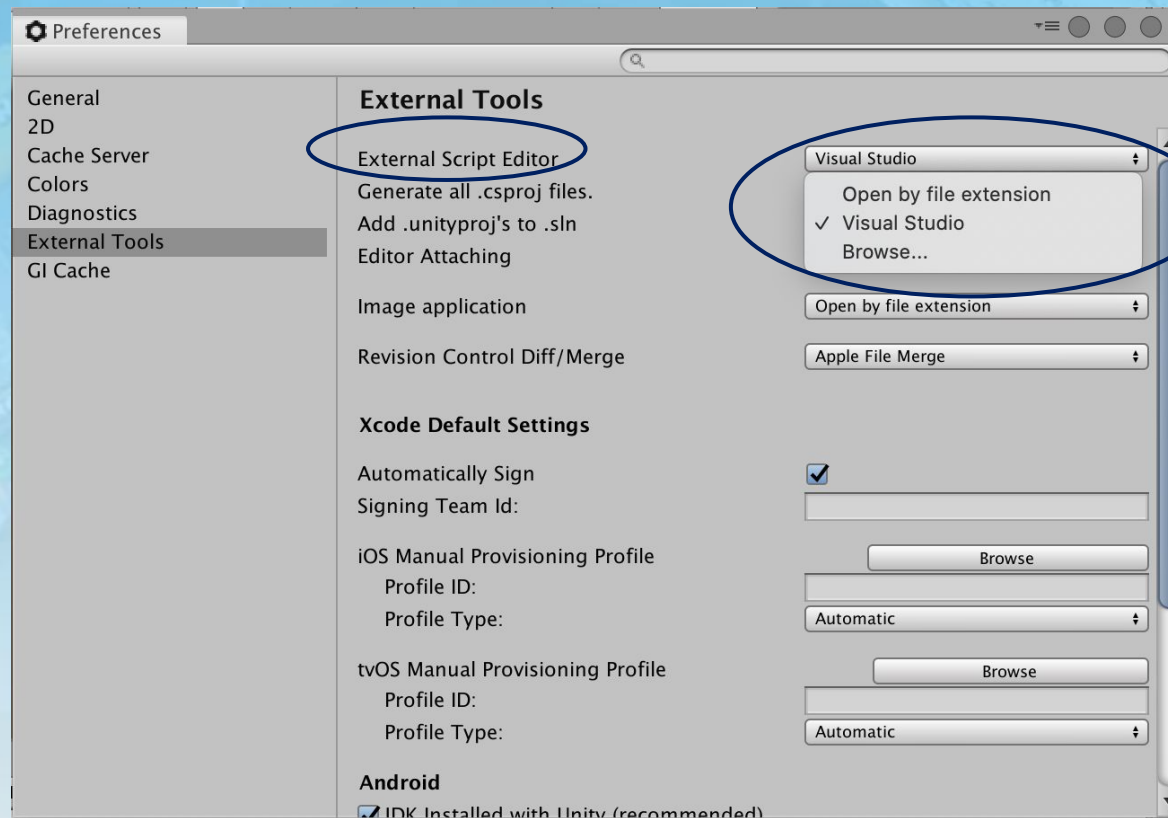
Ver el vídeo [UT1_ConfiguracionEditor.mp4](#)



En esta presentación se está utilizando IOS, la forma de entrar en Windows, aún siendo similar, puede tener las Preferencias en el menú “Edit”.

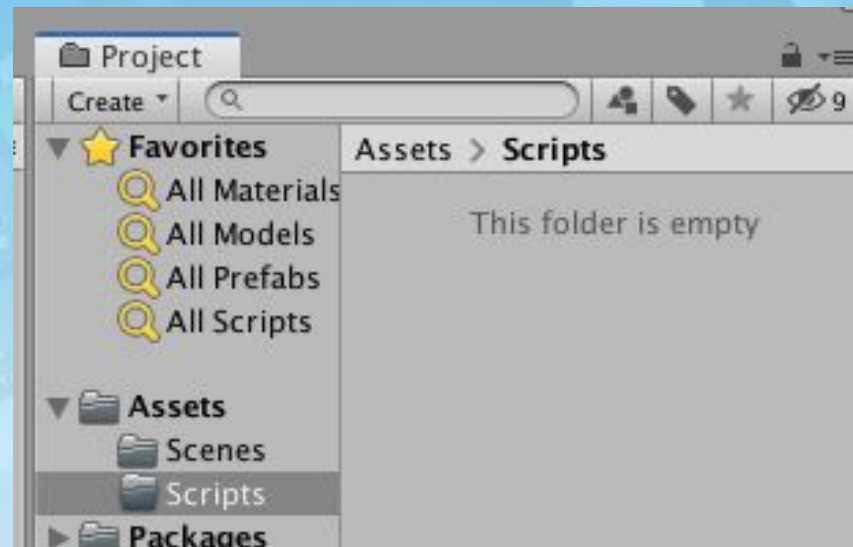
Configuración del Editor de C# en Unity.

Una vez dentro de "External Tools", nos debemos fijar en el apartado que pone "External Script Editor", en el podremos seleccionar el editor que queramos utilizar.



Creación y estructura de un Script en Unity.

- Lo primero y para tener **ordenado** todos los componentes en Unity, vamos a crear una carpeta que se va a denominar “Scripts”, en la que guardaremos todos los scripts creados en nuestra aplicación.



 [Ver el vídeo UT1_CreacionEstructuraScripts.mp4](#)

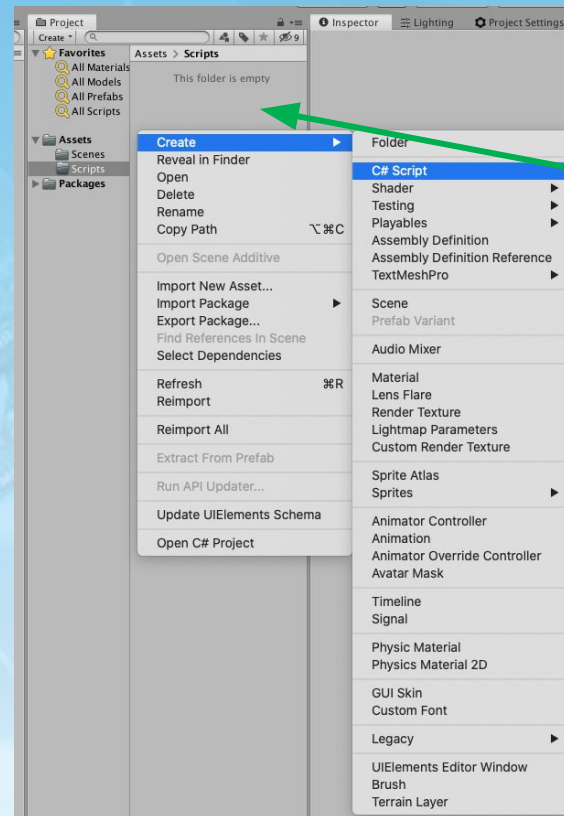


Para crear carpetas, script, meter imágenes, etc, lo podemos hacer desde el entorno de Unity o desde la carpeta que se ha creado al iniciar el proyecto.

Creación y estructura de un Script en Unity.

- Una vez creada la carpeta Script y situados dentro de la misma, vamos a crear un script nuevo. Tenemos varias opciones:

- Menú Contextual:**

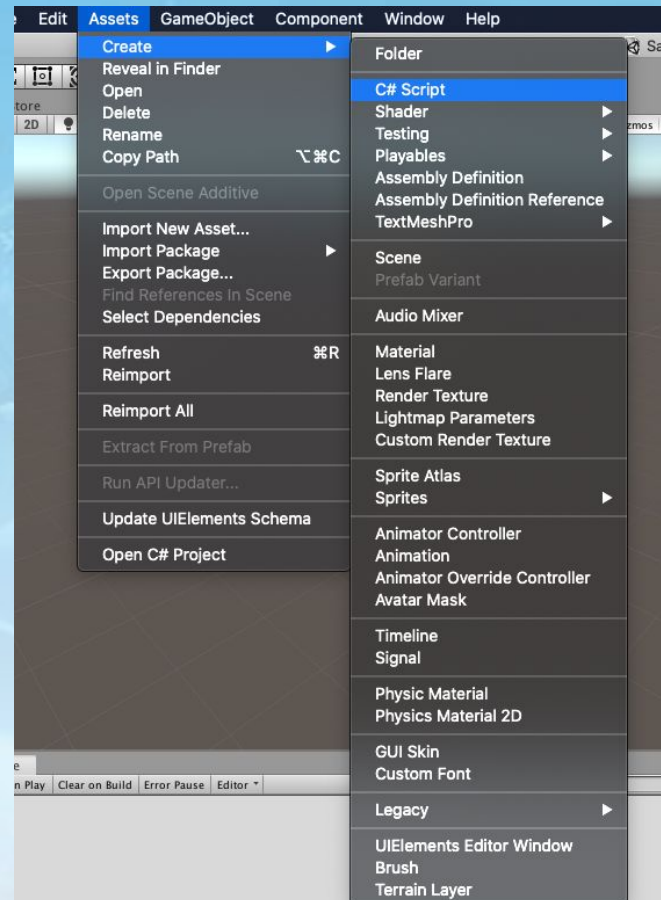


Para sacar el menú contextual, pulsar el botón derecho en el espacio que indica la flecha.

Creación y estructura de un Script en Unity.

- Una vez creada la carpeta Script y situados dentro de la misma, vamos a crear un script nuevo. Tenemos varias opciones:

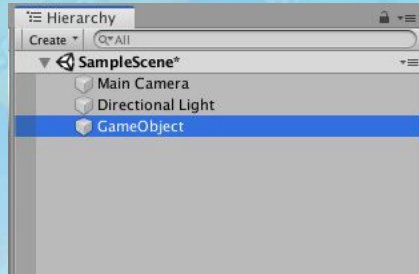
- Menú de Unity:**



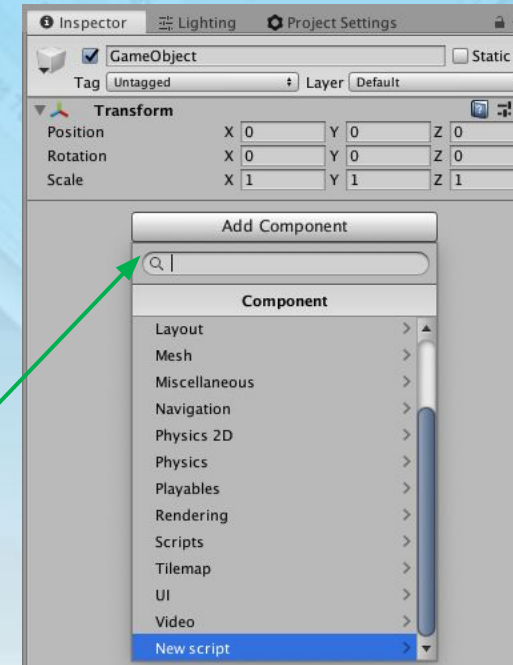
Es necesario tener seleccionada la carpeta de Script en Project para que el script se cree dentro de ella.

Creación y estructura de un Script en Unity.

- Una vez creada la carpeta Script y situados dentro de la misma, vamos a crear un script nuevo. Tenemos varias opciones:
 - **Una tercera opción es crearlo directamente sobre un GameObject:**
 - Seleccionamos el GameObject en el que queremos que se vincule nuestro script:



- Desde la ventana Inspector seleccionamos la siguiente opción:



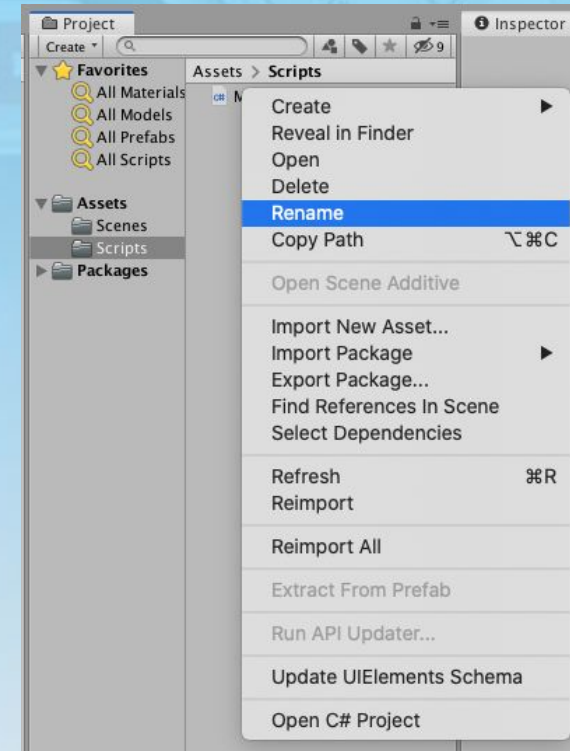
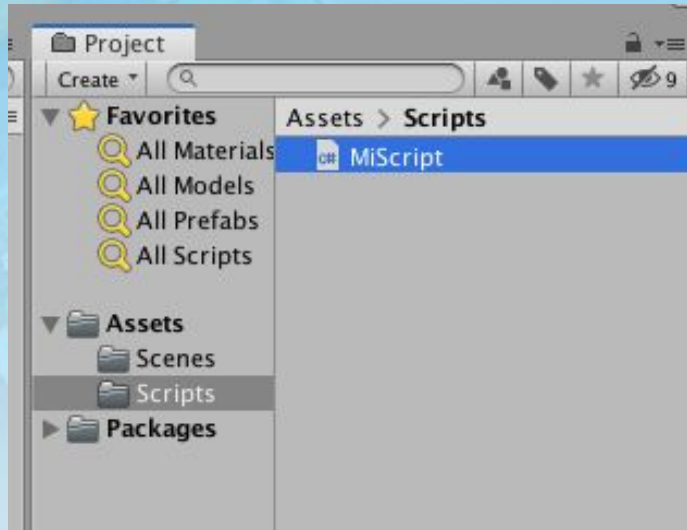
Otra forma de crear un script, es introducir directamente su nombre en el buscador del botón "Add Component". Si no se encuentra un componente con ese nombre, Unity entiende que es un script nuevo.

Creación y estructura de un Script en Unity.

- Una vez seleccionada la opción de crear un script, nos aparecerá dentro de la carpeta de Script (que debemos tener seleccionada). En ese momento, podremos poner el nombre que queramos al script. También podemos utilizar la opción “rename”, desde el menú contextual.



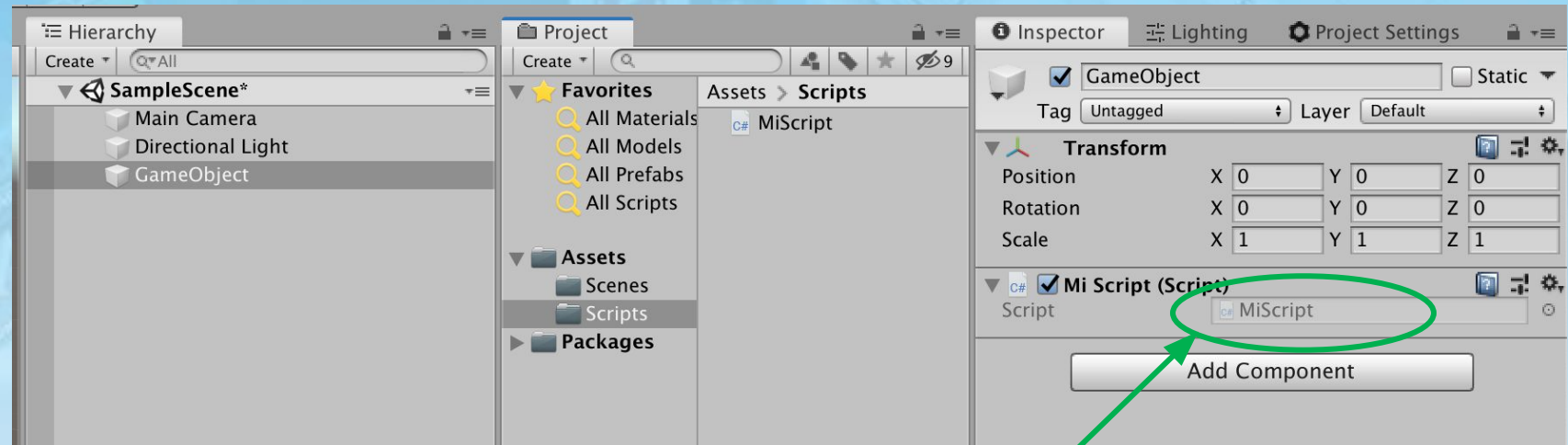
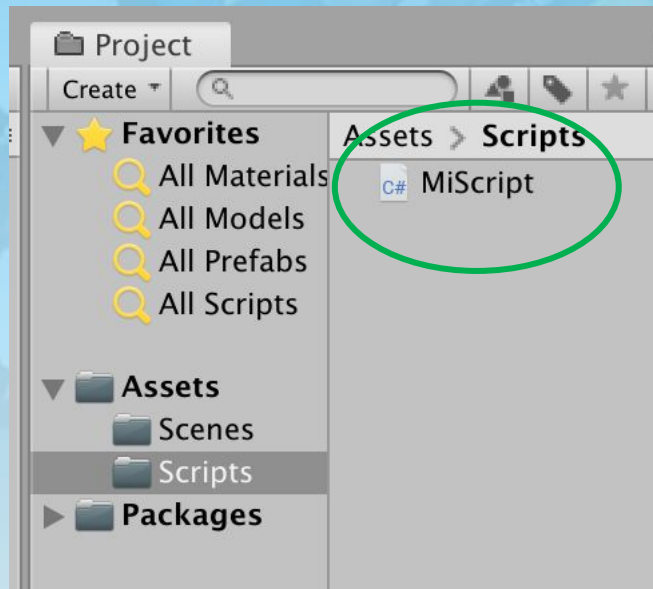
El nombre del Script y de la clase deben coincidir.



El Script por defecto se llama “NewBehaviourScript”.

Creación y estructura de un Script en Unity.

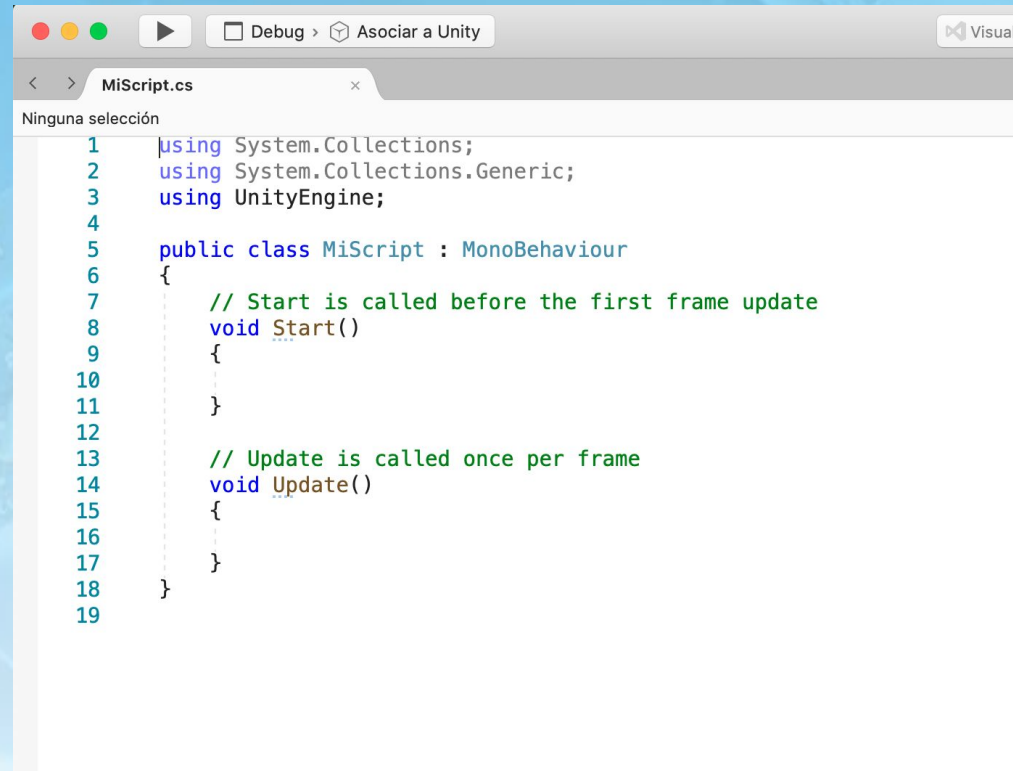
- Una vez creado el script, si queremos editarlo desde el editor que hemos configurado, podemos hacer doble click, con el botón izquierdo del ratón sobre el fichero del script o sobre el script ya asociado a un GameObject.



Hacer doble click encima del nombre, en la posición que indica la flecha.

Creación y estructura de un Script en Unity.

- Una vez abierto el script, nos vamos a encontrar con algo similar a lo que nos muestra la siguiente imagen.



```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class MiScript : MonoBehaviour
6 {
7     // Start is called before the first frame update
8     void Start()
9     {
10
11     }
12
13     // Update is called once per frame
14     void Update()
15     {
16
17     }
18 }
19
```



En Mac, el visual studio, tiene esta apariencia, en Windows, cambiará.

Creación y estructura de un Script en Unity.

- El script que nos ha creado Unity, nos muestra varias cosas.
 - Nos muestra el nombre de las librerías que Unity utiliza por defecto. Veremos que para poder trabajar con la GUI, tendremos que cargar alguna más.

```
1    using System.Collections;
2    using System.Collections.Generic;
3    using UnityEngine;
4
```

- Lo siguiente que nos muestra, es el nombre de nuestro script. Importante ver que lo está declarando como una clase. Después del nombre y seguido de dos puntos, aparece la palabra “MonoBehaviour”, esta es la clase base de Unity.

```
public class MiScript : MonoBehaviour
```


Creación y estructura de un Script en Unity.

- El script que nos ha creado Unity, nos muestra varias cosas.
 - Lo siguiente que nos encontramos es un par de métodos, de los más importantes, aunque existen otros que utilizaremos y además los que creamos nosotros de forma pública (public) o privada (private).

```
// Start is called before the first frame update
void Start()
{
    // Update is called once per frame
    void Update()
    {
    }
```

- El método **Start()** es un método que se ejecuta únicamente una vez, cuando se carga el GameObject en el que está asociado el script.
- El método **Update()** se ejecuta en cada frame. Este tiempo de ejecución entre frame, se puede modificar y lo haremos más adelante.

Creación y estructura de un Script en Unity.

- Algunas cosas características de C#.
 - No existe un begin-End. En este lenguaje de programación se utilizan las {}, tanto para las clases, como para los métodos y para las estructuras de control.
 - Como en otros lenguajes de programación, después del nombre del método, aparecen unos paréntesis, que es en los que vamos a indicar los parámetros de entrada al script.
 - Las variables o métodos, pueden ser públicos o privados.
 - En los métodos, la palabra reservada “public” o “private” se pone al principio del todo.
 - Luego le sigue otra palabra reservada que puede ser “void”, si el método no devuelve nada, float, si el método devuelve un float, string, si el método devuelve un string, etc...

Creación y estructura de un Script en Unity.

- Algunas cosas características de C#.
- Utilizaremos una estructura para atrapar los errores y corregir esas situaciones.
- Al terminar cada sentencia de nuestro código, debemos de añadir un ";", menos en la declaración de una clase o de un método.
- Para añadir comentarios, debemos de utilizar "//" antes de la línea del comentario. En el caso de ser varias líneas, podemos utilizar al principio del comentario "/*" y al final "*/".
- Al trabajar con clases, podemos utilizar herencia y polimorfismo.
- Las clases en C#, disponen de constructores.

Creación y estructura de un Script en Unity.

- Vamos a seguir las siguientes reglas a la hora de realizar nuestros scripts.
 - Las variables se declararán al principio de la clase o de los métodos.
 - Los comentarios nunca se pondrán al lado de una sentencia, siempre en la línea anterior.
 - Si los comentarios constan de varias líneas, no se pondrá nunca una línea inicial y otra final llena de asteriscos.
 - Las variables empezarán siempre con una letra minúscula, nunca un número ni un signo.
 - Las clases y los métodos empezarán con una letra Mayúscula, nunca un número ni un signo.
 - Utilizaremos tabulaciones para indicar los bloques de ejecución.

Creación y estructura de un Script en Unity.

- Vamos a seguir las siguientes reglas a la hora de realizar nuestros scripts.
 - Utilizaremos los siguientes prefijos:
 - Panel: pnl
 - TextBox: txt
 - Button: btn
 - Image: img
 - Toggle: tgl
 - Slider: sld
 - ListBox: lst
 - Checkbox: chk

Creación y estructura de un Script en Unity.

- Vamos a seguir las siguientes reglas a la hora de realizar nuestros scripts.
 - Los métodos que se creen deben de estar correctamente comentados, para saber en que van a ser utilizados.
 - Al principio de cada clase, el alumno añadirá un conjunto de líneas de comentario en el que aparecerá su nombre, su curso, fecha y una breve descripción de para que se usa esa clase.

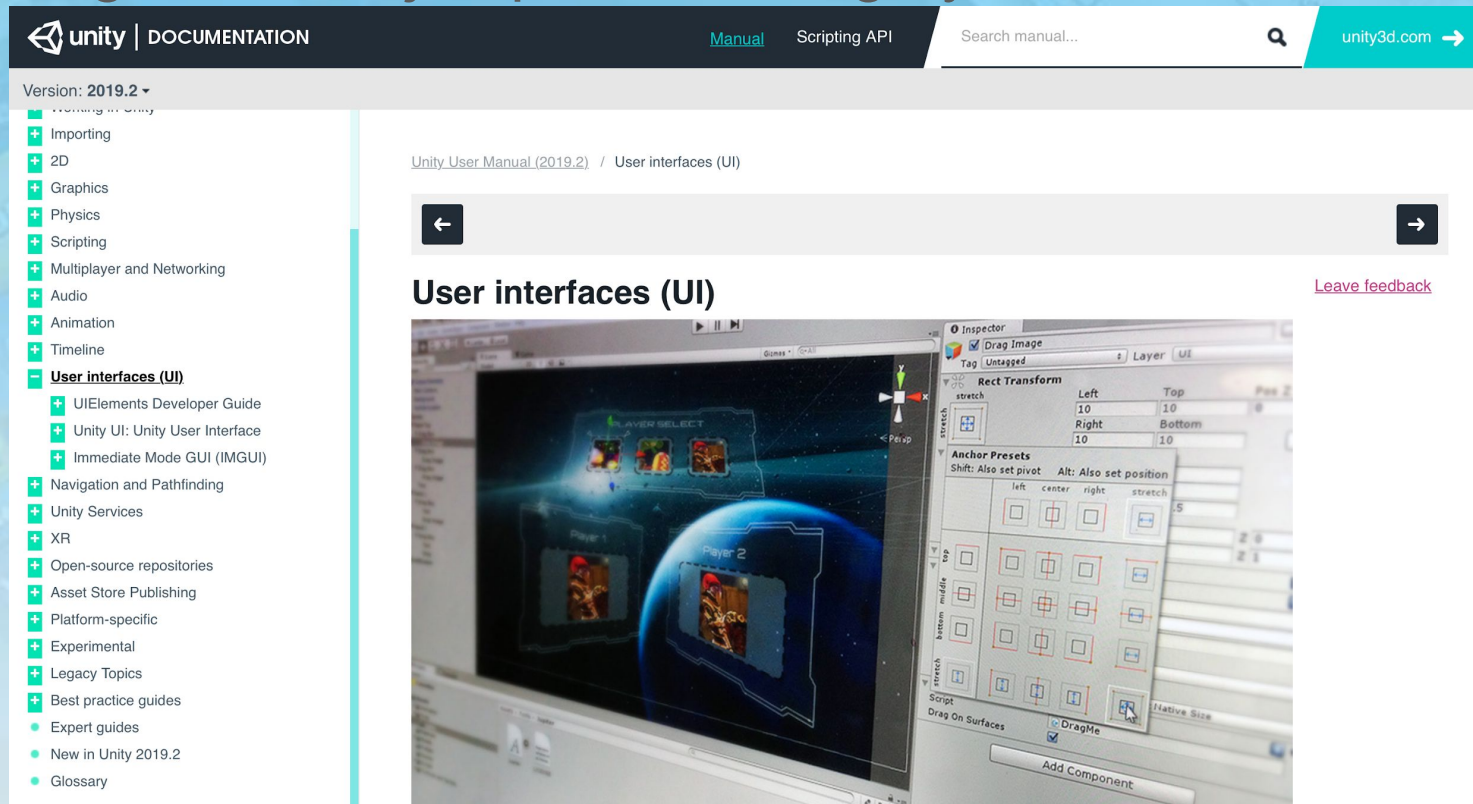
```

1      /*
2      * Nombre:.....
3      * Curso:.....
4      * Fecha:.....
5      * Descripción:.....
6      */
7
8      using System.Collections;
9      using System.Collections.Generic;
10     using UnityEngine;
11

```

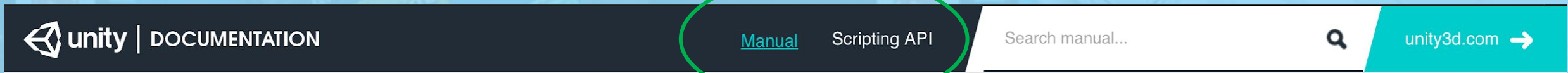
Documentación de Unity. MonoBehaviour.

- Unity dispone de una documentación muy potente en la que se pueden encontrar parámetros de configuración, ejemplos de código y diferentes reseñas.

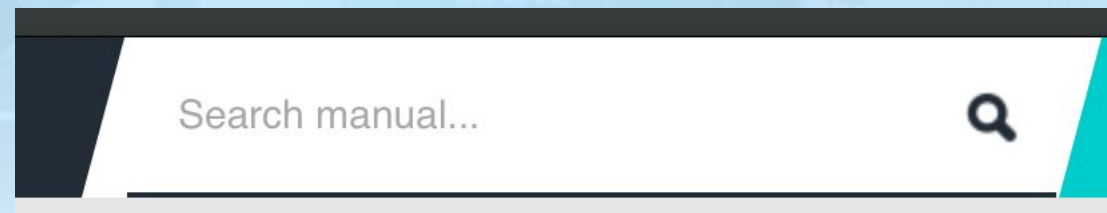


Documentación de Unity. MonoBehaviour.

- Como se observa en la imagen, tenemos los siguientes bloques principales:
 - Manual
 - Scripting API

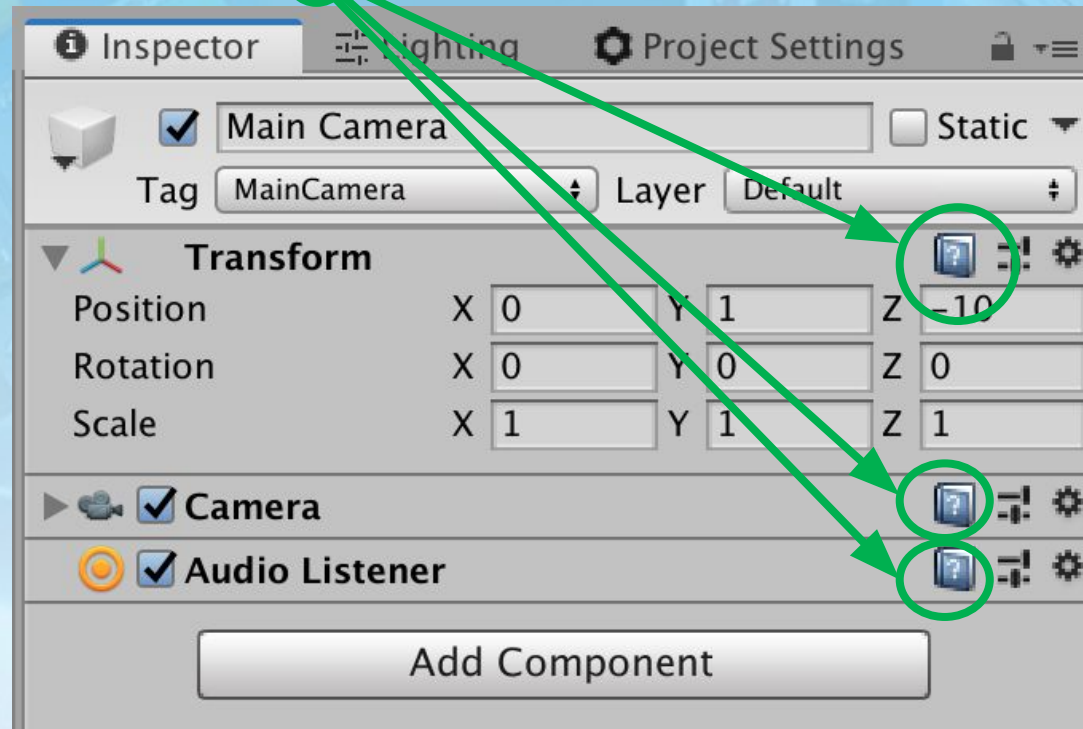


- Además de tener las dos secciones anteriores, tenemos un buscador manual, en el que escribimos el nombre del método, librería, etc., de lo que queremos información y nos la buscará.



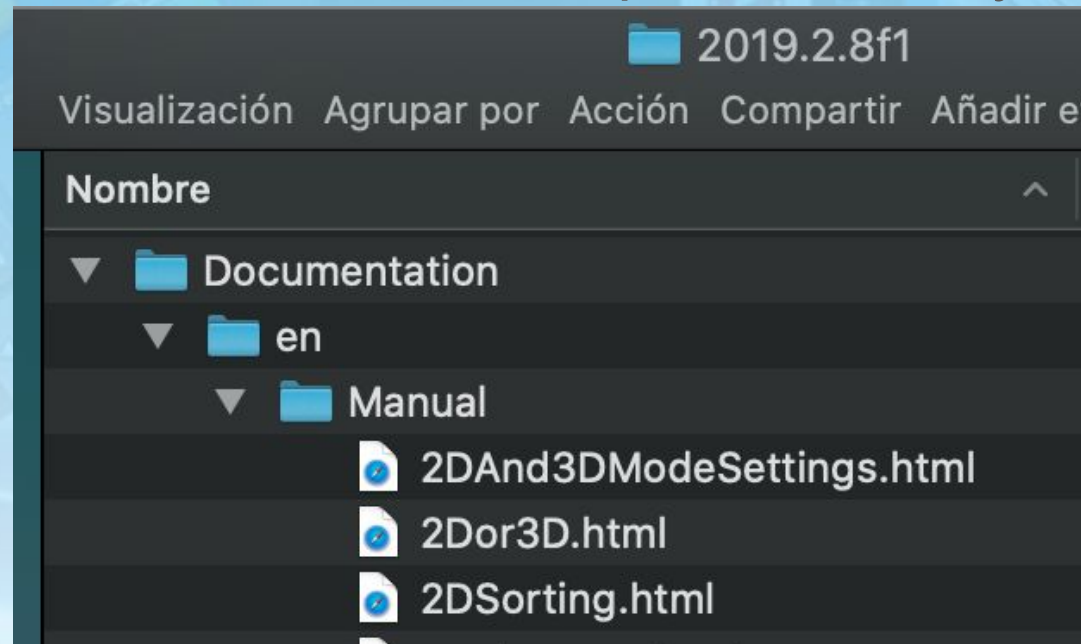
Documentación de Unity. MonoBehaviour.

- Para acceder a la ayuda de Unity, tenemos tres caminos posibles.
 - Desde el propio entorno de Unity. Cuando pulsamos sobre uno de estos iconos, situados en la ventana “Inspector”, se nos abrirá nuestro navegador y nos mostrará la documentación.



Documentación de Unity. MonoBehaviour.

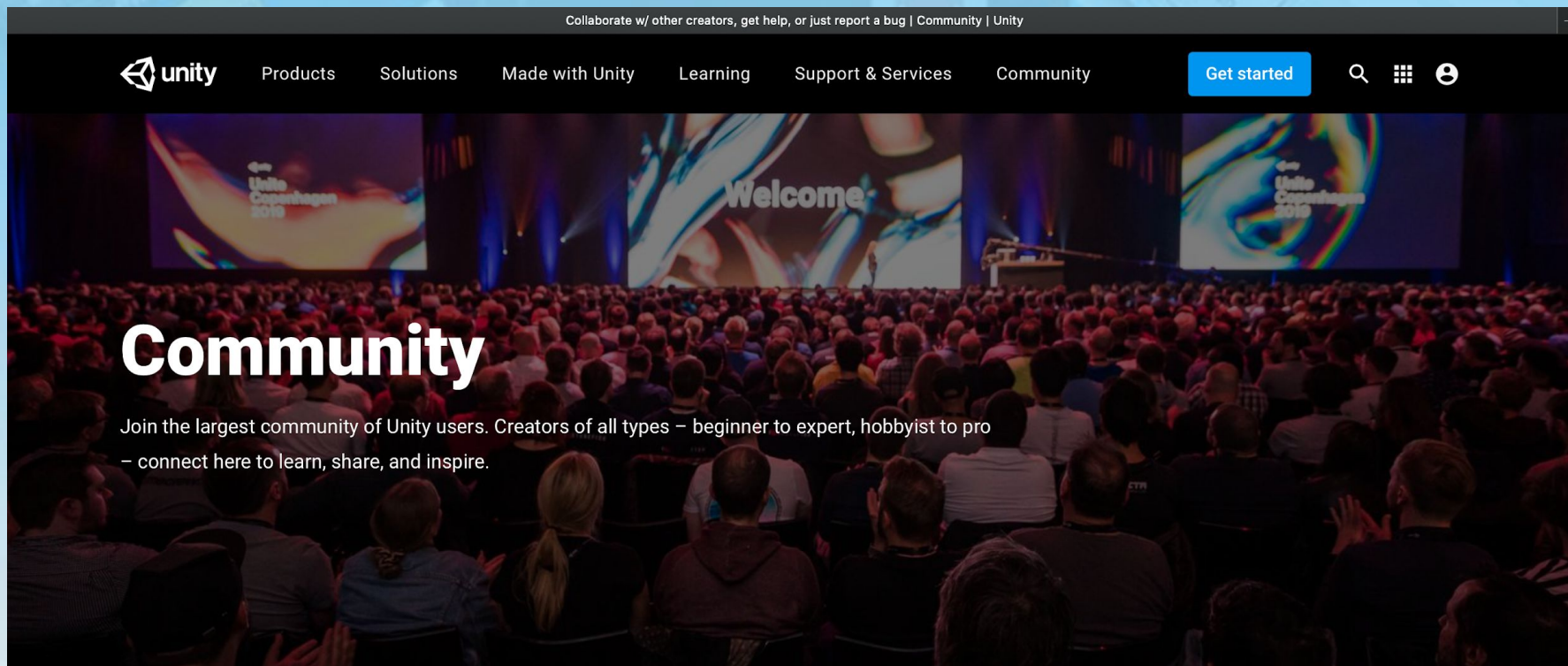
- Para acceder a la ayuda de Unity, tenemos tres caminos posibles.
 - Cuando hacemos la instalación de una versión de Unity, en la carpeta de instalación, se nos guarda una subcarpeta denominada “Manual”, en la que se guarda en local, toda la documentación que tiene Unity en el servidor.



Hacer doble click, con el botón izquierdo del ratón, sobre index.html.

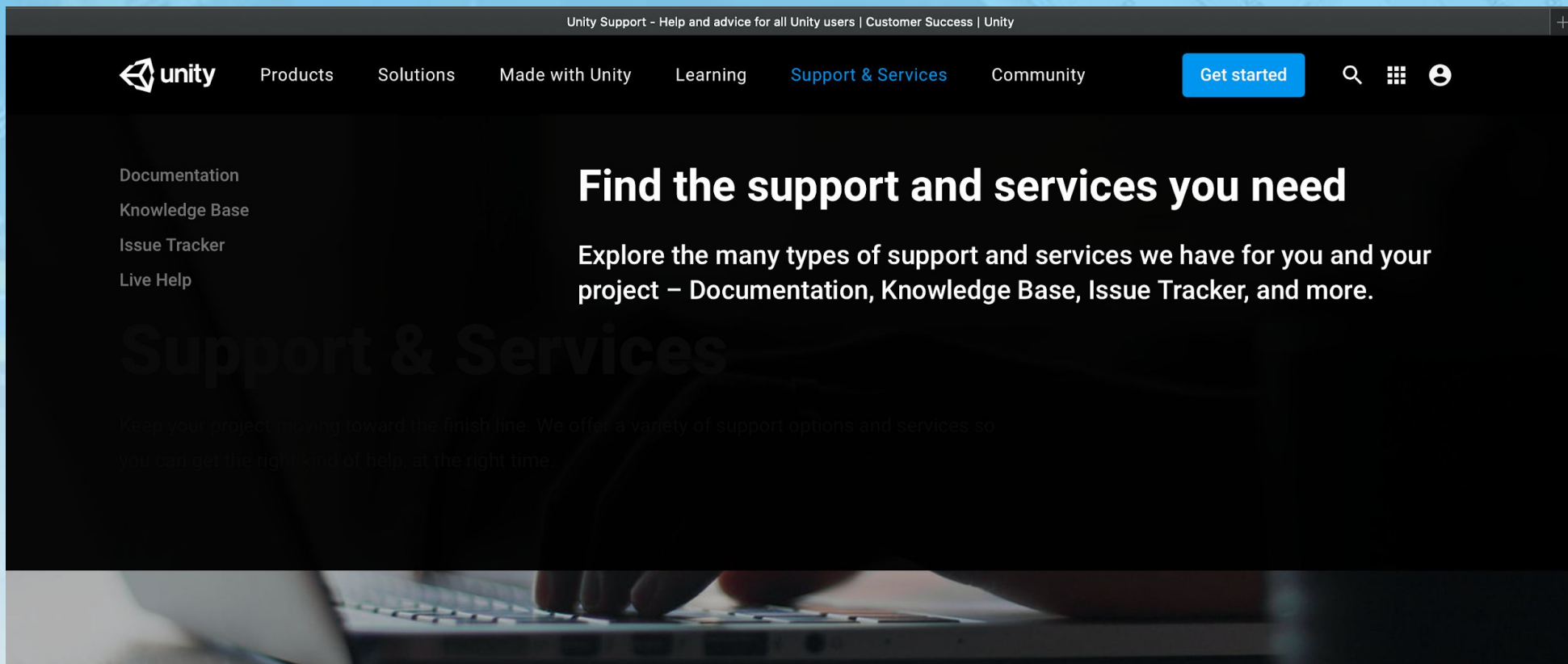
Documentación de Unity. MonoBehaviour.

- Para acceder a la ayuda de Unity, tenemos tres caminos posibles.
 - La tercera y última forma de entrar a la documentación, es directamente a través del servidor de Unity. Lo primero que hacemos es entrar en <http://www.Unity.com>



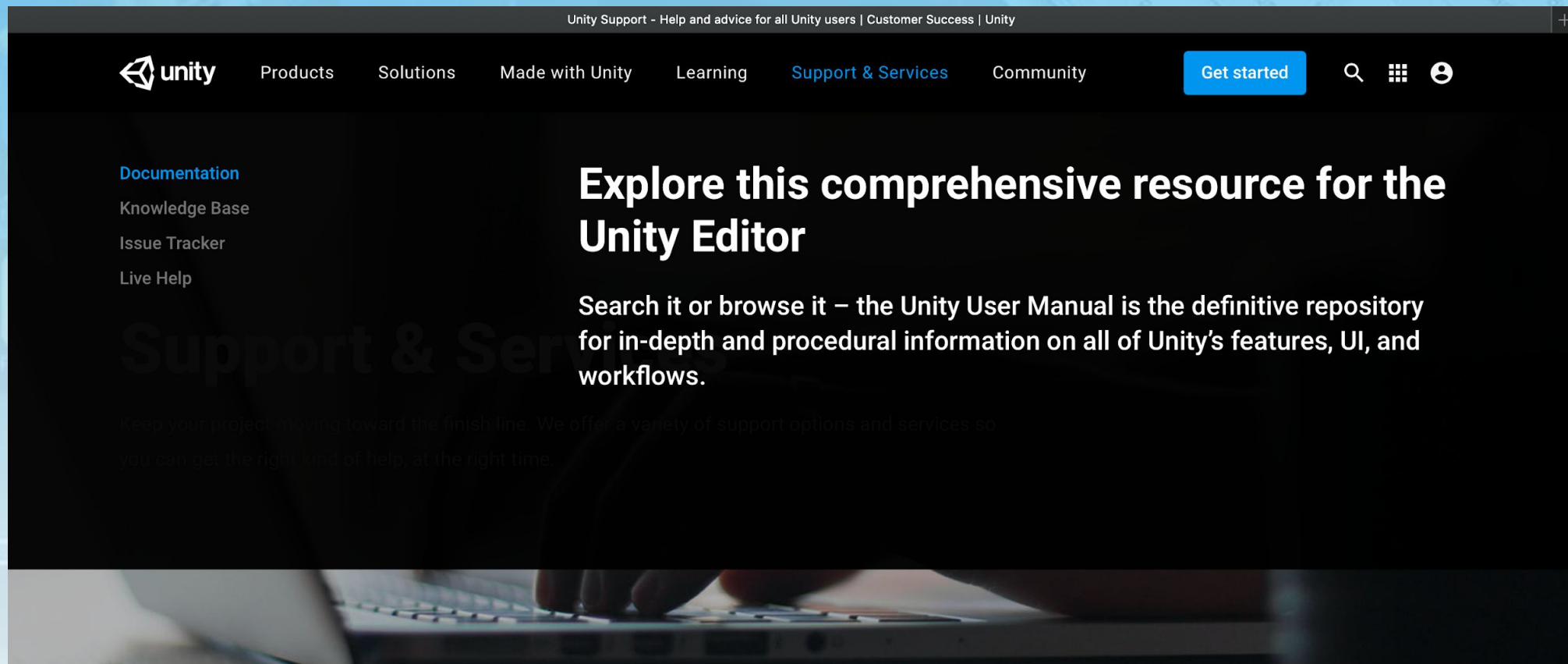
Documentación de Unity. MonoBehaviour.

- Para acceder a la ayuda de Unity, tenemos tres caminos posibles.
 - Una vez situados en la web de Unity, vamos a pulsar sobre “Support & Services”.



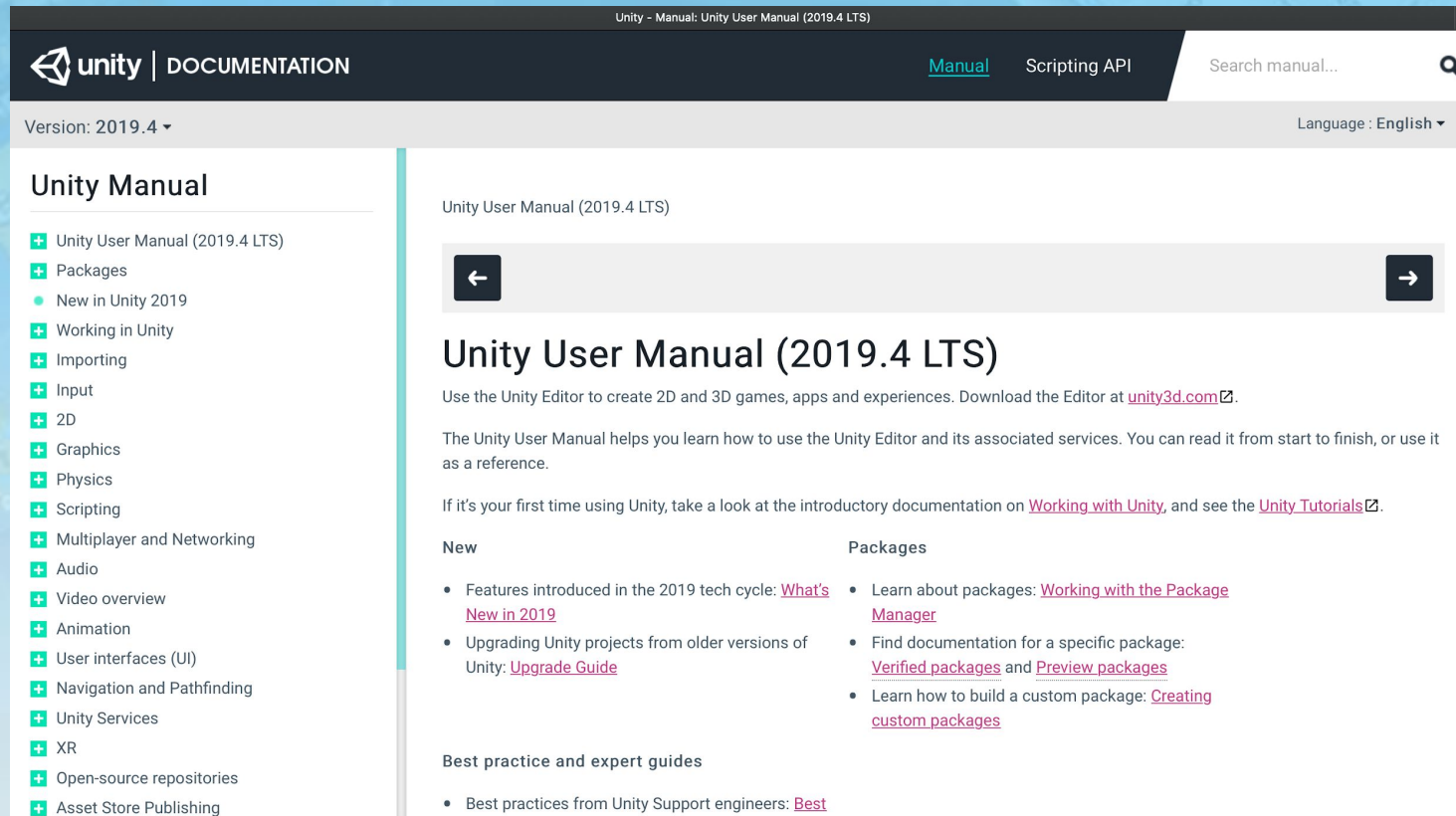
Documentación de Unity. MonoBehaviour.

- Para acceder a la ayuda de Unity, tenemos tres caminos posibles.
 - Ahora pulsamos sobre “Documentation”.



Documentación de Unity. MonoBehaviour.

- Para acceder a la ayuda de Unity, tenemos tres caminos posibles.
- Ahora pulsamos sobre “Documentation”.



The screenshot shows the Unity Documentation website for the 2019.4 LTS version. The page has a dark header with the Unity logo and 'DOCUMENTATION' text. Below the header, there's a navigation bar with 'Manual' and 'Scripting API' links, and a search bar. The main content area is divided into a left sidebar and a main panel. The sidebar lists various topics like 'Unity User Manual (2019.4 LTS)', 'Packages', 'New in Unity 2019', 'Working in Unity', 'Importing', 'Input', '2D', 'Graphics', 'Physics', 'Scripting', 'Multiplayer and Networking', 'Audio', 'Video overview', 'Animation', 'User interfaces (UI)', 'Navigation and Pathfinding', 'Unity Services', 'XR', 'Open-source repositories', and 'Asset Store Publishing'. The main panel displays the 'Unity User Manual (2019.4 LTS)' page, which includes a description of the manual, a list of 'New' features introduced in the 2019 tech cycle, a list of 'Packages' for learning and documentation, and a section for 'Best practice and expert guides'.

Unity - Manual: Unity User Manual (2019.4 LTS)

unity | DOCUMENTATION

Manual Scripting API Search manual...

Version: 2019.4 Language: English

Unity Manual

- Unity User Manual (2019.4 LTS)
- Packages
- New in Unity 2019
- Working in Unity
- Importing
- Input
- 2D
- Graphics
- Physics
- Scripting
- Multiplayer and Networking
- Audio
- Video overview
- Animation
- User interfaces (UI)
- Navigation and Pathfinding
- Unity Services
- XR
- Open-source repositories
- Asset Store Publishing

Unity User Manual (2019.4 LTS)

Use the Unity Editor to create 2D and 3D games, apps and experiences. Download the Editor at unity3d.com.

The Unity User Manual helps you learn how to use the Unity Editor and its associated services. You can read it from start to finish, or use it as a reference.

If it's your first time using Unity, take a look at the introductory documentation on [Working with Unity](#), and see the [Unity Tutorials](#).

New

- Features introduced in the 2019 tech cycle: [What's New in 2019](#)
- Upgrading Unity projects from older versions of Unity: [Upgrade Guide](#)

Packages

- Learn about packages: [Working with the Package Manager](#)
- Find documentation for a specific package: [Verified packages](#) and [Preview packages](#)
- Learn how to build a custom package: [Creating custom packages](#)

Best practice and expert guides

- Best practices from Unity Support engineers: [Best](#)

Documentación de Unity. MonoBehaviour.

- Como ejemplo, podemos buscar información sobre la clase “MonoBehaviour”, que es de la que vamos a heredar y con la que vamos a trabajar siempre. Para ello, vamos a seguir los siguientes pasos:
 - Vamos a seleccionar el bloque de “Scripting API”.

 **unity** | DOCUMENTATION

Manual

[Scripting API](#)

Search scripting...



- En la ventana que se nos muestra, en la parte izquierda, vamos a seleccionar la categoría de “UnityEngine”





 **unity** | DOCUMENTATION

Manual

[Scripting API](#)

Version: 2019.4

Scripting API

-  [UnityEngine](#)
-  [UnityEditor](#)
-  [Unity](#)
-  [Other](#)

MonoBehaviour

class in UnityEngine / Inherits from: [Behaviour](#) / Implemented in: [UnityEngine.CoreModule](#)

Description

MonoBehaviour is the base class from which every Unity script derives.

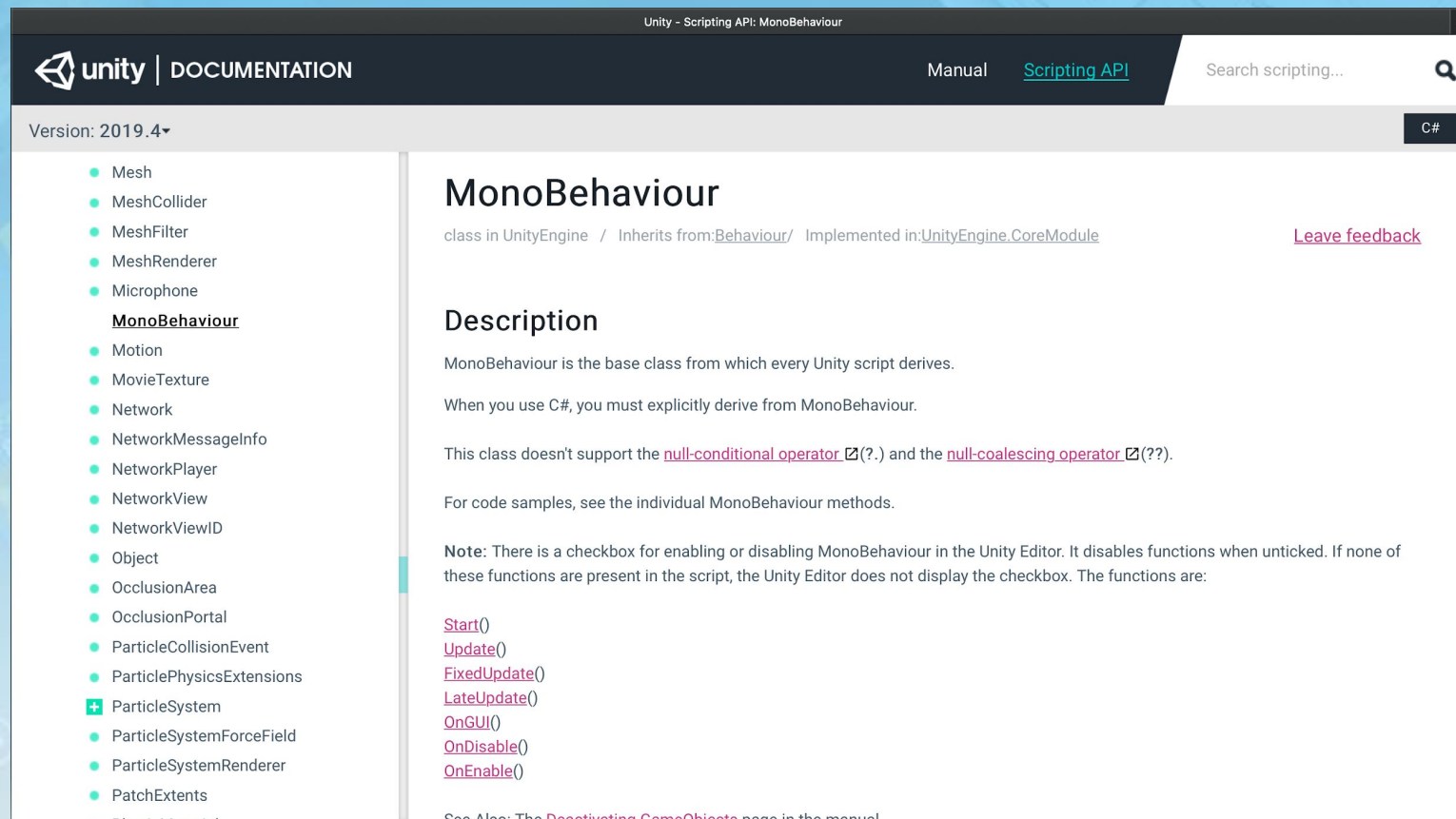
When you use C#, you must explicitly derive from MonoBehaviour.

This class doesn't support the [null-conditional operator](#) `?.` and the [null-coalescing operator](#) `??`.

For code samples, see the individual MonoBehaviour methods

Documentación de Unity. MonoBehaviour.

- Buscaremos la subcategoría de “Classes” y dentro “MonoBehaviour”



Unity - Scripting API: MonoBehaviour

unity | DOCUMENTATION

Manual [Scripting API](#) Search scripting...

Version: 2019.4

MonoBehaviour

class in UnityEngine / Inherits from: [Behaviour](#) / Implemented in: [UnityEngine.CoreModule](#) [Leave feedback](#)

Description

MonoBehaviour is the base class from which every Unity script derives.

When you use C#, you must explicitly derive from MonoBehaviour.

This class doesn't support the [null-conditional operator](#) `?.` and the [null-coalescing operator](#) `??`.

For code samples, see the individual MonoBehaviour methods.

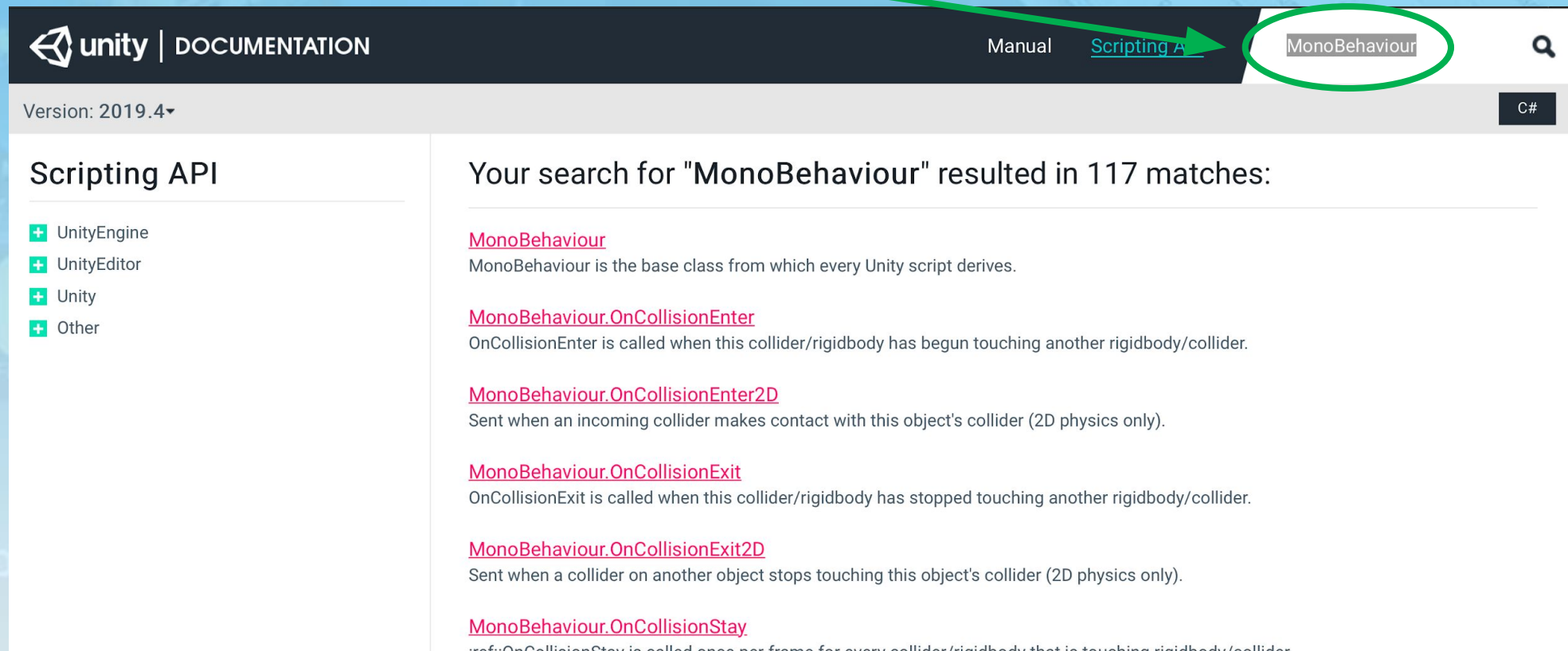
Note: There is a checkbox for enabling or disabling MonoBehaviour in the Unity Editor. It disables functions when unticked. If none of these functions are present in the script, the Unity Editor does not display the checkbox. The functions are:

- [Start\(\)](#)
- [Update\(\)](#)
- [FixedUpdate\(\)](#)
- [LateUpdate\(\)](#)
- [OnGUI\(\)](#)
- [OnDisable\(\)](#)
- [OnEnable\(\)](#)

See Also: The [Deactivating GameObjects](#) page in the manual

Documentación de Unity. MonoBehaviour.

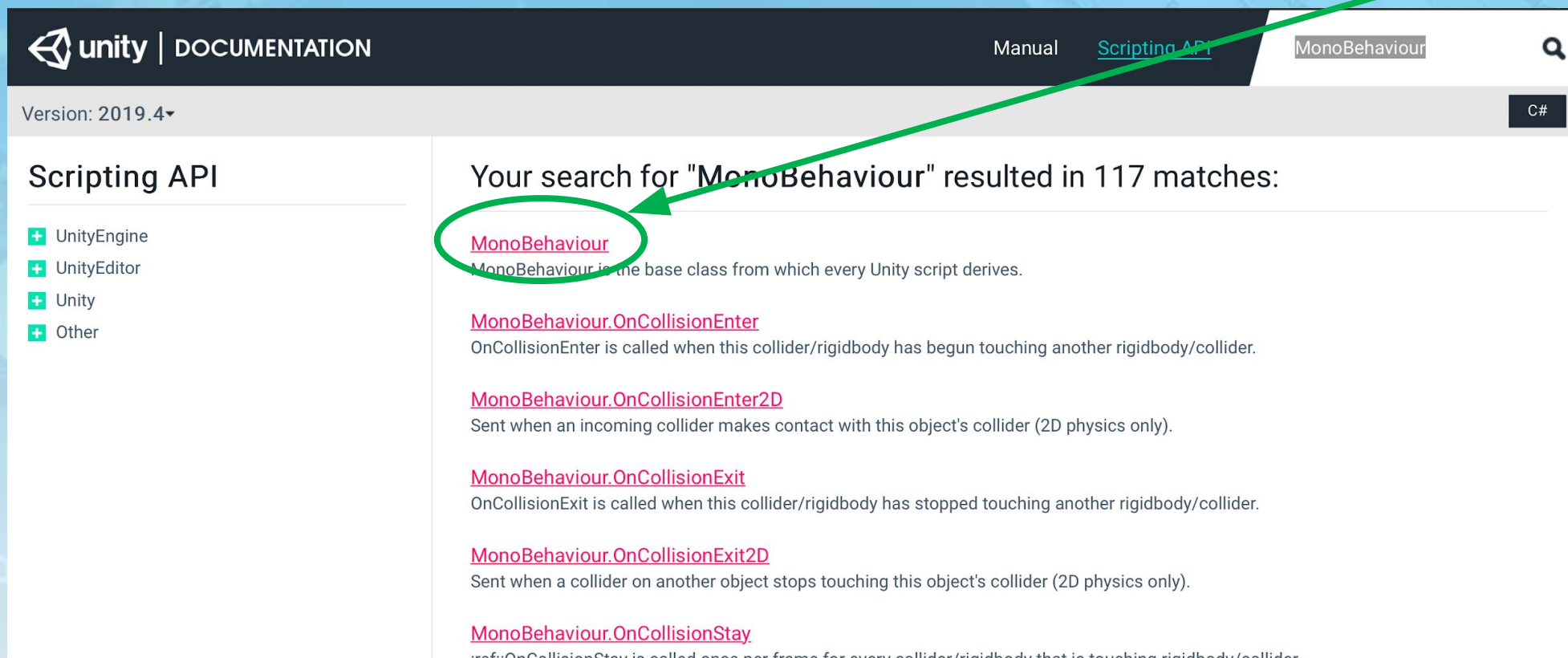
- Otra forma de hacerlo, sería escribir directamente “MonoBehaviour” en el Buscador de la documentación.



The screenshot shows the Unity Documentation website. The top navigation bar includes the Unity logo, 'DOCUMENTATION', and links for 'Manual' and 'Scripting API'. A search bar in the top right corner contains the text 'MonoBehaviour', which is circled in green. A green arrow points from the text 'MonoBehaviour' in the list item above to this search bar. Below the navigation bar, the page title is 'Scripting API' and the version is '2019.4'. The main content area displays the search results for 'MonoBehaviour', stating 'Your search for "MonoBehaviour" resulted in 117 matches:'. The results list several methods: [MonoBehaviour](#), [MonoBehaviour.OnCollisionEnter](#), [MonoBehaviour.OnCollisionEnter2D](#), [MonoBehaviour.OnCollisionExit](#), [MonoBehaviour.OnCollisionExit2D](#), and [MonoBehaviour.OnCollisionStay](#). Each method is followed by a brief description of its function.

Documentación de Unity. MonoBehaviour.

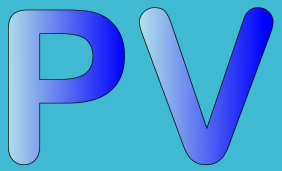
- Y seleccionar la opción que estamos buscando, en este caso, la primera.



The screenshot shows the Unity Documentation website. The search bar at the top right contains the text "MonoBehaviour". Below the search bar, the results are displayed. The first result, "MonoBehaviour", is circled in green. A green arrow points from the search bar to this result. The left sidebar shows the "Scripting API" section expanded, with "UnityEngine", "UnityEditor", "Unity", and "Other" listed. The main content area shows the search results for "MonoBehaviour", stating "Your search for 'MonoBehaviour' resulted in 117 matches:". Below this, the first result is "MonoBehaviour", which is the base class from which every Unity script derives. Other results listed include "MonoBehaviour.OnCollisionEnter", "MonoBehaviour.OnCollisionEnter2D", "MonoBehaviour.OnCollisionExit", "MonoBehaviour.OnCollisionExit2D", and "MonoBehaviour.OnCollisionStay".

Criterios de Evaluación.

- Los conceptos que se deben de poseer, tras finalizar esta Unidad de Trabajo (en adelante U.T.), son:
 - Utilizar de forma correcta la notación de programación.
 - Reconocer los conceptos de programación en código.
 - Utilizar de forma correcta el lenguaje de programación C#.



I.E.S PALOMERAS
VALLECAS

Repaso de POO e introducción al C#



I.E.S. VILLABLANCA



I.E.S PALOMERAS
VALLECAS



I.E.S. VILLABLANCA