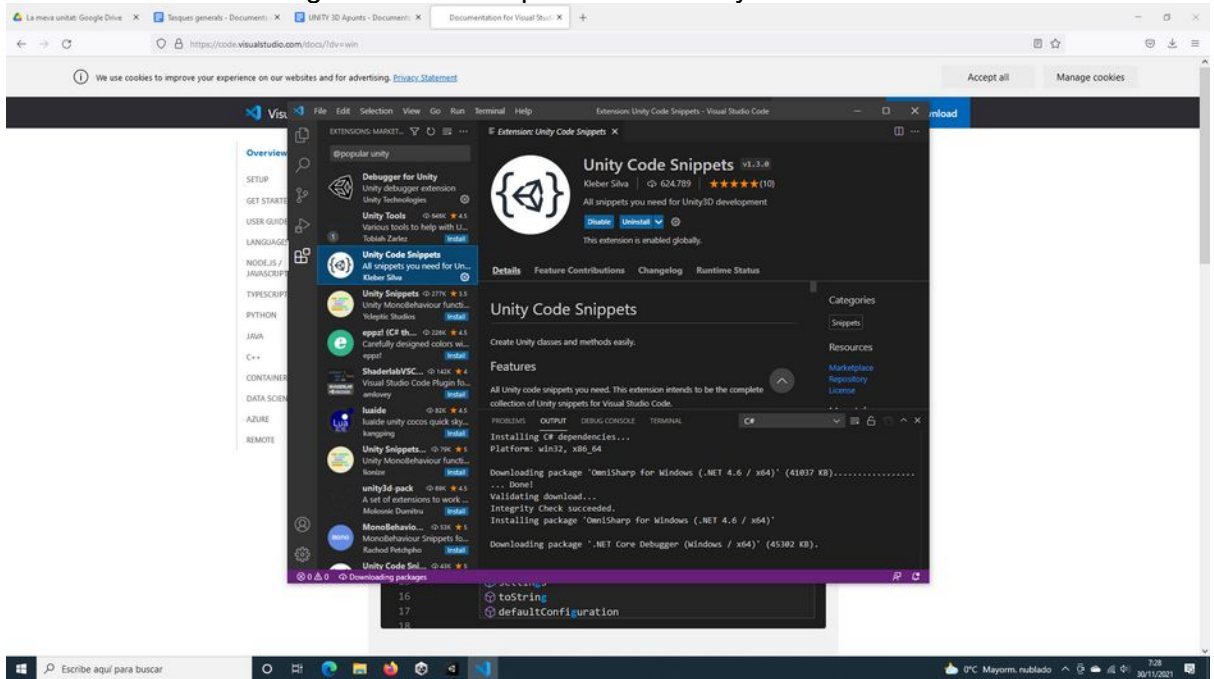


Apunts C# per a Unity

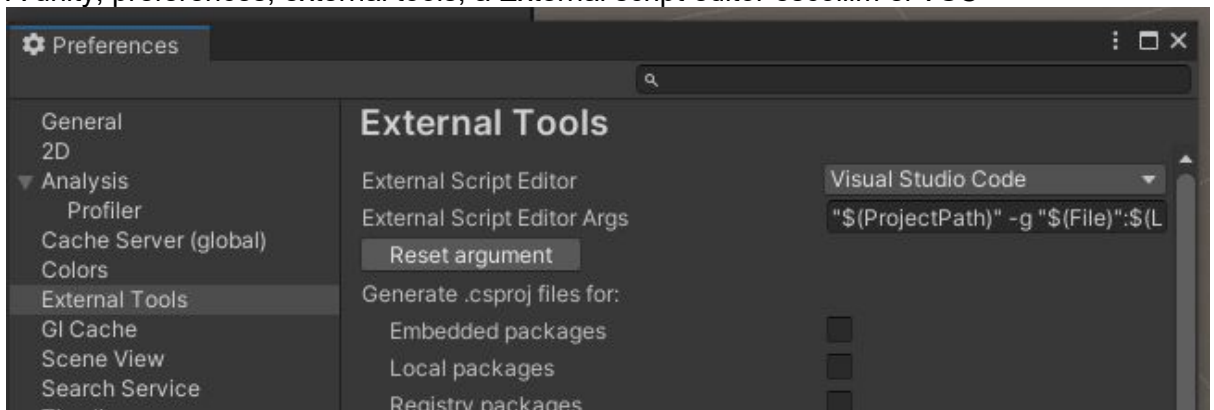
c# a partir del curs de youtube

<https://www.youtube.com/playlist?list=PLNEAWvYbJJ9kf04waxXg3ZsCFnt9EPPYI>

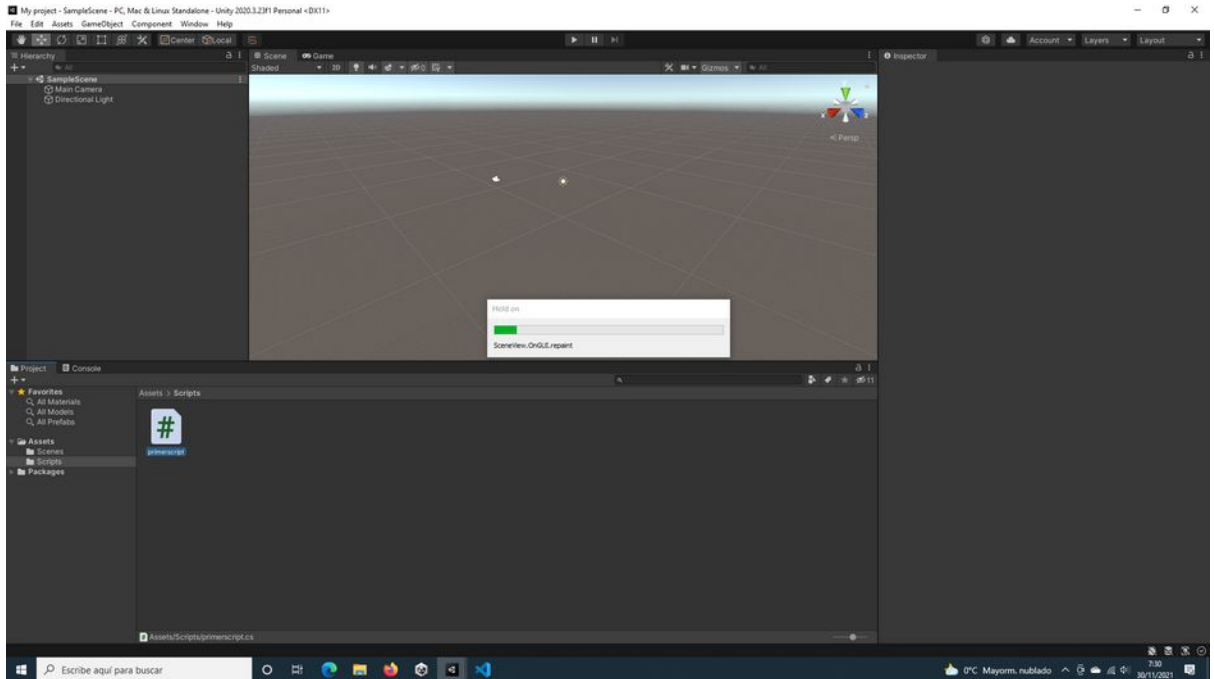
- 1) Utilitzar un IDE (farem servir visual studio code)
- 2) Instal·lem el VSC amb game desenvolupement with unity



- 3) A unity, preferences, external tools, a External script editor escollim el VSC



- 4) Crear un nou projecte i una carpeta que es digui scrips a dins d'Assets



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class primerscript : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

    }
}
```

Hi ha dos mètodes a la classe que hem creat, Start i Update

Utilitzem d'entrada només el mètode start que s'inicia al pulsar al play al videojoc, si hi ha un objecte amb el mètode start, executara aquest codi.

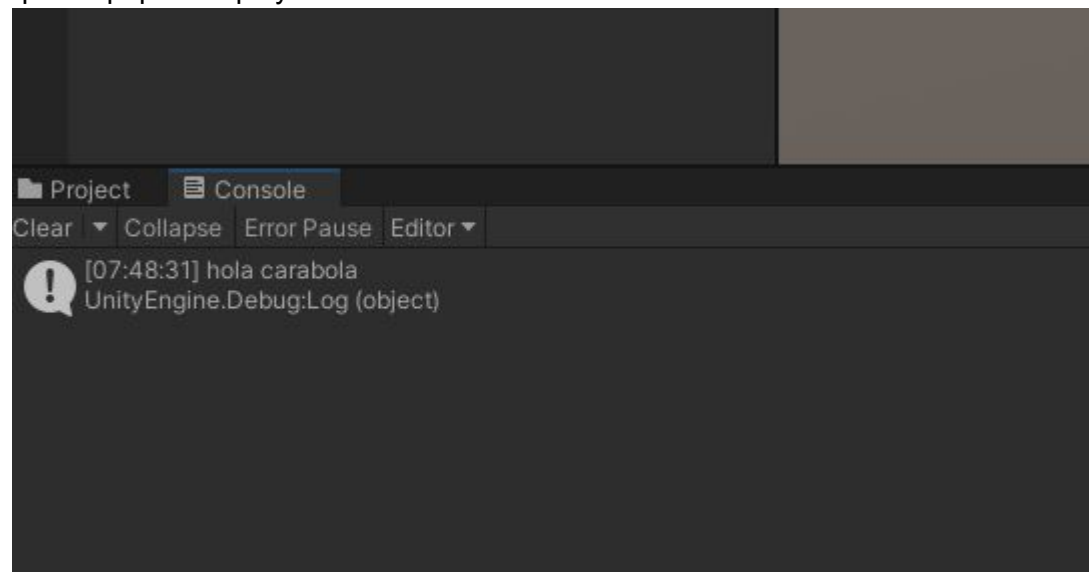
Fem el següent script, creep un empty object i li arrosseguem aquest script

```
using System.Collections;
using System.Collections.Generic;
```

```
using UnityEngine;

public class primerscript : MonoBehaviour
{
    void Start()
    {
        Debug.Log("hola carabola");
    }
}
```

quan li piquem a play surt a la consola



Tipus de variables

```
//tipus de variables
int senser=1;
float flotant=0.5f;
long llarg=1212312132;
double atopefloat=2.2354234354;
string cadena="feucho";
char lletra='a'; //alerta les cometes sencilles
bool bollean=true;

public class primerscript : MonoBehaviour
{
    void Start()
    {
        Debug.Log("hola carabola");
    }
}
```

Declaració de variables a la mateixa línia

```
int a=1,b=3;  
int resultat=0;
```

Les operacions les de sempre, el mòdul es fa amb %

a % b retorna el “resto” de a / b (per saber si un és múltip de l'altre)

a+=b és equivalent a a=a+b

Condicionals &&, ||, ! (and, or, not) if(condicio==valor){}

```
switch(codicio)  
{  
  case 1:  
    blabla bla  
    break;  
  case 2:  
    blabla  
    break;  
  default:  
    bla  
    break;  
}
```

```
while(condicion)  
{  
}
```

```
for (int x=1; x<10;x++){}
```

estructures de dades:

```
int[] array ={2,3,1,7,8};
```

```
for (i=0; i<array.Length;i++)  
{  
  Debug.log(array[i]);  
}
```

```
retorna= array[4];  
array[2]=21; //per a assignar
```

```
float[] arrayfloat = new float[5]; // definim una array sense valors  
arrayfloat[2]=3.4f;
```

```
string[] arraystring = {"pepe", "juan"};  
arraystring[0]="manolo";
```

Llistes és com una array pero es poden crear de tamany indefinit

```
List<int> numerossencers= new List<int>();
```

```
List<float> numerosf= new List<float>();
```

```
List<string> liststr= new List<string>();
```

```
numerossencers.Add(12);
```

```
numerossencers.Add(2);
```

```
numerossencers.Add(1);
```

```
Debug.Log(numerossencer.Count); //quants valors tenim
```

```
//afegir un numero a un lloc especific
```

```
numerossencers.Insert(0,13455);
```

```
//posa el numero 12455 a la posició 0,
```

```
numerossencers.Remove(13455); //borrar la posicio que te aquest valor
```

```
numerossencers.RemoveAt(0); //borra el valor de la posició 0
```

```
foreach ( int numero in numerossencers)
```

```
{
```

```
    Debug.Log (numero);
```

```
}
```

```
// recorre la llista
```

```
string frase ="hola que tal pascual";
```

```
foreach (char c in frase)
```

```
{
```

```
    Debug.Log (c);
```

```
}
```

```
//recòrrer la string
```

```
int [] numeros ={3,5,1,2,-4}
```

```
foreach (int p in numeros)
```

```
{
```

```
    Debug.Log(p);
```

```
}
```

METODES I FUNCIONS A UNITY

Per defecte quan es crea un script apareixen:

Start() Executa quan es crea

Update() Executa a cada frame

FixetUpdate() Calcula les físiques en un frame ????, després de Update

LateUpdate() S'executa després de Update i FixetUpdate

OnDissable() S'executa quan es desactiva

OnEnable() S'executa quan s'activa

Crear un mètode

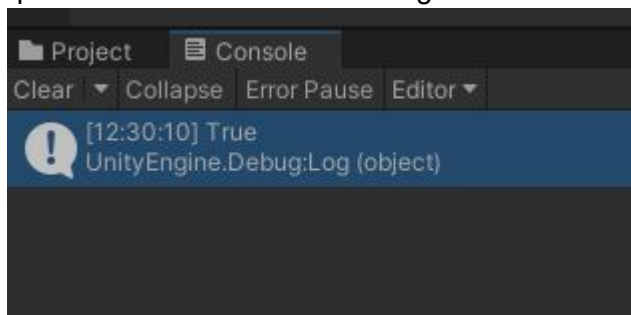
```
public void suma( float a, float b)
{
    suma a+b;
}
```

```
private string resposta (int a)
{
    if (a==1) return "hola"; else return "adeu";
}
```

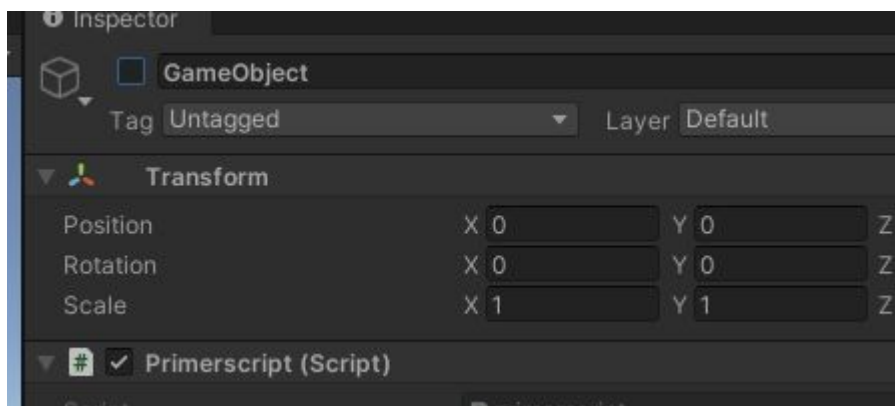
Exemple d'us:

```
public class primerscript : MonoBehaviour
{
    void Start()
    {
        Debug.Log(gameObject.activeSelf);
    }
}
```

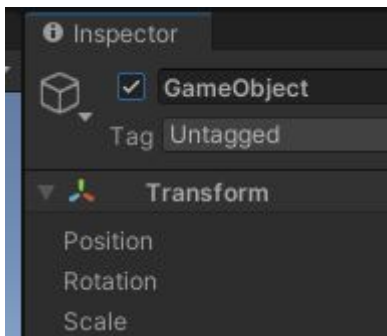
Això ho afegim a un GameObject empty
quan executa a la finestra debug



Si desactivo



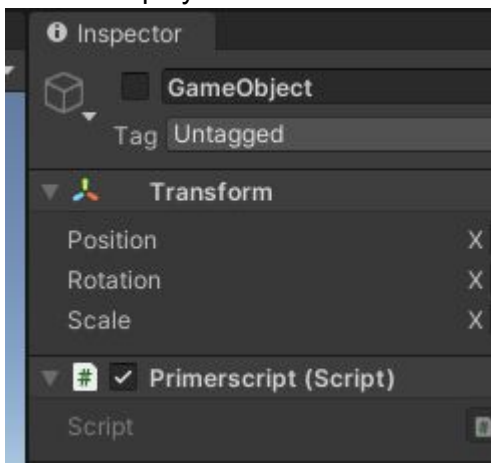
Quan executa debug
no surt res perque com està desactivat no s'executa l'script
Tornem a activar



Canviem el codi

```
public class primerscript : MonoBehaviour
{
    void Start()
    {
        gameObject.SetActive(false);
    }
}
```

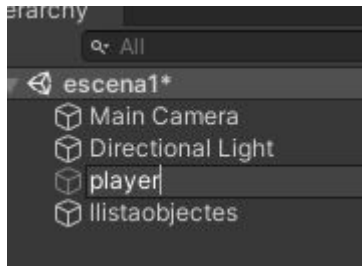
Quan fem play



Ara posem un component de fisica amb gravetat

```
public class primerscript : MonoBehaviour
{
    void Start()
    {
        //afegim un component de física amb gravetat
        gameObject.AddComponent<Rigidbody>();
    }
}
```

i desde el Ui canviem el nom del gameobject a player



Podem preguntar per la consola quin nom es el del objecte

```
void Start()
{
    //afegim un component de física amb gravetat
    gameObject.AddComponent<Rigidbody>();
    Debug.Log(gameObject.name);
}
}
```

Esborrem i definim una posició

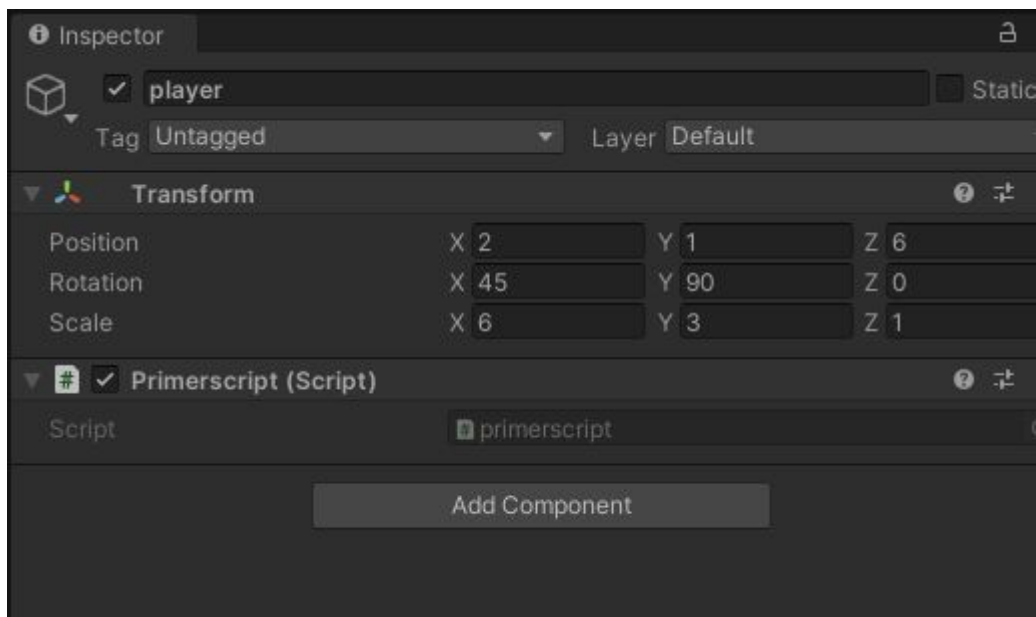
```
void Start()
{

    //ara assignem una posició
    gameObject.transform.position = new Vector3(2,1,6);
}
}
```

Canviem posició, rotació i escala

```
void Start()
{

    //ara assignem una posició
    gameObject.transform.position = new Vector3(2,1,6);
    //aquesta instrucció es idèntica
    transform.position = new Vector3(2,1,6);
    // també podem accedir a veure la posició
    Debug.Log(transform.position);
    //transformar la rotació
    transform.rotation=Quaternion.Euler(45,90,0);
    //transformar la escala
    transform.localScale=new Vector3(6,3,1);
}
}
```

Es pot modificar només algun dels valors, conservant els altres, de tal manera

```
//ara assignem una posició conservant la x  
transform.position = new Vector3(transform.position.x,1,6);
```

Corutines

Son procediments que poden incloure esperes de x segons al codi es declaren amb IEnumerator CoroutineName() i necessiten una línia:

```
yield return new WaitForSeconds(x);
```

I es crida amb StartCoroutine ("nomdecorutine");

Exemple:

```
public class primerscript : MonoBehaviour  
{  
    void Start()  
    {  
        StartCoroutine("Coroutine");  
    }  
  
    IEnumerator Coroutine()  
    {  
        Debug.Log ("ara mesperare un segon");  
        yield return new WaitForSeconds(1);  
        Debug.Log ("YA");  
    }  
}
```

```
}  
}
```

Ara imaginem que volem que cada segon faci una cosa, doncs fàcil, la fem recursiva tornant a cridar-se a sí mateixa

```
public class primerscript : MonoBehaviour  
{  
    void Start()  
    {  
        StartCoroutine("Coroutine");  
    }  
    IEnumerator Coroutine()  
    {  
        Debug.Log ("ara mesperare un segon");  
        yield return new WaitForSeconds(1);  
        Debug.Log ("YA");  
        StartCoroutine("Coroutine");  
    }  
}
```

Per pasar un paràmetre ho fem així

```
public class primerscript : MonoBehaviour  
{  
    void Start()  
    {  
        StartCoroutine("Coroutine",2);  
    }  
    IEnumerator Coroutine(float time)  
    {  
        Debug.Log ("ara mesperare el que digui el paràmetre");  
        yield return new WaitForSeconds(time);  
        Debug.Log ("YA");  
        StartCoroutine("Coroutine",time);  
    }  
}
```

Mètode Invoke per a temporitzar una acció

Exemple: enemic que dispara cada 3 segons:

```
public class primerscript : MonoBehaviour
```

```

{
    void Start()
    {
        Invoke("Disparar",3);
    }

    void Disparar()
    {
        //dispara cada tres segons
        Debug.Log ("ara dispararia");
    }
}

```

Mètode Destroy, per a eliminar objectes que hi ha a la escena

```

public class primerscript : MonoBehaviour
{
    void Start()
    {
        Invoke("Disparar",3);
        Destroy(gameObject,1);
    }
    void Disparar()
    {
        //dispara cada tres segons
        Debug.Log ("ara dispararia");
    }
}

```

En aquest cas tal com passa un segon es destrueix l'objecte i ja no s'executa el Disparar

Com guardar informació; només podem guardar integer, floats i strings amb PlayerPrefs

```

void Start()
{
    //guarda un int assignat a una variable
    int contenidoint=7;
    int retorna;
    PlayerPrefs.SetInt ("valorint",contenidoint);
    retorna =PlayerPrefs.GetInt("valorint");
    Debug.Log (retorna);
    PlayerPrefs.SetString ("valorstring","PEPE NO TE MOJES");
    string retornastring =PlayerPrefs.GetString("valorstring");
    Debug.Log (retornastring);
}

```

