































Character.cs

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

/*
Clase genérica para los personajes del Juego.
*/
public abstract class Character : MonoBehaviour
{
    public HitPoints hitPoints; //Puntos actuales de jugador
    public float maxHitPoints; //Máximos puntos a obtener
}
```

Player.cs

```
healthBar = Instantiate(healthBarPrefab); //Instanciar HealthBar
healthBar.character = this; //Referencia del Player en HealthBar
}

Mensaje de Unity | 0 referencias
public void OnTriggerEnter2D(Collider2D collision)
{
    if (collision.gameObject.CompareTag("CanBePickedUp"))
    {
        Item hitObject = collision.gameObject.GetComponent<Consumable>().item;
        if (hitObject != null)
        {
            Debug.Log("Nombre: " + hitObject.gameObject.name);
            bool shouldDisappear = false;
            switch (hitObject.itemType)
            {
                case Item.ItemType.COIN: //Moneda
                    shouldDisappear = true;
                    break;
                case Item.ItemType.HEALTH://Barra de Salud
                    Debug.Log("Cantidad a Incrementar: " + hitObject.quantity);
                    shouldDisappear = AdjustHitPoints(hitObject.quantity);
                    break;
            }
            if (shouldDisappear)
            {
                collision.gameObject.SetActive(false); //Desaparecer
            }
        }
    }
}

Referencia
private bool AdjustHitPoints(int amount)
{
    if (hitPoints.value < maxHitPoints) // no se puede exceder el máximo de puntos
    {
        hitPoints.value += amount;
        print("Ajustando Puntos: " + amount + ". Nuevo Valor: " + hitPoints.value);
        return true; //Fue modificado
    }
}
```

Screenshot of Visual Studio showing the `HealthBar.cs` script. The code defines a `MonoBehaviour` class named `HealthBar` with a `Player` reference and a `Image` component named `meterImage`. It contains a `Start` method that initializes hit points to 0 and a `Update` method that updates the meter image based on the player's current hit points.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI; //Componentes Interfaz Gráfica

public class HealthBar : MonoBehaviour
{
    [HideInInspector]
    public Player character; //Referencia al jugador
    public Image meterImage; //Medidor Meter de la salud
    public Text hpText; //Texto en barra de salud

    void Start()
    {
        character.hitPoints.value = 0;
    }

    void Update()
    {
        if (character != null)
        {
            //Modifica barra de salud
            meterImage.fillAmount = character.hitPoints.value / character.maxHitPoints;
            //Texto a mostrar
            hpText.text = "HP:" + (meterImage.fillAmount * 100);
        }
    }
}
```

Screenshot of Visual Studio showing the `Player.cs` script. The code defines a `Character` class with a `HealthBar` component. It overrides the `Start` method to instantiate the `HealthBar` and set its character reference. It also implements the `OnTriggerEnter2D` event to handle item pickup logic.

```
using UnityEngine;

public class Player : Character
{
    public HealthBar healthBarPrefab; //Referencia HealthBar Prefab
    private HealthBar healthBar; //Copia de referencia de HealthBar Prefab

    void Start()
    {
        healthBar = Instantiate(healthBarPrefab); //Instanciar HealthBar
        healthBar.character = this; //Referencia del Player en HealthBar
    }

    public void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.CompareTag("CanBePickedUp"))
        {
            Item hitObject = collision.gameObject.GetComponent<Consumable>().item;
            if (hitObject != null)
            {
                Debug.Log("Nombre: " + hitObject.objectName);
                bool shouldDisappear = false;
                switch (hitObject.itemType)
                {
                    case Item.ItemType.COIN: //Moneda
                        shouldDisappear = true;
                        break;
                    case Item.ItemType.HEALTH://Barra de Salud
                        Debug.Log("Cantidad a Incrementar: " + hitObject.quantity);
                        shouldDisappear = AdjustHitPoints(hitObject.quantity);
                        break;
                }
                if (shouldDisappear)
                {
                    collision.gameObject.SetActive(false); //Desaparecer
                }
            }
        }
    }
}
```



