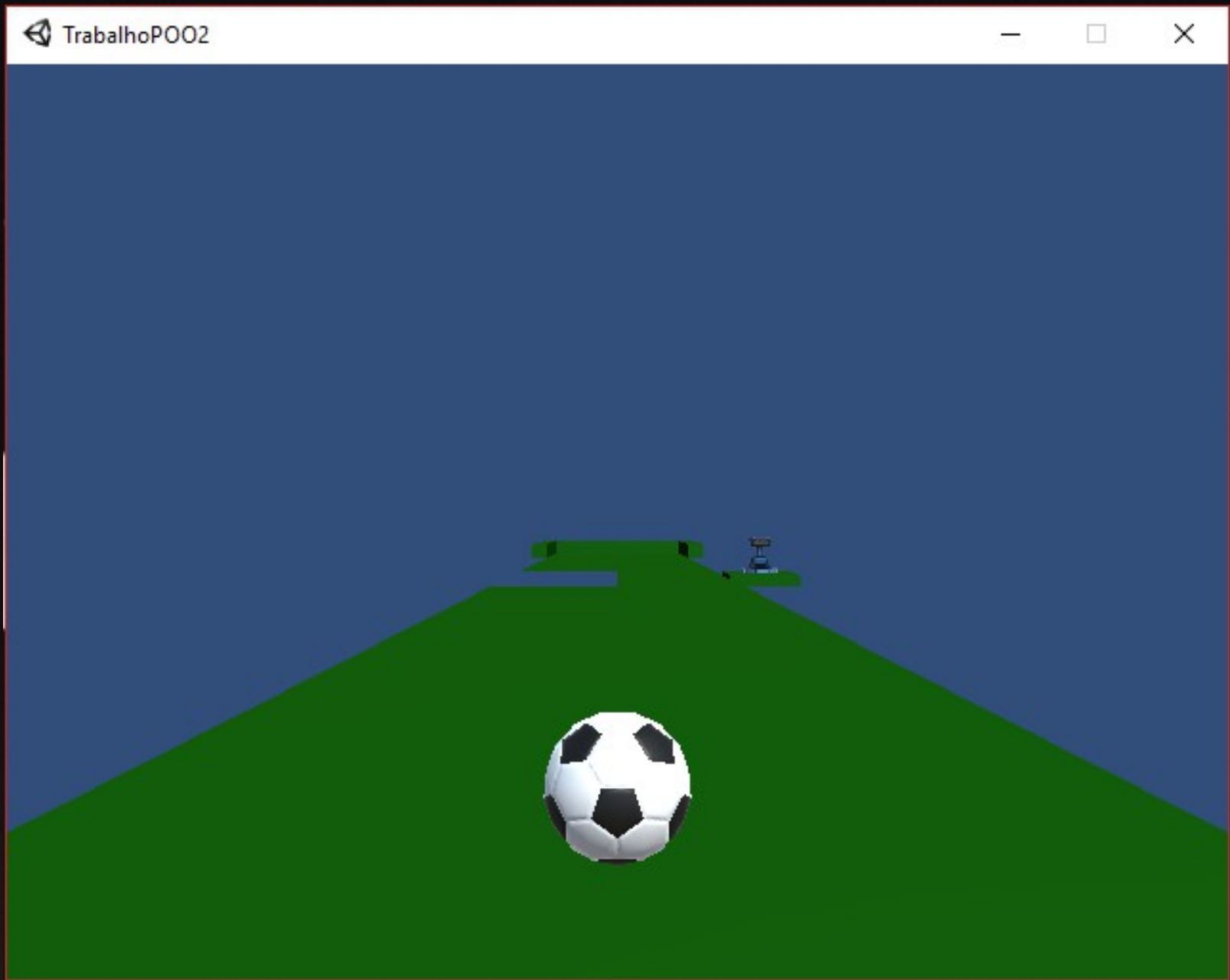


Projeto POO2 - Jogo



PLAY
OPTIONS
QUIT

A soccer ball is positioned at the bottom of the text, partially obscured by the word 'QUIT'. The background is split horizontally into a blue top half and a green bottom half.



LOSE

RESTART
MENU

QUIT



LOSE

**RESTART
MENU**

QUIT



YOU WIN

***RESTART
MENU***

QUIT



Padrão Fabrica Abstrata

```
public class TurrentFactory : MonoBehaviour {  
    public GameObject turrent1;  
    public GameObject turrent2;  
    public GameObject turrent3;  
    public GameObject turrent4;  
    public GameObject turrent5;  
  
    bool x;  
    int i;  
    // Use this for initialization  
    void Start () {  
        x=true;  
        i=1;  
        i=Random.Range(1,5);  
    }  
  
    // Update is called once per frame  
    void Update () {  
        if (x){  
            {  
                switch (i)  
                {  
                    case 1:  
                        Instantiate(turrent1, this.transform.position, this.transform.rotation);  
                        x = false;  
                        break;  
                    case 2:  
                        Instantiate(turrent2, this.transform.position, this.transform.rotation);  
                        x = false;  
                        break;  
                    case 3:  
                        Instantiate(turrent3, this.transform.position, this.transform.rotation);  
                        x = false;  
                        break;  
                    case 4:  
                        Instantiate(turrent4, this.transform.position, this.transform.rotation);  
                        x = false;  
                        break;  
                    case 5:  
                        Instantiate(turrent5, this.transform.position, this.transform.rotation);  
                        x = false;  
                        break;  
                }  
            }  
        }  
    }  
}
```


Padrão Protótipo

```
public class TurrentFactory : MonoBehaviour {  
    public GameObject turrent1;  
    public GameObject turrent2;  
    public GameObject turrent3;  
    public GameObject turrent4;  
    public GameObject turrent5;  
  
    bool x;  
    int i;  
    // Use this for initialization  
    void Start () {  
        x=true;  
        i=1;  
        i=Random.Range(1,5);  
    }  
  
    // Update is called once per frame  
    void Update () {  
        if (x){  
            {  
                switch (i)  
                {  
                    case 1:  
                        Instantiate(turrent1, this.transform.position, this.transform.rotation);  
                        x = false;  
                        break;  
                    case 2:  
                        Instantiate(turrent2, this.transform.position, this.transform.rotation);  
                        x = false;  
                        break;  
                    case 3:  
                        Instantiate(turrent3, this.transform.position, this.transform.rotation);  
                        x = false;  
                        break;  
                    case 4:  
                        Instantiate(turrent4, this.transform.position, this.transform.rotation);  
                        x = false;  
                        break;  
                    case 5:  
                        Instantiate(turrent5, this.transform.position, this.transform.rotation);  
                        x = false;  
                        break;  
                }  
            }  
        }  
    }  
}
```


Padrão Singleton

```
using UnityEngine;
using System.Collections;

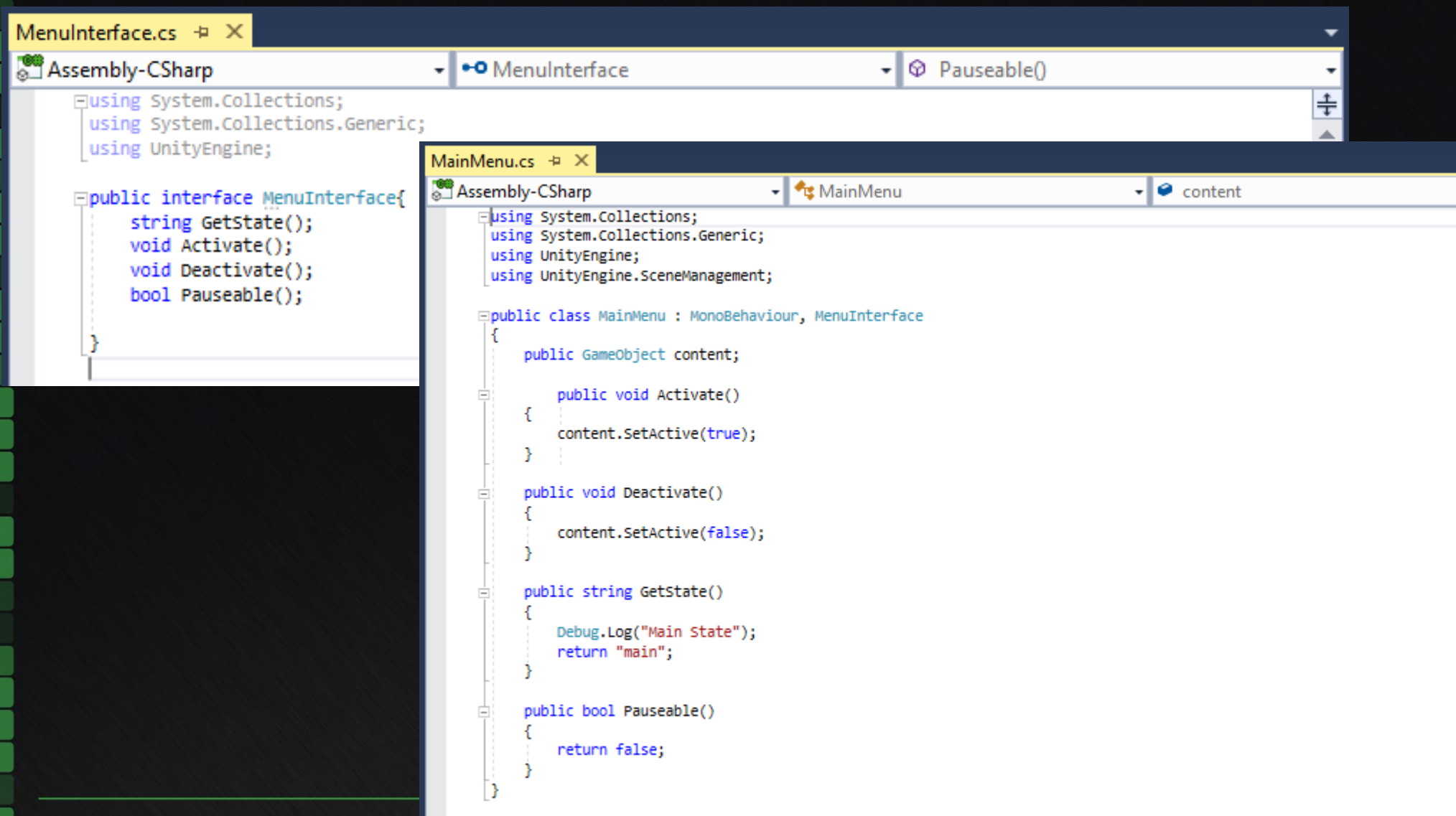
public class GameManager : MonoBehaviour {

    //Used for singleton
    public static GameManager GM;

    //Create Keycodes that will be associated with each of our commands.
    //These can be accessed by any other script in our game
    public KeyCode jump {get; set;}
    public KeyCode forward {get; set;}
    public KeyCode backward {get; set;}
    public KeyCode left {get; set;}
    public KeyCode right {get; set;}

    void Awake()
    {
        //Singleton pattern
        if(GM == null)
        {
            DontDestroyOnLoad(gameObject);
            GM = this;
        }
        else if(GM != this)
        {
            Destroy(gameObject);
        }
        /*Assign each keycode when the game starts.
        * Loads data from PlayerPrefs so if a user quits the game,
        * their bindings are loaded next time. Default values
        * are assigned to each KeyCode via the second parameter
        * of the GetString() function
        */
        jump = (KeyCode) System.Enum.Parse(typeof(KeyCode), PlayerPrefs.GetString("jumpKey", "Space"));
        forward = (KeyCode) System.Enum.Parse(typeof(KeyCode), PlayerPrefs.GetString("forwardKey", "W"));
        backward = (KeyCode) System.Enum.Parse(typeof(KeyCode), PlayerPrefs.GetString("backwardKey", "S"));
        left = (KeyCode) System.Enum.Parse(typeof(KeyCode), PlayerPrefs.GetString("leftKey", "A"));
        right = (KeyCode) System.Enum.Parse(typeof(KeyCode), PlayerPrefs.GetString("rightKey", "D"));
    }
}
```

Padrão State



The screenshot displays two C# files in Visual Studio. The left file, `MenuInterface.cs`, defines an interface `MenuInterface` with methods `GetState()`, `Activate()`, `Deactivate()`, and `Pauseable()`. The right file, `MainMenu.cs`, implements the `MainMenu` class, which inherits from `MonoBehaviour` and implements the `MenuInterface`. The `MainMenu` class has a `GameObject` property `content` and implements the methods to activate/deactivate the content and return the state.

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

public interface MenuInterface{
    string GetState();
    void Activate();
    void Deactivate();
    bool Pauseable();
}
```

```
MainMenu.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class MainMenu : MonoBehaviour, MenuInterface
{
    public GameObject content;

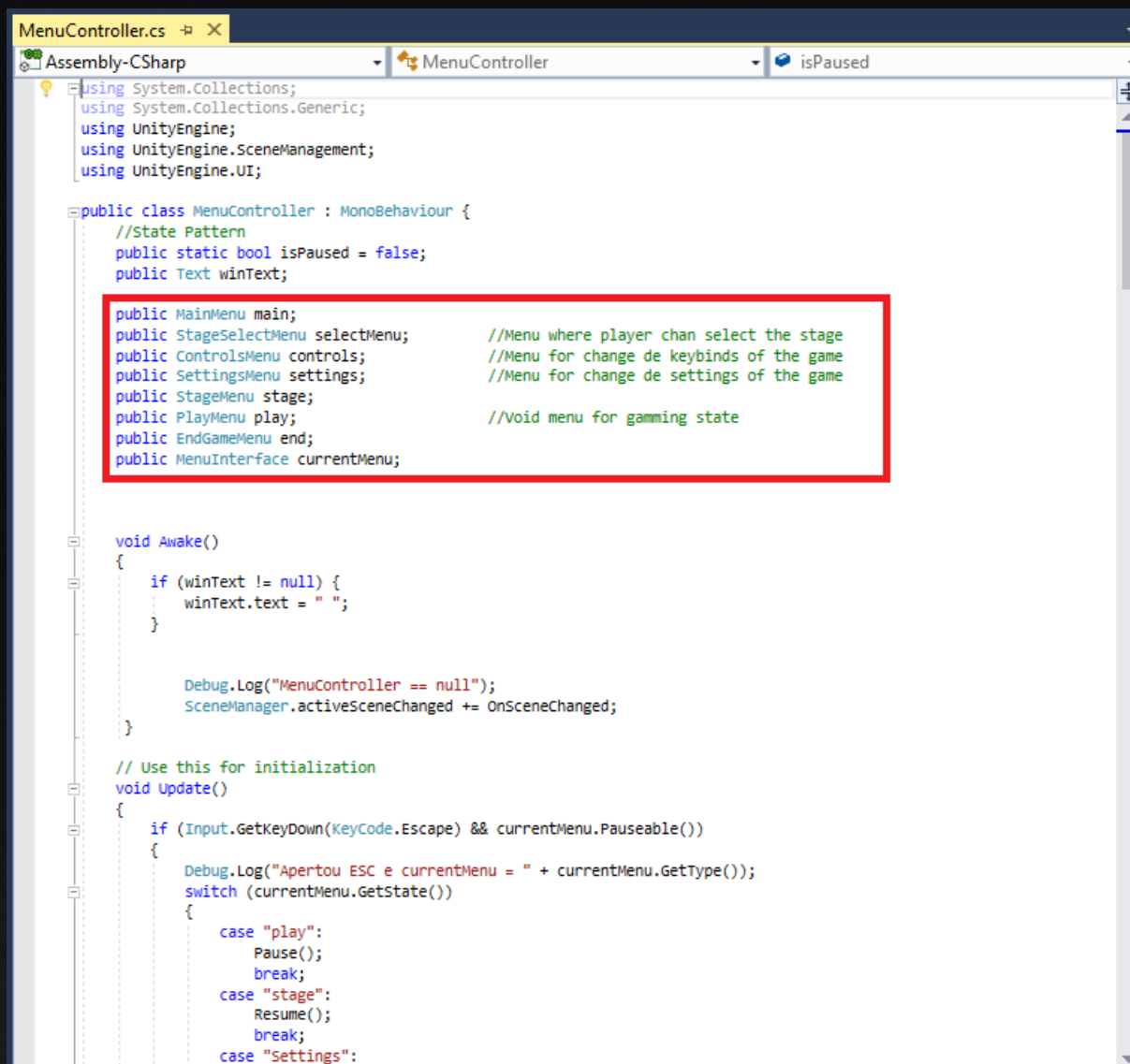
    public void Activate()
    {
        content.SetActive(true);
    }

    public void Deactivate()
    {
        content.SetActive(false);
    }

    public string GetState()
    {
        Debug.Log("Main State");
        return "main";
    }

    public bool Pauseable()
    {
        return false;
    }
}
```

Padrão State



```
MenuController.cs
Assembly-CSharp
MenuController
isPaused

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
using UnityEngine.UI;

public class MenuController : MonoBehaviour {
    //State Pattern
    public static bool isPaused = false;
    public Text winText;

    public MainMenu main;
    public StageSelectMenu selectMenu; //Menu where player can select the stage
    public ControlsMenu controls; //Menu for change de keybinds of the game
    public SettingsMenu settings; //Menu for change de settings of the game
    public StageMenu stage;
    public PlayMenu play; //Void menu for gaming state
    public EndGameMenu end;
    public MenuInterface currentMenu;

    void Awake()
    {
        if (winText != null) {
            winText.text = " ";
        }

        Debug.Log("MenuController == null");
        SceneManager.activeSceneChanged += OnSceneChanged;
    }

    // Use this for initialization
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.Escape) && currentMenu.Pauseable())
        {
            Debug.Log("Apertou ESC e currentMenu = " + currentMenu.GetType());
            switch (currentMenu.GetState())
            {
                case "play":
                    Pause();
                    break;
                case "stage":
                    Resume();
                    break;
                case "Settings":
            }
        }
    }
}
```


Padrão State

```
public void getScripts()
{
    main = GameObject.Find("MainMenu").GetComponent<MainMenu>();
    selectMenu = GameObject.Find("SelectStageMenu").GetComponent<StageSelectMenu>();
    controls = GameObject.Find("ControlsMenu").GetComponent<ControlsMenu>();
    settings = GameObject.Find("SettingsMenu").GetComponent<SettingsMenu>();
    stage = GameObject.Find("StageMenu").GetComponent<StageMenu>();
    play = GameObject.Find("GameManager").GetComponent<PlayMenu>();
    end = GameObject.Find("EndGame").GetComponent<EndGameMenu>();
}
```

```
public void setMenu(string menu)
{
    main.Deactivate();
    selectMenu.Deactivate();
    controls.Deactivate();
    settings.Deactivate();
    stage.Deactivate(); ;
    end.Deactivate(); ;
    play.Deactivate();
    switch (menu)
    {
        case "main":
            Debug.Log("set main");
            currentMenu = main;
            currentMenu.Activate();
            break;
        case "selectMenu":
            Debug.Log("set selectMenu");
            currentMenu = selectMenu;
            currentMenu.Activate();
            break;
        case "controls":
            Debug.Log("set controls");
            currentMenu = controls;
            currentMenu.Activate();
            break;
        case "settings":
            Debug.Log("set settings");
            currentMenu = settings;
            currentMenu.Activate();
            break;
        case "stage":
            Debug.Log("set stage");
            currentMenu = stage;
            currentMenu.Activate();
            break;
        case "play":
            Debug.Log("set play");
            currentMenu = play;
            currentMenu.Activate();
            break;
        case "end":
            Debug.Log("set end");
            currentMenu = end;
            currentMenu.Activate();
            break;
    }
}
```


Padrão Observador

```
void OnSceneChanged(Scene previousScene, Scene changedScene)
{
    Debug.LogError("OnSceneChanged changedScene = " + changedScene.name);
    switch (changedScene.name)
    {
        case "Fase1":
            isPaused = false;
            Debug.Log("play");
            getScripts();
            setMenu("play");
            Debug.Log(currentMenu.GetState());
            break;
        case "FaseMain":
            isPaused = false;
            Debug.Log("main");
            getScripts();
            setMenu("main");
            Debug.Log(currentMenu.GetState());
            break;
        case "Fase3":
            isPaused = false;
            Debug.Log("play");
            getScripts();
            setMenu("play");
            Debug.Log(currentMenu.GetState());
            break;
    }
}

void Awake()
{
    if (winText != null) {
        winText.text = " ";
    }

    Debug.Log("MenuController == null");
    SceneManager.activeSceneChanged += OnSceneChanged;
}
```