Data Science : Save and Load com JSON - Arrays

Prof. Ovídio Francisco

Plano de aula

A notação JSON

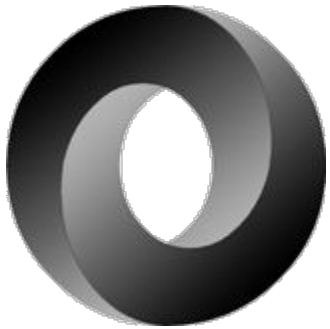
A classe JsonUtility

Métodos para salvar e carregar

Projeto de exemplo

Exercícios

Tipos de dados em JSON



Tipos de dados em JSON

Os dados em JSON podem ser representados respeitando os seguintes tipos de dados:

Tipo	Exemplo
String	`{ "PlayerName" : "Nasrudin" }`
Numbers	`{ "Score" : 1024 }`
Boolean	`{ "IsJumping" : true }`
Arrays	`{ "Items" : [ "Sword", "Shield", "spear" ] }`
Object	`{ "Enemy" : { "Class" : "Clown", "health" : 0.85 } }`
null	`{ "Awards" : null }`

Lista de objetos Unity para Array em JSON

Exemplo com array de objetos

Naturalmente, podemos ter arrays de quaisquer tipos válidos, incluindo objetos e outros arrays.


```
{
    "score": 5,
    "timeLeft": 87.564453125,
    "asteroidsData": [
        {
            "position": {
                "x": 3.64335036277771,
                "y": -4.932304382324219
            }
        },
        {
            "position": {
                "x": -3.505880355834961,
                "y": 3.948585033416748
            }
        },
        {
            "position": {
                "x": -6.235532760620117,
                "y": -3.131162643432617
            }
        }
    ]
}
```

Salvando na prática

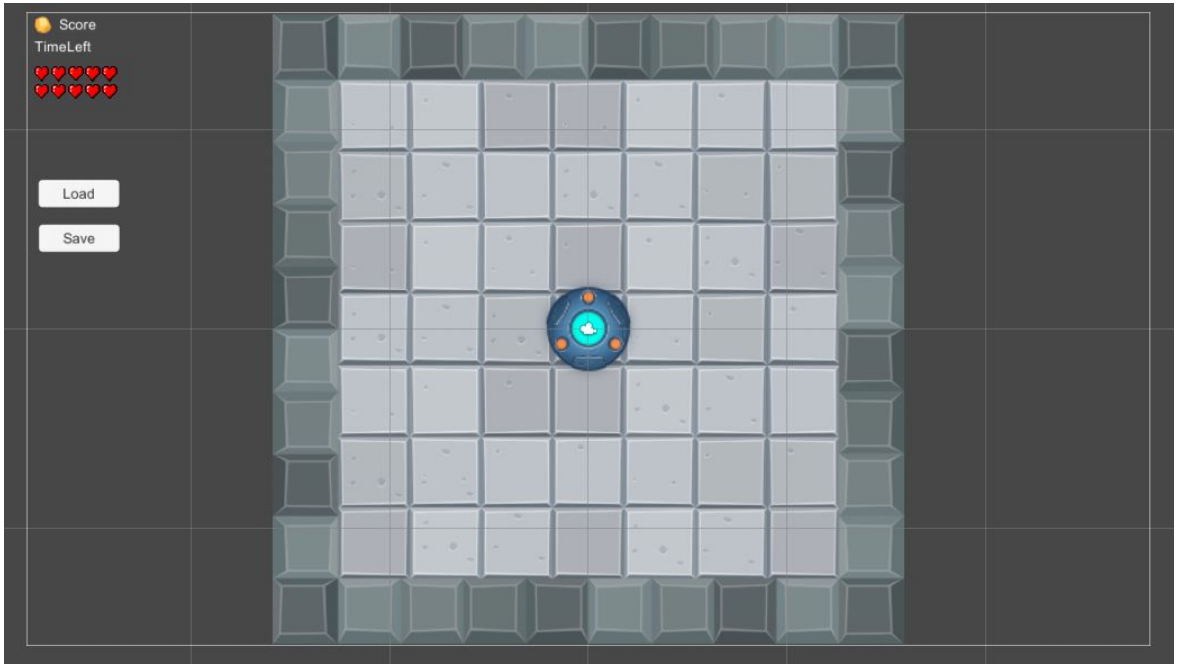
Salvando arrays de objetos C# em JSON

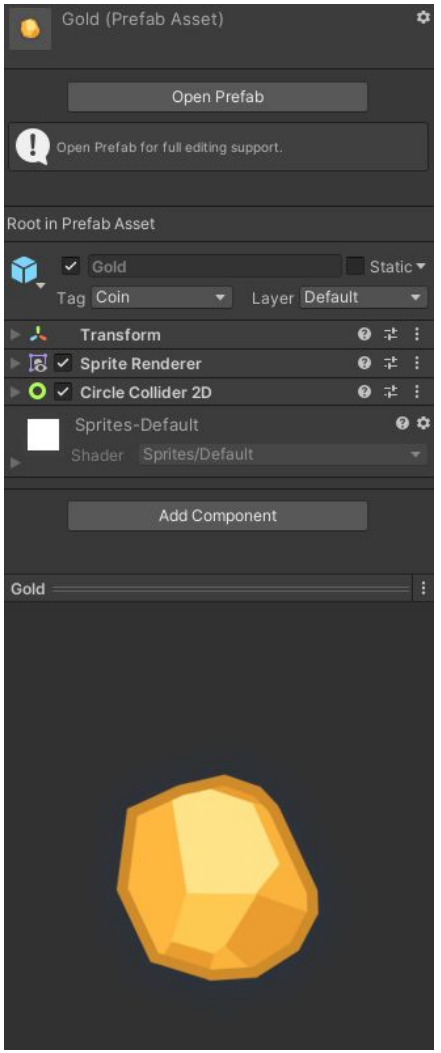
Em um ambiente Unity o desenvolvedor pode salvar **arrays de objetos** do jogo seguindo os passos:

- Identificar os **dados dos itens** que deseja salvar
- Criar uma **classe** que concencentra esses dados
- Criar um **array de objetos**
- Criar uma **representação Json** dos dados
- Salvar resultado em um **arquivo**

Salvando arrays de objetos C# em JSON

- Criar prefabs e excluir os sprites





Salvando arrays de objetos C# em JSON

- Copiar o arquivo AstroManager.cs para o projeto

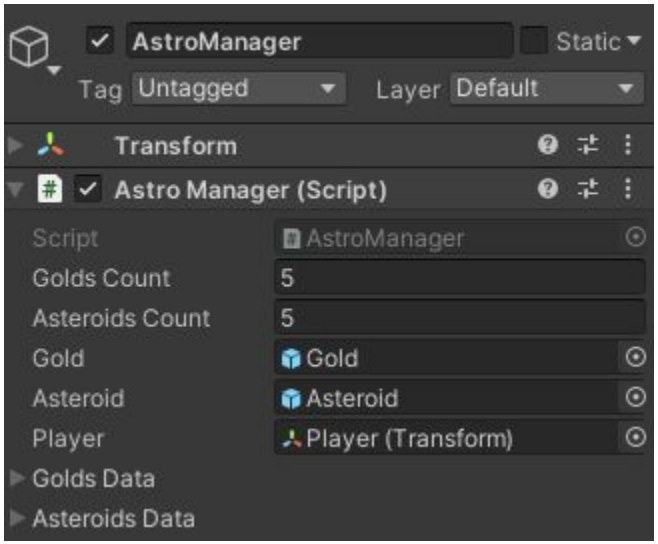
O script contém as classes

- AstroData que concentra dados de um objeto (Gold ou Asteroid) do game.
- AstroManager que cria os objetos Gold e Asteroid em tempo de execução.

```
[Serializable]
public class AstroData {
    public Vector3 position;
    private GameObject reference;

    public void setReference(GameObject reference) {
        this.reference = reference;
    }

    public GameObject getReference() {
        return this.reference;
    }

    public bool isNull() {
        return reference == null;
    }
}



public class AstroManager : MonoBehaviour
{
    public int goldsCount = 5;
    public int asteroidsCount = 5;

    public GameObject gold;
    public GameObject asteroid;
    public Transform player;

    public List<AstroData> goldsData = new List<AstroData>();
    public List<AstroData> asteroidsData = new List<AstroData>();
```

Salvando arrays de objetos C# em JSON

- Copiar o arquivo AstroManager.cs para o projeto
- Inserir um GameObject, renomeá-lo para AstroManager e anexar o script AstroManager.cs
- Associar os atributos **Player**, **Gold** e **Asteroid** aos objetos correspondentes.



Salvando arrays de objetos C# em JSON

- No arquivo SaveLoadJSON.cs, incluir 2 arrays públicos de AstroData na classe GameData

```
public class GameData {
    public Vector3    position;
    public Quaternion rotation;
    public float timeLeft;

    public AstroData[] goldsData;
    public AstroData[] asteroidsData;
}
```

Inserir

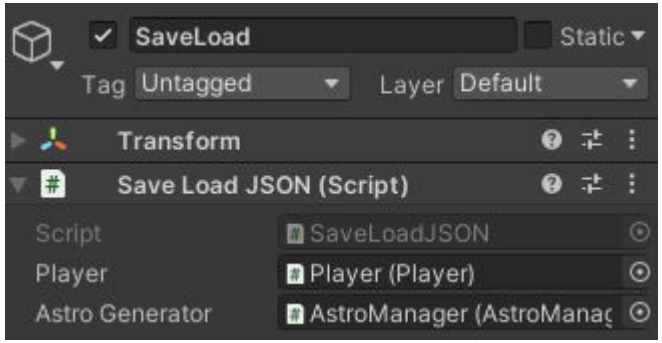
Salvando arrays de objetos C# em JSON

- No arquivo SaveLoadJSON.cs, criar um novo atributo do tipo AstroManager na classe SaveLoadJSON.

```
public class SaveLoadJSON : MonoBehaviour
{
    public Player player;
    public AstroManager astroGenerator;

    public void SaveToJsonFile() {
```

Inserir

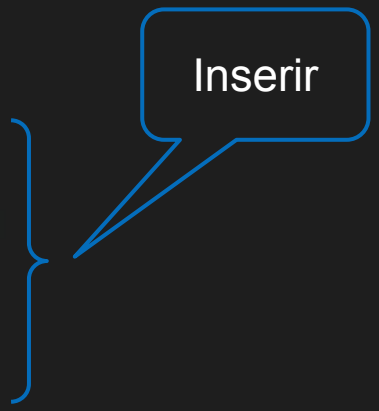
- Anexar o objeto correspondente na interface do Unity



Salvando arrays de objetos C# em JSON

- No método SaveToJsonFile(), inserir as linhas:

```
public void SaveToJsonFile() {

    GameData gd = new GameData();

    gd.position = player.transform.position;
    gd.timeLeft = player.timeLeft;

    // Remove todos os Objetos Gold e Asteroids nulos
    astroGenerator.goldsData.RemoveAll(item => item.isNull());
    astroGenerator.asteroidsData.RemoveAll(item => item.isNull());

    // Copia os arrays de Gold e Asteroids
    gd.goldsData = astroGenerator.goldsData.ToArray();
    gd.asteroidsData = astroGenerator.asteroidsData.ToArray();

    string json = JsonUtility.ToJson(gd, true);
    Debug.Log(json);

    File.WriteAllText(Application.dataPath + "/gamedata.json", json);
}
```



Resgatando na prática

Resgatando dados de um Json na prática

Em um ambiente **Unity** o desenvolvedor pode resgatar **arrays de objetos** de um jogo já encerrado, porém **salvo**, seguindo os passos:

- Carregar o conteúdo de um arquivo Json para um string
- Usar o método FromJson para criar um objeto com os dados salvos
- **Destruir** os objetos do game a serem restaurados
- **Recriar** os objetos a partir dos dados salvos

Salvando arrays de objetos C# em JSON

- No método LoadToJsonFile(), inserir as linhas:

```
public void LoadFromJsonFile() {

    string json = File.ReadAllText(Application.dataPath + "/gamedata.json");

    GameData gd = JsonUtility.FromJson<GameData>(json);

    player.transform.position = gd.position;
    player.timeLeft = gd.timeLeft;


    astroGenerator.DestroyAll();                    // Remove todos os Golds e Asteroids

    foreach (AstroData ad in gd.goldsData)          // Recria os Golds savlos
    {
        astroGenerator.NewAstro(astroGenerator.gold, ad.position);
    }


    foreach (AstroData ad in gd.asteroidsData)      // Recria os Asteroids savlos
    {
        astroGenerator.NewAstro(astroGenerator.asteroid, ad.position);
    }
}
```

Inserir

Para saber mais...

https://www.w3schools.com/js/js_json_datatypes.asp

https://www.json.org/json-en.html

https://docs.unity3d.com/ScriptReference/JsonUtility.html

https://docs.unity3d.com/ScriptReference/JsonUtility.ToJson.html

https://docs.unity3d.com/ScriptReference/JsonUtility.FromJson.html

https://docs.unity3d.com/ScriptReference/JsonUtility.FromJsonOverwrite.html

https://www.w3schools.com/whatis/whatis_json.asp

https://www.w3schools.com/js/js_json_intro.asp

Exercícios

Exercícios AC1

A partir do projeto inicial, fornecido pelo professor, crie as funcionalidades de **salvar** e **carregar** o jogo usando JSON.

Salve e carregue pelo menos **5 atributos** do jogo.

Entrega individual pelo Canvas.

Serão considerados para nota:

- A corretude (deve funcionar como esperado)
- A completude (pelo menos 5 atributos)
- A estética e experiência do usuário.

Opcionalmente o aluno pode fazer melhorias, que serão consideradas para a nota.

Tipos de dados em JSON

Os dados em JSON podem ser representados respeitando os seguintes tipos de dados:

Tipo	Exemplo
String	`{ "PlayerName" : "Nasrudin" }`
Numbers	`{ "Score" : 1024 }`
Boolean	`{ "IsJumping" : true }`
Arrays	`{ "Items" : [ "Sword", "Shield", "spear" ] }`
Object	`{ "Enemy" : { "Class" : "Clown", "health" : 0.85 } }`
null	`{ "Awards" : null }`