# **Dokumentácia k ukladaniu a načítaniu dát v hre Arkanoid**

Táto implementácia ukladania a načítania dát bude rozdelená do dvoch častí:

* **Dáta hráča** – meno hráča, skóre z levelov, hviezdičky.
* **Dáta levelov** – informácie o dizajne levelov vytvorených hráčom v level editore (pozadie levelu, pozície a typy blokov)

# **Ukladanie a načítanie dát hráča**

## **Trieda SaveDate**

Ako prvé sme vytvorili triedu, ktorá bude reprezentovať štruktúru ukladaných dát a bude použitá na manipuláciu s týmito dátami pred uložením a po načítaní. Trieda sa nazýva SaveData.

using System.Collections.Generic;

[System.Serializable]

public class LevelData

{

    public int levelID;

    public int score;

    public int stars;

}

[System.Serializable]

public class SaveData

{

    public string playerName;

    public List<LevelData> levelSaves = new List<LevelData>(new LevelData[10]);

}

Táto trieda obsahuje premennú **playerName** typu string pre uchovávanie mena hráča a list objektov typu **LevelData** o veľkosti 10, keďže v hre sa bude nachádzať 10 levelov. Trieda **LevelData** obsahuje 3 premenné typu integer, **levelID** pre id konkrétneho levelu, **score** pre maximálne score dosiahnuté v danom leveli a **stars** pre maximálny počet hviezdičiek dosiahnutých v danom leveli.

Všetky premenné musia byť verejné, aby k nim bolo možne pristupovať a triedy musia mať atribút **Serializable** aby ich bolo možné serializovať.

# **Funkcie Save() a Load()**

Funkcie **Save()** a **Load()** slúžia na ukladanie a načítanie z a do súboru **save.json**. Na serializovanie a deserializovanie dát do formátu JSON používame Unity triedu **JsonUtility** a pre zapísanie a prečítanie dát z a do súboru používame triedu **File** z knižnice System.IO

## **Funkcia Save()**

Funkcie **Save()** slúžia na serializvanie dát v triede **SaveData** do formátu JSON a uloženie týchto dát do súboru **save.json**.

Táto funkcia má ako argument objekt triedy **SaveData** obsahujúci dáta pre uloženie.

Dáta sa serializujú do premennej **json** typu string formátované pre jednoduchú čitateľnosť človekom (druhy argument vo funkcii ToJson).

Následne sa text v tejto premennej zapíše do súboru save.json na lokácii ....... ak tento súbor ešte neexistuje bude automaticky vytvorený.

    public void Save(SaveData saveData)

    {

        string json = JsonUtility.ToJson(saveData, true);

        File.WriteAllText(Application.dataPath + "/save.json", json);

    }

## **Funkcia Save()**

Funkcie **Load()** slúžia na načítanie dát zo súboru **save.json** a deserializvanie dát z formátu JSON do triedy **SaveData**.

Text zo súboru **save.json** je načítaný do premennej **json**.

Text v tejto premennej je následne deserializovaný a údaje sú uložené do triedy **SaveData**, ktorú táto funkcia následne vráti.

    public SaveData Load()

    {

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

        return JsonUtility.FromJson<SaveData>(json);

    }

# **Práca s dátami pred a po uložení**

V tejto implementácii s dátami pracujeme v skripte **GameManager**.

Na konci funkcie **Awake()** skontrolujeme či súbor **save.json** existuje (týmto zároveň kontrolujeme či bola hra spustená po prvý krát). Ak súbor existuje načítame dáta do premennej **saveData**, ktorá je inštanciou triedy **SaveData** aby sme následne k dátam mohli pristupovať. Ak súbor neexistuje tak premennú **playerName** v premennej saveData nastavíme na hodnotu „UnknownPlayer“ a dáta uložíme, čím sa zároveň aj vytvorí súbor **save.json**.

        if(!File.Exists(Application.dataPath + "/save.json"))

        {

            SetPlayerName("UnknowPlayer");

            Save(saveData);

        }

        else

        {

            saveData = Load();

        }