# Unity Ficheros y Serializar

## Ficheros

Se tiene que importar la librería System.IO

File.WriteAllText(fichero, string): Escribe un texto en el fichero que se le indique.

Ej: File.WriteAllText(C:\text.txt, “Hola Mundo”)

Application.dataPath: devuelve la ruta de la aplicación.

File.ReadAllText(fichero): Devuelve el texto dentro del fichero que se indica.

Ej: File.ReadAllText(Application.dataPath + "/DataBase/data.json");

FileStream: Fichero para poder serializar. Se crea de la siguiente manera:

FileStream file = File.Open(“path+filena”,FileAccess);

Ej: FileStream file = File.Open(Application.dataPath + "/DataBase/data.dat", FileMode.OpenOrCreate);

## Json

JsonUtiliy.toJson(objeto): Utilidad para convertir los objetos en cadenas de texto tipo Json.

Ej: JsonUtiliy.toJson(MonoBehaviour objeto)

JsonUtility.FromJsonOverwrite(json, objeto): Cargar un json y sobrescribe un objeto (MonoBehavour, Class) con los datos de este.

Ej: string json = {“Nivel”:1, “Vidas”:5,”Nombre”:”Alvaro”}

JsonUtility.FromJsonOverwrite(String json, componente);

JsonUtility.FromJson<Typeobjeto>(string json): Devuelve los datos de un json y los inserta en el tipo de objeto que le indiquemos. Tiene que ser una clase, debe estar serializada y no puede heredar del MonoBehavour

Ej: componente = JsonUtility.FromJson<Estados\_class>(json);

## BinaryFormatter

La clase tiene que estar Serializada. No todos los tipos de objeto se pueden Serializar. Los Colores, los GameObjects no.

Se tiene que usar un formateador para transformarlo en binario y insertarlo en un fichero.

Librería para acceder al Serialziador:

using System.Runtime.Serialization.Formatters.Binary;

Se crea un Formateador de la clase BinaryFormatter. En esta clase tenemos los metodos para serializar y deserializar:

BinaryFormatter format = new BinaryFormatter();

Serialize(FileStream, objeto): Para serializar se necesita un fichero de tipo FileStream y una Clase Serializada.

Ej: format.Serialize(file, estado\_obj);

Deserialize(file): Sirve para deserializar un Fichero. Devuelve un objeto que hay que castear al tipo correcto.

Ej: estado = (Estado.Estado\_Class)format.Deserialize(file);