Save Data System

[Random Adjective]

Versión 1.0.0

English manual

Description: This package corresponds to a basic data saving system through "Binary Format" that allows storing volatile information in non-volatile files so that it can be recovered if the application is closed.

- To use this package in a simple way, it is only necessary that you understand the operation of the 3 classes within it.
- The first of these is DataManager, this class is in charge of Create, Load and Save.
- The second class called Data is the main container object, it must be modified so that it stores the information that is necessary for the project.
- The third class StaticData is a supporting class that implements the Singleton design pattern.

How to use:

- In order to store the relevant data of the player, it is necessary to include them within the Data class, all the variables included in it will be stored in a non-volatile file, be careful with the variables you include, all the variables included within this class must be of classes marked as serializable so that the file can be created successfully.
- Within any class in your project you can access the StaticData.Data property, it returns a Data object if you are obtaining it and will overwrite the information if it is being assigned.

Example of obtaining data:
var data = StaticData.Data;

Example of saving data:
StaticData.Data = new Data ();

Sistema de Guardado de Datos

[Random Adjective]

Versión 1.0.0

Manual en español

Descripción: Este paquete corresponde a un sistema básico de guardado de datos a través de “Binary Format” que permite almacenar información volátil en archivos no volátiles de manera que esta pueda ser recuperada si la aplicación es cerrada.

- Para utilizar de manera sencilla este paquete solo es necesario que entienda el funcionamiento de las 3 clases dentro de este.
- La primera de estas es *DataManager,* esta clase es la encargada de Crear, Cargar y Guardar.
- La segunda clase llamada *Data* es el objeto contenedor principal, esta debe ser modificada para que almacene la información que sea necesaria para el proyecto.
- La tercera clase *StaticData* Es una clase de apoyo que implementa el patrón de diseño *Singleton*.

Como usar:

- Para poder almacenar los datos relevantes del jugador es necesario incluirlos dentro de la clase *Data* todas las variables que se incluyan en esta se almacenarán en un archivo no volátil, cuidado con las variables que incluyas, todas las variables incluidas dentro de esta clase deben ser de clases marcadas como serializables para que el archivo pueda ser creado con éxito.
- Dentro de cualquier clase en tu proyecto puedes acceder a la propiedad **StaticData.Data**, esta retorna un objeto *Data* si lo estás obteniendo y sobrescribirá la información si se está asignando.

  Ejemplo de obtención de datos:
  - var data = StaticData.Data;

  Ejemplo de Guardado de datos:
  - StaticData.Data = new Data();