Modelo de documentação JsDC

O JsDC (Json Database Creator) é uma ferramenta que foi desenvolvida para auxiliar a criação de uma interface similar a um banco de dados utilizando a estrutura de um Json no C#, onde é possível ler e gravar objetos em um arquivo de texto e fazer consultas como em uma linguagem de banco de dados (como SQL).

Antes de entender como o JsDC funciona, vamos às dependências. Foi necessário importar o package Newtownsoft, que contém a biblioteca de serialização do JSon. Foi usado também System.IO que contém a biblioteca para manipulação de arquivos.

No JsDC há uma classe chamada *Globals*, dentro dela há variáveis estáticas que recebem valores padrão de configuração. `filePath` recebe o caminho para a pasta onde será salvo o arquivo gerado pelo JsDC, enquanto `fileName` recebe o nome do arquivo que será gerado e seu tipo. Que vem do `.appconfig`

```
using System.Configuration;

    public class Globals
    {
        public static string filePath
    {
        get
        {
            return ConfigurationManager.AppSettings["FILEPATH"];
        }
    }

        public static string fileName
    {
        get
        {
            return ConfigurationManager.AppSettings["FILENAME"];
        }
    }

    }
```

A classe `JsonDictionary` é a classe responsável por armazenar os dados do arquivo Json. Ela tanto receberá os dados quanto será usada para gravar. Ela contém campos que são `Lists<>()` de objetos, que devem ser instâncias de uma classe, cujos atributos são os campos que se deseja gravar no Json. Sendo assim, `JsonDictionary` serve como o Banco de Dados, e as classes como as tabelas dentro dele, de onde se leem os registros.

Ex.: Uma lista para armazenar nome e biografia de personagens.

```
[Serializable]
public class JsonDictionary
{
    public List<Personagem> personagem;
}

public JsonDictionary()
{
        personagens = new List<Personagem>();

}
```

t

```
public class Personagem
{
    int id;
    string nome;
    string bio;
}
```

para gravar um novo *Personagem* basta chamar a instância de *JsonDictionary* com os dados gravados e passar o construtor da classe.

```
JsonDictionary myJsonDictionary;
myJsonDictionary.personagens.Add(new Personagem(0, "Smough", "Um famoso caçador de dragões"));
```

A classe *JsonHandler* é responsável pela manipulação dos dados do arquivo. Sendo usada para gravar os dados e para lê-los.
a função *JsonHandler.GravarJson(JsonDictionary)* grava a instância de *JsonDictionary* para o arquivo especificado em *Globals*
.

```
jsonHandler.GravarJson(myJsonDictionary);
```

*JsonHandler.LerJson()* retorna uma instância de *JsonDictionary* com os dados gravados no arquivo especificado em *Globals.*

```
myJsonDictionary = jsonHandler.LerJson();
```

Para fazer consultas basta chamar a instância do *JsonDictionary* e a classe que deseja obter os valores. E ele retornará o valor desejado para manipulação.

```
string nomeDoPersonagem = myJsonDictionary.personagem.nome;
```

para modificar os campos dentro do *JsonDictionary*, basta saber o *index* dentro da classe correspondente e acessar o campo que deseja alterar e gravar em seguida.

```
myJsonDictionary = jsonHandler.LerJson();
myJsonDictionary.personagem[index].nome = "Ornstein";
jsonHandler.GravarJson(myJsonDictionary);
```