# Working with JSON

# Project Set Up

## Setup Steps for Visual Studio

From within VS (using point and click):

* Create new solution called WorkingWithDataAndFiles
* Add console project called json-demo.csproj to the solution
* Use NuGet to add a reference to the Newtonsoft.json JSON package

## Setup Steps for Visual Studio Code

From Command Prompt:

* Prepare solution by typing:

dotnet new solution --name demos

* Create console project, and add to solution by typing:

dotnet new console --output json-demo

dotnet sln add json-demo

* Install JSON package in the json-demo project by typing:

dotnet add json-demo package newtonsoft.json

## Now do the following regardless of project type:

* Add a code file called PersonFilms.cs
* Replace the file's content with the following:

// PersonFilms class (name, age and FavouriteFilms) –

// NB: need default constructor for JSON deserialization

public class PersonFilms

{

public string Name { get; set; }

public int Age { get; set; }

public List<string> FavouriteFilms { get; set; }

public PersonFilms()

{

FavouriteFilms = new List<string>();

}

public PersonFilms(string name, int age, List<string> favouriteFilms)

{

this.Name = name;

this.Age = age;

this.FavouriteFilms = favouriteFilms;

}

}

* Replace the Program.cs file’s content with the following:

// Include all relevant namespaces including json

using System;

using System.Collections.Generic;

using System.IO;

using Newtonsoft.Json;

namespace json\_demo

{

internal class Program

{

static void Main(string[] args)

{

// Writing and reading a strongly typed class

PersonFilms personFilms = new PersonFilms

{

Name = "Andrew",

Age = 50,

FavouriteFilms = new List<string>() {

"Toy Story",

"Toy Story 2",

"Aliens"

}

};

string pfs = JsonConvert.SerializeObject(personFilms, Formatting.Indented);

Console.WriteLine(pfs);

PersonFilms pfs2 = JsonConvert.DeserializeObject<PersonFilms>(pfs);

Console.WriteLine(pfs2.Name);

Console.WriteLine(pfs2.Age);

pfs2.FavouriteFilms.ForEach(f => Console.WriteLine(f));

// Writing an anonymous object to file

var obj1 = new

{

name = "Maria",

age = 39,

favouriteFilms = new List<string>() {

"Love Story",

"Inception",

"It's a Wonderful Life"

}

};

string s1 = JsonConvert.SerializeObject(obj1);

string json1 = JsonConvert.SerializeObject(obj1, Formatting.Indented);

File.WriteAllText("file1.json", json1);

// Reading JSON into an anonymous, dynamic object then picking out elements

string s2 = File.ReadAllText("file1.json");

dynamic obj2 = JsonConvert.DeserializeObject(s2);

Console.WriteLine(obj2.name);

Console.WriteLine(obj2.age);

Console.WriteLine(obj2.favouriteFilms);

// Alternative way to read elements, if they are non-standard (include dots or hyphens)

int age = obj2["age"];

Console.WriteLine(age);

}

}

}

* Review the code and try to work out what it’s doing.
* Build and run the code and confirm it behaves as you expected.
* Add another class called PersonFilms2 to the project.
* Replace the existing code in the file with the following:

using Newtonsoft.Json;

namespace json\_demo

{

internal class PersonFilms2

{

[JsonProperty(PropertyName = "name", Order = 2)]

public string Name { get; set; }

[JsonProperty(PropertyName = "age-in-years", Order = 1)]

public int Age { get; set; }

[JsonProperty(PropertyName = "favourite-films", Order = 3)]

public List<string> FavouriteFilms { get; set; }

public PersonFilms2()

{

FavouriteFilms = new List<string>();

}

public PersonFilms2(string name, int age, List<String> favouriteFilms)

{

this.Name = name;

this.Age = age;

this.FavouriteFilms = favouriteFilms;

}

}

}

* Add the following lines of code to the foot of the Main function in the Program class:

// Use JsonProperty attributes to control the mapping of property

// names to JSON names (name, age-in-years, favourite-films)

// Note serialisation order specified in class via the JsonPropertyAttribute Order property

PersonFilms2 pf2 = new PersonFilms2("Sharmi", 27, new List<string> { "Top Gun", "Die Hard", "ET" });

s2 = JsonConvert.SerializeObject(pf2, Formatting.Indented);

Console.WriteLine(s2);

PersonFilms2 pd2 = JsonConvert.DeserializeObject<PersonFilms2>(s2);

string name = pd2.Name;

age = pd2.Age;

List<string> favouriteFilms = pd2.FavouriteFilms;

Console.WriteLine(name);

Console.WriteLine(age);

favouriteFilms.ForEach(f => Console.WriteLine(f));

dynamic pdd = JsonConvert.DeserializeObject(s2);

string namedd = pdd["name"];

int ageInYearsdd = pdd["age-in-years"];

List<string> favouriteFilmsdd = pdd["favourite-films"].ToObject<List<string>>();

Console.WriteLine(namedd);

Console.WriteLine(ageInYearsdd);

favouriteFilmsdd.ForEach(f => Console.WriteLine(f));

// Deserialization of unrecognised elements is ignored (not an error)

//Console.WriteLine(obj2.agex);

Console.WriteLine($"{(obj2.agex != null ? obj2.agex : string.Empty)}");

* Review the code and try to work out what it’s doing.
* Build and run the code and confirm it behaves as you expected.