using System;

using System.Collections.Generic;

using System.Linq;

using System.Reflection;

using System.Security.Cryptography.X509Certificates;

using System.Text;

using System.Threading.Tasks;

using System.Web.Services;

namespace ConsoleApp1

{

[AddMetaData(" ghjh")]

internal class Program

{

static void Main(string[] args)

{

/\*Add(23, 34);

Add(new int[]{4,5,6});\*/

// Employee employee = new Employee(); // Compile time

// employee.PrintEmployee();

//Reflection : -- to inspect the library at run time and grab all the details from it

// at run time

/\* int i = 42;

Type type = i.GetType();

Console.WriteLine(type.FullName);\*/

// Let's say we want to fetch all the properties , methods and constructor of a Employee

//Below is to print the metadata of current assembly or current class

//Assembly assembly = Assembly.LoadFile(Assembly.GetExecutingAssembly().Location);

//Below is to print the metadata of any class or assembly by providing the absolute path of it

/\* String path = @"C:\\Program Files (x86)\\Reference Assemblies\\Microsoft\\Framework\\.NETFramework\\v4.7.2\\mscorlib.dll";

//Assembly assembly = Assembly.LoadFile(path);

Type[] types = assembly.GetTypes();

//

foreach (Type type in types)

{

;

// Type type = Type.GetType("ConsoleApp1.Employee");

if (type != null)

{

Console.WriteLine(type.FullName);

Console.WriteLine(type.Name);

GetMethods(type);

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\* Properties \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

GetProperties(type);

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\* Constructors \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

GetConstructors(type);

}

}\*/

Type type = Type.GetType("ConsoleApp1.Employee");

if (type != null)

{

MethodInfo mi = type.GetMethod("PrintEmployee");

if (mi != null)

{

//Employee emp = new Employee() // this was at compile time

object typeinstance = Activator.CreateInstance(type);

object[] param = new object[2];

param[0] = "Preeti";

param[1] = 35;

//emp.printEmployee()

mi.Invoke(typeinstance, param);

}

}

}

public static void GetMethods(Type type)

{

MethodInfo[] methodInfo = type.GetMethods();

foreach (MethodInfo mi in methodInfo)

{

Console.WriteLine($"Method Name : {mi.Name} Return Type : { mi.ReturnType}");

}

}

public static void GetProperties(Type type)

{

PropertyInfo[] propertyInfo = type.GetProperties();

foreach (PropertyInfo pi in propertyInfo)

{

Console.WriteLine($"Method Name : {pi.Name} Return Type : {pi.PropertyType}");

}

}

public static void GetConstructors(Type type)

{

ConstructorInfo[] constructorInfo = type.GetConstructors();

foreach (ConstructorInfo ci in constructorInfo)

{

Console.WriteLine($"Method Name : {ci.Name} Return Type : {ci.IsPublic}");

}

}

/\* // [WebMethod]

[Obsolete("This method will get permanently deleted on 5 th Feb 2025")]

public static void Add(int a, int b)

{

Console.WriteLine("The result is : " + (a + b));

}

[AddMetaData("This is a new method created in place of Add method used previously")]

public static int Add(int[] number)

{

int result = 0;

for (int i = 0; i < number.Length; i++)

{ result += number[i];

Console.WriteLine(result);

}

Console.WriteLine(result);

return result;

}

public static void Update()

{ }

public static void Delete()

{ }\*/

}

}

Employee.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

public class Employee

{

public int Id { get; set; }

public string Name { get; set; }

public string Salary{get; set;}

public int Age { get; set; }

public Employee(int id, string name, string salary, int age)

{

Id = id;

Name = name;

Salary = salary;

Age = age;

}

public Employee()

{

}

private void PrintEmployee()

{

Console.WriteLine("Method created to print Employee Details :");

}

public void PrintEmployee(string name , int age)

{

Console.WriteLine("Method created to print Employee Details :" + name + age);

}

}

}