C# Access Specifiers

What is Access Specifiers in C#?

Access Specifiers defines the scope of a class member. A **class member** can be **variable** or **function**.

List of Access Specifiers

- **Public Access Specifiers**
- **Private Access Specifiers**
- **Protected Access Specifiers**

1.1 Public Access Specifiers (C#)

The class member, that is defined as a **public** can be **accessed** by other class members that are initialized **outside** the **class**.

A **public** member can be **accessed** from anywhere even **outside** the **namespace**.

Sample Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Public_Access_Specifiers
{
    class access
    {
        // String Variable declared as public
        public string name;
        // Public method
        public void print()
```

```
        {
            Console.WriteLine("\nMy name is " + name);
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            access ac = new access();
            Console.Write("Enter your name:\t");
            // Accepting value in public variable that is outside the class
            ac.name = Console.ReadLine();
            ac.print();

            Console.ReadLine();
        }
    }
}
```

Output

```
Enter your name:    Steven Clark

My name is Steven Clark
_
```

1.2 Private Access Specifiers (C#)

The **private** access specifiers **restrict** the member variable or function to be called **outside** of the parent class.

A **private** function or variable **cannot be called outside** of the same **class**.

It **hides** its member variable and method from other class and methods. However, you can store or retrieve the value from private access modifiers using **get/set property**.

Sample Program

```
. using System;
. using System.Collections.Generic;
. using System.Linq;
. using System.Text;
.
. namespace Private_Access_Specifiers
. {
.     class access
.     {
.         // String Variable declared as private
.         private string name;
.         public void print() // public method
.         {
.             Console.WriteLine("\nMy name is " + name);
.         }
.     }
.
.     class Program
.     {
.         static void Main(string[] args)
.         {
.             access ac = new access();
.             Console.Write("Enter your name:\t");
.             // raise error because of its protection level
.             ac.name = Console.ReadLine();
.             ac.print();
.             Console.ReadLine();
.         }
.     }
. }
```

Output

```
Error 1: Private_Access_Specifiers.access.name' is inaccessible due to its protection leve
l __
```

In the above example, you cannot call name variable outside the class because it is declared as private.

1.3 Protected Access Specifiers C#

The **protected** access specifier hides its member variables and functions from other classes and objects.

This type of variable or function can only be **accessed** in **child** class. It becomes very important while implementing **inheritance**.

Sample Program

```
. using System;
. using System.Collections.Generic;
. using System.Linq;
. using System.Text;
.
. namespace Protected_Specifier
. {
.     class access
.     {
.         // String Variable declared as protected
.         protected string name;
.         public void print()
.         {
.             Console.WriteLine("\nMy name is " + name);
.         }
.     }
.
.     class Program
```

```
    {
        static void Main(string[] args)
        {
            access ac = new access();
            Console.Write("Enter your name:\t");
            // raise error because of its protection level
            ac.name = Console.ReadLine();
            ac.print();
            Console.ReadLine();
        }
    }
}
```

Output

```
'Protected_Specifier.access.name' is inaccessible due to its protection level.
```

This is because; the protected member can only be accessed within its child class.

You can use protected access specifiers as follow:

Sample Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Protected_Specifier
{
    class access
    {
        // String Variable declared as protected
        protected string name;
        public void print()
        {
```

```
            Console.WriteLine("\nMy name is " + name);
        }
    }

    class Program : access // Inherit access class
    {
        static void Main(string[] args)
        {
            Program p = new Program();
            Console.Write("Enter your name:\t");
            p.name = Console.ReadLine(); // No Error!!
            p.print();
            Console.ReadLine();
        }
    }
}
```

Output

```
Enter your name:    Steven Clark

My name is Steven Clark
```