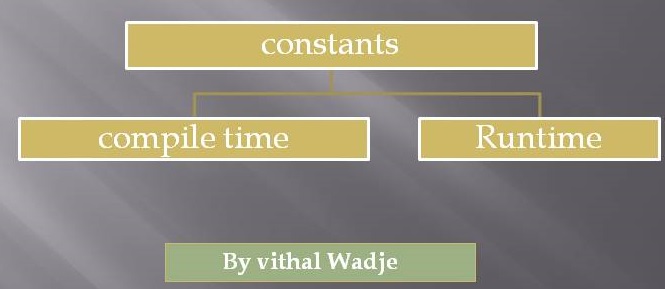
Const vs Readonly in C#.NET

**What is a Constant variable?**  
A variable whose value can not be changed during the execution of the program is called a constant variable.

In the above definition, the value can not be changed during execution of the program, which means we cannot assign values to the constant variable at run time. Instead, it must be assigned at the compile time.

Constants are of the following types:



In the above diagram, we can see the Constants are of two types

1. **Compile time constants** (const)
2. **Runtime constants** (Readonly)

* **Compile time constants**

The compile time constants are declared by using the const keyword, in which the value can not be changed during the execution of the program.

**Syntax**

     int const a=10;

**Some key points about const variable**

* It must be assigned a value at the time of declaration.

             eg.

             int const a=10;

* const only allow constant variables into the expression.

   eg.

          int const a=10;

          int const b=20; // correct

          int const c=a+b;

          int  d=1;

          int const c=d+b;   //compile time error because d variable is the non constant into expression.

* const can be declared at the class level as well as inside the method.
* const can not be declared using static keyword because they are by default static.
* constants are absolute constants in which their value cannot be changed or assigned at the run time.
* constant variables are compile time variables.

**When to use const**

The const is used when its value is absolutely constant. Such PI values cannot be changed, but according to your needs, you can use it as you wish rather than declaring PI values.

Example of **const**  variable.

1. **using** System;
3. **namespace** UsingConst
4. {
5. **class** Program
6. {
7. **const** **int** a = 10;
9. **static** **void** Main(**string**[] args)
10. {
11. **const** **int** b = 20;
12. **const** **int** c = b + a;
13. Console.WriteLine(c);
14. Console.ReadLine();
15. }
16. }
17. }

The output of the above program is 30. From the above example, we clearly see that the const must be assigned the value at declaration time and in expression both the variables must be const.

**Runtime constants**(Readonly)

The Run time constants are declared by using the Readonly keyword which value can not be changed during the execution of the program.

**Syntax**

int Readonly a; or

 int Readonly a=0;

**Some key points about const variable**

* It's not just to assign a value at the time of declaration, we can also assign the value for read-only through the constructor.

eg.

int readonly a;

a=0;

* Readonly allows, readonly constant as well as non-readonly constant variables into the expression.

eg.

int readonly a=10;

int b=1;

int readonly c=a+b;

* Readonly can be declared only at the class level, not inside the method.
* Readonly can not be declared using static keywords because they are by default static.
* Readonly constant's value can be set through the reference variable.
* Readonly constant variables are a runtime time constant variable.

**When to use Readonly**

We can use **Readonly**when its value is not an absolute constant, which means it can be changed frequently, such as dollars vs INR, in this requirement we can set the value through a configuration file or another variable expression so we can avoid changing the class file frequently.

Example of Readonly variable.

Let's define the value from the config file for a readonly constant variable, which will be set through the constructor.

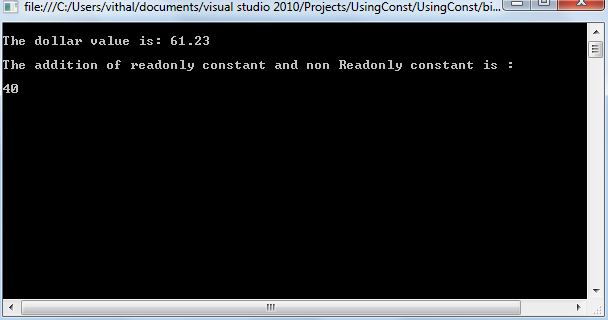
1. <configuration>
2. <appSettings>
3. <add key="DollarPrice" value="61.23"/>
4. </appSettings>
6. </configuration>

Now, let us explain it through this sample program

1. **using** System;
2. **using** System.Configuration;
4. **namespace** Usingreadonly
5. {
6. **class** Program
7. {
8. **readonly** **int** a = 10;
9. **int** b = 30;
10. **int** c;
11. **readonly** **string** r;
12. **public** Program()
13. {
14. r = ConfigurationManager.AppSettings["DollarPrice"];
15. Console.WriteLine("The dollar value is:"+" "+r);
16. Console.WriteLine();
17. c = a + b;
18. Console.WriteLine("The addition of readonly constant and non Readonly constant is :"+Environment.NewLine+ c);
19. Console.ReadLine();
20. }
22. **static** **void** Main(**string**[] args)
23. {
24. Program p = **new** Program();
26. Console.ReadLine();
27. }
28. }
29. }

In the above program ,we have assigned the value for readonly constant through constructor which is defined in the config  file since a readonly constant isn't necessary to assign the value at the time of declaration.

Now run the program, the output will be:



Let us  outline the differences between const and readonly variables.

1. const fields need to be initialized with declaration only, while readonly fields can be initialized at declaration or in the constructor.
2. const variables can be declared in methods, while readonly fields cannot be declared in methods.
3. const fields cannot be used with static modifier, while readonly fields can be used with static modifier.
4. const field is a compile-time constant, the readonly field can be used for run time constants.