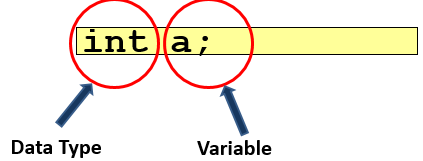
**Variables & Data Types in C#**

**Variables In C#**

* A variable is used to store data in a program and is declared with an associated data type.
* A variable has a name and may contain a value.
* A data type defines the type of data that can be stored in a variable.



A variable is an entity whose value can keep changing during the course of a program.

**Example**

* The age of a student, the address of a faculty member, and the salary of an employee are all examples of entities that can be represented by variables.

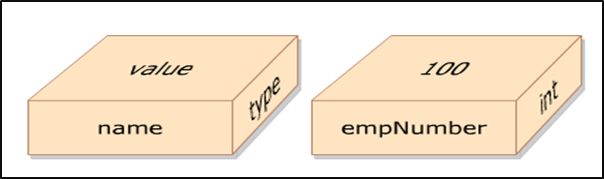
**Variables**

* In C#, a variable is a location in the computer’s memory that is identified by a unique name and is used to store a value. The name of the variable is used to access and read the value stored in it.
* Different types of data such as a character, an integer, or a string can be stored in variables. Based on the type of data that needs to be stored in a variable, variables can be assigned different data types.

**Using Variables**

* In C#, memory is allocated to a variable at the time of its creation and a variable is given a name that uniquely identifies the variable within its scope.
* On initializing a variable, the value of a variable can be changed as required to keep track of data being used in a program. When referring to a variable, you are actually referring to the value stored in that variable.

**The following figure illustrates the concept of a variable:**



**The following syntax is used to declare variables in C#:**

<datatype><variableName>;

Where,

* datatype: Is a valid data type in C#.
* variableName: Is a valid variable name.

**The following syntax is used to initialize variables in C#:**

<variableName> = <value>;

**Where,**

* = Is the assignment operator used to assign values.
* value: Is the data that is stored in the variable.

**The following code declares two variables, namely, empNumber and empName:**

int empNumber;

string empName;

**In Above Code,**

* an integer variable declares empNumber, and a string variable, empName. Memory is allocated to hold data in each variable.
* Values can be assigned to variables by using the assignment operator (=), as follows:
  + empNumber = 100;
  + empName = “David Blake”;
* You can also assign a value to a variable upon creation, as follows:
  + int empNumber = 100;

**Data Types**

* Different types of values such as numbers, characters, or strings can be stored in different variables. To identify the type of data that can be stored in a variable, C# provides different data types.
* When a variable is declared, a data type is assigned to the variable. This allows the variable to store values of the assigned data type.
* In C# programming language, data types are divided into two categories:

**1. Value Types**

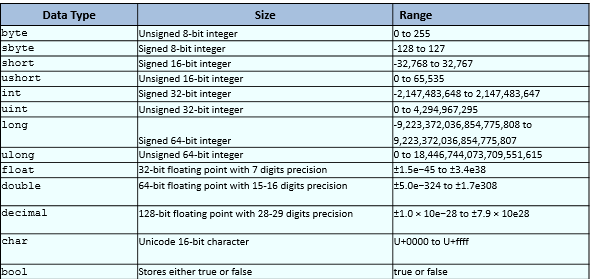
* Variables of value types store actual values that are stored in a stack that results in faster memory allocation to variables of value types.
* Most of the built-in data types are value types.
* The value type built-in data types are int, float, double, char, and bool. User-defined value types are created using the struct and enum keywords.

**2. Reference Types**

* Variables of reference type store the memory address of other variables in a heap.
* These values can either belong to a built-in data type or a user-defined data type.

**Pre-defined Data Types**

* The pre-defined data types are referred to as basic data types in C# that have a pre-defined range and size.
* The size of the data type helps the compiler to allocate memory space and the range helps the compiler to ensure that the value assigned, is within the range of the variable’s data type.



Reference data types store the memory reference of other variables that hold the actual values that can be classified into the following types:

**Object**

* Object is a built-in reference data type that is a base class for all pre-defined and user-defined data types. A class is a logical structure that represents a real world entity. The pre-defined and user-defined data types are created based on the Object class.

**String**

* String is a built-in reference type that signifies Unicode character string values. It allows you to assign and manipulate string values. Once strings are created, they cannot be modified.

**Class**

* A class is a user-defined structure that contains variables and methods. For example, the Employee class can be a user-defined structure that can contain variables such as empSalary, empName, and empAddress. It can also contain methods such as CalculateSalary(), which returns the net salary of an employee.

**Delegates**

* A delegate is a user-defined reference type that stores the reference of one or more methods.

**Interface**

* An interface is a user-defined structure that groups related functionalities which may belong to any class or struct.

**Array**

* An array is a user-defined data structure that contains values of the same data type, such as marks of students.

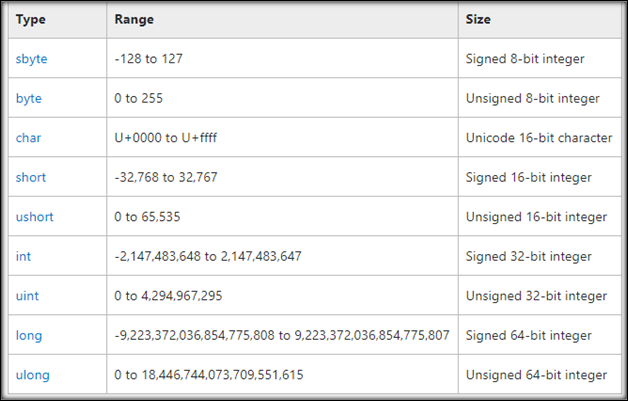
**INTEGRAL TYPE - Integer DataType**

**Signed Integers (Which Takes Negative And Positive Values)**

**Unsigned Integers (Which Only Takes Positive Values)**

* **sbyte**
* **byte**
* **short**
* **ushort**
* **int**
* **uint**
* **long**
* **ulong**

**DataType, Its Range & Its Size**



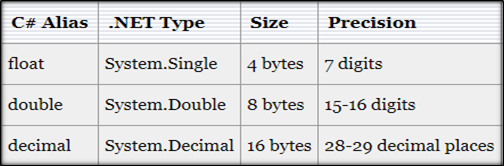
**2 Properties to get maximum & minimum value of data type**

* **MINVALUE** PROPERTY
* **MAXVALUE** PROPERTY

**Boolean Data Type**

**Bool** keyword is used for Boolean data type which only stores **TRUE** or **FALSE**.

**Float Double And Decimal Data Type**



**String And Character Data Type**

* **String** stores multiple characters in a single variable.
* Double quotes will be used with string data type.
* **Char** stores single character at a time in a variable.
* Single quotes will be used for char data type.

**Verbatim Literal**

* Verbatim literal is a string with an **@** symbol.
* Verbatim literal make escape sequences translate as normal printable characters to enhance readability.

**Practical Example:**

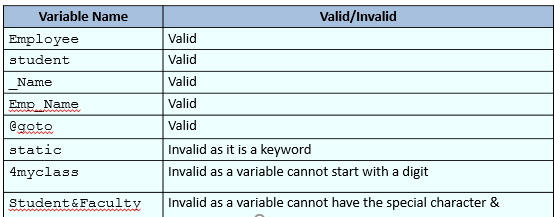
* **WITHOUT VERBATIM LITERAL**: "D:\\Adil\\Csharp\\Tutorials" – Less readable
* **WITH VERBATIM LITERAL**: **@**"D:\Adil\Csharp\Tutorials" – More readable

**Rules For Defining Variables**

A variable needs to be declared before it can be referenced by following certain rules as follows:

* A variable name can begin with an uppercase or a lowercase letter. The name can contain letters, digits, and the underscore character (\_).
* The first character of the variable name must be a letter and not a digit. The underscore is also a legal first character, but it is not recommended at the beginning of a name.
* C# is a case-sensitive language; hence, variable names count and Count refer to two different variables.
* C# keywords cannot be used as variable names. If you still need to use a C# keyword, prefix it with the ‘@’ symbol.
* It is always advisable to give meaningful names to variables such that the name gives an idea about the content that is stored in the variable.

**The following table displays a list of valid and invalid variable names in C#:**



**Variable Declaration**

* In C#, you can declare multiple variables at the same time in the same way you declare a single variable.
* After declaring variables, you need to assign values to them.
* Assigning a value to a variable is called initialization.
* You can assign a value to a variable while declaring it or at a later time.

**The following is the syntax to declare and initialize a single variable:**

<data type><variable name> = <value>;

**Where,**

* data type: Is a valid variable type.
* variable name: Is a valid variable name or identifier.
* value: Is the value assigned to the variable.

**The following is the syntax to declare multiple variables:**

<data type><variable name1>, <variable name2>,..,<variable nameN>;

**Where,**

* data type: Is a valid variable type.
* variable name1, variable name2, variable nameN: Are valid variable names or identifiers.

**The following is the syntax to declare and initialize multiple variables:**

<data <data type><variable name1> = <value1>, <variable name2> = <value2>;

**The following code demonstrates how to declare and initialize variables in C#:**

bool boolTest = true;

short byteTest = 19;

int intTest;

string stringTest = “David”;

float floatTest;

int Test = 140000;

floatTest = 14.5f;

Console.WriteLine(“boolTest = {0}”, boolTest);

Console.WriteLine(“byteTest = ” + byteTest);

Console.WriteLine(“intTest = ” + intTest);

Console.WriteLine(“stringTest = ” + stringTest);

Console.WriteLine(“floatTest = ” + floatTest);

**In Above Code,**

* Variables of type bool, byte, int, string, and float are declared. Values are assigned to each of these variables and are displayed using the WriteLine() method of the Console class.

**The code displays the following output:**

