# char

http://msdn.microsoft.com/it-it/library/x9h8tsay%28v=vs.80%29.aspx

La parola chiave **char** è utilizzata per dichiarare un carattere Unicode interno all'intervallo indicato nella tabella che segue. I caratteri Unicode sono caratteri a 16 bit utilizzati per rappresentare gran parte delle lingue scritte conosciute. Le costanti del tipo **char** possono essere scritte come caratteri effettivi, sequenze di escape esadecimali o rappresentazioni Unicode. È inoltre possibile impostare i codici con caratteri integrali. Tutte le istruzioni che seguono dichiarano una variabile **char** e la inizializzano con il carattere X:

char char1 = 'Z'; // Character literal

char char2 = '\x0058'; // Hexadecima (esadecimale) cerca nella tab

char char3 = (char)88; // Cast from integral type(asci) (esadecimale) cerca nella tab

char char4 = '\u0058'; // Unicode cerca nella tab

# Struttura Char

Rappresenta un carattere Unicode.

.NET Framework utilizza la struttura **Char** per rappresentare i caratteri Unicode. Ciascun valore a 16 bit è compreso tra gli esadecimali 0x0000 e 0xFFFF ed è memorizzato in una struttura **Char**. Il valore di un oggetto **Char** è rappresentato dal valore numerico (ordinale) a 16 bit a esso associato.

Un oggetto String è un insieme sequenziale di strutture **Char** che rappresenta una stringa di testo. La maggior parte dei caratteri Unicode può essere rappresentata da un singolo oggetto **Char**.

#### Funzionalità

La struttura Char fornisce metodi per confrontare gli oggetti Char, convertire il valore dell'oggetto Char corrente su un oggetto di un altro tipo e determinare la categoria Unicode di un oggetto Char:

* Utilizzare i metodi CompareTo e Equals per confrontare gli oggetti **Char**.
* Utilizzare i metodi IsControl, IsDigit, IsLetter, IsLetterOrDigit, IsLower, IsNumber, IsPunctuation, IsSeparator, IsSymbol, IsUpper e IsWhiteSpace per determinare se un carattere appartiene a una categoria Unicode specifica, come ad esempio cifra, lettera, punteggiatura, carattere di controllo e così via.
* Utilizzare i metodi GetNumericValue per convertire un oggetto Char che rappresenta un numero in un tipo di valore numerico. Utilizzare i metodi Parse e TryParse per convertire un carattere appartenente a una stringa in un oggetto Char. Utilizzare il metodo ToString per convertire un oggetto Char in un oggetto String.
* Utilizzare i metodi ToLower e ToUpper per trasformare da maiuscoli a minuscoli o viceversa i caratteri di un oggetto Char.

Metodo Char.IsControl (Char)

Indica se il carattere Unicode specificato è stato categorizzato come carattere di controllo.

Metodo Char.IsSymbol (Char)

Indica se il carattere Unicode specificato è stato categorizzato come carattere simbolo.

I simboli nello standard Unicode sono un set di caratteri definiti senza vincoli che includono gli elementi seguenti:

Simboli di valuta.

Simboli somiglianti a lettere che includono un set di simboli alfanumerici matematici, nonché simboli quali ℅, № e ™.

Forme numeriche, quali apici e pedici.

Operatori matematici e frecce.

Simboli geometrici.

Simboli tecnici.

Modelli braille.

Dingbat.

**C# ASCII Table:**

Decimal ASCII Hex

0 control 00

1 control 01

2 control 02

3 control 03

4 control 04

5 control 05

6 control 06

7 control 07

8 control 08

9 \t 09

10 \n 0A

11 \v 0B

12 \f 0C

13 \r 0D

14 control 0E

15 control 0F

16 control 10

17 control 11

18 control 12

19 control 13

20 control 14

21 control 15

22 control 16

23 control 17

24 control 18

25 control 19

26 control 1A

27 control 1B

28 control 1C

29 control 1D

30 control 1E

31 control 1F

32 space 20

33 ! 21

34 " 22

35 # 23

36 $ 24

37 % 25

38 & 26

39 ' 27

40 ( 28

41 ) 29

42 \* 2A

43 + 2B

44 , 2C

45 - 2D

46 . 2E

47 / 2F

48 0 30

49 1 31

50 2 32

51 3 33

52 4 34

53 5 35

54 6 36

55 7 37

56 8 38

57 9 39

58 : 3A

59 ; 3B

60 < 3C

61 = 3D

62 > 3E

63 ? 3F

64 @ 40

65 A 41

66 B 42

67 C 43

68 D 44

69 E 45

70 F 46

71 G 47

72 H 48

73 I 49

74 J 4A

75 K 4B

76 L 4C

77 M 4D

78 N 4E

79 O 4F

80 P 50

81 Q 51

82 R 52

83 S 53

84 T 54

85 U 55

86 V 56

87 W 57

88 X 58

89 Y 59

90 Z 5A

91 [ 5B

92 \ 5C

93 ] 5D

94 ^ 5E

95 \_ 5F

96 ` 60

97 a 61

98 b 62

99 c 63

100 d 64

101 e 65

102 f 66

103 g 67

104 h 68

105 i 69

106 j 6A

107 k 6B

108 l 6C

109 m 6D

110 n 6E

111 o 6F

112 p 70

113 q 71

114 r 72

115 s 73

116 t 74

117 u 75

118 v 76

119 w 77

120 x 78

121 y 79

122 z 7A

123 { 7B

124 | 7C

125 } 7D

126 ~ 7E

127 control 7F

**Esempio:**

char chA = 'A';

char ch1 = '1';

string str = "test string";

Console.WriteLine(chA.CompareTo('B')); // Output: "-1" (meaning 'A' is 1 less than 'B')

Console.WriteLine(chA.Equals('A'));// Output: "True"

Console.WriteLine(Char.GetNumericValue(ch1)); // Output: "1"

Console.WriteLine(Char.IsControl('\t'));// Output: "True"

Console.WriteLine(Char.IsDigit(ch1)); // Output: "True"

Console.WriteLine(Char.IsLetter(',')); // Output: "False"

Console.WriteLine(Char.IsLower('u')); // Output: "True"

Console.WriteLine(Char.IsNumber(ch1)); // Output: "True"

Console.WriteLine(Char.IsPunctuation('.'));// Output: "True"

Console.WriteLine(Char.IsSeparator(str, 4)); // Output: "True"

Console.WriteLine(Char.IsSymbol('+')); // Output: "True"

Console.WriteLine(Char.IsWhiteSpace(str, 4)); // Output: "True"

Console.WriteLine(Char.Parse("S"));// Output: "S"

Console.WriteLine(Char.ToLower('M')); // Output: "m"

Console.WriteLine('x'.ToString());// Output: "x"

**Esempio -** **Int to Char in C#:**

Console.WriteLine("(char)122 is {0}", (char)122);

// Output:

(char)122 is z

**Esempio –** **Stampa delle prime 10 lettere maiuscole dell’alfabeto:**

char myChar = 'A';

for (int i = 0; i < 10; i++)

{

Console.WriteLine("Valore decimale {0} Carattere {1}", myChar + i, (char)(myChar + i));

}

**Esempio –** **Da Maiuscolo a minuscolo e viceversa:**

I caratteri come le lettere possono essere scritte in maiuscolo o minuscolo e la distanza dei simboli varia di sole 32 posizioni.  
Quindi per convertire una minuscola in maiuscola o viceversa è sufficiente sottrarre o sommare 32 al carattere di partenza.  
Il seguente programma legge i caratteri da tastiera e converte tutte le lettere minuscole in maiuscole senza modificare altri caratteri. Il programma si ferma quando l’utente preme il punto.

char car;

Console.WriteLine ("\*\*\* PROGRAMMA DI CONVERSIONE \*\*\*");

do

{

car =(char) Console.Read(); // casting della variabile

if (car >= 'a' && car <= 'z')

{ car -= (char)32;

Console.WriteLine(car) ;}

else if (car >= 'A' && car <= 'Z')

{ car += (char)32;

Console.WriteLine(car) ;}

}

while (car !='.') ;

Console.WriteLine ("Programma Terminato...");