

Литералы

Литералы представляют неизменяемые значения (иногда их еще называют константами). Литералы можно передавать переменным в качестве значения. Литералы бывают логическими, целочисленными, вещественными, символьными и строчными. И отдельный литерал представляет ключевое слово `null`.

Логические литералы

Есть две логических константы - **true** (истина) и **false** (ложь):

```
1 Console.WriteLine(true);
2 Console.WriteLine(false);
```

Целочисленные литералы

Целочисленные литералы представляют положительные и отрицательные целые числа, например, 1, 2, 3, 4, -7, -109. Целочисленные литералы могут быть выражены в десятичной, шестнадцатеричной и двоичной форме.

С целыми числами в десятичной форме все должно быть понятно, так как они используются в повседневной жизни:

```
1 Console.WriteLine(-11);
2 Console.WriteLine(5);
3 Console.WriteLine(505);
```

Числа в двоичной форме предваряются символами `0b`, после которых идет набор из нулей и единиц:

```
1 Console.WriteLine(0b11);           // 3
2 Console.WriteLine(0b1011);          // 11
3 Console.WriteLine(0b100001);         // 33
```

Для записи числа в шестнадцатеричной форме применяются символы `0x`, после которых идет набор символов от 0 до 9 и от A до F, которые собственно представляют число:

```
1 Console.WriteLine(0x0A);            // 10
2 Console.WriteLine(0xFF);            // 255
3 Console.WriteLine(0xA1);            // 161
```

Вещественные литералы

Вещественные литералы представляют дробные числа. Этот тип литералов имеет две формы. Первая форма - вещественные числа с фиксированной запятой, при которой дробную часть отделяется от целой части точкой. Например:

```
1 3.14
```

```
2 100.001
3 -0.38
```

Также вещественные литералы могут определяться в экспоненциальной форме МЕр, где М — мантисса, Е - экспонента, которая фактически означает " $*10^{\text{п}}$ " (умножить на десять в степени), а р — порядок. Например:

```
1 Console.WriteLine(3.2e3); // по сути равно 3.2 * 103 = 3200
2 Console.WriteLine(1.2E-1); // равно 1.2 * 10-1 = 0.12
```

Символьные литералы

Символьные литералы представляют одиночные символы. Символы заключаются в одинарные кавычки.

Символьные литералы бывают нескольких видов. Прежде всего это обычные символы:

```
1 '2'
2 'A'
3 'T'
```

Также мы можем передать их вывести на консоль с помощью `Console.WriteLine`:

```
1 Console.WriteLine('2');
2 Console.WriteLine('A');
3 Console.WriteLine('T');
```

Специальную группу представляют **управляющие последовательности** Управляющая последовательность представляет символ, перед которым ставится слеш. И данная последовательность интерпретируется определенным образом. Наиболее часто используемые последовательности:

- `'\n'` - перевод строки
- `'\t'` - табуляция
- `'\\'` - слеш

И если компилятор встретит в тексте последовательность `\t`, то он будет воспринимать эту последовательность не как слеш и букву `t`, а как табуляцию - то есть длинный отступ.

Также символы могут определяться в виде шестнадцатеричных кодов, также заключенный в одинарные кавычки.

Еще один способ определения символов представляет использования шестнадцатеричных кодов ASCII. Для этого в одинарных кавычках указываются символы `'\x'`, после которых идет шестнадцатеричный код символа из таблицы ASCII. Коды символов из таблицы ASCII можно посмотреть [здесь](#).

Например, литерал `'\x78'` представляет символ `"x"`:

```
1 Console.WriteLine('\x78');    // x
2 Console.WriteLine('\x5A');    // Z
```

И последний способ определения символьных литералов представляет применение кодов из таблицы символов [Unicode](#). Для этого в одинарных кавычках указываются символы '\u', после которых идет шестнадцатеричный код Unicode. Например, код '\u0411' представляет кириллический символ 'Б':

```
1 Console.WriteLine('\u0420');    // Р
2 Console.WriteLine('\u0421');    // С
```

Строковые литералы

Строковые литералы представляют строки. Строки заключаются в двойные кавычки:

```
1 Console.WriteLine("hello");
2 Console.WriteLine("фыва");
3 Console.WriteLine("hello word");
```

Если внутри строки необходимо вывести двойную кавычку, то такая внутренняя кавычка предваряется обратным слешем:

```
1 Console.WriteLine("Компания \"Рога и копыта\"");
```

Также в строках можно использовать управляющие последовательности. Например, последовательность '\n' осуществляет перевод на новую строку:

```
1 Console.WriteLine("Привет \nмир");
```

При выводе на консоль слово "мир" будет перенесено на новую строку:

```
Привет
мир
```

null

null представляет ссылку, которая не указывает ни на какой объект. То есть по сути отсутствие значения.