IdleNumber: Документация

IdleNumber - это класс, позволяющий использовать простую арифметику с числами гораздо большими, чем могут предложить стандартные типы (максимальное целое число в C# - unsigned long.Max = 18,446,744,073,709,551,615).

С экземплярами класса можно производить простейшие арифметические операции, то есть сложение, вычитание и умножение, а также применять все операции сравнения:

```
IdleNumber first = new IdleNumber();
IdleNumber second = new IdleNumber();

var sum = first + second;
var sub = first - second;
var mul = first * 8;

if(first != second || first == second ||
first > second || first < second ||
first >= second || first <= second)
{
    Console.Write("It's Working!");
}
```

Суффикс

В стандартном виде класс имеет следующий набор суффиксов:

```
public enum Suffix
{
    None, Thousand, Million, Billion, Trillion,
    Quadrillion, Quintillion, Sextillion,
    Septillion, Octillion, Nonillion, Decillion,
    Undecillion, Duodecillion, Tredecillion
}
```

Их Вы можете изменять по своему усмотрению, например, скопировать все буквы алфавита.

Количество суффиксов в перечислении прямым образом влияет на максимально допустимое значение IdleNumber. Например, если в перечислении будет всего три значения, то максимальное значение IdleNumber будет равно 999,999,999.

> Чем больше суффиксов будет в перечислении, тем стабильнее будут работать арифметические операции.

Конструкторы

В классе IdleNumber представлено несколько конструкторов для инициализации экземпляров класса:

- *IdleNumber()* - пустой конструктор, создает экземпляр со значением 0.
- *IdleNumber(long initValue)* - создает экземпляр равный по значению поданному числу.
- *IdleNumber(string initString)* - создает экземпляр, равный по значению числу, представленному строкой. Корректно распознаются только строки без разделителей, например 123456789 или -897567312.
- *IdleNumber(int[] number)* - создает экземпляр с помощью сырого представления числа в IdleNumber. ***Не рекомендуется для использования.*** Но если все же решите, то вот инструкция: возьмем число 3,432,877,842,389. Чтобы подать его в конструктор необходимо создать массив следующего вида:{389, 842, 877, 432, 3}.

Свойства

Свойства в классе разделены на статичные и не статичные. Начнем с рассмотрения статичных свойств:

- *MaxValue* - возвращает экземпляр IdleNumber, который имеет максимально допустимое классом значение.
- *MinValue* - возвращает экземпляр IdleNumber, который имеет минимально допустимое классом значение.
- *Divider* - свойство, позволяющее устанавливать разделитель, используемый при строковом выводе числа. Ничего не возвращает.
- *ShowSuffix* - свойство, принимающее true или false, определяет будет ли печататься суффикс при строковом выводе числа.

Для обращения к свойствам используется следующая конструкция:

```
var maxValue = IdleNumber.MaxValue;
var minValue = IdleNumber.MinValue;
IdleNumber.Divider = ",";
IdleNumber.ShowSuffix = false;
```

Теперь рассмотрим не статичные свойства класса:

- *SmallText* - выводит число в урезанном виде, например, есть число 24,324,635,001 Billion. На экране появится число 24,324 Billion.
- *FullText* - выводит число в полном виде, например, есть число 24,324,635,001 Billion. На экране оно будет выведено также.
- *Negative* - определяет является ли число отрицательным, может как возвращать текущее значение, так и получать новое, таким образом положительное число можно превратить в отрицательное и наоборот без использования умножения.

> На вывод свойств *SmallText* и *FullText* напрямую влияет использование статичных свойств *Divider* и *ShowSuffix*.

Публичные методы

В классе IdleNumber у экземпляров класса есть несколько публичных методов:

- *GetNumArray()* - возвращает целочисленный массив, который является сырым представлением IdleNumber.
- *Assign(IdleNumber other)* - так как IdleNumber это класс, то присваение значения происходит по ссылке. Это значит что присвоив одному экземпляру значение другого через =, Вы будете иметь экземпляры повторящие поведение друг друга (если одно превратить в отрицательное, то же произойдет и с другим). Этот метод позволяет скопировать содержимое одного экземпляра в другой, что позволит иметь собственное поведение.
- *IsEqual(IdleNumber other)* - проверяет равны ли значение поданного числа и текущего, эквивалент сравнения ==.
- *IsGreaterThen(IdleNumber other)* - эквивалент сравнения >.
- *ConvertToLong()* - возвращает целочисленное значение типа long, если это возможно.

Публичные статичные методы

- *ConvertFromLong(long val)* - возвращает экземпляр класса IdleNumber равного значению val. Эквивалент конструктора, принимающего long в качестве параметра.
- *Abs(IdleNumber other)* - возвращает модуль получаемого числа.