New C++ experimental feature: The tadpole operators

devblogs.microsoft.com/oldnewthing/20150525-00

May 25, 2015

Raymond Chen

How often have you had to write code like this:

```
x = (y + 1) % 10;
x = (y + 1) * (z - 1);
x = (wcslen(s) + 1) * sizeof(wchar_t);
```

Since the `+` and `-` operators have such low precedence, you end up having to parenthesize them a lot, which can lead to heavily nested code that is hard to read.

Visual Studio 2015 RC contains a pair of experimental operators, nicknamed tadpole operators. They let you add and subtract one from an integer value without needing parentheses.

```
x = -~y % 10;
x = -~y * ~-z;
x = -~wcslen(s) * sizeof(wchar_t);
```

They're called tadpole operators because they look like a tadpole swimming toward or away from the value. The tilde is the tadpole's head and the hyphen is the tail.

Syntax	Meaning	Mnemonic
`-~y`	`y + 1`	Tadpole swimming toward a value makes it bigger
`~-y`	`y - 1`	Tadpole swimming away from a value makes it smaller

To enable the experimental tadpole operators, add this line to the top of your C++ file

```
#define __ENABLE_EXPERIMENTAL_TADPOLE_OPERATORS
```

For example, here's a simple program that illustrates the tadpole operators.

```
#define __ENABLE_EXPERIMENTAL_TADPOLE_OPERATORS
#include <ios>
#include <iostream>
#include <istream>

int __cdecl main(int, char**)
{
   int n = 3;
   std::cout << "3 + 1 = " << -~n << std::endl;
   std::cout << "(3 - 1) * (3 + 1) " << ~-n * -~n << std::endl;
   return 0;
}
```

Remember that these operators are still experimental. They are not officially part of C++, but you can play with them and ~~give your feedback here~~ learn more about them here.

Raymond Chen

Follow