C++ Tips
Article 1

Aether Lee
Hachioji, Japan

Abstract

We hereby start a new series about C++ programming tips in Go/Semi. In this series, we would like to introduce some C++ programming tips which can be useful for our readers. In this first article of the series, we would like to start with 2 simple ones of manipulating strings in C++. The first is to turn the value of a numeric type (integer, float, or double) variable into a string. The second is to cut string into segments by using member functions of the class itself.

1. Turn a number into a string

Consider a case that you have a value stored in a variable and you would like to make it a part of the file name which is a string so that it resembles an index. In C, a solution would be to use the function, sprintf, to "print" the value onto the string. In C++, there is a similar but easier way to do it: to utilize the "ostringstream" class.

```
#include <sstream>
#include <iostream>
using namespace std;
int main()
{
    ostringstream filename;
    unsigned int index = 18;

    filename << “file_” << index;
    cout << filename.str() << endl;

    return 0;
}
```

Execute the compiled binary executable, and the string "file_18" should be displayed.

As shown in the example, the key is to use the operator "<<" to redirect the value of the integer variable into the string as if we are printing it.

The advantage of ostringstream is that you can use manipulators with it. For example, if you want to have a heximal index in the filename, just introduce the manipulator "hex" in between the target string and the number:

```
filename << “file_” << hex << index;
```

2. Cut a string into segments

Suppose now we have a string containing information fields separated by the delimiter “_”. Again, let’s talk in the language itself.

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string info = “Addr_0x33_Data_0x1E_Ctrl_BTOFF”;

    string::size_type delimiter_pos;
    string::size_type start_pos = 0;

    do {
        delimiter_pos = info.find(‘_’, start_pos);
        if (delimiter_pos == string::npos)
            delimiter_pos = info.size() + 1;
        string field;
        field.assign(info, start_pos, delimiter_pos – start_pos);
        cout << field << endl;
        start_pos = delimiter_pos + 1;
    } while (start_pos < info.size());

    return 0;
}
```

Execute the compiled binary, and those fields of the string separated by underscores, “Addr”, “0x33”, “Data”, “0x1E”, “Ctrl” and “BTOFF”, should be printed in separate lines.

The string class provides a lot of useful functions for accessing and searching. The function “find” is just one of them. It looks for the first appearance of the character or the string to be found, and then returns the position or string::npos if it can not find any. The search starts at the position assigned by the last argument. In the first iteration of the do-while loop, 4 will be assigned to delimiter_pos. The function “assign” erases the current content and copies from the source string for a number of characters starting from the position “start_pos”. In the example, the number of characters is calculated by the distance from delimiter_pos to start_pos.

3. References

[1] Ray Lischner, *C++ in a nutshell*, O’Reilly, USA, 2003.

4. Copyright

Verigy owns the copyrights of this article