Jan 27, 2010

C++ STL Streams

Daniel Maleike

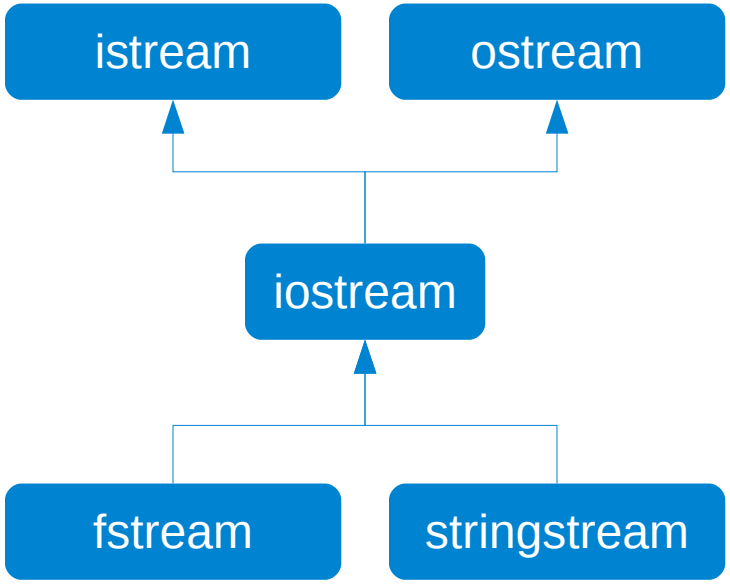
dkfz.
GERMAN
CANCER RESEARCH CENTER
IN THE HELMHOLTZ ASSOCIATION

Why Stream-IO?

- The STL way for I/O
  - Input, output, formatting, file access
  - More type-safe than printf(), scanf()
  - Extensible with user defined types (classes)
  - Inheritable, i.e. custom I/O targets, e.g. logging interface

```
void PrintPatient(std::ostream& s, const Patient& p)
{
  s << “Patient ” << p.PatientID << “ named '” << p.PatientName << “'” << std::endl;
  s << “Total body fat: ” << std::setprecision(1) << p.PatientPercentageFat << sdd::endl;
}
```

Overview




```
namespace std
{
  ostream cout;  // connected to standard output
  ostream cerr;   // connected to error output
  istream  cin;    // connected to standard input
}
```

- Output to any instance of std::ostream
- Input from any instance of std::istream

Identical code for files, strings, console, ...

Reading from file

```
#include <fstream>

std::ifstream numberFile("c:/my/files/numb.ers"); // or numberFile.open(..)
if (numberFile.fail()) return;

std::vector<int> numbers;
int i;

while ( numberFile >> i ) // fails on EOF or INVALID integer (type-safety)
{
  std::cout << "Read " << i << " from file" << std::endl;
  numbers.push_back(i);
}

numberFile.close();
```

Stream status

if (stream)	previous operation was successful, same as !stream.fail()
if (stream.fail())	previous operation failed
if (stream.eof())	reading past end of stream attempted
if (stream.bad())	stream state undefined; cannot be used anymore!
if (stream.good())	none of bad/eof/fail

Formatting for input and output

- Formatting is influenced by a number of flags
- Two versions for setting flags
  - stream.setf(std::ios_base::boolalpha); // write bool values as “true/false”
  - stream << std::ios_base::boolalpha << true;

```
boolalpha, noboolalpha      // 0/1 or false/true
showbase, noshowbase        // 0xA3 or A3
showpoint, noshowpoint      // 1 or 1.
showpos                     // 1 or +1
uppercase                   //  1e+3 or 1E+3
setprecision(int n)
setbase(int n)
...
width(int n), fill(char c), left, right, internal
```

String streams

- Can be used to format numbers into strings

```
#include <sstream>

std::stringstream stream;

str << “Formatted number: ” << std::fixed << std::setprecision(2) << 5.1233;
std::string s( stream.str() );  // s == “5.12”
```

Locales – nice formatting and potential source of errors

A “locale” collects the language / culture specific representations of

- numbers, number punctuation (decimal point, thousands separator)
- time, date, weekdays, currency symbols and formatting
- character collation (order of letters for comparisons: ä > å ?)

Examples for locale names:

- “C”, “POSIX”
- “de_DE”, “de_DE.utf8”, “de_DE@euro”, “lang_german_Germany”

```
#include <locale>
std::locale currentLocale( std::cout.getloc() ); // current output locale (e.g. “de_DE”)
std::cout << 5.43 << std::endl;  // “5,43”
std::locale cLocale( “C” );
std::cout.imbue( cLocale );
std::cout << 5.43 << std::endl;  // “5.43”
```

Custom data types – output

```
#include <iostream>

class Fred {
  public:
    friend std::ostream& operator<< (std::ostream& o, const Fred& fred);
};

std::ostream& operator<< (std::ostream& o, const Fred& fred)
{
  return o << “One Fred at “ << (void*) this;
}

int main() {
  Fred f;
  std::cout << "My Fred object: " << f << "\n";
}
```

Custom data types – input

```
#include <iostream>

class Fred {
  public:
    friend std::istream& operator>> (std::istream& o, Fred& fred);
  private:
    int filling;
};

std::istream& operator>> (std::istream& i, Fred& fred)
{
  return i >> fred.filling;
}
```

Outlook

- Poco library uses streaming for
  - encryption/decryption (CryptoIOS)
  - compression (ZipIOS)
  - socket communication (SocketIOS)
  - ...

- (similar streaming interfaces in Qt)

Useful resources

- Tutorial
  http://www.informit.com/articles/article.aspx?p=170770

- STL references
  http://www.cplusplus.com/reference/iostream/
  http://www.cppreference.com/wiki/

- <iostream>-FAQ
  http://www.parashift.com/c++-faq-lite/input-output.html

- List of POSIX locales
  `locale -a` in any terminal

- MSDN about Windows locales
  http://msdn.microsoft.com/en-us/library/hzz3tw78.aspx