An Introduction to the C++ Network Library

Glyn Matthews

Belgian C++ User Group

INTRODUCTION

NETWORK PROGRAMMING IN C++

C++ NETWORK LIBRARY (0.9.4)

C++1Y

THE FUTURE

About me

- Glyn Matthews
- Software Engineer and ScrumMaster at SoftKinetic, Brussels
- Developer and Project Administrator C++ Network Library
- http://glynos.github.com/
- @glynos

What is the C++ Network Library?

- A collection of libraries for HTTP and application-level protocols
- Network types (URI)

Hosted on Github:

http://cpp-netlib.github.com/

Hello, world! (HTTP Client)

```
using namespace boost::network;
using namespace boost::network::http;

int main(int arg, char *argv[]) {
 client::request
req("http://127.0.0.1:8000/");
 req << header("Connection", "close");
 client c;
 client::response res = c.get(req);
 std::string b = body(res);
}
```

Hello, world! (HTTP Server)

```
namespace http = boost::network::http;

struct handler;
typedef http::server<handler> http_server;

struct handler {
 void operator() (http_server::request const &req,
                  http_server::response &res) {
  res = http_server::response::stock_reply(
   http_server::response::ok,
   "Hello, world!");
 }
};
```

Hello, world! (HTTP Server)

```
int main(int arg, char *argv[]) {
 handler h;
 http_server s("0.0.0.0", "8000", h);
 s.run();
 return 0;
}
```

History

- Started in 2007 by Dean Michael Berris
- Header-only HTTP Client
- Later developed a header-only HTTP Server
- Added a URI class

Objectives

- To develop a high quality, easy-to-use C++ networking library
- To enable developers to extend the library
- To lower the barrier of entry for cross-platform, network-aware C++ applications

INTRODUCTION

NETWORK PROGRAMMING IN C++

C++ NETWORK LIBRARY (0.9.4)

C++1Y

THE FUTURE

Network libraries in C++

- POCO http://pocoproject.org/
- Qt http://qt-project.org/
- libcurl http://curl.haxx.se/libcurl/
- pion http://github.com/cloudmeter/pion

Boost.Asio

- Low level I/O programming
- Portable socket implements
- TCP, UDP, ICMP

Boost.Asio

- C++ Network Library is built on top of Boost.Asio

INTRODUCTION

NETWORK PROGRAMMING IN C++

C++ NETWORK LIBRARY (0.9.4)

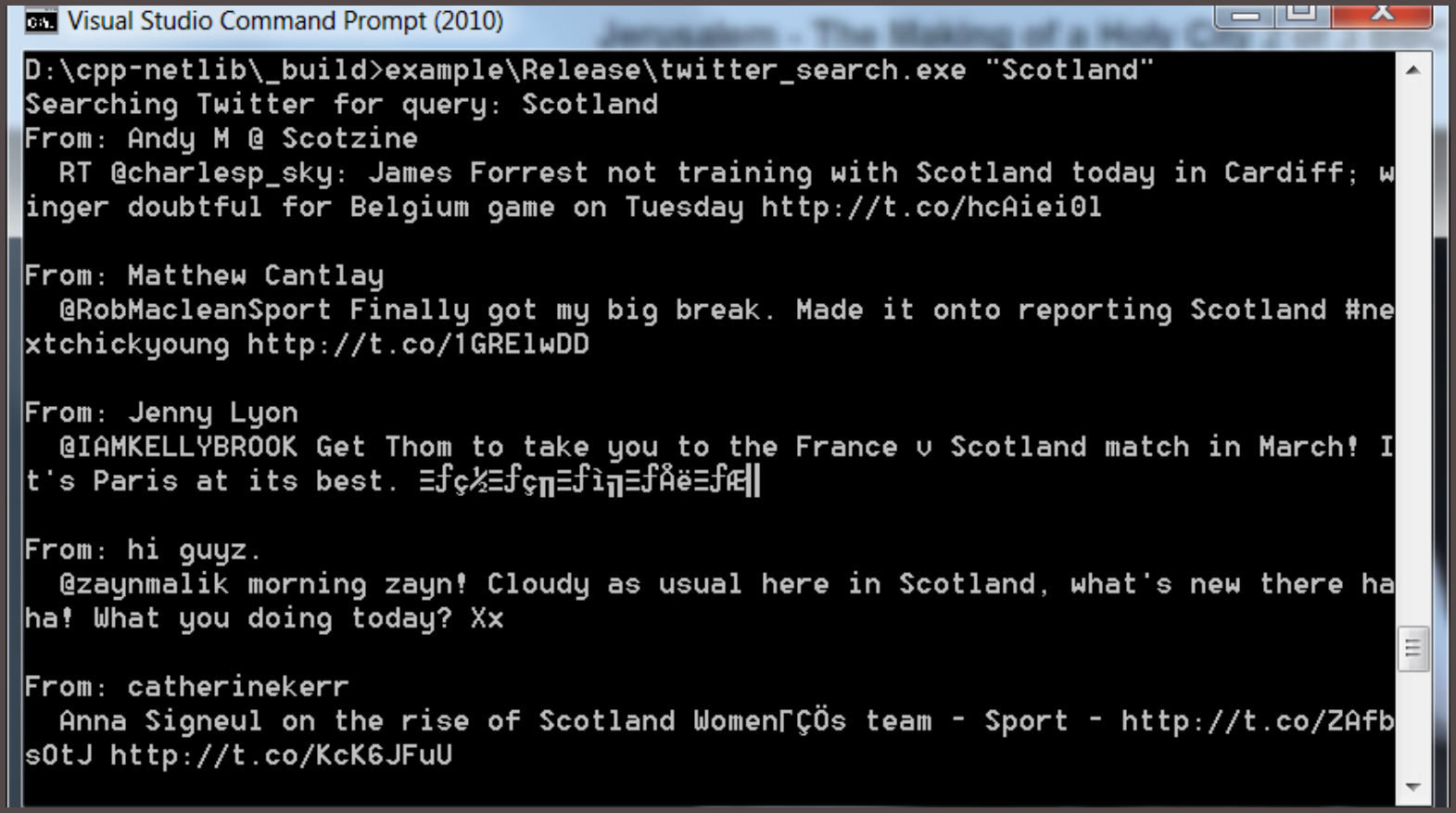
C++1Y

THE FUTURE

Twitter API Example

```
uri::uri
 url("http://search.twitter.com/search.json");
uri::builder builder(url);
builder.query("q", uri::encode(argv[1]));

http::client c;
http::client::request req(url);
http::client::response res = client.get(req);
```



Console output of `twitter_search` example

RSS Feed Example

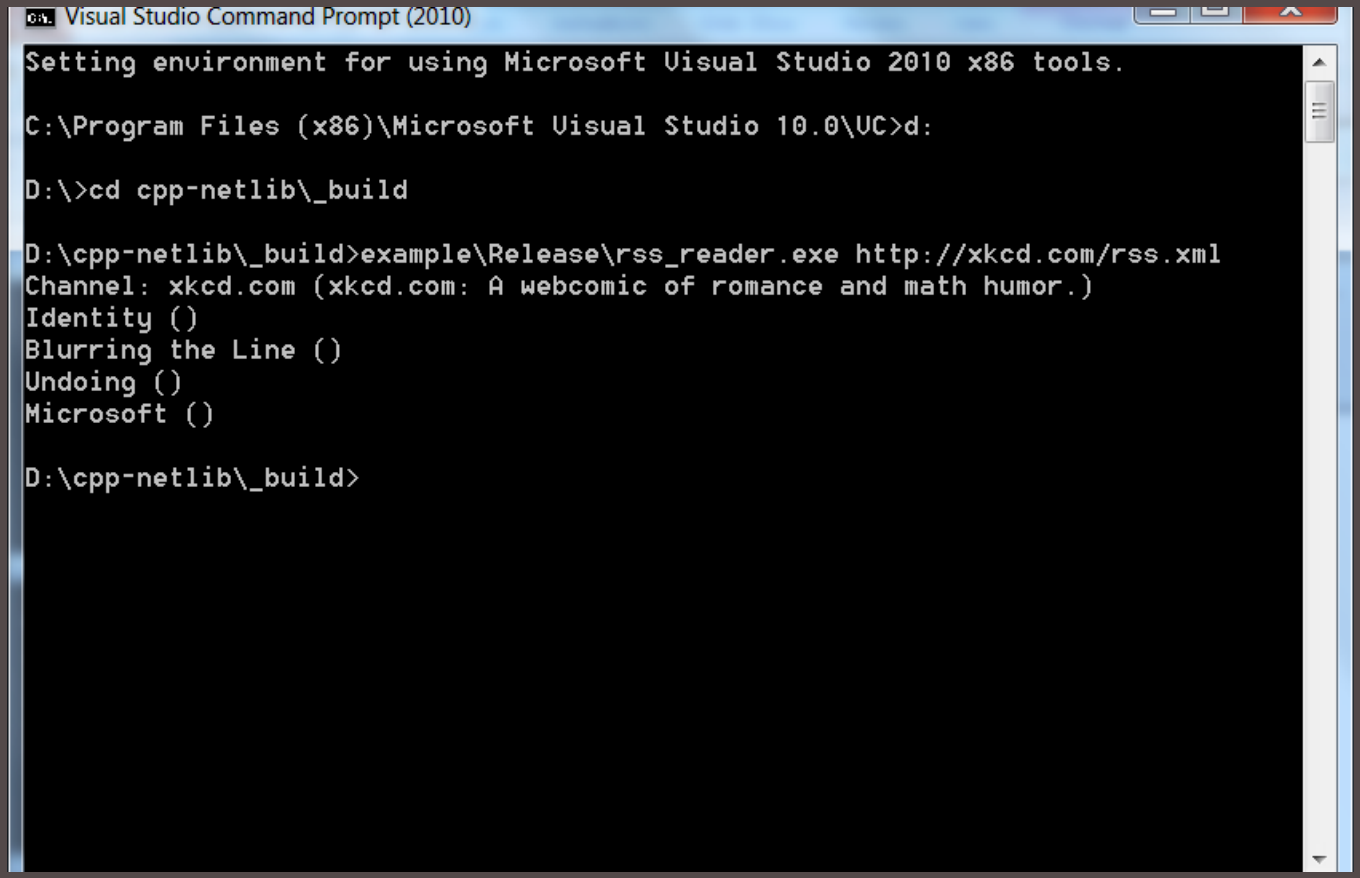
```
using namespace boost::network;
http::client client;
http::client::request req(argv[1]);
request << header("Connection", "close");
http::client::response res = client.get(req);
```

RSS Feed Example

```
rss::channel chan(response);
std::cout << "Channel: " << chan.title()
          << " (" << chan.description() << ")"
          << std::endl;
for (rss::item const &item; chan) {
 std::cout << item.title()
           << " (" << item.author() << ")"
           << std::endl;
}
```

RSS Feed Example

```
using namespace boost::network;
class channel {
public:
 channel(http::client::response const &res) {
  std::string response_body = body(response);
  rapidxml::xml_document<> doc;
  doc.parse<0>(const_cast<char *>(
                response_body.c_str()));
};
```



Console output of RSS Feed example

Issues with 0.9.4

- Configuration options are limited in HTTP client
- Lack of timeout support in HTTP client
- Lack of asynchronous API in HTTP client
- `boost` namespace (we're not a part of boost)

Issues with 0.9.4

- Still too large a burden on users for HTTP server implementation
- Lack of session support in HTTP server
- …

Boost

- Potential long review and post-review process
- SVN and integration process
- Missing support for useful auxiliary libraries (XML, JSON, crypto)
- Issues with licenses for auxiliary dependencies

C++11

Useful features:

- Type inference (`auto`)
- String literals and multi-byte strings
- Move semantics
- Regular expressions

C++11

Useful features:

- Extended iterator support (`std::begin`, `std::end`)
- Concurrency support

INTRODUCTION

NETWORK PROGRAMMING IN C++

C++ NETWORK LIBRARY (0.9.4)

C++1Y

THE FUTURE

C++1y and SG4

- The process for the next round of standardization is already under way
- A study group (SG4) was created with the goal to standardize a set of network libraries
- The target is 2017

SG4

The standardization effort will initially focus on basic socket layer functionality:

- IP v4 / IP v6 Addresses
- TCP/UDP sockets
- URI
- SSL Interface

C++1y URI Proposal

The `network::uri` class forms the basis of a proposal to the C++ standard library

`network::uri` will track the proposal as it evolves.

C++ standard SG4

http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2012/n3420.html

URI

```
#include <network/uri>
class std::network::uri;
class std::network::uri::builder;
```

URI

```
std::network::uri
 uri("http://www.becpp.org/blog/");
std::cout << uri.scheme() << std::endl
          << uri.host() << std::endl
          << uri.path() << std::endl;
```

URI

Output:

```
http
www.becpp.org
/blog/
```

URI Builder

```
std::network::uri uri;
std::network::uri::builder builder(uri);
builder.scheme("http")
       .host(“www.becpp.org")
       .path("/");
assert("http://www.becpp.org/" == uri);
```

INTRODUCTION
NETWORK PROGRAMMING IN C++
C++ NETWORK LIBRARY (0.9.4)
C++1Y
THE FUTURE

Vision

- Abandon submission to Boost
- Focus on C++1y
- Develop application-level protocols

New HTTP Client API

```
struct client {
 enum class method
  { GET, PUT, POST, DELETE, OPTIONS };
 client(client_options);
};
```

New HTTP Client API

```
struct client {
 std::future<response> get(request,
                           request_options);
 std::future<response> put(request,
                           request_options);
 std::future<response> post(request,
                            request_options);
 std::future<response> delete_(request,
                             request_options);
};
```

New HTTP Server API

```
template <
 class Handler,
 class SessionManager,
 class Authenticator,
 class ConnectionManager>
struct basic_server;

typedef basic_server<> server;
```

Extending C++ Network Library

- Logging
- HTTP Web Services
- SNMP
- (E)SMTP
- FTP
- XMPP

Request for Volunteers

We need:

- Protocol implementations
- Users
- Applications and examples

Request for Volunteers

We need:

- Testers
- Documentation

Thank You!

glyn.matthews@gmail.com

http://github.com/cpp-netlib/