URI and UUID

Identifying things on the Web.

applied informatics

Overview

- Uniform Resource Identifiers (URIs)
- URIStreamOpener
- Universally Unique Identifiers (UUIDs)

Uniform Resource Identifiers

- Uniform Resource Identifiers (RFC 3986) are used to identify resources on the Web.
- POCO provides the POCO::URI class that can be used for building and storing URIs and to split URIs into their components.

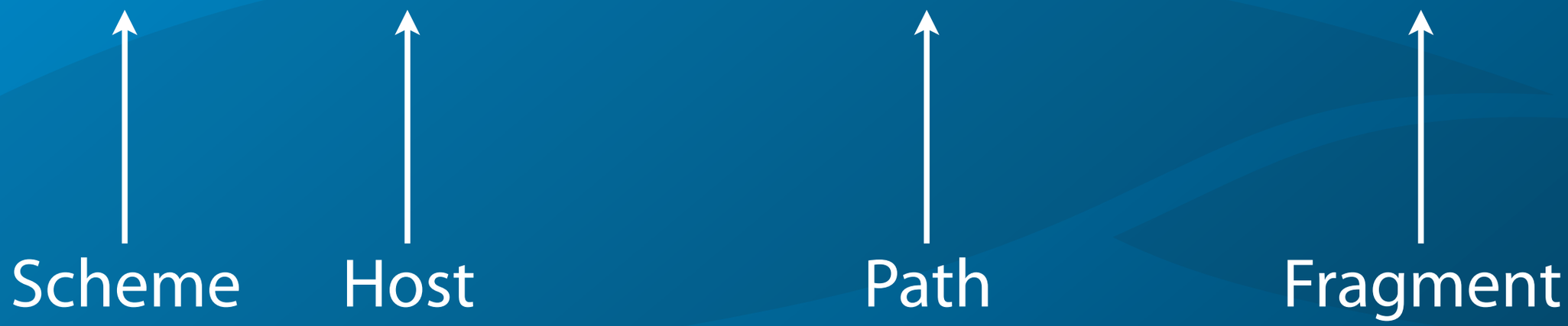
```
#include "Poco/URI.h"
```

URI Structure

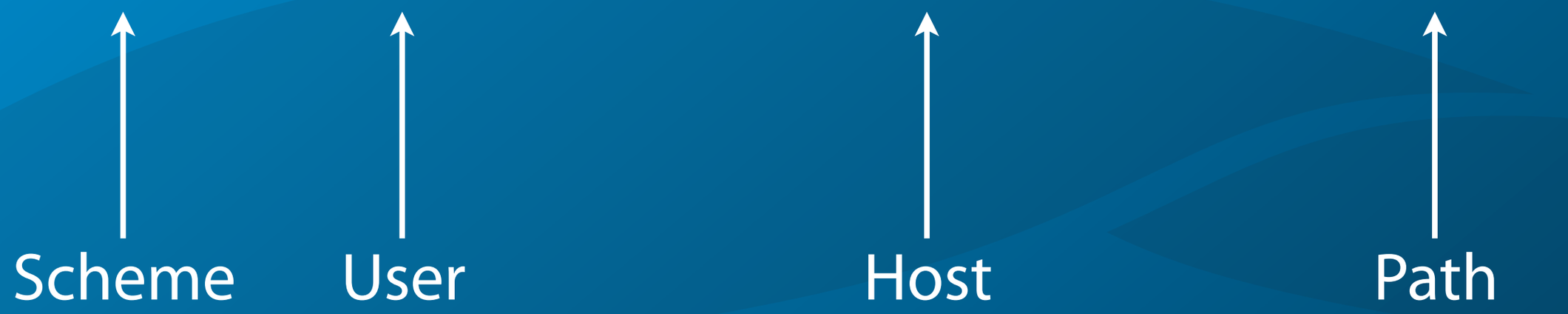
- An URI consists of:
  - a Scheme (or protocol name) part
  - an Authority part
    (user information, host name and port number)
  - a Path part
  - a Query part
  - and a Fragment part

URI Examples

URI Examples (cont'd)



URI Examples (cont'd)



```
#include "Poco/URI.h"
#include <iostream>

int main(int argc, char** argv)
{
    Poco::URI uri1("http://www.appinf.com:88/sample?example-query#frag");

    std::string scheme(uri1.getScheme());    // "http"
    std::string auth(uri1.getAuthority());   // "www.appinf.com:88"
    std::string host(uri1.getHost());        // "www.appinf.com"
    unsigned short port = uri1.getPort();    // 88
    std::string path(uri1.getPath());        // "/sample"
    std::string query(uri1.getQuery());      // "example-query"
    std::string frag(uri1.getFragment());    // "frag"
    std::string pathEtc(uri1.getPathEtc()); // "/sample?example-
    query#frag"

    Poco::URI uri2;
    uri2.setScheme("https");
    uri2.setAuthority("www.appinf.com");
    uri2.setPath("/another sample");
    std::string s(uri2.toString());
        // "https://www.appinf.com/another%20sample"
```

```
	std::string uri3("http://www.appinf.com");
	uri3.resolve("/poco/info/index.html");
	s = uri3.toString(); // "http://www.appinf.com/poco/info/index.html"

	uri3.resolve("support.html");
	s = uri3.toString(); // "http://www.appinf.com/poco/info/support.html"

	uri3.resolve("http://sourceforge.net/projects/poco");
	s = uri3.toString(); // "http://sourceforge.net/projects/poco"

	return 0;
}
```

URIStreamOpener

- Poco::URIStreamOpener is used to create and open input streams for resources identified by URIs.
- #include "Poco/URIStreamOpener.h"
- For every URI scheme used, a subclass of Poco::URIStreamFactory must be registered.
- POCO provides stream factories for files, HTTP, HTTPS and FTP resources.

```
#include "Poco/URIStreamOpener.h"
#include "Poco/Net/HTTPStreamFactory.h"
#include "Poco/Net/FTPStreamFactory.h"
#include <memory>

int main(int argc, char** argv)
{
    Poco::Net::HTTPStreamFactory::registerFactory();
    Poco::Net::FTPStreamFactory::registerFactory();

    Poco::URIStreamOpener& opener =
    Poco::URIStreamOpener::defaultOpener();

    std::auto_ptr<std::istream> istr1(
        opener.open("http://www.appinf.com/index.html")
    );
    std::auto_ptr<std::istream> istr2(
        opener.open("ftp://ftp.appinf.com/pub/poco/poco-1.2.5.tar.gz")
    );
    std::auto_ptr<std::istream> istr3(
        opener.open("file:///usr/include/stdio.h")
    );
    return 0;
}
```

UUIDs

- A UUID (Universally Unique Identifier) is an identifier that is unique across both space and time, with respect to the space of all UUIDs.
- Three flavors:
  - time-based
  - name-based
  - random

The UUID Class

- Poco::UUID stores a UUID, supporting full value semantics including all relational operators.
- #include "Poco/UUID.h"
- UUIDs can be converted to and from strings.

The UUIDGenerator Class

- Poco::UUIDGenerator is used to create UUIDs.
- #include "Poco/UUIDGenerator.h"
- UUIDs can be created
  - time based (ethernet MAC address + timestamp)
  - name-based (usually from an URI)
  - random

```cpp
#include "Poco/UUID.h"
#include "Poco/UUIDGenerator.h"
#include <iostream>

using Poco::UUID;
using Poco::UUIDGenerator;

int main(int argc, char** argv)
{
    UUIDGenerator& generator = UUIDGenerator::defaultGenerator();

    UUID uuid1(generator.create()); // time based
    UUID uuid2(generator.createRandom());
    UUID uuid3(generator.createFromName(UUID::uri(), "http://appinf.com");

    std::cout << uuid1.toString() << std::endl;
    std::cout << uuid2.toString() << std::endl;
    std::cout << uuid3.toString() << std::endl;

    return 0;
}
```

applied informatics

www.appinf.com | info@appinf.com
T +43 4253 32596 | F +43 4253 32096