Platform and Environment

Getting information about the system you're running on.

Overview

- Getting information about the system at build time
- Getting information about the system at run time
- Working with environment variables

System Information at Compile Time

- POCO provides a set of macros that can be used to determine the platform the code is going to run on.
- These macros can be used to determine:
  - the operating system, and
  - the processor architecture,
- and are defined in Poco/Platform.h, which is automatically included by Poco/Foundation.h.

Determining the Operating System

> The POCO_OS macro can be used to determine the operating system. It will have one of the following values:

POCO_OS_AIX	POCO_OS_LINUX	POCO_OS_SOLARIS
POCO_OS_CYGWIN	POCO_OS_MAC_OS_X	POCO_OS_TRU64
POCO_OS_FREE_BSD	POCO_OS_NET_BSD	POCO_OS_VMS
POCO_OS_HPUX	POCO_OS_OPEN_BSD	POCO_OS_VXWORKS
POCO_OS_IRIX	POCO_OS_QNX	POCO_OS_WINDOWS_NT

Note: See the Poco/Platform.h header for current values.

```
#include "Poco/Foundation.h"
#if POCO_OS == POCO_OS_WINDOWS_NT
    // do the Windows thing
#elif POCO_OS == POCO_OS_LINUX
    // do the Linux thing
#endif
```

Determining the Operating System (cont'd)

- If you just want to test whether you are compiling for a Windows platform, you can check if POCO_OS_FAMILY_WINDOWS is defined.
- The same for Unix platforms: POCO_OS_FAMILY_UNIX

```
#include "Poco/Foundation.h"
#if defined(POCO_OS_FAMILY_WINDOWS)
    // do the Windows thing
#elif defined(POCO_OS_FAMILY_UNIX)
    // do the Unix thing
#endif
```

Determining the Hardware Architecture

> The POCO_ARCH macro can be used to determine the hardware architecture. It will have one of the following values:

POCO_ARCH_ALPHA
POCO_ARCH_AMD64
POCO_ARCH_ARM
POCO_ARCH_HPPA
POCO_ARCH_IA32
POCO_ARCH_IA64
POCO_ARCH_MIPS
POCO_ARCH_POWER
POCO_ARCH_PPC
POCO_ARCH_SPARC

Note: See the Poco/Platform.h header for current values.

```
#include "Poco/Foundation.h"
#if POCO_ARCH == POCO_ARCH_IA32
    // do the Intel 32-bit thing
#elif POCO_ARCH == ARM
    // do the ARM thing
#endif
```

Byte Order

- POCO has facilities to deal with byte order issues.
- Macros to determine the current host's byte order:
  - POCO_ARCH_LITTLE_ENDIAN
    macro is defined if architecture is little endian
  - POCO_ARCH_BIG_ENDIAN
    macro is defined if architecture is big endian

System Information at Run Time

- The Poco::Environment class has static functions to determine system and environment information at run time.
- #include "Poco/Environment.h"
- std::string get(const std::string& name)
  Return the value of an environment variable. Throws a Poco::NotFoundException if the variable is undefined.
- bool has(const std::string& name)
  Check whether an environment variable is defined.
- void set(const std::string& name, const std::string& value)
  Set the value of an environment variable.

System Information at Run Time (cont'd)

- std::string osName()
  Return the name of the operating system (uname).
- std::string osVersion()
  Return the version of the operating system (uname -r).
- std::string osArchitecture()
  Return a string describing hardware architecture (uname -m).
- std::string nodeName()
  Return the computer name (uname -n).
  Note: there's also Poco::DNS::hostName() which is a wrapper for gethostname().

System Information at Run Time (cont'd)

- std::string nodeId()

  Return the ethernet address of the first ethernet adapter found on the system in format xx:xx:xx:xx:xx:xx (or all zeros if there is no ethernet adapter).

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

using Poco::Environment;

int main(int argc, char** argv)
{
    std::cout
        << "OS Name: " << Environment::osName() << std::endl
        << "OS Version: " << Environment::osVersion() << std::endl
        << "OS Arch: " << Environment::osArchitecture() << std::endl
        << "Node Name: " << Environment::nodeName() << std::endl
        << "Node ID: " << Environment::nodeId() << std::endl;

    if (Environment::has("HOME"))
        std::cout << "Home: " << Environment::get("HOME") << std::endl;

    Environment::set("POCO", "foo");

    return 0;
}
```

applied informatics

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