Tourist Installation Guide

Dated: October 11, 2007

Version	Description of Change	Author	Date
0.1	First Draft	Faisal Khan	10 – 11 - 2007

1 Introduction

This document walk you through the steps required to get Tourist code, compile and run its test cases. To get more information about working with Tourist code, consult 'Tourist Documentation' – which is still to be written.

2 Platform

Currently, we have tested Tourist only on *NIX based system. The Linux distribution which were specifically tested are:

- Red Hat Linux Enterprise 3 & 4
- Fedora Core 6 & 7
- Ubuntu 7.04

3 Dependencies

Tourist uses heavily components from Portable Components (POCO) library. It is BOOST like library providing cross platform reusable modules. The use of POCO will ensure the availability of Tourist across wide verity of platforms.

Generally, you need to have below mentioned tools installed before compiling Tourist:

- POCO
- Cmake
- Monotone – if you need to use code repository

3.1 POCO

So, as a first step you need to download and compile the POCO. Download the latest tar of poco from http://www.appinf.com/poco/download/ . Here are the steps to compile and install POCO.

```
$ tar -xzvf poco-xxx.tar.gz
$ cd poco-xxx
$ ./configure
$ make
$ make install
```

Note: Make sure to install POCO libraries and header files under /usr/local – default location. This limitation will be removed as we will improve our compile scripts.

Note: We need these components from POCO: libPocoFoundation libPocoNet libPocoUtil libCppUnit

3.2 CMake

Cmake (http://www.cmake.org/HTML/Install.html) is a cross platform build environment that let you configure code compilation using compiler independent scripts. In short, we used it to make our life easy when it comes to configuring source across platforms.

Get the latest release of Cmake from http://www.cmake.org/HTML/Download.html and follow instruction given at http://www.cmake.org/HTML/Install.html to install it. Usually CMake installation is pretty straight forward.

3.3 Monotone

If you are interested in downloading code from the code repository, you will require monotone. Get monotone from http://monotone.ca/. Tourist code is also available as an archive file in case you don't want to download it from repository.

4 Tourist Compilation

4.1 Fetching from Repository

We use monotone repository for managing Tourist code. So, one way to get tourist is fetching from monotone. Assuming that you have successfully installed monotone client, follow these steps to download the code.

```
$ mtn db init  --db=tourist_db
$ mtn --db=tourist_db  pull newman.ultralight.org default
$ mtn –db=tourist_db co --branch=Tourist
```

This will place the Tourist complete code inside 'Tourist' directory.

4.2 Fetching from tar

A latest snapshot of Tourist code can be downloaded from
http://julian.ultralight.org/~faisal/tourist_latest.tar

4.3 Compilation

Make sure that you have fulfilled all the dependencies mentioned in section 3. The build script (Makefile) for Tourist is generated by running CMake. Here are the steps you should follow to genreate Makefile and build tourist.

```
$ cd <tourist directory>
$ mkdir build
$ cd build
$ cmake ../ -DDEBG=1
$ make
```

'-D' switch is used to set configuration parameters for cmake. Here are some of the parameters defined for tourist.

- DEBUG enable (value = 1) debug symbols.

Note: Watch out for cmake errors suggesting some undefined variable as it usually means that cmake is not able to find a particular library or include directory on your system. We hope to make our build scripts more intelligent soon.

4.4 Testing

There are handful of unit test available for almost all of the components of Tourist. Test for each component is available under its' 'testsuite' directory. Here is the generic guideline to run tests.

```
$ cd <tourist directory>
# cd build
$ ./<component>/testsuite/testrunner_<component> -all
```

Test for these component is available with the syntax to run its unit test assuming you are under 'build'

directory':

- Core - ./Core/testsuite/testrunner -all
- Message - ./Message/testsuite/testrunner_message -all
- Net - ./Net/testsuite/testrunner_net -all

Note: You can always run an individual test by giving the name of it instead of '-all'. Use '-print' option to see what kind of tests are available.