NAME

README – Buildtool installation process description

WELCOME TO BUILDTOOL

Buildtool is a portable build infrastructure. From developer's point of view, it provides all tools needed to create powerful source code autoconfiguration scripts and simple makefiles. For the user, this simplifies the process of building and installing third party packages in his own system.

You are encouraged to read this document entirely before starting to play with the Buildtool package, as it contains several things you need to know beforehand.

Remember that you can get the latest Buildtool version, read the manual online and get free support from the official website:

```
http://buildtool.sourceforge.net/
```

LICENSE

Buildtool is licensed under the terms of the BSD license. Please read the `COPYING` file for more details on this topic.

BUILDING AND INSTALLING

Buildtool's build process can be summarized as follows: configure the sources, build them and install the results.

If you have trouble during the process, the section **SYSTEM SPECIFIC NOTES** contains some tips that may help you.

Configuring

Source code configuration is done by the configure script, in a completely automated way. At a very minimun you will need a C compiler, like `cc`(1), a `make`(1) tool, a `sh`(1) interpreter, a lexical analyzer generator, like `lex`(1), and a LARL parser generator, like `yacc`(1). All these tools are standard, so you shouldn't worry about them. If the script is unable to find any of them, it will stop and give you a chance to solve the problems. You can easily launch the configure process by issuing:

```
$ ./configure
```

This is the simplest use of this script, which will work most of the times. Although there are many user-configurable options that you can look at using the **-h** command-line switch. Probably, the most used flag is **-p**, which specifies the installation prefix. The script will also create several makefiles, ready to be parsed by `make`(1).

Building

You can start the build by issuing:

```
$ make
```

Generated makefiles should work with any `make`(1) flavour, as they do not use any kind of advanced features. Environment variables will not affect this step. If they contain incorrect values, you will need to set them in the environment and execute the configuration script again.

Installing

The installation is as simple as all other steps. Depending on the prefix you chose while configuring sources, you will need to be `root` to do this step (note that you can do the installation as a regular user if you have enough privileges on the target directory). Issue:

```
$ make install
```

And that's all! If you used a non-standard prefix you will want to setup your path accordingly. No other change is required in your environment, as Buildtool remembers where it was installed and calls its modules with full paths.

What to do if build fails

If the build process fails in some step, please report it! We must know the problem in order to solve it.

The first thing you must do to provide a good bug report is to take a log of all the process you did. To achieve this you can use the `script`(1) command; this utility starts a subshell where you can run commands (as you usually do), but all output is redirected to a log file.

As a summary, you should do something like this (note that output is removed for simplicity):

```
$ script build.log
Script started, output file is build.log
$ uname -a
$ make cleandir
$ ./configure -p /tmp/debug
$ make
$ make install
$ exit
Script done, output file is build.log
```

Once this is done, send me the `build.log` file, at the following email: Julio M. Merino Vidal ⟨jmmv@users.sourceforge.net⟩. If you can add additional information to this log (that you think can be useful to us), please do so.

SYSTEM SPECIFIC NOTES

This section contains several notes about building and installing Buildtool on specific systems.

Win32 (using Cygwin)

Buildtool can work under Win32 systems using the Cygwin environment. You can download the installer from:

```
http://www.cygwin.com/
```

When you are asked to select which packages you want to install, select (aside from the defaults): bzip2, cygutils, less, make and gcc.

USING BUILDTOOL

All Buildtool commands are run through a small wrapper which selects the appropiate module and its flags each time. To read a brief summary of commands, issue:

```
$ buildtool -h
```

If you are an end-user, who only needs to use Buildtool to configure, build and install third party programs, you will not need to learn many details. Start by reading the `README.bt` file included in any Buildtoolized package. All further instructions are included there, as well as some documentation references.

If you are a developer interested in using Buildtool in your own programs, you will need to learn and understand several concepts. The installed `buildtool`(1) manpage provides a good starting point. The `http://buildtool.sourceforge.net/docs.html` page contains all information available.

TESTING

As Buildtool is still in alpha development state, it is very important that you do some basic testing. Buildtool comes with Buildtool's TestSuite, a sample package designed to see how Buildtool performs in your system.

To start the testing process, launch the following command from the directory where you built Buildtool:

```
$ make test
```

Remember that you will probably need less than 10 minutes to read the file and do all the tests. But you will be helping us a lot by providing porting related information.

SUPPORT

For technical support subscribe and post to Buildtool's mailing lists. See `http://buildtool.sourceforge.net/mlists.html` for a list of available lists.

AUTHORS

This document was written by Julio M. Merino Vidal. For a complete list of authors and contributors, please look at the *PEOPLE* file included with Buildtool's distribution.