Release Notice
BlueGnu Testing Framework
Version 2.0.3

Jan-Willem Neurdenburg
jotOmega dsc
56 Brigham Hill Road
Grafton MA 01519-1135
neurdenburgj@acm.org
Tel: (508) 839-0276
Fax: (508) 839-7267

September 19, 1999

1 Introduction

BlueGnu is a framework for testing other programs. It has been created to be compatible with DejaGnu. Its purpose is to provide a single front end for all tests. Beyond this, BlueGnu offers several advantages for testing:

- The flexibility and consistency of the BlueGnu framework makes it easy to write tests for any program, with the exception of GUI applications.
- BlueGnu provides a layer of abstraction, which makes all tests (if correctly written) portable to any host or target where a program must be tested.
- BlueGnu is written in [incr Tcl], which in turn is based on Tcl (Tool Command Language). The framework comprises two parts:

  1. the testing framework,

2. the test-suites or test-sets themselves.

- BlueGnu will work with any Tcl based interpreter as long as [incr Tcl] has been included. You can include 'expect', 'Tk', and/or other extensions.
- Includes DejaGnu release 1.6

2 Requirements

The following modules should have been installed, before you can install and/or use BlueGnu:

- Tcl release 8.0 or higher,
- incr Tcl release 3.0 or higher.

Any other extensions that is compatible with Tcl release 8.0 can be used as well.

3 Structure and Contents of the Release

The root directory of the release contains the README files with installation instructions and the files needed to build and install this product. It also contains the executable scripts of the BlueGnu testing framework.

The top-level directories are listed below:

lib: the packages and procedures that make the BlueGnu and DejaGnu testing framework. This also includes the default target definition files.

testsets: the BlueGnu test-suites and test examples. It contains the following subdirectories.

> **BlueGnu:** test scripts to test the testing framework itself.
>
> **examples:** test suite and test script examples.
>
> **config, lib, tools:** currently empty, but can be used for test-set dependent configuration files, library files, and tools.

config: currently empty.

doc: the DejaGnu texinfo source and the documentation in 'info', 'dvi', 'ps', and 'pdf' representation, respectively dejagnu.info*, dejagnu.dvi, dejagnu.ps, and dejagnu.pdf. A DejaGnu man page is also available.

It also contains the TEX version (README.tex) of this document as well as the 'dvi', 'ps' 'html', and 'pdf' representation, respectively notice.dvi, notice.ps, notice.html, and notice.pdf.

testsuite: contains a mixture of DejaGnu and BlueGnu test scripts.

contrib: contains examples how DejaGnu is used at Cygnus.

example: contains a full DejaGnu test framework example for testing the program 'calc' which is also included.

4 Installation and use under Unix

Before you can install and use BlueGnu you need to have installed the following three packages:

- Tcl version 8.0.3
- Tk version 8.0.3
- incr Tcl version 3.0.1

The source for these packages should all be located in one directory. The subdirectory in the directory should be:

- tcl8.0.3
- tk8.0.3
- itcl3.0.1

The following examples use the command './configure –prefix=/tools/...'. This will install all packages in a directory "/tools". When you omit the "–prefix"-switch then the installation default will be the directory "/usr/local".

4.1 Installation of needed Packages

When you have not installed Tcl and the other needed extensions, then you need to retrieve the sources from "www.tcltk.com/itcl". You need to 'gunzip' the files and do a 'tar xf' of all these packages in one directory, let's call this directory "TclTk".

From the directory "TclTk", you should do the following to install the packages:

```
% cd tcl8.0.3/unix
% ./configure --prefix=/tools/tcl8.0.3 --enable-gcc --enable-shared
% make
% mkdir /tools/tcl8.0.3
% make install
% cd ../../tk8.0.3/unix
% ./configure --prefix=/tools/tk8.0.3 --enable-gcc --enable-shared
% make
% mkdir /tools/tk8.0.3
% make install
% cd ../../itcl3.0.1
% ./configure --prefix=/tools/itcl3.0.1 --enable-gcc --enable-shared
% make
% mkdir /tools/itcl3.0.1
% make install
```

4.2 Installing BlueGnu

You can now 'gunzip' and 'tar xf' the BlueGnu version 2.0.3 in the directory "TckTk". This will create the directory "bluegnu2.0.3". Now do the following:

```
% cd bluegnu2.0.3
% ./configure --prefix=/tools/bluegnu2.0.3
% make
% mkdir /tools/bluegnu2.0.3
% make install
```

This will install BlueGnu in the directories:

- /tools/bluegnu2.0.3/bin
- /tools/bluegnu2.0.3/lib/bluegnu
- /tools/bluegnu2.0.3/info
- /tools/bluegnu2.0.3/man

4.3 Using BlueGnu

When you have installed [incr Tcl] and BlueGnu and you have the respective "bin" directories in your PATH variable, then you can start running some tests. You can go into the BlueGnu source directory "bluegnu2.0.3/testsets/examples" and run the following:

```
% bluegnu versionTcl.itcl
% bluegnu ts_001
% bluegnu ts_002
% bluegnu ts_003
```

The above test result should all be PASS. The following test will give a result UNKNOWN, because no pass/fail instruction have been given.

```
% bluegnu tc001
```

The last test you can run will fail in its simple form:

```
% bluegnu tc002
```

But will pass if you execute the test as follows:

```
% bluegnu tc002[English]
```

This is because the test scripts need a test case identifier to find the correct benchmark code.

5 Changes

5.1 Version 2.0.3

This being the first public release it is not to useful to list all the changes. BlueGnu has been modeled after DejaGnu and is a complete new implementation which has been tested thoroughly. When documentation is being written more changes will be made. An example of some of these changes as a result of this documentation effort can be found in the test-suites ts_001, ts_002, and ts_003. The first two are not as easy to write as the third. The test suite ts_001 is a script implementation of the command line:

```
% cd bluegnu2.0.3/testsets
% bluegnu examples/tc002[English=B] \
> "examples/tc002[Dutch=B]={MSG=Hallo Wereld}"
```

This may be useful for simple tests but when you want to write more complex test-suite scripts you would like some more flexibility, so two procedures were introduced, which are shown in test-suite ts_002. This makes writing rather complex so the procedures have become part of the procedures 'appendQueue', 'prependQueue', and 'runtest'. The resulting script is shown in test-suit ts_003.

Changes like this will be made in the future!

6 Future Enhancements

The following enhancements are being planned:

- Target code will be made into a class with methods 'start', 'load', 'exit', and 'version'. Instead of the current '<target>_start', '<target>_load', '<target>_exit', and '<target>_version', which have been taken from DejaGnu.
- Procedures will be created that make it easy to test WEB application from the framework.
- Other enhancements will be made depending on the use of the framework in testing different applications.