Password Gorilla

A password manager coded in Tcl/Tk

Part 1: General aspects

- Features
- Screenshots & Demonstration
- Taking over the project „Password Gorilla“
- Maintaining the sources
- Development tools
- Getting Feedback

Part 2: Tcl/Tk related aspects

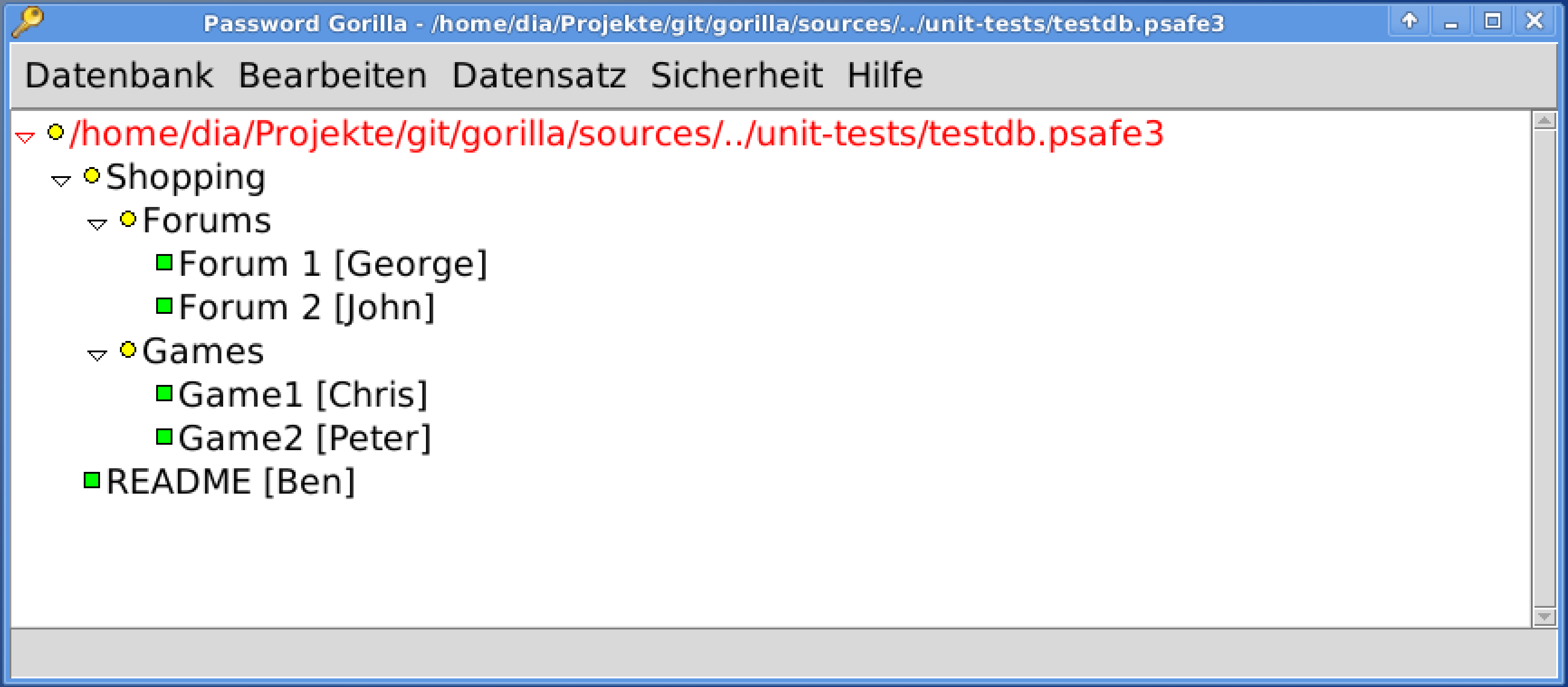
- Creating executables
- Managing the OSX port
- Speeding up the encryption algorithms
- The Hypertext Help System
- Localisation with GNU's *gettext* utility
- Testing with *tcltest*
- Documentation with *Ruff*
- Preview: The Android port

Features

- Cross-platform password manager
- Copy-paste service
- Integrated random password generator
- Cross-platform: Linux, FreeBSD, Windows and MacOS X
- Compatibility to actual Password Safe 3.2 databases
- SHA256 protection for the master password
- Content encryption with Bruce Schneier’s Twofish algorithm
- Prevention of brute force attacks by key stretching.
- Integration of a hypertext help system
- Localization support
- Starpack versions for Linux, Windows, OSX
- Dependency for use with sources: Tk 8.5

Password Gorilla: Linux Version

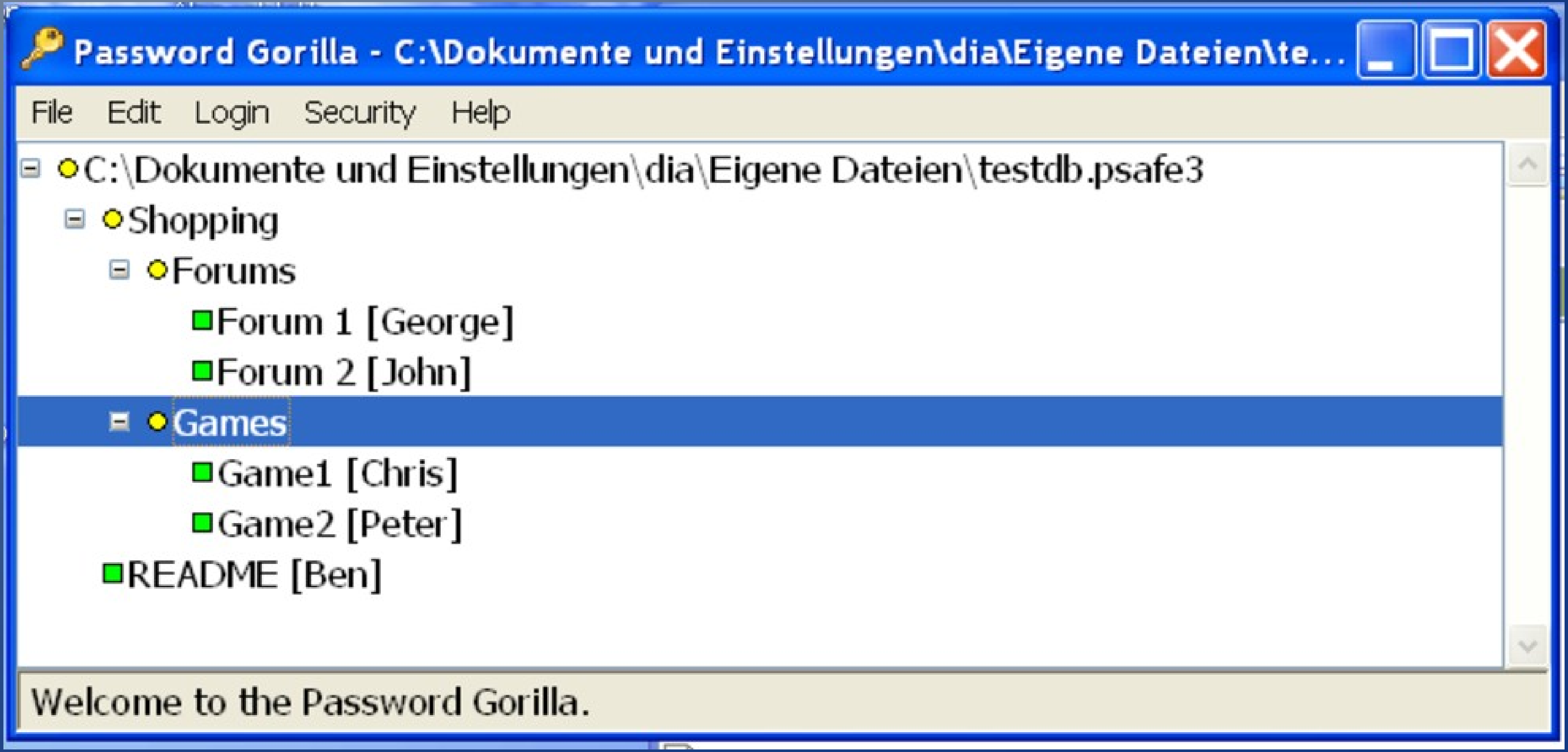
Main screen


Password Gorilla - /home/dia/Projekte/git/gorilla/sources/../unit-tests/testdb.psafe3
Datenbank Bearbeiten Datensatz Sicherheit Hilfe

/home/dia/Projekte/git/gorilla/sources/../unit-tests/testdb.psafe3
- Shopping
  - Forums
    - Forum 1 [George]
    - Forum 2 [John]
  - Games
    - Game1 [Chris]
    - Game2 [Peter]
  - README [Ben]

Password Gorilla: Windows Version

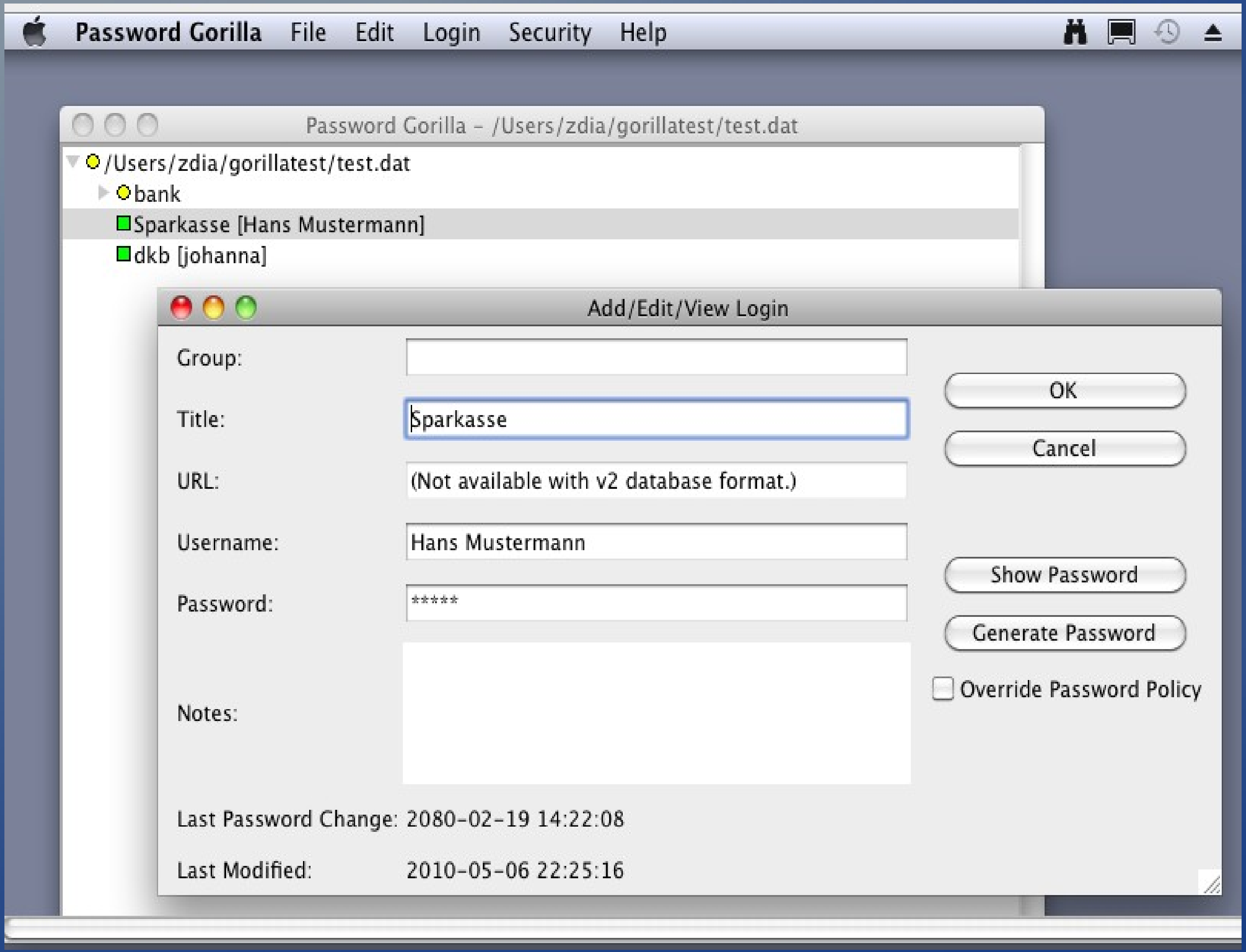
Main screen


Password Gorilla - C:\Dokumente und Einstellungen\dia\Eigene Dateien\te...
File Edit Login Security Help

C:\Dokumente und Einstellungen\dia\Eigene Dateien\testdb.psafe3
Shopping
Forums
Forum 1 [George]
Forum 2 [John]
Games
Game1 [Chris]
Game2 [Peter]
README [Ben]

Welcome to the Password Gorilla.

Password Gorilla: MacOSX Version



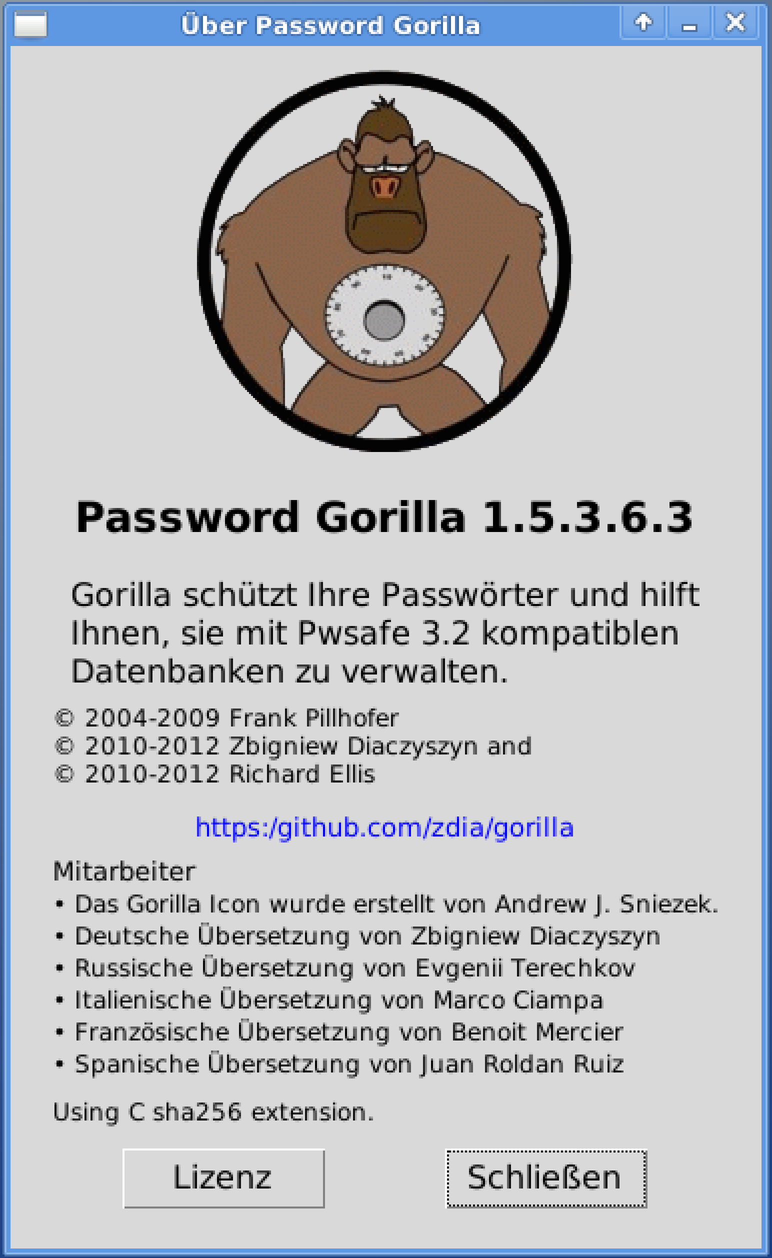
/Users/zdia/gorillatest/test.dat
bank
Sparkasse [Hans Mustermann]
dkb [johanna]

Group:	
Title:	Sparkasse
URL:	(Not available with v2 database format.)
Username:	Hans Mustermann
Password:	*****
Notes:	
Last Password Change:	2080-02-19 14:22:08
Last Modified:	2010-05-06 22:25:16

☐ Override Password Policy

Main screen

Password Gorilla: Linux Version



Password Gorilla 1.5.3.6.3

Gorilla schützt Ihre Passwörter und hilft Ihnen, sie mit Pwsafe 3.2 kompatiblen Datenbanken zu verwalten.

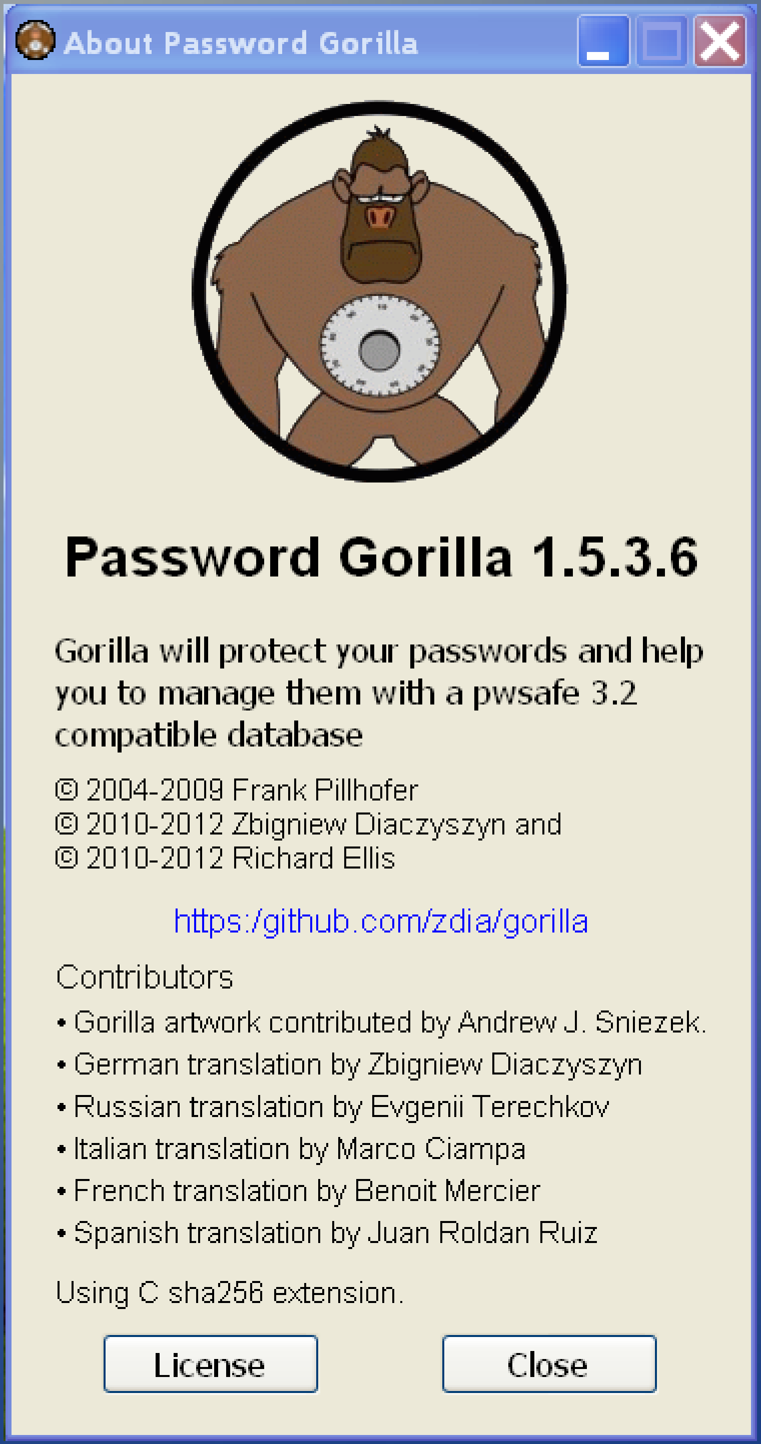
© 2004-2009 Frank Pillhofer
© 2010-2012 Zbigniew Diaczyszyn and
© 2010-2012 Richard Ellis

https:/github.com/zdia/gorilla

Mitarbeiter
- Das Gorilla Icon wurde erstellt von Andrew J. Sniezek.
- Deutsche Übersetzung von Zbigniew Diaczyszyn
- Russische Übersetzung von Evgenii Terechkov
- Italienische Übersetzung von Marco Ciampa
- Französische Übersetzung von Benoit Mercier
- Spanische Übersetzung von Juan Roldan Ruiz

Using C sha256 extension.

About Dialog Window



Password Gorilla 1.5.3.6

Gorilla will protect your passwords and help you to manage them with a pwsafe 3.2 compatible database

© 2004-2009 Frank Pillhofer
© 2010-2012 Zbigniew Diaczyszyn and
© 2010-2012 Richard Ellis

https:/github.com/zdia/gorilla

Contributors

- Gorilla artwork contributed by Andrew J. Sniezek.
- German translation by Zbigniew Diaczyszyn
- Russian translation by Evgenii Terechkov
- Italian translation by Marco Ciampa
- French translation by Benoit Mercier
- Spanish translation by Juan Roldan Ruiz

Using C sha256 extension.

Password Gorilla: Windows Version

About Dialog Window

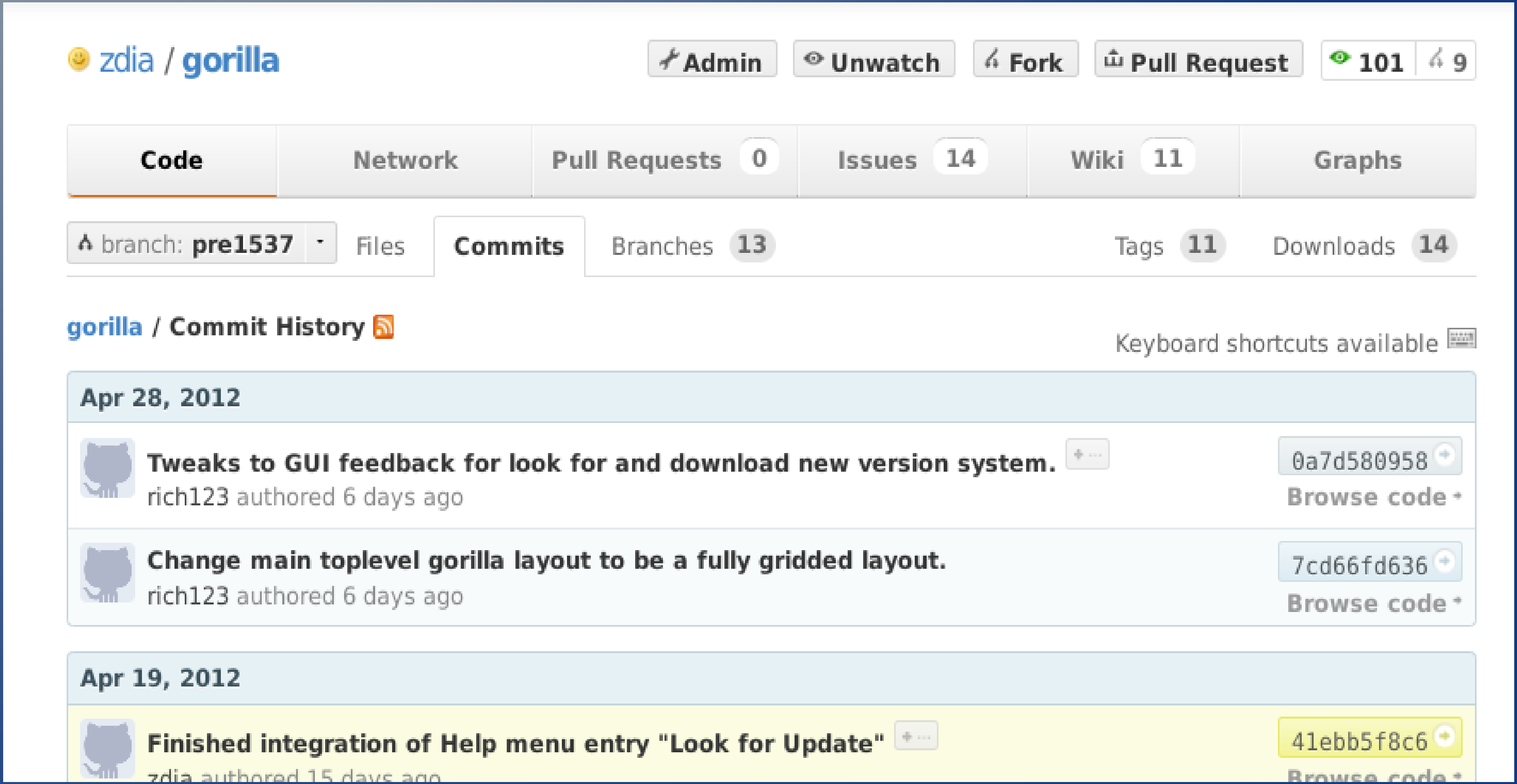
Taking over the project

- Frank Pillhofer (2004-2009)
  - Version 1.4.7
  - Itcl, Bwidget
- Zbigniew Diaczyszyn (2010)
  - Version 1.5.x
  - Bwidget replaced by Ttk widgets
  - msgcat mechanism added
  - Translation into German language
- Richard Ellis (2010)
  - Improving constantly code, adding various modules

Maintaining the sources

- Git
  - Encourages cloning
  - Fully fledged local repository
  - Simple and effective local branches
  - GUI programs: gitk, „git gui“
- Github Server
  - Download area
  - Wiki (Textile)
  - Automatic code tarballs
  - Social coding
  - Issue management

Github: Commits



zdia / gorilla

gorilla / Commit History

Keyboard shortcuts available

Apr 28, 2012

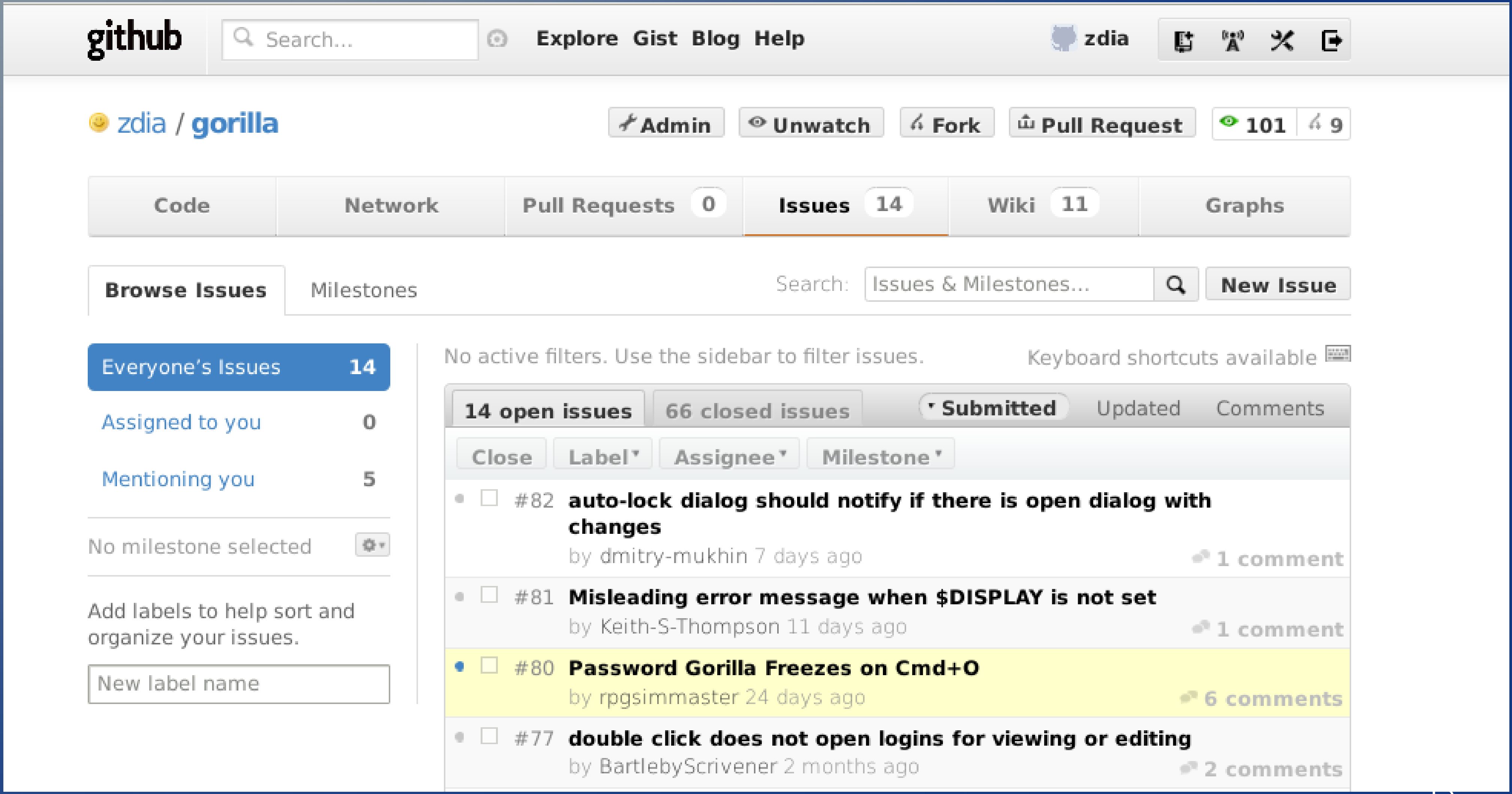
Tweaks to GUI feedback for look for and download new version system.
rich123 authored 6 days ago — 0a7d580958 — Browse code

Change main toplevel gorilla layout to be a fully gridded layout.
rich123 authored 6 days ago — 7cd66fd636 — Browse code

Apr 19, 2012

Finished integration of Help menu entry "Look for Update"
zdia authored 15 days ago — 41ebb5f8c6 — Browse code

Github: Issues



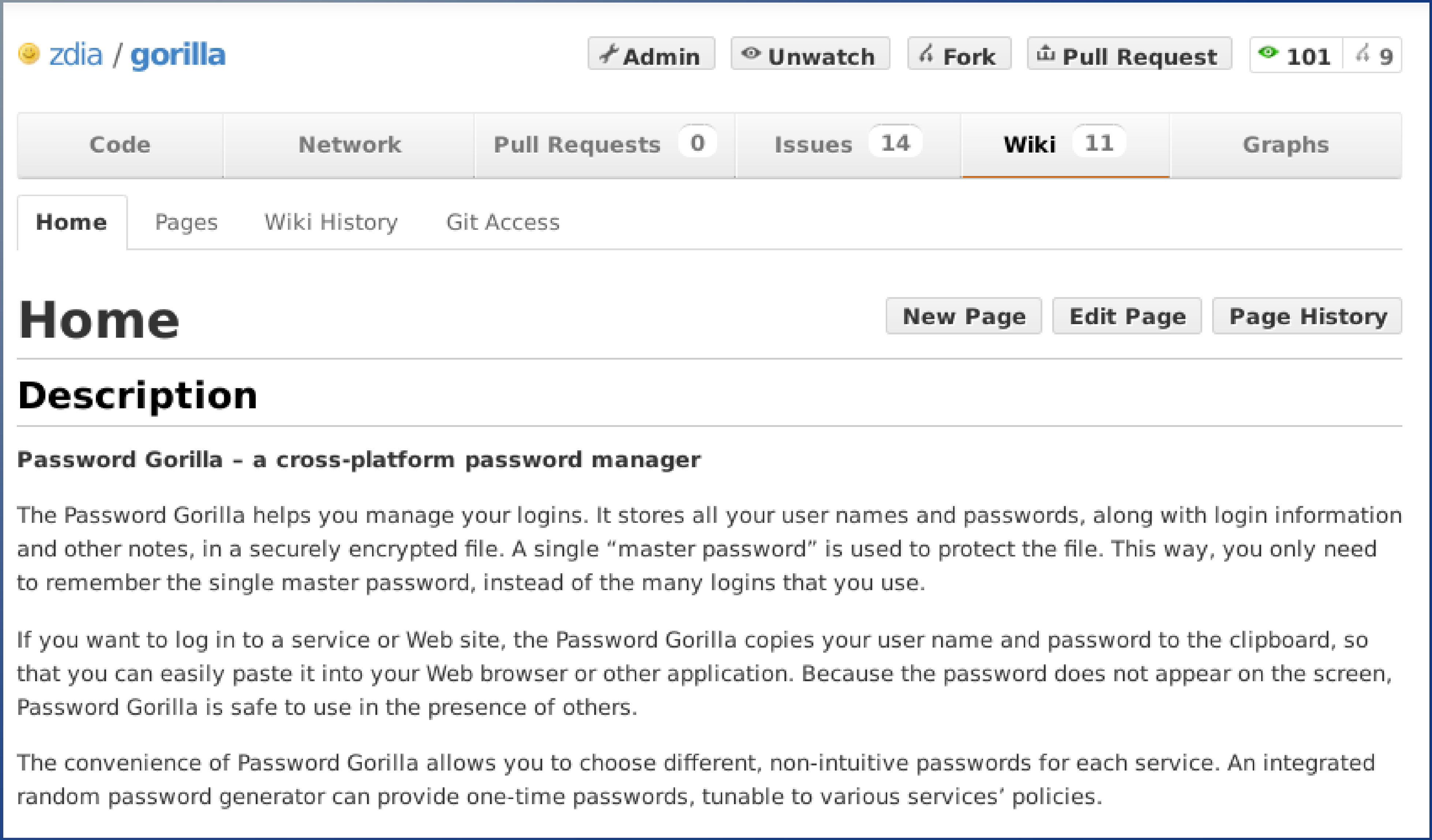
#82 auto-lock dialog should notify if there is open dialog with changes
by dmitry-mukhin 7 days ago — 1 comment

#81 Misleading error message when $DISPLAY is not set
by Keith-S-Thompson 11 days ago — 1 comment

#80 Password Gorilla Freezes on Cmd+O
by rpgsimmaster 24 days ago — 6 comments

#77 double click does not open logins for viewing or editing
by BartlebyScrivener 2 months ago — 2 comments

Github: Wiki



Home

Description

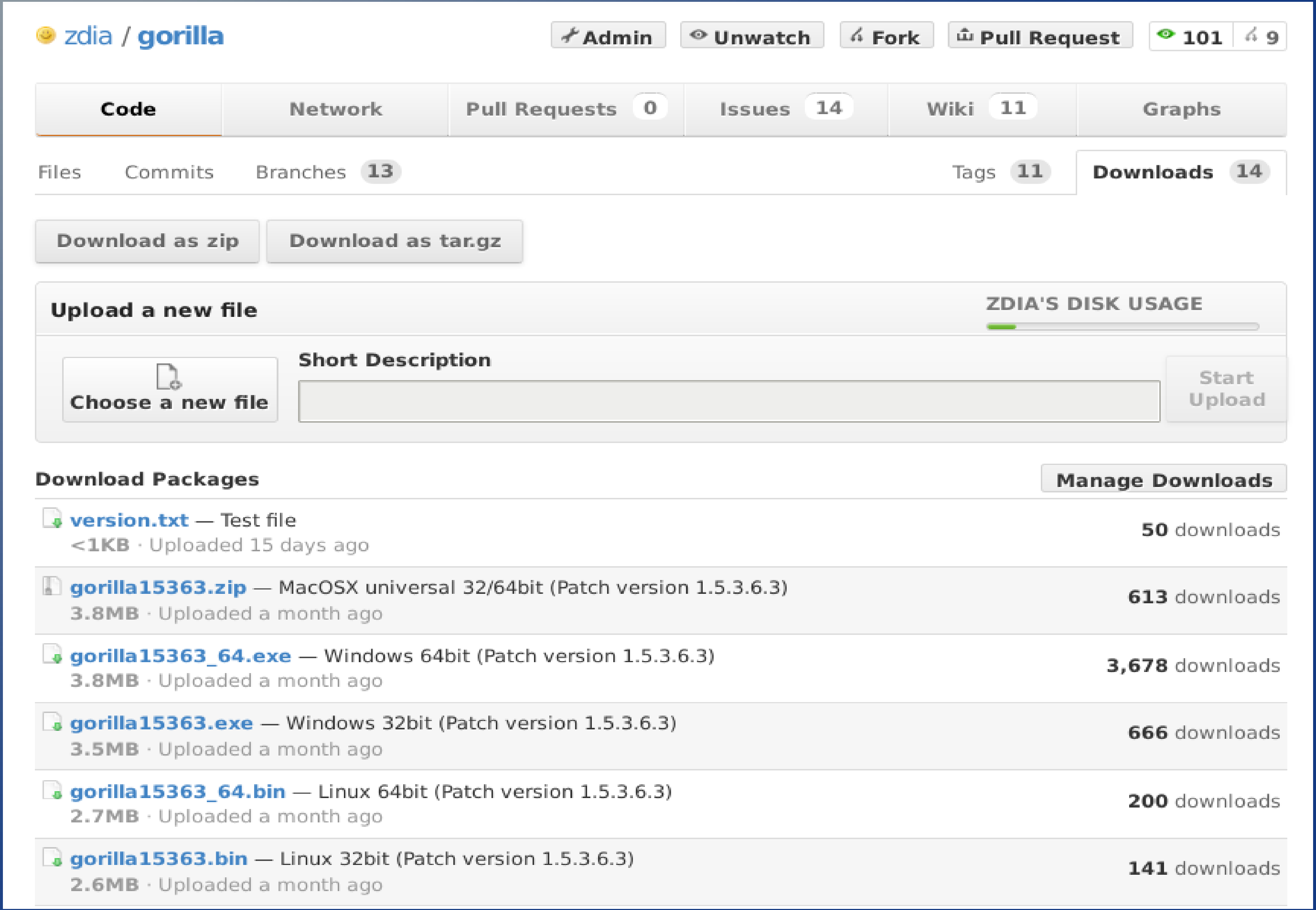
Password Gorilla – a cross-platform password manager

The Password Gorilla helps you manage your logins. It stores all your user names and passwords, along with login information and other notes, in a securely encrypted file. A single “master password” is used to protect the file. This way, you only need to remember the single master password, instead of the many logins that you use.

If you want to log in to a service or Web site, the Password Gorilla copies your user name and password to the clipboard, so that you can easily paste it into your Web browser or other application. Because the password does not appear on the screen, Password Gorilla is safe to use in the presence of others.

The convenience of Password Gorilla allows you to choose different, non-intuitive passwords for each service. An integrated random password generator can provide one-time passwords, tunable to various services’ policies.

Github: Downloads



zdia / gorilla

Admin | Unwatch | Fork | Pull Request | 101 | 9

Code | Network | Pull Requests 0 | Issues 14 | Wiki 11 | Graphs

Files | Commits | Branches 13 | Tags 11 | Downloads 14

Download as zip | Download as tar.gz

Upload a new file | ZDIA'S DISK USAGE

Choose a new file | **Short Description** | Start Upload

Download Packages | Manage Downloads

version.txt — Test file **<1KB** · Uploaded 15 days ago	**50** downloads
gorilla15363.zip — MacOSX universal 32/64bit (Patch version 1.5.3.6.3) **3.8MB** · Uploaded a month ago	**613** downloads
gorilla15363_64.exe — Windows 64bit (Patch version 1.5.3.6.3) **3.8MB** · Uploaded a month ago	**3,678** downloads
gorilla15363.exe — Windows 32bit (Patch version 1.5.3.6.3) **3.5MB** · Uploaded a month ago	**666** downloads
gorilla15363_64.bin — Linux 64bit (Patch version 1.5.3.6.3) **2.7MB** · Uploaded a month ago	**200** downloads
gorilla15363.bin — Linux 32bit (Patch version 1.5.3.6.3) **2.6MB** · Uploaded a month ago	**141** downloads

Development Tools

- Geany
- Tkcon
-

Getting feedback

Editor's Opinion

Password Gorilla - a powerful and highly efficient way in which you can securely store all of your login credentials. This versatile and easy to use piece of software is an excellent tool for any computer user that wants to secure all of their accounts.

Creating executables (1)

- Virtual Filesystem gorilla.vfs

```
gorilla.vfs/
|-- lib/
|   `-- app-gorilla/
|-- main.tcl
`-- tclkit.ico*
```

app-gorilla/ is linked to:

```
/home/dia/Projekte/git/gorilla/sources/
```

Creating executables (2)

- Content of main.tcl

```
package require starkit
starkit::startup
package require app-gorilla 1.0
```

- First line in application:

```
package provide app-gorilla 1.0
```

Creating executables (3)

- Final creation command:

```
./tclkit sdx.kit wrap gorilla.exe -runtime <path/tclkit-version>
```

- Actual runtime versions:

```
tclkit-8.5.11-tk-freebsd-ix86
tclkit-8.5.11-linux
tclkit-tk-8.5.11-linux-x86_64
tclkit-8.5.11-win32.exe
tclkit-tk-8.5.11-win32-x86_64
tclkit-8.5.11-macosx-universal
```

The OSX port : Bundle Structure

The application bundle *gorilla.zip*

```
|-- Password Gorilla.app
|   `-- Contents
|       |-- Info.plist
|       |-- MacOS
|       |   `-- gorilla.aqua
|       `-- Resources
|           `-- Gorilla.icns
`-- gorilla.zip
```

The OSX port: The *Info.plist* file

The Information Property List File *Info.plist*

```
<key>CFBundleExecutable</key>
<string>gorilla.aqua</string>
<key>CFBundleGetInfoString</key>
<string>Password Gorilla</string>
<key>CFBundleIconFile</key>
<string>Gorilla.icns</string>
<key>CFBundleIdentifier</key>
<string>password.gorilla</string>
...
<key>CFBundleVersion</key>
<string>1.5.3.6.1</string>
```

The OSX port: The OSX Menus

```
if {[tk windowingsystem] == "aqua"} {
    # we have to delete the psn_nr in argv
    if {[string first "-psn" [lindex $argv 0]] == 0} {
        set argv [lrange $argv 1 end]}
    }
    proc ::tk::mac::ShowPreferences {} {
            gorilla::PreferencesDialog
    }
    proc ::tk::mac::Quit {} {
        gorilla::Exit
    }
    proc tk::mac::ShowHelp {} {
        gorilla::Help
    }
}
```

The OSX port: The About Dialog

Enable the event:

```
proc tkAboutDialog {} {
     gorilla::About
}
```

Manage the menu entry:

```
menu .mbar.apple
.mbar add cascade -menu .mbar.apple
.mbar.apple add command -label "[mc "About"] Password Gorilla" \
     -command gorilla::About
```

The OSX port: The File Dialog

Native fileselect dialog does not allow filtering of files:

```
tk_getOpenFile -filetypes [list ] -initialdir $::gorilla::dirName
```

C Extensions with CriTcl

Get CriTcl version 3 (Compiled Runtime In Tcl):

```
$ git clone https://github.com/jcw/critcl.git
```

Build critcl starpack:

```
$ ./build.tcl starpack prefix ?destination?
```

Use it to build extension package *sha256c*:

```
$ critcl -pkg sha256c.tcl
```

Get help:

https://github.com/jcw/critcl/blob/master/embedded/www/toc.html

Directory structure for CriTcl created libraries:

```
sha256c/
|-- freebsd-ix86/
|   `-- sha256c.so*
|-- linux-ix86/
|   `-- sha256c.so*
|-- linux-x86_64/
|   `-- sha256c.so*
|-- macosx-ix86/
|   `-- sha256c.dylib*
|-- macosx-x86_64/
|   `-- sha256c.dylib*
|-- win32-ix86/
|   `-- sha256c.dll*
|-- win32-x86_64/
|   `-- sha256c.dll*
|-- critcl.tcl
`-- pkgIndex.tcl
```

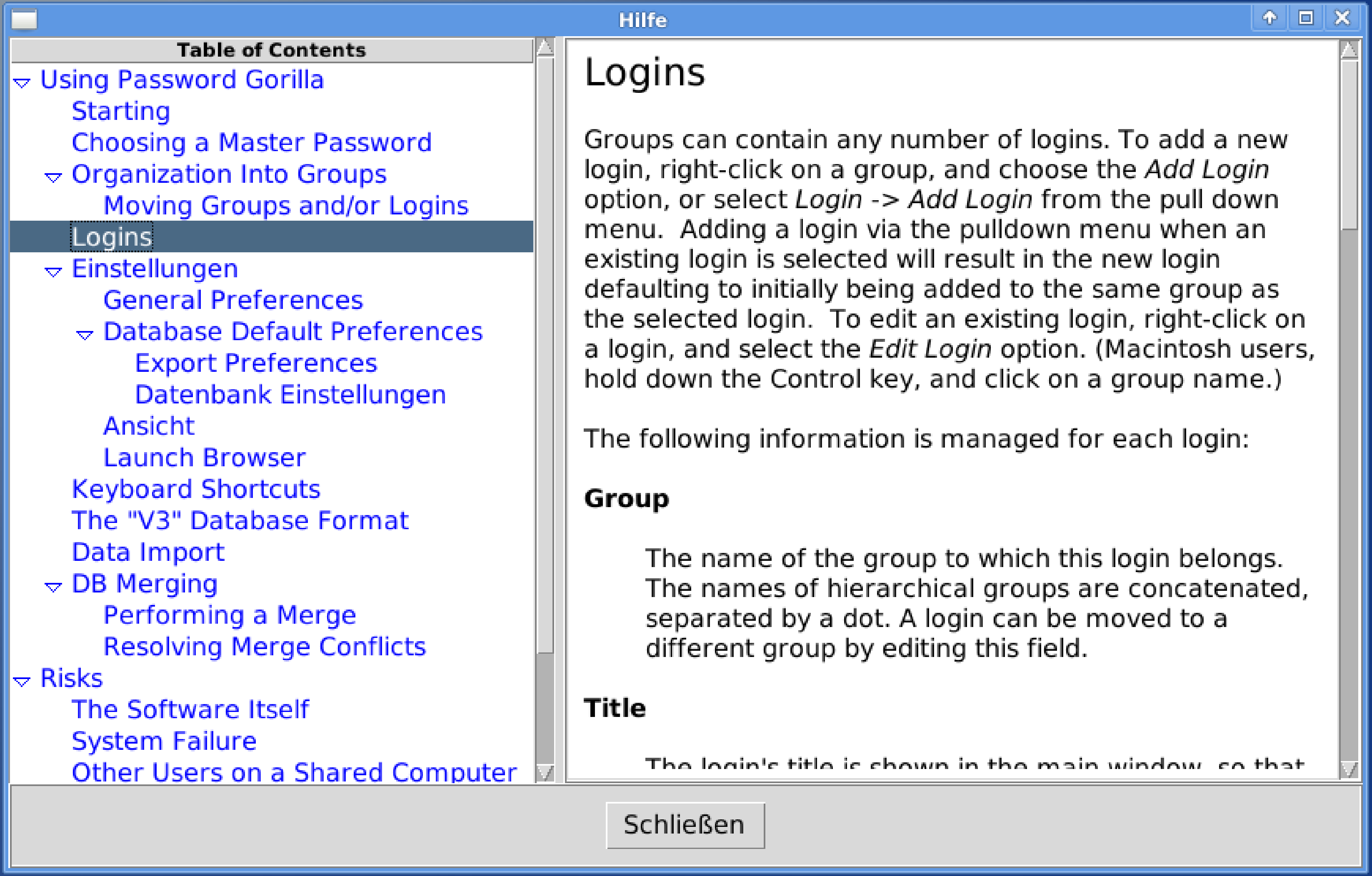
C Extensions with CriTcl

Build and use on native system

Intensiv testen cross-compile → Windows

```
$ critcl -pkg -target mingw32 foo
linux-32-*
linux-64-*
macosx-universal
mingw32
win32-ix86-cl
win32-x86_64-cl-buf
win32-x86_64-cl-nobuf
```

The Hypertext Help System



Logins

Groups can contain any number of logins. To add a new login, right-click on a group, and choose the *Add Login* option, or select *Login -> Add Login* from the pull down menu. Adding a login via the pulldown menu when an existing login is selected will result in the new login defaulting to initially being added to the same group as the selected login. To edit an existing login, right-click on a login, and select the *Edit Login* option. (Macintosh users, hold down the Control key, and click on a group name.)

The following information is managed for each login:

Group

The name of the group to which this login belongs. The names of hierarchical groups are concatenated, separated by a dot. A login can be moved to a different group by editing this field.

Title

The login's title is shown in the main window, so that

Help system features

- A help system originally based on Tcl Wiki 1194 and Tile
- Author: Keith Vetter (May 2007)
- Hyperlinks to other help pages
- Simple searching ability
- History
- Simple wiki formatting:
  - Nested numeric, bullet and dash lists
  - Bold, italic and unformatted text
- Table of Contents
- Adapted for PWGorilla with msgcat support and Ttk widgets

Help system text example

----

title: Logins

Groups can contain any number of logins. To add a new login, right-click on a group, and choose the "Add Login" option, or select "Login" -> "Add Login" from the pull down menu. [...]

The following information is managed for each login:

'''Group'''

| The name of the group to which this login belongs. The names of hierarchical groups are concatenated, separated by a dot. A login can be moved to a different group by editing this field.

Modifying The Help System

It is based on a treeview and a text widget

Example for adding an indented paragraph with sign „|“:

```
$w.t tag config bar -lmargin1 $l2 -lmargin2 $l2
...
if { $op1 eq "|" } {          ; # Bar
        set tag bar
}
```

Using The Help System

```
source viewhelp.tcl

proc gorilla::Help {} {
    # ReadHelpFiles is looking in the given directory
    # for a file named help.txt
    ::Help::ReadHelpFiles $::gorillaDir $::gorilla::preference(lang)
    ::Help::Help Overview
}
```

Localization with *gettext*

Source code format:

```
puts [mc "Hello world"]
```

Create a Portable Object Template:

```
xgettext -kmc -o gorilla.pot -L Tcl gorilla.tcl ?...?
```

Result in file *gorilla.pot*:

```
#: ../../sources/gorilla.tcl:193
#, tcl-format
msgid "Need %s"
Msgstr ""
```

Gettext: Creating locale .po file

Create English version *en.po:*

```
Msgen -o en.po gorilla.pot
```

Edit *gorilla.pot*:

```
#: ../../sources/gorilla.tcl:193
#, tcl-format
msgid "Need %s"
Msgstr ""
```

Save result in file *de.po*:

```
#: ../../sources/gorilla.tcl:193
#, tcl-format
msgid "Need %s"
msgstr "Benötige %s"
```

Gettext: Updating .po files

Merge existing <lang>.po with new *gorilla.pot*:

```
msgmerge --update <lang>.po --backup=simple gorilla.pot"
```

Create a Tcl *.msg language file:

```
msgfmt --tcl -l<lang> -d <path-to-msgs-dir> <lang>.po
```

(lang = en, de, fr, ru ...)

Result in file *de.msg*:

```
::msgcat::mcset de "Need %s" "Ben\u00f6tige %s"
::msgcat::mcset de "File" "Datenbank"
...
```

Gettext: Tweaking .msg files

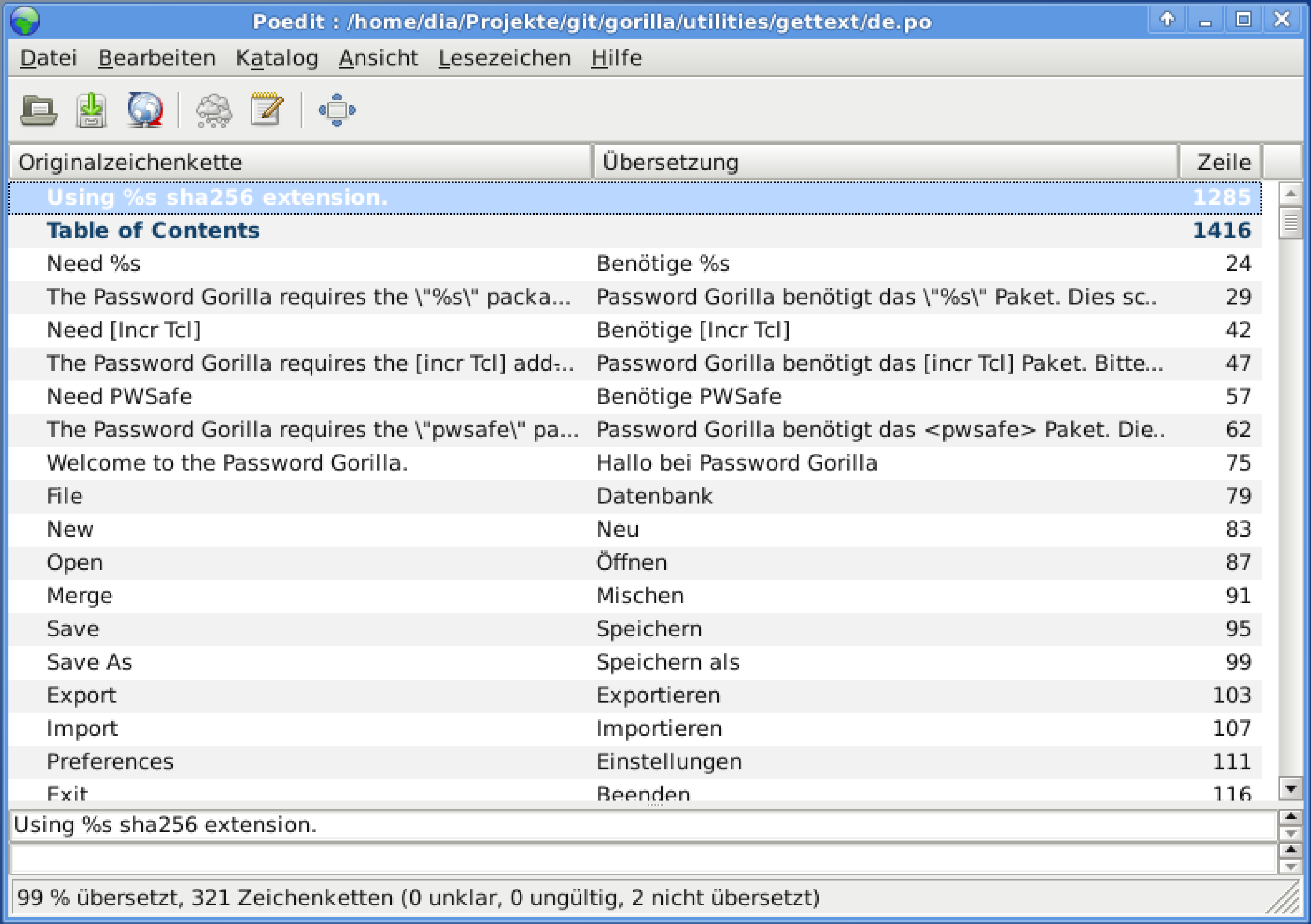
Redefine *mcset*:

```
proc mcset { lang fromstr tostr } {
    variable msgdata
    dict lappend msgdata $lang $fromstr $tostr
  }
```

Use the dictionary to build the final *de.msg:*

```
mcmset de {
{Need %s} {Benötige %s}
File Datenbank
...
}
```

Gettext: Editor *poedit*



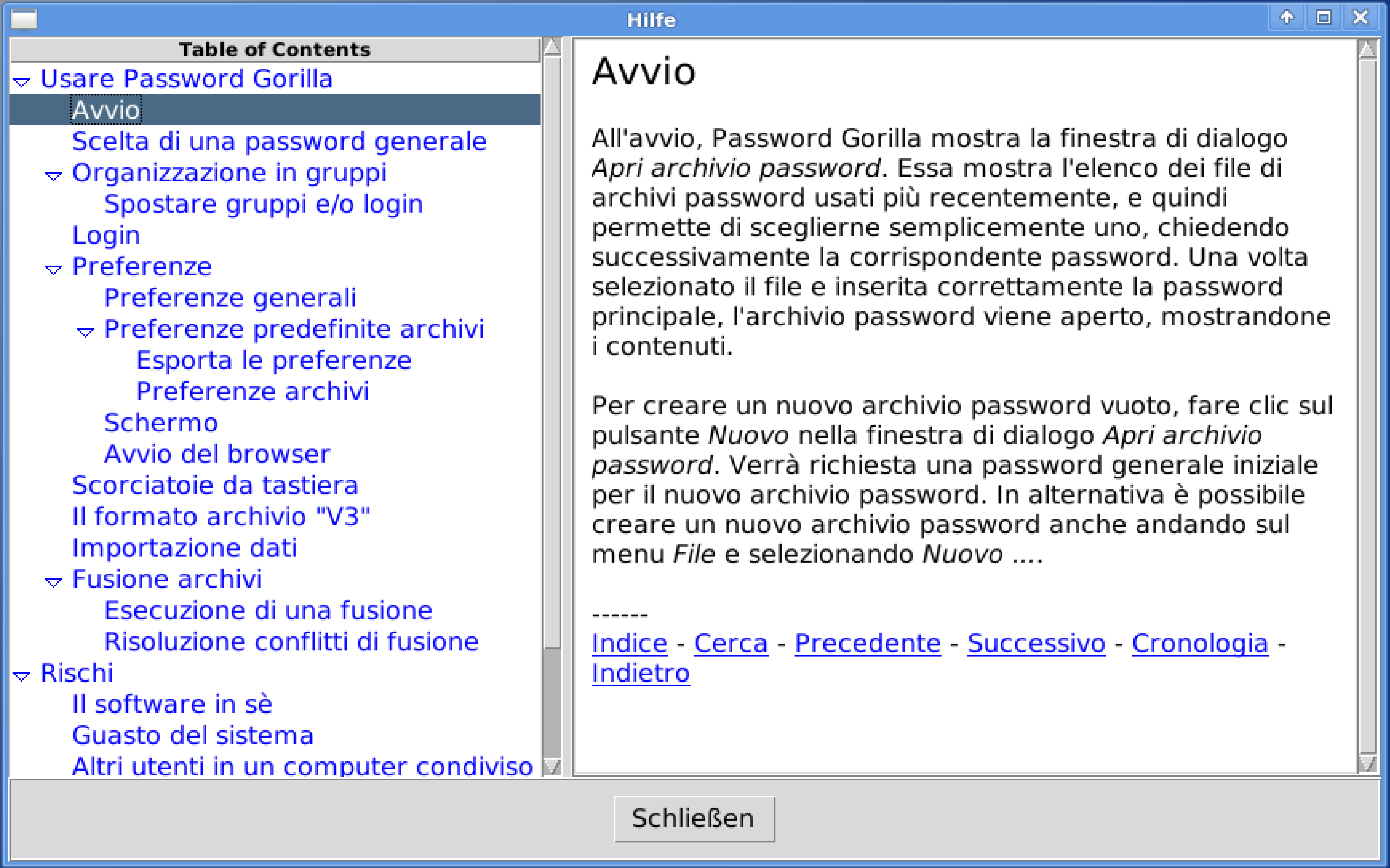
Poedit : /home/dia/Projekte/git/gorilla/utilities/gettext/de.po

Originalzeichenkette	Übersetzung	Zeile
Using %s sha256 extension.		1285
Table of Contents		1416
Need %s	Benötige %s	24
The Password Gorilla requires the \"%s\" packa...	Password Gorilla benötigt das \"%s\" Paket. Dies sc..	29
Need [Incr Tcl]	Benötige [Incr Tcl]	42
The Password Gorilla requires the [incr Tcl] add-...	Password Gorilla benötigt das [incr Tcl] Paket. Bitte...	47
Need PWSafe	Benötige PWSafe	57
The Password Gorilla requires the \"pwsafe\" pa...	Password Gorilla benötigt das <pwsafe> Paket. Die..	62
Welcome to the Password Gorilla.	Hallo bei Password Gorilla	75
File	Datenbank	79
New	Neu	83
Open	Öffnen	87
Merge	Mischen	91
Save	Speichern	95
Save As	Speichern als	99
Export	Exportieren	103
Import	Importieren	107
Preferences	Einstellungen	111
Exit	Beenden	116

Using %s sha256 extension.

99 % übersetzt, 321 Zeichenketten (0 unklar, 0 ungültig, 2 nicht übersetzt)

Gettext: Italian Help Text



Usare Password Gorilla
- Avvio
- Scelta di una password generale
- Organizzazione in gruppi
  - Spostare gruppi e/o login
- Login
- Preferenze
  - Preferenze generali
  - Preferenze predefinite archivi
    - Esporta le preferenze
    - Preferenze archivi
  - Schermo
  - Avvio del browser
- Scorciatoie da tastiera
- Il formato archivio "V3"
- Importazione dati
- Fusione archivi
  - Esecuzione di una fusione
  - Risoluzione conflitti di fusione

Rischi
- Il software in sè
- Guasto del sistema
- Altri utenti in un computer condiviso

Avvio

All'avvio, Password Gorilla mostra la finestra di dialogo *Apri archivio password*. Essa mostra l'elenco dei file di archivi password usati più recentemente, e quindi permette di sceglierne semplicemente uno, chiedendo successivamente la corrispondente password. Una volta selezionato il file e inserita correttamente la password principale, l'archivio password viene aperto, mostrandone i contenuti.

Per creare un nuovo archivio password vuoto, fare clic sul pulsante *Nuovo* nella finestra di dialogo *Apri archivio password*. Verrà richiesta una password generale iniziale per il nuovo archivio password. In alternativa è possibile creare un nuovo archivio password anche andando sul menu *File* e selezionando *Nuovo* ....

Indice - Cerca - Precedente - Successivo - Cronologia - Indietro

Ruff!: Features

Ruff! (Runtime function formatter) v0.4

Author: (c) Ashok P. Nadkarni
http://woof.magicsplat.com/ruff_home

Written in Tcl

Runtime introspection

comment analysis

```
package require ruff
::ruff::document_namespaces html [list ::NS] -output NS.html
-recurse true

File Datenbank
...
}
```

Ruff: Use

Initialization:

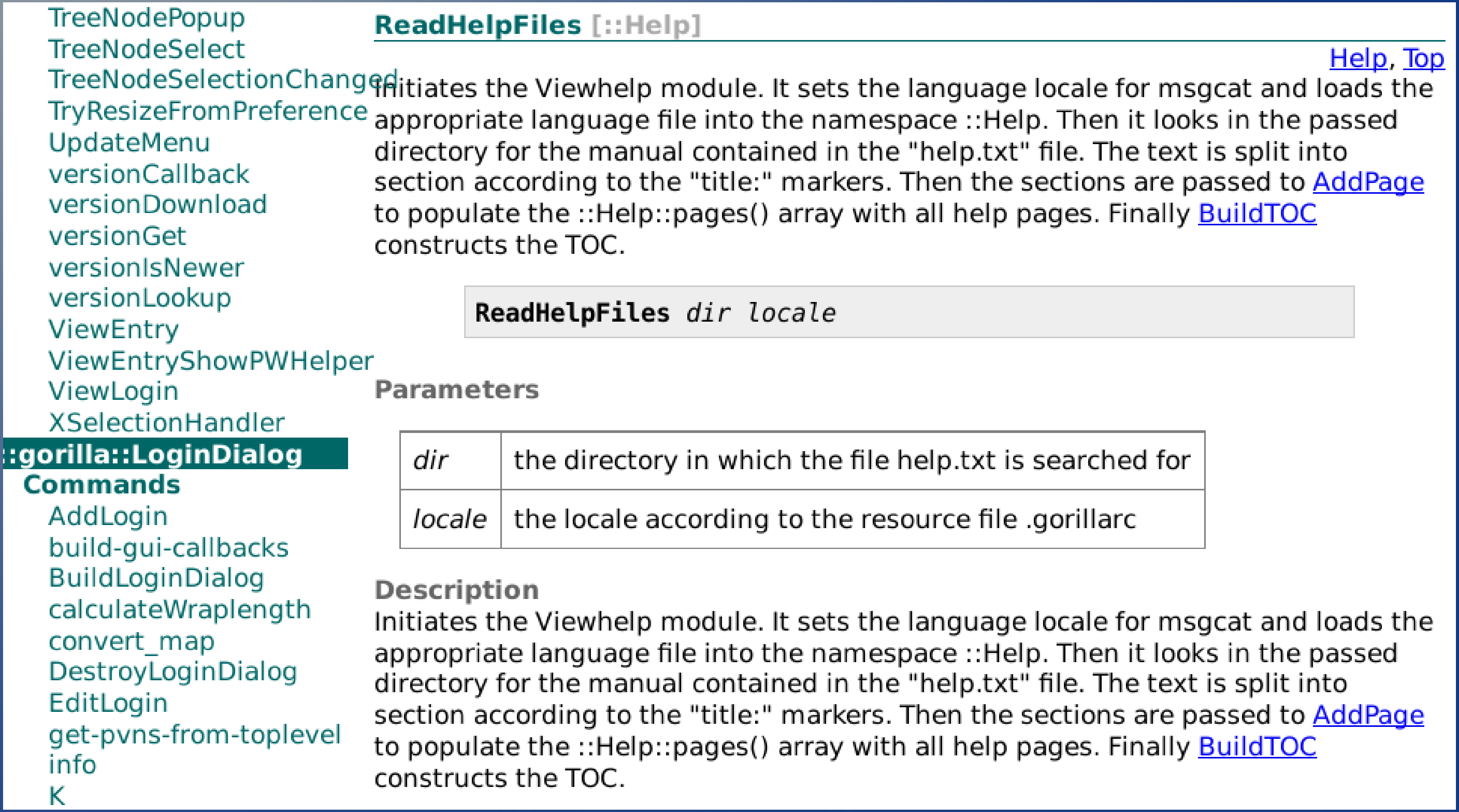
Package require ruff struct::list

```
set nslist [ ::struct::list filterfor z [ namespace children :: ] \
                              { ! [ regexp {^::(ttk|uuid|msgcat|pkg|tcl|
auto_mkindex_parser|itcl|sha2|tk|struct|ruff|textutil|cmdline|
critcl|activestate|platform)$} $z ] } ]

::ruff::document_namespaces html $nslist -output
gorilladoc.html -recurse true

Gorilla --sourcedoc
```

Ruff!: Html Display



TreeNodePopup
TreeNodeSelect
TreeNodeSelectionChanged
TryResizeFromPreference
UpdateMenu
versionCallback
versionDownload
versionGet
versionIsNewer
versionLookup
ViewEntry
ViewEntryShowPWHelper
ViewLogin
XSelectionHandler

::gorilla::LoginDialog

Commands

AddLogin
build-gui-callbacks
BuildLoginDialog
calculateWraplength
convert_map
DestroyLoginDialog
EditLogin
get-pvns-from-toplevel
info
K

ReadHelpFiles [::Help]

Help, Top

Initiates the Viewhelp module. It sets the language locale for msgcat and loads the appropriate language file into the namespace ::Help. Then it looks in the passed directory for the manual contained in the "help.txt" file. The text is split into section according to the "title:" markers. Then the sections are passed to AddPage to populate the ::Help::pages() array with all help pages. Finally BuildTOC constructs the TOC.

```
ReadHelpFiles dir locale
```

Parameters

dir	the directory in which the file help.txt is searched for
locale	the locale according to the resource file .gorillarc

Description

Initiates the Viewhelp module. It sets the language locale for msgcat and loads the appropriate language file into the namespace ::Help. Then it looks in the passed directory for the manual contained in the "help.txt" file. The text is split into section according to the "title:" markers. Then the sections are passed to AddPage to populate the ::Help::pages() array with all help pages. Finally BuildTOC constructs the TOC.

Tcltest: source code integration

Switch of commandline option:

```
--tcltest {
    # TCLTEST 1 and TEST 1 means:
    # skip the OpenDatabase dialog and load testdb.psafe3
    array set ::DEBUG { TCLTEST 1 TEST 1 }
    }
```

After general and GUI initialization:

```
if { $::DEBUG(TCLTEST) } {
    set argv ""
    source [file join $::gorillaDir .. unit-tests RunAllTests.tcl]
}
```

Tcltest: Directory structure

```
unit-tests
|-- RunAllTests.tcl
|-- backup
|   `-- backup.test
|-- csv-export
|   |-- csv-export.test
|   `-- normexport.csv
|-- csv-import
|   |-- csv-import.test
|   |-- import11.csv
|   |-- import110.csv
|   |-- import17.csv
|-- lock-database
|   `-- lock.test
`-- testdb.psafe3
```

Tcltest: RunAllTests.tcl

```
package require tcltest 2.2
...
# default search path is the actual working directory
tcltest::workingDirectory [file dirname [file normalize [info script]]]
tcltest::singleProcess 1      ;# caller environment will be used
tcltest::verbose { pass }
...
foreach testFile $testList {
    source $testFile
}
...
tcltest::cleanupTests    ;# will give the results and exit
```

Tcltest: A Single Test

```
test $testname-1.2 {File Open Error} \
    -setup { set ::DEBUG(CSVIMPORT) 1 } \
    -body { gorilla::Import unknown.csv } \
    -cleanup { set ::DEBUG(CSVIMPORT) 0 } \
    -result GORILLA_OPENERROR
```

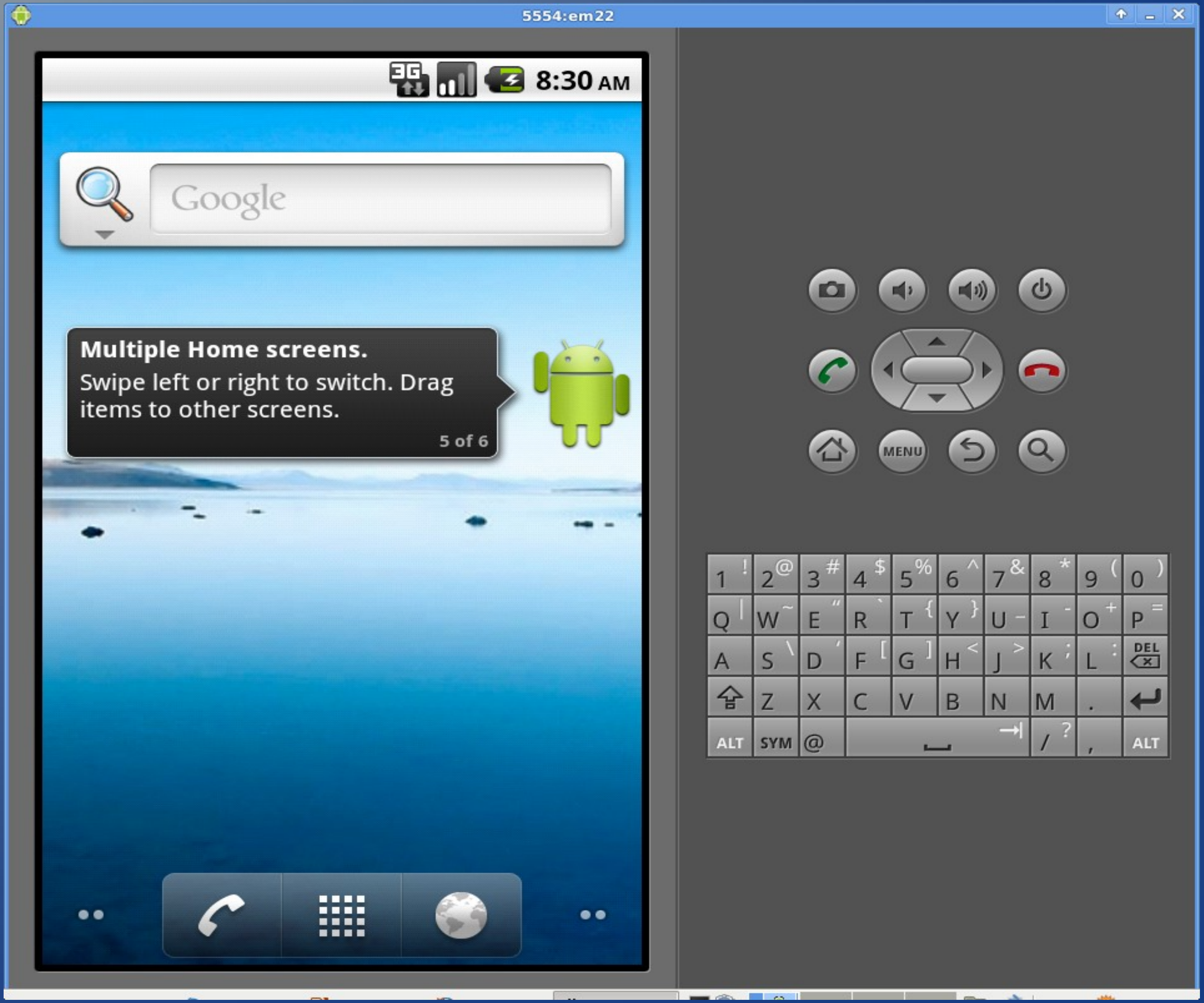
Tcltest: Final Test Run

```
$ gorilla --tcltest
++++ csv-import.test-1.2 PASSED
++++ csv-import.test-1.3 PASSED
++++ csv-import.test-1.4 PASSED
...
++++ csv-import.test-1.13 PASSED
++++ csv-import.test-1.1 PASSED
++++ csv-import.test-2.1 PASSED
++++ csv-import.test-2.2 PASSED
++++ csv-import.test-2.3 PASSED
++++ csv-export.test-1.0 PASSED
++++ lock-database-1.1 PASSED
++++ backup-1.1 PASSED
++++ backup-1.2 PASSED
RunAllTests.tcl:  Total   20  Passed    20  Skipped   0    Failed 0
```

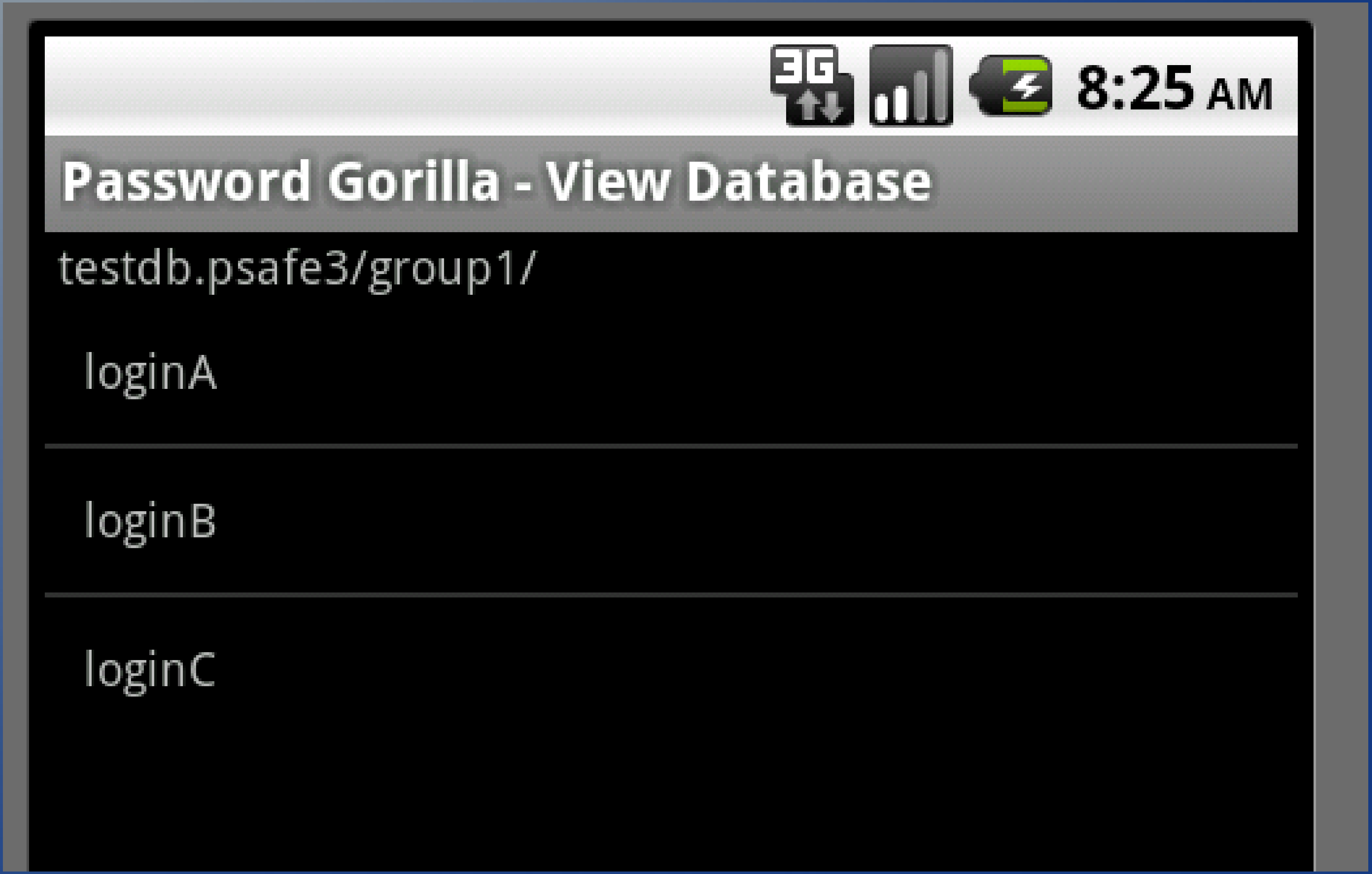
Tcltest: A Single Test

```
test $testname-1.2 {File Open Error} \
    -setup { set ::DEBUG(CSVIMPORT) 1 } \
    -body { gorilla::Import unknown.csv } \
    -cleanup { set ::DEBUG(CSVIMPORT) 0 } \
    -result GORILLA_OPENERROR
```

Android: Emulator



Android: Treeview (Logins)


8:25 AM

Password Gorilla - View Database

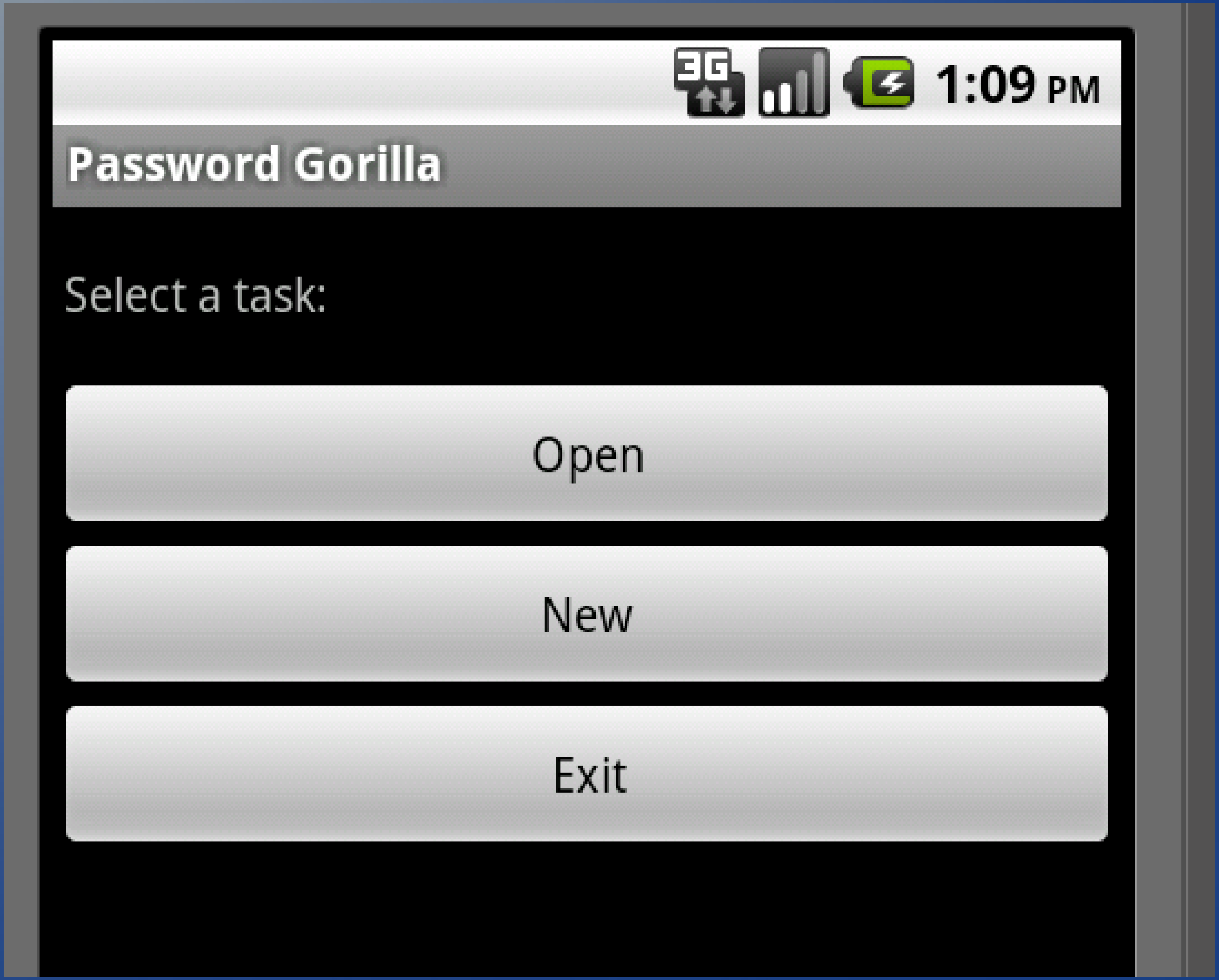
testdb.psafe3/group1/

loginA

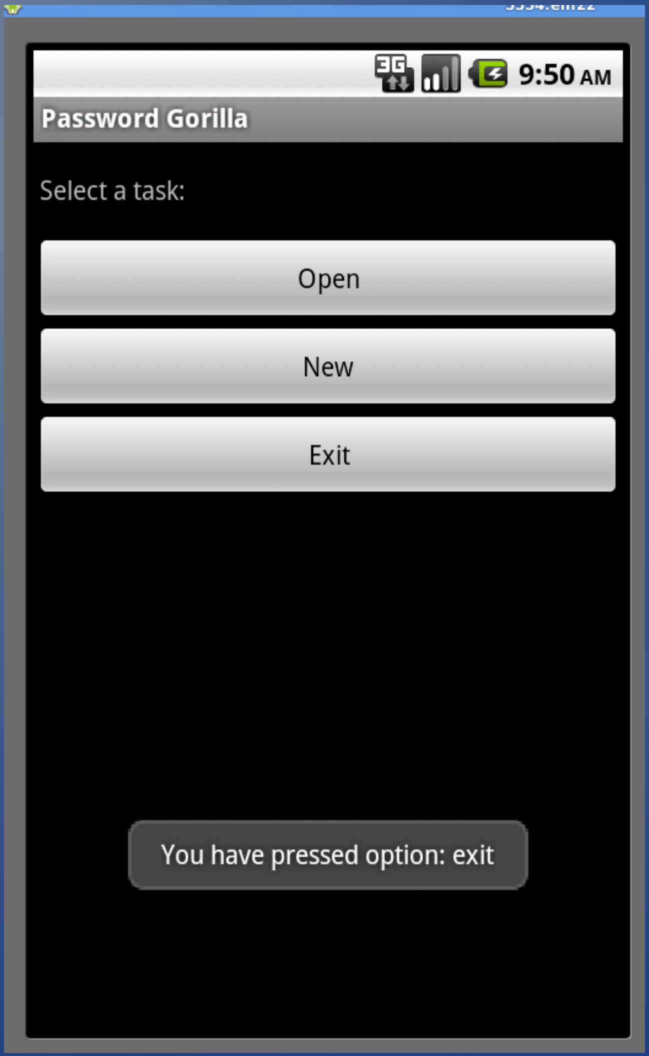
loginB

loginC

Android: Open Dialog



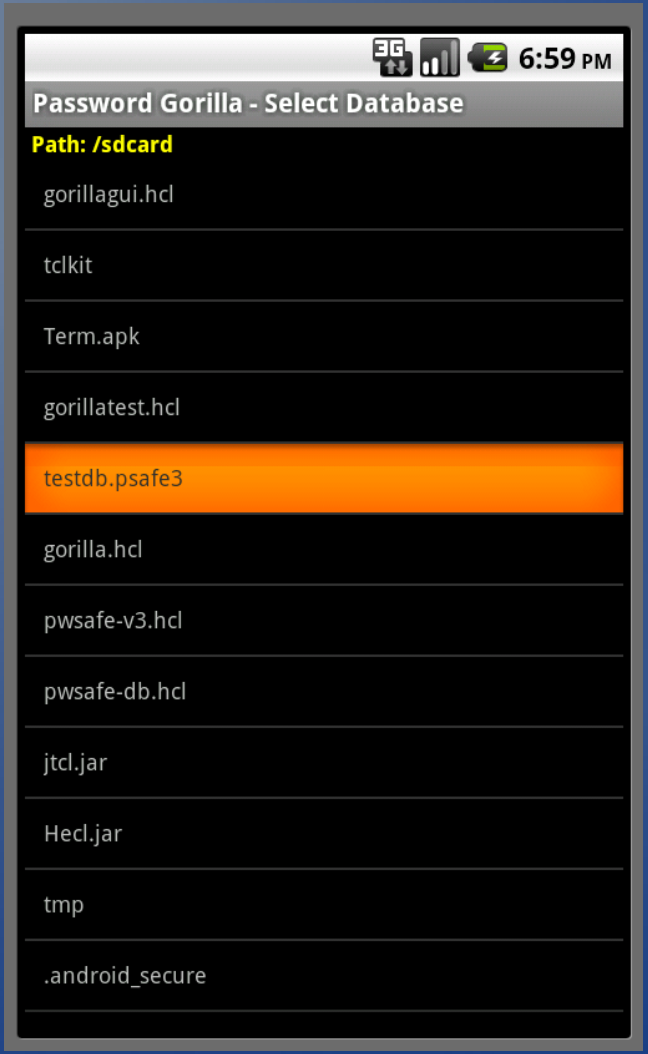
Android: Alert



Select a task:

You have pressed option: exit

Android: File Dialog



6:59 PM

Password Gorilla - Select Database

Path: /sdcard

gorillagui.hcl

tclkit

Term.apk

gorillatest.hcl

testdb.psafe3

gorilla.hcl

pwsafe-v3.hcl

pwsafe-db.hcl

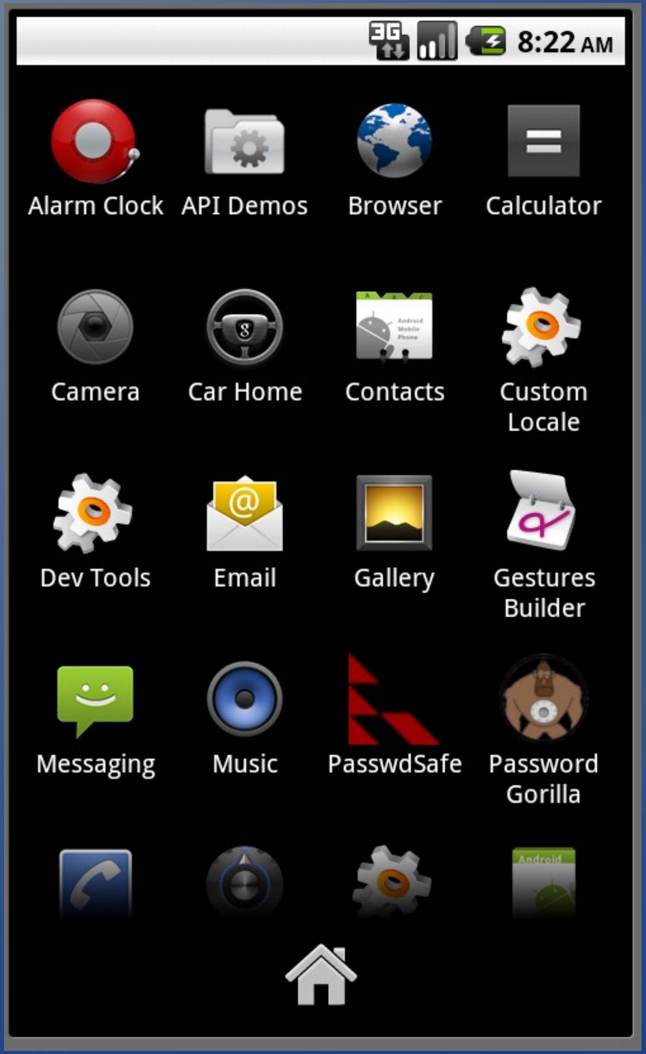
jtcl.jar

Hecl.jar

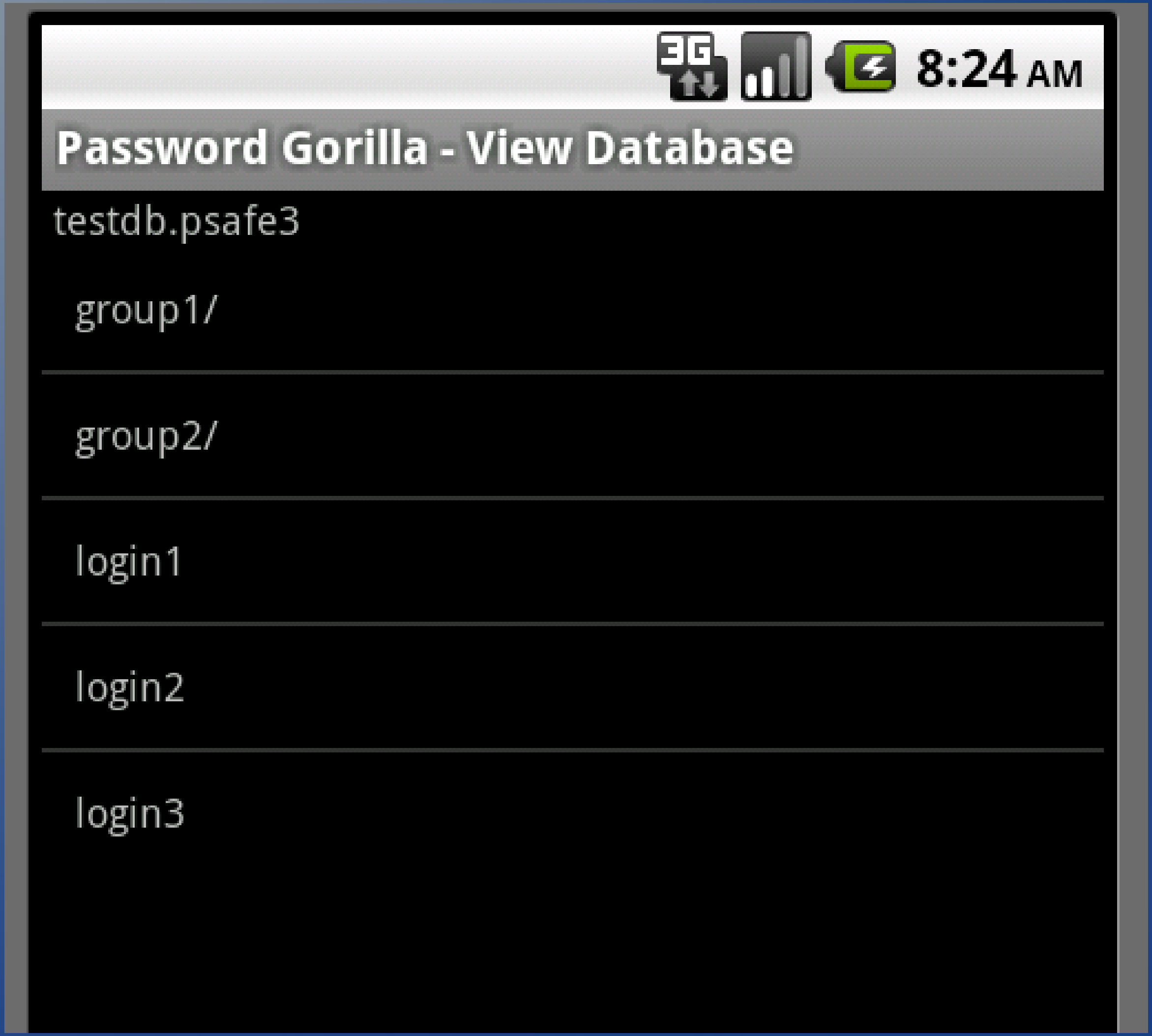
tmp

.android_secure

Android: Launch



Android: Treeview


8:24 AM

Password Gorilla - View Database

testdb.psafe3

group1/

group2/

login1

login2

login3

Hecl: Tcl-like Syntax

Math operations in Polish notation:

```
puts "2 + 2 = [+ 2 2]"
if { < $temp 0 } {   puts "It's freezing" }
```

Command *hash* (instead of array or dict)

```
set foo [hash {a b c d}]
puts [hget $foo a]    ;# prints 'b'
hset $foo c 2
puts [hget $foo c]    ;# prints '2'
puts $foo
# prints 'a b c 2' (although not necessarily in that order)
```

Hecl: Add new command *hello*

Define it in source file *HelloCmd.java:*

```
import org.hecl.Command;
import org.hecl.HeclException;
import org.hecl.Interp;
import org.hecl.Thing;

class HelloCmd implements Command {
    public Thing cmdCode(Interp interp, Thing[] argv)   throws
HeclException {
      System.out.println("Hello world");
      return null;
    }
}
```

Include the following line in *commandline/Hecl.java*:

```
interp.addCommand("hello", new HelloCmd());
```

Hecl: GUI example

```
set context [activity]
set layout [linearlayout -new $context]
$layout setorientation VERTICAL
set layoutparams [linearlayoutparams -new { \
    FILL_PARENT WRAP_CONTENT}]

set tv [textview -new $context -text {Hello World} \
            -layoutparams $layoutparams]

$layout addview $tv
[activity] setcontentview $layout
```

Hecl: The command *java*

Implement new commands with command *java:*

```
hecl> java java.lang.Integer integer
Integer
```

Use the object's methods:

```
# System.out.println( Integer.toHexString(2048i) );
hecl> set hex [integer toHexString 2048]
800
# int decimalNumber = Integer.parseInt(800, 16);
hecl> integer parseInt "800" 16
2048
```

Hecl: GUI example

```
set context [activity]
set layout [linearlayout -new $context]
$layout setorientation VERTICAL
set layoutparams [linearlayoutparams -new { \
    FILL_PARENT WRAP_CONTENT}]

set tv [textview -new $context -text {Hello World} \
                -layoutparams $layoutparams]

$layout addview $tv
[activity] setcontentview $layout
```