libretools - Bug #46

Sourcing PKGBUILDs might change variables used

2012-03-09 08:14 PM - mtjm

Status:	in progress	**% Done:**	20%
Priority:	bug		
Assignee:			
Category:			

Description

E.g. toru used _repo while a PKGBUILD changed the value of this variable, making fullpkg ignore dependencies from the community repo. There should be another solution which wouldn't lead to difficult to debug problems.

History

#2 - 2012-03-16 02:26 PM - fauno

Parsing PKGBUILDs is a mess (see AUR), wouldn't it be better to prefix toru's internal vars ("toru_repo") or something like that?

#3 - 2012-03-16 03:04 PM - mtjm

This problem would be certainly fixed by writing and using a script which would source the PKGBUILD in a different process (properly, with e.g. CARCH defined) and output a script setting specified variables. Then we could implement e.g. caching for it in a single place, so it wouldn't be much slower.

#4 - 2012-06-05 04:19 AM - xihh

Maybe that's the reason why some scripts from «devtools» use subshels when sourcing PKGBUILDs.

#5 - 2012-11-28 07:50 PM - lukeshu

Devtools standards say to not use variables that begin with "_" as that acts as a prefix of variables that are safe to use in PKGBUILDs. And PKGBUILDs should not use variables that do not start with "_" except for the documented ones.

#6 - 2012-12-02 09:38 AM - lukeshu

Also, there are several approaches to this that are used by makepkg and devtools.

- Source them in a subshell and print the results.

```
VAR="$(. FILE; echo $VAR)"
```

- Extract the line that sets it using grep, and `eval` it.

```
eval $(grep '^\s*VAR=' FILE)
```

They both have the possibility to run side-effects. The first is more "correct" and rubust, but is more likely to run unintentional side effects in a poorly written PKGBUILD.

#7 - 2012-12-03 08:35 AM - mtjm

> - Source them in a subshell and print the results. [...]

It will work, not sure about spaces in variables; arrays need different handling. (I have written a script outputting it with nul-separated fields, seems robust although nonobvious to use in shell scripts.)

> - Extract the line that sets it using grep, and `eval` it. [...]

This certainly won't work for makedepends in many mips64el PKGBUILDs, with ifs and +=s.

#8 - 2013-05-02 06:30 PM - lukeshu

mtjm: your script might be over complicated.

Spit out a null-delimited array:

```
printf '%s\0' "${array[@]}"
```

#9 - 2013-05-22 03:07 PM - lukeshu

pkgbuild-check-nonfree now (in git) sources the PKGBUILD in a subshell, so that should be safe. Should it set CARCH?

#10 - 2013-05-23 12:29 AM - mtjm

Maybe source /etc/makepkg.conf? I'm not sure if other variables are used.

#11 - 2013-06-05 08:26 PM - lukeshu

Definitely source makepkg.conf (actually, <tt>load_files makepkg</tt>) if we need to set CARCH. But is setting CARCH important?

Also, as far as exporting arrays from a subshell:

```
eval "$(. FILE &>/dev/null; declare -ap VARNAME)"
```

Other than the possibility of FILE having side-effects, I believe that the only caveat to that is that if it is done in a function, it will be local to that function.

#12 - 2013-06-06 06:25 AM - mtjm

> But is setting CARCH important?

We remove some pkgname array entries depending on CARCH, so setting CARCH affects what toru finds. It is more important in scripts like fullpkg: we add and remove dependencies, changing build orders.

#13 - 2013-06-27 08:38 PM - lukeshu

I've added a "load_PKGBUILD [file]" function to lib/conf.sh, and adjusted all the other programs to use it. It first unsets all PKGBUILD variables and functions, then sets CARCH, then loads the file.

#14 - 2014-01-05 04:17 AM - lukeshu

- Status changed from open to in progress

- % Done changed from 10 to 20

#15 - 2017-05-05 04:43 AM - lukeshu

I'd kind of like to move everything to use makepkg --printsrcinfo. It would be a nice sanitization layer between the PKGBUILD and our runtime. There's a Python parser for the format. I suppose there's also a PHP parser that is part of AUR4. It's an easy format. It would also make it easier to move some of the tooling to a better language than Bash.