NAME
env_parallel - export environment to GNU parallel

SYNOPSIS
env_parallel [--record-env|--session|--end-session] [options for GNU Parallel]

DESCRIPTION
env_parallel is a shell function that exports the current environment to GNU parallel.

If the shell function is not loaded, a dummy script will be run instead that explains how to install the function.

env_parallel is 100 ms slower at startup than pure GNU parallel, and takes up to 30% longer to start a job (typically 15 ms).

Due to the problem with environment space (see below) the recommended usage is either:

# Do --record-env into $PARALLEL_IGNORED_NAMES
env_parallel --session

# Define whatever you want to use
myfunc() { myalias and functions $myvar work. $1.; }
alias myalias='echo Aliases'
myvar='and variables'

# env_parallel will not export names in $PARALLEL_IGNORED_NAMES
env_parallel -S localhost myfunc ::: Hooray

Or:

# Record the "clean" environment (this only needs to be run once)
env_parallel --record-env

# Optionally edit ~/.parallel/ignored_vars (only needed once)

# Define whatever you want to use
myfunc() { myalias and functions $myvar work. $1.; }
alias myalias='echo Aliases'
myvar='and variables'

# Use --env _ to only transfer the names not in the "empty" environment
env_parallel --env_ -S localhost myfunc ::: Hooray

In csh --session is not supported:

# Record the "clean" environment - this only needs to be run once
env_parallel --record-env

# Optionally edit ~/.parallel/ignored_vars - only needed once

# Define whatever you want to use
alias myalias 'echo Aliases $myvar \
set myvar='and variables'

# Use --env _ to only transfer the names not in the "empty" environment
Environment space

By default env_parallel will export all environment variables, arrays, aliases, functions and shell options (see details for the individual shells below).

But this only works if the size of the current environment is smaller than the maximal length of a command and smaller than half of the max if running remotely. E.g. The max size of Bash’s command is 128 KB, so env_parallel will fail if `set | wc -c` is bigger than 128 KB. Technically the limit is in execve(1) which IPC::open3 uses.

Bash completion functions are well-known for taking up well over 128 KB of environment space and the primary reason for causing env_parallel to fail.

Instead you can use --env to specify which variables, arrays, aliases and functions to export as this will only export those with the given name. Or follow the recommended usage in shown in DESCRIPTION.

OPTIONS

Same as GNU parallel in addition to these:

--end-session
    Undo last --session
--record-env
    Record all names currently defined to be ignored every time running env_parallel in the future.
--session
    Ignore all names currently defined. Aliases, variables, arrays, and functions currently defined will not be transferred.
    But names defined after running parallel --session will be transferred.
    This is only valid in the running shell, and can be undone with parallel --end-session.

You can run multiple --session inside each other:

```
env_parallel --session
var=not
# var is not transferred
env_parallel -S localhost 'echo var is $var' ::: ignored
```

SUPPORTED SHELLS

Ash

Installation

Put this in $HOME/.profile:

```
.
`which env_parallel.ash`
```

E.g. by doing:

```
echo './which env_parallel.ash' >> $HOME/.profile
```
Supported use

--env is supported to export only the variable, or alias with the given name. Multiple --envs can be given.

--session is supported.

aliases

  alias myecho='echo aliases'
  env_parallel myecho ::: work
  env_parallel -S server myecho ::: work
  env_parallel --env myecho myecho ::: work
  env_parallel --env myecho -S server myecho ::: work

  alias multiline='echo multiline'
    echo aliases'
  env_parallel multiline ::: work
  env_parallel -S server multiline ::: work
  env_parallel --env multiline multiline ::: work
  env_parallel --env multiline -S server multiline ::: work

functions

  ash cannot list defined functions - thus is not supported.

variables

  myvar=variables
  env_parallel echo "$myvar" ::: work
  env_parallel -S server echo "$myvar" ::: work
  env_parallel --env myvar echo "$myvar" ::: work
  env_parallel --env myvar -S server echo "$myvar" ::: work

arrays

  Arrays are not supported by Ash.

Bash

Installation

Put this in $HOME/.bashrc:

  . `which env_parallel.bash`

E.g. by doing:

  echo `. `which env_parallel.bash`' >> $HOME/.bashrc

Supported use

--env is supported to export only the variable, alias, function, or array with the given name. Multiple --envs can be given.

--session is supported.

aliases

  alias myecho='echo aliases'
  env_parallel myecho ::: work
  env_parallel -S server myecho ::: work
  env_parallel --env myecho myecho ::: work
  env_parallel --env myecho -S server myecho ::: work
alias multiline='echo multiline
    echo aliases'
env_parallel 'multiline {};
    echo but only when followed by a newline' ::: work
env_parallel --env_parallel 'multiline {};
    echo but only when followed by a newline' ::: work
env_parallel --env_parallel --env_parallel 'multiline {};
    echo but only when followed by a newline' ::: work

functions
myfunc() { echo functions $*; }
env_parallel myfunc ::: work
env_parallel --env_parallel --env_parallel 'myfunc' ::: work
env_parallel --env_parallel --env_parallel --env_parallel 'myfunc' ::: work

variables
myvar=variables
env_parallel echo '$myvar' ::: work
env_parallel --env_parallel --env_parallel 'echo $myvar' ::: work
env_parallel --env_parallel --env_parallel --env_parallel 'echo $myvar' ::: work

arrays
myarray=(arrays work, too)
env_parallel --env_parallel --env_parallel --env_parallel --env_parallel --env_parallel 'echo $myarray[{}]' ::: 0 1 2

BUGS
Due to a bug in Bash, aliases containing newlines must be followed by a newline in the command. Some systems are not affected by this bug, but will print a warning anyway.

csh
env_parallel for csh breaks $PARALLEL, so do not use $PARALLEL.

Installation
Put this in $HOME/.cshrc:

source `which env_parallel.csh`

E.g. by doing:

    echo 'source `which env_parallel.csh`' >> $HOME/.cshrc

Supported use

--env is supported to export only the variable, alias, or array with the given name. Multiple --envs can be given.

aliases
GNU Parallel with environment

functions

alias myecho 'echo aliases'
env_parallel myecho ::: work
env_parallel -S server myecho ::: work
env_parallel --env myecho myecho ::: work
env_parallel --env myecho -S server myecho ::: work

variables

set myvar=variables
env_parallel echo '$myvar' ::: work
env_parallel -S server echo '$myvar' ::: work
env_parallel --env myvar echo '$myvar' ::: work
env_parallel --env myvar -S server echo '$myvar' ::: work

arrays with no special chars

set myarray=(arrays work, too)
env_parallel -k echo \$\{myarray\[\{}\]}' ::: 1 2 3
env_parallel -k -S server echo \$\{myarray\[\{}\]}' ::: 1 2 3
env_parallel -k --env myarray echo \$\{myarray\[\{}\]}' ::: 1 2 3
env_parallel -k --env myarray -S server \
escape \$\{myarray\[\{}\]}' ::: 1 2 3

Dash

Installation

Put this in $HOME/.profile:

. `which env_parallel.dash`

E.g. by doing:

echo '. `which env_parallel.dash`' >> $HOME/.profile

Supported use

`--env` is supported to export only the variable, or alias with the given name. Multiple `--envs` can be given.

`--session` is supported.

aliases

alias myecho='echo aliases'
env_parallel myecho ::: work
env_parallel -S server myecho ::: work
env_parallel --env myecho myecho ::: work
env_parallel --env myecho -S server myecho ::: work

alias multiline='echo multiline
            echo aliases'
env_parallel multiline ::: work
env_parallel -S server multiline ::: work
env_parallel --env multiline multiline ::: work
env_parallel --env multiline -S server multiline ::: work
GNU Parallel with environment

functions

dash cannot list defined functions - thus is not supported.

variables

myvar=variables
env_parallel echo "$myvar" :::: work
env_parallel -S server echo "$myvar" :::: work
env_parallel --env myvar echo "$myvar" :::: work
env_parallel --env myvar -S server echo "$myvar" :::: work

arrays

dash does not support arrays.

fish

Installation

Put this in $HOME/.config/fish/config.fish:

source (which env_parallel.fish)

E.g. by doing:

echo 'source (which env_parallel.fish)' \ 
>> $HOME/.config/fish/config.fish

Supported use

--env is supported to export only the variable, alias, function, or array with the given name. Multiple
--envs can be given.

--session is supported.

aliases

alias myecho 'echo aliases'
env_parallel myecho :::: work
env_parallel -S server myecho :::: work
env_parallel --env myecho myecho :::: work
env_parallel --env myecho -S server myecho :::: work

functions

function myfunc
  echo functions $argv
end
env_parallel myfunc :::: work
env_parallel -S server myfunc :::: work
env_parallel --env myfunc myfunc :::: work
env_parallel --env myfunc -S server myfunc :::: work

variables

set myvar variables
env_parallel echo "$myvar" :::: work
env_parallel -S server echo "$myvar" :::: work
env_parallel --env myvar echo "$myvar" :::: work
env_parallel --env myvar -S server echo "$myvar" :::: work
GNU Parallel with environment

arrays

set myarray arrays work, too
env_parallel -k echo "$myarray[{}]' ::: 1 2 3
env_parallel -k -S server echo "$myarray[{}]' ::: 1 2 3
env_parallel -k --env myarray echo "$myarray[{}]' ::: 1 2 3
env_parallel -k --env myarray -S server \  
   echo "$myarray[{}]' ::: 1 2 3

ksh
Installation

Put this in $HOME/.kshrc:

source `which env_parallel.ksh`

E.g. by doing:

echo 'source `which env_parallel.ksh`' >> $HOME/.kshrc

Supported use

--env is supported to export only the variable, alias, function, or array with the given name. Multiple
--envs can be given.

--session is supported.

aliases

alias myecho='echo aliases'
env_parallel myecho ::: work
env_parallel -S server myecho ::: work
env_parallel --env myecho myecho ::: work
env_parallel --env myecho -S server myecho ::: work

alias multiline='echo multiline
   echo aliases'
env_parallel multiline ::: work
env_parallel -S server multiline ::: work
env_parallel --env multiline multiline ::: work
env_parallel --env multiline -S server multiline ::: work

functions

myfunc() { echo functions $*; }
env_parallel myfunc ::: work
env_parallel -S server myfunc ::: work
env_parallel --env myfunc myfunc ::: work
env_parallel --env myfunc -S server myfunc ::: work

variables

myvar=variables
env_parallel echo '$myvar' ::: work
env_parallel -S server echo '$myvar' ::: work
env_parallel --env myvar echo '$myvar' ::: work
env_parallel --env myvar -S server echo '$myvar' ::: work

arrays

myarray=(arrays work, too)
GNU Parallel with environment

env_parallel -k echo '${myarray}[][0]' ::: 0 1 2
env_parallel -k -S server echo '${myarray}[][0]' ::: 0 1 2
env_parallel -k --env myarray echo '${myarray}[][0]' ::: 0 1 2
env_parallel -k --env myarray -S server \
    echo '${myarray}[][0]' ::: 0 1 2

mksh

**Installation**

Put this in $HOME/.mkshrc:

```bash
source `which env_parallel.mksh`
```

E.g. by doing:

```bash
echo 'source `which env_parallel.mksh`' >> $HOME/.mkshrc
```

**Supported use**

--env is supported to export only the variable, alias, function, or array with the given name. Multiple --envs can be given.

--session is supported.

**aliases**

```bash
alias myecho='echo aliases'
env_parallel myecho ::: work
env_parallel -S server myecho ::: work
env_parallel --env myecho myecho ::: work
env_parallel --env myecho -S server myecho ::: work
```

```bash
alias multiline='echo multiline
    echo aliases'
env_parallel multiline ::: work
env_parallel -S server multiline ::: work
env_parallel --env multiline multiline ::: work
env_parallel --env multiline -S server multiline ::: work
```

**functions**

```bash
myfunc() { echo functions $*; }
env_parallel myfunc ::: work
env_parallel -S server myfunc ::: work
env_parallel --env myfunc myfunc ::: work
env_parallel --env myfunc -S server myfunc ::: work
```

**variables**

```bash
myvar=variables
env_parallel echo '$myvar' ::: work
env_parallel -S server echo '$myvar' ::: work
env_parallel --env myvar echo '$myvar' ::: work
env_parallel --env myvar -S server echo '$myvar' ::: work
```

**arrays**

```bash
myarray=(arrays work, too)
env_parallel -k echo '${myarray}[][0]' ::: 0 1 2
env_parallel -k -S server echo '${myarray}[][0]' ::: 0 1 2
```
GNU Parallel with environment

```
env_parallel -k --env myarray echo "$\{myarray[{}]\}" :::: 0 1 2
env_parallel -k --env myarray -S server \n  echo "$\{myarray[{}]\}" :::: 0 1 2
```

**pdksh**

**Installation**

Put this in $HOME/.profile:

```
source `which env_parallel.pdksh`
```

E.g. by doing:

```pdksh
echo 'source `which env_parallel.pdksh`' >> $HOME/.profile
```

**Supported use**

--env is supported to export only the variable, alias, function, or array with the given name. Multiple --envs can be given.

--session is supported.

**aliases**

```
alias myecho="echo aliases";
env_parallel myecho :::: work;
env_parallel -S server myecho :::: work;
env_parallel --env myecho myecho :::: work;
env_parallel --env myecho -S server myecho :::: work
```

**functions**

```
myfunc() { echo functions $*; };
env_parallel myfunc :::: work;
env_parallel -S server myfunc :::: work;
env_parallel --env myfunc myfunc :::: work;
env_parallel --env myfunc -S server myfunc :::: work
```

**variables**

```
myvar=variables;
env_parallel echo "$\{myvar\}" :::: work;
env_parallel -S server echo "$\{myvar\}" :::: work;
env_parallel --env myvar echo "$\{myvar\}" :::: work;
env_parallel --env myvar -S server echo "$\{myvar\}" :::: work
```

**arrays**

```
myarray=(arrays work, too);
env_parallel -k echo "$\{myarray[{}]\}" :::: 0 1 2;
env_parallel -k -S server echo "$\{myarray[{}]\}" :::: 0 1 2;
env_parallel -k --env myarray echo "$\{myarray[{}]\}" :::: 0 1 2;
env_parallel -k --env myarray -S server \n  echo "$\{myarray[{}]\}" :::: 0 1 2
```

**sh**
Installation

Put this in $HOME/.profile:

```
. `which env_parallel.sh`
```

E.g. by doing:

```
echo '. `which env_parallel.sh`' >> $HOME/.profile
```

Supported use

--env is supported to export only the variable, or alias with the given name. Multiple --envs can be given.

--session is supported.

aliases

```
sh does not support aliases.
```

functions

```
myfunc() { echo functions $*; }
env_parallel myfunc ::: work
env_parallel -S server myfunc ::: work
env_parallel --env myfunc myfunc ::: work
env_parallel --env myfunc -S server myfunc ::: work
```

variables

```
myvar=variables
env_parallel echo '$myvar' ::: work
env_parallel -S server echo '$myvar' ::: work
env_parallel --env myvar echo '$myvar' ::: work
env_parallel --env myvar -S server echo '$myvar' ::: work
```

arrays

```
sh does not support arrays.
```

tcsh

```
env_parallel for tcsh breaks $PARALLEL, so do not use $PARALLEL.
```

Installation

Put this in $HOME/.tcshrc:

```
source `which env_parallel.tcsh`
```

E.g. by doing:

```
echo 'source `which env_parallel.tcsh`' >> $HOME/.tcshrc
```

Supported use

--env is supported to export only the variable, alias, or array with the given name. Multiple --envs can be given.

aliases

```
alias myecho 'echo aliases'
env_parallel myecho ::: work
```
GNU Parallel with environment

```
env_parallel -S server myecho ::: work
env_parallel --env myecho myecho ::: work
env_parallel --env myecho -S server myecho ::: work
```

functions

Not supported by `tcsh`.

variables

```
set myvar=variables
env_parallel echo "$myvar" ::: work
env_parallel -S server echo "$myvar" ::: work
env_parallel --env myvar echo "$myvar" ::: work
env_parallel --env myvar -S server echo "$myvar" ::: work
```

arrays with no special chars

```
set myarray=(arrays work, too)
env_parallel -k echo \$'{myarray[]}[]' ::: 1 2 3
env_parallel -k -S server echo \$'{myarray[]}[]' ::: 1 2 3
env_parallel -k --env myarray echo \$'{myarray[]}[]' ::: 1 2 3
env_parallel -k --env myarray -S server \
    echo \$'{myarray[]}[]' ::: 1 2 3
```

Zsh

Installation

Put this in `$HOME/.zshrc`:

```
. `which env_parallel.zsh`
```

E.g. by doing:

```
echo `. `which env_parallel.zsh`' >> $HOME/.zshenv
```

Supported use

--env is supported to export only the variable, alias, function, or array with the given name. Multiple --envs can be given.

--session is supported.

aliases

```
alias myecho='echo aliases'
env_parallel myecho ::: work
env_parallel -S server myecho ::: work
env_parallel --env myecho myecho ::: work
env_parallel --env myecho -S server myecho ::: work
```

```
alias multiline='echo multiline
    echo aliases'
env_parallel multiline ::: work
env_parallel -S server multiline ::: work
env_parallel --env multiline multiline ::: work
env_parallel --env multiline -S server multiline ::: work
```

functions

```
myfunc() { echo functions $*; }
```
GNU Parallel with environment

variables

myvar=variables
env_parallel echo "$myvar" :::: work
env_parallel -S server echo "$myvar" :::: work
env_parallel --env myvar echo "$myvar" :::: work
env_parallel --env myfunc -S server myfunc :::: work

arrays

myarray=(arrays work, too)
env_parallel -k echo "$myarray[{}][]" :::: 1 2 3
env_parallel -k -S server echo "$myarray[{}][]" :::: 1 2 3
env_parallel -k --env myarray echo "$myarray[{}][]" :::: 1 2 3
env_parallel -k --env myvar -S server \ 
    echo "$myarray[{}][]" :::: 1 2 3

EXIT STATUS

Same as GNU parallel.

AUTHOR

When using GNU env_parallel for a publication please cite:


This helps funding further development; and it won't cost you a cent. If you pay 10000 EUR you should feel free to use GNU Parallel without citing.

Copyright (C) 2007-10-18 Ole Tange, http://ole.tange.dk

Copyright (C) 2008-2010 Ole Tange, http://ole.tange.dk

Copyright (C) 2010-2022 Ole Tange, http://ole.tange.dk and Free Software Foundation, Inc.

LICENSE

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 3 of the License, or at your option any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Documentation license I

Permission is granted to copy, distribute and/or modify this documentation under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included in the file LICENSES/GFDL-1.3-or-later.txt.
Documentation license II

You are free:

to Share

to copy, distribute and transmit the work

to Remix

to adapt the work

Under the following conditions:

Attribution

You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).

Share Alike

If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

With the understanding that:

Waiver

Any of the above conditions can be waived if you get permission from the copyright holder.

Public Domain

Where the work or any of its elements is in the public domain under applicable law, that status is in no way affected by the license.

Other Rights

In no way are any of the following rights affected by the license:

- Your fair dealing or fair use rights, or other applicable copyright exceptions and limitations;

- The author’s moral rights;

- Rights other persons may have either in the work itself or in how the work is used, such as publicity or privacy rights.

Notice

For any reuse or distribution, you must make clear to others the license terms of this work.

A copy of the full license is included in the file as LICENCES/CC-BY-SA-4.0.txt

DEPENDENCIES

env_parallel uses GNU parallel.

SEE ALSO

parallel(1), ash(1), bash(1), csh(1), dash(1), fish(1), ksh(1), pdksh(1) tcsh(1), zsh(1).