# Basic Shell Tools, des outils en shell

Bernard Tatin

2017

---

Voici des outils plus ou moins simples, écrits en shell *bash* en général car il est certainement le plus répandu actuellement. Certains ne trouveront d'intérêt que pour moi tandis que d'autres pourront satisfaire un plus grand nombre de lecteurs.

Le choix de 'noweb' provient du simple fait que documentation et sources sont conçus en même temps.[a]

---

[a] Document crée le November 16, 2017à 8:12

# Contents

# I introduction

L'arborescence des scripts est simple, nous avons un répertoire **bin** qui contient les scripts à exécuter, un répertoire **include** qui contient les scripts de configuration globale à sourcer. Le reste sert *simplement* à gérer la documentation et les tests.

## I.1 la licence

C'est la license du *MIT* :

3 ⟨*license.sh* 3⟩≡

```
## The MIT License (MIT)
##
## Copyright (c) 2017 Bernard Tatin
##
## Permission is hereby granted, free of charge, to any person obtaining a copy
## of this software and associated documentation files (the "Software"), to deal
## in the Software without restriction, including without limitation the rights
## to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
## copies of the Software, and to permit persons to whom the Software is
## furnished to do so, subject to the following conditions:
##
```

```
## The above copyright notice and this permission notice shall be included in all
## copies or substantial portions of the Software.
##
## THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
## IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
## FITNESS FOR A PARTICULAR PURPOSE AND NON INFRINGEMENT. IN NO EVENT SHALL THE
## AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
## LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
## OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
## SOFTWARE.
##
```

This code is used in chunk 8.

---

# II le répertoire `include`

Les fichiers de ce répertoire ne sont utiles que lorsqu'ils sont *sourcés*, soit depuis un shell interactif, soit depuis un script. Pour les shells interactifs, il est conseillé de bien vérifier que tout ce qui est nécessaire est bien exporté.

## II.1 le fichier `common-bash.bash`

### II.1.1 les variables

Pour le moment, c'est le seul fichier. Il permet d'exporter une variable, le nom du script :

5 ⟨*common-bash variable* 5⟩≡

```
export script_name=$(basename $0)
```

This code is used in chunk 8.
Defines:
`script_name`, never used.

## II.1.2 les fonctions

### II.1.2.1 onerror

Ensuite, nous avons une série de fonctions très utiles, bien entendu. Tout d'abord, la fonction `onerror` qui affiche un message sur la console d'erreur et force la fin du script. Ici, la redirection ne fonctionne pas avec `sh`.

6a ⟨*common-bash functions* 6a⟩≡

```
function onerror() {
    local exit_code=$1
    shift
    local error_msg="$*"

    echo "ERROR: $error_msg" 1>&2
    exit $exit_code
}
```

This definition is continued in chunk 7.
This code is used in chunk 8.
Defines:
`onerror`, used in chunks 6b and 7.

Voici un exemple d'utilisation qui affiche *ERROR: code 127.x, unknown database error* et force le script à sortir avec une valeur de retour de 2:

6b ⟨*onerror example* 6b⟩≡

```
onerror 2 "ERROR: code 127.x, unknown database error"
```

Root chunk (not used in this document).
Uses `onerror` 6a.

### II.1.2.2 les autres fonctions... à compléter

7 ⟨common-bash functions 6a⟩+≡

```
function safe_source() {
  local file=
  while [ $# -gt 0 ]
  do
  file=$1
    if [ -f ${file} ]
    then
      source ${file}
    else
      onerror 1 "Cannot source ${file}"
    fi
    shift
  done
}

function get_tmp_file() {
  local root_name='another-tmp-file'
  [ $# -gt 0 ] && \
    root_name=$1
  $(mktemp /tmp/${root_name}.XXXXXX)
}

function dohelp() {
  local exit_value=0
  local error_message=

  case "$#" in
    '1')
      exit_value=$1
      ;;
    '2')
      exit_value=$1
      shift
      error_message="$*"
      ;;
  esac
  [ -n "$error_message" ] && \
    echo "ERROR : @" 1>&2

  /usr/bin/printf "${help_text}"

  exit ${exit_value}
}
```

This code is used in chunk 8.
Uses onerror 6a.

Ce qui nous donne au final:

8 ⟨*common-bash.bash* 8⟩≡

```
#!/usr/bin/env bash

⟨license.sh 3⟩

⟨common-bash variable 5⟩

⟨common-bash functions 6a⟩
```

Root chunk (not used in this document).

# III annexes

## III.1 tables et index

### III.1.1 table des extraits de code



### III.1.2 index