

e-rcscan: Resource File Scanner

Karel Kubat
e-tunity

2000 ff.

Contents

1	Introduction	2
2	Invocation	2
3	Real-life Example	2
3.1	Loading the configuration in a C program	3
3.2	Loading the configuration in a shell script	3
3.3	Loading the configuration in a Makefile	4
4	Installation	4

1 Introduction

`e-rcscan` is a very simple tool that uses `e-lib`'s functionality to read a resource file. The results are shown on `stdout`. Optionally, `e-rcscan` formats its output in such a way that the result can be expanded in a shell, to form environment variables, or in a Makefile.

Please see the description of `e-lib`, notably the functions `rc_read()` and `rc_value()`, for an explanation of the resource file handling.

2 Invocation

The basic invocation of `e-rcscan` is `e-rcscan [flags] resource [resource...]`. The stated resource files are scanned. The flags may be `-s`, to force shell-style output, `-m` to force Makefile-like output, or `-c`, to force (t)csh style output.

For a complete explanation of the flags, start `e-rcscan` without arguments to see the usage information.

3 Real-life Example

This section presents a short real-life example of a hypothetical configuration `/etc/my.conf`, used in some given program. E.g., assume that the configuration looks like this:

```
1  # /etc/my.conf -- configuration file
2
3  # Files and directories
4  rootdir = /opt/mypackage           # root of the installation
5  bindir = $(rootdir)/bin           # where are the binaries
6  scanner = $(bindir)/scanner
7  docdir = $(rootdir)/doc           # where are the docs
8  docstart = index.html             # first help page
9
10 # How do we fire up the doc index
11 helpcmd = netscape file://localhost/$(docstart)
12
13 # Error messages
14 err_nobin = Sorry, I failed to locate the directory to hold the \
15             helper programs. It is configured as "$(bindir)".
16 err_nodoc = Sorry, I failed to locate the directory to hold the \
17             documentation.
18 err_noscanner = Sorry, I failed to locate the "scanner" program.
```

3.1 Loading the configuration in a C program

To load the configuration in a C program, the following stanza may be used (for an exact description please see the documentation on *e-lib*.)

```

1  #include <e-lib.h>
2
3  if (rc_read ("/etc/my.conf") < 0)
4      error ("Failed to read configuration /etc/my.conf!\n");
5  if (chdir (rc_value ("bindir")))
6      error (rc_value ("err_nobin"));
7  if (access (rc_value ("scanner"), X_OK))
8      error (rc_value (err_noscanner));
9  .
10 . and so on
11 .

```

3.2 Loading the configuration in a shell script

Now let's assume that you need the same configuration values in a shell script, for example to fire up Netscape to read the index of the help pages. You wouldn't want to "hardwire" the path `/opt/mypackage/doc/index.html` in your shell script, that's what the resource is for!

The tool `e-rcscan` now comes in handy. The stanza might be:

```

1  #!/bin/sh
2
3  # Sample shellscript
4  # -----
5
6  # Load the values from /etc/my.conf into the environment. We're using
7  # a shell-style syntax (-s) and want to have the variables in uppercase
8  # (-u), so the flags are: -su.
9  # The eval '...' tells the shell to evaluate e-rcscan's output right away.
10 eval 'e-rcscan -su /etc/my.conf'
11
12 # Let's see if we got a HELPCMD from the resource. If not, abort.
13 if [ -z "$HELPCMD" ] ; then
14     echo "Oops.. Failed to get the HELPCMD from /etc/my.conf" 2>&1
15     echo "Perhaps a misconfiguration?!" 2>&1
16     exit 1
17 fi
18
19 # Yup, we got a HELPCMD. Let's try to run it.
20 "$HELPCMD"
21 if [ $? -ne 0 ] ; then
22     echo "Oops.. $HELPCMD didn't work." 1>&2
23     echo "Please check the configuration in /etc/my.conf!" 1>&2
24     exit 1
25 fi

```

3.3 Loading the configuration in a Makefile

Similar to the examples above, you probably wouldn't want a separate Makefile configuration, but would want to use `/etc/my.conf` too. The approach would be to convert `/etc/my.conf` into a set of make-compatible commands, e.g. using `e-rcscan -mu /etc/my.conf > config.inc`. The file `config.inc` would then be included in the Makefile, as in:

```
1 # Sample Makefile.
2 # -----
3
4 include config.inc
5
6 $(SCANNER): scanner
7     install -s scanner $(BINDIR)
```

4 Installation

The installation of `e-rcscan` is very simple and straight-forward. Obtain the archive and unpack it, then `chdir` into the created directory `e-rcscan/` and do `make install`. This will install the program `e-rcscan` to `/usr/e/bin/`, the default program directory of the `e` user.

A successful installation will obviously require the presence of `e-lib`.