


Debug, Parser, bashrc, profile,
bash_history,
\$PS1... e TTY



Debug

Debug (em português: Depuração, *depurar*) é o processo de encontrar e reduzir defeitos num aplicativo de software ou mesmo em hardware. Erros de software incluem aqueles que previnem o programa de ser executado e aqueles que produzem um resultado inesperado. De forma geral, a depuração é uma tarefa difícil e trabalhosa, e a dificuldade varia de acordo com o ambiente de desenvolvimento.

A photograph showing two men in a dimly lit office environment, focused on a large computer monitor. The man in the foreground is wearing glasses and a dark shirt, looking intently at the screen. The man behind him is also looking at the screen. The monitor displays a complex software interface with various panels and text. The scene is illuminated by a desk lamp, creating a professional and concentrated atmosphere.

***"Desenvolvimento:
10% programando
90% debugando !!!"***

Formas de **Debugar** em **Shell**

www.terminalroot.com.br

- ❏ **-n** (testa se há erros de sintaxe no código, tudo é somente testado, mas não executado)

bash -n meuscript.sh

- ❏ **echo** (usar o comando echo antes e depois de uma linha, é trabalhoso mais muito eficiente para encontrar erros)

- ❏ **-X** (o Shell exibe na tela os comandos na medida que eles são executados)

bash -x meuscript.sh

- ❏ **-V** (mostra a linha atual do aplicativo que está sendo executado)

bash -v meuscript.sh

- ❏ **set** (o comando pode setar a opção entre as linha que vc deseja)

- ❏ **\$?** (essa variável retorna o código do último comando executado)

- ❏ Outras formas, usar: trap read, condição,...

Existe um endereço que procura erros nos seus scripts online

www.shellcheck.net

Parser

www.terminalroot.com.br

É uma análise de um documento ou arquivo , buscando informações de configurações.

Em **Unix** e **Linux** percebam que há arquivos textos ocultos (iniciados com '.', ponto) geralmente até tem o nome **rC** depois, exemplo: **.bashrc**, **.netrc** ,falamos sobre ele no video: **Dicas Importantes para seu Blog no GitHub** ([terminalroot.com.br/2017/02/dicas-importantes-para-seu-blog-no-github.html](https://www.terminalroot.com.br/2017/02/dicas-importantes-para-seu-blog-no-github.html)),...entre outros, que nada mais são arquivos de configuração lidos por um **parser** pra os aplicativos, você altera ali e muda o funcionamento do programa, sem precisar mexer no código do programa propriamente dito, mais ou menos o que fizemos no video sobre **CGI**.

parser.sh

```
#!/bin/bash
_FILES="arquivo.cfg"
while read LINHA; do
    # ignora as linhas comentadas
    [ "$(echo $LINHA | cut -c1)" = '#' ] && continue
    # pega a versão do .arquivo.cfg
    VERSION=$*
done < $VERSION
```

Dica:

Conheça o **Shell::Parser**

Um analisador de Shell feito em Perl

search.cpan.org/~saper/Shell-Parser-0.04/lib/Shell/Parser.pm

Arquivos de ambiente

Você pode customizar seu ambiente do **Bash** utilizando alguns arquivos como

- ✓ **.bash_profile** - Este arquivo fica localizado no diretório pessoal de cada usuário. É executado por shells que usam autenticação (nome e senha). Ele contém comandos que são executados para o usuário no momento do login no sistema após o `/etc/profile`.
- ✓ **.bashrc** - Possui as mesmas características do `.bash_profile` mas é executado por shells que não requerem autenticação (como uma seção de terminal no X). Os comandos deste arquivo são executados no momento que o usuário inicia um shell.
- ✓ **/etc/profile** - Este arquivo contém comandos que são executados para todos os usuários do sistema no momento de autenticação. Somente o usuário root pode ter permissão para modificar este arquivo.

Quando é carregado através de um shell que requer autenticação (nome e senha), o bash procura estes arquivos em sequência e executa os comandos contidos, caso existam.

1. **/etc/profile**
2. **~/.bash_profile** => **~/.bashrc**
3. **~/.bash_login**
4. **~/.profile** => **~/.bashrc**

Sequência lógica

execute **/etc/profile**

IF **~/.bash_profile** exists THEN
execute **~/.bash_profile**

ELSE

IF **~/.bash_login** exist THEN
execute **~/.bash_login**

ELSE

IF **~/.profile** exist THEN
execute **~/.profile**

END IF

END IF

END IF

+ **.bash_login**

Só existe, se não existir o **.bash_profile**

+ **.bash_logout**

Só existe, se se o usuário estiver logado

.bash_history

Nele fica guardado todos os comandos executados no **Shell**.

history # exibe e numera os comandos que foram executados
cat .bash_history # exibe somente os comandos que foram executados

history -c # limpa o histórico de comandos e seus respectivos numeros, mas não limpa o **.bash_history**

cat /dev/null > ~/.bash_history # limpa o histórico, mas não limpa o history

!! # executa o ultimo comando

fc -s # mesmo que o acima !!

!1120 # executa o comando 1120 se houver no history

history 10 # mostrar os 10 ultimos comandos executados

!nmb # executa o último comando que começa com nmb

fc -s nmb # mesmo que o acima !nmb

Mais:

man history

fc [-e ename] [-lnr] [first] [last] or fc -s [pat=rep] [command]

TTY

TTY é uma abreviatura de **TeleTypewriter**. No início do **UNIX** eles deram esse nome a "tela que vc pode escrever" seria uma "máquina de escrever virtual", e porque existe **TTY1**, **TTY2**,... ela é a **Shell**, **terminal**, **console**... só que fora do ambiente gráfico, experimente dar o comando: **tty**

Dica: abra o **terminal**, depois abra uma nova aba e mantenha a primeira aberta, rode esse comando na segunda aba:

ls /home/\$USER/ > /dev/pts/1

depois volte pra primeira aba e veja se está listado os arquivos/diretório da sua **home** lá! Mais: **man tty || tty --help**

Variáveis PS1, PS2 ... PS4 ...

\$PS1 - Guarda o valor do prompt primário. Ex.: **echo \$PS1**, perceba que há um comando com essa sequencia: **\u@lh**: que é justamente **usuario@host**:

Dica de pesquisa

Pesquise sobre a variável **\$TMOUT**
e descubra uma dica de segurança no seu terminal!