Relatorio Projeto Shell Script

Nome: Wescley Júnior Gonçalves Navarro

Problema: Manter diretórios de trabalho ou de downloads organizados em um ambiente Linux pode ser uma tarefa repetitiva e propensa a desordem. Arquivos de diferentes tipos (imagens, documentos, scripts) acumulam-se rapidamente, dificultando a localização e o gerenciamento. A organização manual consome tempo e exige verificação constante de extensões ou tipos de arquivos.

Solução Proposta: O script desenvolvido, organizer.sh, automatiza a organização de arquivos no diretório atual. Ele lê um arquivo de configuração externo (rules.conf) que define regras de associação entre tipos de arquivos (baseado em MIME type ou extensão) e diretórios de destino. O script oferece um modo de simulação para segurança, evita processar a si mesmo ou arquivos de configuração, previne sobrescrita acidental e gera um log de operações (organizer.log). Esta solução atende aos requisitos de obter informações externas (arquivo de regras) e produzir informações externas (arquivo de log).

#	Conceito	Aula	Linha	Trecho de Código	Descrição
01	10	Parâmetros e Variáveis	4	RULES="rules.conf"	Definição de variável simples.
02	12	Parâmetros e Variáveis	9	declare -A processados	Declaração de variável do tipo array associativo.
03	11	Parâmetros e Variáveis	12	case "$1" in	Uso de parâmetro posicional $1 para argumentos.
04	15	Estruturas de fluxo condicionais	12	case "$1" in	Estrutura de controle de fluxo case.
05	1	Comandos básicos	21	exit 1	Uso do comando exit para encerrar script com erro.
06	14	Estruturas de fluxo condicionais	26	if [[! -e "$RULES"]]; then	Teste condicional if com flag de existência de arquivo.
07	17	Estruturas de fluxo de repetição	32	while read -r linha; do	Estrutura de repetição while para leitura de entrada
08	16	Estruturas de fluxo condicionais	34	[[... || ...]]	Uso do operador lógico "ou" ||`
09	9	Expansões	38	qtd=${#colunas[@]}	Expansão de parâmetro para contar o tamanho do array.
10	13	Parâmetros e Variáveis	54	${DESTINO/#\~/$HOME}	Expansão avançada para substituição de padrão em string.
11	3	Sistema de arquivos	57	MEU_NOME=$(basename "$0")	Uso do comando basename para manipulação de nomes.
12	18	Expressões Regulares	78	[[... =~ ...]]	Uso do operador de expressão regular (=~) no teste.
13	7	Entradas e saídas	102	>> "$LOG"	Redirecionamento de saída padrão (stdout) para append em arquivo.
14	8	Pipes ou redirecionamentos	115	grep ... | wc -l	Uso de pipe |`
15	2	Processamento de texto	115	... | wc -l	Uso do comando wc para contagem de linhas.

```
1   #!/bin/bash
2
3   # Variaveis que serao utilizadas no programa
4   RULES="rules.conf"                  # Arquivo de regras
5   LOG="organizer.log"                 # Arquivo de log
6   SIMULACAO=true
7
8   # Array associativo para rastrear arquivos já movidos (Evita duplicidade)
9   declare -A processados
10
11  # Verifica argumentos passados (Simulacao vs Execucao)
12  case "$1" in
13          "-y")
14                  SIMULACAO=false
15                  ;;
16          "")
17                  echo "Modo de Simulacao Ativo. Use -y para executar."
18                  ;;
19          *)
20                  echo "Argumento invalido. Tente ./organize [-y]"
21                  exit 1
22                  ;;
23  esac
24
25  # Verifica a existencia do arquivo rules.conf
26  if [[ ! -e "$RULES" ]]; then
27          echo "O arquivo de regras ($RULES) não foi encontrado"
28          exit 1
29  fi
30
31  # Lê o arquivo linha a linha ate o EOF
32  while read -r linha; do
33          # Ignora linhas vazias ou comentarios
34          [[ -z "$linha" || "$linha" == \#* ]] && continue
35
36          # Transforma a linha em um array para contar colunas
37          colunas=($linha)
38          qtd=${#colunas[@]}
39
40          # Define as variaveis com base na quantidade de colunas
41          if [ "$qtd" -eq 2 ]; then
42                  TIPO="${colunas[0]}"
43                  REGEX_NOME=""
44                  DESTINO="${colunas[1]}"
45          elif [ "$qtd" -eq 3 ]; then
46                  TIPO="${colunas[0]}"
47                  REGEX_NOME="${colunas[1]}"
48                  DESTINO="${colunas[2]}"
49          else
50                  continue
51          fi
52
```

```
53          # Expansão para corrigir o "~" no destino
54          DESTINO="${DESTINO/#\~/$HOME}"
55
56          # Define o nome do script limpo (sem ./)
57          MEU_NOME=$(basename "$0")
58
59          # Itera sobre os arquivos da pasta atual
60          for arquivo in *; do
61                  # 1. Filtro de Segurança: Ignora script, config, log e o
62                  # próprio diretório destino
63                  if [ ! -f "$arquivo" ] || [ "$arquivo" == "$RULES" ] \
64                          || [ "$arquivo" == "$MEU_NOME" ] || [ "$arquivo" == "$LOG" ];
65                          then
66                          continue
67                  fi
68
69                  # 2. Filtro de Duplicidade: Se já foi processado nesta execução, pula
70                  if [[ -n "${processados[$arquivo]}" ]]; then
71                          continue
72                  fi
73
74                  # Obtém o tipo Mime
75                  MIMETYPE=$(file --mime-type -b "$arquivo")
76
77                  # Logica de Filtro (Mime/Extensao E Nome opcional)
78                  if [[ "$MIMETYPE" =~ "$TIPO" ]] || [[ "$arquivo" =~ \."$TIPO"$ ]]; then
79
80                          if [ -n "$REGEX_NOME" ]; then
81                                  if [[ ! "$arquivo" =~ $REGEX_NOME ]]; then
82                                          continue
83                                  fi
84                          fi
85
86                          echo "Processando: $arquivo ($MIMETYPE) -> $DESTINO"
87
88                          # Marca o arquivo como processado para não cair em outras regras
89                          processados["$arquivo"]=1
90
91                          if "$SIMULACAO"; then
92                                  continue
93                          else
94                                  # Cria diretorio e move arquivos
95                                  if [ ! -d "$DESTINO" ]; then
96                                          mkdir -p "$DESTINO"
97                                  fi
98
99                                  if mv "$arquivo" "$DESTINO"; then
100                                         echo "$(date "+%d/%m/%Y %H:%M:%S") - " \
101                                         "Movido: $arquivo" \
102                                         " -> $DESTINO" >> "$LOG"
103                                 else
104                                         echo "Erro ao mover: $arquivo" >&2
```

```
105                                     fi
106                                 fi
107                             fi
108             done
109     done < "$RULES"
110
111     # Exibe resumo final lendo do log
112     if [ "$SIMULACAO" = false ] && [ -f "$LOG" ]; then
113             echo "---"
114             echo "Total de arquivos movidos hoje:"
115             grep "$(date "+%d/%m/%Y")" "$LOG" | wc -l
116     fi
```

```
[wescleyj@acer-aspire Organizacao-arquivos-automatizados]$ cloc organizer.sh
       1 text file.
       1 unique file.
       0 files ignored.

github.com/AlDanial/cloc v 2.06  T=0.01 s (84.4 files/s, 9786.6 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Bourne Shell                     1             19             19             78
-------------------------------------------------------------------------------
[wescleyj@acer-aspire Organizacao-arquivos-automatizados]$
```