[ARCHIWUM] Message of the day - MOTD

> **Strona zarchiwizowana.** Treść została wchłonięta i rozbudowana. Aktualna wersja: Komunikat powitalny: MOTD i banery. Ta strona zostaje wyłącznie dla historii.

Message Of The Day (MOTD)

> **Co to jest MOTD?**

Message Of The Day (MOTD) to tekst wyświetlany po zalogowaniu do systemu, zazwyczaj zawiera powitanie lub najważniejsze informacje o systemie. W systemach Unix/Linux plik `/etc/motd` jest prostym plikiem tekstowym, który nie interpretuje poleceń, wyświetla jedynie statyczną zawartość.

> **Dlaczego plik /etc/motd nie może wykonywać poleceń?**

Plik `/etc/motd` jest tylko zwykłym plikiem tekstowym i nie jest interpretowany jako skrypt ani polecenie. Aby mieć dynamiczne i kolorowe wiadomości, tworzymy osobny skrypt Bash, który jest wywoływany podczas logowania przez plik startowy powłoki.

Skrypt motd – sekcje i wyjaśnienia poleceń

```
#!/bin/bash
```

```
#!/usr/local/bin/bash
```

>

Wyjaśnienie: Ścieżka do interpretera bash w systemach, gdzie bash jest zainstalowany w /usr/local/bin.

```
RED='\033[31m'
GREEN='\033[32m'
YELLOW='\033[33m'
CYAN='\033[36m'
RESET='\033[0m'
```

Wyjaśnienie: Definiujemy zmienne reprezentujące kody ANSI do kolorowania tekstu na terminalu.

```
echo -e "${CYAN} __  __ _____ ____  ____  ______     __"echo -e "|  \\/  |_   _|___ \\/
___||  _ \\ \\   / /"echo -e "| |\\/| | | |   __) \\___ \\| |_) \\ \\ / / "echo -e "| |  | | |
|  / __/ ___) |  _ < \\ V /  "echo -e "|_|  |_| |_| |_____|____/|_| \\_\\ \\_/   ${RESET}"
```

Wyjaśnienie: Wyświetlamy kolorowy napis ASCII art symbolizujący nazwę serwera w stylu retro. Polecenie `echo -e` interpretuje kody ucieczki dla kolorów, a instrukcje formatowania nadają kolor i resetują go po wyświetleniu.

```
LOAD=$(sysctl -n vm.loadavg | awk '{print $2, $3, $4}')
```

Wyjaśnienie: Pobieramy średnie obciążenie CPU (load average) korzystając z `sysctl`, a `awk` wyciąga drugą, trzecią i czwartą wartość z wyniku (odpowiednio 1-, 5- i 15-minutowy load).

```
MEM_TOTAL=$(sysctl -n hw.physmem)
MEM_TOTAL_MB=$((MEM_TOTAL / 1024 / 1024))
PAGE_SIZE=$(sysctl -n hw.pagesize)
FREE_PAGES=$(sysctl -n vm.stats.vm.v_free_count)
MEM_FREE_BYTES=$((FREE_PAGES * PAGE_SIZE))
MEM_FREE_MB=$((MEM_FREE_BYTES / 1024 / 1024))
```

Wyjaśnienie: Pobieramy całkowitą pamięć RAM i wyliczamy jej wartość w megabajtach. Następnie obliczamy ilość wolnej pamięci na podstawie liczby wolnych stron pamięci i rozmiaru strony. Wynik również jest konwertowany na megabajty.

```
echo -e "${YELLOW}Aktualne obciążenie CPU:${RESET} ${GREEN}${LOAD}${RESET}"
echo -e "${YELLOW}Pamięć całkowita:${RESET} ${GREEN}${MEM_TOTAL_MB} MB${RESET}"
echo -e "${YELLOW}Wolna pamięć:${RESET} ${GREEN}${MEM_FREE_MB} MB${RESET}"
```

Wyjaśnienie: Wyświetlamy pobrane wartości obciążenia CPU i pamięci RAM z kolorami, nadając nagłówkom i wartościom różne kolory dla czytelności.

```
df -h | awk -v CYAN="${CYAN}" -v GREEN="${GREEN}" -v RESET="${RESET}" '
NR>1 {
    free_perc = 100 - gensub(/%/, "", "g", $5)
    printf "%s%s%s: %s%d%%%s  ", CYAN, $6, RESET, GREEN, free_perc, RESET
}
END {print ""}'
```

Wyjaśnienie: Polecenie `df -h` pokazuje rozmiary systemów plików w czytelnej formie. Za pomocą `awk` zliczamy procent wolnego miejsca (100 minus procent użycia z kolumny 5) i wyświetlamy punkt montowania (kolumna 6) razem z procentem wolnego miejsca, kolorując je odpowiednio.

```
count_process() {
  ps aux | grep -w "$1" | grep -v grep | wc -l
}

COUNT_GAME=$(count_process "game")
COUNT_DB=$(count_process "db")
COUNT_AUTH=$(count_process "auth")
```

Wyjaśnienie: Definiujemy funkcję, która liczy liczbę procesów zawierających podany ciąg znaków w pełnej nazwie procesu. Następnie zliczamy liczbę obecnie działających procesów: "game", "db" oraz "auth".

```
color_status() {
  if [ "$1" -gt 0 ]; then
    echo -e "${GREEN}RUNNING (${1})${RESET}"
  else
    echo -e "${RED}NOT RUNNING${RESET}"
  fi
}
```

```
echo -e "${YELLOW}Status procesów gry MT2SRV:${RESET}"
echo -e "game: $(color_status $COUNT_GAME)"
echo -e "db: $(color_status $COUNT_DB)"
echo -e "auth: $(color_status $COUNT_AUTH)"
```

> **Wyjaśnienie:** Funkcja `color_status` przyjmuje liczbę procesów i zwraca tekst z informacją, czy procesy są uruchomione (zielony napis) lub nie (czerwony napis). Następnie wypisujemy status każdej aplikacji gry.

```
echo -e "${YELLOW}Szczegóły procesów game/db/auth:${RESET}"
ps aux | grep -E 'game|db|auth' | grep -v grep | awk -v GREEN="${GREEN}" -v RESET="${RESET}"
'{print GREEN $0 RESET}'
```

> **Wyjaśnienie:** Pokazujemy szczegóły działających procesów o nazwach zawierających "game", "db" lub "auth". Wynik jest kolorowany na zielono dla lepszej czytelności w terminalu.

Jak używać tego skryptu?

> Skrypt zapisujemy jako plik wykonywalny, np. `/etc/motd.sh`. Nadajemy mu uprawnienia wykonania (`chmod +x /etc/motd.sh`) i wywołujemy go z pliku startowego powłoki, np. `~/.bash_profile` lub globalnie w `/etc/profile`, dodając w nich linię:

```
/etc/motd.sh
```

> Dzięki temu podczas każdego logowania do powłoki pojawi się dynamiczny i kolorowy *Message Of The Day* z najważniejszymi informacjami o systemie i stanie procesów.

Revision #5
Created 2025-11-23 13:09:23 UTC by Tomasz Konieczny
Updated 2026-09-18 07:03:12 UTC by Tomasz Konieczny