Projekt Systemy Operacyjne

Temat: Demon synchronizujący dwa podkatalogi

Zespół:

- Damian Maksimowicz
- Karol Kacprzak
- Jakub Kowalewski

Opis włączenia i instalacji demona:

Kompilujemy projekt z użyciem makefile poleceniem:

make

Następnie włączamy demona z użyciem argumentów (-r -s [liczba sekund] [ścieżka źródło] [ścieżka cel] lub -h / -? oznaczający wyświetlenie instrukcji obsługi demona)

np. ./deamon -s 60 -r ~/source ~/destination

Opis funkcji i plików:

1. config.c

   config defaultConfig()

   Zwraca podstawową konfigurację demona (brak rekursywności, 5 minut przerwy między synchronizacją)

   config parseParams(int argc, char *argv[])

   Przekształca podane parametry w format danych zrozumiały dla demona, np. “-s 5” przekształca na przerwę 5 sekundową między synchronizacją katalogów

   void showAvailableParams()

   Ukazuje informacje dot. demona i ich parametrów.

2. converter.c

   int parseTime(char *time)

   Przekształca sekundy podane jako tablica znaków w typ zmiennych int i zwraca podaną wartość sekund.

3. dir.c

int checkIfDirectoriesContainEachOther(char *namea, char *nameb)

Sprawdza czy podane przez użytkownika katalogi nie zawierają się sobie, zwraca 0 lub 1 w zależności czy się zawierają czy nie.

int checkIfDirectoryExists(const char *name)

Sprawdza czy podane przez użytkownika katalogi istnieją w podanych lokalizacjach. Jeśli nie, zwraca 0 jeśli tak zwraca 1

fileList *getFilesFromDirectory(char *path, int recusive)

Zbiera wszystkie pliki do listy z folderu podanego przez użytkownika (path), jeśli zmienna recursive jest równa 1, do listy dodawane są również podkatalogi i ich zawartość. Zwraca listę plików.

int checkIfFileExists(char *name)

Sprawdza czy plik istnieje w podanej lokalizacji i nazwie podanej w zmiennej name.

int checkIfFileIsDirectory(char *name)

Sprawdza czy plik w podanej lokalizacji i nazwie jest katalogiem.

void createFile(char *path)

Tworzy plik w danej lokalizacji podanej w argumencie (path)

void copyFile(char *source, char *destination)

Kopiuje zawartość pliku z lokalizacji podanej w argumencie source, do lokalizacji podanej w argumencie destination

4. fileRepository.c

fileList *createList ()

Tworzy i zwraca pustą listę która będzie służyła do przechowywania listy plików.

fileType getFileType(char *path)

Zwraca typ pliku podany w enumie który zawiera się w pliku fileRepository.h w lokalizacji zawartej w argumencie path

fileList *addToList(fileList *list, char *name, char *path, fileType type)

Dodaje plik o nazwie wskazanej w argumencie name, o typie wskazanym w argumencie type, o ścieżce podanej w argumencie path, do listy wskazanej w argumencie list. Zwraca zaktualizowaną listę.

void *emptyList(fileList *first)

Czyści pamięć która była wykorzystywana przez listę.

fileList *reverseList(fileList *list)

Odwraca i zwraca listę podaną w argumencie (koniec jest początkiem).

fileList *mergeList(fileList *list, fileList *next)

Łaczy i zwraca listę połączoną z dwóch list podanych w argumentach list i next.

void deleteIfNotInSource(config conf)

Usuwa pliki zawierające się w katalogu docelowym a nie znajdujące się w katalogu źródłowym.

void injectTimestamps(char *source, char *dest)

Kopiuje timestampa pliku podanego w argumencie source do pliku podanego w arugumencie dest

int compareTimestamps(char *source, char *dest)

Zwraca 1 gdy timestampy plików podanych w argumentach nie różnią się. Zwróci 0 gdy timestampy plików się nie zgadzają.

5. job.c

void doJob(config conf)

Funkcja zbiorcza, kopiuje pliki z katalogu źródłowego podanego w konfiguracji przez użytkownika, do katalogu docelowego podanego w konfiguracji przez użytkownika.

6. main.c

void forkProcess()

Funkcja, która tworzy nowy proces przez powielenie procesu wywoływanego. Na nowym procesie będzie działał demon synchronizujący foldery.

void signalKillDaemon(int signum)

Sygnał, który ma za zadanie wyłączyć demona.

void signalForceDeamonJob(int signum)

Sygnał, który mówi demonowi, aby zaczął synchronizować foldery.

void setCustomSignals()

Funkcja pozwalająca ustawić własne sygnały.

7. Pliki z dopiskiem .h zawierają definicje podanych wyżej funkcji.

Dodatkowo:

- config.h - zawiera definicję struktury o nazwie config.
- fileRepository.h
    - zawiera definicje enuma używanego do określania typu pliku
    - zawiera definicje struktury o nazwie fileList

Testowanie demona

Do projektu został dodany skrypt o nazwie test.sh który sprawdza czy demon działa poprawnie tworząc w katalogu domowym użytkownika dwa katalogi i włączającym demona a następnie sprawdzającym różnice między dwoma katalogami utworzonymi przez skrypt.