Distributed Software Systems Exercise 6

Python Folder Monitor Daemon for Linux

libraries

- signal: handles system interrupts and signals
- watchdog: used to monitor file system events
- daemon: enables the program to be created as a daemon process
- lockfile: manages concurrent access to files
- syslog: makes the program able to interact with syslogd system module

```
import sys
import time
import os
import signal
from datetime import datetime
from watchdog.observers import Observer
from watchdog.events import FileSystemEventHandler
import daemon
import daemon.pidfile
import lockfile
import syslog
```

listeners

```python
class FileEventHandler(FileSystemEventHandler):
    Codeium: Refactor | Explain | Generate Docstring | ×
    def __init__(self):
        super().__init__()
        syslog.openlog("directory-monitor", syslog.LOG_PID)

    Codeium: Refactor | Explain | Generate Docstring | ×
    def on_created(self, event):
        if not event.is_directory:
            syslog.syslog(syslog.LOG_INFO, f"File created: {os.path.basename(event.src_path)}")

    Codeium: Refactor | Explain | Generate Docstring | ×
    def on_modified(self, event):
        if not event.is_directory:
            syslog.syslog(syslog.LOG_INFO, f"File modified: {os.path.basename(event.src_path)}")

    Codeium: Refactor | Explain | Generate Docstring | ×
    def on_deleted(self, event):
        if not event.is_directory:
            syslog.syslog(syslog.LOG_INFO, f"File deleted: {os.path.basename(event.src_path)}")

    Codeium: Refactor | Explain | Generate Docstring | ×
    def on_moved(self, event):
        if not event.is_directory:
            syslog.syslog(syslog.LOG_INFO, f"File renamed: {os.path.basename(event.src_path)} to {os.path.basename(event.dest_path)}")
```

listeners

- basically creates an event tied to every action happening in the folder (creation, modification, delete, move)
- prints to the syslogd module when one of these action are performed

monitor-directory

- uses the observer library to choose a directory to monitor
- sets the previously created Event Handler as the event handler for the observer

```python
def monitor_directory():
    if not os.path.exists(MONITOR_DIR):
        syslog.syslog(syslog.LOG_ERR, f"Monitor directory {MONITOR_DIR} does not exist!")
        sys.exit(1)

    event_handler = FileEventHandler()
    observer = Observer()
    observer.schedule(event_handler, MONITOR_DIR, recursive=False)

    try:
        syslog.syslog(syslog.LOG_INFO, f"Starting directory monitor for: {MONITOR_DIR}")
        observer.start()

        while True:
            time.sleep(1)

    except Exception as e:
        syslog.syslog(syslog.LOG_ERR, f"Error in monitor_directory: {str(e)}")
        observer.stop()
        raise
    finally:
        observer.join()
```

run_daemon

- runs the program as a system daemon setting also its attributes:
  - umask= 0o002: sets permission for accessing files and directories, rw-rw-r– for files and rwxrwxr-x for directories
  - pidfile= writes its process id to the file specified as value for PID_FILE
  - detach_process: detach from the terminal, run in the background
- run the daemon with directory_monitor_daemon.py start|stop|status

```
def run_daemon():
    context = daemon.DaemonContext(
        working_directory=MONITOR_DIR,
        umask=0o002,
        pidfile=daemon.pidfile.PIDLockFile(PID_FILE),
        detach_process=True
    )

    with context:
        monitor_directory()
```

Testing

- a test_monitor.py file is provided in the repository
- run it after starting the daemon to test it
- crontab can be used to run the test twice per day.
- the test_monitor.py is able to simulate operations for “different days”, if you run it while the daemon is in execution you should see an output like this

```
ratchef@nobara-pc:~/Documenti/GitHub/dis_sys$ sudo grep 'directory-monitor' /var/log/messages | sort
[sudo] password di ratchef:
Nov  1 16:15:37 nobara-pc directory-monitor[30196]: Starting directory monitor daemon
Nov  1 16:16:07 nobara-pc directory-monitor[30466]: Starting directory monitor daemon
Nov  1 16:16:07 nobara-pc directory-monitor[30468]: Starting directory monitor for: /home/ratchef/.directory_monitor/monitored
Nov  1 16:16:24 nobara-pc directory-monitor[30468]: File created: test_file_day0_20241101_161624.txt
Nov  1 16:16:24 nobara-pc directory-monitor[30468]: File modified: test_file_day0_20241101_161624.txt
Nov  1 16:16:25 nobara-pc directory-monitor[30468]: File modified: test_file_day0_20241101_161624.txt
Nov  1 16:16:25 nobara-pc directory-monitor[30468]: File renamed: test_file_day0_20241101_161624.txt to renamed_day0_20241101_161624.txt
Nov  1 16:16:26 nobara-pc directory-monitor[30468]: File deleted: renamed_day0_20241101_161624.txt
Nov  1 16:16:26 nobara-pc directory-monitor-test[30686]: Completed test sequence for day 0
Nov  1 16:16:28 nobara-pc directory-monitor[30468]: File created: test_file_day1_20241102_161628.txt
Nov  1 16:16:28 nobara-pc directory-monitor[30468]: File modified: test_file_day1_20241102_161628.txt
Nov  1 16:16:29 nobara-pc directory-monitor[30468]: File modified: test_file_day1_20241102_161628.txt
Nov  1 16:16:29 nobara-pc directory-monitor[30468]: File renamed: test_file_day1_20241102_161628.txt to renamed_day1_20241102_161628.txt
Nov  1 16:16:30 nobara-pc directory-monitor[30468]: File deleted: renamed_day1_20241102_161628.txt
Nov  1 16:16:30 nobara-pc directory-monitor-test[30686]: Completed test sequence for day 1
```