# Лабораторная работа №5

# Настройка пользовательских демонов с применением systemd

Демон (daemon) – термин, пришедший из UNIX систем и означающий программу, которая работает в фоне и не взаимодействует с пользователем напрямую. Такие программы обычно запускаются при старте системы, по таймеру или событию. Для того, чтобы настроить и контролировать запуск и работу демонов в современных дистрибутивах, используется системный менеджер systemd.

Для настройки демонов systemd использует юнит-файлы (юниты). Фактически они являются обычными текстовыми файлами. Каждый файл отвечает за отдельного демона. Кроме того, существуют специальные юниты, которые позволяют объединять демоны в группы, настраивать зависимости демонов друг от друга, задавать таймеры и т.д.

Юниты могут располагаться в трех местах файловой системы:

* /etc/systemd/system – здесь располагаются пользовательские юниты
* /run/systemd/system – здесь располагаются юниты генерирующиеся “на лету”, т.е. в рантайме
* /usr/lib/systemd/system – здесь располагаются юниты из установленных пакетов стороннего программного обеспечения (mysql, nginx и т.д.)

Юниты делятся на разные типы, которые отличаются друг от друга расширением файла. Примеры юнитов:

* .service – основной файл отвечающий за запуск конкретного демона
* .target – файл выполняющий две функции – синхронизация работы демонов и их группировка. На эти файлы можно ссылаться внутри .service файлов, задавая зависимость демона от других демонов или от уровней запуска ОС.
* .timer – файл для настройки запуска заданного демона по расписанию.
* .path – файл настройки запуска демона по событию в файловой системе (например, появление папки, изменение файла и т.д.)
* .device – файл настройки запуска демона по событию подключения устройства.

Стоит отметить, что systemd – это не одна утилита, а система с большим количеством компонентов. Чтобы работать с systemd из терминала существуют несколько утилит командной оболочки:

* systemctl – позволяет включать/выключать демонов, смотреть их статус
* journalctl – утилита просмотра логов юнитов
* systemd-notify – утилита для записи статуса сообщений
* systemd-analyze – утилита для анализа времени загрузки демонов
* cgls, cgtop, loginctl, nspawn и другие

*Важно*

Далее по тексту, где указываются команды, которые необходимо вводить в терминал, знак **$** вводить не нужно. Это касается только тех мест, где $ находится в самом начале строки.

*Полезные ресурсы*

<https://habr.com/ru/post/503816/> – немного истории и холивара по теме systemd

<https://highload.today/systemd-linux/> – выжимка о возможностях systemd

<https://linuxrussia.com/systemd-create-own-unit.html> – коротенький туториал

<https://sysadminium.ru/adm-serv-linux-systemd-target/> – про типы таргетов в system

<https://wiki.archlinux.org/title/systemd/Timers> – про настройку таймеров в systemd

*Требования к отчету*

Сопроводите свою работу несколькими скриншотами. К отчету приложите файл record\_result\_lab\_5. Это текстовый файл с вашими командами.

## Ход работы

Подготовка

1. Запустите виртуальную машину и дождитесь загрузки ОС.
2. Откройте терминал. Это можно сделать с помощью основного меню или сочетанием клавиш **Ctrl+Alt+T**. Для запуска через основное меню нажмите на Menu -> System Tools -> Mate Terminal.
3. **Запустите запись терминала.**  **Это важно!**  Именно файл с записью истории ваших команд будет основной частью отчета. Для этого введите и запустите команду

$ script -a record\_result\_lab\_5

1. Убедитесь, что команда выполнилась без ошибок. Введите нижеуказанную команду и найдите имя файла record\_result\_lab\_5 в ее выводе:

$ ls -l

Генерация варианта

1. Поместите на рабочий стол файл lab5.py, прилагаемый к этому руководству. А также прилагаемый архив faces.tar.gz .
2. Вернитесь в терминал, в котором уже идет запись (см. Подготовка).
3. Перейдите на рабочий стол. Для этого:

$ cd ./Desktop

в случае с русским языком нужно использовать экранирование пробела

$ cd ./Рабочий\ стол/

1. Распакуйте архив на рабочий стол. Для этого:

$ tar -xvzf faces.tar.gz

1. Выведите список всех файлов и каталогов. Убедитесь, что в выводе есть файл lab5.py и директория images

$ ls -l

1. Запустите генерацию вашего варианта. Нашей задачей будет написать демон, представляющий собой программу для отдыха глаз. Демон будет показывать уведомление о начале отдыха раз в заданный промежуток времени (repetition\_period) и уведомление об окончании отдыха через заданный промежуток времени (relax\_time). Пример генерации варианта на Рисунке 1. Результат генерации добавить в отчет. Для генерации:

$ python lab5.py

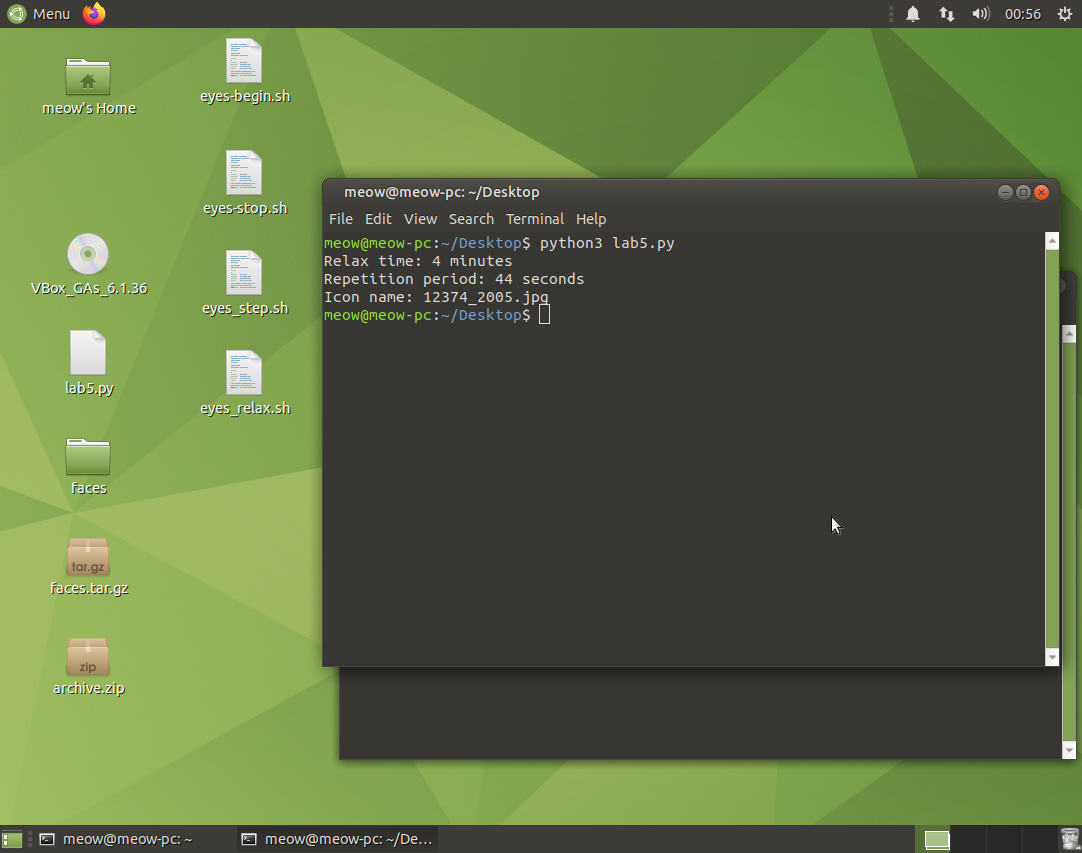


Рисунок 1 – Результат генерации варианта

Создание bash скриптов

1. Напишем bash-скрипт, запуск которого потом настроим в systemd. Для этого перейдем на рабочий стол, создадим файл скрипта и сделаем его исполняемым:

$ cd ~/Desktop

$ touch eyes-start.sh

$ chmod +x eyes-start.sh

1. Откроем файл скрипта в редакторе nano:

$ nano eyes-start.sh

1. Напишем в файл нашу программу

#!/bin/bash

period\_s=$1

period\_ms=$((period\_s\*1000))

icon=/absolute\_/path\_/to\_/icon

notify-send -i $icon -t $period\_ms "Time to relax, senpai!" "$period\_s seconds"

Рассмотрим принцип работы этой программы.

#!/bin/bash  
 Это так называемый шебанг. Шебанг позволяет указать загрузчику программы какой интерпретатор (командную оболочку) использовать для исполнения данной программы.

period\_s=$1

Создаем переменную с именем period\_s и присваиваем ей значение первого параметра, передаваемого нашему скрипту из командной оболочки. Например, при вызове ./eyes\_start.sh 10 переменная period\_s внутри скрипта будет равна 10.

period\_ms=$((period\_s\*1000))

Преобразуем значение секунд в миллисекунды. Синтаксическая конструкция вида $(( арифметическое\_выражение )) позволяет подставлять в текст скрипта результат выполнения арифметического выражения в скобках.

icon=/absolute/path/to/icon

Переменной icon необходимо присвоить абсолютный путь до изображения соответственно вашему варианту. **Внимание!** Не пишите /absolute/path/to/icon, напишите настоящий путь к файлу на вашей машине**.**

notify-send -i $icon -t $period\_ms "Time to relax, senpai!" "$period\_s seconds"

Команда notify-send – это утилита, которая позволяет генерировать всплывающие уведомление на рабочем столе. Ключ -i позволяет задать картинку, располагающуюся в углу уведомления. Ключ -t позволяет задать время отображения уведомления. Далее в кавычках идет основной текст уведомления. Затем в кавычках идет дополнительный текст уведомления.

1. Убедимся в работоспособности программы, запустив ее вручную.

$ ./eyes-start.sh 10

1. Если в консоль не вывелись ошибки, а на рабочем столе вы увидели уведомление (см. Рисунок 2), то можно продолжать. Если этого не произошло, значит что-то пошло не так.

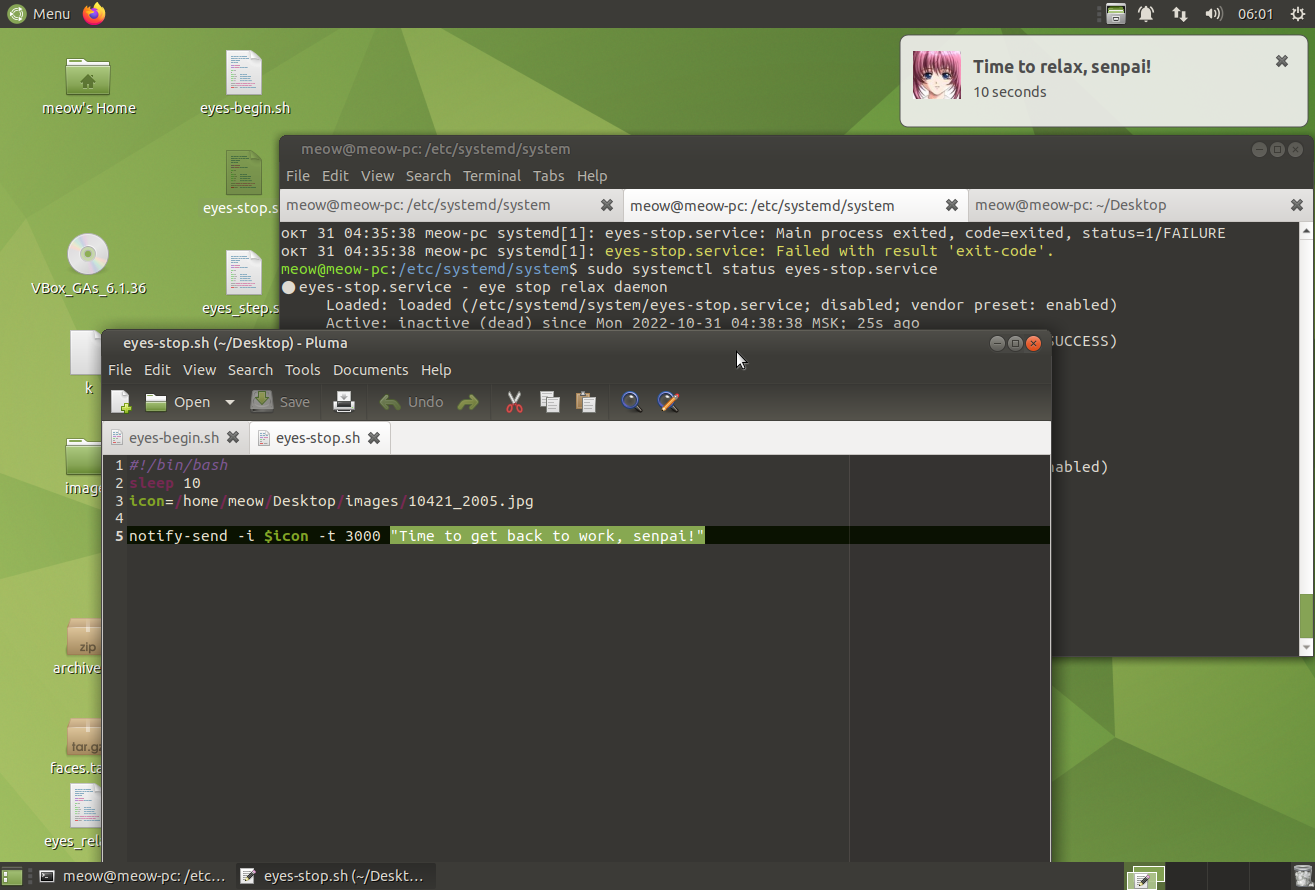


Рисунок 2 – Всплывающее уведомление

1. Теперь самостоятельно напишите второй скрипт. Текст получившейся программы добавьте в отчет. Скрипт должен соответствовать следующим требованиям:
   1. Имя скрипта eyes-stop.sh
   2. Работа скрипта должна начинаться с паузы длительностью relax\_time (значение взять из варианта). Вам пригодится команда sleep.
   3. Затем на рабочий стол должно вывестись уведомление с текстом "Time to get back to work, senpai!" и иконкой в соответствии с вариантом.
   4. Длительность показа уведомления 5 секунд.
   5. Не забудьте сделать файл исполняемым.

Создание юнит-файлов

1. Настроим запуск наших скриптов с использованием systemd. Это позволит нам запускать наши скрипты как демонов. Для начала перейдем в папку с юнит-файлами:

$ cd /etc/systemd/system

1. Создадим юнит-файл и сделаем его исполняемым:

$ sudo touch eyes-relax.service

$ sudo chmod +x eyes-relax.service

1. Откроем его в редакторе:

$ sudo nano eyes-relax.service

1. Узнаем несколько параметров системы, которые пригодятся дальше.

$ xauth

Вывод похож на такое:

Using authority file /home/meow/.Xauthorit **# Запомните этот путь ( у вас он будет отличаться)**

1. Узнаем адрес системной шины межпроцессорного взаимодействия.

$ echo $DBUS\_SESSION\_BUS\_ADDRESS

Вывод похож на такое

unix:path=/run/user/1000/bus **# Запомните этот адрес ( у вас он будет отличаться)**

1. Напишем в файл конфигурацию нашего демона. **# Вместо <ваше имя> напишите имя своего пользователя.**

[Unit]

Description=eye relax daemon

After=eyes-stop.service

Wants=eyes-stop.service

[Service]

Type=simple

Environment="DISPLAY=:0"

Environment="XAUTHORITY=/home/meow/.Xauthority" **# Сюда записать путь из пункта 4**

Environment="DBUS\_SESSION\_BUS\_ADDRESS=unix:path=/run/user/1000/bus" **# Сюда записать адрес из пункта 5**

User=**<ваше имя>**

Group=**<ваше имя>**

WorkingDirectory=/home/<ваше\_имя>/Desktop/

ExecStart=/home/**<ваше имя>**/Desktop/eyes-start.sh 10 **# Вместо 10 поставьте сюда значение relax\_time из вашего варианта**

Рассмотрим структуру юнита подробнее. Файл содержит несколько секций:

* Unit – обязательный раздел, который содержит текстовое описание демона и его зависимость от других демонов (юнитов).
* Service – обязательный раздел, который конфигурирует, что запускать (какую программу) и как запускать.
* Install – обязательный раздел, который может быть вынесен в другие специальные юнит файлы (например, в файл таймера). В файле конфигурируется режим ОС (уровень запуска), в котором запускается демон. Например, это может быть режим с графикой, сетевой режим без графики, режим перезагрузки и т. д.

Рассмотрим атрибуты раздела Unit:

* Description – текстовое описание демона
* Wants – список демонов активация которых желательна для работы данного демона
* After – список демонов, после которых должен запуститься данный демон

Рассмотрим атрибуты Service:

* Type — описывает, как запускается демон. Есть несколько вариантов: simple (по умолчанию) — systemd ожидает, что служба запустится незамедлительно. Процесс не должен разветвляться. forking — после запуска демон ответвляется (делает форк), родительский процесс завершается. Такой подход используется для запуска классических демонов. oneshot — одноразовое выполнение. Используется для скриптов, которые запускаются и завершаются после выполнения.
* Environment —cписок переменных окружения.
* WorkingDirectory — рабочая директория. Становится текущей перед запуском стартовой программы.
* ExecStart — команда для старта демона (программа демона).

1. Чтобы обеспечить периодичность запуска нашего демона, создадим специальный юнит eyes-relax.timer . Его название должно совпадать с service файлом, созданным ранее, благодаря этому они будут связаны.

$ sudo touch eyes-relax.timer

$ sudo chmod +x eyes-relax.timer

$ nano eyes-relax.timer

1. Напишем в файл конфигурацию таймера. В поле OnUnitActiveSec укажите значение времени из варианта (repetition\_period). В отчете опишите, за что отвечают секции и поля в файле таймера.

[Unit]

Description=scheduler for eyes service

[Timer]

OnBootSec=60s

OnUnitActiveSec=50s

[Install]

WantedBy=timers.target graphical.target

1. Создадим юнит файл для написанного вами скрипта (раздел “Написание bash скриптов” пункт 6).

$ sudo touch eyes-stop.service

$ sudo chmod +x eyes-stop.service

$ nano eyes-stop.service

1. В созданный файл написать.

[Unit]

Description=eye stop relax daemon

Requires=eyes-start.service

[Service]

Type=simple

Environment="DISPLAY=:0" Environment="XAUTHORITY=/home/meow/.Xauthority" **# Сюда записать путь из пункта 4**

Environment="DBUS\_SESSION\_BUS\_ADDRESS=unix:path=/run/user/1000/bus" **# Сюда записать адрес из пункта 5**

User=**<ваше имя>**

Group=**<ваше имя>**

WorkingDirectory=/home/<ваше\_имя>/Desktop/

ExecStart=/home/<ваше\_имя>/Desktop/eyes-stop.sh

[Install]

WantedBy=graphical.target

Запуск демона

1. Теперь, когда готовы все нужные файлы, необходимо научиться управлять демоном. Вам пригодятся следующие команды:
2. Итак, перезагрузим всех демонов командой

$ sudo systemctl daemon-reload

1. В лабораторной мы создали три демона – таймер (eyes-relax.timer), демона который уведомляет о начале отдыха (eyes-relax.service) и демона, который ждет заданное время и уведомляет об окончании отдыха (eyes-stop.service).
2. Теперь запустим стартового демона, который в свою очередь активирует выполнение остальных. Из созданной нами логической структуры можно понять, что таким демоном является таймер.

$ sudo systemctl start eyes-relax.timer

1. Проверим статусы всех демонов:

$ sudo systemctl status eyes-relax.timer

$ sudo systemctl status eyes-relax.service

$ sudo systemctl status eyes-stop.service

1. Чтобы установить демон в автозагрузку:

$ sudo systemctl enable eyes-relax.timer

1. Убедитесь, что демон работает, перезагрузив виртуальную машину.
2. Чтобы остановить демон, воспользуйтесь командой:

$ sudo systemctl stop eyes-relax.timer

1. Если демон вам больше не нужен, можно отключить автозагрузку командой:

$ sudo systemctl disable eyes-relax.timer