

## **Часть 1. Написание bash-скрипта для автоматизации установки ПО**

Скрипт написан на языке bash для операционной системы Linux Mint 21. В начале определяются функции для вывода сообщений с цветовым форматированием (информационные, предупреждения, ошибки). Также предусмотрена проверка запуска от имени суперпользователя.

Далее созданы списки программ, устанавливаемых через менеджер пакетов apt и тех, для которых нужно скачать и установить deb-пакет.

Далее программа выполняет следующие шаги:

- обновление списка пакетов (apt update и apt-get update) и уже установленных программ (apt upgrade -y);

Важно: ключ -y отвечает за автоматическую установку по умолчанию – отвечает «yes» на все вопросы установщика.

- установка набора программ через менеджер пакетов apt: Git, Maxima (и wxMaxima), Gimp, Python, Rust, Far Manager, Flameshot, Qualculate!, 7zip и Firefox;
- скачивание и установка .deb-пакетов: Visual Studio Code и Яндекс Браузер;
- установка приложений по отдельным инструкциям: Docker, PyCharm, GitHub Desktop, Julia, Zettlr, MiKTeX, TeXstudio, Anaconda, Google Chrome;
- установка расширений для Visual Studio Code: Julia, Python, Rust;
- очистка временных файлов и кэша системы.

Не были установлены следующие программы, так как они поддерживаются только Windows, либо нужны для функционирования Windows как Linux: MSYS2 UCRT64 Shell, Chocolatey, SumatraPDF, WSL, Яндекс Телемост и Sber Jazz. Яндекс Телемост и Sber Jazz можно использовать в браузере.

Финальным этапом скрипт выводит краткий отчёт об установленных пакетах и рекомендует перезагрузить систему.

Как запускать скрипт:

`chmod +x trofimtsova_software_setup.sh` # делает файл исполняемым

`./ trofimtsova_software_setup.sh` # запуск файла

Листинг кода bash-скрипта (скрипт также прикреплен в репозитории):

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

```
# =====
```

```
# Скрипт автоматической установки ПО для Linux Mint
```

```
# =====
```

```
# Цвета для вывода в терминал
```

```
RED='\033[0;31m'
```

```
GREEN='\033[0;32m'
```

```
YELLOW='\033[1;33m'
```

```
NC='\033[0m' # No Color
```

```
# Функция для печати статусов
```

```
print_status() {
```

```
    echo -e "${GREEN}[INFO]${NC} $1"
```

```
}
```

```
print_warning() {
```

```
    echo -e "${YELLOW}[WARN]${NC} $1"
```

```
}
```

```
print_error() {
```

```
    echo -e "${RED}[ERROR]${NC} $1"
```

```
}
```

```
# Проверка, запущен ли скрипт с правами root
if [ "$EUID" -ne 0 ]; then
    print_error "Этот скрипт требует прав суперпользователя.
Используйте sudo."
    echo "Пример: sudo ./software_setup.sh"
    exit 1
fi
```

```
# Переменные со списками программ для установки
# 1. Программы из официальных репозиториев (через apt)
```

```
APT_PACKAGES=(
```

```
    "git-all"
```

```
    "maxima"
```

```
    "wxmaxima"
```

```
    "gimp"
```

```
    "python3-pip"
```

```
    "rustup"
```

```
    "far2l"
```

```
    "flameshot"
```

```
    "qalculate-gtk"
```

```
    "7zip"
```

```
    "firefox"
```

```
)
```

```
# 2. Программы, которые необходимо скачать как .deb пакеты
```

```
# Формат: "Описательное_имя;URL_на_файл"
```

```
DEB_PACKAGES=(
```

```

"vs-code;
https://code.visualstudio.com/sha/download?build=stable&os=linux-deb-x64"

"yandex-browser;
https://browser.yandex.ru/download?os=linux&package=deb&x64=1&darktheme=
0&banerid=6400000000&portal_testids=1114258%2F-1%2C1114347%2F-
1%2C1124063%2F-1%2C1127618%2F-
1%2C1349550%2F90&signature=U%2FqziHDqzAJF8b1wvllyzQOwNAHE2N1
WyI8T8uQt9i%2FeiKHwPdbI8HEJzTvUHDKHffmeTAmRK9OiqejgWwZqrA%3
D%3D"

)

# =====
# Начало выполнения скрипта
# =====

print_status "Начинается процесс автоматической установки ПО..."
echo ""

# Шаг 1: Обновление списка пакетов
print_status "Шаг 1: Обновление списка пакетов..."
apt update
apt-get update

# Шаг 2: Обновление уже установленных пакетов (рекомендуется)
print_status "Шаг 2: Обновление установленных пакетов..."
apt upgrade -y

# Шаг 3: Установка пакетов из репозитория АРТ
print_status "Шаг 3: Установка пакетов из репозитория АРТ..."
for package in "${APT_PACKAGES[@]}"; do

```

```

print_status "Устанавливается: $package"
apt install -y "$package"
if [ $? -eq 0 ]; then
    print_status "$package успешно установлен."
else
    print_error "Не удалось установить $package."
fi
done

# Шаг 4: Установка .deb пакетов
print_status "Шаг 4: Установка .deb пакетов..."
TEMP_DIR="/tmp/software_setup_deb"
mkdir -p "$TEMP_DIR"

for package_info in "${DEB_PACKAGES[@]}"; do
    # Разделяем строку на имя и URL
    IFS=';' read -r -a package_data <<< "$package_info"
    PACKAGE_NAME="${package_data[0]}"
    PACKAGE_URL="${package_data[1]}"
    DEB_FILENAME=$(basename "$PACKAGE_URL")

    print_status "Скачивается и устанавливается: $PACKAGE_NAME"

    # Скачивание в временную директорию
    wget -O "$TEMP_DIR/$DEB_FILENAME" "$PACKAGE_URL"

    # Установка
    dpkg -i "$TEMP_DIR/$DEB_FILENAME"

```

```
# Исправление возможных зависимостей
apt install -f -y

if [ $? -eq 0 ]; then
    print_status "$PACKAGE_NAME успешно установлен."
else
    print_error "Не удалось установить $PACKAGE_NAME."
fi
done
```

# Шаг 5: Установка приложений по инструкциям с официальных сайтов или с распаковкой архива

```
# Установка Docker

print_status "Скачивается и устанавливается: Docker"

# Add Docker's official GPG key:

sudo apt-get update

sudo apt-get install ca-certificates curl

sudo install -my 0755 -d /etc/apt/keyrings

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc

sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:

echo \

"deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \

$(. /etc/os-release && echo "${UBUNTU_CODENAME:-
$VERSION_CODENAME}") stable" | \
```

```

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-buildx-
plugin docker-compose-plugin

# Установка PyCharm (может не установиться с российского IP-адреса)
print_status "Скачивается и устанавливается: PyCharm"

wget -O pycharm.tar.gz
"http://www.jetbrains.com/pycharm/download/download-
thanks.html?platform=linux"

sudo tar -xzf pycharm.tar.gz -C /opt/

# Установка GitHub Desktop
print_status "Скачивается и устанавливается: GitHub Desktop"

## Direct copy-paste from official instructions
## Github Desktop for Ubuntu
## Get the @shiftkey package feed

wget -qO - https://apt.packages.shiftkey.dev/gpg.key | gpg --dearmor | sudo
tee /usr/share/keyrings/shiftkey-packages.gpg > /dev/null

sudo sh -c 'echo "deb [arch=amd64 signed-by=/usr/share/keyrings/shiftkey-
packages.gpg] https://apt.packages.shiftkey.dev/ubuntu/ any main" >
/etc/apt/sources.list.d/shiftkey-packages.list'

## Install Github Desktop for Ubuntu

sudo apt update && sudo apt install -y github-desktop

# Установка Julia
print_status "Скачивается и устанавливается: Julia"

wget https://julialang-s3.julialang.org/bin/linux/x64/1.11/julia-1.11.7-linux-
x86_64.tar.gz

tar zxvf julia-1.11.7-linux-x86_64.tar.gz

```

# Установка Zettlr

```
print_status "Скачивается и устанавливается: Zettlr"
```

```
curl -s --compressed "https://apt.zettlr.com/KEY.gpg" | gpg --dearmor | sudo
tee /etc/apt/trusted.gpg.d/zettlr_apt.gpg > /dev/null
```

```
sudo curl -s --compressed -o /etc/apt/sources.list.d/zettlr.list
"https://apt.zettlr.com/zettlr.list"
```

```
sudo apt update
```

```
sudo apt install -y zettlr
```

# Установка MiKTeX

```
print_status "Скачивается и устанавливается: MiKTeX"
```

```
curl -fsSL https://miktex.org/download/key | sudo gpg --dearmor -o
/usr/share/keyrings/miktex.gpg
```

```
echo "deb [signed-by=/usr/share/keyrings/miktex.gpg]
https://miktex.org/download/ubuntu jammy universe" | sudo tee
/etc/apt/sources.list.d/miktex.list
```

```
sudo apt-get update
```

```
sudo apt-get install -y miktex
```

# Установка TeXstudio

```
print_status "Скачивается и устанавливается: TeXstudio"
```

```
sudo add-apt-repository ppa:sunderme/texstudio
```

```
sudo apt-get install -y texstudio
```

# Установка Anaconda

```
print_status "Скачивается и устанавливается: Anaconda"
```

```
wget https://repo.anaconda.com/archive/Anaconda3-2019.03-Linux-
x86_64.sh
```



```
bash Anaconda3-2019.03-Linux-x86_64.sh -b
```

```
# Установка Google Chrome
```

```
print_status "Скачивается и устанавливается: Google Chrome"
```

```
wget https://dl.google.com/linux/direct/google-chrome-  
stable_current_amd64.deb
```

```
sudo dpkg -yi google-chrome-stable_current_amd64.deb
```

```
# Установка расширений для VS Code
```

```
code --install-extension julialang.language-julia
```

```
code --install-extension ms-python.python
```

```
code --install-extension rust-lang.rust-analyzer
```

```
# Удаляем временную директорию
```

```
rm -rf "$TEMP_DIR"
```

```
# Шаг 6: Очистка кэша
```

```
print_status "Шаг 6: Очистка кэша пакетов..."
```

```
apt autoremove -y
```

```
apt clean
```

```
print_status
```

```
"=====
```

```
print_status "Установка ПО завершена!"
```

```
print_status
```

```
"=====
```

```
# Краткий отчет
```

```

echo ""

print_status "Краткий отчет:"

print_status "Установленные пакеты из APT: ${#APT_PACKAGES[@]}"

print_status "Установленные .deb пакеты: ${#DEB_PACKAGES[@]}"

print_status "Установленные по отдельной инструкции: Docker,
PyCharm, GitHub Desktop, Julia, Zettlr, MiKTeX, TeXstudio, Anaconda, Google
Chrome"

print_status "Установленные расширения для VS Code: Julia, Python,
Rust"

echo ""

print_warning "Рекомендуется перезагрузить систему."

```

## **Часть 2. Технологии и инструменты для создания бэкапа ОС Linux**

Создание резервной копии (бэкапа) всей системы в Linux чаще всего реализуется путём создания образа диска или снимка файловой системы. Такой подход позволяет полностью восстановить систему после сбоя, повреждения или замены оборудования.

### **1. Подходы к созданию бэкапов**

- Побайтовое копирование диска (disk imaging). Позволяет создать точный клон всего диска или раздела, включая загрузчик, файловую систему и все данные.
- Файловое резервное копирование. Копируются только файлы и каталоги, но не структура разделов. Подходит для сохранения данных, но не для полного восстановления ОС.
- Снимки файловой системы (snapshots). Поддерживаются современными файловыми системами (например, Btrfs, ZFS, LVM). Позволяют создавать моментальные «срезы» состояния системы.

### **2. Инструменты для создания образов и бэкапов**

**Универсальные инструменты побайтового копирования**

### 1. dd

- Базовая утилита Linux для копирования блоков данных.
- Можно создать полный образ диска:

```
sudo dd if=/dev/sda of=/mnt/backup/system.img bs=4M
status=progress
```

- Минусы: создаёт «сырые» образы, большой размер, нет сжатия.

### 2. partclone

- Более эффективная альтернатива dd. Копирует только занятые блоки файловой системы.
- Используется в Clonezilla.

### 3. Clonezilla

- Популярный инструмент для создания и восстановления образов дисков и разделов.
- Работает с множеством файловых систем (ext, NTFS, FAT, Btrfs, XFS и др.).
- Поддерживает сжатие, шифрование, сохранение образов по сети.
- Часто применяется для массового развёртывания систем.

## Снимки файловой системы

### 1. Btrfs snapshots

- Файловая система Btrfs поддерживает встроенные снимки.
- Создание снимка:

```
sudo btrfs subvolume snapshot / /mnt/backup/root_snapshot
```

- Можно быстро откатиться на состояние снимка.

### 2. ZFS snapshots

- Более продвинутая файловая система с поддержкой снимков и репликации (это процесс создания и поддержания точных копий данных на одном или нескольких других устройствах или серверах).
- Создание снимка:

```
sudo zfs snapshot poolname@snapshot1
```

- Можно реплицировать снимки на другой сервер.

### 3. LVM snapshots

- Если система использует LVM, можно создать снимок логического тома:

```
sudo lvcreate --size 1G --snapshot --name root_snap /dev/vg0/root
```

## ◆ Средства системного резервного копирования

### 1. Timeshift

- Очень популярный инструмент в Linux Mint и Ubuntu.
- Позволяет делать «системные точки восстановления» (как в Windows).
- Работает через rsync или Btrfs snapshots.
- Можно легко откатиться, если обновление системы сломало ОС.

### 2. Déjà Dup (Frontend для duplicity)

- Графическая программа для бэкапа.
- Поддерживает инкрементные копии, шифрование, загрузку на Google Drive и другие облака.
- Обычно ориентирована на сохранение пользовательских данных, но может использоваться для бэкапа системы.

### 3. Bacula / Amanda

- Серверные решения для централизованного управления бэкапами.
- Используются в больших организациях, поддерживают хранение на лентах, NAS, в облаке.

## 3. Рекомендации по выбору метода

Для домашнего пользователя:

- Timeshift (удобен, интегрирован в Linux Mint, подходит для восстановления после ошибок).
- Déjà Dup (для резервирования пользовательских данных в облаке).

Для полного клонирования системы:

- Clonezilla (надёжно и универсально).

- partclone + внешний диск (эффективно).

Для серверов и продвинутых пользователей:

- ZFS или Btrfs snapshots + репликация.
- Bacula / BorgBackup для инкрементных копий.

### **Итог**

В Linux можно создать бэкап системы либо на уровне диска (Clonezilla, dd, partclone), либо на уровне файловой системы (Timeshift, Btrfs/ZFS snapshots), либо с помощью специализированных инструментов (Déjà Dup, Bacula).

### Список источников

1. Как автоматизировать установку софта в Linux: подробный гайд с примерами | Сисадмин. — Текст : электронный // Дзен : [сайт]. — URL: <https://dzen.ru/a/aAKWQuKVhyZf1aXH> (дата обращения: 17.09.2025).
2. Скрипт установщик ПО - Ubuntu. — Текст : электронный // Всяко разное : [сайт]. — URL: <https://flammlin.com/blog/2020/07/02/bash-script-auto-install-soft-ubuntu/> (дата обращения: 17.09.2025).
3. Установка и обновление Браузера | Браузер. — Текст : электронный // Яндекс : [сайт]. — URL: [https://yandex.ru/support/browser/ru/about/install.html?hl=ru%3Fflang%3Den%3Fflang%3Den%3Fflang%3Dru%3Fflang%3Dru%3Fflang%3Dru%3Fflang%3Dru&tabs=defaultTabsGroup-eeomo9sq\\_linux%2CdefaultTabsGroup-6srvlnlq\\_linux](https://yandex.ru/support/browser/ru/about/install.html?hl=ru%3Fflang%3Den%3Fflang%3Den%3Fflang%3Dru%3Fflang%3Dru%3Fflang%3Dru%3Fflang%3Dru&tabs=defaultTabsGroup-eeomo9sq_linux%2CdefaultTabsGroup-6srvlnlq_linux) (дата обращения: 17.09.2025).
4. Creating a bash script to install packages. — Текст : электронный // Stack Exchange : [сайт]. — URL: <https://unix.stackexchange.com/questions/717483/creating-a-bash-script-to-install-packages> (дата обращения: 17.09.2025).
5. Download PyCharm: The Python IDE for data science and web development by JetBrains. — Текст : электронный // JetBrains s.r.o. : [сайт]. — URL: <https://www.jetbrains.com/pycharm/download/?section=linux> (дата обращения: 17.09.2025).
6. Download Visual Studio Code - Mac, Linux, Windows. — Текст : электронный // Visual Studio : [сайт]. — URL: <https://code.visualstudio.com/Download> (дата обращения: 17.09.2025).
7. Getting MiKTeX. — Текст : электронный // Christian Schenk : [сайт]. — URL: <https://miktex.org/download> (дата обращения: 17.09.2025).
8. How do I write a shell script to install a list of applications?. — Текст : электронный // Ask Ubuntu : [сайт]. — URL: <https://askubuntu.com/questions/519/how-do-i-write-a-shell-script-to-install-a-list-of-applications> (дата обращения: 17.09.2025).
9. How to Install Anaconda on Ubuntu 20.04. — Текст : электронный // GeeksforGeeks : [сайт]. — URL: <https://www.geeksforgeeks.org/linux->

[unix/how-to-install-anaconda-on-ubuntu-20-04/](#) (дата обращения: 17.09.2025).

10. How to Install Chrome on Ubuntu. — Текст : электронный // GeeksforGeeks : [сайт]. — URL: <https://www.geeksforgeeks.org/linux-unix/how-to-install-chrome-in-ubuntu/> (дата обращения: 17.09.2025).
11. How to Install TeXstudio – A LaTeX Editor in Ubuntu Linux. — Текст : электронный // UbuntuPit : [сайт]. — URL: <https://ubuntupit.com/how-to-install-texstudio-a-latex-editor-in-ubuntu-linux/> (дата обращения: 17.09.2025).
12. How to Write a Script to Install Multiple Applications. — Текст : электронный // Baeldung : [сайт]. — URL: <https://www.baeldung.com/linux/script-install-many-applications#bd-bash-scripting-and-installations> (дата обращения: 17.09.2025).
13. Install Rust - Rust Programming Language. — Текст : электронный // Rust : [сайт]. — URL: <https://rust-lang.org/tools/install/> (дата обращения: 17.09.2025).
14. rust-analyzer - Visual Studio Marketplace. — Текст : электронный // Microsoft : [сайт]. — URL: <https://marketplace.visualstudio.com/items?itemName=rust-lang.rust-analyzer> (дата обращения: 17.09.2025).
15. Setup - Zettlr User Manual. — Текст : электронный // Zettlr : [сайт]. — URL: <https://docs.zettlr.com/en/getting-started/setup/> (дата обращения: 17.09.2025).
16. Ubuntu | Docker Docs. — Текст : электронный // Docker Inc. : [сайт]. — URL: <https://docs.docker.com/engine/install/ubuntu/> (дата обращения: 17.09.2025).