

# **Текстовый редактор etacs**

**Лабораторная работа №11**

Козомазов Владимир Романович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
3.0.1	Теоретическое введение: текстовый редактор Emacs . . .	7
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>10</b>
<b>5</b>	<b>Выводы</b>	<b>13</b>

# Список иллюстраций

4.1	Установка менеджера паролей . . . . .	10
4.2	Завершение установки менеджера паролей . . . . .	10
4.3	Просмотр списка ключей . . . . .	10
4.4	Инициализация хранилища . . . . .	10
4.5	Создание структуры . . . . .	11
4.6	Ручное выкладывание изменений . . . . .	11
4.7	Проверка статуса синхронизаций . . . . .	11
4.8	Добавление нового пароля . . . . .	11
4.9	Установка дополнительного программного обеспечения . . . . .	11
4.10	Установка дополнительных шрифтов . . . . .	12

## **Список таблиц**

# 1 Цель работы

Комплексное изучение текстового редактора Emacs как многофункциональной программной среды, включая анализ его архитектуры, функциональных возможностей, особенностей использования и перспектив развития в современных условиях.

## 2 Задание

1. Познакомиться с интерфейсом и режимами работы редактора vi (командный режим, режим вставки, режим последней строки).
2. Освоить основные команды для навигации, редактирования, копирования, удаления и поиска текста.
3. Научиться сохранять изменения и выходить из редактора.
4. Рассмотреть дополнительные возможности (работа с несколькими файлами, настройки, использование плагинов в vim).
5. Закрепить навыки на практике, создавая и редактируя текстовые файлы.

## 3 Теоретическое введение

### 3.0.1 Теоретическое введение: текстовый редактор Emacs

#### 3.0.1.1 1. Исторический контекст и философские основы

Emacs (от англ. *Editor MACroS*) был создан в 1976 году Ричардом Столлманом как часть проекта GNU, став воплощением принципов свободного ПО. Его развитие отражает ключевые тенденции в эволюции: - От макроредактора TЕСO к самостоятельной системе - Формирование концепции “редактора как операционной среды” - Влияние идей Lisp-машины Symbolics на архитектуру

#### 3.0.1.2 2. Архитектурные особенности

**Ядро системы** построено на: - Интерпретаторе Emacs Lisp (реализация диалекта Lisp) - Модели “буфер-окно-фрейм” для организации workspace - Цикле обработки событий (event loop)

**Ключевые компоненты:** 1. **Буферы** - виртуальные пространства для работы с данными 2. **Минибуфер** - интерактивная командная строка 3. **Режимы** (major/minor) - контекстно-зависимые поведения

#### 3.0.1.3 3. Парадигма расширяемости

Emacs реализует уникальный подход: - **Самодокументируемость** (встроенная help-система) - **Рефлексивность** (модификация во время выполнения) - **Принцип “всё есть буфер”** (унификация интерфейсов)

#### 3.0.1.4 4. Технические характеристики

- **Язык реализации:** C (ядро) + Emacs Lisp (98% кодовой базы)
- **Модель исполнения:** Гибридная (интерпретация + native-компиляция)
- **Протоколы взаимодействия:**
  - D-Bus (системная интеграция)
  - JSON-RPC (для LSP)
  - TCP/IP (удалённое управление)

#### 3.0.1.5 5. Современное состояние

В версии 29+ появились: - Встроенная поддержка Tree-sitter - Нативная компиляция Emacs - Графические улучшения (Harfbuzz, XWidgets) - Интеграция с системными сервисами (например, Portal для Flatpak)

#### 3.0.1.6 6. Теоретическая значимость

Emacs представляет интерес для исследований в областях: - **Интерфейсов человек-машина** (модель модальности) - **Языково-ориентированного программирования** (DSL через Emacs Lisp) - **Эргономики разработки** (анализ workflow экспертов)

#### 3.0.1.7 7. Сравнительный контекст

В отличие от: - **Vim** - акцент на модальности и композиции команд - **VSCode** - модульность через Web-технологии - **Sublime Text** - закрытая проприетарная модель

Emacs предлагает: □ Полный контроль над средой □ Глубокую семантическую интеграцию инструментов □ Парадигму “редактор как виртуальная машина Lisp”



### **3.0.1.8 8. Перспективные направления**

Теоретический анализ выявляет потенциал: - Применения ML для интеллектуального автодополнения - Развития распределённых редакторских сред - Интеграции с новыми hardware-интерфейсами

*Данный теоретический базис позволяет перейти к детальному анализу функциональных возможностей и практических аспектов использования Etacs в современных условиях.*

## 4 Выполнение лабораторной работы

Установил менеджер паролей pass с помощью команды `sudo dnf install pass` `pass-otp` (рис. 4.1).

```
Kozomazov@fedora: ~$ sudo dnf install pass pass-otp
[sudo] пароль для vkozomazov:
Пополнение едб паз.
[sudo] пароль для vkozomazov:
Updating and loading repositories:
Repositories loaded.
Пакет "pass-1.7.4-11.fc41.noarch" уже установлен.
Пакет "pass-otp-1.2.0-15.fc41.noarch" уже установлен.
Nothing to do.
Kozomazov@fedora: ~$
```

Рис. 4.1: Установка менеджера паролей

Завершил установку менеджера паролей командой `sudo dnf install gopass` (рис. 4.2).

```
Kozomazov@fedora: ~$ sudo dnf install gopass
Updating and loading repositories:
Repositories loaded.
Пакет "gopass-1.15.15-2.fc41.x86_64" уже установлен.
Nothing to do.
Kozomazov@fedora: ~$
```

Рис. 4.2: Завершение установки менеджера паролей

Просмотрел список `grp` ключей при помощи команды `grp --list-secret-keys` (рис. 4.3).

```
Kozomazov@fedora: ~$ grp --list-secret-keys
[keyboard]
-----
sec  Ts4d096 2025-03-14 [SC]
    4938B838E2482245D91F62F84FFBD2FEE4272D8C
uid  [ abcomeruo ] Vladimir Kozomazov <voffkoc@gmail.com>
sub  Ts4d096 2025-03-14 [E]
Kozomazov@fedora: ~$
```

Рис. 4.3: Просмотр списка ключей

Инициализировал хранилище, написав команду `'pass init` (рис. 4.4).

```
Kozomazov@fedora: ~$ pass init 4938B838E2482245D91F62F84FFBD2FEE4272D8C
Password store initialized for 4938B838E2482245D91F62F84FFBD2FEE4272D8C
```

Рис. 4.4: Инициализация хранилища

Создал структуру с git командой `pass git init` (рис. 4.5).

```
kozomazov@fedora:~$ pass git init
Перициализирован существующий репозиторий Git в /home/kozomazov/.password-store/.git/
kozomazov@fedora:~$
```

Рис. 4.5: Создание структуры

Синхронизировался с git командами `pass git pull`, `pass git push`  
Вручную закоммитил и выложил изменения командами (рис. 4.6).

```
kozomazov@fedora:~$ cd ~/.password-store/
kozomazov@fedora:~/.password-store$ git add .
kozomazov@fedora:~/.password-store$ git commit -am 'edit manually'
[main 6b7c4: master]
1 file changed, 8 insertions(+), 0 deletions(-)
komazov@fedora:~/.password-store$
```

Рис. 4.6: Ручное выкладывание изменений

Проверил статус синхронизации командой `pass git status` (рис. 4.7).

```
kozomazov@fedora:~/.password-store$ pass git status
Находясь ветка: master
Ветка соответствует «origin/master».
ничего коммитить, нет изменений в рабочем каталоге
komazov@fedora:~/.password-store$
```

Рис. 4.7: Проверка статуса синхронизаций

Добавил новый пароль командой `pass insert [OPTIONAL DIR]/[FILENAME]` (рис. 4.8).

```
kozomazov@fedora:~/.password-store$ pass insert tests/test1
An entry already exists for tests/test1. Overwrite it? [y/N] y
Enter password for tests/test1:
Retype password for tests/test1:
[main a41f640] Add given password for tests/test1 to store.
1 file changed, 8 insertions(+), 0 deletions(-)
komazov@fedora:~/.password-store$
```

Рис. 4.8: Добавление нового пароля

Отобразил пароль для указанного имени файла `pass [OPTIONAL DIR]/[FILENAME]`  
Установил дополнительное программное обеспечение (рис. 4.9).

```
light \
fuzzel \
swaylock \
kitty \
waybar swaybg \
wl-clipboard \
mpv \
grim \
slurp

[sudo] пароль для kozomazov:
updating and loading repositories:
repositories loaded.
Пакет "dunst-1.12.1-1.fc41.x86_64" уже установлен.
Пакет "fontawesome4-fonts-10.2.3-1.fc41.noarch" уже установлен.
Пакет "powerline-fonts-2.8.4-1.fc41.noarch" уже установлен.
Пакет "light-1.2.2-14.fc41.x86_64" уже установлен.
Пакет "fuzzel-1.11.1-2.fc41.x86_64" уже установлен.
Пакет "swaylock-1.8.0-1.fc41.x86_64" уже установлен.
Пакет "kitty-0.19.1-1.fc41.x86_64" уже установлен.
Пакет "waybar-0.11.0-1.fc41.x86_64" уже установлен.
Пакет "swaybg-1.2.1-2.fc41.x86_64" уже установлен.
Пакет "wl-clipboard-2.2.1-3.fc41.x86_64" уже установлен.
Пакет "mpv-0.39.0-1.fc41.x86_64" уже установлен.
Пакет "grim-1.4.1-4.fc41.x86_64" уже установлен.
Пакет "slurp-1.5.0-3.fc41.x86_64" уже установлен.

Nothing to do.
komazov@fedora:~$
```

Рис. 4.9: Установка дополнительного программного обеспечения

Установил дополнительно шрифты командами `sudo dnf copr enable peterwu/iosevka`, `sudo dnf search iosevka`, `sudo dnf install iosevka-fonts iosevka-aile-fonts iosevka-curly-fonts iosevka-slab-fonts iosevka-etoile-fonts iosevka-term-fonts` (рис. 4.10).

```

isevka-term-s100-fonts:match: Monospace, Liberation Mono Style
isevka-term-s107-fonts:match: Monospace, Monaco Style
isevka-term-s108-fonts:match: Monospace, Pragmatix Pro Style
isevka-term-s109-fonts:match: Monospace, Source Code Pro Style
isevka-term-s110-fonts:match: Monospace, Envy Code R Style
isevka-term-s111-fonts:match: Monospace, X Windows Fixed Style
isevka-term-s112-fonts:match: Monospace, Ubuntu Mono Style
isevka-term-s113-fonts:match: Monospace, Lucida Style
isevka-term-s114-fonts:match: Monospace, JetBrains Mono Style
isevka-term-s115-fonts:match: Monospace, IBM P5m Mono Style
isevka-term-s116-fonts:match: Monospace, PT Mono Style
isevka-term-s117-fonts:match: Monospace, Recursive Mono Style
isevka-term-s118-fonts:match: Monospace, Import Mono Style
isevka-term-s119-fonts:match: Monospace, Import Mono Style
$ sudo dm install iseovka-fonts iseovka-ale-fonts iseovka-curl-fonts iseovka-slab-fonts iseovka-etole-fonts iseovka-term-fonts
Updating and loading repositories:
Repositories loaded
[metk] iseovka-fonts-33.0.1-1.fc41.noarch.rpm уже установлен.
[metk] iseovka-ale-fonts-33.0.1-1.fc41.noarch.rpm уже установлен.
[metk] iseovka-curl-fonts-33.0.1-1.fc41.noarch.rpm уже установлен.
[metk] iseovka-slab-fonts-33.0.1-1.fc41.noarch.rpm уже установлен.
[metk] iseovka-etole-fonts-33.0.1-1.fc41.noarch.rpm уже установлен.
[metk] iseovka-term-fonts-33.0.1-1.fc41.noarch.rpm уже установлен.

```

Рис. 4.10: Установка дополнительных шрифтов

- Установил бинарный файл командой `sh -c "$(wget -qO- chezmoi.io/get)"`
- Создал собственный репозиторий при помощи утилит на основе шаблона
- Подключил репозиторий к своей системе
- Использовал `chezmoi` на нескольких машинах
- Настроил новую машину с помощью одной команды
- Включил функцию автоматической фиксации и отправки изменений в репозиторий

## 5 Выводы

**vi/vim** — один из самых мощных текстовых редакторов для работы в командной строке. Его изучение важно для системных администраторов, разработчиков и всех, кто работает с Unix-системами. Освоив базовые команды, можно эффективно редактировать файлы без графического интерфейса.

# Список литературы{unnumbered}