Проект эмуляции терминала

Выполнили:

- Хмельницкий Антон Б01-306
- Рачинский Максим Б01-303

Исходники: https://github.com/khmelnitskiianton/terminal-emulator

🎯 Задачи проекта

Цель проекта - создание терминального эмулятора, включающего графический терминал - **iksTerm** и оболочку **paraShell** для Linux.

Основные задачи:

- Разработка минималистичного терминала с графическим интерфейсом.
- Реализация командной оболочки для взаимодействия с пользователем.
- Изучение работы с псевдотерминалами (PTY) в Linux и взаимодействия между терминалом и оболочкой.
- Создание удобного интерфейса для использования терминала и оболочки пользователем.
- Оформление проекта в соответствии с современными стандартами: система контроля версий, документация, системы сборки, лицензия, описание проекта, а также его использования.

🧩 Структура проекта

1. `iksTerm` — графический терминал

- Отображает текстовый вывод и обрабатывает пользовательский ввод.
- Запускает дочерний процесс с оболочкой.
- Обеспечивает взаимодействие с оболочкой через PTY.
- Написан на C с использованием X11 и низкоуровневых системных вызовов.
- Имеет удобный пользовательский интерфейс, поддерживает различные опции по кастомизации.

2. `paraShell` - оболочка

- Реализует ввод, разбор и выполнение команд (в отдельных процессах, созданных с помощью системного вызова fork()).
- Реализовано межпроцессное взаимодействие посредством файловых дескрипторов и механизма pipe.
- Имеет встроенные команды(такие как *cd, exit*), может быть расширена для поддержки новых функций и команд.

⚙️ Сборка и запуск

Клонирование и сборка проекта

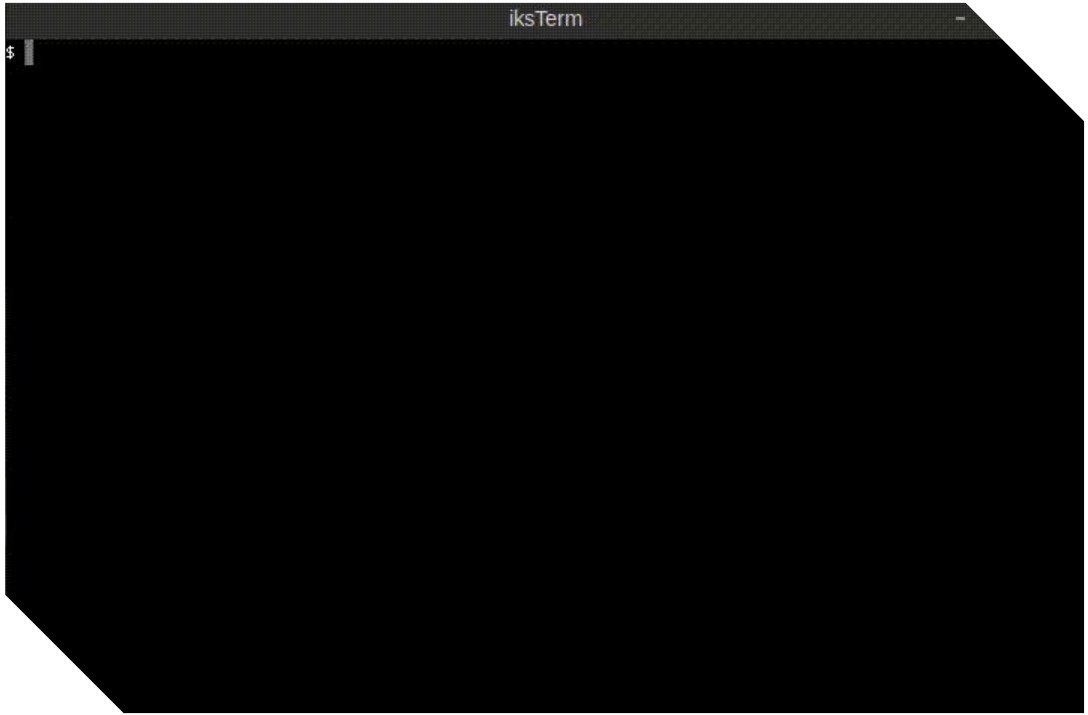
(Полное описание доступно на Github)

Установка и генерация описания

Запуск терминала на оболочке

iksTerm –shell=/usr/bin/paraShell

Альтернатива сборка в контейнере



terminal-emulator

Public

Pin | Unwatch 1

main | 3 Branches | 0 Tags | Go to file t | Add file | Code

maksimra add brief readme in paraShell — fbfe4ef · 2 days ago — 37 Commits

.media	Add docker	4 days ago
iksTerm	add readme and format files	5 days ago
paraShell	add brief readme in paraShell	2 days ago
.clang-format	change codestyle	last week
.gitignore	add brief readme in paraShell	2 days ago
Dockerfile	Add docker	4 days ago
LICENSE	Add WTFPL license	2 months ago
README.md	Add docker	4 days ago

README | WTFPL license

Installation

Build:

```
git clone https://github.com/khmelnitskiianton/terminal-emulator.git # clone repo
cd iksTerm
make # compile program store in "bin/" dir
./bin/iksTerm -h # local run
```

Installation:

```
sudo make install # install in /usr/bin
iksTerm -h
```

Installation

Local build:

```
git clone https://github.com/khmelnitskiianton/terminal-emulator.git # clone repo
cd terminal-emulator/paraShell
cmake -S . -B build
cmake --build build
build/paraShell
```

Installation:

```
cd terminal-emulator/paraShell
chmod +x ./install.sh
sudo ./install.sh # install in /usr/bin
```

Docker

If you want to build & install this project in isolated container and see terminal you can use our docker.

Install Docker:

```
sudo apt install docker.io
```

Build docker image:

```
sudo docker build -t iksterm .
xhost +local:docker
```

Run terminal in container:

```
sudo docker run --rm -it --net=host -e DISPLAY iksterm
```

License

iksTerm

Simple terminal emulator

Good Articles

- Basics of terminal emulator
- Anatomy of terminal emulator
- Theory of terminal emulator
- Linux pty article
- Linux tty understanding
- Linux TTY demystified
- Linux Kernel

Structure of terminal emulation

Terminal-emulation consist of 3 elements:

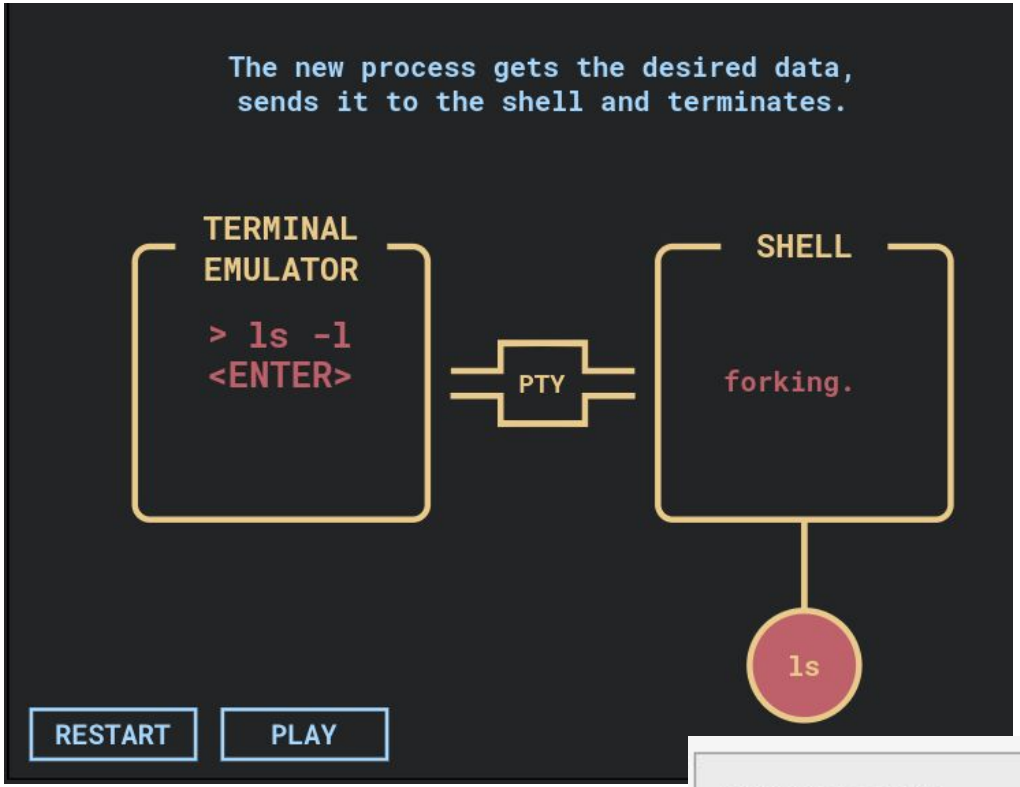
- Terminal
- Shell
- PTY

The shell is a program that provides an interface to the operating system, allowing the user to interact with its file-system, run processes and often have access to basic scripting capabilities.

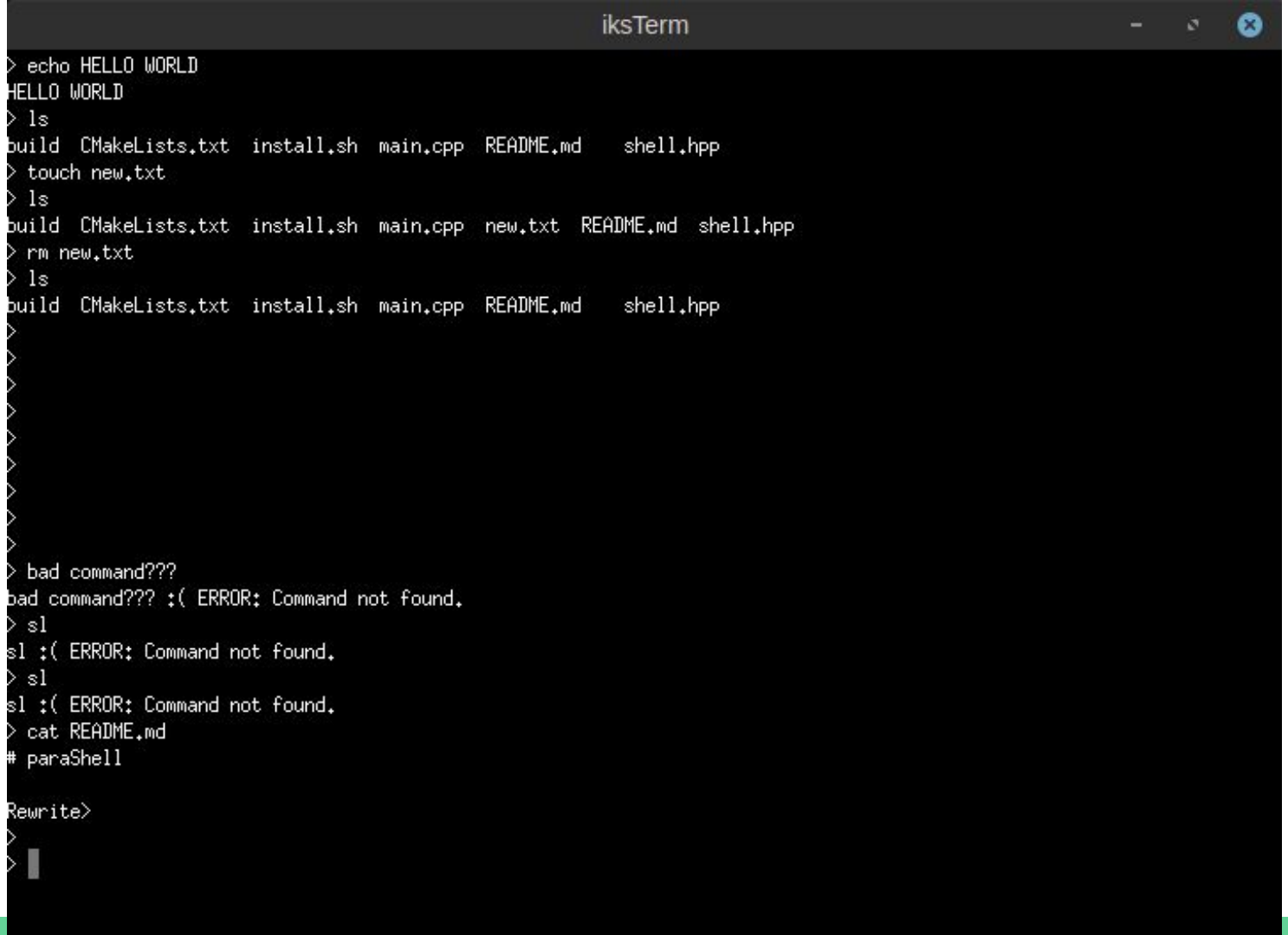
The terminal emulator is a graphical application whose role it is to interpret data coming from the shell and display it on screen.

PTY is a bi-directional asynchronous communication channel between the terminal and shell.

Реализация




```
+------------+      +=================+      +-----------------------+
| X11 client | <--> | pseudoterminal | <--> | terminal application |
+------------+      +=================+      +-----------------------+
```


iksTerm
> echo HELLO WORLD
HELLO WORLD
> ls
build CMakeLists.txt install.sh main.cpp README.md shell.hpp
> touch new.txt
> ls
build CMakeLists.txt install.sh main.cpp new.txt README.md shell.hpp
> rm new.txt
> ls
build CMakeLists.txt install.sh main.cpp README.md shell.hpp
> bad command???
bad command??? :(ERROR: Command not found.
> sl
sl :(ERROR: Command not found.
> sl
sl :(ERROR: Command not found.
> cat README.md
paraShell
Rewrite>

🛠️ Используемый стек технологий

- **Язык программирования:** C++, C
- **Программа разработки:** Visual Studio Code
- **Графическая библиотека:** X11 (X Window System)
- **Стандартные C библиотеки:** POSIX (PTY, процессы, файловые дескрипторы, ioctl), GNU C Lib(аргументы командной строки)
- **Элементы C++:** стандартные контейнеры C++, умные указатели, наследование и полиморфизм.
- **Сборка проекта:** CMake, Make
- **Документация:** Doxygen
- **Система контроля версий:** Git, Github
- **Система развертки приложения:** Docker
- **Описание проекта:** Man, Markdown, License
- **Дополнительные инструменты:** bear (для поддержки LSP и clang-based инструментов), clang-format

✅ Результаты и функциональность

- Реализован графический терминал с простым пользовательским интерфейсом.
- Разработана минималистичная, но расширяемая командная оболочка.
- Применены системные технологии Linux: PTY, fork, execv.
- Использованы стандартные контейнеры C++, умные указатели, наследование и полиморфизм.
- Проект оформлен с использование современных инструментов разработки: документация, структура, автоматизация сборки и развертки приложения.
- Протестирован запуск внешних программ, взаимодействие с пользовательским вводом и выводом.

Future Development

- [x] Resizing
- [ ] Custom font upload(improve)
- [ ] Process control sequences & signals
- [ ] UTF-8 except of ASCII
- [ ] Keep history and get it by arrows
- [x] Handle more control chars (`\b`)

Спасибо за внимание!