1.8. Знакомство с терминалом

~ 9 минут

Терминал запускается так же, как и любая другая программа.

На Windows мы будем запускать Git Bash, установленный вместе с Git. Не забывайте про возможность запускать терминал из Проводника, чтобы не пришлось вводить путь до рабочего проекта руками: эти пути в Windows бывают длинными и неудобными. Для этого нажмите в Проводнике правую кнопку мыши и выберите «Git Bash Here».

На macOS мы будем пользоваться стандартным терминалом, предустановленным в системе. Терминал можно найти в меню приложений в разделе Другие.

Терминал на macOS

Внешний вид

При запуске терминала мы увидим в нём приглашение командной строки:

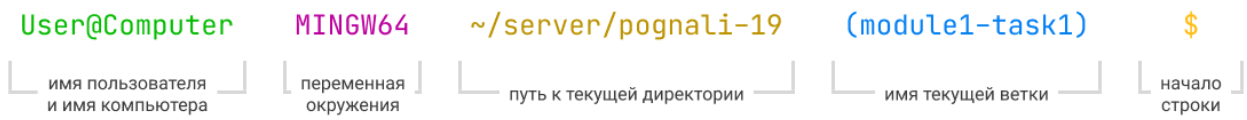


Схема приглашения командной строки

Давайте расшифруем это приглашение. Оно состоит из таких частей:

— имя пользователя и имя компьютера, разделенные символом `@`;

— переменная окружения, которая выводит системную информацию (может выводиться на Windows);

— путь к текущей директории, в которой будут выполняться команды;

  Когда вы работаете в терминале, вы всегда находитесь в какой-то папке или директории. При запуске терминала вы обычно оказываетесь в своей домашней папке. В настройках терминала можно указать, чтобы по умолчанию он открывался в другой папке.

  Важно знать, в какой папке вы находитесь. От этого зависит то, с какими файлами вы можете работать и как попадать в другие папки.

— имя текущей ветки, если в текущей директории есть репозиторий Git;

  Обратите внимание: имя ветки не будет отображаться по умолчанию в терминале macOS. Чтобы оно отображалось, нужно произвести дополнительные настройки.

— начало строки, в виде символа `$` или `%` (бывают и другие варианты). Оно сигнализирует о готовности принять команду. Символ может быть разным, в зависимости от командной строки.

  Например, в Git Bash работает bash (неожиданно), который будет обозначен `$`, а вот в macOS по умолчанию может стоять командная строка zsh, которая будет об символом `%`.

В командной строке могут быть и какие-то дополнительные данные. Всё это зависит от системы и довольно гибко настраивается и модифицируется.

Основные команды

— `pwd` — выводит путь к текущей директории. Можно резонно заметить: зачем нам путь, если он выводится в приглашении командной строки? Например, вы часто будете там видеть символ `~`. Он означает домашнюю директорию пользователя, то есть показывает не полный путь. А команда `pwd` выведет абсолютный путь до текущей директории.

— `cd` — перемещает по директориям. После команды необходимо указать путь.

> Про пути

— `ls` — выводит список папок и файлов. Команда без аргументов покажет список файлов в текущей директории. Если же указать после команды путь, то вы получите список файлов, находящихся по этому пути, например: `ls /c/'Program Files'`.

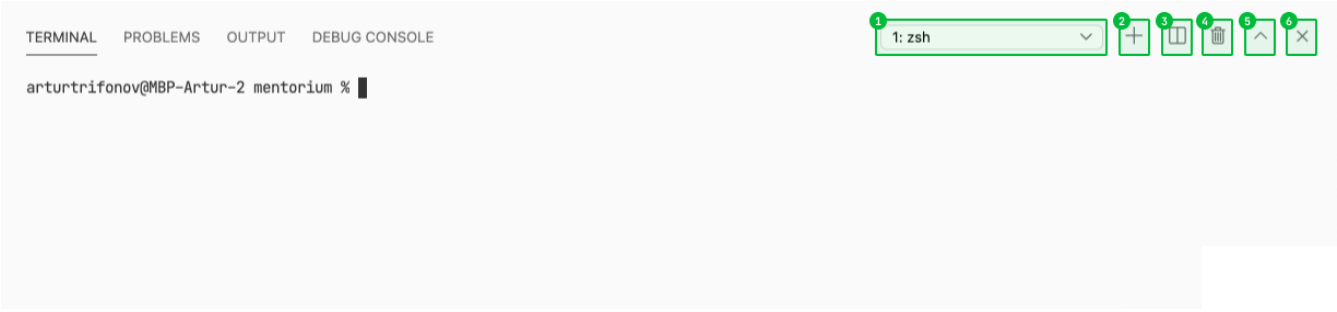
— Автодополнение — автоматически дописывает путь. По нажатию на `tab` терминал дополнит имя файла, либо предложит список имён, которые могут подойти.

> Как включить автодополнение на macOS

Встроенный терминал VS Code

Если вы работаете в VS Code, то можете воспользоваться встроенным в него терминалом. Чтобы его запустить, нужно выбрать в меню «Terminal → New Terminal», либо использовать сочетание клавиш `Ctrl + ` ` на Windows или `Control + ` ` на macOS.

Откроется окно, в котором вы увидите уже знакомую картину:



Вид терминала VS Code

Обратите внимание на правый верхний угол, в нём 6 инструментов:

1. Выпадающий список. VS Code позволяет запустить несколько терминалов и этот список переключает окно между ними.

> Переход на Git Bash для Windows

2. «New Terminal» добавляет новый терминал.
3. «Split Terminal» запускает новое окно терминала рядом.
4. «Kill Terminal» завершает работу терминала.
5. «Maximize Panel Size» расширяет панель с терминалом на всю высоту окна.
6. «Close Panel» закрывает панель. Постарайтесь не путать её с «Kill Terminal» — эта кнопка просто закроет панель, не завершая работу терминала: его состояние и процессы останутся. Открывать и закрывать панель можно через меню «View → Appearance → Show Panel», но намного удобнее будет это делать сочетанием кнопок `Command + J` (`Ctrl + J` для Windows).

Терминал, встроенный в редактор может быть удобен, так как с ним не придётся переключаться на отдельное системное окно. Он полностью функционален и можно смело использовать его для работы. Но не забудьте поставить Git, ведь терминалы — это лишь один из инструментов для работы с ним, и по умолчанию он туда не включен.

Проверим, что Git установлен

После того, как все действия по установке завершены, убедимся, что Git появился в системе компьютера. Откройте терминал и введите `git --version`, должна появиться текущая версия программы на вашей машине. Эта проверка подходит для всех операционных систем.

Настройка Git

После того как Git появился на компьютере, нужно ввести свои данные — имя и адрес электронной почты. Ваши действия в Git будут содержать эту информацию.

Откройте терминал и используйте следующую команду, чтобы добавить своё имя:

```
git config --global user.name "ваше имя"
```

Для добавления почтового адреса вводите:

```
git config --global user.email "ваш email"
```

Обратите внимание, что в командах, указанных выше, есть опция `--global`. Это значит, что такие данные будут сохранены для всех ваших действий в Git и вводить их больше не надо. Если вы хотите менять эту информацию для разных проектов, то в директории проекта вводите эти же команды, только без опции `--global`.

Проверить, что настройки сохранились можно при помощи команды:

```
git config --list
```

Эти настройки хранятся в файле `.gitconfig` в папке вашего пользователя. Посмотреть содержимое этого файла можно при помощи команды:

```
cat ~/.gitconfig
```

Можно открыть этот файл в редакторе и поменять настройки или скопировать его на другой компьютер, например, с домашнего компьютера на рабочий, чтобы на работе были такие же настройки, как дома.

Git настроен.

Прочитали главу?

Нажмите кнопку «Готово», чтобы сохранить прогресс.

Готово

- [ ] Изучите регламент
- [ ] Прочитайте FAQ
- [ ] Добавьте свой Гитхаб
- [ ] Выберите наставника
- [ ] Создайте проект

Мой наставник

Выбрать наставника

Работа с наставником

У вас осталось **10** из 10 консультаций.

История

Курсы

HTML и CSS. Профессиональная вёрстка сайтов

HTML и CSS. Адаптивная вёрстка и автоматизация

JavaScript. Профессиональная разработка веб-интерфейсов

JavaScript. Архитектура клиентских приложений

React. Разработка сложных клиентских приложений

Node.js. Профессиональная разработка REST API

Node.js и Nest.js. Микросервисная архитектура

TypeScript. Теория типов

Алгоритмы и структуры данных

Паттерны проектирования

Webpack

Vue.js 3. Разработка клиентских приложений

Git и GitHub

Анимация для фронтендеров

Блог

С чего начать

Шпаргалки для разработчиков

Отчеты о курсах

Услуги

Работа наставником

Для учителей

Стать автором

Информация

Об Академии

О центре карьеры

Остальное

Написать нам

Мероприятия

Форум

Соглашение

Конфиденциальность

Сведения об образовательной организации

Лицензия № 4696

Sk

Участник