**1. Настройка окружения и знакомство с командной строкой**

**Введение**

**О чём этот модуль**

Добро пожаловать!

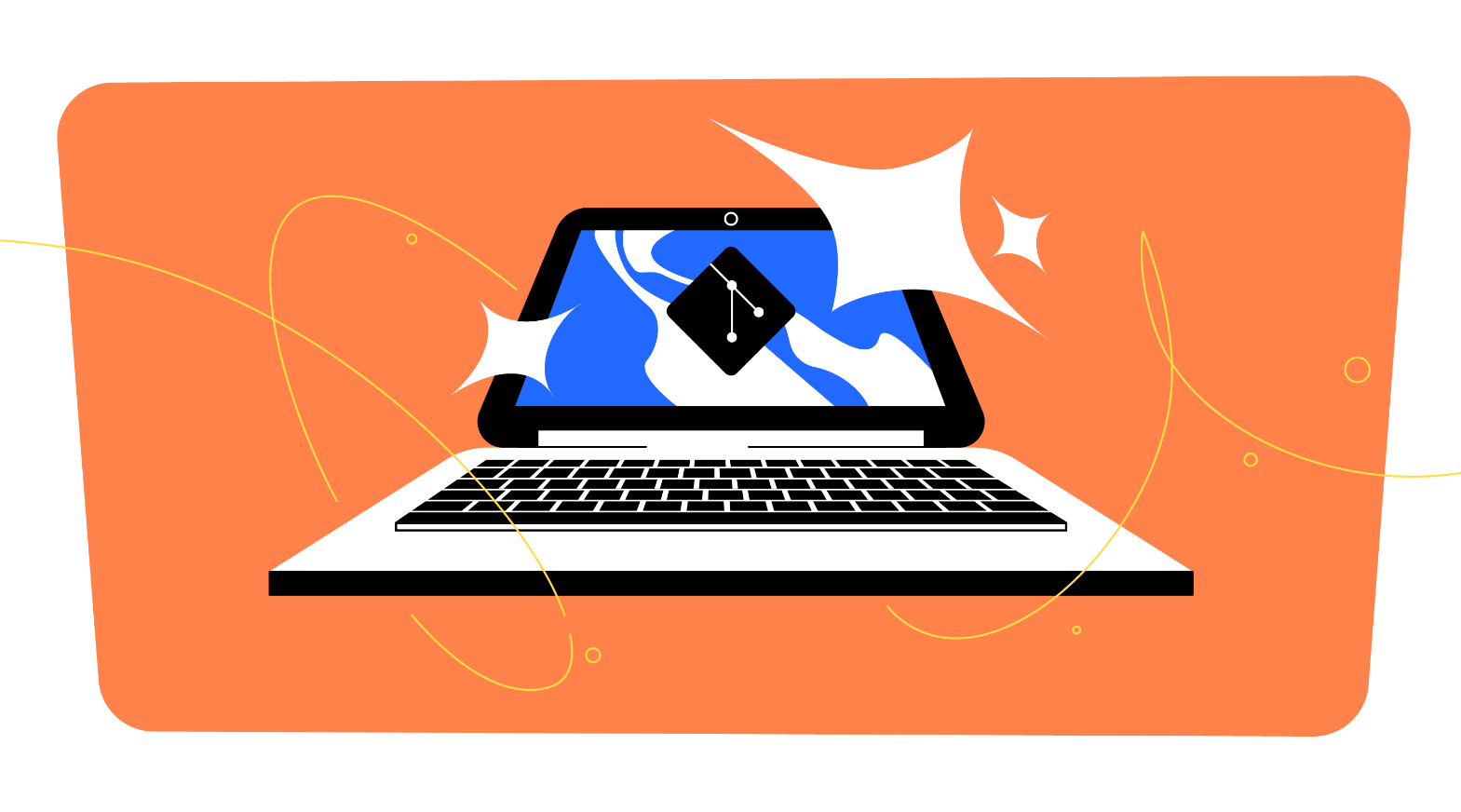
Git — это программа, такая же, как Zoom, MS Office или Photoshop, а значит, она требует установки и предварительной настройки. Но, в отличие от перечисленных программ, у Git нет привычного графического интерфейса, и работа с этим инструментом происходит иначе.

Этот модуль — для тех, кто только начинает знакомиться с Git. Мы расскажем:

* что такое Git и в чём его польза;
* зачем нужна командная строка;
* как перемещаться по файловой системе компьютера с помощью командной строки;
* как выполнять в командной строке ключевые операции с папками и файлами — создание, копирование, перемещение, чтение и удаление;
* как установить Git на компьютер и правильно всё настроить.

Изучение командной строки поможет вам лучше понять процесс установки Git и использовать как можно больше его возможностей. К концу модуля вы будете готовы к полному погружению в мир Git — работе с коммитами.

Желаем удачи!



# 2. Что такое Git?

Любой творческий процесс состоит из набора итераций. Художник рисует эскизы, фотограф делает несколько кадров, студент создаёт файлы диплом.doc, диплом\_final.doc и диплом\_final\_final\_2.doc. Каждая такая попытка имеет ценность. Она позволяет остановиться и убедиться, что работа идёт в правильном направлении.

В отличие от художников, IT-команды не используют скетчбуки. Историю их проектов хранит отдельная программа — **система контроля версий** (англ. ***V***ersion ***C***ontrol ***S***ystem, или коротко ***VCS***).

💡 Для обозначения систем контроля версий используют не только аббревиатуру VCS, но и **SCM** (от англ. ***S***ource ***C***ontrol ***M***anagement — «система управления исходным кодом»).

Система контроля версий, или VCS, — это программное обеспечение, которое помогает отслеживать изменения в программах, текстовых файлах, больших документах, веб-сайтах и так далее.

Одно изменение или группу изменений в VCS называют **ревизией** или **версией**. Каждая такая ревизия содержит информацию о том, что изменилось, кто внёс изменения, когда это было и иногда комментарии к изменению.

Например, первая версия нового проекта может содержать подобную информацию:

* новые файлы: info.txt;
* автор: Иван Иванов;
* дата: 1515 марта 20132013 года 13:5313:53.

А следующая, вторая версия — такую:

* новые файлы: authors.txt;
* изменённые файлы: info.txt;
* автор: Мария Петрова;
* дата: 1717 марта 20132013 года 10:0310:03.

Основные функции системы контроля версий:

* хранит историю изменений в виде отдельных ревизий;
* позволяет манипулировать историей: например, менять порядок ревизий, полностью удалять версии, возвращаться назад в истории;
* помогает анализировать изменения: например, кто и когда вносит изменения, кто чаще всего вносит изменения в определённый файл и так далее.

Одна из ключевых особенностей современных систем контроля версий — поддержка параллельной работы нескольких пользователей, в том числе над одним файлом. Именно поэтому VCS так популярны у IT-команд.

Начало формы

Какую из перечисленных задач решает система контроля версий?



Проверка содержимого проекта.

За проверку обычно отвечают другие компоненты проекта.



Правильный ответ

Хранение истории изменений.

Хранение, изменение и анализ истории — основные функции системы контроля версий.



Компиляция программы.

Основная задача системы контроля версий — хранить историю изменений. Для этого даже необязательно, чтобы проект под контролем VCS был программой.



Своевременное удаление ненужной истории изменений.

Система контроля версий не будет самостоятельно удалять историю — разве что вы сами попросите её об этом.



Контроль версии установленного программного обеспечения.

Этим занимаются системные администраторы.

Конец формы

Система контроля версий — общее название ряда продуктов, таких как Git, Mercurial, Subversion и других. В этом курсе мы будем говорить о самом популярном из них — **Git**.

💡 **Как расшифровывается Git?**

На английском сленге слово git означает «мерзавец». Но, по мнению Линуса Торвальдса, создателя Git, название может расшифровываться как угодно — в зависимости от настроения пользователя.

Например, Global Information Tracker (англ. «глобальный информационный трекер») — когда у вас всё хорошо и Git работает отлично. А если что-то идёт не так, Git превращается в Goddamn Idiotic Truckload of sh\*t (англ. «чёртов идиотский грузовик c ...» — тут вы наверняка справитесь с переводом и без нас).

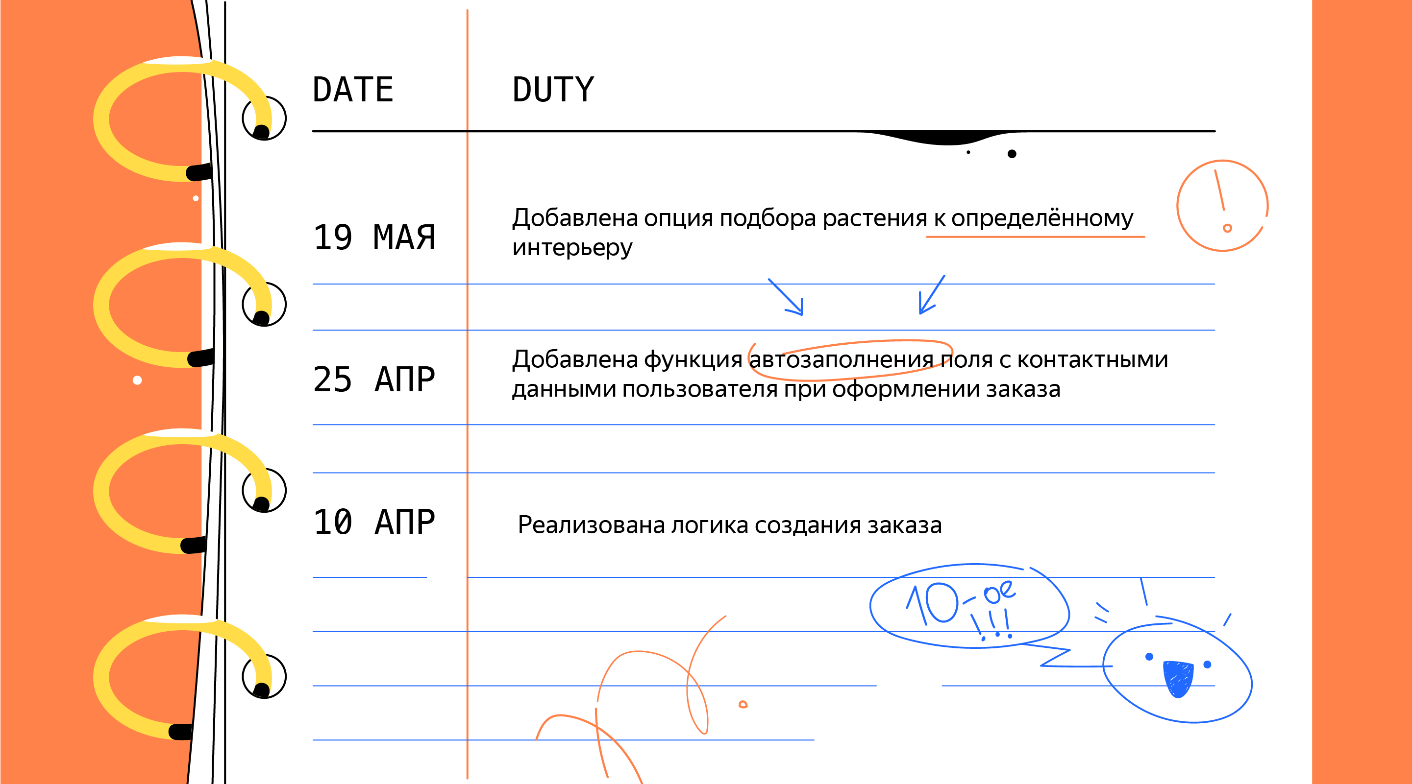
### Git и контроль версий

Мы сказали, что VCS позволяют контролировать версии проекта. Чтобы убедиться, почему это так важно, рассмотрим пример.

Допустим, вы программист и написали приложение для магазина комнатных растений. Оно работает. Пользователи украшают свои квартиры монстерами, бегониями и фикусами. Все довольны. Но однажды вам пишет заказчик и просит добавить новую функциональность — опцию подбора растения к определённому интерьеру. Ему кажется, что так приложение станет ещё более популярным.

Вы внесли в код изменения, обновили проект и на время забыли об этом. Однако через время заказчик сообщает, что его предположение оказалось неверным. Нужно всё вернуть как было. Предыдущие версии файлов у вас не сохранились. Придётся переписывать код заново!

А если бы вы использовали Git, этой проблемы бы не возникло. Git отследил бы все изменения и сохранил разные версии кода на разных этапах написания и переработки программы.



🔑 Чтобы пройти этот курс, вам необязательно уметь программировать. В примерах и заданиях уроков мы будем давать обычные текстовые файлы, а не код.

Тем не менее, чтобы лучше понять работу в IT-командах, вы можете использовать GIt для хранения кода. Не умеете программировать? Не беда! У нас много курсов, на которых мы показываем, как написать свою первую программу.

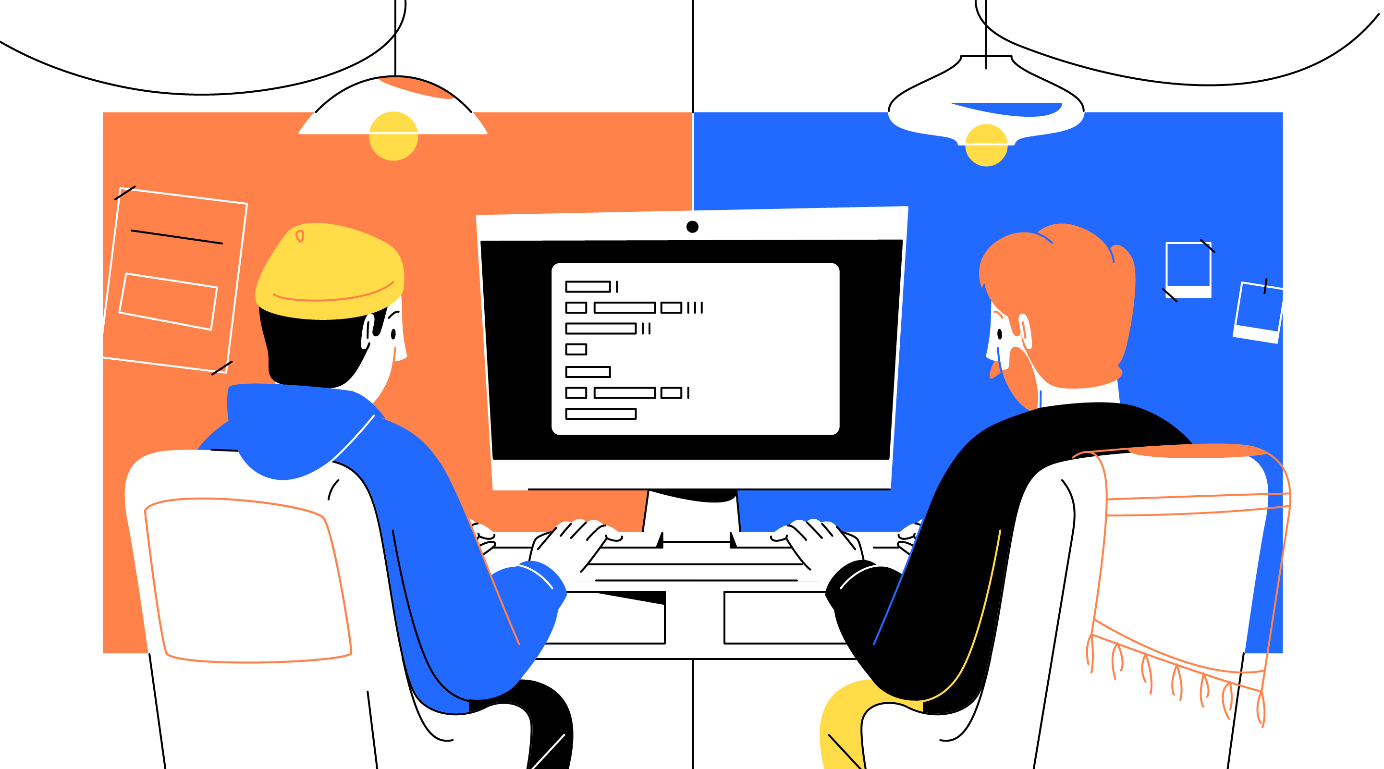
Например, вы можете познакомиться с популярными языками программирования [Python](https://practicum.yandex.ru/backend-developer/?from=catalog), [Java](https://practicum.yandex.ru/java-developer/?from=catalog) и [С++](https://practicum.yandex.ru/cpp/?from=catalog) или попробовать себя [в разработке веб-сайтов](https://practicum.yandex.ru/frontend-developer/?from=catalog). У всех этих курсов есть бесплатная вводная часть, и их можно изучать параллельно с основами Git.

### Git и работа в команде

В этом курсе вы будете решать задачи в одиночку. Но в реальной жизни над одним проектом может работать сразу несколько человек. И без Git им было бы очень сложно!

Представьте: два друга создают приложение, которое строит интересные маршруты для путешествий. Каждый раз, когда один из друзей вносит изменения в код, ему приходится отправлять другому обновлённый архив. Другой скачивает архив, добавляет свои правки и высылает назад. Но в какой-то момент один из них не стал ждать, пока другой завершит свою часть работы, и изменил код.

В результате они одновременно прислали друг другу обновлённые архивы. Совместить их не получится. Правок много, и друзья уже не помнят, где и что меняли. Им придётся открыть код в двух колонках и сверять всё вручную. На это уйдёт много времени и сил. А вот если бы они использовали систему контроля версий, Git сделал бы всё автоматически.



Git стал незаменимым инструментом в командной работе именно благодаря возможности сохранять и «склеивать» труд разных людей. Большинство работодателей в сфере IT ожидают, что соискатель понимает, зачем нужна система контроля версий, и умеет её использовать.

Начало формы

В каких случаях стоит использовать Git?

Правильный ответ

Для синхронизации изменений, сделанных разными участниками команды.

Git используют для синхронизации данных между участниками одной или нескольких команд.

Тоже правильный ответ

Чтобы хранить историю изменений файлов при командной работе.

Git может определить, кто и в какой момент внёс изменения.



Для проверки файлов проекта на плагиат.

Git не поможет бороться с плагиатом — он не сравнивает файлы разных проектов.

Тоже правильный ответ

Для хранения нескольких версий проекта, над которым работает один человек.

И это тоже верный ответ! Git позволяет «откатиться» к предыдущим версиям, если что-то пошло не так.

Конец формы

Поздравляем! Первый урок позади, а ваше знакомство с Git состоялось. Подытожим:

* Система контроля версий, или VCS (SCM), — программа, позволяющая контролировать изменения в проекте.
* Git — один из примеров системы контроля версий: он позволяет хранить, изменять и анализировать историю проекта.
* Git — незаменимый в команде инструмент, ведь он помогает объединять результаты работы нескольких человек.

# Установка командной строки для пользователей Windows

Командная строка — один из основных инструментов взаимодействия с компьютером. Подробнее о ней мы расскажем в следующем уроке. А пока вам может понадобиться выполнить подготовительные действия.

⚠️ Если вы пользователь macOS или Linux, смело пропускайте этот урок — командная строка у вас уже установлена. Но если ваш компьютер работает на базе Windows, вам понадобится установить командную строку вручную. Для этого следуйте инструкции ниже.

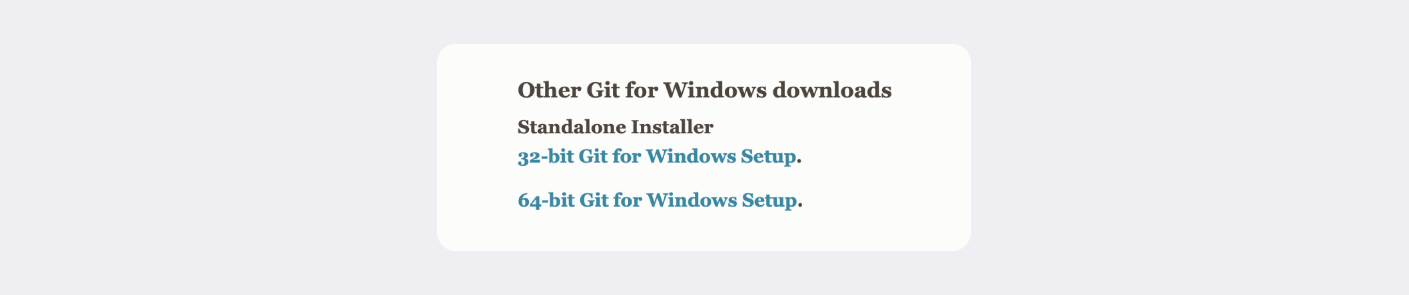
### Инструкция по установке для пользователей Windows

Windows поставляется с консолью — как и другие операционные системы. Но при этом команды консоли Windows и macOS/Linux отличаются друг от друга.

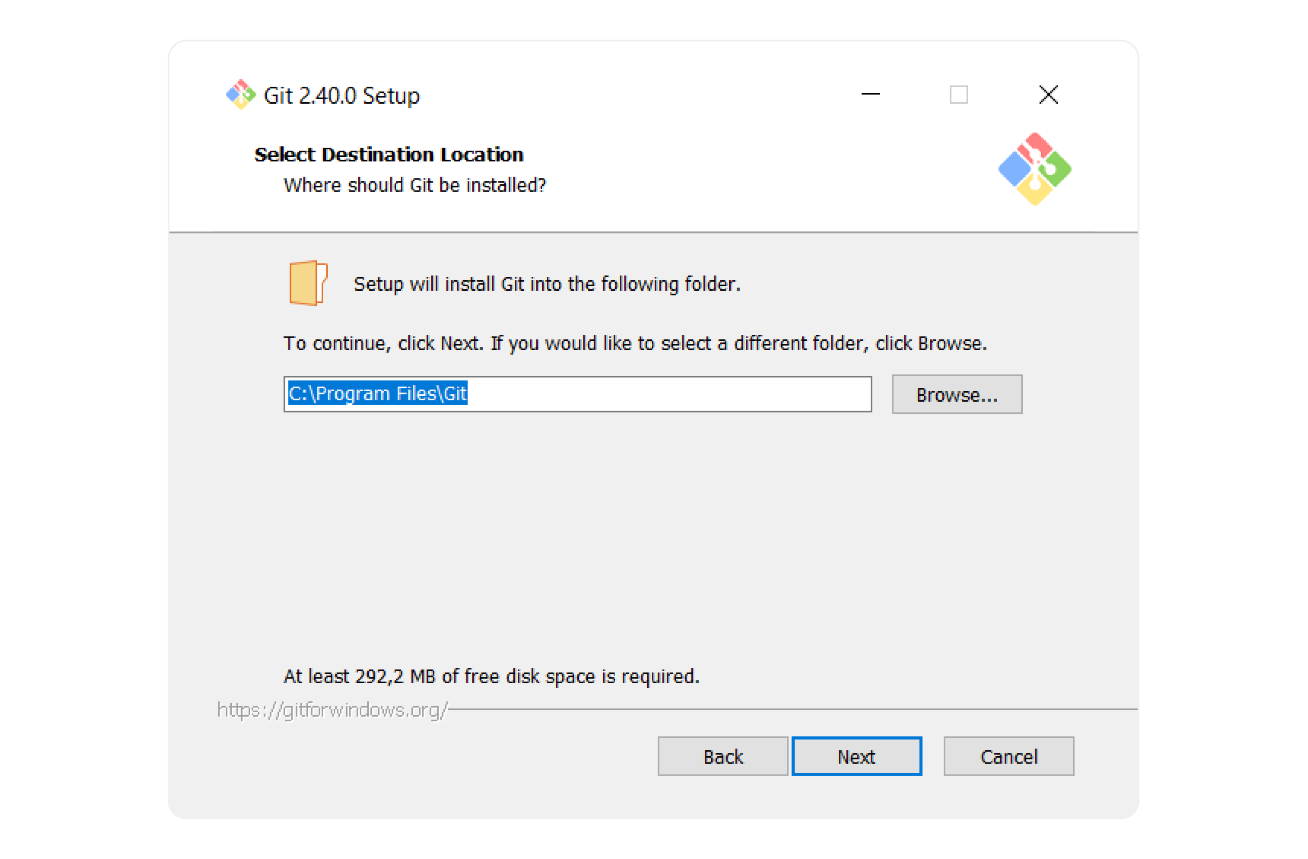
В повседневной работе большинство пользователей Git используют консоли с наборами команд, похожие на те, что применяют в macOS и Linux. В нашем курсе вы будете учиться делать то же самое. Для этого нужно установить специальный консольный инструмент для Windows, который называется **Git Bash**.

Есть несколько способов установки. Мы рекомендуем пакет **Git for Windows**. Он установит не только Bash, но и сам Git, который всё равно понадобится вам дальше. Вот что нужно сделать:

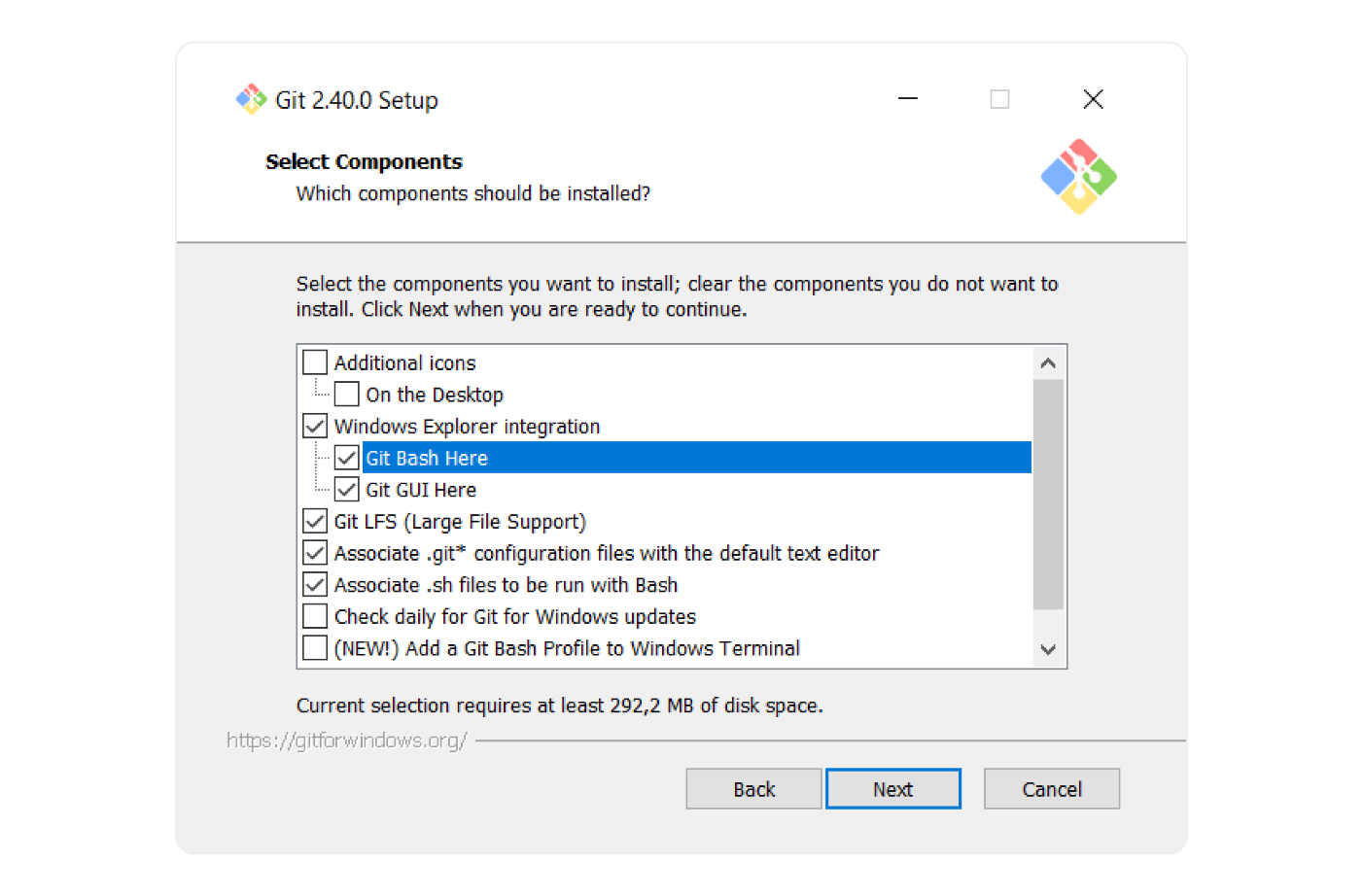
1. Перейдите [на эту страницу официального сайта Git](https://git-scm.com/download/win).
2. Скачайте одну из двух версий из категории **Standalone Installer** (англ. «автономный установщик»). Узнать тип вашей системы Windows можно в настройках.



1. Запустите программу установки. Обратите внимание, куда будет установлен Git. Обычно это директория C:\Program Files\Git.



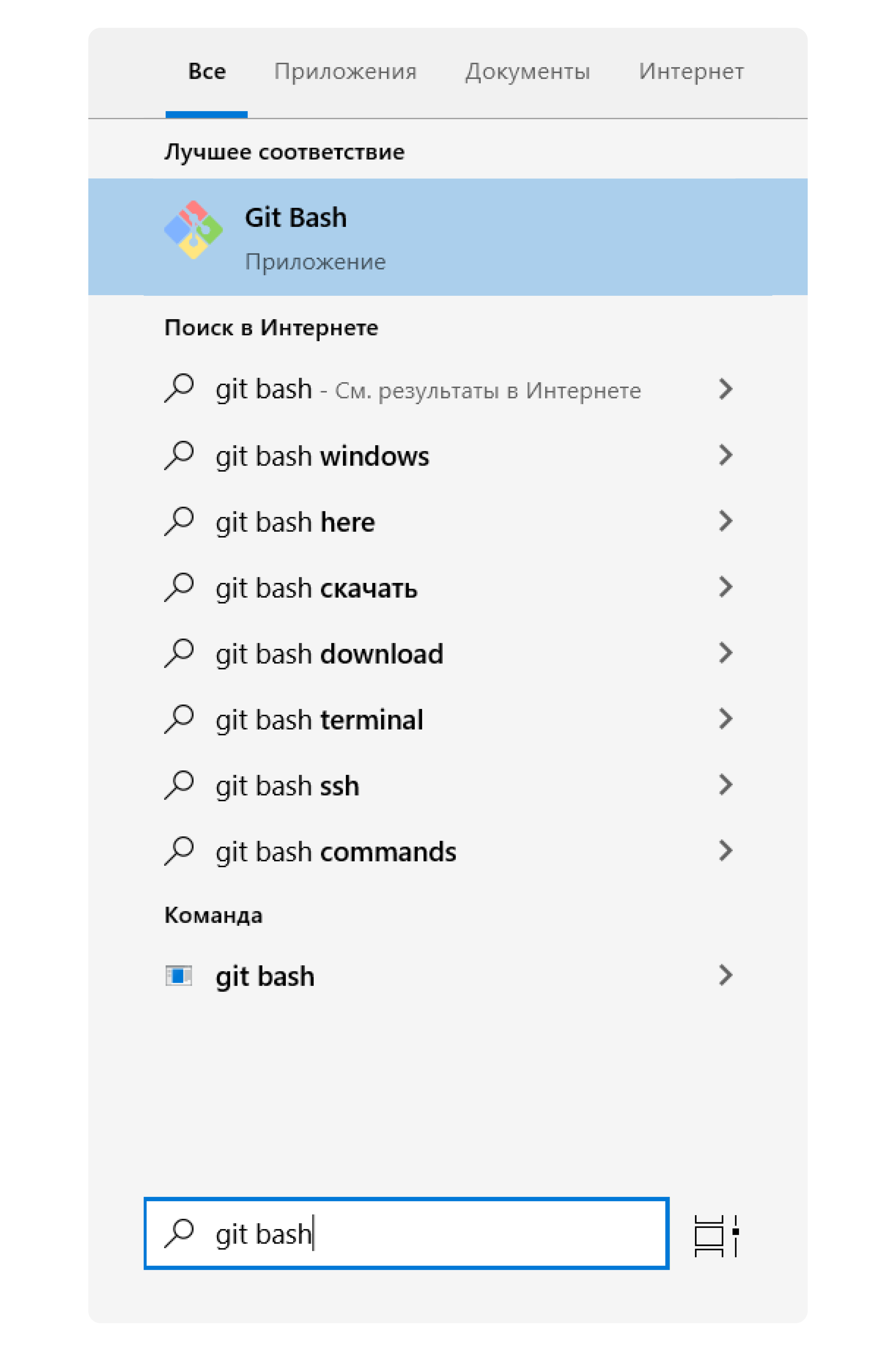
1. Проверьте, что в списке устанавливаемых программ стоит галочка напротив пункта **Git Bash Here** — это позволит открывать консоль с Git в любой папке.



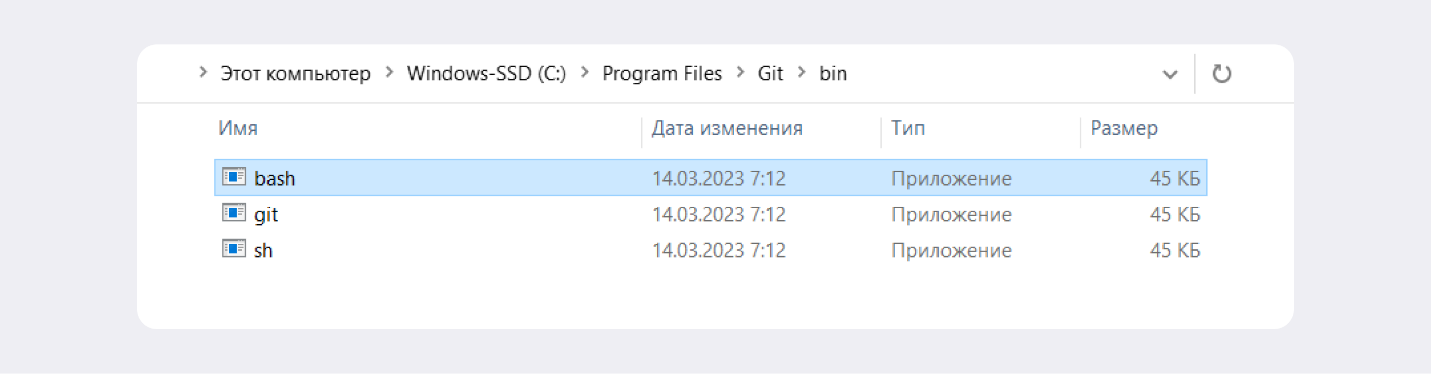
1. Далее установщик предложит много опций. Для нашего курса достаточно оставить все настройки по умолчанию. Несколько раз нажмите **Next** (англ. «далее»), пока не начнётся процесс установки.
2. После окончания установки нажмите **Finish** (англ. «завершить»).

### Первый запуск Git Bash

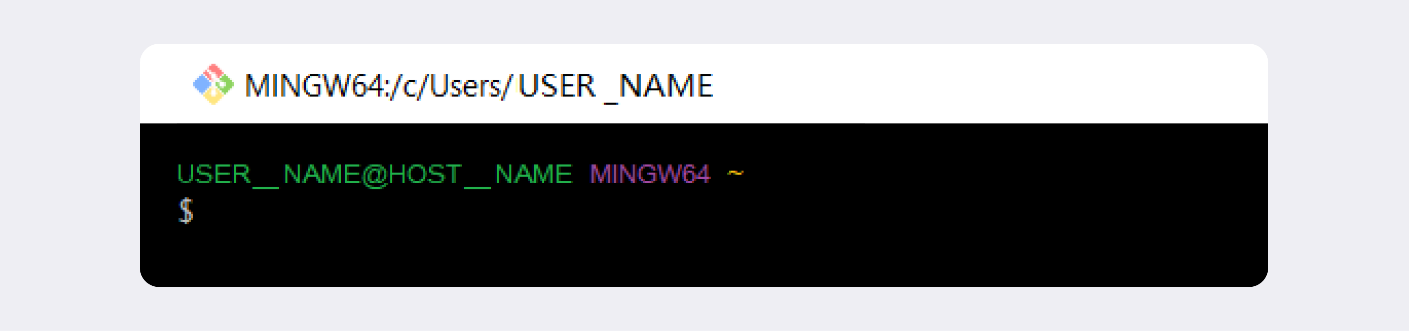
Запустите программу Git Bash. Сделать это можно двумя способами. Можно ввести название программы в окно поиска на панели задач.



А можно открыть директорию, в которую был установлен Git. Обычно это директория C:\Program Files\Git\bin. Перейдите в bin и запустите файл bash.exe.



Откроется консоль, в которой будет написано что-то похожее.



Вместо USER\_NAME будет указано ваше имя пользователя, а вместо HOST\_NAME — имя компьютера. Если вы видите консоль, значит, установка прошла успешно. В нескольких следующих уроках покажем, как работать с ней.

Ура! Теперь на вашем компьютере есть не только командная строка, но и Git. Устанавливать его вам уже не потребуется.

# Знакомство с командной строкой

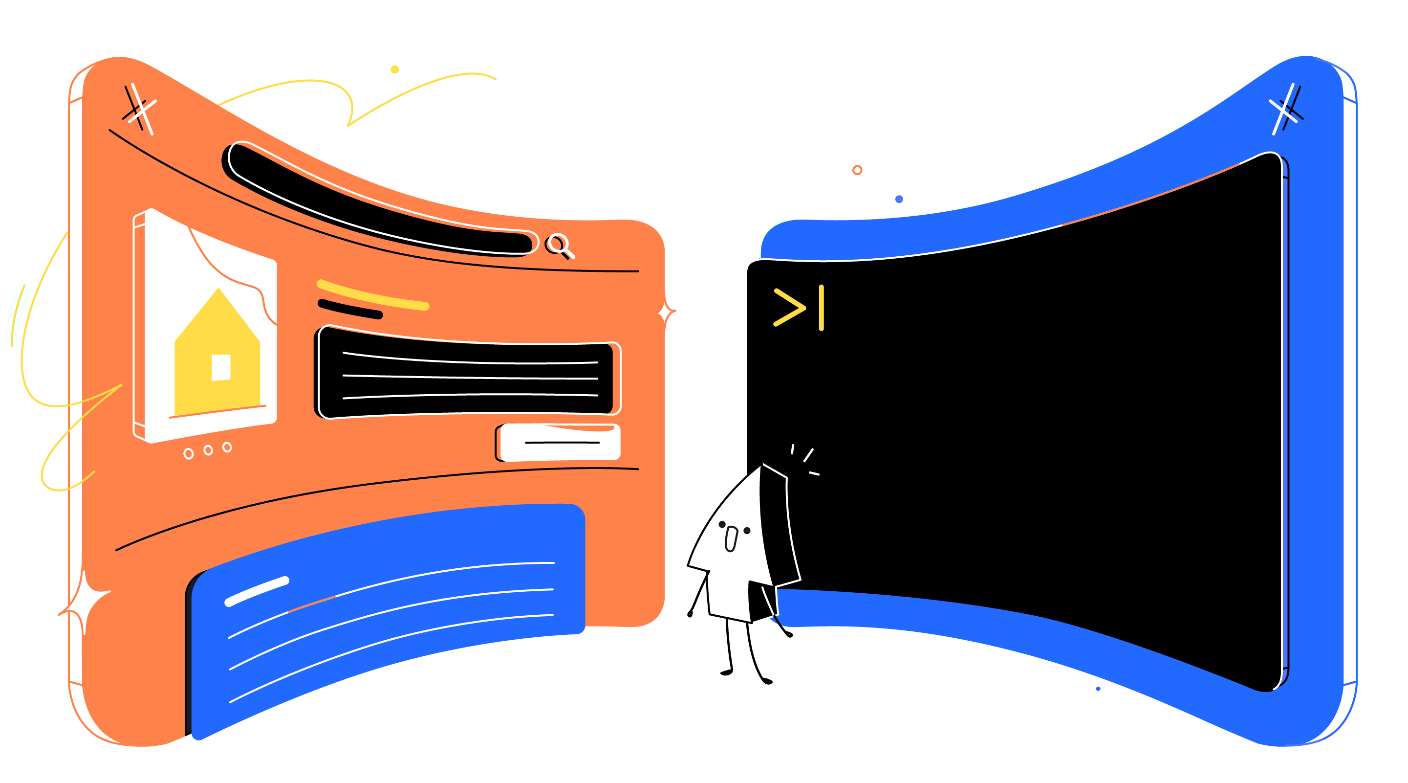
В этом уроке разберём, чем отличается графический интерфейс от командной строки и какова её роль во взаимодействии с Git. А ещё вы сделаете первые шаги в работе с консолью.

### Графический интерфейс vs командная строка

У программ есть привычный **графический интерфейс** (англ. ***G***raphical ***U***ser ***I***nterface, или GUI). Это окна, значки, кнопки, выпадающее меню с настройками и прочие элементы.

**Командная строка** (англ. ***C***ommand-***l***ine ***I***nterface, или CLI) — тоже интерфейс, только текстовый. Пользователь вводит в неё команды. Она принимает их от пользователя и выполняет. Эта строка — обычная программа на вашем компьютере. Такая же, как, например, браузер, в котором вы читаете этот урок.

Чтобы создать папку, вы чаще всего нажимаете правую кнопку мыши и выбираете нужный пункт в графическом меню. Но это можно сделать и через командную строку — достаточно ввести соответствующую команду и имя папки.



Командную строку часто называют **терминалом** или **консолью**. Эти термины пришли из тех времён, когда компьютеры выглядели совсем не так, как сейчас. У них не было привычного интерфейса. Вводить данные или вызывать программы можно было только через специальные устройства: терминалы и консоли. С их помощью происходило взаимодействие человека с компьютером.

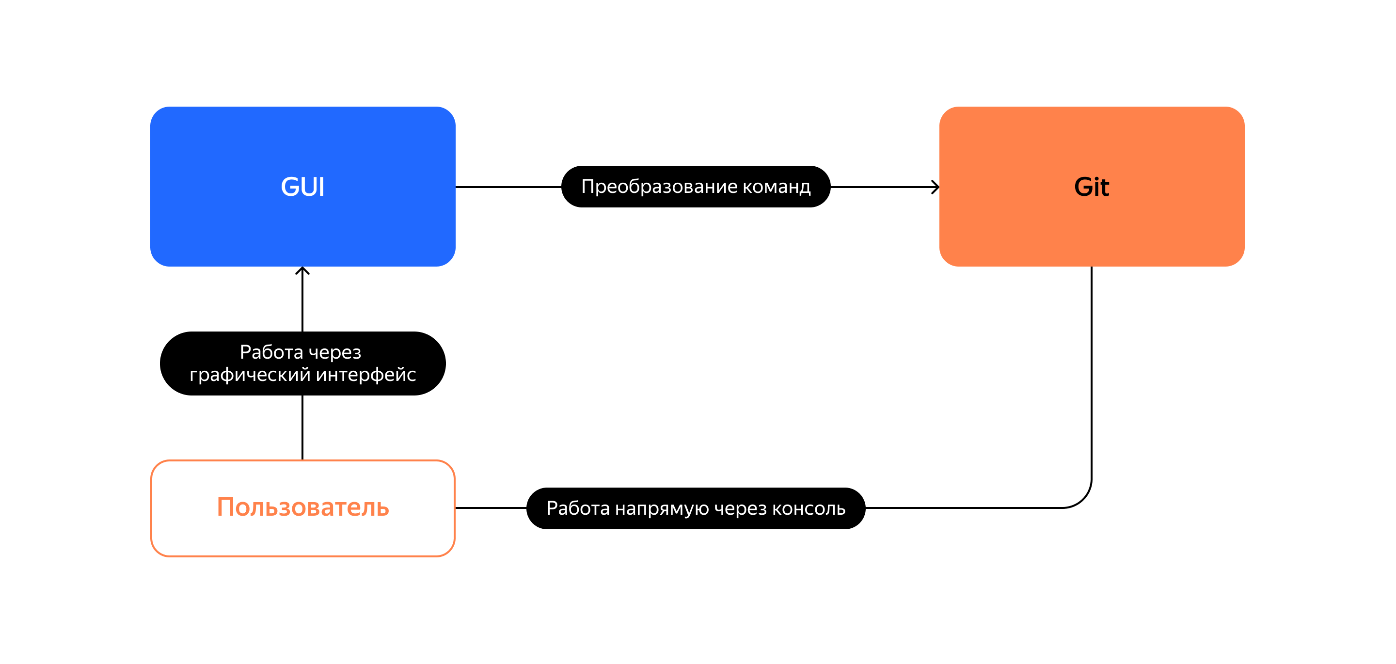
💡 **Консоль: атавизм или необходимость?**

Сначала все операционные системы были консольными, поэтому с компьютерами могли работать только программисты. Графический интерфейс — с окошками и кнопками — появился гораздо позже и изменил мир. Теперь компьютерами могут пользоваться все.

Но и сегодня многие программы изначально пишут в виде консольных приложений — и лишь затем у них появляется графический интерфейс. Это удобно, ведь у некоторых компьютеров, например у серверов в центрах обработки данных, нет GUI. Консоль помогает взаимодействовать с ними.

### Командная строка и Git

Git — это программа, которая в том числе может работать из командной строки. Любой графический интерфейс для Git всего лишь преобразует клики пользователя в вызовы программы.



Если вы научитесь работать через терминал, в дальнейшем вам будет просто освоить графический интерфейс для Git. Поэтому в этом курсе мы сосредоточимся на работе с командной строкой.

Начало формы

Выберите верные утверждения о командной строке.

Правильный ответ

«Терминал», «консоль» и «командная строка» используются для обозначения одного и того же вида программ.

В обиходе вы можете услышать каждый из этих вариантов.



Интерфейс консольных приложений и графический интерфейс не могут быть связаны между собой.

Многие графические интерфейсы вызывают команды консоли.

Тоже правильный ответ

Консоль — это программа на компьютере.

Терминал — такая же программа, как, например, браузер. На вашем компьютере может быть установлено несколько разных видов терминалов.



У всех консольных приложений есть графический аналог.

Далеко не у всех, хотя у части популярных консольных приложений он действительно есть.

Конец формы

### Первые шаги

Пришло время поработать в командной строке! Начните с запуска программы.

Если вы пользователь macOS или Linux, запустите программу **Terminal**. Её можно найти через окно поиска операционной системы. А можно использовать комбинацию горячих клавиш:

* для Linux — Ctrl+Alt+T;
* для macOS — Cmd+Space, затем ввести terminal.

Если вы пользователь Windows, запустите программу Git Bash — как её установить, мы рассказали в прошлом уроке.

В открывшемся окне вы увидите:

* имя вашего компьютера;
* имя, под которым вы авторизовались в компьютере;
* символ доллара ($) — он означает, что программа ждёт ваших команд.

В зависимости от операционной системы порядок компонентов и внешний вид строки могут немного различаться.

Так это будет выглядеть в Windows и Linux.

Скопировать код

BASH

userName@ComputerName ~

А так — в macOS.

Скопировать код

BASH

computerName:~ userName$

В дальнейшем мы будем оставлять в примерах только символ $ — без имени компьютера и имени пользователя.

### Как не потеряться

В графическом интерфейсе всегда ясно, где именно вы находитесь в файловой системе. Если перед вами рабочий стол, значит, вы в папке «Рабочий стол». Если открыта папка «Документы» — вы в ней.

В командной строке вы тоже всегда находитесь в какой-то папке — просто этого не видно. Узнать, где вы сейчас, поможет команда pwd (от англ. ***p***rint ***w***orking ***d***irectory — «показать рабочую папку»). Она выводит путь к текущей директории.

Введите pwd в командную строку и нажмите клавишу Enter для выполнения команды.

Скопировать код

BASH

$ pwd

/c/Users/%USERNAME%

/Users/%USERNAME%

Командная строка выведет путь к папке, в которой вы сейчас находитесь. Вместо %USERNAME% будет ваше имя пользователя. Это путь к **домашней директории** (англ. home directory) — каталогу с файлами пользователя. Обычно здесь хранится такая информация, как загрузки, медиа, скриншоты и так далее. Когда вы открываете командную строку, вы оказываетесь именно в домашней директории.

💡 **Файловые системы — это непросто**

Операционные системы Windows, Linux и macOS отличаются друг от друга. В том числе в том, как работают их файловые системы.

Например, для Windows путь начинается с буквы диска C — /c/ или c:/. В Linux нет букв дисков, а домашняя директория находится в /home вместо /Users. В macOS есть папка /Users, но нет букв дисков.

Так что наши примеры могут немного отличаться от того, что вы видите на вашем компьютере.

С помощью терминала вы всегда можете перейти к домашней директории. Для этого нужно ввести команду cd (от англ. ***c***hange ***d***irectory — «сменить директорию») и символ ~ — обозначение домашней директории. Не забудьте про Enter для выполнения.

Скопировать код

BASH

$ cd ~

Домашняя директория может быть разной для разных пользователей одного и того же компьютера.

Подробнее о структуре файловой системы и команде cd мы расскажем в следующих уроках.

Начало формы

Выберите верные утверждения про домашнюю директорию

Правильный ответ

Перейти к домашней директории можно с помощью команды cd ~.

Символ ~ — обозначение домашней директории.

Неправильный ответ

Чтобы вывести домашнюю директорию, необходимо ввести команду pwd.

Команда pwd выводит текущую рабочую директорию, которая необязательно является домашней.



У одного пользователя может быть несколько домашних директорий.

У одного пользователя может быть только одна домашняя директория.

Правильный ответ

Домашняя директория содержит файлы конкретного пользователя.

По умолчанию бóльшая часть файлов пользователя (например, картинки, медиа, загрузки и прочее) попадёт именно сюда.

Конец формы

## Задание для самостоятельной работы

Настало время попрактиковаться! Откройте командную строку и введите эти команды.

Скопировать код

BASH

pwd

cd ~

pwd

Если консоль открылась в домашней директории, оба вызова pwd выведут одинаковый путь.

Отлично! Теперь вы знаете, зачем нужна командная строка. Коротко подытожим:

* Консольные и графические интерфейсы — два способа взаимодействия с программами.
* Консоль, терминал, или командная строка, — это программа, которая считывает команду пользователя и выполняет её.
* Чтобы вывести текущую рабочую директорию, можно использовать команду pwd.
* У большинства пользователей компьютера есть доступ к домашней директории. Чтобы к ней перейти, используют команду cd ~.

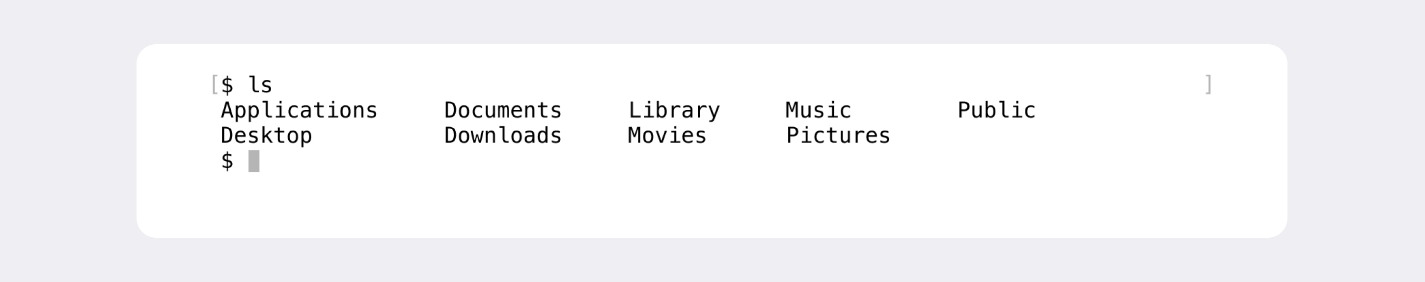
# Навигация в командной строке

Пришло время узнать больше об основных командах для работы с консолью. Начнём с навигации — мы уже затронули эту тему, когда говорили о домашней директории. Теперь подробнее остановимся на файловой системе и разберём, как перемещаться в ней.

### Вывести содержимое директории — ls

Файловая система компьютера состоит из папок, или директорий. В них могут находиться другие папки и файлы.

Когда вы открываете папку через графический интерфейс операционной системы, вы сразу видите её содержимое. В случае с консолью для отображения файлов и папок используют команду — ls (от англ. ***l***i***s***t directory contents — «отобразить содержимое директории»).



### Сменить директорию — cd

Следующая важная команда — cd (от англ. ***c***hange ***d***irectory — «сменить директорию»). Вы познакомились с ней в прошлом уроке. Она меняет текущую рабочую директорию на ту, которая указана в качестве параметра: cd имя\_папки.

Рассмотрим пример. Структура директорий может выглядеть так.

Скопировать код

/projects

/github

/open-source-project

/practicum

/my-project

Допустим, вы находитесь в директории /projects. Если ввести команду cd github, она перенесёт вас в директорию /projects/github.

Скопировать код

BASH

$ pwd

/projects # сейчас мы здесь

$ cd github # переходим в папку github

$ pwd

/projects/github # теперь мы здесь!

Обратите внимание: если в названии папки есть пробелы, при вводе нужно использовать кавычки.

Скопировать код

BASH

$ cd "Фотографии с дня рождения"

Чтобы вернуться в **родительскую директорию** — то есть на уровень выше, — вместо названия папки нужно написать две точки: ...

Скопировать код

BASH

$ pwd

/projects/github # сейчас мы здесь

$ cd .. # переходим на уровень выше

$ pwd

/projects # мы вернулись!

Есть ещё одна команда с точкой. Чтобы обратиться к текущей директории, можно использовать .. Но это нужно довольно редко — например, для запуска скриптов и программ, которые принимают папку в качестве параметра.

Скопировать код

BASH

$ pwd

/projects/github # сейчас мы здесь

$ cd . # переходим в текущую директорию

$ pwd

/projects/github # ничего не поменялось

Также cd позволяет перемещаться сразу через несколько директорий. Для этого нужно разделить их названия знаком /.

Скопировать код

BASH

$ pwd

/projects # сейчас мы здесь

$ cd github/open-source-project # переходим через несколько директорий

$ pwd

/projects/github/open-source-project # переместились сразу в папку open-source-project внутри github

Начало формы

Как посмотреть содержимое папки /с/Users/newUser/pictures/photo, если текущая директория /c/Users/newUser?



Скопировать код

BASH

cd pictures/photo

pwd

Эта последовательность команд выведет не содержимое папки, а её имя — /c/Users/newUser/pictures/photo.

Правильный ответ

Скопировать код

BASH

cd /c/Users/newUser/pictures/photo

ls

Эта последовательность команд покажет то, что нужно!



Скопировать код

BASH

cd photo

ls

Сперва нужно перейти в директорию pictures и только потом в photo. Или прописать весь путь до папки — cd pictures/photo.



Скопировать код

BASH

cd /c/Users/newUser/pictures/photo

pwd

Эта последовательность выведет имя директории — /c/Users/newUser/pictures/photo, а не её содержимое.

Тоже правильный ответ

Скопировать код

BASH

cd ..

cd newUser/pictures/photo

ls

Такая последовательность команд сработает.

Конец формы

### Дополнительные возможности ls

У многих команд консоли есть дополнительные опции. Например, если вы вызовете ls, то увидите список обычных файлов в директории. Но можно вызвать ls с флагом -a и вывести расширенный список. В нём отобразятся все скрытые файлы, которые начинаются с символа . (например, файлы конфигурации). В том числе два особых файла . и .., которые обозначают текущую и родительскую директории.

Скопировать код

BASH

$ ls # вывели список файлов

file.txt

photo.png

$ ls -a # вывели список, в котором отображаются скрытые файлы ., .. и .git

.

..

.git

file.txt

photo.png

А ещё, как и другие команды, ls может работать с символом домашней директории (~) и предыдущей директории (..). Например, ls ~ выведет содержимое домашней директории вне зависимости от того, что показывает pwd. А ls .. покажет содержимое родительской директории.

Подберите команду к описанию.

Сменить текущую директорию

cd

Название нужной команды образовано от **c**hange **d**irectory.

Вывести на экран текущую рабочую директорию

pwd

Имя описанной команды происходит от **p**rint **w**orking **d**irectory.

Напечатать содержимое директории со скрытыми файлами

ls -a

Название этой команды образовано от **l**i**s**t directory contents. Также понадобится флаг.

Начать заново

## Задание для самостоятельной работы

1. Откройте консоль и введите команду pwd.
2. Создайте в текущей директории текстовый файл test.txt с помощью привычного графического интерфейса (правая кнопка мыши → выпадающее меню).
3. Снова откройте консоль и введите ls.

Если всё сделано правильно, вы увидите файл test.txt в выводе.

💡 Не удаляйте файл test.txt после выполнения задания! Он понадобится в следующих уроках.

Замечательно! Через консоль вы можете выполнять те же операции, что и с помощью классических средств работы с операционной системой. А именно:

* менять директории командой cd;
* выводить содержимое директорий с помощью ls;
* просматривать содержимое вместе со скрытыми файлами и папками через ls -a.

# Операции с папками и файлами: создание, копирование, перемещение

Конечная цель перемещения по папкам — получить доступ к данным в них. В этом и следующем уроках покажем, как работать с файлами и директориями в консоли.

### Создание файлов и директорий — touch, mkdir

Чтобы создать файл, нужно ввести в консоль команду touch (англ. «коснуться») с именем файла в качестве параметра: touch %ИМЯ\_ФАЙЛА%.

Скопировать код

BASH

$ touch my-new-file.txt # создали файл my-new-file.txt

Хорошей практикой при создании файла считается указывать его расширение (в примере — .txt). Это позволит операционной системе выбрать подходящую программу, чтобы открыть файл. А ещё поможет другому человеку понять, какое содержимое находится внутри.

Для создания директорий через терминал используют другую команду — mkdir (от англ. ***m***a***k***e ***dir***ectory — «создать директорию»).

Скопировать код

BASH

$ mkdir new-dir # создали директорию new-dir

Можно создать целую структуру директорий одной командой с помощью флага -p.

Скопировать код

BASH

$ mkdir -p dir1/dir-inside/dir-deeper-inside

# создали папку dir-deeper-inside в папке dir-inside, которая находится в папке dir1

По умолчанию touch и mkdir создают файлы и папки в текущей рабочей директории. Например, если вы находитесь в директории abs, команда touch file.txt создаст файл именно там: abs/file.txt.

Также можно использовать обе команды вместе с символом домашней директории (~) или родительской директории (..). Например, команда mkdir ~/my-git-projects создаст папку my-git-projects внутри домашней директории.

А команда touch ../../file.txt создаст файл file.txt на две папки выше по иерархии. Допустим, если вы находитесь в директории projects/git/hello, команда touch ../../file.txt создаст файл по такому пути: projects/file.txt.

### Копирование файлов — cp

Вы наверняка копировали файлы на своём компьютере через графический интерфейс — диспетчер файлов на Windows или Finder на macOS. Посмотрим, как это делать через консоль.

Для копирования файлов через терминал существует команда cp (от англ. ***c***o***p***y — «копировать»). В простом виде cp принимает два параметра: что копируем и куда копируем.

Скопировать код

BASH

$ cp что\_копируем куда\_копируем

$ cp index.html src/

# скопировали index.html в папку src

Но можно указать сразу несколько файлов.

Скопировать код

BASH

$ cp что\_копируем что\_копируем что\_копируем куда\_копируем

$ cp index.html style.css script.js src/

# скопировали три файла (index.html, style.css и script.js) в папку src

Попробуйте вместе с нами! Откройте консоль и создайте папку first-project где угодно на компьютере. Внутри папки создайте два файла: data.txt и table.csv.

Скопировать код

BASH

$ mkdir first-project

$ touch first-project/data.txt first-project/table.csv

Эта запись равноценна поочерёдному вызову следующих команд.

Скопировать код

BASH

$ mkdir first-project

# создали папку

$ touch first-project/data.txt

# создали первый файл

$ touch first-project/table.csv

# создали второй файл

Теперь скопируйте файл data.txt в домашнюю директорию.

Скопировать код

BASH

$ cd first-project

# перешли в директорию

$ cp data.txt ~

# скопировали файл в домашнюю директорию

Перейдите в домашнюю директорию ~ и посмотрите её содержимое.

Скопировать код

BASH

$ cd ~ # перешли в домашнюю директорию

$ pwd # посмотрели, где мы

/Users/%USER\_NAME%

$ ls # файл скопирован, ура!

data.txt

<...>

### Перемещение файлов и папок — mv

Копирование создаёт копию файла или папки. Но иногда вместо копии нужно удалить файл в одном месте и создать в другом. Для этого есть команда mv (от англ. ***m***o***v***e — «переместить»).

Синтаксис команды mv аналогичен синтаксису cp. После имени команды указывают список файлов и папок, которые нужно переместить, а затем — папку, в которую нужно выполнить перемещение.

Создайте папку very-important-files внутри директории first-project. Перейдите в first-project командой cd.

Затем переместите файл table.csv в папку very-important-files и проверьте результат.

Скопировать код

BASH

$ mv table.csv ./very-important-files

# сначала указываем имя файла, который хотим переместить, потом путь — куда перемещаем

$ cd very-important-files

$ ls

table.csv

# перешли в папку very-important-files и проверили, что всё сработало

Подберите к каждой ситуации правильную команду.

Скопировать файлы file.txt и me.png в директорию important-files внутри домашней директории

cp file.txt me.png ~/important-files

Здесь нужна команда, название которой образовано от ***c***o***p***y.

Переместить файлы file.txt из родительской директории и me.png из текущей директории в папку important-files в рабочей директории

mv ../file.txt me.png important-files

Здесь поможет команда, чьё название образовано от ***m***o***v***e.

Создать файл file.txt в директории important-files, которая находится на один уровень выше текущей рабочей директории, и файл me.png в текущей рабочей директории

touch ../important-files/file.txt me.png

В переводе с английского название команды означает «коснуться».

Начать заново

Отлично! Ещё один урок позади. Подведём итоги:

* Команда touch создаёт файл, а команда mkdir — директорию.
* С помощью флага -p можно создать целую структуру директорий одной командой: mkdir -p.
* Для копирования файлов используют команду cp, для перемещения — mv.

# Операции с папками и файлами: чтение и удаление

В этом уроке продолжим рассказывать о том, как работать с файлами и папками с помощью командной строки.

### Чтение файлов — cat

Чтобы прочитать файл, в консоль нужно ввести cat (от англ. con***cat***enate and print — «объединить и распечатать») вместе с именем файла. Команда распечатает то, что содержится в нём.

Скопировать код

BASH

$ cat myfile.txt # распечатали содержимое файла myfile.txt

file-content-1

file-content-2

Команда cat работает только с текстовыми файлами. Вывести этой командой файл другого типа (например, изображение) не получится.

### Удаление файлов и папок — rm, rmdir, rm -r

Чтобы удалить файл, нужно напечатать команду rm (от англ. ***r***e***m***ove — «удалить») и передать ей имя файла.

Скопировать код

BASH

$ rm example.txt # удалили файл example.txt из текущей папки

Удалить папку можно командой rmdir (от англ. ***r***e***m***ove ***dir***ectory — «удалить директорию»). Не забудьте указать имя папки.

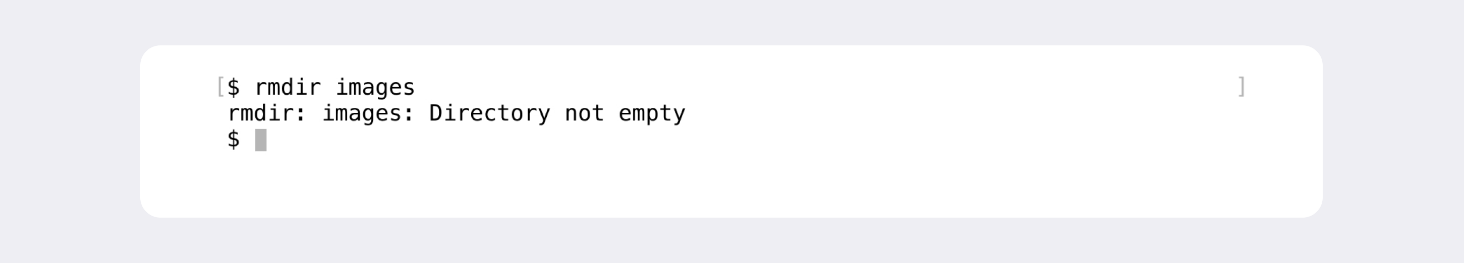
Скопировать код

BASH

$ rmdir images # команда удалит папку images из текущей директории,

# если папка images пуста

Если в папке, которую вы пытаетесь стереть, есть какие-то файлы, то командная строка не удалит её и выведет сообщение о том, что папка не пуста (англ. Directory not empty).



Это защита от случайного удаления нужных файлов. Если папку всё-таки нужно удалить вместе со всем её содержимым, можно использовать команду rm так.

Скопировать код

BASH

$ rm -r images # удалили папку images со всем её содержимым из текущей директории

В этом случае команда rm -r (-r — от англ. ***r***ecursive, «рекурсивный») **рекурсивно** удаляет файлы и папки. Это значит, что удаление будет последовательно применяться к каждому из элементов в этой папке — пока не сотрёт их все. Затем команда удалит пустую директорию.

Например, есть папка "ФОТО", внутри которой — файлы и папка "Фотографии с дня рождения". Если вызвать команду rm -r для "ФОТО", то сначала будут удалены все файлы и папки внутри неё (в том числе папка "Фотографии с дня рождения"), а после — сама директория "ФОТО".

💡 Будьте осторожны: удаление объектов командами rm и rmdir необратимо — в этом случае файлы и папки не попадают в корзину и исчезают навсегда.

Сопоставьте задачу с подходящей для неё командой.

Прочитать файл the-best-file-ever.txt

cat the-best-file-ever.txt

Запомнить эту команду можно по её «кошачьему» названию.

Удалить файл the-best-file-ever.txt

rm the-best-file-ever.txt

Название этой команды происходит от ***r***e***m***ove.

Удалить папку screenshots

rmdir screenshots

Название этой команды складывается из слов ***r***e***m***ove ***dir***ectory.

Удалить папку screenshots и всё, что она содержит

rm -r screenshots

Флаг этой команды происходит от слова ***r***ecursive.

Начать заново

## Задание для самостоятельной работы

1. В уроке о навигации вы создавали файл test.txt. Откройте его в любом текстовом редакторе и введите несколько символов или слов. Затем сохраните файл.
2. Откройте консоль и распечатайте содержимое файла test.txt командой cat test.txt.
3. Сделайте несколько вызовов mkdir и touch для создания файлов и папок с любыми названиями. Для проверки результата используйте команду ls.
4. Удалите все созданные на предыдущем шаге файлы и директории командами rm, rmdir и rm -r.

Так держать! Вот о чём мы рассказали в этом уроке:

* Вывести содержимое файла можно командой cat.
* Для удаления файла используют rm, для удаления пустой директории — rmdir, а для директории с файлами — rm -r.
* Все команды удаления стирают данные безвозвратно — их нельзя будет восстановить из корзины!

# Эффективная работа с командной строкой

Вы познакомились с командной строкой и узнали, как с её помощью можно перемещаться между директориями и управлять папками и файлами. Поначалу вы будете работать в строке медленнее, чем вам хотелось бы, — и это нормально. Дайте себе время привыкнуть к новому интерфейсу, освоиться с базовыми командами. После некоторой практики вы научитесь взаимодействовать с консолью гораздо быстрее.

В этом уроке собрали полезные советы и приёмы, которые помогут ускорить вашу работу с командной строкой.

### Выполняйте сразу несколько команд

Команды в терминале необязательно вбивать и выполнять по очереди. Их можно указывать не по одной, а сразу списком. Для этого их нужно разделить двумя амперсандами (&&).

Скопировать код

BASH

$ mkdir second-project && cd second-project && touch index.html style.css

# создаём папку second-project,

# переходим в папку second-project

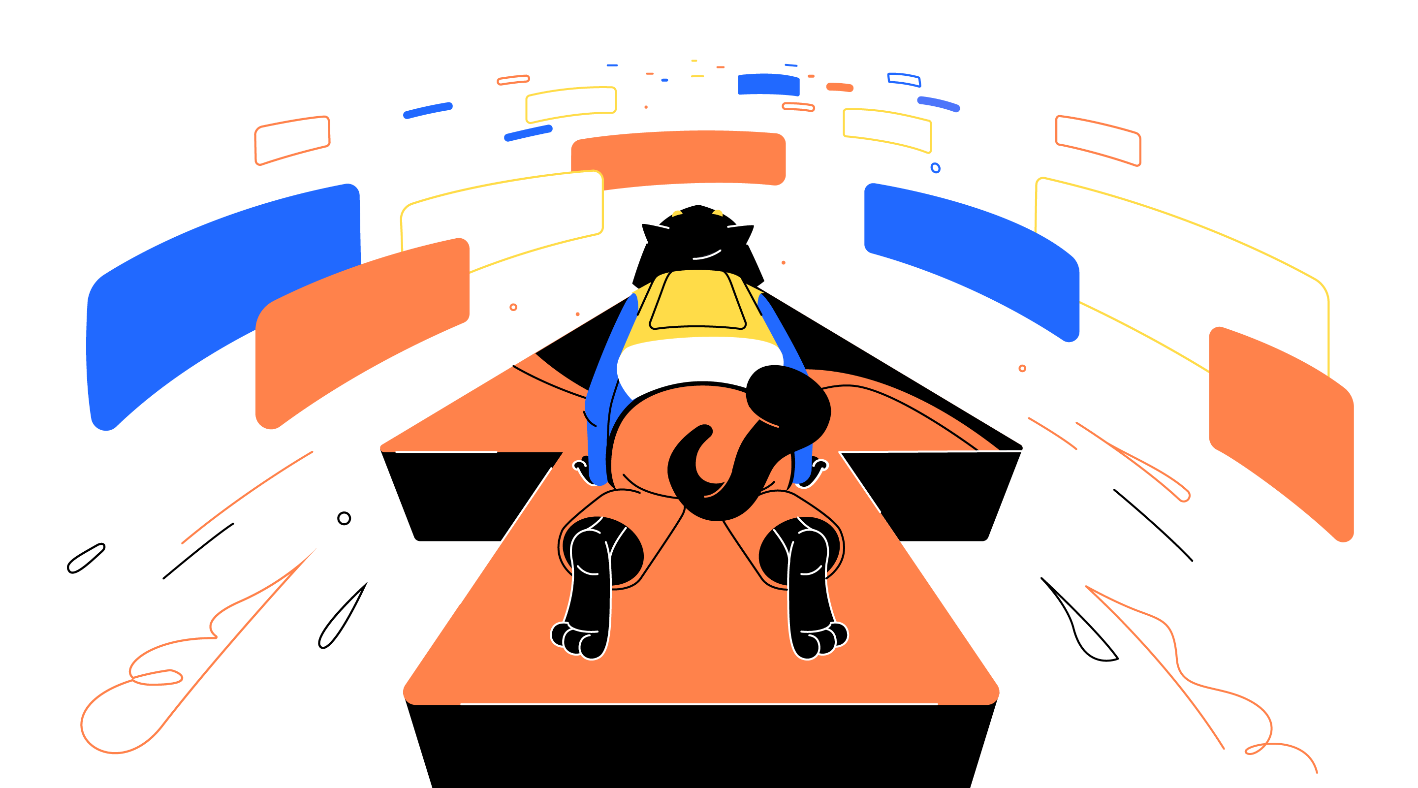
# и создаём в ней два файла: index.html и style.css

Это особенно удобно, когда нужно часто выполнять одну и ту же последовательность команд.

### Вызывайте команды из буфера

Допустим, вчера вы создали пять новых файлов, а сегодня решили добавить к ним ещё один, но не можете вспомнить название нужной команды. На этот случай у терминала есть собственная память. Она называется **буфером** (от англ. buffer — «посредник»). В буфере хранятся все команды, которые вызывались до этого. По их списку можно перемещаться.

Чтобы обратиться к последней введённой команде, нажмите на клавиатуре стрелку вверх (**↑**). Если нажать ещё раз, появится предпоследняя команда; ещё раз — предпредпоследняя; и так далее. Чтобы вернуться — например, от предпоследней команды к последней, — нажмите стрелку вниз (**↓**).



### Используйте автозаполнение

Необязательно заучивать все команды наизусть. Если нужно найти какую-нибудь из них, достаточно вспомнить, с каких букв она начинается. Можно набрать их в командной строке и дважды нажать клавишу Tab. Терминал покажет список всех команд, которые начинаются с этих символов.

Tab автоматически дописывает не только команды, но и пути. Начните печатать имя папки или файла (они должны быть в той же директории) и нажмите Tab. Терминал заполнит имя автоматически.

Если этого не происходит, значит, есть несколько файлов или папок, которые начинаются так же. Нажмите Tab ещё раз, и вы увидите их список. Терминал не знает, как ему дозаполнить такой ввод и что именно выбрать, поэтому показывает все варианты, чтобы вы могли уточнить запрос.

Рассмотрим пример. Перейдём в домашнюю директорию /Users/Username с помощью автозаполнения Tab.

Скопировать код

BASH

$ cd /Users/ # перешли в папку Users

$ cd U[Tab] # ввели первую букву имени пользователя и нажали Tab

# имя папки Username подставится автоматически

$ pwd # теперь проверим, где мы сейчас находимся

/Users/Username # мы в папке Username!

Вместо того чтобы печатать полное имя папки — Username, мы набрали только его первую букву — U, нажали клавишу Tab, и консоль допечатала всё за нас.

Есть ещё один способ использовать Tab при навигации в другую директорию. Если ввести cd с названием папки, а затем нажать Tab, в консоль в качестве подсказки выведутся все возможные пути.

Скопировать код

BASH

$ cd ~/[Tab] # вывели список директорий, чтобы понять, куда переходить

Applications/ Downloads/ Library/ Parallels/ Public/ diagrams/ memes/ python/

Desktop/ Dropbox/ Movies/ Pictures/ bin/ docs/ papers/ tmp/

Documents/ Exercism/ Music/ Postman/ books/ go/ projects/

Если вывод будет слишком большой, консоль спросит, нужно ли показать все возможные варианты.

Скопировать код

BASH

$ cd ~/[Tab]

zsh: do you wish to see all 426 possibilities (429 lines)? # точно хотите увидеть все 426 варианта (429 линий)?

Чтобы подтвердить вывод, нужно нажать y, а чтобы отменить автодополнение — n.

### Применяйте команды для быстрой навигации

Напомним основные:

* pwd — проверить, где мы находимся;
* ls — посмотреть список файлов/папок в директории;
* cd — перейти в выбранную папку.

С помощью этих команд можно быстро перемещаться между каталогами и изучать их содержимое.

А ещё можно почти мгновенно перемещаться в ключевые папки. Допустим, вы хотите увидеть содержимое **корневой директории** (англ. root directory). Это верхняя в иерархии папка, в которой хранится всё, что есть на вашем жёстком диске. Дальнейшие действия зависят от типа операционной системы. Чтобы сделать это на macOS или Linux, напечатайте в командной строке слеш (/) и дважды нажмите Tab.

Скопировать код

BASH

/

Applications/ Network/ Library/

System/ Volumes/ Users/

# содержимое корневой директории macOS

Готово! В терминале напечаталось содержимое корневой директории.

В эту директорию тоже можно быстро перемещаться — с помощью команды cd / + Enter (если у вас macOS или Linux).

Скопировать код

BASH

$ cd / # перемещает в корневую директорию

Чтобы попасть в корневую директорию Windows, нужно перейти на соответствующий диск. Например, cd c:/ + Enter или cd /c + Enter.

Скопировать код

BASH

$ cd c:/ # переместились в корневую директорию

$ ls

Documents and Settings/ Windows/

Program Files/ Users/

Program Files (x86)/

# содержимое корневой директории Windows

💡 **Почему эти действия так различаются для Windows и для macOS и Linux?**

macOS и Linux основаны на операционной системе UNIX. А Windows — отдельная ОС. Поэтому последовательности команд для выполнения одних и тех же действий в этих операционных системах могут различаться.

В домашнюю директорию можно попасть так же быстро. Вы уже знаете, как это сделать. Символ «тильда» (~) по умолчанию хранит ссылку на домашнюю директорию. Поэтому, чтобы переместиться в неё, достаточно напечатать ~ и нажать Enter.

Скопировать код

BASH

$ cd ~

$ pwd

/Users/Username

$ cd ~/Documents # папка Documents хранится в домашней директории

$ pwd

/Users/Username/Documents

# Установка Git

Вы изучили базовые команды для работы с командной строкой. Пора установить Git! Следуйте нашим инструкциям в зависимости от типа вашей операционной системы.

### Windows

Если вы пользователь Windows, то Git у вас уже есть. Вы установили его в составе пакета Git for Windows вместе с командной строкой.

Убедитесь в этом. Откройте консоль и выполните эту команду.

Скопировать код

BASH

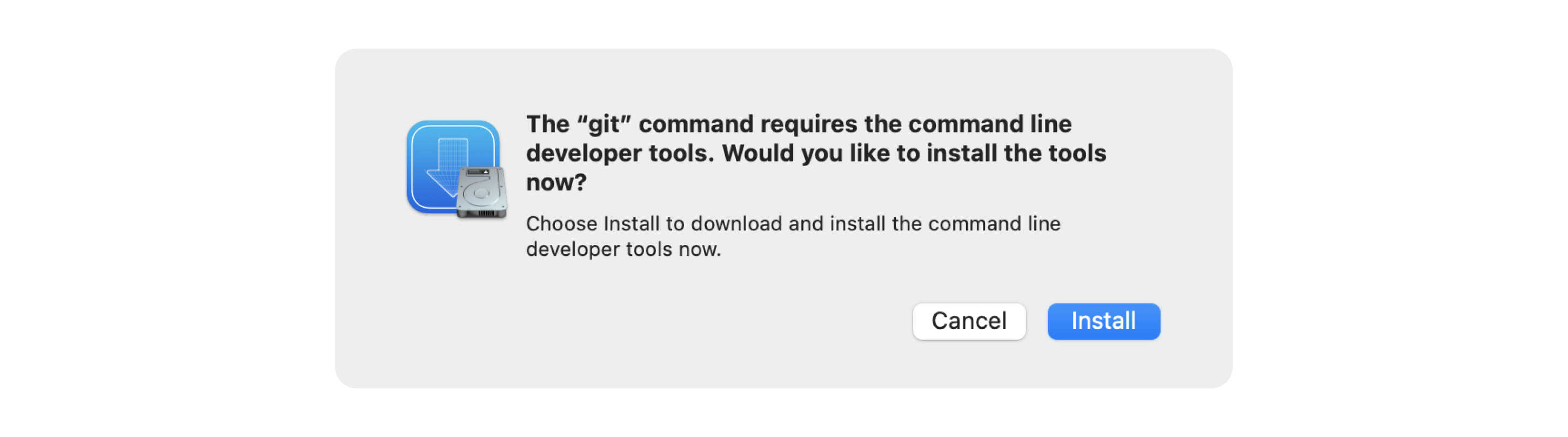
$ git version

Если Git установлен правильно, консоль выведет его текущую версию.

### macOS

Для установки Git на macOS существует два способа.

**Первый способ.** Откройте консоль и выполните команду /usr/bin/git. Она запустит установщик. Нажмите **Install** (англ. «установить») и дождитесь окончания установки.



Когда установка завершится, для проверки выполните эту команду.

Скопировать код

BASH

$ git version

Если на экран выводится текущая версия Git, значит, установка прошла успешно.

**Второй способ.** Используйте **Homebrew**.

1. Установите менеджер пакетов Homebrew:
   * Перейдите [на официальный сайт Homebrew](https://brew.sh/).
   * Скопируйте команду для установки — справа от неё есть символ для копирования. Нажмите на него, чтобы команда попала в буфер обмена.



* + Найдите программу Terminal в поиске Spotlight или в списке программ. Вставьте скопированный текст в окно терминала и нажмите Enter.

1. Установите Git с помощью Homebrew. Скопируйте и введите в терминал следующую команду.

Скопировать код

BASH

$ brew install git

1. Проверьте установку. Для этого откройте терминал и введите эту команду.

Скопировать код

BASH

$ git version

Если на экран выводится текущая версия Git, значит, установка прошла успешно.

### Linux

Для установки Git на Linux нужно использовать терминал. Найдите программу Terminal в поиске или в списке программ. Перейдите [на официальный сайт Git](https://git-scm.com/download/linux) и выберите команду установки для своей версии Linux. Скопируйте её в программу Terminal и нажмите Enter.

После успешной установки введите команду для проверки.

Скопировать код

BASH

$ git version

Если вы видите в консоли текущую версию Git, всё прошло успешно.

Поздравляем, Git установлен! Осталось произвести некоторые настройки, и можно начинать работать с ним.

# Настройка Git

Для настройки Git можно использовать командную строку — теперь вы умеете работать с ней. Если у вас macOS или Linux, запустите программу Terminal. Если Windows — Git Bash.

### Работа с файлом настройки .gitconfig

Сейчас вы работаете в одиночку, но в дальнейшем вам может понадобиться использовать Git в команде. Чтобы участникам проекта было понятно, кто и какие изменения вносил, нужно представиться и указать имя пользователя и адрес электронной почты.

Вы можете указать любую электронную почту и любое имя. Сделать это можно с помощью команды git config (от англ. configuration — «конфигурация», «настройка») с ключом --global (англ. «глобальный»). При этом не имеет значения, в какой директории вы находитесь прямо сейчас: вызов git config --global сработает везде.

В качестве значения user.name нужно указать своё имя или никнейм. Для настройки параметра user.email указывают электронную почту.

Скопировать код

BASH

$ git config --global user.name "User Namovich"

# имя или ник нужно написать латиницей и в кавычках

$ git config --global user.email username@yandex.ru

# здесь нужно указать свой настоящий email

Все глобальные настройки Git хранит в файле .gitconfig в домашней директории. Команда запишет в этот файл указанные имя и почту. Чтобы убедиться в этом, можно вызвать команду для чтения файлов.

Скопировать код

BASH

$ cat ~/.gitconfig

Другой способ проверки — вывести содержимое файла конфигурации Git той же командой git config с флагом --list (англ. «список»).

Скопировать код

BASH

$ git config --list

В ответ командная строка покажет текущие значения настроек.

Скопировать код

BASH

user.name=Username

user.email=username@yandex.ru

Готово!

## Задание для самостоятельной работы

1. Настройте Git командами, которые мы показали в этом уроке.

Скопировать код

BASH

$ git config --global user.name "ваше имя или ник латиницей"

$ git config --global user.email ваша электронная почта

1. Убедитесь, что данные сохранились, с помощью одной из двух команд.

Скопировать код

BASH

$ cat ~/.gitconfig

$ git config --list

Поздравляем! Вы изучили базовые команды консоли, скачали Git и настроили основные параметры. Осталось только подключить Git к проекту и начать им пользоваться! Как это сделать — расскажем дальше в курсе.

# Шпаргалка. Базовые команды в консоли

Чтобы вам было удобнее взаимодействовать с командной строкой, мы подготовили шпаргалку. В ней собраны все команды, о которых мы рассказали в уроках, и их полезные вариации.

### Навигация

* pwd (от англ. ***p***rint ***w***orking ***d***irectory, «показать рабочую папку») — покажи, в какой я папке;
* ls (от англ. ***l***i***s***t directory contents, «отобразить содержимое директории») — покажи файлы и папки в текущей папке;
* ls -a — покажи также скрытые файлы и папки, названия которых начинаются с символа .;
* cd first-project (от англ. ***c***hange ***d***irectory, «сменить директорию») — перейди в папку first-project;
* cd first-project/html — перейди в папку html, которая находится в папке first-project;
* cd .. — перейди на уровень выше, в родительскую папку;
* cd ~ — перейди в домашнюю директорию (/Users/Username);
* cd / — перейди в корневую директорию.

### Работа с файлами и папками

**Создание**

* touch index.html (англ. touch, «коснуться») — создай файл index.html в текущей папке;
* touch index.html style.css script.js — если нужно создать сразу несколько файлов, можно напечатать их имена в одну строку через пробел;
* mkdir second-project (от англ. ***m***a***k***e ***dir***ectory, «создать директорию») — создай папку с именем second-project в текущей папке.

**Копирование и перемещение**

* cp file.txt ~/my-dir (от англ. ***c***o***p***y, «копировать») — скопируй файл в другое место;
* mv file.txt ~/my-dir (от англ. ***m***o***v***e, «переместить») — перемести файл или папку в другое место.

**Чтение**

* cat file.txt (от англ. con***cat***enate and print, «объединить и распечатать») — распечатай содержимое текстового файла file.txt.

**Удаление**

* rm about.html (от англ. ***r***e***m***ove, «удалить») — удали файл about.html;
* rmdir images (от англ. ***r***e***m***ove ***dir***ectory, «удалить директорию») — удали папку images;
* rm -r second-project (от англ. ***r***e***m***ove, «удалить» + ***r***ecursive, «рекурсивный») — удали папку second-project и всё, что она содержит.

### Полезные возможности

* Команды необязательно печатать и выполнять по очереди. Можно указать их списком — разделить двумя амперсандами (&&).
* У консоли есть собственная память — буфер с несколькими последними командами. По ним можно перемещаться с помощью клавиш со стрелками вверх (**↑**) и вниз (**↓**).
* Чтобы не вводить название файла или папки полностью, можно набрать первые символы имени и дважды нажать Tab. Если файл или папка есть в текущей директории, командная строка допишет путь сама.

Например, вы находитесь в папке dev. Начните вводить cd first и дважды нажмите Tab. Если папка first-project есть внутри dev, командная строка автоматически подставит её имя. Останется только нажать Enter.