

*School of
Computer
Science*

СИСТЕМЫ КОНТРОЛЯ ВЕРСИЙ

ПРОГРАММИРОВАНИЕ НА PYTHON
Лекции для ИТ-школы



Как хранить исходные тексты?

Существование без систем контроля версий

- Нет резервных копий
- Копии файла
 - lab.cpp
 - lab_new.cpp
 - lab_newnew.cpp
 - lab_final.cpp
 - lab_final_v2.cpp
 - ...
- Копии папки/архива



Свойства системы контроля версий

Система контроля версий

- Выглядит как одна копия всего проекта
- Хранит все версии, начиная с самой 1-ой
- Возможность групповой работы
- История изменений для каждого файла
- Просмотр изменений между двумя версиями файла



Различные системы контроля версий

Системы контроля версий

- RCS
- CVS
- Subversion (SVN)
- Git
- Mercurial
- ...



История изменений в системе контроля версий

D:\Work\trunk\com.tranzaxis\ads - Log Messages - TortoiseSVN

Filter by Messages, Paths, Authors, Revisions, Bug-IDs, Date, Date Range

From: 02.10.2019 To: 07.10.2019

Revision	Actions	Author	Date	Message
223598	🕒	avostrikov	7 октября 2019 г. 11:16:13	[#] (TXI-252) Interfacing.Nspk.Ips. Исправлена ошибка, из-за которой входящий интерфейс не заполнял ЭБД-20 и ЭБД-47 при
223593	🕒	sshakshin	7 октября 2019 г. 10:05:57	[#] (TXI-304) Interfacing.DInersClub. Исправлено заполнение поля 13 <3.2.12>
223592	🕒	ypalii	7 октября 2019 г. 8:59:47	[.i] .
223588	🕒	dchernienko	6 октября 2019 г. 15:23:32	[.i] (PRJDEV-3450) Subject. Temporary commit
223579	🕒	sshakshin	6 октября 2019 г. 11:19:58	[#] (TXI-60) Interfacing.Mc. Исправлено заполнение поля 23 в ответе <3.2.10>
223577	+🕒	vchernov	5 октября 2019 г. 21:12:55	[+] (TXISS-118) Perso.Icc. Поддержана персонализация Pectoral D n-VSDC f136 v1.1
223575	+🕒	vchernov	5 октября 2019 г. 21:06:47	[+] (TXISS-118) Perso.Icc. Поддержана персонализация Pectoral D n-VSDC SF232 v1.2
223573	+🕒	vchernov	5 октября 2019 г. 21:05:57	[+] (TXISS-116) Perso.Icc. Поддержана персонализация Pectoral n-VSDC f136 v1.1
223561	+🕒	vchernov	5 октября 2019 г. 20:30:04	[+] (TWRBS-27736) Interfacing. Конструктор исходящих файловых интерфейсов для текстового и XML форматов <3.2.12>
223554	+🕒	nvayner	4 октября 2019 г. 19:20:05	[+] (TWRBS-26758) Interfacing.Tip. Для входящего Extract-интерфейса добавлена поддержка кода транзакций
223552	+🕒	ipettrash	4 октября 2019 г. 19:01:38	[.i] (TXI-305) Interfacing.W4. Добавлена новая лицензия W037 для AddSCA (Strong Customer Authentication). Под лицензию пог
223548	+🕒	nvayner	4 октября 2019 г. 18:15:31	[+] (TXI-174) Interfacing.Tip. Для входящего Extract-интерфейса добавлена поддержка кодов транзакций 111, 183, 187, 336,
223545	+🕒	nvayner	4 октября 2019 г. 17:40:09	[.i] (TWRBS-26758) Interfacing.Tip. <3.2.12>
223534	+🕒	kklementyev	4 октября 2019 г. 17:24:16	[#] (TXI-167) Interchange.McIrd, Interfacing.McIpri. Исправлена ошибка при которой игнорировались имеющиеся транзакции
223532	+🕒	eshushunova	4 октября 2019 г. 17:15:04	[.i] (TWRBS-28475) Common. Разграничение фильтров по институту <3.2.12>
223531	+🕒	ashayakhmetov	4 октября 2019 г. 17:12:57	[.i] (TWRBS-30280) Acquiring.Xdc.Config.States.Ndc. Поддержка новых состояний NDC

[#] (TXI-252) Interfacing.Nspk.Ips. Исправлена ошибка, из-за которой входящий интерфейс не заполнял ЭБД-20 и ЭБД-47 при формировании сообщения-ответа с типом С04, если транзакция была отклонена <3.2.11>

Path	Action	Copy from path	Revision
/trunk/dev/trunk/com.tranzaxis/ads/Interfacing.Nspk.Ips/src/adLONOIRZ2PNFFPDTPWYI7IIIZQG4.xml	Modified		
/trunk/dev/trunk/com.tranzaxis/ads/Interfacing.Nspk.Ips/src/preview/adLONOIRZ2PNFFPDTPWYI7IIIZQG4.java	Modified		

Showing 100 revision(s), from revision 223183 to revision 223598 - 1 revision(s) selected, showing 2 changed paths

Show only affected paths Stop on copy/rename Include merged revisions

Statistics Help OK

Show All Next 100 Refresh



ОПРЕДЕЛЕНИЕ И КЛАССИФИКАЦИЯ

Система контроля версий (СКВ) — это система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлов



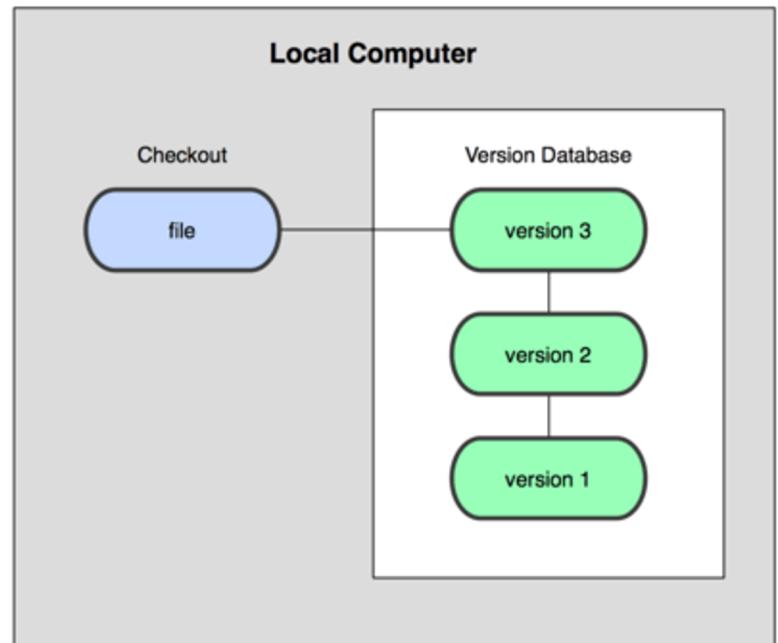


Суть: локальная база данных, в которой хранится список изменений исходного файла

Особенности:

- Работа только с одним файлом;
- Невозможность одновременной работы нескольких пользователей с системой;
- Риск потери данных.

Представители: RCS (Revision Control System)





ЦЕНТРАЛИЗОВАННЫЕ СКВ

Суть: центральный сервер, на котором хранятся все файлы под версионным контролем; клиенты получают копии файлов

Плюсы:

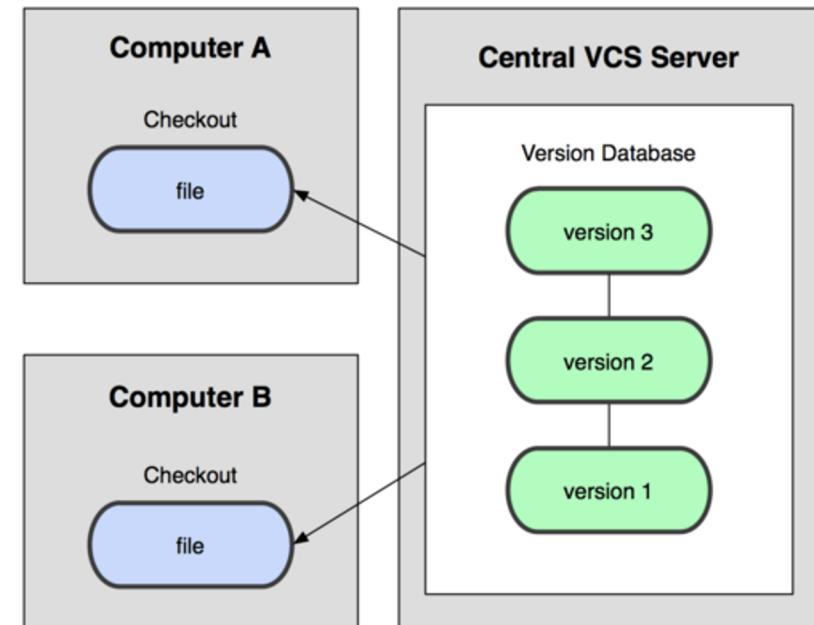
- Под контролем находятся все файлы;
- Возможность работы нескольких пользователей;
- Удобство администрирования.

Минусы:

- Риск потери данных.

Представители: CVS, SVN

В Компас Плюс используется для работы Subversion (SVN)





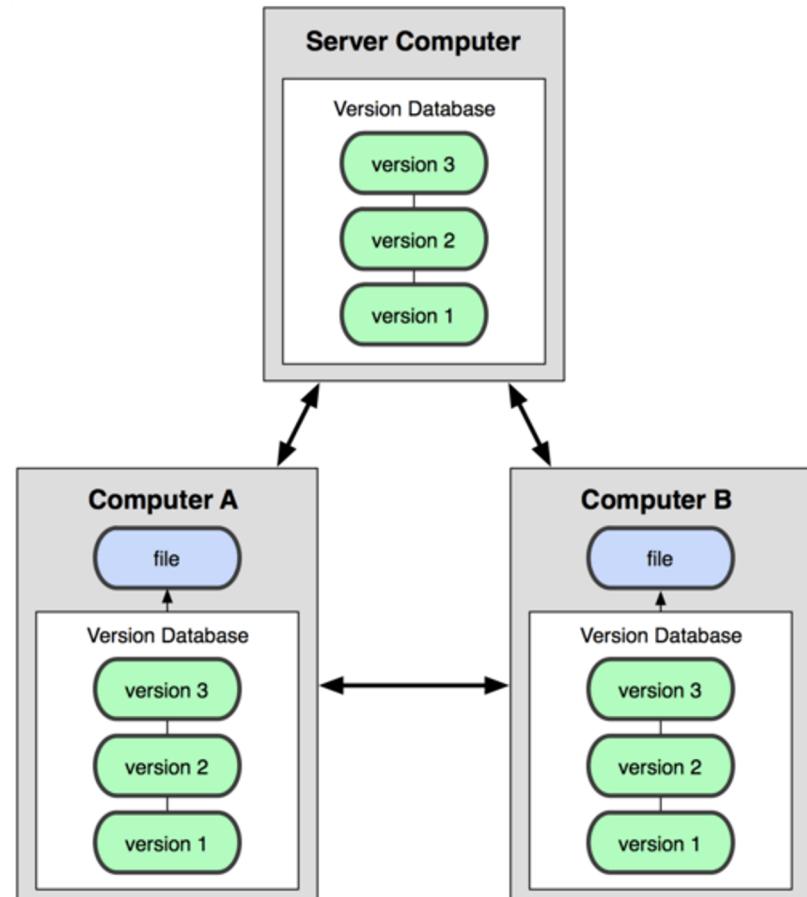
РАСПРЕДЕЛЕННЫЕ СКВ

Суть: клиенты забирают весь репозиторий, который в любой момент может быть скопирован обратно на сервер

Особенности:

- Отсутствие четко выделенного центрального хранилища версий - репозитория
- Возможность работы с несколькими удалёнными репозиториями

Представители: Git, Mercurial



В Компас Плюс используется для работы Git



Git – распределенная система контроля версий

Разработан: Линусом Торвальдсом в 2005 году

Проекты: ядро Linux, Swift, Android, jQuery, PHP, Qt, ряд дистрибутивов Linux

Сервисы для работы: GitHub, GitLab, TortoiseGit



git





Репозиторий – место, где хранятся и поддерживаются какие-либо данные

- **Локальный** – место на вашем компьютере
- **Удаленный** – хранилище в сети (например, GitHub)

Удаленный репозиторий принято называть **origin**

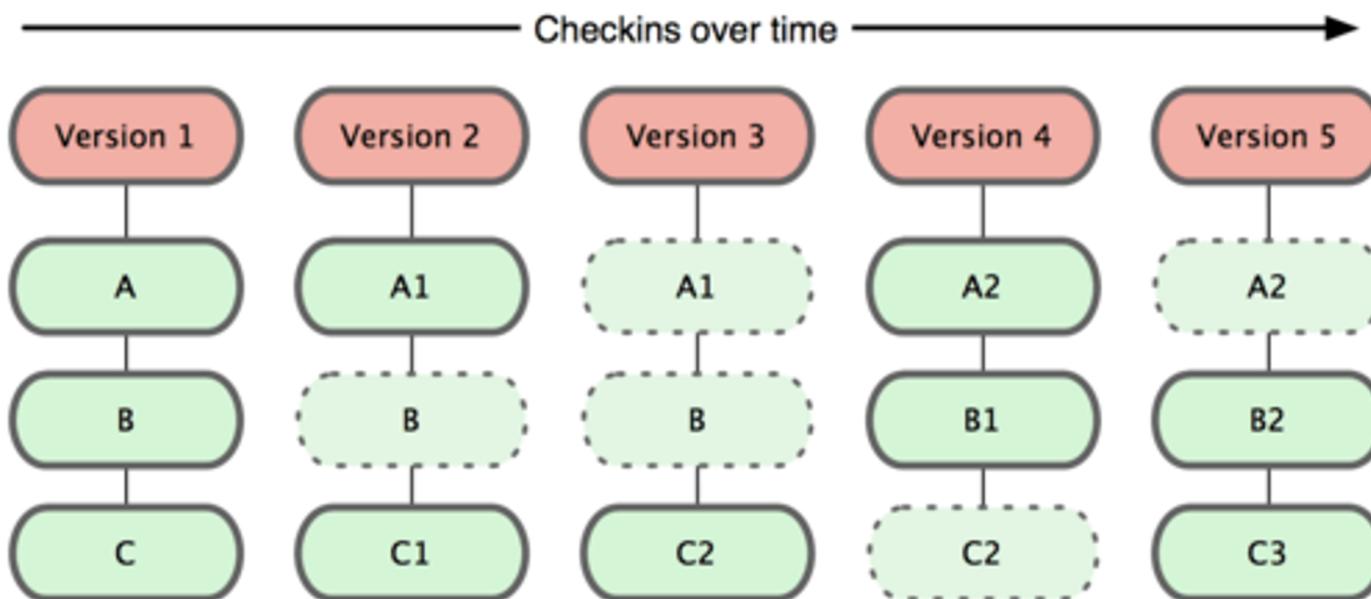


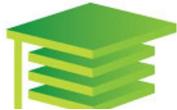
ХРАНЕНИЕ ДАННЫХ

Слепок - состояние всех файлов рабочего каталога на текущий момент времени

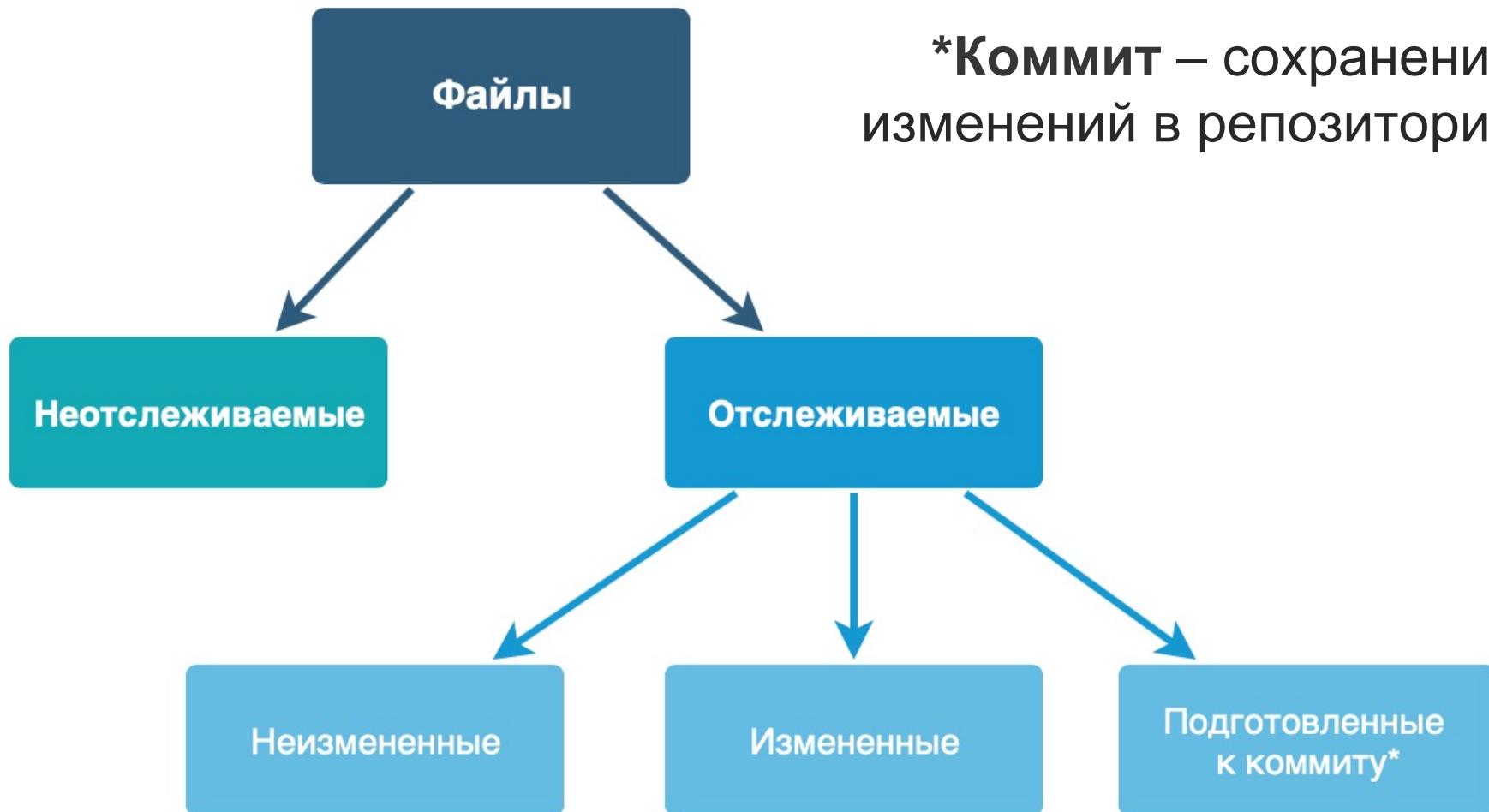
Git: хранит файлы в виде слепков

Другие СКВ: хранят список изменений для файлов





СОСТОЯНИЯ ФАЙЛОВ



Уже зафиксированные в
коммите и сохраненные в
локальном репозитории (.git)

Файлы, которые поменялись,
но еще не зафиксированы

Измененные и подготовленные
для коммита – staged for commit
(проиндексированные)

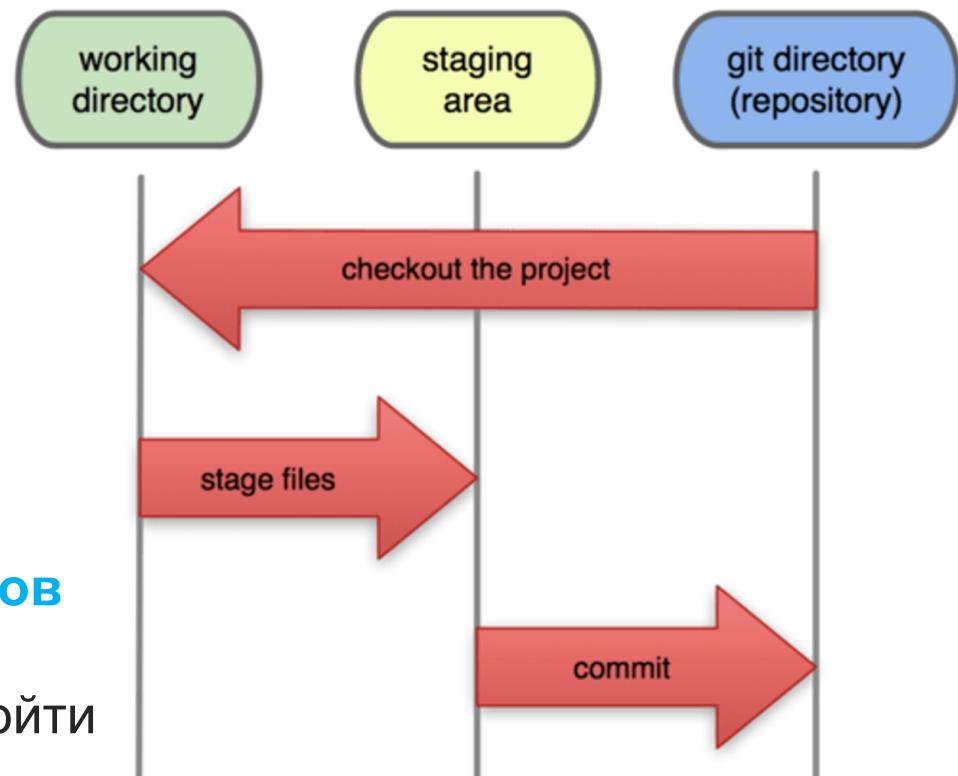


Каталог Git – место, где Git хранит метаданные и базу данных проекта, именуется “`.git`”

Рабочий каталог – копия определенной версии проекта, которая время от времени синхронизируется с удаленным сервером и содержит в себе **Каталог Git**

Область подготовленных файлов (индекс) – файл, содержащий информацию о том, что должно войти в следующий коммит

Local Operations





- Установить git:
<https://git-scm.com/downloads>
- Перейти в командную строку
- Указать данные пользователя:

```
$ git config --global user.name "UserName"
```

```
$ git config --global user.email myname@example.com
```

- Перейти в каталог проекта и создать в нем git-репозиторий:

```
$ git init
```



ОСНОВНЫЕ КОМАНДЫ

- Текущее состояние репозитория:

```
$ git status
```

- Добавить файл под версионный контроль/к коммиту
(другое название – «Проиндексировать изменения»):

```
$ git add <filename>
```

- Коммит (только для проиндексированных файлов):

```
$ git commit -m "commit message"
```

- Commit message должен быть осмысленным и четко передавать содержание коммита
- Обычно придерживаются принципа «Один завершенный цикл разработки – один коммит»



ОСНОВНЫЕ КОМАНДЫ

Удаление файла: git rm

- Удаление файла из Git и рабочего каталога:

```
$ git rm <filename>
```

- Удаление файла из Git, оставив его в рабочем каталоге:

```
$ git rm --cached <filename>
```

```
slukashenko@SLUKASHENKO-WIN MINGW64 /c/test_git (main)
$ ls
file1 file2 file3

slukashenko@SLUKASHENKO-WIN MINGW64 /c/test_git (main)
$ git rm file2
rm 'file2'

slukashenko@SLUKASHENKO-WIN MINGW64 /c/test_git (main)
$ git rm --cached file3
rm 'file3'

slukashenko@SLUKASHENKO-WIN MINGW64 /c/test_git (main)
$ ls
file1 file3

slukashenko@SLUKASHENKO-WIN MINGW64 /c/test_git (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    deleted:   file2
    deleted:   file3

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    file3
```



ОСНОВНЫЕ КОМАНДЫ

Просмотр истории коммитов: git log

- Просмотр всей истории коммитов в обратном хронологическом порядке:

```
$ git log
```

```
slukashenko@SLUKASHENKO-WIN MINGW64 /c/test_git (main)
$ git log
commit 4e2152ed6617ae79d16a573f9f97017401b1cc0f (HEAD -> main)
Author: slukashenko <slukashenko75@gmail.com>
Date:   Sat Sep 11 14:51:06 2021 +0500

    Added 3 files

commit 42b03725088eff7474605580ad6e29f7563c61f5
Author: slukashenko <slukashenko75@gmail.com>
Date:   Sat Sep 11 14:49:12 2021 +0500

    Deleted text.txt

commit ce5fe7cff9a641e43b57ef7a28d3c9846c832c68
Author: slukashenko <slukashenko75@gmail.com>
Date:   Sat Sep 11 14:35:53 2021 +0500

    Added 1st text file
```



ИГНОРИРОВАНИЕ ФАЙЛОВ

- Что игнорировать?

Автоматически генерируемые и временные файлы
(логи, кэш, результаты сборок и т.д.)

- Как?

Добавить в проект файл **.gitignore** с перечислением шаблонов игнорируемых файлов:

```
$ git add .gitignore
```

Сборник готовых **.gitignore** файлов:

<https://gist.github.com/octocat/9257657>

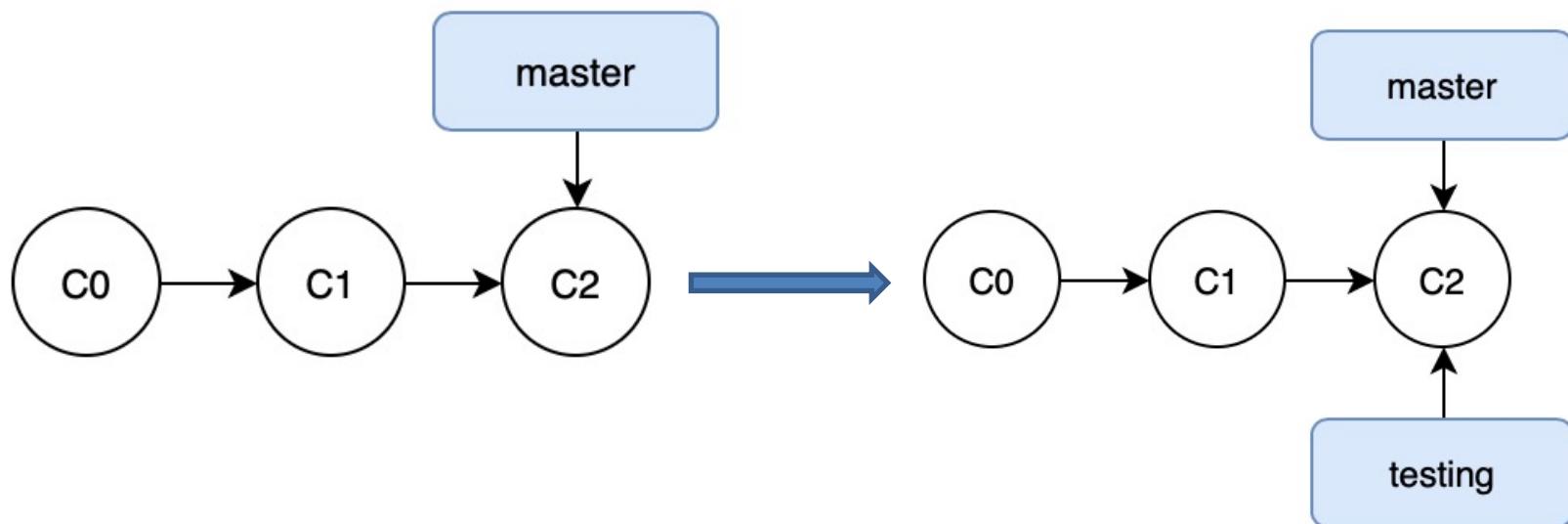
Кроме того, хостинг GitHub предлагает создать файл исключений автоматически при создании проекта на GitHub



Ветка (branch) – это подвижный указатель на один из коммитов. Ветка по умолчанию называется *master* (*main*, *trunk*, ...), т.к. *master* посчитали неполиткорректным 😊

Создание новой ветки:

```
$ git branch testing
```

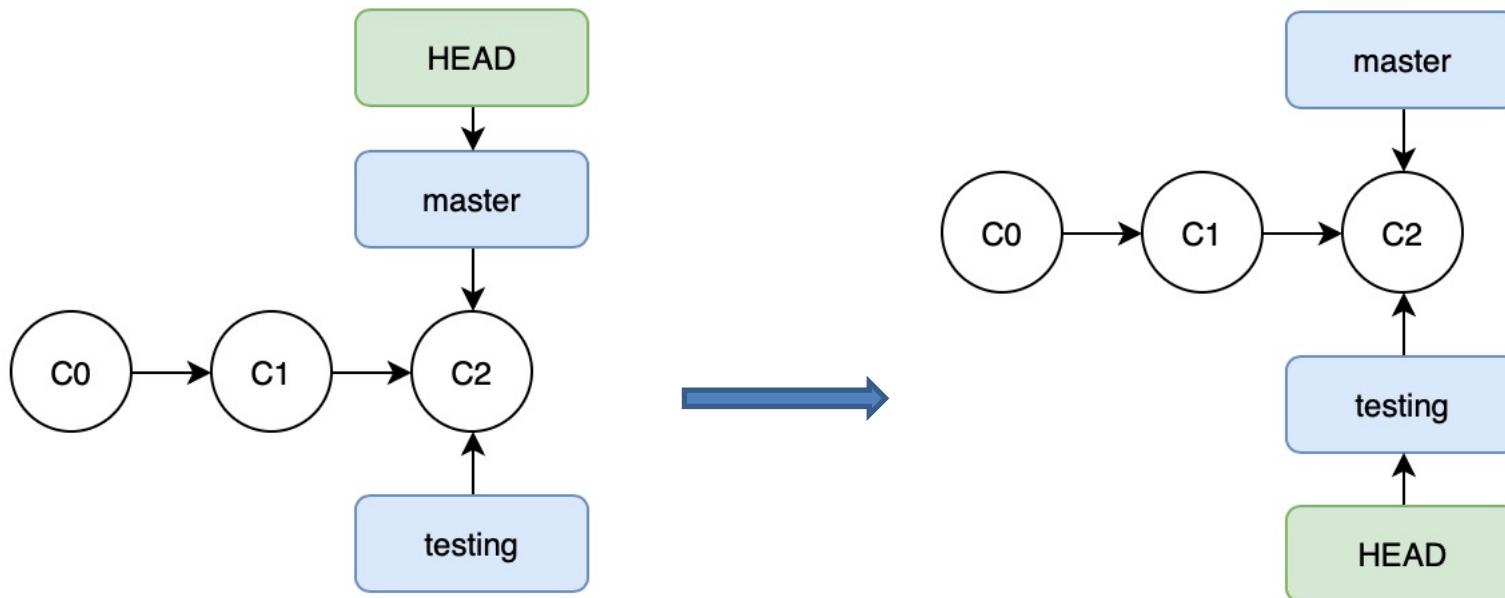




HEAD – это указатель на текущую ветку

Перейти на другую ветку:

```
$ git checkout testing
```





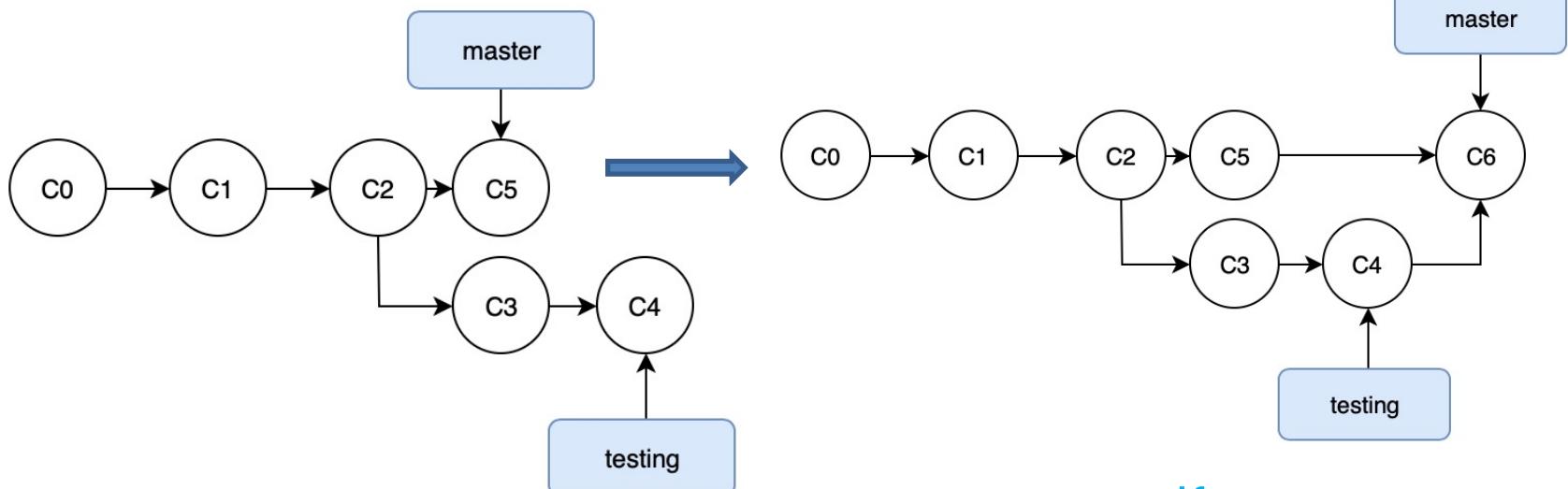
Возврат на основную ветку:

```
$ git checkout master
```

Слияние веток:

```
$ git merge testing
```

При слиянии
возможны
конфликты!



Удаление ветки:

```
$ git branch -d testing
```

Каждая задача
решается в своей
ветке



ЦЕНТРАЛЬНЫЙ РЕПОЗИТОРИЙ

Например, на хостинге GitHub или GitLab

- Связать центральный репозиторий с удаленного хоста для синхронизации с локальным репозиторием:

```
$ git remote add origin <url>
```

- Клонировать центральный репозиторий с удаленного хоста на свой компьютер:

```
$ git clone <url> [local_directory]
```

```
s1ukashenko@SLUKASHENKO-WIN MINGW64 /c
$ git clone https://github.com/ITFI-school/python-course-2021-2022-11.git course_dir
Cloning into 'course_dir'...
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 15 (delta 2), reused 3 (delta 0), pack-reused 0
Receiving objects: 100% (15/15), 1.39 MiB | 3.54 MiB/s, done.
Resolving deltas: 100% (2/2), done.

s1ukashenko@SLUKASHENKO-WIN MINGW64 /c
$ cd course_dir

s1ukashenko@SLUKASHENKO-WIN MINGW64 /c/course_dir (main)
$ ls -a
./  ../  .git/  .gitignore  'Lesson 1'/  README.md

s1ukashenko@SLUKASHENKO-WIN MINGW64 /c/course_dir (main)
```



ЦЕНТРАЛЬНЫЙ РЕПОЗИТОРИЙ

Извлечение данных

- Только получение данных:

```
$ git fetch origin [ветка]
```

- Получение и слияние данных:

```
$ git pull origin [ветка]
```

Пример:

```
$ git pull origin main
```

```
s1ukashenko@SLUKASHENKO-WIN MINGW64 /c/course_dir (main)
$ git pull origin main
From https://github.com/ITFI-school/python-course-2021-2022-11
 * branch            main      -> FETCH_HEAD
Already up to date.
```



ЦЕНТРАЛЬНЫЙ РЕПОЗИТОРИЙ

Отправка данных

- Отправка данные на удаленный хост:

```
$ git push origin[ветка]
```

```
$ git push origin main
```

Выполняется успешно если:

1. Есть права на запись в центральный (удаленный) репозиторий
2. У вас находится актуальная версия репозитория (никто не делал *git push* с момента вашей последней синхронизации: *git pull origin main*)



GITHUB И GITLAB

GitHub и **GitLab** – сервисы для онлайн-хостинга проектов, использующих Git

Цель создания: содействовать взаимодействию разработчиков

Функции:

- Разработка open-source проектов;
- Контроль доступа;
- Багтрекинг;
- Управлением задачами;
- Документация к проекту;

Страница на GitHub – лицо разработчика



ПОЛЕЗНЫЕ ССЫЛКИ

Книга “Pro Git” с переводом на русский:

<https://git-scm.com/book/ru/v2>

Интерактивный курс по использованию Git (English):

<https://githowto.com>

Шпаргалки по командам Git:

<https://eax.me/git-commands/>

<https://training.github.com/downloads/ru/github-git-cheat-sheet/>

Статья ”Git – советы новичкам”:

<https://habr.com/en/company/playrix/blog/345732/>

Видео на YouTube:

1. [Git и GitHub Курс Для Новичков](#)
2. [Git. Большой практический выпуск](#)
3. [Про системы контроля версий из курса МФТИ
"Практика программирования на Python"](#)

СПАСИБО ЗА ВНИМАНИЕ !
ВОПРОСЫ ?

