Linux and more.
Git basics.

Artem Trunov for Ozon Masters

На этом занятии

- Что такое системы контроля версий (CVS)
- Git и GitHub, Gitlab etc
- Начало работы с git
  - Создаем аккаунт на Github
  - Создаем репо
  - Клонируем репо
  - Добавляем файл, commit, push
  - Clone на другой машине (в другой директории) и работа с двумя репо.
- Больше git
  - .gitignore
  - README.md

Система контроля версий

- “Best thing since pizza”
- Система для управления изменениями в коде (тексте etc)
- Хранить все изменения (коммиты), сделанные разработчиками.
- Не обязательно отдельный продукт (например, контроль версий в Word)

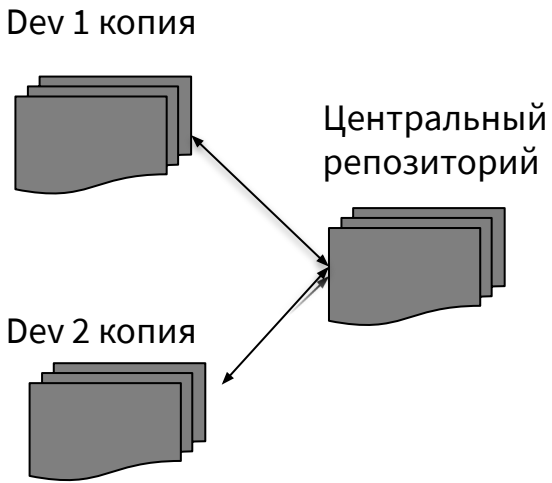
Locking vs. Merging

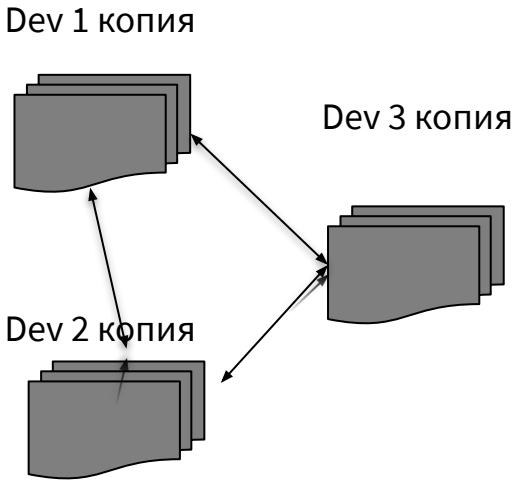
- Locking
  - “Дает” файл на редактирование только одному разработчику.
- Merging
  - Позволяет вносить изменения нескольких разработчиков в один файл

Centralized vs. Distributed

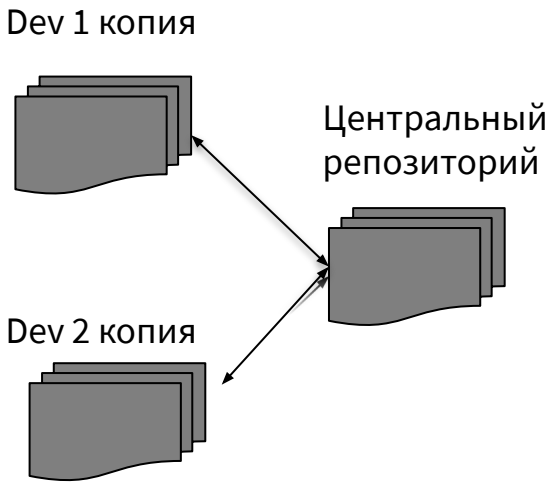
- Centralized
  - Client-server
  - Клон содержит только последнюю версию
  - Коммиты и слияния на стороне сервера.
- Distributed
  - Peer-to-peer
  - Коммиты и слияния локальные
  - Клонируется полная история коммитов etc

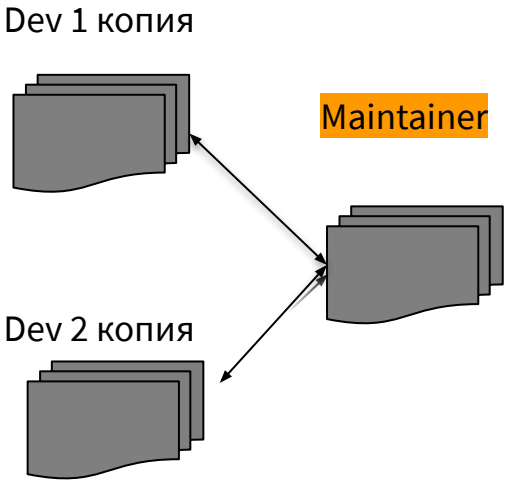
Centralized vs. Distributed



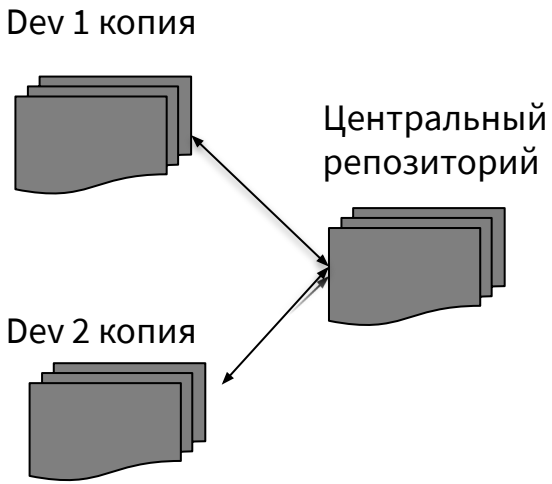


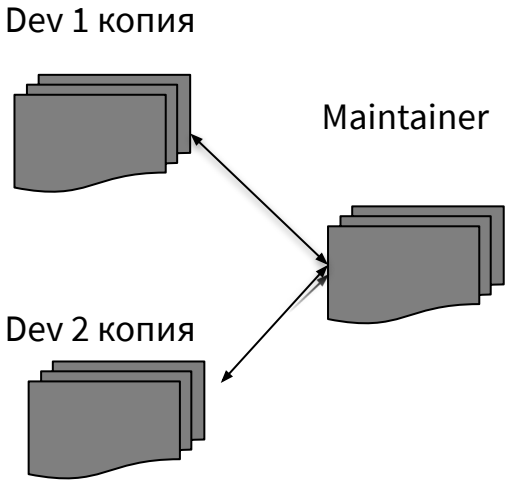
Centralized vs. Distributed





Centralized vs. Distributed





Разработчики не могут напрямую комитить в репозиторий мейнтейнера. Вместо этого просят мейнтейнера “утянуть” свой код в его репозиторий (pull-request)

Терминология

- Clone
  - Создание копии репозитория у себя
- Commit, check in
  - Фиксация изменений в коде
- Tag, label, baseline
  - Слепок состояния репозитория
- Head
  - Слепок состояния с самым последний коммитом
- Branch
  - Ветка репозитория - копия файлов, над которой ведется независимая разработка. Изменения в мей позже могут быть слиты в “главную” ветку (trunk, master)
- Merge
  - Процесс слияния изменений в один файл/ветку
- Pull
  - Синхронизация изменений в другом репо с локальными (push - наоборот)

Git

- Опять Линус Торвальдс!
  - И опять из-за лицензий (BitLocker)
- Распределенная система со слияниями изменений.
- *"git" can mean anything, depending on your mood.*
  - *random three-letter combination that is pronounceable, and not actually used by any common UNIX command. The fact that it is a mispronunciation of "get" may or may not be relevant.*
  - *stupid. contemptible and despicable. simple. Take your pick from the dictionary of slang.*
  - *"global information tracker": you're in a good mood, and it actually works for you. Angels sing, and a light suddenly fills the room.*
  - *"goddamn idiotic truckload of sh*t": when it breaks*

Облачные провайдеры

- Github, Gitlab, BitBucket etc
- De facto являются централизованными хранилищами для “главного” (origin) репозитория проекта.
- GUI для визуализации содержимого репозитория, а так же для для администрирования репозитория.
- Представляют множество дополнительных сервисов
  - CI/CD, Issue tracker
- Сам себе провадер
  - git daemon
  - Gitlab - полностью открытый код

Практика

- Работа с двумя локальными репозиториями
- Создаем репо на сайте Github
- Клонируем репо под персональным аккаунтом на сервере
- Добавляем файл, commit, push
- Клонируем этот репо под гостевым аккаунтом (или у себя на пк и.т.д)
- Вносим изменения, синхронизируем с Github
- Синхронизируем с репо под персональным аккаунтом

Задание. Первая копия репо.
На сервере под персональным аккаунтом

- На сайте Github.com создать новый репозиторий **ozon-masters-linux**
  - .gitignore для питона
  - Инициализировать с README.md
- На нашем сервере под персональным аккаунтом:

```
clone your_repo
echo ‘print(“Hello”)’ > hello.py
git status
git add hello.py
git commit -m “first commit!” hello.py
echo ‘The first programm [hello.py](hello.py)’ >> README.md
git status
git commit -a
# в открывшемся окне напишите описание коммита “link to hello.py from README”
git push
```

Clone

- Если хотите гит без пароля, вы знаете что делать с ключами :)
- Осторожно с копи-пастой кавычек!

```
datamove@linux1:~$ git clone https://github.com/datamove/ozon-masters-linux.git
Cloning into 'ozon-masters-linux'...
Username for 'https://github.com': datamove
Password for 'https://datamove@github.com':
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), 1.57 KiB | 1.57 MiB/s, done.
datamove@linux1:~$ cd ozon-masters-linux/
datamove@linux1:~/ozon-masters-linux$ echo ‘print(“Hello”)’ > hello.py
-bash: syntax error near unexpected token `('
datamove@linux1:~/ozon-masters-linux$ echo 'print("Hello")' > hello.py
datamove@linux1:~/ozon-masters-linux$ cat hello.py
print("Hello")
```

На первом коммите на новом месте

```
datamove@linux1:~/ozon-masters-linux$ git commit -m "first commit!" hello.py
[master 946a52c] first commit!
 Committer: Trunov_Artem <datamove@linux1.ru-central1.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 1 insertion(+)
 create mode 100644 hello.py
```

- Отредактируйте конфиг
  - Email - напишите тот, который привязан к вашему аккаунту в git. Именно по нему гит идентифицирует коммиты.

```
datamove@linux1:~/ozon-masters-linux$ cat ~/.gitconfig
# This is Git's per-user configuration file.
[user]
# Please adapt and uncomment the following lines:
#       name = Trunov_Artem
#       email = datamove@linux1.ru-central1.internal
```

Готовимся к синхронизации с Github

```
datamove@linux1:~/ozon-masters-linux$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
datamove@linux1:~/ozon-masters-linux$ git log
commit 946a52cf2591992fbed2e9cf57b4a7b31ab7654d (HEAD -> master)
Author: Trunov_Artem <datamove@linux1.ru-central1.internal>
Date:   Mon Sep 28 09:40:45 2020 +0000

    first commit!

commit b8f955d307e65bb9012f6f9c63a7a466fe42b0bf (origin/master, origin/HEAD)
Author: Artem Trunov <datamove@gmail.com>
Date:   Mon Sep 28 12:35:00 2020 +0300

    Initial commit
datamove@linux1:~/ozon-masters-linux$ git show
commit 946a52cf2591992fbed2e9cf57b4a7b31ab7654d (HEAD -> master)
Author: Trunov_Artem <datamove@linux1.ru-central1.internal>
Date:   Mon Sep 28 09:40:45 2020 +0000

    first commit!

diff --git a/hello.py b/hello.py
new file mode 100644
index 0000000..2f9a147
--- /dev/null
+++ b/hello.py
@@ -0,0 +1 @@
+print("Hello")
```

Push!

```
datamove@linux1:~/ozon-masters-linux$ git push
Username for 'https://github.com': datamove
Password for 'https://datamove@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 339 bytes | 339.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/datamove/ozon-masters-linux.git
   b8f955d..946a52c  master -> master
```

Задание. Вторая копия репо. На сервере под гостевым аккаунтом

- Работаем со второй копией вашего репозитория. Склонируйте его:
  - либо на лаптоп
  - либо на нашем сервере под гостевым аккаунтом
  - либо под персональным в другую папку:

```
git clone your_repo oml
cd oml
# В файле hello.py вместо “Hello”, напишите “Hello, git”
git status
git diff hello.py
git commit -a -m “changed greetings”
git push
```

Репо под вторым аккаунтом

```
lguest217@linux1:~$ git clone https://github.com/datamove/ozon-masters-linux.git oml
Cloning into 'oml'...
Username for 'https://github.com': datamove
Password for 'https://datamove@github.com':
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 7 (delta 1), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (7/7), 1.80 KiB | 1.80 MiB/s, done.
lguest217@linux1:~$ cd oml
lguest217@linux1:~/oml$ ls -al
total 24
drwxr-xr-x 3 lguest217 students 4096 Sep 28 10:05 .
drwxr-xr-x 7 lguest217 students 4096 Sep 28 10:05 ..
drwxr-xr-x 8 lguest217 students 4096 Sep 28 10:05 .git
-rw-r--r-- 1 lguest217 students 1799 Sep 28 10:05 .gitignore
-rw-r--r-- 1 lguest217 students   15 Sep 28 10:05 hello.py
-rw-r--r-- 1 lguest217 students   20 Sep 28 10:05 README.md
lguest217@linux1:~/oml$
```

Изменяем

```
lguest217@linux1:~/oml$ vi hello.py
lguest217@linux1:~/oml$ cat hello.py
print("Hello, git")
lguest217@linux1:~/oml$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   hello.py

no changes added to commit (use "git add" and/or "git commit -a")
lguest217@linux1:~/oml$ git diff hello.py
diff --git a/hello.py b/hello.py
index 2f9a147..08c6f99 100644
--- a/hello.py
+++ b/hello.py
@@ -1 +1 @@
-print("Hello")
+print("Hello, git")
```

В этом репо - вся история коммитов

```
lguest217@linux1:~/oml$ git log
commit 318dc16456c21c496c74a82230b3b9e79df5d7b1 (HEAD -> master)
Author: Trunov_Artem <lguest217@linux1.ru-central1.internal>
Date:   Mon Sep 28 10:22:02 2020 +0000

    changed greetings

commit 946a52cf2591992fbed2e9cf57b4a7b31ab7654d (origin/master, origin/HEAD)
Author: Trunov_Artem <datamove@linux1.ru-central1.internal>
Date:   Mon Sep 28 09:40:45 2020 +0000

    first commit!

commit b8f955d307e65bb9012f6f9c63a7a466fe42b0bf
Author: Artem Trunov <datamove@gmail.com>
Date:   Mon Sep 28 12:35:00 2020 +0300

    Initial commit
```

Задание. Снова первая копия репо

cd ozon-masters-linux
тут надо сначала сделать git pull, но давайте посмотрим, что будет если не сделать этого
в hello.py замените “Hello” на “Hello World“
git commit
git log

#Ой!
Теперь два варианта
#1. git pull + слияние руками
#2. git stash, git pull, git apply
№3 force copy from the origin?

Первая копия репо.

Ничего (пока) не знает о коммитах, сделанных в другом месте

```
datamove@linux1:~/ozon-masters-linux$ git commit -m "hello world" hello.py
[master 82bc3d9] hello world
 1 file changed, 1 insertion(+), 1 deletion(-)
datamove@linux1:~/ozon-masters-linux$ git log
commit 82bc3d9a4fa5927a5f61e38b265f14609e1dea91 (HEAD -> master)
Author: Artem Trunov <datamove@gmail.com>
Date:   Mon Sep 28 10:36:48 2020 +0000

    hello world

commit 75fccc3354824978ae977a44bf612e2b858a3c7e (origin/master, origin/HEAD)
Author: Artem Trunov <datamove@gmail.com>
Date:   Mon Sep 28 10:35:10 2020 +0000

    first commit!

commit 64905fd3d3c696919c07186c678ded1a85fcddc2
Author: Artem Trunov <datamove@gmail.com>
Date:   Mon Sep 28 13:33:52 2020 +0300

    Initial commit
datamove@linux1:~/ozon-masters-linux$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Первая копия репо. Пытаемся синхронизироваться

git push
#Ой!

```
datamove@linux1:~/ozon-masters-linux$ git push
Username for 'https://github.com': datamove
Password for 'https://datamove@github.com':
To https://github.com/datamove/ozon-masters-linux.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://github.com/datamove/ozon-masters-linux.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

- Теперь два варианта
  - git pull + слияние руками
  - git stash, git pull, git apply
  - force copy from the origin?

1. Git pull + ручное слияние

```
datamove@linux1:~/ozon-masters-linux$ git pull
Username for 'https://github.com': datamove
Password for 'https://datamove@github.com':
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (1/1), done.
remote: Total 3 (delta 1), reused 3 (delta 1), pack-reused 0
Unpacking objects: 100% (3/3), 256 bytes | 256.00 KiB/s, done.
From https://github.com/datamove/ozon-masters-linux
   75fccc3..699f8f8  master     -> origin/master
Auto-merging hello.py
CONFLICT (content): Merge conflict in hello.py
Automatic merge failed; fix conflicts and then commit the result.
datamove@linux1:~/ozon-masters-linux$ cat hello.py
<<<<<<< HEAD
print("Hello, World")
=======
print("Hello, git")
>>>>>>> 699f8f87bea99110b443c9c1743abd43dc649410
```

1. Отредактируем и commit

```
datamove@linux1:~/ozon-masters-linux$ vi hello.py
datamove@linux1:~/ozon-masters-linux$ git diff hello.py
diff --cc hello.py
index e511247,08c6f99..0000000
--- a/hello.py
+++ b/hello.py
datamove@linux1:~/ozon-masters-linux$ git status
On branch master
Your branch and 'origin/master' have diverged,
and have 1 and 1 different commits each, respectively.
  (use "git pull" to merge the remote branch into yours)

You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   hello.py

no changes added to commit (use "git add" and/or "git commit -a")
datamove@linux1:~/ozon-masters-linux$ git commit -m "resolved conflict" hello.py
fatal: cannot do a partial commit during a merge.
datamove@linux1:~/ozon-masters-linux$ git commit -a -m "resolved conflict" hello.py
fatal: paths 'hello.py ...' with -a does not make sense
datamove@linux1:~/ozon-masters-linux$ git commit -a -m "resolved conflict"
[master 5917326] resolved conflict
```

Попробуем второй метод на второй копии репо

- Изменяем файл и пытаемся сделать git pull

```
lguest217@linux1:~/oml$ cat hello.py
print("Hello, git")
lguest217@linux1:~/oml$ vi hello.py
lguest217@linux1:~/oml$ git diff hello.py
diff --git a/hello.py b/hello.py
index 08c6f99..aadab71 100644
--- a/hello.py
+++ b/hello.py
@@ -1 +1 @@
-print("Hello, git")
+print("Hello, git!")
```

```
lguest217@linux1:~/oml$ git pull
Username for 'https://github.com': datamove
Password for 'https://datamove@github.com':
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (1/1), done.
remote: Total 4 (delta 2), reused 4 (delta 2), pack-reused 0
Unpacking objects: 100% (4/4), 378 bytes | 378.00 KiB/s, done.
From https://github.com/datamove/ozon-masters-linux
   699f8f8..5917326  master     -> origin/master
Updating 699f8f8..5917326
error: Your local changes to the following files would be overwritten by merge:
        hello.py
Please commit your changes or stash them before you merge.
Aborting
```

Пробуем git stash

```
lguest217@linux1:~/oml$ git stash
Saved working directory and index state WIP on master: 699f8f8 changed greetings
lguest217@linux1:~/oml$ git pull
Username for 'https://github.com': datamove
Password for 'https://datamove@github.com':
Updating 699f8f8..5917326
Fast-forward
 hello.py | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
lguest217@linux1:~/oml$ git stash list
stash@{0}: WIP on master: 699f8f8 changed greetings
lguest217@linux1:~/oml$ git stash apply
Auto-merging hello.py
CONFLICT (content): Merge conflict in hello.py
lguest217@linux1:~/oml$ git status
On branch master
Your branch is up to date with 'origin/master'.

Unmerged paths:
  (use "git restore --staged <file>..." to unstage)
  (use "git add <file>..." to mark resolution)
        both modified:   hello.py

no changes added to commit (use "git add" and/or "git commit -a")
lguest217@linux1:~/oml$ git diff hello.py
diff --cc hello.py
index e511247,aadab71..0000000
--- a/hello.py
+++ b/hello.py
@@@ -1,1 -1,1 +1,5 @@@
++<<<<<<< Updated upstream
 +print("Hello, World")
++=======
+ print("Hello, git!")
++>>>>>>> Stashed changes
```

После stash тоже надо разрешать merge конфликты, если они есть

И далее, как обычно

git commit

git push

Практические советы

- Всегда делайте git pull перед началом работы
- Перед коммитами делайте git status, git pull
- Если что-то идет не так - сохраните ваши изменения отдельно и удалите локальный репозиторий. Затем склонируйте заново и внесите необходимые изменения.
- Как сделать чтоб показывались названия файлов с кириллицей?