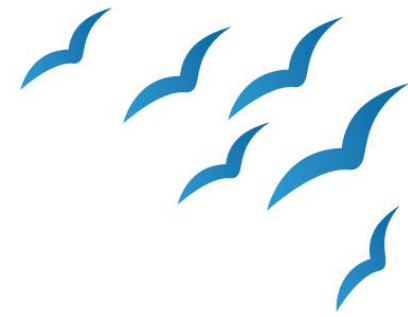




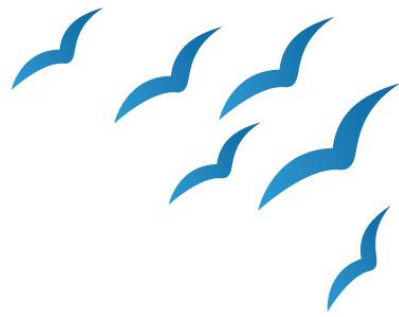
- Система контроля версий



# История создания

Проект был создан Линусом Торвальдсом (Linus Benedict Torvalds) для управления разработкой ядра Linux.

Первая версия выпущена 7 апреля 2005 года меньше чем за неделю разработки.



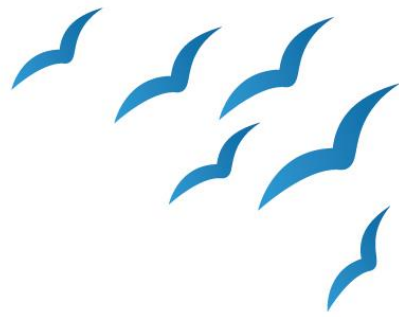
# Git – ХОСТИНГ

- GitHub
- GitLab
- Bitbucket
- SourceForge
- Codebase



# Git – система контроля версии

- Летописец
- Машина времени
- Резервная копия
- Мастер параллельных миров
- Народное вече

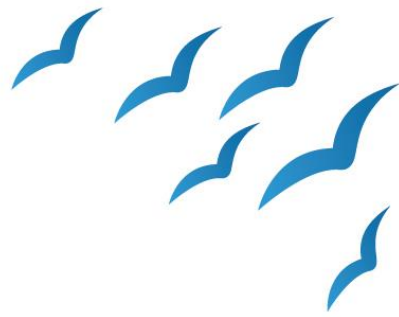


# Летописец

Git – ведет всю историю разработки начиная от сотворения проекта. Делает подшивку черновиков в ветку по контрольным точкам (commit)

> git log

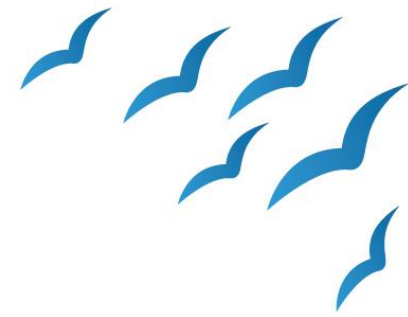
commit c80803cc67c80de23be9fd32626a49dc8c20895e (HEAD -> master, origin/master, origin/HEAD)



# Машина времени

Предоставляет возможность путешествовать в прошлое по дереву летописных сводов.

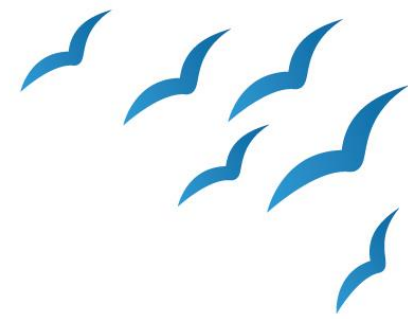
```
>git checkout <hash>
```



# Резервная копия

Создание резервных копий на удаленных серверах git репозитория.

```
>git push origin master
```

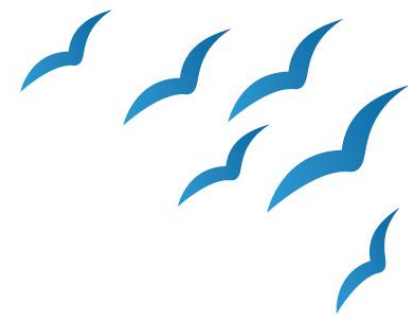


# Мастер параллельных миров

В Git существует возможность распареллелить ветку разработки на несколько , посредством создания дополнительных веток.

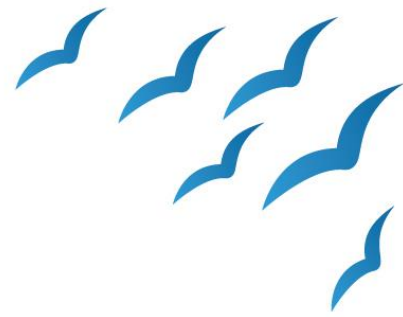
```
>git branch new_branch
```





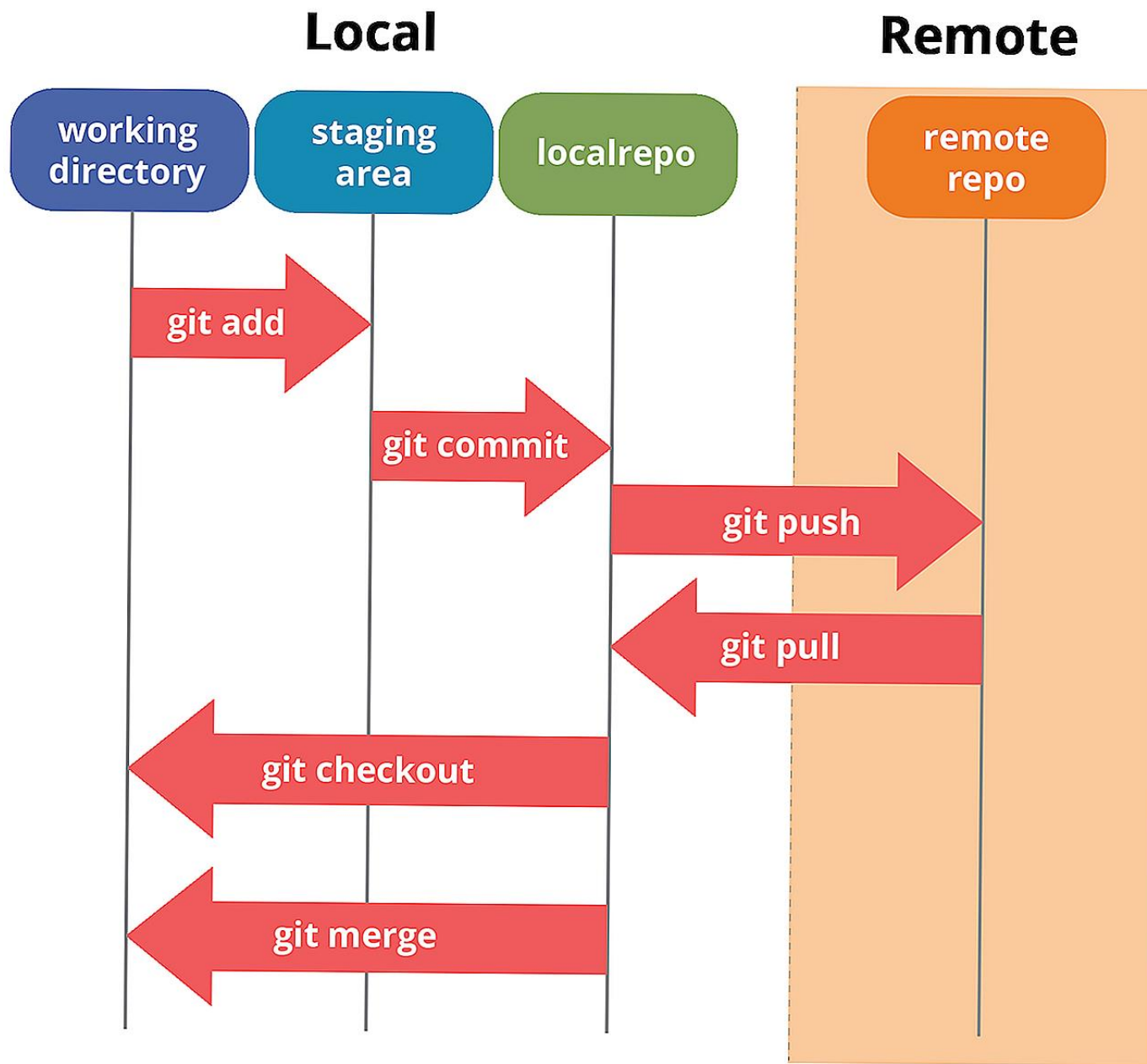
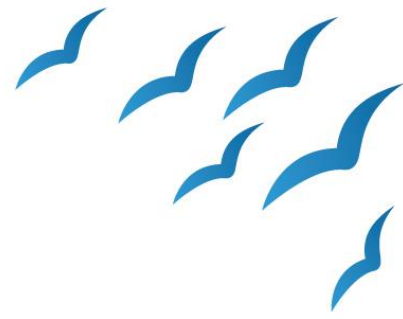
# Народное вече

Git воплощает в себе механизмы коллективного общения разработчиков. Принятия и обсуждения тех или иных технических решений. Идея pull request



Управление репозиторием Git  
сводится к набору команд.  
Рассмотрим диаграмму.





# Установка Git

## (<https://git-scm.com/downloads>)



git-scm.com/downloads

Haskell Python JS

**git** --fast-version-control

Search entire site...

**About**  
**Documentation**  
**Downloads**  
GUI Clients  
Logos  
**Community**

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

## Downloads

macOS Windows Linux/Unix

Older releases are available and the Git source repository is on GitHub.

Latest source Release  
**2.39.0**  
[Release Notes \(2022-12-12\)](#)  
[Download for Windows](#)

### GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients →](#)

### Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

[View Logos →](#)

### Git via Git

If you already have Git installed, you can get the latest development version via Git itself:

```
git clone https://github.com/git/git
```

You can also always browse the current contents of the git repository using the [web interface](#).



# Зарегистрироваться на GitHub

<https://github.com/>

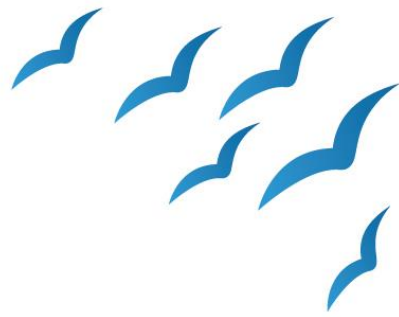
Выбрать разумное имя, типа MikhailZimnev59

Создать репозиторий: **python\_winter\_work\_2023**

Послать мне полное имя репозитория на почту  
[m.zimnev@yandex.ru](mailto:m.zimnev@yandex.ru)

Для проверки домашних заданий.

Срок помещения заданий: 14.00 – 16.00 перед  
следующей лекцией.



# Настройка

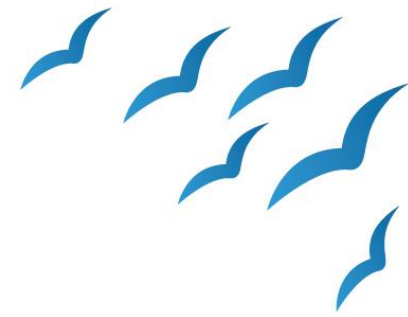
#Устанавливаем в командной строке

```
git config --global user.name "<ваше_имя>"
```

```
git config --global user.email "<адрес_почты@email.com>"
```

# Просмотр настроек

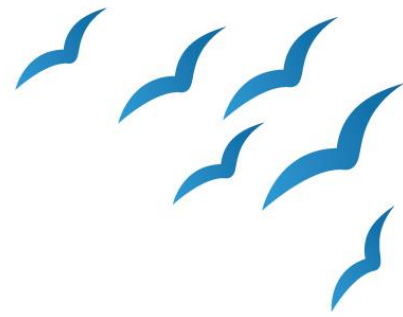
```
>git config --list
```



# Превратить каталог, который не находится под  
# версионным контролем, в репозиторий Git

>**git** init

# После того как была создана папка .git  
# добавляем в текущей каталог файл .gitignore  
# В него заносим все файлы и папки которые не хотим # индексировать



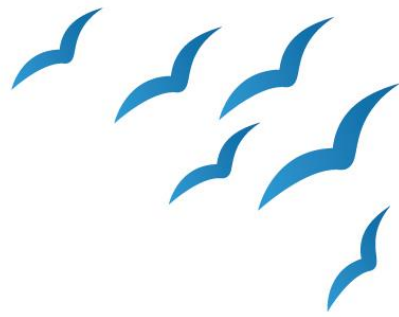
# Индексировать измененный файл

>**git** add [путь к файлу]

# Индексировать измененные файлы

>**git** add .

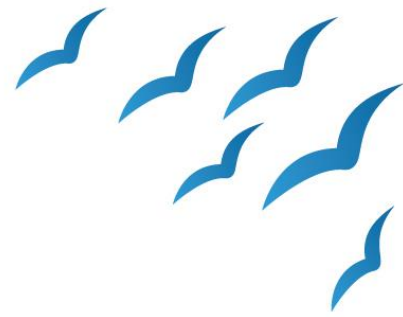




# Просмотр изменений

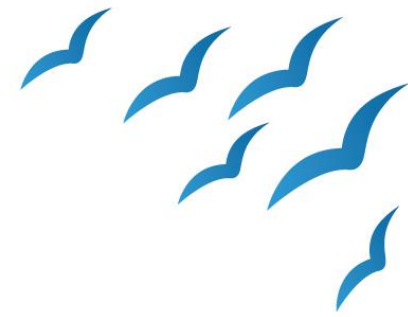
>**git** status





# Фиксация изменения в локальное хранилище  
# В коммит попадут (будут сохранены) только файлы,  
# которые были проиндексированы командой git add

>**git commit -m** “комментарии к коду”



# Просмотр истории коммитов

>**git log**

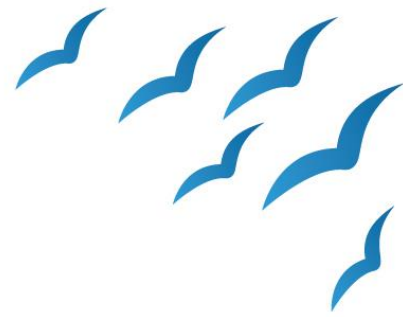
# Информация о коммите (метаданные):

# 1. уникальный идентификатор коммита (хеш);

# 2. имя и email автора коммита;

# 3. дата создания коммита;

# 4. комментарий к коммиту.



# Связывание локального репозитория с GitHub

```
>git remote add origin  
  git@github.com:my\_name/my\_repo.git
```

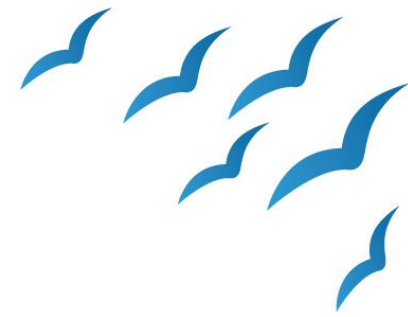
# Где `my_name` — имя пользователя на GitHub  
 `my_repo` — название созданного репозитория

# Проверка свзыывания

```
>git remote get-url origin
```

# Удаление свзыывания

```
>git remote remove origin
```



# Отправка изменений в удаленный  
# репозиторий origin ветки master

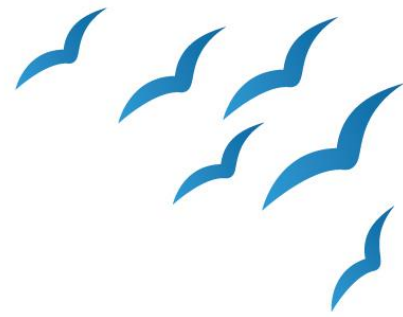
>**git push** origin master

# Отправка изменений с текущей ветки

>**git push**

# Получение изменений

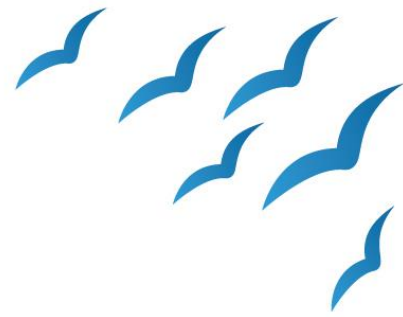
>**git pull**



# Клонировать существующий репозиторий

>**git clone** ссылка-на-репозиторий

# Задача !



# todo: Клонировать существующий репозиторий

>**git clone** [https://github.com/MikhallZimnev59/python\\_winter\\_2023.git](https://github.com/MikhallZimnev59/python_winter_2023.git)

# Задание



- 1) Создать репозиторий на GitHub с названием **python\_winter\_work\_2023**
- 2) Создать папку на локальной машине для домашних работ с названием **python\_winter\_work\_2023**
- 3) Превратить созданный каталог в репозиторий Git
- 4) Создать в папке каталога файл README.MD
- 5) Добавить его в локальный репозиторий
- 6) Связать удаленный репозиторий с локальным
- 7) Отправить изменения в удаленный репозиторий





# Используя PyCharm: Создать новый репозиторий- GitHub

[Pull requests](#) [Issues](#) [Codespaces](#) [Marketplace](#) [Explore](#)

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?  
[Import a repository.](#)

Owner \*



MikhailZimnev59 ▾

Repository name \*

python\_winter\_2023 ✓

Great repository names are short. python\_winter\_2023 is available. ion? How about **potential-umbrella**?

Description (optional)



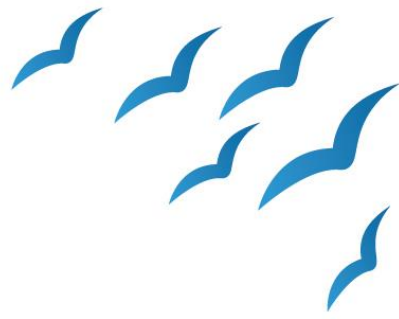
Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.



# Запомнить ссылку

ev59 / python\_winter\_2023 Public Pin Unwatch 1 Fork 0 Star

[Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

**Quick setup — if you've done this kind of thing before**

[Set up in Desktop](#) or [HTTPS](#) [SSH](#)  [Copy](#)

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).



# Создать новый проект в PyCharm

**Create Project**

Location:

▼ Python Interpreter: New Virtualenv environment

☒ New environment using Virtualenv

Location:

Base interpreter:

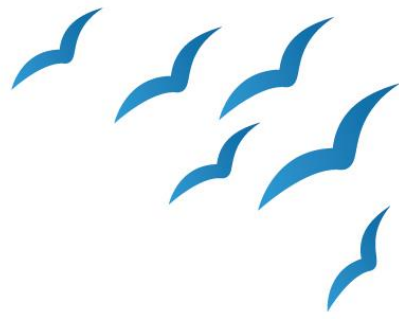
☐ Inherit global site-packages

☐ Make available to all projects

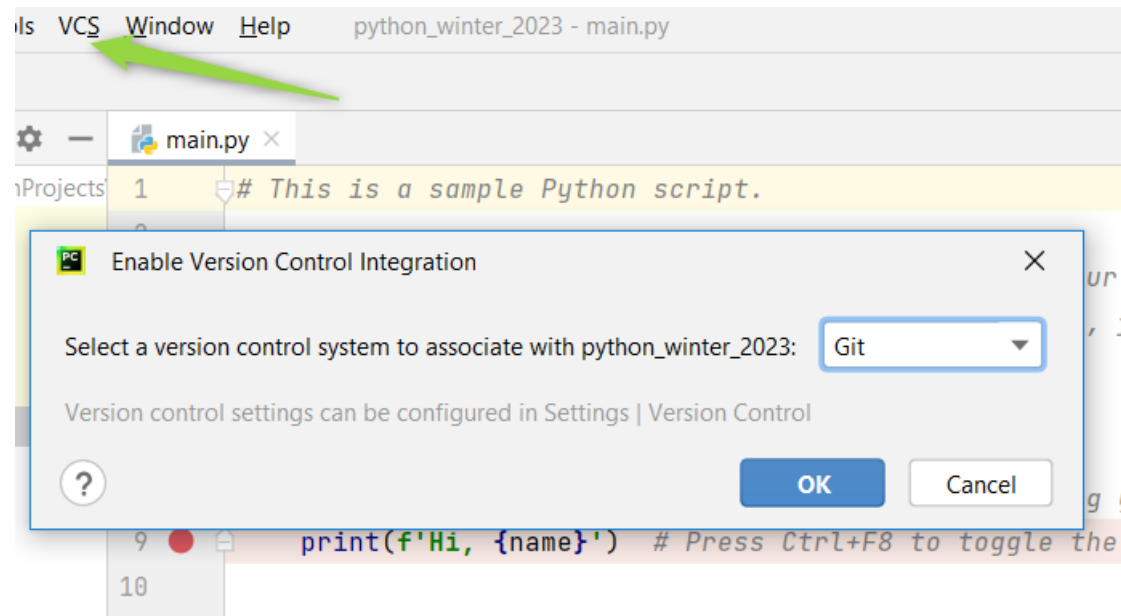
☐ Previously configured interpreter

Interpreter:

☒ Create a main.py welcome script  
Create a Python script that provides an entry point to coding in PyCharm.

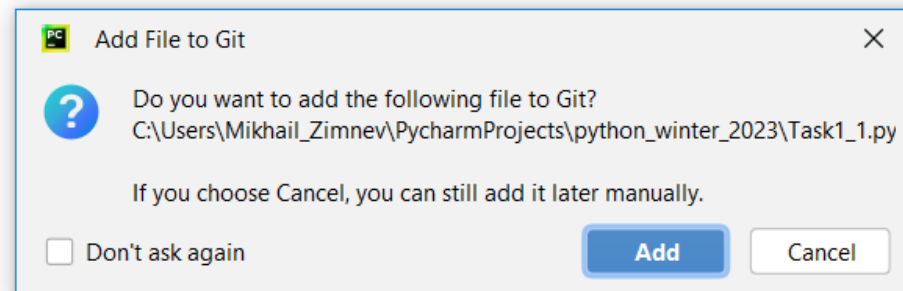
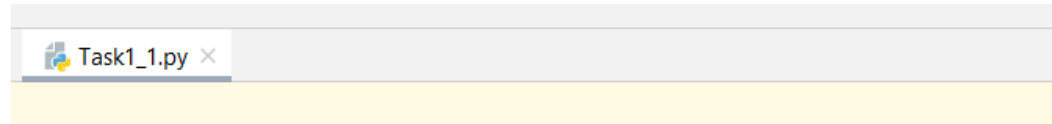


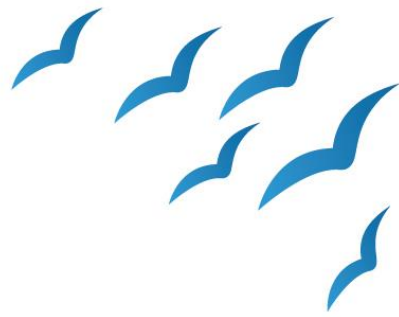
# Enable Version Control - Git



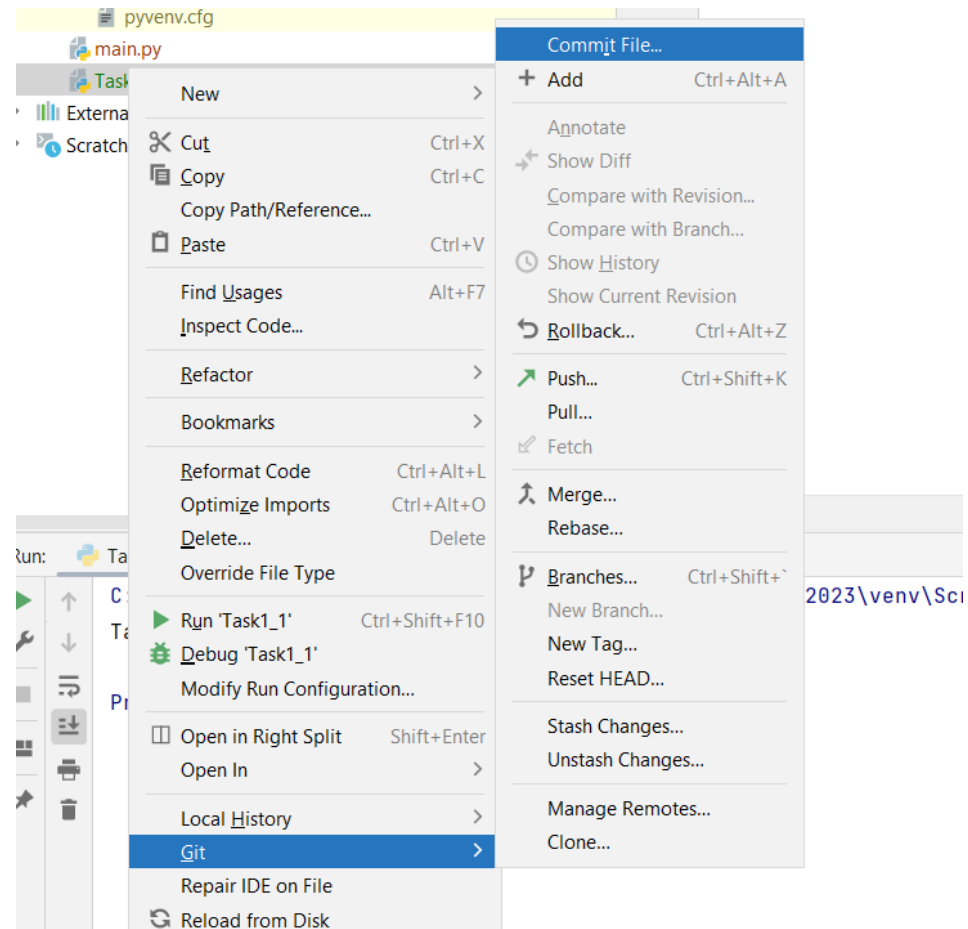


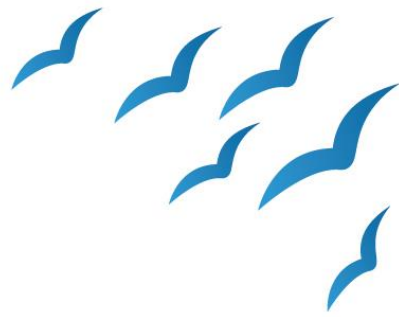
# Создать новый файл и добавить в Git



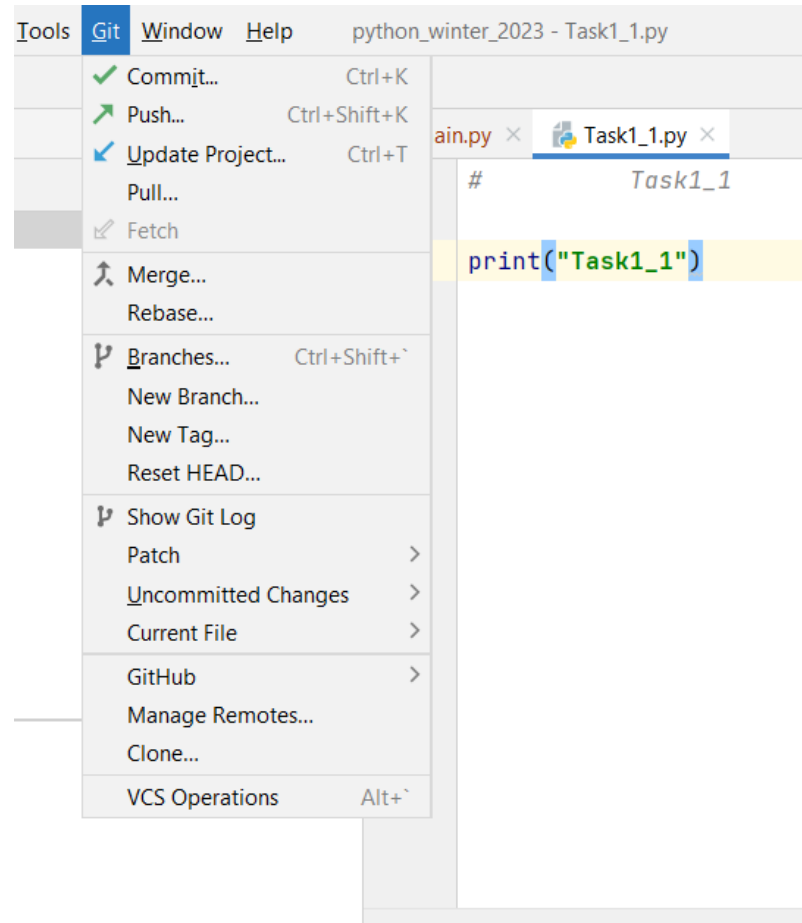


# Commit



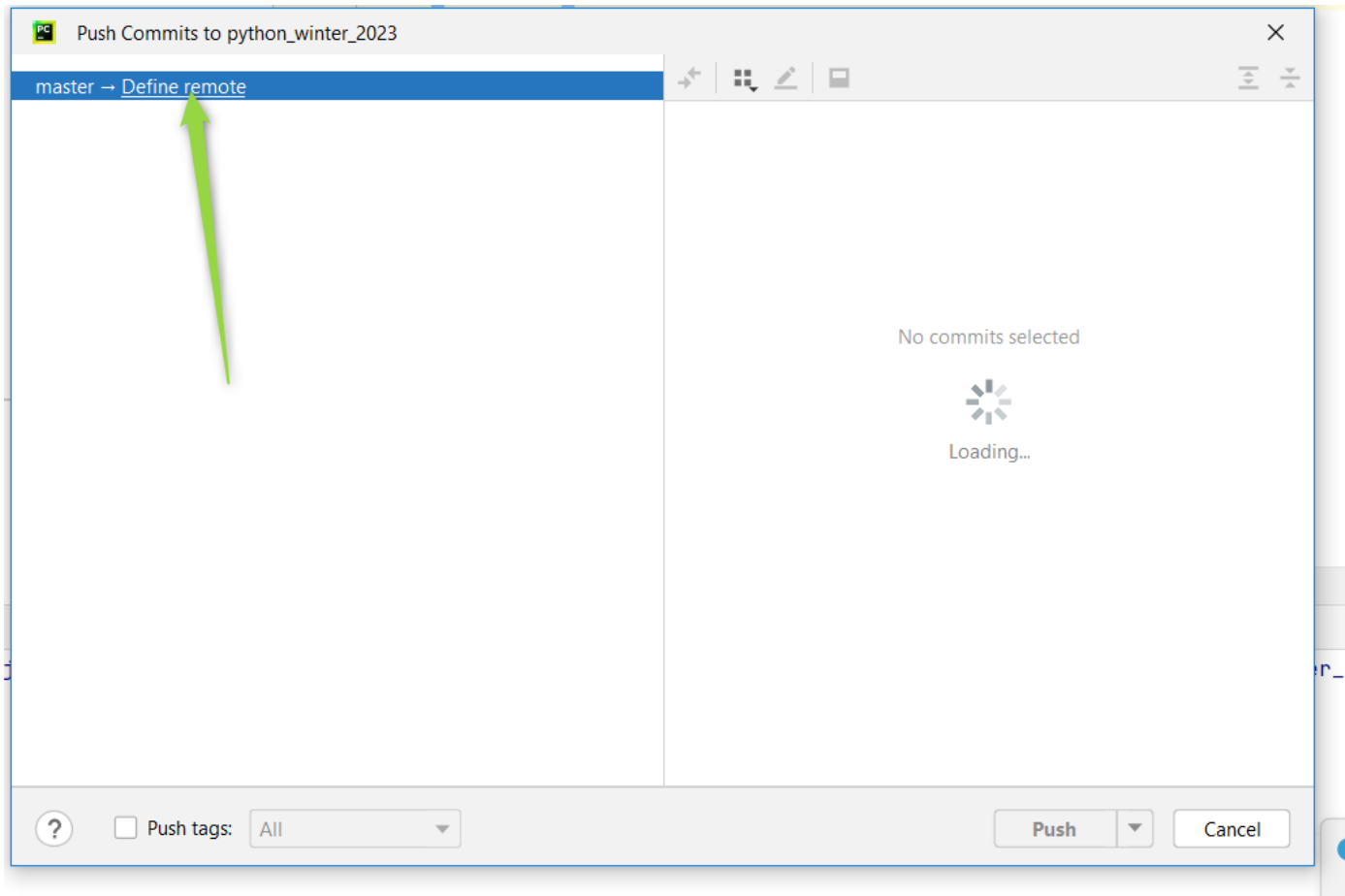


# Push

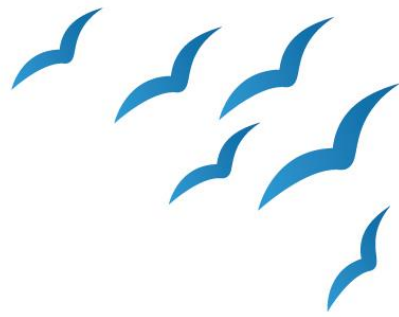





# Скопировать ссылку из GitHub









# Файл попал в репозиторий

 MikhailZimnev59 / python\_winter\_2023 Public

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#)

 master  1 branch  0 tags

Mikhail_Z init	
 Task1_1.py	init

Help people interested in this repository understand your project by adding a README