

занятие 1

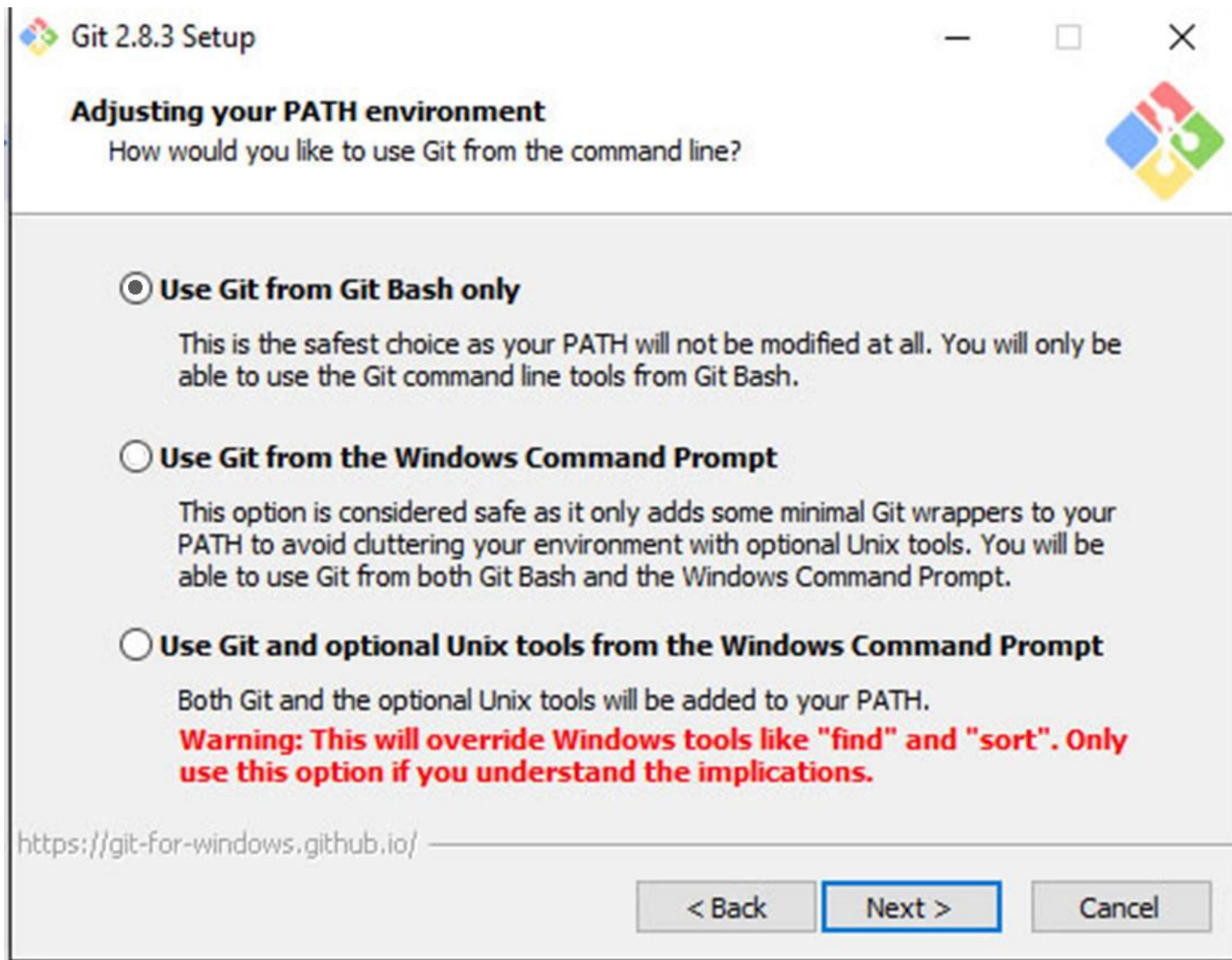


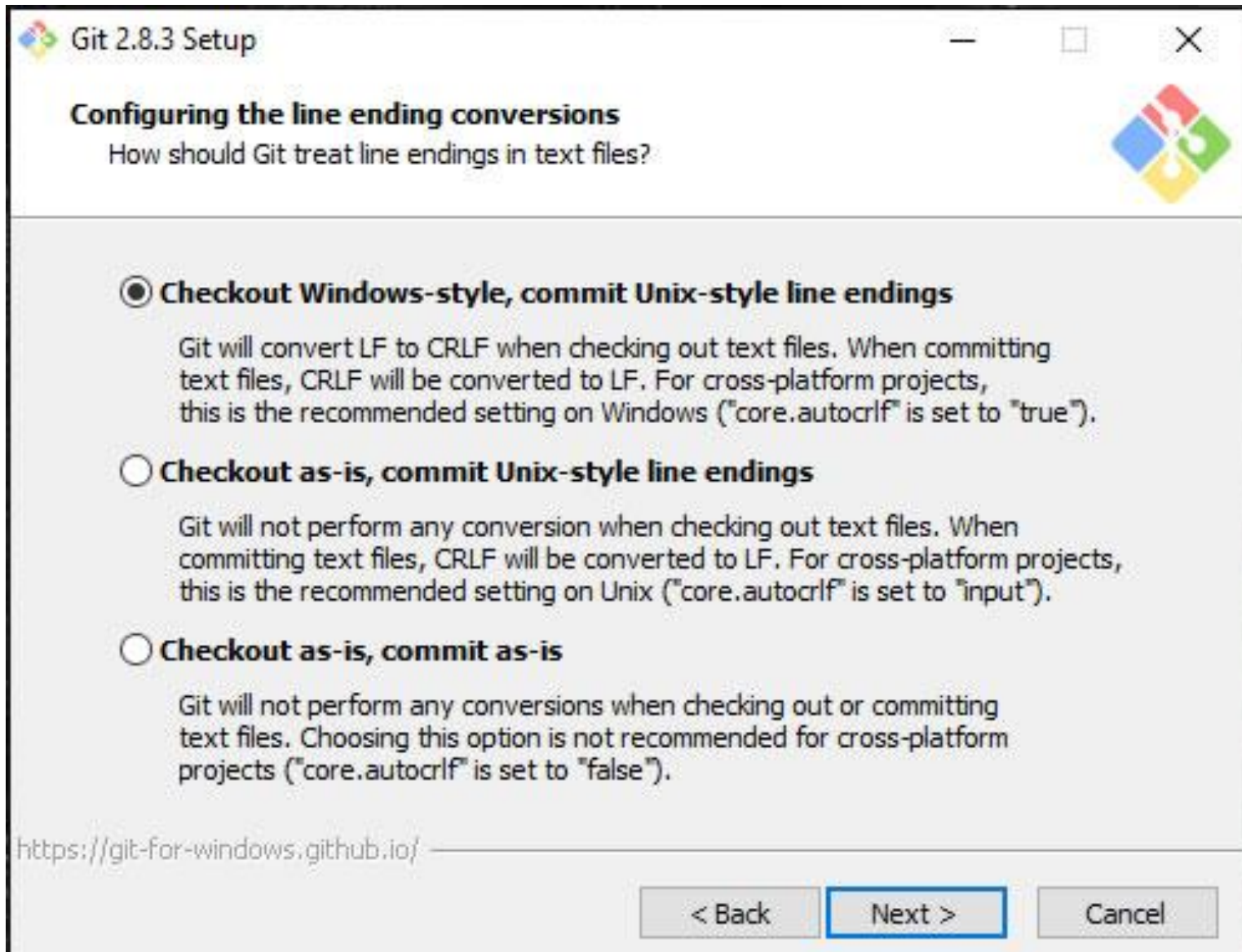
git



Внимание

при работе для именовании файлов, каталогов, коммитов и т.д. использовать только английский язык





Git 2.8.3 Setup



Configuring the terminal emulator to use with Git Bash

Which terminal emulator do you want to use with your Git Bash?



Use MinTTY (the default terminal of MSYS2)

Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `wipty` to work in MinTTY.

Use Windows' default console window

Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.

<https://git-for-windows.github.io/>

< Back

Next >

Cancel

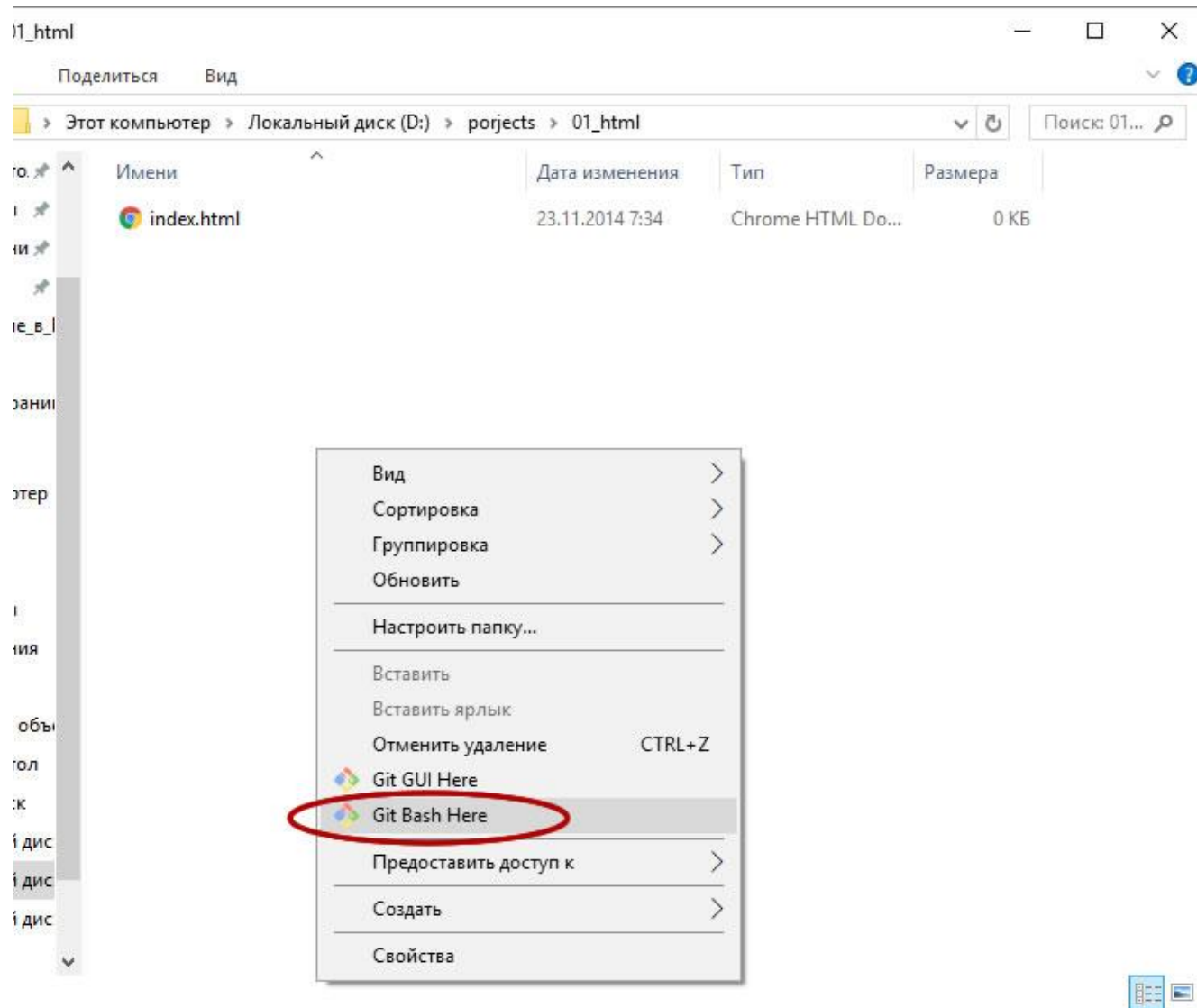
Базовая настройка GIT

Уровень	Linux	Windows
системы <code>--system</code>	<code>/etc/gitconfig</code>	<code>ProgramFiles\Git\etc\config</code>
пользователя <code>--global</code>	<code>/home/имя_польз/</code>	<code>C\Users\имя_польз\gitconfig</code>
проекта	<code>каталог_проекта/.git/.gitconfig</code>	

Уровень	Формат команды для конфигурации
системы	<code>git config --system команда</code>
пользователя	<code>git config --global команда</code>
проекта	<code>git config команда</code>


Практика - базовая настройка GIT

1. В проводнике зайти в каталог D:/projects/01_html
2. Правой кнопкой мыши вызвать контекстное меню и выбрать **Git Bash Here**




3. Набрать поочередно команды


```
git config --global user.name "СВОЕ_ИМЯ"  
git config --global user.email СВОЙ_EMAIL  
git config --global color.ui true
```

 MINGW32:/d/projects/01_html

```
roman@roman-pc MINGW32 /d/projects/01_html  
$ git config --global user.name "Bill Geitz"
```

 MINGW32:/d/projects/01_html

```
roman@roman-pc MINGW32 /d/projects/01_html  
$ git config --global user.email myemail@com.ua
```


 MINGW32:/d/projects/01_html

```
roman@roman-pc MINGW32 /d/projects/01_html  
$ git config --global color.ui true
```


4. Проверка настроек

```
git config -list
```

```
git config user.name
```

 MINGW32:/d/projects/01_html

```
roman@roman-pc MINGW32 /d/projects/01_html  
$ git config --list
```

`git help` -> выведет список команд

`git help log` -> справка по конкретной команде

Перемещение по тексту

`пробел` (или `f`) -> следующая страница

`b` -> предыдущая страница

`q` -> выход из справки

набираем команду `git init`

MINGW32:/d/projects/01_html

```
roman@roman-pc MINGW32 /d/projects/01_html
$ git init
Initialized empty Git repository in D:/projects/01_html/.git/

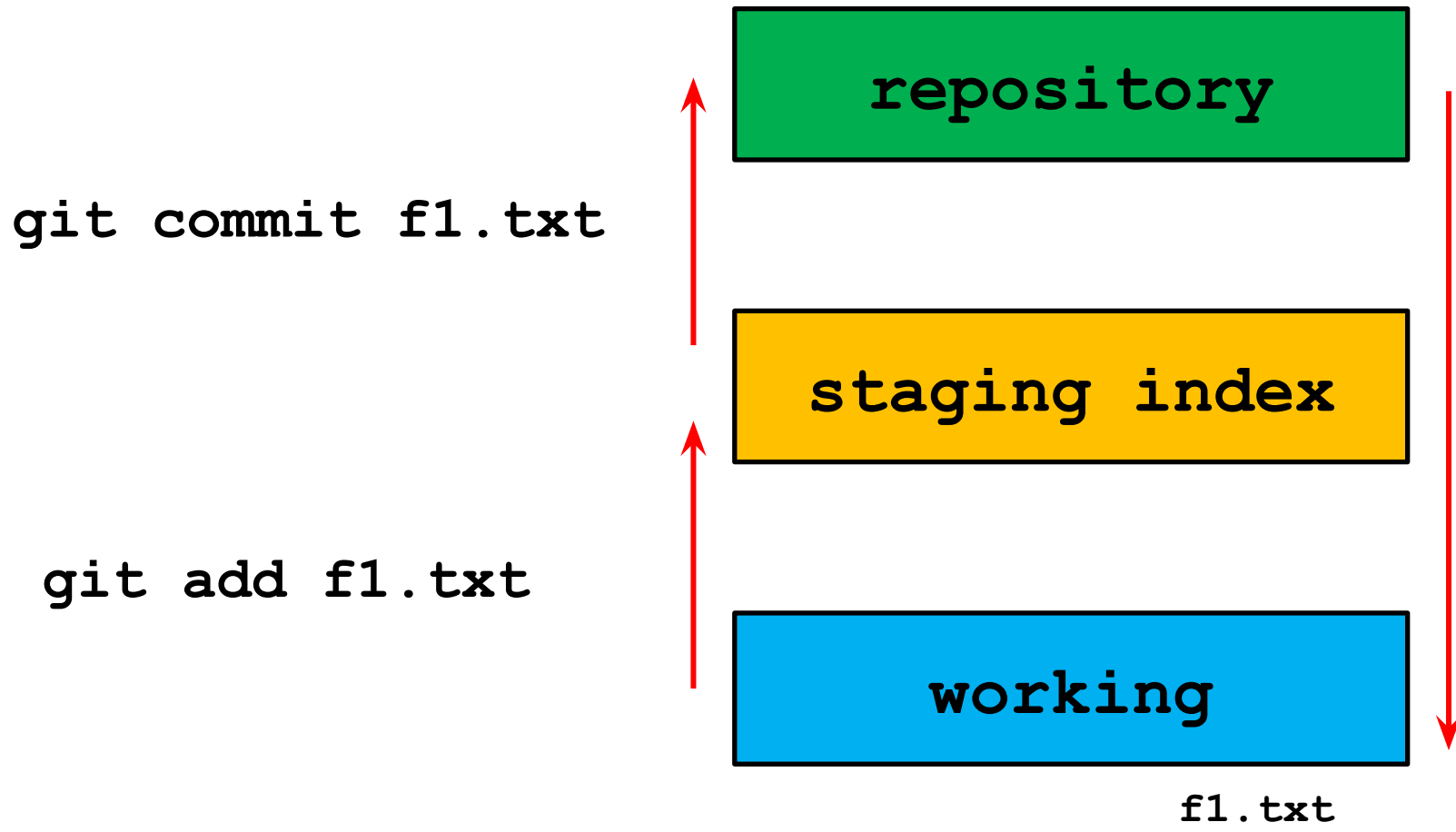
roman@roman-pc MINGW32 /d/projects/01_html (master)
$
```

Теперь каталог проекта – это репозиторий файлов проекта, и GIT будет отслеживать все сделанные в нем изменения.

В каталоге проекта создается каталог `.git` где будут храниться служебные файлы GIT и все изменения, сделанные в проекте

Из командной строки можно увидеть этот каталог набрав команду `ls -la`

Архитектура GIT - three tree



```
git add f1.txt
git commit f1.txt
git add f1.txt
git commit f1.txt
```

repository

f1.txt (v2)

staging index

f1.txt (v2)

working

f1.txt (v2)

B A

Git workflow

1. Создаем (редактируем) файл(ы)
2. Добавляем их в staging area
3. Commit изменения в репозиторий

1. Набрать команду
`git status`

```
MINGW32:/d/projects/01_html

roman@roman-pc MINGW32 /d/projects/01_html (master)
$ git status
On branch master

Initial commit

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

        index.html

nothing added to commit but untracked files present (use "git add" to track)

roman@roman-pc MINGW32 /d/projects/01_html (master)
$
```

3. Дадим команду GIT добавить изменения в staging area

```
git add .
```

где `.` - текущий каталог

MINGW32:/d/projects/01_html

```
roman@roman-pc MINGW32 /d/projects/01_html (master)
```

```
$ git add .
```

```
roman@roman-pc MINGW32 /d/projects/01_html (master)
```

```
$ git status
```

```
On branch master
```

```
Initial commit
```

```
Changes to be committed:
```

```
(use "git rm --cached <file>..." to unstage)
```

```
new file:   index.html
```

```
roman@roman-pc MINGW32 /d/projects/01_html (master)
```

```
$ |
```


Теперь нам нужно отправить изменения в постоянную память GIT

```
git commit -m "initial commit"
```

Где `-m "initial commit"` -> это описание коммита

MINGW32:/d/projects/01_html

```
roman@roman-pc MINGW32 /d/projects/01_html (master)
```

```
$ git commit -m "Initial commit"
```

```
[master (root-commit) 73d282a] Initial commit
```

```
1 file changed, 0 insertions(+), 0 deletions(-)
```

```
create mode 100644 index.html
```

```
roman@roman-pc MINGW32 /d/projects/01_html (master)
```

```
$ |
```

И после коммита олять наберем

`git status`

mingw32:/d/projects/01_html

```
roman@roman-pc MINGW32 /d/projects/01_html (master)
$ git status
On branch master
nothing to commit, working directory clean

roman@roman-pc MINGW32 /d/projects/01_html (master)
$
```

GIT нам показывает, что все изменения которые мы внесли в файл сохранены в его репозитории.

Таким образом можно продолжать работу дальше, НО если что то пойдет не так (например файлы случайно удалены) то можно всегда вернуться к этой сохраненной версии

Для просмотра истории коммитов набираем

`git log`

MINGW32:/d/projects/01_html

```
roman@roman-pc MINGW32 /d/projects/01_html (master)
```

```
$ git log
```

```
commit 73d282a948071362917c6e79e8ccb7e79ed832df
```

```
Author: samson25 <samson25@ukr.net>
```

```
Date: Mon Dec 11 10:20:56 2017 +0200
```

```
Initial commit
```

```
roman@roman-pc MINGW32 /d/projects/01_html (master)
```

```
$ |
```


index.html

```
<h1>Hello Git</h1>  
<p>this is paragraph</p>  
<p>another paragraph</p>
```

