

set up git username and email

```
brwla@AutumnPC MINGW64 /  
$ git config --global user.name "Oleg Ukhov"  
  
brwla@AutumnPC MINGW64 /  
$ git config --global user.email "o_ukhov@kbtu.kz"
```

init main branch

```
brwla@AutumnPC MINGW64 /  
$ git config --global init.defaultBranch main  
  
brwla@AutumnPC MINGW64 /  
$ git config --global core.autocrlf true  
git config --global core.safecrlf warn
```

set autorcrlf true

```
brwla@AutumnPC MINGW64 /  
$ git config --global core.autocrlf true  
git config --global core.safecrlf warn
```

create new directory 'work' and make hello.html file

```
brwla@AutumnPC MINGW64 /  
$ mkdir work  
cd work  
touch hello.html
```

Write something in the hello.html file

```
C: > DownloadedApps > Git > work > <> hello.html  
1 Hello, World!  
2
```

```
brwla@AutumnPC MINGW64 /work
$ git init
Initialized empty Git repository in C:/DownloadedApps/Git/work/.git/
```

git add and commit


```
brwla@AutumnPC MINGW64 /work (main)
$ git add hello.html

brwla@AutumnPC MINGW64 /work (main)
$ git commit -m "Initial Commit"
[main (root-commit) 4bf337e] Initial Commit
1 file changed, 1 insertion(+)
create mode 100644 hello.html
```

check status

```
brwla@AutumnPC MINGW64 /work (main)
$ git status
On branch main
nothing to commit, working tree clean
```

add h1 tag

```
C: > DownloadedApps > Git > work > <> hello.html >  h1
1  <h1>Hello, World!</h1>
```

check status

```
brwla@AutumnPC MINGW64 /work (main)
$ git status
On branch main
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.html

no changes added to commit (use "git add" and/or "git commit -a")
```

add new commit

```
brwla@AutumnPC MINGW64 /work (main)
$ git add hello.html
git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html
```

```
|
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch main
# Changes to be committed:
#       modified:   hello.html
#
```

Add a comment in the first line : Added h1 tag

```
[main a7d8ce9] Added h1 tag
1 file changed, 1 insertion(+), 1 deletion(-)
```

```
brwla@AutumnPC MINGW64 /work (main)
$ git status
On branch main
nothing to commit, working tree clean
```

modify HTML content

```
C: > DownloadedApps > Git > work > <> hello.html > ...
1   <html>
2   |   <body>
3   |       <h1>Hello, World!</h1>
4   |   </body>
5   </html>
6   
```

```
brw1a@AutumnPC MINGW64 /work (main)
$ git add hello.html
```

```
C: > DownloadedApps > Git > work > <> hello.html > ...
```

```
1  √ <html>
2      <head>
3      </head>
4  √ <body>
5      <h1>Hello, World!</h1>
6      </body>
7  </html>
8
```

```
brw1a@AutumnPC MINGW64 /work (main)
```

```
$ git status
```

```
On branch main
```

```
Changes to be committed:
```

```
(use "git restore --staged <file>..." to unstage)
    modified:   hello.html
```

```
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.html
```

```
brwla@AutumnPC MINGW64 /work (main)
$ git commit -m "Added standard HTML page tags"
git status
[main ecdcb2d] Added standard HTML page tags
1 file changed, 5 insertions(+), 1 deletion(-)
On branch main
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.html

no changes added to commit (use "git add" and/or "git commit -a")
```

```
brwla@AutumnPC MINGW64 /work (main)
$ git add .
git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html
```

```
brwla@AutumnPC MINGW64 /work (main)
$ git commit -m "Added HTML header"
[main 11c72d7] Added HTML header
1 file changed, 2 insertions(+)
```

check git log to see commits

```

brwla@AutumnPC MINGW64 /work (main)
$ git log
commit 11c72d79535630f7306752a5b812a34c9f0f4dd3 (HEAD -> main)
Author: Oleg Ukhov <o_ukhov@kbtu.kz>
Date:   Wed Jan 22 00:01:38 2025 +0500

    Added HTML header

commit ecdcb2d800a903fe1398667b090a77f3436fadaa
Author: Oleg Ukhov <o_ukhov@kbtu.kz>
Date:   Wed Jan 22 00:00:19 2025 +0500

    Added standard HTML page tags

commit a7d8ce92765934aa8c3e7e17da2075e2f12518f6
Author: Oleg Ukhov <o_ukhov@kbtu.kz>
Date:   Tue Jan 21 23:51:22 2025 +0500

    Added h1 tag

commit 4bf337e967274a3792e1dae7fade541c5b4d0e96
Author: Oleg Ukhov <o_ukhov@kbtu.kz>
Date:   Tue Jan 21 22:14:54 2025 +0500

    Initial Commit

```

see commit time and author

```

brwla@AutumnPC MINGW64 /work (main)
$ git log --pretty=oneline
11c72d79535630f7306752a5b812a34c9f0f4dd3 (HEAD -> main) Added HTML header
ecdcb2d800a903fe1398667b090a77f3436fadaa Added standard HTML page tags
a7d8ce92765934aa8c3e7e17da2075e2f12518f6 Added h1 tag
4bf337e967274a3792e1dae7fade541c5b4d0e96 Initial Commit

```

try git log with different formats

```

brwla@AutumnPC MINGW64 /work (main)
$ git log --oneline --max-count=2
git log --oneline --since="5 minutes ago"
git log --oneline --until="5 minutes ago"
git log --oneline --author="Your Name"
git log --oneline --all
11c72d7 (HEAD -> main) Added HTML header
ecdcb2d Added standard HTML page tags
11c72d7 (HEAD -> main) Added HTML header
ecdcb2d Added standard HTML page tags
a7d8ce9 Added h1 tag
4bf337e Initial Commit
11c72d7 (HEAD -> main) Added HTML header
ecdcb2d Added standard HTML page tags
a7d8ce9 Added h1 tag
4bf337e Initial Commit

```

```

brwla@AutumnPC MINGW64 /work (main)
$ git log --all --pretty=format:"%h %cd %s (%an)" --since="7 days ago"
11c72d7 wed Jan 22 00:01:38 2025 +0500 Added HTML header (Oleg Ukhov)
ecdcb2d wed Jan 22 00:00:19 2025 +0500 Added standard HTML page tags (Oleg Ukhov)
a7d8ce9 Tue Jan 21 23:51:22 2025 +0500 Added h1 tag (Oleg Ukhov)
4bf337e Tue Jan 21 22:14:54 2025 +0500 Initial Commit (Oleg Ukhov)

```

```

brwla@AutumnPC MINGW64 /work (main)
$ git log --pretty=format:"%h %ad | %s%d [%an]" --date=short
11c72d7 2025-01-22 | Added HTML header (HEAD -> main) [Oleg Ukhov]
ecdcb2d 2025-01-22 | Added standard HTML page tags [Oleg Ukhov]
a7d8ce9 2025-01-21 | Added h1 tag [Oleg Ukhov]
4bf337e 2025-01-21 | Initial Commit [Oleg Ukhov]

```

```

brwla@AutumnPC MINGW64 /work (main)
$ git config --global format.pretty '%h %ad | %s%d [%an]
git config --global log.date short

```

check git log to see commits

```
brwla@AutumnPC MINGW64 /work (main)
$ git log
11c72d7 2025-01-22 | Added HTML header (HEAD -> main) [Oleg Ukhov]
ecdcb2d 2025-01-22 | Added standard HTML page tags [Oleg Ukhov]
a7d8ce9 2025-01-21 | Added h1 tag [Oleg Ukhov]
4bf337e 2025-01-21 | Initial Commit [Oleg Ukhov]
```

```
brwla@AutumnPC MINGW64 /work (main)
$ git checkout <hash>
cat hello.html
bash: syntax error near unexpected token `newline'
<html>
  <head>
</head>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
```

```
brwla@AutumnPC MINGW64 /work (main)
$ git switch main
cat hello.html
Already on 'main'
<html>
  <head>
</head>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
```

```
brwla@AutumnPC MINGW64 /work (main)
$ git tag v1
git log
11c72d7 2025-01-22 | Added HTML header (HEAD -> main, tag: v1) [Oleg Ukhov]
ecdcb2d 2025-01-22 | Added standard HTML page tags [Oleg Ukhov]
a7d8ce9 2025-01-21 | Added h1 tag [Oleg Ukhov]
4bf337e 2025-01-21 | Initial Commit [Oleg Ukhov]
```


check current branch status

```
brw1a@AutumnPC MINGW64 /work (main)
```

```
$ git checkout v1^
```

```
cat hello.html
```

```
Note: switching to 'v1^'.
```

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using `-c` with the switch command. Example:

```
git switch -c <new-branch-name>
```

Or undo this operation with:

```
git switch -
```

Turn off this advice by setting config variable `advice.detachedHead` to false

HEAD is now at ecdcb2d Added standard HTML page tags

```
<html>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
```

```
brw1a@AutumnPC MINGW64 /work ((ecdc2d...))
```

```
$ git tag v1-beta
```

```
git log
```

```
ecdc2d 2025-01-22 | Added standard HTML page tags (HEAD, tag: v1-beta) [Oleg Ukhov]
```

```
a7d8ce9 2025-01-21 | Added h1 tag [Oleg Ukhov]
```

```
4bf337e 2025-01-21 | Initial Commit [Oleg Ukhov]
```

create tag v1

```
brwla@AutumnPC MINGW64 /work ((v1-beta))
$ git checkout v1
git checkout v1-beta
Previous HEAD position was ecdcb2d Added standard HTML page tags
HEAD is now at 11c72d7 Added HTML header
Previous HEAD position was 11c72d7 Added HTML header
HEAD is now at ecdcb2d Added standard HTML page tags
```

```
brwla@AutumnPC MINGW64 /work ((v1-beta))
$ git tag
v1
v1-beta
```

```
brwla@AutumnPC MINGW64 /work ((v1-beta))
$ git log main --all
11c72d7 2025-01-22 | Added HTML header (tag: v1, main) [Oleg Ukhov]
ecdcb2d 2025-01-22 | Added standard HTML page tags (HEAD, tag: v1-beta) [Oleg Ukhov]
a7d8ce9 2025-01-21 | Added h1 tag [Oleg Ukhov]
4bf337e 2025-01-21 | Initial Commit [Oleg Ukhov]
```

```
brwla@AutumnPC MINGW64 /work ((v1-beta))
$ git switch main
Previous HEAD position was ecdcb2d Added standard HTML page tags
Switched to branch 'main'
```

add bad comment to revert

C: > DownloadedApps > Git > work > <> hello.html > ...



```
1  <html>
2    <head>
3    </head>
4    <body>
5      <h1>Hello, World!</h1>
6      <!-- This is a bad comment. We want to revert it. -->
7    </body>
8  </html>
9  |
```

```
brw1a@AutumnPC MINGW64 /work (main)
$ git status
On branch main
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.html

no changes added to commit (use "git add" and/or "git commit -a")
```

restore command

```
brw1a@AutumnPC MINGW64 /work (main)
$ git restore hello.html
git status
cat hello.html
On branch main
nothing to commit, working tree clean
<html>
  <head>
</head>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
```

C: > DownloadedApps > Git > work >  hello.html >  html

```
1  <html>
2    <head>
3      <!-- This is an unwanted but staged comment -->
4    </head>
5    <body>
6      <h1>Hello, World!</h1>
7    </body>
8  </html>
```

```
brw1a@AutumnPC MINGW64 /work (main)
$ git add hello.html

brw1a@AutumnPC MINGW64 /work (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html
```

```
brw1a@AutumnPC MINGW64 /work (main)
$ git restore --staged hello.html
```

back to clean state

```
brw1a@AutumnPC MINGW64 /work (main)
$ git restore hello.html
git status
On branch main
nothing to commit, working tree clean
```