📚 آموزش کامل Git از صفر تا پیشرفته

.1 نصب و تنظیمات اولیه

1.1 نصب Git

- دانلود ازhttps://git-scm.com/downloads

- همه گزینه‌ها را در نصب پیش‌فرض بگذارید

1.2 تنظیم هویت

git config --global user.name "نام شما"

git config --global user.email "ایمیل شما"

2. شروع یک پروژه جدید

2.1 ساخت پوشه پروژه

mkdir my-project

cd my-project

2.2 تبدیل پوشه به ریپوی Git

git init

3. کار با فایل‌ها

3.1 ساخت فایل جدید

echo "# پروژه من" > README.md

3.2 بررسی وضعیت

git status

3.3 اضافه کردن فایل‌ها

git add README.md

# یا برای اضافه کردن همه فایل‌ها:

git add .

3.4 ثبت تغییرات (کامیت)

git commit -m "اولین کامیت"

4. اتصال به GitHub

-ساخت ریپوی جدید در GitHub

-به github.com/new بروید

-نام ریپو را وارد کنید )مثلاً (my-project

Create repository -را بزنید

4.2 اتصال ریپوی محلی به GitHub

git remote add origin https://github.com/نام-کاربری/my-project.git

5. ارسال تغییرات به GitHub (Push)

git push -u origin master

یا اگر از main استفاده می‌کنید:

git push -u origin main

6. دریافت تغییرات از GitHub (Pull)

git pull origin master

7. همگام‌سازی (Sync)

git pull origin master

git push origin master

8. برگرداندن تغییرات (Revert)

8.1 مشاهده تاریخچه کامیت‌ها

git log --oneline

8.2 برگرداندن یک کامیت خاص

git revert کامیت-ایدی

📌 توضیح مفاهیم اصلی:

:Push ارسال تغییرات از کامپیوتر شما به GitHub

:Pull دریافت تغییرات از GitHub به کامپیوتر شما

:Sync ترکیب Pull و Push برای همگام‌سازی کامل

:Revert برگرداندن یک تغییر خاص بدون حذف تاریخچه

🧩 آموزش کامل Git و GitHub - از صفر تا صد

.1 نصب) Git اولین قدم(

چرا نیاز داریم؟

Git یک سیستم کنترل نسخه است که تغییرات فایل‌های شما را ثبت و مدیریت می‌کند.

چگونه نصب کنیم؟

.1 به سایت https://git-scm.com/downloads بروید

.2 نسخه مناسب سیستم عامل خود را دانلود کنید

.3نصب را با تنظیمات پیش‌فرض انجام دهید

بررسی نصب

git --version

اگر نسخه Git را نشان داد، یعنی درست نصب شده است.

2. تنظیمات اولیه Git

چرا نیاز داریم؟

برای اینکه Git بداند چه کسی تغییرات را ایجاد کرده است.

دستورات:

git config --global user.name "نام شما"

git config --global user.email "ایمیل شما"

مثال:

git config --global user.name "Ali Mohammadi"

git config --global user.email "ali@example.com"

3. شروع یک پروژه جدید

3.1 ساخت پوشه پروژه:

mkdir my-first-project

cd my-first-project

3.2 تبدیل پوشه به ریپوی Git:

git init

این دستور یک پوشه مخفی به نام `.git` می‌سازد که تمام اطلاعات نسخه‌بندی در آن ذخیره می‌شود.

4. کار با فایل‌ها

4.1 ساخت فایل جدید:

echo "# پروژه اول من" > README.md

4.2 بررسی وضعیت:

git status

این دستور نشان می‌دهد:

- چه فایل‌هایی تغییر کرده‌اند

- چه فایل‌هایی آماده کامیت هستند

3.4 اضافه کردن فایل به مرحله :Staging

git add README.md

یا برای اضافه کردن همه فایل‌ها:

git add .

Staging یعنی آماده کردن تغییرات برای ثبت دائم

4.4 ثبت تغییرات (کامیت):

```bash

git commit -m "اضافه کردن فایل README"

```

کامیت مانند یک عکس فوری از پروژه شما در این لحظه است.

## 5. اتصال به GitHub

\*\*5.1 ساخت ریپوی جدید در GitHub:\*\*

1. به [github.com](https://github.com) وارد شوید

2. روی + کلیک کنید و "New repository" را انتخاب کنید

3. نام ریپو را وارد کنید (مثلاً my-first-project)

4. Create repository را بزنید

\*\*5.2 اتصال ریپوی محلی به GitHub:\*\*

```bash

git remote add origin https://github.com/نام-کاربری/my-first-project.git

```

این دستور یک ارتباط بین پروژه محلی و GitHub ایجاد می‌کند.

## 6. ارسال تغییرات به GitHub (Push)

```bash

git push -u origin master

```

یا اگر از شاخه main استفاده می‌کنید:

```bash

git push -u origin main

```

\*\*توضیح:\*\*

- `push`: ارسال تغییرات به سرور

- `-u`: ذخیره تنظیمات برای دفعات بعد

- `origin`: نام مخزن دور

- `master/main`: نام شاخه

## 7. دریافت تغییرات از GitHub (Pull)

```bash

git pull origin master

```

این دستور تغییرات از GitHub را دریافت و با نسخه محلی ادغام می‌کند.

## 8. همگام‌سازی (Sync)

همگام‌سازی یعنی هم دریافت تغییرات و هم ارسال تغییرات:

```bash

git pull origin master # اول تغییرات جدید را بگیر

git push origin master # سپس تغییرات خود را بفرست

```

## 9. برگرداندن تغییرات (Revert)

\*\*9.1 مشاهده تاریخچه کامیت‌ها:\*\*

```bash

git log --oneline

```

خروجی چیزی شبیه این خواهد بود:

```

a1b2c3d (HEAD -> master) اضافه کردن فایل جدید

e4f5g6h ایجاد فایل README

```

\*\*9.2 برگرداندن یک کامیت خاص:\*\*

```bash

git revert a1b2c3d

```

این دستور تغییرات آن کامیت را برمی‌گرداند اما تاریخچه را حذف نمی‌کند.

## 📝 خلاصه دستورات مهم:

| دستور | توضیح | مثال |

|-------|-------|------|

| `git init` | شروع یک ریپوی جدید | `git init` |

| `git add` | اضافه کردن فایل به مرحله staging | `git add .` |

| `git commit` | ثبت تغییرات | `git commit -m "پیام"` |

| `git push` | ارسال تغییرات به سرور | `git push origin master` |

| `git pull` | دریافت تغییرات از سرور | `git pull origin master` |

| `git revert` | برگرداندن یک کامیت | `git revert a1b2c3d` |

## 🧠 مفاهیم کلیدی:

1. \*\*Repository (ریپو)\*\*: پوشه پروژه شما که توسط Git مدیریت می‌شود

2. \*\*Commit (کامیت)\*\*: یک نسخه از پروژه شما در یک زمان خاص

3. \*\*Branch (شاخه)\*\*: خط توسعه مستقل (پیش‌فرض master/main)

4. \*\*Remote (ریموت)\*\*: نسخه سروری پروژه (معمولاً روی GitHub)

## 🚀 مسیر یادگیری پیشنهادی:

1. ابتدا فقط با `git add`, `git commit`, `git push` کار کنید

2. بعد از تسلط، `pull` و `sync` را یاد بگیرید

3. در نهایت `revert` و سایر دستورات پیشرفته را تمرین کنید

اگر در هر مرحله سوالی داشتید یا چیزی نامفهوم بود، خوشحال می‌شوم کمک کنم! 😊