



Git & GitHub

المحاضرة الأولى

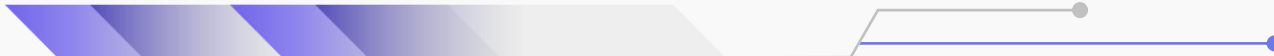
م. محمد السلوم





أساسيات Git

ما هي Git	مرحلة الادراج Staging في Git
كيفية نشأة Git ولماذا	إيداع المساهمات Committing في Git
كيفية تنصيب وتثبيت Git	إيداع نفس الملفات
تعريف حساب Git محلي	الإطلاع على سجل الايداعات
البدء باستخدام Git	التراجع عن عملية إيداع في Git
حالات الملفات في Git	





ما هي Git

01

هو نظام لإدارة الإصدارات الموزعة (Distributed Version Control System) ، والذي يتيح للمطورين تتبع التغييرات في شفرة المصدر خلال مراحل تنفيذ المشروع المختلفة والتعاون مع الآخرين على نفس المشروع وهي أداة قوية ومهمة لكل مطور برمجيات يسعى لإدارة مشاريعه بكفاءة وتعاون فعال مع زملائه



كيفية نشأة Git ولماذا

02

نشأة Git كانت نتيجة للحاجة إلى أداة أفضل لإدارة الإصدارات في تطوير البرمجيات. في البداية، كان مجتمع لينكس (Linux) يستخدم نظام إدارة الإصدارات المركزي يسمى BitKeeper ومع ذلك، في عام 2005، نشأ خلاف بين مجتمع لينكس ومطوري BitKeeper، مما أدى إلى سحب الأخير لرخصة الاستخدام المجانية التي كانت تستخدمها لينكس. نظرًا للحاجة الملحة لنظام إدارة إصدارات جديد ومفتوح المصدر، قرر لينوس تورفالدس مبتكر نواة لينكس



كيفية نشأة Git ولماذا

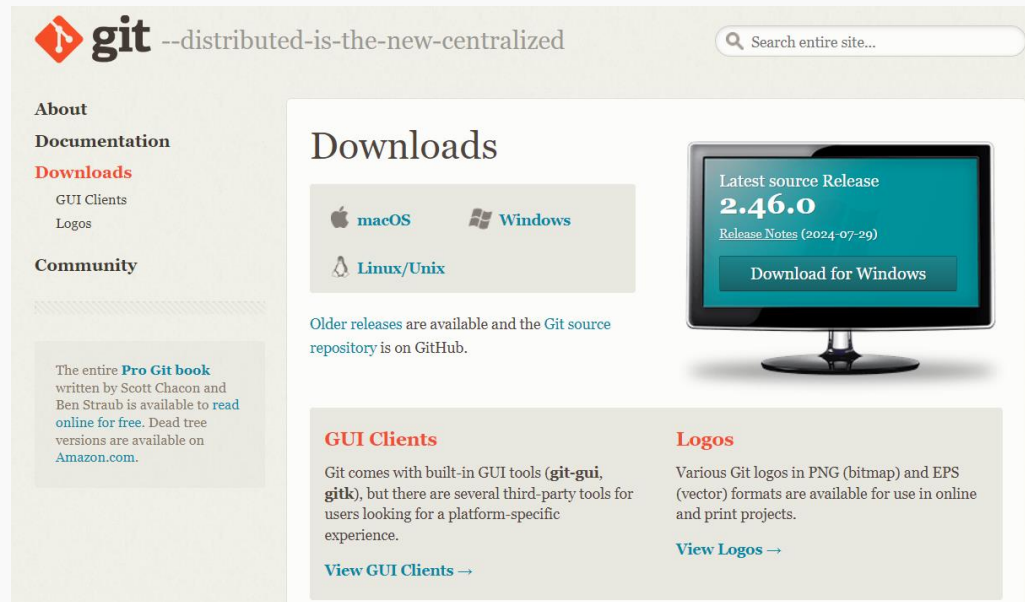
02

تطوير نظام جديد يلبي احتياجات المجتمع وكان الهدف من هذا النظام الجديد أن يكون:
موزعًا بالكامل: كل مطور يمتلك نسخة كاملة من المشروع مع تاريخه الكامل، مما يزيد من
المرونة والقدرة على العمل بدون اتصال بالإنترنت.
سريعًا وفعالًا: ينبغي أن يكون النظام قادرًا على التعامل مع المشاريع الكبيرة ومعالجة
التغييرات بسرعة.
أمان البيانات: ضمان سلامة ودقة البيانات مع إمكانيات التتبع القوي للتغييرات
والتراجع عنها ومعرفة من قام بهذه التعديلات وإمكانية الغاء التعديلات



تنصيب Git

03



The screenshot shows the Git website's Downloads page. The header features the Git logo and the tagline "--distributed-is-the-new-centralized". A search bar is located in the top right. The left sidebar contains links for "About", "Documentation", "Downloads" (highlighted), "GUI Clients", "Logos", and "Community". The main content area is titled "Downloads" and includes a section for "Latest source Release 2.46.0" with a "Download for Windows" button. Below this, there are links for "macOS", "Windows", and "Linux/Unix". A section for "Older releases" mentions that they are available on GitHub. At the bottom, there are sections for "GUI Clients" and "Logos", both with "View" links.

git --distributed-is-the-new-centralized

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

Latest source Release
2.46.0
Release Notes (2024-07-29)
Download for Windows

macOS Windows Linux/Unix

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

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 →](#)

تختلف آلية تنصيبه من نظام تشغيل الى آخر
بالنسبة لنظام التشغيل Windows نقوم
بتحميله من الموقع الرسمي عبر الرابط التالي

<https://git-scm.com/downloads>



تعريف حساب محلي

04

قبل أن نبدأ باستخدام git يفضل أن نقوم بمعرفة اصدار git و بتعريف الحساب الخاص بنا على git، و ذلك من خلال تنفيذ هذه الاوامر:

```
git -v
```

```
git config --global user.name "UserName"
```

```
git config --global user.email "Email"
```

والان للتحقق ان المعلومات الخاصة بالمستخدم تم تعيينها بنجاح نستخدم الامر:

```
git config --global user.name
```

```
git config --global user.email
```



تعريف حساب محلي

04

MINGW64; c:/Users/MOHAMD-SALOUM

```
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~  
$ git config --global user.name "MohamadSaloum"  
  
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~  
$ git config --global user.email "mohamdsaloum667@gmail.com"  
  
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~  
$ git config --global user.name  
MohamadSaloum  
  
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~  
$ git config --global user.email  
mohamdsaloum667@gmail.com  
  
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~  
$ |
```

تأكد من تغيير اسم المستخدم والبريد الإلكتروني

إلى البيانات الخاصة بك

وبسبب استعمالنا الوسيط العام global فهذا

يعني أن الحساب سيكون هو الحساب المستخدم

في كل المستودعات بهذا الحاسوب



تعريف حساب محلي

04

إذا لم تستخدم الخيار `global--`، فإن الإعدادات ستُطبق فقط على المستودع الحالي الذي تعمل عليه، وهذه الإعدادات ستُحفظ في ملف التكوين المحلي (`git/config`) داخل المستودع.

ملخص الخيارات في `Git config` :

`global --` : إعدادات عامة لجميع المستودعات الخاصة بالمستخدم الحالي.

بدون أي خيار: إعدادات خاصة بالمستودع الحالي فقط.

`system --` : إعدادات عامة للنظام بأكمله (نادر الاستخدام ويتطلب صلاحيات المسؤول).



بدء استخدام Git

05

و الآن، يمكننا البدء باستخدام git لإدارة إصدارات المشروع الذي نعمل عليها حاليا او ذلك من خلال الانتقال الى مجلد المشروع، وادخال الامر التالي كما تبين الصورة أدناه، و ذلك لتحويله الى مستودع git، حيث أننا نستخدم مصطلح "مستودع Repository" للإشارة الى أي مشروع تدار إصداراته باستخدام git، و أمر التهيئة هو

```
git init
```

و بهذا نكون قد قمنا بانشاء أوّل مستودع git بشكل ناجح



بدء استخدام Git

05

```
MINGW64:/c:/Users/MOHAMD-SALOUM/Desktop/test
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~
$ cd Desktop/

MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop
$ cd test

MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test
$ git init
Initialized empty Git repository in C:/Users/MOHAMD-SALOUM/Desktop/test/.git/

MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)
$ |
```

او يمكننا اختيار المجلد ومن ثم النقر
بزر الفأرة الأيمن
ثم نقوم بفتحه عن طريق git bash
ثم كتابة الامر git init



حالات الملفات في Git

06

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <h1> Welcome In Our Session </h1>
</body>
</html>
```

قمنا بإنشاء مستودع git محليا في
الفقرة السابقة ولا يحتوي على اي
ملفات فلذلك الان سنقوم بإنشاء
ملف جديد وليكن باسم main.html
ولنقم بتجربة حفظ الملف وإضافته



حالات الملفات في Git

06

نعود الان الى أوامر git ولكي نستعرض الملفات الموجودة في مجلدنا الجديد فسوف نرى الملف main.html كما في الصورة التالية :

```
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)
$ dir
main.html
```

مع الانتباه الى ان الامر **dir** هو في نظام التشغيل Windows بينما في أنظمة linux and Mac فيستخدم الامر **ls** لاستعراض الملفات الموجودة بالمجلد



حالات الملفات في Git

06

والان نستطيع التحقق من حالة تتبع الملفات باستخدام الامر التالي:

```
git status
```

والامر الذي سنستعمله كثيرا حيث يستعمل هذا الامر لكي نرى الوضع الحالي لملفات المشروع

بالنسبة ل git وفي مثالنا الملف جديد فسيظهر لنا انه ملف جديد ولم يتم تتبعه بعد

Untracked ولم يتم الاحتفاظ باي إصدار منه كما هو موضح بالصورة :

```
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)
$ git status
On branch master

No commits yet

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



حالات الملفات في Git

06

بالنسبة لل git فتكون الملفات في المستودعات بإحدى حالتين :

□ **Tracked**: هذه هي الملفات التي تم تضمينها في نظام التحكم في الإصدارات الخاص بـ Git

يمكن أن تكون هذه الملفات في واحدة من ثلاث حالات فرعية:

1 - غير معدلة 2- معدلة 3- محضرة

□ **Untracked** : هي الملفات التي لم يتم تضمينها في نظام التحكم في الإصدارات الخاص بـ

Git لم يتم إضافتها بعد إلى منطقة التحضير أو الالتزام بها. Git يعرف بوجودها، لكنها

ليست تحت إدارته بعد.



مرحلة الادراج Staging

07

مرحلة الإدراج في Git، والتي تُعرف أيضًا بمنطقة التحضير (staging area) أو الفهرس (index)، هي منطقة وسيطة حيث يمكنك تجميع التغييرات التي ترغب في تضمينها في الالتزام التالي (commit) تساعد هذه المرحلة في التحكم بشكل دقيق في ما يتم تضمينه في الالتزام، مما يمنحك مرونة أكبر في إدارة التغييرات.

حيث لا يمكننا نقل الملف أو تغييراته من حالة عدم التتبع الى حالة التتبع مباشرة دون المرور بمرحلة الإدراج.



مرحلة الادراج Staging

07

والان لنقوم بإضافة الملف الخاص بنا الى مرحلة الادراج باستخدام الامر التالي :

```
git add main.html
```

كما هو موضح بالصورة ادناه :

```
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)  
$ git add main.html
```

هذا الأمر يضيف التغييرات التي أجريتها على الملف main.html إلى منطقة التحضير.



مرحلة الادراج Staging

07

التحقق من الملفات في منطقة التحضير:

لمعرفة أي الملفات تمت إضافتها إلى منطقة التحضير وأيها لم يتم إضافتها بعد، يمكنك استخدام الأمر `git status` سيعرض هذا الأمر قائمة بالملفات المحضرة للالتزام والملفات التي تم تعديلها ولكن لم تتم إضافتها بعد

```
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   main.html
```



مرحلة الادراج Staging

07

ملاحظة : يمكننا اضافة أكثر من ملف واحد الى حالة الادراج دفعة واحدة من خلال الامر

```
git add all
```

حيث أن الوسيط all يعني جميع الملفات. كما يمكننا اضافة مجلد كامل الى حالة الادراج من خلال كتابة اسم المجلد بدلا من اسم الملف عند تنفيذ تعليمة add
كما لاضافة جميع التغيرات نستخدم الامر :

```
git add .
```



مرحلة الادراج Staging

07

وفي حال قمنا برفع عدة ملفات الى منطقة Staging area وقبل عملية ال Committing اردنا ان نزيل احد هذه الملفات من منطقة الادراج لسبب ما نستخدم الامر التالي :

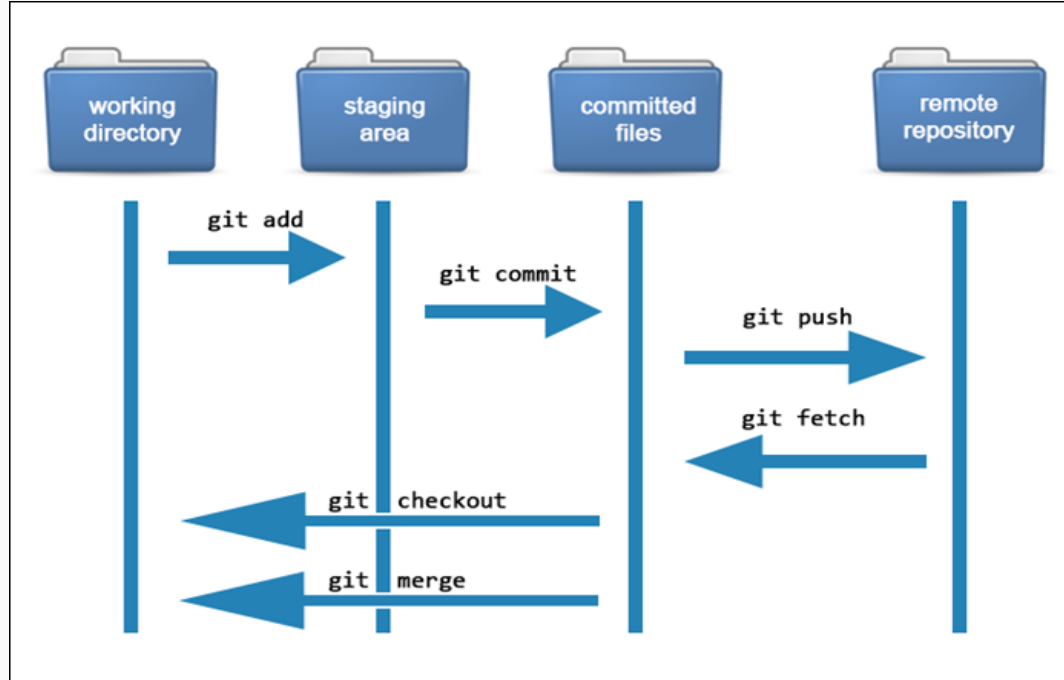
```
git reset head fileName
```

تاكد من وضع اسم الملف الذي تريد استبداله



مرحلة الادراج Staging

07





إيداع المساهمات Committing

08

يعتمد git في ادارة الاصدارات على مفهوم "المساهمات commits" وهي مجموعة من التعديلات التي تتم على الملفات و التي تحفظ ككتلة واحدة في شجرة تاريخ git، بحيث تعتبر كل محطات سير المشروع و يتيح git العودة الى أية محطة مستقبلا على أية حال، لإيداع مساهمة يجب أو لا ادراج جميع التغييرات المطلوب ايداعها، ثم وبعد الأدرج Staging يتم الإيداع في المستودع من خلال الامر commit مع مراعاة ترك رسالة مناسبة ترافق عملية الإيداع و تبين ما تم فيها بشكل مختصر عادة لسهولة المراجعة من قبل المساهم نفسه أو زملائه في فريق العمل.



إيداع المساهمات Committing

08

الان لايداع العمل الذي قمنا به ومع إضافة رسالة نقوم بتنفيذ الامر التالي :

```
git commit -m "FirstCommit"
```

الان تمت عملية الإيداع ومعها الرسالة " الإيداع الأول " وال git تقوم بمنح ترقيم فريد لكل

عملية إيداع ويسمى بمعرف الإيداع Commit ID كما هو موضح في الصورة :

```
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)
$ git commit -m "FirstCommit"
[master (root-commit) 805db08] FirstCommit
1 file changed, 11 insertions(+)
create mode 100644 main.html
```

إيداع الملفات مرة أخرى

09

الان بعد عملية الإيداع لأول مرة سوف نقوم بتغيير ضمن الملف main.html بإضافة أي شي وذلك لمشاهدة ال git كيف سيقوم بالتعرف على التغيير الحاصل عند تنفيذ الامر git status

كما هو موضح في الصورة :

```
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)
$ git status
On branch 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:   main.html
```

كلمة modified والتي تعني أن الملف مدرج سابقا لكنه الآن خضع لتعديل جعل نسخته الحالية

معدّلة modified بمعنى أنّها مختلفة عن آخر ادراج تم لها في git



إيداع الملفات مرة أخرى

09

يمكننا الآن تكرار نفس خطوات عملية الإيداع السابقة، و اجراء ايداع ثان لنفس الملف مع رسالة تفيد بسبب الإيداع كما تبينه الصورة التالية:

```
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)
$ git add main.html

MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)
$ git commit -m "SecoundCommit"
[master 4408f7b] SecoundCommit
1 file changed, 1 insertion(+)
```



الاطلاع على سجل الإيداع

10

الآن وكوننا قمنا بعمل ايداعين في مستودعنا، ربما يكون الوقت مناسب للاطلاع على الإيداعات التي

`git log`

تمت بالفعل في المستودع من خلال تنفيذ الامر :

و الذي يعرض معرفّات عمليات الإيداعات السابقة و تاريخ كل منها و الرسالة التي تبين سبب

الإيداعات كما أدخلها المستخدم الذي قام بعملية الإيداع كما يظهر في الصورة أدناه مع ملاحظة أن

ترتيب ظهور عمليات الإيداع عكسي من الاحداث الى الاقدم كما في الصورة:

```
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)
$ git log
commit 4408f7bf6066721435e74f8cbac6a82a2646b473 (HEAD -> master)
Author: MohamadSaloum <mohamdsaloum667@gmail.com>
Date: Sat Aug 3 16:10:40 2024 +0300

    SecoundCommit

commit 805db08441d08c6361210623c543ff971efdc577
Author: MohamadSaloum <mohamdsaloum667@gmail.com>
Date: Sat Aug 3 15:50:07 2024 +0300

    FirstCommit
```



التراجع عن عملية إيداع

10

التراجع عن عملية ايداع في git يحدث أحيانا أن نقوم بعملية ايداع ما أو أكثر، ثم ولسبب أو لآخر نكتشف أننا بحاجة للتراجع عن عملية الإيداع تلك، تتيح git التراجع عن عمليات الإيداع بأسلوبين اثنين أحدهما بسيط و سنناقشه في هذه الفقرة، والآخر متقدم سنناقشه في فصل لاحق الامر البسيط فهو أمر **git revert** والذي يقوم بالتراجع عن عملية ايداع معينة من خلال القيام بعملية ايداع أخرى معاكسة تماما بحيث تكون النتيجة العودة بالملفات الى حالتها التي كانت عليها قبل عملية الإيداع الاساسية التي نرغب بالتراجع عنها

التراجع عن عملية إيداع

10

لكي تكون الأمور بشكل أوضح نقوم بالتراجع عن اخر عملية إيداع قمنا بها وهي العملية او الإيداع الثاني والتي بها اضعفنا بعض الاسطر الى الملف الخاص بنا والذي قمنا بإنشائه مسبقا ولكي نقوم بالتراجع نحتاج معرف عملية الإيداع Commit Id المطلوب التراجع عنها ويمكن ان نقوم بجلبه عن طريق نسخه من ناتج تعليمة git log كما يظهر في الصورة :

```
MOHAMD-SALOUM@DESKTOP-K2UL453 MINGW64 ~/Desktop/test (master)
$ git log
commit 4408f7bf6066721435e74f8cbac6a82a2646b473 (HEAD -> master)
Author: MohamadSaloum <mohamdsaloum667@gmail.com>
Date: Sat Aug 3 16:10:40 2024 +0300

    SecoundCommit

commit 805db08441d08c6361210623c543ff971efdc577
Author: MohamadSaloum <mohamdsaloum667@gmail.com>
Date: Sat Aug 3 15:50:07 2024 +0300

    FirstCommit
```



التراجع عن عملية إيداع

10

الملف قبل عملية التراجع عن الإيداع كما هو موضح بالصورة ادناه :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-s
  <title>Document</title>
</head>
<body>
  <h1> Welcome In Our Session </h1>
  <h2>The First Session</h2>
</body>
```

ثم نقوم بتنفيذ الامر التالي مع استبدال " Commit Id " بمعرف عملية الإيداع:

```
git revert "Commit ID"
```



التراجع عن عملية إيداع

10

الملف بعد عملية التراجع عن الإيداع كما هو موضح بالصورة ادناه :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Document</title>
</head>
<body>
  <h1> Welcome In Our Session </h1>
</body>
</html>
```

بمجرد ادخال اول 8 محارف من معرف العملية سوف يتعرف عليها git في حال لم ترد نسخ كامل

المعرف



والحمد لله رب العالمين