



git + GitHub

اعداد :

م. أحمد حاج محمد سلامة


الجامعة الدولية للعلوم والنهضة




# ماذا سوف نتعلم في هذا المقرر

## مواضيع متقدمة في Git

ملف التجاهل GitIgnore 

السفر في الزمن Git Checkout 

الامر Reset 

استخدام Commit –amend 

git blame 




# مواضيع متقدمة في Git



لاشك أنه ستكون أجزاء أو ملفات من المشروع الذي تعمل عليه لا تريد مشاركتها مع غيرك ممن يطلع على شيفرة المشروع، و من أمثلة تلك الملفات:

- ملفات السجلات log files
- الملفات المؤقتة temporary files
- الملفات المخفية.
- الملفات الشخصية.
- ملفات الكلمات السريّة و مفاتيح المصادقة.
- غير ذلك من الملفات ذات الحساسية.

ملف التجاهل GitIgnore 

السفر في الزمن Git Checkout 

الامر Reset 

استخدام Commit –amend 

git blame 



# مواضيع متقدمة في Git

تستطيع أن تطلب من git أن يتجاهل تلك الملفات أو الأجزاء من المشروع التي ينبغي له تجاهلها باستخدام ملف gitignore. لاحظ أن اسم الملف يبدأ بنقطة وبالتالي سيكون مخفياً بشكل افتراضي في أنظمة لينكس و ماك أو إس) و لن يتعقب git الملفات و المجلدات المحددة في gitignore. رغم أنه يتتبع ملف gitignore نفسه.

يجب أن يتم إنشاء الملف gitignore في أعلى مستوى من مجلدات المشروع (في المجلد الجذر root بتعبير آخر)، ثم و بعد إنشاء الملف نستخدم أي محرر نصي، و نضيف القواعد التي تحدد الملفات التي نرغب بتجاهلها، و حين نقول قواعد فنحن نعني ذلك لأننا قد نكون في حالات نحتاج فيها لتجاهل الكثير من الملفات دفعة واحدة كتلك التي لها امتداد محدد مثلاً أو التي تطابق أسماؤها نمطاً معيناً ... إلخ

ملف التجاهل GitIgnore

السفر في الزمن Git Checkout

الامر Reset

استخدام Commit –amend

git blame




# مواضيع متقدمة في Git

فيما يلي القواعد التي نستطيع تعريفها:

Pattern	Explanation/Matches	Examples
	Blank lines are ignored	
<code># text comment</code>	Lines starting with # are ignored	
<code>name</code>	All <i>name</i> files, <i>name</i> folders, and files and folders in any <i>name</i> folder	<code>/name.log</code> <code>/name/file.txt</code> <code>/lib/name.log</code>
<code>name/</code>	Ending with / specifies the pattern is for a folder. Matches all files and folders in any <i>name</i> folder	<code>/name/file.txt</code> <code>/name/log/name.log</code>  <b>no match:</b> <code>/name.log</code>
<code>name.file</code>	All files with the <i>name.file</i>	<code>/name.file</code> <code>/lib/name.file</code>

ملف التجاهل GitIgnore 

السفر في الزمن Git Checkout 

الامر Reset 

استخدام Commit –amend 

git blame 




# مواضيع متقدمة في Git



<code>/name.file</code>	Starting with <code>/</code> specifies the pattern matches only files in the root folder	<code>/name.file</code>  <b>no match:</b> <code>/lib/name.file</code>
<code>lib/name.file</code>	Patterns specifying files in specific folders are always relative to root (even if you do not start with <code>/</code> )	<code>/lib/name.file</code>  <b>no match:</b> <code>name.file</code> <code>/test/lib/name.file</code>
<code>**/lib/name.file</code>	Starting with <code>**</code> before <code>/</code> specifies that it matches any folder in the repository. Not just on root.	<code>/lib/name.file</code> <code>/test/lib/name.file</code>
<code>**/name</code>	All <code>name</code> folders, and files and folders in any <code>name</code> folder	<code>/name/log.file</code> <code>/lib/name/log.file</code> <code>/name/lib/log.file</code>

ملف التجاهل GitIgnore 

السفر في الزمن Git Checkout 

الامر Reset 

استخدام Commit –amend 

git blame 



# مواضيع متقدمة في Git



<code>name?.file</code>	? matches a <b>single</b> non-specific character	<code>/names.file</code> <code>/name1.file</code>  <b>no match:</b> <code>/names1.file</code>
<code>name[a-z].file</code>	[range] matches a <b>single</b> character in the specified range (in this case a character in the range of a-z, and also be numeric.)	<code>/names.file</code> <code>/nameb.file</code>  <b>no match:</b> <code>/name1.file</code>
<code>name[abc].file</code>	[set] matches a <b>single</b> character in the specified set of characters (in this case either a, b, or c)	<code>/namea.file</code> <code>/nameb.file</code>  <b>no match:</b> <code>/names.file</code>
<code>name[!abc].file</code>	[!set] matches a <b>single</b> character, <b>except</b> the ones specified in the set of characters (in this case a, b, or c)	<code>/names.file</code> <code>/namex.file</code>  <b>no match:</b> <code>/namesb.file</code>
<code>*.file</code>	All files with the <code>.file</code> extension	<code>/name.file</code> <code>/lib/name.file</code>
<code>name/</code> <code>!name/secret.log</code>	! specifies a negation or exception. Matches all files and folders in any <code>name</code> folder, except <code>name/secret.log</code>	<code>/name/file.txt</code> <code>/name/log/name.log</code>  <b>no match:</b> <code>/name/secret.log</code>

ملف التجاهل GitIgnore



السفر في الزمن Git Checkout



الامر Reset



استخدام Commit –amend



git blame







# مواضيع متقدمة في Git



<code>*.file</code>	! specifies a negation or exception. All files with the <code>.file</code> extension, except <code>name.file</code>	<code>/log.file</code>
<code>!name.file</code>		<code>/lastname.file</code>
		<b>no match:</b> <code>/name.file</code>
<code>*.file</code>	Adding new patterns after a negation will re-ignore a previous negated file	<code>/log.file</code>
<code>!name/*.file</code>	All files with the <code>.file</code> extension, except the ones in <code>name</code> folder. Unless the file name is	<code>/name/log.file</code>
<code>junk.*</code>	<code>junk</code>	<b>no match:</b> <code>/name/junk.file</code>

ملف التجاهل GitIgnore



السفر في الزمن Git Checkout



الامر Reset



استخدام Commit –amend



git blame







# مواضيع متقدمة في Git



نستطيع باستخدام git أن نعيد عقارب الساعة إلى الوراء مجازياً، بمعنى أن نعيد الملفات الموجودة في المستودع إلى الحالة التي كانت فيها لحظة مساهمة ما تماماً، ذلك من خلال الأمر git checkout ثم ذكر معرّف المساهمة المطلوب العودة إليها.

لقد أنشأت ملف html وملف js باستخدام الامر

Touch <اسم الملف>

```
XPRISTO@DESKTOP-STN36A1 MINGW64 ~/Desktop/test (master)
$ touch index.html
```

```
XPRISTO@DESKTOP-STN36A1 MINGW64 ~/Desktop/test (master)
$ touch main.js
```

ملف التجاهل GitIgnore 

السفر في الزمن Git Checkout 

الامر Reset 

استخدام Commit –amend 

git blame 



# مواضيع متقدمة في Git



بعد تتبع الملفات السابقة و عمل الإيداع

```
XPRISTO@DESKTOP-STN36A1 MINGW64 ~/Desktop/test (master)
$ git add *

XPRISTO@DESKTOP-STN36A1 MINGW64 ~/Desktop/test (master)
$ git commit -m "one-v1"
[master (root-commit) 0102ba9] one-v1
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 index.html
```

استعراض عمليات الإيداع

```
XPRISTO@DESKTOP-STN36A1 MINGW64 ~/Desktop/test (master)
$ git log
commit cdc878669b7bb2f246b3c735e26a7ed2fc5786ec (HEAD -> master)
Author: ahmadSalameh1995 <ahmadsalama998877@gmail.com>
Date:   Fri Aug 25 16:06:05 2023 +0300

    one-v2

commit 0102ba9151dc7884682b6be3f3cf80c1ccda21f2
Author: ahmadSalameh1995 <ahmadsalama998877@gmail.com>
Date:   Fri Aug 25 16:05:29 2023 +0300

    one-v1
```

ملف التجاهل GitIgnore



السفر في الزمن Git Checkout



الامر Reset



استخدام Commit -amend



git blame





# مواضيع متقدمة في Git



للرجوع الى الوراء بالزمن عند عملية إيداع

```
XPRISTO@DESKTOP-STN36A1 MINGW64 ~/Desktop/test (master)
$ git checkout 0102ba9151dc7884682b6be3f3cf80c1ccda21f2
Note: switching to '0102ba9151dc7884682b6be3f3cf80c1ccda21f2'.

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 0102ba9 one-v1

XPRISTO@DESKTOP-STN36A1 MINGW64 ~/Desktop/test ((0102ba9...))
$ git log
commit 0102ba9151dc7884682b6be3f3cf80c1ccda21f2 (HEAD)
Author: ahmadSalameh1995 <ahmadsalama998877@gmail.com>
Date:   Fri Aug 25 16:05:29 2023 +0300

    one-v1

XPRISTO@DESKTOP-STN36A1 MINGW64 ~/Desktop/test ((0102ba9...))
$
```

ملف التجاهل GitIgnore 

السفر في الزمن Git Checkout 

الامر Reset 

استخدام Commit -amend 

git blame 



# مواضيع متقدمة في Git



ذكرنا في الفصل الأول أنه توجد طريقة ثانية متقدمة للتراجع عن مساهمة أو أكثر قمنا بها، وقلنا أننا سنغطيها في الفصل المتقدم ، و الآن قد حان الوقت للحدث عنها، ألا وهي طريقة الأمر `reset` يُستخدم الأمر `reset` عند الرغبة في إرجاع المستودع في التاريخ إلى مساهمة سابقة و له ثلاثة أنماط

- النمط الخشن `—hard` :هو أخطر الأنماط حيث أنه يعيد المستودع إلى مساهمة سابقة مع إلغاء جميع المساهمات و التغييرات التي حدثت بعد تلك المساهمة.

**`git reset —hard <commit-id>`**

- النمط الناعم `—soft` : هو شبيه بالسابق إلا أنه لا يلغي المساهمات التالية، بل يبقها و كأنها تمت و تم عمل `add` لها إنما لم يتم عمل `commit` و لهذا اسمه سوفت).

**`git reset —soft <commit-id>`**

ملف التجاهل `GitIgnore`

السفر في الزمن `Git Checkout`

الأمر `Reset`

استخدام `Commit —amend`

`git blame`



# مواضيع متقدمة في Git




النمط الوسطي mixed شبيه بالنمط الناعم مع فارق أنّه يبقى التغييرات وكأنّها حدثت للتو و لم تسجل في تاريخ المستودع بعد، و بالتالي إذا أردت الاحتفاظ بها فيجب أن تقوم بعمل `git add` ثم `git commit`

`git reset --mixed <commit-id>`

بغض النظر عن النمط الذي تريد استخدامه فيجب أولاً أن تقوم بتحديد المساهمة التي ترغب في العودة إليها من خلال تحديد معرفها Commit ID من خلال الامر `git log` او الامر `gitk`

ملف التجاهل GitIgnore 

السفر في الزمن Git Checkout 

الامر Reset 

استخدام Commit –amend 

`git blame` 



# مواضيع متقدمة في Git



تخيل أنك قمت بعمل commit و من ثم نسيت إضافة أحد الملفات لها، أو قمت بعمل commit و أخطأت في كتابة الرسالة وأردت أن تعيد صياغتها مثلاً أو لأي سبب ما، أردت أن تتراجع عن الأمر، عندها نقول أنه في حالة أردت المحاولة مرة أخرى، فقم بتنفيذ commit مع الخيار amend – كالتالي

## **git commit –amend**


بمثال بسيط، تخيل أنك قمت بعمل commit و بعد ذلك وجدت أنك نسيت أحد الملفات، و لنقل file.cpp و أردت أن تضيفه، عندها يمكننا القيام بالتالي بعد commit التي قمنا بها

```
git commit -m 'initial commit'
```

```
git add file.cpp
```

```
git commit --amend
```

ملف التجاهل GitIgnore 

السفر في الزمن Git Checkout 

الأمر Reset 

استخدام Commit –amend 

git blame 





# مواضيع متقدمة في Git





أحياناً نرغب و لسبب ما بمعرفة من هو المساهم الذي تسبب بتغيير سطر ما من ملف ما أو مجموعة أسطر، يتيح لنا git ذلك من خلال الأمر git blame و أرى تسميته طريقة لأن blame باللغة الإنجليزية تستعمل في الحالات السلبية لإلقاء اللوم على أحد ما . صيغة الأمر هي:

```
git blame -L <start>,+<end> -- <file-path>
```

```
XPRISTO@DESKTOP-STN36A1 MINGW64 ~/Desktop/test (master)
$ git blame -L 2,+4 -- asd.js
3beed2c1 (ahmadSalameh1995 2023-08-25 18:11:36 +0300 2) var t=6-5;
3beed2c1 (ahmadSalameh1995 2023-08-25 18:11:36 +0300 3) var x=5*5;
3beed2c1 (ahmadSalameh1995 2023-08-25 18:11:36 +0300 4) var t=6-5;
3beed2c1 (ahmadSalameh1995 2023-08-25 18:11:36 +0300 5) var x=5*5;
```

ملف التجاهل GitIgnore 

السفر في الزمن Git Checkout 

الأمر Reset 

استخدام Commit --amend 

git blame 