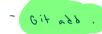
- Git init

Git Init

เป็นคำสั่งที่ใช้สร้างระบบของ Git ขึ้นมาภายใต้โฟลเดอร์หรือ Path นั้น โดยจะสร้าง โฟลเดอร์ .git ขึ้นมาเพื่อใช้เก็บ สำรองข้อมูล การเปลี่ยนแปลงและคุณสมบัติอื่นๆ ของ Git

\$git init



Git Add

เป็นคำสั่งที่ใช้เพิ่มการเปลี่ยนแปลงของ Source Code เข้าไปที่สถานะ Staged

```
$git add <file_name>
$git add README.md #เพิ่มไฟส์ชื่อ README.md เข้าไปที่สถานะ Staged
$git add . #ใช้ในกรณีที่มีหลายๆ ไฟล์และด้องการเพิ่มเข้าไปทั้งหมด
```

Git connit -n

Git Commit

เป็นคำสั่งที่ใช้ยืนยัน Source Code ที่อยู่ในสถานะ Staged เข้าไปเก็บไว้ที่ Local Repository

```
$git commit -m "message" #ขึ้นขันการเปลี่ยนแปลงพร้อมข้อความ
$git commit -am "message" #เพิ่มการเปลี่ยนแปลงและฮันชันหร้อมข้อความ
$git commit #เพิ่มซ้อความในโปรแกรม vi
#ขึ้นขันการเปลี่ยนแปลงพร้อมข้อความและ merge ลงใน commit ล่าสุด
$git commit --amend -m "message"
```

ถ้าต้องการเขียน Commit Message ยาวๆ สามารถใช้คำสั่ง Git commit ระบบจะเปิด โปรแกรม vi ให้เขียน Message [<u>วิธีใช้ vi, vim]</u>



Git Push

เป็นคำสั่งที่ใช้ส่งการเปลี่ยนแปลงของ Source Code ที่เก็บอยู่บน Local Repository ขึ้น ไปยัง Remote Repository

#ส่งการเปลี่ยนแปลง Branch master ไปยัง Remote ที่ชื่อ origin \$git push origin master

- Git stetus

Git Status

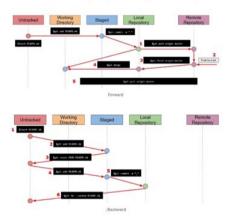
เป็นคำสั่งที่ใช้ตรวจสองสถานะของ Source Code ในระบบของ Git ซึ่งจะแสดงสถานะดั่ง ที่ได้อธิบายข้างต้นไปแล้ว

```
$git status
```

Git Status

สถานะของ Source Code ที่เก็บอยู่ในระบบของ Git นั้นมีดั่งนี้

- Untracked เป็นสถานะที่ Source Code ถูกเพิ่มเข้ามาใหม่และยังไม่ได้ถูกเก็บไว้ใน ระบบของ Git
- Working Directory เป็นสถานะที่กำลังมีการเปลี่ยนแปลงหรือแก้ไข Source Code หรืออาจจะเรียกสถานะนี้ว่า Modified
- Staged เป็นสถานะที่ Source Code กำลังเตรียมที่จะ Commit เพื่อยืนยันการ เปลี่ยนแปลงก่อนที่จะเก็บลง ในสถานะ Local Repository
- Local Repository เป็นสถานะที่มีการเก็บบันทึกข้อมูลการเปลี่ยนแปลงของ Source Code ลงไปที่ Git Repository ที่เป็น Local (ที่เครื่องตัวเอง)
- Remote Repository เป็นสถานะที่มีการเก็บบันทึกข้อมูลการเปลี่ยนแปลงของ Source Code ลงไปที่ Git Repository ที่เป็น Hosting (ที่เครื่องเชิร์ฟเวอร์)



Git Log

เป็นคำสั่งที่ใช้แสดงประวัติของ Commit ที่เก็บไว้ใน Repository

- Git Ing

```
$git log --oneline
$git log --oneline --decorate
$git log --oneline --decorate
$git log --oneline --decorate --graph

$git log --stat #Diff from log
$git log --grep="Message" #log by message
$git log --after="2017-2-14" #log in date
$git log --before="2017-2-14" #Log in date
$git log --author=pakin #Log by user
```

- Git ignore (gitignore)

Git Ignore [.gitignore]

Git Ignore ไม่ได้เป็นคำสั่งแต่เป็นคุณสมบัติของ Git โดยการเพิ่มไฟล์ที่ชื่อ .gitignore เข้าไปในระบบของ Git เพื่อทำการบอกให้ Git ไม่ต้องสนใจไฟล์หรือ โฟลเดอร์นั้นๆ เช่น ไฟล์หรือโฟลเดอร์ที่เป็น Output ของการ Build ใน Java (.class) ไฟล์ที่เป็นคุณสมบัติ เฉพาะของ IDE หรือ Working Space ก็ไม่ควรแชร์ไปให้คนอื่นๆ ในทีม

```
$touch .gitignore #สร้างไฟล์ .gitignore
#เพิ่ม String เจ้าไปในไฟล์ .gitignore เพื่อ ignore ไฟล์ .class ทั้งหมด และ
โฟลเดอร์ Debug, Build
Secho >> .gitignore "%-class"
$echo >> .gitignore "/Debug"
$echo >> .gitignore "/Dulld"
$git add .gitignore #เพิ่มไฟล์ชื่อ .gitignore เข้าไปที่สถานะ Staged
$git commit -m "Add .gitignore file"
```

ในกรณีที่มีการเพิ่มไฟล์ที่ไม่ต้องการเข้าไปยังสถานะ Staged แล้ว และเพิ่มไฟล์ .gitignore เข้าไปทีหลัง สามารถใช้คำสั่งนี้เพื่อลบไฟล์หรือโฟล์เตอร์ที่ไม่ต้องการออก จากสถานะ Staged และ Commit Apply .gitignore เข้าไปอีกครั้ง

```
$git rm —cached <file name>
$git rm —cached <path to file>
$git rm —cached .class
$git rm —cached bebug/*
$git rm —cached Build/*
$git rm —r —cached Build/*
$git rm -r —cashed *
$git commit —am "apply .gitignore file"
#ผมคาไฟฟ์ที่ Track ไว้ในขะบบของ Git ไฟฟ์ที่ ignore จะพายไปแม้จะยังอยู่ในโฟฟ์เดอร์
$git ls-files
```

Git Ignore ที่มีการรวบรวมไว้บน GitHub <u>https://github.com/github/gitignore</u>



- Git clane

Git Clone

เป็นคำสั่งที่ใช้ดึงประวัติทั้งหมดบน Remote Repository ของเพื่อนร่วมทีม ของคนอื่น หรือของเราเองที่มีอยู่แล้วบน Git Hosting มาที่เครื่องของเรา คำสั่งนี้จะคล้ายๆ Git Init ที่ใช้สร้างระบบ Git ขึ้นมาตอนเริ่มค้น แต่เราจะได้ประวัติเดิมของ Repository มาด้วย ทำให้เราเริ่มพัฒนาต่อจากตรงจุดนี้ได้เลย

\$git clone https://github.com/NewGame0/Android_HelloWorld.git

คำสั่ง Git Clone นั้นจะ Checkout Branch หลักมาเป็น Master และดึง Tag ลงมา ทั้งหมด