**พื้นฐานเกี่ยวกับ version control และ git**

11. ระบบที่คอยจัดการ backup source code ของเรา โดยเก็บเป็นลักษณะ version ต่างๆ เช่น ในกรณีเกิดปัญหาขึ้น ก็สามารถหยิบเอา source code ตัวเก่าที่เคยใช้งานได้มาแทน Version Control ยังเป็นตัวกลางที่ทำให้ source code ของแต่ละเครื่อง (แต่ละ programmer) มี source code ที่ตรงกันด้วย ซึ่งโดยปกติทั่วไปแล้ว programmer มักจะทำโปรแกรมให้เสร็จสมบูรณ์ และใช้งานได้ก่อน จึงค่อยโยนขึ้นไปที่ repository เพื่อให้คนอื่นดึงไปใช้ต่อไป

12. **Distributed Version Control Systems (DVCSs) ข้อได้เปรียบ**

- DVC ลดภาระในหลาย ๆ action ให้ไม่ต้องเชื่อมต่อกับ remote repository ทำให้มีความเร็วมากกว่า

- DVC แบ่งเฟสการทำงานได้ละเอียดคือมี local repository ไว้ในเครื่องเรา ถึงแม้จะ commit ผิดเราก็สามารถแก้ไขได้โดยไม่ต้องเชื่อมต่อไปที่ remote repository ถ้าแก้ไขเสร็จเรียบร้อยก็ค่อย push ทั้งหมดไปทีเดียว

- DVC ไม่ต้องพึ่งอินเตอร์เน็ต ทำไว้เท่าไหร่ก็ได้ มีเน็ตใช้ก็ค่อย push

- การมี working repository ทำให้ DVC สามารถแชร์การเปลี่ยนแปลงให้คนอื่นดูได้โดยที่ไม่ต้อง push ขึ้น remote repository

13. **Distributed Version Control Systems (DVCSs)** สามารถรู้ได้ว่าคนอื่นในโปรเจคกำลังทำอะไร ผู้ควบคุมระบบสามารถควบคุมได้อย่างละเอียดว่าใครสามารถแก้ไขอะไรได้บ้าง การจัดการแบบรวมศูนย์ในที่เดียวทำได้ง่ายกว่าการจัดการฐานข้อมูลใน client แต่ละเครื่องเยอะ

14. ทำการ edit แล้ว commit ไปใหม่นั่นเอง

15. ทำการ merge บ่อย ๆ

16. Git คือ Version Control ตัวหนึ่ง ซึ่งเป็นระบบที่มีหน้าที่ในการจัดเก็บการเปลี่ยนแปลงของไฟล์ในโปรเจ็คเรา มีการ backup code ให้เรา สามารถที่จะเรียกดูหรือย้อนกลับไปดูเวอร์ชั่นต่างๆของโปรเจ็คที่ใด เวลาใดก็ได้ หรือแม้แต่ดูว่าไฟล์นั้นๆใครเป็นคนเพิ่มหรือแก้ไข หรือว่าจะดูว่าไฟล์นั้นๆถูกเขียนโดยใครบ้างก็สามารถทำได้ ฉะนั้น Version Control ก็เหมาะอย่างยิ่งสำหรับนักพัฒนาไม่ว่าจะเป็นคนเดียวโดยเฉพาะอย่างยิ่งจะมีประสิทธิภาพมากหากเป็นการพัฒนาเป็นทีม

ส่วน Github เป็นเว็บเซิฟเวอร์ที่ให้บริการในการฝากไฟล์ Git (ทั่วโลกมักนิยมใช้ในการเก็บโปรเจ็ค Open Source ต่างๆ ที่ดังๆ ไม่ว่าจะเป็น Bootstrap, Rails, Node.js, Angular เป็นต้น)

17. เพื่อแยกการพัฒนาความสามารถใหม่ๆ ออกมาจากส่วนหลัก

18. คือ เป็นพฤติกรรมของการที่แตก Branch ออกไปทำงาน และในการ Merge เข้ามาที่ Branch ใดๆ นั้น Git คำนวณแล้วว่า ไม่มีการเกิด Conflict ระหว่างไฟล์ใดๆ ก็ตามในการ Merge จึงสามารถนำ commit ที่เกิดขึ้นใน Branch ที่แตกออกไปทำงาน มาต่อได้เลย

เพราะ จะได้ตรวจสอบโค๊ด local ของเรานั้นล้าสมัย หรือว่าตามหลัง master บน remote อยู่หรือเปล่า ถ้าตามหลังก็ควรอัพเดทโค๊ดให้ล่าสุดเสมอก็คือการ git merge

19. รวมโค๊ดจาก remote มายัง local โดยที่เราไม่สามารถรู้ได้เลยว่าจะรวมโค๊ดอะไรบ้าง รู้แค่หลังจาก pull เสร็จแล้วนั่นเอง ซึ่งจริงๆแล้ว git pull มันก็คือการทำ git fetch และต่อด้วย git merge อัตโนมัตินั่นเอง

20. วิธีการตั้งเวอร์ชั่นของโปรแกรม