โปรแกรม ruby พื้นฐาน

5.

US\_states.keep\_if {|v| v =~ /[MNO]/}

เลือกเฉพาะขึ้นต้นด้วย M N O ได้ แต่ลงท้ายด้วย aeiou หนูทำไม่ได้อะค่ะ

6.

puts US\_states.reverse!

โปรแกรม Ruby เชิงวัตถุ

10.

pi = 3.14159;

r = 5;

area = pi\*r\*r ;

circumference = 2\*pi\*r ;

puts "Area is #{area}"

puts "Diameter is #{r}"

puts "Circumference is #{circumference}"

พื้นฐานเกี่ยวกับ Version control และ git

11.

การใช้ Version Control System (VCS) เป็นสิ่งที่ชาญฉลาดมาก เพราะมันช่วยให้คุณสามารถย้อนไฟล์บางไฟล์หรือแม้กระทั่งทั้งโปรเจคกลับไปเป็นเวอร์ชั่นเก่าได้ นอกจากนั้นระบบ VCS ยังจะช่วยให้คุณเปรียบเทียบการแก้ไขที่เกิดขึ้นในอดีต ดูว่าใครเป็นคนแก้ไขคนสุดท้ายที่อาจทำให้เกิดปัญหา แก้ไขเมื่อไร ฯลฯ และยังช่วยให้คุณสามารถกู้คืนไฟล์ที่คุณลบหรือทำเสียโดยไม่ตั้งใจได้อย่างง่ายดาย

12.

ถึงแม้ว่าเซิร์ฟเวอร์จะเสีย client ก็ยังสามารถทำงานร่วมกันได้ต่อไป และ repository เหล่านี้ของ client ยังสามารถถูกก๊อปปี้กลับไปที่เซิร์ฟเวอร์เพื่อกูข้อมูลกลับคืนก็ได้ การ checkout แต่ละครั้งคือการทำสำรองข้อมูลทั้งหมดแบบเต็ม ๆ

13.

ทุกคนสามารถรู้ได้ว่าคนอื่นในโปรเจคกำลังทำอะไร ผู้ควบคุมระบบสามารถควบคุมได้อย่างละเอียดว่าใครสามารถแก้ไขอะไรได้บ้าง การจัดการแบบรวมศูนย์ในที่เดียวทำได้ง่ายกว่าการจัดการฐานข้อมูลใน client แต่ละเครื่องเยอะ

14.

การใช้คำสั่ง git rebase --no-ff (P)

ซึ่งวิธีนี้จะช่วยให้ เราสามารถ Merge ลง Master Branch ได้ทันที โดยไม่เกิดการ Conflict กับ (M) / (W) ที่เรา reverted ไป แต่ๆๆ...... จริงๆ มันก็ยังเกิดโอกาส Conflict ได้นะ แต่เป็น Conflict แบบปกติ ที่ถ้าหากมีการเขียนบรรทัดเดียวกัน ก็ต้องแก้ไขปัญหากันไป ไม่ได้เกิดจาก Conflict ที่ Revert ไปมา

ข้อดีของวิธีนี้คือ ประวัติศาสตร์ของแต่ละบรรทัดจะอยู่ครบถ้วน ทำให้วันหลังเราสามารถตรวจสอบได้ง่าย

15.

แนวทางที่ได้รับความนิยมสูงคือ git-flow

16.

Git เป็น revision control แบบ distributed (หมายความว่าไม่มีศูนย์กลาง) และ แบบ non-linear history (หมายความว่ามีประวัติการเปลี่ยนแปลงแบบไม่ใช่เส้นตรง) ดังนั้นทำให้คอนเซปต์ของ Git นั้นต่างจาก revision control รุ่นก่อนหน้าหลายอย่าง

แตกต่างกับ Github อย่างไร?

GitHub คือ เว็บที่ให้บริการพื้นที่จัดเก็บโครงการโอเพ่นซอร์สด้วยระบบควบคุมเวอร์ชันแบบ Git

17.

เพื่อให้งานดำเนินต่อไปได้โดยไม่ต้องรอแก้บั๊กใน master ให้เสร็จก่อน ใน Git เราใช้คำสั่ง branch เพื่อแตก branch ใหม่ออกมา

18.

If Master has not diverged, instead of creating a new commit, git will just point master to the latest

commit of the feature branch.

19.

ใช้ดึงความเปลี่ยนแปลงจาก remote มายัง local และรวมเข้าด้วยกัน (มีค่าเท่ากับ fetch+merge

20.

merge และ branch