



VERSION CONTROL

030513702 : Selected Topics in Software Development
Asst. Prof. Dr. Phollakrit Wongsantisuk

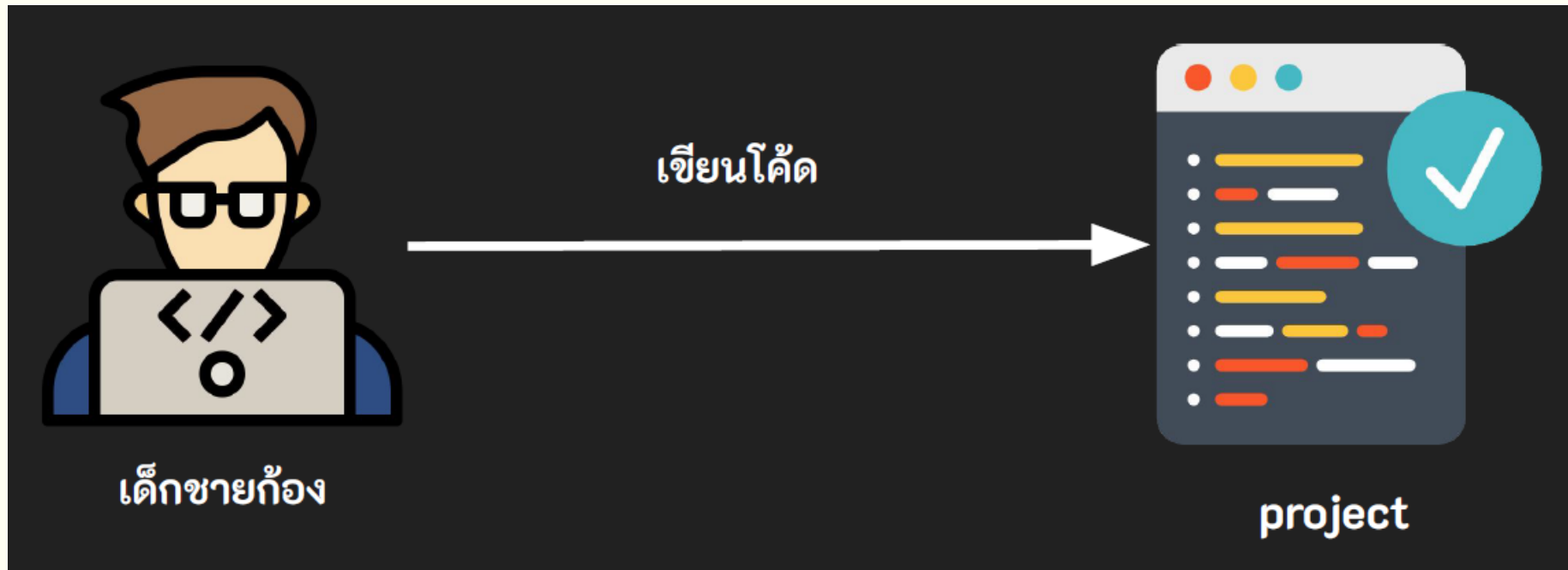
อ้างอิงจาก :

<https://www.youtube.com/c/KongRuksiamOfficial/>

รู้จักกับการจัดเก็บเวอร์ชัน (Version Control)

- **Version Control หรือ Source Control** หมายถึง เครื่องมือช่วยติดตามการเปลี่ยนแปลงของ Source Code โดยการเก็บประวัติการเปลี่ยนแปลงลงฐานข้อมูลชนิดพิเศษ ซึ่งจุดประสงค์ของการเก็บบันทึกทุกการเปลี่ยนแปลง ก็คือ หากมีความผิดพลาดเกิดขึ้น ไม่ว่าจะผิดพลาดเล็กน้อย ไปจนถึงขั้นร้ายแรงที่จะส่งที่จะส่งผลให้ซอฟต์แวร์ที่พัฒนาขึ้นมาขึ้นพังทั้งหมด
- การบันทึกประวัติเก็บไว้อย่างต่อเนื่อง จะช่วยทำให้นักพัฒนาซอฟต์แวร์สามารถย้อนกลับไปในช่วงเวลาต่างๆ โดยเปรียบเทียบโค้ดใหม่กับโค้ดเวอร์ชันก่อนหน้านี้ได้ และตรวจสอบความผิดพลาดเพื่อแก้ไขให้มีผลกระทบต่อทีมพัฒนาน้อยที่สุด คล้ายๆกับการ Undo / Redo ในโปรแกรม (แต่ถ้าหากปิดโปรแกรมไปก็จะไม่สามารถ Undo / Redo อีกรอบได้)

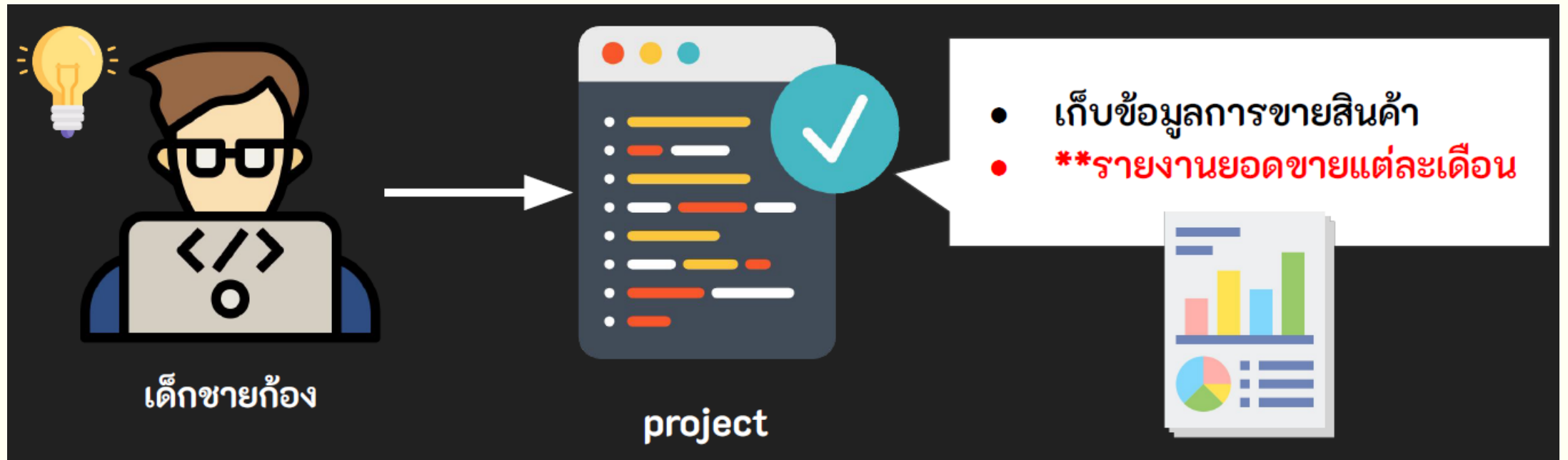
สาเหตุที่ต้องใช้ Version Control



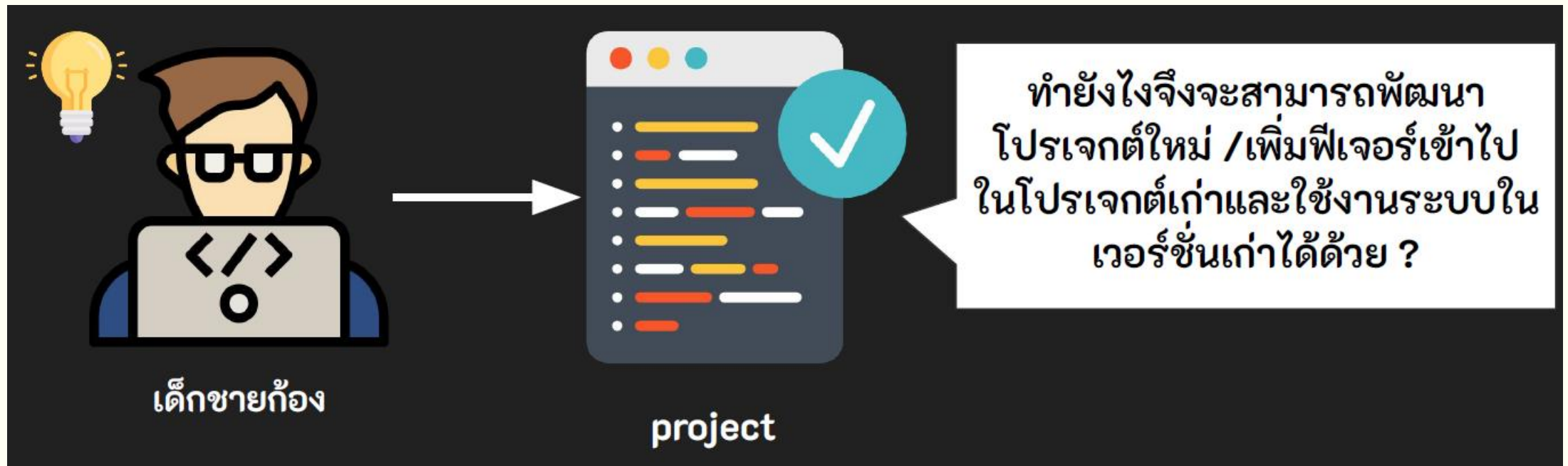
สาเหตุที่ต้องใช้ Version Control



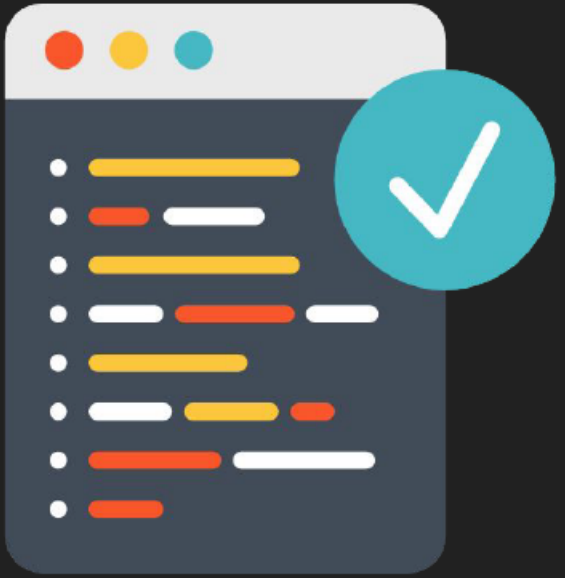
สาเหตุที่ต้องใช้ Version Control



สาเหตุที่ต้องใช้ Version Control

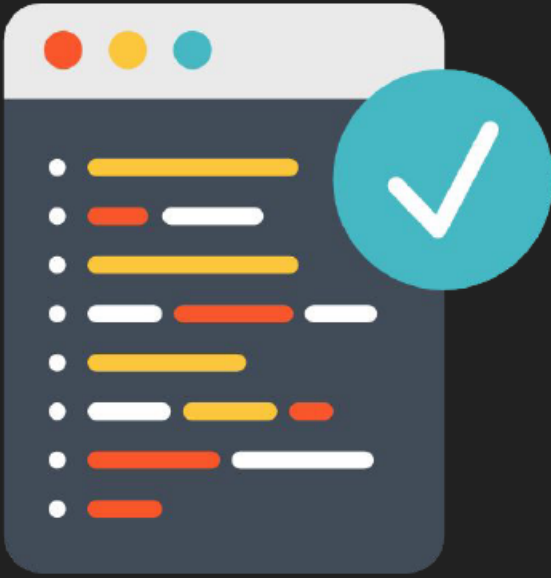


สาเหตุที่ต้องใช้ Version Control



1. Backup (สำรอง) โค้ดที่เขียนเพื่อให้สามารถกลับมาแก้ไขได้ในภายหลังได้

สาเหตุที่ต้องใช้ Version Control



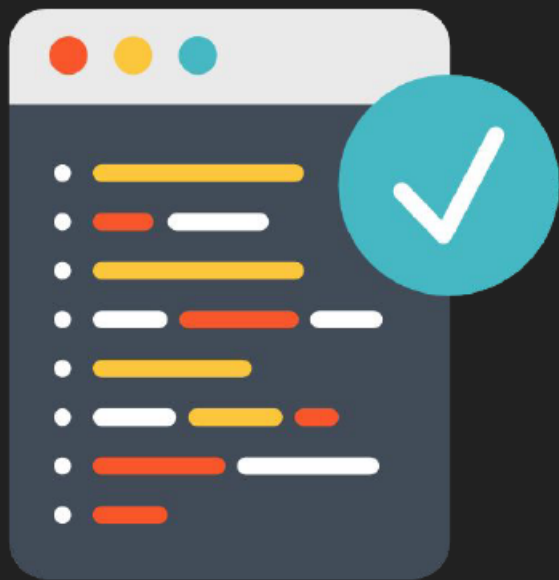
2. คัดลอกไฟล์โค้ดเก่าและสร้าง
ไฟล์ใหม่ทุกครั้งที่มีการแก้ไขโค้ด

สาเหตุที่ต้องใช้ Version Control



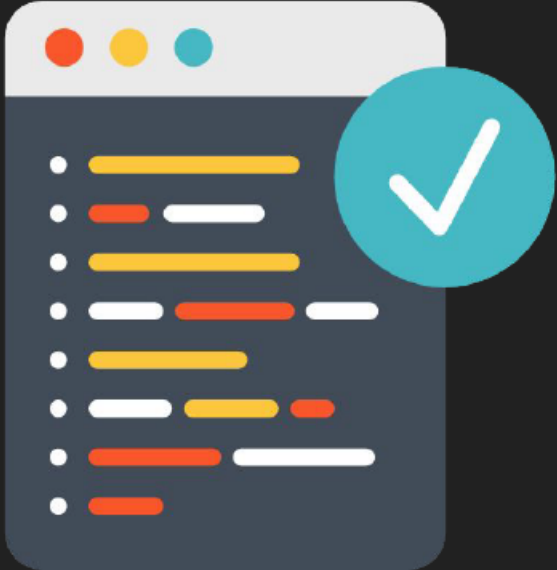
- **project**
- **projectv2**
- **projectv3 fixbug**

สาเหตุที่ต้องใช้ Version Control



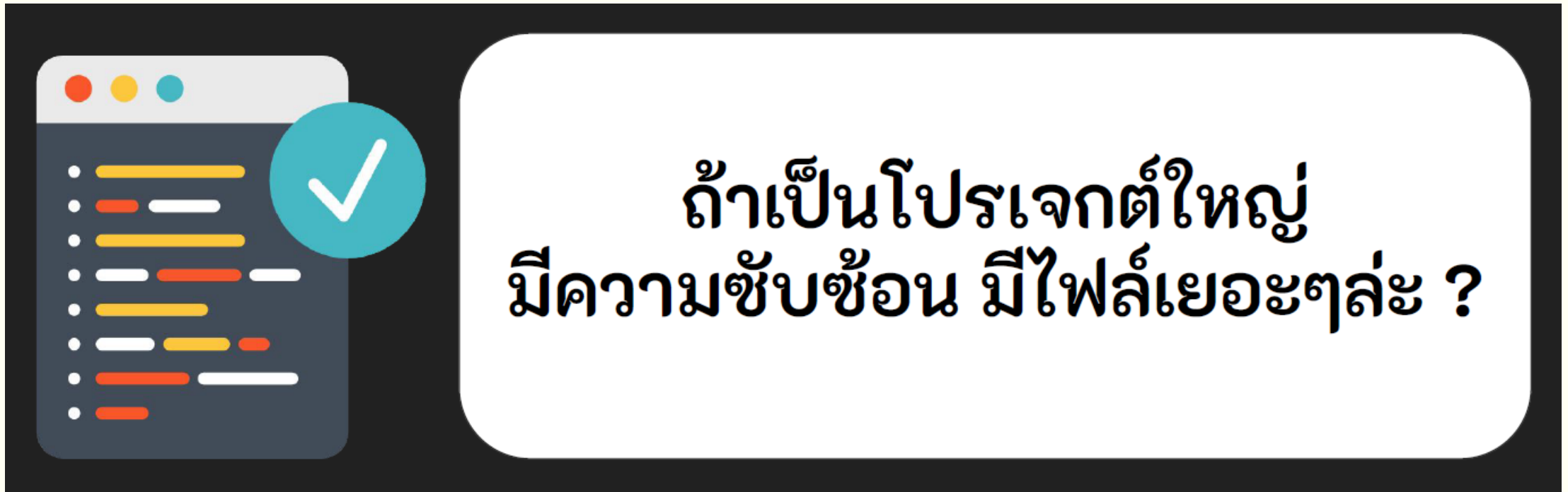
**** เก็บโค้ดเวอร์ชันเก่าไว้เพื่อเวอร์ชันใหม่ที่กำลังพัฒนานั้นเกิดพังขึ้นมา จะได้กลับไปใช้ โค้ดเวอร์ชันเก่าได้นั่นเอง**

สาเหตุที่ต้องใช้ Version Control

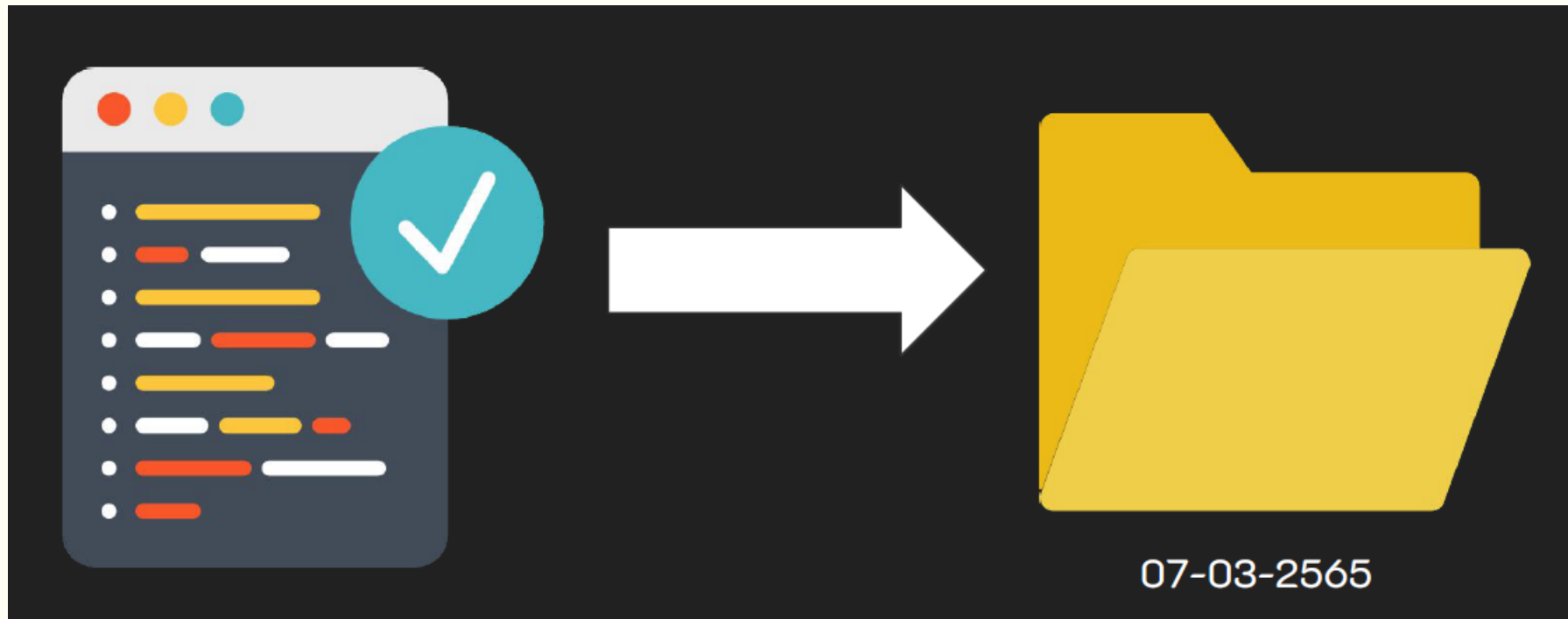


- project
- projectv2
- projectv3 fixbug
- projectv4 add feature
- projectv5 final
- projectv5 final fixbug

สาเหตุที่ต้องใช้ Version Control



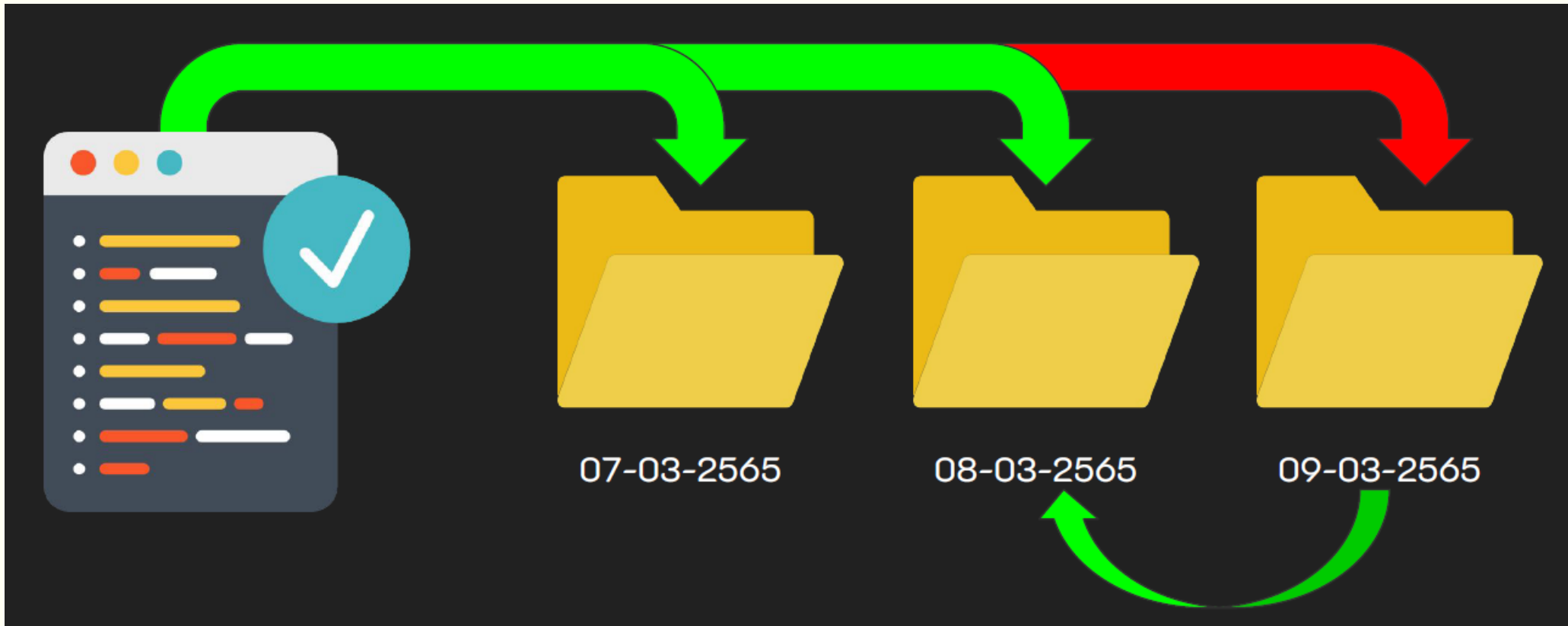
สาเหตุที่ต้องใช้ Version Control



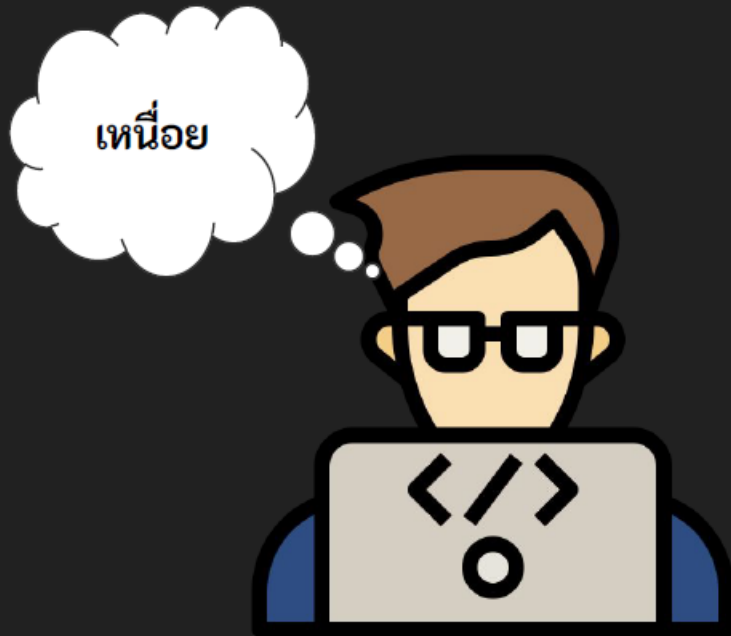
สาเหตุที่ต้องใช้ Version Control



สาเหตุที่ต้องใช้ Version Control

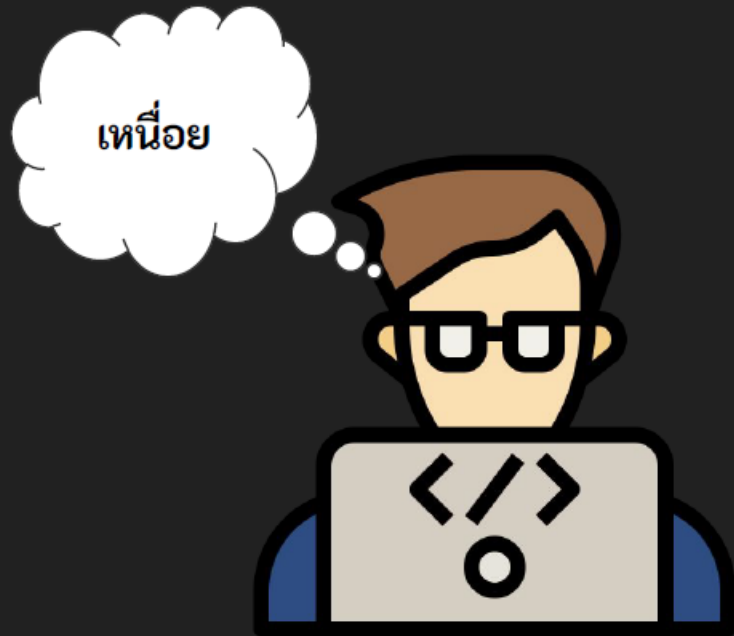


สาเหตุที่ต้องใช้ Version Control



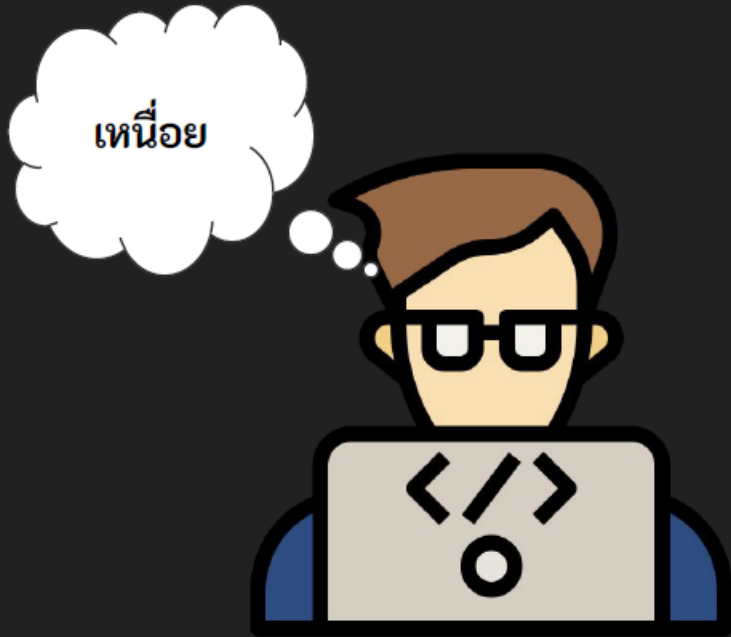
อยากกลับไปแก้งานเก่าที่ทำไว้ จะต้อง
ไปหาไฟล์ Backup แล้วจำได้ว่าไฟล์
Backup นั้น ถูกแก้ไขอะไรไปบ้าง
แก้ไขล่าสุดวันไหน แก้ไขเวลาใด ? / **ไม่รู้**

สาเหตุที่ต้องใช้ Version Control



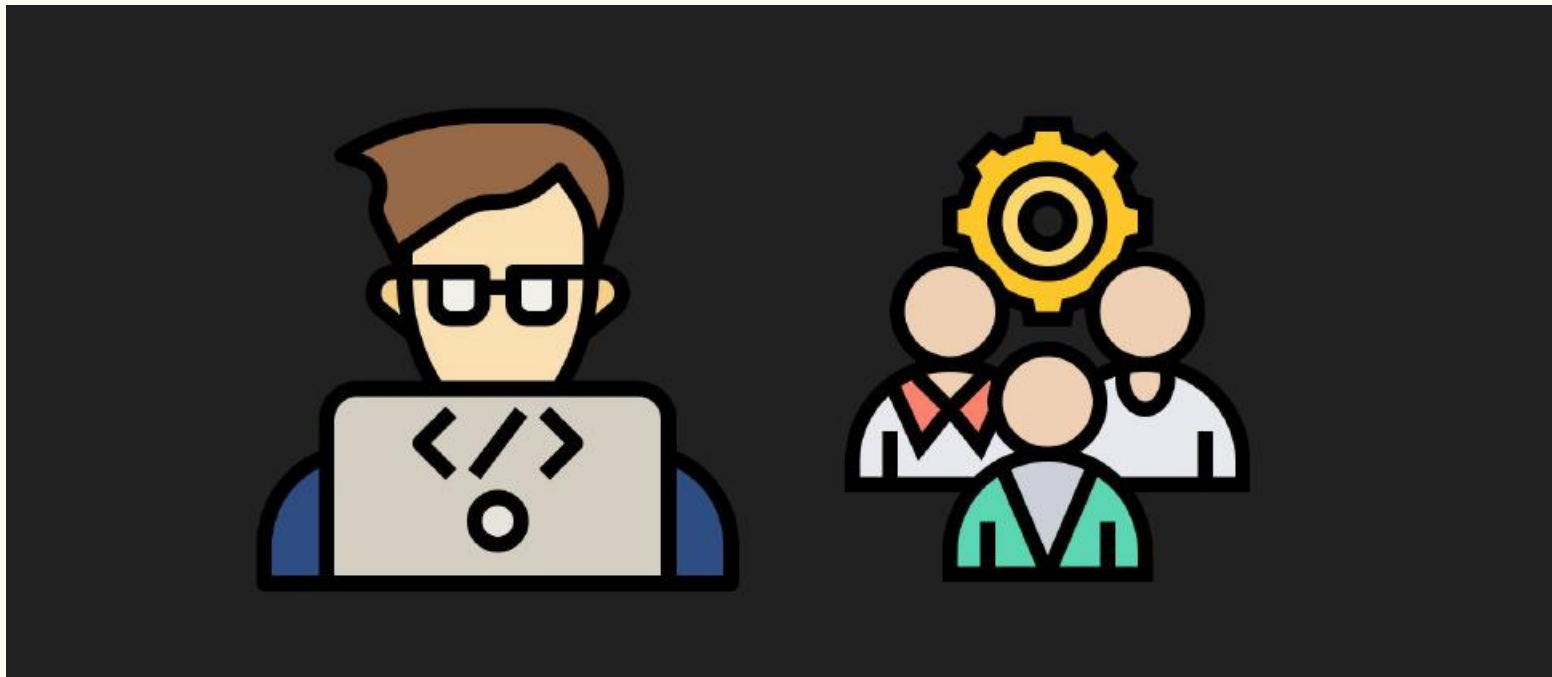
อยากทดลองเพิ่มฟีเจอร์ใหม่เข้าไป
ในโปรเจกต์เก่า จะทำอย่างไร
โดยที่ไม่ต้อง Copy โค้ดทั้งโปรเจ็ค
แล้วมาเปลี่ยนชื่อในภายหลัง / **ไม่รู้**

สาเหตุที่ต้องใช้ Version Control

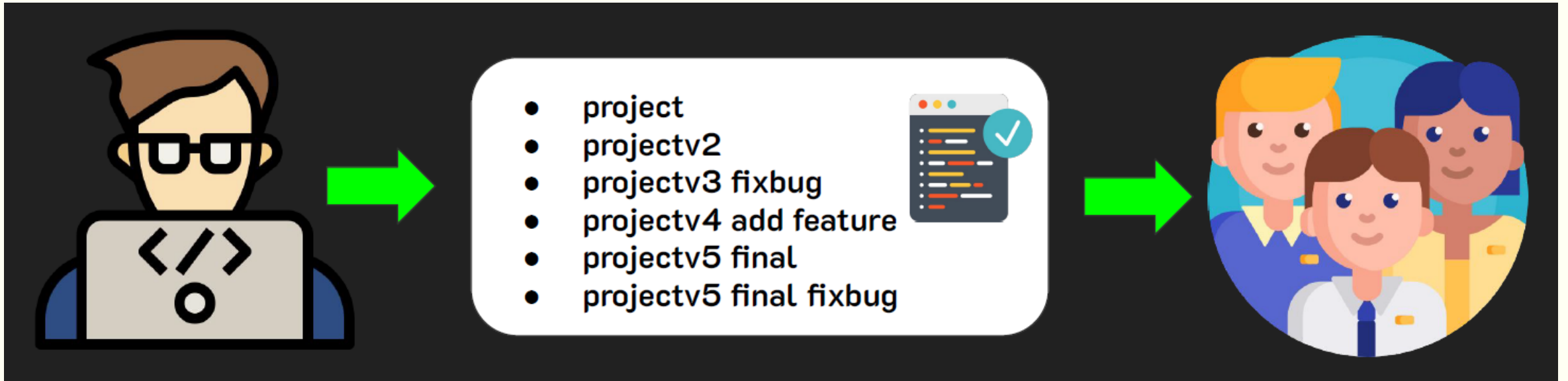


หลังเขียนโค้ดเสร็จต้องทำการ Backup
โค้ดเก่าเก็บไว้ทุกวัน ถ้าวันหนึ่งพื้นที่เก็บ
ข้อมูลเต็มจะทำอย่างไร ? / **ไม่รู้**

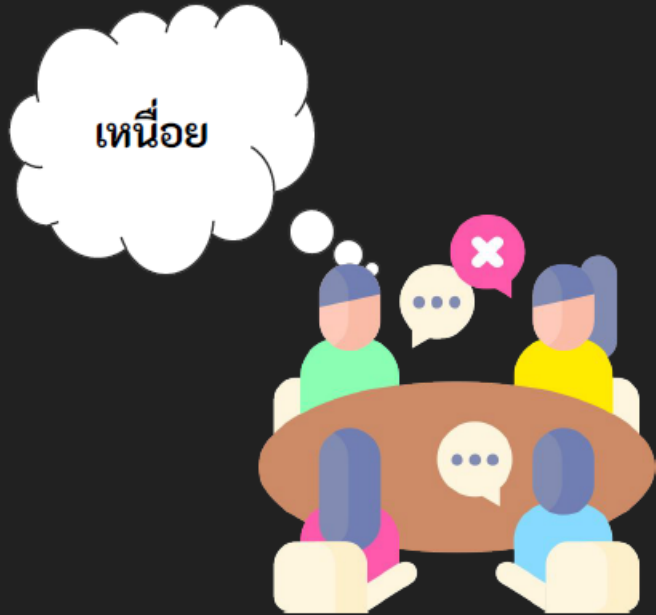
สาเหตุที่ต้องใช้ Version Control



สาเหตุที่ต้องใช้ Version Control

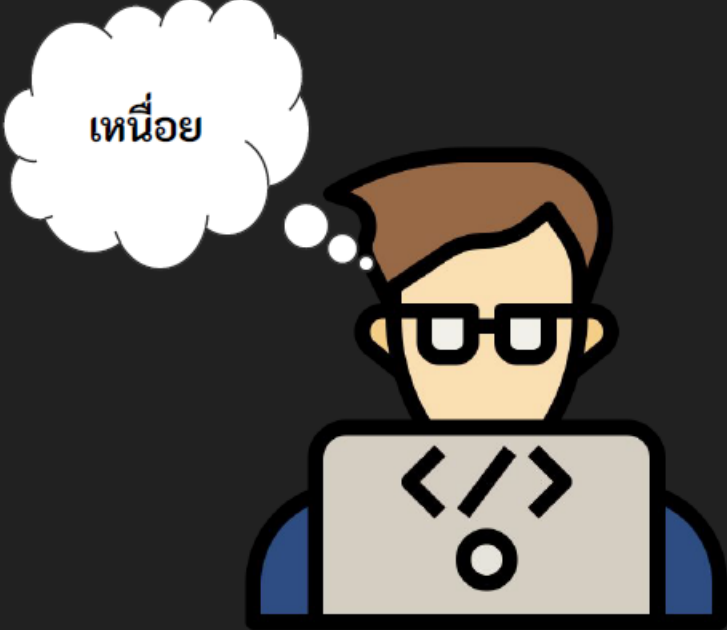


สาเหตุที่ต้องใช้ Version Control



- ต้องส่งงานที่ทำล่าสุดให้กันอย่างไร ?
- คนในทีมทำการปรับปรุงแก้ไขโค้ดอะไรไปบ้าง ?
- ถ้าแก้ไขโค้ดจุดเดียวกันจะเกิดปัญหาหรือไม่ ?
- อยากพัฒนาระบบที่มีอยู่ไม่ให้กระทบกับ Production จะทำอย่างไร ?

สาเหตุที่ต้องใช้ Version Control



เหนื่อย

- หาเครื่องมือมาช่วยในการเก็บการเปลี่ยนแปลงของไฟล์ได้ โดยการเก็บประวัติไฟล์ว่าถูกสร้าง/ลบ/แก้ไข โดยใคร เมื่อไหร่
- สามารถติดตามการเปลี่ยนแปลงของโค้ดในไฟล์ได้หรือย้อนเวลาโค้ดกลับไปก่อนตอนที่จะพังได้

สาเหตุที่ต้องใช้ Version Control



“ Version Control ”

วิวัฒนาการของ Version Control


- Copy File & Folder
- Patch
- Local Version Control System
- Centralized Version Control System (CVCS)
- Distributed Version Control System (DVCS)

วิวัฒนาการของ Version Control

- Copy File & Folder

คือ ลักษณะการบันทึกเอกสารและแยกออกเป็นหลายๆไฟล์ แล้วตั้งชื่อไฟล์พร้อมระบุเวอร์ชันตามลำดับ



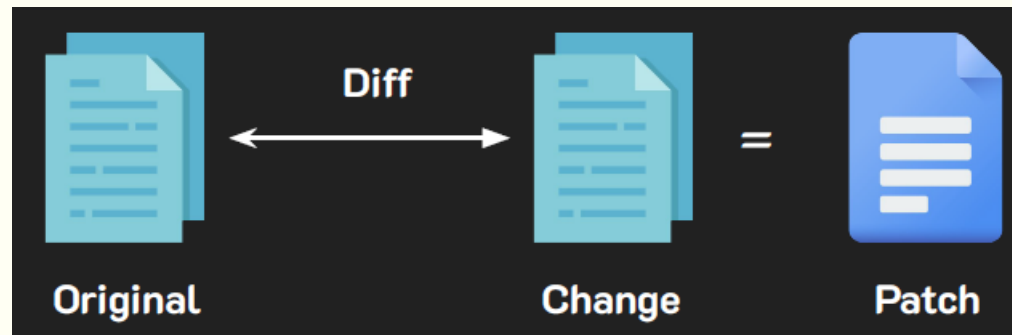
	project.py	07-03-2565
	projectv1.py	10-03-2565
	projectv1.2.py	15-03-2565
	projectfinal.py	30-03-2565

ข้อเสีย : เปลืองพื้นที่จัดเก็บข้อมูลตามจำนวนเวอร์ชันของเอกสาร เพราะต้องคัดลอกไฟล์ทั้งโฟลเดอร์เพื่อสร้างเอกสารเวอร์ชันใหม่ขึ้นมา

วิวัฒนาการของ Version Control

- Patch (แพตช์)

เพื่อเลี่ยงความเสี่ยงพื้นที่ในการจัดเก็บแบบวิธี Copy File & Folder เกิดจากการเปรียบเทียบไฟล์เก่ากับไฟล์ใหม่ที่มีการเปลี่ยนแปลง



ข้อเสีย : อาจมีความเสี่ยงต่อการทำ Patch บาง Patch หาย จนไม่สามารถประกอบร่าง Source Code กลับไปยังเวอร์ชันต่างๆได้

วิวัฒนาการของ Version Control

- Local Version Control System

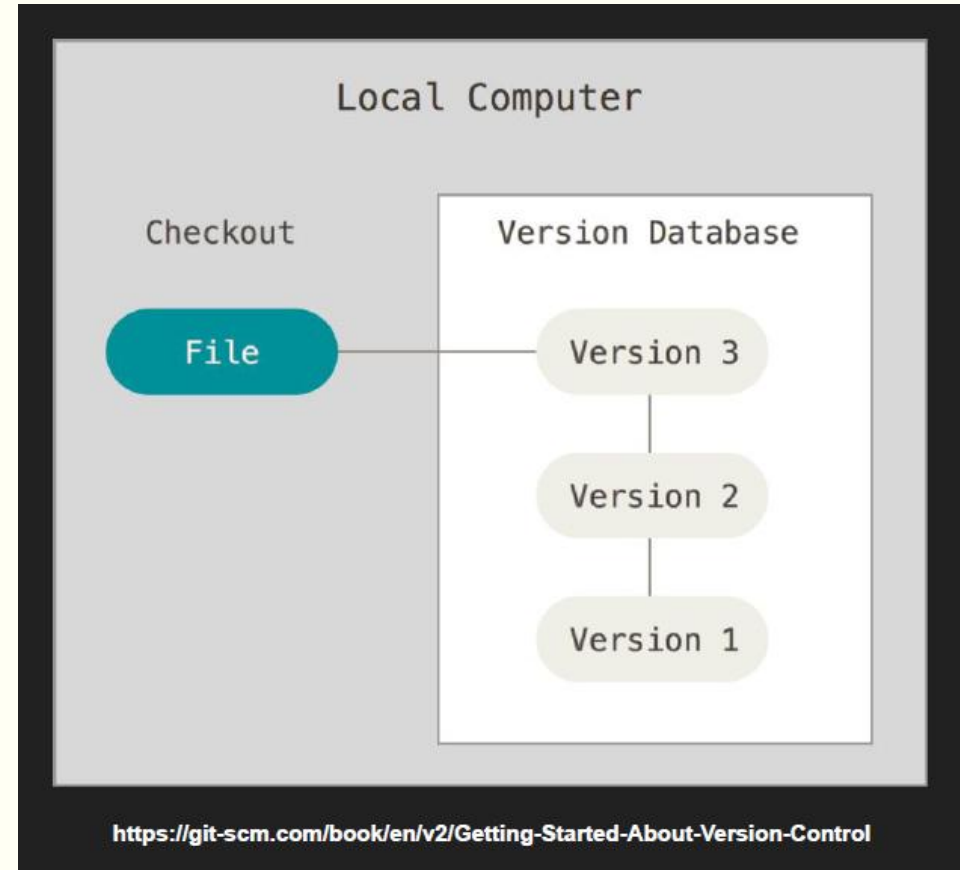
เพื่อแก้ปัญหา Patch หาย จนไม่สามารถประกอบร่าง Source Code กลับไปยังเวอร์ชันต่างๆได้ จึงได้มีการพัฒนา Version Control System (VCS) ที่มีฐานข้อมูลเฉพาะคอยจัดเก็บทุกการเปลี่ยนแปลงของ Source Code โดยจะเรียกการจัดเก็บเวอร์ชันของ Source Code ลงฐานข้อมูลเรียกว่า “Check-In” และเรียกคืน Source Code จากฐานข้อมูลเพื่อทำงานต่อว่า “Check-Out”



วิวัฒนาการของ Version Control

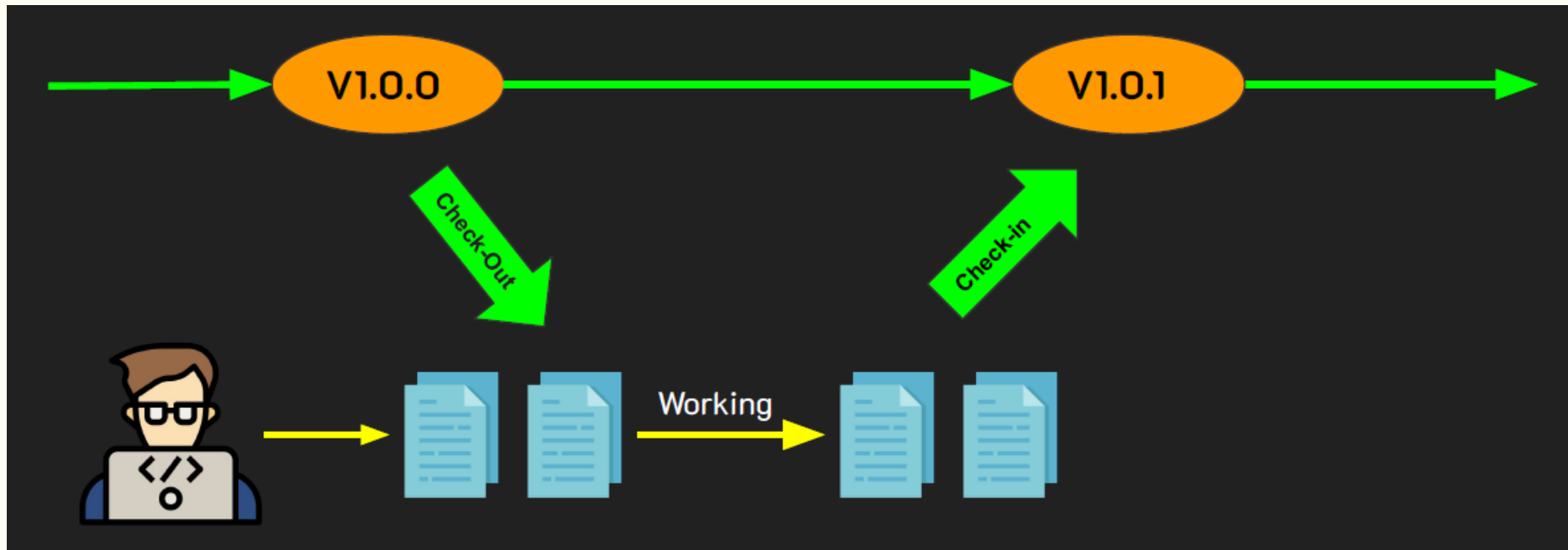
- Local Version Control System

มีความสามารถในการจัดเก็บเวอร์ชัน (Check-In) พร้อมทั้งข้อความช่วยจำ (Log Message) ลงในฐานข้อมูลและเรียกคืนเวอร์ชันจากฐานข้อมูล (Check-Out) กลับมายังพื้นที่ทำงาน เพื่อให้ นักพัฒนาซอฟต์แวร์แก้ไข Source Code ต่อไปได้



วิวัฒนาการของ Version Control

- Local Version Control System



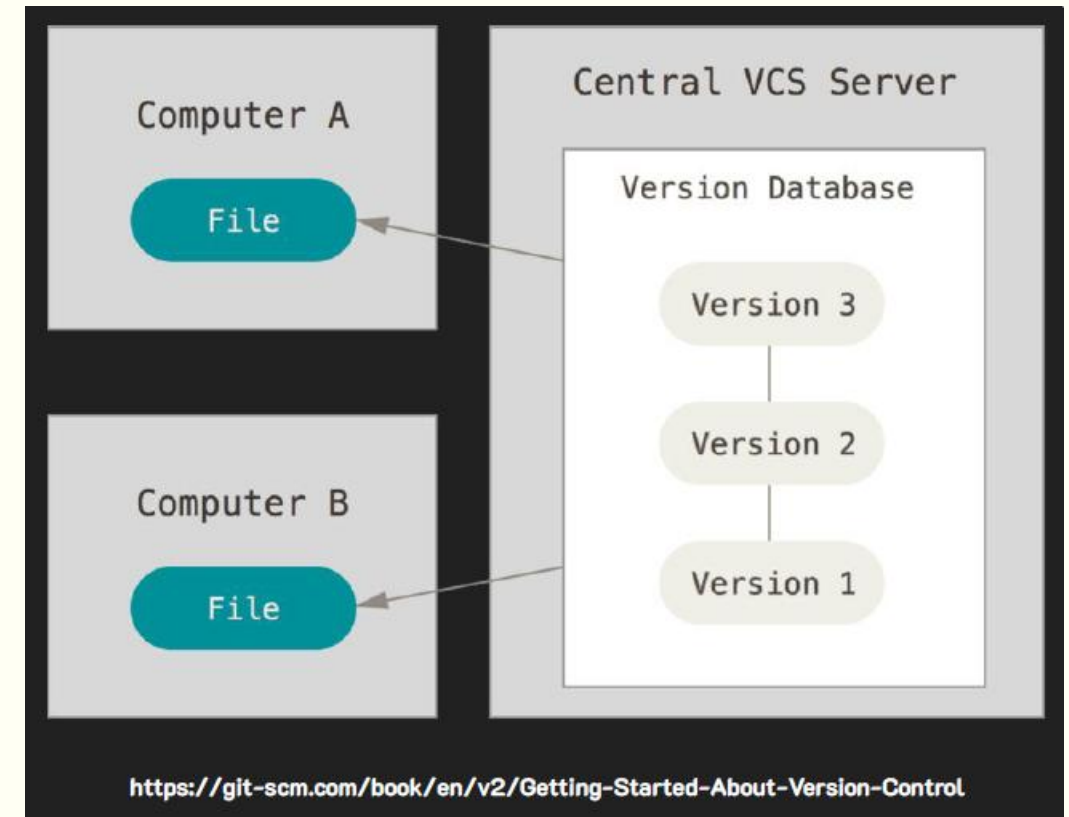
ข้อเสีย : การ Check-Out จากผู้ใช้หลายคน อาจจะทำให้เกิดปัญหาขึ้น

วิวัฒนาการของ Version Control

- Centralized Version Control System

เป็น Version Control System แบบรวมศูนย์ เพื่อแก้ปัญหakerณีที่มีผู้ใช้งานหลายคนโดยการเก็บข้อมูลไว้บน Server ซึ่งเมื่อผู้ใช้ Check-Out งานเวอร์ชันเดียวกันแล้วกลับมา Check-In ระบบจะพยายามรวบรวมเนื้อหาเข้าด้วยกัน (Merge)

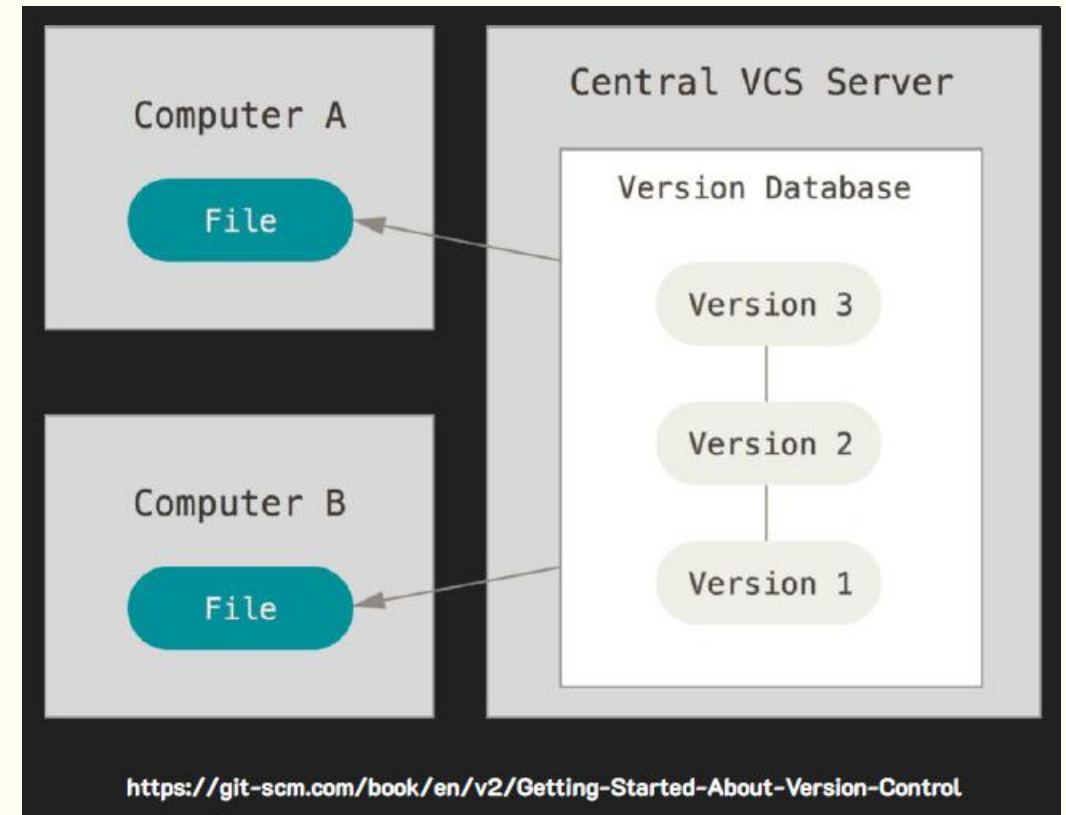
- ถ้ารูปแบบการรวบรวมเนื้อหามีความเรียบง่ายระบบจะรวมเนื้อหาให้อัตโนมัติ
- ถ้าการรวมมีความซับซ้อน ระบบจะแจ้งให้ผู้ใช้งานตัดสินใจแทน



วิวัฒนาการของ Version Control

ข้อเสีย :

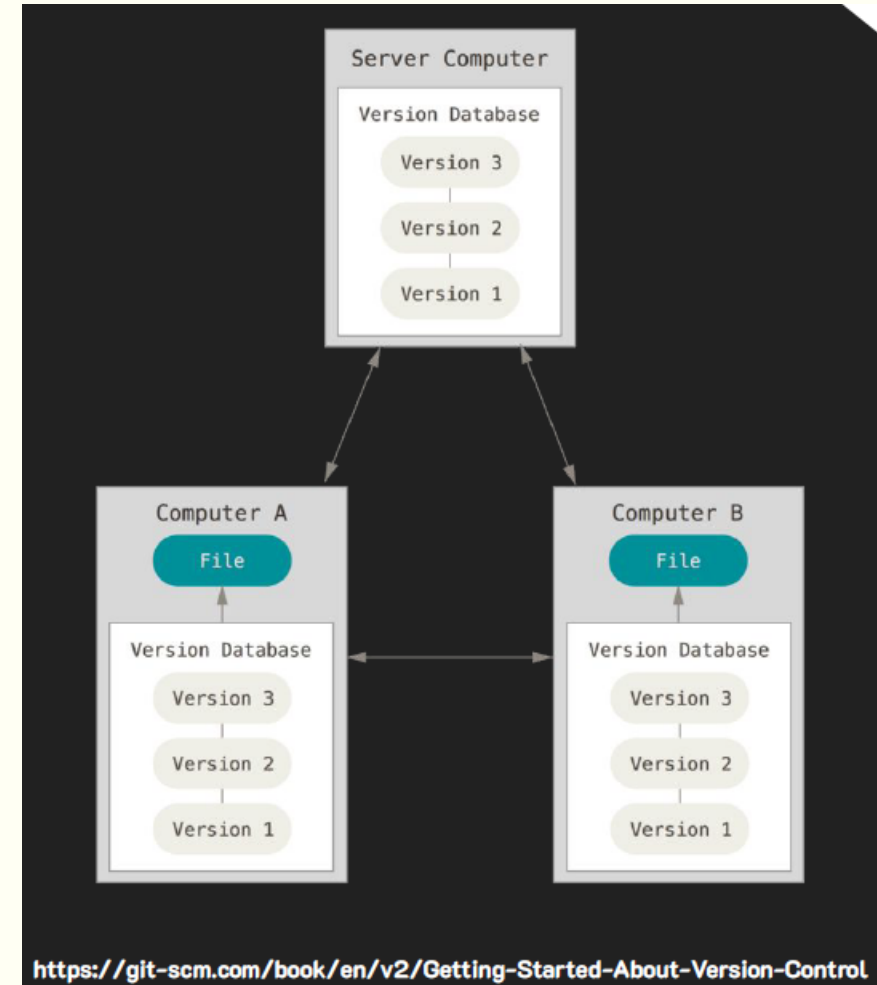
- การทำงานแบบรวมศูนย์ หาก Server ล่มชั่วคราว ผู้ใช้จะไม่สามารถ Check-In หรือ Check-Out ได้
- ถ้า Server ล่มถาวร เวอร์ชันของ Source Code ทั้งหมดก็จะได้รับผลกระทบไปด้วย
- ทำงานแบบ Online เท่านั้น



วิวัฒนาการของ Version Control

- Distributed Version Control System

จะแก้ปัญหการทำงานแบบรวมศูนย์โดยการโคลนฐานข้อมูลมาทั้งหมด (เรียก Database ที่เก็บเวอร์ชันของโค้ดว่า Repository) ซึ่งเมื่อ Clone มาแล้ว สามารถ Check-In และ Check-Out บน Local Host แบบ Offline ก่อนจะ Push ขึ้น Server ในภายหลัง



สรุป

จุดประสงค์ที่สำคัญของการใช้งาน Version Control System คือ เพื่อให้สามารถย้อนกลับไปยังเวอร์ชันก่อนหน้าได้ เมื่อพบปัญหาระหว่างพัฒนาโปรแกรม การ Check-In เพื่อเปลี่ยนแปลง Source Code ไปยังระบบจัดเก็บเวอร์ชัน (Version Control System) จึงเป็นเรื่องที่ต้องทำเป็นประจำ

เพราะถ้าคนในทีมพัฒนาโปรแกรมไม่ทำการ Check-In เป็นประจำ แล้วพบปัญหาขึ้นกับซอร์ฟแวร์ที่อยู่ระหว่างพัฒนา ก็อาจก่อให้เกิดปัญหาใหญ่ตามมาคือ ไม่สามารถย้อนกลับไปยังเวอร์ชันต่างๆก่อนหน้าได้ หรืออาจจะต้องยุ่งยากหรือใช้เวลานานกว่าจะย้อนกลับไปได้