Claude Code for Everyone

ระยะเวลา 1 วัน

หลักการและเหตุผล

เวิร์กช็อป 1 วันสำหรับคนทำงานทุกสายงานที่อยากใช้ Claude Code เป็นเครื่องมือทำงานจริง โดยไม่ต้องเขียนโค้ดเป็น ตลอดทั้งวันไม่มีการเขียน โปรแกรมแม้แต่บรรทัดเดียว สิ่งที่ผู้เรียนได้คือความเข้าใจว่า agent ประกอบด้วยอะไรบ้าง ควบคุมมันอย่างไร และจะวางมันลงในงานประจำของ ตัวเองตรงไหน

ช่วงเช้าออกแบบเป็นทัวร์ส่วนประกอบของ Claude Code โดยสลับระหว่างการบรรยายกับการลงมือทุก 6 ถึง 10 นาที มี micro-lab สั้น 10 ครั้ง ก่อนพักเที่ยง ผู้เรียนจึงได้แตะคีย์บอร์ดตลอดเวลา แทนการนั่งฟังยาว จากนั้นช่วงบ่ายเปลี่ยนเป็น lab ยาวกับงานจริงของตัวเอง จบวันด้วย workflow ส่วนตัวหนึ่งหน้าที่กลับไปใช้ได้ทันที

วัตถุประสงค์

- อธิบายส่วนประกอบของ Claude Code ได้ครบ: session, context window, tools, permissions, checkpoints, CLAUDE.md, skills, plugins, MCP, subagents และ hooks
- ควบคุมความปลอดภัยด้วย permission mode และย้อนงานที่พังด้วย checkpoint ได้
- ใช้ลูป ซัก สั่งทำ ล้าง ร่วมกับ TSOS framework กับงานจริงของตัวเองได้ เขียน CLAUDE.md และสร้าง skill ของตัวเองสำหรับงานที่ทำซ้ำทุกสัปดาห์
- ออกแบบ feedback loop และจุดที่คนต้องตัดสินใจ สำหรับงานที่ไม่ใช่โค้ด

รายละเอียดหลักสูตร

เวลา	หัวข้อ	เนื้อหา
9:00 - 09:30 น.	**Agent ต่างจาก Chat อย่างไร**	▪ Agentic loop: agent อ่านไฟล์ วางแผน ลงมือ และตรวจงานตัวเองได้ ข้อจำกัดจริงของ LLM ที่ต้องรู้ก่อนสั่งงาน ▪ สั่งเหมือนเดิมแต่ได้ผลไม่เหมือนเดิม แปลว่าต้องออกแบบวิธีตรวจ ไม่ใช่หวังดวง **Micro LAB A:** ▪ ติดตั้งและเปิด session แรก ถามคำเดียวว่าโฟลเดอร์นี้มีอะไรบ้าง ทุกคนต้อง ผ่านจุดนี้ก่อนไปต่อ
09.30 – 10.30 น.	**ส่วนประกอบที่ 1: ตัวเซสชัน**	▪ Session: เปิด ปิด ตั้งชื่อ และ resume กลับมาทำต่อ **Micro LAB B:** ▪ ปิด session แล้ว resume กลับมา ดูว่าอะไรกลับมาบ้าง ▪ Context window: งบประมาณของ agent ไม่ใช่ความจำถาวร **Micro LAB C:** ▪ พิมพ์ /context ดูตัวเลข สั่งให้อ่านไฟล์ใหญ่ 1 ไฟล์ แล้วกด /context ซ้ำเพื่อเทียบ ▪ Tools ที่ agent มีในมือ: อ่านไฟล์ แก้ไฟล์ รันคำสั่ง ค้นเว็บ **Micro LAB D:** ▪ สั่งงานที่ต้องใช้หลาย tool แล้วสังเกตว่ามันหยิบอะไรมาใช้บ้าง ▪ เว็บ Permissions: plan, acceptEdits, auto และเส้นแบ่งว่าอะไรควรอนุมัติอัตโนมัติ **Micro LAB E:** ▪ สลับเข้า plan mode สั่งงานเดิม เทียบว่ามันขออนุญาตต่างกันอย่างไร ▪ Checkpoints: ย้อนเวลาเมื่อ agent ทำงานพัง **Micro LAB F:** ▪ ตั้งใจสั่งงานให้พัง แล้วใช้ /rewind ย้อนกลับ ▪ ช่องทางใช้งานทั้งหมด: Desktop, CLI, VS Code, Web, Slack และมือถือ
10:30 - 10:45 น.	**BREAK**	

10:45 - 12:00 น.	**ส่วนประกอบที่ 2: ของที่ต่อเติมได้**	▪ CLAUDE.md: ความรู้ประจำโฟลเดอร์ที่ agent อ่านทุกครั้งที่เปิดงาน **Micro LAB G:** ▪ ใช้ /init ให้ Claude เขียน CLAUDE.md ให้ตัวเอง แล้วเปิดอ่านว่าเขียนอะไรไว้ ▪ Skills: ขั้นตอนงานที่แพ็กไว้เรียกใช้ซ้ำได้ ต่างจากการพิมพ์ prompt ซ้ำอย่างไร **Micro LAB H:** ▪ เรียก bundled skill กับไฟล์จริง แปลงข้อมูลเป็น xlsx หรือสร้างสไลด์ pptx ▪ Plugins: ชุดรวม skill และเครื่องมือ ติดตั้งจาก marketplace **Micro LAB I:** ▪ เปิด /plugin เดินดู marketplace แล้วติดตั้ง 1 ตัว ▪ MCP: ท่อต่อข้อมูลนอกเครื่อง เช่น Google Drive, Notion, SharePoint **Micro LAB J:** ▪ ต่อ MCP 1 ตัว แล้วถามข้อมูลจากที่นั่น ▪ Subagents และ hooks: รู้จักไว้ก่อน ยังไม่ลงมือในคอร์สนี้ ▪ ตารางเทียบ: อันไหนใช้ตอนไหน และอันไหนกิน context เท่าไหร่
12:00 - 13:00 น.	**LUNCH BREAK**	
13:00 – 14:00 น.	**ลูปหลัก: ซัก สั่งทำ ล้าง**	▪ ก่อนสั่งงานต้องซักให้ agent เข้าใจบริบทก่อน แล้วค่อยสั่งทำ จบแล้วล้าง context เริ่มรอบใหม่ ▪ TSOS framework (Task, Source, Output, Safety) วางในขั้นสั่งทำ ▪ Compaction และการส่งงานข้าม session โดยไม่เสียบริบท **LAB 3:** ▪ ทำงานจริงของตัวเองครบหนึ่งลูป
14:00 - 15.00 น.	**สอน agent ให้รู้จักงานคุณ**	▪ เขียน CLAUDE.md อย่างไรให้ agent ทำตามจริง ▪ Progressive disclosure: ไม่ยัดทุกอย่างลงไฟล์เดียว ให้ agent ค่อยเปิดอ่านตามต้องการ ▪ Automatic memory: ตรวจและแก้สิ่งที่ agent จำไว้เอง **LAB 4:** ▪ เขียน CLAUDE.md ของตัวเอง แล้วสร้าง skill งานประจำ 1 ตัว
15.00 – 15.15 น.	**BREAK**	
15.15 – 16.00 น.	**ตรวจงาน AI และ workflow ปิดท้าย**	▪ งานร่างมันถูก: ทิ้งแล้วสั่งใหม่มักเร็วกว่านั่งแก้ทีละบรรทัด ▪ Feedback loop สำหรับงานที่ไม่ใช่โค้ด: เช็คลิสต์ ไฟล์ตัวอย่าง และการเปิดผลลัพธ์ดูจริง ▪ จุดที่คนต้องกดอนุมัติเสมอ และงานที่ไม่ควรปล่อยให้ agent ทำเอง **LAB 5:** ▪ เขียน workflow ส่วนตัวหนึ่งหน้า ปิดท้ายด้วยหลัก AI drafts, Human decides