

(ฉบับลงมือทำ)

Pydantic

การตรวจสอบและจัดการข้อมูล
ใน Python อย่างมืออาชีพ

อนุชิต ฐไลรร

สำนักพิมพ์ท้อปวาง

คู่มือ Pydantic ฉบับลงมือทำ

การตรวจสอบและจัดการข้อมูลใน
Python อย่างมืออาชีพ

อนุชิต ชโลธร

คำนำสำนักพิมพ์

ในโลกการพัฒนาซอฟต์แวร์ยุคใหม่ ข้อมูลเปรียบเสมือนเส้นเลือดใหญ่ของทุกแอปพลิเคชัน ความถูกต้อง ความน่าเชื่อถือ และการคาดการณ์ได้ของข้อมูลคือหัวใจสำคัญ แต่การทำให้มั่นใจว่าข้อมูลเหล่านั้นถูกต้องกลับไม่ใช่เรื่องง่าย มักเต็มไปด้วยความซับซ้อนและโอกาสเกิดข้อผิดพลาด โดยเฉพาะสำหรับผู้เพิ่งเริ่มต้นในสายงานนี้ และนี่เองคือจุดที่ Pydantic โดดเด่นขึ้นมา

เรามีความยินดีอย่างยิ่งที่ได้เสนอ *คู่มือ Pydantic ฉบับลงมือทำ : ตรวจสอบและจัดการข้อมูลใน Python อย่างมืออาชีพ* ซึ่งถูกสร้างขึ้นเพื่อเติมเต็มช่องว่างสำคัญของนักพัฒนา นั่นคือการเข้าถึงเครื่องมือที่จำเป็นที่สุดอย่างหนึ่งของ Python ในรูปแบบที่ตรงไปตรงมาและใช้งานได้ง่าย

หนังสือเล่มนี้ออกแบบมาสำหรับผู้ทุกระดับถึงความสำคัญของการตรวจสอบข้อมูล แต่ยังมองหาหนทางที่ชัดเจนในการเรียนรู้และใช้งานโดยไม่ต้องหลงไปกับเอกสารที่ซับซ้อนหรือภาษาที่เป็นวิชาการเกินไป ผู้เขียนได้ถ่ายทอดแนวคิดที่ซับซ้อนให้ง่ายต่อการเข้าใจ ตั้งแต่โมเดลแรกของคุณ ไปจนถึงการทำงานร่วมกับเฟรมเวิร์กยอดนิยมอย่าง FastAPI

แต่ละบทเชื่อมโยงต่อกันอย่างมีเหตุผล การเรียนรู้จึงเป็นไปอย่างราบ
รื่นและคุ้มค่า ที่สำคัญ ทุกบทมุ่งเน้นการนำไปใช้จริง เพื่อให้คุณ
สามารถประยุกต์สิ่งที่ได้เรียนรู้กับโปรเจกต์ของคุณทันที

เรามั่นใจว่าหนังสือเล่มนี้จะเป็นเสมือนเพื่อนคู่คิดในเส้นทางการเป็น
นักพัฒนา ไม่เพียงแต่สอนการใช้ Pydantic เท่านั้น แต่ยังปลูกฝัง
ความเข้าใจที่ลึกซึ้งยิ่งขึ้น ว่าทำไมการตรวจสอบข้อมูลจึงเป็นรากฐาน
สำคัญของซอฟต์แวร์ที่แข็งแกร่งและมีคุณภาพสูง

ขอให้คุณสนุกกับการเรียนรู้

คำนำผู้เขียน

ครั้งแรกที่ผมได้ลองเขียนโปรแกรมด้วยภาษา Python ผมรู้สึกทั้งในความเรียบง่ายและทรงพลังของมัน แต่เมื่อโปรเจกต์เริ่มใหญ่ขึ้น ความซับซ้อนก็มากขึ้นตามไปด้วย ผมใช้เวลาจำนวนมากไปกับการเขียนโค้ดซ้ำๆ เพื่อตรวจสอบข้อมูลที่เข้ามา ซึ่งเป็นงานที่น่าเบื่อ ซ้ำซาก และมักก่อให้เกิดบักอยู่เสมอ จนกระทั่งผมได้ค้นพบ Pydantic และทุกอย่างก็เปลี่ยนไป

ทันใดนั้น งานตรวจสอบข้อมูลที่เคยยุ่งเหยิงกลับกลายเป็นเพียงการประกาศที่ชัดเจน สะอาด และแทบไม่ต้องใช้ความพยายาม ทำให้ผมสามารถทุ่มเทเวลาให้กับการพัฒนาตรรกะหลักของแอปพลิเคชัน โดยมั่นใจได้ว่าข้อมูลที่ไหลผ่านนั้นถูกต้องตามที่คาดหวัง มันให้ความรู้สึกเหมือนกับได้พลังพิเศษ และเป็นพลังที่ผมอยากถ่ายทอดให้นักพัฒนาทุกคน โดยเฉพาะผู้ที่เพิ่งเริ่มต้น

หนังสือเล่มนี้คือคู่มือที่ผมหวังว่าจะได้อ่านในวันที่เริ่มต้น ผมออกแบบเนื้อหาให้ตรงไปตรงมา ลงมือปฏิบัติได้จริง โดยไม่ติดอยู่กับทฤษฎีที่ยืดยาว แต่จะพาคุณกระโดดเข้าสู่การเขียนโค้ดทันที แต่ละบทถูกวางโครงสร้างให้เป็นสูตรสำเร็จที่ช่วยแก้ปัญหาที่พบบ่อย เป้าหมายของผมคือให้คุณอ่านหนังสือเล่มนี้จบพร้อมทั้งความรู้และความมั่นใจที่จะใช้ Pydantic ได้อย่างมีประสิทธิภาพ

ขอบคุณที่เลือกหยิบหนังสือเล่มนี้ขึ้นมา ผมรู้สึกตื่นเต้นที่ได้ร่วมเป็นส่วนหนึ่งในเส้นทางการเรียนรู้ Python ของคุณ และหวังว่า Pydantic จะมอบทั้งความชัดเจนและความสุขให้กับการเขียนโค้ดของคุณ เหมือนที่มันได้มอบให้ผม

ขอให้สนุกกับการเขียนโค้ดครับ

สารบัญ

แนะนำ Pydantic	1
Pydantic แก้ปัญหาอะไร	1
ปัญหา: ความโกลาหลของข้อมูลและความไม่แน่นอน	2
ทางออก: การตรวจสอบเชิงประกาศด้วย Type Hint	3
การติดตั้งและการตั้งค่าโปรเจกต์	5
การติดตั้งมาตรฐาน	5
การติดตั้งส่วนเสริม	6
การตั้งค่าโปรเจกต์	6
โมเดลข้อมูล	8
การกำหนดโมเดล	8
Schema คืออะไร	10
การอ่านข้อมูลเข้าสู่โมเดลของคุณ	11
การใช้ <code>model_validate()</code>	11
การจัดการข้อมูลที่ไม่ถูกต้อง	12
การตรวจสอบข้อมูล JSON	14
การแปลงข้อมูลจากโมเดลของคุณ	15
การแปลงเป็น dict ด้วย <code>model_dump()</code>	16
การส่งออกเป็น JSON ด้วย <code>model_dump_json()</code>	18

การตรวจสอบข้อมูลพื้นฐาน	20
การทำงานกับชนิดข้อมูลมาตรฐาน	20
ชนิดข้อมูลพื้นฐาน: int, float, str, bool	20
ตัวอย่างโมเดลและการแปลงประเภท	21
ชนิดข้อมูลมาตรฐานอื่น ๆ ที่มีประโยชน์	23
ตัวอย่างโมเดลคอลเลกชัน	24
การจัดการกับฟิลด์ที่เป็นทางเลือก	25
ประเภท Optional	25
พฤติกรรมของฟิลด์ที่เป็นทางเลือก	26
Optional เทียบกับ ค่าเริ่มต้น	29
การกำหนดค่าเริ่มต้น	29
ทำไมต้องใช้ค่าเริ่มต้น	29
วิธีการกำหนดค่าเริ่มต้น	30
ค่าเริ่มต้น (Default Values) กับฟิลด์ที่อาจไม่มีค่า (Optional Fields)	32
การตรวจสอบข้อมูลวัน-เวลา	34
การทำงานกับ datetime	35
การทำงานกับ date และ time	37
การทำงานกับ timedelta	38
การใช้ Enums เพื่อจำกัดค่าที่เป็นไปได้	40
ทำไมต้องใช้ Enums	40

การสร้างโมเดลด้วย Enum	41
ประโยชน์ในการใช้งาน	44
การสร้างโครงสร้างข้อมูลที่ซับซ้อน	45
การซ้อนโมเดลเข้าไปในโมเดลอื่น (Nesting Models)	45
ทำไมต้องโมเดลซ้อนโมเดล	45
วิธีการซ้อนโมเดล	46
การสร้างลิสต์ของโมเดล	48
วิธีการสร้างลิสต์ของโมเดล	49
การใช้ Dictionaries กับประเภทข้อมูลพื้นฐาน	53
วิธีการใช้ Dictionaries	53
การตรวจสอบความถูกต้องแบบกำหนดเองอย่างง่าย	58
บทนำสู่ตัวตรวจสอบความถูกต้องแบบกำหนดเอง (Validators)	58
บทบาทของ Custom Validators	58
Validators ทำงานอย่างไร	59
การตรวจสอบฟิลด์เดียวด้วย @field_validator	60
วิธีใช้ @field_validator	60
การใช้ Validator ซ้ำสำหรับหลายฟิลด์	64
การตรวจสอบแบบหลายฟิลด์ด้วย @model_validator	65
วิธีใช้ @model_validator	66
การควบคุม Input และ Output	70

การเปลี่ยนชื่อฟิลด์ด้วย Aliases	70
ประโยชน์การใช้ Aliases	70
วิธีการใช้ Alias	71
การยกเว้นฟิลด์ออกจากผลลัพธ์ตอนส่งออก	74
ทำไมต้องยกเว้นฟิลด์?	74
วิธีการยกเว้นฟิลด์	75
การเพิ่มค่าที่คำนวณได้ด้วย @computed_field	78
ประโยชน์ของ Computed Fields	78
วิธีการใช้ @computed_field	79
การตั้งค่าแอปพลิเคชัน	83
การสร้างโมเดล BaseSettings แรกของคุณ	83
ทำไมต้องใช้ BaseSettings	83
จาก BaseModel สู่ BaseSettings	84
การใช้งาน AppSettings	85
การโหลดการตั้งค่าจากตัวแปรสภาพแวดล้อม	87
การจับคู่ฟิลด์กับ Environment Variables	88
การโหลดการตั้งค่าจากไฟล์ .env	91
วิธีทำงานของ Dotenv ใน Pydantic	92
ตัวอย่าง การใช้ไฟล์ .env	92
การนำทุกอย่างมารวมกันด้วย FastAPI	96
แนะนำ FastAPI	96

ทำไมถึงเลือกใช้ FastAPI ร่วมกับ Pydantic	96
“Hello World” ง่ายๆ ใน FastAPI	97
การใช้โมเดล Pydantic สำหรับ Input ของ API	100
วิธีการนิยาม Request Bodies	100
การโต้ตอบกับ API	102
FastAPI ทำอะไรไปบ้าง?	105
การใช้โมเดล Pydantic สำหรับ Output ของ API	106
วิธีการนิยาม Response Models	106
ตัวอย่างการแยกโมเดล Input และ Output	107
การโต้ตอบและตรวจสอบ	109
บทส่งท้าย	112
ภาพรวมของคุณสมบัติขั้นสูง	112
การตรวจสอบความถูกต้องขั้นสูง (Advanced Validation)	112
ประเภทฟิลด์ขั้นสูง (Advanced Field Types)	113
การปรับแต่ง Serialization	113
การกำหนดค่าโมเดล (Model Configuration)	114
แหล่งข้อมูลสำหรับเรียนรู้เพิ่มเติม	114
เอกสาร Pydantic อย่างเป็นทางการ	114
เอกสาร FastAPI	115
ชุมชนและการขอความช่วยเหลือ	115
ข้อคิดสุดท้าย	115

แนะนำ Pydantic

ยินดีต้อนรับสู่โลกของ Pydantic ก่อนที่เราจะเริ่มเขียนโค้ด สิ่งสำคัญคือต้องเข้าใจ “เหตุผล” ที่อยู่เบื้องหลังไลบรารีนี้ก่อน Pydantic แก้ปัญหาอะไร และทำไมมันถึงกลายเป็นเครื่องมือที่ขาดไม่ได้สำหรับนักพัฒนา Python จำนวนมาก หัวใจหลักของ *Pydantic* คือการจัดระเบียบความโกลาหลของข้อมูล

Pydantic แก้ปัญหาอะไร

ลองจินตนาการว่าคุณกำลังสร้างแอปพลิเคชัน อาจจะเป็นเว็บ API เครื่องมือ Command-line หรือสคริปต์ประมวลผลข้อมูล แอปพลิเคชันของคุณจะต้องโต้ตอบกับข้อมูลจากแหล่งภายนอก ข้อมูลนี้อาจมาจาก

- JSON request body ที่ส่งมายัง API ของคุณ
- ข้อมูลจากฐานข้อมูล
- ไฟล์การตั้งค่า เช่น YAML หรือ TOML
- ไฟล์ CSV ที่คุณต้องประมวลผล

ข้อมูลภายนอกเหล่านี้มักจะยุ่งเหยิงและคาดเดาไม่ได้ อาจมีฟิลด์ที่ขาดหายไป, ประเภทข้อมูลไม่ถูกต้อง เช่น ตัวเลขที่ส่งมาเป็นสตริงอย่าง “42” หรือค่าที่ไม่สมเหตุสมผลสำหรับแอปพลิเคชันของคุณ