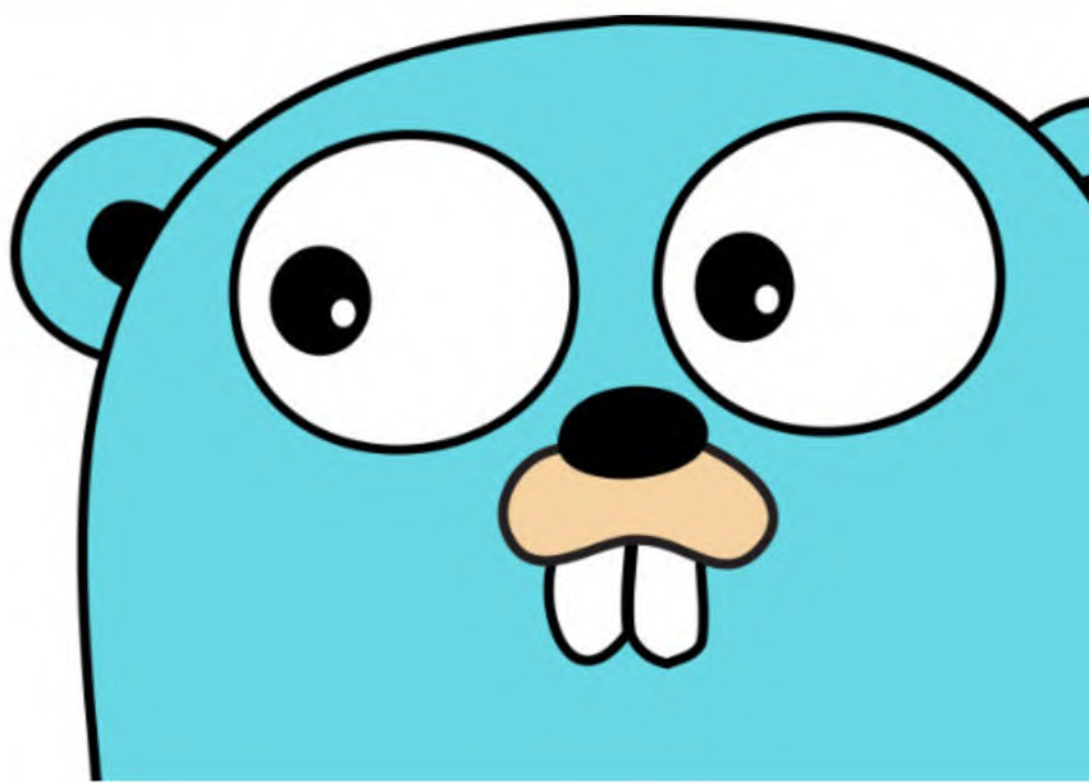


เขียนภาษา Go ถ่ายนิคเดียว

คู่มือสำหรับผู้เริ่มต้นเขียนโปรแกรมภาษา Go

อนุชิต ชโลธ

สำนักพิมพ์ก๊อปวาง



เขียนภาษา Go ง่ายนิดเดียว

คู่มือสำหรับผู้เริ่มต้นเขียนโปรแกรมภาษา Go

อนุชิต ชโลธร

บทนำ

มาเรียนรู้ภาษา Go กันเถอะ! คู่มือนี้เหมาะสำหรับคนที่เพิ่งเริ่มเขียนโปรแกรมหรืออยากลองใช้ภาษาใหม่ ถ้าคุณไม่เคยเขียนโปรแกรมมาก่อนหรือเคยเขียนมาบ้างแล้ว ภาษา Go เป็นตัวเลือกที่ดีมาก เราจะช่วยให้คุณเริ่มต้นได้ง่ายๆ ด้วยคำอธิบายที่เข้าใจง่ายและตัวอย่างสนุกๆ ที่คุณสามารถลองทำตามได้

ในคู่มือนี้ เราจะเริ่มจากพื้นฐาน เช่น การเขียนโค้ดแบบง่ายๆ วิธีสร้างตัวแปร เขียนฟังก์ชัน และใช้คำสั่งพื้นฐาน จากนั้นค่อยๆ เรียนเรื่องที่ซับซ้อนขึ้น เช่น การจัดการข้อมูลในคอมพิวเตอร์ การทำให้โปรแกรมทำงานหลายๆ อย่างพร้อมๆ กัน (Concurrency) และการใช้ฟังก์ชันสำเร็จรูปในภาษา Go

ถ้าคุณเคยเขียนโปรแกรมมาก่อน คู่มือนี้จะช่วยให้คุณเข้าใจว่าภาษา Go มีจุดเด่นอะไร และช่วยให้คุณเขียนโปรแกรมที่เร็วและง่ายต่อการดูแล นอกจากนี้ เรายังมีตัวอย่างโปรเจกต์ให้คุณลองทำ เพื่อฝึกฝนทักษะและได้เห็นว่าคุณสามารถใช้ Go ทำอะไรได้บ้าง

สารบัญ

บทที่ 1 แนะนำภาษา Go	1
ทำไมต้องภาษา Go?	1
ความเรียบง่ายและประสิทธิภาพในการพัฒนาโปรแกรม	1
การยอมรับและการใช้งาน	2
คุณสมบัติเด่น	2
การตั้งค่าสภาพแวดล้อมการพัฒนา	3
บทที่ 2 พื้นฐานภาษา Go	6
Hello, World!	6
ตัวแปรและชนิดข้อมูล	7
ค่าคงที่และตัวดำเนินการ	7
บทที่ 3 โครงสร้างการควบคุม (Control Structures)	9
คำสั่งเงื่อนไข (Conditional Statements)	9
ลูป for, break และ continue	11
ลูป for พื้นฐาน	11
ลูป for ที่ใช้เงื่อนไข	12
ลูปไม่มีที่สิ้นสุด	12
คำสั่ง break และ continue	12
คำสั่ง Switch	13
Switch รูปแบบพื้นฐาน	13
Switch โดยไม่มีการใช้เงื่อนไข	14

พฤติกรรมของ Fallthrough	14
ตรวจสอบหลายค่าในกรณีเดียว	15
บทที่ 4 ฟังก์ชันในภาษา Go	16
การประกาศและการเรียกใช้ฟังก์ชัน	16
พารามิเตอร์และค่าที่ส่งกลับ	17
พารามิเตอร์หลายตัว	17
การใช้รูปแบบที่สั้นลงสำหรับพารามิเตอร์ที่มีประเภทเดียวกัน	18
การส่งกลับค่าหลายค่า	18
ค่าที่ส่งกลับมีชื่อ	18
ฟังก์ชันไม่ระบุชื่อและฟังก์ชันระดับสูง (Higher-Order Functions)	19
ฟังก์ชันไม่ระบุชื่อ	19
ฟังก์ชันระดับสูง	19
ฟังก์ชัน Variadic	20
defer, panic และ recover	21
defer	21
panic และ recover	21
บทที่ 5 คุณสมบัติเด่นในภาษา Go	23
Goroutines การทำงานพร้อมกัน	23
Channels การสื่อสารระหว่าง Goroutines	24
กลไก defer, panic และ recover	25
ใช้ defer รับประกันการคืนค่าทรัพยากร	25
ใช้ panic ในการส่งสัญญาณข้อผิดพลาด	26

ใช้ recover ในการจัดการกับ Panic	26
บทที่ 6 การทำงานกับข้อมูล	29
Arrays และ Slices	29
Arrays ใน Go	29
Slices: อาร์เรย์ที่ยืดหยุ่นและสามารถปรับขนาดได้	30
Maps: คู่คีย์และค่า	31
Structs: ประเภทข้อมูลที่กำหนดเอง	32
บทที่ 7 การเขียนโปรแกรมเชิงวัตถุ	35
เมธอดและอินเตอร์เฟซ	35
เมธอดในภาษา Go	35
อินเตอร์เฟซในภาษา Go	36
การฝัง (Embedding) และการรวมกัน (Composition)	38
พอลิมอร์ฟิซึมใน Go	39
บทที่ 8 การจัดการข้อผิดพลาด	42
ประเภทข้อผิดพลาดใน Go	42
ประเภท error	42
การสร้างข้อผิดพลาดแบบกำหนดเอง	43
การตรวจสอบข้อผิดพลาด	44
การห่อหุ้มข้อผิดพลาด (Error Wrapping)	45
แนวทางที่ดีที่สุดในการจัดการข้อผิดพลาด	46
จัดการข้อผิดพลาดอย่างชัดเจน	46
คืนค่าข้อผิดพลาดที่มีข้อมูลครบถ้วน	47

ใช้ข้อผิดพลาด Sentinel สำหรับปัญหาที่รู้จัก	47
จัดการข้อผิดพลาดให้เร็วที่สุด	47
หลีกเลี่ยงการใช้ Panic สำหรับข้อผิดพลาดที่สามารถจัดการได้	48
ใช้ defer สำหรับการคืนทรัพยากร	48
การสร้างแอปพลิเคชันที่ทนทาน	49
การลดความสามารถในการทำงาน	49
การลองใหม่ (retry)	49
ใช้ Context สำหรับการหมดเวลา	50
บทที่ 9 การทำงานกับไฟล์	52
การอ่านและเขียนไฟล์	52
การเปิดไฟล์	52
การอ่านไฟล์	53
การเขียนไฟล์	55
การทำงานกับ JSON และ XML	57
การทำงานกับ JSON	58
การทำงานกับ XML	59
การจัดการอินพุต	61
การใช้ os.Args	62
การใช้แฟลคเกจ flag	62
บทที่ 10 การทำงานกับเครือข่าย	65
การส่งคำขอ HTTP	65
การส่งคำขอ GET	65

การส่งคำขอ POST, PUT และ DELETE	66
การสร้างเว็บเซิร์ฟเวอร์ง่ายๆ	68
การสร้างเว็บเซิร์ฟเวอร์พื้นฐาน	68
การจัดการเส้นทางที่แตกต่างกัน	69
การทำงานกับ REST API	70
การใช้งาน REST API	70
การสร้าง REST API ด้วย Go	72
บทที่ 11 การจัดการโมดูลและแพ็คเกจ	74
การสร้างและใช้ Go Modules	74
การเริ่มต้นใช้งาน Go Modules	74
การเพิ่ม dependencies	75
การดูและอัปเดต dependencies	76
จัดการ Dependencies ให้เรียบร้อย	76
การจัดการ Dependencies	77
ไฟล์ go.mod	77
ไฟล์ go.sum	78
การจัดการหลายเวอร์ชัน	78
การจัดโครงสร้างโปรเจกต์ภาษา Go	79
ไคเรกทอรี cmd	80
ไคเรกทอรี internal	80
ไคเรกทอรี pkg	80
การทำงานกับ Version Control	80

การคอมมิตต์ go.mod และ go.sum	81
บทที่ 12 การทดสอบ	82
การเขียนยูนิตเทส	82
ตัวอย่างยูนิตเทสพื้นฐาน	82
การรันทดสอบ	83
ผลลัพธ์ของการทดสอบ	84
การวัดประสิทธิภาพของโค้ด	84
การเขียนการทดสอบการวัดประสิทธิภาพ	84
การรันการทดสอบการวัดประสิทธิภาพ	85
การวัดประสิทธิภาพด้วยตัวเลือก	86
การใช้เครื่องมือทดสอบในภาษา Go	86
การครอบคลุมการทดสอบ	87
การทดสอบแบบ Table-Driven	87
การ Mocking ขึ้นอยู่กับพฤติกรรม	88
บทที่ 13 ตัวอย่างโปรเจกต์	90
เครื่องมือ CLI	90
การสร้างเครื่องมือ CLI ง่ายๆ	90
การปรับปรุงเครื่องมือ CLI	91
เว็บแอปพลิเคชันง่ายๆ	91
การตั้งค่าเว็บเซิร์ฟเวอร์พื้นฐาน	91
การเพิ่มเส้นทางอื่น	92
การให้บริการไฟล์สแตติก	93

การดีพลอยแอปพลิเคชัน	93
การคอมไพล์แอปพลิเคชัน	93
การดีพลอยเว็บแอปพลิเคชัน	93
การดีพลอยด้วยเครื่องมือ CLI	94
บทที่ 14 หัวข้อขั้นสูงสำหรับผู้เริ่มต้น	95
การสะท้อน (Reflection) ในภาษา Go	95
การสะท้อนคืออะไร?	95
ตัวอย่างการใช้การสะท้อน	95
การปรับค่าโดยใช้การสะท้อน	96
การทำงานกับ Go Runtime	97
ฟังก์ชันใน Go Runtime	97
แนะนำ Generics ใน Go	99
Generics คืออะไร?	99
ตัวอย่างฟังก์ชัน Generic	100
ชนิดข้อมูล Generic	100
บทที่ 15 เรียนรู้เพิ่มเติม	103
การมีส่วนร่วมในชุมชนโอเพ่นซอร์ส	103
ทำไมต้องมีส่วนร่วมในโอเพ่นซอร์ส?	103
ที่ไหนบ้างที่สามารถค้นหาโครงการ Go โอเพ่นซอร์ส	104
วิธีเริ่มมีส่วนร่วม	104
การสำรวจไลบรารีและเฟรมเวิร์กของ Go	105
ไลบรารีและเฟรมเวิร์ก Go ที่ได้รับความนิยม	105

วิธีการสำรวจไลบรารีเพิ่มเติม	107
การเข้าร่วมชุมชนภาษา Go	107
วิธีการเข้าร่วมชุมชน Go	107
การมีส่วนร่วมในการพัฒนาภาษา Go	109

บทที่ 1 แนะนำภาษา Go

ทำไมต้องภาษา Go?

Go หรือที่มักเรียกกันว่า Golang เป็นภาษาการเขียนโปรแกรมสมัยใหม่ที่ถูกออกแบบมาให้มีความเรียบง่ายและมีประสิทธิภาพสูง สร้างขึ้นที่ Google โดย Go ผสมผสานคุณสมบัติที่ดีที่สุดจากภาษาการเขียนโปรแกรม เช่น C และ Python ภาษา Go มีไวยากรณ์ที่เรียบง่ายและเรียนรู้ได้ง่าย มุ่งเน้นการลดความซับซ้อน ทำให้นักพัฒนาทั้งผู้เริ่มต้นและผู้มีประสบการณ์สามารถเรียนรู้และใช้งานได้อย่างรวดเร็ว โค้ดที่คอมไพล์ด้วย Go มีประสิทธิภาพสูง และให้ความเร็วใกล้เคียงกับภาษา C โดยยังคงความเรียบง่ายไว้

หนึ่งในคุณสมบัติเด่นของ Go คือการรองรับการทำงานพร้อมกัน (Concurrency) ผ่าน Goroutines และ Channels ซึ่งช่วยให้สามารถดำเนินการหลายงานพร้อมกันได้อย่างมีประสิทธิภาพ นอกจากนี้ ภาษา Go ยังเหมาะสำหรับโปรเจกต์ทุกขนาด ตั้งแต่เครื่องมือขนาดเล็กไปจนถึงระบบขนาดใหญ่ และรองรับการทำงานในสภาพแวดล้อมยุคใหม่ ไลบรารีมาตรฐานของ Go มาพร้อมกับเครื่องมือและพีเอชเออร์ที่มีคุณภาพสูง ทำให้การพัฒนาซอฟต์แวร์เป็นไปอย่างสะดวกและไม่จำเป็นต้องพึ่งพาไลบรารีภายนอก

ความเรียบง่ายและประสิทธิภาพในการพัฒนาโปรแกรม

ภาษา Go ถูกสร้างขึ้นในปี 2007 โดย Robert Griesemer, Rob Pike และ Ken Thompson ที่ Google ด้วยเป้าหมายในการแก้ไขปัญหาดังต่อไปนี้ที่พบใน