สถาบันไอทีจีเนียส
IT GENIUS INSTITUTE

COURSE ID
WDC-85
PROGRAMMING / JAVA

COURSE OUTLINE · รายละเอียดหลักสูตรอบรม

Java OOP

ออกแบบและเขียน Java แบบเชิงวัตถุอย่างถูกหลัก ด้วย Java 25 LTS, SOLID และ Design Pattern

- 18 ชั่วโมง
- 3 วัน
- พื้นฐาน **เขียนภาษา Java ได้**
- ระดับ **Intermediate**



ระยะเวลา: 3 วัน
18 ชั่วโมง (3 วัน วันละ 6 ชม.) · 09:00-16:00 น.

รูปแบบ: Classroom / Hybrid
Onsite หรือ Online ผ่าน Microsoft Teams · Public / In-house

รายละเอียดหลักสูตร
www.itgenius.co.th

ผู้ที่เขียน Java ได้แล้วจำนวนมากยังเขียนโปรแกรมแบบรวมทุกอย่างไว้ใน Class ใหญ่ไม่กี่ตัว พอระบบโตขึ้นก็แก้ยาก ทดสอบยาก และนำโค้ดกลับมาใช้ซ้ำไม่ได้ การคิดแบบ Object Oriented ที่ถูกต้องช่วยแบ่งความรับผิดชอบของโค้ดให้ชัด และ Java รุ่นใหม่ยังเพิ่มเครื่องมือที่ทำให้ออกแบบได้ดีขึ้น เช่น Record, Sealed Class และ Pattern Matching ซึ่งตำรา OOP รุ่นเก่ายังไม่ได้พูดถึง

หลักสูตรนี้พาเรียนหลักการ OOP บน Java 25 LTS อย่างลึกและนำไปใช้ได้จริง เริ่มจากการคิดแบบ Object การออกแบบ Class, Abstraction, Encapsulation, Inheritance และ Polymorphism ต่อด้วย Abstract Class กับ Interface, Record, Sealed Class, Pattern Matching, Composition และ Generics จากนั้นยกระดับสู่การออกแบบด้วยหลัก SOLID, Class Diagram และ Design Pattern ที่ใช้บ่อย เขียน Unit Test ด้วย JUnit 6 เพื่อ Refactor อย่างมั่นใจ แล้วปิดท้ายด้วย Capstone ออกแบบและสร้างระบบด้วย OOP ตั้งแต่ต้นจนจบ (อบรม 3 วัน วันละ 6 ชั่วโมง รวม 18 ชั่วโมง ระดับ Intermediate)

เครื่องมือ: Java 25 LTS · OOP · Design Patterns · JUnit 6 · Maven

ราคาค่าอบรม · PUBLIC / IN-HOUSE

9,500 บาท / ท่าน

สมัครผ่านเว็บไซต์ ลด 10%

ราคายังไม่รวมภาษีมูลค่าเพิ่ม (VAT 7%) · รวมเอกสารประกอบการอบรม Lab Guide ซอร์สโค้ดตัวอย่างและโปรเจกต์ Maven ครบทุก Lab และ Certificate · รับจัด In-house ขอใบเสนอราคาได้

เพิ่มเพื่อน LINE

02-570-8449 · 088-807-9770 · @itgenius · www.itgenius.co.th

01 วัตถุประสงค์ Objectives

- เข้าใจเหตุผลของการเขียนโปรแกรมแบบ OOP และความต่างจากแบบ Top-Down
- ออกแบบ Class และ Object ที่มีหน้าที่ชัดเจน พร้อม Abstraction และ Encapsulation
- ใช้ Inheritance, Polymorphism, Overloading, Overriding และ final ได้ถูกที่
- เลือกใช้ Abstract Class, Interface, Record และ Sealed Class ให้เหมาะกับงาน
- ใช้ Composition และ Generics เพื่อให้โค้ดยืดหยุ่นและนำกลับมาใช้ซ้ำได้
- ออกแบบตามหลัก SOLID และอ่านเขียน Class Diagram ได้
- ประยุกต์ใช้ Design Pattern ที่พบบ่อยในงานจริงได้
- เขียน Unit Test ด้วย JUnit 6 และ Refactor โค้ด OOP ได้อย่างปลอดภัย

02 กลุ่มเป้าหมาย Target Audience

- ผู้ที่เขียน Java ได้แล้วและต้องการเขียนโปรแกรมแบบเชิงวัตถุอย่างถูกหลัก
- Developer ที่ต้องการออกแบบโค้ดให้ดูแลและขยายต่อได้ง่าย
- ผู้ที่จะต่อยอดไปใช้ Framework อย่าง Spring Boot หรือ Jakarta EE
- นักศึกษาและครูอาจารย์สายคอมพิวเตอร์ที่ต้องการเข้าใจ OOP และ Design Pattern
- IT Manager และ Tech Lead ที่ต้องรีวิวโค้ดและวางแนวทางการออกแบบของทีม

03 จุดเด่นของหลักสูตร Highlights

- **อัปเดตเป็น Java 25 LTS** Record, Sealed Class และ Pattern Matching
- **OOP ครบทุกหลัก** Abstraction, Encapsulation, Inheritance, Polymorphism
- **จากหลักการสู่การออกแบบ** SOLID, Composition และ Class Diagram
- **Design Pattern ที่ใช้จริง** Factory, Builder, Strategy, Observer และอื่น ๆ
- **โค้ดที่ทดสอบได้** JUnit 6 และการ Refactor อย่างปลอดภัย

04 ความรู้พื้นฐาน Prerequisites

- เขียนภาษา Java ได้ เช่น ตัวแปร เงื่อนไข ลูป Method และ ArrayList
- เคยสร้าง Class และ Object มาบ้าง แม้ยังไม่คล่อง
- ใช้ IDE อย่าง IntelliJ IDEA หรือ VS Code ได้เบื้องต้น
- โน้ตบุ๊กที่ติดตั้ง JDK 25 และ IDE ได้ พร้อมสิทธิ์ติดตั้งโปรแกรม

05 สิ่งที่ต้องเตรียมและเครื่องมือ Setup & Tools

รายการ	รายละเอียด
อุปกรณ์	Notebook Windows, macOS หรือ Linux เชื่อมต่ออินเทอร์เน็ต พร้อมสิทธิ์ติดตั้งโปรแกรม
ภาษาและ JDK	Java 25 LTS (OpenJDK เช่น Eclipse Temurin)
IDE	IntelliJ IDEA Community หรือ VS Code พร้อม Extension Pack for Java
Build และทดสอบ	Maven, JUnit 6 และ Mockito
ออกแบบ	PlantUML หรือ Mermaid สำหรับวาด Class Diagram และ Git

หมายเหตุ: หลักสูตรนี้ปรับปรุงใหม่ปี 2026 จากฉบับเดิมที่สอนหลัก OOP บน Java รุ่นเก่า เนื้อหาเปลี่ยนเป็น Java 25 LTS ทั้งหมด คงหัวข้อเดิมเรื่อง Class, Abstraction, Encapsulation, Inheritance, Polymorphism, final, Abstract Class และ Interface ไว้ แล้วเพิ่ม Record, Sealed Class, Pattern Matching, Flexible Constructor Bodies, Composition, หลัก SOLID, Design Pattern และการเขียน Unit Test ด้วย JUnit 6 เครื่องมือทั้งหมดเป็นโอเพนซอร์สหรือใช้ฟรี หลักสูตรนี้ต่อยอดจาก Basic Java Programming ที่สอนไวยากรณ์และ OOP ขั้นต้น โดยเน้นการออกแบบเชิงวัตถุให้ลึกขึ้น ผู้ที่ยังไม่เคยเขียน Java ควรเริ่มจากหลักสูตร Basic Java Programming หรือ Java for Non-Programmers

06 ภาพรวมการเรียน 3 วัน Big Picture

วัน	หัวข้อหลัก
Day 1 Section 1-5	คิดแบบ OOP เทียบกับแบบ Top-Down, ออกแบบ Class และ Object, Constructor แบบใหม่ใน Java 25, Abstraction และ Encapsulation, Inheritance, Polymorphism และการใช้ final
Day 2 Section 6-10	Abstract Class กับ Interface, Record, Sealed Class และ Pattern Matching, Composition แทน Inheritance, Generics กับ Collections ในการออกแบบ และออกแบบ Exception ให้เป็นระบบ
Day 3 Section 11-15	หลัก SOLID และ Class Diagram, Design Pattern กลุ่ม Creational, Structural และ Behavioral, เขียน Unit Test และ Refactor ด้วย JUnit 6 แล้วปิดท้ายด้วย Capstone ออกแบบระบบด้วย OOP

DAY 1

หลักการ OOP บน Java 25

เนื้อหาการอบรม · Section 1-5

รวม
6 ชั่วโมง

1 Lab: แนวคิด OOP และเตรียมเครื่องมือ

- ทำไมต้องเขียนแบบ OOP ข้อดีข้อเสียเทียบกับแบบ Top-Down
- คิดแบบ Object: แยกสิ่งของ หน้าที่ และความสัมพันธ์จากโจทย์
- ติดตั้ง JDK 25, IntelliJ IDEA หรือ VS Code และสร้างโปรเจกต์ Maven
- Lab: วิเคราะห์โจทย์ระบบห้องสมุดเป็นรายการ Class เบื้องต้น

2 Class, Object และ Constructor

- Field, Method และ Object Reference ในหน่วยความจำ
- Constructor, Overloading ของ Constructor และ this
- Flexible Constructor Bodies ใน Java 25 สำหรับตรวจค่าก่อนเรียก super
- Static Member เทียบกับ Instance Member
- Lab: สร้าง Class หนังสือและสมาชิกพร้อมตรวจความถูกต้องของข้อมูล

3 Abstraction และ Encapsulation

- Abstraction: เปิดเผยเฉพาะสิ่งที่ผู้ใช้ Class ต้องรู้
- ซ่อนสถานะด้วย private และ Access Modifier อื่น ๆ
- Getter, Setter และเมื่อใดไม่ควรมี Setter
- Immutable Object และ Package สำหรับจัดกลุ่ม Class
- Lab: ปรับ Class บัญชีเงินฝากให้ปกป้องยอดเงินจากการแก้ไขผิดวิธี

4 Lab: Inheritance

- หลักการสืบทอดและความสัมพันธ์แบบ is-a
- super, protected และลำดับการสร้าง Object
- Override เมธอด toString, equals และ hashCode ให้ถูกต้อง
- ข้อควรระวังของลำดับชั้นการสืบทอดที่ลึกเกินไป
- Lab: ลำดับชั้นของสื่อในห้องสมุด เช่น หนังสือ นิตยสาร และ E-book

5 Polymorphism และ final

- Overloading กับ Overriding ต่างกันอย่างไร
- Dynamic Dispatch และการอ้างอิงผ่านชนิดของคลาสแม่
- final กับตัวแปร Method และ Class
- Lab: คำนวณค่าปรับคืนช้าที่ต่างกันตามชนิดสื่อด้วย Polymorphism

DAY 2

เครื่องมือออกแบบของ Java สมัยใหม่

เนื้อหาการอบรม · Section 6-10

รวม 6 ชั่วโมง

6 Abstract Class และ Interface

- Abstract Class ต่างจาก Class ทั่วไปอย่างไร
- Interface พร้อม default, static และ private Method
- เลือกระหว่าง Abstract Class กับ Interface
- Functional Interface และ Lambda ในมุมของ OOP
- Lab: ออกแบบ Interface ช่องทางแจ้งเตือนหลายแบบ

7 Lab: Record, Sealed Class และ Pattern Matching

- Record สำหรับข้อมูลที่ไม่เปลี่ยนค่า และ Compact Constructor
- Sealed Class และ Sealed Interface จำกัดชนิดลูกที่อนุญาต
- Pattern Matching กับ instanceof และ switch
- Record Pattern และ Unnamed Pattern
- Lab: โมเดลการชำระเงินหลายรูปแบบด้วย Sealed Interface และ switch

8 Composition แทน Inheritance

- ความสัมพันธ์แบบ has-a, Association และ Aggregation
- Delegation และเหตุผลที่ควรเลือก Composition ก่อน
- Enum ที่มีพฤติกรรมของตัวเอง
- Lab: Refactor ลำดับชั้นที่ซับซ้อนให้เป็น Composition

9 Generics และ Collections ในการออกแบบ

- Generic Class และ Generic Method
- Bounded Type และ Wildcard
- Comparable กับ Comparator สำหรับเรียงลำดับ Object
- เลือก List, Set หรือ Map ให้เหมาะกับโมเดลข้อมูล
- Lab: Repository แบบ Generic สำหรับข้อมูลในระบบห้องสมุด

10 Lab: ออกแบบ Exception ให้เป็นระบบ

- Checked กับ Unchecked Exception และเมื่อใดใช้แบบไหน
- สร้าง Custom Exception ที่สื่อความหมายทางธุรกิจ
- try-with-resources และการไม่กลืน Error
- ใช้ Optional แทนการคืนค่า null
- Lab: จัดการกรณียืมหนังสือเกินสิทธิ์และหนังสือไม่ว่าง

DAY 3

หลักการออกแบบ Design Pattern และ Test

เนื้อหาการอบรม · Section 11-15

รวม 6 ชั่วโมง

11 หลัก SOLID และ Class Diagram

- Single Responsibility และ Open/Closed
- Liskov Substitution, Interface Segregation และ Dependency Inversion
- Coupling และ Cohesion
- อ่านและเขียน Class Diagram ด้วย PlantUML หรือ Mermaid
- Lab: วิเคราะห์โค้ดที่ผิดหลัก SOLID แล้วเสนอการแก้

12 Design Pattern กลุ่ม Creational

- Design Pattern คืออะไร และใช้อย่างไรไม่ให้เกินจำเป็น
- Factory Method และ Static Factory
- Builder สำหรับ Object ที่มีค่าหลายตัว
- Singleton และข้อควรระวัง
- Lab: สร้าง Object สมาชิกหลายประเภทด้วย Factory และ Builder

13 Lab: Pattern กลุ่ม Structural และ Behavioral

- Adapter และ Decorator
- Strategy และ Template Method
- Observer สำหรับแจ้งเหตุการณ์
- เขียน Pattern ให้กระชับด้วย Lambda, Record และ Sealed Class
- Lab: ใช้ Strategy คำนวณค่าปรับ และ Observer แจ้งเตือนหนังสือว่าง

14 Lab: Unit Test และ Refactoring

- เขียน Unit Test ด้วย JUnit 6 และ Assertion ที่อ่านง่าย
- Mock Dependency ด้วย Mockito
- Code Smell ที่พบบ่อยในโค้ด OOP
- Refactor ทีละขั้นโดยมี Test คุ้มครอง
- ใช้ AI Assistant ช่วยรีวิวการออกแบบพร้อมตรวจทานผล

15 Workshop: Capstone ออกแบบระบบด้วย OOP

- วิเคราะห์โจทย์ระบบจองหรือระบบยืมคืนที่ผู้เรียนเลือก
- วาด Class Diagram และเลือก Pattern ที่เหมาะสม
- ลงมือสร้างพร้อม Unit Test ครอบคลุมส่วนสำคัญ
- นำเสนอการออกแบบ รีวิวร่วมกัน และสรุปเส้นทางต่อยอด

07 เรียนจบแล้วได้อะไร Outcomes

- **คิดแบบ OOP ได้** แยก Class และหน้าที่จากโจทย์
- **ใช้หลัก OOP ถูกที่** Encapsulation, Inheritance, Polymorphism
- **ใช้ Java สมัยใหม่** Record, Sealed Class และ Pattern Matching
- **ออกแบบโค้ดที่ดี** SOLID, Composition และ Class Diagram
- **ใช้ Design Pattern ได้** Creational, Structural และ Behavioral
- **โค้ดทดสอบได้** JUnit 6, Mockito และ Refactoring

08 สิ่งที่ผู้เรียนจะได้รับ Deliverables

- เอกสารประกอบการอบรมฉบับสมบูรณ์ครบทั้ง 3 วัน และ Lab Guide ทีละขั้นตอน
- ซอร์สโค้ดตัวอย่างและโปรเจกต์ Maven ครบทุก Lab รวมถึงตัวอย่าง Design Pattern
- Class Diagram และโค้ดของ Capstone พร้อม Unit Test
- ประกาศนียบัตรจบหลักสูตร (Certificate of Completion) และช่องทางสอบถามวิทยากรหลังจบหลักสูตร

09 ข้อมูลผู้สอน Instructor

อ.สามิตร โกยม ผู้สอนและนักพัฒนาซอฟต์แวร์ที่ใช้ Java และการออกแบบเชิงวัตถุในการพัฒนาระบบจริง จุดเด่นคือการสอนผ่านโจทย์ระบบเดียวที่ค่อย ๆ ขยาย เพื่อให้เห็นว่าแต่ละหลักการแก้ปัญหาอะไร เปรียบเทียบโค้ดก่อนและหลังออกแบบให้เห็นชัด และพาผู้เรียนลงมือคิด ออกแบบ และรีวิวโค้ดร่วมกัน เพื่อนำวิธีคิดไปใช้กับงานของตนเองได้ทันที

ราคาค่าอบรม 9,500 บาท/ท่าน · สนใจ IN-HOUSE ขอใบเสนอราคา

9,500 บาท / ท่าน (ยังไม่รวม VAT 7%)

สมัครผ่านเว็บไซต์ ลด 10%

02-570-8449 · 088-807-9770

@itgenius · contact@itgenius.co.th

เพิ่มเพื่อน LINE