สถาบันไอทีจีเนียส
IT GENIUS INSTITUTE

COURSE ID
WDC-86

COURSE OUTLINE · รายละเอียดหลักสูตรอบรม

WEB DEVELOPMENT / JAVASCRIPT

JavaScript OOP

ออกแบบโค้ด JavaScript แบบ OOP และ Design Pattern ให้ดูแลง่าย

- 18 ชั่วโมง
- 3 วัน
- พื้นฐาน **เขียน JavaScript ได้แล้ว**
- 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

เมื่อโปรเจกต์ JavaScript โตขึ้น โค้ดที่เขียนเป็นฟังก์ชันต่อกันยาว ๆ มักแก้ยาก ใช้ซ้ำยาก และพังง่ายเมื่อมีคนแก้หลายคน การคิดแบบ Object Oriented ช่วยแบ่งระบบเป็นชิ้นที่มีหน้าที่ชัดเจน ซ่อนรายละเอียดที่ไม่จำเป็น และต่อยอดได้โดยไม่ต้องรื้อของเดิม แนวคิดนี้อยู่เบื้องหลัง Framework และไลบรารีที่นักพัฒนาใช้ทุกวัน

หลักสูตรนี้พาเข้าใจ OOP ในแบบของ JavaScript ซึ่งทำงานบน Prototype ตั้งแต่ Object, Prototype และ this เขียน Class ตามมาตรฐาน ECMAScript ล่าสุด ฝึก Encapsulation ด้วย Private Element, Abstraction, Inheritance และ Polymorphism เลือกใช้ Composition แทนการสืบทอดเมื่อเหมาะสม ออกแบบตามหลัก SOLID และใช้ TypeScript เพิ่ม Interface กับ Abstract Class จากนั้นเรียน Design Pattern ทั้งกลุ่ม Creational, Structural และ Behavioral เขียน Test และ Refactor โค้ดเดิม แล้วปิดท้ายด้วย Capstone ออกแบบระบบด้วย Pattern (อบรม 3 วัน วันละ 6 ชั่วโมง รวม 18 ชั่วโมง ระดับ Intermediate)

เครื่องมือ

- JS ECMAScript 2026
- TS TypeScript
- N Node.js LTS
- GoF Design Patterns
- UML Class Diagram

ราคาค่าอบรม · PUBLIC / IN-HOUSE

9,500 บาท / ท่าน

สมัครผ่านเว็บไซต์ ลด 10%

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

เพิ่มเพื่อน LINE

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

01 วัตถุประสงค์ Objectives

- อธิบายแนวคิด OOP และความต่างระหว่าง Prototype กับ Class ใน JavaScript ได้
- เขียน Class พร้อม Private Element, Static Member, Getter และ Setter ได้ถูกต้อง
- ใช้หลัก Encapsulation, Abstraction, Inheritance และ Polymorphism ออกแบบโค้ดได้
- เลือกใช้ Composition หรือ Inheritance ได้เหมาะกับปัญหา
- ออกแบบ Class ตามหลัก SOLID และสื่อสารด้วย Class Diagram ได้
- ใช้ TypeScript เพิ่ม Interface, Abstract Class และ Access Modifier ให้โค้ด OOP ได้
- นำ Design Pattern ที่ใช้บ่อยมาแก้ปัญหาในงาน JavaScript ได้
- เขียน Test และ Refactor โค้ดเดิมให้เป็นโครงสร้างแบบ OOP ได้

02 กลุ่มเป้าหมาย Target Audience

- Developer ที่เขียน JavaScript ได้แล้วและต้องการออกแบบโค้ดให้เป็นระบบ
- Front-end และ Full-stack Developer ที่ต้องดูแลโปรเจกต์ขนาดกลางถึงใหญ่
- ผู้ที่ต้องการเข้าใจแนวคิดเบื้องหลัง Framework อย่าง Angular, NestJS และไลบรารีต่าง ๆ
- IT Manager และ Tech Lead ที่ต้องวางโครงสร้างโค้ดและรีวิวงานของทีม

03 จุดเด่นของหลักสูตร Highlights

- **OOP ในแบบของ JavaScript** Prototype, Class และ Private Element
- **ออกแบบเป็นระบบ** SOLID, Composition และ Class Diagram
- **เสริมด้วย TypeScript** Interface และ Abstract Class
- **Design Pattern ที่ใช้จริง** Creational, Structural และ Behavioral
- **ปิดท้ายด้วย Capstone** ออกแบบระบบพร้อม Test

04 ความรู้พื้นฐาน Prerequisites

- เขียน JavaScript ได้ ทั้งตัวแปร ฟังก์ชัน Array, Object และ Arrow Function
- เคยใช้ import และ export หรือไลบรารีจาก npm มาบ้าง
- ใช้ Command Line และ Git ขั้นพื้นฐานได้
- โน้ตบุ๊กที่ติดตั้ง Node.js รุ่น LTS และ VS Code ได้ พร้อมสิทธิ์ติดตั้งโปรแกรม

05 สิ่งที่ต้องเตรียมและเครื่องมือ Setup & Tools

รายการ	รายละเอียด
อุปกรณ์	Notebook เชื่อมต่ออินเทอร์เน็ต พร้อมสิทธิ์ติดตั้งโปรแกรม
Runtime	Node.js รุ่น LTS ปัจจุบัน และเบราว์เซอร์รุ่นล่าสุดพร้อม DevTools
ภาษาและเครื่องมือ	JavaScript ตามมาตรฐาน ECMAScript ล่าสุด, TypeScript, VS Code และ ESLint
ทดสอบ	Node.js Test Runner และ node:assert
ออกแบบ	Class Diagram ด้วย Mermaid หรือเครื่องมือวาด UML แบบใช้ฟรี

หมายเหตุ: เครื่องมือทั้งหมดเป็นโอเพนซอร์สหรือใช้ฟรี หลักสูตรนี้ปรับปรุงใหม่ปี 2026 ใช้ไวยากรณ์ Class ตามมาตรฐาน ECMAScript ล่าสุด เช่น Private Method, Static Block และการตรวจด้วย #field in obj เพิ่มหลัก SOLID, Composition, Mixin, TypeScript สำหรับ Interface และ Abstract Class และการเขียน Test ปรับตัวอย่าง Design Pattern ให้ใช้ ES Modules, EventTarget และ Proxy แทนวิธีเดิมที่จำลองด้วยฟังก์ชัน หัวข้อ Class และ Prototype ทับซ้อนกับหลักสูตร Advanced JavaScript บางส่วน หลักสูตรนี้เน้นการออกแบบเชิงวัตถุและ Design Pattern ส่วน Asynchronous, Node.js API และ Express อยู่ในหลักสูตร Advanced JavaScript

06 ภาพรวมการเรียน 3 วัน Big Picture

วัน	หัวข้อหลัก
Day 1 Section 1-5	เข้าใจแนวคิด OOP และเตรียมเครื่องมือ, Object, Prototype และ this, เขียน Class ตามมาตรฐาน ECMAScript ล่าสุด, ห่อหุ้มข้อมูลด้วย Private Element และออกแบบ Abstraction ที่ใช้งานง่าย
Day 2 Section 6-10	สืบทอดด้วย extends และ super, Polymorphism, Composition และ Mixin, หลักการ SOLID กับ Class Diagram และใช้ TypeScript เพิ่ม Interface กับ Abstract Class ให้โค้ด OOP
Day 3 Section 11-15	Design Pattern กลุ่ม Creational, Structural และ Behavioral ในแบบ JavaScript สมัยใหม่, เขียน Test และ Refactor โค้ดเดิมให้เป็น OOP แล้วปิดท้ายด้วย Capstone ออกแบบระบบด้วย Pattern

DAY 1

พื้นฐาน OOP ใน JavaScript

เนื้อหาการอบรม · Section 1-5

รวม
6 ชั่วโมง

1 Lab: แนวคิด OOP และเตรียมเครื่องมือ

- ทำไมต้องเขียนแบบ OOP ข้อดี ข้อจำกัด และความต่างกับแบบ Procedural
- หลักการคิดแบบ Object และการแยกหน้าที่ของแต่ละ Class
- ติดตั้ง Node.js รุ่น LTS, VS Code และ ESLint
- จัดโปรเจกต์ด้วย ES Modules และรัน Test ด้วย node --test
- Lab: โปรเจกต์ตั้งต้นที่ใช้ต่อทุก Lab

2 Object, Prototype และ this

- Object Literal, Method และ Shorthand
- Constructor Function และ new ทำงานอย่างไร
- Prototype Chain, Object.create และ Object.getPrototypeOf
- ค่า this ใน Method, Callback และ Arrow Function
- Lab: สร้าง Object สินค้าด้วยสามวิธีแล้วเปรียบเทียบ

3 Lab: Class และ Object

- Class Declaration, Constructor, Field และ Method
- Static Field, Static Method และ Static Initialization Block
- Getter และ Setter สำหรับตรวจค่าก่อนเก็บ
- Class ทำงานบน Prototype อย่างไรเบื้องหลัง
- Lab: Class บัญชีลูกค้าและคำสั่งซื้อ

4 Encapsulation

- ห่อหุ้มข้อมูลด้วย Private Field และ Private Method
- ตรวจว่าเป็น Object ของ Class ด้วย #field in obj
- Closure และ Module Pattern เป็นอีกทางของการซ่อนข้อมูล
- Object.freeze และการออกแบบ Object ที่แก้ไขไม่ได้
- Lab: Class กระเป๋าเงินที่ป้องกันยอดเงินถูกแก้ตรง ๆ

5 Abstraction

- ออกแบบ Public API ของ Class ให้ใช้ง่ายและซ่อนรายละเอียด
- แนวคิด Abstract Class ใน JavaScript ที่ไม่มีคีย์เวิร์ด abstract
- บังคับให้ Class ลูกเขียน Method ด้วย new.target และการโยน Error
- Duck Typing และการกำหนดสัญญาของ Object ด้วย JSDoc
- Lab: Class แม่ของช่องทางชำระเงินหลายแบบ

DAY 2

Inheritance, Polymorphism และหลักการออกแบบ

เนื้อหาการอบรม · Section 6-10

รวม
6 ชั่วโมง

6 Lab: Inheritance

- สืบทอดด้วย extends และเรียก Class แม่ด้วย super
- Method Overriding และการต่อยอด Method เดิม
- สืบทอดจาก Built-in เช่น Error เพื่อสร้าง Custom Error
- instanceof และความสัมพันธ์ของ Class แบบต่าง ๆ
- Lab: ลำดับชั้นของพนักงานและการคำนวณเงินเดือน

7 Polymorphism

- Polymorphism คืออะไร และช่วยลด if-else ได้อย่างไร
- Overriding เทียบกับ Overloading ที่ JavaScript ไม่มีโดยตรง
- ทำ Object ให้ทำงานกับภาษาด้วย Symbol.iterator และ toString
- เรียกใช้ Object หลายชนิดผ่าน Interface เดียวกัน
- Lab: ระบบคำนวณค่าขนส่งหลายรูปแบบ

8 Composition และ Mixin

- ปัญหาของการสืบทอดหลายชั้น
- Composition over Inheritance และการประกอบ Object จากส่วนย่อย
- Mixin สำหรับเพิ่มความสามารถให้หลาย Class
- Dependency Injection เพื่อให้เปลี่ยนส่วนประกอบและทดสอบได้ง่าย
- Lab: ปรับลำดับชั้น Class ให้เป็น Composition

9 หลักการ SOLID และ Class Diagram

- Single Responsibility และ Open/Closed
- Liskov Substitution, Interface Segregation และ Dependency Inversion
- อ่านและเขียน Class Diagram ด้วย UML
- สร้าง Diagram จากโค้ดด้วย Mermaid
- Lab: วิเคราะห์โค้ดที่ผิดหลัก SOLID และออกแบบใหม่

10 Lab: OOP ด้วย TypeScript

- ทำไมทีมนิยมใช้ TypeScript กับโค้ด OOP
- Interface และ implements
- Abstract Class, Abstract Method และ Access Modifier
- Parameter Property, readonly และ Generic Class เบื้องต้น
- Lab: แปลงระบบชำระเงินจาก Section 5 เป็น TypeScript

DAY 3

Design Pattern, Test และ Capstone

เนื้อหาการอบรม · Section 11-15

รวม
6 ชั่วโมง

11 Design Pattern และกลุ่ม Creational

- ที่มาของ Design Pattern จาก Gang of Four และวิธีเลือกใช้
- Singleton และทางเลือกด้วย ES Module
- Factory Method สำหรับสร้าง Object ตามเงื่อนไข
- Builder สำหรับ Object ที่มีค่าตั้งต้นหลายตัว
- Lab: Factory สร้างช่องทางแจ้งเตือนหลายแบบ

12 Lab: Structural Pattern

- Composite สำหรับข้อมูลแบบต้นไม้ เช่น เมนูและหมวดหมู่
- Adapter สำหรับเชื่อมกับ API หรือไลบรารีที่หน้าตาต่างกัน
- Decorator สำหรับเพิ่มความสามารถโดยไม่แก้ Class เดิม
- Facade สำหรับซ่อนระบบย่อยที่ซับซ้อน
- Proxy ด้วย Proxy Object ของ JavaScript

13 Behavioral Pattern

- Observer ด้วย EventTarget ฝั่งเบราว์เซอร์และ EventEmitter ฝั่ง Node.js
- Strategy สำหรับสลับวิธีคำนวณตอนทำงาน
- Command สำหรับงานที่ต้อง Undo ได้
- State สำหรับ Object ที่พฤติกรรมเปลี่ยนตามสถานะ
- Lab: สถานะคำสั่งซื้อด้วย State และ Observer

14 Lab: Test และ Refactor โค้ด OOP

- เขียน Unit Test ให้ Class ด้วย Node.js Test Runner
- ใช้ Dependency Injection และ Mock เพื่อทดสอบแยกส่วน
- Refactor โค้ดแบบฟังก์ชันยาว ๆ ให้เป็น Class ทีละขั้นโดยมี Test คุม
- ใช้ AI Assistant ช่วยเสนอการ Refactor แล้วตรวจผลด้วย Test
- Lab: Refactor ระบบคำนวณส่วนลดจากโค้ดเดิม

15 Workshop: Capstone ออกแบบระบบด้วย OOP และ Pattern

- วิเคราะห์โจทย์ระบบจองหรือระบบคำสั่งซื้อและเขียน Class Diagram
- เลือก Pattern ที่เหมาะกับแต่ละส่วนพร้อมเหตุผล
- เขียนโค้ดตามหลัก SOLID พร้อม Test ส่วนสำคัญ
- นำเสนอผลงานและรีวิวการออกแบบร่วมกัน

07 เรียนจบแล้วได้อะไร Outcomes

- **เข้าใจ OOP ของ JavaScript** Prototype และ Class
- **เขียน Class ได้ถูกหลัก** Encapsulation และ Abstraction
- **ออกแบบลำดับชั้นได้** Inheritance, Polymorphism และ Composition
- **ออกแบบเป็นระบบ** SOLID และ Class Diagram
- **ใช้ Pattern เป็น** Creational, Structural และ Behavioral
- **โค้ดมีคุณภาพ** TypeScript, Test และ Refactoring

08 สิ่งที่ผู้เรียนจะได้รับ Deliverables

- เอกสารประกอบการอบรมฉบับสมบูรณ์ครบทั้ง 3 วัน และ Lab Guide ทีละขั้นตอน
- โค้ดตั้งต้นและโค้ดเฉลยครบทุก Lab ทั้ง JavaScript และ TypeScript
- ตัวอย่าง Design Pattern พร้อม Class Diagram และโปรเจกต์ Capstone สำหรับใช้ต่อ
- ประกาศนียบัตรจบหลักสูตร (Certificate of Completion) และช่องทางสอบถามวิทยากรหลังจบหลักสูตร

09 ข้อมูลผู้สอน Instructor

อ.สามิตร โกยม ผู้สอนและนักพัฒนาที่ออกแบบระบบด้วย JavaScript และ TypeScript ในงานจริง เน้นอธิบายเหตุผลของการออกแบบแต่ละแบบด้วยตัวอย่างจากงานธุรกิจ ให้ผู้เรียนลองเขียน เปรียบเทียบ และ Refactor ด้วยตัวเอง เพื่อให้กลับไปออกแบบโค้ดที่แก้ไขและต่อยอดง่ายให้กับทีมได้

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