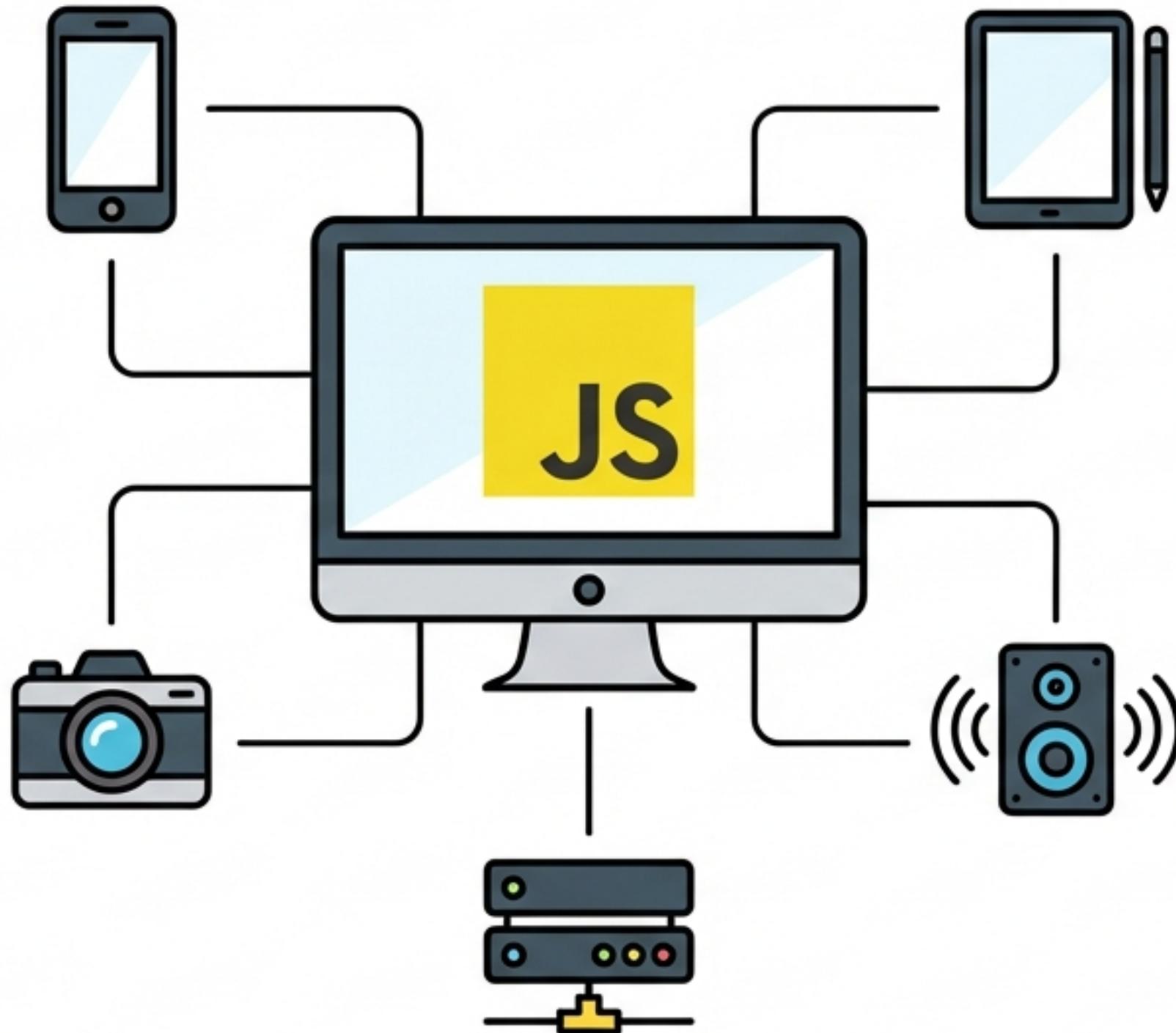


JavaScript: เริ่มต้นการ ผจญภัยเขียนโค้ดของคุณ!

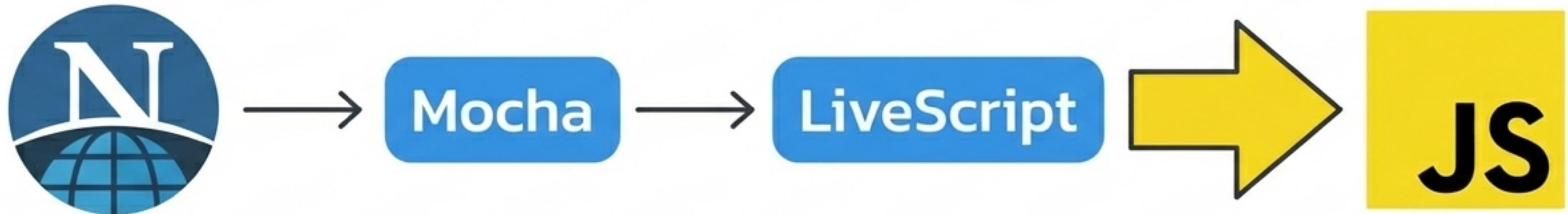
คู่มือฉบับเริ่มต้นสำหรับสร้างเว็บแบบอินเทอร์แอคทีฟ

JavaScript คืออะไร?



- JavaScript คือภาษาโปรแกรมที่ถูกออกแบบมาให้เว็บไซต์มีการโต้ตอบและเปลี่ยนแปลงตามการกระทำของผู้ใช้
- เป็นส่วนสำคัญของเทคโนโลยีหลักของเว็บร่วมกับ HTML (โครงสร้าง) และ CSS (สไตล์) JavaScript คือ 'สมอง' ที่ทำให้เว็บมีชีวิตชีวา
- ในปัจจุบัน JS ไม่ได้อยู่แค่ในเบราว์เซอร์อีกต่อไป ด้วย Node.js ทำให้เราสามารถใช้ JavaScript พัฒนาได้ทั้งฝั่งหน้าบ้าน (Front-End) และหลังบ้าน (Back-End)

จุดกำเนิดของ JavaScript



ผู้สร้าง: พัฒนาโดย Brendan Eich ที่บริษัท Netscape

วิวัฒนาการของชื่อ: เริ่มต้นด้วยชื่อ 'Mocha' จากนั้นเปลี่ยนเป็น 'LiveScript' และสุดท้ายคือ 'JavaScript' เพื่อใช้ความนิยมของภาษา Java ในขณะนั้นมาช่วยในการตลาด (แม้ว่าทั้งสองภาษาจะแตกต่างกันมาก!)

การเป็นมาตรฐาน: ในปี 1997 ได้ถูกกำหนดเป็นมาตรฐานโดยองค์กร ECMA ภายใต้ชื่อ 'ECMAScript' (ES) ซึ่งมีการอัปเดตพีเจอร์ใหม่ๆ อยู่เสมอ (เช่น ES6 ในปี 2015)

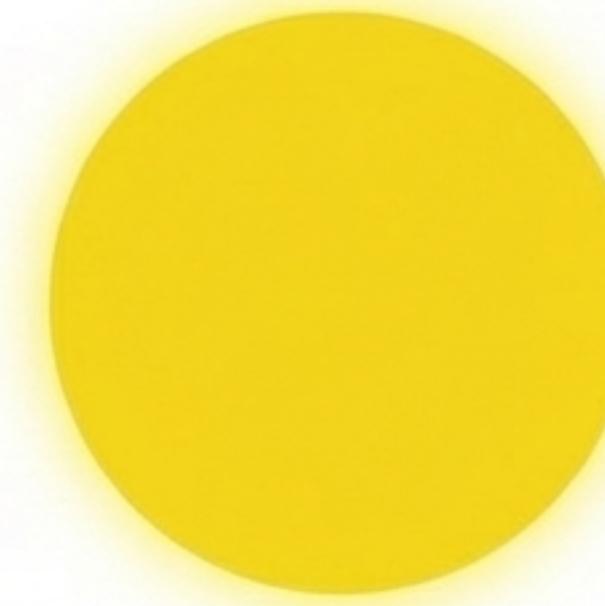
JavaScript vs. ECMAScript: ต่างกันอย่างไร?

ECMAScript



คือ ‘มาตรฐาน’ หรือ ‘พิมพ์เปี้ยว’ ที่กำหนด
ว่าภาษาควรมีฟีเจอร์และทำงานอย่างไร

JavaScript

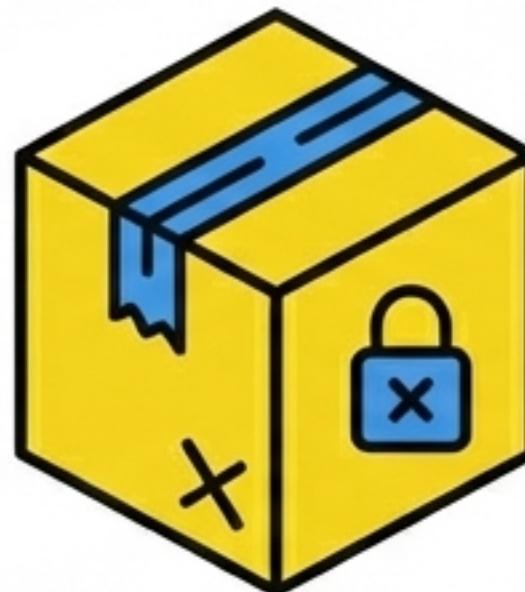
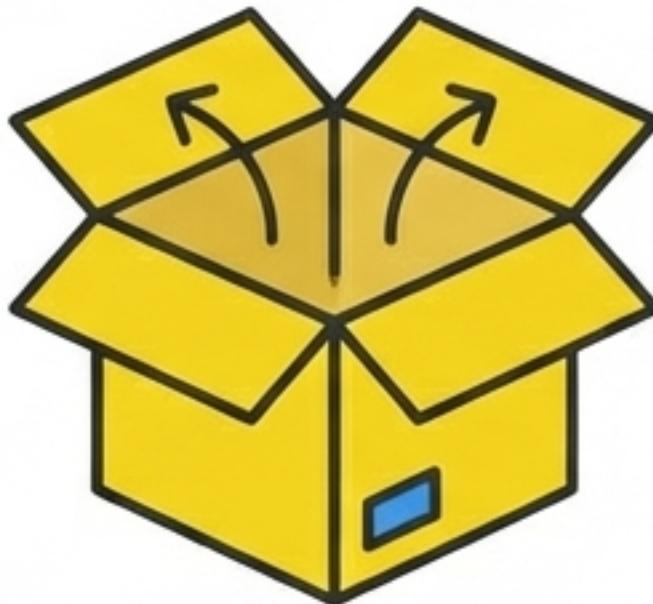


คือ ‘ภาษา’ ที่ถูกพัฒนาขึ้นตาม
มาตรฐานของ ECMAScript

ดังนั้น เวลาที่เราเห็นคำว่า ES6, ES7, ES2016 นั้นคือกำลังพูดถึงเวอร์ชันของ ‘มาตรฐาน’ ECMAScript ที่ JavaScript นำไปใช้งาน

ตัวแปร: กล่องเก็บข้อมูลของเรา

ตัวแปรใช้สำหรับเก็บข้อมูลที่เราสามารถอ้างอิงและเปลี่ยนแปลงได้ในโปรแกรม
เหมือนกล่องที่เราตั้งชื่อแล้วใส่ของเข้าไปข้างในได้ ใน JavaScript เรา มีคำสั่งสำหรับสร้างกล่อง 3 แบบ:



'let': กล่องที่เปลี่ยนของข้างในได้
เหมาะสำหรับข้อมูลที่อาจมีการ
มีการเปลี่ยนแปลงค่า

'const': กล่องที่ 'ปิดตาย'
เมื่อใส่ของไปแล้วจะเปลี่ยนไม่ได้
เหมาะสำหรับค่าคงที่

'var': วิธีการประกาศแบบดั้งเดิม
ปัจจุบันไม่นิยมใช้แล้ว
แนะนำให้ใช้ 'let' และ 'const' แทน

```
// กล่องที่เปลี่ยนของข้างในได้  
let age = 30;
```

```
// กล่องที่ปิดตาย เปลี่ยนค่าไม่ได้  
const isStudent = true;
```

ชนิดของข้อมูล: ของที่ใส่ในกล่องได้

JavaScript สามารถจัดเก็บข้อมูลได้หลายประเภท แบ่งเป็น 2 กลุ่มหลัก:

ข้อมูลพื้นฐาน (Primitive Types)



String: `let name = "John";` - สำหรับเก็บข้อความ



Number: `let num = 42;` - สำหรับเก็บตัวเลข (ทั้งจำนวนเต็มและทศนิยม)



Boolean: `let bool = true;` - สำหรับเก็บค่าจริง (true) หรือเท็จ (false)



Undefined: `let something;` - ตัวแปรที่ประกาศแล้วแต่ยังไม่มีค่า



Null: `let emptyValue = null;` - การระบุว่า 'ไม่มีค่า' อย่างตั้งใจ

ข้อมูลไม่พื้นฐาน (Non-Primitive Types)



Object: `{ name: "John", age: 30 }` - ชุดข้อมูลที่เก็บเป็นคู่ key-value



Array: `[1, 2, 3];` - ชุดข้อมูลที่เก็บเรียงกันเป็นลำดับ

การควบคุมทิศทาง: เมื่อโค้ดต้องตัดสินใจ



เราใช้ 'คำสั่งเงื่อนไข' เพื่อสั่งให้โปรแกรมทำงาน
แตกต่างกันไปตามเงื่อนไขที่กำหนด
คำสั่งที่ใช้บ่อยที่สุดคือ 'if...else'

- 'if' (ถ้าเงื่อนไขเป็นจริง) ให้ทำสิ่งนี้...
- 'else if' (หรือถ้าเงื่อนไขนี้เป็นจริง) ให้ทำสิ่งนั้นแทน...
- 'else' (ถ้าไม่มีเงื่อนไขไหนเป็นจริงเลย) ให้ทำอีกอย่างหนึ่ง...

```
let age = 20;  
  
if (age < 18) {  
    console.log("You are a minor");  
} else {  
    console.log("You are an adult");  
}
```

การทำงานช้า: สั่งให้โค้ดทำอะไรiron



การวนซ้ำ (Loop) ใช้เพื่อสั่งให้โค้ดบล็อกหนึ่งทำงานซ้ำๆ จนกว่าเงื่อนไขที่กำหนดจะไม่เป็นจริงอีกต่อไป วนซ้ำที่พบบ่อยคือ `for` loop

```
for (let i = 0; จุดเริ่มต้น: เริ่มนับที่ 0
```

เงื่อนไข: นำไปเรื่อยๆ ตราบใดที่ i ยังน้อยกว่า 5

การเปลี่ยนแปลง: หลังจากกำเสริจแต่ละรอบ
ให้เพิ่มค่า i ขึ้น 1

```
console.log(i);  
}  
//พิมพ์ 1 บน 1  
สิ่งที่ต้องทำ: พิมพ์ค่า i ออกมาก่อน
```

Output: ผลลัพธ์ที่ได้คือ: `0, 1, 2, 3, 4

ฟังก์ชัน: สูตรสำเร็จที่เรียกใช้ช้าได้

ฟังก์ชันคือบล็อกของโค้ดที่ออกแบบมาเพื่อทำงานอย่างใดอย่างหนึ่งโดยเฉพาะ เราสามารถตั้งชื่อให้มัน และเรียกใช้ช้าที่ไหนก็ได้ในโปรแกรม ช่วยให้โค้ดเป็นระเบียบและจัดการง่าย



Standard Function

```
// สร้าง "สูตร" สำหรับทักทาย
function greet(name) {
  return "Hello, " + name;
}

// เรียกใช้ "สูตร"
let message = greet("Alice");
console.log(message); // "Hello, Alice"
```

ES6 Arrow Function

แบบนำรูปแบบที่ทันสมัยและสั้นกว่า

```
// "สูตร" แบบสั้นด้วย Arrow Function
const multiply = (a, b) => a * b;
console.log(multiply(2, 3)); // 6
```



ໄດ້ເວລາລົງມື່ອ! ເຕຣີຍມເຄຣື່ອງມື່ອໃຫ້ພຣູມ

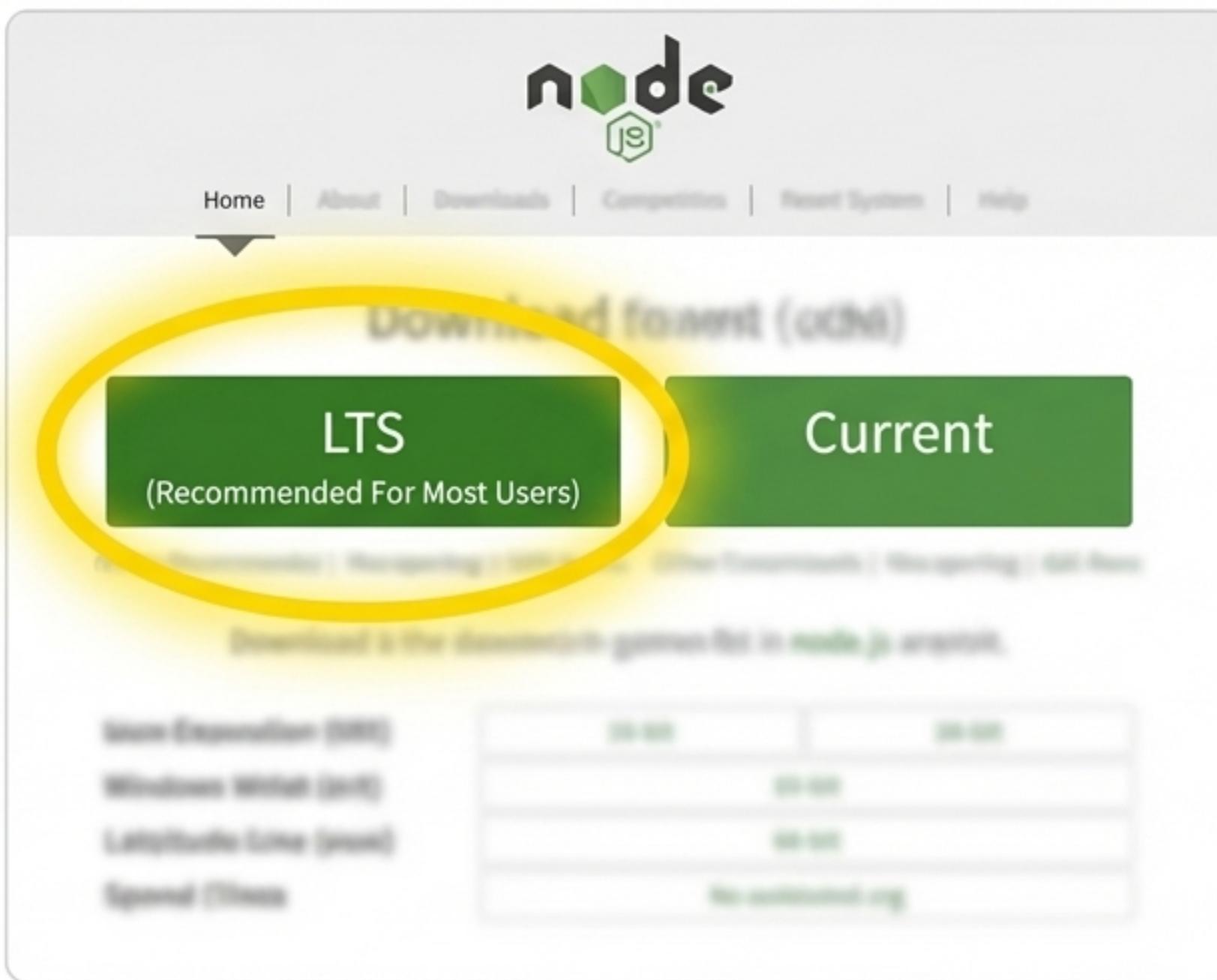


ກ່ອນຈະເຮັ່ມເບີຍນໂຄ້ດ ເຮັດວຽກ ເຕຣີຍມເຄຣື່ອງມື່ອທີ່ຈຳເປັນ 3 ອຍ່າງ ເປົ້າຍບເສມື່ອນກາຮັດເຕຣີຍມຫ້ອງກົດລອງຂອງເຮົາ:

1. **Node.js**: ກຳໃຫ້ເຮົາສາມາຮຽນໂຄ້ດ JavaScript ນອກເວັບເບຣາວໆເຊວຣໄດ້ (ເຊັ່ນ ໃນເຄຣື່ອງຂອງເຮົາໄດຍຕຮງ)
2. **Visual Studio Code (VS Code)**: Text Editor ທີ່ຮູ້ອໂປຣແກຣມສໍາຮັບເບີຍນໂຄ້ດທີ່ໄດ້ຮັບຄວາມນິຍມສູງ
3. **Git**: ເຄຣື່ອງມື່ອສໍາຮັບຈັດກາຮັດວຽກ (Version Control)

ໄປຖີ່ລະບົ້ນຕອນກັນແລຍ!

ขั้นตอนที่ 1: ติดตั้ง Node.js



1. ไปที่เว็บไซต์ <https://nodejs.org/>
2. คลิกดาวน์โหลดเวอร์ชัน **LTS** (Long Term Support) ซึ่งเป็นเวอร์ชันที่เสถียรและแนะนำสำหรับผู้ใช้ส่วนใหญ่
3. ติดตั้งโปรแกรมตามขั้นตอนที่ปรากฏบนหน้าจอ

Pro-tip: 'เมื่อติดตั้งเสร็จแล้ว ลองเปิด Terminal (หรือ Command Prompt) และพิมพ์ `node --version` เพื่อตรวจสอบว่าการติดตั้งสำเร็จหรือไม่'

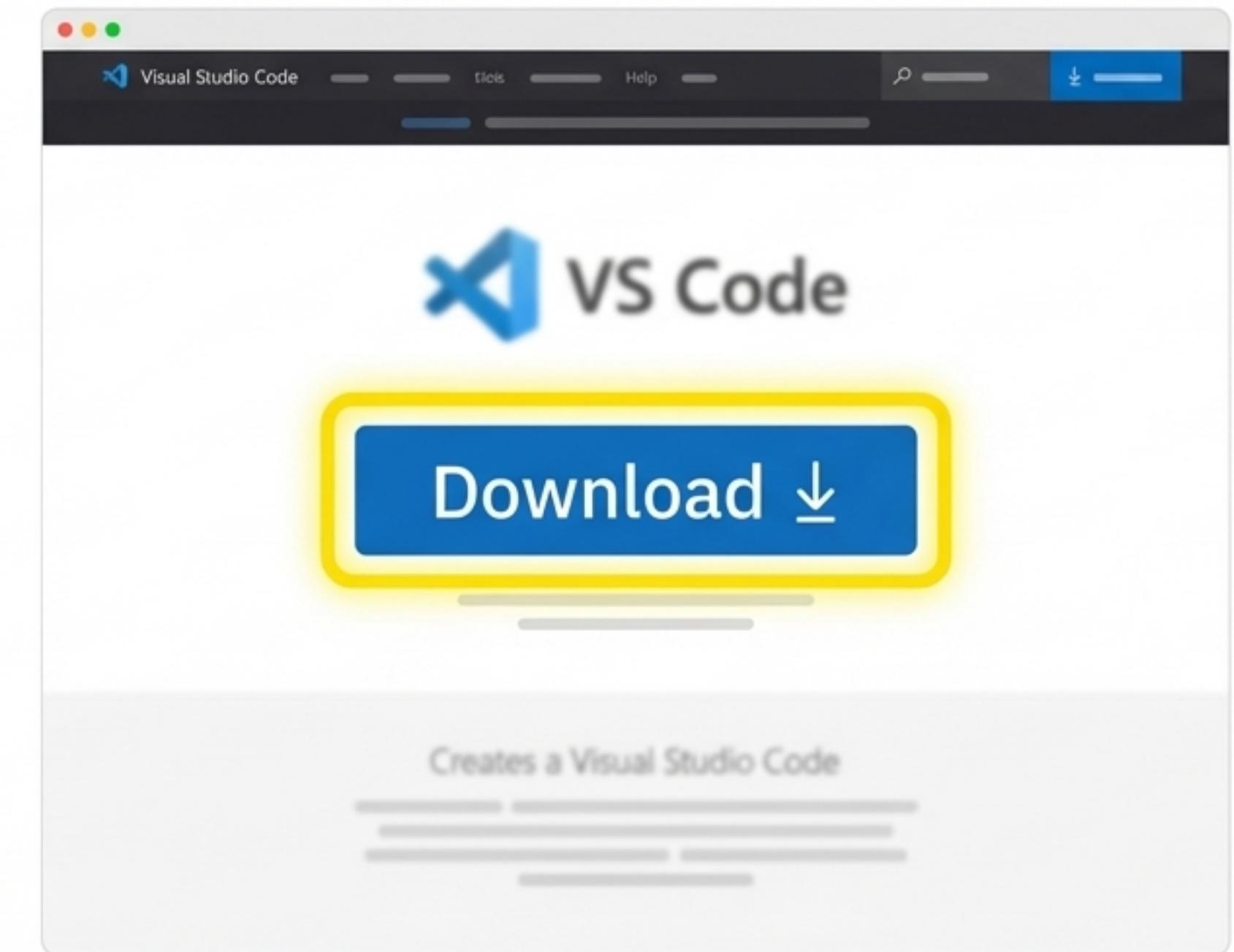
ขั้นตอนที่ 2: ติดตั้ง Visual Studio Code

- 1. ไปที่เว็บไซต์

<https://code.visualstudio.com/>

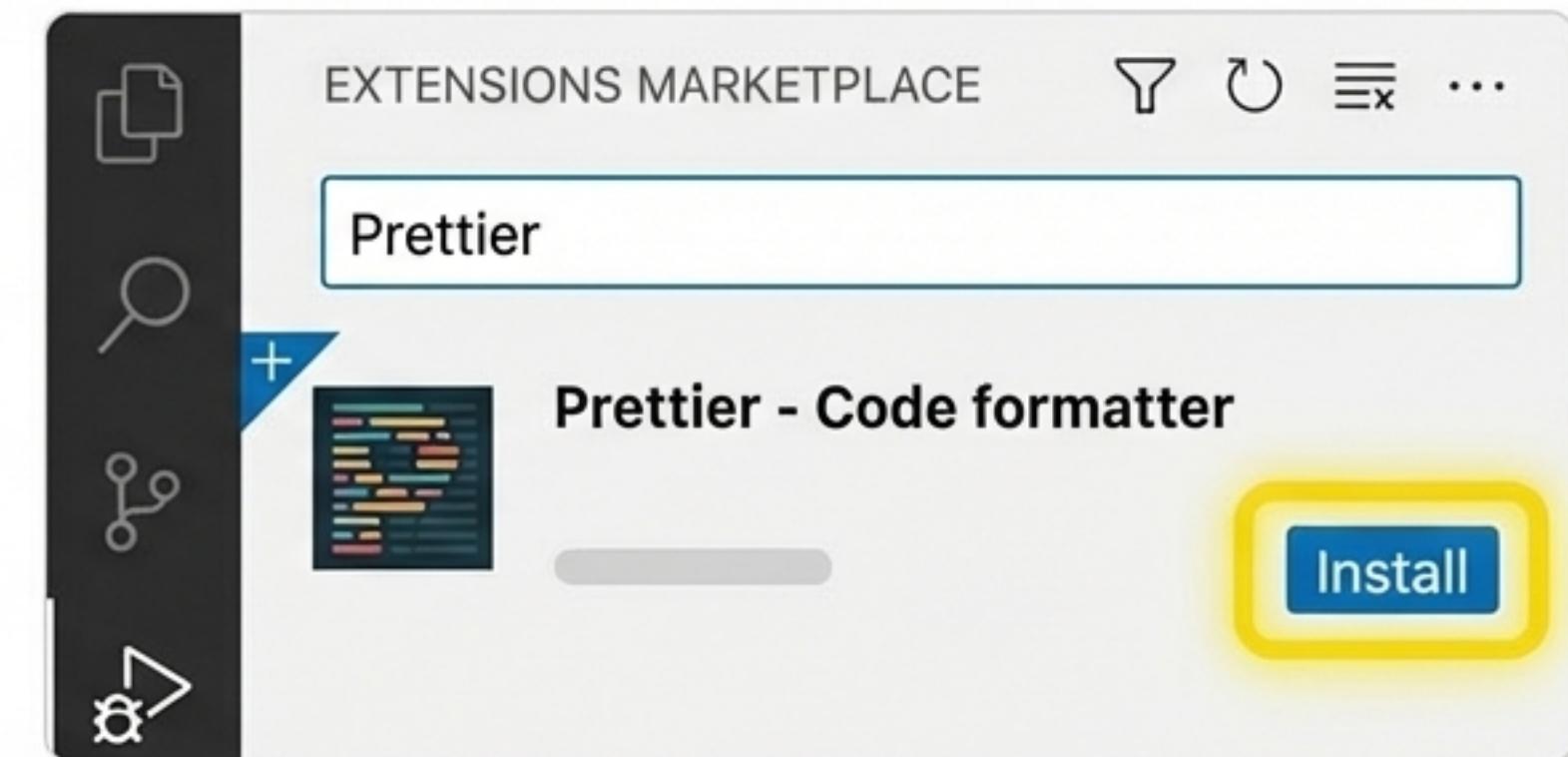
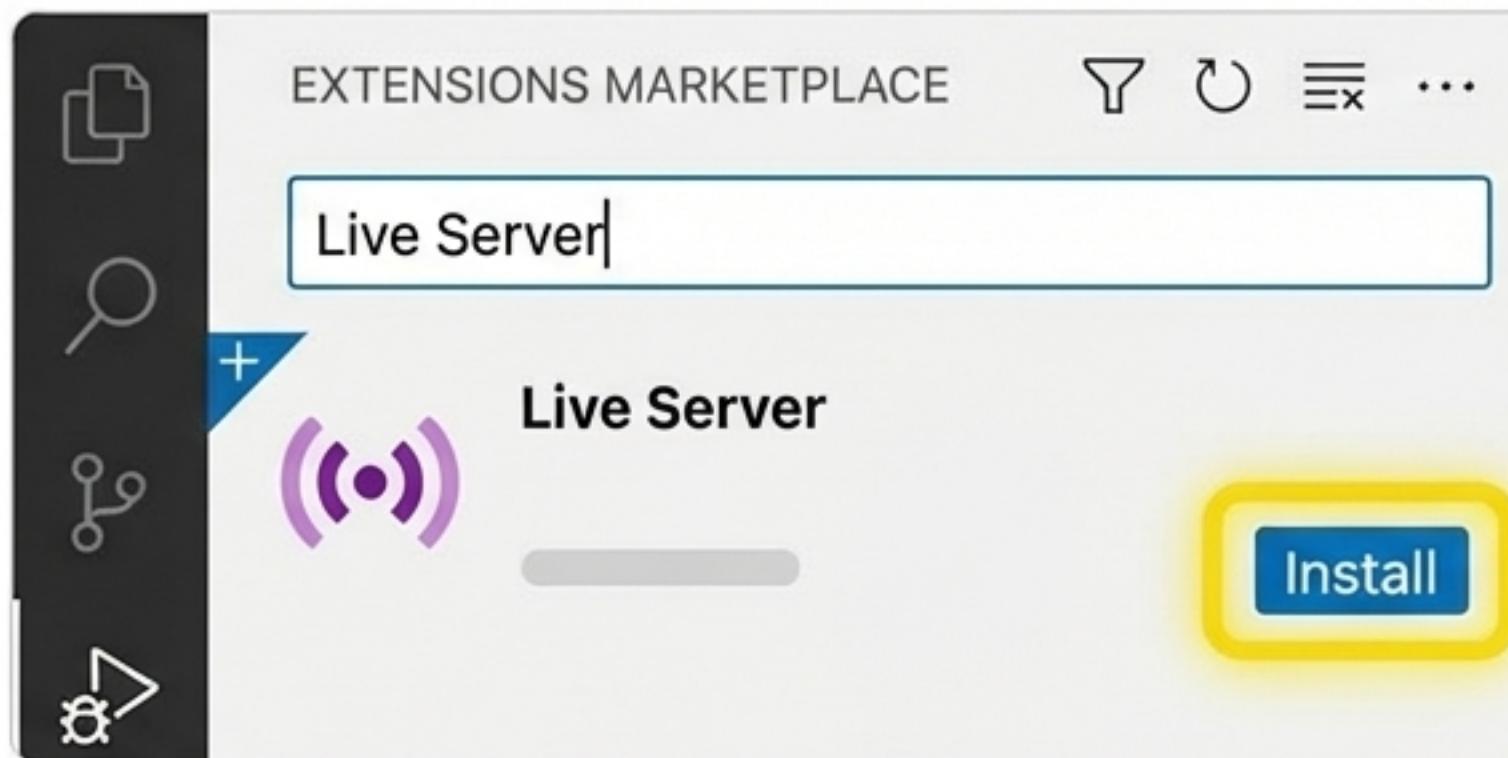
- 2. ดาวน์โหลดและติดตั้งโปรแกรมสำหรับระบบปฏิบัติการของคุณ (Mac, Windows, or Linux)

'VS Code เป็นเหมือนบ้านของโค้ดเดอร์ ที่เราจะใช้เวลาส่วนใหญ่ในการเขียนและแก้ไขโปรแกรม'



ขั้นตอนที่ 3: ติดตั้งส่วนเสริม (Extensions) ใน VS Code

เพิ่มพลังให้ VS Code ของเราด้วยส่วนเสริมยอดนิยม:



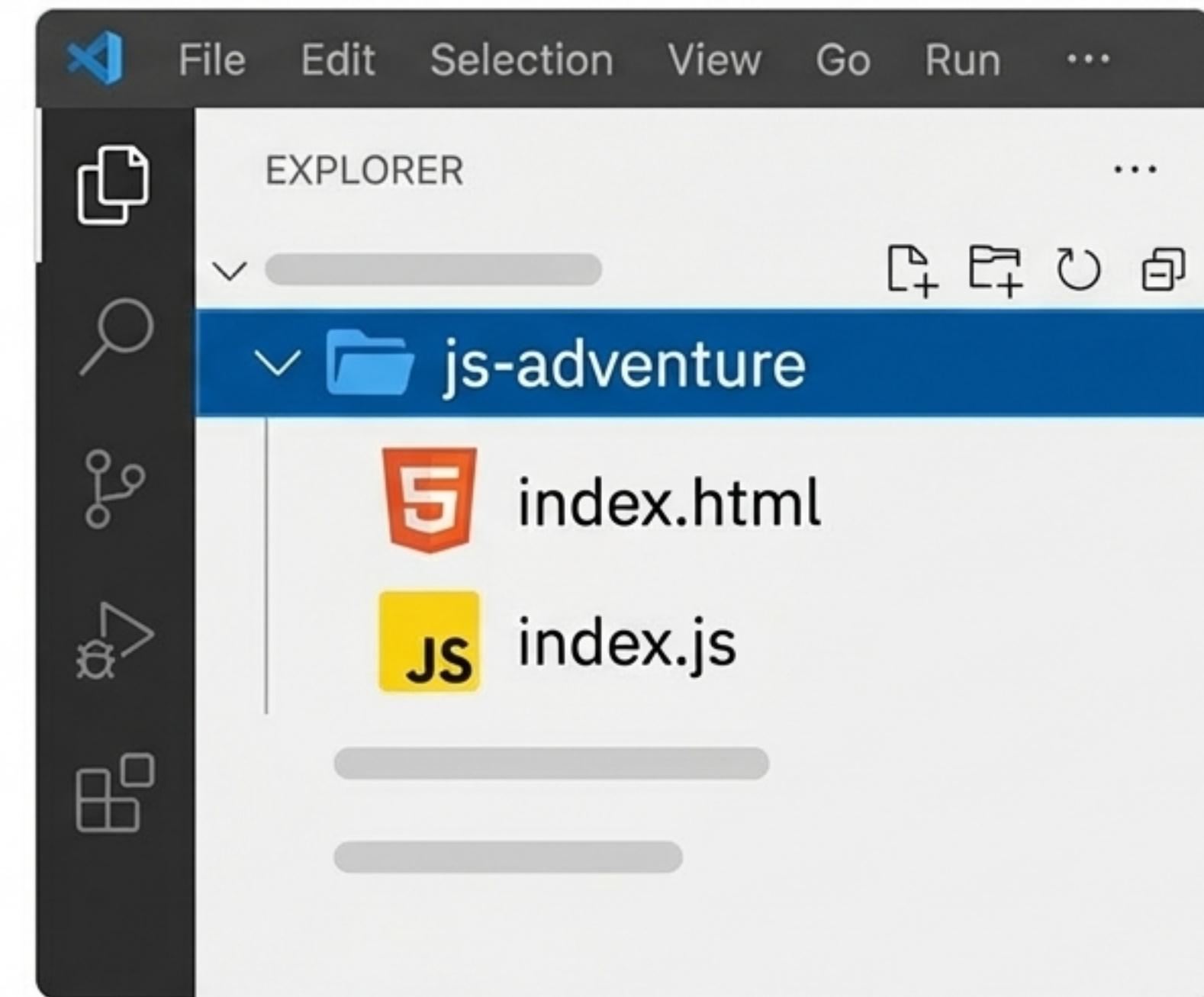
Live Server: ช่วยให้เราเห็นผลลัพธ์ของหน้าเว็บที่เรากำลังแก้ไขได้กันทีโดยไม่ต้องกดรีเฟรชเอง

Prettier - Code formatter: ช่วยจัดรูปแบบโค้ดให้อ่านง่ายและสวยงามโดยอัตโนมัติ

How-to: ‘เปิด VS Code, ไปที่แท็บ Extensions (ไอคอนรูปสี่เหลี่ยม), ค้นหาชื่อส่วนเสริมแล้วกด Install’

สร้างโปรเจกต์แรกของเรา: Hello World

1. สร้างโฟลเดอร์ใหม่บนคอมพิวเตอร์ของคุณ ตั้งชื่อว่า `js-adventure`
2. เปิดโปรแกรม VS Code แล้วเลือก `File > Open Folder...` และเลือก โฟลเดอร์ `js-adventure` ที่เพิ่งสร้าง
3. ใน VS Code Explorer, คลิกที่ไอคอน **New File** เพื่อสร้างไฟล์ 2 ไฟล์: `index.html` (สำหรับโครงสร้างหน้าเว็บ), `index.js` (สำหรับเขียนโค้ด JavaScript ของเรา)



เขียนโค้ดเบร็ฟๆ และรัน!

Step 1: เขียนโค้ด

ในไฟล์ `index.js` , พิมพ์โค้ดต่อไปนี้:

```
// นี่คือคอมเม้นต์ โค้ดเบร็ฟๆของฉัน!  
console.log("Hello World");
```

Step 2: รันโค้ด

เปิด Terminal ใน VS Code (ไปที่เมนู `Terminal > New Terminal`)

ในหน้าต่าง Terminal, พิมพ์คำสั่ง: `node index.js` และกด Enter

The screenshot shows the Visual Studio Code interface. In the top left, there's a code editor with two tabs: 'index.js' and 'index.js > ...'. The content of 'index.js' is:

```
// นี่คือคอมเม้นต์ โค้ดเบร็ฟๆของฉัน!  
console.log("Hello World");
```

Below the code editor is a 'TERMINAL' tab. Inside the terminal window, the command `node index.js` is entered, followed by the output `Hello World`.

★ Output : คุณจะเห็นข้อความ `Hello World` ปรากฏขึ้นใน Terminal!

นำ JavaScript ไปแสดงผลในเบราว์เซอร์

ตอนนี้เราจะเชื่อมไฟล์ `index.js` ของเราเข้ากับ `index.html` เพื่อให้โค้ดทำงานบนหน้าเว็บ

Step 1: เขียนโค้ด HTML

ในไฟล์ `index.html`, พิมพ์โค้ดพื้นฐานนี้:

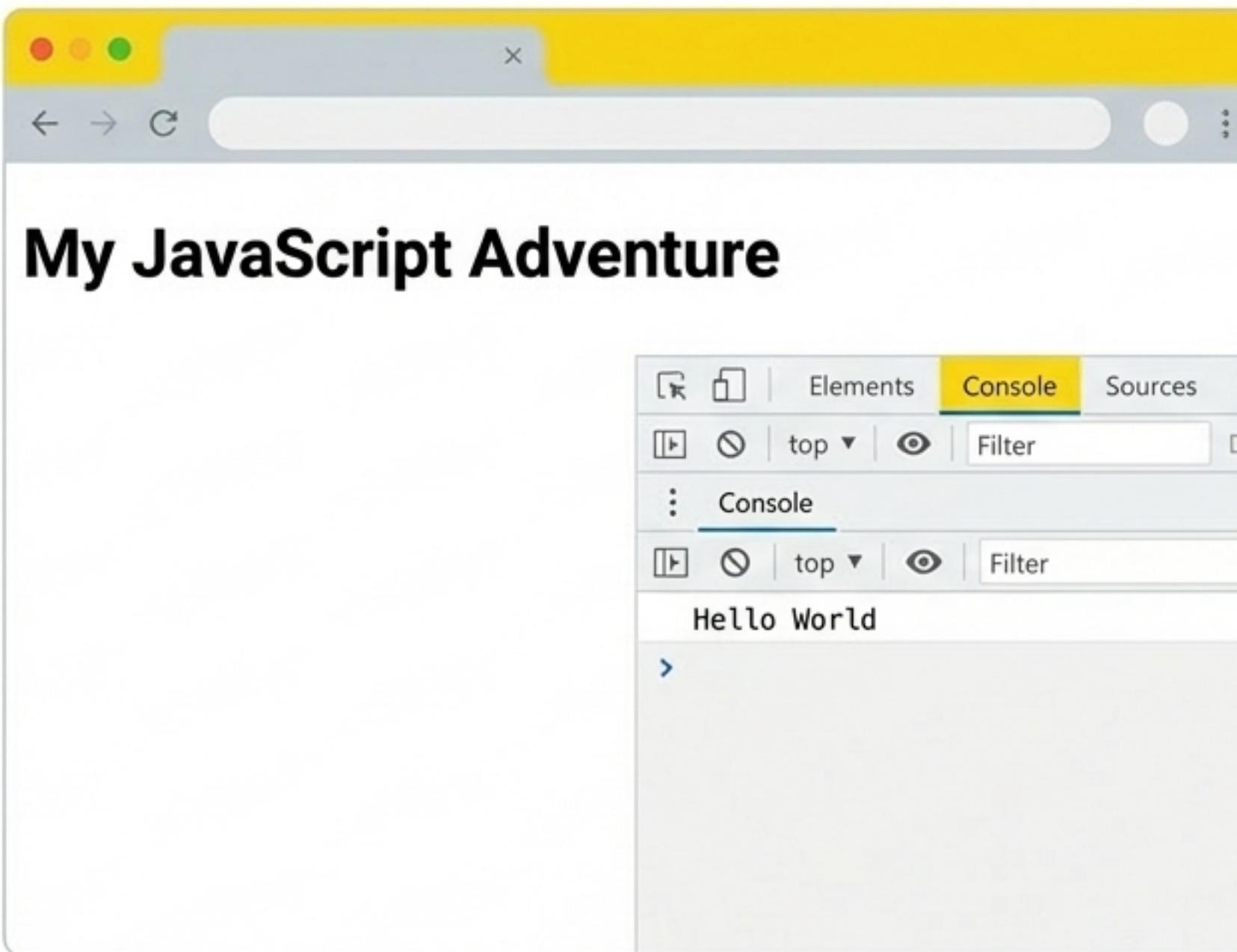
```
<!DOCTYPE html>
<html>
  <head>
    <title>My First JS Page</title>
  </head>
  <body>
    <h1>My JavaScript Adventure</h1>
    <script src="index.js"></script>
  </body>
</html>
```

Key Point

บรรทัดนี้คือหัวใจสำคัญ! เป็นการบอกให้ไฟล์ HTML โหลดและรันโค้ดจากไฟล์ `index.js` ของเรา



เปิดดูผลลัพธ์ใน Browser Console



Instructions:

1. ใน VS Code, คลิกขวาที่ไฟล์ `index.html` แล้วเลือก ‘Open with Live Server’
2. หน้าเว็บจะเปิดขึ้นในเบราว์เซอร์ของคุณ
3. เปิด Developer Tools โดยการคลิกขวาบนหน้าเว็บแล้วเลือก ‘Inspect’ จากนั้นไปที่แท็บ `Console`

Result:

คุณจะเห็นข้อความจาก `console.log` ข่องเราปรากฏอยู่ใน Console!
นี่คือเครื่องมือสำคัญสำหรับ debug โค้ด

พลังของ JavaScript: เปลี่ยนแปลงหน้าเว็บได้ทันที!

JavaScript สามารถตั้งต่อและเปลี่ยนแปลงเนื้อหา, โครงสร้าง, และสไตล์ของหน้าเว็บได้ผ่านสิ่งที่เรียกว่า Document Object Model (DOM)

Before



Turn on the light



Turn off the light

After



Turn on the light

Turn off the light

```
<!-- รูปภาพหลอดไฟ -->


  Turn on the light

```

‘**onclick**’ คือ event ที่เกิดขึ้นเมื่อเราคลิกปุ่ม โค้ด JavaScript ที่อยู่ข้างในจะไปค้นหา element ที่มี **‘id="myImage”** และเปลี่ยน **‘src’** (source) ของรูปภาพ!

การผจญภัยของคุณเพื่อเริ่มต้น

javascript.info
The Modern JavaScript Tutorial



**MDN
Web Docs**

{~} **exercism.org**

เส้นทางการเรียนรู้ JavaScript ยังมีอะไรอีกมากmany! นี่คือแหล่งข้อมูลชั้นเยี่ยมสำหรับก้าว
ก้าวต่อไปของคุณ:

- **The Modern JavaScript Tutorial (javascript.info)**: แหล่งข้อมูลเชิงลึกและ
ทันสมัย เหมาะสำหรับการเรียนรู้ตั้งแต่พื้นฐานจนถึงขั้นสูง
- **MDN Web Docs (Mozilla Developer Network)**: คู่มืออ้างอิงที่ดีที่สุดสำหรับ
เทคโนโลยีเว็บทั้งหมด รวมถึง JavaScript
- **Exercism (exercism.org)**: แพลตฟอร์มสำหรับฝึกฝนการเขียนโค้ดผ่านแบบฝึกหัด
พร้อมรับ feedback จาก mentor

ขอให้สนุกกับการเขียนโค้ด!