STUDENT PRICE BOOK CENTER

Svelte Web Programming

Beginner Integrative-Generative AI Edition

svelte

Table of Contents

Author Name

คำนำ

หนังสือ *Svelte Web Programming: Beginner* เล่มนี้ถูกจัดทำขึ้นเพื่อเป็นคู่มือฉบับสมบูรณ์สำหรับผู้เริ่มต้นที่ต้องการเรียนรู้การพัฒนาเว็บแอปพลิเคชันด้วย Svelte ซึ่งเป็นเฟรมเวิร์กสมัยใหม่ที่กำลังได้รับความนิยมอย่างรวดเร็วในกลุ่มนักพัฒนาเว็บทั่วโลก ด้วยแนวคิดที่แตกต่างจากเฟรมเวิร์กทั่วไป Svelte ทำให้การสร้างเว็บแอปพลิเคชันเป็นเรื่องที่เรียบง่าย เร็ว และมีประสิทธิภาพสูง โดยไม่ต้องพึ่ง Virtual DOM และยังคอมไพล์โค้ดออกมาเป็น JavaScript ล้วนๆ ในขั้นตอน build ทำให้ได้ผลลัพธ์ที่เบาและทำงานได้เร็วกว่าเฟรมเวิร์กอื่นหลายเท่าตัว

หนังสือเล่มนี้เริ่มต้นจากพื้นฐานจริงๆ โดยใน **บทที่ 1: แนะนำ Svelte** ผู้อ่านจะได้เรียนรู้ที่มาของเฟรมเวิร์กนี้ รวมถึงแนวคิดเบื้องหลัง สถิติผู้ใช้งาน และองค์กรที่นำ Svelte ไปใช้จริง ตลอดจนคำแนะนำด้านเครื่องมือ เช่น Visual Studio Code และการติดตั้งโปรเจกต์ Svelte ด้วย degit + Vite ที่เป็นมาตรฐานของปี 2024–2025 บทนี้ยังครอบคลุมโครงสร้างพื้นฐานของโปรเจกต์และความเข้าใจเบื้องต้นสำหรับผู้ที่ไม่เคยใช้ Svelte มาก่อน

เมื่อผู้อ่านมีพื้นฐานแล้ว **บทที่ 2: สร้าง Component แรก** จะพาไปเริ่มสร้างไฟล์ .svelte อย่างเต็มรูปแบบ พร้อมอธิบายโครงสร้างของ <script>, <style> และ markup ภายใน Component ซึ่งช่วยให้ผู้อ่านเข้าใจการรวม logic, UI และ style ไว้ในไฟล์เดียวได้อย่างเป็นระบบ นอกจากนี้ยังสอนการรับค่าผ่าน export let (props) และการจัดการ event ด้วยคำสั่งที่เป็นธรรมชาติของ Svelte เช่น on:click={handleClick} พร้อมด้วยตัวอย่างประกอบที่สามารถนำไปต่อยอดได้ทันที

ถัดมา **บทที่ 3: การแสดงผลและโครงสร้างเงื่อนไข** จะเน้นการแสดงข้อมูลบน UI อย่างยืดหยุ่น ไม่ว่าจะเป็นการแสดงค่าด้วย {} (Interpolation) การใช้ if/else if/else เพื่อแสดงเนื้อหาแบบมีเงื่อนไข หรือการวนลูปข้อมูลด้วย each block รวมถึง await block สำหรับจัดการข้อมูลแบบ asynchronous ทุกหัวข้อมีการอธิบายอย่างละเอียด พร้อมตัวอย่างโค้ดและการประยุกต์ใช้งานจริงเพื่อให้เข้าใจง่ายยิ่งขึ้น

ใน **บทที่ 4: การทำงานกับฟอร์มและการ binding** ผู้อ่านจะได้เรียนรู้เทคนิคการเชื่อมโยงข้อมูลระหว่างฟอร์ม (เช่น <input> และ <select>) กับตัวแปรใน script ด้วย bind:value และแนวคิด two-way data binding ซึ่งช่วยลดโค้ดที่ซับซ้อนในการจัดการค่าภายในฟอร์ม พร้อมทั้งแนะนำวิธีการจัดการการส่งข้อมูล และการตรวจสอบความถูกต้องของฟอร์มแบบเบื้องต้น เพื่อเตรียมความพร้อมสำหรับโปรเจกต์จริง

สุดท้ายใน **บทที่ 5: การจัดการ Style** จะเน้นการเขียน CSS ภายใน Component อย่างเป็นระบบ ผ่าน <style> tag ที่รองรับ **Scoped CSS** โดยอัตโนมัติ ทำให้ไม่ต้องกังวลว่าสไตล์จะชนกับ Component อื่น และยังแนะนำการใช้ class แบบมีเงื่อนไข เช่น class:active={isActive} เพื่อควบคุมการเปลี่ยนรูปลักษณ์ตามสถานะ ซึ่งเป็นอีกหนึ่งจุดเด่นของ Svelte ที่ทำให้การพัฒนา UI มีความเป็นธรรมชาติและไม่ซับซ้อน

หนังสือเล่มนี้เขียนขึ้นโดยเน้นภาษาที่เข้าใจง่าย มีตัวอย่างโค้ดให้ทดลองจริง และออกแบบมาให้เหมาะทั้งสำหรับผู้ที่ไม่มีพื้นฐาน Svelte มาก่อน และผู้ที่มีพื้นฐาน JavaScript แต่ต้องการย้ายจากเฟรมเวิร์กอื่นมาสู่โลกของ Svelte ผู้อ่านจะสามารถเรียนรู้แบบก้าวต่อก้าว ตั้งแต่เริ่มติดตั้ง ไปจนถึงการพัฒนา Component ที่ซับซ้อนมากขึ้น

ผู้เขียนหวังว่า *Svelte Web Programming: Beginner* จะเป็นคู่มือที่ทรงคุณค่าสำหรับนักพัฒนาในยุคใหม่ และช่วยให้ทุกท่านสามารถนำ Svelte ไปใช้พัฒนาเว็บแอปพลิเคชันได้อย่างมั่นใจ ทั้งในโปรเจกต์ส่วนตัว การเรียนการสอน หรือแม้แต่การพัฒนาระบบจริงในองค์กร ขอให้การเรียนรู้ในทุกบทของหนังสือนี้เต็มไปด้วยความสนุก และเปิดโลกใหม่ของการเขียนเว็บให้กับทุกท่านอย่างแท้จริง.

ด้วยรักและปรารถนาดี

ศูนย์หนังสือราคานักเรียน

สารบัญ

หน้า

บทที่ 1

แนะนำ Svelte

(Introduction to Svelte)

เนื้อหา

- แนะนำ Svelte (Introduction to Svelte)
- แนะนำ Svelte (ฉบับเชิงลึก)
- ประวัติและวิวัฒนาการของ Svelte (Svelte History & Evolution)
- แนวโน้มการใช้งาน Svelte + สถิติ + บริษัทที่ใช้งานจริง
- IDE/Editor ที่แนะนำสำหรับ Svelte (ปี 2024–2025)
- การติดตั้ง Svelte กับ VS Code อย่างละเอียดที่สุด (ตั้งแต่เริ่มต้น)
- คู่มือติดตั้ง Svelte + VS Code บน macOS / Linux
- การติดตั้งและตั้งค่าเริ่มต้น Svelte ด้วย degit และ Vite
- โครงสร้างโปรเจกต์ของ Svelte (พื้นฐาน)

บทนำ

ในยุคที่การพัฒนาเว็บแอปพลิเคชันมีความซับซ้อนมากขึ้นอย่างต่อเนื่อง เฟรมเวิร์กและไลบรารีสำหรับสร้างส่วนติดต่อผู้ใช้ (User Interface – UI) ได้กลายเป็นเครื่องมือสำคัญที่นักพัฒนาจำเป็นต้องเลือกใช้อย่างเหมาะสม เพื่อให้สามารถสร้างสรรค์งานที่มีประสิทธิภาพ ตอบสนองได้รวดเร็ว และง่ายต่อการดูแลรักษา ท่ามกลางเฟรมเวิร์กที่ได้รับความนิยมอย่างกว้างขวาง เช่น React, Vue และ Angular ได้มีการถือกำเนิดของ **Svelte** ซึ่งนำเสนอแนวคิดที่แตกต่างออกไป โดยมุ่งเน้นการคอมไพล์โค้ดล่วงหน้าเพื่อลดภาระการประมวลผลในฝั่งเบราว์เซอร์

Svelte ได้รับการออกแบบมาเพื่อลดความซับซ้อนของการพัฒนาเว็บแอปพลิเคชัน ด้วยแนวคิดที่ไม่พึ่งพา Virtual DOM แต่ใช้วิธีการคอมไพล์คอมโพเนนต์ให้เป็นโค้ด JavaScript ที่มีประสิทธิภาพสูงตั้งแต่ขั้นตอนการ Build แนวทางนี้ไม่เพียงแต่ช่วยให้แอปพลิเคชันมีประสิทธิภาพดีขึ้นเท่านั้น แต่ยังทำให้ขนาดไฟล์ที่ได้มีความกระชับ ลดการใช้ทรัพยากรระบบ และเพิ่มความเร็วในการตอบสนองของส่วนติดต่อผู้ใช้ได้อย่างมีนัยสำคัญ อีกทั้งยังช่วยให้การเรียนรู้และการเริ่มต้นพัฒนาเป็นไปอย่างราบรื่น เนื่องจากรูปแบบไวยากรณ์ใกล้เคียงกับ HTML, CSS และ JavaScript มาตรฐาน

เมื่อเปรียบเทียบกับเฟรมเวิร์กอื่น ๆ Svelte มีความโดดเด่นด้านโครงสร้างที่เรียบง่าย และปราศจากการใช้ Runtime ขนาดใหญ่เหมือนใน React หรือ Angular การอัปเดตข้อมูลและการเชื่อมโยง

สถานะของคอมโพเนนต์ทำได้อย่างเป็นธรรมชาติ โดยไม่จำเป็นต้องพึ่งพาไลบรารีภายนอกเพิ่มเติมในหลายกรณี ซึ่งเหมาะสมอย่างยิ่งทั้งสำหรับผู้ที่เพิ่งเริ่มต้นเรียนรู้การพัฒนาเว็บ ไปจนถึงนักพัฒนาที่ต้องการเครื่องมือที่สามารถสร้างโปรเจกต์ขนาดใหญ่ได้อย่างมีประสิทธิภาพ

หนังสือเล่มนี้ถูกจัดทำขึ้นเพื่อเป็นคู่มือเชิงลึกสำหรับการเรียนรู้ Svelte โดยเฉพาะ โดยเนื้อหาเริ่มต้นจากการอธิบายพื้นฐานสำคัญ เช่น แนวคิดหลักของ Svelte ความแตกต่างเมื่อเทียบกับเฟรมเวิร์กที่ได้รับความนิยมอื่น ๆ ตลอดจนการติดตั้งและการตั้งค่าเบื้องต้น ทั้งด้วย **degit** และ **Vite** เพื่อให้ผู้อ่านสามารถเลือกใช้เครื่องมือที่เหมาะสมกับรูปแบบการทำงานของตนเองได้อย่างมีประสิทธิภาพ

นอกจากเนื้อหาเกี่ยวกับการติดตั้งแล้ว หนังสือยังให้ความสำคัญกับการทำความเข้าใจโครงสร้างโปรเจกต์ของ Svelte อย่างละเอียด เพื่อให้ผู้อ่านสามารถจัดการไฟล์และคอมโพเนนต์ได้อย่างถูกต้องและเป็นระบบ ซึ่งถือเป็นปัจจัยสำคัญที่จะช่วยให้การพัฒนามีความคล่องตัวและลดปัญหาที่อาจเกิดขึ้นในระยะยาว การวางรากฐานความเข้าใจในส่วนนี้จึงเป็นขั้นตอนสำคัญก่อนที่จะก้าวเข้าสู่เนื้อหาเชิงลึกเกี่ยวกับการสร้างคอมโพเนนต์ การจัดการสถานะ (State Management) และเทคนิคการเพิ่มประสิทธิภาพแอปพลิเคชัน

เพื่อให้เหมาะสมกับผู้อ่านในหลากหลายระดับ หนังสือเล่มนี้มีการอธิบายเนื้อหาอย่างเป็นลำดับขั้น เริ่มตั้งแต่ผู้ที่ไม่มีประสบการณ์มาก่อน ไปจนถึงนักพัฒนาที่มีพื้นฐานและต้องการขยายความเข้าใจเชิงลึก โดยใช้ตัวอย่างโค้ดที่ชัดเจน พร้อมคำอธิบายเชิงวิเคราะห์ เพื่อให้สามารถนำไปประยุกต์ใช้ได้จริงในงานพัฒนาเว็บในสถานการณ์ต่าง ๆ

ท้ายที่สุด หนังสือเล่มนี้ไม่ได้มุ่งเน้นเพียงการสอนให้ใช้งาน Svelte ได้เท่านั้น แต่ยังให้ความสำคัญกับการทำความเข้าใจแนวคิดเชิงสถาปัตยกรรมของเฟรมเวิร์กที่อาจเปลี่ยนแปลงทิศทางของการพัฒนาเว็บแอปพลิเคชันในอนาคต ความเข้าใจในหลักการทำงานของ Svelte จึงไม่เพียงแต่ช่วยให้ผู้อ่านสามารถพัฒนาแอปพลิเคชันได้อย่างมีประสิทธิภาพ แต่ยังเป็นการเปิดมุมมองใหม่ต่อการออกแบบและสร้างสรรค์ซอฟต์แวร์ในยุคที่ความเร็วและประสิทธิภาพเป็นปัจจัยสำคัญที่สุด

แนะนำ Svelte (Introduction to Svelte)

☐ **หัวข้อ 1: Svelte คืออะไร? เปรียบเทียบกับ React / Vue / Angular**

☐ **Svelte คืออะไร?**

Svelte เป็น **JavaScript Framework สมัยใหม่** ที่ออกแบบมาเพื่อสร้าง **Web Application ที่เบาและเร็ว** โดยมีแนวคิดที่ต่างจาก React/Vue/Angular ตรงที่:

- Svelte ทำงานในขั้น **Compile Time**
- ไม่มี Virtual DOM
- สร้างโค้ด JavaScript ที่เรียกใช้ DOM โดยตรง
- ไม่มี Runtime Framework เหลือใน browser

นิยามโดยผู้สร้าง:

“Svelte is a radical new approach to building user interfaces.”

☐ **เปรียบเทียบกับ React / Vue / Angular:**

ด้านเปรียบเทียบ	Svelte	React	Vue	Angular
ลักษณะหลัก	Compiler-based	Virtual DOM	Virtual DOM	Full-featured Framework
ขนาด bundle	เล็กที่สุด (~2KB gzip)	ปานกลาง (~40KB gzip)	ปานกลาง (~30KB gzip)	ใหญ่ (~100KB+)
Performance	เร็วมาก (DOM ตรง)	ดี (ผ่าน Virtual DOM)	ดี (ผ่าน Virtual DOM)	ดี (ขึ้นกับการ config)
Learning Curve	ต่ำ	ปานกลาง	ต่ำถึงปานกลาง	สูง
Syntax	HTML-like (Single File)	JSX (JavaScript-based)	Template + Script	TypeScript + Decorators
SSR / Static Site	รองรับผ่าน SvelteKit	รองรับผ่าน Next.js	รองรับผ่าน Nuxt.js	มี Angular Universal

---

☐ **หัวข้อ 2: จุดเด่นของ Svelte: Compiler-based Framework**

☐ **Svelte มีจุดเด่นที่สำคัญคือ:**

1. **ไม่มี Virtual DOM**

   ⟶ ทำให้ DOM อัปเดตเร็วขึ้นและโค้ดเบากว่า

2. **Compile เป็น JavaScript บริสุทธิ์**

   ⟶ ไม่มี runtime library ใหญ่ ๆ เหลือใน browser

   ⟶ ใช้ทรัพยากรน้อยลง

3. **ใช้งานง่ายและเขียนโค้ดน้อย**

   ⟶ ใช้ HTML/CSS/JS แบบตรง ๆ

   ⟶ ไม่ต้อง import useState, useEffect ฯลฯ

4. **Reactive โดยธรรมชาติ**

   ⟶ แค่เปลี่ยนค่าตัวแปร, UI ก็อัปเดตทันที

   ⟶ ใช้ $: เพื่อบอกว่าเป็น statement ที่ต้อง re-compute

5. **Built-in Transition และ Animation**

6. **Bundle Size เล็กมาก**

   ⟶ โหลดเร็วขึ้น เหมาะกับ mobile-first app

☐ **หัวข้อ 3: การติดตั้งและตั้งค่าเริ่มต้น (Svelte Project Setup)**

☐ **วิธีที่ 1: ใช้ degit (แนะนำสำหรับผู้เริ่มต้น)**

```
npx degit sveltejs/template my-svelte-app
cd my-svelte-app
npm install
npm run dev
```

☐ degit จะ copy เทมเพลต starter มาให้

☐ มีไฟล์พื้นฐาน: App.svelte, main.js, public/index.html

☐ **วิธีที่ 2: ใช้ Vite (แนะนำสำหรับการเริ่ม SvelteKit หรือใช้ในองค์กร)**

```
npm create vite@latest my-svelte-app -- --template svelte
cd my-svelte-app
npm install
npm run dev
```

☐ Vite เป็น bundler ที่เร็วมาก รองรับ hot reload และ TypeScript

---

☐ **หัวข้อ 4: โครงสร้างโปรเจกต์ของ Svelte**

โครงสร้างไฟล์พื้นฐาน (เมื่อใช้ degit):

```
my-svelte-app/
├── public/
│   └── index.html       # HTML entry point
├── src/
│   ├── App.svelte       # Component หลัก
│   └── main.js          # Entry point JS ที่ mount App
├── package.json         # รายการ dependencies และ script
├── rollup.config.js     # การตั้งค่า build (ถ้าใช้ degit)
└── README.md
```

☐ ถ้าใช้ Vite จะใช้ vite.config.js แทน rollup.config.js

ไฟล์หลัก:

- App.svelte

  ⟶ Component หลักที่แสดง UI

- main.js
  ⟶ ใช้ new App({ target: ... }) เพื่อ mount ไปยัง DOM

```
// main.js
import App from './App.svelte';

const app = new App({
  target: document.body,
});

export default app;
```

- index.html
  ⟶ ไฟล์ static HTML ที่มี <script type="module" src="/main.js">

☐ **สรุปภาพรวมบทที่ 1:**

สิ่งที่คุณจะได้รู้	รายละเอียด
☐ Svelte คืออะไร	Framework ที่ compile เป็น JS ล้วน ไม่มี runtime
☐ จุดเด่น	เบา, เร็ว, เขียนน้อย, Reactive โดยธรรมชาติ
☐ วิธีติดตั้ง	ใช้ degit หรือ Vite
☐ โครงสร้างโปรเจกต์	src/, App.svelte, main.js, index.html

แนะนำ Svelte (ฉบับเชิงลึก)

☐ **1. Svelte คืออะไร? (What is Svelte?)**

Svelte คือ JavaScript Framework สำหรับการพัฒนา Web UI ที่ใช้แนวคิด **"Compile-time Framework"** คือ:

“Instead of using a virtual DOM, Svelte compiles your components into highly efficient imperative code that directly updates the DOM.”

☐ **จุดเปลี่ยนสำคัญ:**

- React/Vue/Angular: ทำงานใน Runtime ⟶ Virtual DOM ⟶ Diff ⟶ Patch DOM
- Svelte: Compile ตั้งแต่ตอน Build ⟶ DOM Manipulation ถูก generate มาเลย

Svelte ไม่ต้องมี framework runtime ขนาดใหญ่ใน browser

☐ เปรียบเทียบเชิงเทคนิค

ประเด็น	Svelte	React	Vue	Angular
DOM Update Mechanism	DOM update แบบ imperative โดยตรง	Virtual DOM (diff + patch)	Virtual DOM (Template ⟶ VDOM)	Change Detection + Zone.js
Runtime Overhead	ต่ำมาก (≈0KB)	สูง (≈40KB)	กลาง (≈30KB)	สูงมาก (≈100KB+)
Syntax	HTML, CSS, JS ในไฟล์เดียว	JSX	Template + Script	HTML Template + TS Decorator
Re-render Trigger	Assignment (count = count + 1)	setState(), useState()	data / reactive ref()	NgZone + Observables
Learning Curve	ต่ำ	ปานกลาง	ต่ำ	สูง
SSR/Static Gen Tool	SvelteKit	Next.js	Nuxt.js	Angular Universal

☐ **2. สถาปัตยกรรมและกระบวนการ Compile ของ Svelte**

☐ **Compiler-based Framework = เปลี่ยนโค้ด Component ⟶ Native JS**

เช่น โค้ดนี้ใน Svelte:

```
<script>
  let count = 0;
</script>

<button on:click={() => count += 1}>
  Clicked {count} times
</button>
```

เมื่อ Compile จะกลายเป็น JavaScript แบบนี้ (ย่อ):

```
let count = 0;
const button = document.createElement('button');
button.textContent = `Clicked ${count} times`;

button.addEventListener('click', () => {
  count += 1;
```

```
  button.textContent = `Clicked ${count} times`;
});
```

สังเกตว่าไม่มี Virtual DOM ใด ๆ เลย

Svelte แปลง template ⟶ DOM Instruction ⟶ จัดการตรงๆ

⚙ 3. วิธีติดตั้งและเริ่มต้นโปรเจกต์

วิธีที่ 1: ใช้ degit (Official Starter Template)

```
npx degit sveltejs/template my-svelte-app
cd my-svelte-app
npm install
npm run dev
```

โครงสร้างแบบ Rollup:

```
my-svelte-app/
├── public/
│   └── index.html
├── src/
│   ├── App.svelte
│   └── main.js
├── rollup.config.js
└── package.json
```

ข้อดี:

- เหมาะกับโปรเจกต์เล็ก
- เข้าใจพื้นฐานง่าย

ข้อจำกัด:

- ไม่มี routing
- ไม่มี SSR
- ไม่มี file-based structure

วิธีที่ 2: ใช้ Vite (แนะนำสำหรับโปรเจกต์จริง)

```
npm create vite@latest my-svelte-app -- --template svelte
cd my-svelte-app
npm install
npm run dev
```

โครงสร้างแบบ Vite:

```
my-svelte-app/
├── index.html
├── src/
│   ├── App.svelte
│   └── main.js
├── vite.config.js
└── package.json
```

ข้อดี:

- Build เร็ว
- รองรับ Hot Module Reloading (HMR)
- ใช้ได้กับ SvelteKit

□ 4. โครงสร้างโปรเจกต์ Svelte (แบบลึก)

public/index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Svelte App</title>
  </head>
  <body>
    <script type="module" src="/src/main.js"></script>
  </body>
</html>
```

- ไม่มี เหมือน React ⟶ Mount ที่ document.body แทน

---

src/main.js

```
import App from './App.svelte';

const app = new App({
  target: document.body,
  props: {
    name: 'world'
  }
```

```
});

export default app;
```

- target: ตำแหน่งที่ mount component
- props: ส่งค่า initial เข้า App

---

src/App.svelte

```
<script>
  export let name;
</script>

<h1>Hello {name}!</h1>
```

- ใช้ <script> เพื่อจัดการ logic
- ใช้ {} สำหรับแสดงค่าตัวแปร
- Export ตัวแปรด้วย export let

☐ **จุดเด่นเชิงลึกของ Svelte (จากการออกแบบ)**

☐ **Zero Virtual DOM**

- ประหยัด memory และ CPU
- ไม่ต้อง diff tree ใหญ่ ๆ ในทุกการอัปเดต

☐ **Compile-time Optimization**

- ตรวจจับว่า component ใช้อะไร ⟶ generate code เท่าที่จำเป็น
- ไม่มี overhead จาก unused logic

☐ **Reactive by assignment**

- ไม่ต้องใช้ setState(), ref(), หรือ signal
- เขียนง่าย เช่น count += 1; ⟶ UI อัปเดตเลย

☐ **Integrated Transition / Animation**

- ไม่ต้องลง plugin เพิ่ม เช่น react-transition-group

☐ **สรุปเชิงลึกบทที่ 1:**

ประเด็น	รายละเอียด
สถาปัตยกรรม	Compile-time framework ที่สร้าง DOM operation แบบ native
จุดเด่น	ไม่มี runtime, เล็ก เร็ว Reactive โดย assignment

ประเด็น	รายละเอียด
การติดตั้ง	degit (Rollup) หรือ Vite (แนะนำ)
โครงสร้างโปรเจกต์	มี App.svelte, main.js, index.html เป็น core
ความต่างกับ React/Vue	ไม่มี Virtual DOM, ไม่มี state hook, ไม่มี runtime overhead

---

ประวัติและวิวัฒนาการของ Svelte (Svelte History & Evolution)

☐ **2016 – จุดเริ่มต้นของ Svelte 1**

- **ผู้สร้าง:** Rich Harris (นักพัฒนาที่ The New York Times ณ เวลานั้น)
- **แนวคิดหลัก:**
  - ใช้ **compiler แทน runtime framework**
  - สร้าง UI โดยไม่มี virtual DOM
- **จุดเด่น:**
  - น้ำหนักเบา, ไม่มี runtime, ทำงานเร็ว
- **ข้อจำกัด:**
  - ไม่มี ecosystem / tooling ที่ครบ
  - ยังไม่รองรับ reactive syntax แบบ $: หรือ store

Rich Harris พัฒนา Svelte เพื่อช่วยในการสร้าง interactive graphic ในสื่อข่าวที่โหลดเร็ว

☐ **2018 – Svelte 2**

- ปรับโครงสร้างภายในเพื่อความเสถียร
- แก้ไขเรื่อง **component reactivity และ event binding**
- ยังไม่ถือว่า “พร้อมสำหรับแอปจริง” นัก
- ยังใช้โค้ด imperative แบบ manual (ต้องเขียน logic UI เองมาก)

☐ **2019 – Svelte 3 (เปิดตัว 26 เมษายน 2019)**

จุดเปลี่ยนที่แท้จริงของเฟรมเวิร์กนี้

☐ **สิ่งใหม่ใน Svelte 3:**

- **Reactivity โดย assignment** ⟶ เปลี่ยนค่าตัวแปร = อัปเดต DOM ทันที
- **Reactive statements** ($: derived = a + b)
- **Stores** (writable, readable, derived)
- **Slot, Context API, Lifecycle hooks**

- เขียนโค้ดได้ในรูปแบบที่ใกล้เคียง HTML + JS ปกติมาก
- **ไม่มี template language แบบ framework อื่น**

□ **ผลกระทบ:**

- ชนะใจนักพัฒนาจำนวนมากด้วยความเรียบง่าย
- ได้รับความนิยมใน GitHub อย่างรวดเร็ว
- เริ่มมี Third-party tools เช่น Routify, Sapper ฯลฯ

□ **2020 – การพัฒนา SvelteKit เริ่มต้น**

- **Sapper** (เฟรมเวิร์ก SSR เดิมของ Svelte) เริ่มล้าสมัย
- Rich Harris และทีมเริ่มพัฒนา **SvelteKit**

  ⟶ รองรับ File-based routing, SSR, Static generation, Adapters

---

□ **2021 – SvelteKit เปิด beta**

- ใช้ Vite เป็น bundler แทน Rollup
- รองรับ SSR, Prerender, File-based Routing
- มี adapter สำหรับ deploy หลายแบบ: Static, Vercel, Netlify, Node
- ช่วยให้ Svelte พร้อมใช้งานระดับ production แบบ Next.js

□ **2022 – Svelte เติบโตใน ecosystem**

- Svelte ติดอันดับสูงในผลสำรวจของ State of JS:
  - Satisfaction Rate สูงสุด (มากกว่า React/Vue/Angular)
  - Developer Happiness สูง
- เริ่มมี library เพิ่มขึ้น: svelte-motion, svelte-query, svelte-forms-lib

---

□ **2023 – SvelteKit เวอร์ชัน 1.0 (Stable Release)**

เปิดตัวช่วงปลายปี 2022 ⟶ ใช้งานจริงใน production ได้เต็มที่

- รองรับ SSR, Static, SPA, Hybrid
- รองรับ TypeScript เต็มรูปแบบ
- โครงสร้างชัดเจนแบบ File System Routing
- รองรับ transitions, hooks, endpoints, preloading ฯลฯ
- ได้รับการสนับสนุนจาก Vercel

⚡ **2024–ปัจจุบัน (Svelte 4 และการเตรียมสู่ Svelte 5)**

□ **Svelte 4 (เปิดตัวมิถุนายน 2023)**

- ไม่เปลี่ยน syntax ของ Svelte 3
- โฟกัสที่ **ประสิทธิภาพของ Compiler และ Developer Experience**
- ลดขนาดของ runtime ลงอีก
- รองรับระบบ tooling ใหม่ เช่น Vite, pnpm, TypeScript ได้ดีขึ้น

☐ **Svelte 5 (อยู่ระหว่างการพัฒนา ณ ปี 2024–2025)**

Svelte 5 จะเป็น "การเปลี่ยนแปลงเชิงลึก" อีกครั้ง

เป้าหมายของ Svelte 5:

- **การจัดการ reactivity ที่ดีกว่าเดิม (ผ่าน Signals)**
- ใช้ concept ของ **runes** (syntax ใหม่ เช่น @state, @effect)
- มีแนวโน้มว่าจะคล้าย React Signals หรือ Solid.js
- เพิ่มความสามารถในการ debug และ performance instrumentation
- รองรับระบบ animation, server streaming, partial hydration

☐ **สรุปลำดับเวลาโดยย่อ**

ปี	เหตุการณ์สำคัญ
2016	Svelte 1 เปิดตัวโดย Rich Harris
2018	Svelte 2 ปรับปรุง logic และ structure
2019	Svelte 3 เปลี่ยนรูปแบบ syntax และ reactive system
2020	เริ่มพัฒนา SvelteKit
2021	SvelteKit เปิด beta
2022	Svelte เป็น framework ที่มีความพึงพอใจสูงสุด
2023	SvelteKit 1.0 เปิดตัว, Svelte 4 ปรับ performance
2024+	Svelte 5 (รอเปิดตัว) มาพร้อม runes/reactivity model ใหม่

แนวโน้มการใช้งาน Svelte + สถิติ + บริษัทที่ใช้งานจริง

☐ **แนวโน้มการเติบโตของ Svelte**

☐ **1. ความพึงพอใจของนักพัฒนา (Developer Satisfaction)**

ปี	ความพึงพอใจใน Svelte (State of JS)	เปรียบเทียบ React/Vue
2019	88% (เริ่มปรากฏในแบบสำรวจ)	React ~71%, Vue ~74%

ปี	ความพึงพอใจใน Svelte (State of JS)	เปรียบเทียบ React/Vue
2021	**90.8%** (สูงที่สุดในบรรดา Frameworks)	React ~84%, Vue ~83%
2022	91.5% (ยังสูงที่สุด)	React ~80%, Vue ~79%
2023	**92.7%** (อันดับ 1 อย่างต่อเนื่อง)	React ~78%, Vue ~76%

☐ **Svelte ชนะใจนักพัฒนาสาย UI ด้วยความง่ายและ performance ที่ดีเยี่ยม**

☐ **2. แนวโน้ม Adoption โดยรวม**

- สัดส่วนผู้เคยใช้ Svelte ยังน้อยกว่า React/Vue แต่ **อัตราการเติบโตเร็วมาก**
- Developer จำนวนมากเริ่มใช้ Svelte สำหรับ:
  - **Internal Tools**
  - **Static Site / Blog / Docs**
  - **Dashboard UI**
  - **Mobile Web App**

☐ **3. เหตุผลที่นักพัฒนาเลือกใช้ Svelte**

เหตุผล	คำอธิบาย
☐ Performance สูง	ไม่มี Virtual DOM / Render เร็ว
☐ Syntax เข้าใจง่าย	HTML-first + JS-native
☐ โครงสร้างโปรเจกต์เบา	ไม่ต้อง config มาก
☐ Tooling ดีขึ้นทุกปี	SvelteKit, Vite, TS support
☐ ใช้สร้างแอป Production ได้จริง	SvelteKit 1.0 เปิดเส้นทางนี้

☐ **บริษัทที่ใช้งาน Svelte / SvelteKit ใน Production**

☐ บริษัทหลายแห่งใช้ Svelte จริงในระบบ production, ทั้งแบบ frontend, dashboard, documentation, และ internal tools

☐ **The New York Times**

- Rich Harris (ผู้สร้าง Svelte) เคยทำงานที่นี่
- ใช้ Svelte สร้าง interactive graphic ข่าว เช่น Election Maps

☐ **Square Enix**

- ใช้สำหรับ web component & in-game UI tools

☐ **Rakuten**

- ใช้ Svelte ในการพัฒนา internal tools

☐ **Spotify**

- ใช้ Svelte ใน internal UI components + rapid prototyping

☐ **IBM Carbon Design System**

- มี support สำหรับ Svelte (นอกเหนือจาก React)

☐ **Figma Community**

- บางเครื่องมือของ Figma Community ถูกเขียนด้วย Svelte

☐ **Cloudflare**

- Cloudflare Pages / Dashboard บางส่วนใช้ SvelteKit

☐ **Open Source Projects**

- Svelte.dev เว็บไซต์หลักของ Svelte
- Routify - File-based routing สำหรับ Svelte
- Svelte Material UI - Component UI library
- SvelteKit - SSR framework

☐ **การค้นหาบน Google Trends (2020–2024)**

- คำค้นหา "Svelte" และ "SvelteKit" เพิ่มขึ้นต่อเนื่อง
- การเติบโตแบบ *exponential* ในกลุ่ม startup และ frontend devs
- อัตราการค้นหาในประเทศที่นิยมสูงสุด:
    1. ☐☐ Germany
    2. ☐☐ USA
    3. ☐☐ Japan
    4. ☐☐ India
    5. ☐☐ **Thailand** (ติดอันดับกลุ่มที่เริ่มสนใจ)

☐ **Ecosystem ที่เติบโตเร็ว**

Libraries และ Tools ที่สนับสนุน:

- ☐ **SvelteKit**: Fullstack framework
- ☐ **Svelte Query**: Inspired by React Query
- ☐ **Svelte Motion**: สำหรับ animation
- ☐ **Svelte Testing Library**
- ☐ **Skeleton UI**, **Smelte**, **Carbon-Svelte**: UI Libraries

---

☐ **ข้อจำกัดที่ยังต้องพัฒนา**

ประเด็น	รายละเอียด
☐ Community ยังไม่ใหญ่เท่า React	จำนวนนักพัฒนา, package ยังน้อยกว่ามาก
☐ Ecosystem ของ plugin	ยังต้องสร้างเองหลายอย่าง
☐ Debugging ขั้นสูง	ยังไม่ดีเท่า DevTools ของ React/Vue
☐ Learning Material	มีแต่ยังไม่หลากหลายเท่าคู่แข่ง

☐ **บทสรุปเชิงกลยุทธ์**

ประเด็น	สถานะ
☐ ใช้ใน Production ได้ไหม	☐ ได้เต็มที่ โดยเฉพาะถ้าใช้ SvelteKit
☐ เหมาะกับโปรเจกต์ประเภทใด	UI-heavy app, dashboard, SPA, site ที่ต้องโหลดเร็ว
☐ ควรเรียนรู้ในปี 2025 หรือไม่	☐ มาก! โดยเฉพาะถ้าคุณชอบ developer experience ดี ๆ
☐ บริษัทใหญ่ใช้งานไหม	☐ หลายแห่งเริ่มใช้ หรือทดลองใช้อย่างจริงจัง

IDE/Editor ที่แนะนำสำหรับ Svelte (ปี 2024–2025)

☐ **1. Visual Studio Code (VS Code) – แนะนำที่สุด**

- ฟรี, น้ำหนักเบา, มี Extensions สำหรับ Svelte โดยเฉพาะ
- รองรับทั้ง Svelte 3 และ SvelteKit
- มี IntelliSense, Syntax Highlighting, Linting, Format, Go to Definition

☐ **ปลั๊กอินที่แนะนำ:**

Plugin	คำอธิบาย
☐ Svelte for VS Code	รองรับ syntax highlighting, autocomplete, error checking
☐ SvelteKit Snippets	ชุดคำสั่งลัดในการเขียนโค้ด SvelteKit
☐ Prettier + Prettier Svelte	จัดรูปแบบโค้ดอัตโนมัติ (Formatting)
☐ ESLint	ตรวจจับปัญหา syntax/code smell
☐ Tailwind CSS IntelliSense	ถ้าใช้ Tailwind กับ Svelte
☐ TypeScript ESLint	หากใช้ Svelte + TypeScript

☐ **2. WebStorm (by JetBrains)**

- มี Svelte plugin (ต้องติดตั้งเพิ่ม)
- สภาพแวดล้อมคุณภาพสูงสำหรับ TypeScript/Svelte

- เหมาะสำหรับองค์กรหรือมืออาชีพที่มี license อยู่แล้ว

วิธีติดตั้ง Plugin:

- ไปที่ **Settings ⟶ Plugins ⟶ Marketplace**
- ค้นหา Svelte และกดติดตั้ง

☐ **3. Codesandbox / StackBlitz (Online IDE)**

- สำหรับทดลองหรือสอนแบบรวดเร็ว ไม่ต้องติดตั้งบนเครื่อง
- รองรับ SvelteKit template (บางอันต้องเลือก Vite)

☐ ลองที่: https://stackblitz.com ⟶ เลือก Svelte

⚙☐ **วิธีติดตั้ง VS Code + ปลั๊กอิน Svelte (Step-by-Step)**

☐ **ขั้นตอนที่ 1: ติดตั้ง VS Code**

- ดาวน์โหลดจาก: https://code.visualstudio.com/

☐ **ขั้นตอนที่ 2: ติดตั้ง Extension ที่เกี่ยวข้อง**

เปิด VS Code แล้วกด Ctrl + Shift + X ⟶ ค้นหาและติดตั้ง:

1. **Svelte for VS Code**
2. **Prettier - Code Formatter**
3. **ESLint**
4. **SvelteKit Snippets** *(optional)*

☐ **ขั้นตอนที่ 3: ตั้งค่า Prettier สำหรับไฟล์ .svelte**

ไฟล์ .prettierrc ตัวอย่าง:

```
{
  "semi": false,
  "singleQuote": true,
  "plugins": ["prettier-plugin-svelte"],
  "pluginSearchDirs": false,
  "overrides": [
    {
      "files": "*.svelte",
      "options": {
        "parser": "svelte"
      }
    }
```

```
  ]
}
```

ติดตั้ง plugin เพิ่ม:

```
npm install -D prettier prettier-plugin-svelte
```

☐ **ขั้นตอนที่ 4: ใช้งานกับ TypeScript (ถ้าต้องการ)**

```
npm create vite@latest my-svelte-app -- --template svelte-ts
cd my-svelte-app
npm install
npm run dev
```

VS Code จะอ่าน tsconfig.json โดยอัตโนมัติ

หากติดตั้ง ESLint + TypeScript plugin ⟶ จะได้โค้ดที่ปลอดภัยมากขึ้น

☐ **เสริม: Terminal ภายใน VS Code**

- ใช้คำสั่งต่างๆ ผ่าน Terminal ใน VS Code ได้เลย:

```
npm run dev       # รันเซิร์ฟเวอร์
npm run build     # สร้าง production build
```

☐ **สรุป**

รายการ	แนะนำ
IDE หลัก	☐ Visual Studio Code
ปลั๊กอินสำคัญ	Svelte, Prettier, ESLint
Online Editor	StackBlitz, CodeSandbox
แบบองค์กร	WebStorm + Svelte Plugin
เสริม	Tailwind IntelliSense, TypeScript Support

การติดตั้ง Svelte กับ VS Code อย่างละเอียดที่สุด (ตั้งแต่เริ่มต้น)

☐ **ขั้นที่ 1: ติดตั้ง Node.js (จำเป็นก่อนใช้ Svelte)**

☐ **ทำไมต้องติดตั้ง?**

Svelte ต้องใช้ Node.js สำหรับรัน dev server (npm run dev) และ build โค้ด

☐ **วิธีติดตั้ง Node.js:**

1. ไปที่เว็บไซต์: □ https://nodejs.org
2. เลือกเวอร์ชัน:
    - □ **LTS (Long Term Support)** แนะนำ
3. ติดตั้งตามขั้นตอน Wizard (กด Next ไปเรื่อย ๆ)

□ **ตรวจสอบว่า Node.js ติดตั้งแล้ว:**

เปิด Terminal หรือ CMD แล้วพิมพ์:

```
node -v
npm -v
```

ควรได้ output เช่น:

```
v20.11.0
10.2.3
```

□ **ขั้นที่ 2: ติดตั้ง VS Code**

1. ดาวน์โหลด: □ https://code.visualstudio.com/
2. ติดตั้งตามปกติ
3. เปิดโปรแกรมแล้วไปที่ Extensions (Ctrl + Shift + X)

□ **ขั้นที่ 3: สร้างโปรเจกต์ Svelte (ด้วย Vite)**

□ **วิธีที่แนะนำ (ใช้ Vite template)**

```
npm create vite@latest my-svelte-app -- --template svelte
cd my-svelte-app
npm install
npm run dev
```

□ คำสั่งนี้จะ:

- สร้างโฟลเดอร์ my-svelte-app
- ติดตั้ง dependency
- เริ่ม dev server ที่ http://localhost:5173/

□ โครงสร้างที่ได้:

```
my-svelte-app/
├── index.html
├── package.json
├── vite.config.js
├── src/
│   ├── App.svelte
```

```
└── main.js
```

□ ขั้นที่ 4: ติดตั้ง Extension ใน VS Code

เปิด VS Code แล้วกด Ctrl + Shift + X แล้วค้นหาคำว่า:

□ **1. Svelte for VS Code**

⟶ ให้ Syntax Highlighting + IntelliSense + Error Checking

□ **2. Prettier – Code Formatter**

⟶ จัดรูปแบบโค้ดให้อัตโนมัติ

□ **3. ESLint**

⟶ ตรวจ syntax / เตือน error แบบ real-time

□ **4. Tailwind CSS IntelliSense *(ถ้าใช้ Tailwind)***

□□ ขั้นที่ 5: เปิดโปรเจกต์ใน VS Code

วิธีเปิด:

cd my-svelte-app

code .

คำสั่ง code . จะเปิด VS Code ที่โฟลเดอร์โปรเจกต์นี้

ถ้ายังไม่ใช้ได้ ให้เพิ่ม VS Code เข้า PATH ก่อน

⚙□ ขั้นที่ 6: ตั้งค่า Prettier + ESLint (จัดโค้ดให้สวยและถูกต้อง)

□ **ติดตั้ง Prettier Plugin สำหรับ Svelte:**

npm install -D prettier prettier-plugin-svelte

□ **สร้างไฟล์ .prettierrc:**

```
{
  "semi": false,
  "singleQuote": true,
  "plugins": ["prettier-plugin-svelte"],
  "pluginSearchDirs": false,
  "overrides": [
    {
      "files": "*.svelte",
      "options": {
        "parser": "svelte"
```

```
      }
    }
  ]
}
```

☐ **ตั้งค่าใน VS Code (ใน .vscode/settings.json):**

```
{
  "editor.formatOnSave": true,
  "[svelte]": {
    "editor.defaultFormatter": "esbenp.prettier-vscode"
  }
}
```

☐ เมื่อกด Save ⟶ โค้ดจะถูกจัดรูปแบบอัตโนมัติ

☐ **ขั้นที่ 7: รันโปรเจกต์ Svelte ใน VS Code**

ใน VS Code ⟶ เปิด Terminal (Ctrl + ~) แล้วพิมพ์:

```
npm run dev
```

ผลลัพธ์ที่ควรได้:

```
VITE v5.x.x  ready in 300ms
```

➜ Local: http://localhost:5173/

- เปิดเบราว์เซอร์ไปที่ http://localhost:5173
- คุณจะเห็นข้อความ “Hello world!” จาก App.svelte

☐ **ขั้นที่ 8: ทดสอบว่า Editor ทำงานครบ**

รายการ	วิธีตรวจสอบ
☐ Syntax Highlight	ลองพิมพ์ <script> และ <style> ใน .svelte ไฟล์
✍ ☐ IntelliSense	พิมพ์ let count = 0 แล้วใช้ตัวแปรใน HTML
☐ Error Checking	ลองลืม let หรือเขียนโค้ดผิด ๆ ดู
☐ Auto Formatting	กด Ctrl + S แล้วดูว่าโค้ดถูกจัด format หรือไม่
☐ ESLint (optional)	ลองพิมพ์โค้ดไม่ถูก style แล้วดู warning

☐ **แนะนำเพิ่มเติม**

☐ **ถ้าใช้ TypeScript:**