Día 2 — Ejercicio explicado a nivel granular

Explicación línea por línea y parámetros/retornos de cada instrucción clave.

Objetivo: Capturar datos del formulario y mostrar un mensaje sin recargar la página.

Ficha técnica — Instrucciones y parámetros clave

• form.addEventListener("submit", listener)
  - tipoEvento: String — "submit".
  - listener: Function(SubmitEvent event) — callback que recibe el evento del envío.
  - return: undefined.
• event.preventDefault()
  - Cancela la acción por defecto (recarga/navegación).
• document.getElementById(id: string) → Element|null
  - Retorna la referencia al nodo DOM con ese id (o null si no existe).
• HTMLInputElement.value / HTMLSelectElement.value → string
  - Contiene el valor actual del control. Lectura/escritura.
• String.prototype.trim() → string
  - Quita espacios en blanco al inicio/fin.
• Node.textContent: string
  - Asigna/lee el texto del nodo.

index.html — Fragmento con comentarios línea a línea

```
<!-- Línea 1: Sección con formulario de pedido -->
<section id="contacto"> <!-- id: facilita navegación y selección desde JS -->
  <h2>Contacto</h2> <!-- Encabezado semántico -->

  <!-- Línea 2: Formulario con id para enganchar JS -->
  <form id="form-pedido">
    <!-- Línea 3: Campo Nombre (required) -->
    <label>
      Nombre:
      <input id="inp-nombre" type="text" required> <!-- required: validación HTML5 -->
    </label>

    <!-- Línea 4: Campo Email (type=email) -->
    <label>
      Email:
      <input id="inp-email" type="email" required> <!-- type=email valida formato -->
    </label>

    <!-- Línea 5: Selección de producto -->
    <label>
      Producto:
      <select id="sel-producto" required> <!-- required: obliga a seleccionar -->
        <option value="">Selecciona...</option> <!-- value vacío: inválido -->
        <option value="americano">Café Americano</option>
        <option value="latte">Latte</option>
        <option value="pastel">Pastel de chocolate</option>
      </select>
    </label>

    <!-- Línea 6: Botón de envío (dispara 'submit') -->
    <button type="submit">Enviar pedido</button>
  </form>

  <!-- Línea 7: Área para feedback al usuario -->
  <p id="msg-resultado" aria-live="polite"></p> <!-- aria-live: anuncia cambios -->
</section>
```

js/app.js — Código explicado línea por línea

```
// Línea 1: Obtenemos referencias a elementos que usaremos más de una vez.
```

```
const formPedido   = document.getElementById("form-pedido");     // HTMLFormElement|null
const msgResultado = document.getElementById("msg-resultado");   // HTMLElement|null

// Línea 2: Solo si existe el formulario registramos el listener de 'submit'.
if (formPedido) {
  /** Línea 3: addEventListener('submit', onSubmit)
   * @param {string} tipoEvento - "submit"
   * @param {Function} listener - function onSubmit(event: SubmitEvent): void
   * El navegador llamará a 'onSubmit' cada vez que el usuario envíe el formulario.
   */
  formPedido.addEventListener("submit", function onSubmit(event) {
    /** Línea 4: event.preventDefault()
     * Evita que el navegador recargue la página y nos permite manejar el flujo con JS.
     * @param {SubmitEvent} event - objeto evento, no requiere argumentos adicionales.
     */
    event.preventDefault();

    // Línea 5: Leemos los valores actuales de los campos (type string).
    const nombre   = document.getElementById("inp-nombre").value.trim();   // trim(): limpia
espacios
    const email    = document.getElementById("inp-email").value.trim();
    const producto = document.getElementById("sel-producto").value;

    // Línea 6: Validación mínima extra (complementa a 'required' de HTML).
    if (!nombre || !email || !producto) {
      msgResultado.textContent = "Por favor, completa todos los campos.";
      return; // salida temprana si falta información
    }

    // Línea 7: Construimos feedback final usando template literals (interpolación).
    msgResultado.textContent =
      `Gracias, ${nombre}. Recibimos tu pedido de: ${producto}. Te contactaremos a ${email}.`;
  });
}
```