

Flujo completo del código de Tartas TypeScript

Leyenda de colores

-  Clase/Entidad
-  Función/Lógica
-  Evento (click / interacción)
-  HTML/UI
-  Resultado/Salida

Versión “cajas y flechas”

Clase base: **Tarta**

- Propiedades: **sabor**, **imagen**
- Métodos abstractos: **hornear()**, **ingredientesEspecificos()**
- Método común: **ingredientes()** → base + específicos
- Getter: **nombre** → “Tarta de {sabor}”



■ Clases concretas (heredan con **extends**)

- `TartaChocolate`, `TartaFresa`, `TartaQueso`, `TartaLimon`, `TartaZanahoria`
- Cada una: `super(sabor, imagen)` + implementa `hornear()` + define `ingredientesEspecificos()`



■ Estado inicial

- `const tartas: Tarta[] = []` (array vacío)
- Se irá llenando al pulsar “Añadir tarta”



🖥️ HTML inicial

- `<select id="tipo">` (elige el tipo de tarta)
- Botones: `#btn-add` (Añadir), `#btn-hornear-todas` (Hornear todas)
- Contenedores: `#lista` (tarjetas), `#resultados` (mensajes)



■ Evento: click en “Añadir”

- Lee `select.value`
- Crea instancia con `crearTarta(tipo)`
- `tartas.push(nueva)`
- Llama a `renderLista()`



■ Función: `crearTarta(tipo)`

- Devuelve instancia concreta según el tipo
- Fallback: chocolate si no coincide



■ Función: `renderLista()`

- Recorre `tartas` y genera `<article class="card">` por cada una
- Inserta imagen, nombre, ingredientes y botón **Hornear**
- Añade evento al botón para hornear esa tarjeta



■ Evento: click en “Hornear” (individual)

- Llama `tartas[idx].hornear()`
- Pasa el texto a `pushResultado()`



■ Evento: click en “Hornear todas”

- Recorre `tartas` → `t.hornear()` para cada una
- Envía textos a `pushResultado()`





■ Función: `pushResultado(texto)`

- Crea `` y lo **prepend** en `#resultados`
- Muestra el último horneado arriba



Resultados visibles

- Ejemplos:
 -  Tarta de chocolate: ... ¡Lista!
 -  Tarta de fresa: ... ¡Lista!