# **Resumen Conceptual sobre Template Strings en JavaScript**

 Definición de Template Strings

Los template strings son una manera moderna de concatenar strings y variables, o de insertar resultados de operaciones y expresiones JavaScript dentro de una cadena de texto.  
➡️ Ejemplo: `const nombreCompleto = `${nombre} ${apellido}`;` es más legible que usar concatenaciones con `+`.

 Uso de Backticks (` `)

Los template strings se escriben entre backticks (` `), que es un carácter que puede estar en diferentes lugares del teclado, como al lado del número 1 o cerca de la tecla Z. Estos permiten crear strings que pueden incluir variables o saltos de línea.  
➡️ Ejemplo: `const saludo = `Hola, ${nombre}`;` imprime 'Hola, Fernando'.

 Inclusión de Expresiones en Template Strings

Dentro de un template string se pueden incluir expresiones JavaScript, utilizando el símbolo `${}`. Esto permite ejecutar operaciones o invocar funciones dentro del string.  
➡️ Ejemplo: `const resultado = `${1 + 1}`;` mostrará el valor 2 como parte del string.

 Multi-línea en Template Strings

Una de las ventajas de los template strings es que permiten crear cadenas de texto en múltiples líneas sin necesidad de usar caracteres especiales como `\n`.  
➡️ Ejemplo: `const texto = `Línea 1  
Línea 2`;` crea un string con saltos de línea de manera natural.

 Funciones dentro de Template Strings

Es posible ejecutar funciones dentro de un template string y pasarles argumentos directamente. Esto permite combinar lógica y presentación en un solo lugar.  
➡️ Ejemplo: `const saludo = `${getSaludo(nombreCompleto)}`;` llamará la función y mostrará su retorno en el string.

 Buenas Prácticas con Template Strings

Es recomendable utilizar template strings para mejorar la legibilidad del código y evitar errores comunes en la concatenación de cadenas. Además, permite manejar fácilmente resultados de funciones o expresiones dentro de los strings.  
➡️ Ejemplo: Usar template strings siempre que se necesite concatenar variables y texto en una sola línea.

 Respaldo del Archivo

Es importante realizar copias de seguridad del código para evitar pérdidas de información. Guardar los archivos en carpetas organizadas permite mantener un registro de los ejercicios y avances.  
➡️ Ejemplo: Copiar `index.js` en la carpeta `bases` y renombrarlo como `02-template-string.js`.