El concepto de JavaScript Tagged Templates es un concepto que sirve para extender el concepto de JavaScript Template String y aporta flexibilidad sobre este último . Vamos a ver un ejemplo partiendo de una plantilla de JavaScript estandar en la cual imprimimos la información de de una persona en una página html.

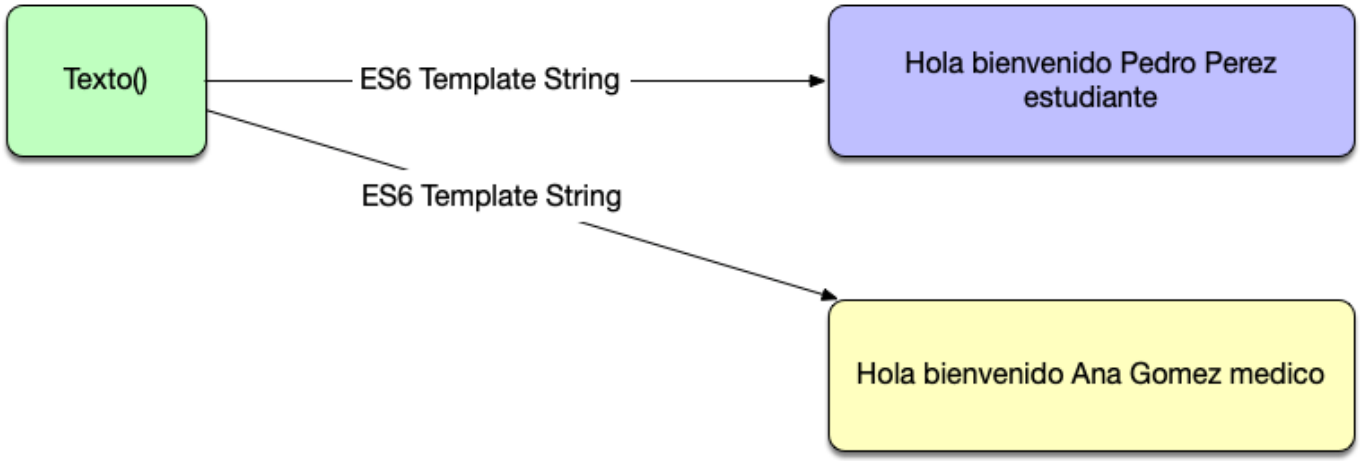
```
let nombre="pedro";
let apellidos="perez";
let profesion="estudiante";
let plantilla=texto(nombre,apellidos,profesion);
const documento= document.getElementsByTagName("body")[0];
const p= document.createElement("p");
p.innerHTML=plantilla;
document.body.appendChild(p);

nombre="ana";
apellidos="gomez";
profesion="medico";
const p2= document.createElement("p");
p2.innerHTML=texto(nombre,apellidos,profesion);
document.body.appendChild(p2);


function texto(nombre ,apellidos, profesion) {

    return `hola bienvenido ${nombre}, ${apellidos} eres
${profesion}`;
}
```

En este caso hemos usado una función que contiene la plantilla y que nos genera un sencillo párrafo con el contenido del nombre , los apellidos y la profesión de la persona utilizando plantillas de ES6.



Normalmente esta estructura nos es suficiente para el 80% de los casos.

```
hola bienvenido pedro, perez eres estudiante

hola bienvenido ana, gomez eres medico
```

Sin embargo hay situaciones en las que podemos necesitar una mayor flexibilidad y para ello nos pueden ser útiles los JavaScript Tagged Templates

JavaScript Tagged Templates

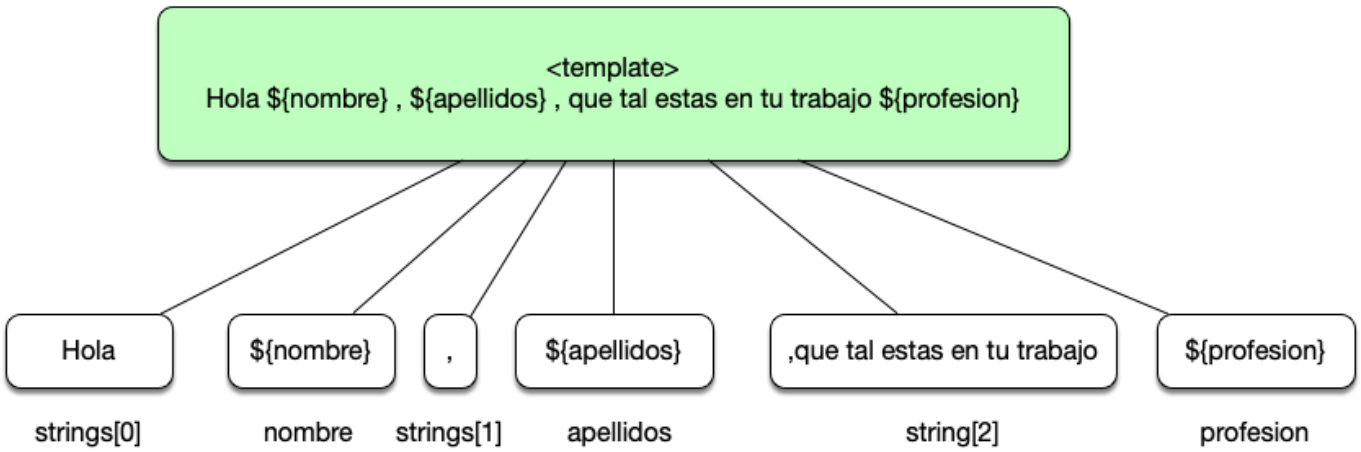
Este tipo de plantillas nos aporta una mayor flexibilidad ya que permite pasar como parametro a la función no una estructura de datos sino una plantilla en sí . Esta plantilla sera descompuesta en en su totalidad.

```
let nombre="pedro";
let apellidos="perez";
let profesion="estudiante";
const documento= document.getElementsByTagName("body")[0];
const p= document.createElement("p");
p.innerHTML=infoPersona `hola ${nombre} ,${apellidos} que tal estas en
tu trabajo de ${profesion}` ;
document.body.appendChild(p);
```

```
nombre="ana";
apellidos="gomez";
profesion="medico";
const p2= document.createElement("p");
p2.innerHTML=infoPersona `bienvenido ${nombre} ${apellidos} aqui
trabajaras de ${profesion} gracias por habernos elegido`;
document.body.appendChild(p2);


function infoPersona(strings ,nombre,apellidos, profesion) {
    return strings[0]+ nombre + strings[1] +apellidos + strings[2] +
`${profesion}` + strings[3];
}
```

En este caso la función se denomina infoPersona la función recibe una plantilla y la convertirá en un grupo de cadenas , unas cadenas serás implemente los parametros que pasa la plantilla y el resto será el texto que contiene convertido en un array



De esta forma podemos usar un tagged template para generar unas plantillas similares pero no idénticas para grupos de información comunes como se puede ver aquí

hola pedro ,perez que tal estas en tu trabajo de estudiante

bienvenido ana gomez aqui trabajaras de medico gracias por habernos elegido

El código de cada plantilla tiene un aire pero no es idéntico y genera flexibilidad.

```
p.innerHTML=infoPersona `hola ${nombre} ,${apellidos} que tal estas en tu trabajo de ${profesion}` ;

p2.innerHTML=infoPersona `bienvenido ${nombre} ${apellidos} aqui trabajaras de ${profesion} gracias por habernos elegido`;
```

Otros artículos relacionados

- Javascript Template Strings
- ES6 REST parameter
- Fetch-api
- Introducción a ES6