Template

在 ES6 之前对字符串的处理是相当的麻烦，看如下场景：

1. 字符串很长要换行

字符串很长包括几种情形一个是开发时输入的文本内容，一个是接口数据返回的文本内容。如果对换行符处理不当，就会带来异常。

2. 字符串中有变量或者表达式

如果字符串不是静态内容，往往是需要加载变量或者表达式，这个也是很常见的需求。之前的做法是字符串拼接：

```
var a = 20
var b = 10
var c = 'JavaScript'
var str = 'My age is ' + (a + b) + ' and I love ' + c
console.log(str)
```

如果字符串有大量的变量和表达式，这个拼接简直是噩梦。

3. 字符串中有逻辑运算

我们通常写代码都是有逻辑运算的，对于字符串也是一样，它包含的内容不是静态的，通常是根据一定的规则在动态变化。

```
var retailPrice = 20
var wholesalePrice = 16
var type = 'retail'

var showTxt = ''

if (type === 'retail') {
  showTxt += '您此次的购买单价是：' + retailPrice
} else {
  showTxt += '您此次的批发价是：' + wholesalePrice
}
```

看到这样的代码一定会感到很熟悉，通常大家的做法是使用上述的字符串拼接+逻辑判断，或者采用字符串模板类库来操作。

String Literals

看了上述的应用场景，就要引入 String Literals 话题，这个是用来解决字符串拼接问题，从 ES6 开始可以这样定义字符串了。

```
`string text`

`string text line 1
 string text line 2`

`string text ${expression} string text`
```

在这里你可以任意插入变量或者表达式，只要用 `${}` 包起来就好。

> [!WARNING]
> 这里的符号是反引号，也就是数字键 1 左边的键，不是单引号或者双引号

这样就可以轻松解决字符串包含变量或者表达式的问题了，对于多行的字符串，之前是这样处理

```
console.log('string text line 1\n' +
'string text line 2');
```