Templates

HTML templates are a way to define reusable markup that can be used in different parts of a web page or in different web pages altogether. A template is defined using the *<template>* element and can contain any valid HTML markup.They are not displayed by default, but rather are stored in the document's DOM tree and can be cloned and inserted into the document as needed using JavaScript.They can also include placeholders for dynamic content, which can be filled in using JavaScript.

Examples :

1. Templates :

HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <template id="myTemplate">
        <h2>Hello World</h2>
        <p>This is a pg</p>
    </template>

</body>
<script>
    const data={
        title:"This is a title",
        description : "This is description"
```

```
    }
    const template=document.getElementById("myTemplate");
    const content=template.content.cloneNode(true);
    content.querySelector("h2").innerHTML=data.title
    content.querySelector("p").innerHTML=data.description
    document.body.appendChild(content)

</script>
</html>
```

Output :

Templates

Explanation :

The html code defines a template element with the ID *myTemplate*. This template element contains a heading element *(<h2>)* with the text "Hello World", and a paragraph element *(<p>)* with the text "This is a pg".In the JavaScript code, a constant named *data* is defined with two properties: *title* and *description*.Then, the template element is retrieved using the *getElementById* method and stored in the *template* constant. The *content* of the template element is then cloned using the *cloneNode* method with the *true* argument to make a deep copy. This creates a new document fragment that is an exact copy of the template's content.The *querySelector* method is used to select the *h2* and *p* elements within the cloned content. The *innerHTML* property of these elements is then set to the corresponding values of the *data* object.Finally, the modified content is appended to

the *body* element using the *appendChild* method, effectively adding the template to the DOM.

2. Nested Templates :

HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <template id="parentTemplate">
    <h1>Parent Template</h1>
    <template id="childTemplate">
        <h2>Child Template</h2>
    </template>
    </template>
</body>
<script>
    const parentTemplate=document.getElementById("parentTemplate")
    const parentContent=parentTemplate.content.cloneNode(true)
    const
childTemplate=parentContent.querySelector("#childTemplate")
    const childContent=childTemplate.content.cloneNode(true)
    parentContent.appendChild(childContent);
    document.body.appendChild(parentContent);
</script>
</html>
```

Output :

Nested Templates

Explanation :

In the Html code the parent template has a heading and the child template has a sub-heading.In the JavaScript code, we first get the parent template by using the *getElementById()* method and create a clone of its content by using the *cloneNode()* method. Then, we get the child template from the parent content by using the *querySelector()* method and create a clone of its content as well.Finally, we append the child content to the parent content by using the *appendChild()* method and append the parent content to the HTML body. This results in the parent template being displayed on the web page with the child template as a nested element.It is important to note that we use the *content* property to access the content of the template element and not the element itself. This property returns a *DocumentFragment* object which represents the content of the template. The *cloneNode()* method is used to create a copy of the content so that we can manipulate it without affecting the original template.

JavaScript Quick Reference :

JavaScript is a programming language that is commonly used to create interactive effects on web pages. Here are some basic JavaScript concepts and methods:

1. *const:* The *const* keyword is used to declare a constant variable that cannot be reassigned a new value. For example, *const PI = 3.14.*
2. *getElementById*: The *getElementById* method is used to get an element by its ID attribute. For example, *document.getElementById("myButton")* would return the element with the ID "myButton".

3. *querySelector*: The *querySelector* method is used to get the first element that matches a specified CSS selector. For example, *document.querySelector(".myClass")* would return the first element with the class "myClass".
4. *appendChild:* The *appendChild* method is used to append a new child element to an existing element. For example, *document.getElementById("myList").appendChild(newListItem)* would append a new list item to the element with the ID "myList".
5. *cloneNode:* The *cloneNode* method is used to create a duplicate of an existing element. For example, *var clonedNode = document.getElementById("myElement").cloneNode(true)* would create a copy of the element with the ID "myElement" and all of its children. The *true* argument specifies that all child nodes should be cloned as well.