In an HTML document, the **`document.createElement()`** method creates the HTML element specified by *tagName*, or an `HTMLUnknownElement` if *tagName* isn't recognized.

Syntax

```
let element = document.createElement(tagName[, options]);
```

Parameters

tagName
A string that specifies the type of element to be created. The `nodeName` of the created element is initialized with the value of *tagName*. Don't use qualified names (like "html:a") with this method. When called on an HTML document, `createElement()` converts *tagName* to lower case before creating the element. In Firefox, Opera, and Chrome, `createElement(null)` works like `createElement("null")`.

options Optional
An optional `ElementCreationOptions` object, containing a single property named `is`, whose value is the tag name of a custom element previously defined via `customElements.define()`. See Web component example for more details.

Return value

The new `Element`.

Examples

Basic example

This creates a new `<div>` and inserts it before the element with the ID "`div1`".

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title>||Working with elements||</title>
</head>
<body>
  <div id="div1">The text above has been created dynamically.</div>
</body>
</html>
```

JavaScript

```
document.body.onload = addElement;

function addElement () {
  // create a new div element
  const newDiv = document.createElement("div");

  // and give it some content
  const newContent = document.createTextNode("Hi there and greetings!")

  // add the text node to the newly created div
  newDiv.appendChild(newContent);

  // add the newly created element and its content into the DOM

  const currentDiv = document.getElementById("div1");
  document.body.insertBefore(newDiv, currentDiv);
}
```

Web component example

The following example snippet is taken from our expanding-list-web-component example (see it live also). In this case, our custom element extends the `HTMLUListElement`, which represents the `<ul>` element.

```
// Create a class for the element
```

```
class ExpandingList extends HTMLUListElement {

  constructor() {
    // Always call super first in constructor
    super();

    // constructor definition left out for brevity
    ...
  }
}

// Define the new element
customElements.define('expanding-list', ExpandingList, { extends: "ul"
```

If we wanted to create an instance of this element programmatically, we'd use a call along the following lines:

```
let expandingList = document.createElement('ul', { is : 'expanding-list
```

The new element will be given an `is` attribute whose value is the custom element's tag name.

> **Note**: For backwards compatibility with previous versions of the Custom Elements specification, some browsers will allow you to pass a string here instead of an object, where the string's value is the custom element's tag name.