



JavaScript scrollToView

Summary: in this tutorial, you'll learn how to scroll an element into the view using its `scrollIntoView()` method.

Suppose you have a list of elements and you want a specific element to be highlighted and scrolled into view.

To achieve this, you can use the `element.scrollToView()` method. The `element.scrollToView()` accepts a boolean value or an object:

```
element.scrollToView(alignToTop);
```

or

```
element.scrollToView(options);
```

The method accepts one of the following two arguments:

alignToTop

The `alignToTop` is a boolean value.

If it is set to `true`, the method will align the top of the element to the top of the viewport or the top of the visible area of the scrollable ancestor.

If the `alignToTop` is set to `false`, the method will align the bottom of the element to the bottom of the viewport or the bottom of the visible area of the scrollable ancestor.

By default, the `alignToTop` is `true`.

options

The `options` argument is an object that gives more control over of alignment of the element in the view. However, the web browser support may be slightly different.

The `options` object has the following properties:

- `behavior` property defines the transition animation. The `behavior` property accepts two values: `auto` or `smooth`. It defaults to `auto`.
- `block` property defines the vertical alignment. It accepts one of four values: `start`, `center`, `end` or `nearest`. By default, it is `start`.
- `inline` property defines horizontal alignment. It also accepts one of four values: `start`, `center`, `end` or `nearest`. It defaults to `nearest`.

JavaScript scrollToView() Example

Suppose that you have an HTML page with a list of the programming languages as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>JS scrollToView Demo</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div class="container">
    <button class="btn">Scroll Into View</button>
    <ul>
      <li>C</li>
      <li>Java</li>
      <li>Python</li>
      <li>C++</li>
      <li>C#</li>
      <li>Go</li>
      <li>Visual Basic</li>
      <li>JavaScript</li>
      <li>PHP</li>
```

```
<li>SQL</li>
<li>R</li>
<li>Swift</li>
<li class="special">JavaScript</li>
<li>MATLAB</li>
<li>Assembly language</li>
<li>Ruby</li>
<li>PL/SQL</li>
<li>Classic Visual Basic</li>
<li>Perl</li>
<li>Scratch</li>
<li>Objective-C</li>
</ul>
</div>
<script src="scrollIntoView.js"></script>
</body>
</html>
```

Without scrolling, the `JavaScript` list item, which has a class called `special`, is not in the viewport. When the button `"Scroll Into View"` is clicked, the `JavaScript` list item is scrolled into the view:

```
let btn = document.querySelector('.btn');
let el = document.querySelector('.special');

btn.addEventListener('click', function () {
  el.scrollIntoView(true);
});
```

How it works:

- First, `select` the button with the `btn` class and list items with the `special` class.
- Then, attach an event listener to the click event of the button.
- Finally, scroll the `JavaScript` list item into the viewport by calling the `el.scrollIntoView(true)` method in the click event handler.



Java

Python

C++

Here is the [JavaScript scrollToView\(\)](#) demo.

To align the `JavaScript` list item to the bottom of the view, you pass `false` value to the `scrollIntoView()` method:

```
let btn = document.querySelector('.btn');
let el = document.querySelector('.special');

btn.addEventListener('click', function() {
  el.scrollIntoView(false);
});
```

In this tutorial, you have learned how to use the JavaScript `scrollIntoView()` method to scroll an element into the viewport.