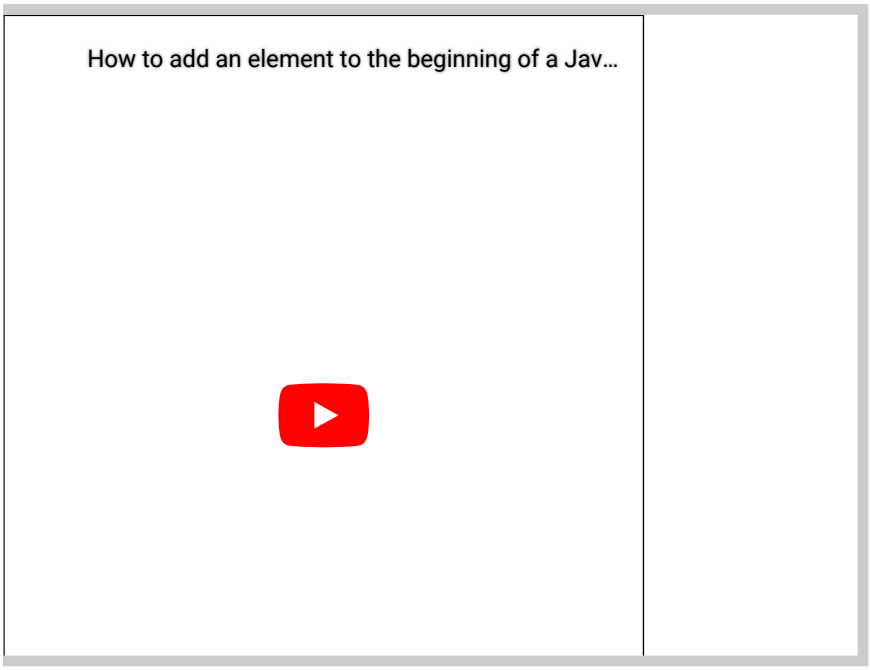
JavaScript Array.prototype.unshift()

The JavaScript unshift method adds the specified value to the beginning of the array and returns the new length of the array. It is the most efficient way to add an element to the beginning of a JavaScript array.

While some may be tempted to use the Array.prototype.splice() method to add an element to the beginning of a JavaScript array, you can trust me when I say, the Array.prototype.unshift() is the way to go. I will add though, just as an observation, I've always felt that the method name "unshift" is clunky and unintuitive. Nevertheless, it is the one that the ECMAScript specification has given us, so we'll just have to hold our noses : –) and dive right in.

So, the syntax for the Array.prototype.unshift() method is quite simple. You just chain unshift() onto your array variable name, and pass one argument to that method: the element that you want to add to the beginning of your array. For example: myArray.unshift("hello") would add the string "hello" to the beginning of the "myArray" array.

So you can pass any valid JavaScript value as the argument to the unshift() method. This could be a number, an object, another array or even an expression such as an executed function. But it's important to keep in mind that the unshift() method returns the new length of the array.
So, if an array has five elements, calling the unshift() method will return "6", because adding one element to that array has increased the length of the array to "6".



Try it yourself !

Edit in JSFiddle

- JavaScript
- Result

```
var foo = ['d', 'e', 'f'];


//the unshift method adds the element
//to the beginning of the array,
//and returns the new length

console.warn('foo.unshift("c") -> ' + foo.unshift("c")); // 4
console.dir(foo); // ['c', d', 'e', 'f']

console.warn('foo.unshift("b") -> ' + foo.unshift("b")); // 5
console.dir(foo); // ['b', c', d', 'e', 'f']

console.warn('foo.unshift("a") -> ' + foo.unshift("a")); // 6
console.dir(foo); // ['a', 'b', 'c', 'd', 'e', 'f']
```

In the above example, click the **JavaScript** tab. You'll see that we have the **foo** array, which has three elements. Each time that we call the **unshift** method, the value that we pass as an argument is added to the beginning of the array. Notice that we show the return value of the **unshift** method in the console. This allows us to see that **shift** returns the new length of the array.

Click the **Result** tab. Notice how we call the **unshift** method a total of three times. Each time, we show the return of that call to **unshift**: **"4"**, **"5",** and **"6"**. We also show that we use the **console.dir** method to inspect foo. This is so we can see the changes that are happening to the array with each call to **unshift**.

Video Example Code

If you want to download the example code, visit this page: bit.ly/kcv-array-unshift

Summary

So, big picture: the ECMAScript specification provides a number of methods on the Array.prototype object that are designed for handling mutations to the beginning and end of an array. The Array.prototype.unshift() method, for example, is specifically designed to efficiently add an element to the beginning of an array. The way it works is, the element that you pass as the sole argument is added to the beginning of the array, and the new length of the array is returned. Simple and efficient… that's what we like, right?