- **Microsoft Tech(https://www.sitepoint.com/developer-center/)** February 01, 2016

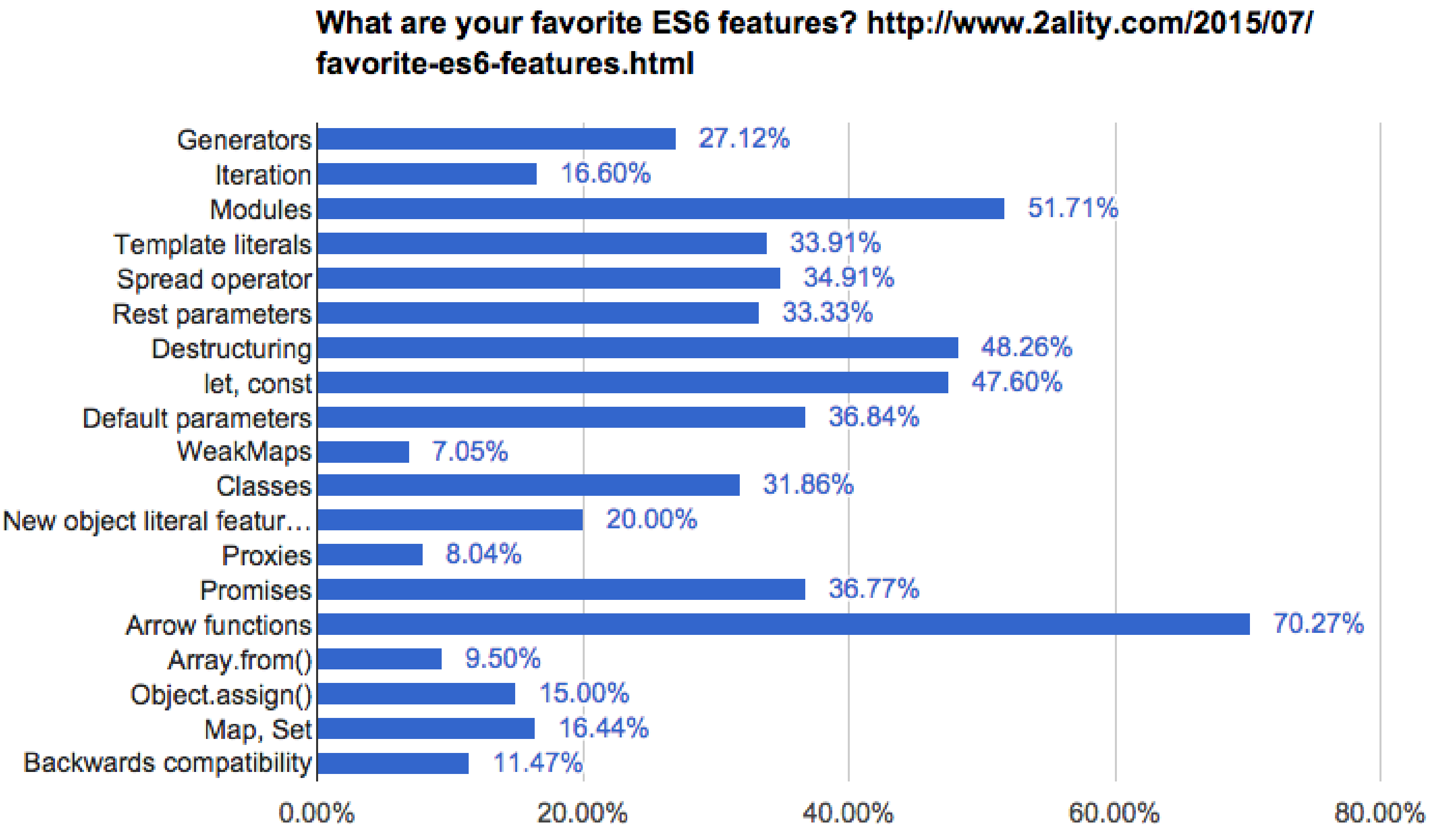
# ES6 Arrow Functions: The New Fat & Concise Syntax in JavaScript

By Kyle Pennell (https://www.sitepoint.com/author/kyle-pennell/)

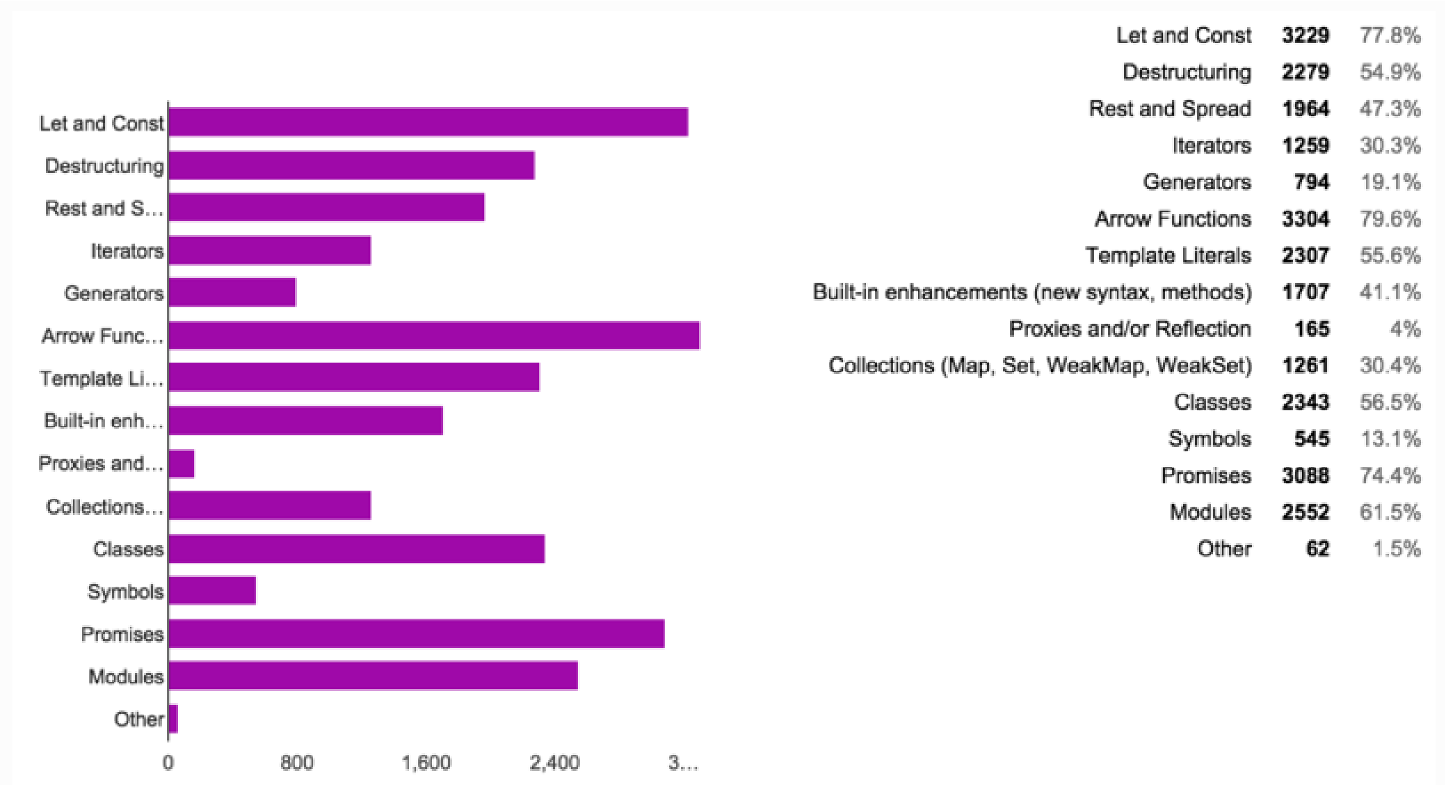
For a high-quality, in-depth introduction to ES6, you can't go past Canadian full-stack developer Wes Bos. Try his course here (https://es6.io/friend/SITEPOINT), and use the code SITEPOINT to get 25% off and to help support SitePoint.

*This article is part of a web development series from Microsoft. Thank you for supporting the partners who make SitePoint possible.*

Arrow functions are a new ES6 syntax for writing JavaScript functions. They will save developers time and simplify function scope. Surveys show they are the most popular ES6 feature:



Source: Axel Rauschmayer survey on favorite ES6 features (http://www.2ality.com/2015/07/favorite-es6-features.html)



| | | |
|---|---|---|
| Let and Const | **3229** | 77.8% |
| Destructuring | **2279** | 54.9% |
| Rest and Spread | **1964** | 47.3% |
| Iterators | **1259** | 30.3% |
| Generators | **794** | 19.1% |
| Arrow Functions | **3304** | 79.6% |
| Template Literals | **2307** | 55.6% |
| Built-in enhancements (new syntax, methods) | **1707** | 41.1% |
| Proxies and/or Reflection | **165** | 4% |
| Collections (Map, Set, WeakMap, WeakSet) | **1261** | 30.4% |
| Classes | **2343** | 56.5% |
| Symbols | **545** | 13.1% |
| Promises | **3088** | 74.4% |
| Modules | **2552** | 61.5% |
| Other | **62** | 1.5% |

Source: Ponyfoo's survey on the most commonly used ES6 features (https://ponyfoo.com/articles/javascript-developer-survey-results)

The good news is that many major modern browsers support the use of arrow functions (https://dev.windows.com/en-us/microsoft-edge/platform/status/arrowfunctionses6/?utm_source=SitePoint&utm_medium=article83&utm_campaign=SitePoint).

## More from this author

Using the New ES6 Collections: Map, Set, WeakMap, WeakSet (https://www.sitepoint.com/using-the-new-es6-collections-map-set-weakmap-weakset/?utm_source=sitepoint&utm_medium=relatedinline&utm_term=&utm_campaign=relatedauthor)

The Joys of Block Scoping with ES6 (https://www.sitepoint.com/joys-block-scoping-es6/?utm_source=sitepoint&utm_medium=relatedinline&utm_term=&utm_campaign=relatedauthor)

This post will cover the details of Arrow functions, specifically, how to use them, common syntaxes, common use cases, and gotchas/pitfalls.

# What Are Arrow Functions?

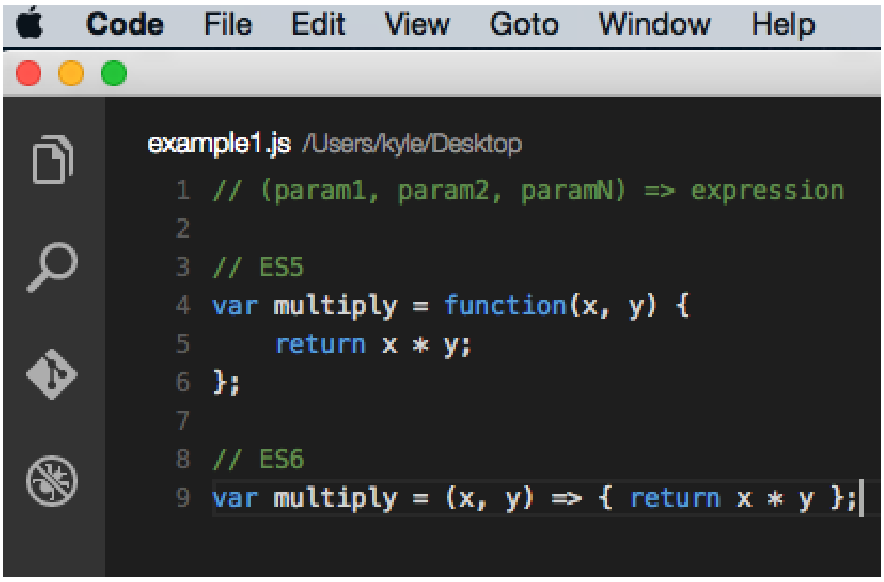
Arrow functions – also called "fat arrow" functions, from CoffeeScript (a transcompiled language (http://blogs.msdn.com/b/cdnstudents/archive/2013/09/17/visual-studio-tips-for-javascript-coders-try-coffeescript.aspx?WT.mc_id=16547-DEV-sitepoint-article83)) are a more concise syntax for writing function expressions. They utilize a new token, `=>`, that looks like a fat arrow. Arrow functions are anonymous and change the way `this` binds in functions.

Arrow functions make our code more concise, and simplify function scoping and the `this` keyword. They are one-line mini functions which work much like Lambdas in other languages like C# (https://msdn.microsoft.com/en-us/library/bb397687.aspx?WT.mc_id=16547-DEV-sitepoint-article83) or Python (http://www.diveintopython.net/power_of_introspection/lambda_functions.html). (See also lambdas in JavaScript (http://stackoverflow.com/questions/7190439/is-there-a-c-like-lambda-syntax-in-javascript)). By using arrow function we avoid having to type the `function` keyword, `return` keyword (it's implicit in arrow functions), and curly brackets.

# Using Arrow Functions

There are a variety of syntaxes available in arrow functions. EcmaScript.org has a thorough list of the syntaxes (http://wiki.ecmascript.org/doku.php?id=harmony:arrow_function_syntax) and so does MDN (https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow_functions). We'll cover the common ones here to get you started.
Let's compare how ES5 code with function expressions can now be written in ES6 using arrow functions.

### Basic Syntax with Multiple Parameters (from MDN (https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow_functions))



```
// (param1, param2, paramN) => expression

// ES5
var multiply = function(x, y) {
    return x * y;
};

// ES6
var multiply = (x, y) => { return x * y };
```

Code Example: http://codepen.io/DevelopIntelligenceBoulder/pen/wMdPoj?editors=101 (http://codepen.io/DevelopIntelligenceBoulder/pen/wMdPoj?editors=101)

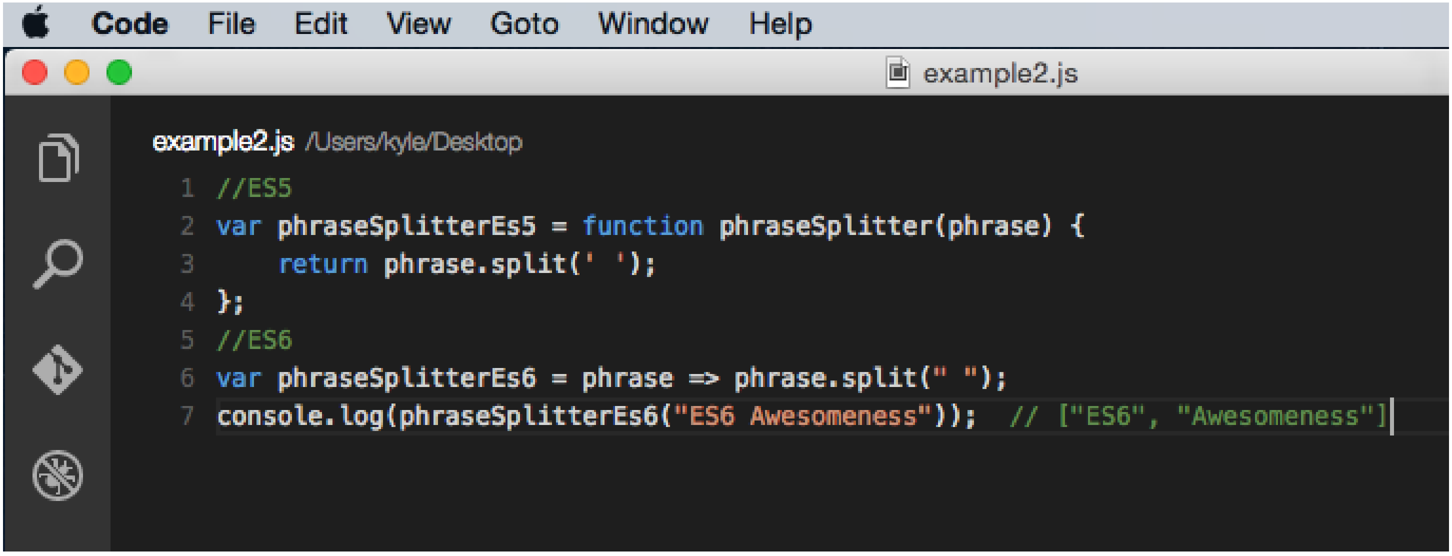
The arrow function example above allows a developer to accomplish the same result with fewer lines of code and approximately half of the typing.

Curly brackets are not required if only one expression is present. The preceding example could also be written as:

```
var multiply = (x, y) => x*y;
```

## Basic Syntax with One Parameter

Parentheses are optional when only one parameter is present

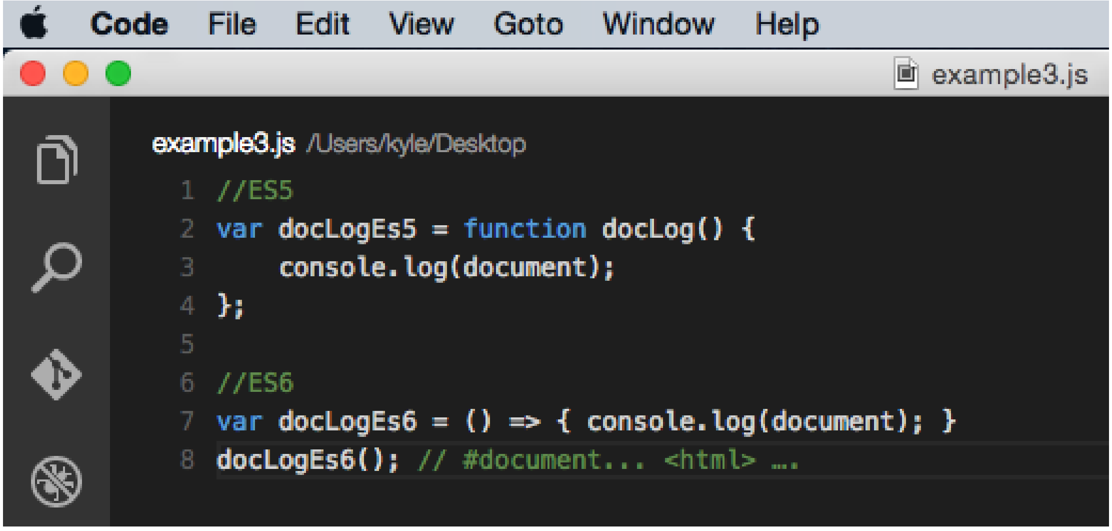


```
//ES5
var phraseSplitterEs5 = function phraseSplitter(phrase) {
    return phrase.split(' ');
};
//ES6
var phraseSplitterEs6 = phrase => phrase.split(" ");
console.log(phraseSplitterEs6("ES6 Awesomeness"));  // ["ES6", "Awesomeness"]
```

http://codepen.io/DevelopIntelligenceBoulder/pen/PZmOWQ?editors=101 (http://codepen.io/DevelopIntelligenceBoulder/pen/PZmOWQ?editors=101)

## No Parameters

Parentheses are required when no parameters are present.



example3.js /Users/kyle/Desktop

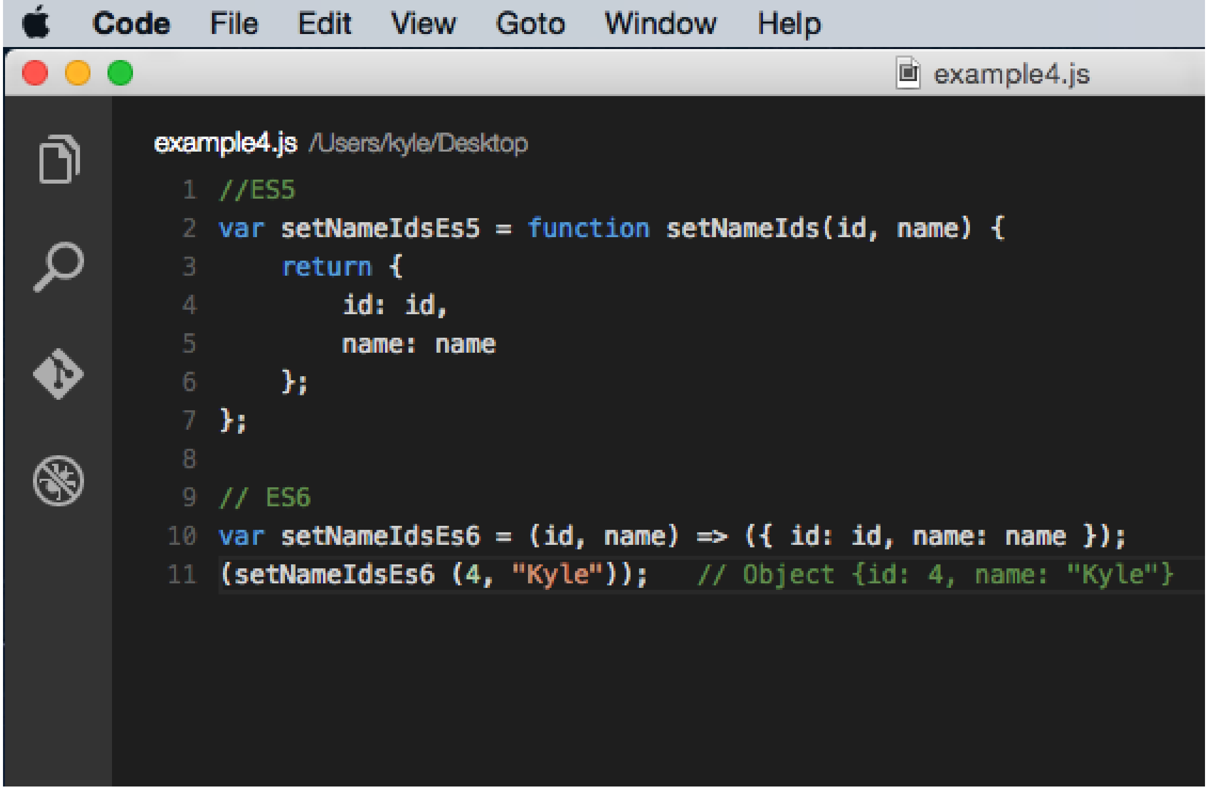
```
//ES5
var docLogEs5 = function docLog() {
    console.log(document);
};

//ES6
var docLogEs6 = () => { console.log(document); }
docLogEs6(); // #document... <html> ….
```

Code Example: http://codepen.io/DevelopIntelligenceBoulder/pen/GomOWO?editors=101 (http://codepen.io/DevelopIntelligenceBoulder/pen/GomOWO?editors=101)

## Object Literal Syntax

Arrow functions, like function expressions, can be used to return an object literal expression. The only caveat is that the body needs to be wrapped in parentheses, in order to distinguish between a block and an object (both of which use curly brackets).



example4.js /Users/kyle/Desktop

```
//ES5
var setNameIdsEs5 = function setNameIds(id, name) {
    return {
        id: id,
        name: name
    };
};

// ES6
var setNameIdsEs6 = (id, name) => ({ id: id, name: name });
(setNameIdsEs6 (4, "Kyle"));   // Object {id: 4, name: "Kyle"}
```

Code example: http://codepen.io/DevelopIntelligenceBoulder/pen/zrwPwx?editors=101 (http://codepen.io/DevelopIntelligenceBoulder/pen/zrwPwx?editors=101)

# Use Cases for Arrow Functions

Now that we've covered the basic syntaxes, let's get into how arrow functions are used.

One common use case for arrow functions is array manipulations and the like. It's common that you'll need to map or reduce an array. Take this simple array of objects:

```
varsmartPhones = [
    { name:'iphone', price:649 },
    { name:'Galaxy S6', price:576 },
    { name:'Galaxy Note 5', price:489 }
];
```

We could create an array of objects with just the names or prices by doing this in ES5:

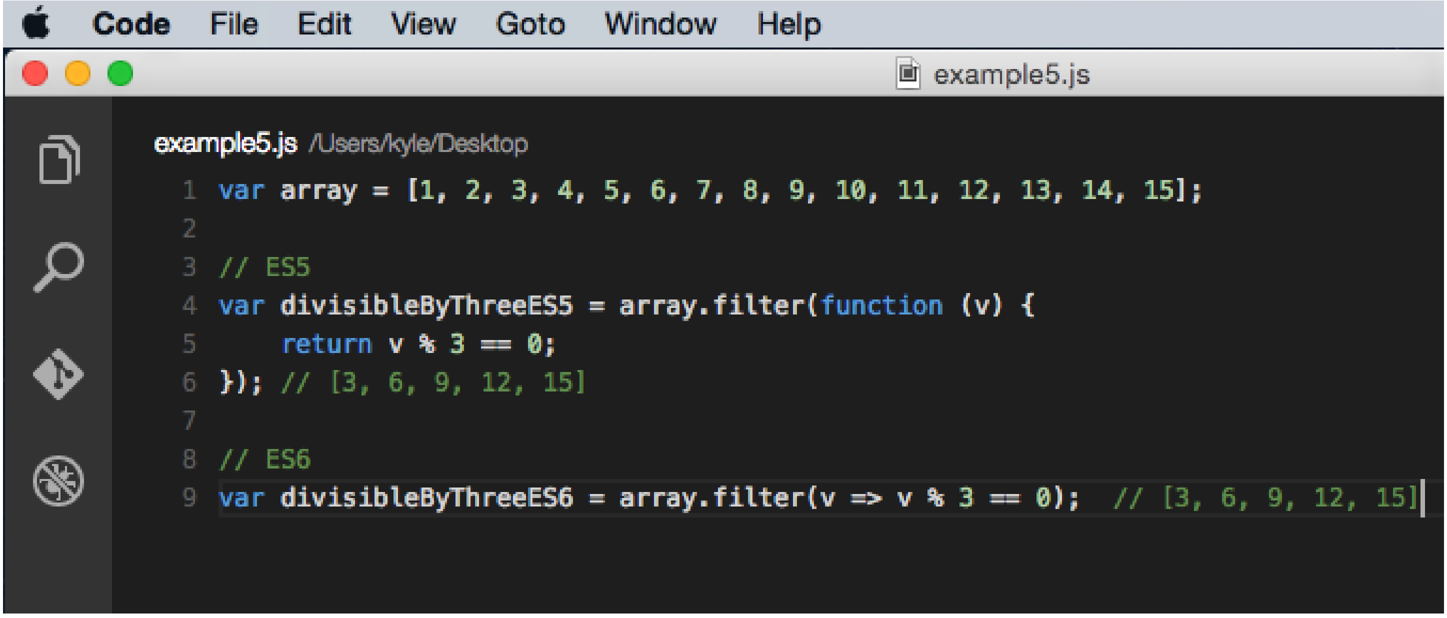
```
// ES5
console.log(smartPhones.map(
    function(smartPhone) {
    returnsmartPhone.price;
    }
)); // [649, 576, 489]
```

An arrow function is more concise and easier to read:

```
// ES6
console.log(smartPhones.map(
    smartPhone=&gt;smartPhone.price
)); // [649, 576, 489]
```

Code Example: http://codepen.io/DevelopIntelligenceBoulder/pen/jWmamX?editors=101 (http://codepen.io/DevelopIntelligenceBoulder/pen/jWmamX?editors=101)

Here's another example using the array filter method (https://msdn.microsoft.com/en-us/library/ff679973(v=vs.94).aspx):



```
var array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15];

// ES5
var divisibleByThreeES5 = array.filter(function (v) {
    return v % 3 == 0;
}); // [3, 6, 9, 12, 15]

// ES6
var divisibleByThreeES6 = array.filter(v => v % 3 == 0);  // [3, 6, 9, 12, 15]
```

Code Example: http://codepen.io/DevelopIntelligenceBoulder/pen/RrVjgL?editors=101 (http://codepen.io/DevelopIntelligenceBoulder/pen/RrVjgL?editors=101)

## Promises and Callbacks

Code that makes use of asynchronous callbacks or promises often contains a great deal of `function` and `return` keywords. When using promises, these function expressions will be used for chaining. Here's a simple example of chaining promises from the MSDN docs (https://msdn.microsoft.com/en-us/library/windows/apps/hh700334.aspx?WT.mc_id=16547-DEV-sitepoint-article83):

```
// ES5
aAsync().then(function() {
    returnbAsync();
}).then(function() {
    returncAsync();
}).done(function() {
    finish();
});
```

This code is simplified, and arguably easier to read using arrow functions:

```
// ES6
aAsync().then(() =&gt; bAsync()).then(() =&gt; cAsync()).done(() =&gt;
finish);
```

Arrow functions should similarly simplify callback-laden NodeJS code.

***What’s the meaning of this?!***

The other benefit of using arrow functions with promises/callbacks is that it reduces the confusion surrounding the `this` keyword. In code with multiple nested functions, it can be difficult to keep track of and remember to bind the correct `this` context. In ES5, you can use workarounds like the `.bind` method (which is slow (https://jsperf.com/function-bind-performance/5)) or creating a closure using `var self = this;`.

Because arrow functions allow you to retain the scope of the caller inside the function, you don’t need to create `self = this` closures or use bind.

Developer Jack Franklin (https://twitter.com/jack_franklin) provides an excellent practical example of using the arrow function lexical `this` to simplify a promise (http://javascriptplayground.com/blog/2014/04/real-life-es6-arrow-fn/):

Without Arrow functions, the promise code needs to be written something like this:

```
// ES5
API.prototype.get = function(resource) {
    var self = this;
    return new Promise(function(resolve, reject) {
        http.get(self.uri + resource, function(data) {
            resolve(data);
          });
    });
};
```

Using an arrow function, the same result can be achieved more concisely and clearly:

```
// ES6
API.prototype.get = function(resource) {
    return new Promise((resolve, reject) =&gt; {
        http.get(this.uri + resource, function(data) {
            resolve(data);
         });
       });
};
```

You can use function expressions if you need a dynamic `this` and arrow functions for a lexical `this`.

# Gotchas and Pitfalls of Arrow Functions

The new arrow functions bring a helpful function syntax to ECMAScript, but as with any new feature, they come with their own pitfalls and gotchas.

Kyle Simpson, a JavaScript developer and writer, felt there were enough pitfalls with Arrow Functions to warrant this flow chart when deciding to use them (https://github.com/getify/You-Dont-Know-JS/blob/master/es6 & beyond/fig1.png). He argues there are too many confusing rules/syntaxes with arrow functions. Others have suggested that using arrow functions saves typing but ultimately makes code more difficult to read. All those `function` and `return` statements might make it easier to read multiple nested functions or just function expressions in general.

Developer opinions vary on just about everything, including arrow functions. For the sake of brevity, here are a couple things you need to watch out for when using arrow functions.

## More about *this*

As was mentioned previously, the `this` keyword works differently in arrow functions. The methods call(), apply(), and bind() (http://javascriptissexy.com/javascript-apply-call-and-bind-methods-are-essential-for-javascript-professionals/) will not change the value of `this` in arrow functions. (In fact, the value of `this` inside of a function simply can't be changed–it will be the same value as when the function was called.) If you need to bind to a different value, you'll need to use a function expression.

## Constructors

Arrow functions cannot be used as constructors (https://msdn.microsoft.com/en-us/library/c1hcx253(v=vs.94).aspx?WT.mc_id=16547-DEV-sitepoint-article83) as other functions can. Don't use them to create similar objects as you would with other functions. If you attempt to use `new` with an arrow function, it will throw an error. Arrow functions, like built-in functions (https://msdn.microsoft.com/en-us/library/c6hac83s(v=vs.94)?WT.mc_id=16547-DEV-sitepoint-article83) (aka methods), don't have a prototype property or other internal methods. Because constructors are generally used to create classlike objects in JavaScript, you should use the new ES6 classes (https://blogs.msdn.microsoft.com/ie/2014/12/15/classes-in-javascript-exploring-the-implementation-in-chakra/?WT.mc_id=16547-DEV-sitepoint-article83) instead.

## Generators

Arrow functions are designed to be lightweight and cannot be used as generators (https://msdn.microsoft.com/en-us/library/dn858237(v=vs.94).aspx?WT.mc_id=16547-DEV-sitepoint-article83). Using the `yield` keyword in ES6 will throw an error. Use ES6 generators (https://davidwalsh.name/es6-generators) instead.

### Arguments object

Arrow functions do not have the local variable `arguments` as do other functions. The arguments object is an array-like object that allows developers to dynamically discover and access a function's arguments. This is helpful because JavaScript functions can take an unlimited number of arguments. Arrow functions do not have this object.

## How Much Use Is There for Arrow Functions?

Arrow functions have been called one of the quickest wins (http://javascriptplayground.com/blog/2014/04/real-life-es6-arrow-fn/) with ES6. Developer Lars Schöning (http://stackoverflow.com/users/578454/lyschoening) lays out how his team decided where to use arrow functions (http://stackoverflow.com/questions/22939130/when-should-i-use-arrow-functions-in-ecmascript-6):

> Use `function` in the global scope and for `Object.prototype` properties.
> Use `class` for object constructors.
> Use `=>` everywhere else.

Arrow functions, like `let` and `const` (http://www.sitepoint.com/joys-block-scoping-es6/), will likely become the default functions unless function expressions or declarations are necessary. To get a sense for how much arrow functions can be used, Kevin Smith (https://twitter.com/zenparsing), counted function expressions in various popular libraries/frameworks (https://docs.google.com/spreadsheets/d/1G-zoOMJm3iu_lQF54YRd5BvLGPs8TTwiw7hpaEAw-s8/edit) and found that roughly 55% of function expressions (http://www.2ality.com/2012/04/arrow-functions.html) would be candidates for arrow functions.

Arrow functions are here (c'mon Safari!)–they are powerful, concise, and developers love them. Perhaps it's time for you to start using them!

## More Hands-on with Web Development

This article is part of the web development series from Microsoft and DevelopIntelligence (http://www.developintelligence.com/) on practical JavaScript learning, open source projects, and interoperability best practices including Microsoft Edge (http://blogs.windows.com/msedgedev/2015/05/06/a-break-from-the-past-part-2-saying-goodbye-to-activex-vbscript-attachevent/?WT.mc_id=16547-DEV-sitepoint-article83) browser and the new EdgeHTML rendering engine (http://blogs.windows.com/msedgedev/2015/02/26/a-break-from-the-past-the-birth-of-microsofts-new-web-rendering-engine/?WT.mc_id=16547-DEV-sitepoint-article83). DevelopIntelligence (http://www.developintelligence.com/) offers instructor-led JavaScript Training (http://www.developintelligence.com/catalog/javascript-training), AngularJS Training (http://www.developintelligence.com/catalog/web-development-training/angularjs) and other Web Development Training (http://www.developintelligence.com/catalog/web-development-training) for technical teams and organizations.

We encourage you to test across browsers and devices including Microsoft Edge – the default browser for Windows 10 – with free tools on dev.microsoftedge.com (https://dev.windows.com/en-us/microsoft-edge/?utm_source=SitePoint&utm_medium=article83&utm_campaign=SitePoint):

- Scan your site for out-of-date libraries, layout issues, and accessibility (http://dev.modern.ie/tools/staticscan/?utm_source=SitePoint&utm_medium=article83&utm_campaign=SitePoint)
- Download free virtual machines for Mac, Linux, and Windows (http://dev.modern.ie/tools/vms/windows/?utm_source=SitePoint&utm_medium=article83&utm_campaign=SitePoint)
- Check Web Platform status across browsers including the Microsoft Edge roadmap (https://dev.modern.ie/platform/status/?utm_source=SitePoint&utm_medium=article83&utm_campaign=SitePoint)
- Remotely test for Microsoft Edge on your own device (https://remote.modern.ie/?utm_source=SitePoint&utm_medium=article83&utm_campaign=SitePoint)

More in-depth learning from our engineers and evangelists:

- **Interoperability best practices** (series (https://channel9.msdn.com/Blogs/BeLux-Developer/Riding-the-Modern-Web-5-things-to-consider-as-a-web-developer?WT.mc_id=16547-DEV-sitepoint-article83)):
  - How to avoid Browser Detection (https://channel9.msdn.com/Blogs/BeLux-Developer/Riding-the-Modern-Web-Avoiding-Browser-Detection?WT.mc_id=16547-DEV-sitepoint-article83)
  - Using CSS Prefix best practices (https://channel9.msdn.com/Blogs/BeLux-Developer/Riding-the-Modern-Web-CSS-Vendor-Prefixes?WT.mc_id=16547-DEV-sitepoint-article83)

    - Keeping your JS frameworks & libs updated (https://channel9.msdn.com/Blogs/BeLux-Developer/Riding-the-Modern-Web-Dealing-with-JavaScript-Libraries?WT.mc_id=16547-DEV-sitepoint-article83)
    - Building plug-in free web experiences (https://channel9.msdn.com/Blogs/BeLux-Developer/Riding-the-Modern-Web-Dealing-with-Plugins?WT.mc_id=16547-DEV-sitepoint-article83)
- Coding Lab on GitHub: Cross-browser testing and best practices (https://github.com/deltakosh/interoperable-web-development?WT.mc_id=16547-DEV-sitepoint-article83)
- Woah, I can test Edge & IE on a Mac & Linux! (https://channel9.msdn.com/Events/WebPlatformSummit/2015/Woah-I-Can-Test-Edge--IE-on-a-Mac--Linux?WT.mc_id=16547-DEV-sitepoint-article83) (from Rey Bango)
- Advancing JavaScript without Breaking the Web (http://channel9.msdn.com/Events/WebPlatformSummit/2015/Advancing-JavaScript-without-breaking-the-web?WT.mc_id=16547-DEV-sitepoint-article83) (from Christian Heilmann)
- Unleash 3D rendering with WebGL (https://channel9.msdn.com/Events/WebPlatformSummit/2015/Unleash-3D-rendering-with-WebGL-and-Microsoft-Edge?WT.mc_id=16547-DEV-sitepoint-article83) (from David Catuhe)
- Hosted web apps and web platform innovations (https://channel9.msdn.com/Events/WebPlatformSummit/2015/Hosted-web-apps-and-web-platform-innovations?WT.mc_id=16547-DEV-sitepoint-article83) (from Kiril Seksenov)

Our community open source projects:

- vorlon.JS (http://vorlonjs.com/?WT.mc_id=16547-DEV-sitepoint-article83) (cross-device remote JavaScript testing)
- manifoldJS (http://manifoldjs.com/?WT.mc_id=16547-DEV-sitepoint-article83) (deploy cross-platform hosted web apps)
- babylonJS (http://babylonjs.com/?WT.mc_id=16547-DEV-sitepoint-article83) (3D graphics made easy)

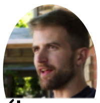
More free tools and back-end web dev stuff:

- Visual Studio Code (https://code.visualstudio.com/?WT.mc_id=16547-DEV-sitepoint-article83) (lightweight code-editor for Mac, Linux, or Windows)
- Visual Studio Dev Essentials (https://www.visualstudio.com/en-us/products/visual-studio-dev-essentials-vs.aspx?WT.mc_id=16547-DEV-sitepoint-article83) (free, subscription-based training and cloud benefits)
- Code with node.JS (https://www.microsoftvirtualacademy.com/en-US/training-

courses/building-apps-with-node-js-jump-start-8422?WT.mc_id=16547-DEV-sitepoint-article83) with trial on Azure Cloud (https://azure.microsoft.com/en-us/pricing/free-trial/?WT.mc_id=16547-DEV-sitepoint-article83)

Was this helpful? 👍 👎

More: arrow functions (https://www.sitepoint.com/tag/arrow-functions/), es6 (https://www.sitepoint.com/tag/es6/), javascript (https://www.sitepoint.com/tag/javascript-2/), JavaScript functions (https://www.sitepoint.com/tag/javascript-functions/)



Meet the author

**Kyle Pennell (https://www.sitepoint.com/author/kyle-pennell/) (https://twitter.com/kyleapennell)**

Kyle is a Technical Instructor at DevelopIntelligence (http://www.developintelligence.com/instructors/kyle). He spends his time reading, coding, biking, and exploring live music in Denver. He enjoys trying to make technical concepts more approachable and likes tinkering with music and mapping APIs.

## Login or Create Account to Comment

Login (/Premium/Sign-In?Ref_source=Sitepoint&Ref_medium=Comments&Redirect_path=%2Fes6-Arrow-Functions-New-Fat-Concise-Syntax-Javascript%2F%23commentsSection)

Create Account (/Premium/Sign-Up?Ref_source=Sitepoint&Ref_medium=Comments&Redirect_path=%2Fes6-Arrow-Functions-New-Fat-Concise-Syntax-Javascript%2F%23commentsSection)

## Popular In the Community

**WHY IS A STRING CALLED A STRING? — SITEPOINT**

**Mark Richards**
25 Jul

You were right to commence your…

**CREATING BEAUTIFUL CHARTS USING VUE.JS…**

**Caleb**
6d

Hey Yomi, thanks for the tutorial! I think yo…

**HOW TO ADD REAL-TIME NOTIFICATIONS TO…**

**Jay Bardeleben**
27 Jul

Thank you for this Yazid, I am very much..

**UI VS UX: WHAT IS THE DIFFERENCE? —…**

**brian sheridan**
4h

Super article. I was beginning to give up…

## Conversation (19)

Sort by **Oldest**

**Panayiotis “PVgr” Velisarakos**

Great article, thorough and well written, thanks for sharing!
I’m with Kyle Simpson on this feature.
I’d prefer more strict rules than optional parentheses and optional curly braces — too many optionals leave too much room for error and confusion.
I’ll get used to it no doubt, but this is not a feature that makes the developer’s life easier as it is — a few keystrokes are saved but one gets “gotchas and pitfalls” in return.

Reply · Share · *6 Likes* · 👍 👎

**M⦸ ⦸S⦸ ⦸i⦸ ⦸N⦸ ⦸L⦸u⦸n⦸d⦸**

Actually the old way lets you do the same thing with much fewer lines.

Check out the proof (using the same method as in the article):

```
//ES5
var docLogEs5 = function docLog() { console.log(document); };

//ES6
var docLogEs6 = () => {
```

See more

Reply · Share · *2 Likes* · 👍 👎

**Andrew Williams** ➦ M⦸ ⦸S⦸ ⦸i⦸ ⦸N⦸ ⦸L⦸u⦸n⦸d⦸

ES6 arrow function is still shorter and can be written in the same number of lines

```
let docLogEs6 = () => {  console.log(document) }
```

Reply · Share · *2 Likes* · 👍 👎

**M⦸ ⦸S⦸ ⦸i⦸ ⦸N⦸ ⦸L⦸u⦸n⦸d⦸** ➦ Andrew Williams

No, but compare:

1.)
WOOSH

2.)
W
O

See more

Reply · Share · *3 Likes* · 👍 👎

**Kyle Pennell** ➦ M⦸ ⦸S⦸ ⦸i⦸ ⦸N⦸ ⦸L⦸u⦸n⦸d⦸

Rekt. (I think that's what they say on Reddit?)

Funny way to show a point :)

Reply · Share · 👍 👎

Show 1 more ▾

**Jason**

Great article! I think the best feature of arrow functions is concise code, especially when dealing with callbacks or chains of filters.

```
$.get( "/api/users").done(result =>
  result.filter(user => user.isActive)
  .map(user => `${user.fName} ${user.lName}`)
)
```

Something like this gets quite a bit more messy with ES5.

Reply · Share · *2 Likes* · 👍 👎

**OlegM**

Good article man

Reply · Share · 👍 👎

**AnonymousUser**

nice job of writing a "balanced" article. that is most appreciated.

Reply · Share · 👍 👎

**ZetCoby**

pffff I dont know but for me ES5 is much more intuitive, maybe I am wrong but isn`t ES5 more similar to other languages then ES6 is?

Reply · Share · 👍 👎

**ZetCoby** ➦ ZetCoby

I am only referring to the functions here, i love the let, const and classes declarations though :D

Reply · Share · 👍 👎

**Uki C**

Well I'm trying to get some clarification as a part of my learning and to be sincere, some of these don't make sense especially when this is run it doesn't work, instead result is undefined once it logs smartPhone.price :

```
// ES5
console.log(smartPhones.map(
   function(smartPhone) {
      returnsmartPhone.price;
   }
)); // [649, 576, 489]
```

See more

Reply · Share · 👍 👎

**sarab**

Great article, very well written.
Thank you

Reply · Share · 👍 👎

Load more replies...

## LATEST MICROSOFT TECH BOOKS AND COURSES › (/premium?q=&limit=24&offset=0&page=1&content_types[]=All&slugs[]=developer-center&states[]=available&order=)

**PREMIUM COURSE**

**3h 4m**

**AtoZ: Sass** (https://www.sitepoint.com/premium/courses/atoz-sass-2977)

**PREMIUM BOOK**

**Designing with CSS Grid Layout** (https://www.sitepoint.com/premium/books/designing-with-css-grid-layout)

**PREMIUM COURSE**

**34m**

**Ember 2.0** (https://www.sitepoint.com/premium/courses/ember-2-0-2973)

**PREMIUM COURSE**

**1h 10m**

**Exploring Chrome** (https://www.sitepoint.com/premium/courses/exploring-chrome-2975)

**All Microsoft Tech Books And Courses (/Premium?Q=&Limit=24&Offset=0&Page=1&Content_types[]=All&Slugs[]=Developer-Center&States[]=Available&Order=)**

**About**

**Visit**

Our Story (/about-us/)

Advertise (/advertise/)

Press Room (/press/)

Reference (http://reference.sitepoint.com/css/)

Terms of Use (/legals/)

Privacy Policy (/legals/#privacy)

FAQ (https://sitepoint.zendesk.com/hc/en-us)

Contact Us (/contact-us/)

Contribute (/write-for-us/)

SitePoint Home (/)

Themes (/themes/?utm_source=blog&utm_medium=footer)

Podcast (/versioning-show/)

Forums (https://www.sitepoint.com/community/)

Newsletters (/newsletter/)

Premium (/premium/)

References (/sass-reference/)

Versioning (https://www.sitepoint.com/versioning/)

**Connect**

(https://www.sitepoint.com/feed/) (/newsletter/) (https://www.facebook.com/sitepoint) (http://twitter.com/sitepointdotcom) (https://plus.google.com/+sitepoint)