**Immutable Object Creation**

**Object.seal(), Object.freeze()**

[**Object.seal**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/seal)

* It prevents adding and/or removing properties from the sealed object
* Can throw a TypeError when attempting to modify (most commonly in *strict mode*)
* Using delete will return false
* It makes every property non-configurable, and they cannot be converted from data accessors to properties (and vice versa)

[**Object.freeze**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/freeze)

* Exactly what Object.seal does, plus:
* It prevents *changing* any existing properties

Neither one affects children objects (e.g. if obj is frozen or sealed, obj.el can't be changed, but obj.el.id can)

**Event Delegation**

<https://javascript.info/event-delegation>

<https://javascript.info/>

**Asynchronous Methods in JS**

[**https://medium.com/@\_bengarrison/javascript-es8-introducing-async-await-functions-7a471ec7de8a**](https://medium.com/@_bengarrison/javascript-es8-introducing-async-await-functions-7a471ec7de8a)

**CORS**

<https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>

**Angular Forms**

<http://blog.ng-book.com/the-ultimate-guide-to-forms-in-angular-2/>

**Observables**

<https://angular-2-training-book.rangle.io/handout/observables/>

**Specificity**

<https://developer.mozilla.org/en-US/docs/Web/CSS/Specificity>

**Object Comparison in JS**

<http://adripofjavascript.com/blog/drips/object-equality-in-javascript.html>

**Array.from(S) -** The **Array.from()** method creates a new, shallow-copied Array instance from an array-like or iterable object.

**new Set()** function is used to remove duplicates..

**Array.sort**(function(I,j){

return i-j;

}); ------- Is used for sorting an array

<https://github.com/wwwebman/front-end-interview-questions#middle-candidate>

<https://medium.com/@debug_mode/20-javascript-articles-to-brush-up-your-javascript-language-a2a918416821>

**What is the difference between Deferred and Promise objects? Where is Deferred object used?**

**Promise**

**Defer**

[**https://nodeschool.io/#workshopper-list**](https://nodeschool.io/#workshopper-list)

**Testing in angular -** [**https://medium.com/frontend-fun/angular-unit-testing-jasmine-karma-step-by-step-e3376d110ab4**](https://medium.com/frontend-fun/angular-unit-testing-jasmine-karma-step-by-step-e3376d110ab4)

**Angular 2 directives vs components(difference) -** [**https://thinkster.io/tutorials/angular-2-directives**](https://thinkster.io/tutorials/angular-2-directives)

**Angular ngoninit vs constructor -** [**https://toddmotto.com/angular-constructor-ngoninit-lifecycle-hook**](https://toddmotto.com/angular-constructor-ngoninit-lifecycle-hook)

**Angular breadcrumbs -** [**https://brianflove.com/2016/10/23/angular2-breadcrumb-using-router/**](https://brianflove.com/2016/10/23/angular2-breadcrumb-using-router/)

**Filters in Angular**

**Pipes in Angular**

**Dependency Injection in angular**

[**https://blog.upstate.agency/core-concepts-for-app-development-in-angular-4b121bc1b522**](https://blog.upstate.agency/core-concepts-for-app-development-in-angular-4b121bc1b522)

[**https://angularfirebase.com/lessons/angular-components-basics-top-ten/**](https://angularfirebase.com/lessons/angular-components-basics-top-ten/)

[**https://webapplog.com/node-js-fundamentals-a-concise-overview-of-the-main-concepts/**](https://webapplog.com/node-js-fundamentals-a-concise-overview-of-the-main-concepts/)

[**https://gist.github.com/lenafaure/d4ddb8c442dfbe2c7c69fb48f3caf608#file-closure6-js**](https://gist.github.com/lenafaure/d4ddb8c442dfbe2c7c69fb48f3caf608#file-closure6-js)

**2/12/2018**

[**https://gist.github.com/Geoff-Ford/c985b67a1a27deadb970d828b6a90282**](https://gist.github.com/Geoff-Ford/c985b67a1a27deadb970d828b6a90282)

[**https://medium.com/dev-bits/a-perfect-guide-for-cracking-a-javascript-interview-a-developers-perspective-23a5c0fa4d0d**](https://medium.com/dev-bits/a-perfect-guide-for-cracking-a-javascript-interview-a-developers-perspective-23a5c0fa4d0d)

[**https://performancejs.com/post/hde6d32/The-Best-Frontend-JavaScript-Interview-Questions-%28written-by-a-Frontend-Engineer%29**](https://performancejs.com/post/hde6d32/The-Best-Frontend-JavaScript-Interview-Questions-%28written-by-a-Frontend-Engineer%29)

[**https://www.maxpou.fr/js-exercises-explained**](https://www.maxpou.fr/js-exercises-explained)

[**https://github.com/ganqqwerty/123-Essential-JavaScript-Interview-Questions**](https://github.com/ganqqwerty/123-Essential-JavaScript-Interview-Questions)

[**https://www.toptal.com/javascript/interview-questions**](https://www.toptal.com/javascript/interview-questions)

[**https://medium.freecodecamp.org/3-questions-to-watch-out-for-in-a-javascript-interview-725012834ccb**](https://medium.freecodecamp.org/3-questions-to-watch-out-for-in-a-javascript-interview-725012834ccb)

[**https://www.greycampus.com/blog/programming/top-javascript-interview-questions-and-answers**](https://www.greycampus.com/blog/programming/top-javascript-interview-questions-and-answers)

[**https://blog.udemy.com/javascript-interview-questions/**](https://blog.udemy.com/javascript-interview-questions/)

[**https://davidwalsh.name/event-delegate**](https://davidwalsh.name/event-delegate)

[**https://medium.com/@dis\_is\_patrick/practical-uses-for-recursive-javascript-b8f142552f8b**](https://medium.com/@dis_is_patrick/practical-uses-for-recursive-javascript-b8f142552f8b)

[**http://altitudelabs.com/blog/what-is-the-javascript-event-loop/**](http://altitudelabs.com/blog/what-is-the-javascript-event-loop/)

**Difference between session and cookies?**

**Angular lifecycle hooks in order?**

**What is bootstrapping in angular?**