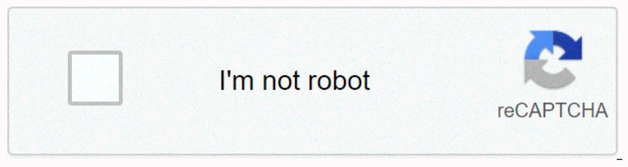

Continue

Angularjs binding function performance

I'm curious about the impact of performance code variations below as a scale of complexity. Part of the answer to this (those who use properties) has already been addressed in AngularJS: Why is ng-binding better than {{}} angular? but I would like to understand the impact of using functions instead of properties. It would seem to me that with the properties Corner knows in a sense when there are changes, while the function is opaque, so the corner would not know, and will have to evaluate every time. However, according to another SO issue mentioned above, Corner already scores each time at a direct rate anyway. So is there really a penalty for performing for using a function instead of property? And what are the pros and cons of each of them? 1 Direct viscosity with property <div>Hello, {{user.name}}</div> 2 ng-bind template with property <div ng-bind-template=Hello, {{user.name}}></div> 3 ng-link to property <div>Hello </div> 4 Straight viscosity with function <div>Hello, {{GetUserName()}}</div> 5 ng-bind template with <div ng-bind-template=Hello, {{GetUserName()}}></div> 6 ng-link to &lt function;div>Hello </div> Interpolation is one of the first things newcomers to corner training. You can't get around to writing your first corner Hello, a world without some simple interpolation like Hello, {{ $ctrl.name }}. But you may have stumbled upon the ng-bind directive, seen that it was being used, and realised it was almost the same as simple interpolation. The example above with ng-bind will look like this: 1 Hello, Why do we have both {{}} and ng-binding? Is there any reason to use over each other? In this post we will see exactly what differences and why ng-bind is better. This article will discuss some best practices to ensure maximum performance for AngularJS (1.x). We'll go over the basics of optimizing your AngularJS templates (and some animation tips) to get every possible quick win, such as reducing CPU memory and time by reducing the number of listeners and observers in your components. All code examples are written in TypeScript because this is my preferred cup of coffee broadcaster. BasicsSuppose you have a controller where the header property is assigned only once in the event cycle $onInitlife:There are some things we can do in order to optimize the template for this component: Optimize template analysisE have four compelling reasons to use ng-bind rather than pattern expressions or interpolation:You avoid visible flash expressions in the browser before they are compiled. The template expression is kept completely in memory and recalcaled (dirty check) on each binding $digest cycle.ng will only be recalcaled if the actual value of changes.ng-bind or other directives - at least in theory - is faster than analyzing expressions you need to analyze and calculate all text nodes in the template to handle template expressions. Source: 7B%7B%7D%7D/One-time bindingsSince header property is assigned only once (and never updated), we can use one-time AngularJS bindings (available from version 1.4) to make sure that property does not look:Conditional compilation and renderingLet say that you want to show the title only when a certain condition is met, You have a choice between theng show and ng-if directives:So what is the difference between them? When using theng-showdirective, the basic elements (which may well be a whole component tree) will be compiled and hidden from view immediately when AngularJS adds the CSS .ng-hide class that is added to <span> element. This means that all observers are active in the background and everything is kept up to date, even if nothing is visible to the user. When you use ng-ifdirective, the underlying component tree is removed from the DOM and inserted only back into the DOM after the condition is met meaning that additional CPU compilation and rendering of the main component tree will not be consumed. In many cases, you will experience a noticeable increase in speed to your program simply by changing the theng show directive to theng-if directive. It's one thing to note, however, that if you refer to $element in your controller and expect to change it immediately from a $onInit lifecycle event, your code will be short and more likely to throw an error in the console. Make sure to only change the item using built-in directives such as the ng class or - if you have to do so in the controller - only if you are absolutely sure that the item exists at the time. Preparing for productionE have several providers that you would like to configure before running your app in the wild:$compileProvider and $animateProvider (if you are using ngAnimate). When AngularJS analyzes templates, it will add comments throughout the DOM for debug purposes. This should be disabled. It will also check CSS class comments and directives that you should not use in any way. If you use ngAnimate to handle your animations, you should know that ngAnimate will add its ng-enter and ng-leave classes to all your elements, which in time will cause many unnecessary layout update events in the browser. They can be optimized by configuring the app this way: Some of you are likely to notice that the feature written in the example above is annotated in two lines - an injection of addiction. This is a way for AngularJS to know exactly which modules to enter into the function. If you use this everywhere, your app wins by en on the strict addiction injection that is done by adding the ng-strict-di property along with your NG   the guide will explain everything you need to maximize your app's performance for production: side note, do yourself a service and read this article explaining which CSS animations to use that don't cause heavy rendering cycles. Notes to observersThe is a common myth that too many observers equals poor performance. In fact, AngularJS can manage thousands of observers on one page without affecting either scrolling performance or upgrade time. It all depends on the nature of the observers, that is, if you look deeply at hundreds of complex objects, you will begin to see the negative impact of the user experience. Of course, with this being said, it should be noted that having thousands of observers on the page for no reason at all is definitely useful for performance - we may need cpu time for something else. Hint: I use this great Chrome extension called Angular Watchers to measure and monitor the number of watches during development! Angular observers like chromestick extension with ng-bindover template expressions, Therefore, statements are cachedUsing one-time bindings ::variable where applicablePrefer ng-if over ng showRemember to install classNameFilter on $animateProvider if you use ngAnimateRead via Running the AngularJS application in the production guide AngularJS is an open source JavaScript framework developed and maintained by Google. The tool provides everything you need to create and manage dynamic interfaces for web applications. Its modular approach to web design and the mass support community make AngularJS a popular tool among professional developers. In fact, AngularJS powers some of the internet's most high-profile traffic websites, including Google and Virgin America. This guide will serve as an introduction to AngularJS and offer tips on how to improve AngularJS performance. What is AngularJS? AngularJS was created to simplify the complex process of building and managing JavaScript applications. Based on model-view-controller, or MVC, programming structure, AngularJS is especially useful for creating web applications on one page. With a JavaScript library based on standard JS and HTML, AngularJS automatically cares about things like DOM manipulation and AJAX glue, which would otherwise have to be encoded by developers. The tool provides modular JavaScript building blocks for developers to mix, match and test. AngularJS can be quickly added to any HTML page with a simple tag. The pros and cons of AngularJSA are several features distinguishing AngularJS from its competition, including: Simplified two-way data binding. AngularJS allows you to bind data to HTML using expressions, and AngularJS directives allow developers to expand their HTML functions to create new constructs. Things like DOM manipulation and code data condensed into simple elements that can be quickly and easily embedded in HTML templates. Since AngularJS was designed as a very versatile framework, it can be used to create any type of web application. If you build a dynamic program on one page, there's most likely no better alternative. AngularJS is part of the MEAN software package, which also includes MongoDB, Express.js and Node.js. Thus, it allows you to manage both front and server projects using only JavaScript. In addition, Ruby on Rails makes a great free server. ASP.NET and C# also connect well with AngularJS.Since AngularJS was built with first-thinking functionality, it is best suited to the top-down development process. The modular nature of AngularJS makes it easy to divide the workforce in large-scale projects between different teams. It also greatly simplifies the testing and debugging process. Because they prioritize using a minimum amount of code, AngularJS apps tend to be compact and easy to edit. However, there are some things you should take into account when deciding if AngularJS is suitable for your project. First of all, AngularJS is considered a very opinion, which means that it imposes the structure on developers. For beginners and even expert programmers, this is usually good news. AngularJS was designed to be as convenient as possible, so its tools are quite intuitive. However, developers who crave more flexibility may find that you need to bypass the framework. For some projects, the use of AngularJS may be excessive. Lightweight frameworks such as .js may be the best option for static websites. AngularJS is also not equipped to handle intense DOM manipulation because it relies on dirty validation to manage DOM changes, which means that any variable changes cause DOM updates. Although this is not a problem for many websites, it can cause applications such as GUI editors and video games to fall behind. AngularJS is also trying to maintain high-traffic photo gallerys, so Instagram wasn't built on a frame. You can solve these performance issues, but it might be better to get away with an alternative like React. Otherwise, AngularJS is able to maintain forms with a high level of user interaction; After all, it does power Gmail.AngularJS optimization tipsAngularJS has plenty of built-in optimization tools, but performance complaints still plague the framework. If you don't have the massive infrastructure Google has, you might need to implement some best practices to improve the performance of the AngularJS program. If you know you need performance improvements, or if you just want to see if there's room for improvement, here are some tips for getting your AngularJS apps up to speed:1. Keep an eye on your digest cycleSjuget your AngularJS app is a good indicator of its performance. Think of the digest cycle as a cycle that tests changes in variables that are controlled. The shorter the digest cycle, the faster your program2. Limit your observers who, anytime you enter data bindings, you create $$watchers and $scopes that prolongs the digest cycle. Too much $$watchers lead to a backlog, so limit their use as much as possible.3. Use a one-time binding, if possibleIf you use an older version of AngularJS, you can use a single-time binding. To do this, simply add a colon in front of the value. If applied correctly, the value will resolve once and then disappear from the observer list. Importantly, the one-time binding feature, which was introduced in AngularJS 1.3, is not available in corner 4.0.4. Use scope.$evalAsyncIf when you try to manually activate a digest loop when it is already running, you may receive an error message. To prevent this from happening, use scope.$evalAsync instead $apply when you need to initiate a digest cycle. It queues for operations that will be performed at the end of the current cycle without installing a new one. Both DevTools Profiler and timeline tools will help you find performance bottlenecks to guide your optimization efforts. Read our in-depth guide in Chrome DevTools.6. Restricting access to DOMAccessing DOM can get expensive, so keep your DOM trees small. Don't change DOM if you can help, and don't set built-in styles to avoid JavaScript overcomputing. When you create a directive, you can assign it as an element, attribute, CSS class, or comments. If you don't need css class directives and comments, turn them off to improve performance.8 Disable debug dataInstances such as Batarang and Protractor rely on data about bindings and areas that AngularJS attaches to DOM elements. Using LodashLodash allows you to quickly rewrite the logic of the program to improve the built-in methods of AngularJS and improve the performance of the program. If the web application does not use Lodash, you can rewrite the methods yourself using native JavaScript.10. Use ng-if or ng-switch instead of ng-show Directive ng-show simply toggle CSS display for the specified item. To remove an item from DOM, use ng-if or ng-switch.11. Avoid ng-repetition when it is possibleperformance directive ng-repetition can significantly reduce performance. Fortunately, there are alternatives. For example, instead of using ng-repeat to play global navigation, you can make your own with the $interpolate provider to play the template against the object before converting it to a DOM.12 node. Use $watchCollectionWhen where you're $watch, the two options are great, but three crowds. Adding a third parameter causes AngularJS to run a deep check that ates many resources. Developers were nice enough to include work around: $watchCollection. It behaves as the third parameter for $watch, but it checks the first layer of properties of each object, so it doesn't slow down the situation as much. Use $cacheFactoryIf you want to store data that you may need to later, use the $cacheFactory. It works like any other method of memorization.14 Use console.timeIf you're having trouble debugging, console.time (Chrome DevTools) is a great tool for measuring runtime and other performance benchmarks.15 Debounce ng-modelDebunsin inputs using the ng-model directive can limit the digest cycle. For example, applying ng model parameters ={debounce:200} ensures that the digest cycle does not start more than once every 200 ms.16. Use $filterAngularJS runs DOM filters twice during each digest cycle: first to detect changes, and then to update the values that have changed. To save some time, the $filter allows you to pre-process the data before it is sent to the submission, and thus skips the DOM17 analysis process. Dense scopingKeep your variables are tight within, so that JavaScript garbage collector can release some memory every now and then. If all else fails, you can reduce the number of items that get looped by numbering or endless scrolling. AngularJS even has a directive called ngInfiniteScroll for this purpose. It is always more effective to use best practices from the beginning rather than continuing to come back and make changes. Before you start coding, think carefully about how you can limit tethers, observers and expensive directives like ng replay. Refer to the official AngularJS documents for troubleshooting and additional help to get started. You can find a number of tools dedicated to improving the testing and performance of AngularJS applications. Here are some of the best options:1. ProtractorProtactor comes straight from the corner team. This software package makes it easy to run automatically soothing testing. Because Protractor is built over the WebDriverJS and Selenium server, it boasts all its features, which means you can use selenium's grid feature to run tests in multiple browsers at the same time. You can write your own test suts using Jasmine or Mocha.2. WebDriverIOHow to implement the W3C webDriver API, WebDriverIO is more flexible than WebDriverJS. Its command line interface makes setting up tests simple enough for non-programmers to find out. WebDriverIO users receive excellent support and an active developer community.3. NightwatchJSNightwatchJS is also a custom

implementation of the W3C webdriver API. Easy to expand and customize, this tool comes with its own testing base and assertion mechanisms, but it lacks the level of support that WebDriverIO and Protractor.4 have. TestingTaksodynamic commands allow you to testWhich synchronize with different server timeouts to ensure accurate cutting-cut testing of angular applications. Script-free function does TestingWhy very popular among non-programmers.5. The BatarangThe Batarang tool is a Chrome extension created by the Angular team to make debugging easier. It boasts several handy features, but its most useful for tracking performance metrics. As a last note, also considering using CDN to assets of AngularJS. This will optimize download times for visitors around the world thanks to the smart cache. While there are dozens of JavaScript frameworks these days, AngularJS remains a favorite for many, so it won't be going anywhere anytime soon. To get more time out of your apps, you need to make optimizing AngularJS a common habit. Fine-tuning the performance of the app can allow you to provide more content when using less code, which frees up resources that could be better spent elsewhere. elsewhere.