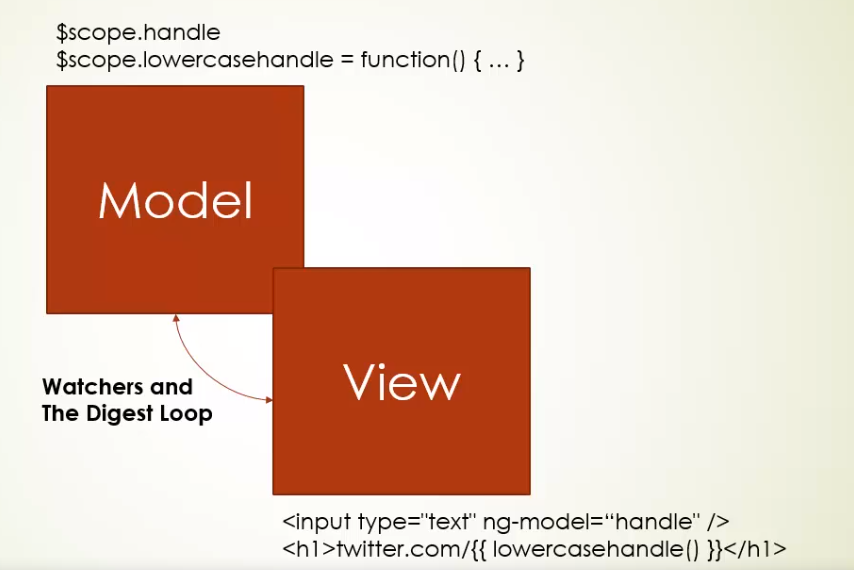
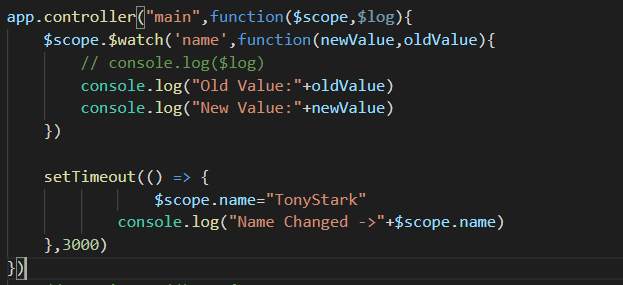
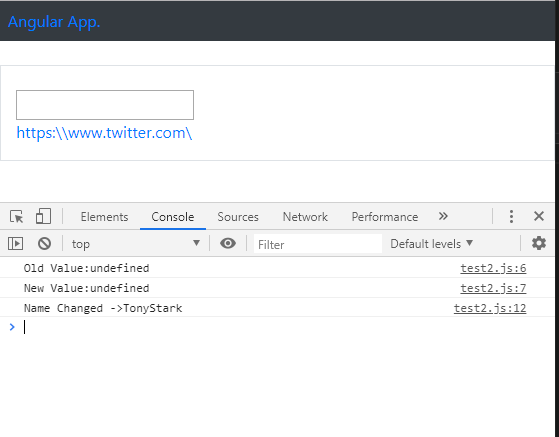
**Watchers and Digestion Loop in Angular.js**

****

Angular.js loops continuously to check for an update into the modal so that view could be changed, This occurs on a single thread, though it could be tricked, means, we can change the modal without letting angular know about it, but how, by suppling data onto another thread, Let’s see how.



Changes reflected in the browser:



In the above function, we have used watcher to check for old and new value held by angular, but suddenly through an another thread (used by setTimeout function due to async. behavior) it updates the modal without being known to angular, this way, angular doesn’t fire up digestion loop and view doesn’t get updated. THIS IS A MAJOR FLAW.

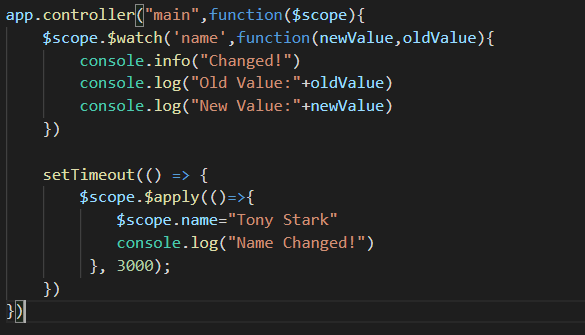
The name variable has been altered and updated value has been reflected into the console, but not into the angular eco-system, the above URL still stands short of the name.

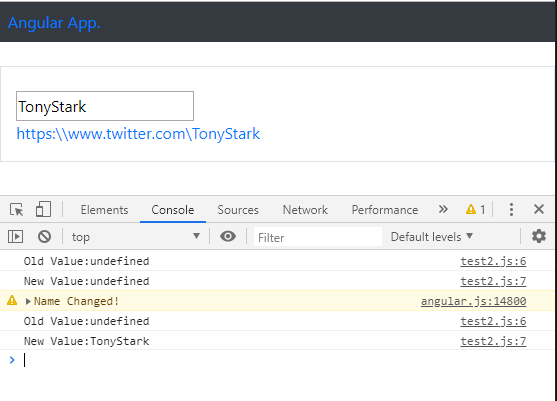
**Solution:**

But this can be debugged by invoking angular or telling in advance about tentative future updates, by wrapping any async. Function into **$apply()** directive. This will invoke angular to fire up the digestion loop and check for updates and update the view.

**How to know when to use:**

You need to use $scope.$apply only when you are performing an operation which is outside the Angular context - any changes will not get picked up by the digest loop. The alternative is to use the angular implementations for what you're trying to do, Eg. use the $http service as opposed to the native XMLHttpRequest.

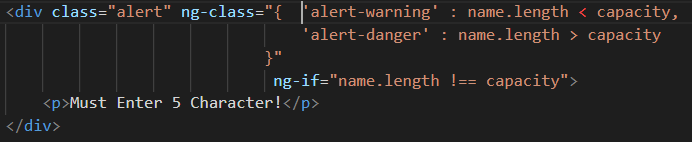


Changes reflected into the browser:

Now by wrapping setTimeout function into **$apply** directive, Angular.js automatically updates the content.  we need **$apply** to tell Angular - 'run a digest loop to check if things need to be updated'.

**AngularJS Common Directives**

* **ng-if:** used as an attribute to extend HTML tag, Ex. <div ng-if = ”parameter”> < /div>. parameter = JS statement which returns Boolean value.
* **Ng-class:** used as an attribute for dynamically injecting classes into HTML tags, Ex. <div ng-class=”parameter”> < /div>. parameter = JS object containing class and expression which returns boolean. Ex.



* **Ng-cloak:** Most of time we have seen when Internet speed isn’t fast and browser render the static text like this {{name}} which was supposed to be updates by angular js with respective modal value, but delay in JS execution it appears that way, what ng-cloak does is, It doesn’t render the static text i.e. {{name}} until JS execution happens, and once JS kicks in and updates the view only then the modal value appears on the screen.
* **http:** (Very Important)
* **$routeProvider:** in-Short -> what should I do when I see a particular thing in hash. Used for defining routes.
* Explain Singleton objects in Angular.
* Write about Scope in Angular -> where to use “@ “(Text), “=” (Object) and “&” (function) operator.
* **Transclusion:** Include one document inside other. (Place a copy of one document at a particular point inside another)