Merging of EventHandler / ShodowDOMBoundry with View

*Status: (Draft, Final)*

*Authors: misko@google.com*

*This document is published to the web as part of the public* [*Angular Design Docs*](https://drive.google.com/#folders/0BxgtL8yFJbacUnUxc3l5aTZrbVk) *folder*

# Objective

EventHandler (also known as ShodowDOMBoundry) relies on ElementProbe to get its job done. ElementProbe is optional hence EventHandler will not work when it is disabled.

# Detailed Design

EventHandler functionality should be merged with View. This would allow the event handling to be cacheable together with the View since registering listeners can now be reused. Currently the EventHandler spans multiple views. The issue is that when an event happens EventHandler needs to find scope on which to fire the event. The searching for scope is what requires ElementProbe. By tying the ElementHandler to view the problem goes away since each View has exactly one Scope.

This will also allow forms to register change listeners (see [PR#985](https://github.com/angular/angular.dart/pull/985)) to input, since currently this functionality requires the storage of a callback function with the element. The new approach should make the event listeners declarative in the directive annotation.

|  |
| --- |
| @Decorator(  selector: 'input',  events: {'change': 'onChange'}  )  class Directive {  onChange(Event e) => ...;  } |

This would allow the decoration of template HTML as

|  |
| --- |
| <input class="ng-binding ng-binding-123"> |

The ng-binding-123 would allows us to know that we have to find the DirectiveInjector index 123 in the View. Than we can look through all of the directives on the DirectiveInjector and search for the ones which need to have the event delivered.

# Performance Considerations / Test Strategy

How you’ll be fast.

# Work Breakdown

Description of development phases and approximate time estimates.