# Proposal: Drop methods and getter/setter support in Databinding to get rid of polling during digest

Author: [tbosch@google.com](mailto:tbosch@google.com)

Date: 4/21/2014

## Why?

* Expressions that do not use getters/setters nor methods (except for pure functions) can be observed without polling them!
  + either use Object.observe if available
  + or inject a setter/getter to know about the changes
* With this we would not need optimizations like
  + separating reaction functions in digest vs flush phase to eliminate one polling iteration
  + onetime binding vs live binding
  + binding regions, ...

Pro

* Performance
* People don't write code that they would need to optimize afterwards

Con

* Makes writing simple examples more verbose
* Dart will not support Object.observe in the near future, so the codebase will get out of sync

|  |
| --- |
| @DecoratorDirective({  bind: {  'input': 'input',  'result': 'result'  }  })  class Adder {  **get** result() {  return this.input + 10;  }  }  @DecoratorDirective({  bind: {  'input': 'input',  'result': 'result'  },  **observe: {'input': 'inputChanged'}**  })  class Adder {  **inputChanged((newValue) => {**  this.result = newValue + 10;  });  } |

## Optimization Roadmap

step 1 - use dirty checking for everything

step 2 - support O.o for observable paths + dirty checking for everything else

step 3 - allow unobservable paths to be annotated with observable input path that determine the output of the unobservable path

step 4 - when O.o is unavailable patch objects and arrays with getter/setter interceptors that would tell us when something has changed without dirty checking

**End result**: Writing an application that does not use dirty checking at all should be possible easily.