

# Advanced Jest Matchers

---



**Daniel Stern**  
CODE WHISPERER  
[@danieljackstern](#)



# What Is a Matcher?



Also known as an  
*assertion* or  
*expectation*



Represents a claim  
that a value will be  
equal (or not) to  
something



Throws an error (test  
fails) if matcher's  
claim is not validated

```
function test() {  
  const value = getValue42();  
  if (value !== 42) {  
    throw new Error(/**/);  
  }  
}
```

```
function test2() {  
  const value = getValue42();  
  expect(value).toEqual(42);  
}
```

- ◀ Handwritten test
  - ◀ If statement (condition)
  - ◀ Throws an error
  - ◀ In test runner, the error results in test failing, not in the application failing
- 
- ◀ Equivalent test using a matcher
  - ◀ Matcher is equivalent to *if* statement above



# Exploring Matchers

---



“To be or not to be.  
That is the question.”

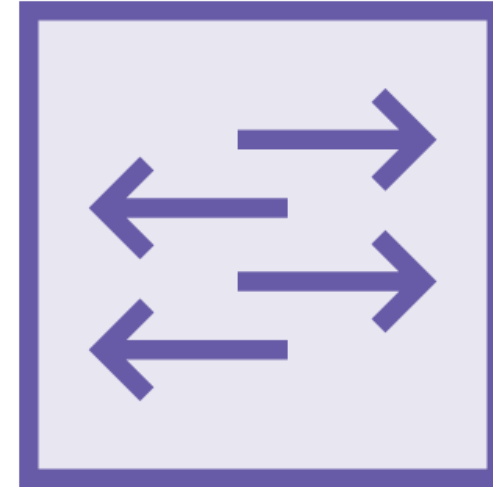
– William Shakespeare



Not



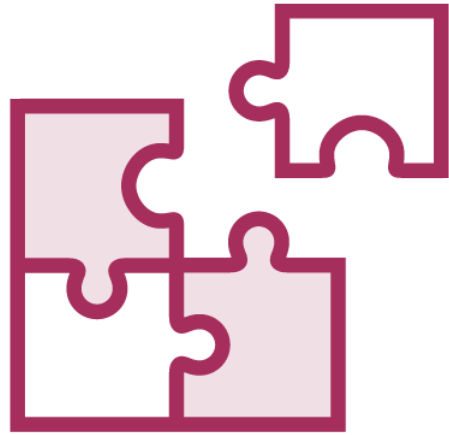
Reverses an assertion



If assertion without not would pass,  
assertion with not will fail



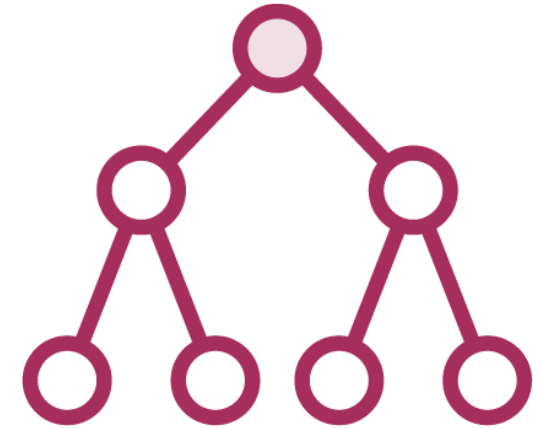
# To Be and To Equal



Verify that two values  
are equivalent



Two arrays with  
matching elements  
are equal, but not  
identical



Very general – use  
more specific matcher  
when possible

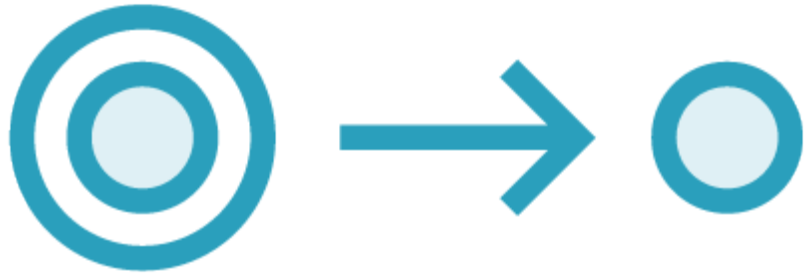
“Unlike most other programming languages, there is no separate integer type, so 1 and 1.0 are the same value.”

– Douglas Crockford





# To Be Close To



Like *toEqual* for numbers, but  
assertion still passes if numbers  
are close but not equal



For assertions involving floating  
point numbers

# To Contain & To Have Length



Array matchers which verify the contents and size of a collection



More succinct than combining *toEqual* with array operators *includes*, *length*, etc.

# Demo



Add unit tests for reducer

Utilize *toContain* and *toHaveLength* for testing an array

Work with *not* Matchers

