## Controlling How Parameters Are Matched



Nathan Taylor
SOFTWARE ENGINEER
@taylonr taylonr.com



## Verifying Creation Date



## What Are Matchers?



## Matcher

Objects that contain a rule, and allow you to control the specificity of your test.



## Matcher

Objects that contain a **rule**, and allow you to control the specificity of your test.



#### Where Are Matchers Used?

calledOn calledWith returned withArgs



```
stub
   .withArgs({...})
   .returns(true);

stub
   .withArgs(<matcher>)
   .returns(true);
```

■ Must supply the entire object

Allow more fuzziness in withArgs



#### What Can Be Matched?



```
const stub = sinon.stub();
stub.withArgs(sinon.match(1517)).returns(true);
```



```
const stub = sinon.stub();
stub.withArgs(sinon.match(1517)).returns(true);
stub.withArgs(1517).returns(true);
```



```
const stub = sinon.stub();
stub.withArgs(sinon.match(1517)).returns(true);
stub.withArgs(1517).returns(true);
stub(1517);
stub('1517');
```



## String Matcher

```
const stub = sinon.stub();
stub.withArgs(sinon.match('sight')).returns(true);
```



#### String Matcher

```
const stub = sinon.stub();
stub.withArgs(sinon.match('sight')).returns(true);
stub('Pluralsight');
```



#### Regex Matcher

```
const stub = sinon.stub();
stub.withArgs(sinon.match(/[abc](1|9)/)).returns(true);
```



#### Regex Matcher

```
const stub = sinon.stub();
stub.withArgs(sinon.match(/[abc](1|9)/)).returns(true);
stub('a9');
stub('a2');
```



## Object Matcher

```
const stub = sinon.stub();
stub.withArgs(sinon.match({id: 1})).returns(true);
```

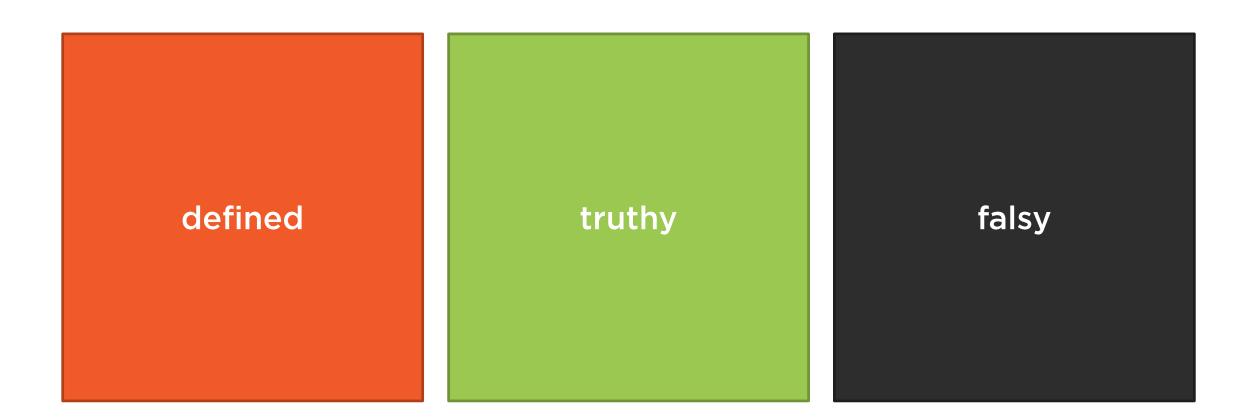


## Object Matcher

```
const stub = sinon.stub();
stub.withArgs(sinon.match({id: 1})).returns(true);
stub({id: 1});
stub({id: 1, name: 'Charles Spurgeon'});
```



#### More Value Matchers



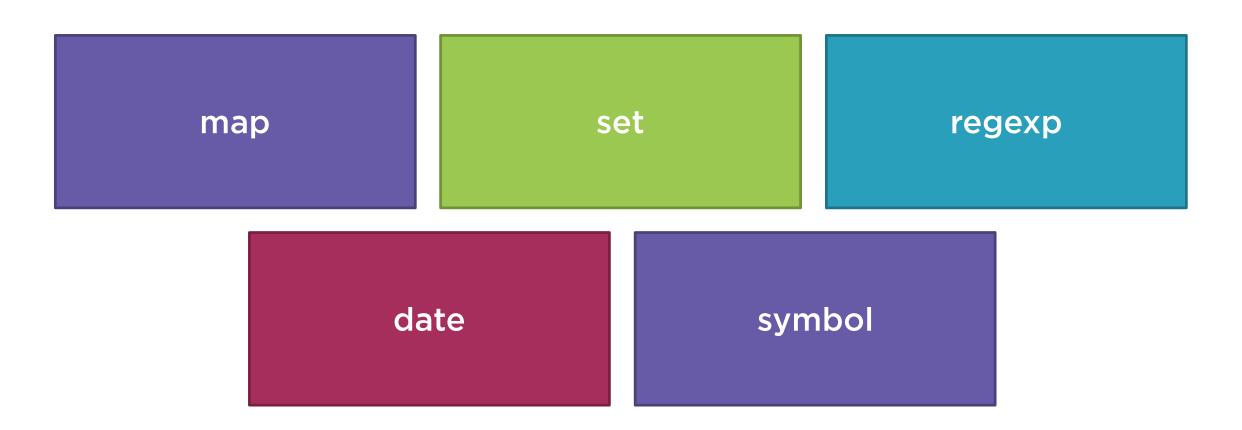


## Data Type Matchers

bool number string object func array



## Data Type Matchers (Continued)





```
const stub = sinon.stub();
stub.withArgs(sinon.match.number.or(sinon.match.string))
    .returns(true);
```

Combining Matchers





Equipped to complete the test



## Completing Date Verification



## Array Matchers



## Matching an Array

```
const stub = sinon.stub();
stub.withArgs(sinon.match.array).returns(true);
stub(['a', 'b']); //true
```



#### Array Starts With...

```
const stub = sinon.stub();
stub.withArgs(sinon.match.array.startsWith([1]))
    .returns('Numeric');
stub.withArgs(sinon.match.array.startsWith(['a']))
    .returns('Alpha');
```



## Array Functions

endsWith startsWith contains deepEquals



## Checking Every Value

```
const stub = sinon.stub();
stub.withArgs(sinon.match.every(sinon.match.number))
    .returns('numeric');
```



## Checking Every Value

```
const stub = sinon.stub();
stub.withArgs(sinon.match.every(sinon.match.number))
    .returns('numeric');
stub.withArgs(sinon.match.some(sinon.match.string))
    .returns('alpha');
```



## Object Array Matcher

```
const stub = sinon.stub();
stub.withArgs(sinon.match.array.contains([{
    name: 'Spurgeon'
}])).returns(true);
stub([{name: 'Spurgeon'}]);
```



## **Custom Matchers**



## Object Contains Array

```
const stub = sinon.stub();
stub.withArgs(arrayContainsObjectMatch({
   id: 1
})).returns(true);
stub([{id: 2}, {id: 1}]);
```



```
var arrayContainsObjectMatch = expectation =>
    sinon.match.some(sinon.match(expectation));
```

Custom Matcher Function



#### Custom Matcher Function

```
var arrayContainsObjectMatch = expectation =>
sinon.match.some(sinon.match(expectation));
```



#### Custom Matcher Function

```
var arrayContainsObjectMatch = expectation =>
    sinon.match.some(sinon.match(expectation));
```



## Object Contains Array

```
const stub = sinon.stub();
stub.withArgs(arrayContainsObjectMatch({
   id: 1
})).returns(true);
stub([{id: 2}, {id: 1}]);
```



## Check Every Object in Array

```
const arrayEveryObjectMatch = expectation =>
sinon.match.every(sinon.match(expectation));
const stub2 = sinon.stub();
stub2.withArgs(arrayEveryObjectMatch({
   shouldDelete: true
})).returns('Deleted');
stub2([ {id: 2, shouldDelete: true}]);
```



```
const stub = sinon.stub();
stub.withArgs(sinon.match(1517)).returns(true);
```



## Custom Matchers

```
(value) => {
...
}
```



## Using a Custom Matcher

```
sinon.match((value) => {
          ...
});
```



## Creating a Custom Matcher

```
const isCharlesSpurgeon = value => value.name === 'Charles
Spurgeon';

const stub = sinon.stub();
stub.withArgs(sinon.match(isCharlesSpurgeon))
    .returns('Hi Charles!');

stub({name: 'Charles Spurgeon'});
```



```
const isDivisible = expectation =>
    value =>
    value % expectation === 0;

const stub = sinon.stub();
stub.withArgs(sinon.match(isDivisible(2))).returns('even');
```

Is Divisible Matcher



# Matchers give <u>you</u> power over your tests





Faking out XHR and timers

