

Matching Strings

```
assertThat(s, is(@"FooBar"));
```

```
assertThat(s, startsWith(@"Foo"));
```

```
assertThat(s, endsWith(@"Bar"));
```

```
assertThat(s, containsString(@"oo"));
```

```
assertThat(s,
equalToIgnoringCase(@"foobar"));
```

```
assertThat(@" X \n Y \t\t Z \n",
equalToIgnoringWhiteSpace(@"X Y Z"));
```

Given NSString *s = @"FooBar";

is â€œ match the complete string

startsWith â€œ match the beginning of a string

endsWith â€œ match the end of a string

containsString â€œ match part of the string

equalTo â€œ match the complete string

equalToIgnoringCase â€œ match the complete

string but ignore case

equalToIgnoringWhiteSpace â€œ match the

complete string but ignore extra whitespace

(new line, tab, or double spaces)

Matching Dictionaries

```
NSDictionary *d = [NSDictionary
dictionaryWithObjectsAndKeys: @"valA",
@"keyA", @"valB", @"keyB", @"valC",
@"keyC", nil];
```

```
assertThat(d, hasCountOf(3));
```

```
assertThat(d, isEmpty());
```

```
assertThat(d, hasKey(@"keyA"));
```

```
assertThat(d, isNot(hasKey(@"keyX")));
```

```
assertThat(d, hasValue(@"valA"));
```

```
assertThat(d, hasEntry(@"keyA", @"valA"));
```

```
assertThat(d, hasEntries(@"keyA", @"valA",
@"keyC", @"valC", nil));
```

hasKey â€œ match a key

hasValue â€œ match a value

hasEntry â€œ match a key-value pair

hasEntries â€œ match a list of k-v pairs

Matcher Error Messages

```
NSString *s = @"bar";
```

```
assertThat(s, is(@"foo")); //Expected "foo", but
was "bar"
```

```
assertThat(s, describedAs(@"doh! this should
be 'foo", equalTo(@"foo", nil)); //Expected
doh! this should be 'foo', but was "bar"
```

```
assertThat(s, describedAs(@"doh! this should
be foo, %0, %1", equalTo(@"foo", @"baz",
[NSNumber numberWithInt:42], nil));
//Expected doh! this should be foo, "baz",
<42>, but was "bar"
```

Combining Matchers

```
assertThat(s, allOf(startsWith(@"Foo"),
endsWith(@"Bar"), nil));
```

```
assertThat(s, anyOf(startsWith(@"Foo"),
startsWith(@"Bar"), nil));
```

```
assertThat(s, anyOf(endsWith(@"Foo"),
endsWith(@"Bar"), nil));
```

allOf â€œ AND together all matchers

anyOf â€œ OR together all matches

The list of matchers must be nil terminated.

Matching Nil

```
assertThat(s, notNilValue());
```

```
assertThat(o, nilValue());
```

Given NSObject *o = nil;

nilValue() â€œ stands in for nil

notNilValue() â€œ stands in for !nil

Matching Numbers

```
assertThatInt(5, equalToInt(5));
```

```
assertThatFloat(3.14, equalToFloat(3.14f));
```

```
assertThatBool( false, equalToBool(NO) );
```

```
NSNumber *f = [NSNumber
numberWithFloat:3.14f];
```

```
assertThat(f, closeTo(3.0f, 0.25f));
```

```
assertThat(f, lessThan([NSNumber
numberWithInt:4]));
```

Matching Numbers (cont)

```
assertThat(f, greaterThan([NSNumber
numberWithInt:3]));
```

closeTo â€œ match a number with a target number plus or minus a delta (both params are double)

lessThan â€œ match a number less than the given number (param is NSNumber), also lessThanOrEqual

hasProperty Matcher

```
Person *p = [Person
personWithFirstName:@"Joe"
andLastName:@"Doe"];
```

```
assertThat(p, hasProperty(@"firstName",
@"Joe"));
```

```
NSArray *a = [NSArray arrayWithObjects:
[Person personWithFirstName:@"Joe"
andLastName:@"Doe"], [Person
personWithFirstName:@"Joe"
andLastName:@"Smith"], [Person
personWithFirstName:@"Jane"
andLastName:@"Allen"], nil];
```

```
assertThat(a, contains(
hasProperty(@"firstName", @"Joe"),
hasProperty(@"firstName", @"Joe"),
hasProperty(@"firstName", @"Jane"), nil));
```

Any method, without arguments, that returns an object

Invert Matcher

```
assertThat(s, isNot(@"foo"));
```

```
assertThat(s, isNot(endsWith(@"Baz"));
```

```
assertThat(s, isNot(allOf(startsWith(@"Baz"),
endsWith(@"Baz"), nil));
```

```
assertThat(s, isNot(anyOf(startsWith(@"Baz"),
startsWith(@"Baz"), nil));
```

isNot â€œ negate the matcher

Matching Classes

```
assertThat(s, instanceof([NSString class]));
```

instanceOf â€œ match the class



By **appdeft**

cheatography.com/appdeft/

Published 23rd September, 2013.

Last updated 23rd September, 2013.

Page 1 of 2.

Sponsored by **Readability-Score.com**

Measure your website readability!

<https://readability-score.com>

Matching Arrays

```
NSArray *a = [NSArray array];
```

```
assertThat(a, isEmpty());
```

```
assertThat(a, hasCountOf(0));
```

```
NSArray *a = [NSArray arrayWithObjects:@"a", @"b", @"c",  
nil];
```

```
assertThat(a, hasItem(@"a"));
```

```
assertThat(a, isNot(hasItem(@"X"));
```

```
assertThat(a, hasItem(equalIgnoringCase(@"A"));
```

```
NSArray *a = [NSArray arrayWithObjects: [NSNumber  
 numberWithInt:2], [NSNumber numberWithInt:3], [NSNumber  
 numberWithInt:5], nil];
```

```
assertThat(a, hasItem(equalToInt(2)));
```

```
assertThat(a, isNot(hasItem(equalToInt(13))));
```

```
assertThat(a, contains(equalToInt(2), equalToInt(3),  
equalToInt(5), nil));
```

```
NSArray *a = [NSArray arrayWithObjects:@"a", @"b", @"c",  
nil];
```

```
assertThat(a, hasItems(@"b", @"a", nil));
```

```
assertThat(a, contains(@"a", @"b", @"c", nil));
```

```
assertThat(a, containsInAnyOrder(@"c", @"b", @"a", nil));
```

```
assertThat([a componentsJoinedByString:@","], is(@"a,b,c"));
```

hasItem â€œ match if given item appears in the list

hasItems â€œ match if all given items appear in the list (in any order)

contains â€œ match exactly the entire array

containsInAnyOrder â€œ match entire array, but in any order

hasCountOf â€œ match the size of the array

empty â€œ match an empty array



By **appdeft**

cheatography.com/appdeft/

Published 23rd September, 2013.

Last updated 23rd September, 2013.

Page 2 of 2.

Sponsored by **Readability-Score.com**

Measure your website readability!

<https://readability-score.com>