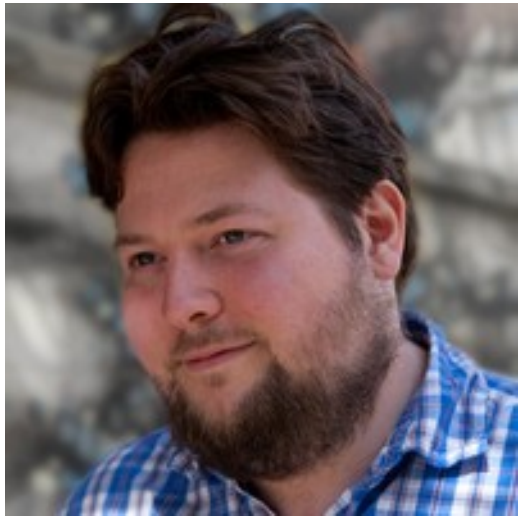
Reflection



Richard Warburton

@richardwarburto | www.insightfullogic.com

Class Literals

What types are Reflectable?

Reflecting Generics

Class Literals

A Pattern for Generic instantiation

Reflecting Types

Types that are reified can be reflected

Reifiable Types

Primitives

```
int, long
```

Non Parameterized Class or Interface

```
String, ActionListener
```

All type arguments are unbounded Wildcards

```
List<?>, Map<?, ?>
```

Raw Types

```
List, Map
```

Arrays of reifiable components

```
int[][], List<?>[]
```

Non Reifiable Types

Type Variables

```
T
```

Parameterized Type with Parameters

```
ArrayList<String>, Map<Integer, String>
```

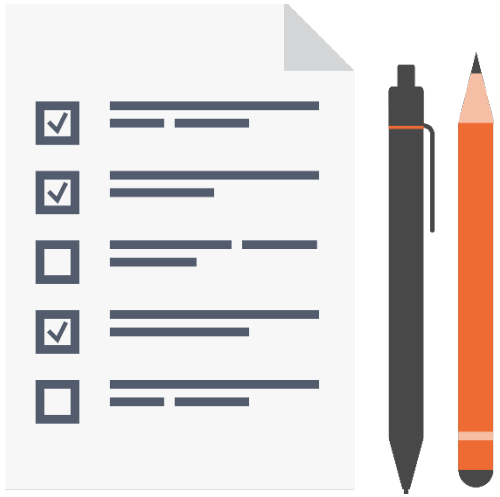
Parameterized Type with Bounds

```
List<? extends Number>, Consumer<? super String>
```

Reflecting Generic Information

Advanced but Cool!

Summary



- Expanded on Implementation
- Reflectable types are Reifiable Types
- Class Literals and Reflection

Reflection

Richard Warburton
@richardwarburto | www.insightfullogic.com

Reflection is an incredibly powerful and commonly used tool in Java

It allows us to see an image of what your application is doing at runtime.

This module is about how reflection and generics interplay, not about reflection at large.

Class Literals

What types are Reflectable?

Reflecting Generics

Class Literals

A Pattern for Generic instantiation

Reflecting Types

Types that are reified can be reflected

Reifiable means the type information is completely represented at runtime.

Reifiable Types

Primitives

```
int, long
```

Non Parameterized Class or Interface

```
String, ActionListener
```

All type arguments are unbounded Wildcards

```
List<?>, Map<?, ?>
```

Raw Types

```
List, Map
```

Arrays of reifiable components

```
int[][], List<?>[]
```

Non Reifiable Types

Type Variables

```
T
```

Parameterized Type with Parameters

```
ArrayList<String>, Map<Integer, String>
```

Parameterized Type with Bounds

```
List<? extends Number>, Consumer<? super String>
```

Reflecting Generic Information

Advanced but Cool!

Summary

- Expanded on Implementation
- Reflectable types are Reifiable Types
- Class Literals and Reflection