Raw Types and Compatibility

Richard Warburton

@richardwarburto | www.insightfullogic.com

Migration Compatibility

You can upgrade to a new Java version and your code doesn't “break”.

Binary Compatibility for Generics

You can replace a legacy class file with a generic class file without changing or recompiling any client code.

Raw Types

What is Erasure

The Implications of Erasure

Arrays and Reification

Raw Types

Erasure

How Generics get implemented

Translation at compile time, not runtime.

How Are Generics Implemented?

Erase Type Parameters

Add casts

Add Bridge Methods

Erasure

List<String>, List<Integer>, List<List<Integer>> → List

List<String>[] → List[]

T without bounds → Object

T extends Foo → Foo

Implications of Erasure

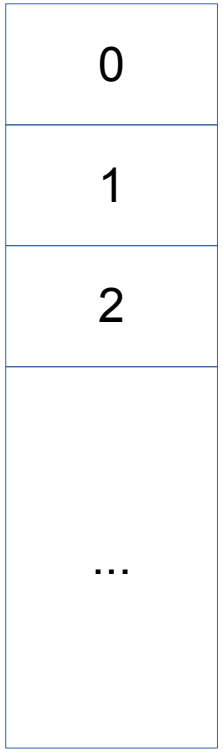
Downsides and Tradeoffs

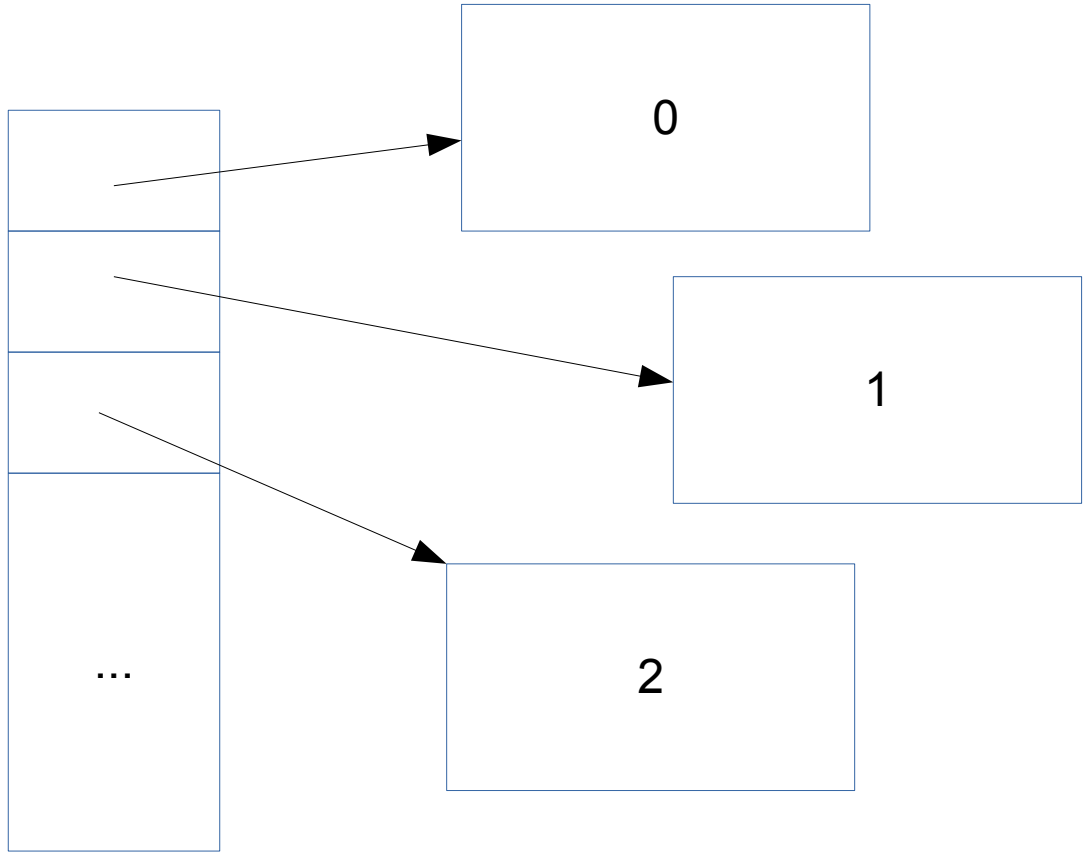
Overloads

Checking the type of an Instance

Performance

Flatness: Int[] vs. Integer[]





Size: int vs. Integer

int

4 bytes

Nothing else for arrays!

java.lang.Integer

4 bytes for int value

Object Header: 8-16 bytes

Alignment: allocate a multiple of 8 bytes

Pointer for arrays: 4 or 8 bytes

Worst case: 32bytes (8x fatter)

Reifiable Types and Arrays

Summary

Summary

There's a bad and ugly side of generics as well as a good one

Erasure has significant downsides and limitations

Raw Types achieve compatibility, but are error prone