Collections with Uniqueness: Sets

Richard Warburton
JAVA CHAMPION, AUTHOR AND PROGRAMMER

@richardwarburto www.monotonic.co.uk

Sets are collections of distinct elements. There are no duplicates

Outline

Set Features

The Why and How of Sets

Hashcode and Equals

Understanding the contract behind HashSet

Set Implementations

Performance tradeoffs and features

Set Features

Hashcode and Equals


Signatures
graphs, and hereby grants the
graphs for which the Client pays.
provide reasonable
Photographer's Signature

```
object.equals(other)
```

➔

```
object.hashCode() == other.hashCode()
```

Hashcode / Equals Contract

One way implication

Equality

It can be reference based or value based. Reference based just needs to inherit equals from Object. Value based requires a custom equals method.

```
result = 31 * result + obj.hashCode();


// Arrays

Arrays.hashCode()


// Primitives (Java 8+)

Long.hashCode(longValue)


// Old Primitives

(int) (l ^ (l >>> 32))
Float.floatToIntBits(f);
```

- Combine hashcode information from each field

- IDE Can auto-generate

- Objects.hash() (Java 7+)

- ALWAYS use the same fields as equals()

Set Implementations

HashSet

Based upon HashMap

Uses hashcode() and looks up location

Good General Purpose Implementation

Use by default

TreeSet

Based Upon TreeMap

Red/Black binary tree with defined sort order

Provides Extra Features

Implements SortedSet and NavigableSet

Conclusion

Summary

Sets are a commonly used collection

Different implementations for different purposes

Remember to get the hashcode/equals contract correct


LINEX