Java Generics

Parametric Polymorphism

OVERVIEW OF THE COURSE

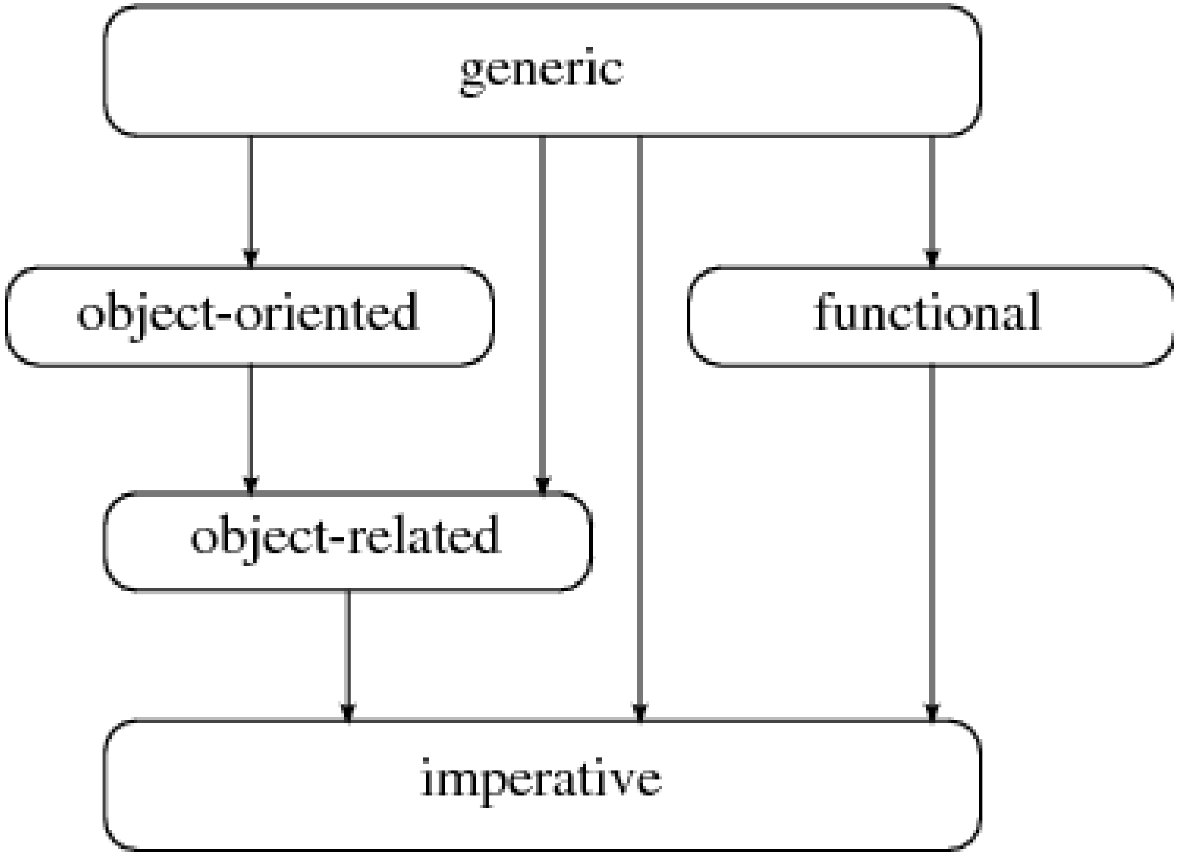
DR. ERIC CHOU

IEEE SENIOR MEMBER

What is Generic Programming?

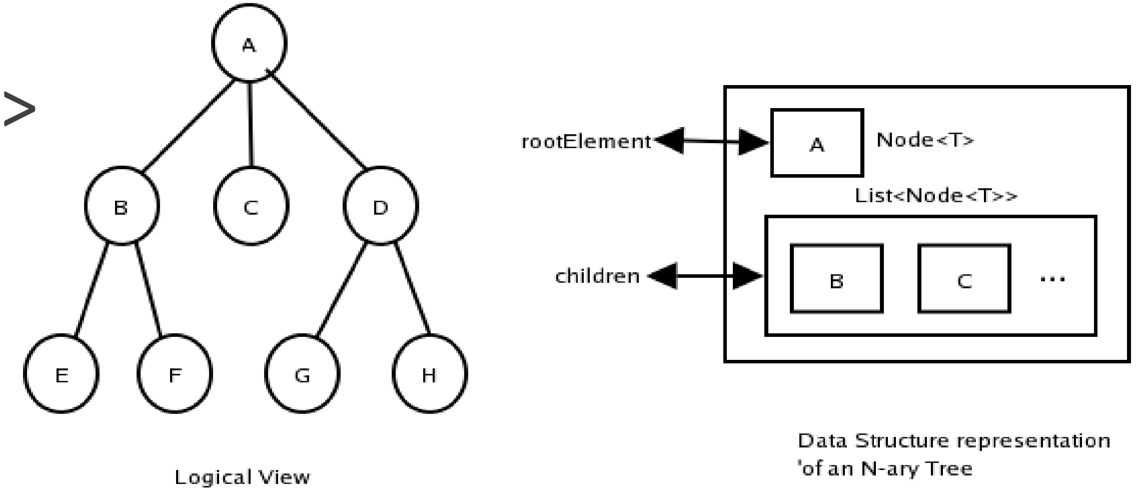
- Generic programming is another way of overloading your functions (methods) for different data type. And, it can do more than overloading.
- Overloading can only be applied to methods.
- Generic Programming can be applied to both data field and methods. Generic Data Container, Generic Library Functions, Generic Polymorphic methods.
- Generic Programming further expands Object-Oriented Programming.

Generic Programming

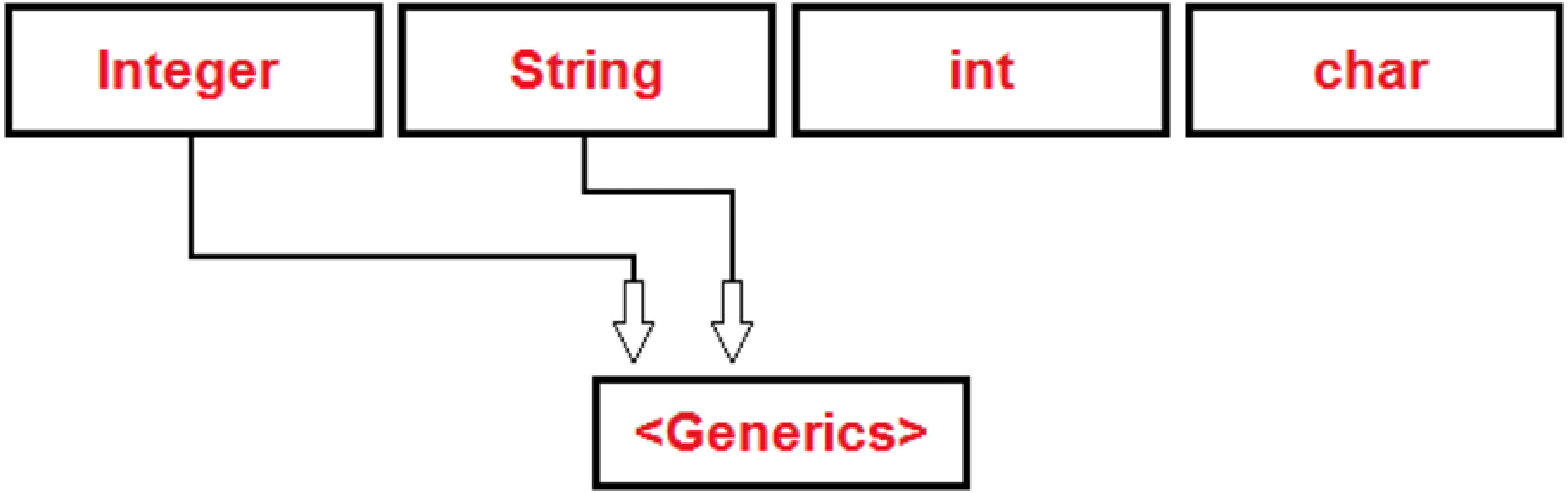


Generic Data Containers

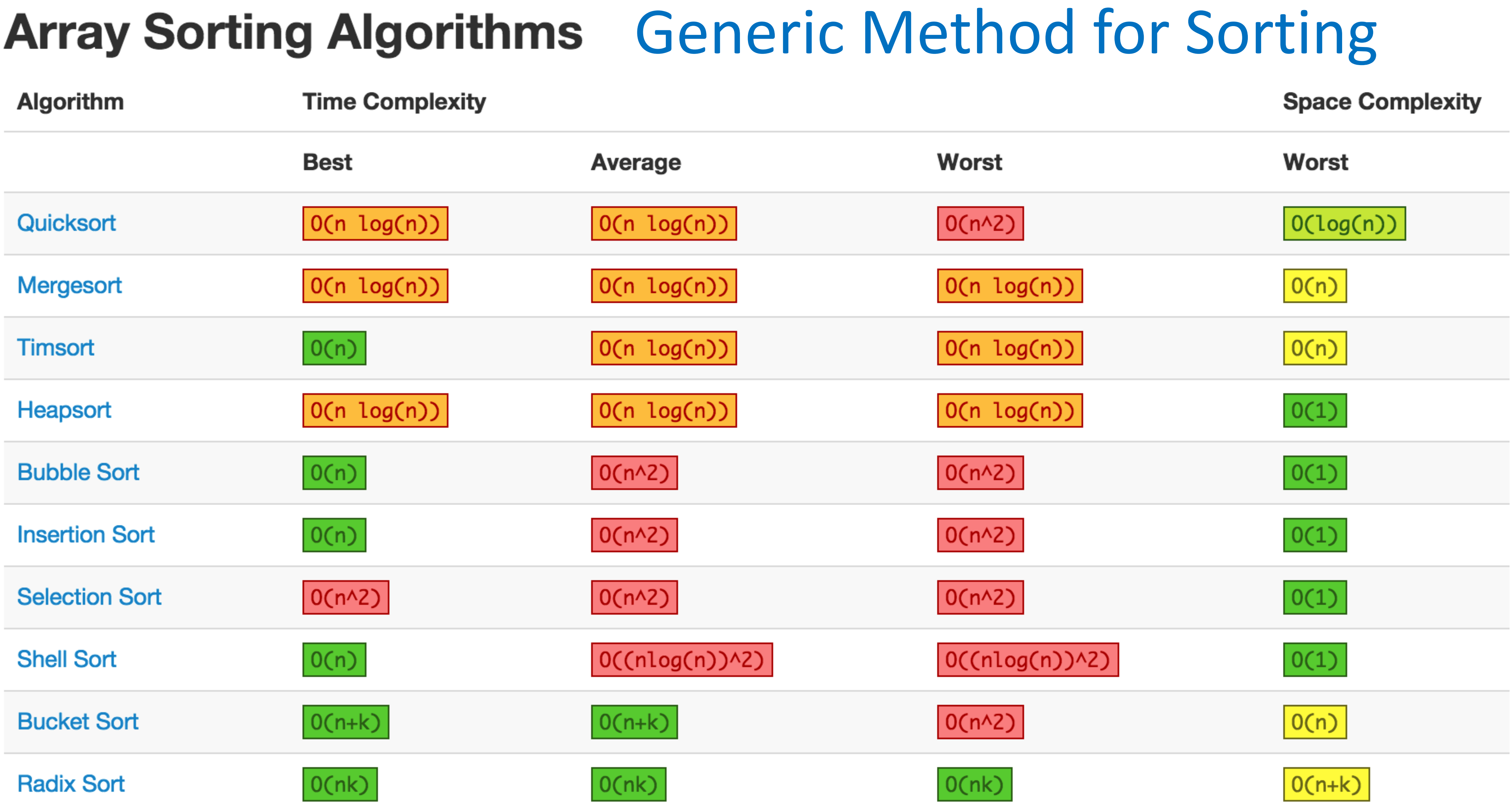
- Abstract Data Types (such as, Queue, Stack, Map, Set) tend to be Generic.
- Generic Programming can be realized by inheritance and polymorphism, or parametric polymorphism (Generic language structure)
- Object versus Type Variable<T>



Generic Methods



Generic Method for Sorting

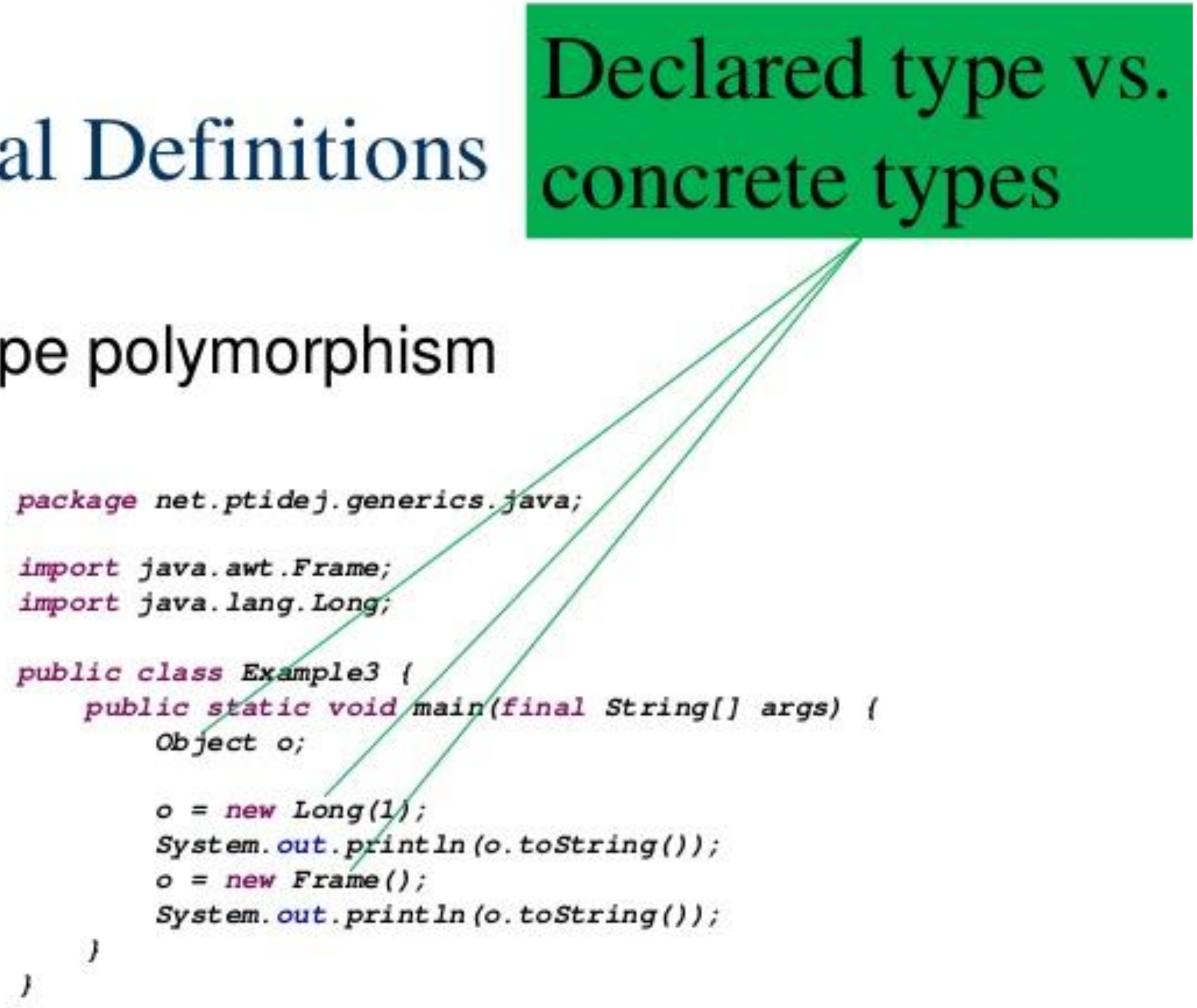


Array Sorting Algorithms

Algorithm	Time Complexity			Space Complexity
	Best	Average	Worst	Worst
Quicksort	O(n log(n))	O(n log(n))	O(n^2)	O(log(n))
Mergesort	O(n log(n))	O(n log(n))	O(n log(n))	O(n)
Timsort	O(n)	O(n log(n))	O(n log(n))	O(n)
Heapsort	O(n log(n))	O(n log(n))	O(n log(n))	O(1)
Bubble Sort	O(n)	O(n^2)	O(n^2)	O(1)
Insertion Sort	O(n)	O(n^2)	O(n^2)	O(1)
Selection Sort	O(n^2)	O(n^2)	O(n^2)	O(1)
Shell Sort	O(n)	O((nlog(n))^2)	O((nlog(n))^2)	O(1)
Bucket Sort	O(n+k)	O(n+k)	O(n^2)	O(n)
Radix Sort	O(nk)	O(nk)	O(nk)	O(n+k)

General Definitions

- Subtype polymorphism



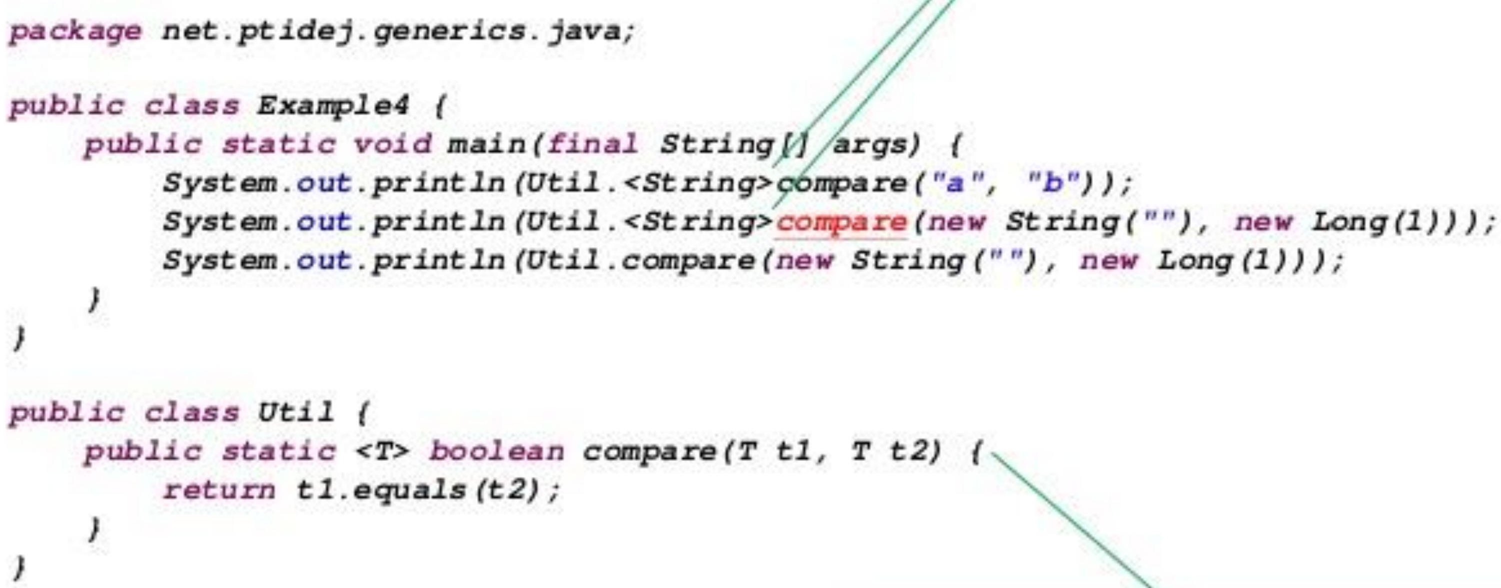
Declared type vs. concrete types

```java
package net.ptidej.generics.java;

import java.awt.Frame;
import java.lang.Long;

public class Example3 {
    public static void main(final String[] args) {
        Object o;

        o = new Long(1);
        System.out.println(o.toString());
        o = new Frame();
        System.out.println(o.toString());
    }
}
```

General Definitions

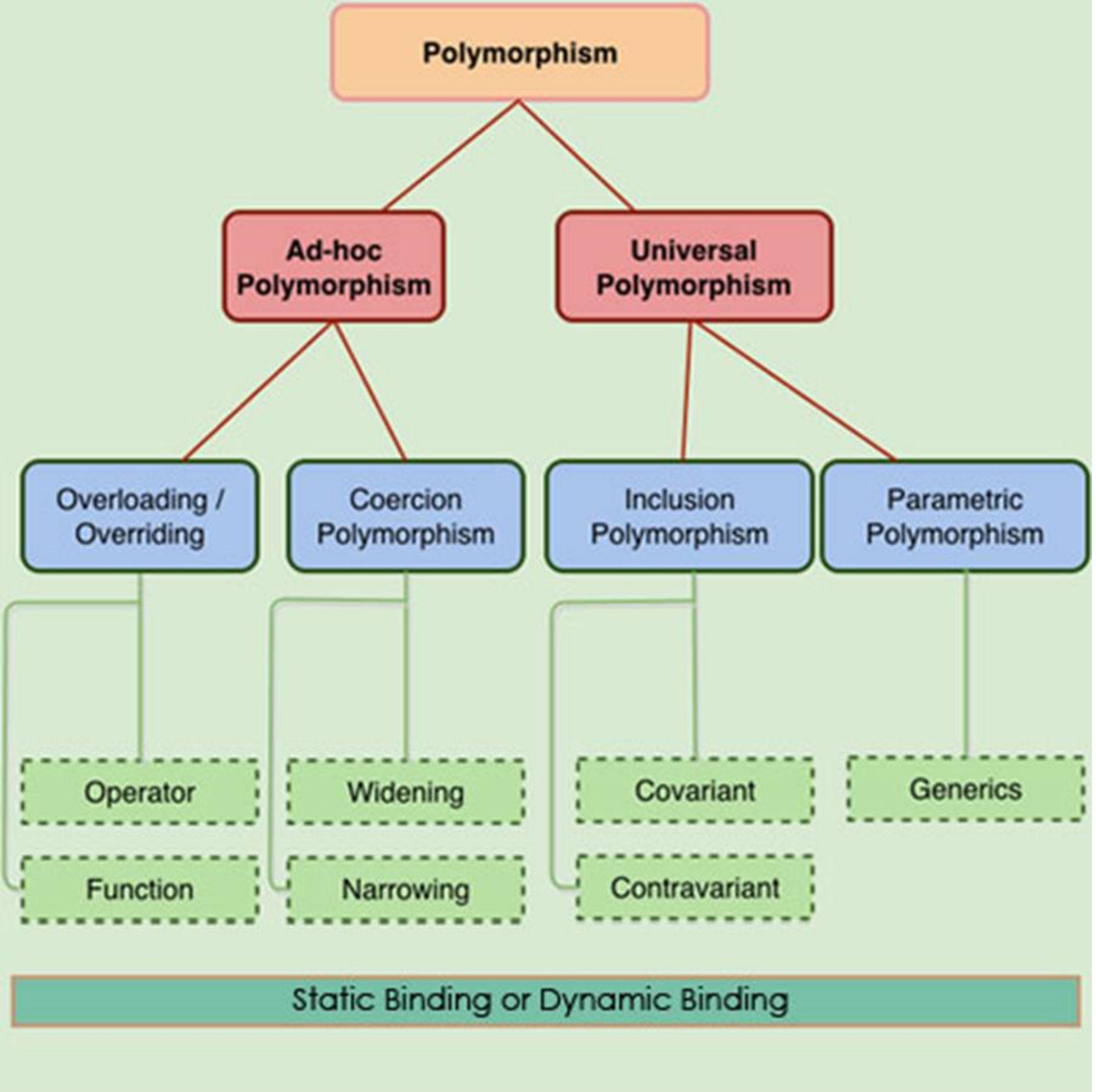
Explicit calls

- Parametric polymorphism



```
package net.ptidej.generics.java;

public class Example4 {
    public static void main(final String[] args) {
        System.out.println(Util.<String>compare("a", "b"));
        System.out.println(Util.<String>compare(new String(""), new Long(1)));
        System.out.println(Util.compare(new String(""), new Long(1)));
    }
}

public class Util {
    public static <T> boolean compare(T t1, T t2) {
        return t1.equals(t2);
    }
}
```

Generic method



Polymorphism

Ad-hoc Polymorphism

Universal Polymorphism

Overloading / Overriding

Coercion Polymorphism

Inclusion Polymorphism

Parametric Polymorphism

Operator

Function

Widening

Narrowing

Covariant

Contravariant

Generics

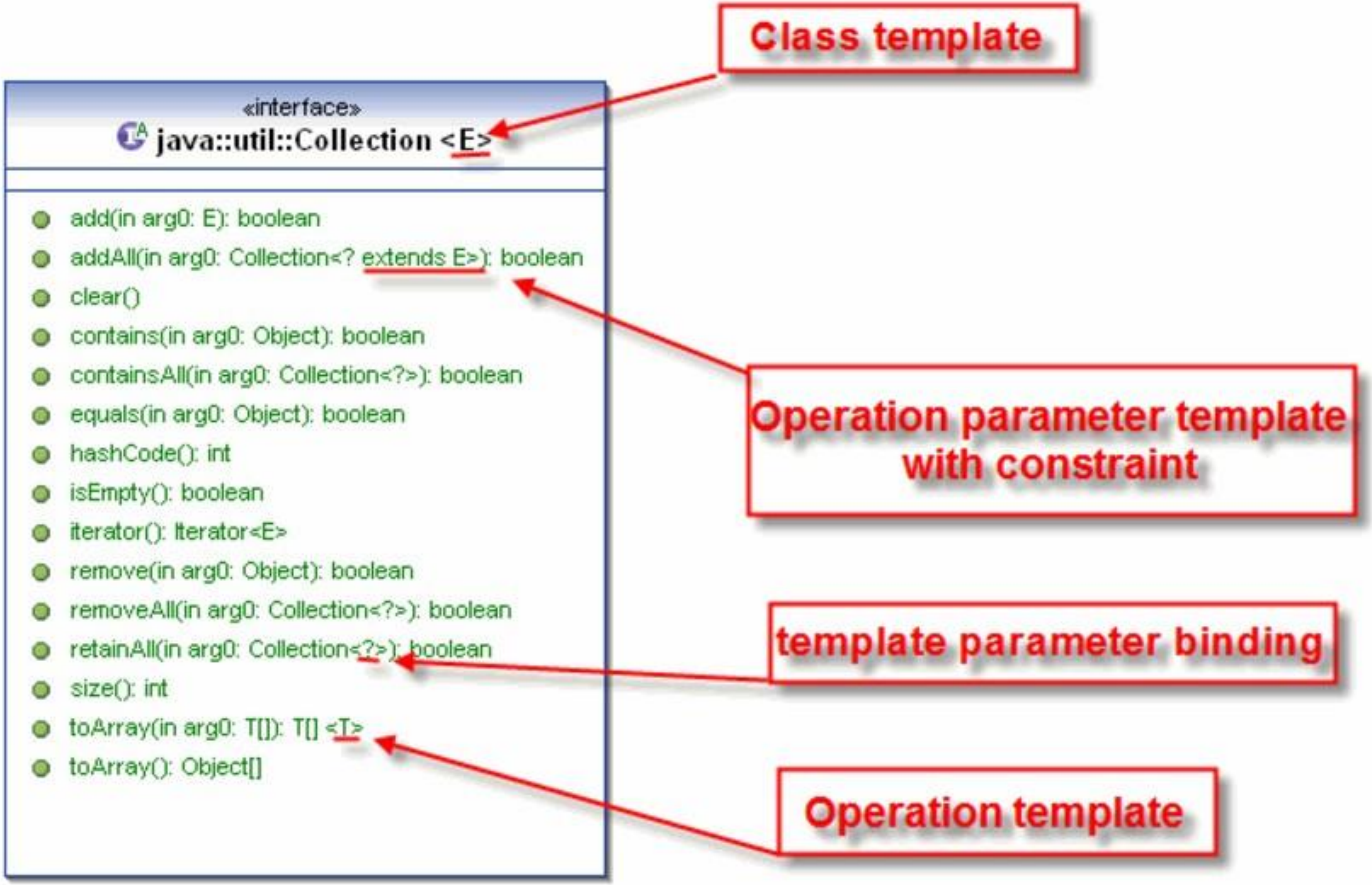
Static Binding or Dynamic Binding

Types of Polymorphism: (By assignment time)

- **Ad hoc Polymorphism:** polymorphism assigned by programmer at any time in design time.
- **Universal Polymorphism:** polymorphism assigned by language definition.

Types of Polymorphism: (By purpose)

- **Overloading/Overriding:** Re-definition of functions
- **Coercion Polymorphism:** Data Casting
- **Inclusion Polymorphism:** Sub-type polymorphism or polymorphism by inheritance
- **Parametric Polymorphism:** Generics, Generic data type



«interface»
java::util::Collection <E>

- add(in arg0: E): boolean
- addAll(in arg0: Collection<? extends E>): boolean
- clear()
- contains(in arg0: Object): boolean
- containsAll(in arg0: Collection<?>): boolean
- equals(in arg0: Object): boolean
- hashCode(): int
- isEmpty(): boolean
- iterator(): Iterator<E>
- remove(in arg0: Object): boolean
- removeAll(in arg0: Collection<?>): boolean
- retainAll(in arg0: Collection<?>): boolean
- size(): int
- toArray(in arg0: T[]): T[] <T>
- toArray(): Object[]

Class template

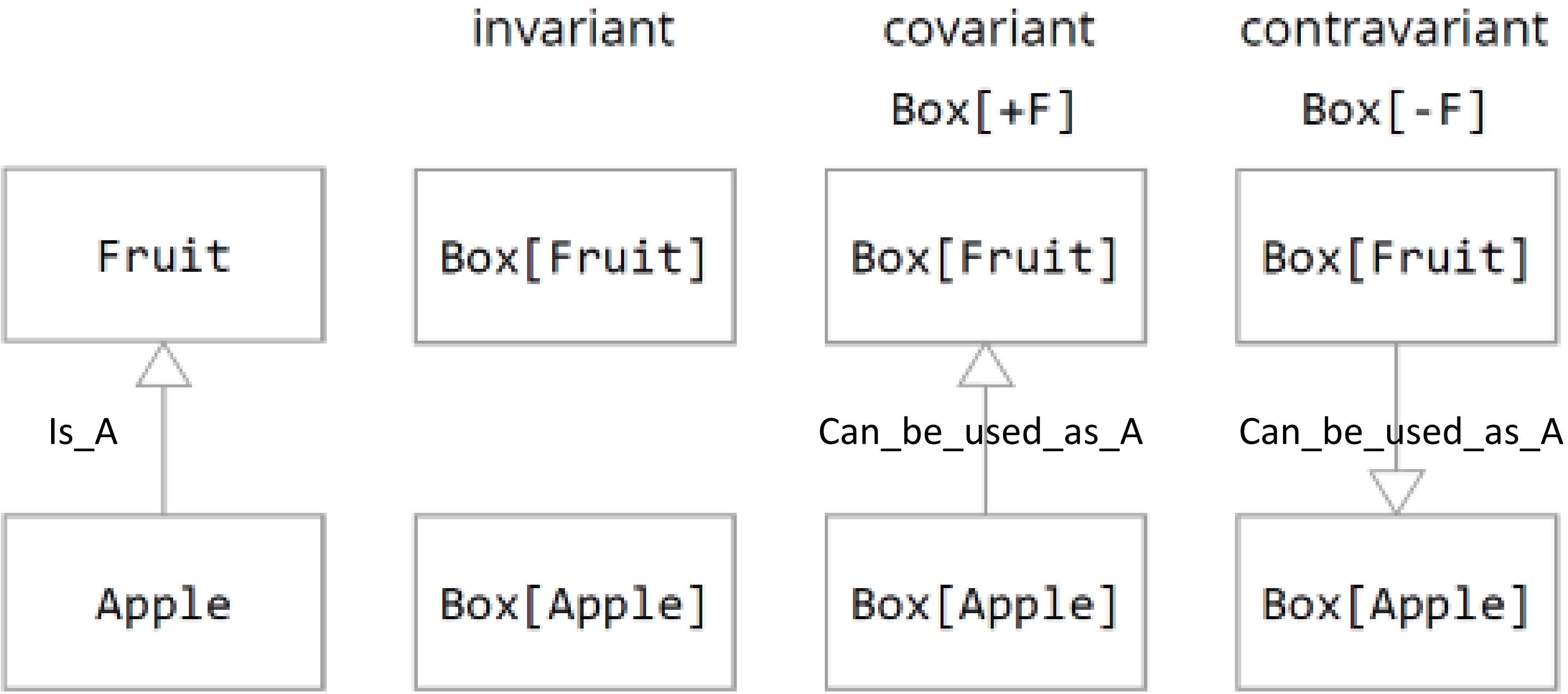
Operation parameter template with constraint

template parameter binding

Operation template

Scala's Covariant and Contravariant

(Future Expansion)



invariant

covariant

Box[+F]

contravariant

Box[-F]

Fruit

Is_A

Apple

Box[Fruit]

Box[Apple]

Box[Fruit]

Can_be_used_as_A

Box[Apple]

Box[Fruit]

Can_be_used_as_A

Box[Apple]