Java is a trademark of Sun Microsystems, Inc.

JavaOne℠

Upgrading OSGi™

BJ Hargrave
Senior Technical Staff Member
IBM

Peter Kriens
Technical Director
OSGi

OSGi Archeology

JDK 1.0	1996	
JDK 1.1	1997	
Personal Java	1998	Connected Alliance
J2SE 1.2	1999	OSGi Alliance
J2SE 1.3	2000	R1
J2ME CDC/FP 1.0	2001	R2
J2SE 1.4	2002	
	2003	R3
J2SE 5.0	2004	
J2ME CDC/FP 1.1	2005	R4
JavaSE 6	2006	
	2007	R4.1
	2008	
	2009	R4.2

OSGi Archeology

OSGi Archeology

Dictionary

OSGi Archeology

Dictionary

Arrays[]

OSGi Archeology

Dictionary

Arrays[]

Checked Exceptions

OSGi Archeology

Dictionary

Null return for Empty arrays

Arrays[]

Checked Exceptions

OSGi Archeology

Busy Bundle Context

Dictionary

Null return for Empty arrays

Arrays[]

Checked Exceptions

OSGi Archeology

Busy Bundle Context

Dictionary

Null return for Empty arrays

Arrays[]

Java 1.4 Level

Checked Exceptions

OSGi Archeology

OSGi Archeology

Generics?

OSGi Archeology

Generics?

Collections?

OSGi Archeology

Generics?

Enums?

Collections?

OSGi Archeology

Generics?

Enums?

Collections?

Refactor Types?

OSGi Archeology

Generics?

Annotations?

Enums?

Collections?

Refactor Types?

OSGi Archeology

Generics?

Annotations?

...?

Enums?

Collections?

Refactor Types?


OSGi Cool???


OSGi Cool???
SONY


OSGi Cool???

Guidelines

- Interoperability
- Simplify
- Increase Type Safety
- Improve “Looks”
- Java 5 Features

SCSI

DVD Drive

ADB

Firewire

PowerPC

68000

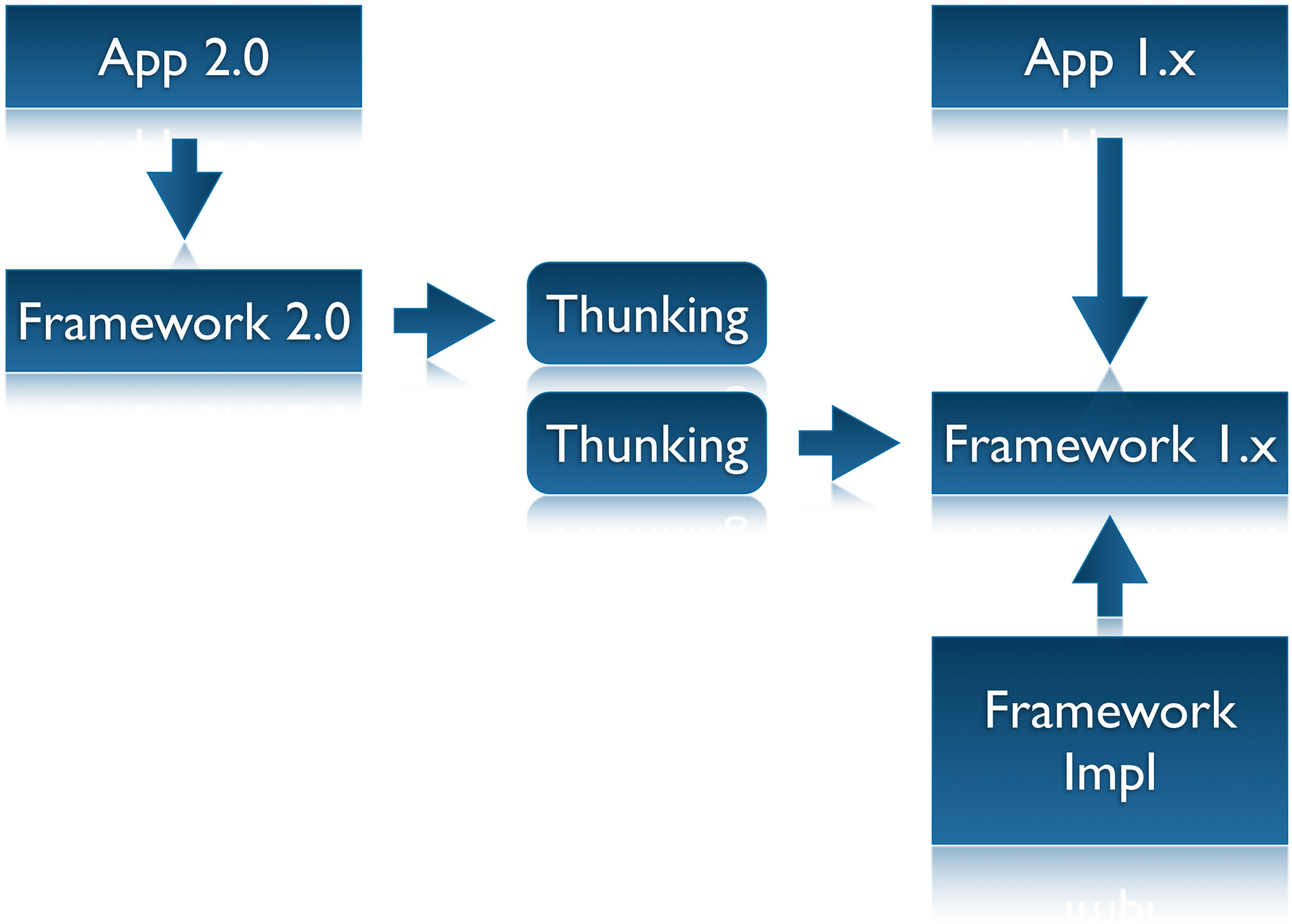
I'm a PC.

I'm a Mac.

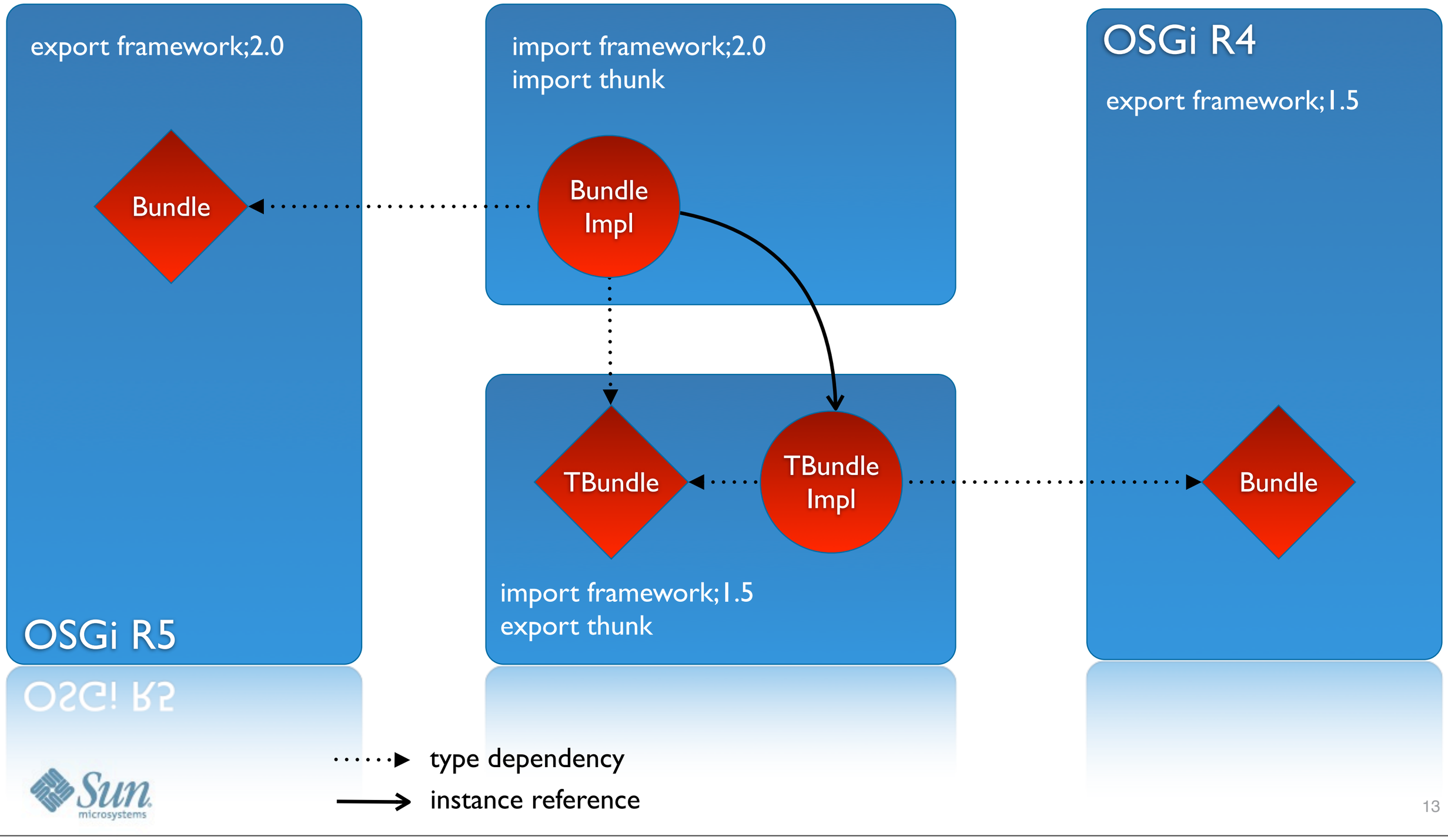

自転車を除く

Thunking Layer

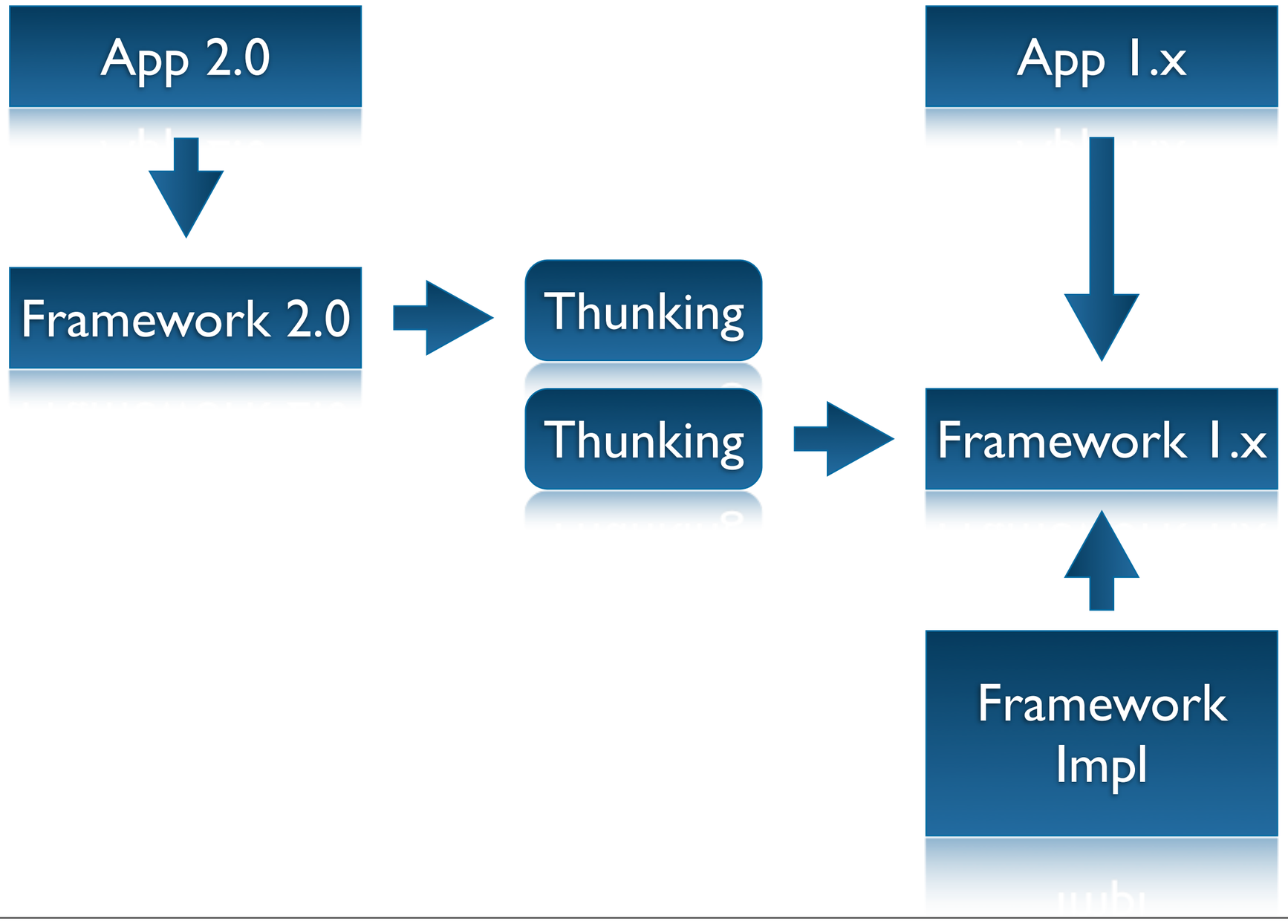
Thunking Layer



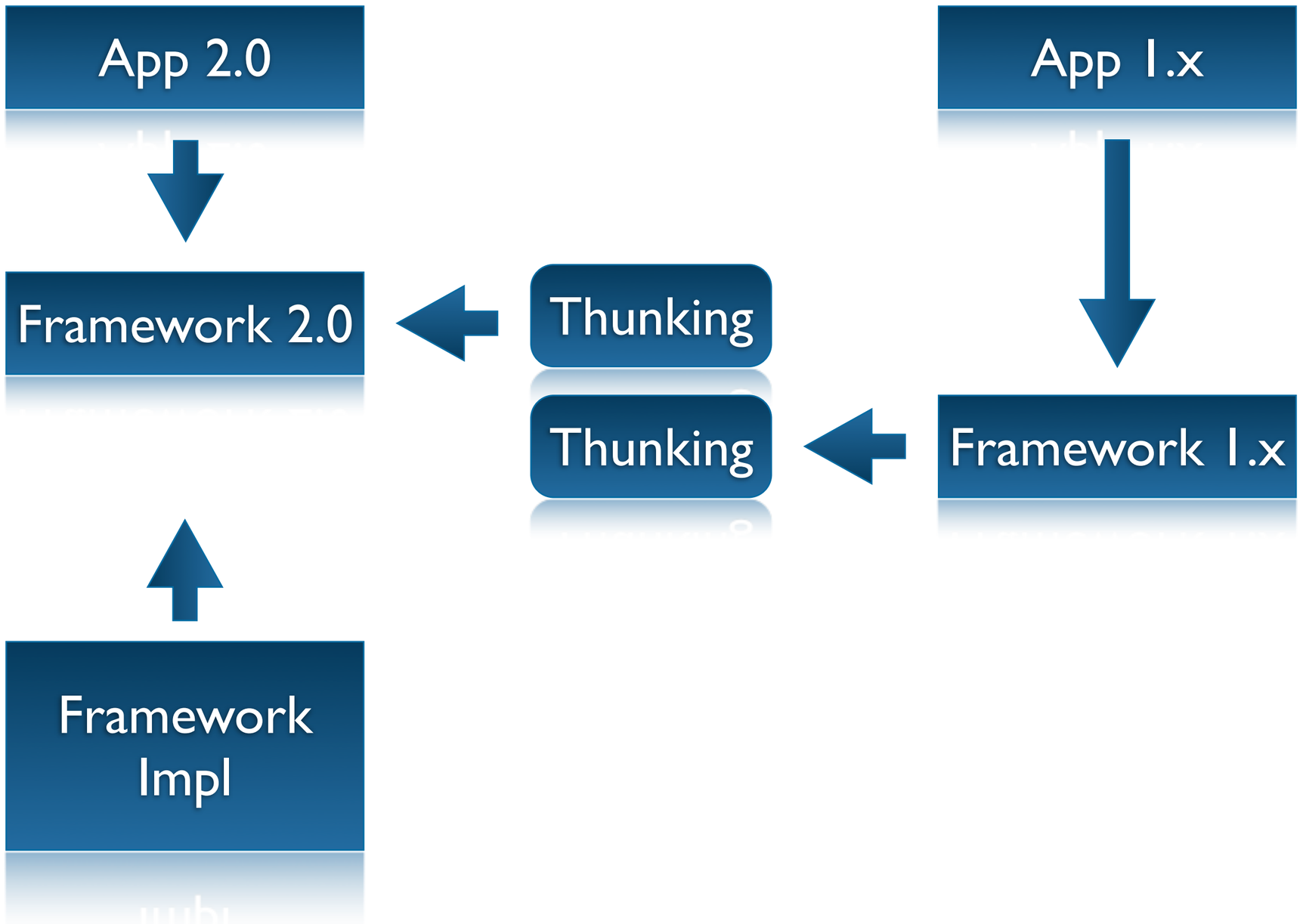
Thunking Layer



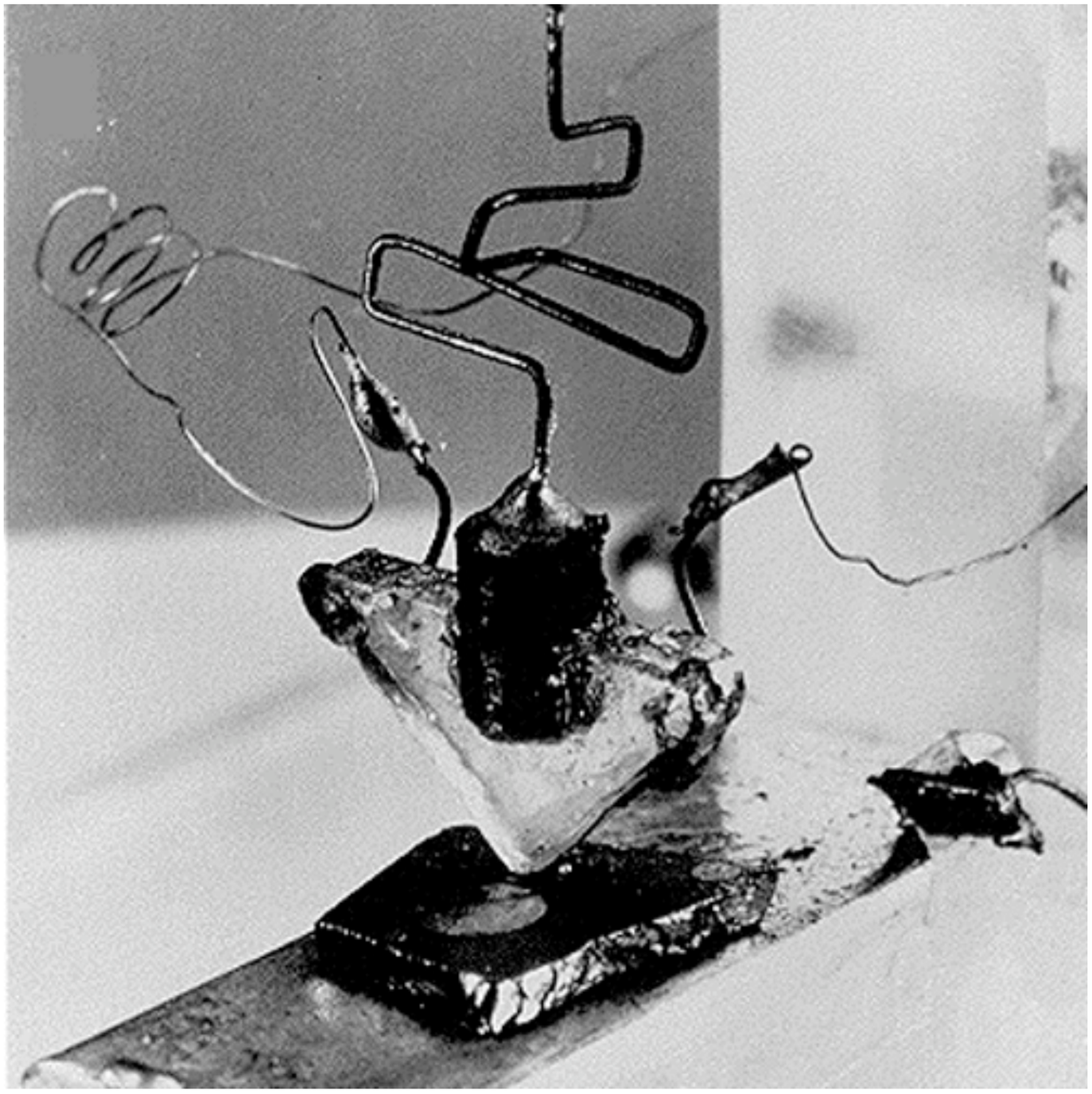
Thunking Layer



Thunking Layer



Thunking Layer



Thunking Layer

```
> exports org.osgi.framework

org.osgi.framework-1.5.0 exported by 0        ACTIVE      org.eclipse.osgi
 imported by 1        ACTIVE      org.osgi.wrapped.framework
 imported by 3        ACTIVE      org.osgi.impl.service.log
 imported by 4        ACTIVE      org.osgi.impl.bundle.console
org.osgi.framework.launch-1.0.0 exported by 0        ACTIVE      org.eclipse.osgi
org.osgi.framework.hooks.service-1.0.0 exported by 0        ACTIVE      org.eclipse.osgi
org.osgi.framework-0.0.0 exported by 0        ACTIVE      org.eclipse.osgi
 imported by 1        ACTIVE      org.osgi.wrapped.framework
 imported by 3        ACTIVE      org.osgi.impl.service.log
 imported by 4        ACTIVE      org.osgi.impl.bundle.console
org.osgi.framework.launch-0.0.0 exported by 0        ACTIVE      org.eclipse.osgi
org.osgi.framework.hooks.service-2.0.0 exported by 2        ACTIVE      org.osgi.thunk.framework
org.osgi.framework-2.0.0 exported by 2        ACTIVE      org.osgi.thunk.framework
 imported by 5        ACTIVE      test.bundle.alpha
org.osgi.framework.bundle-2.0.0 exported by 2        ACTIVE      org.osgi.thunk.framework
 imported by 5        ACTIVE      test.bundle.alpha
org.osgi.framework.launch-2.0.0 exported by 2        ACTIVE      org.osgi.thunk.framework

>
```

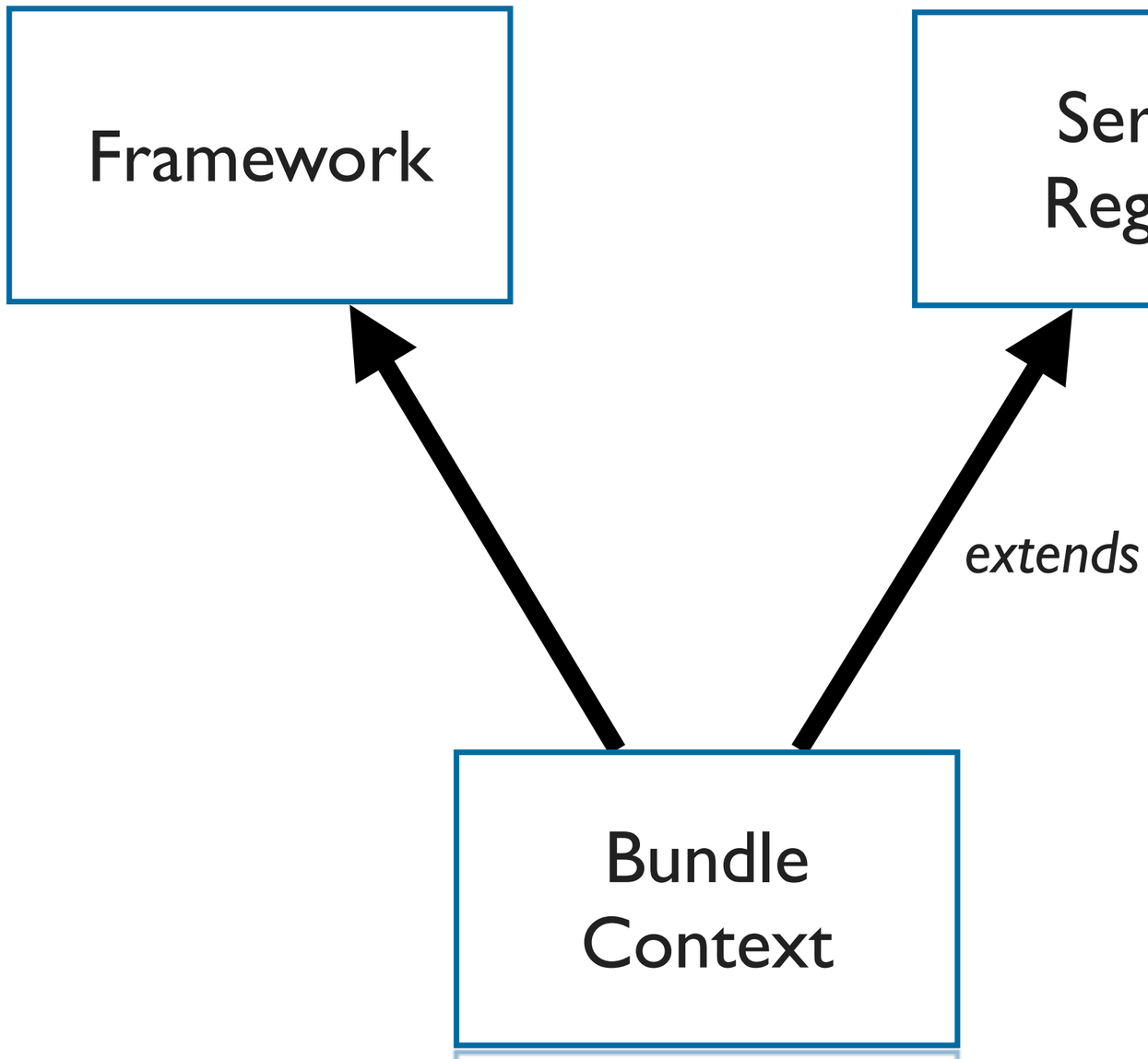
Thunking Layer

```
> exports org.osgi.framework

org.osgi.framework-1.5.0 exported by 0        ACTIVE      org.eclipse.osgi
 imported by 1         ACTIVE      org.osgi.wrapped.framework
 imported by 3         ACTIVE      org.osgi.impl.service.log
 imported by 4         ACTIVE      org.osgi.impl.bundle.console
org.osgi.framework.launch-1.0.0 exported by 0        ACTIVE      org.eclipse.osgi
org.osgi.framework.hooks.service-1.0.0 exported by 0        ACTIVE      org.eclipse.osgi
org.osgi.framework-0.0.0 exported by 0        ACTIVE      org.eclipse.osgi
 imported by 1         ACTIVE      org.osgi.wrapped.framework
 imported by 3         ACTIVE      org.osgi.impl.service.log
 imported by 4         ACTIVE      org.osgi.impl.bundle.console
org.osgi.framework.launch-0.0.0 exported by 0        ACTIVE      org.eclipse.osgi
org.osgi.framework.hooks.service-2.0.0 exported by 2        ACTIVE      org.osgi.thunk.framework
org.osgi.framework-2.0.0 exported by 2        ACTIVE      org.osgi.thunk.framework
 imported by 5         ACTIVE      test.bundle.alpha
org.osgi.framework.bundle-2.0.0 exported by 2        ACTIVE      org.osgi.thunk.framework
 imported by 5         ACTIVE      test.bundle.alpha
org.osgi.framework.launch-2.0.0 exported by 2        ACTIVE      org.osgi.thunk.framework

>
```

Layering

Bundle
Context

Layering



Enums versus ints

- Type Safe
- More Readable

Bundle State with Ints Old

```
public interface Bundle {
   int UNINSTALLED       = 0x00000001;
   int INSTALLED         = 0x00000002;
   int RESOLVED          = 0x00000004;
   int STARTING          = 0x00000008;
   int STOPPING          = 0x00000010;
   int ACTIVE            = 0x00000020;
   int getState();
   . . .
}


// Usage
void foo() {
   if ( (bundle.getState()
        & (Bundle.STARTING|Bundle.RESOLVED))!=0)
     bar();
}
```

Bundle State With Enums New

```
interface Bundle {
  public enum State {
    UNINSTALLED,
    INSTALLED,
    RESOLVED,
    STARTING,
    STOPPING,
    ACTIVE;
  }
  State getState();
  . . . }
// Usage
void foo() {
   EnumSet<State> mySet =
     EnumSet.of(State.STARTING,State.RESOLVED);
   if ( (mySet.contains(bundle.getState()) )
     bar();
}
```

Bundle Start Options with Enums, Attempt 1

```
interface Bundle {
  public enum StartOption {
     ACTIVATION_POLICY,
     TRANSIENT;

     public final static
          EnumSet<StartOption> NONE =
             EnumSet.noneOf(StartOption.class);
  }

  void start(EnumSet<StartOption> options);
  . . .
}
// Usage
void foo() {
   bundle.start(StartOption.NONE);
   bundle.start(EnumSet.of(
            StartOption.TRANSIENT));
}
```

Bundle Start Options with Enums, Attempt 1

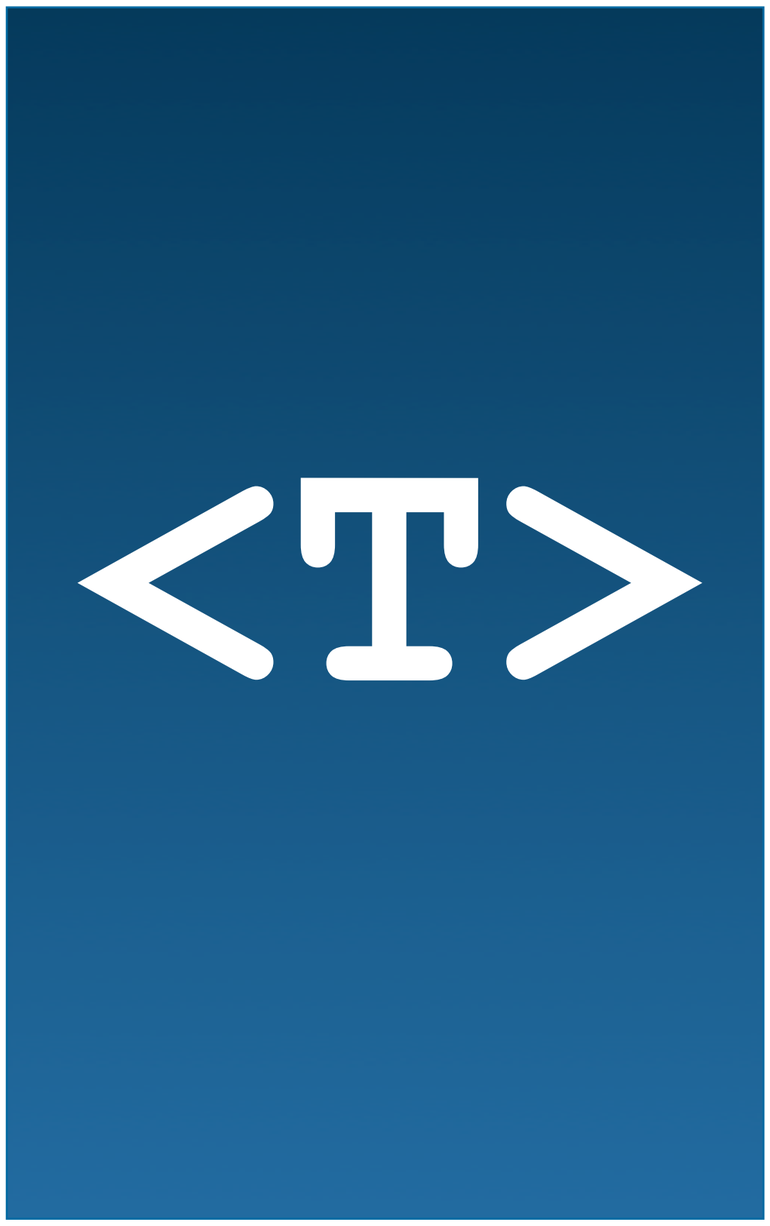
Hmmm ...

Bundle Start Options with Enums and varargs

```
interface Bundle {
  public enum StartOption {
     ACTIVATION_POLICY,
     TRANSIENT;
  }

  void start( StartOption... options );
  . . .
}

// Usage
void foo() {
   bundle.start();
   bundle.start( StartOption.TRANSIENT );
   bundle.start( StartOption.TRANSIENT,
        StartOption.ACTIVATION_POLICY );
}
```

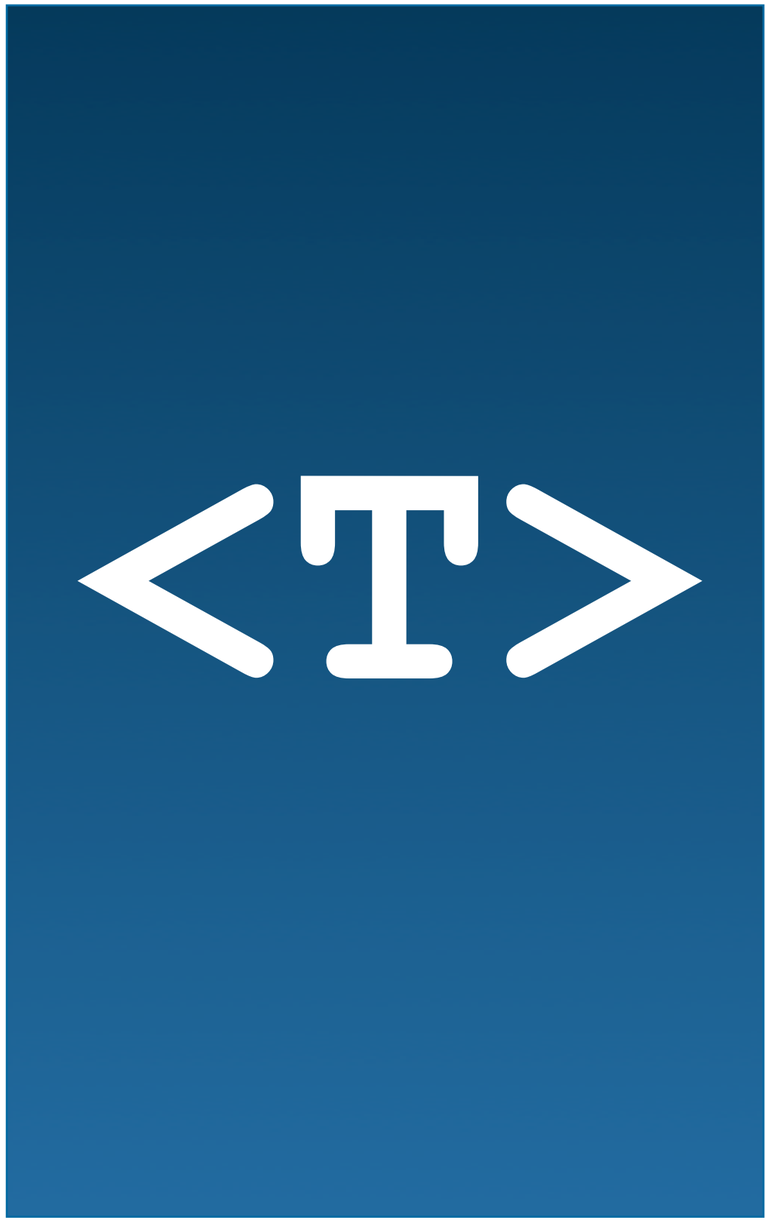
Generifying

- No more casting!
- Type safety



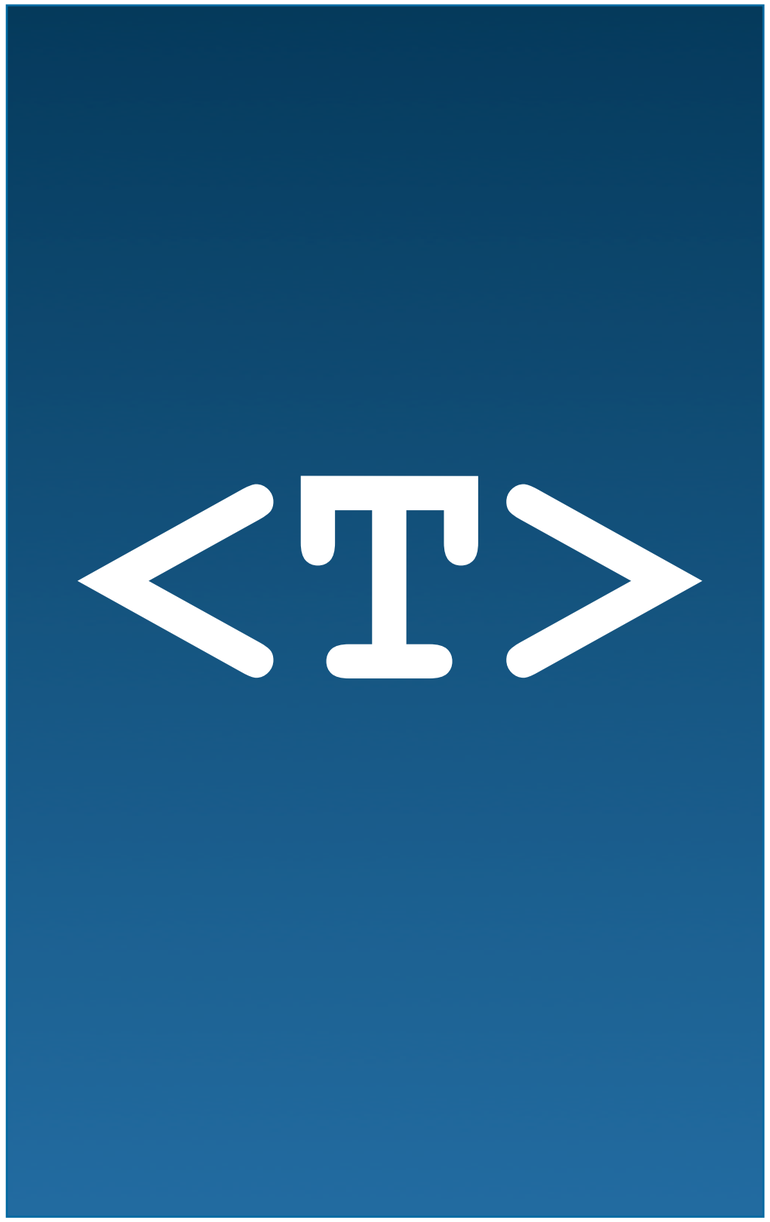
Generifying Services

```
// Old Usage

BundleContext context;

void foo() {
  ServiceReference r =
    registry.getServiceReference(
       LogService.class.getName() );
  if ( r != null ) {
    LogService ls = (LogService)
       registry.getService( r );
    ...
  }
}
```



Generifying Services

```
<S> S getService(ServiceReference<S> ref);
<S> S getService(Class<S> clazz);
// Usage
ServiceRegistry registry;

void foo() {
  ServiceReference<LogService> r =
    registry.getServiceReference(
          LogService.class );

  if ( r != null ) {
     LogService ls = registry.getService( r );
     ...
  }
}

void bar(){
   LogService ls = registry.getService(
       LogService.class);
}
```



Yes!

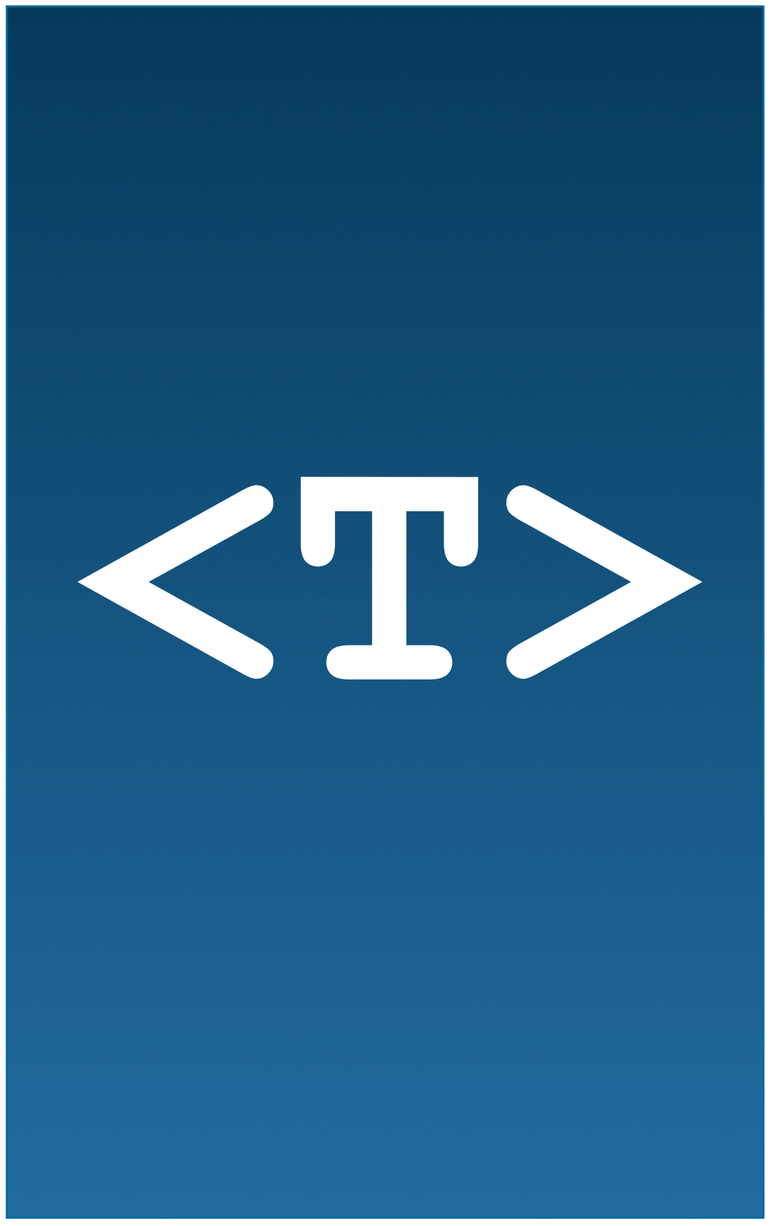
Yes!

No More Casting!

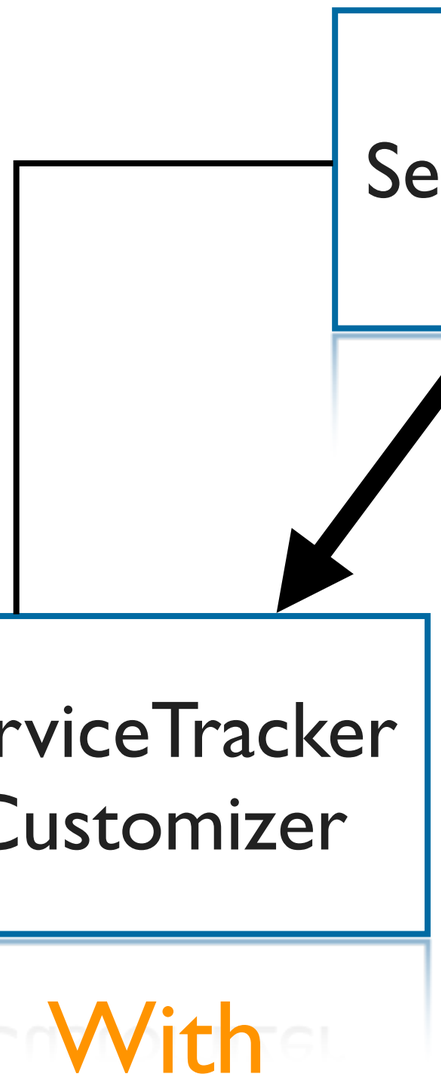
Generifying ServiceTracker

ServiceTracker

As Is

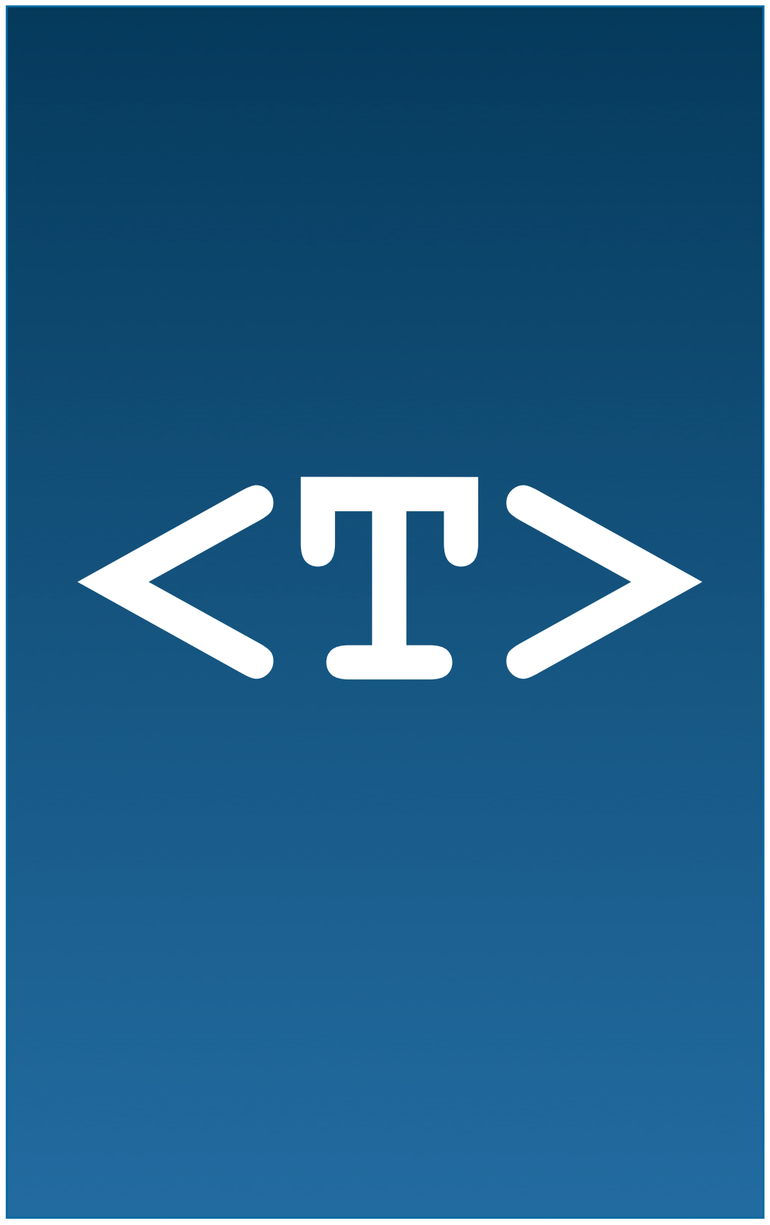


Generifying ServiceTracker

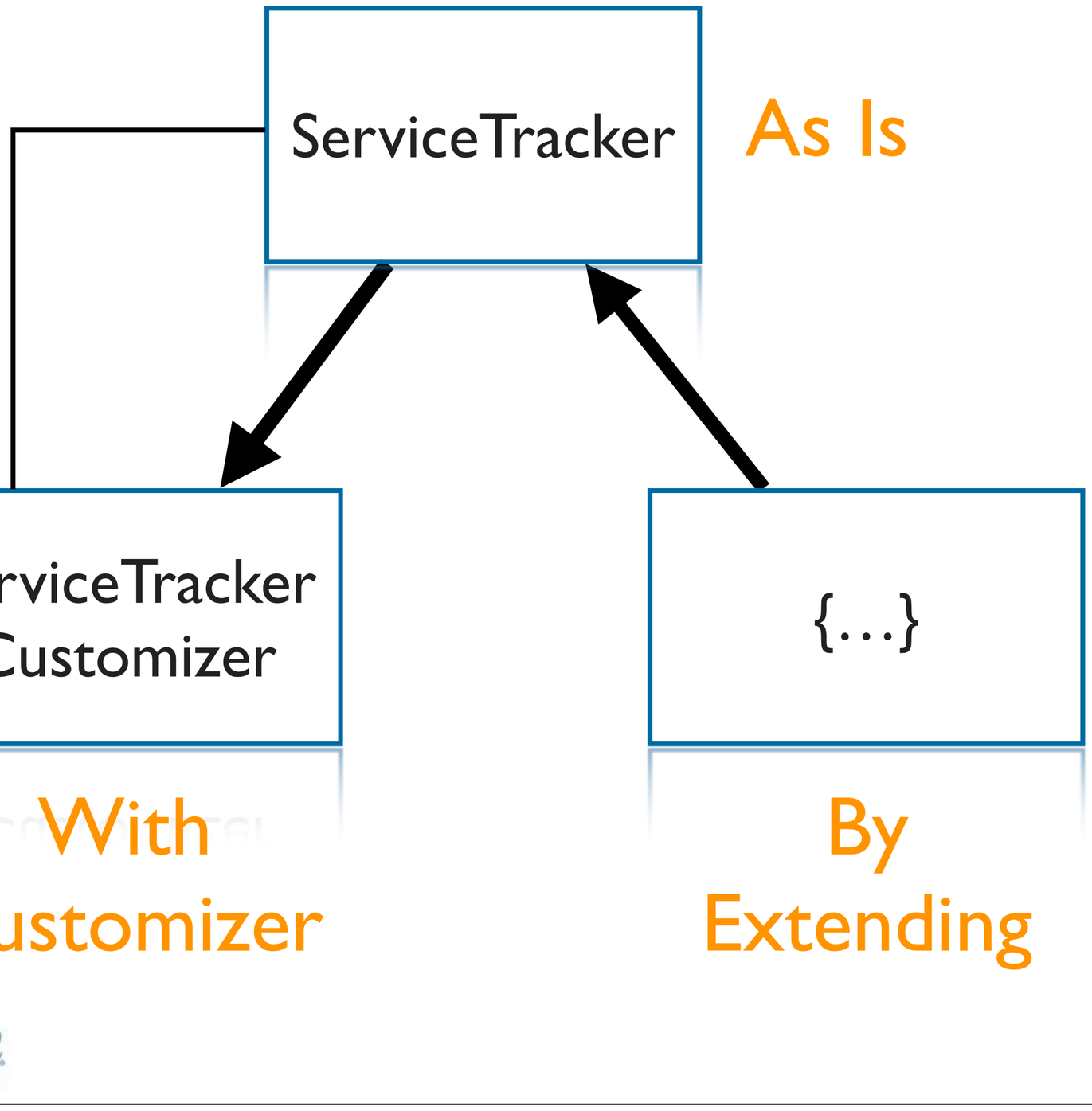


As Is

With Customizer



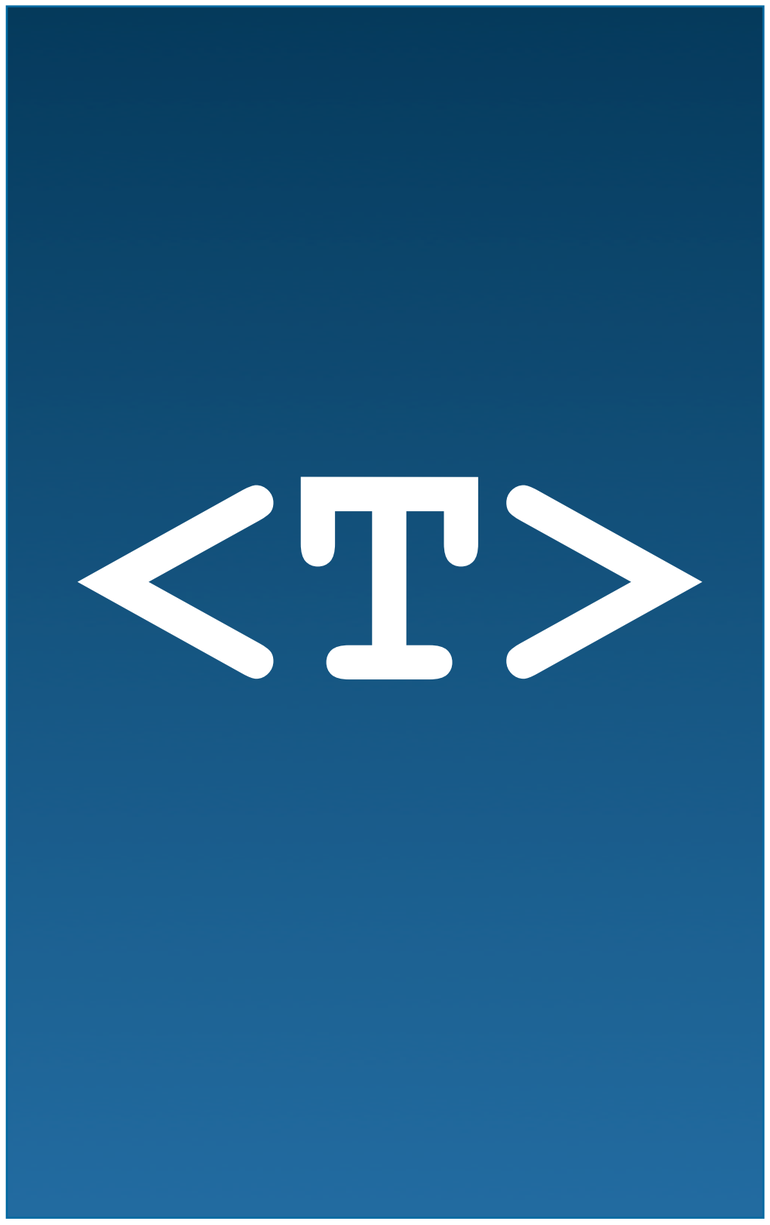
Generifying ServiceTracker



As Is

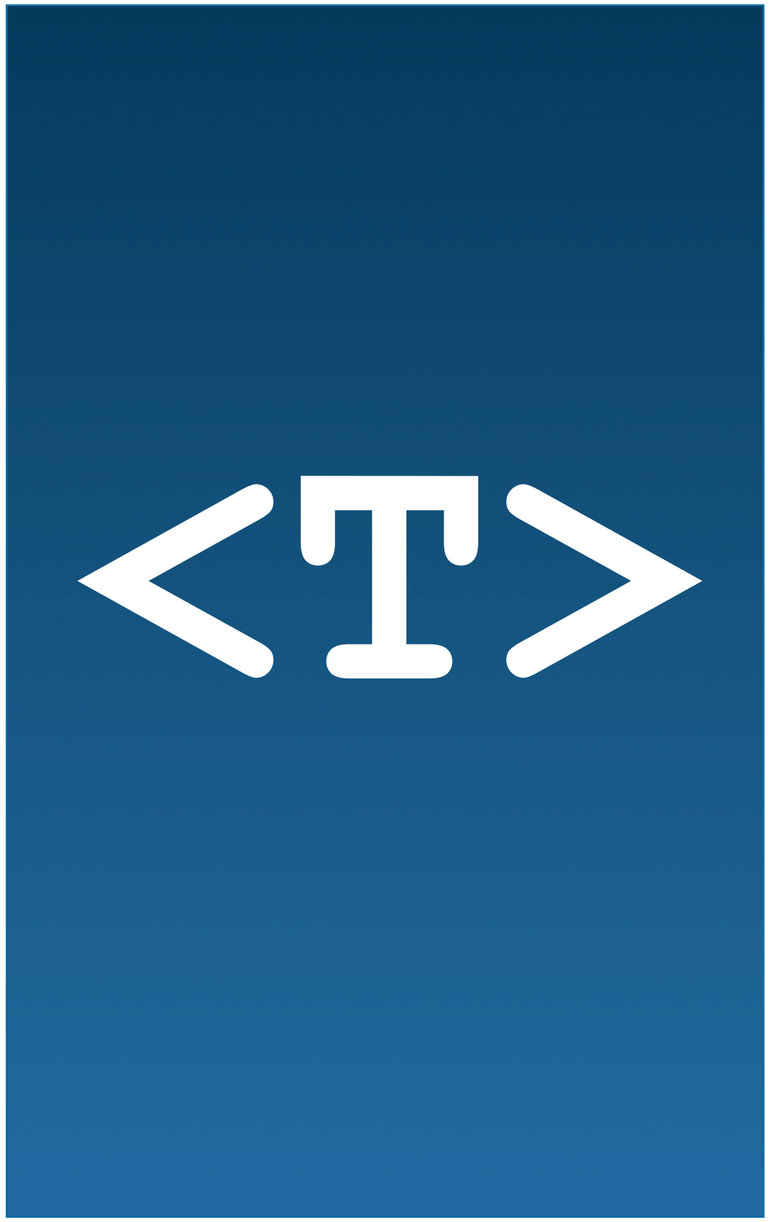
With Customizer

By Extending



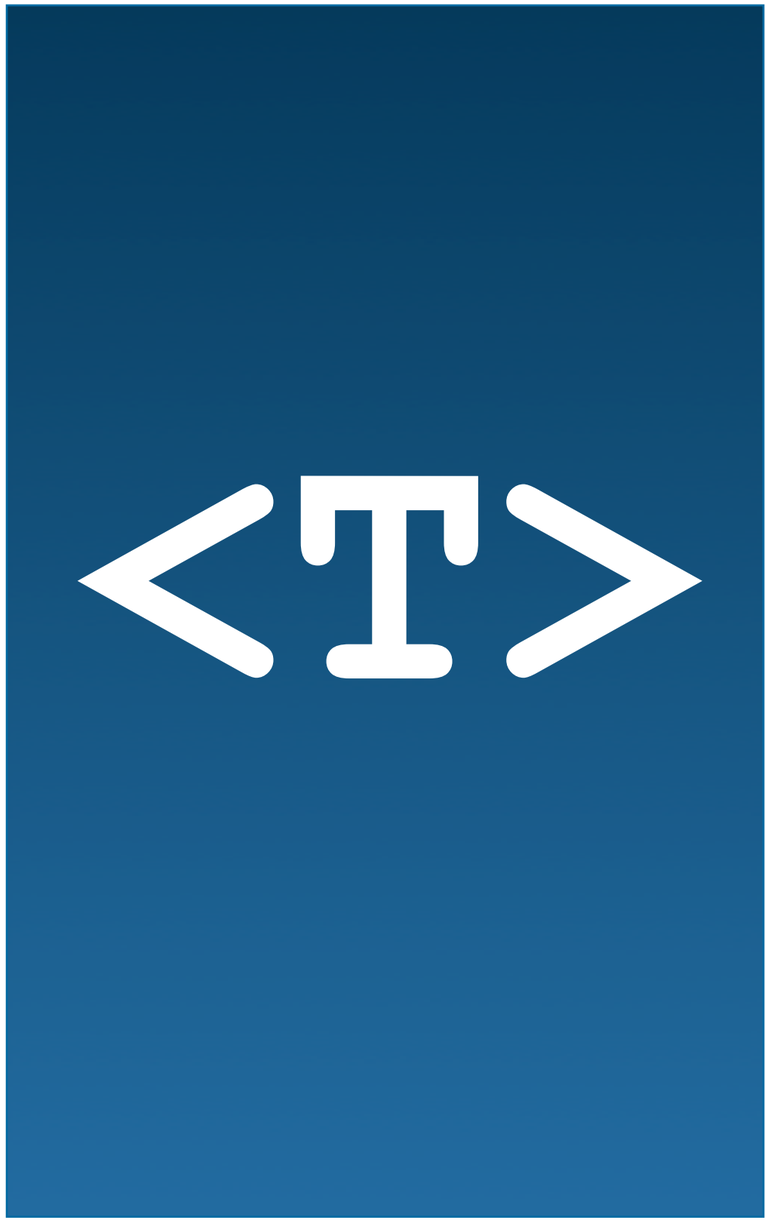
Generifying ServiceTracker As Is

```
void foo() {
  ServiceTracker st = new ServiceTracker(
    context, LogService.class.getName(),
           null );
  st.open();

  LogService log = (LogService)
                          st.getService();
  . . .
}
```



Generifying ServiceTracker by Extending

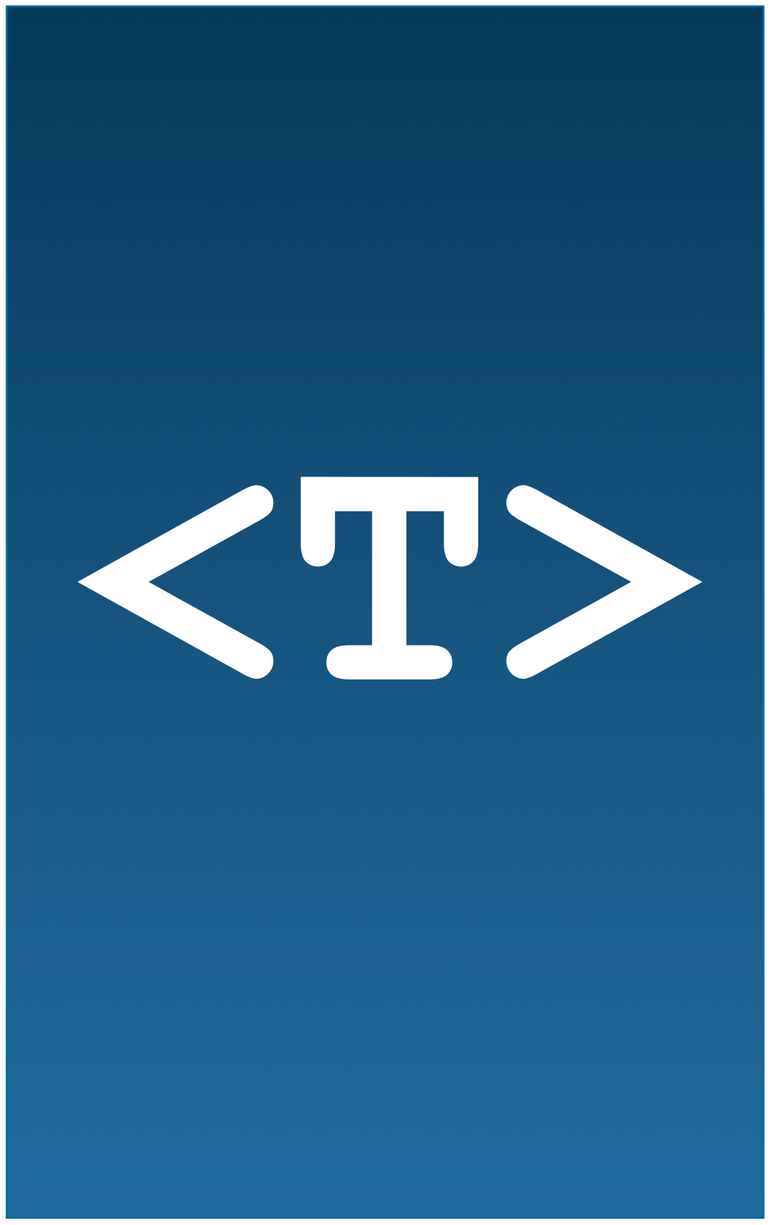
```
void foo() {
  ServiceTracker st1 = new ServiceTracker(
    context, LogService.class.getName(),null) {
      public Object addingService(
        ServiceReference ref) {
          return
            new LogTracked(
              super.addingService(ref));
      } . . .}
  );

  st.open();
  LogTracked t = (LogTracked) st.getService();
}
```



Generifying ServiceTracker with Customizer

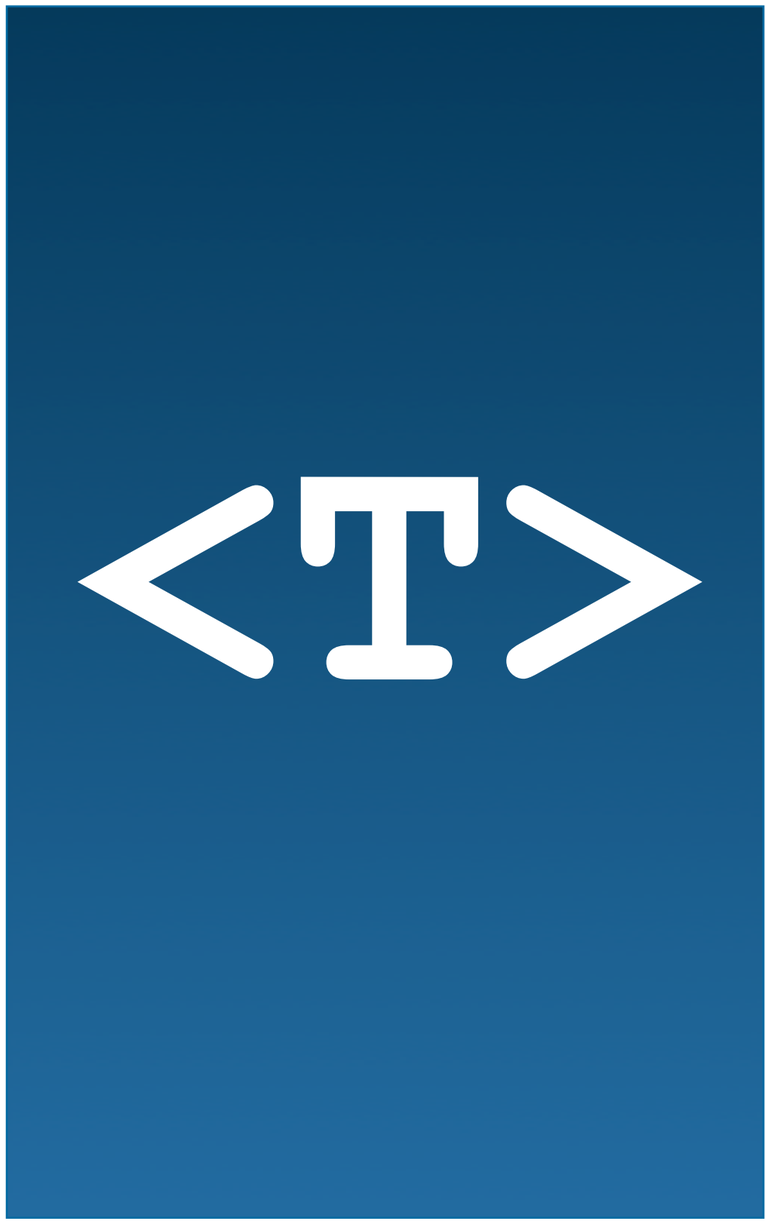
```
void foo() {
  ServiceTracker st1 = new ServiceTracker(
    context, LogService.class.getName(),
     this );
  st.open();
  LogTracked t = (LogTracked) st.getService();
  ...
}
```

```
Object addingService(ServiceReference ref) {
   return new LogTracked(ref);
}
void removedService(
  ServiceReference ref, Object o) {
  LogTracked lt = (LogTracked) o;
  context.ungetService(lt.getRef());
}
void modifiedService(
  ServiceReference ref, Object o) {
}
```

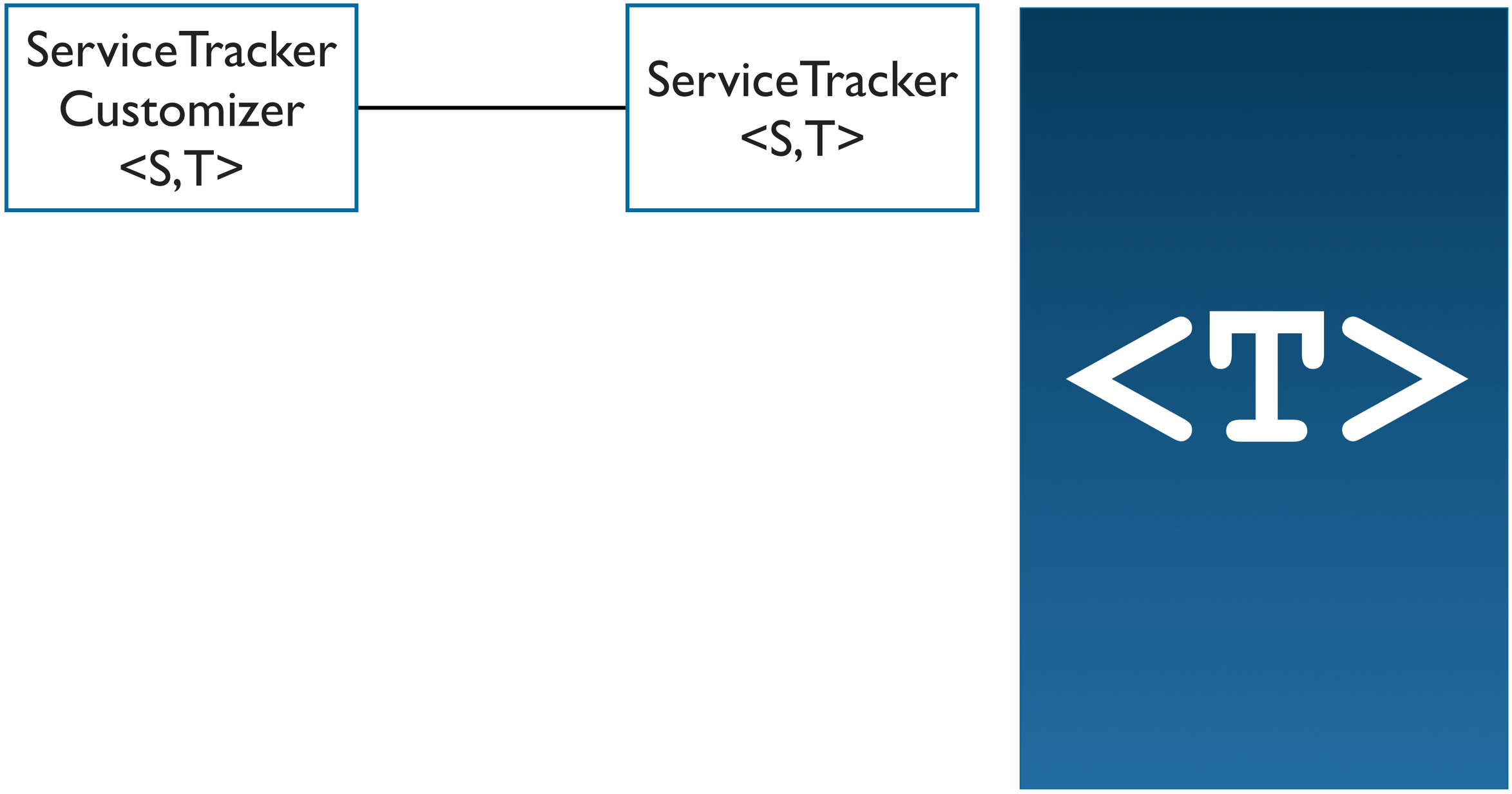


Generifying ServiceTracker

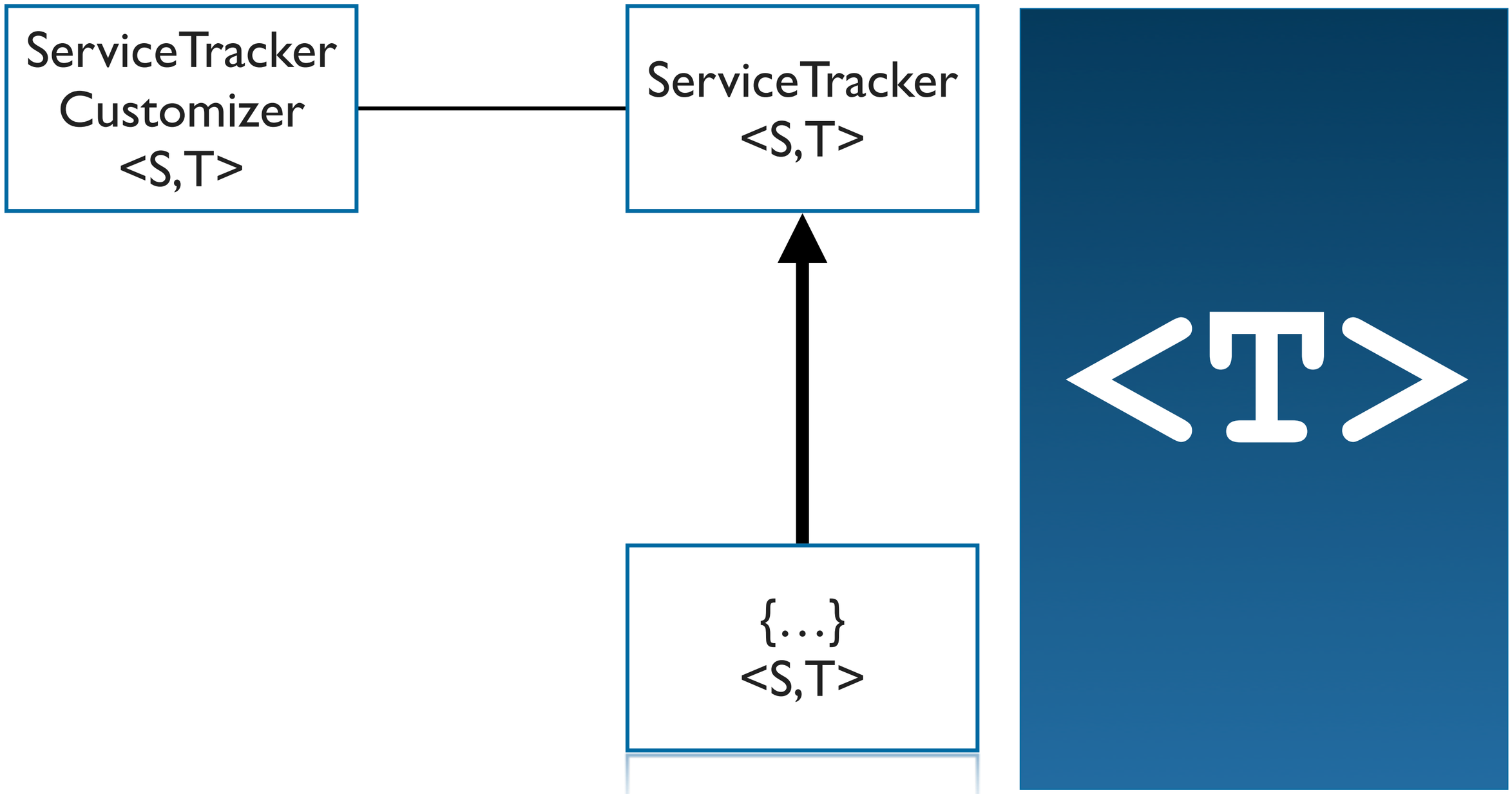
ServiceTracker
<S,T>



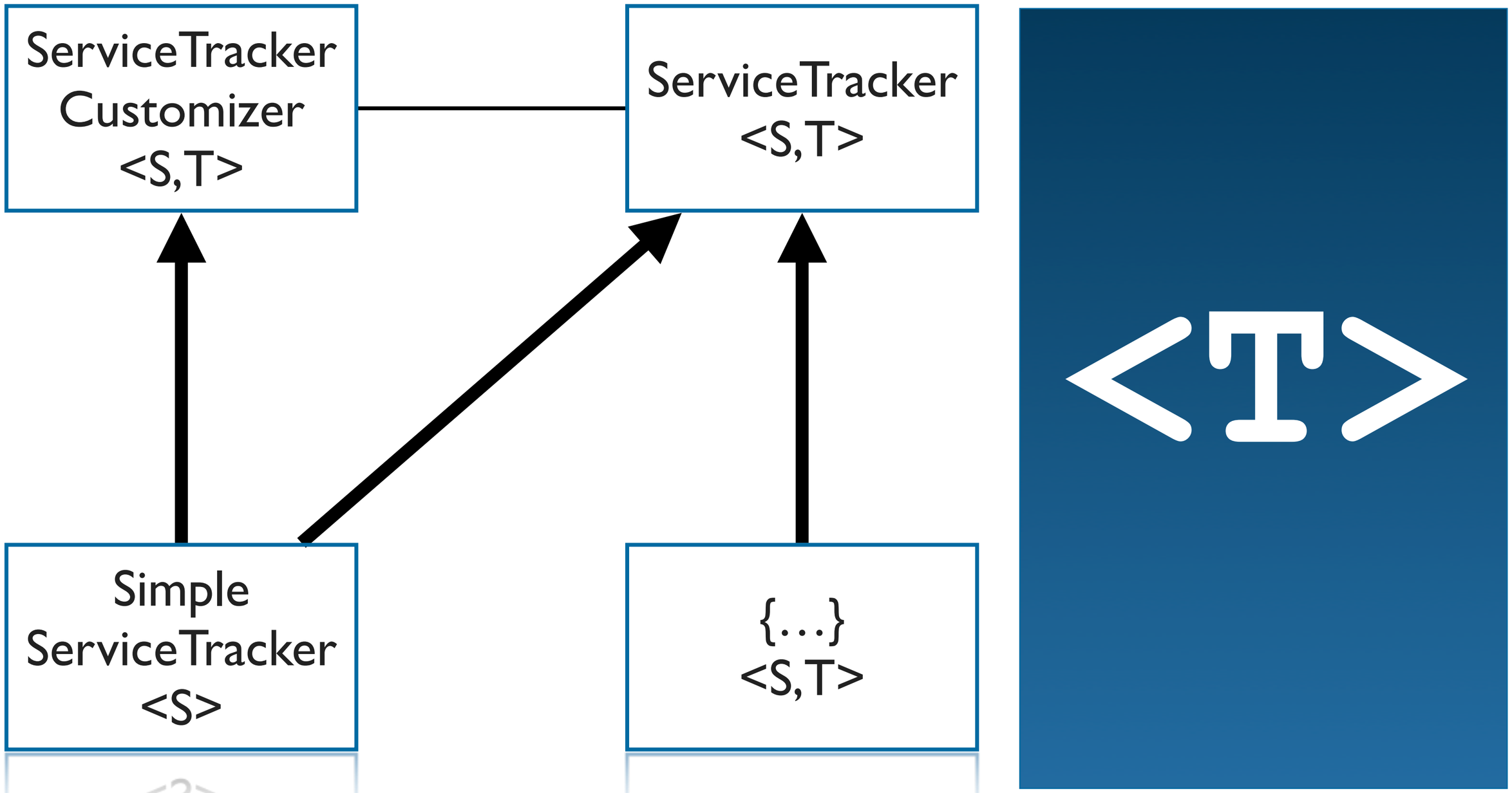
Generifying ServiceTracker



Generifying ServiceTracker

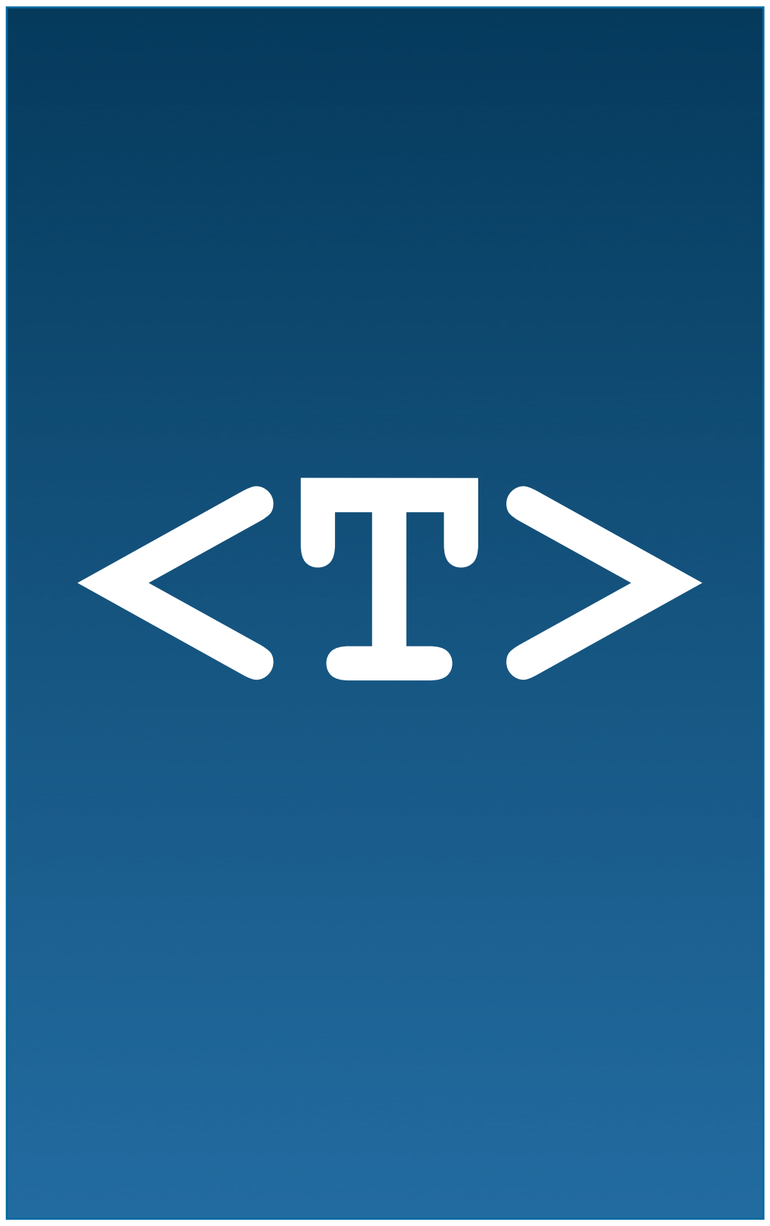


Generifying ServiceTracker



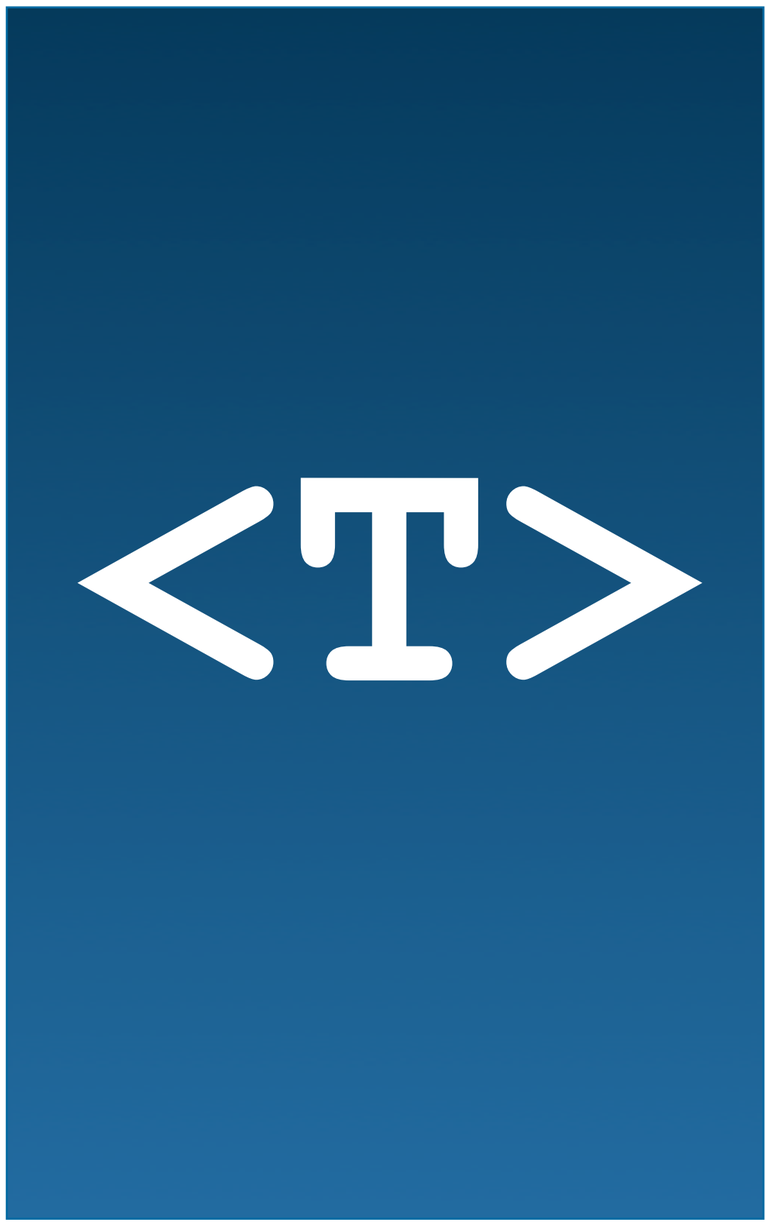
Generifying ServiceTracker with Simple Service Tracker

```
void foo() {
  SimpleServiceTracker<LogService> st =
     new SimpleServiceTracker(
         registry,
         LogService.class);

  st.open();

  LogService log = st.getService();
  . . .
}
```



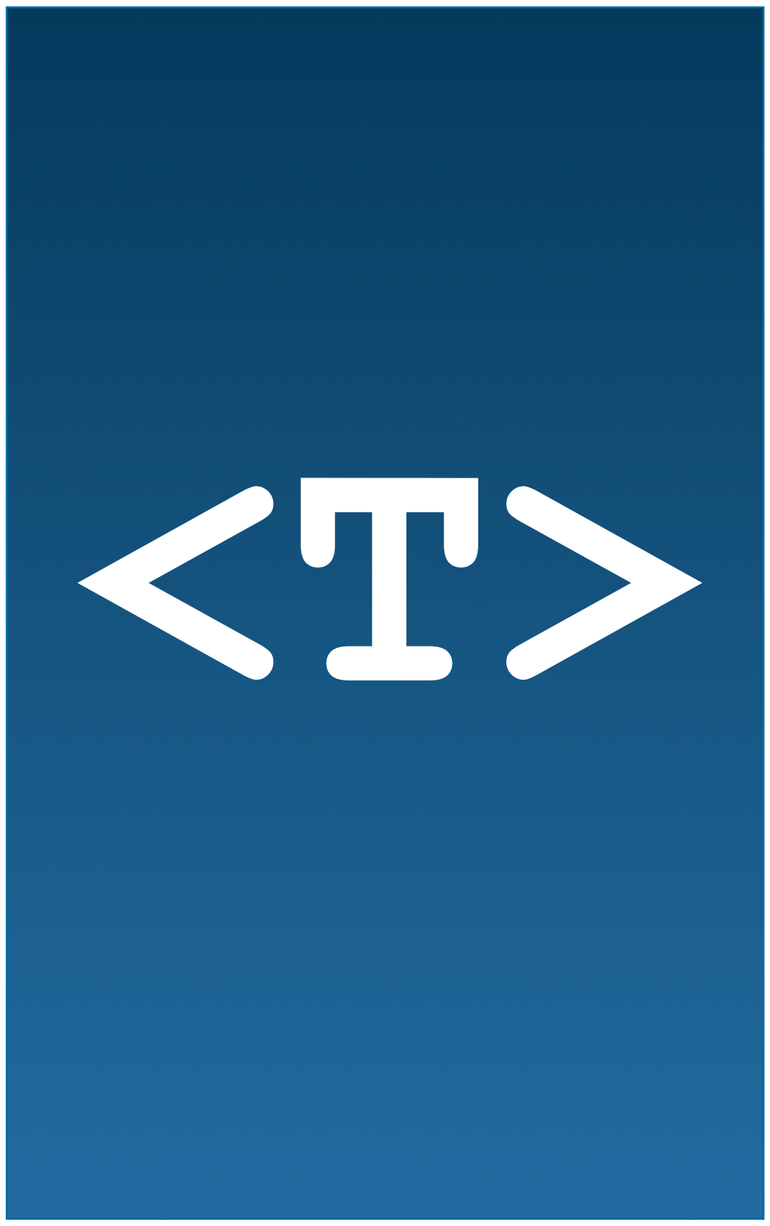
Generifying ServiceTracker by Extending

```
void foo() {
  ServiceTracker<LogService,LogTracked> st =
   new ServiceTracker<LogService,LogTracked>(
    context, LogService.class, null) {
       public LogTracked addingService(
         ServiceReference<LogService> ref) {
            return
              new LogTracked(
                  super.addingService(ref));
       } . . .}
  );

  st.open();
  LogTracked t = st.getService();
}
```



Generifying ServiceTracker with Customizer

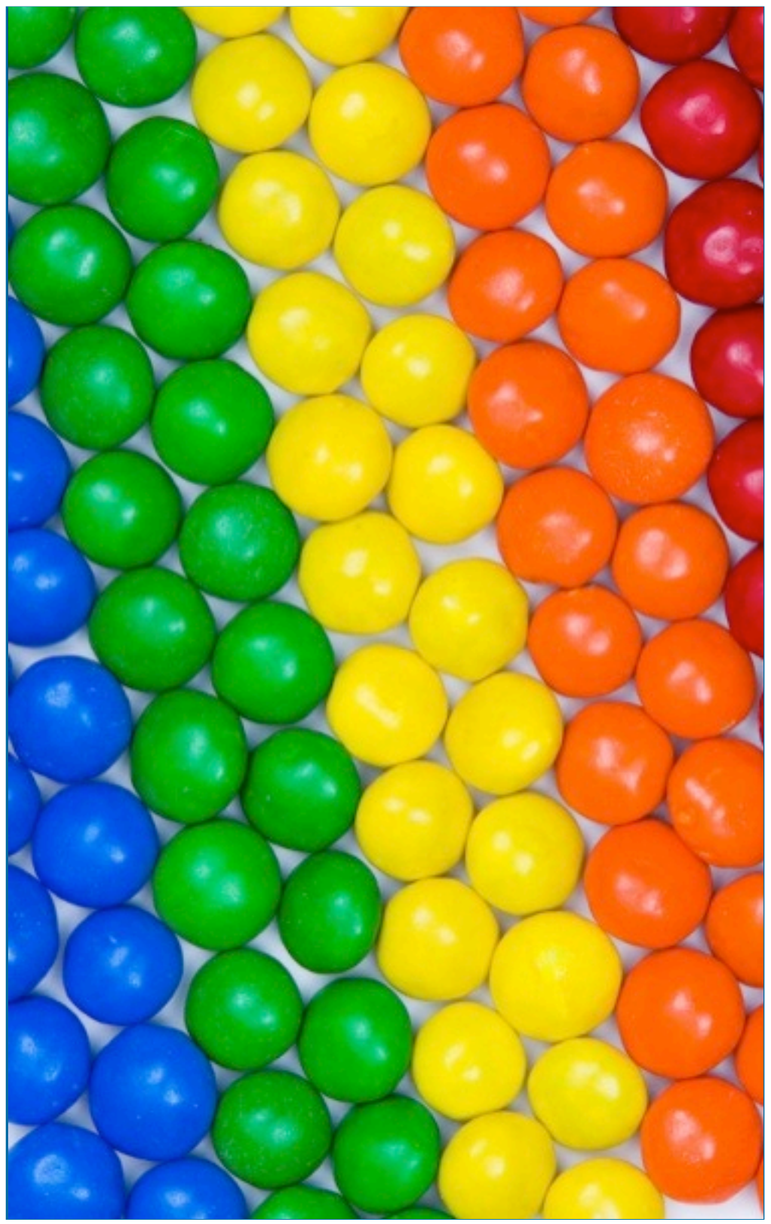
```
void foo() {
  ServiceTracker<LogService,LogTracked> st1 =
   new ServiceTracker<LogService,LogTracked>(
    context, LogService.class,
     this );
  st.open();
  LogTracked t = st.getService();
  ...
}

LogTracked addingService(
  ServiceReference<LogTracked> ref) {
   return new LogTracked(ref);
}

void removedService(
  ServiceReference<LogTracked> ref, LogTracked
lt) {
  lt.unget();
}
```



Arrays

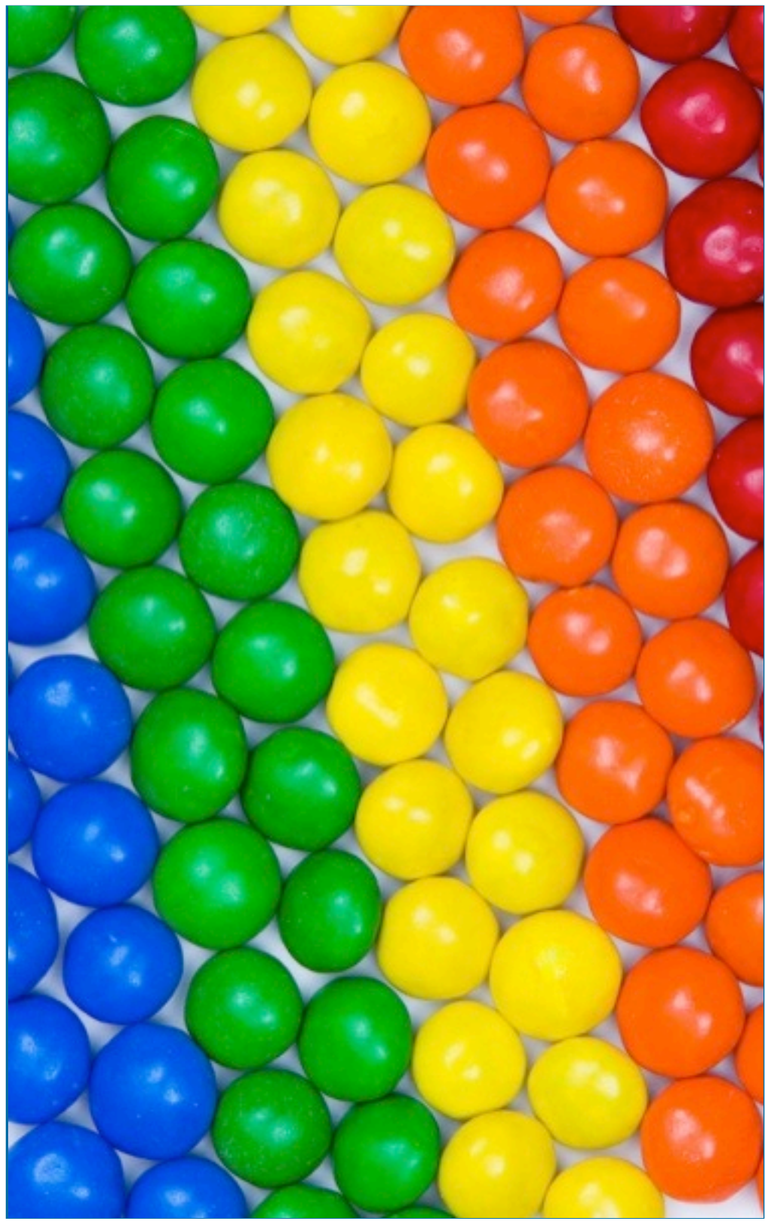
```
// Old

  Bundle[] bundles = context.getBundles();
  for ( int i = 0;
        bundles != null && i<bundles.length;
        i++ )
     foo(bundles[i]);
```

```
// New
  for ( Bundle bundle : framework.getBundles())
    foo(bundle);
```



Dictionary

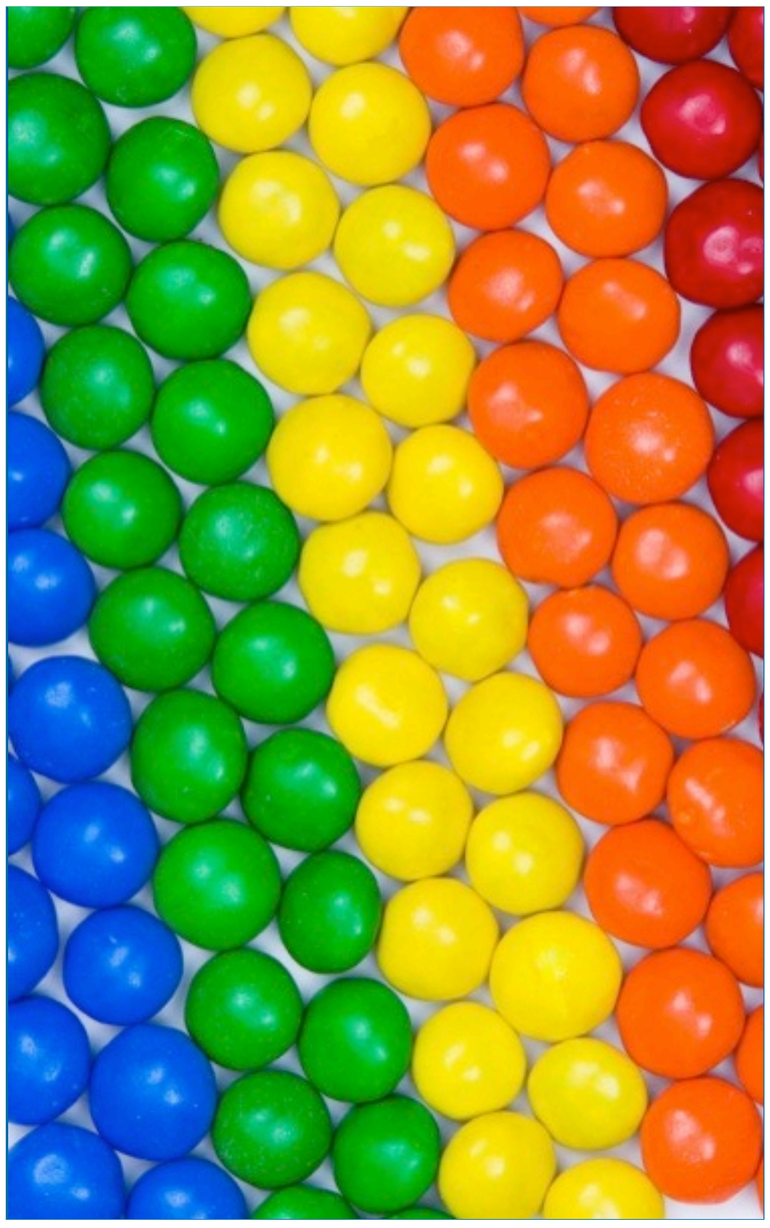
```
// Old

   Dictionary d = bundle.getHeaders();
   for ( Enumeration e = d.keys();
         e.hasMoreElements(); )
      foo((String) e.nextElement);
```



```
// New
   Map<String,String> d = bundle.getHeaders();
   for ( String key: d.keySet())
      foo( key );
```

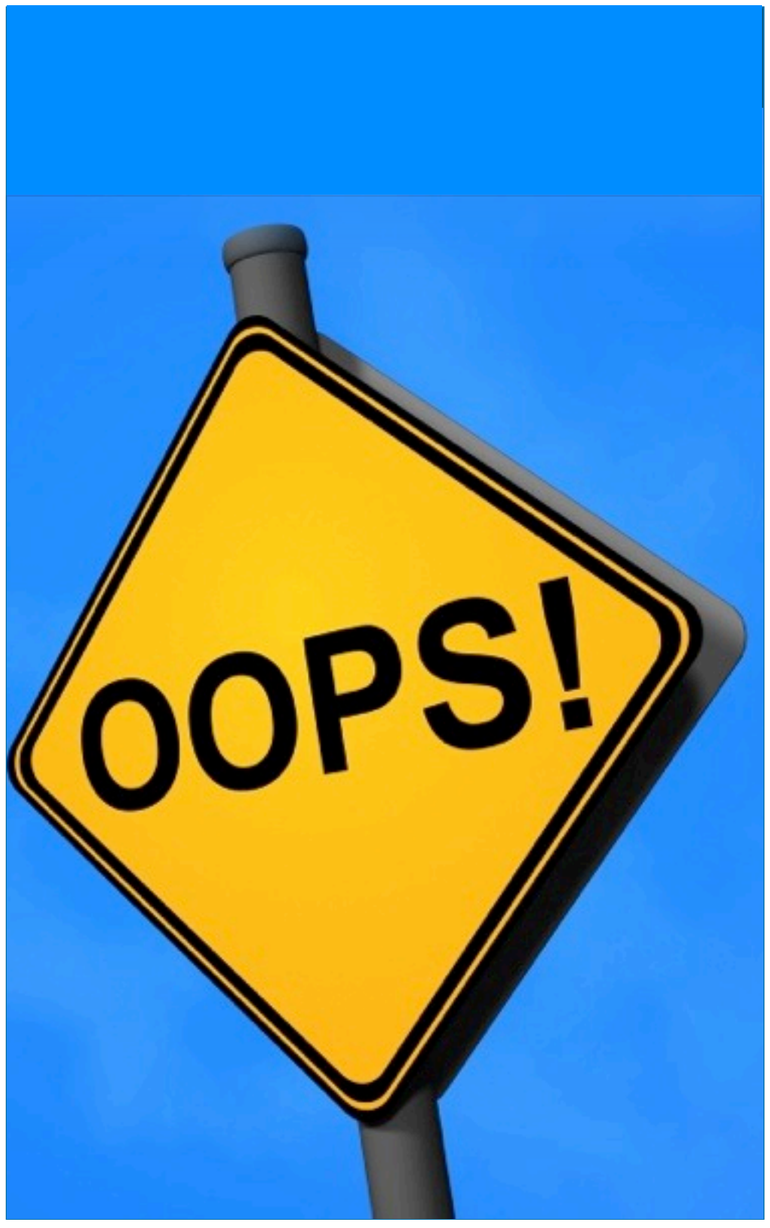
Dictionary

```
// Old

  BundleContext context;
  void foo() {
    Dictionary p = new Hashtable();
    p.put("service.pid",MYPID);
    context.registerService(
          MyService.class.getName(),
          service, p );

// New
  ServiceRegistry reg;
  Map<String,Object> p =
        HashMap<String,Object>();
  p.put("service.pid",MYPID);

  reg.registerService(
          MyService.class,
          service, p );
```



Checked Exceptions: BundleException & InvalidSyntaxException

```
//Old

    ServiceReference[] refs;
    try {
      refs = context.getServiceReferences(
          LogService.class.getName(),
          null );
    } catch( InvalidSyntaxException e ) {
       // Cannot happen
    }

// New
    Collection<ServiceReference<LogService>>
        refs = context.getServiceReferences(
          LogService.class,
          null );
```



Annotations

```
@Retention(RetentionPolicy.CLASS)
@Target(ElementType.PACKAGE)

public @interface export {
    String version() default "0";
    String mandatory() default "";
    String uses() default "";
    Class[] include() default {};
    Class[] exclude() default {};
    String[] extra() default {};
}
```

Annotations

```
// package-info.java

@export(
  version="2.3",
  uses="com.acme.bar",
  exclude={Bundle.class})
```

```
package com.acme.foo;
```

Annotations

```
@Bundle(
  require_bundle = {
    "com.acme.foo; version='[2.1,3)'",
    "com.acme.foo2",
    "com.aQute.bndlib; version='2.1" },
  bundle_name     = "OSGi Annotations",
  bundle_description  =
    "Provides annotations to OSGi bundles",
  bundle_docurl   = "http://www.osgi.org",
  bundle_copyright=
    "Copyright (c) OSGi Alliance 2009",
)

module com.acme.foo;
```

Annotations

JSR 294 Improved Modularity Support in the JavaTM Programming Language

JSR 294 Improved Modularity Support in the JavaTM Programming Language

294

JSR
294

JSR 294 Improved Modularity Support in the JavaTM Programming Language

$$\frac{294}{7}$$

JSR
294

JSR 294 Improved Modularity Support in the JavaTM Programming Language

42

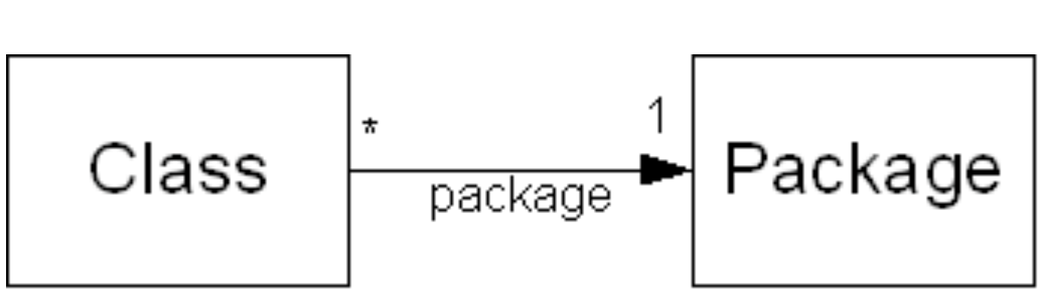
JSR
294

JSR 294 Improved Modularity Support in the JavaTM Programming Language

Java 7 × Answer to Life, the Universe, and Everything

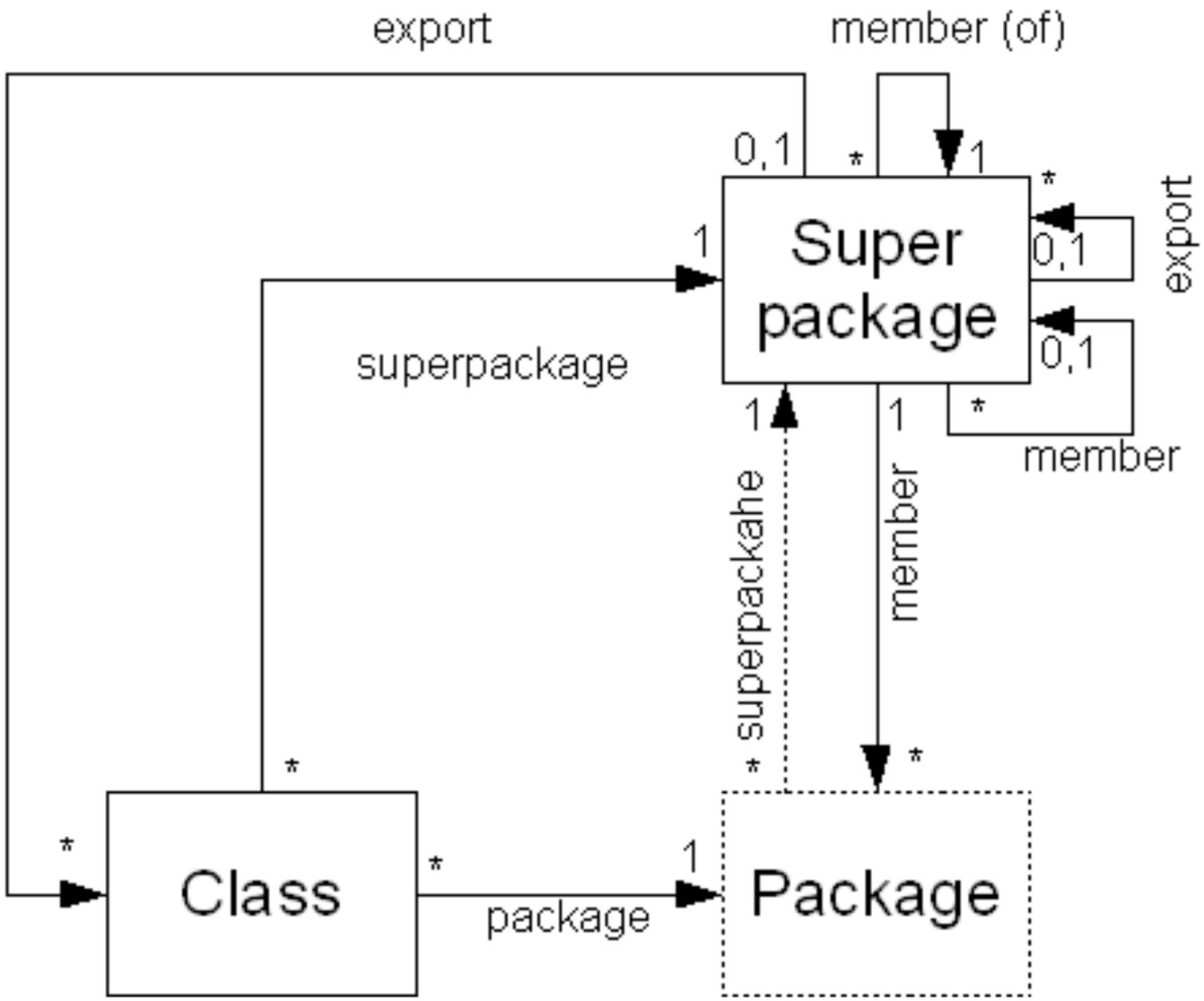
JSR 294

JSR 294 Improved Modularity Support in the JavaTM Programming Language



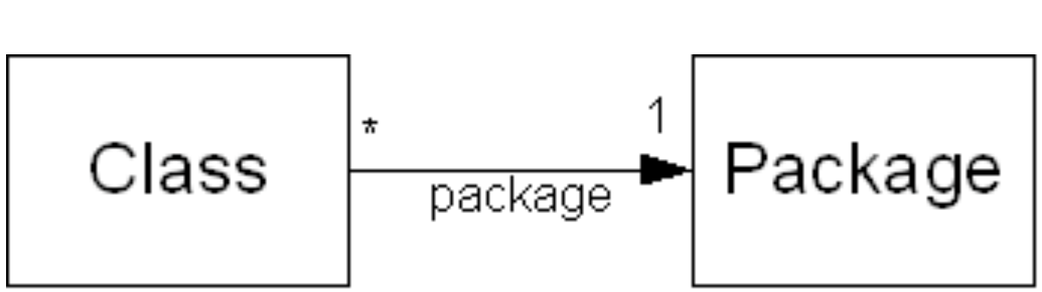
JSR 294

JSR 294 Improved Modularity Support in the JavaTM Programming Language



JSR 294

JSR 294 Improved Modularity Support in the JavaTM Programming Language



JSR 294

JSR 294 Improved Modularity Support in the JavaTM Programming Language

JSR 294 Improved Modularity Support in the JavaTM Programming Language

```
module
```

JSR
294

JSR 294 Improved Modularity Support in the JavaTM Programming Language

```
module class Foo {}

public class Bar {
  module int foo() {}
}

module interface Baz {}
```

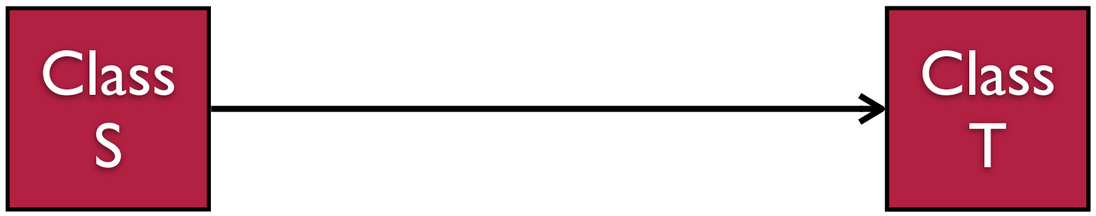
JSR 294 Improved Modularity Support in the JavaTM Programming Language

Visibility

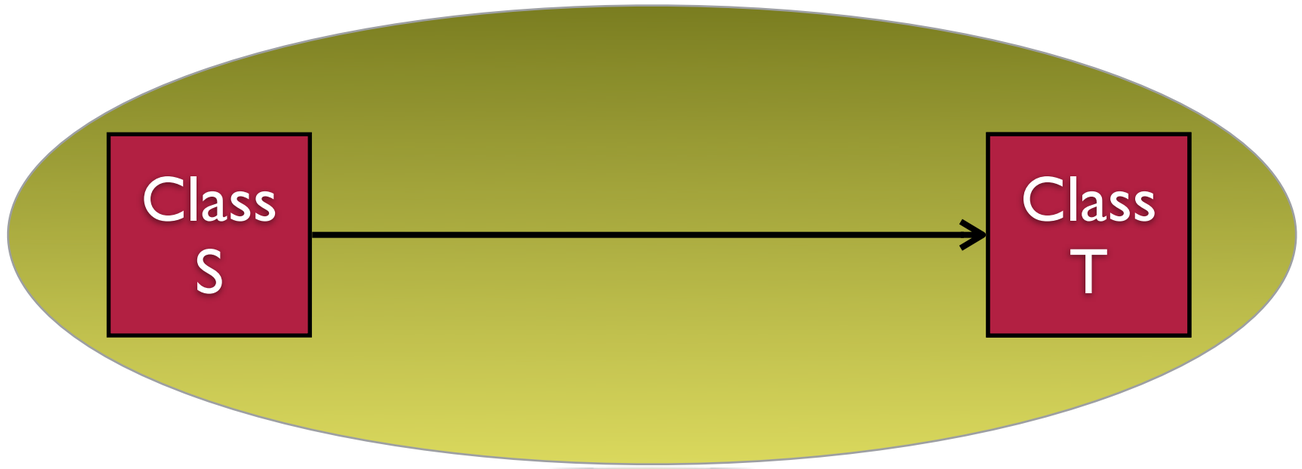
vs

Accessibility

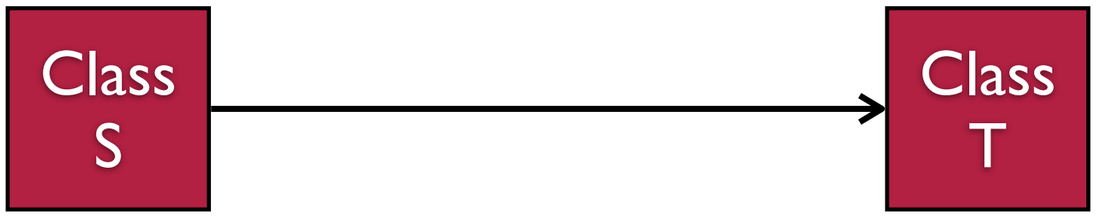
Visibility



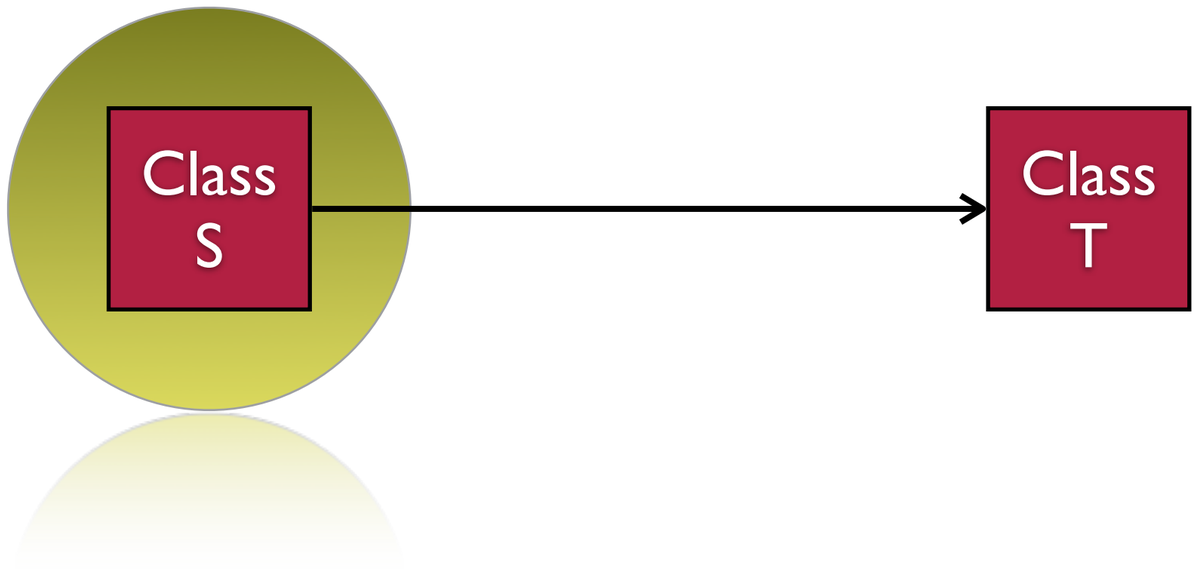
Visibility



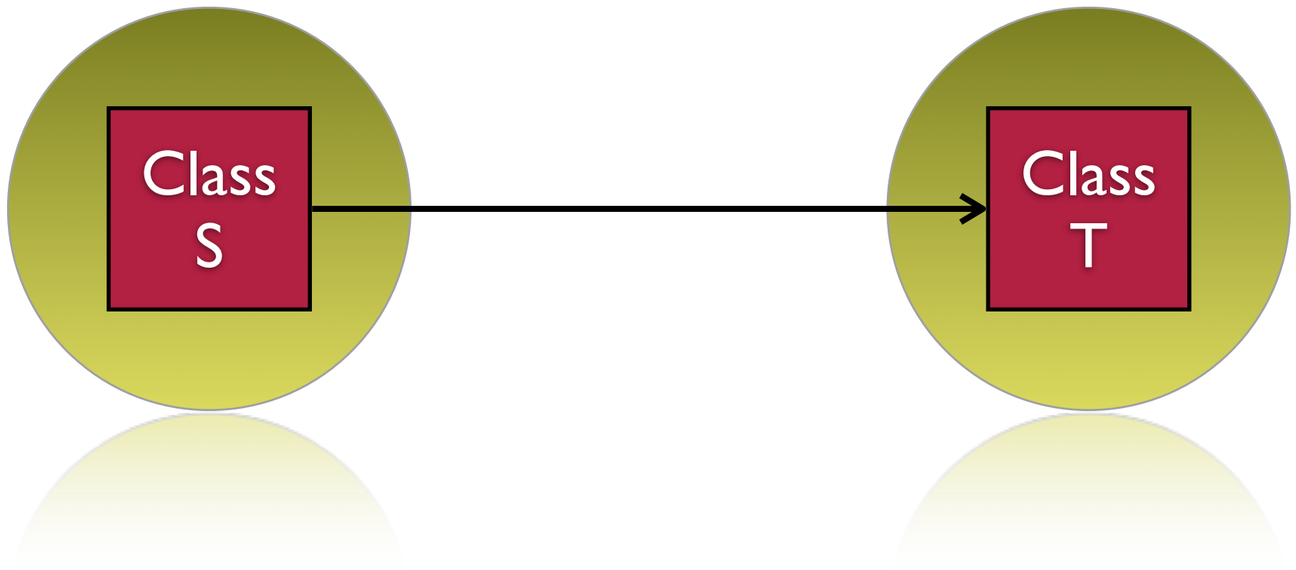
Visibility



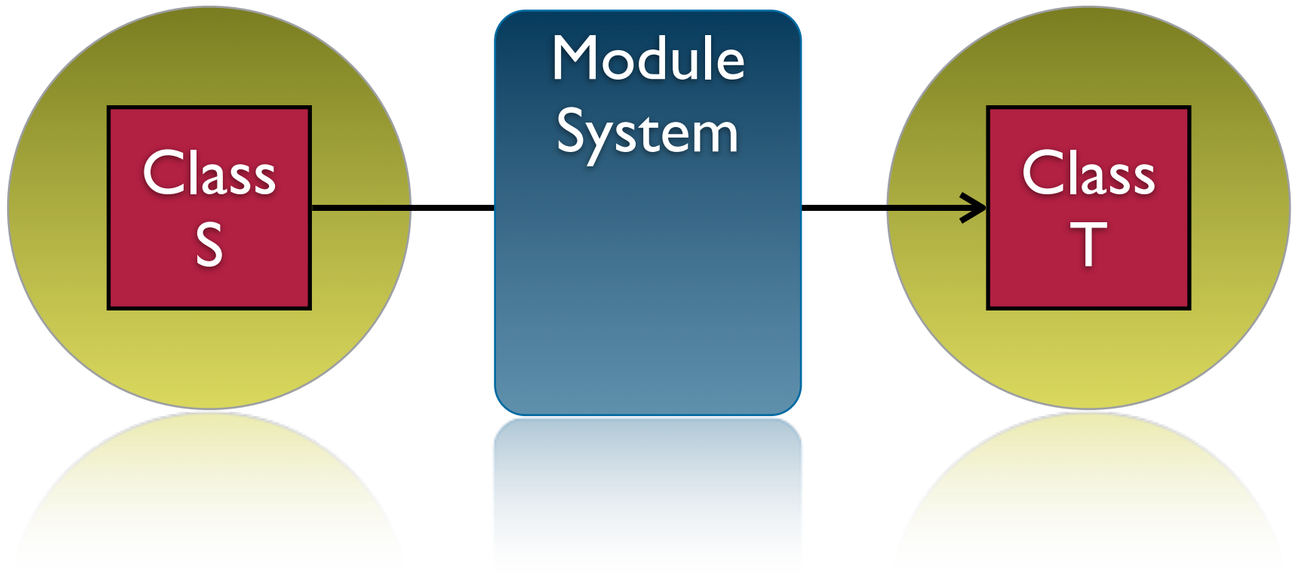
Visibility



Visibility



Visibility



Accessibility

public

package

private

Accessibility

public

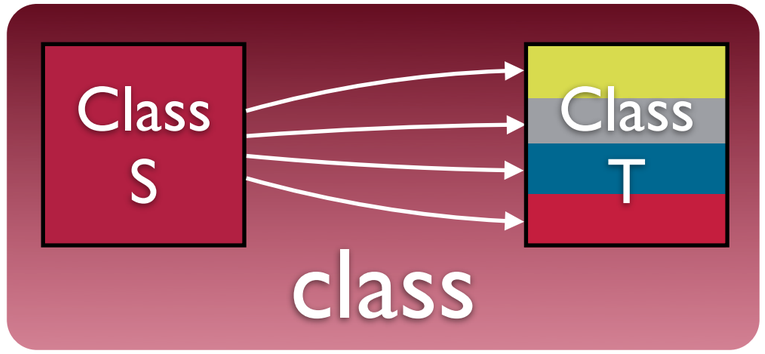
module

package

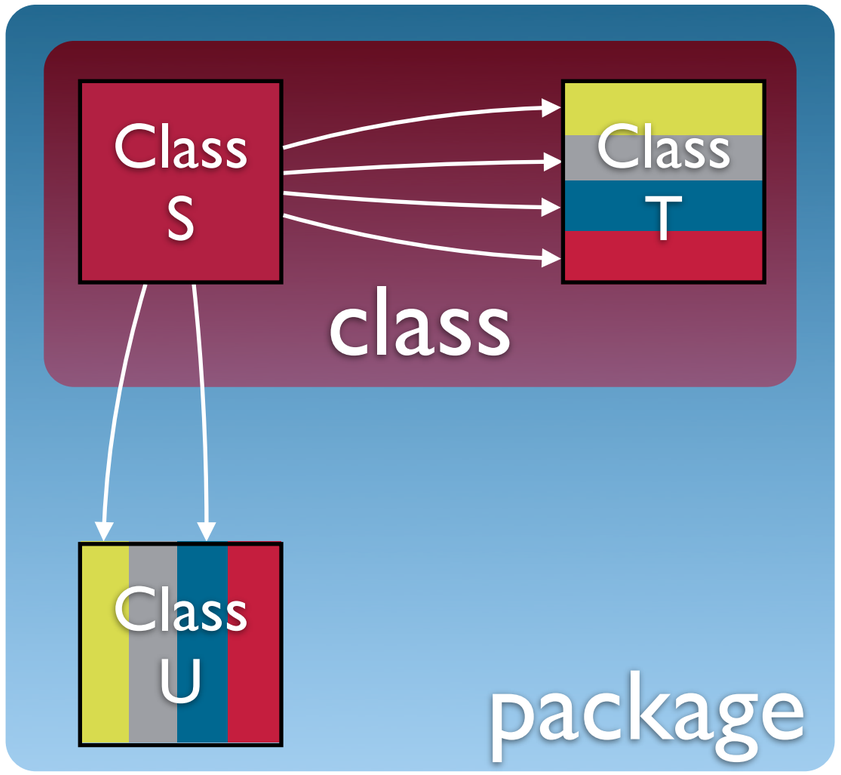
private

Accessibility

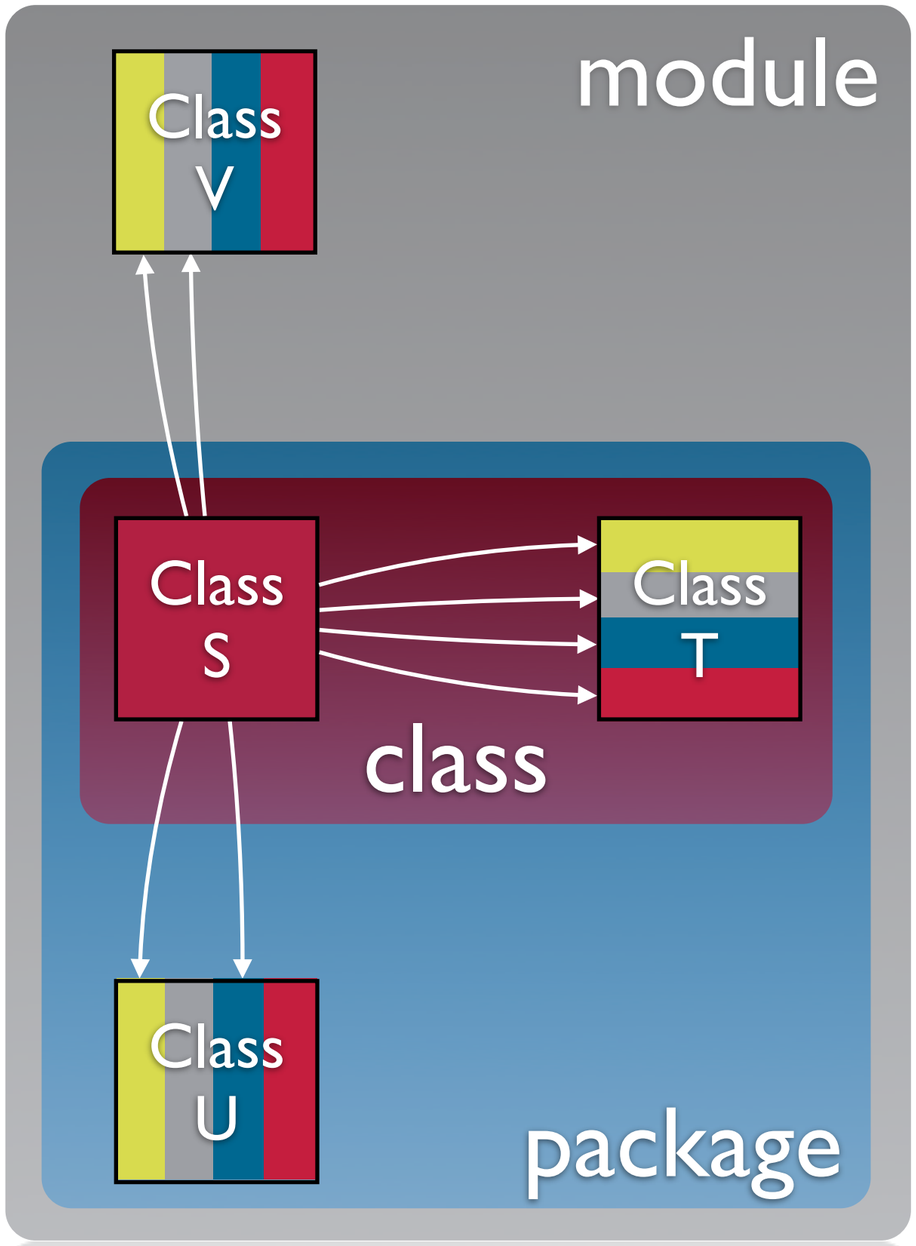
Accessibility



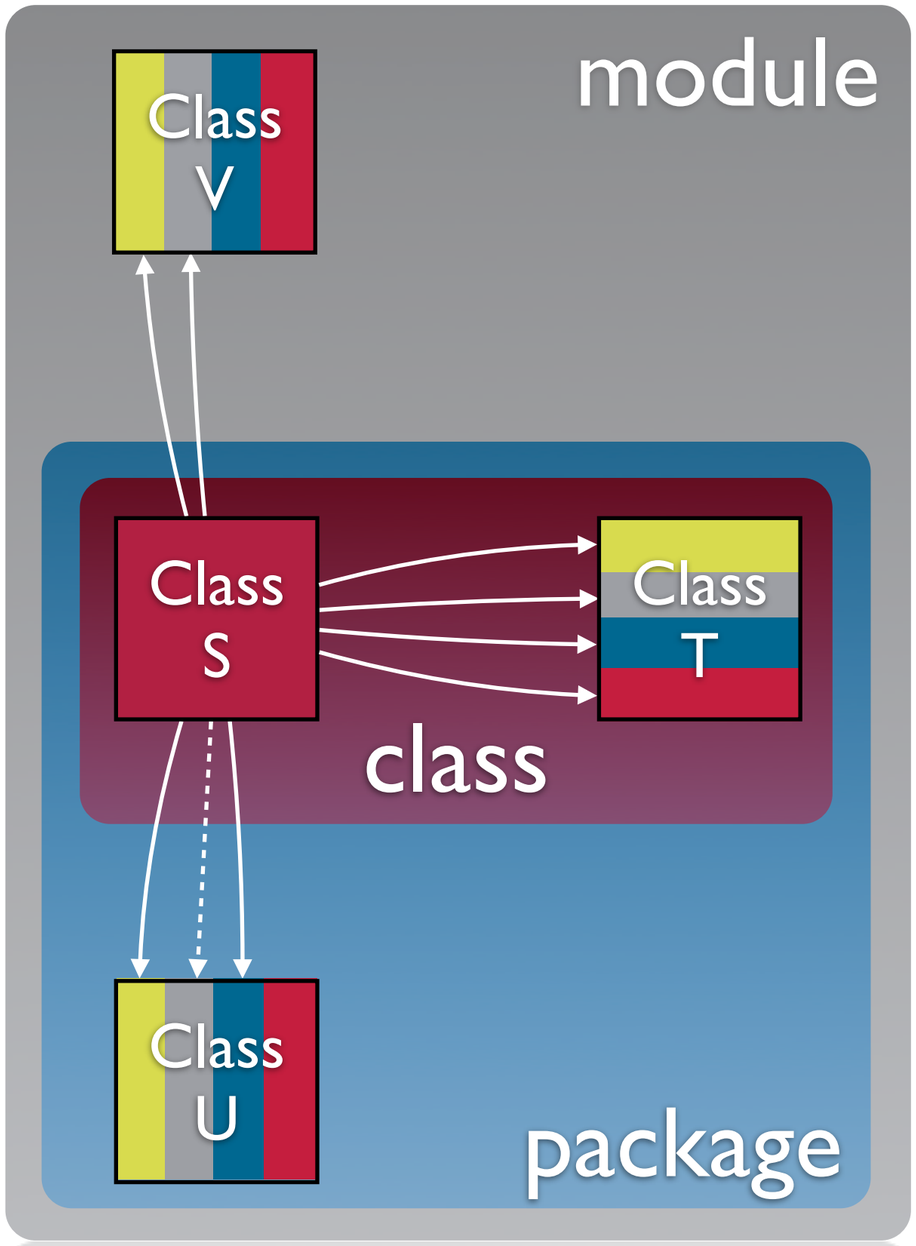
Accessibility



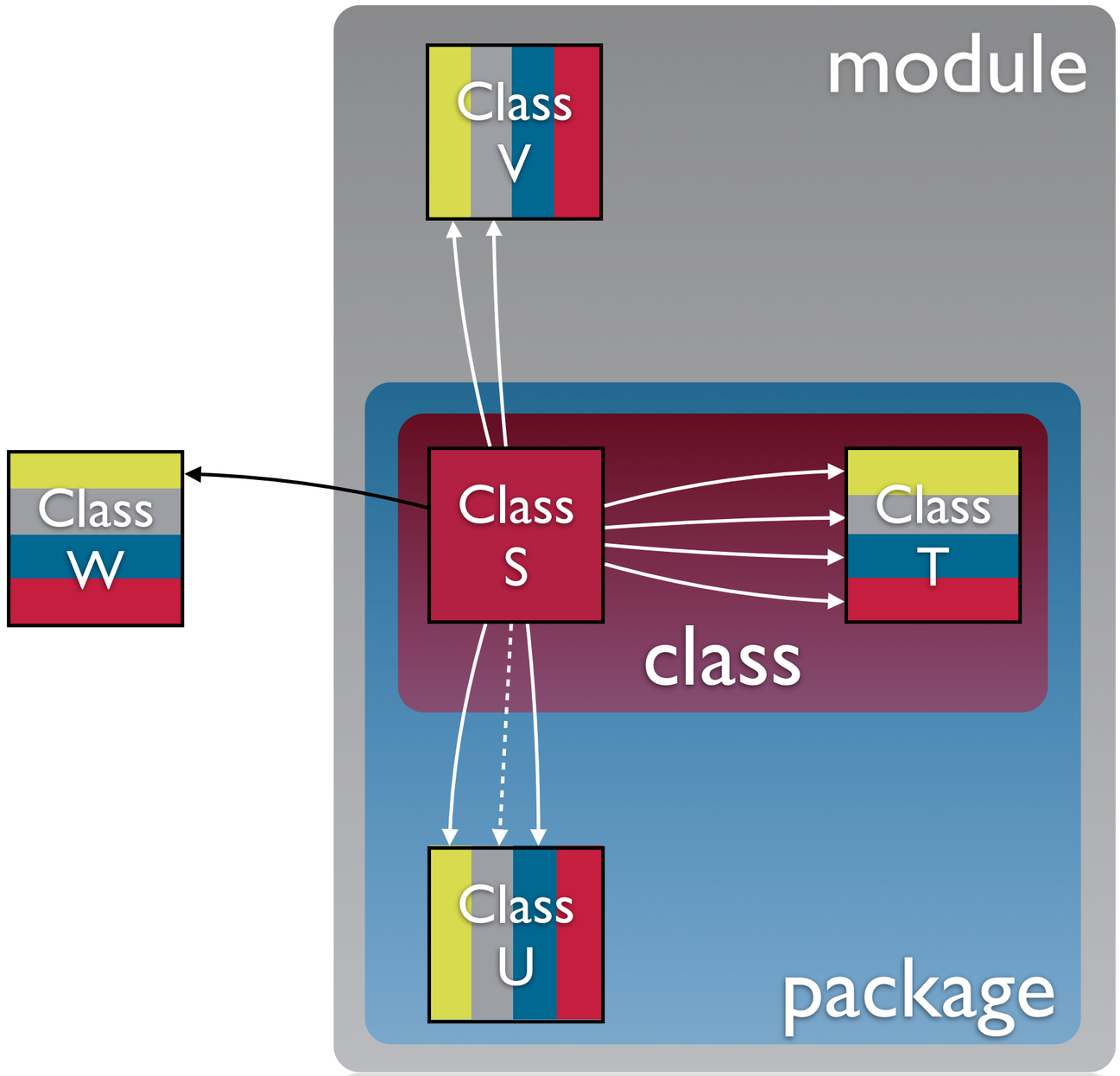
Accessibility



Accessibility



Accessibility



Is JSR 294 a *Meta* Module System?

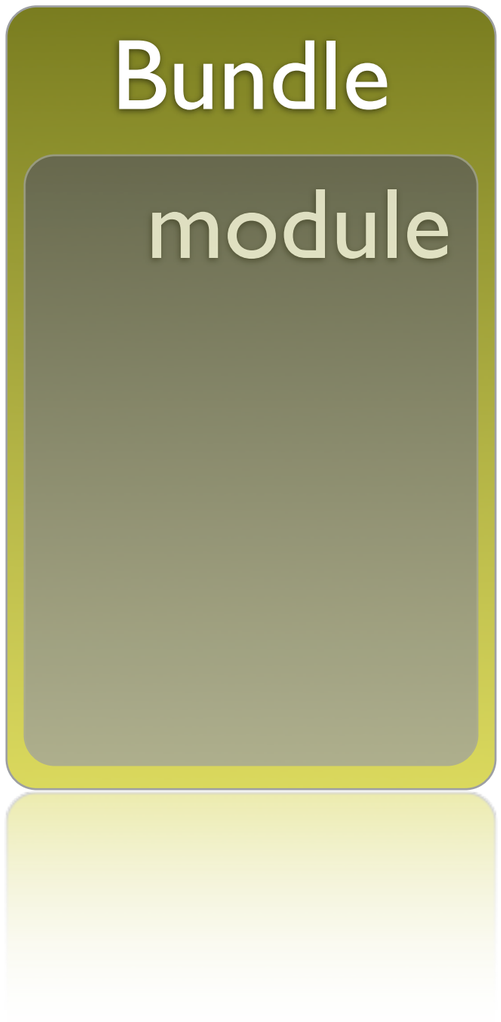
(Implemented by OSGi™, Jigsaw, Guice, ...)

Is JSR 294 a *Meta* Module System?

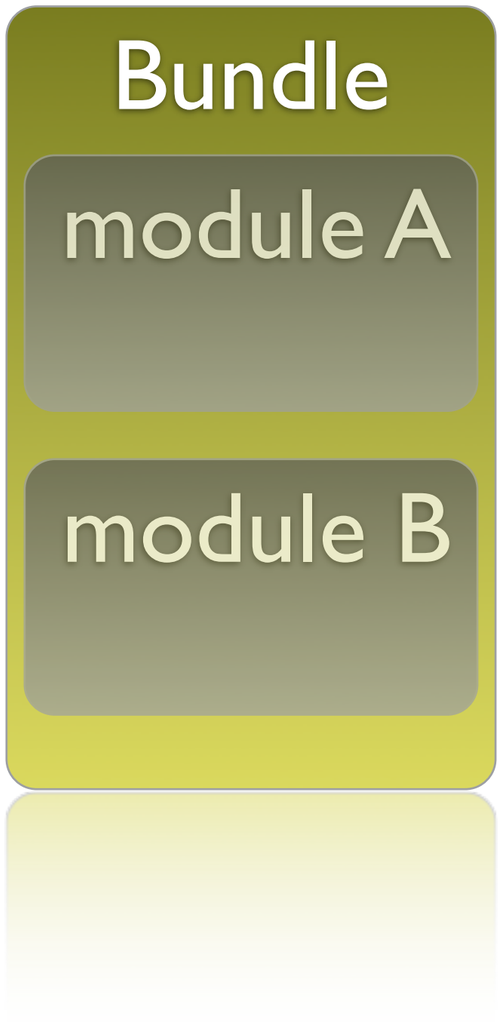
(Implemented by OSGi™, Jigsaw, Guice, ...)

No Java™ Standard for Modules!

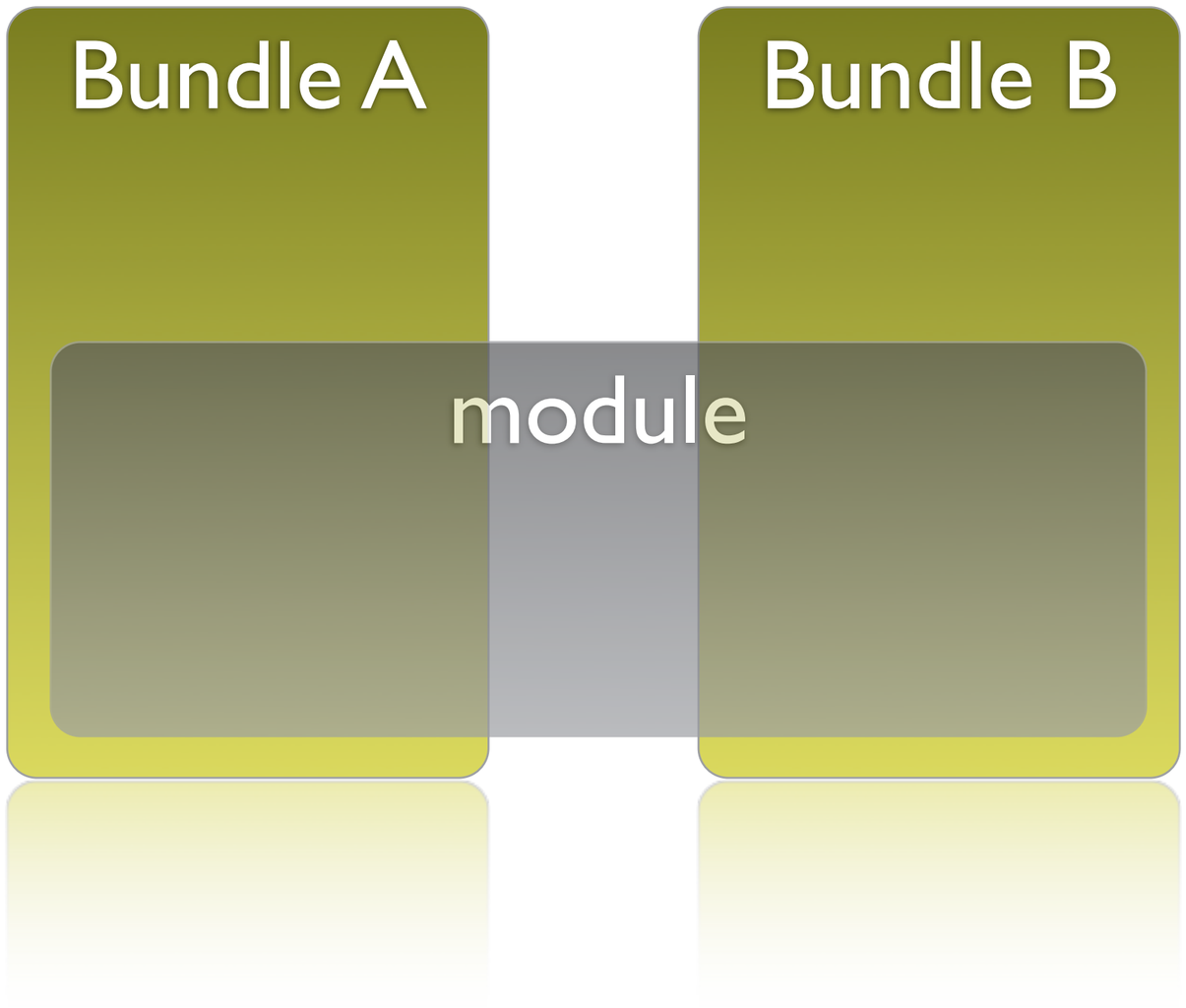
OSGi Modules: Bundle = Module



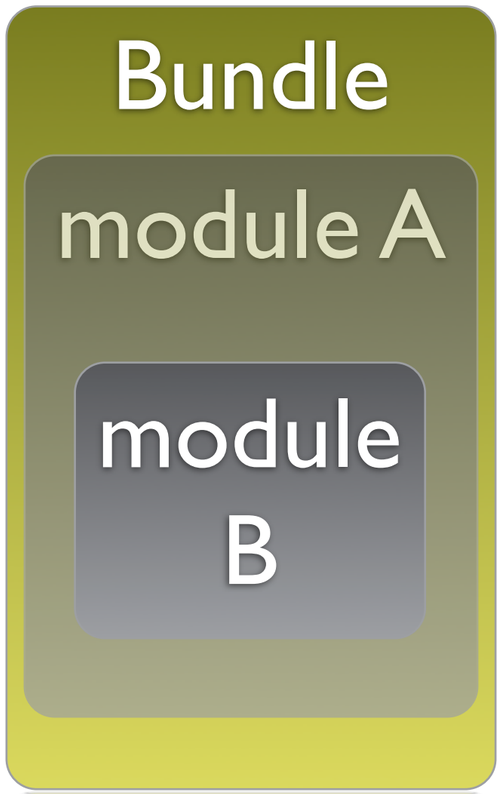
OSGi Modules: Bundle = Multiple Modules



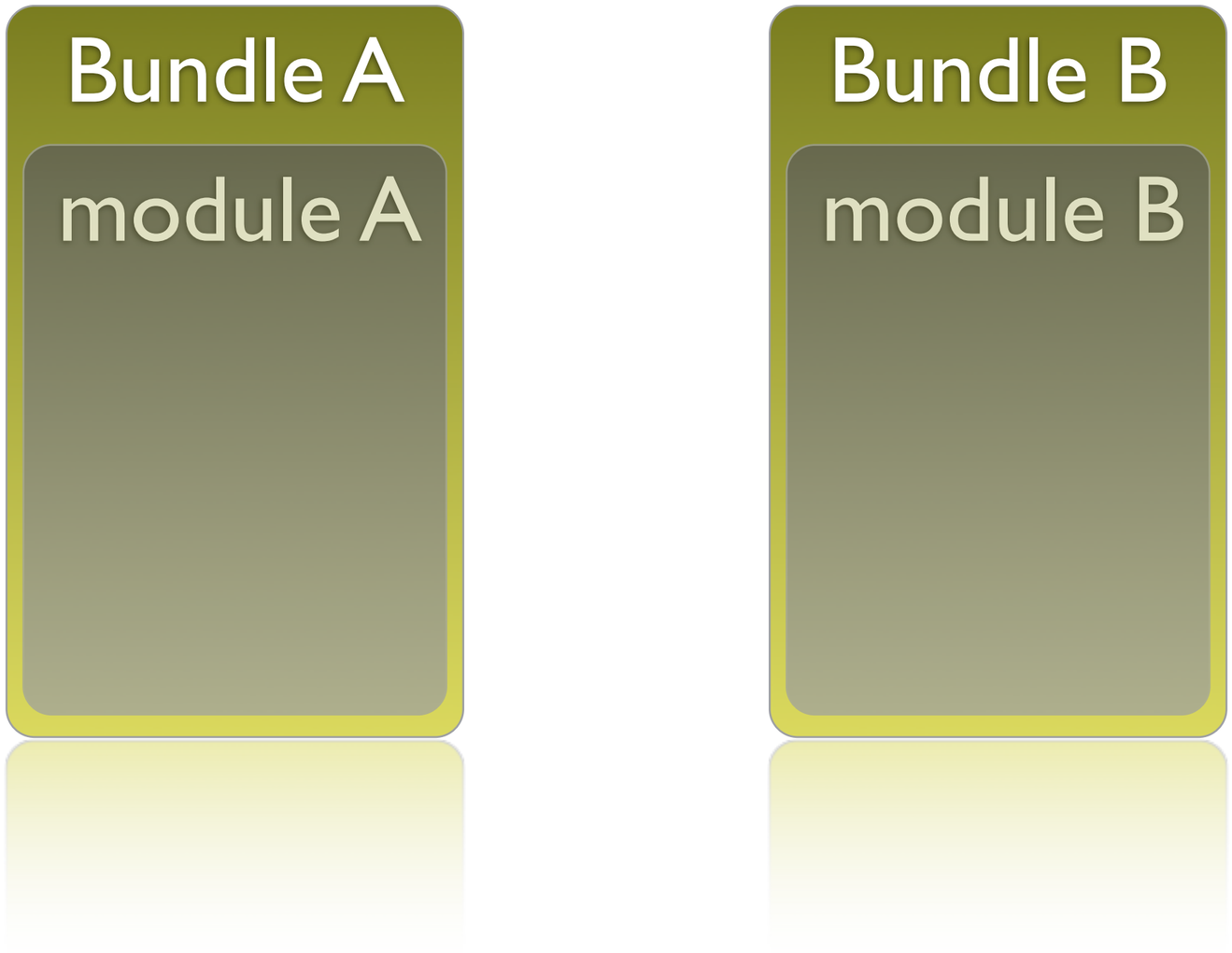
OSGi Modules: Modules Cross Bundles



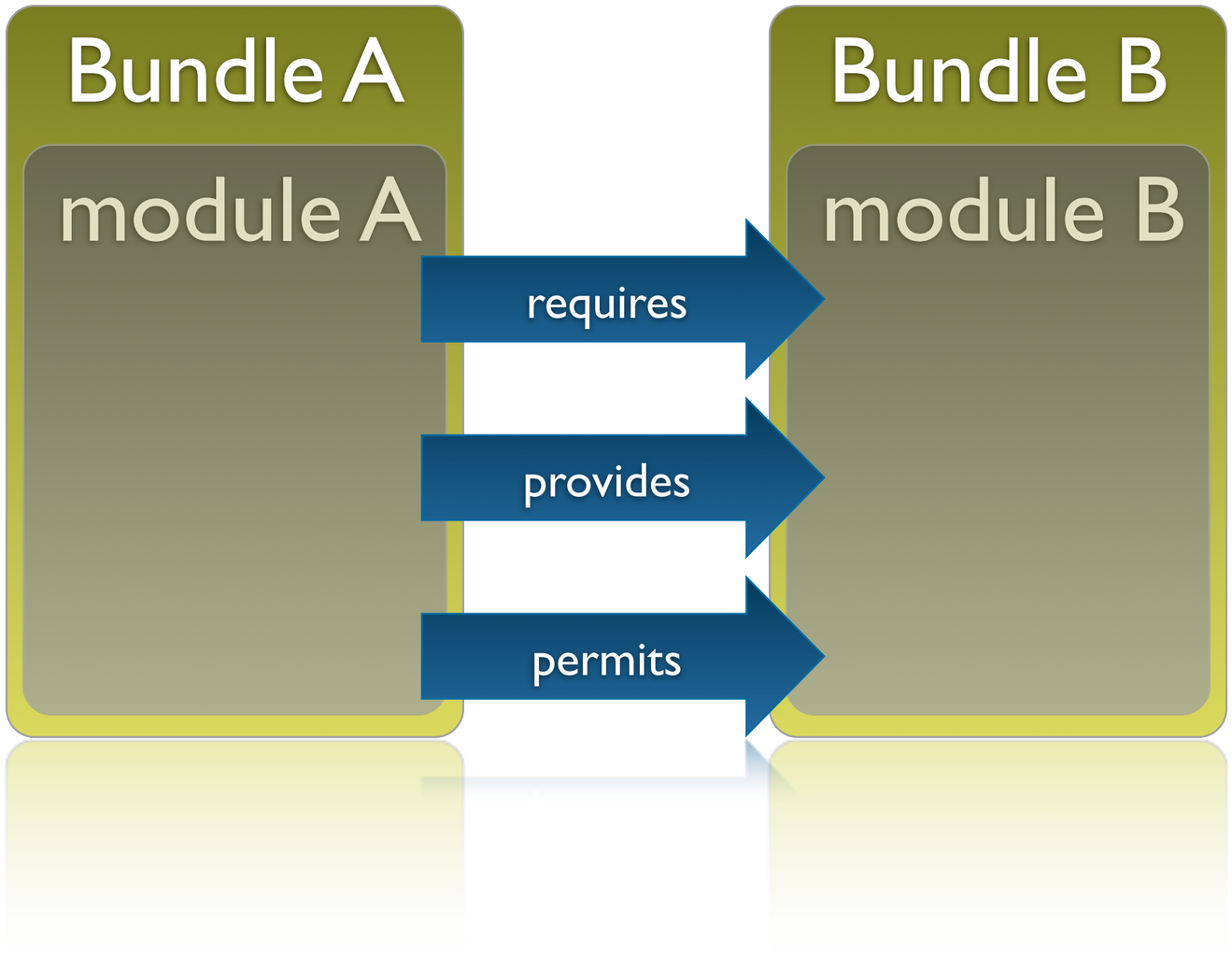
OSGi Modules: Nested?



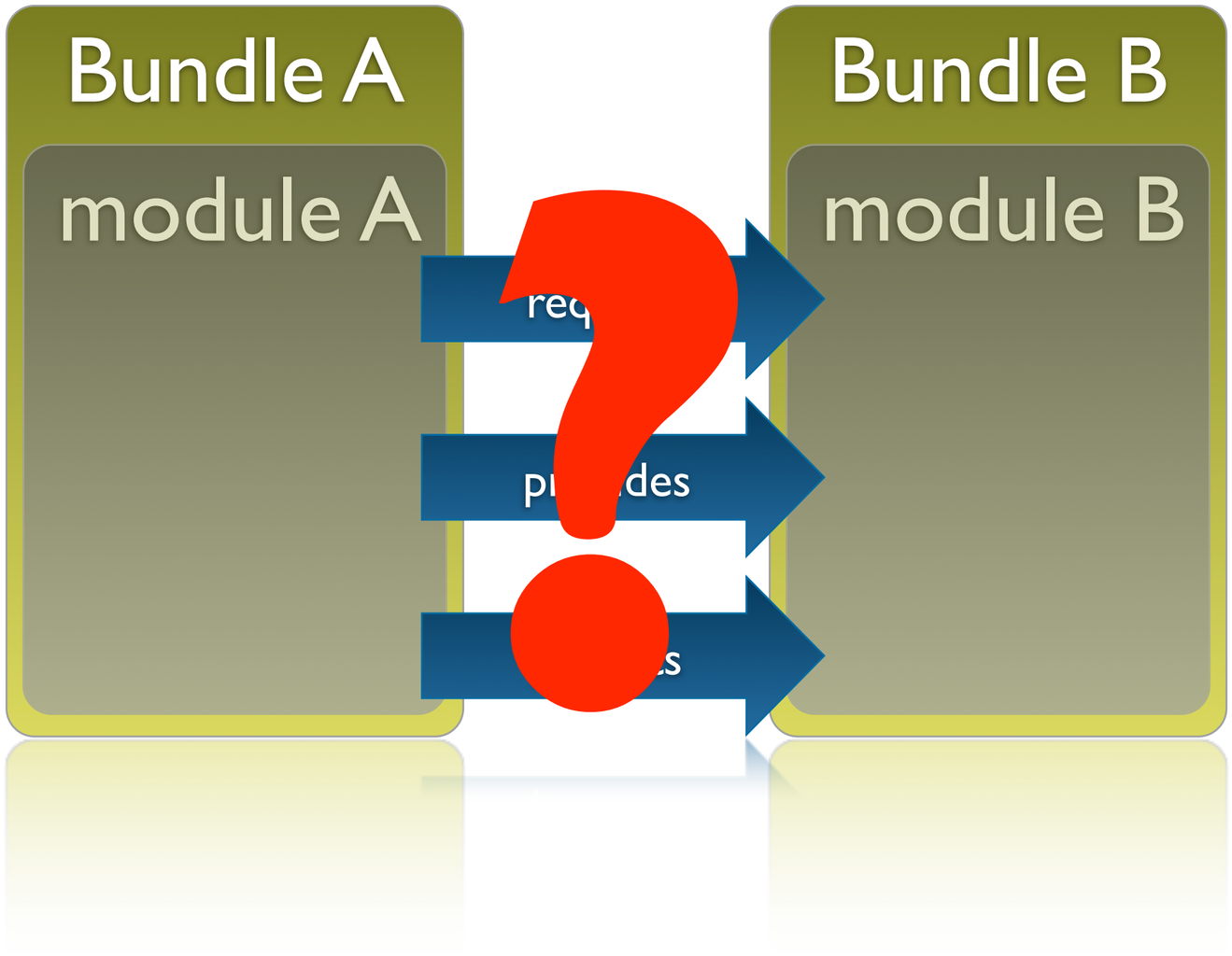
Module Dependencies

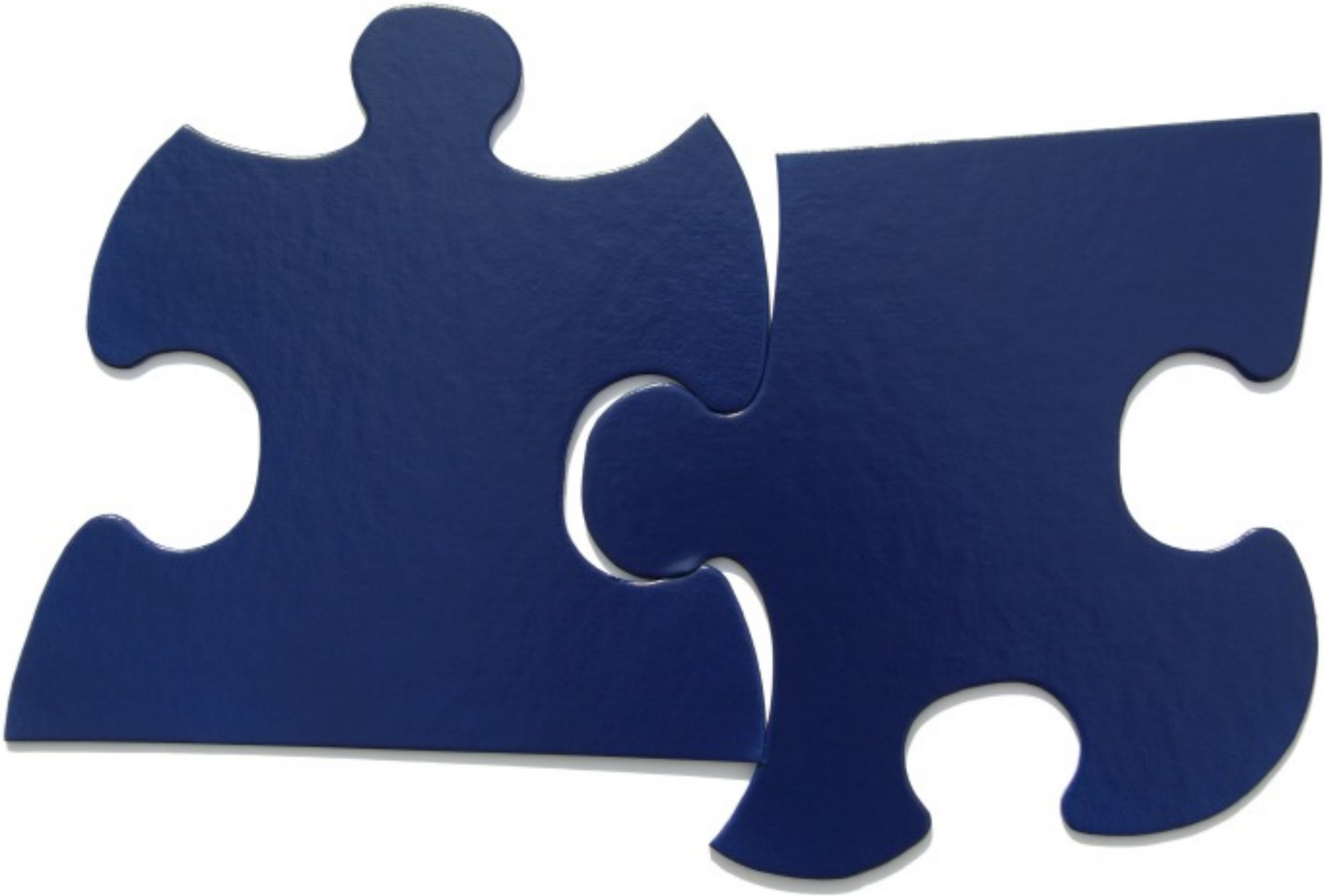


Module Dependencies



Module Dependencies





Lessons Learned

- Many bundles import unbounded versions
  - Import-Package: x; version=1.2
- Bundle Activation needed to be linked to package wiring:
  - XBundle-Activator: com.acme.Activator

Lessons Learned

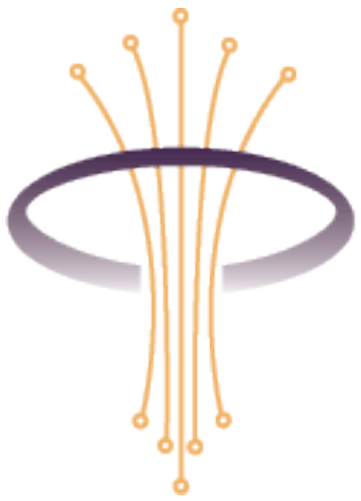
- Uses works!
  - Incompatibilities were discovered early
- The varargs ellipse (…) really cleans up the API

Lessons Learned

- JSR 294 has a long way to go
  - … if doing it right is the goal
  - Accessibility (Module keyword) is good!
  - Visibility with meta module system (Dependencies model) is too much

JavaOne℠

Thank You



OSGi™ Alliance

BJ Hargrave
Senior Technical Staff Member
IBM

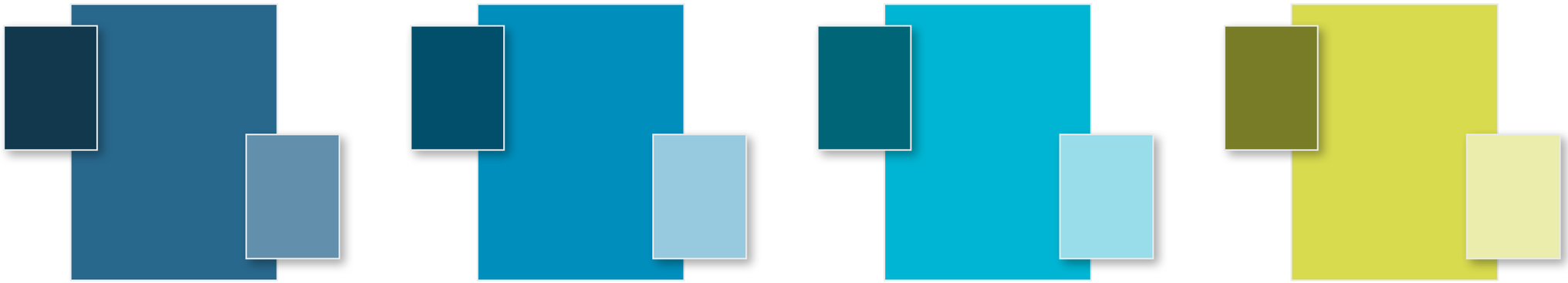
Peter Kriens
Technical Director
OSGi

Slide Heading: 52pt

Subheading: 42pt

- All text is **Helvetica**
- Level One bullet point: 50pt
  - Level Two bullet: 48pt
    - Level Three: 36pt
      - Level Four and subsequent: 32
- Text block is aligned to the left

Color Palette



Filter Builder