CHARIOT SOLUTIONS

Apt Usage of `apt`: How and When To Use the Annotation Processing Tool

John Shepard

Andrea O. K. Wright

Chariot Solutions, LLC

www.chariotsolutions.com

BOF-9385

Apt Usage of `apt`

Our goal is to ...

present ideas for tasks well suited to **`apt`** and to show how to write a processor that puts them to good use.

Agenda

Introduction to Annotation Processing

What is **`apt`**? How do you use it?

`apt` vs. Doclet, XDoclet 2, ANTLR

Practical Uses for **`apt`**

`apt` and Aspect Oriented Programming

`apt` and Design By Contract

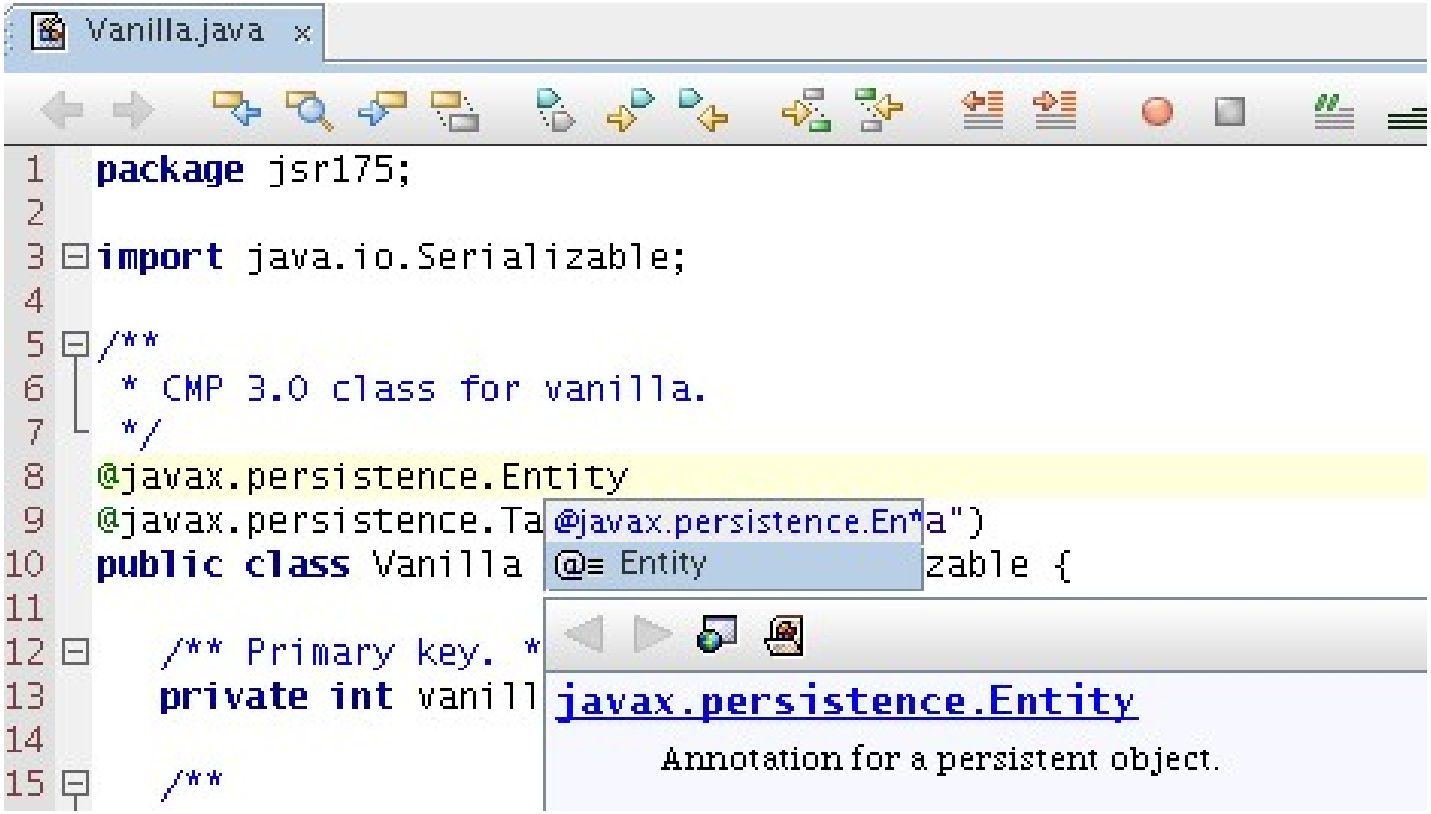
The **`apt`** track

Vintage Annotation

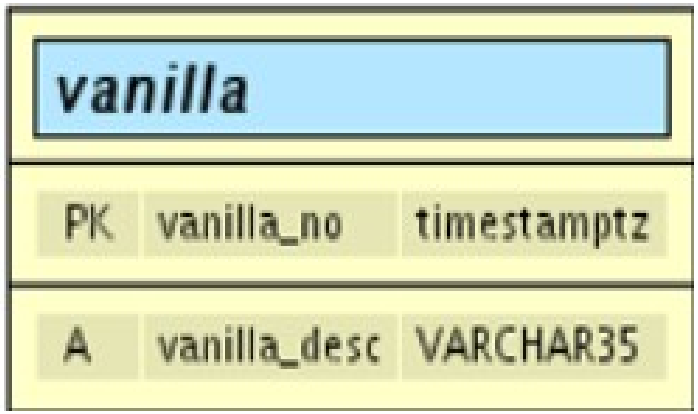
VanillaBean.java

```
1  package vintage;
2
3  /**
4   * @ejb.bean
5   *    type="CMP"
6   *    cmp-version="2.x"
7   *    name="Vanilla"
8   *    local-jndi-name="ejb/VanillaHome"
9   *    view-type="local"
10  *    primkey-field="vanillaNo"
11  *
12  * @ejb.local-interface
13  *
14  * @ejb.persistence table-name="vanilla"
15  * @ejb.transaction type="Required"
16  */
17 public abstract class VanillaBean
18     implements javax.ejb.EntityBean {
```

JSR 175 Annotation (Contemporary)



Domain Model



```
apt=> select * from vanilla;
 vanilla_no |            vanilla_desc
------------+-----------------------------------
          1 | TAHITIAN VANILLA
          2 | MADAGASCAR BOURBON VANILLA
          3 | MEXICAN VANILLA
          4 | INDIAN BOURBON VANILLA
          5 | PAPUA NEW GUINEA BOURBON VANILLA
          6 | UGANANDAN BOURBON VANILLA
```

XDoclet Interface Template

```
/**
 * Generated file - Do not edit!
 */
package <XDtPackage:packageName/>;
public interface <XDtClass:className/>Interface extends
  javax.ejb.EJBLocalObject {
<XDtTagDef:tagDef namespace="Example"
  handler="slides.xdoclet1.InterfaceHandler"/>
<XDtMethod:forAllMethods sort="true">
<XDtMethod:ifHasMethodTag tagName="ejb.interface-method">
   <XDtExample:generate/>
</XDtMethod:ifHasMethodTag>
</XDtMethod:forAllMethods>
}
```

XDoclet Processor

```
public class InterfaceHandler extends
AbstractProgramElementTagsHandler {
  public void generate(Properties attributes) throws
  XDocletException {
    StringBuffer method = new StringBuffer();
    method.append(
      getCurrentMethod().getReturnType().getType().toString());
    method.append(" ");
    method.append(getCurrentMethod().getName());
    method.append("(");
    boolean firstParameter = true;
    for (XParameter parameter :
    (List<XParameter>)getCurrentMethod().getParameters()) {
      if (firstParameter) {
        firstParameter = false;
      } else {
        method.append(", ");
      }
      method.append(parameter.getType());
      method.append(" ");
      method.append(parameter.getName());
    }
    method.append(");");
    getEngine().print(method.toString()); }}
```

Vintage Annotated Code

```
/* * @ejb.local-interface */
public abstract class VanillaBean implements
  javax.ejb.EntityBean {
   /** @ejb.interface-method */
   public abstract java.lang.Integer getVanillaNo();
   public abstract void setVanillaNo(java.lang.Integer
  vanillaNo);
   /** @ejb.interface-method */
   public abstract java.lang.String getVanillaDesc();
   /** @ejb.interface-method */
   public abstract void setVanillaDesc(java.lang.String
  vanillaDesc);
  ...
}
```

Generated Interface

```
/**
 * Generated file - Do not edit!
 */
package vintage;
public interface VanillaBeanInterface extends
javax.ejb.EJBLocalObject {
  java.lang.Integer getVanillaNo();
  java.lang.String getVanillaDesc();
  void setVanillaDesc(java.lang.String vanillaDesc);
  ...
}
```

XDoclet Ant Task

```
<xdoclet destDir="${preprocess.dir}">
  <xmlTemplate
    destinationFile="{0}Interface.java"
    templateFile="${template.dir}/interface.xdt"
    havingClassTag="ejb.local-interface"
    mergeDir="${merge.dir}"
    subtaskName="vintage.interface"
  />
  <fileset dir="${example.dir}">
    <include name="**/*.java" />
  </fileset>
</xdoclet>
```

Agenda

Introduction to Annotation Processing

What is `apt`? How do you use it?

`apt` vs. Doclet, XDoclet 2, ANTLR

Practical Uses for `apt`

`apt` and Aspect Oriented Programming

`apt` and Design By Contract

The `apt` track

APT recurses, traverses and intersperses...

-- abstract for a presentation on APT at the 2004 JAOO Conference by Joseph Darcy, Spec Lead, JSR 269, Pluggable Annotation Processing API

How do you use `apt`?

- Create a Factory
  - Implements AnnotationProcessorFactory
  - Instantiates one or more processors
- Run a Processor
  - Implements AnnotationProcessor
- Delegate to a Visitor
  - Extends SimpleDeclarationVisitor
  - Implements DeclarationVisitor
- Custom Validation; Display Messages
- Generate Code

JSR 175 Annotation

Annotation Definition

```
@Target(TYPE) @Retention(RUNTIME)
public @interface Entity {
  String name() default "";
  AccessType access() default PROPERTY;
  EntityType entityType() default CMP;
  int version() default 3;
}
```

Annotated Code

```
@javax.persistence.Entity
@javax.persistence.Table(name = "vanilla")
public class Vanilla implements Serializable {
```

Vanilla.java

```
@javax.persistence.Entity
@javax.persistence.Table(name = "vanilla")
public class Vanilla implements Serializable {
  private int vanillaNo;
  @javax.persistence.Id(generate =
  javax.persistence.GeneratorType.AUTO)
  @javax.persistence.Column(name = "vanilla_no",
  primaryKey = true)
  public int getVanillaNo() {
     return vanillaNo;
  }
  ...
}
```

Annotation Processor Factory Imports

```
import com.sun.mirror.apt.AnnotationProcessor;
import com.sun.mirror.apt.AnnotationProcessorEnvironment;
import com.sun.mirror.apt.AnnotationProcessorFactory;
import com.sun.mirror.declaration.AnnotationTypeDeclaration;
...

public class EjbFactory implements AnnotationProcessorFactory {
...
}
```

Annotation Processor Factory

```
public class YourAnnotationProcessorFactory
    implements AnnotationProcessorFactory {

  public AnnotationProcessor getProcessorFor(
   Set<AnnotationTypeDeclaration> declarations,
   AnnotationProcessorEnvironment environment) {

      return // Your AnnotationProcessor Here

   }

  public Collection<String> supportedAnnotationTypes() {
     return // Your Annotation Type(s) Here
  }

  public Collection<String> supportedOptions() {
     return // Your Supported Option(s) Here
  }
}
```

Annotation Processor Factory

```
public class EjbFactory implements AnnotationProcessorFactory {
  private static final Collection<String> ANNOTATION_TYPES =
  Collections.unmodifiableCollection(Arrays.asList(
  new String[]{"javax.persistence.Entity",
   "javax.persistence.Table",
   "javax.persistence.Id"}));

  public AnnotationProcessor getProcessorFor(
   Set<AnnotationTypeDeclaration> declarations,
   AnnotationProcessorEnvironment environment) {
    return new EjbProcessor(declarations, environment);
  }

  public Collection<String> supportedAnnotationTypes() {
    return ANNOTATION_TYPES; }

  public Collection<String> supportedOptions() {
    return Collections.emptySet(); }
}
```

Annotation Processor

```
public class YourAnnotationProcessor
  implements AnnotationProcessor {

  public void process() {
   // Your Logic for Processing Annotation and/or
   // Other Kinds of Declarations Here
  }

}
```

Annotation Processor

```
public class EjbProcessor implements AnnotationProcessor {

  private Set<AnnotationTypeDeclaration>
  annotationTypeDeclarations;
  private AnnotationProcessorEnvironment environment;

  public void process() {
    Filer filer = environment.getFiler();
    Set<Declaration> declarations = new HashSet();
    ...
    // populate declarations (package, class, method, etc)
    ...
   for (Declaration declaration : declarations) {
     declaration.accept(
     DeclarationVisitors.getSourceOrderDeclarationScanner( new
     EjbVisitor(filer), DeclarationVisitors.NO_OP));
   }
 }

}
```

SimpleDeclarationVisitor API

```
visitAnnotationTypeDeclaration(AnnotationTypeDeclaration d)
visitAnnotationTypeElementDeclaration(
  AnnotationTypeElementDeclaration d)
visitClassDeclaration(ClassDeclaration d)
visitConstructorDeclaration(ConstructorDeclaration d)
visitDeclaration(Declaration d)
visitEnumConstantDeclaration(EnumConstantDeclaration d)
visitEnumDeclaration(EnumDeclaration d)
visitExecutableDeclaration(ExecutableDeclaration d)
visitFieldDeclaration(FieldDeclaration d)
visitInterfaceDeclaration(InterfaceDeclaration d)
visitMemberDeclaration(MemberDeclaration d)
visitMethodDeclaration(MethodDeclaration d)
visitPackageDeclaration(PackageDeclaration d)
visitParameterDeclaration(ParameterDeclaration d)
visitTypeDeclaration(TypeDeclaration d)
visitTypeParameterDeclaration(TypeParameterDeclaration d)
```

From EjbVisitor

```
public void visitClassDeclaration(ClassDeclaration declaration) {
  writer.println("Methods:" + declaration.getMethods());
}

public void visitDeclaration(Declaration declaration) {
  for (AnnotationMirror mirror:
  declaration.getAnnotationMirrors()) {
    writer.println("Parameters = " + mirror.getElementValues());
    for (AnnotationTypeElementDeclaration type :
    mirror.getElementValues().keySet()) {
      AnnotationValue value =
      mirror.getElementValues().get(type);
      writer.println("Default = " + type.getDefaultValue());
      writer.println("Value = " + value);
      writer.println("Value.getValue() = " +
      value.getValue());
    }
  }
}
```

Generated EjbVisitor Output: apt.txt

```
Methods:[getVanillaNo(), setVanillaNo(int), getVanillaDesc(),
setVanillaDesc(java.lang.String)]
Parameters = {generate()=javax.persistence.GeneratorType.AUTO}
Default = javax.persistence.GeneratorType.NONE
Value = javax.persistence.GeneratorType.AUTO
Value.getValue() = AUTO
Parameters = {name()="vanilla_no", primaryKey()=true}
Default = ""
Value = "vanilla_no"
Value.getValue() = vanilla_no
Default = false
Default.getValue() = false
Value = true
Value.getValue() = true
...
```

Generated EjbVisitor Output: apt.txt

```
Methods:[getVanillaNo(), setVanillaNo(int), getVanillaDesc(),
setVanillaDesc(java.lang.String)]
Parameters = {generate()=javax.persistence.GeneratorType.AUTO}
Default = javax.persistence.GeneratorType.NONE
Value = javax.persistence.GeneratorType.AUTO
Value.getValue() = AUTO
Parameters = {name()="vanilla_no", primaryKey()=true}
Default = ""
Value = "vanilla_no"
Value.getValue() = vanilla_no
Default = false
Default.getValue() = false
Value = true
Value.getValue() = true
...
```

apt Ant Task

```
<apt srcdir="${example.dir}"
  compile="true"
  factorypathref="lib.ref"
  classpathref="lib.ref"
  preprocessdir="${preprocess.dir}"
  destdir="${build.dir}">
</apt>
```

Agenda

Introduction to Annotation Processing

What is **`apt`**? How do you use it?

`apt` vs. Doclet, XDoclet 2, ANTLR

Practical Uses for **`apt`**

`apt` and Aspect Oriented Programming

`apt` and Design By Contract

The **`apt`** track

Multiple Processors for Doclet

```
private final RootDoc root;
public void execute() {
  XmlBeanFactory factory = new XmlBeanFactory(new
  ClassPathResource("beans.xml"));

  for (String annotationProcessorName :
  factory.getBeanNamesForType(AnnotationProcessor.class)) {
    ...
    AnnotationProcessor annotationProcessor =
    (AnnotationProcessor) factory.getBean(
    annotationProcessorName);
    ...
    annotationProcessor.process(root);
  }
}
```

AbstractProcessor.java

```
private void processClassRoot(ClassDoc clazz, boolean
   innerClass) {
  for (AnnotationDesc annotation : clazz.annotations()) {
    if (isSupportedAnnotation(annotation)) {
      processClassAnnotation(clazz, annotation);
    }
  }
  for (MethodDoc method : clazz.methods()) {
    processMethod(clazz, method);
    for (AnnotationDesc annotation : method.annotations()) {
      if (isSupportedAnnotation(annotation)) {
        processMethodAnnotation(clazz, method, annotation);
      }
    }
  }
...
}
```

Doclet EjbProcessor.java

```
public class EjbProcessor extends EmptyProcessor implements
AnnotationProcessor {
  public void processClassAnnotation(ClassDoc clazz,
  AnnotationDesc annotation) {
    System.out.println("Class:" + annotation);
  }
  public void processMethodAnnotation(ClassDoc clazz,
  MethodDoc method, AnnotationDesc annotation) {
    System.out.println("Method:" + annotation);
    for (ElementValuePair pair :
    annotation.elementValues()) {
      System.out.println("Parameter:" + pair);
      System.out.println("Default Value:" +
      pair.element().defaultValue().value());
      System.out.println("Value:" + pair.value().value());
    }
  }
  ...
}
```

Doclet EjbProcessor Output

```
Class:@javax.persistence.Entity
Class:@javax.persistence.Table(name="vanilla")
Method:@javax.persistence.Id(generate=AUTO)
Parameter:generate=AUTO
Default Value:javax.persistence.GeneratorType.NONE
Value:javax.persistence.GeneratorType.AUTO
Class:@javax.persistence.Entity
Class:@javax.persistence.Table(name="recipe")
Method:@javax.persistence.Id(generate=AUTO)
Parameter:generate=AUTO
Default Value:javax.persistence.GeneratorType.NONE
Value:javax.persistence.GeneratorType.AUTO
```

Doclet ant Task

```
<javadoc
classpathref="lib.ref"
docletpathref="lib.ref"
doclet="slides.doclet.Main"
sourcefiles=
  "example/jsr175/Vanilla.java,
  example/jsr175/Recipe.java"
/>
```

XDoclet 2

- velocity
- picocontainer
- QDox – JSR 175 annotations not accessible
- No recursion

CookBookIndex Annotation

```
package slides.antlr;

import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Target(ElementType.LOCAL_VARIABLE)
@Retention(RetentionPolicy.SOURCE)
public @interface CookBookIndex {
}
```

Recipe.java

```
@javax.persistence.Entity
@javax.persistence.Table(name = "recipe")
public class Recipe implements Serializable {
   ...
   public void measureIngredients() {
        @CookBookIndex int feeds = 8;
   }
}
```

antlr Grammar (java.tree.g)

```
header {package antlr;}
class JavaTreeParser extends TreeParser;
options {
importVocab = Java;
}
{
public void processAnnotationIdentifier(AST ast) {}
}
compilationUnit
...
annotation
   :   #(ANNOTATION {processAnnotationIdentifier(_t);} identifier
   annotationInit )
;
...
```

AnnotationTreeParser.java

```
public class AnnotationTreeParser extends JavaTreeParser {
  public void processAnnotationIdentifier(AST ast) {
    System.out.println("annotation:" + unDot(ast));
  }

  public static void main(String[] args) throws
  FileNotFoundException, RecognitionException,
  TokenStreamException {
    JavaLexer lexer = new JavaLexer(new FileReader(
    "/projects/slides/example/jsr175/Recipe.java"));
    JavaRecognizer parser = new JavaRecognizer(lexer);
    ASTFactory factory = parser.getASTFactory();
    AnnotationTreeParser walker = new AnnotationTreeParser();
    parser.compilationUnit();
    CommonAST parseTree = (CommonAST)parser.getAST();
    walker.compilationUnit(parseTree);
 }
 ...
```

antlr Output

```
annotation:javax.persistence.Entity
annotation:javax.persistence.Table
annotation:javax.persistence.Id
annotation:javax.persistence.Column
annotation:javax.persistence.Column
annotation:javax.persistence.ManyToOne
annotation:javax.persistence.JoinColumn
annotation:CookBookIndex
```

Strengths of `apt`

- Provides Helper Methods
- Recurses
- Compiles generated files
- Compiles other java files
- Provides convenient hooks for custom processing

Limitations of `apt`

- Local variable annotation are not visible
- Not part of java/javax packages
- Default values and optional parameters
- No built-in support for template engine
- No tree
- Requires Java 5 or later
- Only processes Java files
- Does not access annotations in the class or at runtime (Reflection)

Agenda

Introduction to Annotation Processing

What is **`apt`**? How do you use it?

`apt` vs. Doclet, XDoclet 2, ANTLR

Practical Uses for `apt`

`apt` and Aspect Oriented Programming

`apt` and Design By Contract

The **`apt`** track

The rapt Project on java.net

rapt
Project home

If you were registered and logged in, you could join this project.

Summary	A collection of Annotations with APT processors
Categories	None
License	Berkeley Software Distribution (BSD) License
Owner(s)	brucechapman

Message from the owner(s)

Still building this project. @LongString is built and working with javadocs and jar in download area. Still working on tutorials. Want to link each Annotation to its own discussion

Description

Java 5, "Tiger" introduces annotations and the apt tool. The apt tool opens up the compiler as a runtime environment and presents many exciting opportunities.

This project seeks to develop and share useful annotations and their apt processors.

It also provides a place to explore the use of annotations and the apt tool (This is new for all of us, we will make mistakes, but share them for everyones benefit).

Initially we have annotations for multi-line String literals and XML DOM literals. We are experimenting with delegators or closure like constructs.

The Rapt Library should be thought of as a collection of little things, mostly unrelated, except that they all use or enhance the use of the apt tool in some way.

rapt: Mixins

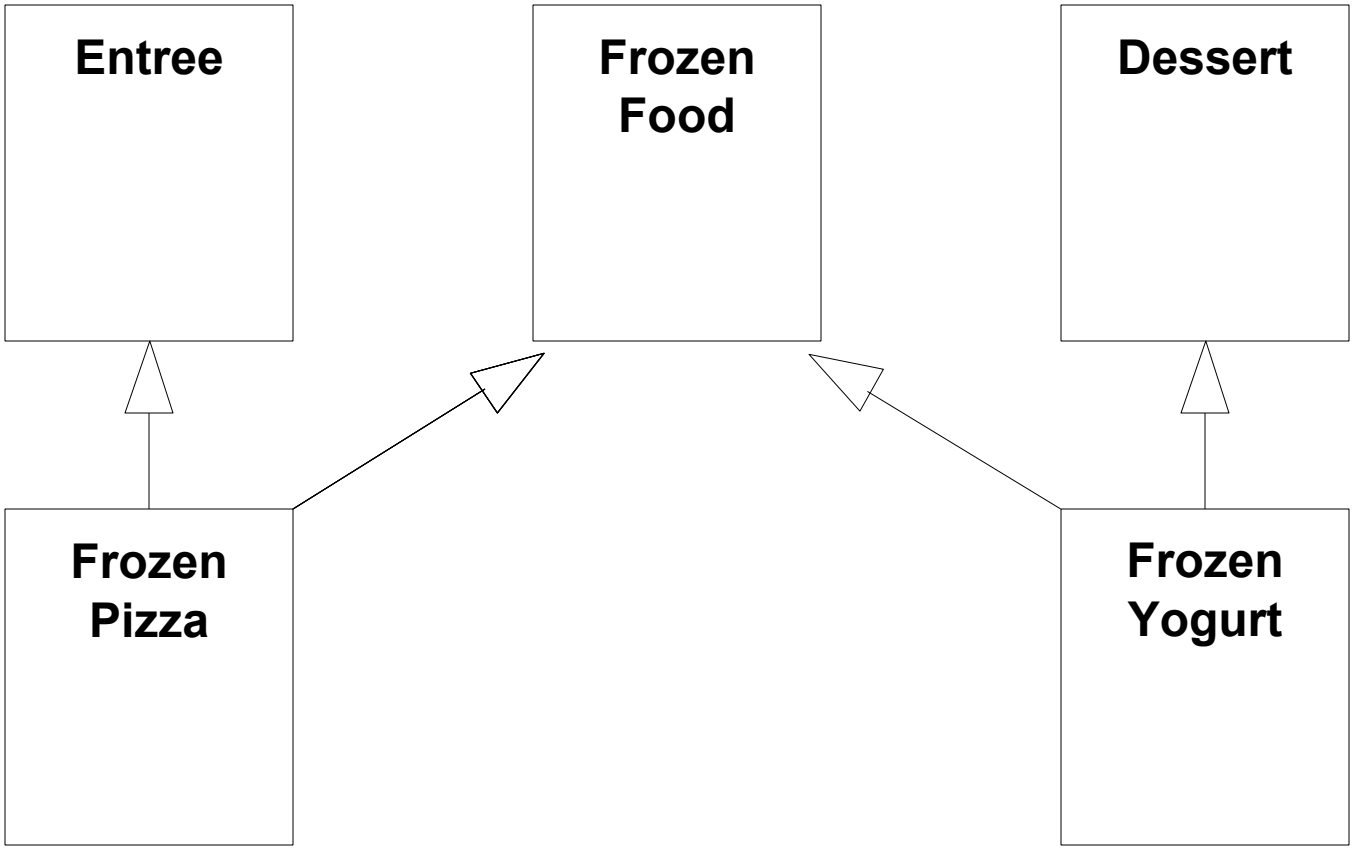
FrozenPizza.java

```
public class FrozenPizza extends Entree {
  public void freeze () {
    System.out.println("Freezing.....");
  }
}
```

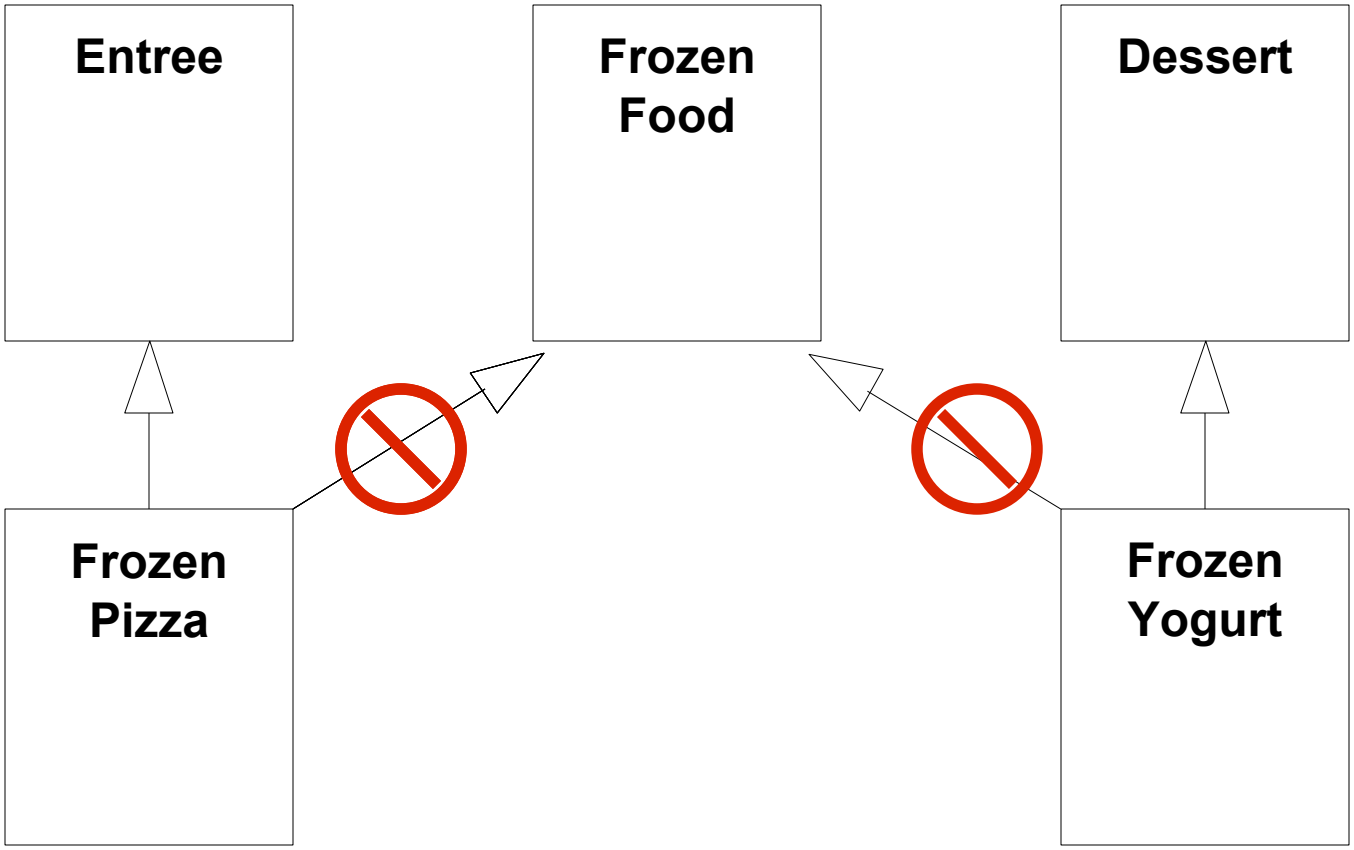
FrozenYogurt.java

```
public class FrozenYogurt extends Dessert {
  public void freeze () {
    System.out.println("Freezing.....");
  }
}
```

rapt: Mixins



rapt: Mixins



rapt: Mixins

Mixin Interface: Frozen

```
@Implementation(FrozenImpl.class)
interface Frozen extends Mixin {
  void freeze();
}
```

Mixin Implementation: FrozenImpl

```
abstract class FrozenImpl implements Frozen,
  MixinImpl<Frozen,Any> {
  public void freeze() {
    System.out.println("Freezing.....");
  }
}
```

rapt: Mixins

Mix Annotation Definition

```
@Target(ElementType.TYPE)
@Retention(RetentionPolicy.SOURCE)
public @interface Mix {
  Class<?> base();
  Flavor[] add();
}
```

Flavor Annotation Definition

```
@Target({})
public @interface Flavor {
  Class<? extends Mixin> value();
  String named() default "";
  String baseNamed() default "";
}
```

rapt: Mixins

Annotated FrozenPizza.java

```
@Mix(base=Entree.class,add=@Flavor(value=Frozen.class))
public class FrozenPizza extends Entree {
}
```

rapt: Mixins

Annotated FrozenPizza.java

```
@Mix(base=Entree.class,add=@Flavor(value=Frozen.class))
public class FrozenPizza extends FrozenEntree {
}
```

rapt: Mixins

Generated Class: FrozenEntree.java

```
class FrozenEntree extends Entree implements Frozen {
  protected FrozenEntree() {
    super();
    ...
    construct$Frozen();
   }
   ...
   // code for @Flavor(value=Frozen)
   private Frozen frozen;
   private void construct$Frozen() {
       frozen = new FrozenImpl() {
           ...
       };
   }
   public void freeze() { frozen.freeze(); }
}
```

JAX-WS 2.0 RI Early Access 2 (Formerly JAX-RPC 2.0)

Java Community Process

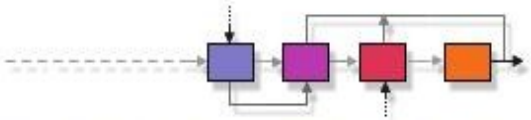
JCP Home

- What's New
- JSRs: Java Specification Requests
- JCP Procedures
- Community Resources
- Participation
- Press & Success
- Introduction

Go to JSR:

On this page:

Identification
Request
Contributions



Community Development of Java Technology Specifications

List of all JSRs JCP2 Spec Lead Guide Expert Group Login
Calendar

JSRs: Java Specification Requests

JSR 224: Java™ API for XML-Based RPC (JAX-RPC) 2.0

The JAX-RPC 2.0 specification extends the existing JAX-RPC 1.0 specification with new features.

Status: In Progress

Stage		Start	Finish
Early Draft Review 3	Download page	05 Apr, 2005	05 May, 2005
Early Draft Review 2	Download page	04 Feb, 2005	06 Mar, 2005
Early Draft Review	Download page	23 Jun, 2004	23 Jul, 2004
Expert Group Formation		10 Jun, 2003	23 Apr, 2004
JSR Review Ballot	View results	27 May, 2003	09 Jun, 2003

JCP version in use: 2.6
Java Specification Participation Agreement version in use: 2.0
Please direct comments on this JSR to: jsr-224-comments@jcp.org

JAX-WS Reference Implementation

Annotated Class: DinnerReservations.java

```
@WebService
public class DinnerReservations {
  public String makeReservations (String name,
    int sizeOfParty, Date time) {

   return "Your table is reserved under: " + name + ".";
  }
}
```

Generated MakeReservationsResponse.java

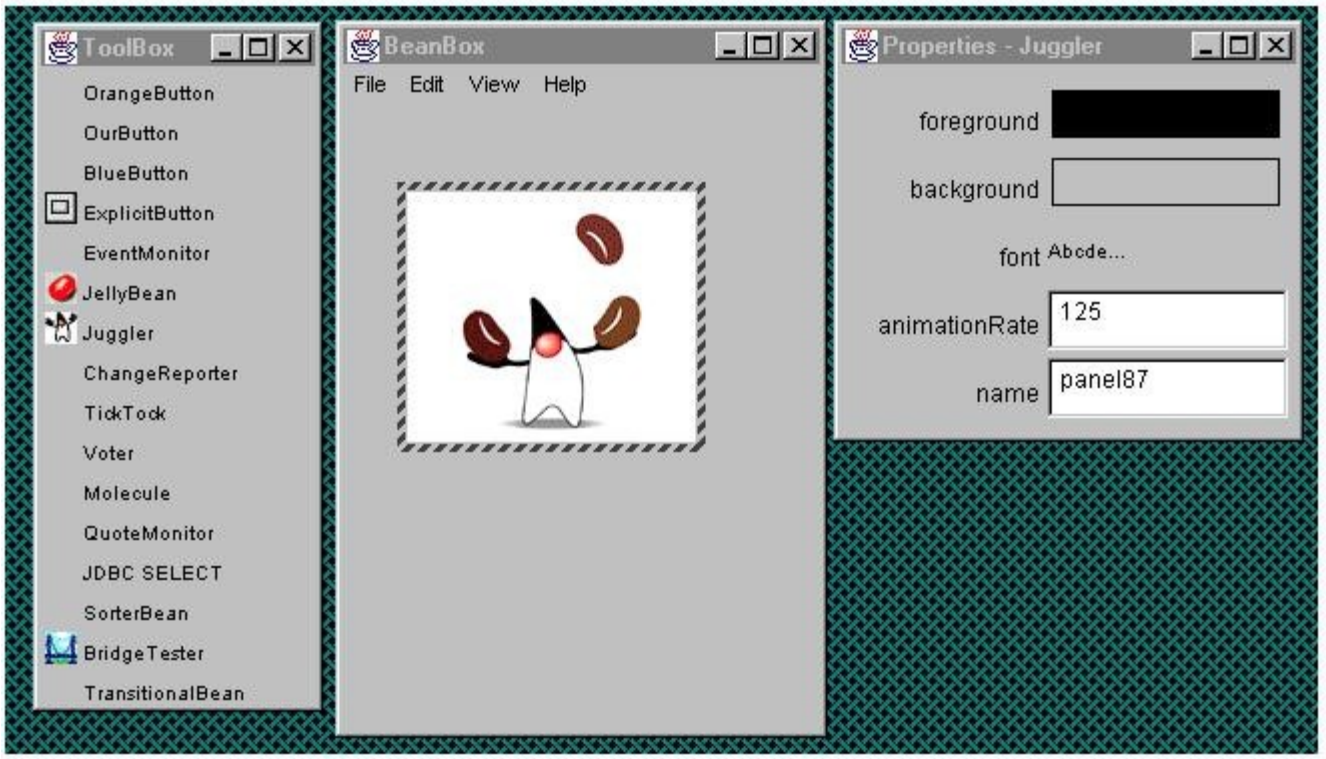
```
@XmlRootElement(name="makeReservationsResponse",
    namespace="http://jaxws")
@XmlAccessorType(AccessType.FIELD)
@XmlType(name="makeReservationsResponse",
namespace="http://jaxws",
propOrder={"_return"})
public class MakeReservationsResponse {
    @XmlElement(namespace="http://jaxws", name="return")
    @ParameterIndex(value=-1)
   public java.lang.String _return;

   public MakeReservationsResponse(){}
}
```

Generated MakeReservations.java

```
@XmlRootElement(name="makeReservations",
  namespace="http://jaxws")
@XmlAccessorType(AccessType.FIELD)
@XmlType(name="makeReservations",
namespace="http://jaxws",
  propOrder={"name", "sizeOfParty", "time"})
public class MakeReservations {
  @XmlElement(namespace="http://jaxws", name="name")
  @ParameterIndex(value=0)
  public java.lang.String name;
  @XmlElement(namespace="http://jaxws",
  name="sizeOfParty")
  @ParameterIndex(value=1)
  public java.lang.Integer sizeOfParty;
  @XmlElement(namespace="http://jaxws", name="time")
  @ParameterIndex(value=2)
  public java.util.Date time;
  public MakeReservations(){}
}
```

Apache Beehive

Beehive: Generates JavaBeans



Beehive: Client View

Placing a Takeout Order at JMStaurant

```
@Control
JMStaurant jmstaurant;
jmstaurant.placeTakeOutOrder(new TakeoutOrder("JJ", 15));
```

TakeoutOrder.java

```
public class TakeoutOrder implements java.io.Serializable {
  String customerName;
  int platterNo;

    public Order(String customerName, int platterNo) {
      this.customerName = name;
      this.platterNo = platterNo;
    }
  }
}
```

Unified APIs for Serverside Resources

JMS: Placing a Takeout Order at JMStaurant

```
@Control
JMStaurant jmstaurant;
jmstaurant.placeTakeOutOrder(new TakeoutOrder ("JJ", 15);
```

EJBs: Placing a Takeout Order at ChezEJB

```
@Control
ChezEJB chezEJB;
ejbEatery.placeTakeOutOrder(new TakeoutOrder ("JJ", 15);
```

JDBC: Placing a Takout Order at JDBSeafood

```
@Control
JDBSeafood jdbSeafood;
jdbSeafood.placeTakeOutOrder(new TakeoutOrder ("JJ", 15);
```

Control Interface

```
@ControlInterface
public interface DailySpecials  {
  public String[] getSpecials();
  public void setSpecials(String[] specials);
}
```

Control Implementation

```
@ControlImplementation
public class DailySpecialsImpl implements DailySpecials {
 private  String[] specials;
 public String[] getSpecials() {
     return specials;
 }
 public void setSpecials(String[] specials) {
     this.specials = specials;
 }
}
```

Generated DailySpecialsBean

```
public class DailySpecialsBean
extends org.apache.beehive.controls.runtime.bean.ControlBean
implements pkg.DailySpecials {
  static final Method _getSpecialsMethod;
  static final Method _setSpecialsMethod;
...
  public  void setSpecials(java.lang.String[] specials) {
    Object [] argArray = new Object[] { specials };
    Throwable thrown = null;
    pkg.DailySpecials target =
      (pkg.DailySpecials)ensureControl();
    ...
    preInvoke(_setSpecialsMethod, argArray);
    target.setSpecials(specials);
    ...
    Object rv = null;
    postInvoke(_setSpecialsMethod, argArray, rv, thrown);
  }
```

A JMS Control

JMStaurant.java

```
@ControlExtension
@JMSControl.Destination(sendJndiName="jms.SimpleJmsQ",
jndiConnectionFactory=
"weblogic.jws.jms.QueueConnectionFactory")
public interface JMStaurant extends JMSControl {

  @Message
  public void placeTakeoutOrder (TakeoutOrder order);

 }
```

A Database Control

JDBSeafood.java

```
@ControlExtension
@JdbcControl.ConnectionDataSource(jndiName=
"java:comp/env/jdbc/JdbcControlSampleDB")
public interface JDBSeafood extends JdbcControl {

 @JdbcControl.SQL(statement="INSERT INTO orders VALUES
     ({order.customerName}, {order.platterNo})")
  public void placeTakeOutOrder(TakoutOrder order)
    throws SQLException;

 @JdbcControl.SQL(statement="INSERT INTO reservations VALUES
     ({customerName}, {hour})")
  public void makeReservations(String customerName, int hour)
    throws SQLException;
}
```

Menu Configuration

DinerMenuItem.java

```
Public class DinerMenuItem extends JMenuItem() {
  int idNo;

  public void setIdNo(int idNo) {
    this.idNo = idNo;
  }

  public int getIdNo {
    return idNo;
  }

  public int hashCode () {return uniqueKey;}

  public boolean equals (Object o) {
    return o instanceof DataKey &&
              uniqueKey == ((DataKey)o).uniqueKey;
   }
}
```

XML Menu Configuration File

```
<menu>
    <name>"Breakfast"</name>
    <menuItem id=1, name="Waffles", price="1.99",
              action="BreakfastSpecialAction"/>
    <menuItem id=2,name="Pancakes",price="2.99"/>
    <menuItem name=id=3,"Scrambled Eggs",price="2.99"/>
</menu>
<menu>
    <name>"Lunch"</name>
    <menuItem id=4,name="Pizza", price="5.99",
              action="LunchSpecialAction"
     <menuItem/>
    <menuItem id=5,name="Grilled Cheese",price="2.45"/>
    <menuItem id=6,name="Pasta",price="4.99"/>
</menu>
```

Annotations for Menu Configuration

MenuInfo Annotation

```
@Retention(RetentionPolicy.SOURCE)
@Target({ElementType.FIELD})
public @interface MenuInfo {
  String name();
  MenuItemInfo[] items();
}
```

MenuItemInfo Annotation

```
@Retention(RetentionPolicy.SOURCE)
@Target({ElementType.FIELD})
public @interface MenuItemInfo {
  int id() default -1;
  String name();
  double price();
  Class action() default MenuAction;
}
```

Annotated Menu Configuration File

Diner.java

```
public class Diner {
@MenuInfo(name="Breakfast",
  items = {@MenuItemInfo(id=1,name="Waffles",price=1.99),
           @MenuItemInfo(id=2,name="Pancakes",price=2.99),
           @MenuItemInfo(id=3,name="Scrambled Eggs",price=2.99))
private JMenu breakfastMenu;

@MenuInfo(name="Lunch",
  items = {@MenuItemInfo(id=4,name="Pizza",price=5.99,
                         action=LunchSpecialAction),
           @MenuItemInfo(id=5,name="Grilled Cheese",price=2.45),
            @MenuItemInfo(id=6,name="Pasta",price=4.99)
private JMenu lunchMenu;
    ...
}
```

Generated Swing code

```
...
JMenu breakfastMenu = new JMenu("Breakfast");
DinerMenuItem menuItem = new DinerMenuItem();
menuItem.setIdNo(1);
menuItem.setAction(new MenuAction("Waffles",1.99));
breakfastMenu.add(menuItem);
menuItem = new DinerMenuItem();
menuItem.setIdNo(2);
menuItem.setAction(new MenuAction("Pancakes",2.79));
breakfastMenu.add(menuItem);
menuItem.setIdNo(3);
menuItem = new DinerMenuItem();
menuItem.setAction(new MenuAction("Scrambled Eggs",2.99));
breakfastMenu.add(menuItem);
JMenu lunchMenu = new JMenu("Lunch");
...
```

Annotated Java Configuration File

BURGERS

All Burgers are 1/3 pound 100% USDA inspected fresh Ground Beef. They include Thousand Island, Onions, Pickles, Lettuce & Tomato. Add French Fries or Side Salad $1.00 Extra

Hamburger	$2.99
Cheeseburger	$3.49
Avocado	$4.49
Green Chile & Swiss Cheese	$3.99
Mushroom	$3.99
Pastrami	$3.99
Bacon	$3.99
Bacon Cheese	$4.49
Chili	$3.99
Chili Size	$4.49
Chili Cheese	$4.49
Metro 1/4 lb. Hot Dog	$3.99
VOLCANO (Mushrooms, Onions, Green Chili, American Cheese, Swiss Cheese)	$4.99
Add Extra Cheese	$0.50
Add Any Item	$1.00
Fish	$3.49
Turkey	$2.99
Teriyaki	$3.49
Veggie/Garden	$3.99

ENTREES

All Entrees served with Roll

Fish & Chips	$6.99
Shrimp & Chips	$6.99
Grilled Breast of Chicken	$6.99
Chicken in the Basket	$6.99
Open Face Turkey Sandwich	$6.99
Pasta of the Day	$6.99
Lasagna (Veggie or Meat)	$6.99
Meat Loaf	$6.99
Hamburger Steak	$6.99

SANDWICHES

All Sandwiches are made fresh to order and include Lettuce & Tomato and your choice of Coleslaw or Potato Salad.
French Fries or Side Salad $1.00 Extra.
Bread Choices: White, Rye, Wheat, Sourdough or French Roll.

Cheese	$3.99
Ham	$3.99
Ham & Cheese	$3.99
Turkey	$4.49
Tuna	$4.29
Egg Salad	$3.99
Chicken Salad	$3.99
Pastrami	$3.99
Roast Beef	$3.99
Salami	$3.99
Veggie	$3.99
B.L.T.	$4.29
Meat Loaf	$4.49
Club	$5.99
Combo (any 3 items)	$5.99
Any Sandwich Grilled	$0.50

GRILLED

Patty Melt	$5.99
Tuna Melt	$5.99
Meat Loaf Melt	$4.99
Chicken Salad Melt	$4.99
Marinated Grilled Chicken Breast	$5.99
Reuben	$4.99

HOT DOGS

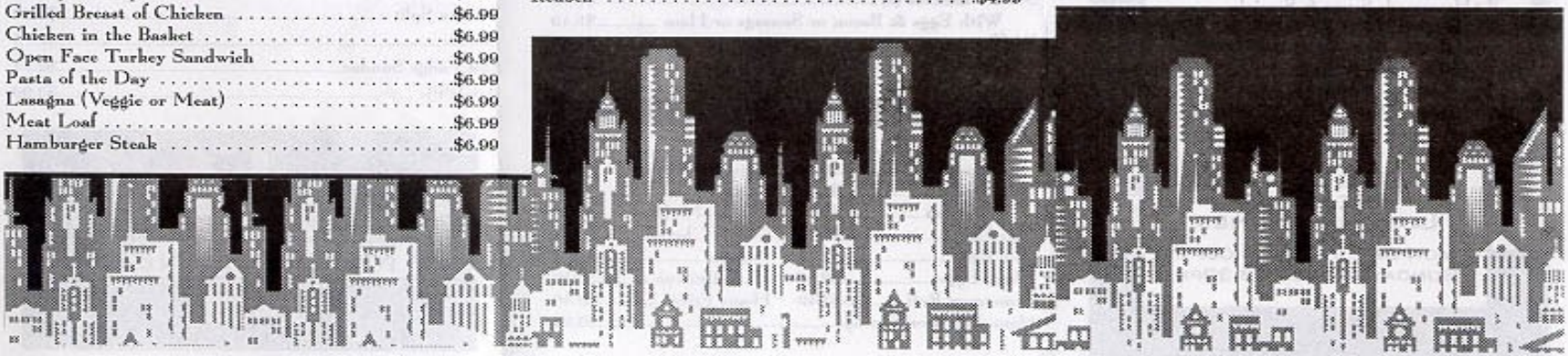
Hot Dog	$1.99
Chili Dog	$2.49
Chili Dog With Cheese	$2.99
Corn Dog	$1.99

SALADS

Dinner	$2.49
Caesar	$3.99
Caesar with Chicken Breast	$5.49
Chef	$5.49
Tuna	$3.99
Chicken	$3.99
Chinese Chicken Salad	$5.49

SIDES

Side Salad	$1.29
French Fries	$1.49
Chili Fries	$1.99
Hand Dipped Onion Rings, Zucchini or Mushrooms	$1.99
Bag of Chips	$0.79
Garlic Bread	$1.29
Baked Beans	$1.29
Mashed Potatoes	$1.99
Coleslaw	$1.29
Potato Salad	$1.29
Soup of the Day	Cup $1.95 Bowl $2.75
Chili	Cup $2.25 Bowl $2.99



Annotated Menu Configuration File

Diner.java

```
...
@MenuInfo(name="Breakfast",
          items =(id=443,name="Waffles",price=1.99),
          @MenuItemInfo(id=451,name="Pancakes",price=2.99),
          @MenuItemInfo(id=501,name="Scrambled Eggs",
          price=2.99),
          @MenuItemInfo(id=502,name="Grits",price=2.49),
          @MenuItemInfo(id=503,name="Sausage",price=1.99),
          @MenuItemInfo(id=504,name="Bacon",price=1.79),
          @MenuItemInfo(id=510,name="Bagel",price=.50),
          @MenuItemInfo(id=511,name="Toast",price=1.00),
          @MenuItemInfo(id=520,name="Danish",price=1.20),
          @MenuItemInfo(name="Donut",price=.60),
          ...
          })
private JMenu breakfastMenu;
...
```

Validation and Error Handling

```
...
if (!idNos.contains(idNo)) {
  idNos.add(idNo);
} else {
  env.getMessager().printError(menuItem.getPosition(),
   "The  id#" + idNo + " is specified more than once."));
   throw new RuntimeException("Duplicate Id#s.”);
  }
  ...
```

`apt` Messager Output

```
Diner.java:9: The id# 10 is specified more than once.
private JMenu breakfastMenu;
             ^
Problem encountered during annotation processing;
see stacktrace below for more information.
java.lang.RuntimeException: Duplicate Id#s.
        at
MenuAnnotationProcessorFactory$MenuProcessor.testIdNo(MenuAnn
otationProcessorFactory.java:171)
```

Agenda

Introduction to Annotation Processing

What is **`apt`**? How do you use it?

`apt` vs. Doclet, XDoclet 2, ANTLR

Practical Uses for **`apt`**

`apt` and Aspect Oriented Programming

`apt` and Design By Contract

The **`apt`** track

Aspect Oriented Programming: Crosscutting Concerns

- Logging
- Security
- Persistence
- Caching

Aspect Oriented Programming

Sample Aspect for AspectJ

```
public aspect PureeAspect {
   pointcut callFruitConstructor() : call(Fruit+.new(..))
   after() : callFruitConstructor() {
       System.out.println("Pureeing a " + target);
   }
}
```

AOP Term	Value
JoinPoint	`new Fruit()`
Pointcut	`callFruitConstructor`
Advice	`System.out.println("Pureeing a " + target);`
Aspect	`PureeAspect`

```
public class Apple extends Fruit {
}
```

```
public class Pear extends Fruit {
}
```

```
public class Banana extends Fruit {
}
```

```
public class Carrot extends Vegetable {
}
```

```
public class Celery extends Vegetable {
}
```

```
 Directory of C:\smoothie

06/14/2005   10:29 AM    <DIR>          .
06/14/2005   10:29 AM    <DIR>          ..
06/14/2005   10:28 AM                 5 Apple.java
06/14/2005   10:28 AM                 5 Banana.java
06/14/2005   10:28 AM                 5 Carrot.java
06/14/2005   10:28 AM                 5 Celery.java
06/14/2005   10:28 AM                 5 Pear.java
               5 File(s)             25 bytes
               2 Dir(s)  25,001,152,512 bytes free
```

FoodProcessor.java

```
private static class FoodProcessor implements
      AnnotationProcessor {
      private final AnnotationProcessorEnvironment env;
      FoodProcessor(AnnotationProcessorEnvironment env) {
          this.env = env;
      }

      public void process() {
        for (TypeDeclaration typeDeclaration :
            env.getTypeDeclarations()) {
            typeDeclaration.accept
            (getSourceOrderDeclarationScanner(
                 new FruitPicker(),
                 NO_OP));
        }
        makeSmoothie();
    }
```

FruitVisitor.java

```
private static class FruitPicker extends
  SimpleDeclarationVisitor {
    public void visitClassDeclaration (ClassDeclaration cd) {
      ClassType superclass = cd.getSuperclass();
      String superName = superclass.getQualifiedName()
      if (superName.equals("Fruit")) {
        if (!fruits.contains(d.getQualifiedName()))
          fruits.add(d.getQualifiedName());
        }
      }
    }
}
```

Pureeing Fruit for a Smoothie

```
private void makeSmoothie() {

    PrintWriter out = env.getFiler().createSourceFile
        ("Smoothie");
    out.println("public class Smoothie {");
    out.println("public Smoothie() {");

    for (Object fruit : fruits) {
        writePureeLogic(fruit);
    }
    writePureeMethod();
    out.close();

}
```

Generated File: Smoothie. java

```
public class Smoothie {

    public Smoothie() {
        Apple apple = new Apple();
        puree(apple);
        Pear pear = new Pear();
        puree(pear);
        Banana banana = new Banana();
        puree(banana);
    }

    public void puree(Fruit fruit) {
        System.out.println("Pureeing " +  fruit);
    }
}
```

Generated File: FruitSalad. java

```
public class FruitSalad {

    public FruitSalad() {
        Apple apple = new Apple();
        cube(apple);
        Pear pear = new Pear();
        cube(pear);
        Banana banana = new Banana();
        cube(banana);
    }

    public void cube(Fruit fruit) {
        System.out.println("Cubing " +  fruit);
    }
}
```

Agenda

Introduction to Annotation Processing

What is **`apt`**? How do you use it?

`apt` vs. Doclet, XDoclet 2, ANTLR

Practical Uses for **`apt`**

`apt` and Aspect Oriented Programming

`apt` and Design By Contract

The **`apt`** track

Design By Contract

- Preconditions
  - 50% deposit provided?
  - Null checks; Upper and lower value limits
- Postconditions
  - Where the requested services delivered?
  - Inspection of return value
- Invariants
  - Caterer has current Dept of Health certification
  - The internal value variable for EvenInteger is always divisible by 2

Design By Contract with `apt`

Contract Annotation Definition

```
@Retention(RetentionPolicy.SOURCE)
@Target({ElementType.METHOD})
public @interface Contract {
  String precondition () default "";
  String postcondition() default "";
  String invariant () default "";
};
```

Design By Contract with `apt`

Annotated Class

```
Public class LobsterPlatter {
 @Contract(precondition="lobster != null &&
              lobster.getWeight() > 1")
 public Lobster steamLobster(Lobster lobster) {
   lobster.setSteamed(true);
   return lobster;
 }
}
```

Design By Contract with `apt`

Generated Class: LobsterPlatterContractEnforcer.java

```
Public class LobsterPlatterContractEnforcer
    extends LobsterPlatter {


    public Lobster steamLobster(Lobster lobster) {
        if (lobster == null || lobster.getWeight < 1)
            throw new (ContractViolationException());
        return super.steam(lobster);
    }
}
```

Contract 4J

Contract4J

- Home
- What is AOP?
- *Projects:*
  - Contract4J
- White Papers
- About Us
- Contacts

Contract4J: *Design by Contract for Java*

We are pleased to announce the first release of a new open-source tool called **Contract4J**, *Contract for Java*, a t
that supports Design by Contract® (DbC) programming in Java 5, using annotations and generated AspectJ code

In DbC, the designer considers the conditions required for a module or method to successfully perform its intende
services, the *preconditions*, and what results it can guarantee to deliver, the *postconditions*, if the preconditions ar
satisfied. The designer can also specify *invariants* that must be satisfied.

Contract4J, like the original implementation of DbC in Eiffel and the XDoclet-based Barter for Java before Java 5
provides support for defining DbC tests and performing them at runtime. Whereas Barter uses Javadoc-style tags
XDoclet for generation of AspectJ test code, **Contract4J** uses Java 5 annotations to define the condition tests. Th
advantage of annotations over javadoc-style tags is the fact that the JVM can be aware of the annotations at runtim
That awareness can be used to manipulate runtime behavior, *etc.* **Contract4J** is also designed to support back-e
generation of contract test code for other aspect systems, like Spring, AspectWerkz, JBoss AOP, and others.

DbC is an underused technique for improving software quality, by thinking through design issues more carefully an
testing compliance during development and testing. It also complements other techniques very well, such as Test-D
Development (here and here).

AnnotatedClass (Contract4J)

```
@Contract
public class LobsterPlatter {

  public static Lobster lobster = new Lobster();

  public static void main (String[] args) {
          steamLobster(lobster);
   }

   public static void steamLobster(
           @Pre("lobster.weight > 1")
           Lobster lobster) {
       lobster.setSteamed(true);
       return  lobster;
   }

}
```

Generated Aspect (Contract4J)

```
static aspect ParamPrecondition_lobster_N65575 {
  before (Lobster lobster) :
  execution (* LobsterPlatter.steamLobster (Lobster)) &&
    args (lobster) {

      if ( !(lobster.weight > 1)) {
        String msg = messagePrefix + " Parameter
        Precondition failed: \"lobster.weight < 1\":  "
         + messageSuffix;
        ContractTest.assertTrue (false, msg);
      }
    }
  }
```

Agenda

Introduction to Annotation Processing

What is **`apt`**? How do you use it?

`apt` vs. Doclet, XDoclet 2, ANTLR

Practical Uses for **`apt`**

`apt` and Aspect Oriented Programming

`apt` and Design By Contract

The `apt` track

The "apt Track"

- Exploring Annotation-Based Programming Through the APT and Mirror APIs (BOF-9161)
  - 6/28 9:30 pm Moscone Center/Gateway 102/103
- Using Annotations in Large-Scale Systems: Code Automation in the "Tiger" Era (BOF-9563)
  - 6/29 9:30 pm Marriott Hotel/G Gate A3
- Annotation Processing for the Java™ Programming Language (TS-7425)
  - 6/30 10:45 am Moscone Center/N Mtg Rm121/122

For More Information

- JSR 269 Pluggable Annotation Processing API Home Page: http://jcp.org/en/jsr/detail?id=269
- JSR 175 A Metadata Facility for the Java Programming Language: http://jcp.org/en/jsr/detail?id=175
- The rapt Project: https://rapt.dev.java.net/
- Contract4J: http://www.contract4j.org/
- Apache Beehive: http://incubator.apache.org/beehive/
- These slides will be posted at www.chariotsolutions.com

Q&A

John Shepard
Andrea O. K. Wright

Chariot Solutions, LLC
www.chariotsolutions.com

CHARIOT SOLUTIONS

Apt Usage of APT: How and When To Use the Annotation Processing Tool

John Shepard

Andrea O. K. Wright

Chariot Solutions, LLC

www.chariotsolutions.com

BOF-9385