

JavaDoc doclets for Java Extension APIs

jax-doclets

0.10.0



Stéphane Épardaud (stephane@lunatech.com)

Copyright © 2009

| | |
|--|-----|
| Preface | iii |
| 1. Overview | 1 |
| 1.1. Information | 1 |
| 1.2. Example | 1 |
| 2. Running the jax-doclets | 7 |
| 2.1. Running jax-doclets in standalone | 7 |
| 2.1.1. JAXB doclet | 7 |
| 2.1.2. JAX-RS doclet | 7 |
| 2.1.3. JPA doclet | 7 |
| 2.2. Running jax-doclets in ant | 7 |
| 2.2.1. JAXB doclet | 7 |
| 2.2.2. JAX-RS doclet | 8 |
| 2.2.3. JPA doclet | 8 |
| 2.3. Running jax-doclets in Maven | 8 |
| 2.3.1. JAXB doclet | 8 |
| 2.3.2. JAX-RS doclet | 9 |
| 2.3.3. JPA doclet | 9 |
| 2.4. Doclet parameters | 10 |
| 2.4.1. Generic parameters | 10 |
| 2.4.2. JAX-RS doclet parameters | 10 |
| 2.4.3. JAXB doclet parameters | 10 |
| 2.4.4. JPA doclet parameters | 11 |
| 2.5. About stylesheets | 11 |
| 3. JAX-RS doclet documentation | 12 |
| 3.1. Where should you write JavaDoc | 12 |
| 3.2. Supported standard JavaDoc tags | 12 |
| 3.3. Supported specific JavaDoc tags | 12 |
| 3.4. Supported JAX-RS annotations | 13 |
| 3.5. Supported RESTEasy JAX-RS extension annotations | 13 |
| 4. JAXB doclet documentation | 14 |
| 4.1. Where should we write JavaDoc | 14 |
| 4.2. Supported standard JavaDoc tags | 14 |
| 4.3. Supported specific JavaDoc tags | 14 |
| 4.4. Supported JAXB annotations | 14 |
| 4.5. Mapping Java types to XML tpes | 15 |
| 5. JPA doclet documentation | 16 |
| 5.1. Supported standard JavaDoc tags | 16 |
| 5.2. Supported specific JavaDoc tags | 16 |
| 5.3. Supported JPA annotations | 16 |
| 5.4. Supported Hibernate JPA extension annotations | 16 |
| 6. License | 17 |

Preface

This is pre-release software, please bear with it.

This book is produced by the [Wikbook](#) tool. Wikbook is an open source project for converting wiki files into a set of docbook files.

1. Overview

jax-doclets allows you to generate [JavaDoc](#) documentation for specific Java annotation-based extensions such as:

- [JAX-RS](#): the RESTful API for Java
- [JAXB](#): the XML binding API for Java
- [JPA](#): the Java Persistence API

The goal of jax-doclets is to let you write documentation for your JAX-RS API, JAXB structures and JPA model in JavaDoc, where it belongs, where it is maintainable, and produce a quality JavaDoc-style documentation.

1.1 Information

jax-doclets is an open-source project maintained by [Lunatech Labs](#).

| | |
|---------------------------|---|
| Home page | http://www.lunatech-labs.com/content/jax-doclets |
| Download | https://github.com/FroMage/jax-doclets/downloads |
| Issue Tracker | https://github.com/FroMage/jax-doclets/issues |
| Source Control Management | https://github.com/FroMage/jax-doclets |

1.2 Example

Here is an example of documented JAX-RS and JAXB code:

Example 1.1. Example of documented JAX-RS and JAXB code

```
package com.lunatech.doclets.jax.test;

import javax.ws.rs.*;
import javax.xml.bind.annotation.*;

/**
 * An example JAX-RS resource
 */
@Path("/example")
@Produces( { "application/xml", "application/*+xml" })
public class JAXRSExample {

    /**
     * An example resource
     */
    @XmlRootElement
    public static class JAXBExample {

        /**
         * The resource ID
         */
        @XmlID
```

```
@XmlElement
String id;

/**
 * The example contents
 */
@XmlValue
String contents;

/**
 * An optional attribute
 */
@XmlAttribute
String type;
}

/**
 * Gets an example resource
 *
 * @param id
 *         the example id
 * @param type
 *         the type of resource we prefer
 * @param startIndex
 *         the start index
 * @return an example resource suitable for the given parameters
 * @HTTP 404 if there is no such example resource
 * @RequestHeader X-Example-Auth the authentication header
 * @ResponseHeader Location a pointer to the example details
 */
@Path("/{id}")
@GET
public JAXBExample getExample(@PathParam("id") String id,
                             @MatrixParam("type") String type,
                             @QueryParam("start") int startIndex) {
    return new JAXBExample();
}
}
```



[Overview](#) [Index](#) [Root resource](#)

SUMMARY: [RESOURCE](#) | [METHOD](#)

DETAIL: [METHOD](#)

Path: / [rest](#) / [example](#) / {id}

Gets an example resource

Path parameters:

id - the example id

| Method Summary | |
|---|--------------------------|
| Resource | Description |
| GET /rest/example/{id};type=...?start=... | Gets an example resource |

Method Detail

HTTP Example:

GET /rest/example/{id};type=...?start=...

API Example:

```
JAXRSEExample.getExample({'type': /* type
the type of resource we prefer */ ,
'start': /* startIndex the start index
*/ ,
'id': /* id the example id */});
```

Gets an example resource

Output:

[JAXRSEExample.JAXBExample](#) - an example resource suitable for the given parameters

Query parameters:

start - the start index

Matrix parameters:

type - the type of resource we prefer

Produces:

application/xml
application/*+xml

HTTP return codes:

404 - if there is no such example resource

HTTP response headers:

Location - a pointer to the example details

HTTP request headers:

X-Example-Auth - the authentication header

[Overview](#) [Index](#) [Root resource](#)

SUMMARY: [RESOURCE](#) | [METHOD](#)

DETAIL: [METHOD](#)

Generated by [Lunatech Labs jax-doclets v0.7](#)

Figure 1.1. Result of documented JAX-RS code



Overview

DETAIL: [ELEMENT](#) | [ATTRIBUTE](#) | [VALUE](#)

Name: JAXBExample

An example resource

XML Example:

```
<JAXBExample
  type="xsd:string">
  <id>xsd:ID[xsd:string]</id>
  xsd:string
</JAXBExample>
```

JSON Example:

```
{'JAXBExample':
  {
    '@type': String,
    'id': String /* ID */,
    String,
  }
}
```

ID

[id](#)

| Elements | | |
|----------|--------------------|-----------------|
| Name | Type | Description |
| id | xsd:ID[xsd:string] | The resource ID |

| Attributes | | |
|------------|------------|-----------------------|
| Name | Type | Description |
| type | xsd:string | An optional attribute |

| Value | |
|------------|----------------------|
| Type | Description |
| xsd:string | The example contents |

Overview

DETAIL: [ELEMENT](#) | [ATTRIBUTE](#) | [VALUE](#)

Generated by [Lunatech Labs jax-doclets v0.7](#)

Figure 1.2. Result of documented JAXB code

Here is an example of documented JPA code:

Example 1.2. Example of documented JPA code

```
package com.lunatech.doclets.jax.test.jpa;
```

```

import java.util.List;
import java.util.Set;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.OneToMany;
import javax.persistence.OneToOne;
import javax.persistence.Transient;

/**
 * This is the order type.
 *
 * @author stephane
 */
@Entity
public class Order {

    /**
     * This is my ID and I love it
     */
    @Id
    @GeneratedValue
    public Long id;

    /**
     * Column with an explicit name
     */
    @Column(name = "column_with_name")
    public String columnWithName;

    /**
     * Column with an annotation
     */
    @Column
    public String columnWithAnnotation;

    /**
     * Column with no annotation
     */
    public String columnWithoutAnnotation;

    /**
     * Transient property
     */
    public transient String transientColumn;

    /**
     * Transient property
     */
    @Transient
    public String transientAnnotatedColumn;

    /**
     * A Bill
     */
    @JoinColumn(name = "bill_id")
    @OneToOne
    public Bill bill;

    /**
     * A list of Orderlines
     */
    @OneToMany(mappedBy = "order")
    public List<Orderline> orderlineList;

    /**
     * A set of Orderlines
     */
    @OneToMany(mappedBy = "order")
    public Set<Orderline> orderlineSet;

    /**
     * A set of Items, via a link table
     */

```

```

@JoinTable(name = "order2item",
           joinColumns = @JoinColumn(name = "item_id"),
           inverseJoinColumns = @JoinColumn(name = "order_id"))
@ManyToMany
public Set<Item> itemSet;
}

```



Overview [Graph](#)

DETAIL: [ID COLUMN RELATION](#)

Name: Order

This is the order type.

| IDs | | |
|--------------------|------|-----------------------------|
| Name | Type | Description |
| id | long | This is my ID and I love it |

| Columns | | |
|-------------------------|----------|------------------------------|
| Name | Type | Description |
| column_with_name | varchar | Column with an explicit name |
| columnWithAnnotation | varchar | Column with an annotation |
| columnWithoutAnnotation | varchar | Column with no annotation |
| id | long[ID] | This is my ID and I love it |

| Relations | | | |
|--------------------|-----------------------------|------------|----------------------------------|
| Name | Type | Relation | Description |
| bill_id | Bill | ONE..ONE | A Bill |
| order2item.item_id | [item] | MANY..MANY | A set of Items, via a link table |
| orderlineList | [Orderline] | ONE..MANY | A list of Orderlines |
| orderlineSet | [Orderline] | ONE..MANY | A set of Orderlines |

Overview [Graph](#)

DETAIL: [ID COLUMN RELATION](#)

Generated by [Lunotech Labs lax-doclets](#) v0.9.1

Figure 1.3. Result of documented JPA code

2. Running the jax-doclets

The jax-doclets are run by JavaDoc either as standalone, via ant or using Maven.



Note

Since the JAX-RS supports links to JAXB documentation, you should first run the JAXB doclet, then the JAX-RS doclet using the [-link](#) parameter.

2.1 Running jax-doclets in standalone

2.1.1 JAXB doclet

You can use the following command to run the JAXB doclet on your code:

```
javadoc -doclet com.lunatech.doclets.jax.jaxb.JAXBDoclet \  
-docletpath lib/jax-doclets-0.10.0.jar \  
com.lunatech.doclets.jax.test
```

2.1.2 JAX-RS doclet

You can use the following command to run the JAX-RS doclet on your code:

```
javadoc -doclet com.lunatech.doclets.jax.jaxrs.JAXRSDoclet \  
-docletpath lib/jax-doclets-0.10.0.jar \  
com.lunatech.doclets.jax.test
```

2.1.3 JPA doclet

You can use the following command to run the JPA doclet on your code:

```
javadoc -doclet com.lunatech.doclets.jax.jpa.JPADoclet \  
-docletpath lib/jax-doclets-0.10.0.jar \  
com.lunatech.doclets.jax.test
```

2.2 Running jax-doclets in ant

2.2.1 JAXB doclet

You can use the following ant XML to run the JAXB doclet on your code:

```
<target depends="jars" description="Run the JAXB doclet" name="doc-jaxb">  
  <javadoc doclet="com.lunatech.doclets.jax.jaxb.JAXBDoclet" docletpath="lib/jax-doclets-0.10.0.jar">  
    <package name="com.lunatech.doclets.jax.test.*" />  
  </javadoc>
```

```
</target>
```

2.2.2 JAX-RS doclet

You can use the following ant XML to run the JAX-RS doclet on your code:

```
<target depends="jars" description="Run the JAXRS doclet" name="doc-jaxrs">
  <javadoc doclet="com.lunatech.doclets.jax.jaxrs.JAXRSDoclet" docletpath="lib/jax-doclets-0.10.0.jar">
    <package name="com.lunatech.doclets.jax.test.*" />
  </javadoc>
</target>
```

2.2.3 JPA doclet

You can use the following ant XML to run the JPA doclet on your code:

```
<target depends="jars" description="Run the JPA doclet" name="doc-jpa">
  <javadoc doclet="com.lunatech.doclets.jax.jpa.JPADoclet" docletpath="lib/jax-doclets-0.10.0.jar">
    <package name="com.lunatech.doclets.jax.test.*" />
  </javadoc>
</target>
```

2.3 Running jax-doclets in Maven

2.3.1 JAXB doclet

You can use the following Maven POM extract to run the JAXB doclet on your code:

```
<reporting>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-javadoc-plugin</artifactId>
      <version>2.7</version>
      <reportSets>
        <reportSet>
          <id>jaxb</id>
          <configuration>
            <doclet>com.lunatech.doclets.jax.jaxb.JAXBDoclet</doclet>
            <docletArtifacts>
              <docletArtifact>
                <groupId>com.lunatech.jax-doclets</groupId>
                <artifactId>doclets</artifactId>
                <version>0.10.0</version>
              </docletArtifact>
            </docletArtifacts>
          </configuration>
          <reports>
            <report>javadoc</report>
          </reports>
        </reportSet>
      </reportSets>
    </plugin>
  </plugins>
</reporting>
```

Which you run with:

```
$ mvn site
```

2.3.2 JAX-RS doclet

You can use the following Maven POM extract to run the JAX-RS doclet on your code:

```
<reporting>
<plugins>
<plugin>
<groupId>org.apache.maven.plugins</groupId>
<artifactId>maven-javadoc-plugin</artifactId>
<version>2.7</version>
<reportSets>
<reportSet>
<configuration>
<doclet>com.lunatech.doclets.jax.jaxrs.JAXRSDoclet</doclet>
<docletArtifacts>
<docletArtifact>
<groupId>com.lunatech.jax-doclets</groupId>
<artifactId>doclets</artifactId>
<version>0.10.0</version>
</docletArtifact>
</docletArtifacts>
</configuration>
<reports>
<report>javadoc</report>
</reports>
</reportSet>
</reportSets>
</plugin>
</plugins>
</reporting>
```

Which you run with:

```
$ mvn site
```

2.3.3 JPA doclet

You can use the following Maven POM extract to run the JPA doclet on your code:

```
<reporting>
<plugins>
<plugin>
<groupId>org.apache.maven.plugins</groupId>
<artifactId>maven-javadoc-plugin</artifactId>
<version>2.7</version>
<reportSets>
<reportSet>
<id>jpa</id>
<configuration>
<doclet>com.lunatech.doclets.jax.jpa.JPADoclet</doclet>
<docletArtifacts>
<docletArtifact>
<groupId>com.lunatech.jax-doclets</groupId>
<artifactId>doclets</artifactId>
<version>0.10.0</version>
</docletArtifact>
</docletArtifacts>
</configuration>
<reports>
<report>javadoc</report>
</reports>
</reportSet>
</reportSets>
</plugin>
</plugins>
</reporting>
```

```

</reportSet>
</reportSets>
</plugin>
</plugins>
</reporting>

```

Which you run with:

```
$ mvn site
```

2.4 Doclet parameters

2.4.1 Generic parameters

These parameters are valid for all jax-doclets

| Parameter | Function |
|-------------|--|
| -stylesheet | The CSS stylesheet to copy and use. |
| -header | The header which is inserted on every page header. |
| -footer | The footer which is inserted on every page footer. |
| -charset | The charset to use for source files and produced HTML documentation. |
| -link | Path to another JavaDoc documentation. This is used to produce links to other package's documentation, either regular JavaDoc or to JAXB documentation in the case of the JAX-RS doclet. |

2.4.2 JAX-RS doclet parameters

These parameters are only valid for the JAX-RS doclet

| Parameter | Function |
|---------------------------|---|
| -jaxrscontext url | The URL path to your RESTful API, if there is a prefix prepended to it on your deploy site. |
| -disablehttpexample | Disables the generated HTTP examples. |
| -disablejavascriptexample | Disables the generated JavaScript examples. |

2.4.3 JAXB doclet parameters

These parameters are only valid for the JAXB doclet

| Parameter | Function |
|----------------------|--|
| -disablejsontypename | If you want to hide the serialised type name from the JSON examples. |
| -disablexmlexample | Disables the generated XML examples. |
| -disablejsonexample | Disables the generated JSON examples. |

2.4.4 JPA doclet parameters

There are currently no special parameters for the JPA doclet.

2.5 About stylesheets

If you do not specify a stylesheet, one will be provided for you that closely matches the default JavaDoc style. If you do specify a stylesheet with the `-stylesheet` parameter, it will be copied to the stylesheet named `doclet.css` and the default JavaDoc stylesheet will be available as `default-doclets.css` which means you can write your stylesheet like this to extend the default stylesheet without having to restyle everything:

```
@IMPORT url("default-doclet.css");

table.info .TableCaption, td.NavBarCell11 {
  background: #EEEEEE;
}

td.NavBarCell11 table th.selected {
  background-color: #61911B;
}

img.logo {
  margin: 1em;
  border: none;
}

a {
  color: #61911B;
}

a:visited {
  color: #3F5E12;
}
```

Note that wherever possible we've stuck with the JavaDoc CSS class names so you can reuse your existing JavaDoc stylesheets.

3. JAX-RS doclet documentation

The JAX-RS doclet generates documentation for your RESTful service based on JAX-RS annotations and JavaDoc comments on your JAX-RS resource methods.

3.1 Where should you write JavaDoc

JavaDoc is read either on the JAX-RS resource methods, or their interface. Only method-level JavaDoc is used. Documentation for a given RESTful URL is taken from the method annotated with `@GET`, `@HEAD`, `@POST`, `@PUT` or `@DELETE` for that URL (in order of preference).

JAX-RS resource locators are supported.



Note

Since the JAX-RS supports links to JAXB documentation, you should first run the JAXB doclet, then the JAX-RS doclet using the `-link` parameter.

3.2 Supported standard JavaDoc tags

The following standard JavaDoc tags are supported on resource methods:

| Tag | Function |
|------------------------------|---|
| <code>@param name doc</code> | This is used to document the corresponding resource method parameters annotated with <code>@PathParam</code> , <code>@QueryParam</code> or <code>@MatrixParam</code> . Can be used at most once per parameter name. |
| <code>@return doc</code> | Documents the entity returned from this resource method. Can only be used once. |

3.3 Supported specific JavaDoc tags

The following specific JavaDoc tags are supported on resource methods:

| Tag | Function |
|--|---|
| <code>@HTTP code doc</code> | This is used to document the codes that the method can return. Can be used multiple times. |
| <code>@inputWrapped fq-classname</code> | Specifies the real type of input when declared as a <code>String</code> parameter. Can only be used once. |
| <code>@returnWrapped fq-classname doc</code> | Used in place of <code>@return</code> when output type is <code>String</code> , <code>void</code> or <code>Response</code> to specify the real type of output and documentation for each possible type. Can be used multiple times. |

| Tag | Function |
|---|---|
| <code>@RequestHeader</code> header doc | This is used to document HTTP request headers. Can be used multiple times. |
| <code>@ResponseHeader</code> header doc | This is used to document HTTP response headers. Can be used multiple times. |
| <code>@include</code> file | Includes the specified relative file in the resource documentation. Can be used multiple times. |

3.4 Supported JAX-RS annotations

The following standard JAX-RS annotations are supported on resource methods or classes:

- `@Path`
- `@PathParam`
- `@FormParam`
- `@CookieParam`
- `@HeaderParam`
- `@QueryParam`
- `@MatrixParam`
- `@Produces`
- `@Consumes`
- `@Context` (ignored)

3.5 Supported RESTEasy JAX-RS extension annotations

If the optional RESTEasy dependency is present, the following RESTEasy annotations are supported on resource methods or classes:

- `@Form`

4. JAXB doclet documentation

The JAXB doclet generates documentation for your XML schema based on JAXB annotations and JavaDoc comments on your JAXB classes.

4.1 Where should we write JavaDoc

JavaDoc is read either on the JAXB properties (getter methods or fields), or their interface (only for the getters) as well as on the JAXB classes.



Note

Since the JAX-RS supports links to JAXB documentation, you should first run the JAXB doclet, then the JAX-RS doclet using the [-link](#) parameter.

4.2 Supported standard JavaDoc tags

There are no standard JavaDoc tags supported. Everything comes from JavaDoc comments.

4.3 Supported specific JavaDoc tags

There are no specific JavaDoc tags supported.

4.4 Supported JAXB annotations

The following standard JAXB annotations are supported on properties or classes:

- `@XmlAccessorType`
- `@XmlRootElement`
- `@XmlElement`
- `@XmlElementWrapper`
- `@XmlAttribute`
- `@XmlValue`
- `@XmlID`
- `@XmlIDREF`
- `@XmlTransient` (ignored)

4.5 Mapping Java types to XML types

The following Java types have a special mapping in XML:

| Type | XML mapping |
|---|---------------------------|
| <code>java.lang.String</code> | <code>xsd:string</code> |
| <code>java.lang.Character</code> , <code>char</code> | <code>xsd:string</code> |
| <code>java.lang.Date</code> | <code>xsd:datetime</code> |
| <code>java.lang.Integer</code> , <code>int</code> | <code>xsd:int</code> |
| <code>java.lang.Long</code> , <code>long</code> | <code>xsd:long</code> |
| <code>java.lang.Short</code> , <code>short</code> | <code>xsd:short</code> |
| <code>java.lang.Byte</code> , <code>byte</code> | <code>xsd:byte</code> |
| <code>java.lang.Float</code> , <code>float</code> | <code>xsd:float</code> |
| <code>java.lang.Double</code> , <code>double</code> | <code>xsd:double</code> |
| <code>java.lang.Boolean</code> , <code>boolean</code> | <code>xsd:boolean</code> |
| <code>java.lang.Object</code> | <code>xsd:any</code> |
| <code>java.lang.Enum</code> | List of enum values |
| <code>java.util.Collection</code> | <code>xsd:list</code> |

Any other type is taken to be either a Java type or a JAXB type, for whom proper links will be generated.

5. JPA doclet documentation

The JPA doclet generates documentation for your data model based on JPA annotations and JavaDoc comments on your JPA classes.

5.1 Supported standard JavaDoc tags

There are no standard JavaDoc tags supported. Everything comes from JavaDoc comments.

5.2 Supported specific JavaDoc tags

There are no specific JavaDoc tags supported.

5.3 Supported JPA annotations

The following standard JPA annotations are supported on classes or properties:

- `@Entity`
- `@Id`
- `@ManyToMany`
- `@OneToMany`
- `@ManyToOne`
- `@OneToOne`
- `@Table`
- `@Transient`
- `@GeneratedValue`
- `@Id`
- `@Lob`

5.4 Supported Hibernate JPA extension annotations

If the optional Hibernate dependency is present, the following Hibernate annotations are supported on classes or properties:

- `@GenericGenerator`

6. License

jax-doclets is distributed under the LGPL license. It does not distribute any thirdparty libraries that are GPL. It does ship thirdparty libraries licensed under Apache ASL 2.0 and LGPL.