The Java EE 5 Tutorial

Previous: Header Example **Next**: Attachments Example

DOM and DOMSource Examples

The examples `DOMExample.java` and `DOMSrcExample.java` show how to add a DOM document to a message and then traverse its contents. They show two ways to do this:

- `DOMExample.java` creates a DOM document and adds it to the body of a message.
- `DOMSrcExample.java` creates the document, uses it to create a `DOMSource` object, and then sets the `DOMSource` object as the content of the message's SOAP part.

You will find the code for `DOMExample` and `DOMSrcExample` in the following directory:

tut-install/javaeetutorial5/examples/saaj/dom/src/

Examining the `DOMExample` Class

`DOMExample` first creates a DOM document by parsing an XML document. The file it parses is one that you specify on the command line.

```
static Document document;
...
    DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
    factory.setNamespaceAware(true);
    try {
        DocumentBuilder builder = factory.newDocumentBuilder();
        document = builder.parse( new File(args[0]) );
        ...
```

Next, the example creates a SOAP message in the usual way. Then it adds the document to the message body:

```
        SOAPBodyElement docElement = body.addDocument(document);
```

This example does not change the content of the message. Instead, it displays the message content and then uses a recursive method, `getContents`, to traverse the element tree using SAAJ APIs and display the message contents in a readable form.

```
public void getContents(Iterator iterator, String indent) {

    while (iterator.hasNext()) {
        Node node = (Node) iterator.next();
        SOAPElement element = null;
        Text text = null;
        if (node instanceof SOAPElement) {
            element = (SOAPElement)node;
            QName name = element.getElementQName();
            System.out.println(indent + "Name is " + name.toString());
            Iterator attrs = element.getAllAttributesAsQNames();
            while (attrs.hasNext()){
                QName attrName = (QName)attrs.next();
                System.out.println(indent + " Attribute name is " +
                     attrName.toString());
                System.out.println(indent + " Attribute value is " +
                     element.getAttributeValue(attrName));
            }
            Iterator iter2 = element.getChildElements();
            getContents(iter2, indent + " ");
        } else {
            text = (Text) node;
            String content = text.getValue();
            System.out.println(indent + "Content is: " + content);
            }
```

```
        }
    }
```

Examining the `DOMSrcExample` Class

`DOMSrcExample` differs from `DOMExample` in only a few ways. First, after it parses the document, `DOMSrcExample` uses the document to create a `DOMSource` object. This code is the same as that of `DOMExample` except for the last line:

```
static DOMSource domSource;
...
try {
    DocumentBuilder builder = factory.newDocumentBuilder();
    Document document = builder.parse(new File(args[0]));
    domSource = new DOMSource(document);
    ...
```

Then, after `DOMSrcExample` creates the message, it does not get the header and body and add the document to the body, as `DOMExample` does. Instead, `DOMSrcExample` gets the SOAP part and sets the `DOMSource` object as its content:

```
// Create a message
SOAPMessage message = messageFactory.createMessage();

// Get the SOAP part and set its content to domSource
SOAPPart soapPart = message.getSOAPPart();
soapPart.setContent(domSource);
```

The example then uses the `getContents` method to obtain the contents of both the header (if it exists) and the body of the message.

The most important difference between these two examples is the kind of document you can use to create the message. Because `DOMExample` adds the document to the body of the SOAP message, you can use any valid XML file to create the document. But because `DOMSrcExample` makes the document the entire content of the message, the document must already be in the form of a valid SOAP message, and not just any XML document.

Building and Running the DOM and DOMSource Examples

When you run `DOMExample` and `DOMSrcExample`, you can specify one of two sample XML files in the directory *`tut-install`*`/javaeetutorial5/examples/saaj/dom/`:

- `slide.xml`, a file that consists only of a message body
- `domsrc.xml`, an example that has a SOAP header (the contents of the `HeaderExample` SOAP 1.1 output) and the same message body as `slide.xml`

You can use either of these files when you run `DOMExample`. You can use `domsrc.xml` to run `DOMSrcExample`.

To build the programs using NetBeans IDE, follow these steps:

1. In NetBeans IDE, choose Open Project from the File menu.
2. In the Open Project dialog, navigate to *`tut-install`*`/javaeetutorial5/examples/saaj/`.
3. Select the `dom` folder.
4. Select the Open as Main Project check box.
5. Click Open Project.

   A Reference Problems dialog appears. Click Close.
6. Right-click the `dom` project and choose Resolve Reference Problems.
7. In the Resolve Reference Problems dialog, select the first of the missing JAR files and click Resolve.

   The missing files are `activation.jar`, `javaee.jar`, and `appserv-ws.jar`.
8. Navigate to the *`as-install`*`/lib/` directory.
9. Select the missing JAR file (`activation.jar`, for example) and click Open.

   In the Resolve Reference Problems dialog, all the files have green check marks to the left of their names.
10. Click Close.
11. Right-click the project and choose Build.

To run `DOMExample` using NetBeans IDE, follow these steps:

1. Right-click the `dom` project and choose Properties.
2. Select Run from the Categories tree.
3. Click Browse next to the Main Class field.

4. In the Browse Main Classes dialog, select `DomExample` .
5. Click Select Main Class.
6. In the Arguments field, type the following:

```
slide.xml
```

7. Click OK.
8. Right-click the project and choose Run.

To run `DOMSrcExample` using NetBeans IDE, follow these steps:

1. Right-click the `dom` project and choose Properties.
2. Select Run from the Categories tree.
3. Click Browse next to the Main Class field.
4. In the Browse Main Classes dialog, select `DomSrcExample` .
5. Click Select Main Class.
6. In the Arguments field, type the following:

```
domsrc.xml
```

7. Click OK.
8. Right-click the project and choose Run.

To run the examples using Ant, go to the directory *tut-install*`/javaeetutorial5/examples/saaj/dom/` .

To run `DOMExample` using Ant, use the following command:

```
ant run-dom -Dxml-file=slide.xml
```

To run `DOMSrcExample` using Ant, use the following command:

```
ant run-domsrc -Dxml-file=domsrc.xml
```

When you run `DOMExample` using the file `slide.xml` , you will see output that begins like the following:

```
    Running DOMExample.
    Name is slideshow
     Attribute name is author
     Attribute value is Yours Truly
     Attribute name is date
     Attribute value is Date of publication
     Attribute name is title
     Attribute value is Sample Slide Show
     Content is:
...
```

When you run `DOMSrcExample` using the file `domsrc.xml` , you will see output that begins like the following:

```
    Running DOMSrcExample.
    Header contents:
    Content is:

    Name is {http://gizmos.com/NSURI}orderDesk
     Attribute name is SOAP-ENV:actor
     Attribute value is http://gizmos.com/orders
    Content is:
   ...
```

If you run `DOMSrcExample` with the file `slide.xml` , you will see runtime errors.

Previous: Header Example **Next**: Attachments Example