ANDROID CORE JAVA DESKTOP JAVA ENTERPRISE JAVA JAVA BASICS JVM LANGUAGES SOFTWARE DEVELOPMENT DEVOPS

Home » Enterprise Java » ws-soa » Metro Web Service Example

ABOUT BYRON KIOURTZOGLOU

Byron is a master software engineer working in the IT and Telecom domains. He is an applications developer in a wide variety of applications/services. He is currently acting as the team leader and technical architect for a proprietary service creation and integration platform for both the IT and Telecom industries in addition to a in-house big data real-time analytics solution. He is always fascinated by SOA, middleware services and mobile development. Byron is co-founder and Executive Editor at Java Code Geeks.

Metro Web Service Example

Posted by: Byron Kiourtzoglou in ws-soa November 11th, 2012

```
package com.wordpress.jdevel.ws;

import java.io.File;
import java.io.FileFilter;
import java.io.FilenameFilter;
import java.util.ArrayList;
import java.util.List;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

@WebService(serviceName = "Music")
public class Music {

    private static final File FOLDER = new File("D:/TEMP/SONGS");

    @WebMethod(operationName = "listSongs")
    public Song[] listSongs(@WebParam(name = "artist") String artist) {


  List<Song> songs = new ArrayList<Song>();

  System.out.println("ARTIST: " + artist);

  if (artist != null) {


File folder = new File(FOLDER, artist);


if (folder.exists() && folder.isDirectory()) {


    File[] listFiles = folder.listFiles(new FilenameFilter() {




  public boolean accept(File dir, String name) {




return name.toUpperCase().endsWith(".MP3");




  }


    });



    for (File file : listFiles) {



  String fileName = file.getName();



  String author = file.getParentFile().getName();



  int size = (int) (file.length() / 1048576); //Megabytes



  songs.add(song);


    }


}

  }


  return songs.toArray(new Song[songs.size()]);
    }

    @WebMethod(operationName = "listArtists")
    public String[] listArtists() {

  File[] folders = getFolders(FOLDER);

  List<String> artists = new ArrayList<String>(folders.length);

  for (File folder : folders) {


artists.add(folder.getName());

  }

  return artists.toArray(new String[artists.size()]);
    }

    private File[] getFolders(File parent) {

  FileFilter filter = new FileFilter() {



public boolean accept(File pathname) {


    return pathname.isDirectory();


}

  };


  File[] folders = parent.listFiles(filter);


  return folders;
    }

    public static void main(String[] args) {

  Music listFiles = new Music();

  String[] artists = listFiles.listArtists();

  System.out.println("Artists: " + artists);

  for (String artist : artists) {


Song[] listSongs = listFiles.listSongs(artist);


for (Song song : listSongs) {


    System.out.println(song.getArtist() + " : " + song.getFileName() + " : " + song.getSize() + "MB");


}

  }
    }
}
```

Needed also a simple bean to get some more complex types:

```
package com.wordpress.jdevel.ws;

import java.io.Serializable;

public class Song implements Serializable {
    String fileName;
    String artist;
    int size;

    public Song() {
    }

    public Song(String fileName, String artist, int size) {

  this.fileName = fileName;

  this.artist = artist;

  this.size = size;
    }
```

```java
    }

    public void setArtist(String artist) {

  this.artist = artist;
    }

    public String getFileName() {

  return fileName;
    }

    public void setFileName(String fileName) {

  this.fileName = fileName;
    }

    public int getSize() {

  return size;
    }

    public void setSize(int size) {

  this.size = size;
    }
}
```

To make it a web service all you have to do is annotate class with @WebService(serviceName = “Music”) and every method you want to expose as web service operation has to be marked with @WebMethod(operationName = “listArtists”).

This should be all if you’re deploying it on GlassFish, but I’ve used Tomcat, so 3 more steps were needed:

1. Add Metro 2.0 jars to WEB-INF/lib

2. Add Metro servlet and listener to web.xml:

```xml
<listener>
    <listener-class>
        com.sun.xml.ws.transport.http.servlet.WSServletContextListener
    </listener-class>
</listener>
<servlet>
    <servlet-name>Music</servlet-name>
    <servlet-class>com.sun.xml.ws.transport.http.servlet.WSServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>Music</servlet-name>
    <url-pattern>/Music</url-pattern>
</servlet-mapping>
```

You probably shouldn’t change anything here. Just paste it to your web.xml in web-app node.

3. Add sun-jaxws.xml file to WEB-INF with endpoint declaration:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<endpoints version="2.0" xmlns="http://java.sun.com/xml/ns/jax-ws/ri/runtime">
    <endpoint implementation="com.wordpress.jdevel.ws.Music" name="Music" url-pattern="/Music"/>
</endpoints>
```

- implementation has to match your @WebService class
- name has to match serviceName in @WebService annotation
- url-pattern has to match url-pattern you have declared in servlet mapping

There should also be no need to edit these xml files if you create it in NetBeans.

Now start Tomcat and deploy your app. You should be able to access your service via something like

http://localhost:8080/WSServer/Music

Related Article:

- Web Services in Ruby, Python and Java

Reference: Web Services in Ruby, Python and Java from our JCG partner at the “Development world stories” blog

Tagged with: ENTERPRISE JAVA WS SOA

4. Java Annotations Tutorial
5. Java Interview Questions
6. Spring Interview Questions
7. Android UI Design

and many more ....

Email address:

Your email address

☑ Receive Java & Developer job alerts in your Area

Sign up

Leave a Reply

Be the First to Comment!

Notify of new follow-up comments Email ›

Start the discussion

NEWSLETTER

weekly updates and complimentary whitepapers!

Join them now to gain exclusive access to the latest news in the Java world, as well as insights about Android, Scala, Groovy and other related technologies.

Email address:

Your email address

☑ Receive Java & Developer job alerts in your Area

Sign up

JOIN US

With **1,240,600** monthly unique visitors and over **500** authors we are placed among the top Java related sites around. Constantly being on the lookout for partners; we encourage you to join us. So If you have a blog with unique and interesting content then you should check out our **JCG** partners program. You can also be a **guest writer** for Java Code Geeks and hone your writing skills!

KNOWLEDGE BASE

- Courses
- Minibooks
- News
- Resources
- Tutorials

HALL OF FAME

- Android Alert Dialog Example
- Android OnClickListener Example
- How to convert Character to String and a String to Character Array in Java
- Java Inheritance example
- Java write to File Example
- java.io.FileNotFoundException – How to

ABOUT JAVA CODE GEEKS

JCGs (Java Code Geeks) is an independent online community focused on creating the ultimate Java to Java developers resource center; targeted at the technical architect, technical team lead (senior developer), project manager and junior developers alike. JCGs serve the Java, SOA, Agile and Telecom communities with daily news written by domain experts, articles, tutorials, reviews, announcements, code snippets and open source projects.

DISCLAIMER

All trademarks and registered trademarks appearing on Java Code Geeks are the property of their respective owners. Java is a trademark or registered trademark of Oracle Corporation in the United States and other countries. Examples Java Code

- .NET Code Geeks
- Java Code Geeks
- System Code Geeks
- Web Code Geeks

- – How to handle Array Index Out Of Bounds Exception
- java.lang.NoClassDefFoundError – How to solve No Class Def Found Error
- JSON Example With Jersey + Jackson
- Spring JdbcTemplate Example