JAX-WS

（この文書は EMBL-EBI の http://www.ebi.ac.uk/Tools/webservices/tutorials/06_programming/java/soap/jax-ws の和訳です。このドキュメントは、12/10/02 時点の情報にもとづいて書かれています。このドキュメントは ecobioinfo.com で独自に訳したもので、EMBL-EBI とは直接関係ありません。）

はじめに

JAX-WS [1)] 標準仕様は、多くの SOAP ツールキットによって実装されています。ほとんどの場合、私たちが JAX-WS について話題にするのは標準仕様そのものより JAX-WS リファレンス実装 [2)]のことを言っています。

注：JAX-WS は RPC/encoded スタイルをサポートしていません。そのため、いくつかの EMBL-EBI Web サービスでは使えません。SOAP スタイルについてのより詳しい情報は解説の SOAP と WSDL のセクションをご覧ください。

インストール

JAX-WS は Java 6 プラットフォームの一部として含まれているため、Java 6 以降がインストールされている場合に利用できます。もし Java 5 を使っているなら、JAX-WS は http://jax-ws.java.net/からダウンロードできます。

EMBL-EBI サンプルクライアント依存関係

JAX-WS ベースのサンプルクライアントのために、追加で要求される依存関係（例えば、Apache commons-cli [3)]）を含むアーカイブを用意しています：lib-1.4.zip

これは、クライアント実行のときには -Djava.ext.dirs オプションを使って実行できます。例えば仮に lib-1.4.zip アーカイブがカレントディレクトリに解凍（lib サブディレクトリを生成）されて、サンプルの WSDbfetch クライアント jar がカレントディレクトリにあるとすると、クライアントは：

```
java -Djava.ext.dirs=lib -jar WSDbfetch_JAXWS.jar
```

で実行できます。

wismport を使ったスタブの作成

wsimport ツールは、WSDL インターフェース定義からのサービスインターフェースのクライアントクラスの生成に使われます。デフォルトでは、生成された.class ファイルは、WSDL で使われている名前空間に基づいたパッケージに配置されます。クラス生成で特別なパッケージを指定するためには -p オプションを使用してください。

EBI Web Services で利用可能な WSDL 文書の詳細は、興味のあるサービスのサービスパッケージを参照してください。

WSDL 文書からのスタブ生成のために、特別な Apache ant [4] と Apache Maven [5] タスクが利用できます。詳細は JAX-WS ドキュメントを参照してください。

コマンドライン

WSDL からスタブを生成するには、この場合は EB-eye ですが、wsimport コマンドを使ってください：

```
wsimport -d bin -p uk.ac.ebi.webservices.ebeye -keep -s src
http://www.ebi.ac.uk/ebisearch/service.ebi?wsdl
```

この場合は、スタブは uk.ac.ebi.www.ebeye Java パッケージ（-p）に作られ、生成されたクラスは bin/ ディレクトリ（-d）、生成されたソースコードは保持され（-keep）、ソースは src/ ディレクトリ（-s）に配置されます。

Apache ant: generic task

追加タスクのインストール無しで、Apache ant ビルドツールを使ってスタブをビルドするために、WSDL 文書からスタブコードを生成する包括的なターゲットの定義ができます：

```
<!-- Generate JAX-WS stubs from a WSDL -->
<!--
  Using the wsimport command:
  wsimport -d bin -p uk.ac.ebi.webservices.ebeye -keep -s src
http://www.ebi.ac.uk/ebisearch/service.ebi?wsdl
  NB: assumes wsimport is on the current PATH.
-->
<target name="genJaxwsStubs"
  description="JAX-WS RI: generate stubs from a WSDL using wsimport
command">
  <echo message="${stubPkg}" />
  <echo message="${wsdlUrl}" />
  <exec executable="wsimport">
    <arg value="-d" />
    <arg value="${build.dir}" />
    <arg value="-p" />
    <arg value="${stubPkg}" />
    <arg value="-keep" />
    <arg value="-s" />
    <arg value="${src.dir}" />
    <arg value="-Xnocompile" />
    <arg value="${wsdlUrl}" />
  </exec>
</target>
```

これは wsimport コマンドが使えるように PATH 設定されていることを前提としています。

そして genJaxwsStubs ターゲットを、スタブ生成のためにサービス固有ターゲットのパラメータで呼

び出すことができます。例えば WSDbfetch では:

```
<!-- Generate JAX-WS stubs for WSDbfetch -->
<target name="jaxws-stubs-wsdbfetch" description="JAX-WS RI: generate
stubs for WSDbfetch (document/literal SOAP)">
  <antcall target="genJaxwsStubs">
    <param name="stubPkg"
value="uk.ac.ebi.webservices.jaxws.stubs.wsdbfetch" />
    <param name="wsdlUrl"
value="http://www.ebi.ac.uk/ws/services/WSDbfetchDoclit?wsdl" />
  </antcall>
</target>
```

次に、スタブ生成のために ant を呼び出すときターゲットを参照させることができます。

```
ant jaxws-stubs-wsdbfetch
```

この方法を利用して JAX-WS のスタブを生成する Apache ant build.xml ファイルは、練習プロジェクト: java_exercises.zip に含まれています。この仕組みを使うことによって、プロジェクトは Apache Ivy configuration によって操作される最少セットの依存関係を持つことになります(ant 用の Ivy プラグインは、ビルド処理の間にダイナミックにダウンロードされインストレーションに追加されます)。

練習1:サービススタブを生成

練習プロジェクトに含まれる Apache ant ビルドファイル(build.xml)を調べてください。多くの ant タスクが見られ、それらはグループ:

- 環境設定
- ビルドディレクトリの消去
- 依存関係アーカイブを獲得して解凍
- WSDL から JAX-WS スタブを生成
- ソースコードのコンパイル

に分けられます。

Eclipse で ant ビルドファイルに記述されたタスクを実行するためには、build.xml ファイルを右クリック、“実行”→”Ant ビルド...”で必要とするターゲットを選択してください。

JAX-WS のサンプルをビルドするには 2 つのターゲットを実行する必要があります:

1. jaxws-stubs:コードスタブの生成
2. jaxws-compile:生成されたスタブとクライアントコードのコンパイル

これらを Eclipse で実行するには:

1. build.xml ファイルを右クリックしてメニューを表示
2. 利用可能なターゲットを表示するダイアログを開くため、“実行”→”Ant ビルド...”を選択
3. スタブ生成のために jaxws-stubs ターゲットを選択
4. “順序...”ボタン使って、jaxws-stubs を“compile”の前へ
5. “実行”ボタンをクリックして ant ビルドを呼び出し
6. 生成されたコードを Eclipse に確実に認識させるためにプロジェクトをリフレッシュ:

Ⅰ. プロジェクト（例えば EBIWS_Java_Exercises）を右クリック
Ⅱ. “リフレッシュ”を選択

これらのターゲットをコマンドラインから実行するには：

```
ant jaxws-stubs
ant jaxws-compile
```

Apache ant: WsImport task

Apache ant では、プラグインメカニズムを通して固有のタスクを追加することが許されています。WsImport タスク（http://jax-ws.java.net/nonav/2.2.5/docs/UsersGuide.html#3.1.2.1_Generate_a_Service_Endpoint_Interface ）WSDL 文書からの JAX-WS スタブの生成をサポート、そして、もしインストールされているなら、上の一般タスクと入れ替えて使うことができます。

Apache maven

Maven プラグインも WSDL 文書から JAX-WS コードスタブを生成するために利用可能です。詳細は、http://jax-ws-commons.java.net/jaxws-maven-plugin/ を参照してください。

スタブの利用

wsimport によって生成されたスタブは、WSDL で記述され相当する Java 言語に変換されたインターフェースとデータタイプの直接的なマッピングです。生成されたコードに目を通すことと、WSDbfetch service documentation と http://www.ebi.ac.uk/ws/services/WSDbfetchDoclit?wsdl は以下のセクションで役立つでしょう。

プログラムに生成されたクラスをインポートしてください（uk.ac.ebi.webservices.wsdbfetch パッケージを想定）：

```
import uk.ac.ebi.webservices.wsdbfetch.*;
```

サービスと交信するサービスプロキシを生成するためには：

```
WSDBFetchDoclitServerService service = new WSDBFetchDoclitServerService();
WSDBFetchServer srvProxy = service.getWSDbfetchDoclit();
```

サービスプロキシのメソッド（srvProxy）は WSDL で定義されたものとマップして、WSDbfetch サービスの fetchData メソッドを呼出して使用できます：

```
String result = srvProxy.fetchData("UNIPROT:ADH1A_HUMAN", "default", "raw");
```

fetchData メソッドは、要求されたデータベースエントリを含む文字列を返します。これは：

```
System.out.println(result);
```

での出力と同じになります。

そして通常の方法でコンパイルします：

1. Eclipse のような IDE の使用：
   - “自動的にビルド”が有効なら、クラスはソースファイルが保存されたときにコンパイルされます。
   - 別の方法としては、”プロジェクト”,”すべてビルド”を使ってプロジェクト内の全てのソースをコンパイルできます。
2. Apache ant と提供されている build.xml ファイルを使う：

```
ant jaxws-compile
```

3. コマンドラインから Java コマンドを使う：

```
javac uk/ac/ebi/webservices/wsdbfetch/*.java
javac Dbfetch.java
```

そして動作確認のためプログラムを実行します：

1. Eclipse のような IDE を使う：ソースファイル（例えば Dbfetfch.java）を右クリック、”実行”で”Java アプリケーション”を選択する。
2. コマンドラインから：

```
java Dbfetch
```

練習 2：WSDbfetch（document/literal SOAP）

WSDbfetch Web サービス（http://www.ebi.ac.uk/Tools/webservices/services/dbfetch）は、dbfetch と同等の基本的な機能性を提供していますが、SOAP インターフェースのコレクションを通しています。JAX-WS は RPC/encoded SOAP サービスをサポートしていないので、document/literal SOAP を使わなければなりません。

サンプルの WSDbfetch クライアント（examples/soap/jaxws/Dbfetch.java）を参考として利用して、アクセッション: P13569, P26361, P35071 で、fasta シーケンス書式で、UniProtKB エントリを獲得するサービスを使ってみてください。

解答例： solutions/soap/jaxws/Q4Dbfetch.java

練習 3：WSDbfetch メタデータ

WSDbfetch サービスには多くのメタデータが含まれていて、利用できるデータベース名と、それぞれのデータベースで使われるフォーマットとスタイルのようなサービスについての情報を提供しま

す(http://www.ebi.ac.uk/Tools/webservices/services/dbfetch を参照)。

サンプルクライアント(examples/soap/jaxws/Dbfetch.java)から始めて、WSDbfetch サービスの getDbFormats() メソッドを使って、UniProtKB データベースに利用可能なデータフォーマットは何かを判断してください。

解答例:solutions/soap/jaxws/Q5Dbfetch.java

複雑なデータ構造

WSDbfetch service のメソッドは全て簡単な文字列のパラメータを使っています。多くのほかの EBI サービスは、より複雑な入力データ構造を使っています。例えば、**NCBI BLAST (SOAP)** では run(email, title, params)メソッドへ渡す多様なパラメータと入力シーケンスを含んだ構造が求められます:

```
// Object factory for creating objects to be exchanged with the web
service.
ObjectFactory objFactory = new ObjectFactory();
// Populate input parameters structure
InputParameters params = new InputParameters();
params.setProgram("blastp");
ArrayOfString databaseList = new ArrayOfString();
databaseList.getString().add("uniprotkb_swissprot");
params.setDatabase(databaseList);
params.setStype("protein");
params.setSequence(objFactory.createInputParametersSequence(">Seq1 Example
search sequence\n"
  + "MKFLILLFNILCLFPVLAADNHGVGPQGASGVDPITFDINSNQTGPAFLTAVEMAGV"));
String email = "email@example.org"; // Your e-mail address
// Get the NCBI BLAST (SOAP) service proxy
JDispatcherService_Service service = new JDispatcherService_Service();
JDispatcherService ncbiblast = service.getJDispatcherServiceHttpPort();
// Submit the job
String jobid = ncbiblast.run(email, "", params);
System.out.println("Job Id: " + jobid);
```

注:(minOccurs="0") と nillable (nillable="true") の両方の sequence パラメータが選択できるので、JAXBElement<String> オブジェクトとマップされ、オブジェクトが createInputParametersSequence メソッド を使って ObjectFactory から利用可能となります。これは 3 通りの表現で起こり得る状態の全てが許されることになります:

1. 明示的な値、例えば、
   params.setSequence(objFactory.createInputParametersSequence("UNIPROT:WAP_RAT"));
2. Null 値、例えば、
   params.setSequence(objFactory.createInputParametersSequence(null));
3. **値がない、例えば** params.setSequence(null);

wsimport でカスタマイズバインディングを使ってこの振る舞いを変更する方法の詳細は The WSIT Tutorial の Customizations for WCF Service WSDL セクションを参照してください。

run(email, title, params) メソッドは、ジョブのステータス（例えば、RUNNING, FINISHED, ERROR）を獲得する getStatus(jobId) メソッド、終了したジョブの利用可能な結果の詳細を獲得する getResultTypes(jobId) メソッド、そして、ジョブの結果を獲得する getResult(jobId, type) で使えるジョブ識別子を返します。

```
// Poll until job has finished
String status = "RUNNING";
while (status.equals("RUNNING")) {
  Thread.sleep(3000); // Wait 3 seconds
  status = ncbiblast.getStatus(jobid); // check job status
  System.out.println(status);
}
// If the job completed successfully...
if (status.equals("FINISHED")) {
  // Get the text result
  byte[] resultbytes = ncbiblast.getResult(jobid, "out", null);
  String result = new String(resultbytes);
  // Output the result
  System.out.println(result);
}
```

練習 4：NCBI BLAST（SOAP）

今まで見てきた全てのサービスは比較的短い時間内に結果を返信することができました。結果が返信されるまでに極端な場合は何日もかかる分析ツールを実行するときには、このケースは当てはまりません。そのため、処理申請でその後にジョブの状態をチェックするため使うジョブ識別子を返信して、終了したら結果を獲得するように、これらのツールには非同期メソッドを使わなければならなりません。シーケンス相同性検索処理に適している NCBI BLAST を使った NCBI BLAST (SOAP) サービスにはこのアプローチが要求されます。
サンプルクライアント（examples/soap/jaxws/NcbiBlastSoap.java ）をよく調べて、”run”, ”getStatus”, ”getResult”の処理の流れに注目してください。

サンプルクライアントを参考として利用して、UniPortKB の CFTR_MOUSE（uniprot:CFTR_MOUSE）を UniProtKB/SwissProt（uniprotkb_swissprot）に対して NCBI BLAST 検索 を実行してください。

解答例： solutions/soap/jaxws/Q8NcbiBlastSoap.java

練習 5：NCBI BLAST（SOAP）メタデータ

NCBI BLAST (SOAP) サービスは、多くの種類のフォーマットで結果を返すことができます。 getResultTypes(jobId) メソッドを使って、他の利用可能な結果のタイプを検出するようにクライアントを修正してください。

解答例：solutions/soap/jaxws/Q9NcbiBlastSoap.java

プロキシ

一部の環境では、クライアントが外部のサービスに接続できるように HTTP プロキシを設定する必要があります。JAX-WS はプロキシ設定のための通常の Java メカニズムをサポートしています：

1. Java システムプロパティ
    - JVM に設定

```
java -Dhttp.proxyHost=proxy.example.org -Dhttp.proxyPort=8080
ExampleClientClass
```

    - クライアントコードで設定

```
System.setProperty("http.proxyHost", "proxy.example.org");
System.setProperty("http.proxyPort", "8080");
```

2. Java.net.Proxy と java.net.ProxySelector クラスを使用

詳細と実例は下記を参照:

- Networking Properties:
  http://download.oracle.com/javase/6/docs/technotes/guides/net/properties.html
- Java Networking and Proxies:
  http://download.oracle.com/javase/6/docs/technotes/guides/net/proxies.html

User-Agent

HTTP クライアントは通常そのクライアントが何者かについての情報を提供し、もし必要なら特定のクライアントのみ異なる操作をするサービスを可能にし、サービスがどのように利用されているかについて、サービス提供者にいくらかの情報が与えられます。デフォルトで JAX-WS は HTTP User-Agent ヘッダ（RFC2616 section 14.43 参照）を"JAX-WS RI 2.1.6 in JDK 6"、バージョン番号（2.1.6）は JAX-WS のバージョン、のように設定します。もし付加的にクライアントの識別が要求された場合、より詳述なプロダクトトークン（RFC2616 section 3.8 参照）が User-Agent 文字列の先頭に追加されなければなりません。例えば:

```
// Modify the user-agent to add a more specific prefix (see RFC2616
section 14.43)
String clientUserAgent = "Example-Client/1.0 (" +
                         System.getProperty("os.name") + ")";
((BindingProvider)srvProxy).getRequestContext().put(
                   MessageContext.HTTP_REQUEST_HEADERS,
  Collections.singletonMap("User-Agent"
                  ,Collections.singletonList(clientUserAgent))
);
```

ここで、srvProxy は User-Agent が変更されるサービスプロキシオブジェクトです。

サービスエンドポイントと名前空間

Web サービスの代替インスタンスへのアクセスが必要とされた場合、サービスのエンドポイントアドレスと、場合によっては SOAP メッセージに使われる XML 名前空間を変更する必要があります。JAX-WS では、要求されたエンドポイントアドレスとサービス SML 名前空間を記述している WSDL 文書を指定して、サービスプロキシのオブジェクトを獲得するとき、それらをオーバーライドできます。:

```
String srvWsdl
            = "http://wwwdev.ebi.ac.uk/ws/services/WSDbfetchDoclit?wsdl";
String srvXmlNamespaceUri =
            "http://www.ebi.ac.uk/ws/services/WSDbfetchDoclit";
String srvServiceName = "WSDBFetchDoclitServerService";
WSDBFetchDoclitServerService service = new WSDBFetchDoclitServerService(
  new java.net.URL(srvWsdl),
  new javax.xml.namespace.QName(srvXmlNamespaceUri, srvServiceName));
WSDBFetchServer srvProxy = service.getWSDbfetchDoclit();
```

サンプルクライアント

EMBL-EBI の大部分の SOAP Web サービスには、サービスへのコマンドラインでのアクセスと、サンプルコードを提供するサンプルクライアントがあります。Java 用に、いくつかのクライアントは JAX-WS ベースにしています。

Document/literal SOAP

Service	Sample client
EB-eye	Executable jar: EBeye_JAXWS.jar; Source: EBeyeClient.java
InterProScan (SOAP)	Executable jar: IPRScan_JAXWS.jar; Source: AbstractWsToolClient.java, IPRScanClient.java
NCBI BLAST (SOAP)	Executable jar: NCBIBlast_JAXWS.jar; Source: AbstractWsToolClient.java, NCBIBlastClient.java
WSDbfetch (SOAP)	Executable jar: WSDbfetch_JAXWS.jar; Source: WSDbfetchClient.java

1) JAX-WS - http://jcp.org/en/jsr/detail?id=224
2) JAX-WS Reference Implementation - http://jax-ws.java.net/
3) Apache commons-cli - http://commons.apache.org/cli/
4) Apache ant - http://ant.apache.org/
5) Apache Maven - http://maven.apache.org/