

Home

Blog?

Blog stream

Create a blog post to share news and announcements with your team and company.



What?

The AndroidSOAP is a "yet another" SOAP client for Android platform, it is compatible with 1.5+ versions and uses Reflection API.



But... this is only a **proof-of-concept** library, I've tested it with JBoss WebService stack, see and/or try the example.

Why?

Using AndroidSOAP is a very easy way to call SOAP services, because it is based on JAX-WS interfaces. You can use all interfaces that the 'wsimport' generates from the WSDL.

How?

```

/**
 * Creates a request, it is a JAX-WS generated class
 */
ListSkinPacksRequest request = new ListSkinPacksRequest();

/**
 * Put it into a 'request' map
 */
Map<String, Object> parameters = new HashMap<String, Object>();
parameters.put("request", request);

/**
 * Create an envelope with namespace and create a body with operation name 'listSkinPacks'
 */
Envelope envelope = new SimpleEnvelope("http://skinpack.pop.javaforum.hu/");
envelope.setHeader(new SimpleHeader());
envelope.setBody(new SimpleBody("listSkinPacks", parameters));

/**
 * Creates a transport (HTTP or HTTPS) with an optional username/password
 */
HttpsTransport transport = new HttpsTransport("https://services.power.of.planets.hu/PoP-SkinPack-remote/listSkinPacks",
    "androidsoap.demo@javaforum.hu", "demopassword");

/**
 * Set the 'trust all' flag, if necessary
 */
transport.setTrustAll(Boolean.TRUE);

/**
 * Call a service, the result arrives into the JAX-WS class
 */
ListSkinPacksResponse response = transport.call(envelope, ListSkinPacksResponse.class);

/**
 * You can use the result as a normal bean
 */
for (SkinPackMetadata metadata : response.getReturn().getSkinPacksList())
{
    String fileName = metadata.getFileName();
    String name = metadata.getName();
}

```

Can you see? The lists are lists, the classes are classes, the values are in the properties, just like in the JAX-WX client... 😊



If any field has XmlElement annotation in the generated source, you need add jaxb-api into your Android project, like this:

```

<dependency>
  <groupId>javax.xml.bind</groupId>
  <artifactId>jaxb-api</artifactId>
  <version>2.2.6</version>
  <scope>compile</scope>
</dependency>

```

And add coreLibrary into the configuration of the android-maven-plugin:

```

<dex>
  <coreLibrary>true</coreLibrary>
</dex>

```

Where?

Source: <http://svn.javaforum.hu/svn/android/AndroidSOAP/trunk/>

Wiki: <http://wiki.javaforum.hu/display/ANDROIDSOAP/Home>

Issues: <http://traq.javaforum.hu/browse/ANDROIDSOAP>

Example APK: <http://nexus.javaforum.hu/nexus/content/repositories/releases/hu/javaforum/android/androidsoap/AndroidSOAP-example/0.0.5/AndroidSOAP-example-0.0.5.apk>

Tutorial?

Step by step tutorial with Maven3 project: [Step by Step Tutorial](#)

Patches and feature requests?

If you have a patch, don't hesitate, [send me](#) as fast you can... 😊

If you need a feature, [create an issue...](#) 😊

If I'm not using Maven?

The entire project (with dependencies) managed by Maven, if you aren't using Maven, you need some jars to add your project's CLASSPATH:

- [slf4j-android-1.6.1-RC1.jar](#) or above
- [commons-codec-1.3.jar](#) exactly
- [javaforum20-commons-1.0.2.jar](#) or above
- [AndroidSOAP-lib-0.0.5.jar](#) or above
- [validation-api-1.1.0.Alpha1.jar](#) exactly

Example screenshot

