EXECUTE A SERVICE TYPE ON THE SERVER ASYNCHRONOUSLY

The code below is an example of how to execute a method on a service type on the server. A service type will select any service of that type and try to execute the method on it. The services may typically be owned by the server itself, for query or other purposes. This can include the server-based web folder searches. This example executes a server web folder search for a file name, but it can also be on any service type. Note that 2 methodinfos are required, where one is the parameter of the other.

```
String commID;                    //comm id
String condition;                 //comparison condition
ArrayList parameters;             //methodinfo parameters
QueryModel queryModel;            //query model
MessageInfo reply;                //message reply
MethodInfo methodInfo;            //method info
MethodInfo methodInfo2;           //method info
CallObject callObject;            //call object

serverUri = //set the server URI
callObject = new CallObject();

if (serverUri != null)
{
    condition = LicasQueryConst.FILENAME;

    //make server async call to search web folder for file type
    //QueryModel and QueryConst have some help methods to determine query type
    if (QueryConst.otherOps().contains(condition))
    {
            queryModel = QueryModel.createModel(
                        QueryModel.typeFromOpOther(condition));
    }
    else
    {
            queryModel = QueryModel.createModel(
                        QueryModel.typeFromOpText(condition));
    }

    //another help method to easily set he query condition
    queryModel.setSingleCondition(condition, QueryConst.CONTAINS, "anim");

    //make an asynchronous method call on server, so nested methodinfo's
    methodInfo2 = new MethodInfo();
    methodInfo2.setName(LicasQueryConst.SEARCHWEBFOLDER);
    methodInfo2.setRtnType(TypeConst.ELEMENT);
    methodInfo2.setCallTimeout(30000);
    methodInfo2.addParam(QueryModel.toXml(queryModel));
    commID = UuidHandler.getUuid(20);
    methodInfo2.setCommunicationID(commID);

    parameters = new ArrayList();
    parameters.add(LicasQueryConst.SEARCHWEBFOLDER);
    parameters.add(methodInfo2);

    methodInfo = MethodFactory.createMethodCall(
            MethodConst.EXECUTESERVICETYPEASYNC, TypeConst.VOID,
            serverUri, parameters, passwordHandler);
    callObject.call(methodInfo);

    //delay
    Thread.sleep(1000);

    //get result using comm id and display, server services return Elements only
    reply = null;
    parameters = new ArrayList();
    parameters.add(commID);
    methodInfo = MethodFactory.createMethodCall(
            MethodConst.SYNCTOASYNCTRANSIT, TypeConst.BOOLEAN, serverUri,
            parameters, passwordHandler);
    inTransit = ((Boolean)callObject.call(methodInfo)).booleanValue();
    System.out.println("In transit: " + String.valueOf(inTransit));

    while ((reply == null) && inTransit) {
            parameters = new ArrayList();
            parameters.add(commID);
            methodInfo = MethodFactory.createMethodCall(
                    MethodConst.SYNCTOASYNCREPLY, MessageInfo.class.getName(),
                    serverUri, parameters, passwordHandler);
            reply = (MessageInfo) callObject.call(methodInfo);

            if (reply == null) {
                    Thread.sleep(1000);
                    methodInfo = MethodFactory.createMethodCall(
                            MethodConst.SYNCTOASYNCTRANSIT,
                            TypeConst.BOOLEAN, serverUri,
                            parameters, passwordHandler);
                     inTransit = ((Boolean) callObject.call(methodInfo)).booleanValue();
                     System.out.println("In transit: " + String.valueOf(inTransit));
            }
    }

    System.out.println("Reply: " +
            XmlHandler.xmlToFormattedString((Element) reply.getMessage()));
}
```