ADD A SERVICE TO THE SERVER

To add a service to a server remotely, you need the server details and also the details of the class that you want to add as a service. If this class is part of a separate jar file, then you need to include the jar file paths as well. The code might look as follows:

```
//some variables that might be required
int i;
boolean isOK;                   //true if OK
String parentService;           //parent service
String theUuid;                 //the new service id
String serviceType;             //service type
String serviceClass;            //service class
String servicePassword;         //service password
String adminKey;                //service admin key
String jarFile;                 //jar file path
Vector<String> jarFiles;        //all jar files
Vector convertParams;           //converted parameters
Element serviceUri;             //service uri
Element newServicePath;         //new service path
File filePath;                  //file path
File[] allFiles;                //all files in a path
MethodInfo methodInfo;          //used to store the method information
CallObject callObject;          //used to make the remote method call


//retrieve the server url, of the form server URL and Const.HTTPSERVER value.
serverUri = <Handle><U>http://123.4.5.6:8888/</U><S>HttpServer</S></Handle>
callObject = new CallObject();

//get the passwords for the URI Handle
servicePassword = passwordHandler.getServicePassword(serverUri);
adminKey = passwordHandler.getadminKey(serverUri);

//create the haandle of the service that you want to add the new service to
serviceUri = Handle.createNewUrlHandle(Handle.getURI(serverUri));
serviceUri = Handle.addToHandle(serviceUri, Handle.asHandleElement(serviceToAddTo));

...

//if the server uri exists then try to add the service
if ((serverUriStr != null) && !serverUriStr.equals(""))
{
    //service passwords can either be stored under just the uuid name or the full URI path
    parentService = Handle.getLastHandle(serviceUri);

    newServiceUuid = ???        //new service uuid
```

```
newServiceType = ???          //a description of the service type

//fully qualified class description of the service, can use something similar to
newServiceClass = YourClass.class.getName();

//TYPICALLY - this is enough
jarFiles = new Vector();

//IF YOUR OWN MODULES - you might need to add new jar files
//if the classes are to be remotely loaded then add the required jar files
//if all classes are local then you do not need to do this
jarFile =                                                    //jar file path
jarFile = getFullJarPath(jarFile);                           //checks this path
jarFiles.add(LoadObject.addJarPrefix(serverUriStr, jarFile)); //converts to proper format

//OPTIONAL - add any additional jar files in a related lib folder
filePath = new File(jarFile);
filePath = filePath.getAbsoluteFile();
jarFile = filePath.getParent();

filePath = new File(jarFile + LIBPATH);
if (filePath.isDirectory())
{
    allFiles = filePath.listFiles();
    for (i = 0; i < allFiles.length; i++)
    {
        if (allFiles[i].getName().endsWith(".jar"))
        {
            jarFile = (filePath.getAbsolutePath() + "/" + allFiles[i].getName());
            jarFiles.add(LoadObject.addJarPrefix(serverUriStr, jarFile));
        }
    }
}

...

//if service details are available then you can try to add the service to the network
if ((newServiceUuid != null) && !newServiceUuid.equals(""))
{
    //new service path is the path of the parent service plus the new uuid
    newServicePath = Handle.addToHandle(serviceUri,
                                        Handle.asHandleElement(newServiceUuid));

    if ((newServiceType != null) && !newServiceType.equals(""))
    {
        //check if adding to the base server - then need whole uri again
        if (parentService.equals(Const.HTTPSERVER)) parentService = serverUriStr;

        //also need to add any parameters required by the new service constructor
        //this would probably include the password and admin key (shown here)
```

```
                //it might also include an admin document
                params = new Vector();
                params.addElement(Const.ANON);
                params.addElement(Const.ANON);

                //add to the specified service
                methodInfo = new MethodInfo();
                methodInfo.setName(MethodConst.ADDSERVICE);
                methodInfo.setRtnType(TypeConst.BOOLEAN);
                methodInfo.setServiceURI(serviceUri);
                methodInfo.setServerPassword(passwordHandler.getServicePassword(serverUri));
                methodInfo.setPassword(getServicePassword(parentService));
                methodInfo.addParam(getServicePassword(parentService));
                methodInfo.addParam(newServiceUuid);
                methodInfo.addParam(newServiceType);
                methodInfo.addParam(jarFiles);
                methodInfo.addParam(newServiceClass);
                methodInfo.addParam(false);
                methodInfo.addParam(params);

                //method parameters are a single list to the MethodInfo object
                //but then the constructor 'params' is also a list that gets added

                //then invoke the add service method
                isOK = ((Boolean)call.call(methodInfo)).booleanValue();

                .....
            }
        }
}
```