

Using Symbian Web Runtime to access web services

Example widget [Media:Wstutor.zip](#)

According to W3C a web service is a "a software system designed to support interoperable Machine to Machine interaction over a network". So web services offer a way of communication between different kind of applications. This communication usually takes place over HTTP. Web services are widely used by popular web sites, so to allow developers to incorporate the functionality offered in the web sites into their applications. Just to name a few [Google](#), [Yahoo!](#), [Amazon](#), [Facebook](#) and many others. These web services can be accessed by widgets easily, offering new exciting possibilities for the developers! Web services can be divided into two categories:

- RESTful web services and
- SOAP web services.

Accessing RESTful web services

RESTful web services can be considered as functions that are executed remotely. These functions accept parameters and they return a result. The parameters can be send using HTTP GET or HTTP POST and the result can be a JSON object or some XML text. An example of a RESTful web service that receives parameters via HTTP GET and returns a JSON object is the [Google AJAX Language API](#) which allows developers to translate and detect the language of blocks of text. The code that follows uses that web service in order to translate a phrase from English to French:

```
var word      = "The phrase to translate;
var getString = "v=1.0&q="+escape(word)+"&langpair=en%7Cfr";
var ajaxGet   = null;
ajaxGet       = new XMLHttpRequest();
ajaxGet.open ("GET",
"http://ajax.googleapis.com/ajax/services/language/translate?" + getString, true);
ajaxGet.onreadystatechange = function() { //Call a function when the state changes.
  if(ajaxGet.readyState == 4 ) {
    if(ajaxGet.status==200){
      //JSON objects can be manipulated extremely easily!!
      var transObj = eval('(' + ajaxGet.responseText + ')');
      alert(transObj.responseData.translatedText);
    }
  }
}
```

[Yahoo! Traffic web service](#) is a web service that displays traffic information for a certain road. It accepts input parameters either by HTTP GET or HTTP POST and the response is some XML text. The code that follows invokes that web service, in order to get traffic information about Broadway in New York, using as input method HTTP POST:

```
var appID = "You have to get an APP ID from yahoo!";
var street= "Broadway";
var city  = "New+York";
var state = "NY";
var postQ = "appid="+appID+"&street="+street+"&city="+city+"&state="+state;
var ajaxPost = null;
ajaxPost     = new XMLHttpRequest();
ajaxPost.open("POST", "http://local.yahooapis.com/MapsService/V1/trafficData", true);
ajaxPost.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
ajaxPost.setRequestHeader("Content-length", postQ.length);
ajaxPost.onreadystatechange = function() { //Call a function when the state changes.
  if(ajaxPost.readyState == 4 ) {
    if(ajaxPost.status==200){
```

```

    \\Do something with the returned xml
        }else{
    alert("error" );
    }
    }
}
}
ajaxPost.send(postQ);

```

SOAP web services

SOAP web services are web services that are invoked using a XML based protocol named [SOAP](#). The advantage of these web services is that are fully described in a language, named [WSDL](#). WSDL is a XML based language which automation tools can parse and generate the appropriate SOAP messages. The disadvantage of these web services is that usually SOAP messages are lengthy which may lead to undesirable HTTP traffic size. SOAP messages are send using HTTP POST. The following example uses [Lyrics Wiki SOAP web services](#) in order to find the lyrics of the song "eat the rich" of "Aerosmith"

```

var artist = "Aerosmith";
var song   = "Eat the rich";
var soapReq = "<?xml version=\"1.0\" encoding=\"ISO-8859-1\"?>"+
    "<SOAP-ENV:Envelope SOAP-
ENV:encodingStyle=\"http://schemas.xmlsoap.org/soap/encoding/\""+
    " xmlns:SOAP-ENV=\"http://schemas.xmlsoap.org/soap/envelope/\"
xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\""+
    " xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" xmlns:SOAP-
ENC=\"http://schemas.xmlsoap.org/soap/encoding/\""+
    " xmlns:tns=\"urn:LyricWiki\">"+
    "<SOAP-ENV:Body><tns:getSong xmlns:tns=\"urn:LyricWiki\">"+
    "<artist xsi:type=\"xsd:string\">"+artist+"</artist>"+
    "<song xsi:type=\"xsd:string\">"+song+"</song>"+
    "</tns:getSong></SOAP-ENV:Body></SOAP-ENV:Envelope>";
var ajaxPost = null;
ajaxPost = new XMLHttpRequest();
ajaxPost.open("POST", "http://lyricwiki.org/server.php", true);
ajaxPost.setRequestHeader("Content-type", "text/xml");
ajaxPost.setRequestHeader("Content-length", soapReq.length);
ajaxPost.onreadystatechange = function() { //Call a function when the state changes.
    if(ajaxPost.readyState == 4 ) {
    if(ajaxPost.status==200){
    //Do something with the xml result
    }else{
    alert("error" );
    }
    }
}
}
ajaxPost.send(soapReq);

```

Limitations

Current XMLHttpRequest object implementation does not allow to set custom HTTP headers. However some web services that offered only to authentication users, demand the usage of custom HTTP headers in order to send user credentials.

