



Contents

- [1 Introduction](#)
- [2 Concept](#)
- [3 Connecting to the Web service](#)
- [4 XML parsing](#)
- [5 Trimming string data](#)
- [6 Temperature gauge animation](#)
- [7 Final output](#)
- [8 Source code](#)
- [9 Author](#)

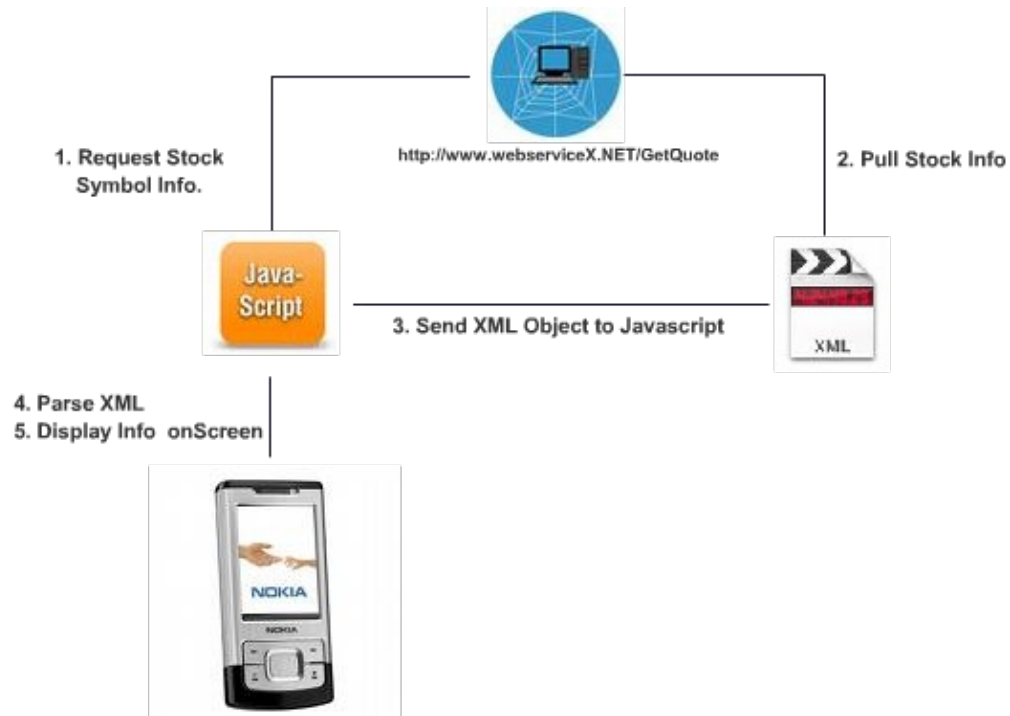
Introduction

Like any other application development technology, such as Symbian C++ or Java ME, Flash Lite can connect to a Web service and thus send/receive data. This article illustrates again the power of Flash Lite.

Concept

Here is a reference to the [WSDL page](#) on Wikipedia. To learn on the WSDL concept in detail, you may refer to this page. To learn more about its implementation in flash Lite, read below.

How_to_create_a_Flash_Lite_weather_application_using_Web_services



Connecting to the Web service



The client's device establishes communication with the Web services by creating a new Web service object in the ActionScript and invokes its `GetWeather()` method once the "GO" button is pressed.

```
import mx.services.*;
// Create a new instance of Web service called - wthrService.
var wthrService = new WebService("http://www.webservices.net/globalweather.asmx?WSDL");

// set it up so that the service is called when the button is pressed.
```

How to create a Flash Lite weather application using Web services

```
result_btn.onPress = function() {
    loading_mc._visible = true;
    //Call the getWeather method and assign it to the pending call object to handle results.
    resultObj = wthrService.getWeather(citybox.text, "");
    wthrService.addEventListener(Event.COMPLETE, trace("loading"));
    // the onResult function is called if the service is successful
    resultObj.addEventListener(Event.COMPLETE, function(result) {
if(result != "Data Not Found")
{
    _visible = false;
    resultObj.addEventListener(Event.COMPLETE, function(result) {
        xml = null;
        delete xml;
        xml = new XML();
        xml.ignoreWhite = true;
        xml.parseXML(resStr);
        city_txt.text = citybox.text;
        gotoAndStop(2);
    }
    else
    {
        gotoAndStop(3);
    }
    resultObj.addEventListener(Event.COMPLETE, function(fault) {
        // If there is any error, e.g. the service is not working, the onFault handler will be invoked.
        loading_mc._visible = false;
        fault_txt.text = "ERROR: " + fault.faultCode + ", " + fault.faultstring;
    }
    }
    }
}
```

XML parsing

The server sends the XML object containing weather information that must be parsed before it can be used.

- XML FORMAT

```
<?xml version="1.0" encoding="utf-16"?>
<CurrentWeather>
  <Location>Islamabad Airport, Pakistan (OPRN) 33-37N 073-06E 508M</Location>
  <Time>May 29, 2008 - 07:00 PM EDT / 2008.05.29 2300 UTC</Time>
  <Visibility> 3 mile(s):0</Visibility>
  <SkyConditions> clear</SkyConditions>
  <Temperature> 77 F (25 C)</Temperature>
  <DewPoint> 59 F (15 C)</DewPoint>
  <RelativeHumidity> 53%</RelativeHumidity>
  <Pressure> 29.65 in. Hg (1004 hPa)</Pressure>
  <Status>Success</Status>
</CurrentWeather>
```

The XML object may have more or less nodes as it depends on the level of information available on the server at any given time. Hence using the following code, the XML node's name/value pairs are stored in an array of a dataclass object.

- DataClass

```
class dataClass
```

How_to_create_a_Flash_Lite_weather_application_using_Web_services

```
{
    //Establishes the variables in dataClass
    public var nodeName: String;
    public var nodeValue:String;
}

import dataClass;

for (var aNode:XMLNode = xml.firstChild; aNode != null; aNode=aNode.nextSibling) {
    if (aNode.nodeType == 1) {
    var t:dataClass = new dataClass();
        nodeName = aNode.nodeName;
        nodeValue = aNode.firstChild.nodeValue;
        array.push(t);
    }
}
```

Trimming string data

Once data is saved in the array, it can be retrieved from the array and trimmed to be displayed on the screen.

```
for (j=0; j<array.length; j++)
{
    if(array[j].nodeName eq "Time")
    {
        array[j].nodeValue;
        getLocalTimes=getLocalTime.getTime().indexOf("EDT");
    }
    if(array[j].nodeName eq "Wind")
    {
        array[j].nodeValue;
        Number=getWind=slice(getWind.indexOf("at ") +3, getWind.indexOf("MPH")-1);
    }
    if(array[j].nodeName eq "Visibility")
    {
        array[j].nodeValue;
        getVisibility =slice(0, getVisibility.length-2);
    }
    if(array[j].nodeName eq "Temperature")
    {
        array[j].nodeValue;
        getTemperature_F =getTemperature.indexOf("F")-1;
        getTemperature_C =getTemperature.indexOf("(")+1, getTemperature.indexOf("C")
    }
    if(array[j].nodeName eq "DewPoint")
        array[j].nodeValue;
    if(array[j].nodeName eq "RelativeHumidity")
        array[j].nodeValue;
    if(array[j].nodeName eq "Pressure")
    {
        array[j].nodeValue;
        getPressure =slice(0, getPressure.indexOf("(")+1, getPressure.length-1);
    }
}
```

Temperature gauge animation

```
new mx.transitions.Tween(temp_mc.temp_lvl_mc, "_height", mx.transitions.easing.Strong.easeOut, 0,
```

Final output



Source code

The source code files are available for download: [media:FLWeather.zip](#)

Author

--NINJA 03:16, 30 May 2008 (EEST) RAHEAL AKHTAR