HTTP adapter - Communicating with HTTP back-end systems

- Download Studio project (http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v630/MobileFirstAdaptersProject.zip)

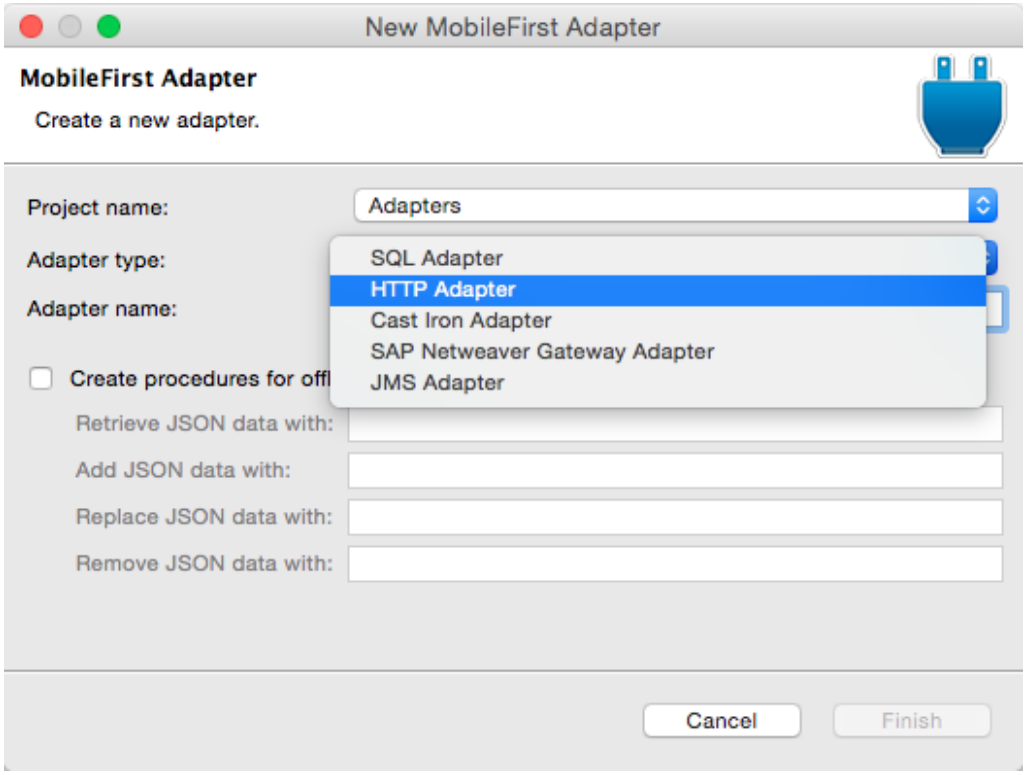
Overview

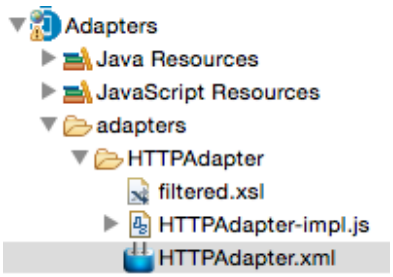
By using IBM MobileFirst Platform Foundation HTTP adapters, you can send GET or POST HTTP requests and retrieve data from the response headers and body. HTTP adapters work with RESTful and SOAP-based services and can read structured HTTP sources such as RSS feeds.

You can easily customize HTTP adapters with simple server-side JavaScript code. For example, you could set up server-side filtering if necessary. The retrieved data can be in XML, HTML, JSON, or plain text format.

The adapter is configured with XML to define the adapter properties and procedures. Optional: It uses XSL to filter received records and fields.

Creating the adapter

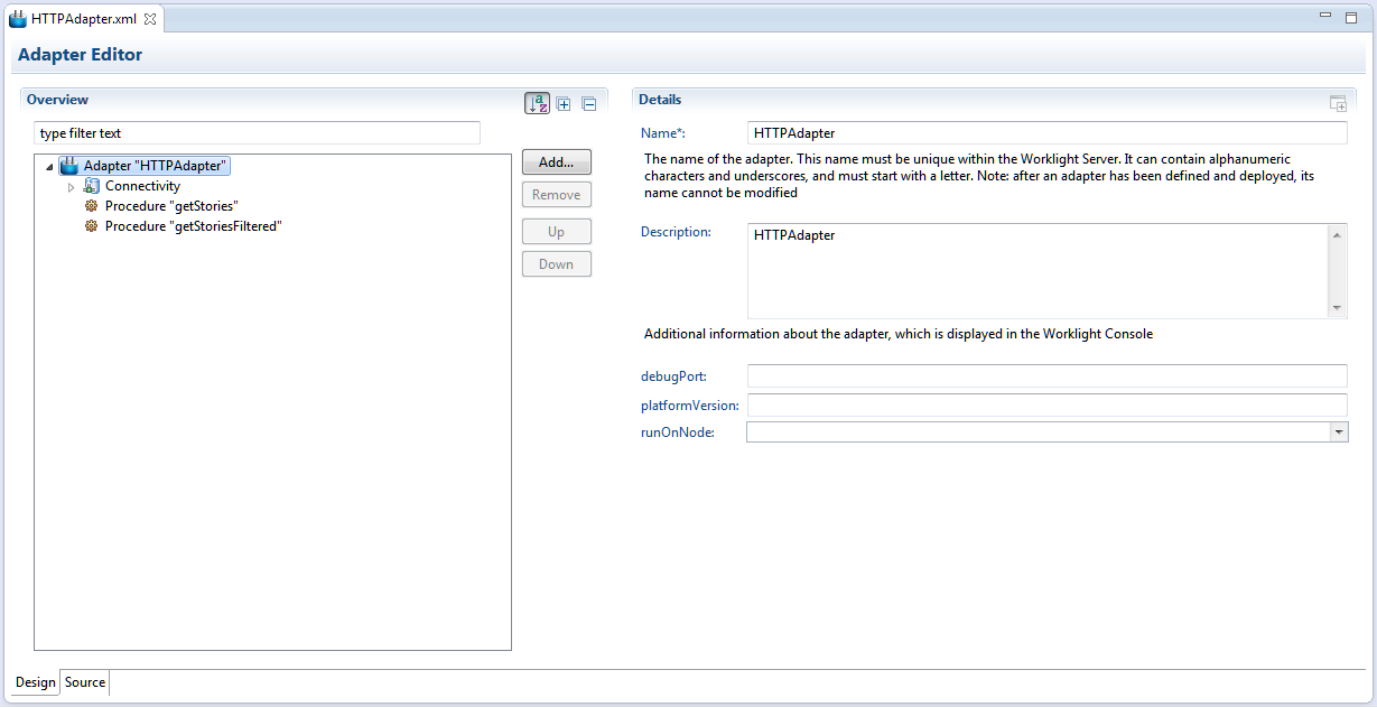




In MobileFirst Studio, create an adapter and select the **HTTP Adapter** type. A standard HTTP adapter structure is created.

Adapter XML

Settings and metadata are stored in the adapter XML file. You can use either the **Design** or **Source** editor to modify the adapter XML file.



To edit the adapter XML file, you must:

- Set the protocol to HTTP or HTTPS
- Set the HTTP domain to the domain part of HTTP URL
- Set the TCP Port

Declare the required procedures below the connectivity element.

```
<wl:adapter name="HTTPAdapter"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:wl="http://www.ibm.com/mfp/integration"
 xmlns:http="http://www.ibm.com/mfp/integration/http"></p>
 <displayName>HTTPAdapter</displayName>
 <description>HTTPAdapter</description>
 <connectivity>
  <connectionPolicy xsi:type="http:HTTPConnectionPolicyType">
   <protocol>http</protocol>
   <domain>rss.cnn.com</domain>
   <port>80</port>
   <connectionTimeoutInMilliseconds>30000</connectionTimeoutInMilliseconds>
   <socketTimeoutInMilliseconds>30000</socketTimeoutInMilliseconds>
   <maxConcurrentConnectionsPerNode>50</maxConcurrentConnectionsPerNode>
   <!-- The following properties are used by the adapter's key manager for choosing specific certificate from the k
eystore.
   <sslCertificateAlias></sslCertificateAlias>
   <sslCertificatePassword></sslCertificatePassword>
   -->
  </connectionPolicy>
 </connectivity></p>
 <procedure name="getStories"/></p>
 <procedure name="getStoriesFiltered"/></p>
</wl:adapter>
```

JavaScript implementation

Procedures are implemented in the adapter JavaScript file. The service URL is used for procedure invocations.

Some parts of the URL are constant; for example, http://example.com/. They are declared in the XML file. Other parts of the URL can be parameterized; that is, substituted at run time by parameter values that are provided to the MobileFirst procedure.

The following URL parts can be parameterized.

- Path elements
- Query string parameters
- Fragments

> See the "The connectionPolicy element of the HTTP adapter" user documentation topic for advanced options for adapters, such as cookies, headers, and encoding.

In the JavaScript file, use the same procedure name as in the XML file. The mandatory parameters to call the procedure are `method`, `path`, and `returnedContentType`. The procedure can be parameterized at run time.

```
function getFeeds() {
 var input = {
    method : 'get',
    returnedContentType : 'xml',
    path : "rss.xml"
 };
return WL.Server.invokeHttp(input);
}
```

To call an HTTP request, use the `WL.Server.invokeHttp` method. Provide an input parameters object, which must specify:

- The HTTP method: `GET`, `POST`, `PUT`, `DELETE`
- The returned content type: `XML`, `JSON`, `HTML`, or `plain`
- The service `path`
- The query parameters (optional)
- The request body (optional)
- The transformation type (optional)

> See the "WL.Server.invokeHttp" user documentation topic for a complete list of options.

XSL transformation filtering

You can apply XSL transformation to the received data, for example to filter the data. To apply XSL transformation, specify the transformation options in the input parameters of the procedure invocation.

```
function getFeedsFiltered() {
 var input = {
    method : 'get',
    returnedContentType : 'xml',
    path : "rss.xml",
    transformation : {
     type : 'xslFile',
     xslFile : 'filtered.xsl'
    }
 };
return WL.Server.invokeHttp(input);
}
```

Creating a SOAP-based service request

You can use the `WL.Server.invokeHttp` method to create a **SOAP** envelope, which can be sent directly.

To call a SOAP-based service in an HTTP adapter, you must encode the SOAP XML envelope within the request body. Encoding XML within JavaScript is simple: just use **E4X**, which is officially part of JavaScript 1.6. You can use this technology to encode any type of XML document, not only SOAP envelopes.

If you receive error messages for SOAP envelopes, disable the JavaScript validator. Click **Project > Properties > Builders** and clear **JavaScript Validator**.

Use JavaScript to create a SOAP Envelope. It is possible to insert JavaScript code and variables into SOAP XML. Such additional code is evaluated at run time.

```
var request =
 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/20
01/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
   <CelsiusToFahrenheit xmlns="http://www.w3schools.com/webservices/">
    <Celsius>{celsiusTemp}</Celsius>
   </CelsiusToFahrenheit>
  </soap:Body>
 </soap:Envelope>;
```

The `WL.Server.invokeHttp(options)` method is used to call a request for a SOAP service.

The Options object must include the following properties:

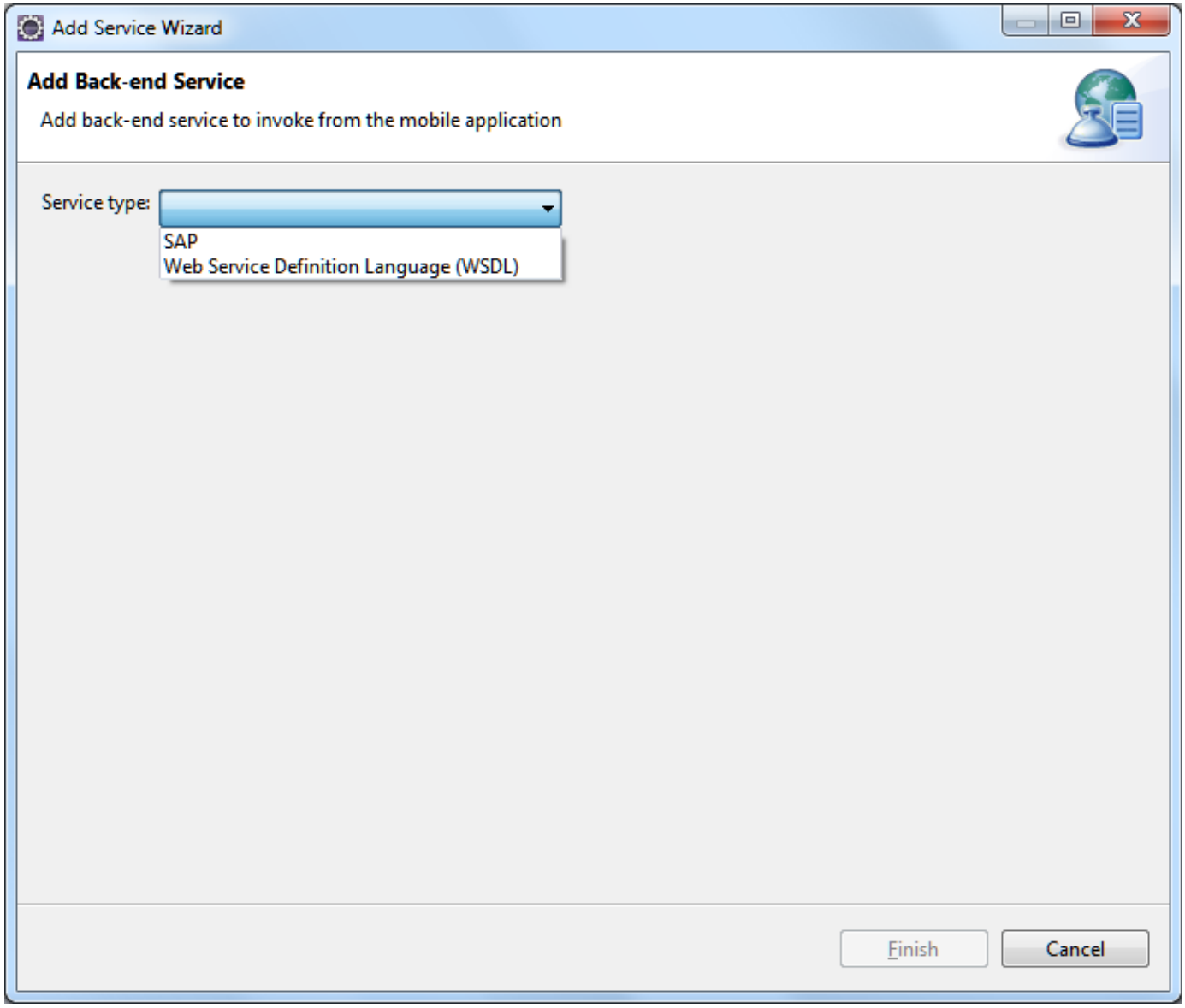
- A `method` property: usually `POST`
- A `returnedContentType` property: usually `XML`
- A `path` property: a service path
- A `body` property: `content` (SOAP XML as a string) and `contentType`

```
var input = {
    method: 'post',
    returnedContentType: 'xml',
    path: '/webservices/tempconvert.asmx'
,
    body: {
     content: request.toString(),
     contentType: 'text/xml; charset=utf-8',
    },
 };
 var result = WL.Server.invokeHttp(input);
```

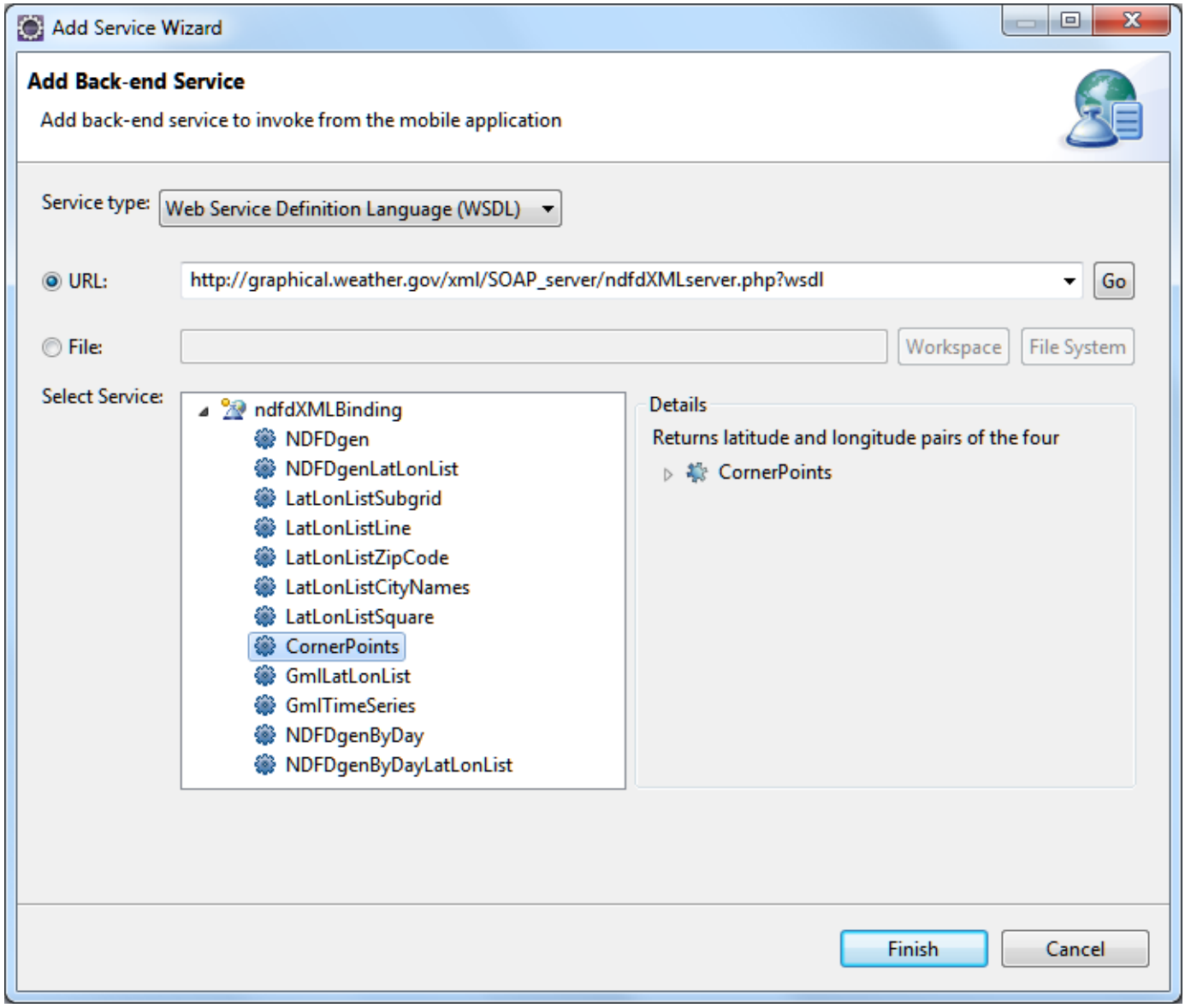
Back-end service discovery

If you are developing HTTP adapters for SOAP (or SAP) services, you can reduce development time by using the Discover Back-end Services tool to autogenerate the adapter with procedures that are based on the provided WSDL.

Right-click the services folder in a MobileFirst project and choose **Discover back-end services**. Select the type of service: SAP or SOAP.



Add the services location to use and select the service that you want to add to the adapter. Repeat for each service that you want to add. The service will be added to the same adapter.



MobileFirst Studio will then auto-generate an adapter.

> For more information, review the "Generating adapters with the services discovery wizard" user documentation topic.

Sample application

Click to download (http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v630/MobileFirstAdaptersProject.zip) the Studio project.