DEVELOPING WEB SERVICES

Remote Procedure Calls (RPC)

REMOTE PROCEDURE CALLS

What is an RPC

- As name suggests RPCs **call** a **remote procedure**
- Calls a procedure or function that is inside another process running on same or different server by using http protocol
- We write a function such that it accepts **parameters**
- The function completes some operation
- The function returns result if it succeeded
  - (or error message if it failed).
- Most common formats are
  - XML-RPC and JSON-RPC

XML-RPC

What is XML-RPC ?

- RPC - Remote Procedure Calls
  Allows you to make calls to procedures or functions that are on another computer

- XML-RPC is among the simplest and most foolproof web service approaches that makes it easy for computers to call procedures on other computers.

XML-RPC

XML-RPC consists of three parts

- **Data types**
  - A set of types used in passing parameters, return values, and faults (error messages).
- **Request structures**
  - An HTTP request containing the method and the parameter information.
- **Response structures**
  - An HTTP response that contains return values or fault information.

XML-RPC

Basic Data Types

- There are 6 basic data types and 2 complex types
  - <int>123</int> *or* <i4>123</i4>
  - <double>98.7654</double>
  - <boolean>1</boolean>
  - <string>Hello World</string>
  - <dateTime.iso8601>20170517T21:55:07</dateTime.iso8601>
  - <base64>SGVsbG8gV29ybGQh</base64>
    - *Hello World!*

XML-RPC

Basic Data Types

```
<value>
   <array>
      <data>
         <value><boolean>1</boolean></value>
         <value>Montreal</value>
         <value><string>Québec</string></value>
         <value><int>-40</int></value>
         <value><double>42.14159265</double></value>
      </data>
   </array>
</value>
```

XML-RPC

Complex Data Types

Array

```
<value>
   <array>
      <data>
         <value>
            <array>
               <data>
                  <value><int>10</int></value>
                  <value><int>20</int></value>
               </data>
            </array>
         </value>

         <value>
            <array>
               <data>
                  <value><int>15</int></value>
                  <value><int>25</int></value>
               </data>
            </array>
         </value>
      </data>
   </array>
</value>
```

Struct

```
<value>
   <struct>

      <member>
         <name>givenName</name>
         <value><string>John</string></value>
      </member>

      <member>
         <name>familyName</name>
         <value><string>Smith</string></value>
      </member>

      <member>
         <name>age</name>
         <value><int>27</int></value>
      </member>

   </struct>
</value>
```

XML-RPC

Examples - array of struct

```
<methodCall><methodName>method.name</methodName>
<params>
<param>
   <value>
      <array>
         <data>
            <value>
               <struct>
                  <member><name>name</name><value><string>Lisa</string></value></member>
                  <member><name>age</name><value><int>9</int></value></member>
               </struct>
            </value>
            <value>
               <struct>
                  <member><name>name</name><value><string>Bart</string></value></member>
                  <member><name>age</name><value><int>10</int></value></member>
               </struct>
            </value>
         </data>
      </array>
   </value>
</param>
</params>
</methodCall>
```

XML-RPC

Request Format

- XML-RPC requests are a combination
- XML content
  - Uses the data typing to pass parameters
  - Contains additional information identifying which procedure is being called
- HTTP headers
  - Provides a wrapper for passing the request over the Web.

XML-RPC

XML Request Example

```
<?xml version="1.0"?>
<methodCall>
  <methodName>Math.Multiply</methodName>
  <params>
    <param>
        <value><int>2</int></value>
    </param>
    <param>
        <value><int>3</int></value>
    </param>
  </params>
</methodCall>
```

XML-RPC

Resulting HTTP Request

```
POST /xmlrpc HTTP 1.0
User-Agent: myXMLRPCClient/1.0
Host: 127.0.0.1
Content-Type: text/xml
Content-Length: 169
<?xml version="1.0"?>
<methodCall>
  <methodName>Math.Multiply</methodName>
  <params>
    <param>
        <value><int>2</int></value>
    </param>
    <param>
        <value><int>3</int></value>
    </param>
  </params>
</methodCall>
```

XML-RPC

Response Format

- An XML-RPC response can only contain one parameter.
- That parameter may be an array or a struct, so it is possible to return multiple values.
- It must return a value in response.
- If the response is successful - the procedure was found, executed correctly, and returned results
- If there was a problem in processing a request, a fault element will be used to indicate something went wrong.

XML-RPC

Response Format

```
HTTP/1.1 200 OK
Date: Mon, 27 Nov 2018 12:20:04 GMT
Server: Apache.1.3.12 (Unix)
Connection: close
Content-Type: text/xml
Content-Length: 124

<?xml version="1.0"?>
<methodResponse>
   <params>
      <param>
         <value><int>6</int></value>
      </param>
   </params>
</methodResponse>
```

XML-RPC

Response Format (fault)

```
HTTP/1.1 200 OK
Date: Mon, 27 Nov 2018 12:20:04 GMT
Server: Apache.1.3.12 (Unix)
Connection: close
Content-Type: text/xml
Content-Length: 124

<?xml version="1.0"?>
<methodResponse>
   <fault>
      <value><string>No such method!</string></value>
   </fault>
</methodResponse>
```

JSON-RPC

JSON-RPC Request

- Must contain the following keys:
  - **jsonrpc** - String specifying the version of the JSON-RPC protocol.
    - MUST be exactly "2.0"
  - **method** - Name of method/service
  - **params** - Array of arguments to be passed
  - **id** - Should be an integer and makes it easier for client to know which request it got response to

```
{"jsonrpc": "2.0", "method": "Math.Multiply",
    "params": [{A: 2, B: 3}], "id": 1}
```

JSON-RPC

JSON-RPC Responce

- Must contain the following keys:
  - **jsonrpc - S**tring specifying the version of the JSON-RPC protocol.
    - MUST be exactly "2.0"
  - **result -** Contains return value of method called
    - Value will be null if an error occurred
- **error -** If error occurred, this will indicate error code or error message, otherwise it’s null
- **id -** The id of the request it is responding to.

```
{"jsonrpc": "2.0", "result": 6, "error": null, "id": 1}
```

JSON-RPC

JSON-RPC Responce (Error)

- Error parameter must contain the following keys:
- **code -** An integer that indicates the error type that occurred.
- **message** - A String providing a short description of the error.
- **data** - An optional value that contains additional information about the error.

```
{"jsonrpc": "2.0", "error": {"code": -32601, "message": "Method
                    not found"}, "id": "5"},
```

JSON-RPC

JSON-RPC Responce Error Codes

code	message	meaning
-32700	Parse error	Invalid JSON was received by the server. An error occurred on the server while parsing the JSON text.
-32600	Invalid Request	The JSON sent is not a valid Request object.
-32601	Method not found	The method does not exist / is not available.
-32602	Invalid params	Invalid method parameter(s).
-32603	Internal error	Internal JSON-RPC error.
-32000 to -32099	Server error	Reserved for implementation-defined server-errors.