

熱血講義

프리렉의 열혈강의 시리즈

Python 파이썬



Python



22

: XML-RPC

(gslee@mail.kw.ac.kr)₂



- 1.
2. XML-RPC
3. XML-RPC

- Procedure Call



- RPC(Remote Procedure Call)



가

- XML(Extensible Markup Language)



- XML-RPC



HTTP XML RPC



HTTP XML

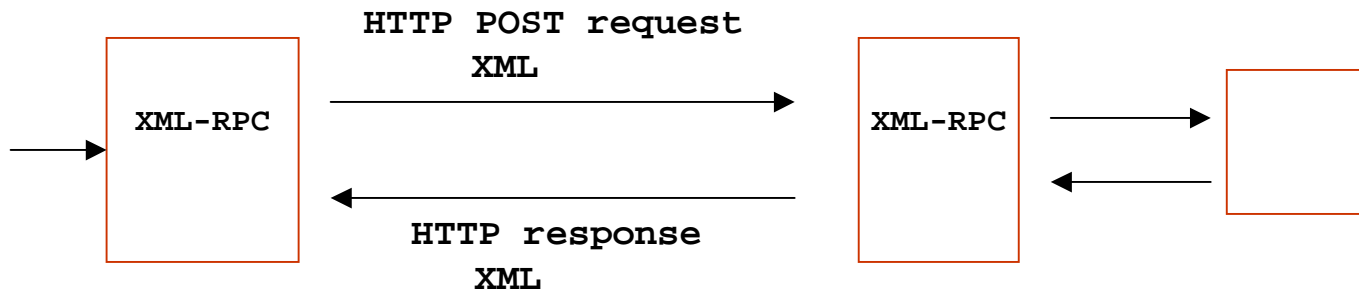
/



,

XML-RPC

- <http://www.xmlrpc.com/spec>



Request

POST /RPC2 HTTP/1.0

User-Agent: Frontier/5.1.2 (WinNT)

Host: betty.userland.com

Content-Type: text/xml

Content-length: 181

```
<?xml version="1.0"?>
```

```
<methodCall>
```

```
<methodName>examples.getStateName</methodName>
```

```
  <params>
```

```
    <param>
```

```
      <value><i4>41</i4></value>
```

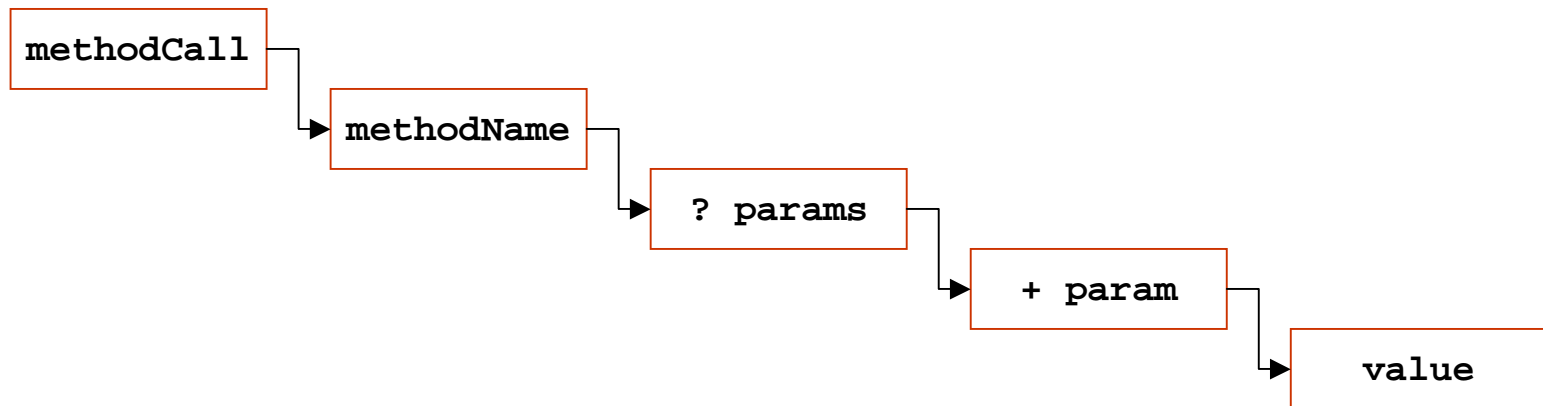
```
    </param>
```

```
  </params>
```

```
</methodCall>
```

Request

- - Method : POST
 - User Agent, Host
 - Content-Type: text/xml
- XML Payload



Request

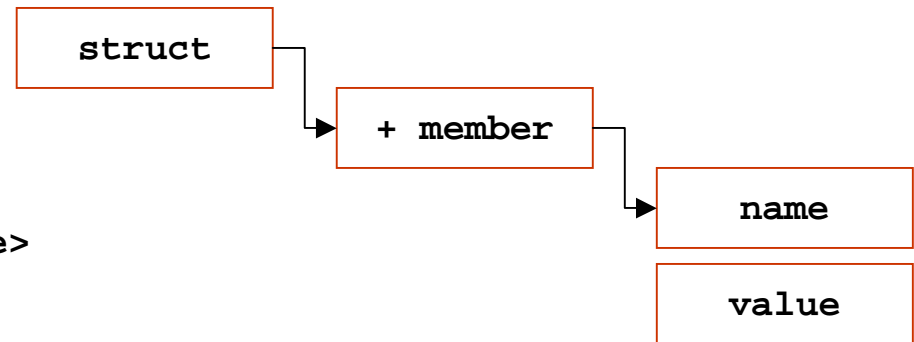
● value ()

<i4> <int>	4	-12
<boolean>	0 () 1 ()	1
<string>	ASCII	hello world
<double>		-12.214
<dateTime.iso8601>		19980717T14:08:55
<base64>	Base64	eW91IGNhbid0IHJlY WQgdGhpcyE=

Request

(<struct>
가

```
<struct>
  <member>
    <name>lowerBound</name>
    <value><i4>18</i4></value>
  </member>
  <member>
    <name>upperBound</name>
    <value><i4>139</i4></value>
  </member>
</struct>
```



Request

(**<array>**)
가

```
<array>
  <data>
    <value><i4>12</i4></value>
    <value><string>Egypt</string></value>
    <value><boolean>0</boolean></value>
    <value><i4>-31</i4></value>
  </data>
</array>
```



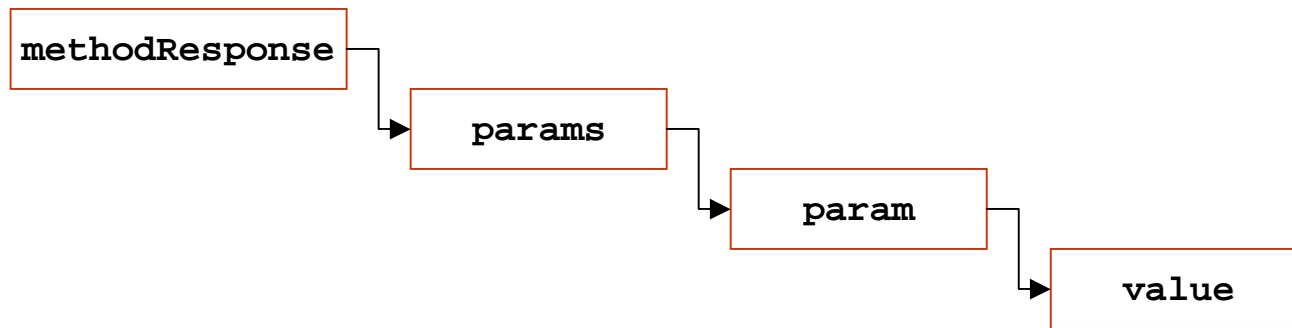
Response

●

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 158
Content-Type: text/xml
Date: Fri, 17 Jul 1998 19:55:08 GMT
Server: UserLand Frontier/5.1.2-WinNT
```

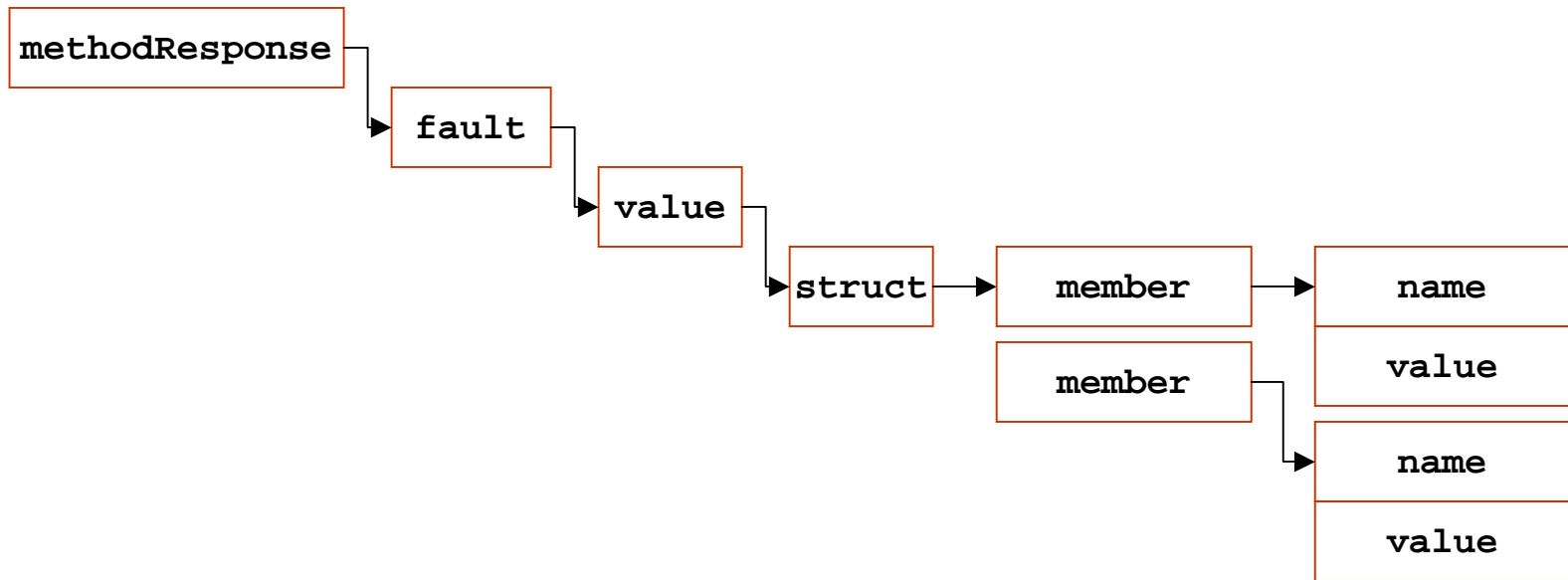
```
<?xml version="1.0"?>
<methodResponse>
  <params>
    <param>
      <value><string>South
Dakota</string></value>
    </param>
  </params>
</methodResponse>
```

Response



Response

가



Response

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 426
Content-Type: text/xml
Date: Fri, 17 Jul 1998 19:55:02 GMT
Server: UserLand Frontier/5.1.2-WinNT
```

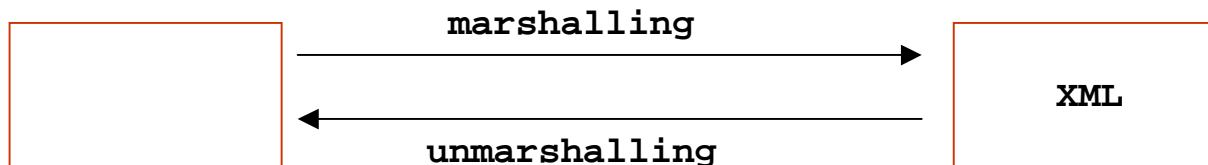
```
<?xml version="1.0"?>
<methodResponse>
  <fault>
    <value>
      <struct>
        <member>
          <name>faultCode</name>
          <value><int>4</int></value>
        </member>
        <member>
          <name>faultString</name>
          <value><string>Too many parameters.</string></value>
        </member>
      </struct>
    </value>
  </fault>
</methodResponse>
```

XML-RPC DTD

- <http://www.ontosys.com/xml-rpc/xml-rpc.dtd>

XML-RPC

- **xmlrpclib**
 - **2.2**
- **(marshalling)**
 - **XML**
- **(unmarshalling)**
 - **XML**

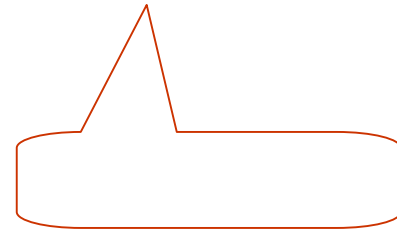


>>> import xmlrpclib
>>> print xmlrpclib.dumps((1, 'abc', [10, 20]))
<params>
<param>
<value><int>1</int></value>
</param>
<param>
<value><string>abc</string></value>
</param>
<param>
<value><array><data>
<value><int>10</int></value>
<value><int>20</int></value>
</data></array></value>
</param>
</params>

(struct)

```
>>> print xmlrpclib.dumps(({ 'one':1, 'two':2 },))  
<params>  
<param>  
<value><struct>  
<member>  
<name>two</name>  
<value><int>2</int></value>  
</member>  
<member>  
<name>one</name>  
<value><int>1</int></value>  
</member>  
</struct></value>  
</param>  
</params>
```

```
>>> import xmlrpclib
>>> print xmlrpclib.dumps((1, 'abc', [10, 20]), 'funcationCall')
<?xml version='1.0'?>
<methodCall>
<methodName>funcationCall</methodName>
<params>
<param>
<value><int>1</int></value>
</param>
<param>
<value><string>abc</string></value>
</param>
<param>
<value><array><data>
<value><int>10</int></value>
<value><int>20</int></value>
</data></array></value>
</param>
</params>
</methodCall>
```



가

```
>>> from xmlrpclib import *
>>> import sys
>>> Boolean('true?').encode(sys.stdout)
<value><boolean>1</boolean></value>

>>> DateTime(time.time()).encode(sys.stdout)
<value><dateTime.iso8601>20021230T07:22:04</dateTime.iso8601></value>

>>> Binary('acc').encode(sys.stdout)
<value><base64>
YWNj
</base64></value>
```



```
>>> import xmlrpclib
>>> xmlp = xmlrpclib.dumps((1, 'abc', [10, 20]), 'funcationCall')
>>> xmlrpclib.loads(xmlp)
((1, 'abc', [10, 20]), u'funcationCall')
```

XML-RPC

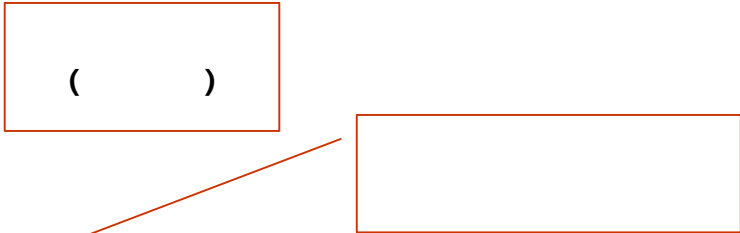
```
# xmlrpcsvr01.py
from SimpleXMLRPCServer import *

class MyRequestHandler(SimpleXMLRPCRequestHandler):
    def _dispatch(self, method, params):
        try:
            server_method = getattr(self, "do_"+method)
        except:
            raise AttributeError, "No XML-RPC procedure do_%s" % method
        return server_method(*params)

    def log_message(self, format, *args):
        print format % args

    def do_incr(self, k):
        return k+1

if __name__ == '__main__':
    server = SimpleXMLRPCServer(('', 8000), MyRequestHandler)
    server.serve_forever()
```



XML-RPC

- ```
$ python xmlrpcsvr01.py
```

- ```
>>> import xmlrpclib
```

```
>>> svr = xmlrpclib.ServerProxy('http://localhost:8000')
```

```
>>> svr.incr(2)
```

```
3
```

XML-RPC

```
# xmlrpcsvr02.py
from SimpleXMLRPCServer import *

class MyRequestHandler(SimpleXMLRPCRequestHandler):
    def _dispatch(self, method, params):
        try:
            server_method = getattr(self, "do_"+method)
        except:
            raise AttributeError, "No XML-RPC procedure do_%s" % method
        return server_method(*params)

    def do_otherObj(self, bool, datetime, bin):
        print bool, datetime, bin
        print datetime.value, bin.data
        return bool, datetime, bin

if __name__ == '__main__':
    server = SimpleXMLRPCServer(('', 8000), MyRequestHandler)
    server.serve_forever()
```


XML-RPC

```
>>> from xmlrpclib import *
>>> import time
>>> svr = ServerProxy('http://localhost:8000')
>>> bool, datetime, bin = svr.otherObj(Boolean(0), DateTime(time.time()),
    Binary('acc'))
>>> bool
<Boolean False at 81b2bc4>
>>> datetime
<DateTime 20021230T07:07:17 at 8224354>
>>> datetime.value
u'20021230T07:07:17'
>>> bin
<xmlrpclib.Binary instance at 0x82246fc>
>>> bin.data
'acc'
```

```
<Boolean False at 81a5704> <DateTime 20021230T07:17:16 at 81d627c>  
    <xmlrpclib.Binary instance at 0x81d3fc4>  
20021230T07:17:16 acc
```