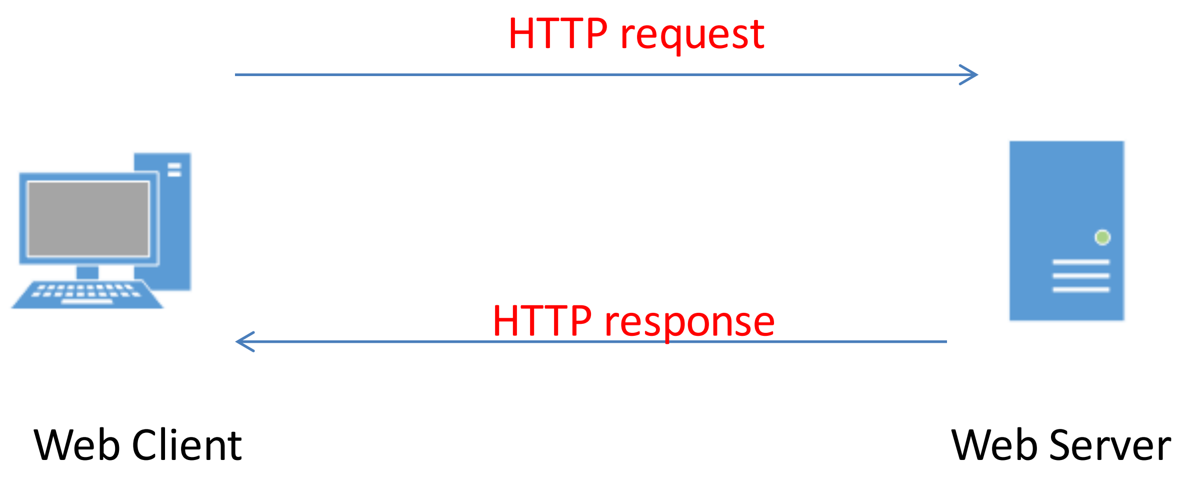
Web Application Development

REST

Contents

- HTTP Review
- JSON
- REST Principles

HTTP



- HTTP (Hypertext Transfer Protocol): is a set of rules governing the format and content of the conversation between a web client and a web server

HTTP

- The most popular protocol used on the Internet
- Text-based protocol
- Independent of operating systems and programming languages

Resources

- Servers contain resources such as document files, images, etc.
- Resources may be generated by a program running on the server

Resources

- Each resource is identified by a Uniform Resource Identifier (URI)
- Two types of URIs:
  - URL (Uniform Resource Locator): is a subset of the URIs that include a network location
    http://www.usth.edu.vn/index.html
  - URN (Uniform Resource Name): is a subset of URIs that include a name within a given space, but no location
    urn:isbn:978-0-495-82616-3

HTTP Requests

- Example of an HTTP request to fetch hello.html page from web server running on www.usth.edu.vn

```
GET /hello.html HTTP/1.1
User-Agent: Mozilla/4.0
Host: www.usth.edu.vn
Accept:text/html,application/xhtml+xml,application/xml
Accept-Language: en-us, en
Accept-Encoding: gzip
Connection: keep-alive
```

HTTP Responses

- Example of an HTTP response for a request to fetch hello.html page from web server running on www.usth.edu.vn

```
HTTP/1.1 200 OK
Date: Mon, 27 Jul 2016 23:59:59 GMT
Server: Apache/2.0.54 (Fedora)
X-Powered-By: PHP/5.0.4
Location: http://www.usth.edu.vn/hello
Content-Length: 1354
Content-Type: text/html
Connection: close

<html> <body>
<h1> <Hello World!</h1>
</body> </html>
```

HTTP Methods

Method	Description
GET	requests data from a web server by specifying parameters in the URL of the request
POST	submits data to be processed (e.g., from an HTML form) to the identified resource. The data is included in the body of the request
PUT	requests server to store the included data at a location specified by the given URL
DELETE	requests server to delete a file at a location specified by the given URL

HTTP Status Codes

Code	**Explanation**
200	OK, the document is sent from server to Web browser
302	Redirect, the requested resource is moved to another place
404	Not Found, the server can not find the requested resource
405	Method Not Allowed, the method specified in the request is not allowed
500	Internal Server Error, the request was not completed. The server met an unexpected condition

JSON

- JSON: JavaScript Object Notation
- A method for text-based data representation
- Independent of programming languages

JSON Syntax

- Data are in key-value pairs:

```
{
      "firstName": "Son",
      "lastName": "Nguyen"
}


"students":
[
     {"firstName":"Son", "lastName":"Nguyen"},
     {"firstName":"Trang", "lastName":"Nguyen"},
     {"firstName":"Loc", "lastName":"Hoang"}
]
```

JSON Values

- A number (integer or floating point)
- A string (in double quotes)
- A Boolean (true or false)
- An object (in curly braces {})
- An array (in square brackets [])
- null

JSON Example

```
<html>
<body>
    <h2> Example of creating **JSON Object** in JavaScript </h2>
    <p id="demo"></p>
    <script>
        var txt = '{"name":"Thuy Tran", "address":"HQV 18", "phone":"0951234567"} ';
        var obj = JSON.parse(txt);

        document.getElementById("demo").innerHTML =
        obj.name + "<br>" +
        obj.address + "<br>" +
        obj.phone;
    </script>
</body>
</html>
```

JSON vs. XML

- Both JSON and XML are "self describing" (human readable)
- Both JSON and XML are hierarchical (values within values)
- Both JSON and XML can be parsed and used by lots of programming languages
- Both JSON and XML can be fetched with an XMLHttpRequest
- However, XML has to be parsed with an XML parser while JSON can be parsed directly by a standard JS function

REST

- REST (Representational State Transfer) is *an architecture style* for designing networked applications
- REST is currently the predominant Web service design model
- To connect machines in the network, REST uses simple HTTP methods such as GET, POST, PUT, DELETE, etc.

REST Design Principles

- Use HTTP methods explicitly
- Be stateless
- Expose directory structure-like URIs
- Transfer XML, JSON, or both

Use HTTP methods explicitly

- To create/update a resource on the server, use POST
- To retrieve a resource, use GET
- To create/replace a resource on the server, use PUT
- To remove/delete a resource, use DELETE

Use POST to create a resource

```
POST /users HTTP/1.1
Host: myserver
Content-Type: application/xml
<?xml version="1.0"?>
<user>
  <name>Robert</name>
</user>
```

Use GET to retrieve a resource

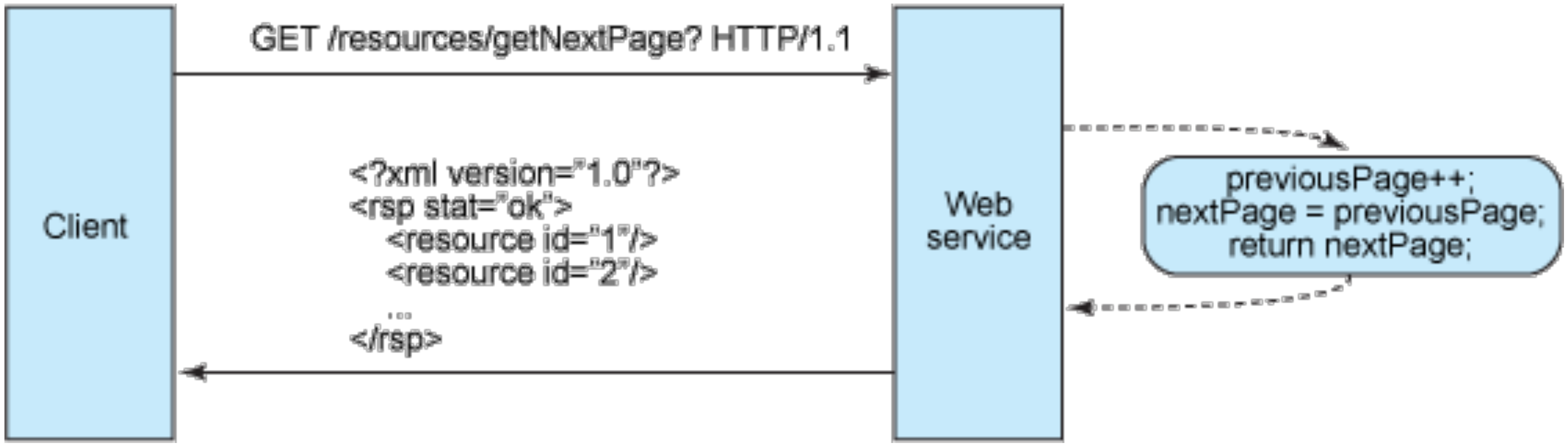
```
GET /users/Robert HTTP/1.1
Host: myserver
Accept: application/xml
```

Use PUT to replace a resource

```
PUT /users/Robert HTTP/1.1
Host: myserver
Content-Type: application/xml
<?xml version="1.0"?>
<user>
  <name>Rob</name>
</user>
```

Be stateless

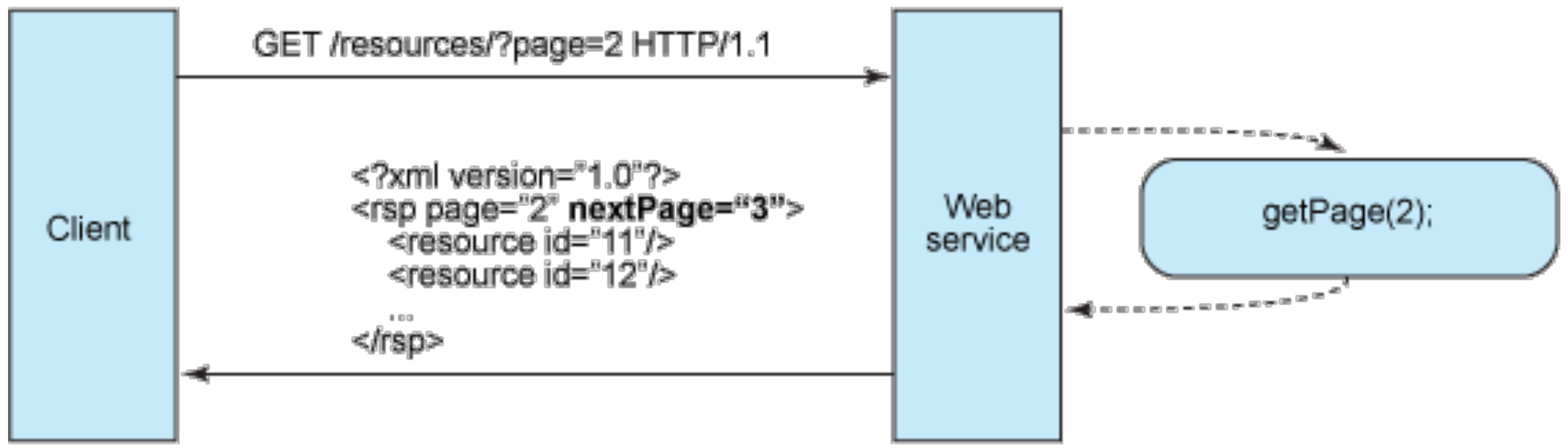
- In REST architecture, server does not keep the state of client when processing client's requests



```
<?xml version="1.0"?>
<rsp stat="ok">
   <resource id="1"/>
   <resource id="2"/>
   ...
</rsp>
```

Example of Stateful Design

Be stateless



```
<?xml version="1.0"?>
<rsp page="2" nextPage="3">
   <resource id="11"/>
   <resource id="12"/>
   ...
</rsp>
```

Stateless Design in REST

Expose directory structure-like URIs

- A hierarchical tree-like structure is used to present REST URIs. All branches are rooted at a single path where sub-paths expose different services
- For example, in a discussion threading service that gathers various topics, a structured set of URIs can be defined as follows:

http://www.myservice.org/discussion/topics/{topic}

Expose directory structure-like URIs

http://www.myservice.org/discussion/topics/{topic}

The root, **/discussion**, has a **/topics** node beneath it. Underneath that there are a series of topic names (e.g. gossip, technology), where each of which points to a discussion thread

Transfer XML, JSON, or both

- A resource representation reflects the current state of a resource at the time a client application requests it

- Following is a XML representation of a discussion thread:

```
<?xml version="1.0"?>
<discussion date="{date}" topic="{topic}">
  <comment>{comment}</comment>
  <replies>
    <reply from="joe@mail.com" href="/discussion/topics/{topic}/joe"/>
    <reply from="bob@mail.com" href="/discussion/topics/{topic}/bob"/>
  </replies>
</discussion>
```

- JSON is currently the most popular format being used in web services

Transfer XML, JSON, or both

- REST-based services use the built-in HTTP Accept header, where the value of the header is a MIME (Multipurpose Internet Mail Extensions) type

MIME-Type	Content-Type
JSON	application/json
XML	application/xml
XHTML	application/xhtml+xml