CISC 3120

C24: Web API: Passing Arguments and Parsing Returns

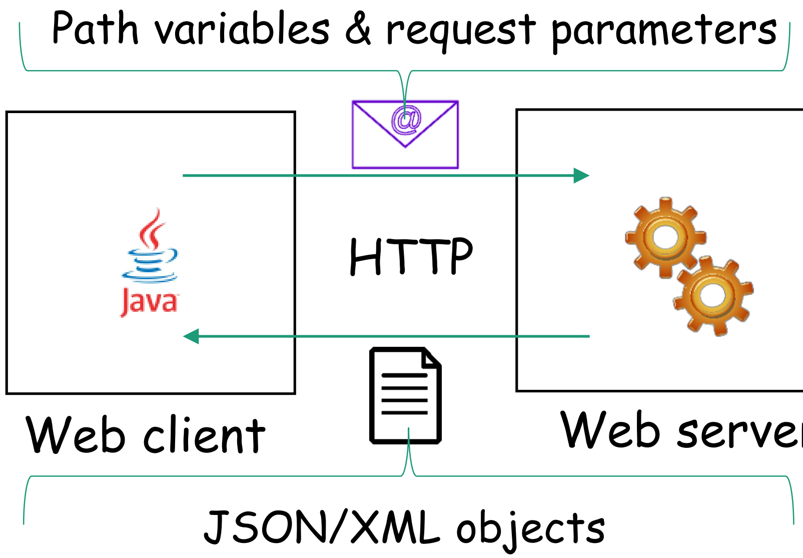
Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Outline

- Parsing arguments/data to Web server
- Parsing returned value/data from Web server



Passing Arguments via URL

- URL syntax
  - $$\text{scheme://authority}\overbrace{[\underset{\substack{\downarrow\\ \text{Path variables}}}{\text{path}}][\underset{\substack{\downarrow\\ \text{Query parameters}\\ \text{(request parameters)}}}{\text{?query}}][\#\text{fragment}]}^{\text{Contains arguments}}$$

Passing Argument via Request Body

- Request with the HTTP POST method

header

```
POST /index.html HTTP/1.1
User-Agent: Java Web Client
Host: localhost:61235
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive


......
```

body

Request method, URN, protocol version

Two blank lines

Contains arguments (request parameters)

Passing Arguments

- Use java.net.HttpURLConnection
  - Path variables
  - Request parameters
    - In URL (with the HTTP GET method)
    - In HTTP request Body (with the HTTP POST method)
- Note
  - Libraries and 3rd Party APIs may provide convenient methods or mechanisms
    - Example: jdk.incubator.http.HttpRequest
      - "incubator" are Java features that are under development
    - Example: org.apache.http.HttpRequest
      - Apache HTTP client API

Path Variables: Example

- Use String.format(…) method
- URL-encode strings for compatibility
  - Not the entire URL!
  - java.net.URLEncoder
- Example
  - final static String WEB_API_FMT = “http://example.com/%s/%d”;
  - String itemName = “blue moon”; int type = 5;
  - String urlResource = String.format(WEB_API_FMT, URLEncoder.encode(itemName, StandardCharsts.UTF_8.name()), type);
  - Example: the Address-auto-fill-by-zipcode example

Request Parameters

- General guideline for preparing the parameters
  - Form key-value pairs, separate with “&”
  - Send the key-value pairs to the server
  - Examples
    - Using a Map data structure
      - A map data structure is a list of key-value pair
      - URL-encode and key and value

URL Query Component

- URL syntax
  - scheme://authority[path][?query][#fragment]

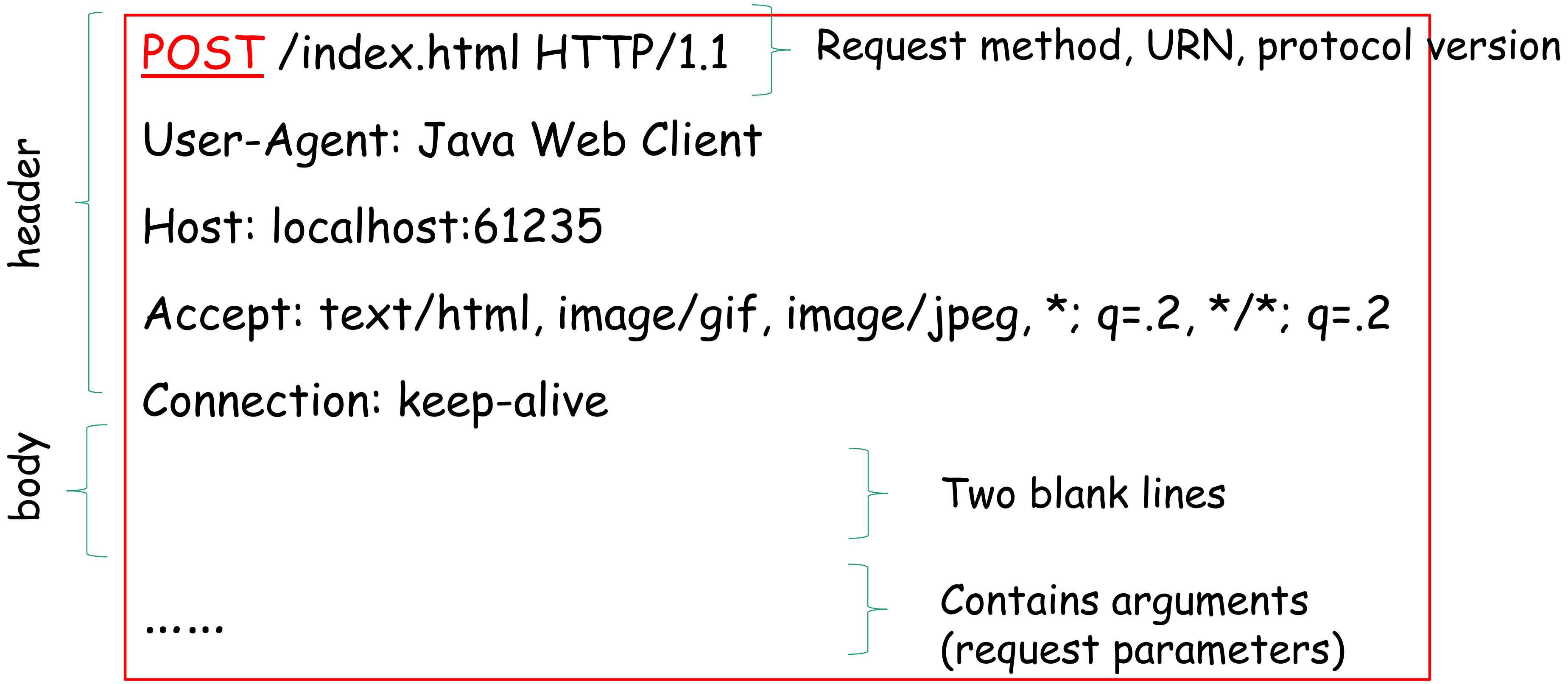
$$\Big\downarrow$$

Query parameters
(request parameters)

- Concatenate the prepared parameters (name-value pairs) with the URL
  - Example:
    - apiResource = url + “?” + preparedQuery;

Parameters in Request Body

- Request with the HTTP POST method



```
POST /index.html HTTP/1.1
User-Agent: Java Web Client
Host: localhost:61235
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive



......
```

Request method, URN, protocol version
header
body
Two blank lines
Contains arguments (request parameters)

Parameters in Request Body: HttpURLConnection

- HttpURLConnection or HttpsURLConnection
  - Example:
    - given HttpURLConnection conn = … and prepared query string in query, do,

```
conn.setDoOutput(true);
try (OutputStream out = conn.getOutputStream()) {
   out.write(query.getBytes());
}
```

Which Approach to Use?

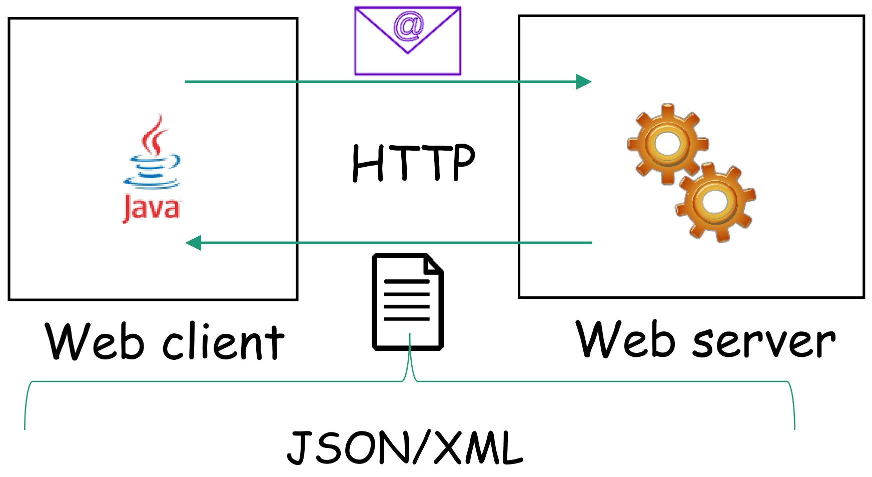
- Which method to use in your client?
- Depends on the design and implementation of Web API
  - Some uses path variables
  - Some only allows GET method (query in URL)
  - Some only allows POST method (query in REQUEST body)
  - Some support both

Questions?

- Passing arguments/data to the Web server
- Examine Web API determine which one to use
  - Path variables
  - Request parameters

Return Value in the Response

- Parsing return values in the response message
    - JSON or XML
- Only discuss JSON



JSON

- A simple data exchange format
  - Easy for humans to read
  - Easy for programs to process
  - https://www.json.org/
- Two structures
  - JSON object
  - JSON array

JSON Object

- An unordered set of name-value pairs
  - Enclosed in a pair of braces "{" and "}".
  - Name and value separated by a ":" in a name-value pair
  - Name-pairs are separated by ","
- Example

```
{
        "country": "US",
        "state": "NY",
        "city": "BROOKLYN"
}
```

JSON Array

- An unordered collection of values
  - Enclosed in a pair of brackets “[“ and “]”
  - Values are separated by “,”
- Examples

```
[
    “Brooklyn College”,
    “Hunger College”,
    “City College”,
    “Lehman College”
]
```

JSON Value

- What can be a value?
  - String: quoted character sequence
    - e.g., "Brooklyn College"
  - Number: an integer, or a float pointing number
    - e.g., 3, 3.14, 3.14e0, 0.314e1, 0.314e+1, 31.4e-1
  - JSON object, i.e., JSON objects can be nested with JSON objects or arrays
  - JSON array, i.e., JSON arrays can be nested with JSON objects or arrays
  - true
  - false
  - null

JSON Object: Example

```
{
    “club”: “Alpha Beta Gamma”,
    “president”: {“name”:”Ben Jefferson”, “gpa”:3.8},
    “vice president”: null,
    member: [
        {“name”:”Jane Doe”, “gpa”:4.0, “graduated”: false },
        {“name”:”John Doe”, “gpa”:4.0, “graduated”: true},
    ]
}
```

Dealing with JSON Objects and Arrays

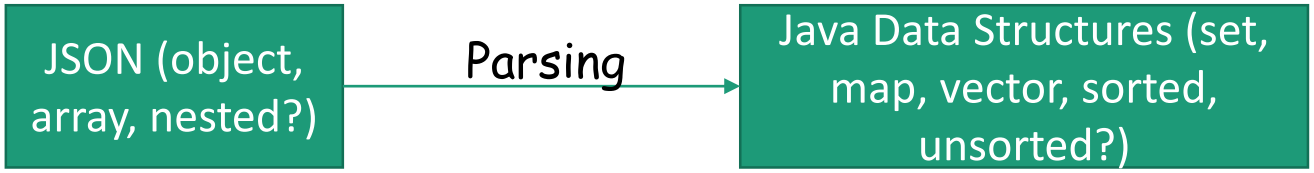
- Many APIs and libraries
  - https://www.json.org/
- In this course,
  - JSONP by Oracle
  - https://javaee.github.io/jsonp/
  - Examples
    - https://javaee.github.io/jsonp/getting-started.html

Questions?

- Dealing with JSON array/object in HTTP response?

Full Example in Application

- Full Web client example
  - Charting multiple equities price over time (using Alpha Vantage Web API)
- First, understand the format of the JSON object that Alpha Vantage API returns
- Second, determine to create JSON object or array
- Third, determine Java data structure


JSON (object, array, nested?)
Parsing
Java Data Structures (set, map, vector, sorted, unsorted?)

Main Steps in the Web Client

- Create a URL connection
- Prepare a HTTP request
- Send request
- Read response
- Disconnect

Create URL Connection

- URL's openConnection() method
- Determine if it is HttpURLConnection or HttpsURLConnection
    - Can only be one of the two, if Web

Prepare HTTP Request

- Request method: GET or POST or …?
- Path variables and request parameters
- Additional items
  - Some additional request header fields: “content-type”, “user-agent”
  - Network timeout: connect or read timeout, use default or set a desired value
  - Use Web cookie: send cookie in the request?
  - Handling redirection

Send Request

- Use connect() method
  - of the HttpURLConnection or the HttpsURLConnection object
- *As* side effect of any one of the three methods of the connection object
  - getResponseCode()
  - getInputStream()
  - getOutputStream()

Read Response

- Error or not?
  - If error, read the error message via the stream from getErrorStream()
- Parse JSON array/object

Disconnect

- Each HttpURLConnection instance is used to make a single request
- However, the underlying network connection to the HTTP server may be shared by other instances.
  - Calling the close() methods on the InputStream or OutputStream of an HttpURLConnection after a request may free network resources associated with this instance
  - but has no effect on any shared persistent connection.
- Calling the disconnect() method may close the underlying socket if a persistent connection is otherwise idle at that time.

Questions?

- Concept of JSON?
- Full application example

HTTP Cookie

- Also called Web Cookie, Browser Cookie, Cookie
- A small piece of data stored by the user agent sent from the Web server
  - HTTP is stateless
    - i.e., whenever the Server finishes sending the response, it forgets about the client
    - Cookie is invented for the Web server to remember about a client
      - Web server sends a cookie to the client (user agent)
      - The client may choose to store the cookie and send it back with the next request to the server

HTTP Cookie: Main Purposes

- Session management
  - Example: login/logout, shopping cart, game score, anything else the server should remember about the client
- Personalization
  - User preferences, themes, and other settings
- Tracking
  - Recording and analyzing user behavior

Handling Cookies

- A few classes and interfaces in the java.net package,
  - CookieHandler, CookieManager, CookiePolicy, CookieStore, and HttpCookie.

Questions?

- Concept about HTTP Cookie