Web Application Architectures

Module 5: Middleware
Lecture 3: HTTP – Request

THE UNIVERSITY of NEW MEXICO

HTTP – Client Side

Let us consider the client side of this protocol. An HTTP/1.1 request message consists of three parts:

1. Request line
2. Header
3. Message body

We will now consider each of these in turn.

HTTP – Client Side: Request Line

THE UNIVERSITY of NEW MEXICO

- The request line identifies the resource, along with the desired action (also called a "request", "verb" or "method") that should be applied to it.
- The resource is typically identified by a Universal Resource Identifier (URI).
  Note: a Uniform Resource Locator (URL) is a specific type of URI.
- There are nine request types that can be specified.

HTTP – Client Side: Request Line, Methods

THE UNIVERSITY of NEW MEXICO

1. HEAD - the response the resource would supply to a GET request, but without the response body.
2. GET – return a representation of the resource.
3. POST – submit data (e.g., from an HTML form) to the resource, where the data is supplied in the body of the request, and the result may be the creation of a new resource, or the update of an existing one.
4. PUT – submit a representation of the resource.
5. DELETE – delete the resource.
6. TRACE – Echoes back the received requested (the client can use this to see if any changes were made by intermediate servers).
7. OPTIONS – returns the HTTP methods that the server supports for the specified resource.
8. CONNECT – converts the request connection to a transparent TCP/IP tunnel (usually to facilitate SSL through HTTPS).
9. PATCH – apply partial modifications to a resource.

HTTP – Client Side: Request Line, Methods

THE UNIVERSITY of NEW MEXICO

- HEAD, GET, OPTIONS and TRACE are referred to as safe methods.
- Safe methods should not produce side effects on the server. I.e., these methods are only intended for information retrieval, and should not change the state of the server.
- Note: If a GET method is implemented in a safe way, then a browser can make arbitrary GET requests without worrying about the state of the web application. Furthermore, they can be cached.

HTTP – Client Side: Request Line, Methods

THE UNIVERSITY of NEW MEXICO

- In contrast, because the POST, PUT, and DELETE methods may cause side effects on the server, they are not considered safe methods.
- Furthermore, the PUT and DELETE methods should be idempotent, which means that multiple identical requests should have the same effect as a single request.
- Note that the safe methods, since they don't change the state of the server, are also idempotent.

HTTP – Client Side: Headers

THE UNIVERSITY of NEW MEXICO

- The HTTP message header is the primary part of an HTTP request. It contains the operating parameters of an HTTP request.
- Header fields start with the field name, followed by a colon, and then the field value. E.g., the Accept field specifies the content types that are acceptable to the client.
  Ex. `Accept: text/plain`
  The Accept-Language header specifies the languages that are acceptable to the client.
  Ex. `Accept-Language: en-US`
- Field names and values may be any application-specific strings, but a core set of fields is standardized by the Internet Engineering Task Force (IETF).
- An HTTP message header must be separated from the message body by a blank line.

HTTP – Client Side: Message Body

THE UNIVERSITY of NEW MEXICO

- The message body in HTTP request is optional.
- In an HTTP request, the message body typically includes user-entered data or files that are being uploaded to the server. It defines various characteristics of the data that is being requested. It may contain a number of fields.
- If an HTTP request includes a body, there are usually header lines in the message that describe the body.

Ex.

The `Content-Type:` specifies the MIME-type of the data in the message body, such as `text/html` or `image/gif`.

The `Content-Length:` specifies the number of bytes in the message body.