Mittuniversitetet
MID SWEDEN UNIVERSITY

HTTP and REST

Experimental Work in Intelligent IoT Networks

Stefan Forsström
Department of Computer and Electrical Engineering (DET)

Mittuniversitetet
MID SWEDEN UNIVERSITY

Overview

- History
- HTTP
- Web Services
- REST
- Tips for the Lab Project

Mittuniversitetet
MID SWEDEN UNIVERSITY

History

Mittuniversitetet
MID SWEDEN UNIVERSITY

History

- 1989 Tim Berners-Lee proposed three technologies
  - HTML: Hypertext markup language
  - URI: Uniform resource identifier
  - HTTP: Hypertext Transfer Protocol
- First web server were created in 1991
- 1993 CERN announced that all WorldWideWeb technology will be royalty free
- Web 1.0 took off in 1994 when Tim founded the World Wide Web Consortium (W3C)

Mittuniversitetet
MID SWEDEN UNIVERSITY

History

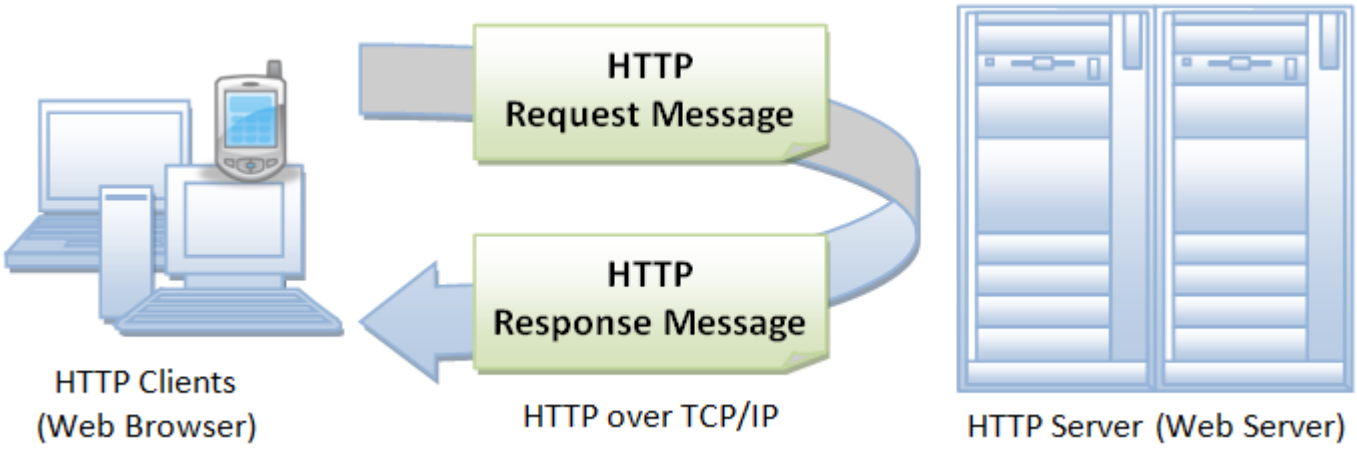
- The work on turning HTTP into an official IETF Internet standard
  - Happened between 1995 and 1999
- The first official HTTP/1.1 standard is defined in RFC 2068
  - Which was officially released in January 1997
  - In June of 1999, a number of improvements and updates were incorporated into the standard and were released as RFC 2616
- Now deprecated by
  - RFC 7230, 7231, 7232, 7233, 7234, and 7235

Mittuniversitetet
MID SWEDEN UNIVERSITY

Hypertext Transfer Protocol

Hypertext Transfer Protocol (HTTP)

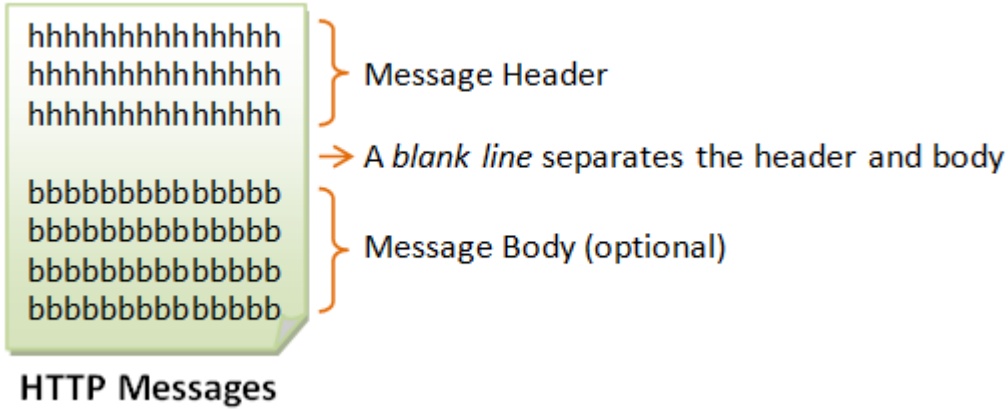
- Brief Example



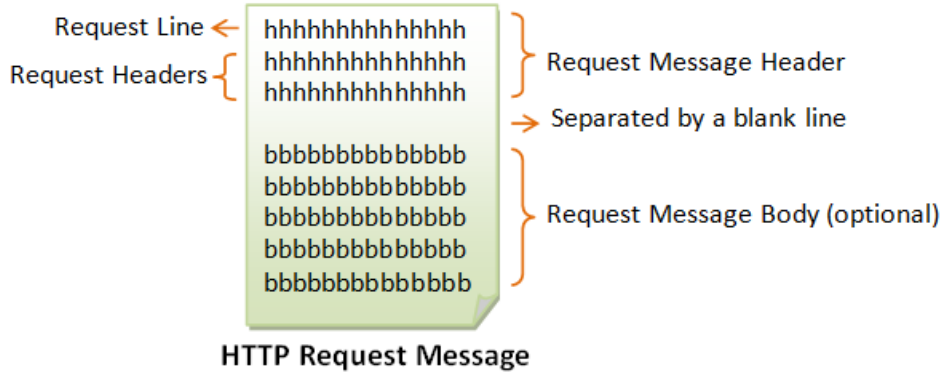
Mittuniversitetet
MID SWEDEN UNIVERSITY

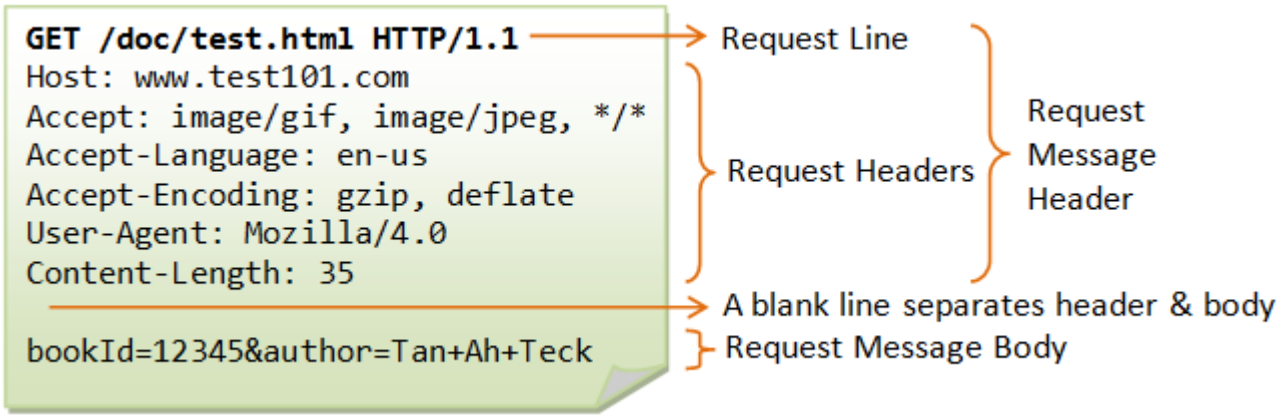
HTTP Messages

- HTTP-message structure
  - Header and Body
  - Uses CR/LF, Which is two separate ASCII characters
    - Carriage Return \r
    - Line Feed \n



HTTP Requests (Client to Server)





Mittuniversitetet
MID SWEDEN UNIVERSITY

HTTP Requests (Client to Server)

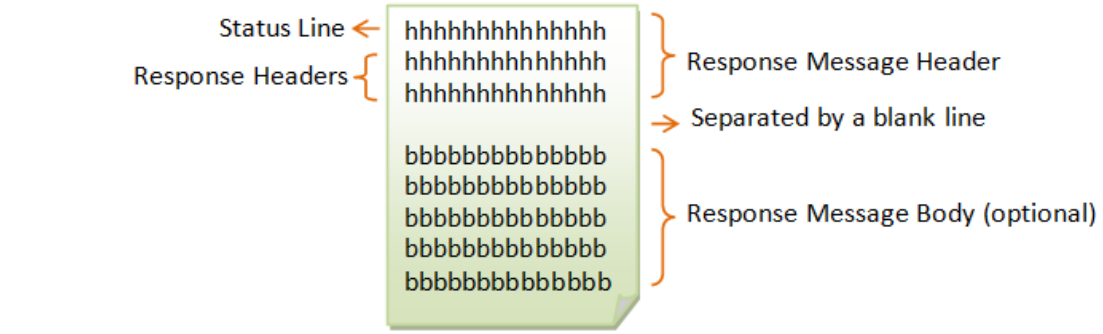
- Request-line
  - **method request-target HTTP-version CRLF**
- Defined methods in HTTP/1.1 are
  - GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, TRACE
    - GET and HEAD are the only requests a server must implement
- Headers can include for example:
  - HOST, DATE, CONTENT-TYPE, TRANSFER-ENCODING, CONTENT-LENGTH
- Can be made very simple:

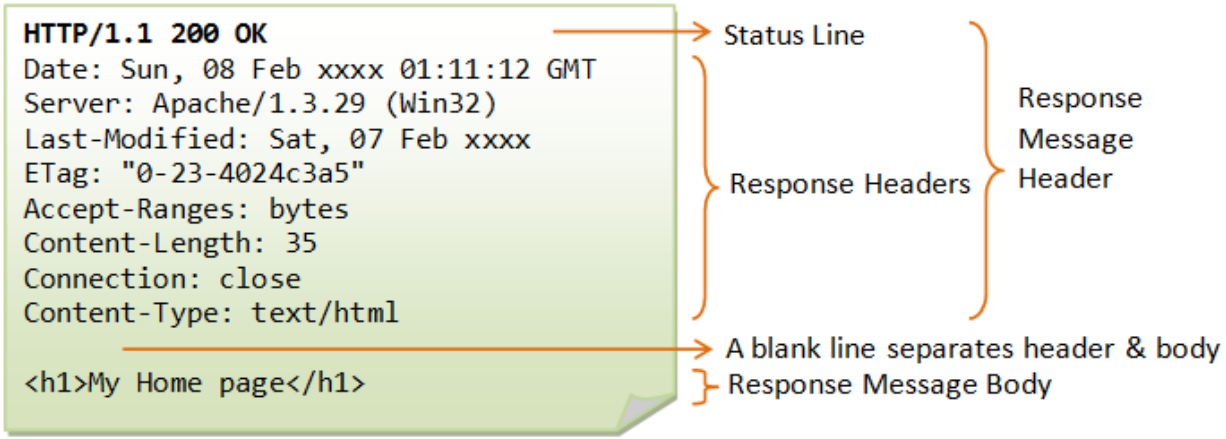
```
GET /student/ HTTP/1.1
Host: www.miun.se
```

```
GET / HTTP/1.1
Host: www.miun.se
```

```
GET /moodle/my/ HTTP/1.1
Host: elearn20.miun.se
```

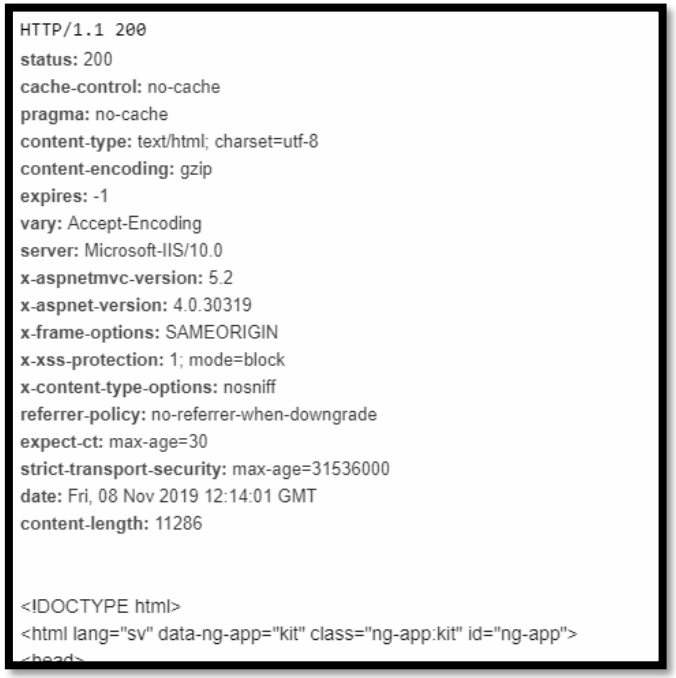
HTTP Response (Server to Client)





HTTP Response (Server to Client)

- Response-line
  - HTTP-version status-code reason-phrase CRLF
- Example
  - www.miun.se/student/



Mittuniversitetet
MID SWEDEN UNIVERSITY

HTTP Response (Server to Client)

- Status Codes
  - 1XX
    - Informal, request received, processing
  - 2XX
    - Successful, The request was successfully received, understood, and accepted
  - 3XX
    - Redirection: Further action needs to be taken in order to complete the request
  - 4XX
    - Client Error: The request contains bad syntax or cannot be fulfilled
  - 5XX
    - Server Error: The server failed to fulfill an apparently valid request

Mittuniversitetet
MID SWEDEN UNIVERSITY

HTTP Response Codes

- Cats?
  - https://http.cat/
- Or dogs?
  - https://http.dog/

- Complete list:
  - https://en.wikipedia.org/wiki/List_of_HTTP_status_codes
  - https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

Mittuniversitetet
MID SWEDEN UNIVERSITY

Hypertext Transfer Protocol Secure (HTTPS)

- Is an extension of HTTP for secure communication
  - Most commonly encrypted using TLS (or previously SSL)
  - Signed and verifiable certificates
- HTTPS can
  - Provide authentication of the accessed website
    - To know it is the right one
  - Protect the privacy and integrity of the data
    - To prevent eavesdropping and tampering when in transit
- Historically it was mostly used for payments, transactions, etc.
  - But is now used by almost all websites on the World Wide Web
  - Primarily to ensure page authenticity and keep user communications, identity, and web browsing private

Mittuniversitetet
MID SWEDEN UNIVERSITY

Web Services

Mittuniversitetet
MID SWEDEN UNIVERSITY

What are Web Services

- A Web service is defined by the W3C as:
  - "a software system designed to support interoperable Machine to Machine interaction over a network”
- Web services are Web APIs that can be accessed over a network, such as the Internet, and executed on a remote system hosting the requested services
- Many other definitions also exists
  - But in short you can say that a web service is any piece of software that makes itself available over Internet and uses a standardized messaging system

Mittuniversitetet
MID SWEDEN UNIVERSITY

What are Web Services

- There are two prominent approaches for web services
  - SOAP and REST
- SOAP = Simple Object Access Protocol
  - Publishes a service description file and which is then used to exchange standardized and well-formed XML messages with the server
  - A W3C standard
- REST = Representational State Transfer
  - Is today much more common than SOAP
  - Is a programming approach/technique
  - Follow the same “grammar” as HTTP
    - Get, post, put, delete, update…

Mittuniversitetet
MID SWEDEN UNIVERSITY

REST

- Uses the standard HTTP primitives POST, GET, PUT and DELETE
  - To and returns answers, often in the form of XML or JSON
- Coined by Roy Fielding but is not an actual standard
  - Was included as the final chapter of his 2000 PhD dissertation
- Is used in almost all commercial and over the shelf services
  - Most IoT platforms have some type of REST API
  - Because of its simplicity and existing adaptations
    - Also solves problems with firewalls, NAT, security, etc.
- A little heavy for the weakest type of IoT devices
  - Since it used TCP sockets and HTTP style headers
  - And security with HTTPS is generally quite heavy

Mittuniversitetet
MID SWEDEN UNIVERSITY

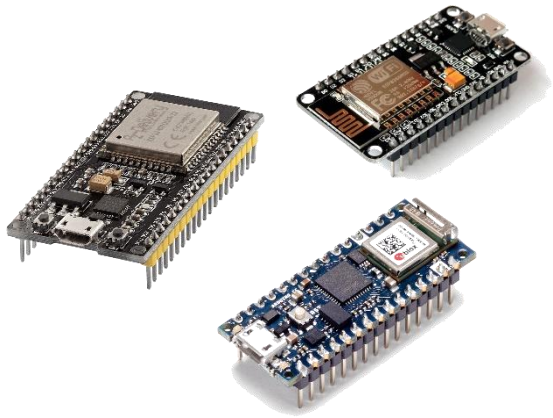
REST

- Client-Server
  - Separation of roles
- Stateless
  - No client state on server
  - (Although many use API keys)
- Cacheable
  - Some responses may be cacheable
- Layered system
  - Intermediate servers for caching, scalability and security
- Well known interface and works with many different libraries
  - Including CURL etc.

Mittuniversitetet
MID SWEDEN UNIVERSITY

REST and the IoT

- More and more IoT devices can handle WiFi, 802.11 b/g/n
  - ESP32, NodeMCU, Arduino wifi, etc.
  - (Moore's law is catching up…)



- And if they can do that
  - They can also do simple REST as well!

- Most simple one-way applications:
  - Monitoring, data visualization, etc.
  - Only require to push sensor values to the cloud
    - No need for anything fancier…

Mittuniversitetet
MID SWEDEN UNIVERSITY

REST IoT Documentation Examples

- Microsoft Azure Example

HTTP

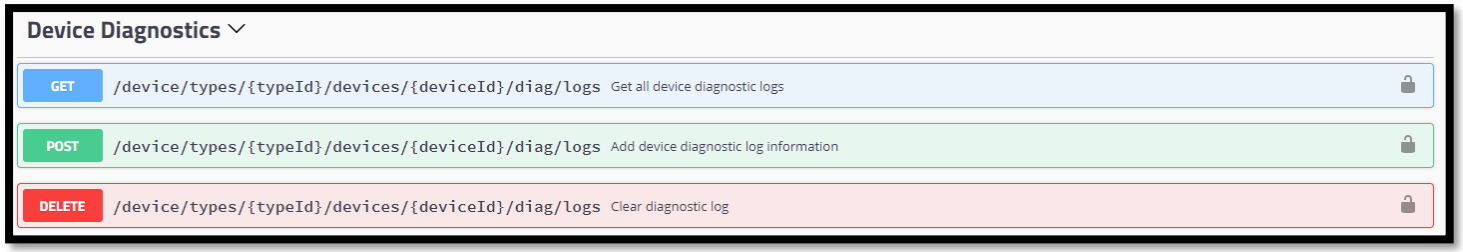
```
GET https://apps.azureiotcentral.com/api/preview/applications/{application_id}
```

- ThingBoard Documentation Example

```
# Please replace $HOST_NAME and $ACCESS_TOKEN with corresponding values.
curl -v -X POST -d "{\"temperature\": 25}" $HOST_NAME/api/v1/$ACCESS_TOKEN/telemetry --header "Content-Type:application/json"

# For example, $HOST_NAME in case of live demo server:
curl -v -X POST -d "{\"temperature\": 25}" https://demo.thingsboard.io/api/v1/$ACCESS_TOKEN/telemetry --header "Content-Type:application/json"
```

- IBM Watson IoT Example



Mittuniversitetet
MID SWEDEN UNIVERSITY

Tips for the Exercise Lab

Mittuniversitetet
MID SWEDEN UNIVERSITY

Tips for the Exercise Lab

- Start by creating a server socket listening on port 80
  - Use your browser to simply trigger an incoming /GET connection
  - Use one of the testing tools to trigger the other commands

- Upon receiving an incoming connection
  - Read and parse first line, take action depending on its contents
    - I like to use String split
  - Return the correct response code and answer

- Use simple text files to persistently save the data
  - You can even use one file per sensor/path

Mittuniversitetet
MID SWEDEN UNIVERSITY

Tips for the Exercise Lab

- Make your server scale a little better
  - By having a non main thread listen to incoming connection
  - And then move each new incoming connection to another new thread
- Reading data from socket until the end?
  - EoF is however not triggered, because thats how HTTP works…
- CR/LF
  - Remember that the protocol should use CR/LF (\r\n)
  - Meaning you can't check for \n\n because it is actually \r\n\r\n
- Almost all headers etc. can be ignored/removed
  - Focus on parsing the first line
  - Only the content length is important from the headers
    - (But some browsers doesn’t even need that…)

Mittuniversitetet
MID SWEDEN UNIVERSITY

Contact Information

STEFAN FORSSTRÖM
Assoc. Prof. Computer Engineering

MID SWEDEN UNIVERSITY
Department of Computer and Electrical Engineering (DET)
Campus Sundsvall, Room L426
Email: stefan.forsstrom@miun.se