

R-Pi

Team Emertxe



IoT Protocols

HTTP

HTTP

Introduction

- HTTP stands for hypertext transfer protocol
- Is used to transfer data across the Web
- The first version of the protocol had only one method, namely GET, which would request a page from a server
- The response from the server was always an HTML page
- There have been several versions of HTTP starting with the original 0.9 version
- Latest version is 2.0, last revised in 2015

HTTP Working

- It is a **command** and **response** text based protocol using a client server communications model
- The HTTP protocol is also a stateless protocol
 - Meaning that the server isn't required to store session information and each request is independent of the other
 - All requests originate at the client (your browser)
 - The server responds to a request
 - The requests(commands) and responses are in readable text
 - The requests are independent of each other and the server doesn't need to track the requests


HTTP

Request and Response Structure

- Request and response message structures are the same

```
generic-message = start-line  
                  *(message-header CRLF)  
                  CRLF  
                  [ message-body ]
```

Optional



HTTP Request or Response Message Format

- A request consists of: A command or request + optional headers + optional body content
- A response consists of: A status code + optional headers + optional body content
- A simple CRLF combination is used to delimit the parts and a single blank line (CRLF) indicates end of the headers
- If the request or response contains a message body then this is indicated in the header

HTTP Request

- The start line is mandatory and is structured as follow

Method + Resource Path + protocol version

- Example if we try to access the web page testpage.htm on www.testsite5.com
- The the start line of the request would be:

GET /test.htm HTTP/1.1

where:

GET is the method

/testpage.htm is the relative path to the resource

HTTP/1.1 is the protocol version we are using

HTTP

Response + Codes

- Each request has a response
- The Response consists of a
 - STATUS code And Description
 - 1 or more optional headers
 - Optional Body message can be many lines including binary data
- Response Status codes are split into 5 groups each group has a meaning and a three digit code

1xx	Informational
2xx	Successful
3xx	Multiple Choice
4xx	Client Error
5xx	Server Error

- For example a successful page request will return a 200 response code and an unsuccessful a 400 response code

HTTP

Response + Codes: Example



- Each request has a response
- The Response consists of a
 - STATUS code And Description
 - 1 or more optional headers
 - Optional Body message can be many lines including binary data
- Response Status codes are split into 5 groups each group has a meaning and a three digit code

1xx	Informational
2xx	Successful
3xx	Multiple Choice
4xx	Client Error
5xx	Server Error

- For example a successful page request will return a 200 response code and an unsuccessful a 400 response code

THANK YOU