E01 Hypertext Internet and the Web

1. Quiz

Decide whether the following statements are true or false.

Q1.1 The terms “internet” and “web” can be used synonymously

FALSE: Internet denotes a network of computers, the Web is a collection of resources (accessible via Internet)

Q1.2 Metcalfe’s law states that the effect of a network is proportional to the square of the number of users connected to it.

FALSE. Metcalfe's law states that the value of a telecommunications network is proportional to the square of the number of connected users of the system

Q1.3 Web links and Web pages form a directed graph
TRUE. Links are edges and Web pages are nodes.

Q1.4 The header field “Accept” in HTTP is sent by the server to indicate the type of requests that the server supports

FALSE: The header field “Accept” is sent by the client to indicate the fornats of message bodies that the client supports

Q1.5 A body of a response to an HTTP request is always in HTML format

FALSE: HTTP responses can be encoded in other (valid) formats, e.g. XML, JSON

2. Exercises

E2.1 For each of the following resources, decide whether it is an 'Information Resource' or a 'Non-Information Resource'.

Resource	Information Resource	Non-Information Resource
A PNG image of Tim Berners-Lee	x	
Tim Berners-Lee		x
The homepage of Tim Berners-Lee	x	
World Peace		x
The Foundation of Linked Data Syllabus	x	
The ACM Web Conference 2022		x
The Web Conference Series		x

E2.2 Resolve the following relative references in conjunction with the base URI : **http://a/b/c/d;p?q**

Relative URI	Absolute URI
/foo	http://a/foo
../baz	http://a/b/baz
./	http://a/b/c/
Foobar	http://a/b/c/Foobar
#bar	http://a/b/c/d;p?q#bar

See https://paul.ti.rw.fau.de/~xy55peqe/base.html.

E2.3 Provide absolute URIs for the following URIs without the "." and "..".

Dot URI	Resolved URI
http://localhost/foo/../bar.html	http://localhost/bar.html
http://localhost/foo/./bar.html	http://localhost/foo/bar.html
http://localhost/foo/../baz/bar.html	http://localhost/baz/bar.html

E2.4 Creating valid URIs

```
URI = scheme ":" hier-part [ "?" query ] [ "#" fragment ]

scheme      = ALPHA *( ALPHA / DIGIT / "+" / "-" / "." )
hier-part   = "//" authority path-abempty
                  / path-absolute
                  / path-rootless
                  / path-empty

authority   = [ userinfo "@" ] host [ ":" port ]
query       = *( pchar / "/" / "?" )
fragment    = *( pchar / "/" / "?" )
```

Figure 1: Subset of the generic syntax for URIs in ABNF.

```
 foo://example.com:8042/over/there?name=ferret#nose
  \_/   \______________/\_________/ \_________/ \__/
   |           |             |            |        |
scheme     authority       path        query   fragment
   |   _____________________|__
  / \ /                        \
  urn:example:animal:ferret:nose
```

Figure 2: Basic URI examples

In the RFC for URIs the generic syntax for URIs is formulated in the Augmented Backus-Naur Form notation (ABNF) (see Figure 1). The relevant syntax for this task in ABNF is:

- **/** denotes choice/alternative
- **[ ]** brackets are used to display optional content
- **( )** parentheses are used for groupings
- **n*** is the prefix for n repetitions (3*"s" = "sss"). Single ***** denotes variable repetitions.

Your task is to write three valid URIs and name their components. See Figure 2 for two demonstrative examples of a valid URI and its components based on the ABNF rules.

(1)
```
https://www.ti.rw.fau.de:443/courses/
 \___/  \___________________/\_______/
   |           |                 |
scheme     authority            path
```

(2)
```
http://142.250.184.227:80/search?q=FAU
 \__/   \________________/\_____/ \___/
  |             |             |       |
scheme       authority      path    query
```

(3)
```
https://www.google.com/search?q=FAU
```

```
      \___/   \_______________/\______/ \___/
        |             |            |       |
     scheme       authority      path    query
```

E2.5 Based on the HTTP Method descriptions for GET, POST, PUT and DELETE provided by Mozilla, explain safe and idempotent and their relation

Safe	A HTTP method is safe when it does not change the servers' internal state.
Idempotent	Identical requests have the same result
Relation	An idempotent method should also be safe because a change of the internal state might change the result. Idealistically, the method PUT is not safe, but idempotent, because it overwrites values without affecting the server's internal state, but this depends on the system.

E2.6 Provide the Reason-Phrase of the following HTTP status codes (according to RFC 7231).

Status Code	**Reason-Phrase**
200	OK
301	Moved Permanently
403	Forbidden
404	Not Found
500	Internal Server Error

E2.7 When replying to HTTP requests, servers typically provide information about the type of the response in the Content-Type header. Match the MIME (Multipurpose Internet Mail Extensions) types below with the following characteristic file extensions: **.html, .js, .mp4, .pdf, .png, .txt, .xml.**

MIME	**File Extension**
application/pdf	.pdf
application/xml	.xml
image/png	.png
text/html	.html
text/javascript	.js
text/plain	.txt

video/mp4	.mp4

3. Practices

P3.1 Install VSCode[1] and create a file with the name “test.html” file in folder “FLD-E01”. Add the line “`<h1>This is a headline.</h1>`”. Now you can open the file with your favorite browser. The URI should be valid to the File URI scheme[2].

P3.2 If we want to access the document via HTTP, we need a dummy server. Install the Live Server extension[3] for VSCode. After the installation, you can publish the file using the button with the label “Go Live” on the bottom-right. Instead of “127.0.0.1”, you can use “localhost”.

P3.3 Instead of using the Web browser, we want to retrieve the document’s content with the cURL program. Open Powershell (curl.exe) on Windows and any terminal (curl) on Linux, cURL should be already installed. Retrieve the document using the following command:

```
curl.exe -v http://127.0.0.1:5500/test.html
```

P3.4 The dummy server is fine for testing GET requests, but there is no server logic for handling the POST method. Go to the Post Test Server V2[4] and open a new random toilet. Use the below command with the POST URL of your random toilet. [Note: The V2 server is currently off, please try https://httpbin.org/post instead.]

```
curl.exe -X POST  -H "Content-Type: text/html" -d “<h1>This is a headline.</h1>” [POST URL]
```

Refresh your page afterwards and you should see that a new dump item was added.

P3.5 If you do not like to work in terminals, checkout Postman[5]. Postman provides a graphical user interface for the cURL program. Try to repeat P3.3 and P3.4 with Postman.

P3.6 There are two additional commands, that are quite handy to know when working with internet-connected applications: ping and tracert (traceroute). Ping checks the reachability of a internet resource while tracert additionally lists routing hubs. If you are unable to retrieve a Web resource, you can ensure with ping that at least your request was not lost on the way, but rejected by the server. If all ping packets are lost, you might find out with tracert, where they get lost (e.g., the router has no access to the internet). Execute the following commands in your terminal for a better understanding:

```
ping  www.fau.de
ping 131.188.16.209
```

[1] https://code.visualstudio.com/
[2] https://datatracker.ietf.org/doc/html/rfc8089
[3] https://marketplace.visualstudio.com/items?itemName=ritwickdey.LiveServer
[4] http://ptsv2.com/
[5] https://www.postman.com/

```
tracert    www.google.de
```