UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

WEB TECHNOLOGIES — LECTURE 01

INTRODUCTION TO WEB TECHNOLOGIES

Luigi Libero Lucio Starace, PhD

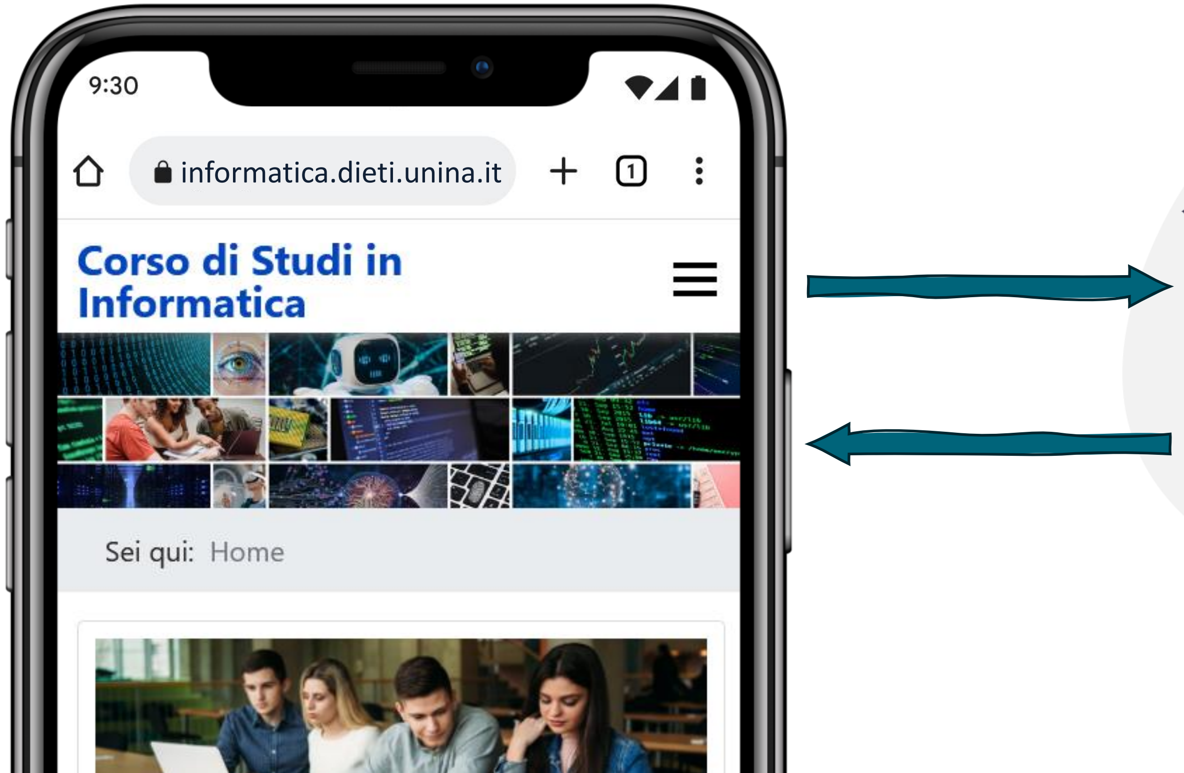
luigiliberolucio.starace@unina.it

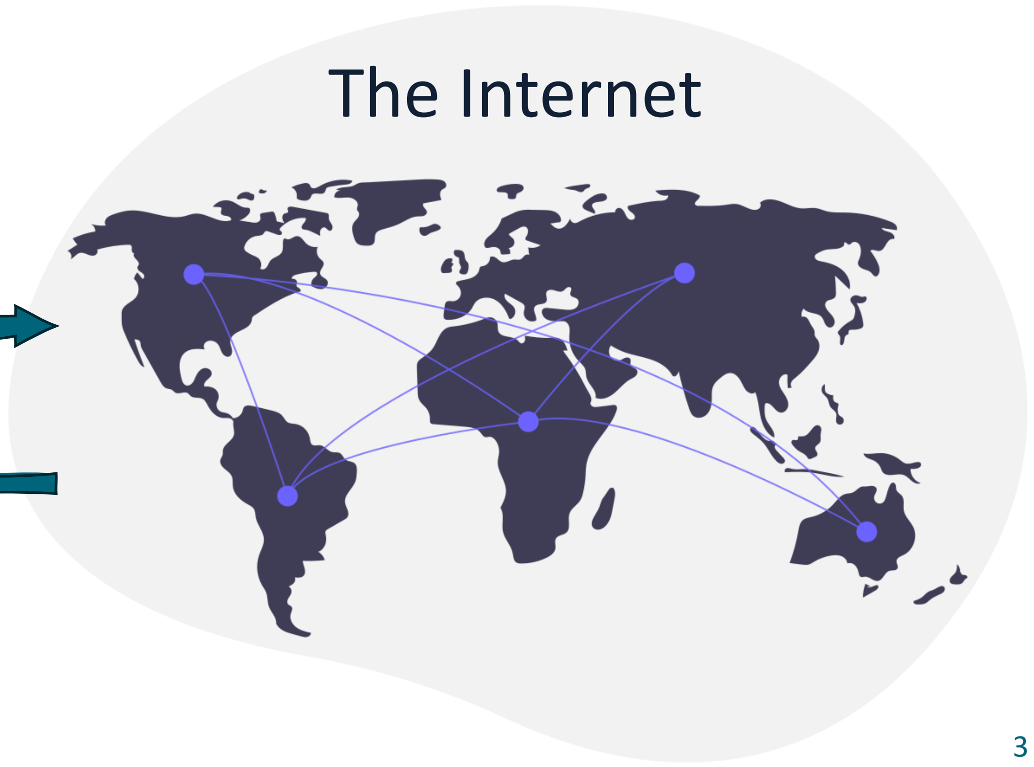
https://luistar.github.io

https://www.docenti.unina.it/luigiliberolucio.starace

WELCOME TO WEB TECHNOLOGIES!

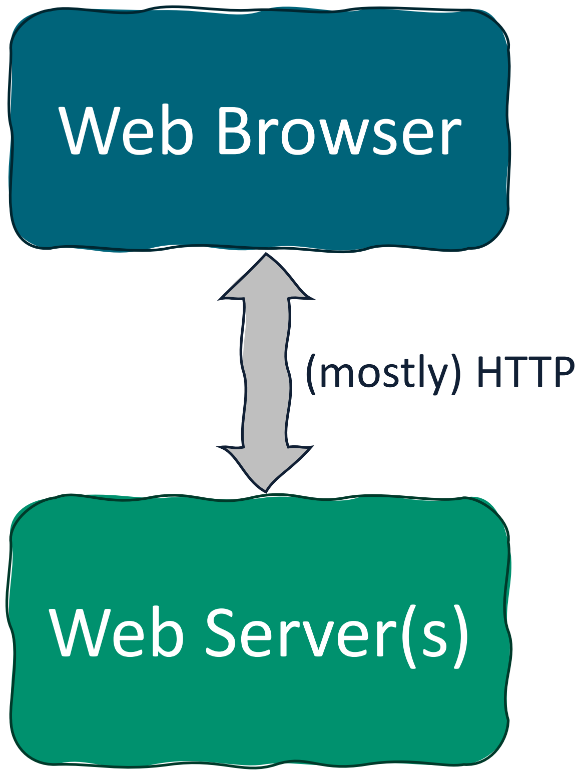
The **goal** of the course is to provide a comprehensive introduction to **fundamental concepts**, **technologies** and **tools** involved in building **modern web applications**.



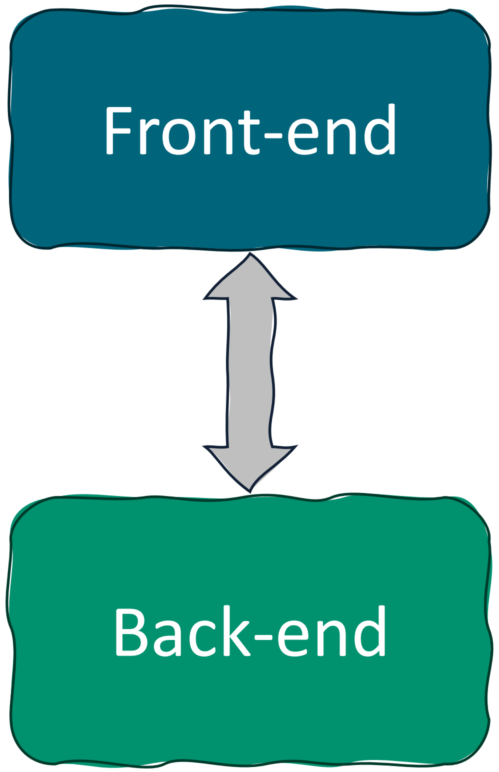


FULL STACK WEB APP ARCHITECTURE

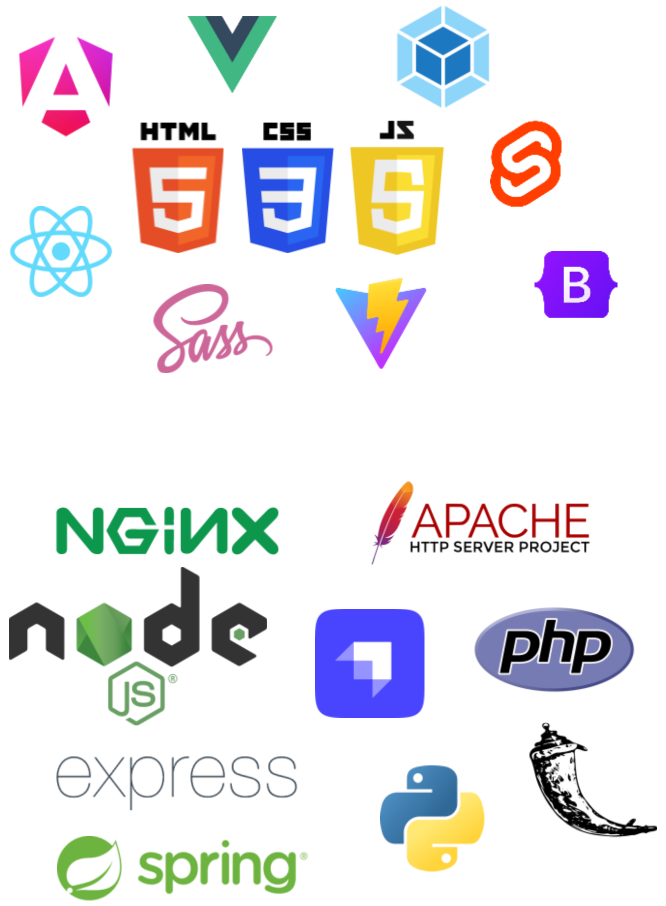
ARCHITECTURE



COMPONENTS

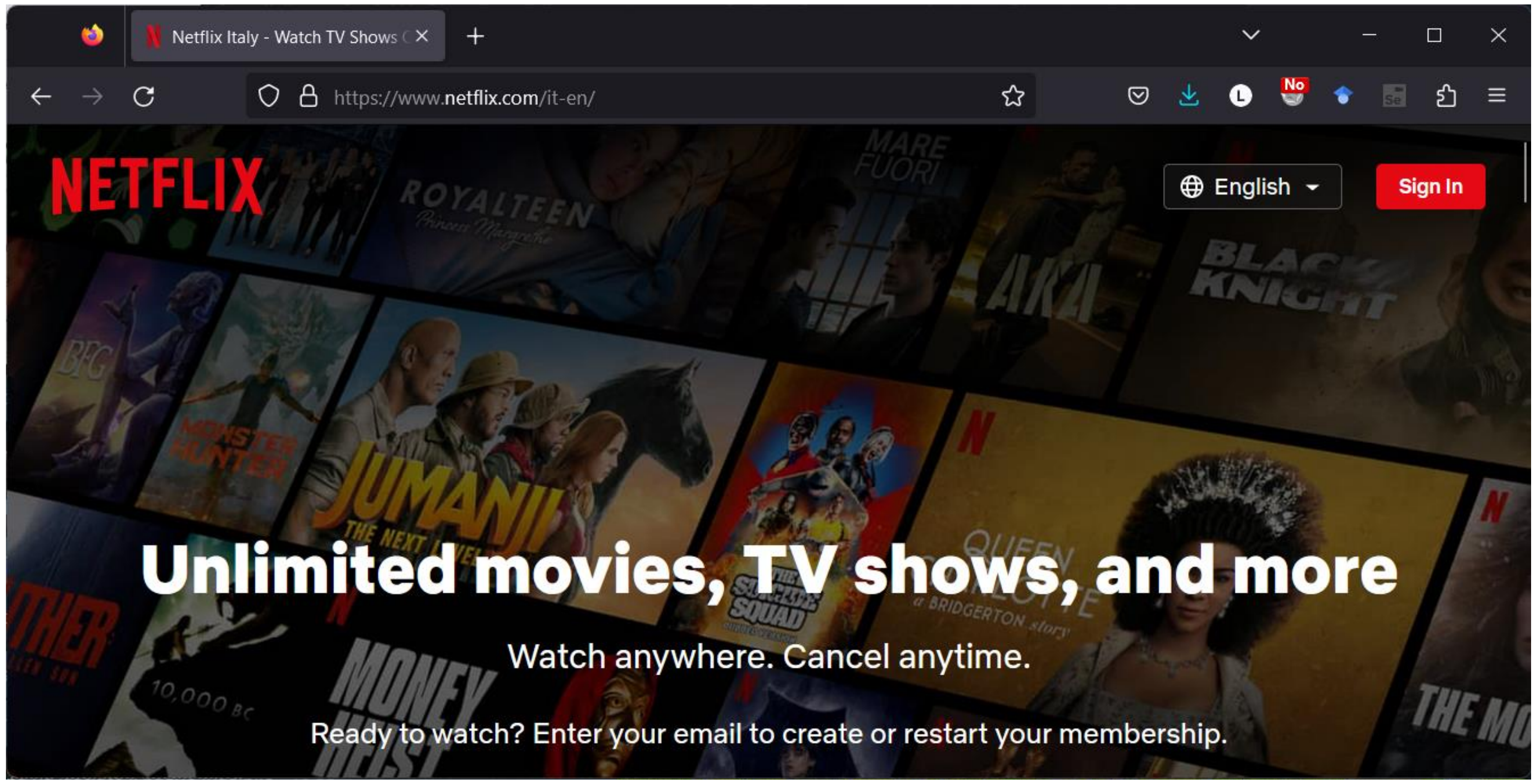


TECHNOLOGIES



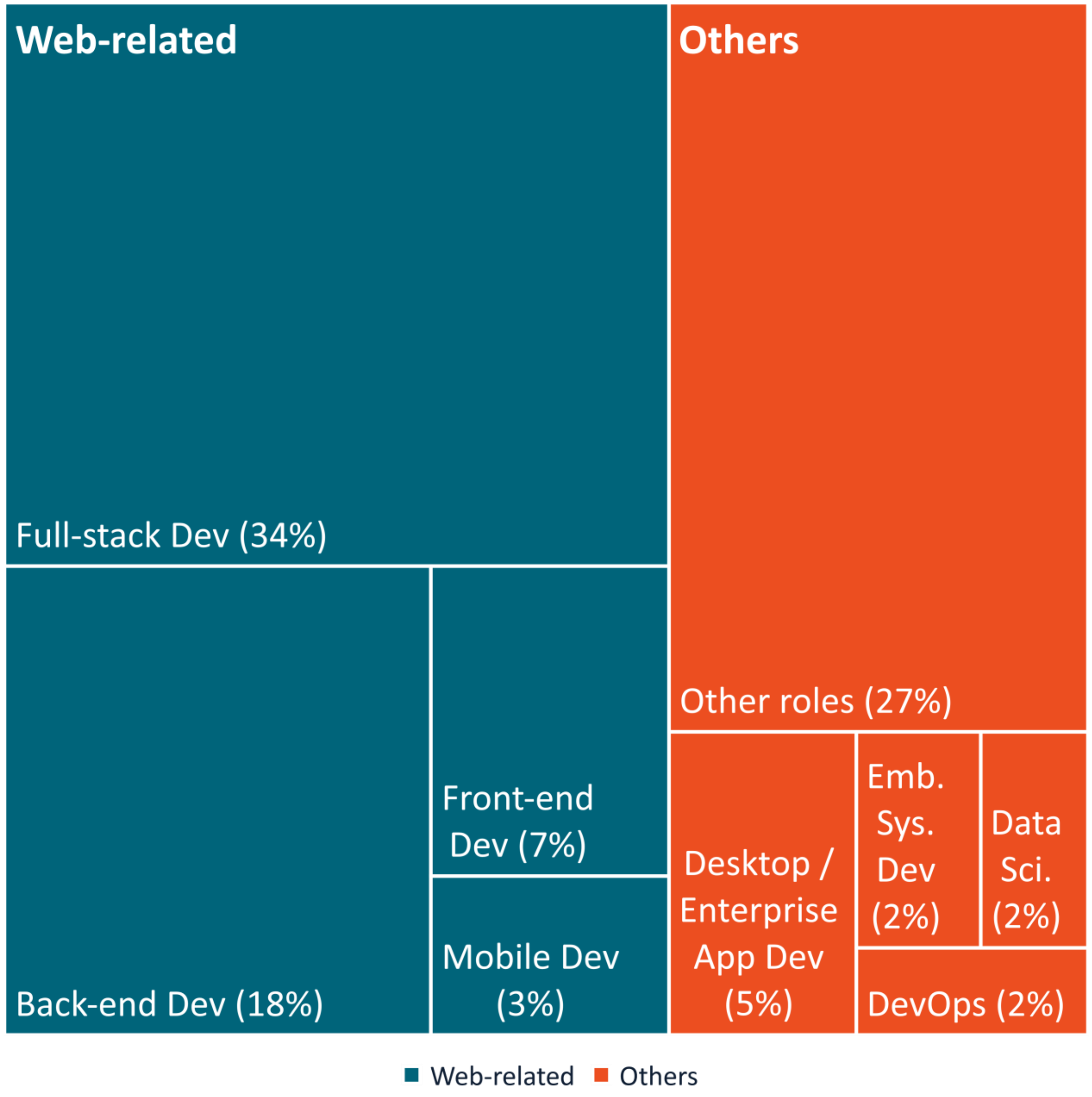
WHY BOTHER?

Everyday, we **interact** with the Web and with Web Apps


Netflix Italy - Watch TV Shows
https://www.netflix.com/it-en/
NETFLIX
English
Sign In
Unlimited movies, TV shows, and more
Watch anywhere. Cancel anytime.
Ready to watch? Enter your email to create or restart your membership.

WHY BOTHER?

- A significant part of all software developed nowadays is **web-based**
- We use web apps everyday
  - buy stuff, plan trips, reserve hotels, study, get news, watch tv shows, …
- Even when we use mobile apps
  - The app «talks» with a remote server using web technologies
  - The GUI with which we interact is often developed using web technologies!



Software Developer Roles (StackOverflow Dev Survey 2023)

LEARNING OBJECTIVES

Understand the **fundamentals** of the **World Wide Web**

Learn the basics of **full-stack** web development

Develop **responsive, secure, modern** web apps

Chose the best **technologies** and **tools** to suit your needs

Stay updated autonomously in this fast-evolving field

TEACHER

Luigi Libero Lucio Starace, PhD

luigiliberolucio.starace@unina.it

https://www.docenti.unina.it/luigiliberolucio.starace

https://luistar.github.io

- Assistant Professor (RTDa) @ UniNA since Oct. 2023
- B.Sc. and M.Sc. in Computer Science @ UniNA
- Worked as a full-stack web dev as a student
- Research topics include web application engineering and testing

THE WEB TECHNOLOGIES COURSE

- **6** credits = **24** lectures
  - Tuesday 8.30 – 10.30 CL-T-3, Friday 12.30 – 14.30 CL-T-3
  - Healthy mix of **theory** and **practice**, intertwined together
- Full course description is available (in english and in italian) on:
  - https://www.docenti.unina.it/luigiliberolucio.starace/2023/N86/14404

REQUIRED PRELIMINARY COURSES

According to the regulations of the B.Sc. in Computer Science degree:

- **Algebra** (1st year course)
- **Programming Languages I** (2nd year course)
- **Object-Oriented Programming** (2nd year course)

PREREQUISITES

- Prerequisites for understanding course concepts:
  - Basic programming knowledge
  - Understanding of the object-oriented programming paradigm
- The following are helpful (but not mandatory):
  - Basic networking concepts, client-server architectures, HTTP(S), REST from the **Computer Networks** course held in the 1st semester
  - Basic understanding of **software design principles** and **automated testing,** from the **Software Engineering** course held in the 1st semester
  - Basic understanding of **Docker** from the **Operating Systems Lab** course held in the 1st semester (useful to run some examples)

COURSE CONTENTS OVERVIEW

Course schedule (tentative):

1. 🟦 Course Introduction
2. 🟦 HTML
3. 🟦 CSS (2 lectures)
4. 🟦 JavaScript (4 lectures)
5. 🟩 Traditional Web Apps
6. 🟩 Node.js and Express (5 lectures)
7. 🟩 REST, CMS, GraphQL (2 lectures)
8. 🟦 TypeScript
9. 🟥 Front-end Tooling
10. 🟥 Single Page Apps
11. 🟥 Angular (3 lectures)
12. 🟦 Web App Security
13. 🟦 Web Testing

- 🟦 Fundamental concepts
- 🟩 Back-end
- 🟥 Front-end

COURSE MATERIALS

- There is no single official textbook
- The main materials will be the **course slides**
- Course slides will be made available the day **before** presentation
- At the end of each lecture, there will be a list of **reference materials**
  - **books**, **articles**, **documentation** and **tutorials freely available** on the Internet
- If you want to learn more about some topic, or prefer a more discursive source when studying, check out these references
- All materials are in **english**!

COURSE ASSIGNMENTS

- At the end of some lectures (in the first half of the course), you will be given an **assignment**
- Assignments are just a set of exercises designed to test your knowledge of the lecture's topics and ability to put them in practice
- These assignments are **completely optional**
  - No need to submit them, no impact on your final grade
- I **recommend** you do them as a study and self-assessment tool

WHAT ELSE YOU'LL NEED

To replicate the course examples and carry out the assignments you'll need a PC with a modern OS and the following:

Web Browser
(Firefox recommended)

Text Editor / IDE
(VS Code recommended)

Node.js Runtime
v. 20.9.0 recommended

Docker
(nice to have)

COURSE ASSESSMENT AND GRADES

To pass the Web Technologies course, you'll need to pass:

1. Written Exam → **2. Project Discussion** (after you passed the written exam)

- You will need a **passing grade** (≥ 18/30) in both parts
- The **final grade** will be determined as the **average** of the two grades

WRITTEN EXAM

- **The written exam consists in multiple-choice/open questions**
  - You may be asked to discuss some concept or methodology
  - You may be asked to write or analyze some (basic) code
  - Topics include all the contents covered during the course
- You may also pass the written exam by taking **two partial exams**
  - One **midterm partial** (tentatively some day in 22 April 2024 – 30 April 2024)
  - One **final partial** (first days of June 2024, before the first official exam date)
  - The midterm partial will focus on topics covered up to the date of the exam
  - The final partial will focus on topics covered in the second half of the course
  - You will need a passing grade (≥ 18/30) in both partials
  - The final **written exam grade** will be determined as the **average of the partials**

PROJECT DISCUSSION

After passing the written exam, you will submit and discuss a **project**

- The project consists in developing a **modern web application**
  - **Frontend Single Page App** + **REST backend**, both developed with frameworks
- Projects are to be developed **individually**
- You may choose among several alternative themes proposed by the teacher, or you may propose your own theme
- Free to use any technology you want
- You will be able to start working on the project in the second half of the course. More details will be provided later on during the course.

PROJECT DISCUSSION

After submitting your project, you will be able to discuss it.

During the discussion, you will:

- Showcase the functionality of the web app using your own laptop
- Answer some technical questions to ensure you actually did the project and understood the technologies you used
- Grades will be determined based on the discussion and on the quality of the developed web app

COMMUNICATION PROCESS

- **Teacher → Students**
  - Istitutional website: https://www.docenti.unina.it/luigiliberolucio.starace
  - Subscribe to the course, and **make sure to activate the mailing list**
  - Team on Microsoft Teams (see news on my istitutional website)
- **Students → Teacher**
  - Send me an email
    - Put «**[TECWEB]**» in the subject
    - Do not forget to say who you are!
    - **Try not to use Teams chat messages**. They are a pain to manage and I may not respond
  - Come see me during office hours
    - Details available on the Istitutional website (link above)

FEEDBACK IS IMPORTANT!

- Your feedback is **valuable** and **much appreciated**
- If there's any issue with the course, or you have suggestions for improvement, let me know ASAP!
  - Come talk to me during office hours
  - Let's talk during lecture break or before/after lectures
  - Send me an email
  - If you're really scared of me (no reason to be!) ask the student representatives to bring any issue or suggestion to my attention
- Don't forget to fill the course evaluation forms at the end of the course!

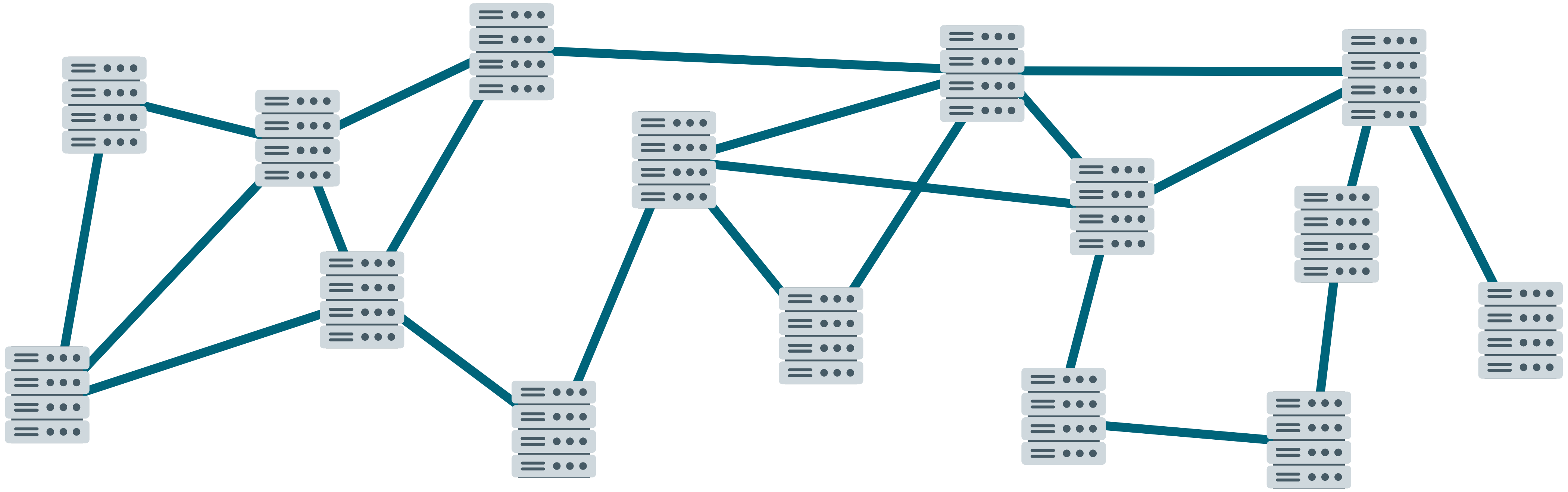
THE WORLD WIDE WEB

The Web, and its core protocol HTTP

THE INTERNET AND THE WEB

The **Internet** is a global network of interconnected computers, sharing information using Internet Protocols,

- Origins trace back to ARPANET (1969)



THE INTERNET AND THE WEB

- The **World Wide Web**, commonly referred to as the **WWW** or simply **the Web**, is a subset of the broader Internet.
  - Invented by Sir Tim Berners-Lee in the early 1990s.
  - It is a system of interconnected **hypertext documents**, which are linked to each other through **hyperlinks**.
  - Core components are **HTTP** and **HTML**

HYPERTEXT DOCUMENTS

- Traditional documents are merely sequences of characters
- Hypertexts are documents that contain also **links** (a.k.a. **hyperlinks**) to other content (e.g.: other hypertexts, documents, or media)

Fragment from «The Book of Programming»

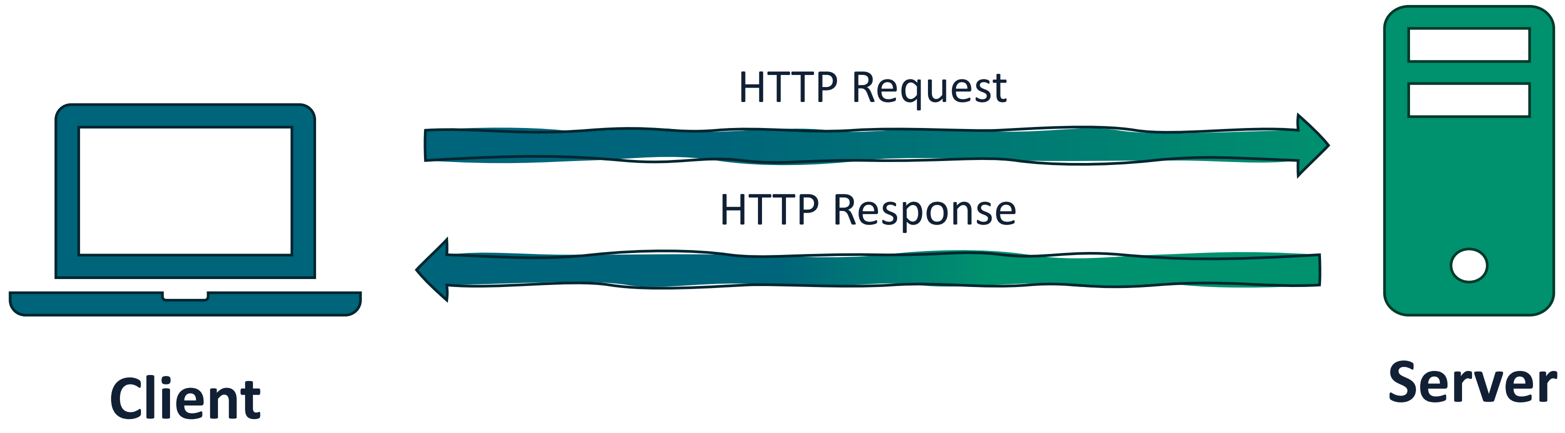
A student asked: “The programmers of old used only simple machines and no programming languages, yet they made beautiful programs. Why do we use complicated machines and programming languages?”. Fu-Tzu replied: “The builders of old used only sticks and clay, yet they made beautiful huts”.

Fu-Tzu, the legendary programmer

With a long beard and robes adorned with clever coding jokes, Fu-Tzu was a legend among programmers. It is said his first word was “printf”.

HTTP: HYPERTEXT TRANSFER PROTOCOL

- Application protocol, built on top of **TCP/IP**
- Foundation and backbone of the **WWW**
- Developed for hypertexts, nowadays used also for other **resources**
- **Client** requests a particular resource, **Server** responds.
- **Resources** are identified by their **URL**s (**U**niform **R**esource **L**ocators)



URL: UNIFORM RESOURCE LOCATOR

```
https://www.informatica.it:4242/corsi/tecweb.html
```

Scheme: specifies the **protocol** used to access the resource.

- Most common web protocols are **http** and **https**
- **HTTPS** (HTTP **Secure**) is HTTP on top of an **encrypted** connection
  - Encryption is achieved using **Transport Layer Security** (**TLS**)
  - Plays an important role in mitigating some kinds of web application attacks

URL: UNIFORM RESOURCE LOCATOR

```
https://www.informatica.it:4242/corsi/tecweb.html
```

Domain Name: domain name of the web server hosting the resource

- Made of several parts separated by dots and **read from right to left.**
- First part («**.it**») is the **Top Level Domain (TLD)**
  - The Internet Assigned Numbers Authority (IANA) maintains a list of TLDs
- Second part («**informatica**») is the **Secondary Level Domain (SLD)**
- Additional parts define **subdomains**, which are used to differentiate different content on the same domain
  - E.g.: blog.mozilla.org, informatica.dieti.unina.it or luistar.github.io

URL: UNIFORM RESOURCE LOCATOR

```
https://www.informatica.it:4242/corsi/tecweb.html
```

Port: the port to use when establishing a connection to the server

- It can be omitted if the server uses standard ports
- Standard port for HTTP is **80**, for HTTPS is **443**
- Must be specified if the server uses a non-standard port

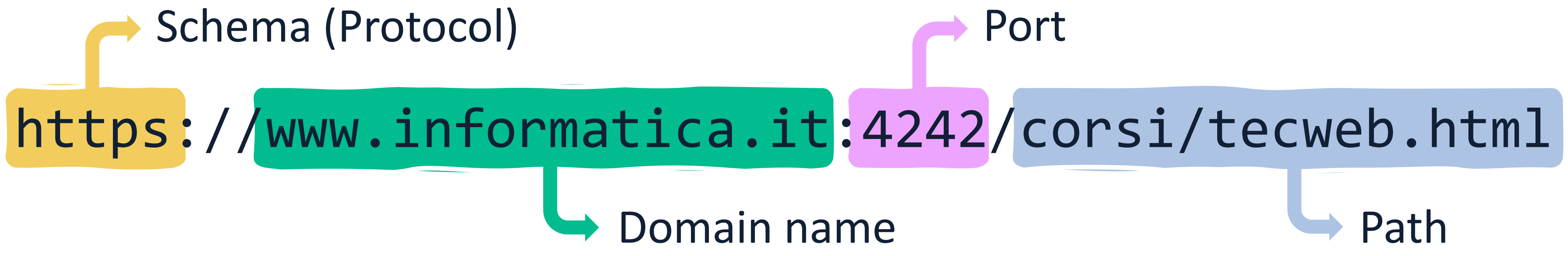
URL: UNIFORM RESOURCE LOCATOR

```
https://www.informatica.it:4242/corsi/tecweb.html
```

Path: the specific location on the server where the resource is stored

- Path is typically relative to a **web root** directory on the server
- Server only serve files within the web root directory
  - We do not want all our files to be accessible on the web!

URL: UNIFORM RESOURCE LOCATOR

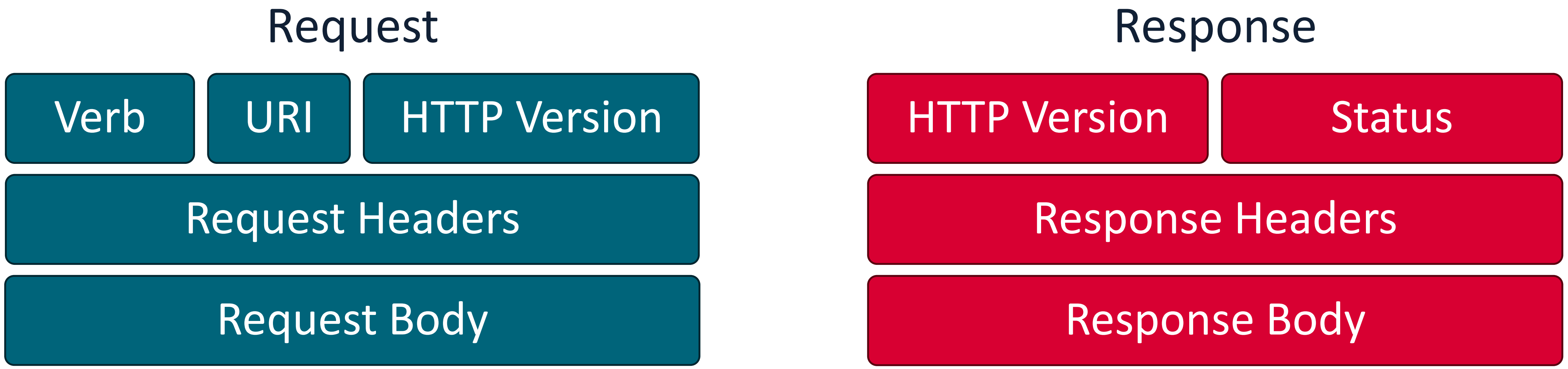


URLs may also contain **query parameters** and **anchors**

- We'll see about those in the next lecture!

HTTP MESSAGES

- HyperText Transfer Protocol
- Two types of messages: **Request** and **Response**



HTTP REQUEST METHODS (OR VERBS)

- Indicate the desired action to be performed on a given resource
- Common methods include:

Method	Description
GET	Retrieve (a representation of) a resource
POST	Submit new data to the specified resource
PUT	Replace the current resource with the specified payload
DELETE	Delete the specified resource

- Full list of HTTP Request Methods available here.

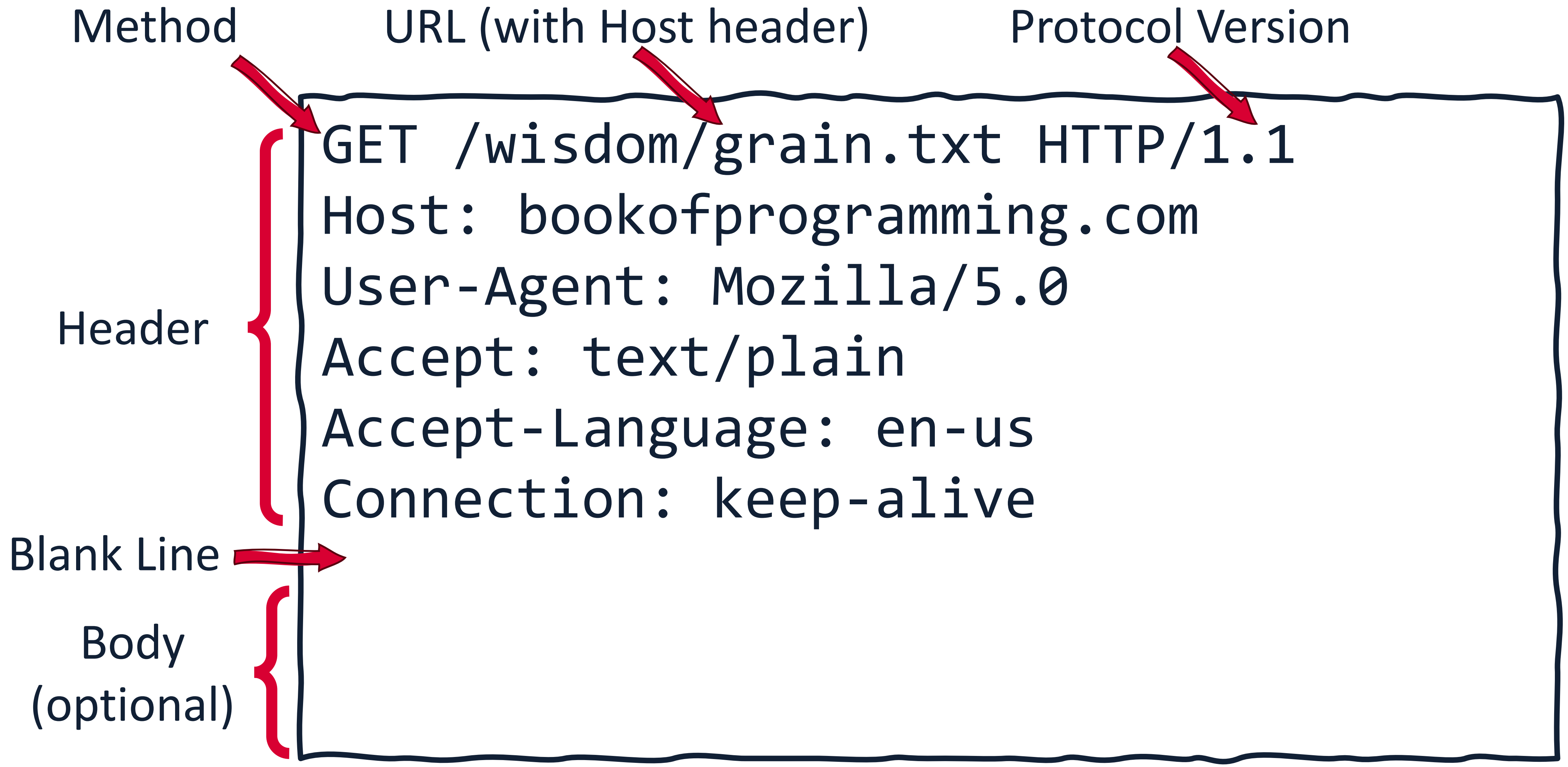
HTTP REQUEST HEADERS

- Headers are ways to pass additional information in HTTP requests and responses
- An header consists of its (case-insensitive) **name** followed by a colon (:), then by its **value**:

```
HEADER_NAME: value
```

- The IANA (Internet Assigned Numbers Authority) maintains a list of permanent and provisional headers
- It is also possible to define custom headers
- More information on MDN HTTP Headers reference

HTTP: REQUEST EXAMPLE



```
GET /wisdom/grain.txt HTTP/1.1
Host: bookofprogramming.com
User-Agent: Mozilla/5.0
Accept: text/plain
Accept-Language: en-us
Connection: keep-alive

```

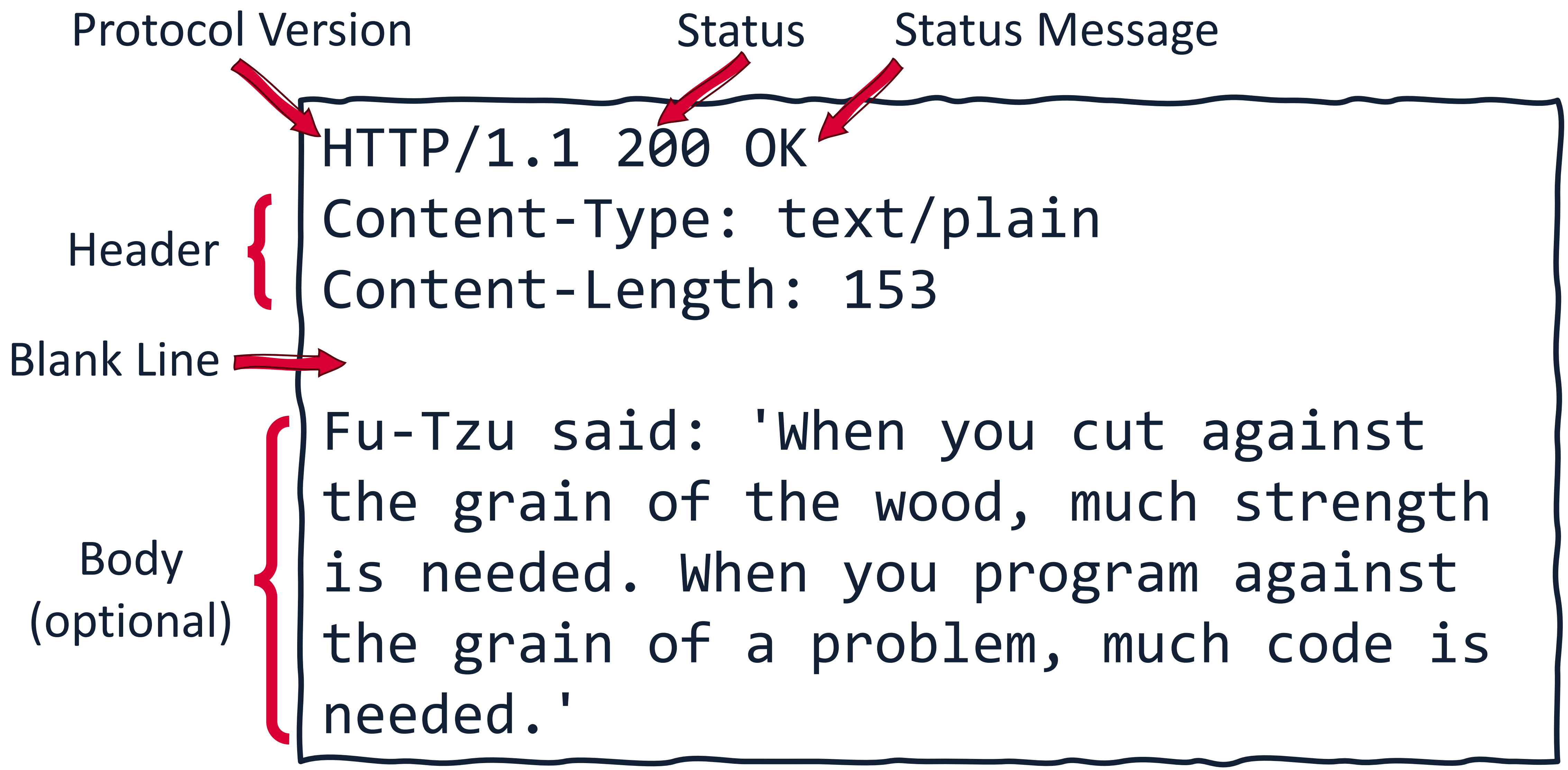
HTTP RESPONSE STATUS CODES

- Indicate whether a request has been successfully completed
- Response status codes are grouped in five classes

Informational (100-199)	**Success** (200-299)	**Redirection** (300-399)	**Client Error** (400-499)	**Server Error** (500-599)
100 Continue	200 OK	301 Moved	400 Bad Req.	500 App. Error
			403 Forbidden	503 Unavail.
			404 Not Found	

- More details available here.

HTTP: RESPONSE EXAMPLE



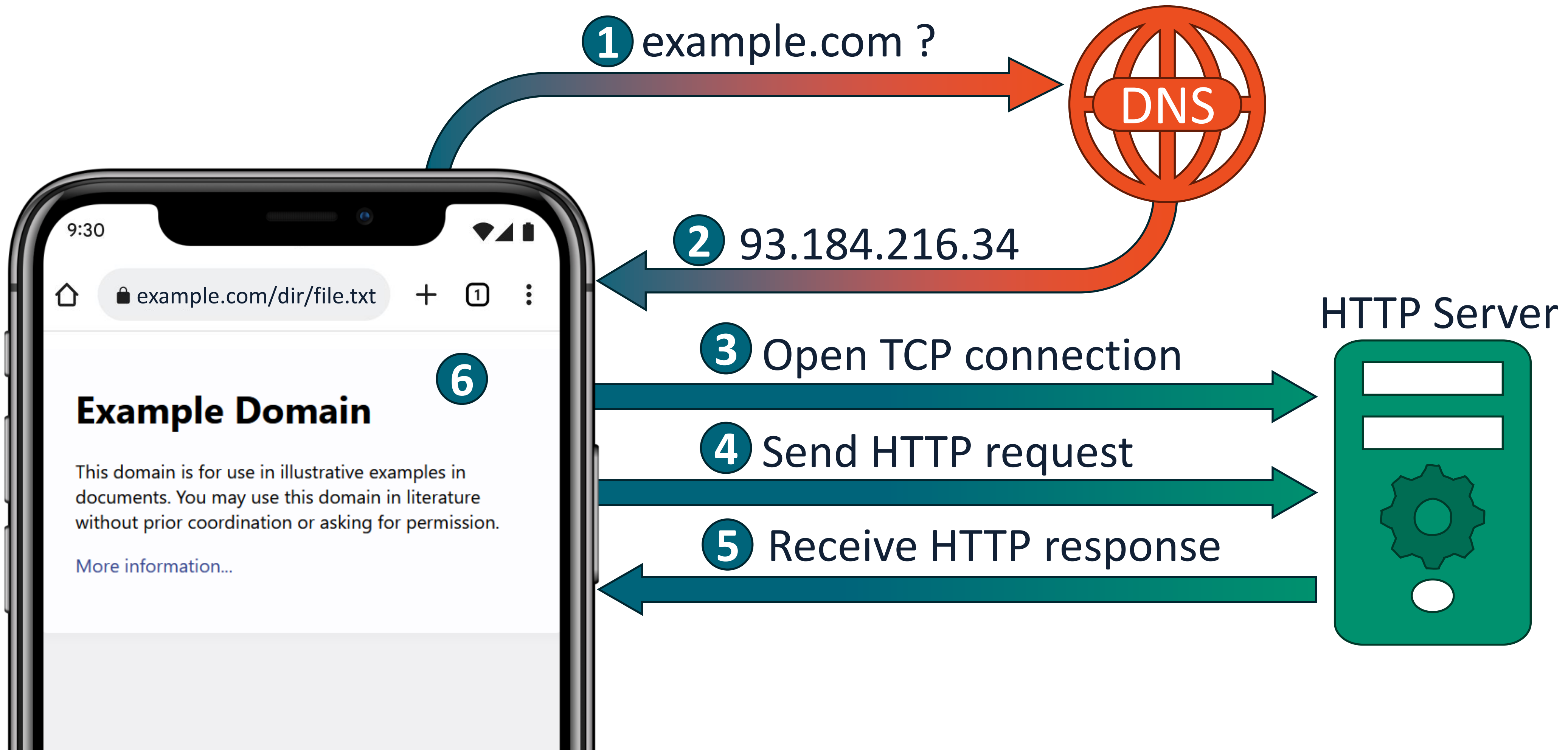
```
HTTP/1.1 200 OK
Content-Type: text/plain
Content-Length: 153

Fu-Tzu said: 'When you cut against
the grain of the wood, much strength
is needed. When you program against
the grain of a problem, much code is
needed.'
```

HTTP: STATELESSNESS

- HTTP is a **stateless** protocol
  - Each request is **independent** from previous ones
  - Server does not retain information about prior requests from a client
- Specific mechanisms (e.g.: **cookies**) need to be put in place to allow **stateful communication** in an otherwise stateless protocol
  - We will learn about Cookies in a few lectures!

HTTP FOR WEB BROWSING: OVERVIEW

When we type http://example.com/dir/file.txt in a Web Browser:



HTML: HYPERTEXT MARKUP LANGUAGE

HTML is the standard for representing hypertext documents in the Web

- The web pages we interact with in web browsers are **documents** defined using **HTML**
- HTML allows us to define web pages with headings, paragraphs, images, lists, tables, and more
- We'll get to know HTML in the next lecture!

REFERENCES (1/2)

- **Introduction to Web Applications Development**
  By Carles Mateu
  Freely available on archive.org under the GNU Free Documentation Licence
  Relevant parts: Module 1 (Introduction to Web Applications)
- **How the web works**
  MDN web docs
  https://developer.mozilla.org/en-US/docs/Learn/ Getting_started_with_the_web/How_the_Web_works
- **What are hyperlinks?**
  MDN web docs
  https://developer.mozilla.org/en-US/docs/Learn/Common_questions/Web_mechanics/What_are_hyperlinks
- **What is a URL?**
  MDN web docs
  https://developer.mozilla.org/en-US/docs/Learn/Common_questions/Web_mechanics/What_is_a_URL

REFERENCES (2/2)

- **An overview of HTTP**
  MDN web docs
  https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview
- **What is the difference between webpage, website, web server, and search engine?**
  MDN web docs
  https://developer.mozilla.org/en-US/docs/Learn/Common_questions/Web_mechanics/Pages_sites_servers_and_search_engines