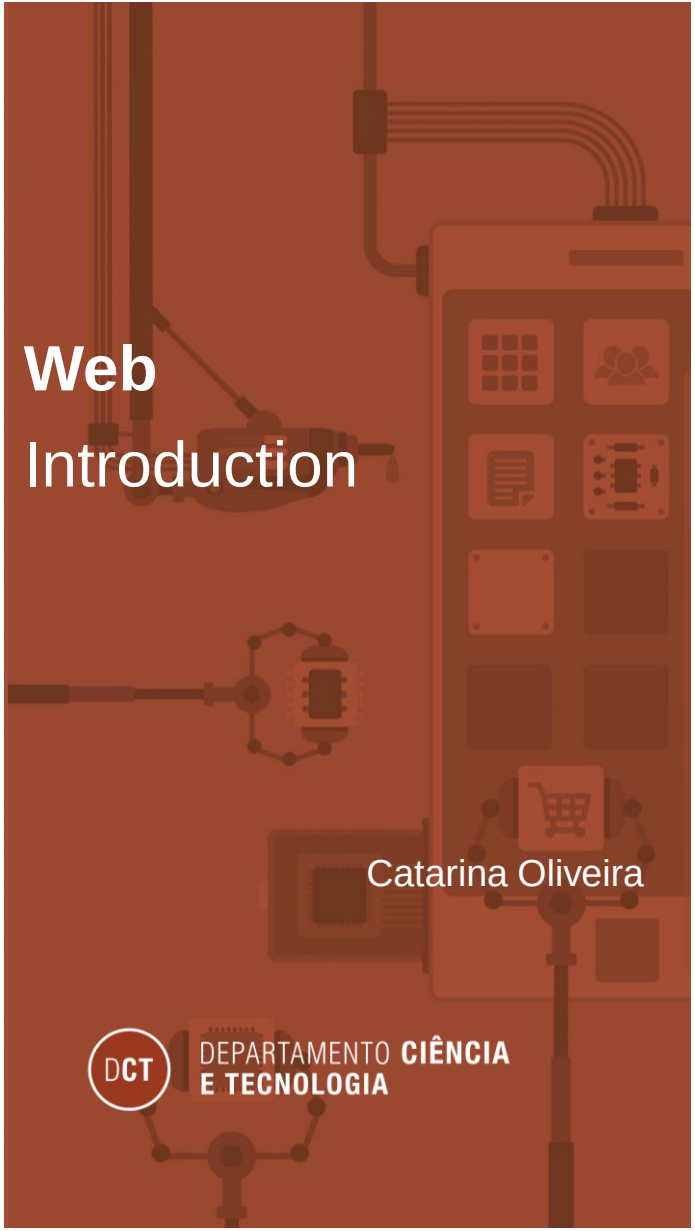


UPT UNIVERSIDADE PORTUCALENSE

CONTEÚDO

1. Internet History
2. The World Wide Web (www)
3. How it works
4. Web 1.0, 2.0, 3.0 and 4.0
5. The Internet is a global network
6. Client-Server model
7. URLs
8. HTTP protocol
9. HTTP messages
10. Web languages
11. Front-end vs Back-end
12. Web

História da Internet

A evolução da comunicação

por Carlos Diniz

1957 – A Russia lança o satélite Sputnik. Em resposta os EUA criam a Advanced Research Projects Agency (ARPA)

1961 – Desenvolvimento da transmissão de dados por pacote

1962 – J.C.R. Licklider cria o conceito da chamada "Rede Galáctica"

1969 – Nasce a ARPANET.

1969 – Foi estabelecida com sucesso ligação entre os modems de dois computadores distantes.

1969
Tenex

1971 – A Xerox lança o protótipo de um computador pessoal.

1972 – Surge o The Creeper, o primeiro vírus para computador.

1972 – Ray Tomlinson cria a primeira aplicação de envio de email.

1973 – É feita a primeira conexão continental entre os EUA e a Noruega.

1974
Apple I

1977 – Criação do protocolo de transmissão TCP/IP

1978 – A empresa DEC envia 393 emails para funcionários da Arpanet. Nasce o SPAM.

1981 – É lançado o PC da IBM.

1983 – A ARPANET cria a MILNET, uma divisão só para assuntos militares.

1988 – Jarkko Oikarinen cria o canal de chat IRC.

1985
Windows 1.x

1989 – Tim Berners-Lee propõe o sistema World Wide Web

WWW

A história continua...

1990 - Portugal adere à Internet.
1991 - É aberto o uso da rede a entidades comerciais.
1994 - Surge o MOSAIC, primeiro navegador gráfico.
1998 - Ano de lançamento do Google e do NAPSTER.
2004 - Mark Zuckerberg e os seus colegas de quarto lançam o Facebook.
2005 - Chad Hurley, Steve Chen, e Jawed Karim fundam o Youtube.

The World Wide Web (www)

WWW: Space of global information where the web resources are identified by URLs, linked by hiperlinks or links and accessible via the Internet.

URL: Uniform Resource Locator

Tim Berners Lee, 1990, “now”

Robert Cailliau, 1990, “now”

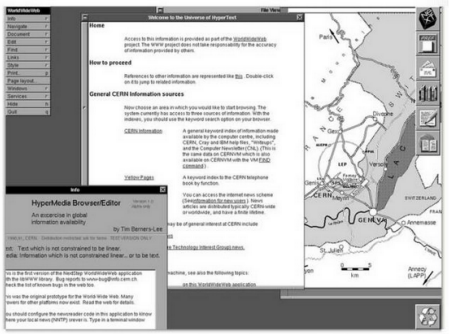
CERN (*Conseil Européen pour la Recherche Nucléaire*), 1990

- Tim Berners Lee e Robert Cailliau
  - Published a proposal on how to manage information through a hypertext distributed system (https://www.w3.org/History/1989/proposal.html)

- To operationalise the proposed concepts, Tim Berners-Lee created:
  - **WorldWideWeb**, the first web browser (https://www.w3.org/People/Berners-Lee/WorldWideWeb.html)
  - **HTML**, *HyperText Markup Language* (https://www.w3schools.com/html/html_intro.asp)
  - **HTTP**, *HyperText Transfer Protocol* (https://www.w3schools.com/whatis/whatis_http.asp)
  - **CERN httpd**, Web server (https://www.w3.org/Daemon/)
  - (4 years later) **W3C**, World Wide Web Consortium, to regulate and standardise the technologies (https://www.w3.org)

DECEMBER 1990

The world's first browser/editor, website and server go live at CERN



By Christmas 1990, Sir Berners-Lee had defined the Web's basic concepts, the html, http and URL, and he had written the first browser/editor and server software. info.cern.ch was the address of the world's first web server, running on a NeXT computer at CERN. The world's first web page address provided information about the World Wide Web project.

https://timeline.web.cern.ch/timeline-header/89

How it works

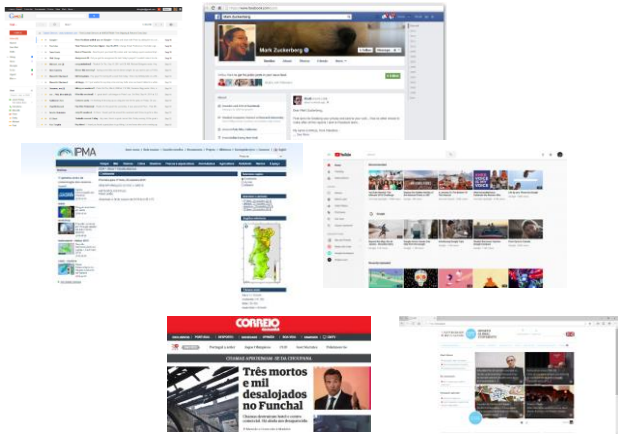
Browser
- Download
- Interpret
- Render

access

Documents
- Texts
- Images
- Videos
- Sounds
- Hipertext
- Hiperlinks

Allow “browsing”

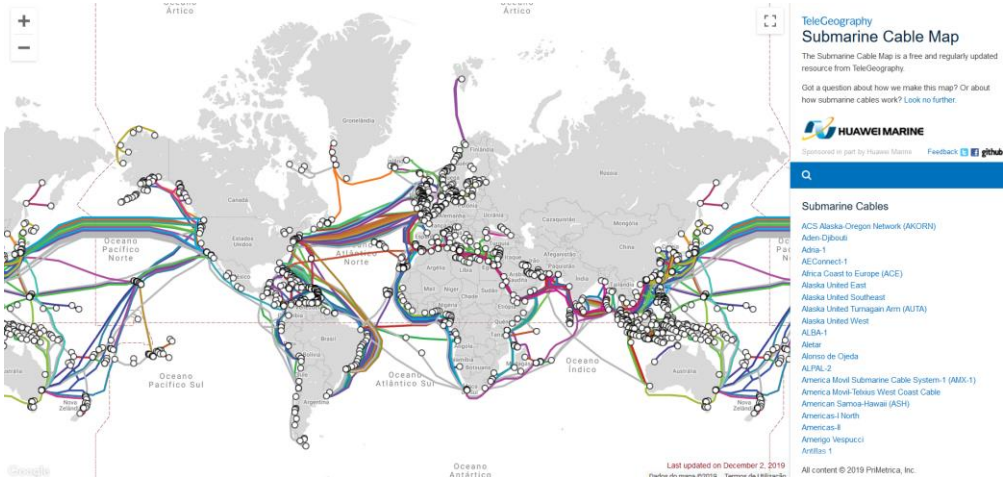
Managed by the HTTP protocol

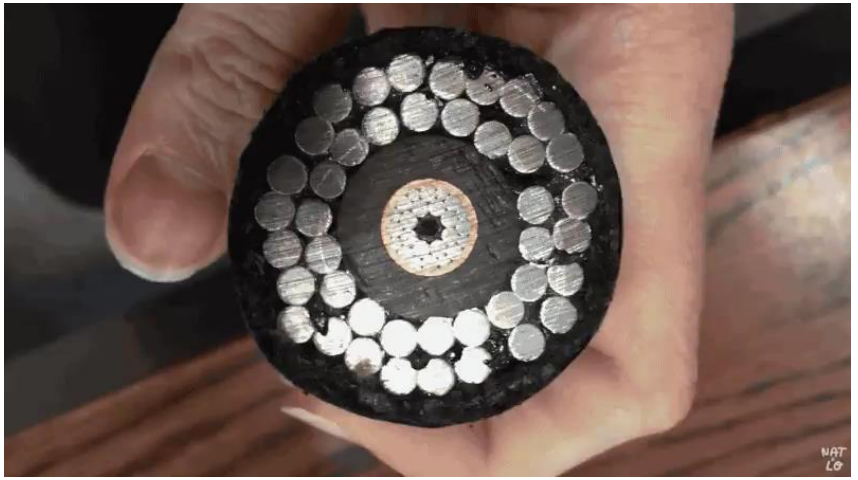


Web 1.0, 2.0, 3.0 e 4.0

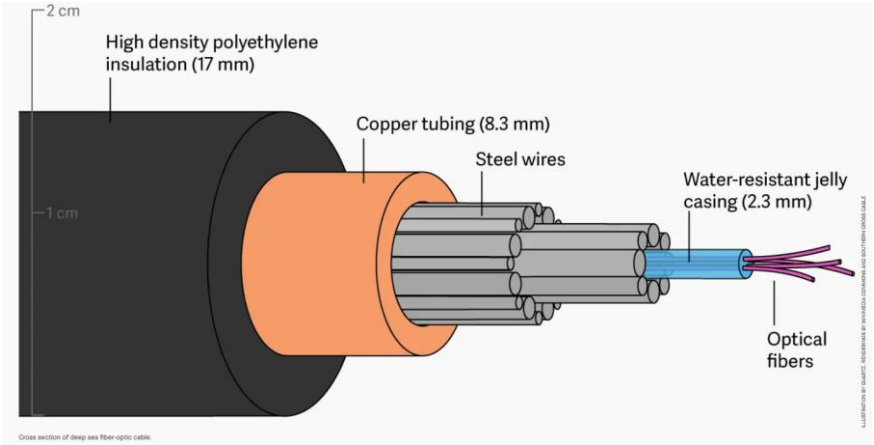
	Web 1.0	Web 2.0	"The Internet of things"	Web 3.0	Web 4.0	Web 5.0
	1994 - 2000	2000 - 2010		2010 – 2020	2020 - 2030	2030 - …
	"The Information Web"	"The Social Web"		"The Semantic Web"	"The intelligent Web"	"The Telepathic Web" "The Symbionet Web"
Focus	Read-only. Static content.	Read-write. Dynamic content.	Device comunication	Recommendation systems	Artificial Intelligence	Cerebral implants
Interaction	Users can not create or interact with the websites	Users can create and interact with websites and other users	Connection between "smart" devices and the internet through the "cloud"	Smart connection between people and machines	Smart connection between machines	EVERYTHING is connected
Examples	Altavista, Geocities, Hotmail, Yahoo! Google	Blogs, Facebook, YouTube, Wikipedia	Visit: https://www.postscapes.com/internet-of-things-examples/	Amazon, YouTube	Computers used as personal assistentes, virtual reality, holograms, implants, …	Ability to comunicate with the internet only by thinking. Payments throug implants.

The Internet is a global network



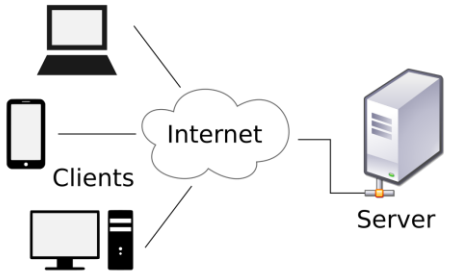


https://www.submarinecablemap.com/

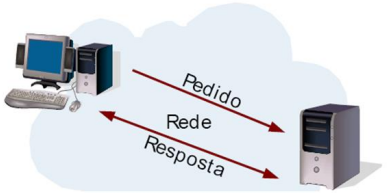


Client-Server model

- Modelo developped by Xerox PARC during the 1970's
- Describes the relations between programs in na application

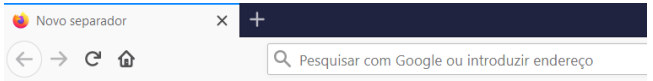


- How it works:



1. Client (e.g.: browser) makes the ***request*** of a resource (ex: web page) to a server
2. Server (e.g.: Apache) receives the request and sends [***response***] the resource

- Client and server communicate through messages
  - **Messages** act over the resources on the server
    - **Resources** are identified by URLs
      - **URL** is the link we write on the browser



URLs

Structure

- **Protocol**: defines the communication rules (e.g.: HTTP, HTTPS*, FTP)
- **Domain**: address of the server that provides the resource
- **Path**: specifies the location of the resource on the server's file system
- ***Query string*** (optional): one or more pairs in the form *name=value* that are sent to the server to filter the resource

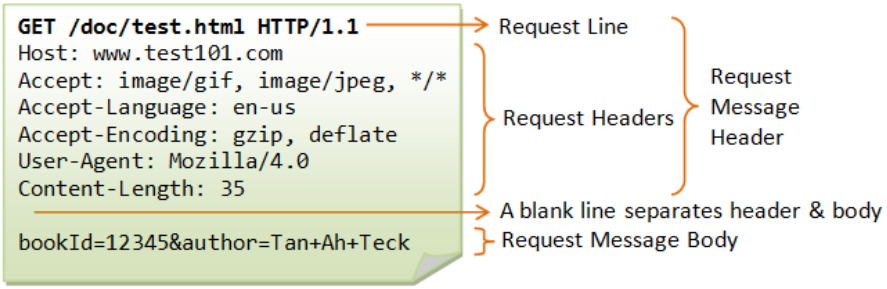
* **HTTPS** (*HTTP Secure*) is a HTTP protocol implementation with an additional security layer that uses the SSL/TLS (*Secure Sockets Layer* / *Transport Layer Security*) protocol. It allows the data to be sent through a cryptographed connection and the identities of the server and the client to be verified through certificates

HTTP protocol

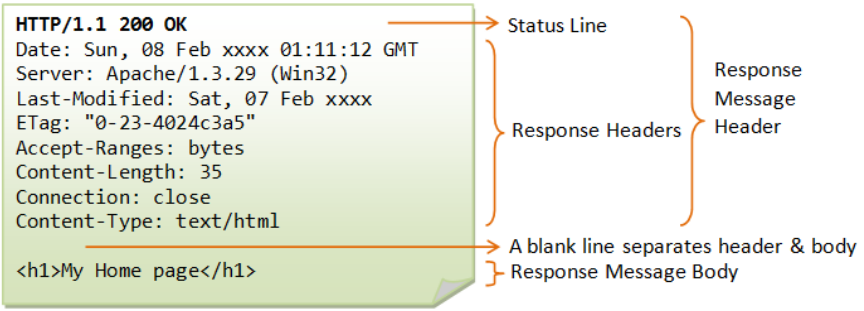
- Hypertext Transfer Protocol
- Based on client-server model
- Allows message exchange by:
  - **Browsers**: softwares that request different format contents
  - **Web servers**: softwares that work automatically and provide static (saved files) and dynamic (generated) content
  - ***Proxies***: act as intermediate to the request and response, execute routing of requests and responses, implement access policies and avoid redundancy (*cache*)
- Differentiating characteristics:
  - ***Connectionless*:** the client/server connection is made at each request. Does not allow persistent connections.
  - ***Stateless*:** there is not a client/server connection status. Each request is independent. Cookies simulate the interaction between requests.
  - **M*edia independent*:** any resource can be sent through the protocol. The agents deal with the resource types by reading metainformation included on the protocol to characterise the resources

HTTP messages

Request



Response



Request line – Methods:

- **get**: requests resource. Should return only data
- **post**: data submission to create resource
- **put**: Replace the resource
- **delete**: remove the resource

Header:

- **Host**: server host
- **Accept**: Accepted type of content
- **Accept-Language**: Accepted Language
- **Accept-Encoding**: Accepted encoding
- **User-Agent**: Client description
- **Content-Length**: Request size (bytes)

Request body

Status line – Codes:

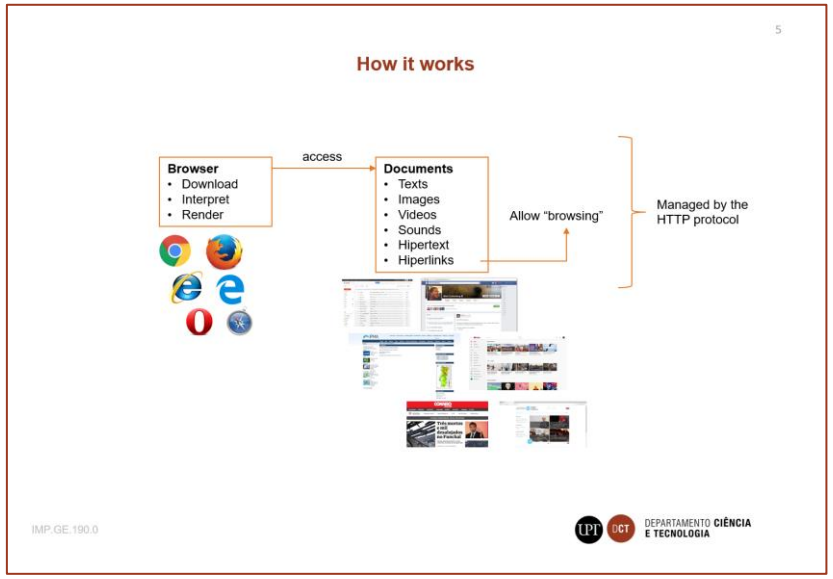
- 1**: informative responses
- 2**: success
- 3**: redirect
- 4**: client errors
- 5**: server errors

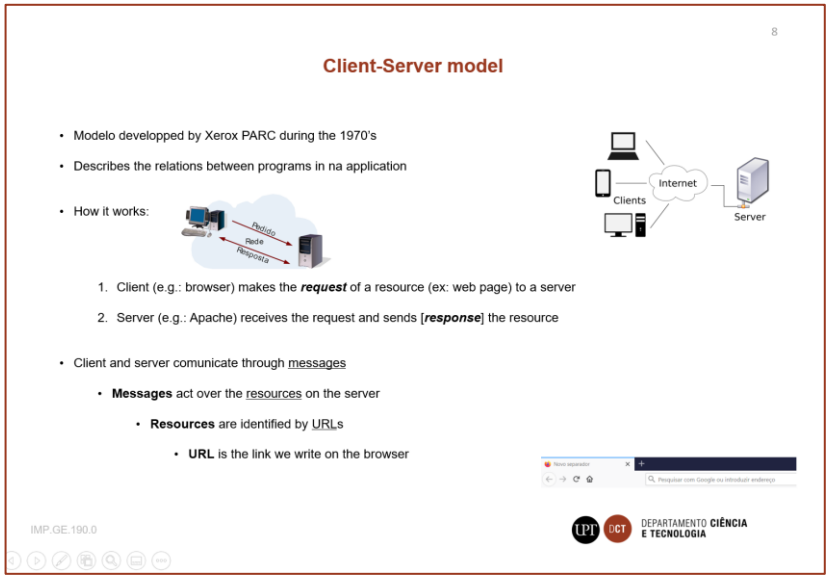
Header:

- **Date**: of the beginning of the transfer
- **Server**: software used on the server
- **Last-Modified**: last update to the resource
- **ETag**: resource version id
- **Accept-Ranges**: unit to define partial requests
- **Content-Length**: size of the response (bytes)
- **Connection**: controls what to do with the connection
- **Content-type**: response content type

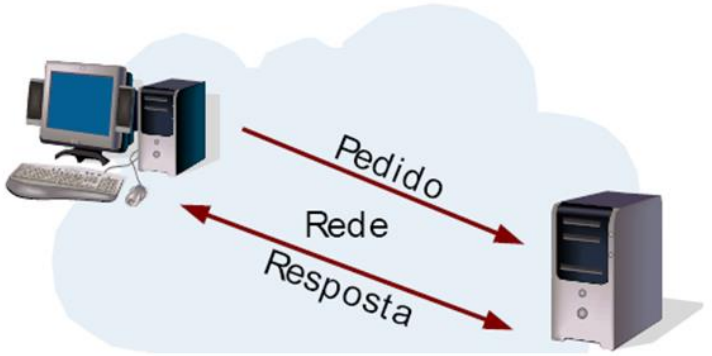
Response body

Web Languages





Frontend



Backend

DEPARTAMENTO CIÊNCIA E TECNOLOGIA

Front-end vs Back-end

Frontend

- Client side
- The part of the application that interacts directly with the user
- Graphical part of the website
- Languages:
  - **HTML**: content
  - **CSS**: appearance
  - **JavaScript**: interactivity

Full Stack

- Both sides
- "Stack of software that completely meets the needs of frontend and backend development"
  (in *Introdução ao Desenvolvimento Moderno para a Web*)
- Example: **MEAN**
  - (**M**ongoDB, **E**xpress, **A**ngular, **N**odeJS)
  - Allows using JS only both on client and server sides

Backend

- Server side
- Part of the application that interacts with services
  - **API RESTful** *
- …and databases:
  - **Relational**
  - **NoSQL**

* API RESTful
API: Application Programming Interfaces
REST: Representational State Transfer

Web Frameworks

Web Framework: standardised set of concepts and practices used to deal with web development problems. Reference for future problem solving.

- **Frontend Frameworks**
  - Focus: presentation/interaction layer (ex: responsiveness)
  - Efficient libraries
  - Combine markup, styling and scripting kanguages
  - Present in **responsive web frameworks** (Ex: **Bootstrap**, **Foundation**, **Kube**, **Semantic-UI**, **Skeleton**, **Materialize**, **Pure**)
    - **CSS style sheet**: responsive positioning of elements, style, modeling
    - **JS Plugins**: implementation of advanced interaction elements
- **Backend frameworks**
  - Focus: logical and data layers: access to the business model and data
  - Libraries: database access, mapping data to objects, session management, …
  - **Node.js**: JS interpreter to help developing high scalability applications
    - Node.js Framework – **Express**
  - **MongoDB**: non-relational database
    - High availability, scalability and flexibility
    - Allows storing **JSON** (JavaScript Object Notation) files without a fixed structure

UNIVERSIDADE
PORTUCALENSE

Do conhecimento à prática.