

YOU ARE NOW DEPRECATED



Les diplômes techniques sont presque inutiles au moment où vous obtenez votre diplôme.

La technologie progresse plus vite que jamais (La quantité d'informations techniques double tous les deux ans). Ainsi, si vous commencez un diplôme en quatre ans, la moitié de ce que vous apprenez à la fin sera obsolète.

WEB 1: Initiation à la programmation WEB

A dense, abstract word cloud centered around web design and development terms. The most prominent words include "Design", "Wordpress", "SEO", "HTML", "CSS", "AJAX", "MySQL", "DHTML", "GFX", "CGI", "PHP", "JavaScript", "Database", "Responsive", "Content", "Blog", and "E-commerce". Smaller words like "HTML5", "AJAX", "MySQL", "DHTML", "GFX", "CGI", "PHP", "JavaScript", "Database", "Responsive", "Content", "Blog", and "E-commerce" are also visible, though less frequently.

Qui suis je ?

QUI ?



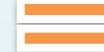
Emilien Micard

Ingénieur Développement Logiciel

Programmation
JAVA, JEE
Python, C++, ...

Traitement d'images

OU ?

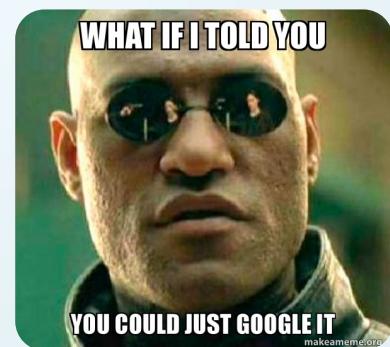
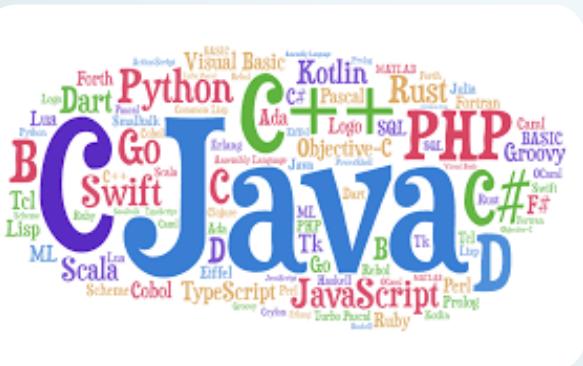


OBJECTIFS DE CE COURS

Comprendre comment fonctionne le Web

Apprendre à concevoir des sites Web dynamiques

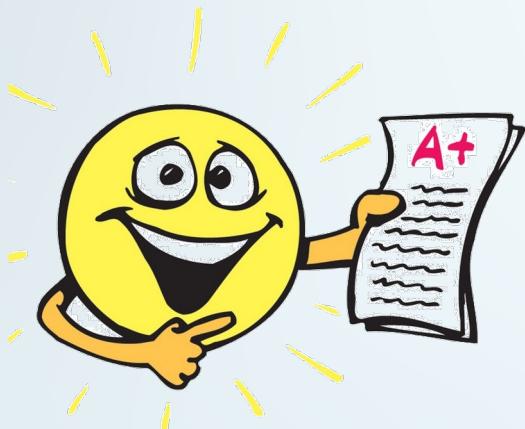
PRÉREQUIS



MODALITÉS D'ÉVALUATIONS



TP Notés



Note individuelle : Petit Quick-Quizz à chaque séance

CONTACTEZ MOI



emilien.micard@univ-lorraine.fr

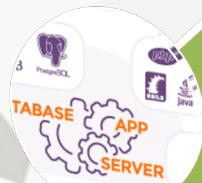
Il n'y a pas de questions idiotes... Il y a juste des questions posées un peu trop vite ...



WEB 1 : Initiation à la programmation WEB



Le Web ?



Dev Web / Backend : PHP



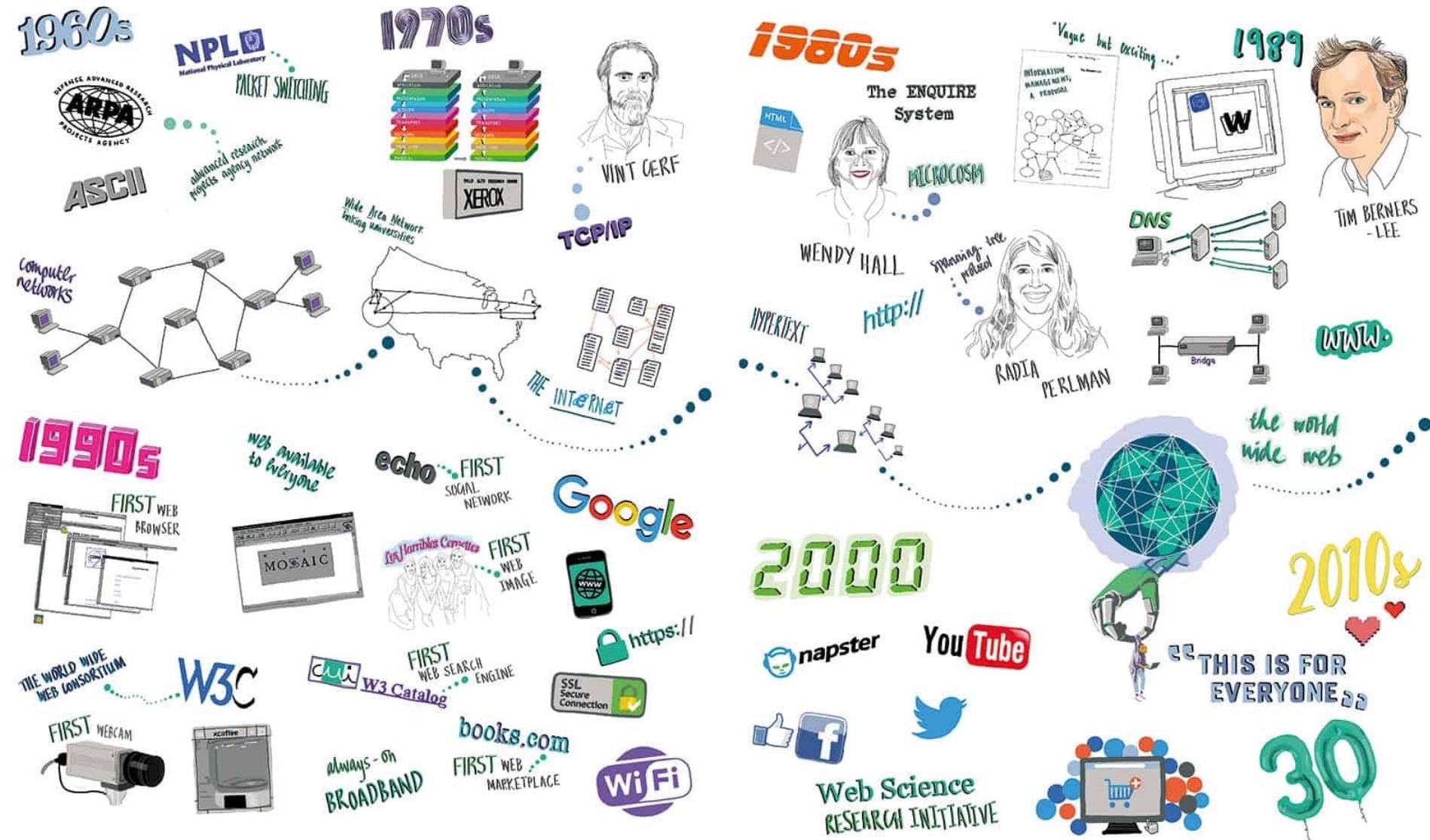
Dev Web / Frontend



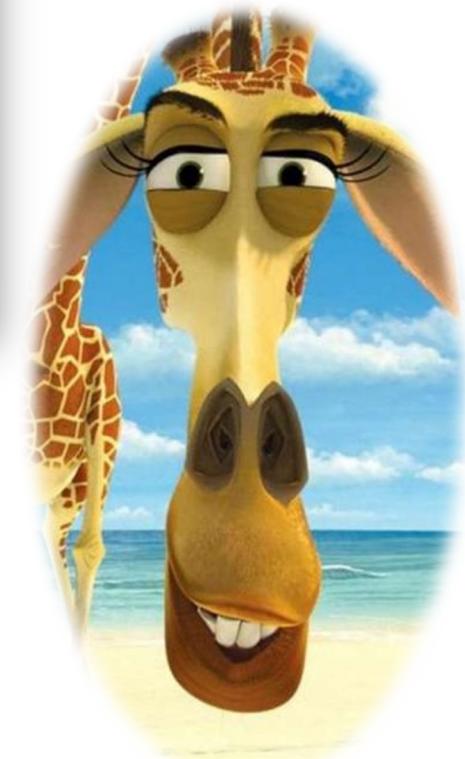
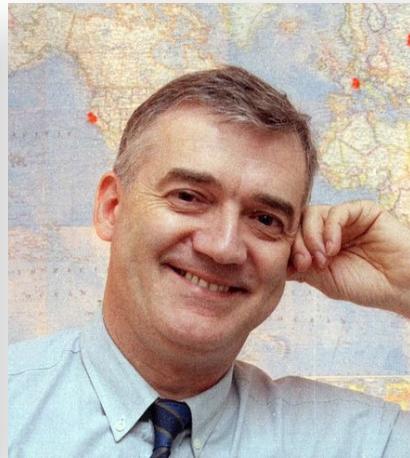
Un peu de Sécurité



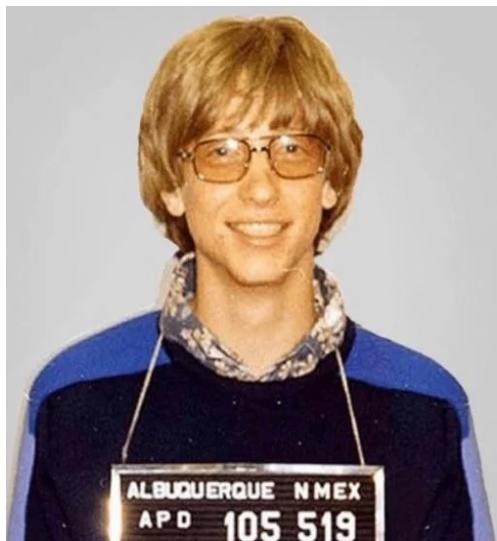
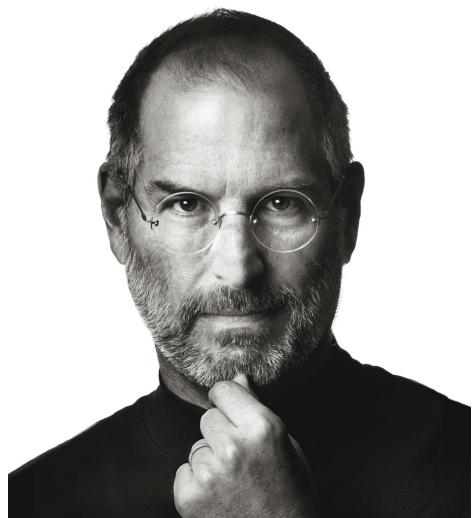
WEB 1 : Le Web ?



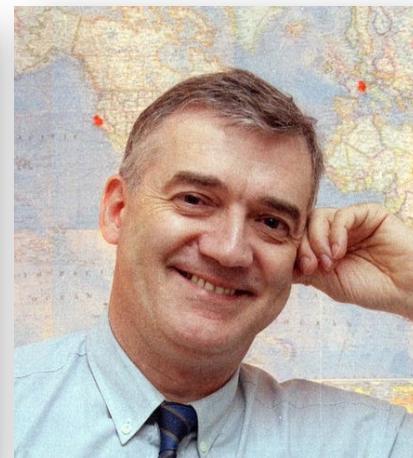
Qui est ce ?



Qui est ce ?



Qui est ce ?



Tim Berners-Lee

Robert Cailliau

Wendy Hall

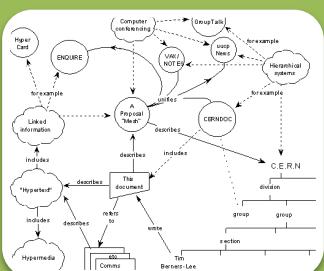
Ils ont juste inventé le WEB ☺



Le Web ? : Plan



Un peu d'histoire



Retour sur les éléments du Web

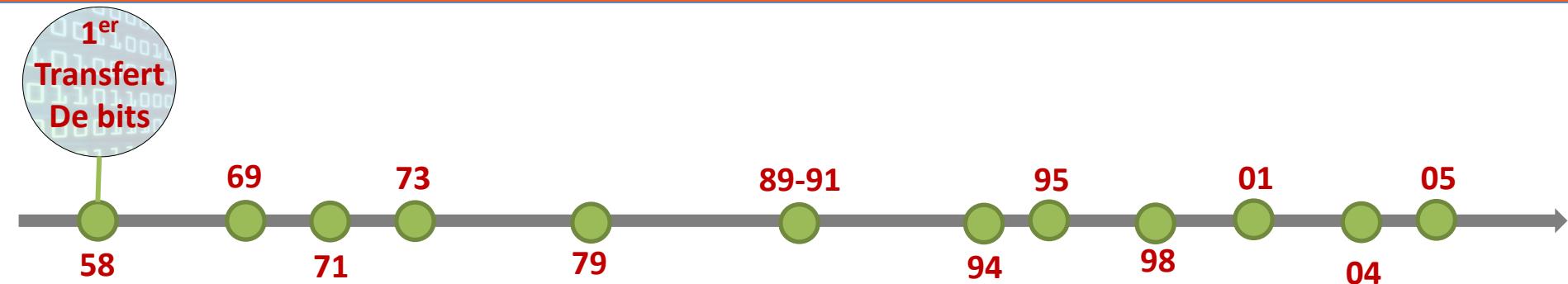
WEB 1 : Un peu d'histoire



WEB 1 : Un peu d'histoire

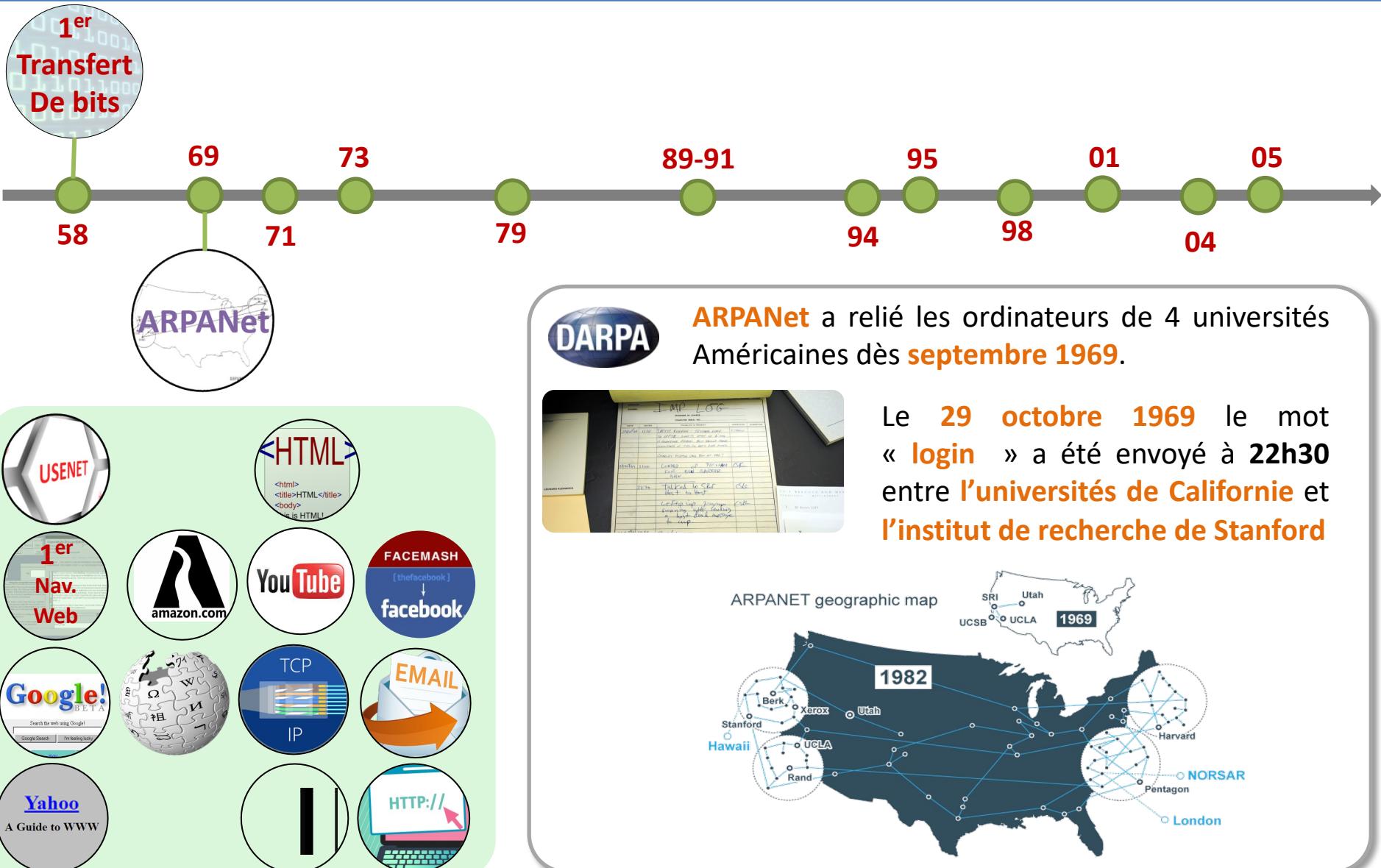


WEB 1 : Un peu d'histoire

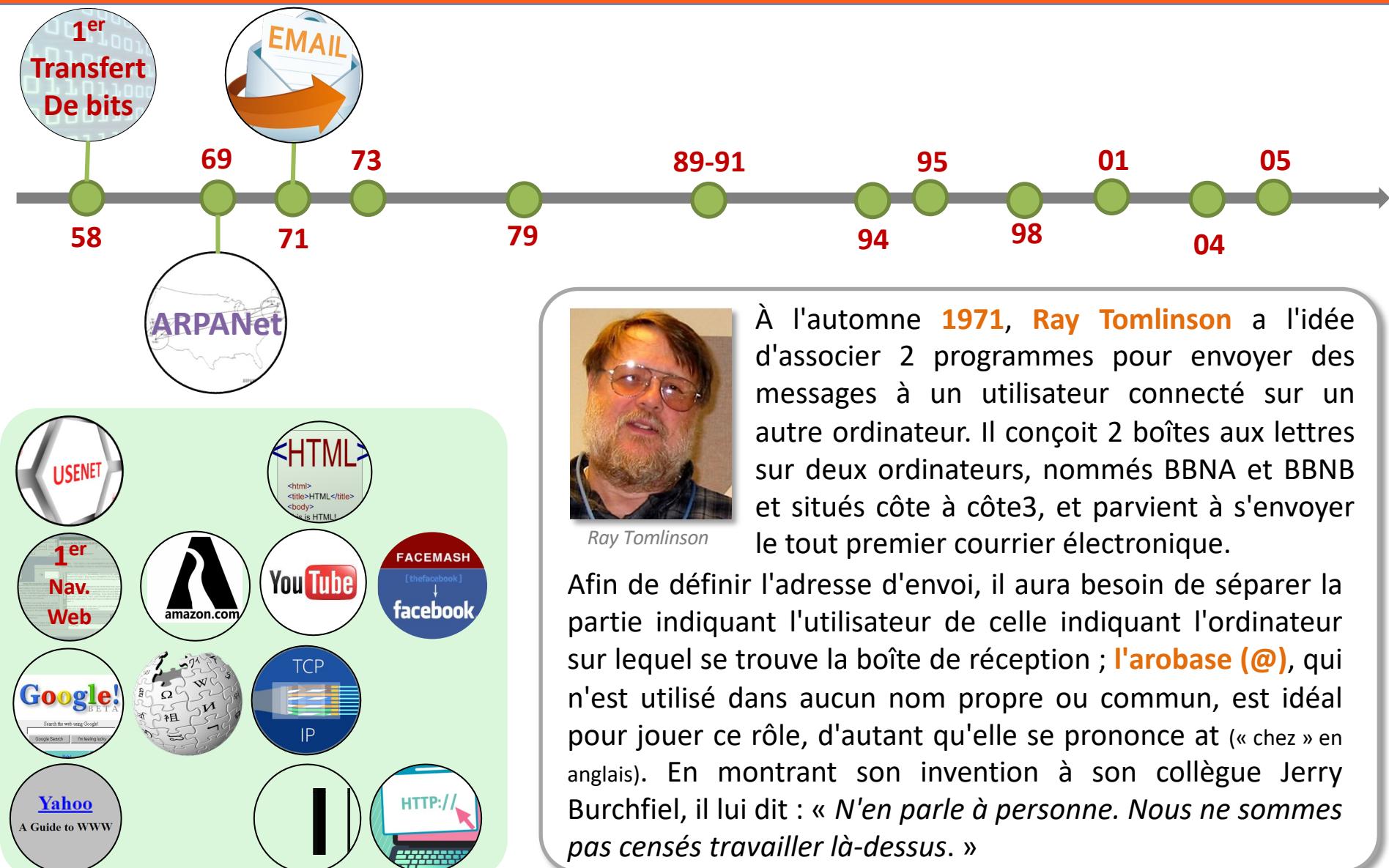


Les **Laboratoires Bell** créent le premier **modem** permettant de transmettre des **données binaires** sur une simple ligne téléphonique.

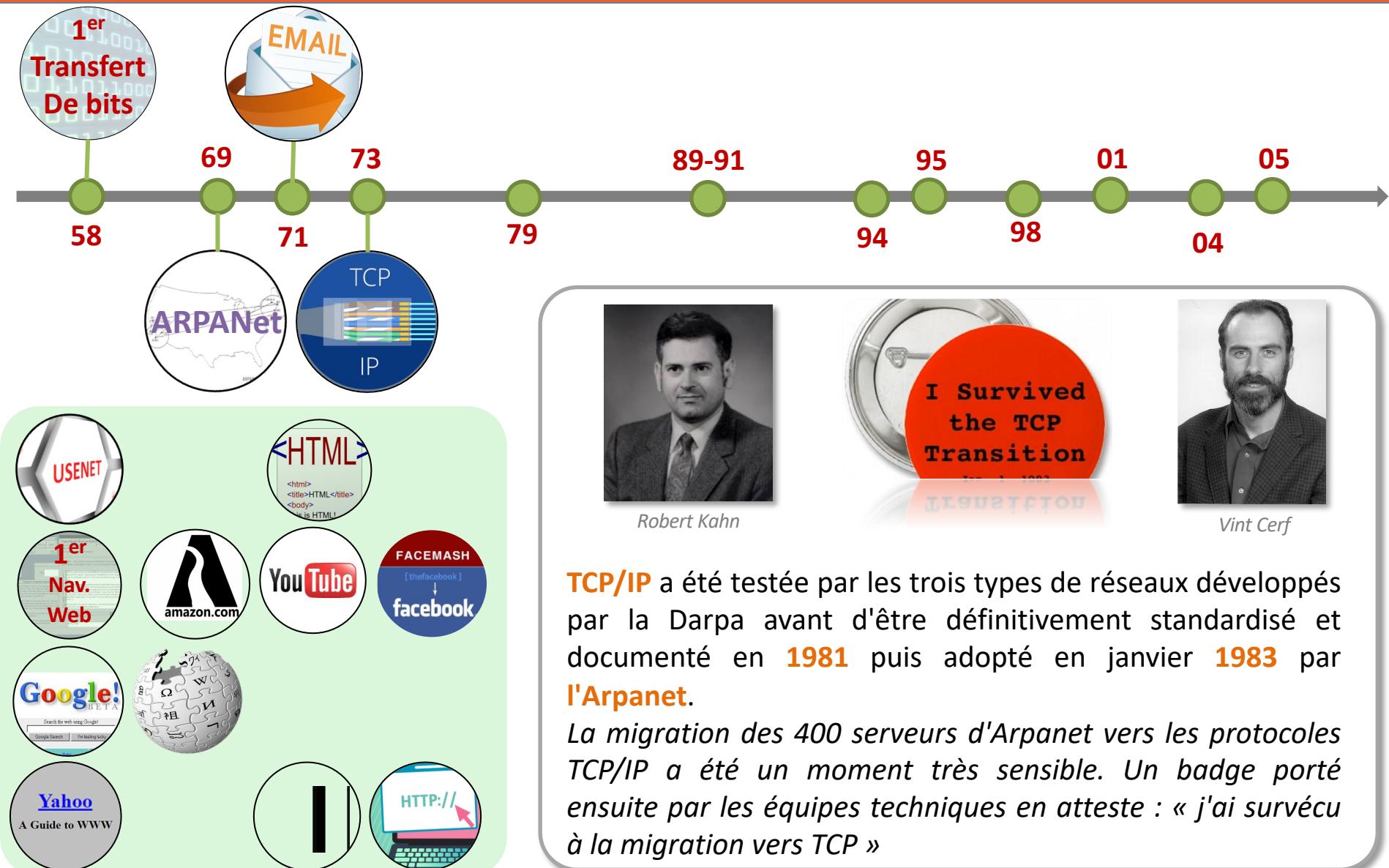
WEB 1 : Un peu d'histoire



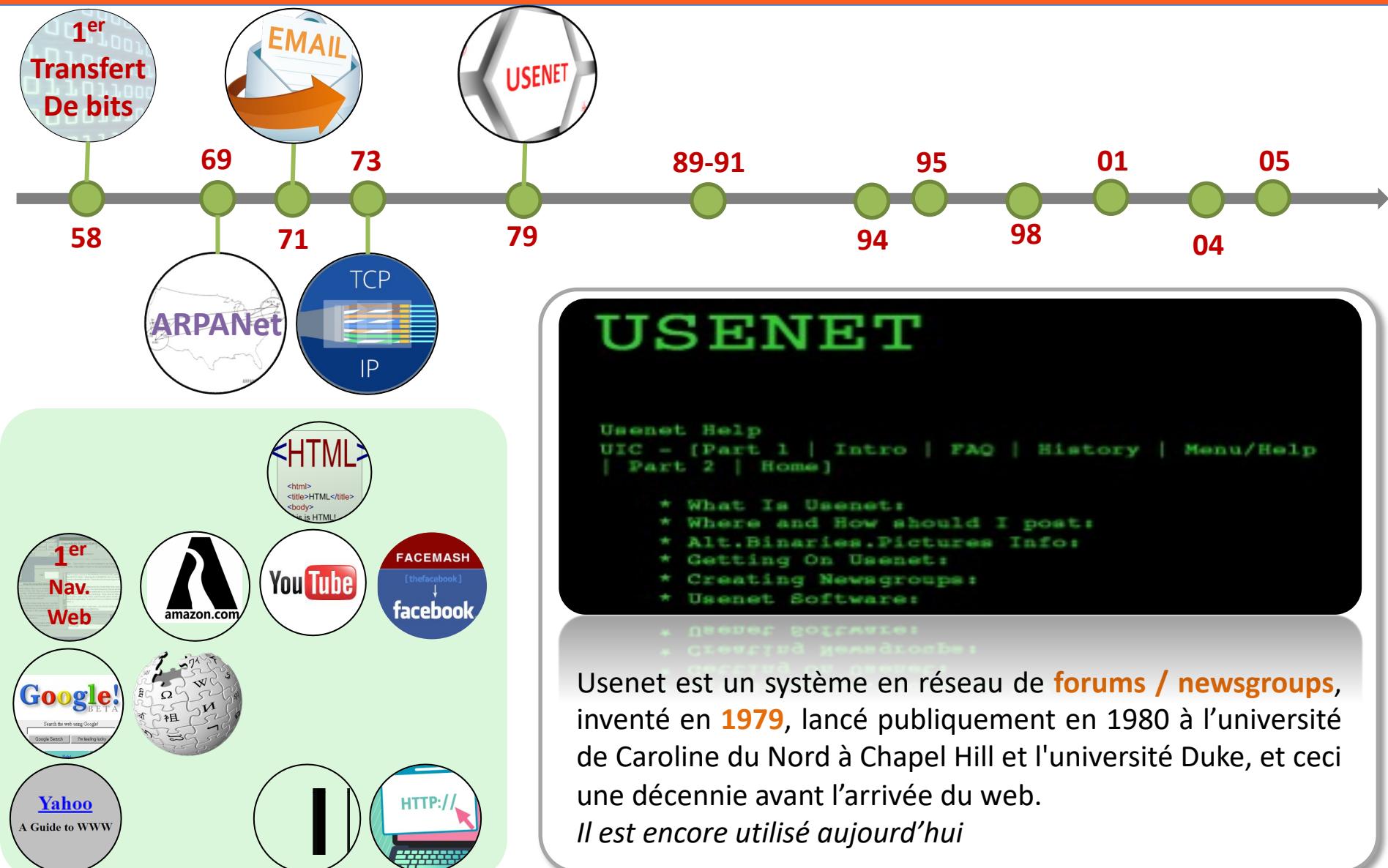
WEB 1 : Un peu d'histoire



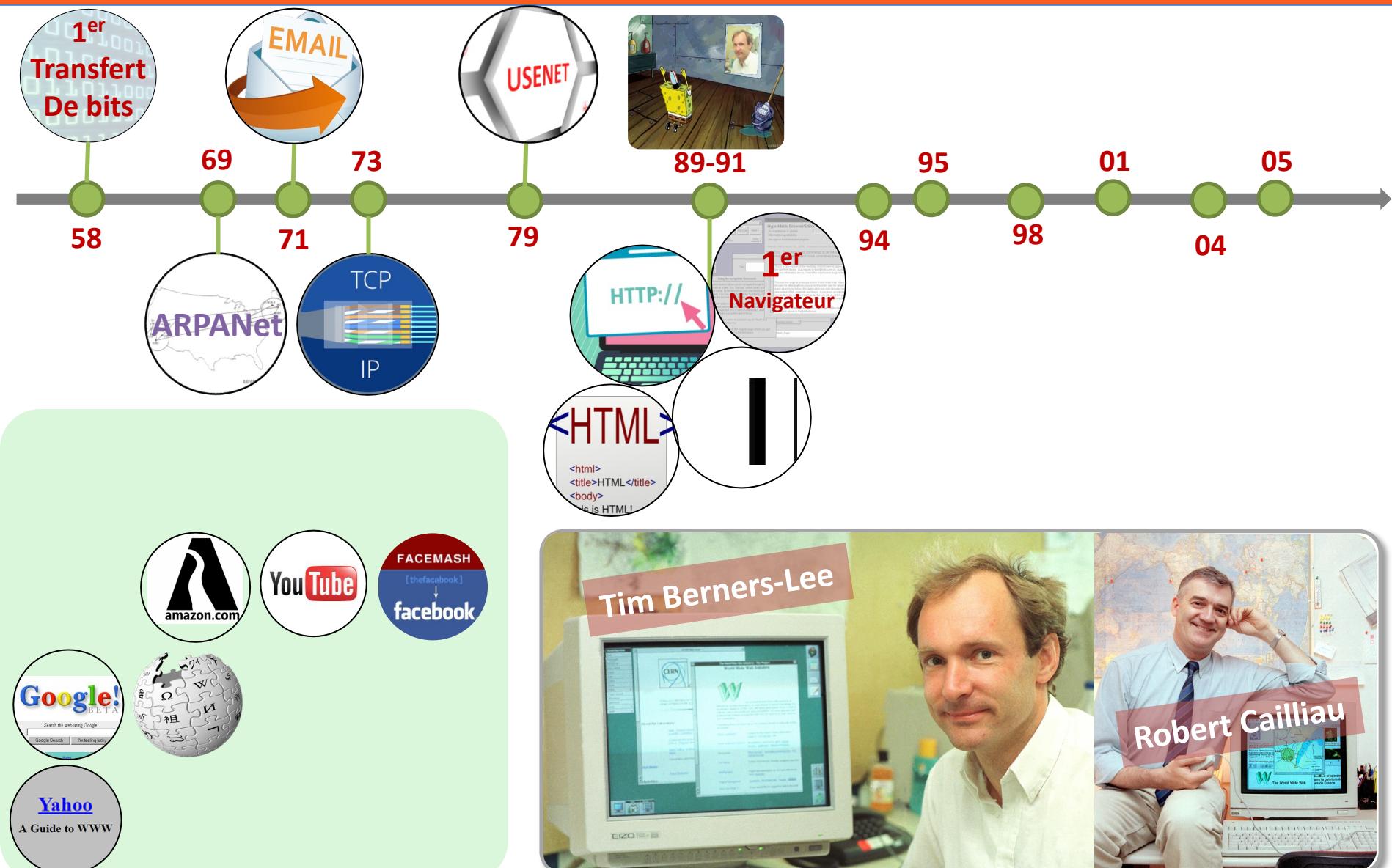
WEB 1 : Un peu d'histoire



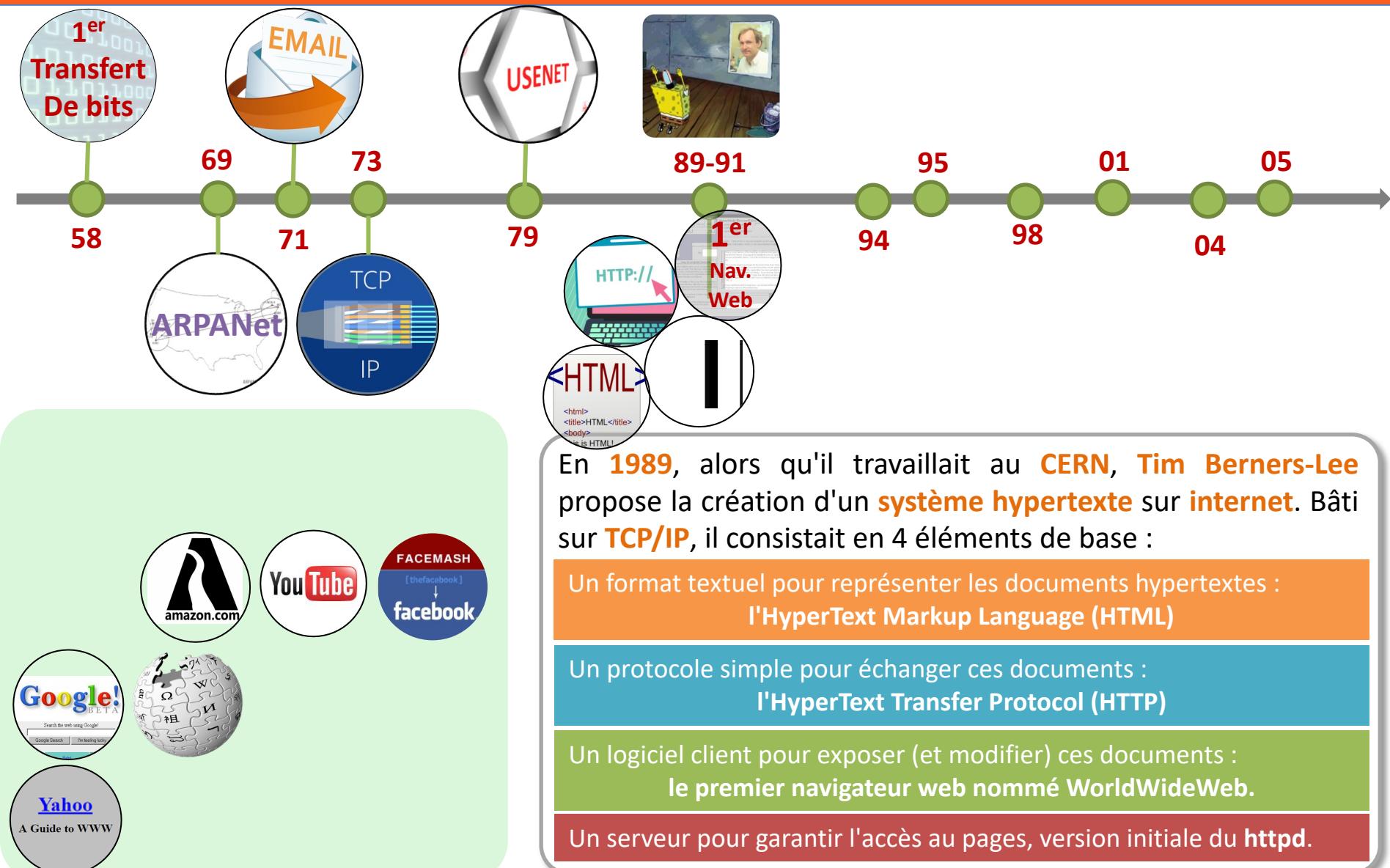
WEB 1 : Un peu d'histoire



WEB 1 : Un peu d'histoire



WEB 1 : Un peu d'histoire



WEB 1 : Un peu d'histoire

World Wide Web

The WorldWideWeb (W3) is a wide-area [hypermedia](#) information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an [executive summary](#) of the project, [Mailing lists](#) , [Policy](#) , November's [W3 news](#) , [Frequently](#)

What's out there?

Pointers to the world's online information, [subjects](#) , [W3 servers](#), etc.

Help

on the browser you are using

Software Products

A list of W3 project components and their current state. (e.g. [Line Mode](#) ,[X11 Viola](#) , [NeXTStep](#) , [Servers](#) , [Tools](#) , [Mail robot](#) , [Library](#))

Technical

Details of protocols, formats, program internals etc

Bibliography

Paper documentation on W3 and references.

People

A list of some people involved in the project.

History

A summary of the history of the project.

How can I help ?

If you would like to support the web..

Getting code

Getting the code by [anonymous FTP](#) , etc.



Le tout premier site et le tout premier serveur Web voient le jour sur un ordinateur NeXT **le 20 décembre 1990**. Le site, qui n'est consultable que sur le **réseau interne du CERN**, est affiché sur un outil baptisé « **navigateur** » que TBL nomme **WorldWideWeb**. Son adresse : info.cern.ch.

Tim Berners-Lee, qui travaille désormais avec le belge **Robert Caillau**, a durant l'année défini les bases de la technologie qui sera **rendu publique** en **1991**. Ils utilisent pour cela l'ancêtre du forum : Usenet.

A partir de 93/94, le Web s'emballe ➔ ~10.000 sites fin 94.

WEB 1 : Un peu d'histoire

World Wide Web

The WorldWideWeb (W3) is a wide-area [hypermedia](#) information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an [executive summary](#) of the project, [Mailing lists](#) , [Policy](#) , November's [W3 news](#) , [Frequently](#)

What's out there?

Pointers to the world's online information, [subjects](#) , [W3 servers](#), etc.

Help

on the browser you are using

Software Products

A list of W3 project components and their current state. (e.g. [Line Mode](#) ,[X11 Viola](#) , [NeXTStep](#) , [Servers](#) , [Tools](#) , [Mail robot](#) , [Library](#))

Technical

Details of protocols, formats, program internals etc

Bibliography

Paper documentation on W3 and references.

People

A list of some people involved in the project.

History

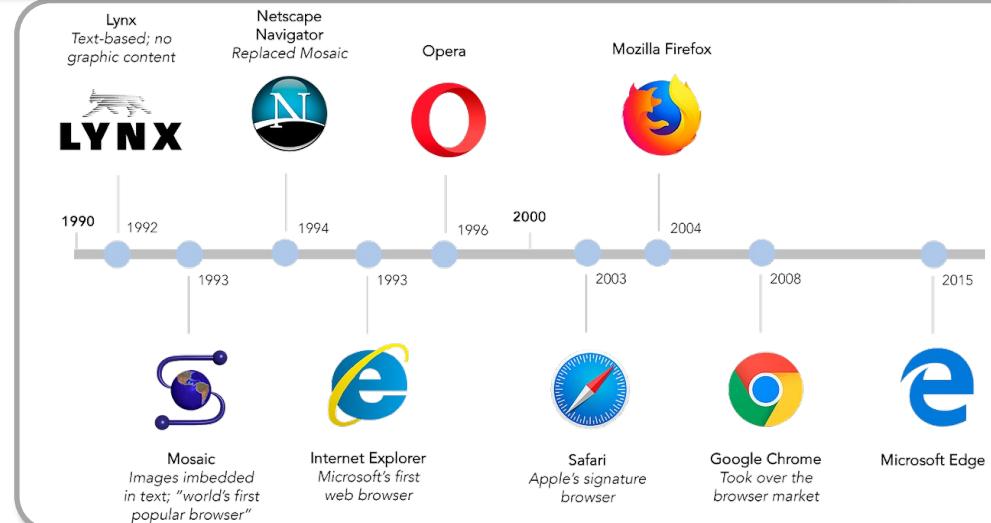
A summary of the history of the project.

How can I help ?

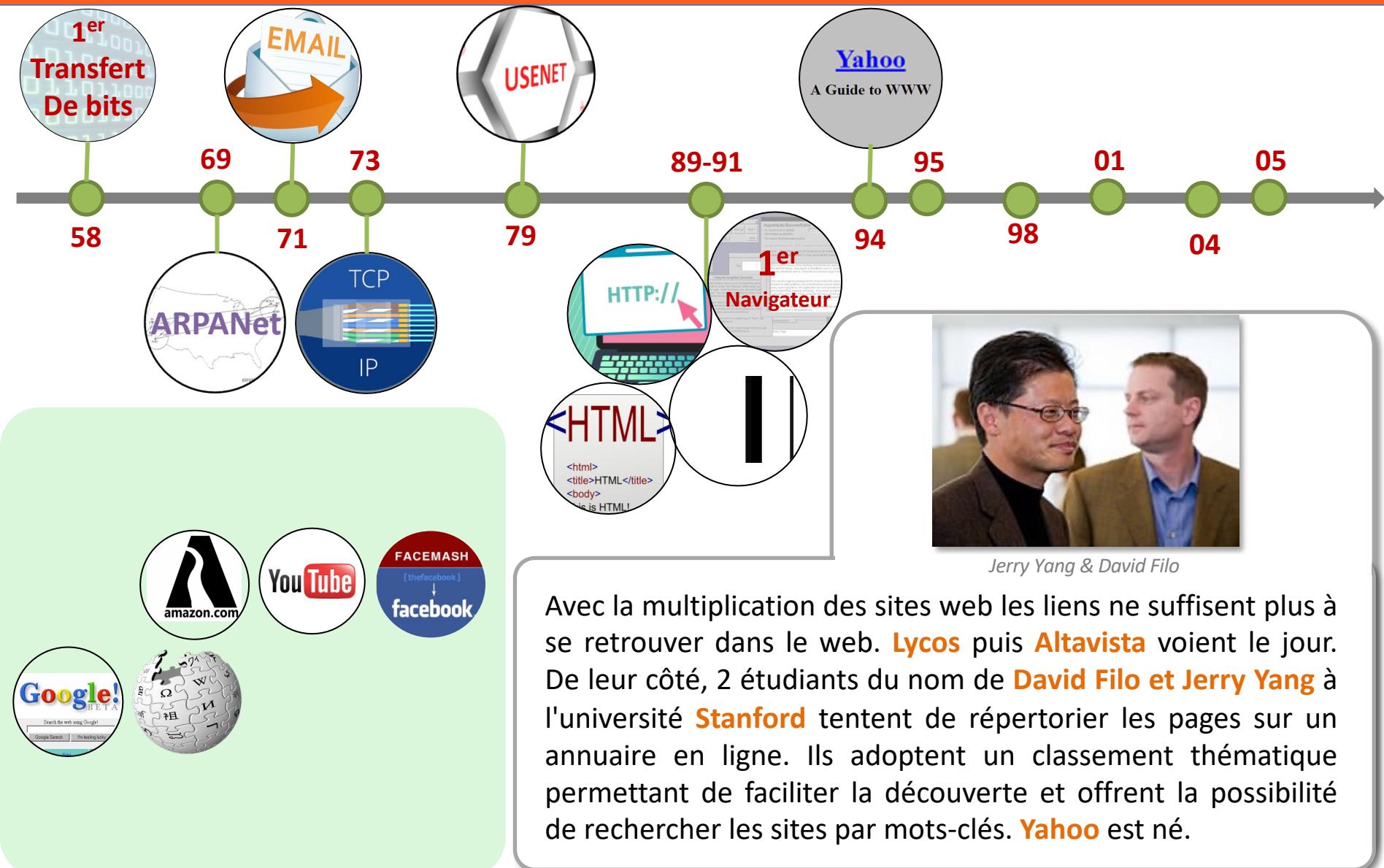
If you would like to support the web..

Getting code

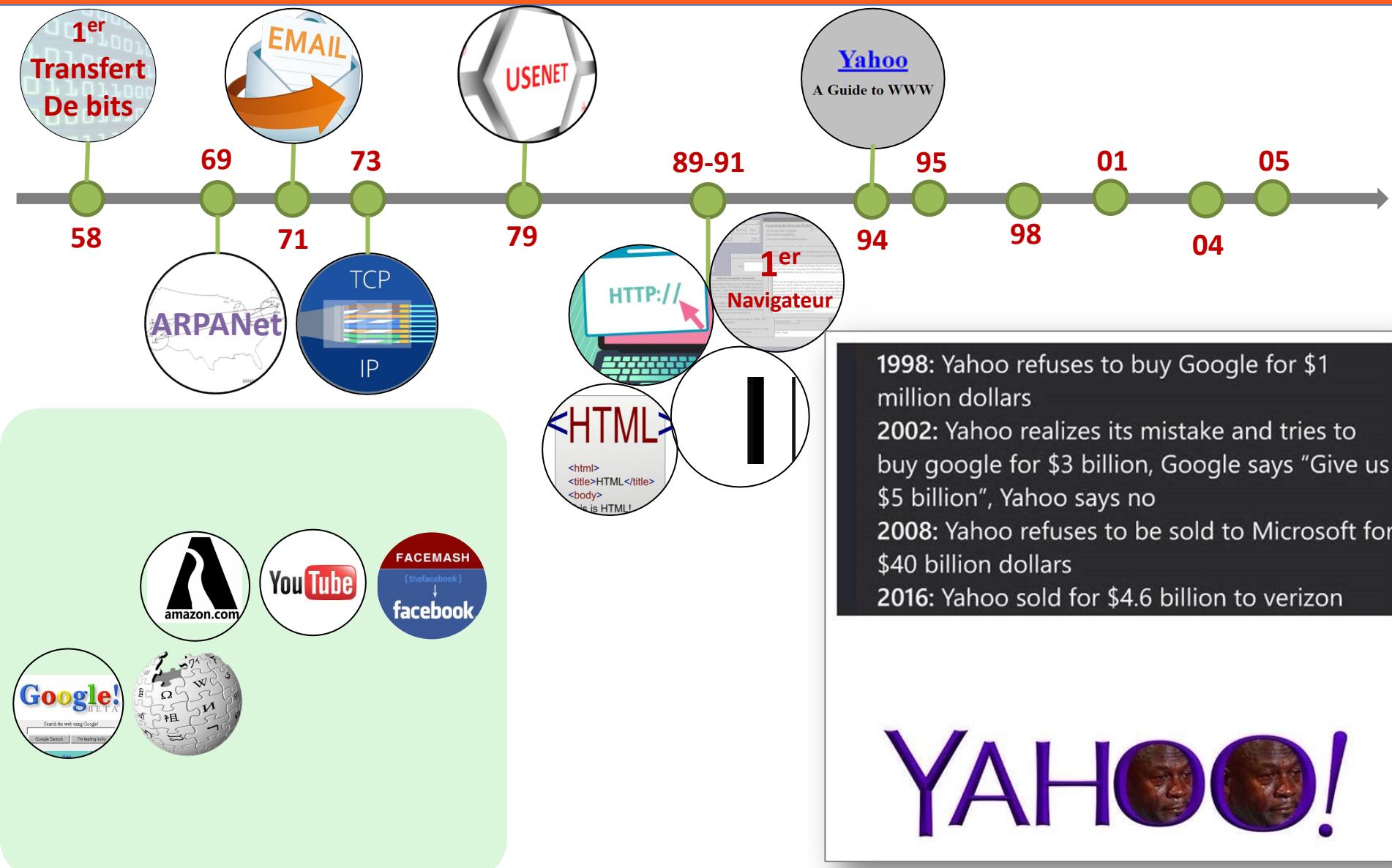
Getting the code by [anonymous FTP](#) , etc.



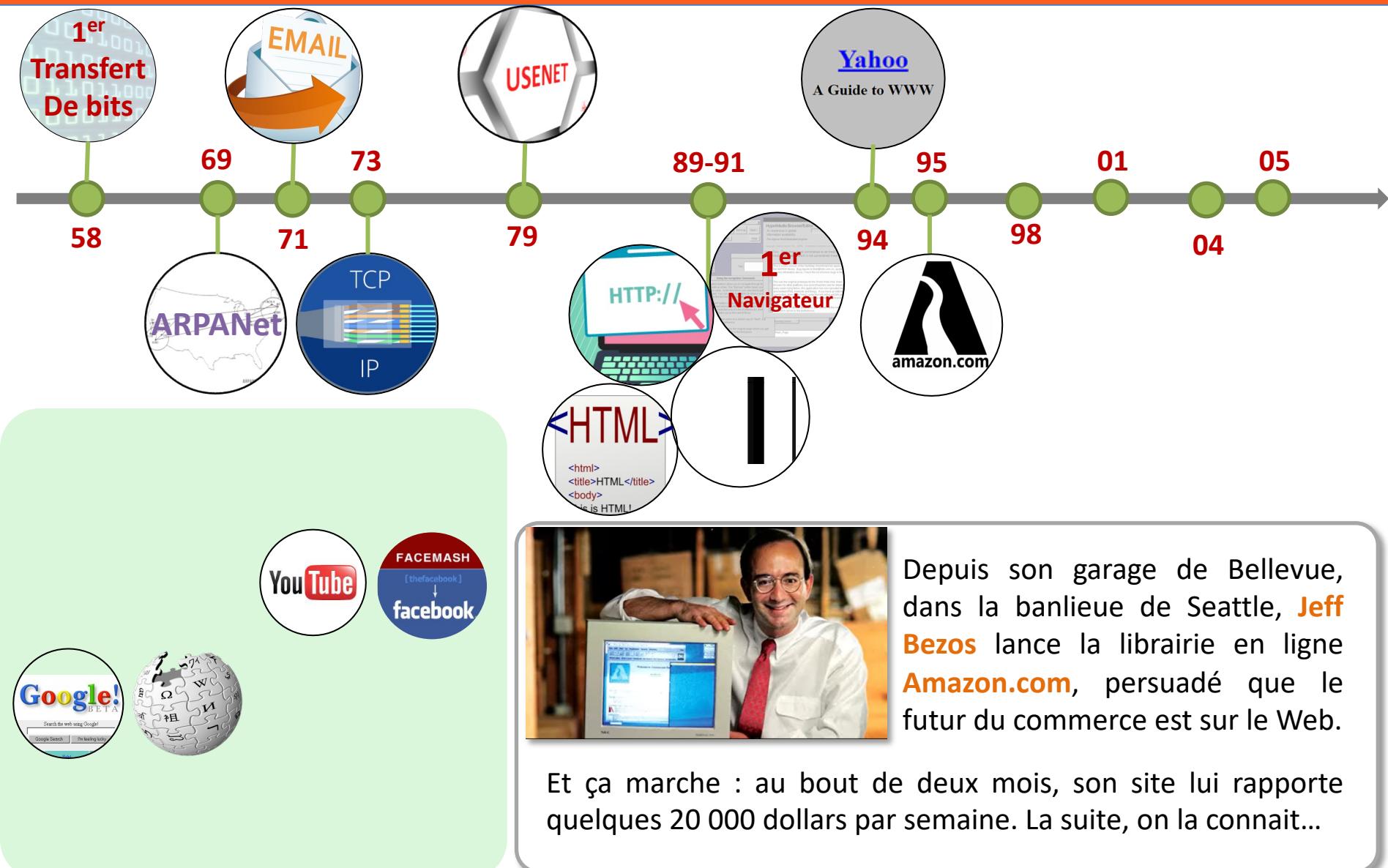
WEB 1 : Un peu d'histoire



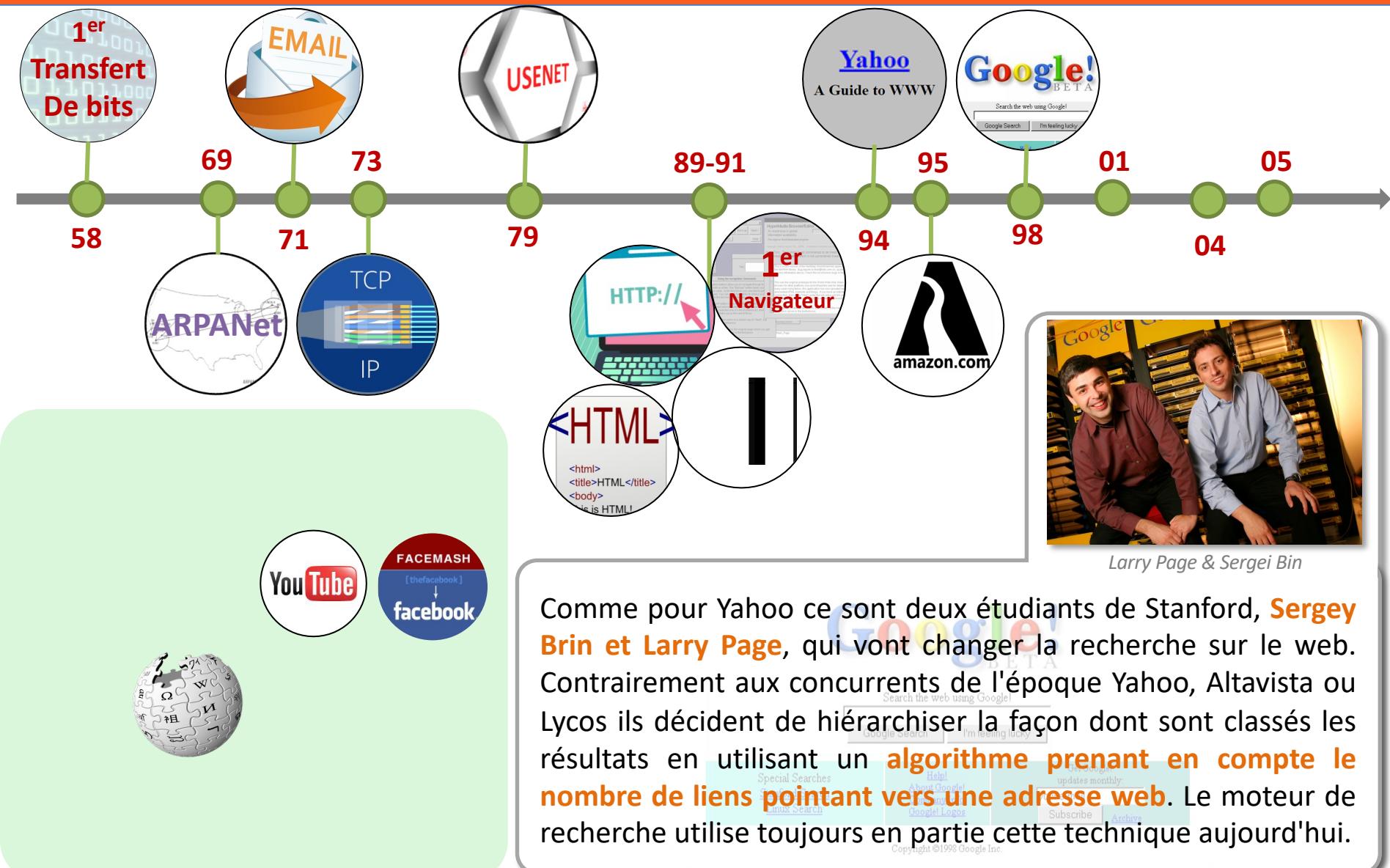
WEB 1 : Un peu d'histoire



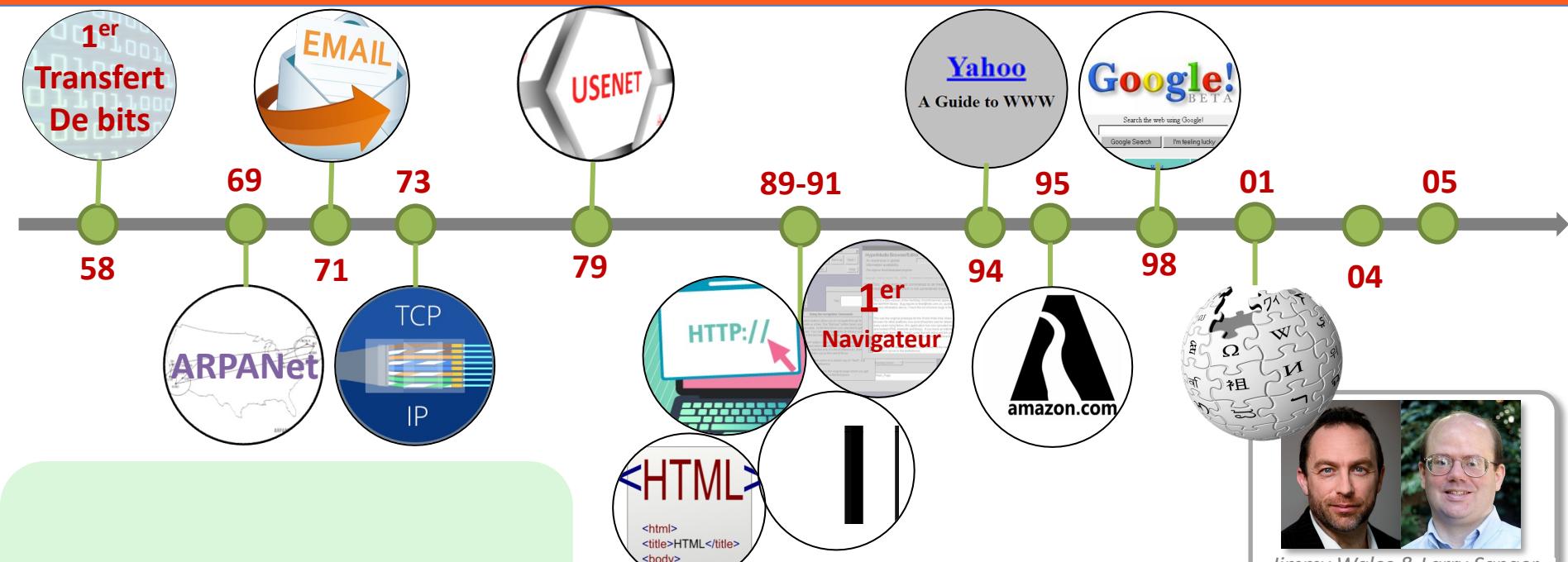
WEB 1 : Un peu d'histoire



WEB 1 : Un peu d'histoire

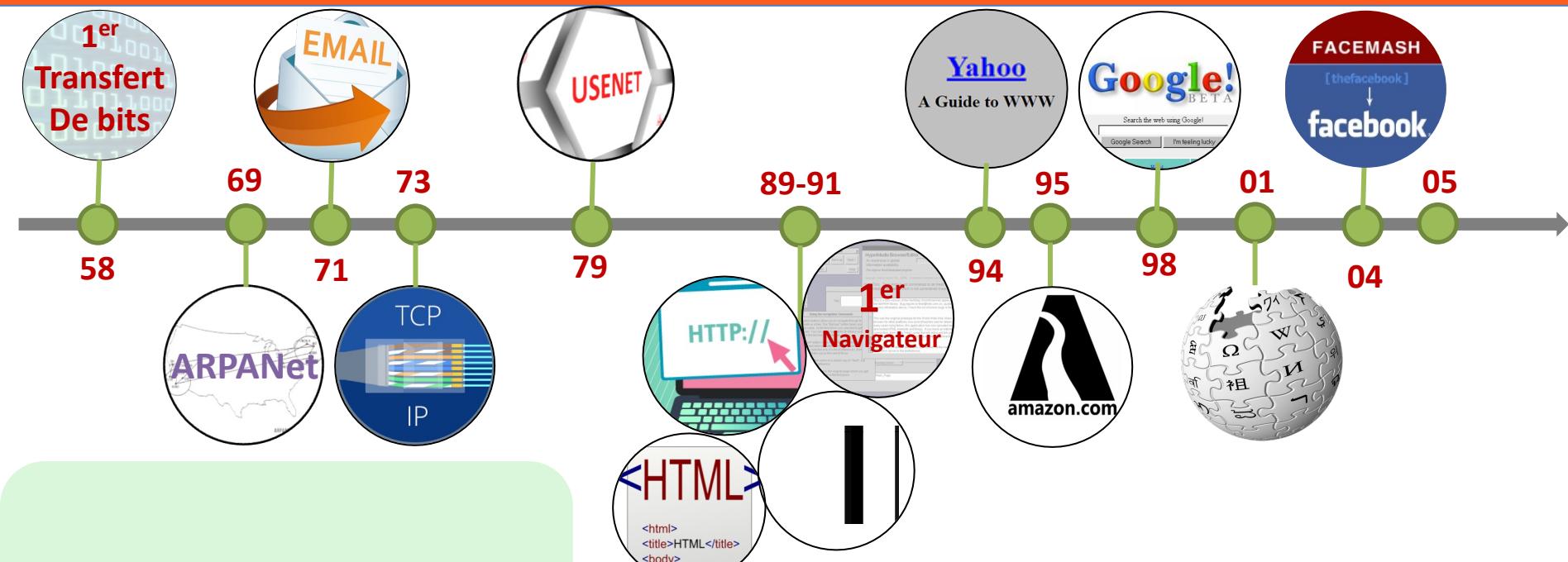


WEB 1 : Un peu d'histoire



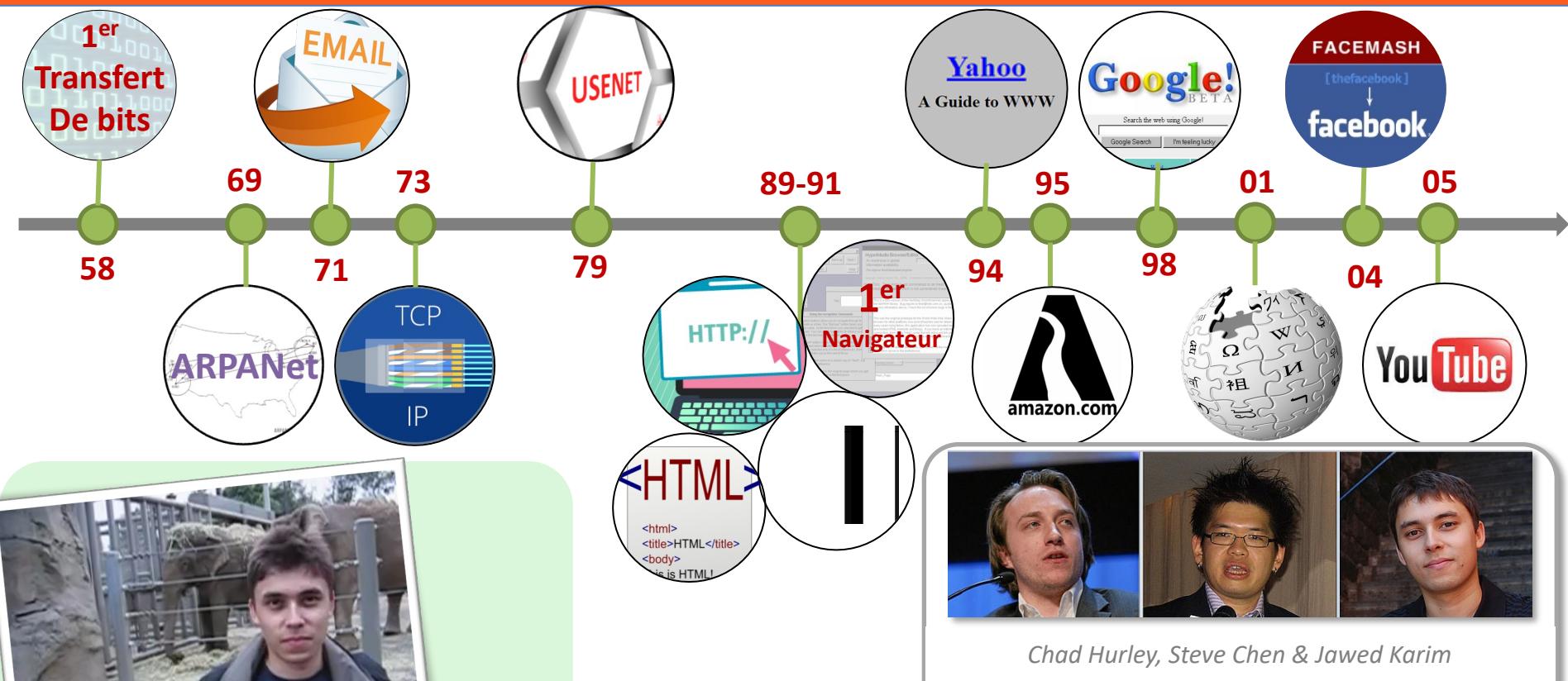
L'encyclopédie libre est née de l'échec de **Nupedia**, un projet d'encyclopédie gratuite mais dont les articles, écrits par des experts, devaient subir un contrôle scientifique strict avant d'être publiés. Devant le faible nombre d'articles, **Jimmy Wales**, le patron de Nupedia et **Larry Sanger** son rédacteur en chef, imaginent un tout autre système de validation, basé sur une technologie encore peu connue à l'époque, **le Wiki**, permettant aux visiteurs d'un site d'en modifier eux-mêmes le contenu. *Wikipedia compte désormais 500 millions de visiteurs uniques par mois !*

WEB 1 : Un peu d'histoire



Le **réseau social** créé par **Mark Zuckerberg** est devenu un monstre rassemblant des milliards d'utilisateurs, bien loin du trombinoscope pour étudiants qu'il était au début. A tel point qu'il est devenu la cible de toutes les interrogations concernant la préservation des données personnelles de chacun. D'ailleurs, Tim Berners Lee n'apprécie pas le service, qu'il estime être un "Web dans le Web" qui pourrait finir par fragmenter son invention.

WEB 1 : Un peu d'histoire



La première vidéo de YouTube est d'ailleurs « **Me at the zoo** », dans laquelle **Jawed Karim** commente sa visite au zoo de San Diego

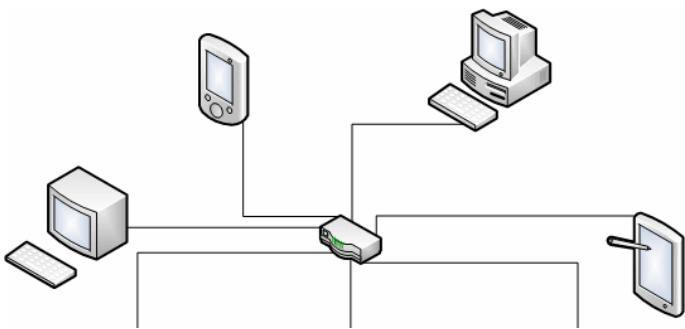
La société **YouTube**, symbole du **Web 2.0**, a été créée en **février 2005** par trois ex-employés de l'entreprise PayPal : **Chad Hurley**, **Steve Chen** et **Jawed Karim**. La plateforme devait être à l'origine un site de rencontres reposant sur l'utilisation de vidéos. Google rachète la prometteuse plate-forme un an et demi après son lancement, contre 1,65 milliard de dollars...

WEB 1 : Modèle Client-Serveur – La Base



imgflip.com

WEB 1 : Le modèle « Client / Serveur » : Terminologie



Architecture informatique distribuée dans laquelle les tâches et les charges de travail sont partagées entre :

- Les fournisseurs d'une ressource ou d'un service : **SERVEURS**
- Les demandeurs de services : **CLIENTS**

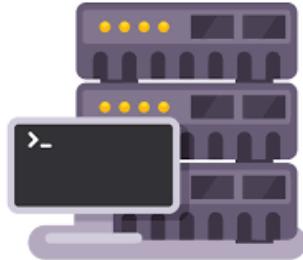
Les tâches sont exécutées par des programmes qui sont répartis entre clients et serveurs.

Souvent, les clients et les serveurs tournent sur des machines séparées et communiquent via un réseau informatique.

Mais le client et le serveur peuvent résider sur le même système.

Une machine hôte (anglais : host) peut exécuter un ou plusieurs programmes serveurs qui partagent leurs ressources avec des clients.

WEB 1 : Le modèle « Client / Serveur » : Terminologie



SERVEUR :

Programme qui peut se connecter à un programme client pour lui donner accès à un service.

Remarque : "serveur" est un rôle, pas un ordinateur en soi.

Un serveur et un client peuvent être hébergés par la même machine hôte.



CLIENT :

Programme qui peut demander un service à un serveur qui fournit ce service.



SERVICE :

Fourniture d'une fonctionnalité spécifique offert par un serveur et qu'un client peut utiliser.



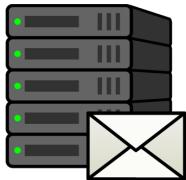
REQUÊTE :

Demande de la part d'un client, d'un service dont il a besoin à un serveur.

RÉPONSE :

Réponse d'un serveur à une demande d'un client.

WEB 1 : Le modèle « Client / Serveur » : Terminologie



Serveur de messagerie :

Envoie des mails à des clients de messagerie.



Serveur de fichier :

Permet de stocker et consulter des fichiers sur le réseau. (*exemple : FTP*)



Serveur de Données :

exécute un système de gestion de base de données (SGBD).



De grandes quantités de données, stockées en central, sont mises à disposition.



Plusieurs clients peuvent travailler simultanément avec ces données.

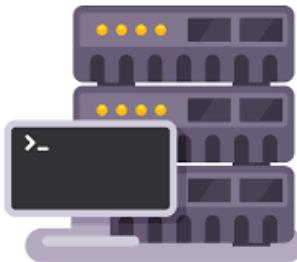


Serveur WEB:

Renvoie des pages web demandées par des navigateurs web.

WEB 1 : Le modèle « Client / Serveur » : Principe

Processus serveur



- Attende une connexion entrante sur un ou plusieurs ports réseaux ;
- À la connexion d'un client sur le port en écoute, il ouvre un socket ;
- Suite à la connexion, il communique avec le client suivant le protocole prévu ;
- Un serveur est généralement capable de servir plusieurs clients simultanément ;

Processus client



- Établit la connexion au serveur à destination d'un ou plusieurs ports réseaux ;
- Suite à la connexion (acceptée par le serveur), il communique comme le prévoit la couche applicative du modèle OSI.

Le client et le serveur doivent utiliser le même protocole de communication au niveau de la couche transport du modèle OSI.

WEB 1 : Le modèle « Client / Serveur » : Principe



WEB 1 : Le modèle « Client / Serveur » : Bilan

Centralisation des données :

- Simplifie les contrôles de sécurité, l'administration, la mise à jour des données et des logiciels.
- Architecture particulièrement adaptée et rapide pour retrouver et comparer de vastes quantités d'information (*moteur de recherche sur le Web*).



Puissance de calcul :

- Traitements lourds à la charge du serveur.
- Client + léger dont les traitements peuvent être simplifiés au maximum

Centralisation :

- Les clients ne peuvent communiquer entre eux.



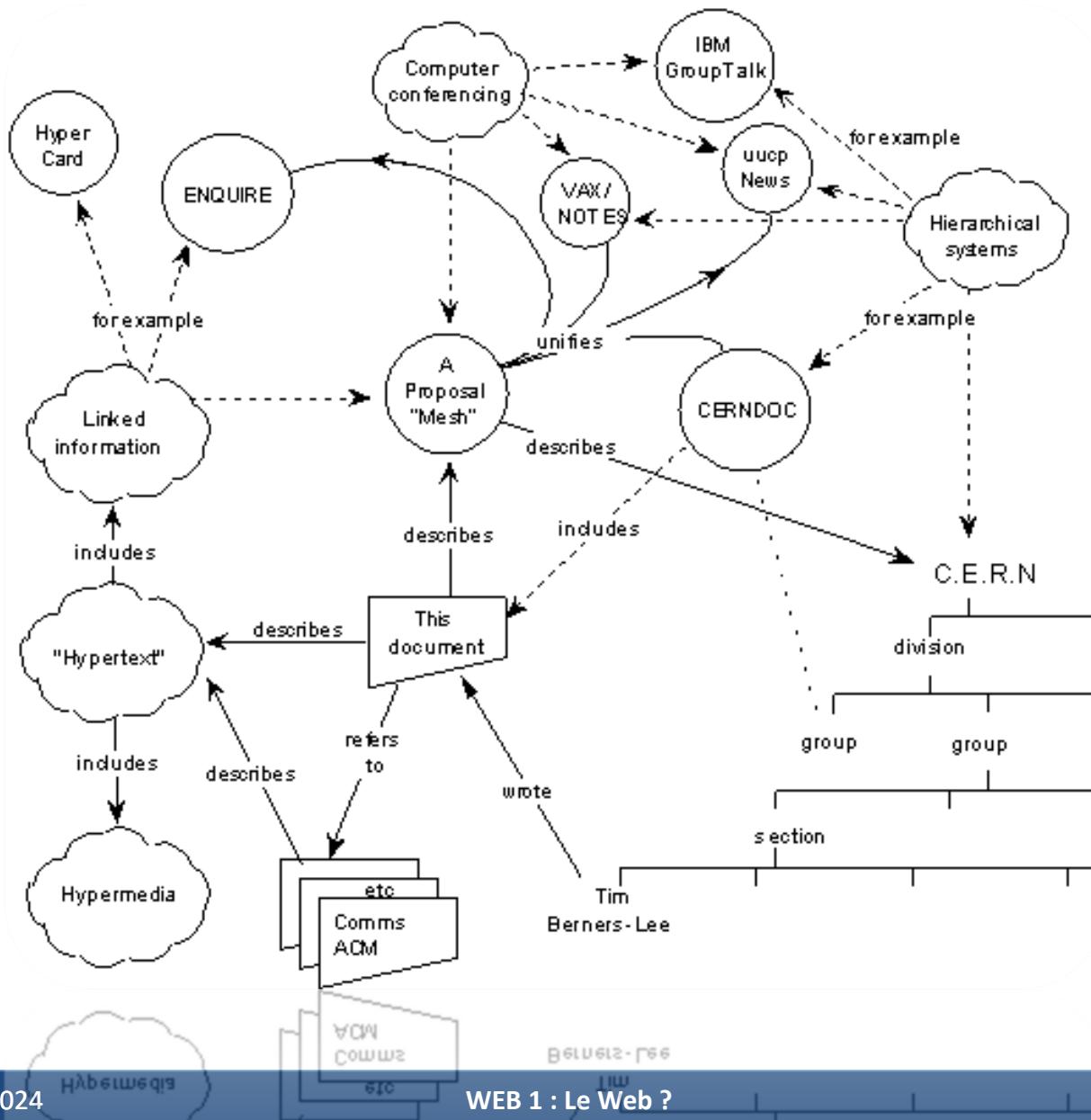
Charge des serveurs :

- Si trop de clients veulent communiquer avec le serveur au même moment, ce dernier risque de ne pas supporter la charge.

Maintenance :

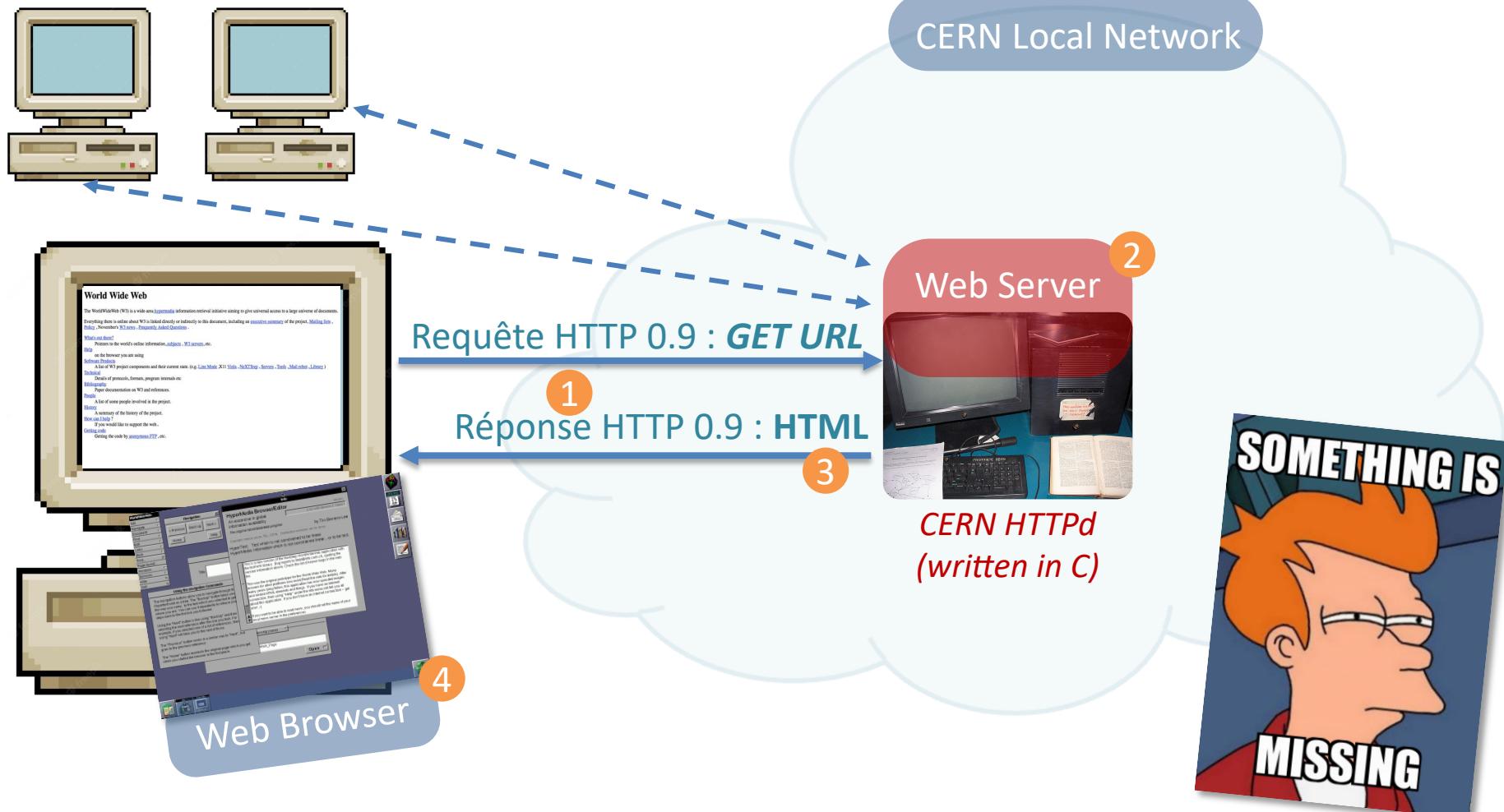
- Si le serveur n'est plus disponible, plus aucun des clients ne fonctionne.
- Les coûts de mise en place et de maintenance peuvent être élevés.

WEB 1 : Evolution du Web



WEB 1 : Evolution du Web

Modèle TBL (89) :



WEB 1 : Evolution du Web



Pages statiques en local

Pas de Base de données

Pas de personnalisation

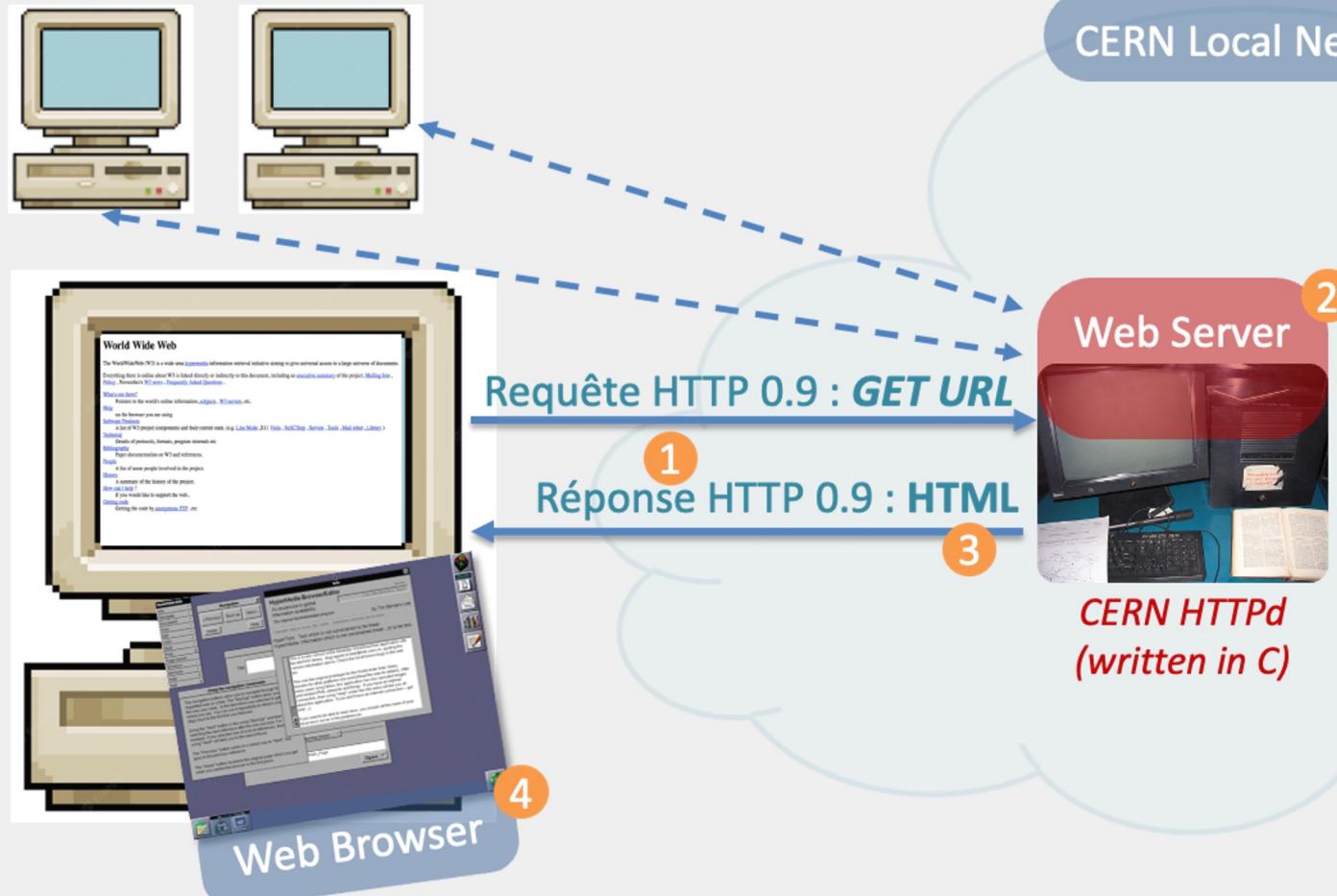
Pas d'interaction (Web 2.0)

Any ideas?



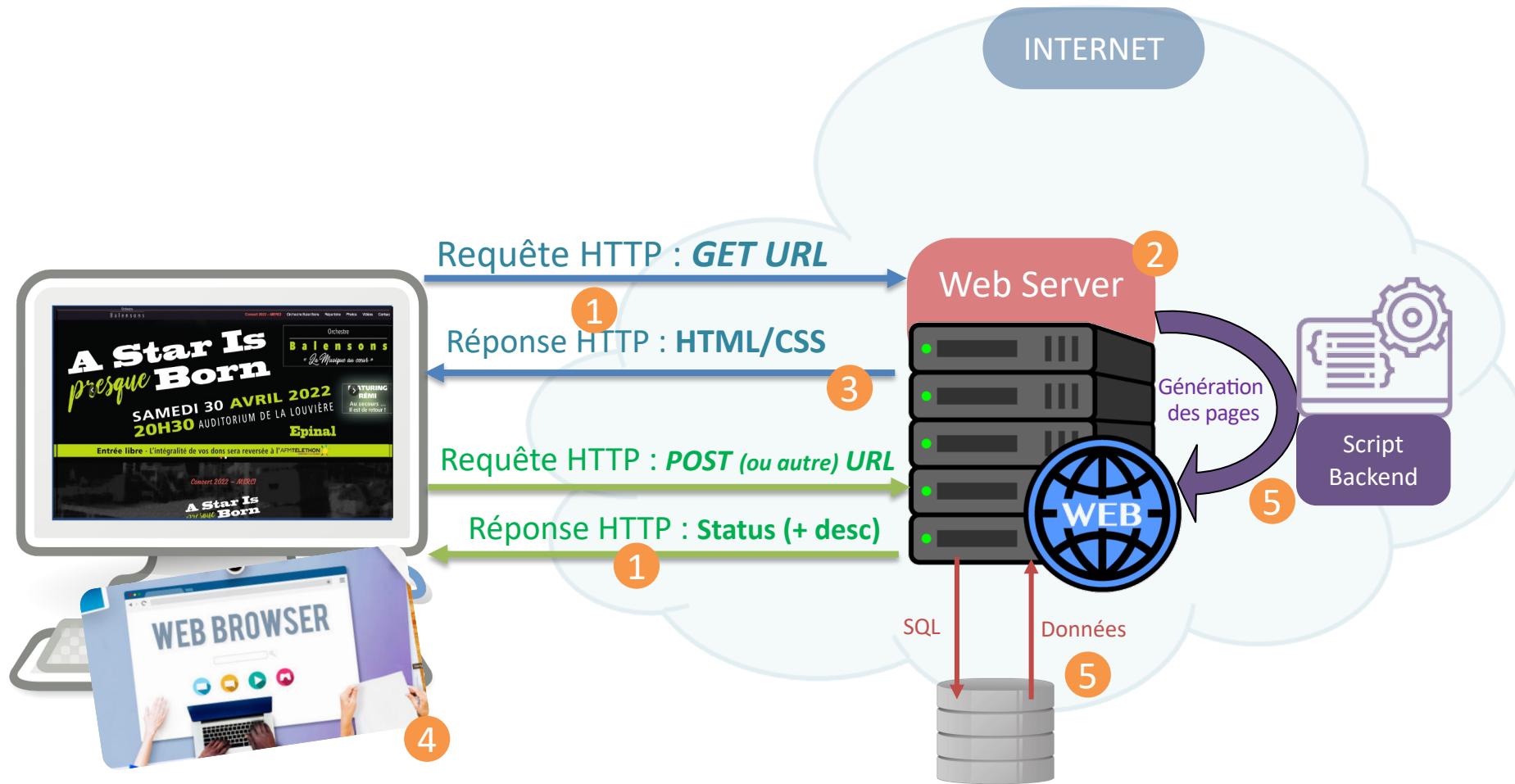
WEB 1 : Evolution du Web

Modèle TBL (89) :



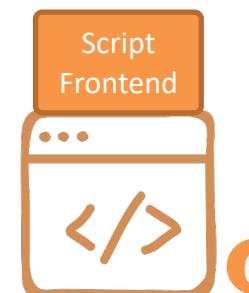
WEB 1 : Evolution du Web

Évolution ...



WEB 1 : Evolution du Web

Évolution ...



6



Requête HTTP : **GET URL**

1

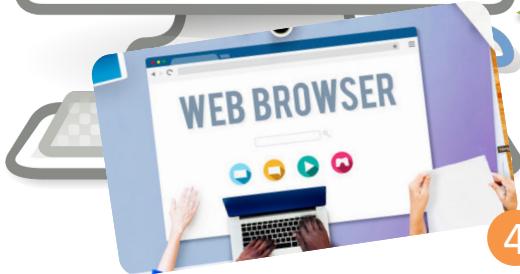
Réponse HTTP : **HTML/CSS / JS**

3

Requête HTTP : **POST (ou autre) URL**

1

Réponse HTTP : **Status (+ desc)**

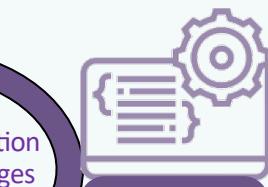


4

INTERNET

Web Server

2



Script Backend

5



SQL

Données

5

1 Le protocole

http://

1.1



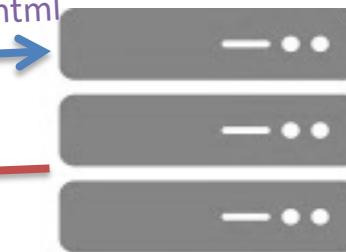
Requête HTTP sur une URL

Réponse HTTP

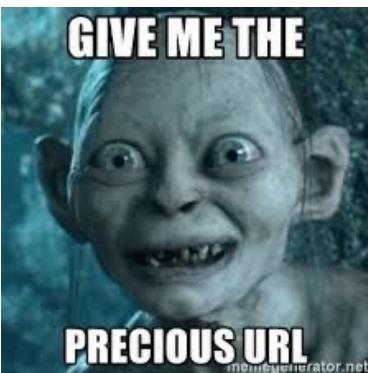


GET http://info.cern.ch/hypertext/WWW/TheProject.html

Code de réponse
+ HTML de la page



1 Le protocole **http://** : l'URL (Unique Resource Location)



- Est composée de 4 parties :
 - Le protocole : http / https / ftp, ...
 - L'Hôte : www.google.fr, www.monservice.fr, 192.168.1.28, ...
 - Le port : 1664, 80 (par défaut pour http), 443 (par défaut pour https), ...
 - Le chemin : /mesimages/image.php, /index.php, ...



- Identifie la **ressource** de manière **unique** (*document html ou ressource*)
- Si le fichier demandé est :

Un fichier statique (ex. HTML) :

Son contenu est simplement renvoyé au navigateur client, précédé de quelques headers

Un fichier de script (ex. PHP) :

- Celui ci est exécuté. (*Il peut récupérer des informations contenues dans la partie requête de l'URL*)
- Pendant son exécution, il écrit du texte dans sa sortie standard.
- Le texte produit par le script est finalement renvoyée au navigateur client, précédé de headers.

Introuvable
(*URL ne correspond pas à un fichier*) mais qu'il existe des règles de réécritures permettant de l'associer à un script existant :

- Celui ci est exécuté.



Cf. URL-REWRITING

1 Le protocole **http://** : La requête

- Permet à un client d'envoyer des messages à un serveur

- Format d'un message HTTP :

- Message Header

- Request Line : Infos générales (URL, ...)
 - Request Headers : Données d'entête [Optionnel]

- Message Body [Optionnel] ←



Dans le cas d'un POST
(envoi de données au serveur)

1 Le protocole : La requête

- Permet à un client d'envoyer des messages à un serveur

- Format

- Message Head

- Request URL: `http://annemilienmicard.freeboxos.fr:55555/api/books/`

- Request Method: GET (ou POST, PUT, DELETE, ...), voir plus loin ...

- Message Head

- Request Line : Infos générales (URL, ...)

- Request Headers : Données d'entête [Optionnel]

- Message Body [Optionnel]



1 Le protocole **http://** : La requête

- Permet à un client d'envoyer des messages à un serveur

- Format

- Mess

- Request Line
 - Request Headers : Données d'en-tête [Optionnel]

- Message Body [Optionnel]

▼ Request Headers [view source](#)

Accept: */*

Accept-Encoding: gzip, deflate

Accept-Language: fr-FR,fr;q=0.8,en-US;q=0.6,en;q=0.4

Cache-Control: no-cache

Connection: keep-alive

Host: www.tplm.org/balensons

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.116 Safari/537.36

générales (URL, ...)

1 Le protocole **http://** : La requête

- Permet à un client d'envoyer des messages à un serveur

- Format d'un message HTTP :

- **Message Header**

- **Request Method**
 - **Request URI**
- ▼ **Form Data** [view source](#) [view URL encoded](#)

login: maitreyoda@force.be.u
password: xxxxxxxxx

- **Message Body [Optionnel]**



1 Le protocole **http://** : La réponse

- Permet au client d'avoir un retour sur la requête;
(Qu'il soit ou non en attente d'un contenu)

- Format d'une réponse HTTP
 - Message Header
 - Response Line
 - Request Headers : Données d'entête
 - Response Message [Optionnel]



1 Le protocole **http://** : La réponse

- Permet au client d'avoir un retour sur la requête;
(Qu'il soit ou non en attente d'un contenu)



▼ General

Request URL: <http://www.tplm.org/balensons>

Status Code: ● 200 OK (ou 404, 403, ...), voir plus loin ...

Remote Address: 23.21.201.133:443

• Format

– Message Header

- Response Line

- Request Headers : Données d'entête



– Response Message [Optionnel]

1 Le protocole **http://** : La réponse

- Permet (Qu'il soit

- Format

- Message

- Response Line
 - Request Headers : Données d'entête

- Response Message [Optionnel]

▼ Response Headers [view source](#)

Connection: keep-alive

Content-Length: 216

Content-Type: text/html; charset=UTF-8

Date: Tue, 01 Mar 2016 18:14:46 GMT

Etag: W/"d8-sas2ykRj0XM5VS1ZQpkUNg"

Server: Cowboy

Via: 1.1 vegur

X-Powered-By: Express



1 Le protocole **http://** : La réponse

- Permet au client d'avoir un retour sur la requête;
(Qu'il soit ou non en attente d'un contenu)

- Format de la réponse

– Message [Optionnel]

- Response Message [Optionnel]
- Request Message [Obligatoire]

```
<!DOCTYPE html>
<html lang="fr-FR">

<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1,
minimum-scale=1">
<link rel="profile" href="http://gmpg.org/xfn/11">
<title>Concert 2022 – MERCI – Orchestre Balloons Orchestre variété  
Vosges Nancy Meurthe-et-Moselle Epinal Groupe  
Musique mariages danser repas dansant concert</title>
```

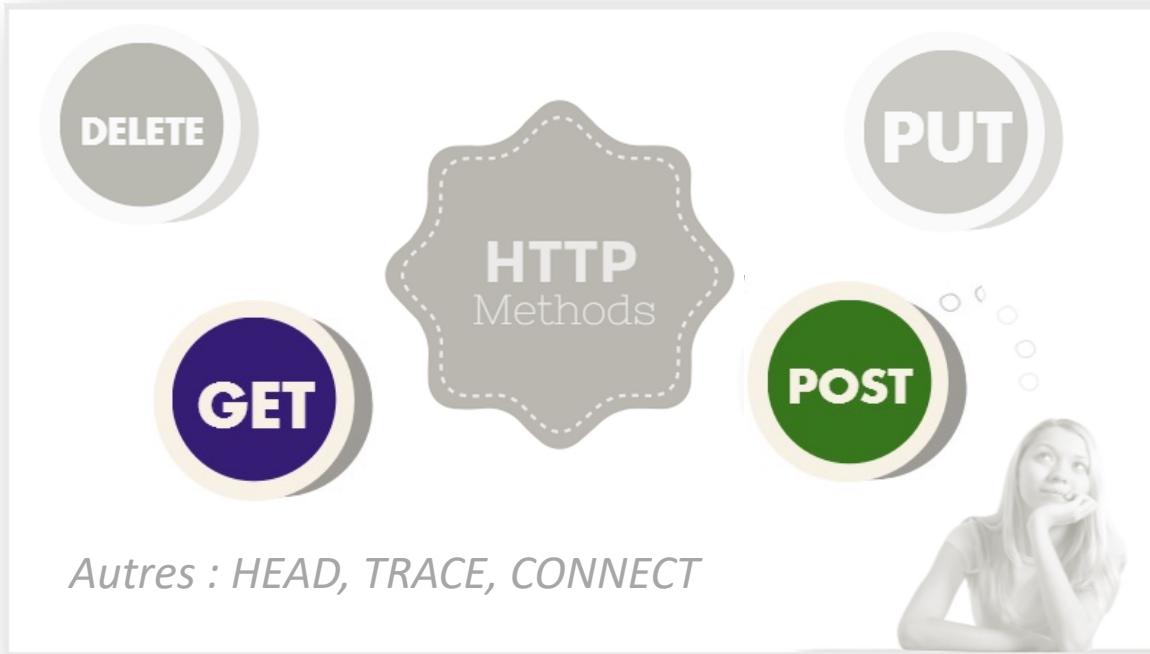
– Response Message [Optionnel]

1

Le protocole <http://> : Les méthodes

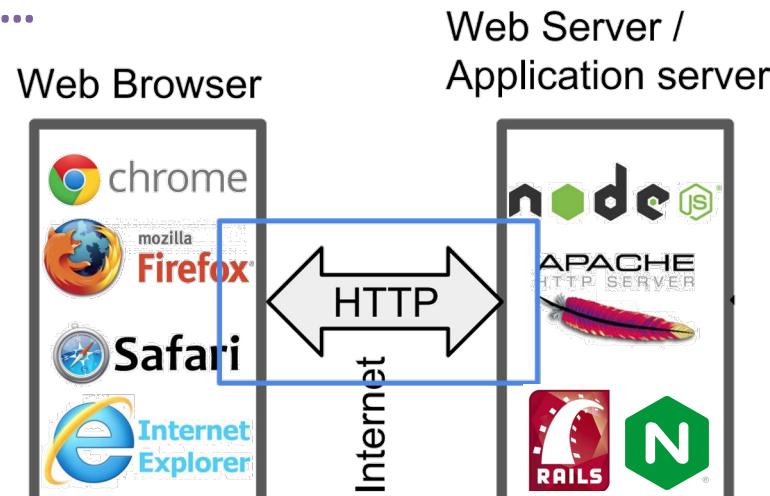


- HTTP définit un ensemble de méthodes permettant de caractériser les requêtes



On va y revenir ...

- Un **serveur Web** est une **application côté serveur** qui est en mesure de répondre à des **requêtes HTTP**.
 - Il communique avec les clients en ouvrant des connexions **TCP/IP (host+ports)**
 - Il prend en charge les requêtes HTTP des différents clients (avec ou sans parallélisme) et renvoi une réponse HTTP, le plus souvent avec du **contenu HTML, CSS, JS...**



LE SAVIEZ-VOUS ?



Il est possible de communiquer avec le client et le serveur web sur la même machine en utilisant l'IP **localhost (127.0.0.1)** ➔ en TP

3 HTML / CSS

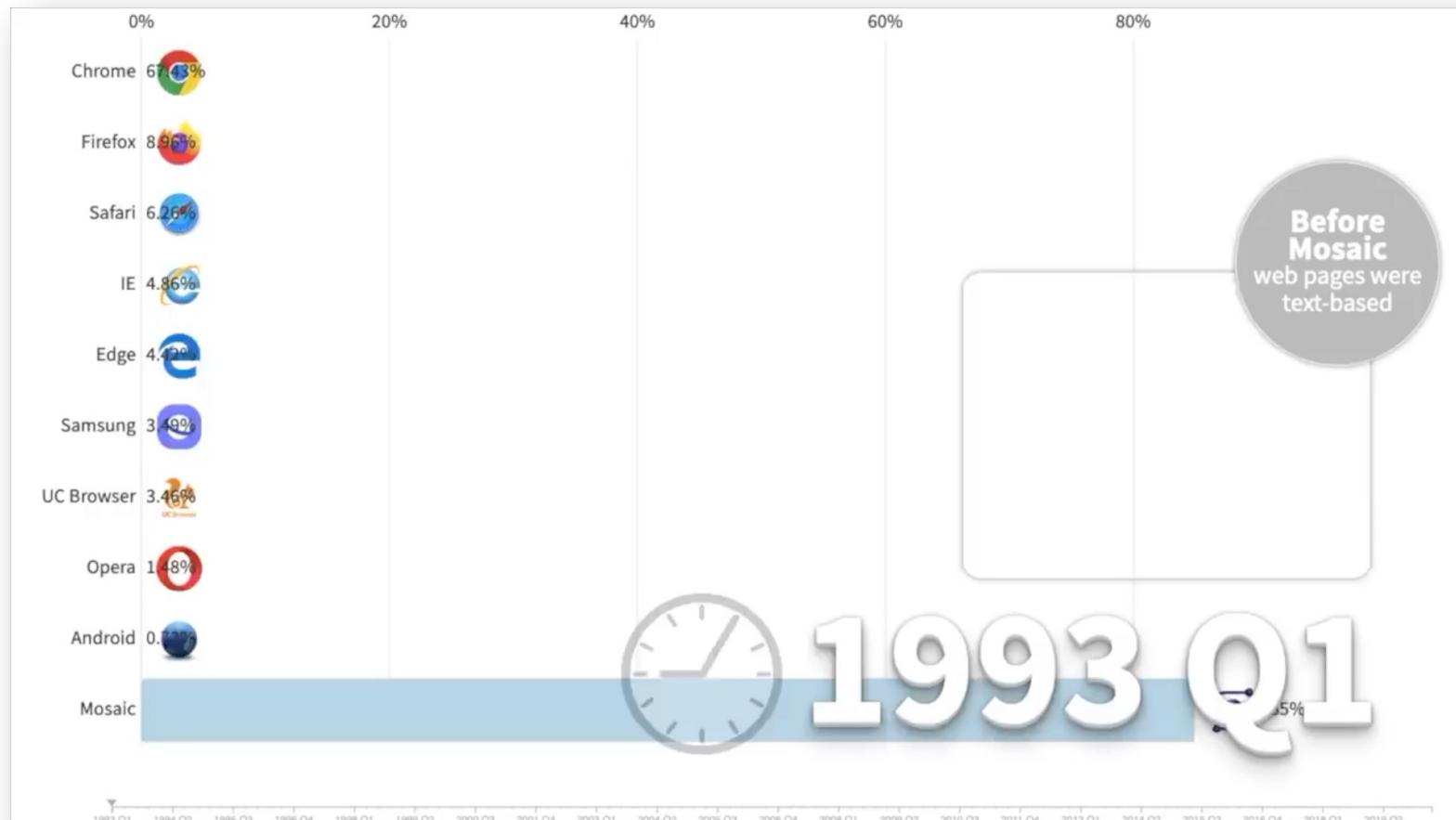
OH I REMEMBER

WAIT OH I REMEMBER

4

Navigateur Web

- Un navigateur web (web browser) est un logiciel permettant d'accéder à une **page web** et de l'afficher sur l'écran de l'utilisateur. Un tel logiciel doit donc être capable de **lire** et **d'interpréter** des fichiers **HTML**. Les navigateurs web accèdent au web à travers le protocole HTTP. On parle de « **client HTTP** ».



5 Langages de prog. « Backend » (côté serveur)

- Un langage de programmation « **Backend** » permet de générer, côté serveur, des **pages web dynamiques**.
- Une **page web dynamique** est une page web **générée à la demande**. Le contenu peut donc varier en fonction d'informations (heure, nom de l'utilisateur, formulaire rempli par l'utilisateur, etc.) qui ne sont connues qu'au moment de sa consultation. À l'inverse, le contenu d'une page web statique est a priori identique à chaque consultation

 Java	 PHP	 JavaScript	 NodeJS
<ul style="list-style-type: none">• Airbnb• Uber• Pinterest• LinkedIn• eBay	<ul style="list-style-type: none">• Facebook• Viber• Hootsuite• Buffer• Yahoo• Wordpress• Wikipedia	<ul style="list-style-type: none">• Netflix• Candy Crush• Facebook	<ul style="list-style-type: none">• Airbnb• eBay• Square• Asana
 Kotlin	 Python	 Ruby	 C#
<ul style="list-style-type: none">• Google• Amazon• Pinterest• Foursquare• Trello	<ul style="list-style-type: none">• Google• Instagram• Spotify• Quora	<ul style="list-style-type: none">• GitHub• Kickstarter• Basecamp• Scribd	<ul style="list-style-type: none">• Twitch• GitHub• Telegram• MasterCard

LE SAVIEZ-VOUS ?



HTML n'est pas un langage de Programmation mais un langage descriptif

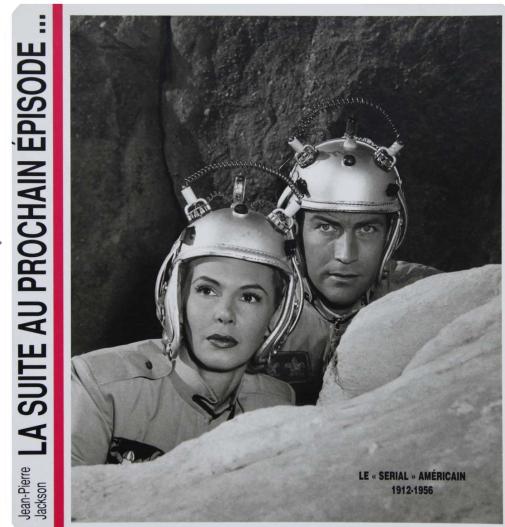
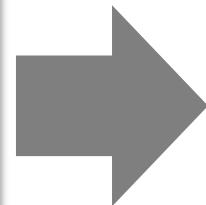
On ne retrouve pas les structures de contrôle (*tests, boucles, ...*) et structure de données (*variables*)

5 Langages de prog. « Backend » (côté serveur)

- Pour développer des sites plus gros, on va privilégier l'utilisation de **Frameworks**...

Frameworks / Langages

 Node.js <ul style="list-style-type: none">• Express.js• Noa.js• Meteor.js	 Ruby <ul style="list-style-type: none">• Ruby on Rails• Hanami• Sinatra	 Python <ul style="list-style-type: none">• Django• Flask• Web2py
 Java <ul style="list-style-type: none">• Spring• Grails• JSF	 PHP <ul style="list-style-type: none">• Laravel• CodeIgniter• Symfony	 C# <ul style="list-style-type: none">• .NET



6 Langages de prog. « Frontend » (côté client)



- Un site Web peut être composé uniquement d'HTML et de CSS, mais si on veut insuffler un peu de dynamisme on aura besoin de **Javascript** : un vrai langage de programmation... Il sera responsable de **l'interactivité** et de la **logique** qu'il y a derrière nos pages web et s'exécute uniquement **côté client**, sur le **navigateur**.
- Un tel langage est aussi utilisé pour charger du contenu depuis le serveur de manière asynchrone → **AJAX**
- Comme pour les langages « Backend » il y a plusieurs **Frameworks** basé sur **JS**

« Frontend VS Backend » : Bilan

