DÉVELOPPEMENTS, WEB ET BASE DE DONNÉES

- Nicolas Perrier

Introduction

Objectifs de ce module:

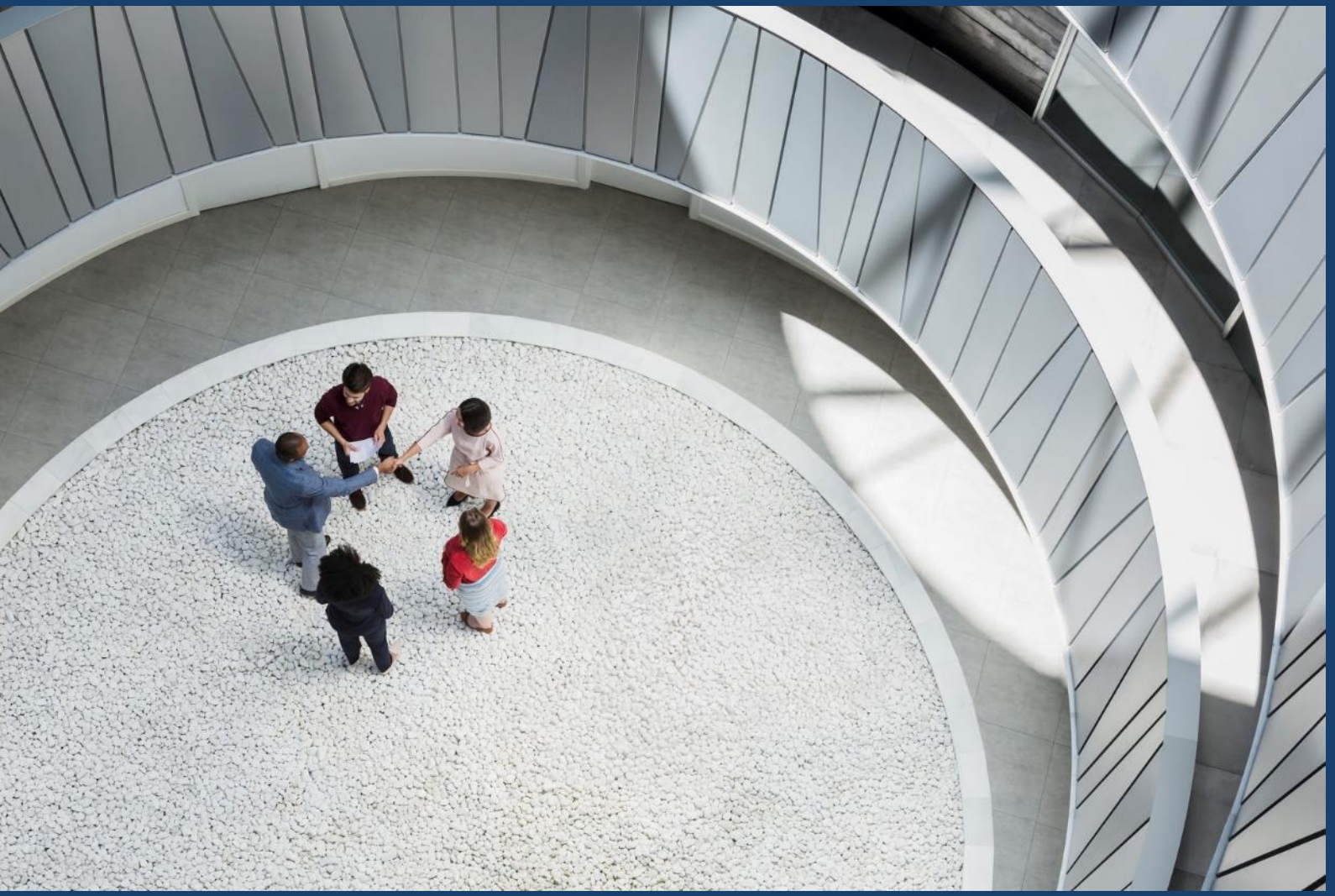
- Vous permettre d'appréhender la programmation informatique (via Python)
- Comprendre l'environnement WEB
- Réaliser des premières applications WEB (formulaire, restitution de données, rendu visuel, etc)
- Comprendre les IA

Pas de panique !!!

- Toutes les notions seront revues, approfondies lors des TD
- Le rythme sera adapté !

Contenu cours

- Programmation informatique
- Internet&Co, langages Web
- Bases de données (rappel)
- IA



Programmation

- Définitions
- Python
- Premier pas
- Syntaxe
- Structure et POO

Programmation - Définitions

Pourquoi ?

- Elle sert à résoudre un problème (faire un calcul, exécuter des actions, aider à la prise de décision, modéliser, etc)
- Elle est écrite dans un langage qui contient des ordres ou instructions élémentaires que l'ordinateur peut « comprendre, interpréter et exécuter »

Langages

- Il existe de très nombreux langages informatiques
- Ces langages peuvent être regroupés en deux catégories principales: impérative (déroulement) ou déclarative (objectif à atteindre)
- La programmation impérative est la plus répandue

Programmation - Définitions

Programmation impérative

- La programmation impérative consiste à écrire un programme en donnant des instructions qui modifient les données rangées dans des variables identifiées par des noms
- Les **programmations structurée**, **procédurale** et **modulaire** sont trois approches majeures d'écriture et de structuration du code logiciel

Contexte et langage

- Web: via le réseau Internet, avec le HTML, Javascript, CSS, PHP
- Client lourd: Python, Java, C#, etc
- Serveur/API: Python, Java, etc
- Embarqué/temps réel : C

Programmation - Définitions

Instruction

- Une instruction comporte des mots-clés ou symboles définis par le langage et des expressions qui ont une valeur au moment de l'exécution
- Exemple :
  - **if (x == 0)**
  - **y = 5 * x;**
- Les types d'instruction dépendent du langage qu'on utilise

Variable

- Une variable correspond à un emplacement dans la mémoire centrale
- Une variable est utilisée par le programme pour enregistrer une valeur qu'il réutilisera dans la suite de son exécution
- Une variable est identifiée par son nom
- **Affectation** : action d'enregistrer une nouvelle valeur dans une variable ; par exemple « x = 8; »

Programmation - Définitions

Natures des instructions

- Pour les langages habituels, de type impératif, les grands types d'instructions sont :
  - *L'affectation d'une valeur à une variable (et déclaration de type selon langage)*
  - *L'alternative ou « if »: selon la valeur d'une expression, une séquence d'instructions est exécutée, ou une autre*
  - *La répétition (for, while) : une séquence d'instructions est répétée un certain nombre de fois*

Attention avec le «=»

- Ne pas confondre le « = » de l'affectation avec le « = » mathématique (souvent écrit «==»)
- Il n'est pas symétrique :
  - à gauche doit se trouver un nom de variable qui désigne un emplacement mémoire
  - la droite peut contenir n'importe quelle expression qui calcule une valeur qui peut être rangée dans l'emplacement mémoire désigné par la gauche

Programmation - Définitions

« Portée » d'une variable

- La portée d'une variable désigne la portion du programme où la variable peut être utilisée.
- Portée d'une variable en python : dans le module, dans la fonction/méthode

Langage typé… ou pas

- Java est un langage typé : on doit déclarer le type d'une variable avant de l'utiliser
- Il existe d'autres langage non typés (Javascript, Python) dans lesquels les types des variables ne sont pas indiqués dans le programme
- Les langages typés sont moins souples mais plus sûrs car davantage d'erreurs peuvent être détectées en amont

Programmation - Définitions

Fonction vs méthode

- En programmation, on distingue les deux termes:
  - La fonction est « **indépendante** »: elle dispose de données d'entrée, définit un traitement et propose des données de sortie
  - La méthode est **associée à un contexte d'appel**, on parle de méthode « objet ». Exemple: mettre en minuscule la chaine de caractères, cette dernière étant le contexte

Interface (prog)

- Dans un langage de programmation, une **interface** est une trame minimum à respecter. Exemple: la méthode « Avancer » est nécessaire, mais pas la méthode « Reculer » pour un objet « Voiture ». Seule la méthode « Avancer » sera donc présente dans l'interface et devra donc être décrite
- Pour utiliser une interface, on parle **d'implémentation**.

Programmation - Définitions

POO

- Acronyme pour Programmation Orientée Objet. Cela désigne une approche de programmation où les concepts sont modélisés via des objets « numériques ». Trois notions à retenir :
  - Classe
  - Instance
  - Héritage

Classe Objet

- Définition de l'objet: ses propriétés, ses capacités exprimées par des méthodes. On peut parler de « plan » ou « modèle »
- Cette définition doit être exhaustive, elle peut évoluer dans le temps (plus ou moins difficile selon le langage)
- Exemple: une voiture, qui a une couleur, un poids, une puissance et peut avancer, reculer, etc.
- Ces propriétés peuvent être communes à tous les objets (statiques), ou bien propres à chacun

Programmation - Définitions

Instance d'objet

- Il s'agit de la **concrétisation** d'un objet: il existe en mémoire, et fonctionne selon sa **classe d'appartenance** (un objet est toujours associé à une classe)
- Ses propriétés évoluent dans le temps, elles sont **persistantes** en mémoire vive (tant que le programme tourne), ou bien après une **sérialisation** et un stockage (dans un fichier, une base de données, etc)
- Passage entre une **instance** d'objet et une **BDD relationnelle**: via des outils appelés **ORM** (object relational mapping)

Héritage

- Objectif: **mutualiser** des **propriétés** et **méthodes** pour des objets de classes différentes, éviter de ré-écrire la roue !
- Exemple: la classe Véhicule définit les propriétés poids et couleur, ainsi que les méthodes avancer et reculer. La classe Voiture dispose des mêmes besoins. Aussi, on créé la classe Voiture en héritant de la classe Véhicule, i.e., ses propriétés et méthodes. Seules les particularités sont donc à décrire pour la classe Voiture
- On peut **surcharger**: par exemple, dans notre cas, il s'agit de **redéfinir** la méthode « avancer » pour la classe « Voiture »

Programmation - Définitions

M(odèle)V(ue)C(ontrôleur)

- Concept architectural pour développer des applications. Objectif: ne pas mélanger la modélisation, le traitement et la restitution !
- **Modèle**: il s'agit de la définition des objets qu'il faut utiliser. Les classes d'objet !
- **Vue**: il s'agit des écrans utilisateurs qui lui permettent d'interagir avec l'application
- **Contrôleur**: outil qui assure le pont entre les données sortante à destination de la vue (qui ne doit que restituer des données, pas de calcul) et celles entrantes (saisies par l'utilisateur) à destination du programme

Interface/api

- Objectif: permettre à différentes applications, i.e., programmes de **communiquer ensemble**. On parle de **contrat d'interface**, ou interface, l'idée étant de décrire la façon d'échanger.
- Il faut donc décrire:
  - Des points d'entrées
  - Des points de sorties
  - Des langages d'échanges: REST, SOAP
  - Un format de données: JSON, XML
- On parle aussi de bus applicatif (ESB): application centrale qui assure la communication entre tous les programmes

Programmation - Python

Origines

- La première version de Python a été créé en 1989 par Guido Von Rossum pendant une semaine de vacances
- Pourquoi Python ? Car Guido Von Rossum est un fan de la série télévisée « Monty Python »

Caractéristiques

- langage interprété
- langage orienté objet
- langage à typage dynamique
- portable (unix, windows, MAC)
- interfaçable (Java, .NET)
- apprécié des pédagogues pour sa syntaxe claire et intuitive

Programmation - Python

Domaines d'application

- Framework web (Flask, Django, Zope)
- Calcul scientifique (Panda, Numpy)
- Bibliothèques graphiques (wxPython)
- Administration système
- ORM (SQLAlchemy)
- Big Data, Deep Learning, IA

Fonctionnement: l'interpréteur

```
artymon@kerouac:~$ python
Python 2.7.1+ (r271:86832, Apr 11 2011, 18:13:53)
[GCC 4.5.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Programmation – Python

Cours en ligne

- Syntaxe et langage:

https://cours.artymon.com/python/syntaxe.html

Merci !

Programmation - Python

Un interpréteur, deux interpréteurs, etc

- Un système peut avoir plusieurs interpréteurs
- Les versions de ces interpréteurs peuvent différer
- Un interpréteur pour un projet est une bonne pratique !

Installation – Par Windows

- Des installateurs graphiques sont disponibles sur https://www.python.org
- Une fois l'installation terminée l'interpréteur est disponible dans le menu *Programmes*

Programmation - Python

Installation Linux - Par paquet

- Par défaut, les environnements Linux proposent l'installation du Python via des commandes comme:
  - apt-get install pythonX.X
  - yum install pythonX.X
- Attention: ce python ainsi installé est parfois utilisé par le système de votre ordinateur ! Il est donc nécessaire de créer des pythons **virtuels** ou par compilation.

Installation Linux – Par compilation

Téléchargement et extraction

```
wget http://www.python.org/ftp/python/X.X.X
tar xjf Python-X.X.X.tar.bz2
cd Python-X.X.X/
```

Option de configuration

```
./configure --help
`configure' configures python X.X to adapt to many kinds of systems.

Usage: ./configure [OPTION]... [VAR=VALUE]...

...

Installation directories:
  --prefix=PREFIX         install architecture-independent files in PREFI
                          [/usr/local]


By default, `make install' will install all the files in
`/usr/local/bin', `/usr/local/lib' etc.  You can specify
an installation prefix other than `/usr/local' using `--prefix',
for instance `--prefix=$HOME'.
```

Configure, make, make install

```
./configure --prefix=/home/artymon/pythons/pyXXX
make
make install
```

L'interpréteur est disponible dans /home/artymon/pythons/pyXXX/bin/python

Programmation - Python

PIP – Python Index Package

- pip est un programme en ligne de commande
- but: permettre l'installation de librairies supplémentaires, non embarquées dans python par défaut
- Site officiel https://pypi.org, solution à tous vos besoins !

PIP – Usages

- Lorsque vous installez pip, une commande pip est ajoutée à votre système, qui peut être exécutée à partir de l'invite de commande comme suit :

pip option nom_librairie

Les principales options à passer: install, uninstall, update, etc

Programmation - Python

1 python, 2 python, 3 python…

- Quand on commence à beaucoup programmer, il est rapide d'accumuler plusieurs projets en cours de développement sur sa machine. Certains sont vieux, d'autres récents… Mais ils ont tous un point commun: ils utilisent tous des bibliothèques différentes, avec forcément des versions différentes… voire incompatibles.

  Le jour où ça casse, c'est le chaos.

- La solution: des environnements **virtuels**, des « Python » isolés de l'OS, et séparés les uns des autres pour chaque projet.

Virtual ENV

- Depuis le python « **socle** », une commande est disponible: *venv*. Pour chacun de vos projets, créer un environnement virtuel devient simple :

venv /path/vers/projet/env_nom_du_projet

- Venv va créer un dossier (à l'emplacement désigné) contenant un environnement complet : l'interpréteur Python, les librairies, les commandes, etc. C'est votre installation **isolée**.

Programmation - Python

Pour aller plus loin…

- D’autres outils existent pour faciliter la création d’environnement de travail:
  - Pipenv
  - virtualenvwrapper

Programmation – Python

Cours en ligne

- Structuration du code:

https://cours.artymon.com/python/structuration_code.html

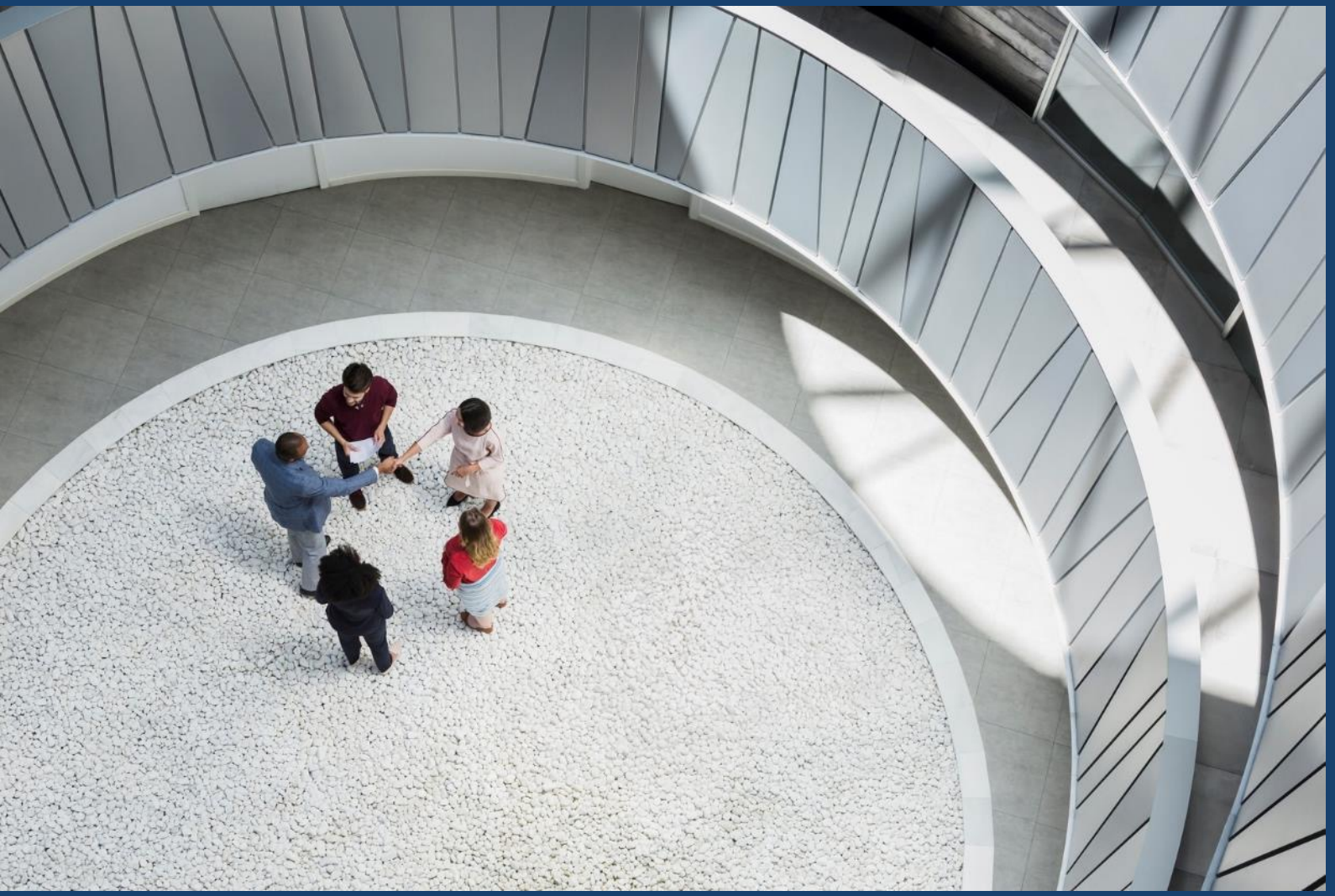
- Outils transverses:

https://docs.python.org/3/library/functions.html#open

https://docs.python.org/3/library/functions.html#input

https://docs.python.org/3/library/os.html (walk)

Merci !



Internet&co

- Définitions
- DNS
- Client et serveur
- Protocole HTTP

Le WEB

D’où vient le terme Web ?

- Créé en 1990 par **Tim Berners-Lee** (considéré comme l’inventeur d’internet), père du premier **navigateur** et **éditeur** web nommé ***WorldWideWeb***
- Aidé dans ses développements par son collègue **Robert Cailliau,** l’auteur du premier logo du WWW

Kesako ?

- **Web ou WWW (***World Wide Web* ou *Toile mondiale* en français): consulter via un navigateur des pages regroupées sur des sites via le réseau **internet** (créé en 1969)
- Toile d’araignée ? Découle des liens hypertextes qui relient les pages entre elles
- Le Web est l’une des applications d’**internet: courrier électronique** (email ou *courriel* en français), la **messagerie instantanée**, le **partage de fichier en peer to peer** ou l**e partage de fichier** via le protocole **FTP, etc**

Le WEB

Premier site web

- Toujours consultable à cette adresse: http://info.cern.ch/
- Hébergé à l'époque sur le réseau du CERN en Suisse
- Ce site décrivait le fonctionnement, donnait des conseils et contraintes d'écriture d'une page web

Attention ! Piège(s)

- Par extension, il désigne couramment (et donc de façon inexacte) **internet** dans sa globalité
- Des traces dans les adresses des sites internet (ex: *https://**www**.google.com*) appelées également URL (*Uniform Resource Locator*)
- Attention, les ***www*** ne sont pas nécessaires pour accéder à un site web. Faites le test : accéder à un site avec ou sans les 3 w

Le WEB

Le web 2.0, c'est quoi ?

- Terme **web 2.0** utilisé pour la première fois par Dale Dougherty en 2003 et popularisé par Tim O'Reilly dès 2004
- Très à la mode après l'éclatement de la bulle internet en 2001
- Désigne une nouvelle version d'internet: une infrastructure plus **complexe** et **évoluée**, de nouveaux usages et un **internaute** « **acteur** »
- **Mais:** terme est très largement galvaudé car trop lié au « marketing »
- En bref, à retenir: un web 2.0 est un web plus **simple** pour l'internaute, plus **collaboratif** et **participatif**
- La meilleure illustration: les réseaux sociaux… et ses dérives

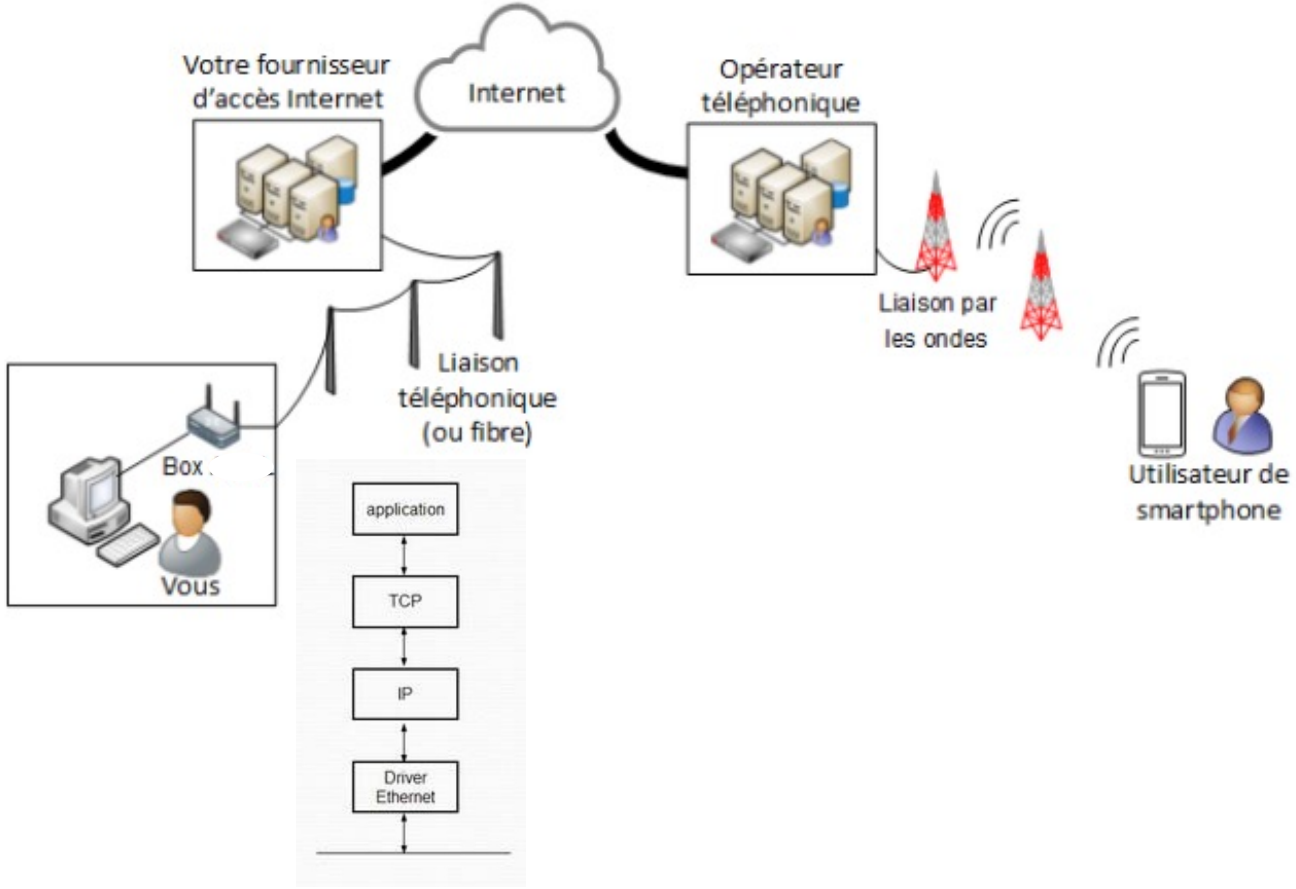
Internet

Internet

- **Internet est un réseau reliant des ordinateurs du monde entier.**

Quelques règles fondamentales :

- Les ordinateurs connectés à Internet doivent avoir une adresse pour être contactés. C'est l'adresse **IP** suite de chiffres séparé par des points avec 2 formats (IPV4, IPV6)
- Pour communiquer entre eux, les ordinateurs doivent utiliser le même langage: le **protocole**.
- Quelques termes: ethernet, TCP/IP, modèle OSI



Internet

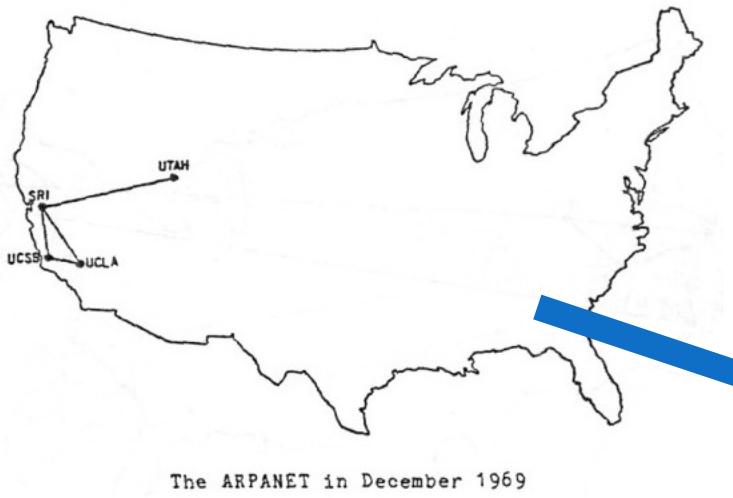
Quelques acteurs « internet »

- Les **fournisseurs d'accès Internet**: Orange, Free, etc. La porte d'entrée
- Les **fournisseurs de contenus et de services**, (GAFAM, sites d'actualités, moteurs de recherche, OpenDATA, etc)
- Des **CDN** (Content Delivery Network) : mise à disposition du contenus avec des optimisations selon la provenance du demandeur
- Des opérateurs de transit IP (permettre de faire le lien entre les différents acteurs, pays, etc).

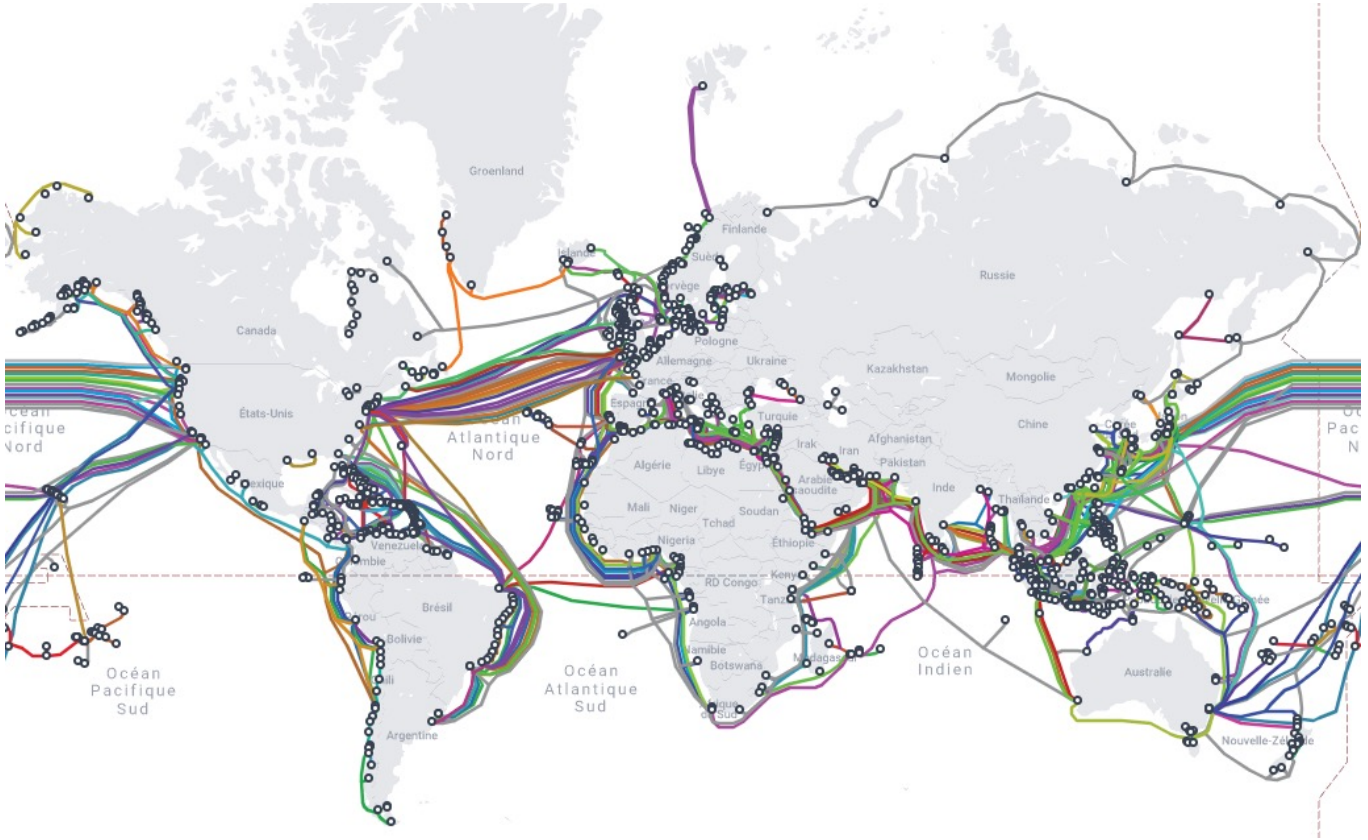
« Backbone » ou Dorsale Internet

- Au départ: le réseau Internet est organisé autour d'une dorsale: ARPANET
- 1989: d'autres dorsales arrivent: NSFNet, MILNET de l'armée américaine
- Rapidement: la centralisation du routage devient obsolète.
- Depuis 1995, Internet repose entièrement sur des réseaux appartenant à des entreprises de services Internet.
- « l'Internet backbone » n'a plus de sens à date: aucun réseau n'est officiellement au cœur d'Internet.

Internet



The ARPANET in December 1969



DNS

Vous avez dit DNS ?

- Signifie « **Domain Name System** » ou système de nom de domaine
- Principe: l'exemple du téléphone ! Vous souhaitez appeler quelqu'un dont vous ne connaissez pas le n° de téléphone ?

Que faites-vous ?…. vous utilisez un annuaire !

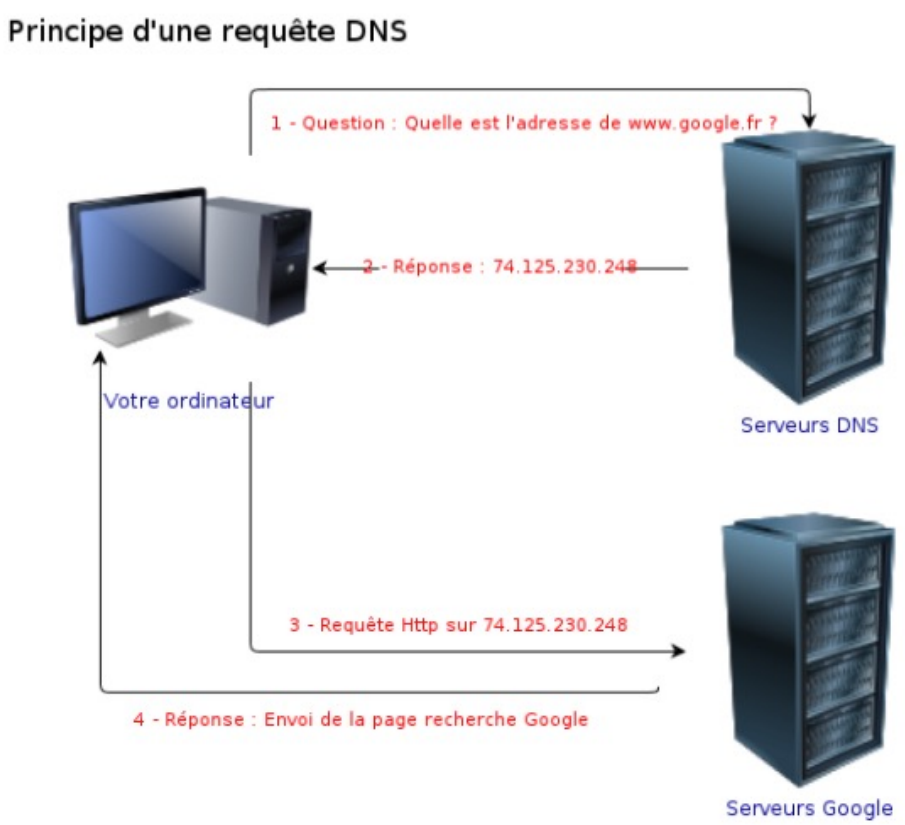
Un **serveur DNS est un annuaire pour ordinateur**.

Définition

- Le serveur DNS permet la relation entre le nom d'ordinateur et l'adresse IP. *(ramené au téléphone, on obtient : nom de l'abonné et n°)*

DNS

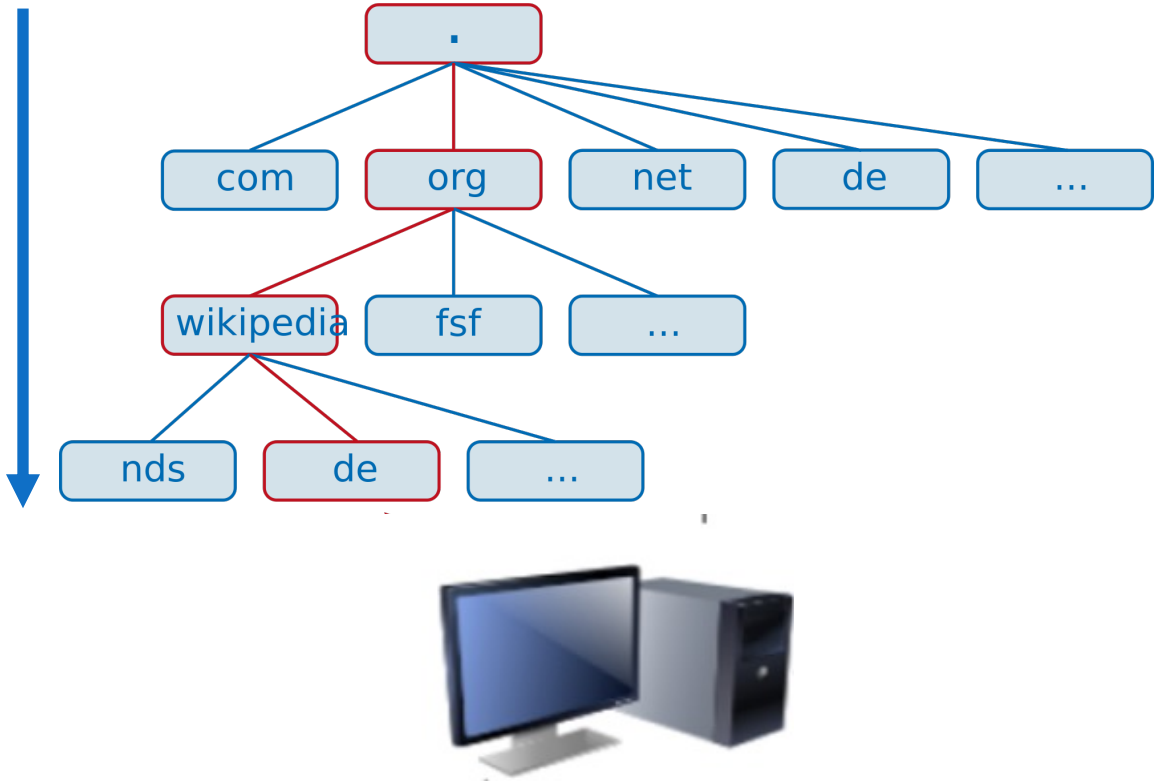
Par l'exemple: aller sur www.google.fr



DNS

Plus complexe car en « cascade »

- de.wikipedia.org(.), c'est dans l'ordre :
  1. « . »: serveur racine DNS renvoie vers serveur « org »
  2. « org » vers « wikipedia »
  3. « wikipedia » vers « de »
  4. « de »: vous êtes arrivé !
- Serveurs racine, le premier « . » sont gérés par 12 organisations dans le monde



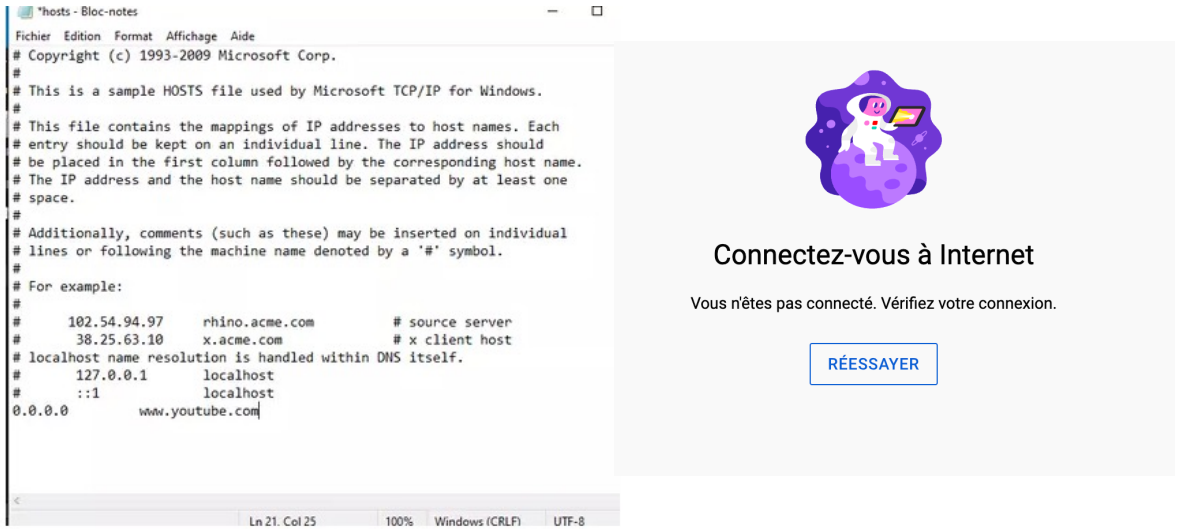
DNS

Cybersécurité

- DNS: nœuds très sensibles sur Internet, surtout les « racines » au début de tout !
- Pourquoi: exemple concret : prenons un faux annuaire (piraté): vous appelez votre banque via cet annuaire… mais vous tombez sur des pirates , et vous ne le savez pas !
- Le DNS avec l'annuaire AD) sont le « Saint Graal » des pirates
- Sur le WEB, le HTTPS permet de palier en partie à cela (correspond entre autre à une empreinte pour chaque site garantissant que vous êtes au bon endroit !)

Bonne blague

- Changer (si possible !) le fichier hosts
- Sous Windows: **C:**, **Windows, System32, drivers** et enfin **etc**
- Permettre de forcer l'IP associée à un DNS !



Dark Web

Dark web

- Aussi appelé **web clandestin** ou encore **web caché**
- Ce sont des réseaux « superposés » qui utilisent l'Internet public
- Réseaux seulement accessibles via des logiciels, des configurations ou des protocoles spécifiques
- Le dark web forme une petite partie du deep web (ou web invisible, toile profonde), la partie d'Internet qui n'est pas indexée par les moteurs de recherche
- **Web invisible ou Deep Web:** environ 75% du trafic internet

Darknet

- Nom donné aux réseaux qui hébergent le dark web: réseaux ami-à-ami, de pair à pair, ainsi que de grands réseaux populaires tels que **Freenet, I2P et Tor (The Onion Router)**

Architecture d’échanges – Client / Serveur

Définitions

- Architecture **client/serveur:** désigne un mode de communication entre plusieurs ordinateurs d'un réseau
- Deux profils se distinguent le **poste client** et le **serveur**
- Chaque logiciel client peut envoyer des requêtes à un serveur qui envoie une réponse
- Un serveur est souvent spécialisé: applications, fichiers, ou encore de messagerie électronique, etc

Caractéristiques du serveur

- Il est **passif** (i.e., ne travaille pas par défaut)
- Il est **toujours à l'écoute**, prêt à répondre aux requêtes envoyées par des clients
- Dès qu'une requête lui parvient, il la traite et envoie une réponse: il devient « actif »

Architecture d'échanges – Client / Serveur

Caractéristiques d'un client

- Il est **actif**
- Il envoie des requêtes au serveur
- il attend et reçoit les réponses du serveur

Règles

- Le client et le serveur doivent utiliser le même protocole de communication: https, ftp, pop, smtp, etc
- Un serveur est généralement capable de servir plusieurs clients simultanément

A noter:

D'autres types d'architectures réseau existent: le poste à poste (ou peer-to-peer), dans lequel chaque ordinateur ou logiciel est à la fois client et serveur, le distribué (tout le monde participe au même niveau), etc.

Architecture d’échanges – Client / Serveur

Avantages (VS distribués)

- Toutes les données sont **centralisées** sur un **seul serveur**: facilité des contrôles de sécurité, de la mise à jour des données et des logiciels
- Les technologies supportant l'architecture client/serveur sont plus matures que les autres.

Inconvénients (VS distribués)

- Si trop de clients: surcharge et risque de déni de service (VS le réseau distribué dilue la charge)
- Si le serveur n'est plus disponible (panne): plus aucun des clients ne marche (VS le réseau distribué continue à marcher, même si plusieurs participants quittent le réseau)
- 1 cible à attaquer !

Architecture d’échanges – Client / Serveur

Exemples

- **Consultation de pages sur un site internet**:
  - un internaute connecté au réseau via son ordinateur et un navigateur web est le client
  - le serveur est constitué par le ou les ordinateurs contenant les applications qui délivrent les pages demandées

  Dans ce cas, c'est le protocole de communication HTTP(S) qui est utilisé.
- Les **courriels**:
  - sont envoyés et reçus par des clients
  - gérés par un serveur de messagerie

  Les protocoles utilisés sont le SMTP, et le POP ou l'IMAP.
- **La gestion d'une base de données**:
  - centralisée sur un serveur
  - Interrogeable à partir de plusieurs postes clients qui permettent de visualiser et saisir des données

Protocole HTTP

Définition

- Signifie « Hypertext Transfer Protocol (traduction: protocole de transfert hypertexte) »
- Définie la communication entre un **client** (exemple: navigateur) et un **serveur** sur le **World Wide Web** via **Internet**

Rappels

- Protocole inventé par Tim-Berner Lee au début des années 1990
- Fonctionne sur le principe « requête-réponse »
- Protocole orienté « fichier/document/page »
- Utilise des URL : adresse d'un site ou d'une page hypertexte sur Internet (ex. https://www.lerobert.com).
- Origine :(sigle anglais de *uniform* [ou *universal*] *resource locator* « repère uniforme [ou universel] des ressources »)

Protocole HTTP

Schéma

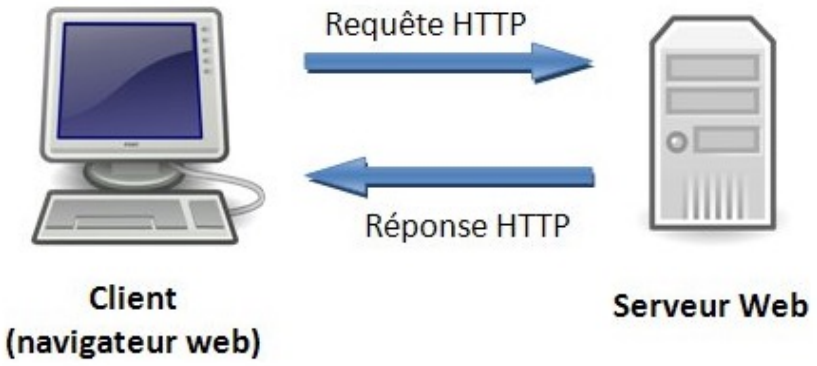


Schéma d'une requête HTTP

Principe

- L'ordinateur de l'internaute utilise le navigateur pour envoyer une requête à un serveur web
- Cette requête demande un document (exemple: page HTML, image, fichier de style CSS, vidéo, etc)
- Le serveur cherche les informations, effectue éventuellement des calculs
- Le serveur envoie la réponse. Cette réponse contient
  - les entêtes. Exemple: la date de modification
  - du contenu, et « normalement » le contenu demandé/attendu

Protocole HTTP

La requête: composition

- Une requête HTTP, c'est un ensemble de lignes de texte (contenant des instructions) envoyé au serveur par le navigateur.
- **On décompose la requête en:**
  - **Une ligne obligatoire**: le type de document demandé, la méthode qui doit être appliquée, et la version du protocole utilisée
  - **Des champs d'en-tête** : ensemble de lignes facultatives permettant de donner des informations supplémentaires sur la requête et/ou le client (Navigateur, système d'exploitation, etc)
  - **Un corps de la requête**: option, envoi de données (exemple: un formulaire web)

Principales méthodes

Commande	**Description**
GET	Demande une ressource située à l'URL spécifiée avec argument(s) possible(s) https://www.qwant.com/?client=ext-chrome-sb&q=test&t=web
POST	Envoi de données au programme situé à l'URL spécifiée (formulaire)

Protocole HTTP

La réponse: composition

- Réponse HTTP: un ensemble de lignes envoyées au navigateur par le serveur.
- Elle comprend :
  - **Une ligne de statut**: état de la réponse et autres informations
  - **Les champs d'en-tête** : ensemble de lignes facultatives permettant de donner des informations supplémentaires sur la réponse et/ou le serveur.
  - **Le corps de la réponse**: il contient le document demandé (image, texte, vidéo, etc)

Focus « code statut »

- Le code de réponse est constitué de trois chiffres : le premier indique la classe de statut et les suivants la nature exacte de l'erreur
- 2XX: succès
- 5XX: erreur
- Pour aller plus loin:

https://fr.wikipedia.org/wiki/Liste_des_codes_HTTP

Protocole HTTP

« Debug tools (F11) »

- Présent par défaut sur tous les navigateurs
- Analyse du contenu de la requête http (avec vos « traces »)

https://www-ensibs.univ-ubs.fr/fr/index.html

Protocole HTTP

Sécurité (faille)

- HTTP est un système sans état, ce qui signifie qu'il permet de créer des connexions à la demande
- L'unique objectif de HTTP: affiche les informations demandées (sans se soucier de la façon dont ces informations se déplacent)

-> HTTP peut être intercepté et éventuellement détourné, ce qui rend les informations et leurs destinataires vulnérables !

HTTPS

- HTTPS et HTTP sont « cousins »: deux protocoles de transfert hypertexte qui permettent à des données web d'être affichées sur votre écran lorsque vous envoyez une requête
- Protocole HTTPS est une extension de HTTP, le « S » à la fin est l'initiale du mot « Secure » (sécurisé)
- L'extension « secure » est assurée grâce au protocole **TLS** (Transport Layer Security, successeur du protocole **SSL** (Secure Sockets Layer), la technologie de sécurité standard pour établir une **connexion chiffrée entre un serveur web et un navigateur.**

Protocole HTTP

Intérêts

- Sans HTTPS, toutes les données que vous envoyez sur un site sont « en clair » (ex : nom d'utilisateur, mot de passe, carte bancaire, RIB, etc), donc vulnérables aux interceptions et à l'espionnage
- HTTPS vous garantit l'identité du serveur pendant le transfert
- HTTPS garantit la non altération des échanges de données

A venir

- Demande croissante du public sur le HTTPS: les visiteurs ne « consomment » plus sur des sites HTTP
- HTTPS est désormais considéré comme un facteur de référencement naturel. Moins de visibilité pour les sites ne le proposant pas
- Les navigateurs se joignent également à l'effort: nombreux messages d'avertissement lorsque le site n'est pas sécurisé !
- Demain: navigation sera impossible sans HTTPS

Protocole HTTP

Exemples

Site sans HTTPS:

?

Site HTTPS:

https://www-actus.univ-ubs.fr/fr

Merci !

Langages Web

- Définitions
- Langage HTML
- CSS
- Divers

Langages WEB

Définitions

- **Langages informatiques** qui permettent de créer des sites/pages web
- **Tous les sites web sont basés sur ces langages**, ils sont incontournables et universels aujourd'hui.
- L'évolution du Web est géré par le World Wide Web Consortium (W3C), qui définit les nouvelles versions des langages liés au Web

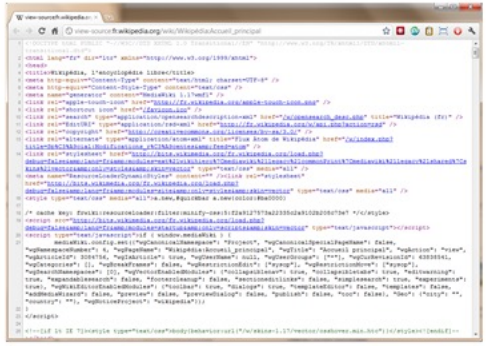
https://www.w3.org/

Basiques

- Uniquement du code informatique au format « texte », donc, humainement lisible (et modifiable !)
- Le navigateur web fait la traduction entre ces langages informatiques et créé la magie qui va s'afficher à l'écran

Langages WEB

Langages
HTML et CSS



Traduction par l'ordinateur

Résultat visible
à l'écran

Du HTML à l'écran

Expériences

- Via le « debug tools », modifier le rendu d'un site

https://www-actus.univ-ubs.fr/fr/index.html

Langages WEB

HTML

- **HTML :** *HyperText Markup Language*
- Son rôle est de gérer et organiser le contenu
- Permet d'afficher du texte, des liens, des images
- Langage de « balises »
- Version 5 à date

CSS

- **CSS** (*Cascading Style Sheets*, aussi appelées *feuilles de style*)
- Son rôle est de gérer l'apparence de la page web (agencement, positionnement, décoration, couleurs, taille du texte…)
- Ce langage est venu compléter le HTML en 1996
- Version 3 à date

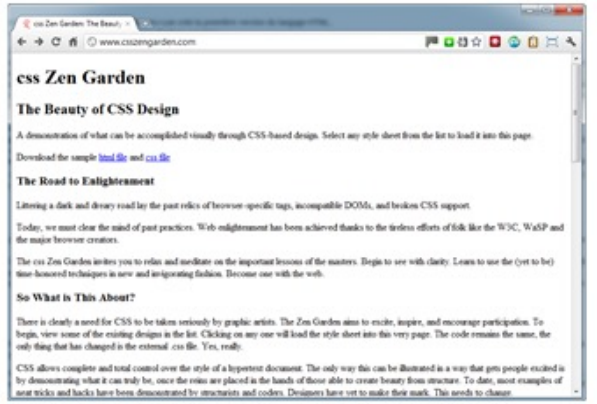
Langages WEB

Avec ou sans CSS ?

- Le minimum pour une page WEB: le HTML, mais pas de « cosmétique »
- Le CSS apporte grandement dans l'esthétique et simplifie les rendus en industrialisant les styles (cf outils de bureautique)

https://www.csszengarden.com/

HTML
(pas de CSS)



HTML + CSS

Avec et sans CSS

Langages WEB

Historique HTML

- **HTML 1** : première version créée par Tim Berners-Lee en 1991
- **HTML 2** : 1994 à 1996. Pose les bases des versions suivantes du HTML. Les règles et le fonctionnement de cette version sont donnés par le W3C
- **HTML 3** : 1996, ajout de nombreuses possibilités au langage, comme les tableaux, les applets, les scripts, le positionnement du texte autour des images, etc
- **HTML 4** : 1998: arrivée des frames (qui découpent une page web en plusieurs parties), des tableaux plus complexes, des améliorations sur les formulaires, etc. Lancement du CSS !

HTML 5 : Dernière version. Nombreuses améliorations, comme la possibilité d'inclure facilement des vidéos, un meilleur agencement du contenu, de nouvelles fonctionnalités pour les formulaires, etc

Langages WEB

Historique CSS

- **CSS 1** : dès 1996, première version du CSS. Pose les bases du langage pour travailler la page sur les couleurs, les marges, les polices de caractères, etc
- **CSS 2** : apparue en 1999 puis complétée par CSS 2.1. Ajoute de nombreuses options. Techniques de positionnement très précises notamment
- **CSS 3** : Dernière version, avec des fonctionnalités particulièrement attendues comme les bordures arrondies, les dégradés, les ombres, etc

Attention !

HTML5 et CSS3 ne sont pas **encore des versions "officiellement" finalisées par le W3C**... mais les modifications seront mineures, et les navigateurs sont compatibles avec la plupart des nouvelles fonctionnalités

Langages WEB

Construction de pages WEB

- Via deux approches:
  - Le **WYSIWYG** (*What You See Is What You Get* – "ce que vous voyez est ce que vous obtenez") : programmes très faciles d'emploi. Ils permettent de créer des sites web sans apprendre de langage particulier… mais attention à la « boite noire »
  - les **éditeurs de texte** : ce sont des programmes dédiés à l'écriture de code. On peut en général les utiliser pour de multiples langages, pas seulement HTML et CSS.

Un outil ?

- Sublime Texte : simple, épuré et facile à lire dès le départ… mais bien d'autres encore: UltraEdit, NotePad++, etc
- Le bloc note peut suffire… mais attention aux yeux!

Langages WEB

Importance du navigateur

- Le navigateur est le programme qui nous permet de voir les sites web!
- Le principal problème: *les différents navigateurs n'affichent pas le même site exactement de la même façon* ! Vérifier régulièrement que votre site fonctionne correctement sur la plupart des navigateurs
- Pour compliquer les choses, **plusieurs versions des navigateurs coexistent**. Aujourd'hui, un navigateur comme Chrome sort une nouvelle version presque tous les mois !
- Existence de variantes de ces navigateurs conçues pour les téléphones portables, en particulier pour les **smartphones**
- **Un vrai casse-tête !!**
- Pour aider: https://caniuse.com/

Langages WEB - HTML

HTML, premiers pas

- En HTML, on utilise des **balises**.
- Les balises se repèrent facilement. Elles sont entourées de « chevrons », c'est-à-dire des symboles < et > , comme ceci : **<balise>**
- Elles indiquent la nature du texte qu'elles encadrent. « Ceci est le titre de la page », « Ceci est une image », « Ceci est un paragraphe de texte », etc
- On distingue deux types de balises : les balises en paires et les balises orphelines.

Les balises en paires

- Elles s'ouvrent, contiennent du texte (voire d'autres balises), et se ferment plus loin.

<titre>Ceci est un titre</titre>

- On distingue une balise ouvrante (<titre>) et une balise fermante (</titre>) qui indique ici que le titre se termine

Ceci n'est pas un titre <titre>Ceci est un titre</titre>
Ceci n'est pas un titre

Langages WEB - HTML

Les balises orphelines

- Des balises qui servent à insérer un élément à un endroit précis (par exemple une image). Il n'est pas nécessaire de délimiter le début et la fin de l'image
- Une balise orpheline s'écrit comme ceci :

```
<image />
```

Les attributs

- Les « options » des balises. Compléter pour donner des informations supplémentaires. L'attribut se place après le nom de la balise ouvrante et a le plus souvent une valeur, comme ceci :

```
<balise attribut="valeur">
```

- À quoi cela sert-il ? Prenons la balise <image />. Seule, elle ne sert pas à grand-chose. On pourrait rajouter un attribut qui indique le nom de l'image à afficher :

```
<image url="urlimage/photo.jpg" />
```

L'ordinateur comprend alors qu'il doit afficher l'image contenue dans le fichier photo.jpg .

Langages WEB - HTML

Attributs (suite)

- Balise « par paire »: attributs uniquement dans la balise ouvrante
- Autre exemple:

<citation auteur="Neil Armstrong" date="21/07/1969"> C'est un petit pas pour l'homme, mais un bond de géant pour l'humanité. </citation>

- La citation est de Neil Armstrong et et elle date du 21 juillet 1969 :

Toutes les balises « officielles »

https://www.w3schools.com/tags/default.asp

Langages WEB - HTML

Première page WEB

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>Titre</title>
    </head>

    <body>

    </body>
</html>
```

Langages WEB - HTML

Remarques première page WEB

- Les balises s'ouvrent et se ferment dans un ordre précis. Par exemple, la balise <html> est la première que l'on ouvre et c'est aussi la dernière que l'on ferme (tout à la fin du code, avec </html>). *Les balises doivent être fermées dans le sens inverse de leur ouverture.*
- Un exemple :

<html><body></body></html> : **correct**. Une balise qui est ouverte à l'intérieur d'une autre doit aussi être fermée à l'intérieur.

<html><body></html></body> : **incorrect**, les balises s'entremêlent.

- Présence d'espaces au début de certaines lignes pour « décaler » les balises. On appelle cela l'***indentation.*** Non obligatoire, aucun impact sur l'affichage de la page. Permet une meilleure lisibilité

Langages WEB - HTML

Première page WEB

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>Titre</title>
    </head>

    <body>

    </body>
</html>
```

Langages WEB - HTML

Doctype

- Toute première ligne! Indispensable car indique qu'il s'agit bien d'une page web HTML. Balise à part et unique !

- Incroyablement complexe par le passé. Exemple HTML1 :
  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

- Depuis HTML5, simplification: <!DOCTYPE html> (attention, page forcément en HTML5 !)

Langages WEB - HTML

Première page WEB

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>Titre</title>
    </head>

    <body>

    </body>
</html>
```

Langages WEB - HTML

Balise HTML

- **En « paire »:**

  <html> </html>

- Balise principale du code. Englobe tout le contenu de votre page.

Langages WEB - HTML

Première page WEB

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>Titre</title>
    </head>

    <body>

    </body>
</html>
```

Langages WEB - HTML

Balise « head »

- **En « paire »:**

  <head> </head>

- Cette section donne quelques informations générales sur la page: titre, l'encodage
- Les informations que contient l'en-tête ne sont pas affichées sur la page, ce sont simplement des informations générales à destination de l'ordinateur, moteur de recherche, etc

Balise « body »

- **En « paire »:**

  <body> </body>

- C'est là que se trouve la partie principale de la page. Tout ce que nous écrirons ici sera affiché à l'écran.
- Pour le moment, le corps est vide

Langages WEB - HTML

Première page WEB

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>Titre</title>
    </head>

    <body>

    </body>
</html>
```

Langages WEB - HTML

Balise « meta »

- **Apporter des informations sur la page**
- Encodage: <meta charset="utf-8" /> : indique la façon dont le fichier est enregistré et comment les caractères spéciaux vont s'afficher (accents, idéogrammes chinois et japonais, caractères arabes, etc)
- Langue : <meta content="fr" name="language" />
- Etc

Balise « title »

- C'est le titre de votre page, probablement l'élément le plus important ! Toute page doit avoir un titre qui décrit ce qu'elle contient
- Il est conseillé de garder le titre assez court (moins de 100 caractères, en général)
- Le titre ne s'affiche pas dans votre page mais en haut de celle-ci (souvent dans l'onglet du navigateur)

Langages WEB - HTML

Commentaires

- Un commentaire est une balise HTML avec une forme bien spéciale :
- <!-- Ceci est un commentaire -->
- Vous pouvez le mettre où vous voulez au sein de votre code source
- Aucun impact sur votre page
- Attention ! Visible de tous !

Langages WEB - HTML

De façon concrète:

En regardant le code source des pages (clique droit ou F11) :

- https://www.qwant.com/?l=fr
- https://fr.wikipedia.org/wiki/Wikip%C3%A9dia:Accueil_principal

A bien noter

Tout le monde peut voir vos commentaires et tout votre code HTML !

Langages WEB - HTML

Paragraphes

- Le langage HTML propose la balise pour délimiter les paragraphes.

Bonjour et bienvenue sur mon site !

- signifie « Début du paragraphe », signifie « Fin du paragraphe »
- A écrire entre les balises <body></body>

Saut de lignes

- Taper frénétiquement sur la touche Entrée ne sert strictement à rien ! Il faut :
  - Un deuxième paragraphe: une deuxième balise
  - Utiliser une balise **orpheline** qui sert juste à indiquer qu'on doit aller à la ligne :

Langages WEB - HTML

Les titres (corps de la page)

- Avec de nombreux paragraphes, les titres deviennent utiles !
- Six niveaux de titres différents.
  - <h1> mon titre </h1> : signifie « titre très important ». En général, on s'en sert pour afficher le titre de la page au début de celle-ci
  - <h2> mon titre </h2> : signifie « titre important »
  - <h3> mon titre </h3> : pareil, c'est un titre un peu moins important (on peut dire un « sous-titre », si vous voulez)
  - <h4> mon titre </h4> : titre encore moins important
  - <h5> mon titre </h5> : titre pas important
  - <h6> mon titre </h6> : titre vraiment, mais alors là, vraiment pas important du tout ! Très peu utilisé
- Les moteurs de recherche tiennent compte de la bonne utilisation de cette hiérarchie !

Langages WEB - HTML

La mise en valeur

- Au sein de vos paragraphes, certains mots sont parfois plus importants que d'autres et vous aimeriez les faire ressortir.
- Pour mettre *un peu* en valeur votre texte, vous devez utiliser la balise

   le mot

Alternative forte

- Pour mettre un texte bien en valeur, balise :

  c'est très visible
- <mark> propose aussi cela

Langages WEB - HTML

Attention !

- **HTML pour le fond, CSS pour la forme**
- Les balises , , <mark> … servent à indiquer le *sens* du texte, donc, aide au référencement
- La plupart des navigateurs affichent les textes importants en gras, mais rien ne les y oblige
- Il est nécessaire de travailler sur le CSS pour faire le rendu souhaité !

Langages WEB - HTML

Les listes

- Deux types de listes : les listes non ordonnées ou *listes à puces* ET les listes ordonnées ou *listes numérotées*, ou encore *énumérations*.
- **Liste non ordonnée.** Une liste non ordonnée ressemble à ceci :
  - Fraises
  - Framboises
  - Cerises
- Il suffit d'utiliser la balise <ul> que l'on referme un peu plus loin avec . Chaque item est encadré par la balise

Langages WEB - HTML

Code HTML

```
<ul>
  <li>Fraises</li>
  <li>Framboises</li>
  <li>Cerises</li>
</ul>
```

Langages WEB - HTML

Créer des liens

- Pour faire un lien, la balise à utiliser est <a>. L'ajout d'un attribut, « href », et obligatoire pour indiquer vers quelle page le lien doit conduire.
- Exemple du code pour un lien qui amène vers le Site du Zéro, situé à l'adresse https://www.siteduzero.com :

```
<a href="https://www.siteduzero.com">Site du Zéro</a>
```

- Plaçons ce lien au sein d'un paragraphe :

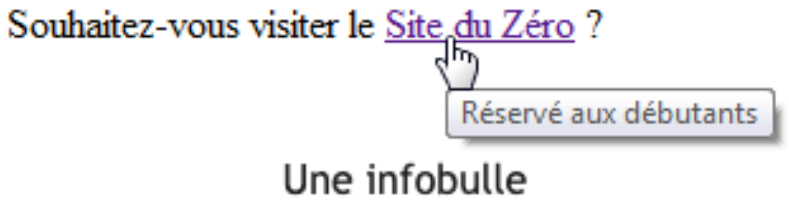
```
<p>Bonjour. Souhaitez-vous visiter le <a href="https://www.siteduzero.com">site du Zéro</a> ? </p>
```

Langages WEB - HTML

Compléments liens: infobulle

- Pour un lien qui affiche une infobulle au survol, il faut utiliser l'attribut « title »
- Cet attribut est facultatif

Bonjour. Souhaitez-vous visiter le Site du Zéro ?



Une infobulle

Langages WEB - HTML

Compléments liens: nouvelle fenêtre

- Il est possible de « forcer » l'ouverture d'un lien dans une nouvelle fenêtre
- Utilisation de l'attribut target="_blank" pour la balise <a>

```
<p>Bonjour. Souhaitez-vous visiter le <a href="http://www.siteduzero.com" target="_blank">Site du Zéro</a> ?<br /></p>
```

- Selon la configuration du navigateur, la page s'affichera dans une nouvelle fenêtre ou un nouvel onglet
- Attention: perturbant pour la navigation de l'internaute !

Langages WEB - HTML

Compléments liens: télécharger un fichier

- Identique à un lien vers une page, simplement avec le nom du fichier à télécharger l'url de l'attribut « href »

```
<p><a href="monfichier.zip">Télécharger le fichier</a></p>
```

Langages WEB - HTML

Image – aparté (1)

- Il existe deux types de format d'image:
  - **Raster** : le fichier stocke des informations sur les pixels. On applique une couleur à chaque pixel.
  - **Vectoriel** : les données sont stockées sous la forme d'opérations géométriques avec des points, lignes et courbes.
- Un fichier image a plusieurs caractéristiques, i.e. formats :
  - la **définition** d'une image ou **résolution** : c'est le nombre de pixels qui la composent. Plus y en a, plus l'image sera net. Elle s'exprime
    - **pixel par pouce** (exprimé en ppp et ppi en anglais). C'est le nombre de pixels par unité de longueur soit donc la densité de pixels.
    - La taille en pixels, le nombre de pixel en longueur sur la largeur. Ex : 1980x1800.
  - La **profondeur de la couleur**. Il indique le nombre de bits utilisés pour représenter la couleur d'un pixel dans une image (**bpp**). Par ex : 24/32 bits (16bits : 65536 couleurs)
  - Le **taux de compression**. Si le format de l'image gère la compression, on peut appliquer un algorithme de compression pour réduire le taille du fichier image.

Langages WEB - HTML

Image – aparté (2)

- Dans le monde du WEB, il faut:
  - Adapter l'image à la situation: pas de fichier haut résolution pour des pictogrammes !
  - Privilégier les formats JPG, PNG et récemment, les formats vectoriels

Langages WEB - HTML

Afficher des images

- Via une balise de type orpheline: **« img ». Objectif:** insérer une image à un endroit précis

- La balise doit être accompagnée de deux attributs obligatoires :
  - **src** : indiquer où se trouve l'image que l'on veut insérer. Soit en chemin absolu (ex. : https://www.site.com/fleur.png), soit en chemin en relatif (ex.: images/fleur.png si votre image est dans un sous-dossier « images
  - **alt** : signifie « texte alternatif ». Court texte qui décrit ce que contient l'image, très utile pour l'accessibilité (non-voyants). Utiles aussi aux robots des moteurs de recherche

* Voici une photo que j'ai prise lors de mes dernières vacances à la montagne :
  *

- Eviter à tout prix les accents, majuscules et espaces dans les noms de fichiers et dossiers

Langages WEB - HTML

Formulaires (1/9)

- Fonctionne avec la balise <form> </form>:

```
<p>Texte avant le formulaire</p>
<form>
  <p>Texte à l'intérieur du formulaire</p>
</form>
<p>Texte après le formulaire</p>
```

- Deux problèmes:
  - **Problème n°1** : comment envoyer le texte saisi par l'internaute?
  - **Problème n°2** : comment les traiter ?

Langages WEB - HTML

Formulaires (2/9)

- Deux attributs à la balise <form> pour répondre :
  - **method** : cet attribut indique par quel moyen les données vont être envoyées (réponse au **problème n°1**). Il existe deux solutions pour envoyer des données sur le Web :
    - method="get" : méthode limitée à 255 caractères. Informations reportées dans l'url. A proscrire
    - method="post" méthode la plus utilisée pour les formulaires. Compatible avec un grand nombres d'informations (envoi de fichiers par exemple).
  - **action** : adresse/url de la page ou du programme qui va **traiter** les informations (réponse au **problème n°2**). Cela se fait avec un autre langage. Exemple: PHP, javascript, Python, etc.

Langages WEB - HTML

Formulaires (3/9)

Exemple de code HTML pour un formulaire « vide »:

```
<p>Texte avant le formulaire</p>

<form method="post" action="traitement.php">
  <p>Texte à l'intérieur du formulaire</p>
</form>

<p>Texte après le formulaire</p>
```

https://www-actus.univ-ubs.fr/fr/index/actualites/dseg/soiree-anniversaire-10-ans/formulaire-10-ans.html

Langages WEB - HTML

Formulaires (4/9)

Zone de texte mono-ligne (saisie):

- Utiliser la balise <input /> :

  <input type="text" />

- Donner un nom à votre zone de texte via l'attribut « **name** »:

  <input type="text" name="pseudo" />

Nom *

https://www-actus.univ-ubs.fr/fr/index/actualites/dseg/soiree-anniversaire-10-ans/formulaire-10-ans.html

Langages WEB - HTML

Formulaires (5/9)

Les libellés:

- Indiquer à l'utilisateur ce qu'il doit saisir
- Via la balise **« <label> »**

```
<form method="post" action="traitement.php">
  <p>
    <label>Votre pseudo</label> : <input type="text" name="pseudo" />
  </p>
</form>
```

https://www-actus.univ-ubs.fr/fr/index/actualites/dseg/soiree-anniversaire-10-ans/formulaire-10-ans.html

Langages WEB - HTML

Formulaires (6/9)

Les cases à cocher

- Via la balise <input />, en spécifiant cette fois le type « **checkbox** » :

  <input type="checkbox" name="choix" />

- Rajoutez un <label> bien placé, et le tour est joué !
- Pour un choix d'option, il suffit d'ajouter l'attribut « **name** » avec la même valeur pour chaque balise « **input** », et le type **« radio »**

https://www-actus.univ-ubs.fr/fr/index/actualites/dseg/soiree-anniversaire-10-ans/formulaire-10-ans.html

Formulaires (7/9)

Les listes déroulantes

- Permettre un choix parmi plusieurs possibilités. Via la balise <select> </select> qui indique le début et la fin de la liste déroulante
- A l'intérieur du <select> </select>, plusieurs balises <option> </option> (une par choix possible), chacune avec un attribut « value »
- Exemple:

```
<label for="pays">Dans quel pays habitez-vous ?</label><br />
    <select name="pays" id="pays">
      <option value="france">France</option>
      <option value="espagne">Espagne</option>
      <option value="italie">Italie</option>
    </select>
```

Formulaires (8/9)

Le bouton d'envoi

- Elément indispensable via la balise <input /> en quatre versions :
  - type="submit" : le principal bouton d'envoi de formulaire
  - type="reset" : remise à zéro du formulaire.
  - type="image" : équivalent du bouton submit, présenté cette fois sous forme d'image. Rajoutez l'attribut src pour indiquer l'URL de l'image.
  - type="button" : bouton générique, qui n'aura (par défaut) aucun effet. Utile pour des processus plus complexe
- On peut changer le texte affiché à l'intérieur des boutons avec l'attribut « **value** »
- Exemple :

<input type="submit" value="Envoyer" />

Langages WEB - HTML

Formulaires (9/9)

Vous avez dit « captcha » ?

- Protection **indispensable** pour vos formulaires publiques
- Permet d'éviter l'envoi de données via des robots !
- Nombreuses solutions gratuites et libres de droits (mais attention, qui est le produit ?!)

Merci !

Langage WEB - CSS

CSS

Rappel: il vous permet de choisir la couleur de votre texte, de sélectionner la police utilisée sur votre site, définir la taille du texte, les bordures, le fond…

On peut écrire du code en langage CSS à trois endroits différents :

- dans un fichier .css (*méthode la plus recommandée*)
- dans l'en-tête <head> du fichier HTML via la balise **« <style> »**
- directement dans les balises du fichier HTML *via* un attribut **« style »** (*méthode la moins recommandée*).

Langage WEB - CSS

Dans un fichier .css

Dans la balise « head », nous ajoutons :

<link rel="**stylesheet**" href**="fichier_style.css**" />

Dans le fichier, nous insérons :

```
p {
   color: blue;
}
```

Langage WEB - CSS

Dans une balise <style>

Dans la balise « head », nous ajoutons :

```
<style type="text/css">
p {
    color: blue;
}
</style>
```

Langage WEB - CSS

Via l’attribut « style »

```
<p style="color: blue;">Bonjour et bienvenue sur mon site !</p>
```

Langage WEB - CSS

Appliquer un style: sélectionner une balise !

Le code CSS se décompose en 3 informations:

```
p{color: blue;}
```

- Des noms de balises : on écrit les noms des balises dont on veut modifier l'apparence. Ici, si je veux modifier l'apparence de tous les paragraphes <p>
- Des propriétés CSS : les « effets de style » de la page sont rangés dans des propriétés. Exemple ici : la propriété color qui permet d'indiquer la couleur du texte. D'autres propriétés: font-size (taille), font-weight (le poids), etc.
- Les valeurs : pour chaque propriété CSS, on doit indiquer une valeur. Par exemple ici: « blue ».

Schématiquement, une feuille de style CSS ressemble donc à cela :

```
balise1
{
   propriete1: valeur1;
   propriete2: valeur2;
   propriete3: valeur3;
}

 balise2…
```

Langage WEB - CSS

Appliquer un style « ciblé »

- Idée : appliquer un style à certaines balises. Exemple: une partie des paragraphes
- Solution: usage des attributs « class » et « id »
- Exemple:

```
<h1 class="titre_rouge"> </h1>
```

- Dans mon CSS, j'aurai alors:

```
.titre_rouge {
    color: red;
}
```

Je peux ainsi choisir les balises qui disposeront de l'effet « titre rouge »

Langage WEB - CSS

Appliquer un style « ciblé »

- Via l'attribut « id », cela donne:

```
<h1 id="titre_rouge"> </h1>
```

- Dans mon CSS, j'aurai alors:

```
#titre_rouge {
    color: red;
}
```

- Attention, les « id » doivent être unique !
- Il est possible de reprendre le nom de la balise dans le CSS pour plus de précision (en id ou class):

```
H1#titre_rouge {
    color: red;
}
```

Langage WEB

Quelques liens pour approfondir:

- Liste des propriétés de styles les plus répandue:

https://yard.onl/sitelycee/cours/html5css3/_debut.html?Unlienversuneancre.html

- Le cours dont sont inspirés les slides :

https://yard.onl/sitelycee/cours/html5css3/_debut.html?Unlienversuneancre.html

- Mémento des balises HTML:

https://yard.onl/sitelycee/cours/html5css3/Mmento.html

- Quelques bons exemples :

https://www.w3schools.com

https://www.alsacreations.com/tutoriels

Langage WEB

En synthèse :

- HTML: la structure de la page, son contenu. Statique.
- CSS: la mise en forme
- Un autre langage existe: le **javascript**, qui permet d'effectuer des traitements dans la page:

https://www.w3schools.com/js/DEFAULT.asp

- « **Responsive** »: technique CSS pour optimiser le rendu de vos pages selon le terminal (smartphone, PC fixe, tablette, etc)
- **Les langages comme Python ou PHP permettent de rendre « dynamique » votre code HTML**

Règlements WEB

RGAA

- Attention à l'accessibilité de vos pages: contraste, couleurs, images et alternatives textuelles, etc
- Toutes les règles:

https://www.numerique.gouv.fr/publications/rgaa-accessibilite/

- Un outil:

https://wave.webaim.org

RGPD, via la CNIL

- Les formulaires peuvent contenir des saisies de données personnelles
- Vos pages WEB peuvent stocker des informations pour profiler le comportement des vos utilisateurs
- Etc

Il existe des règles à respecter pour ces situations !

https://www.cnil.fr/fr/rgpd-de-quoi-parle-t-on

Eco-conception

Numérique responsable, être attentif au:

- Poids de la page, et ses ressources (image, fichiers, etc)
- Nombre d'éléments composant la page
- Nombre de requêtes pour afficher votre page
- WebSiteCarbon / LightHouse / GreenIT / etc

La BDD

- Définitions
- SQL / NOSQL / Document
- Solutions
- Langages

Base de données

Contexte

- Nées à la fin des années **1960** pour combler les lacunes des systèmes de fichiers et faciliter la gestion qualitative et quantitative des données informatiques
- Les Systèmes de Gestion de Base de Données (**SGBD**) sont des **applications informatiques** permettant de créer et de gérer des **Bases de Données** (Oracle, PostgreSQL par exemple)
- Les **BD relationnelles** sont celles qui ont connu le plus grand essor et reste encore aujourd'hui les plus utilisées.
- Le **langage SQL** est le langage commun à tous les SGBD Relationnels
- L'accroissement de l'utilisation du numérique a engendré l'émergence de solutions conceptuelles et technologiques nouvelles, les bases de données du mouvement **NoSQL**

Base de données

Définitions (1/4)

- **base de données:** un ensemble de données numériques qui possède une structure, c'est à dire dont l'organisation répond à une logique systématique
- Par extension, représente tout ensemble de données stocké numériquement et pouvant servir à un ou plusieurs programmes: un disque dur, un fichier de tableur, voire un fichier de traitement de texte peuvent constituer des bases de données
- Fonctions fondamentales:
  - **Stocker** l'information de façon fiable (c'est à dire être capable de restituer l'information entrée dans le système)
  - **Traiter de grands volumes** de données (massification)
  - **Traiter rapidement** les données (optimisation)
  - **Sécuriser** les accès aux données (gérer les autorisations selon les utilisateurs)
  - **Contrôler** la qualité des données (par exemple la cohérence par rapport à un modèle préétabli)
  - **Partager** les données (entre plusieurs applications dédiées à plusieurs métiers)
  - Rendre accessible les données en réseau (gérer la **concurrence** des accès parallèles ou **transaction**)

Base de données

Définitions (2/4)

- **SQL:**
  - SQL est un langage d'interrogation pour la gestion des bases de données relationnelles.
  - La syntaxe et les commandes SQL originales sont définies par la norme ANSI.
  - En raison de l'existence de plusieurs types de SQL, certaines commandes SQL peuvent ne pas fonctionner sur des systèmes de bases de données spécifiques.

- **Table :** une base de données relationnelle est principalement constituée de tables (ou « relations » d'où le nom de relationnel). Une table est basiquement un élément d'organisation de l'information constitué de colonnes (ou attributs) et de lignes (ou enregistrements)

Exemple: des livres sont représentés par des enregistrements dans une table, avec pour colonne « titre », « date de parution », etc.

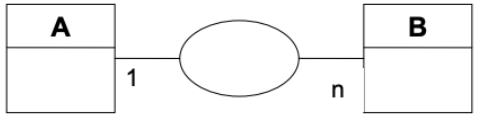
- **Instruction:** enchainement de mots clés permettant d'interagir avec la BDD

Définitions (3/4)

- Lien « **fonctionnel** » ou « **association** » : permet de savoir si une table peut être associée à une autre table

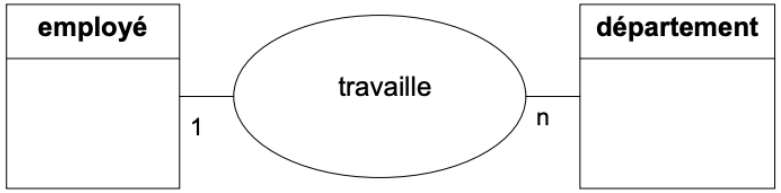
Exemple : un employé ne peut travailler que dans un seul département, un département a plusieurs employés

Lien fonctionnel 1:n



Une instance de A ne peut être associée qu'à une seule instance de B

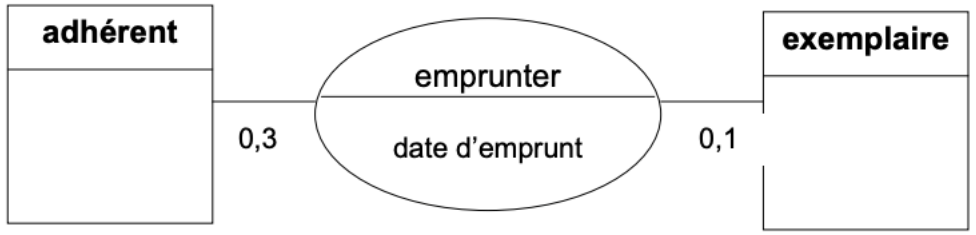
Par exemple :



Base de données

Définitions (4/4)

- **Cardinalité**: c'est le nombre d'associations possibles entre les enregistrements des deux tables
- « **Clé primaire**»: une information (i.e., attribut ou colonne de la table) uniquement permettant d'identifier un enregistrement. Exemple: le numéro d'adhérent



- La cardinalité 0,3 indique qu'un adhérent peut être associé à 0, 1, 2 ou 3 livres, c'est à dire qu'il peut emprunter au maximun 3 livres.
- A l'inverse un livre peut être emprunté par un seul adhérent, ou peut ne pas être emprunté.

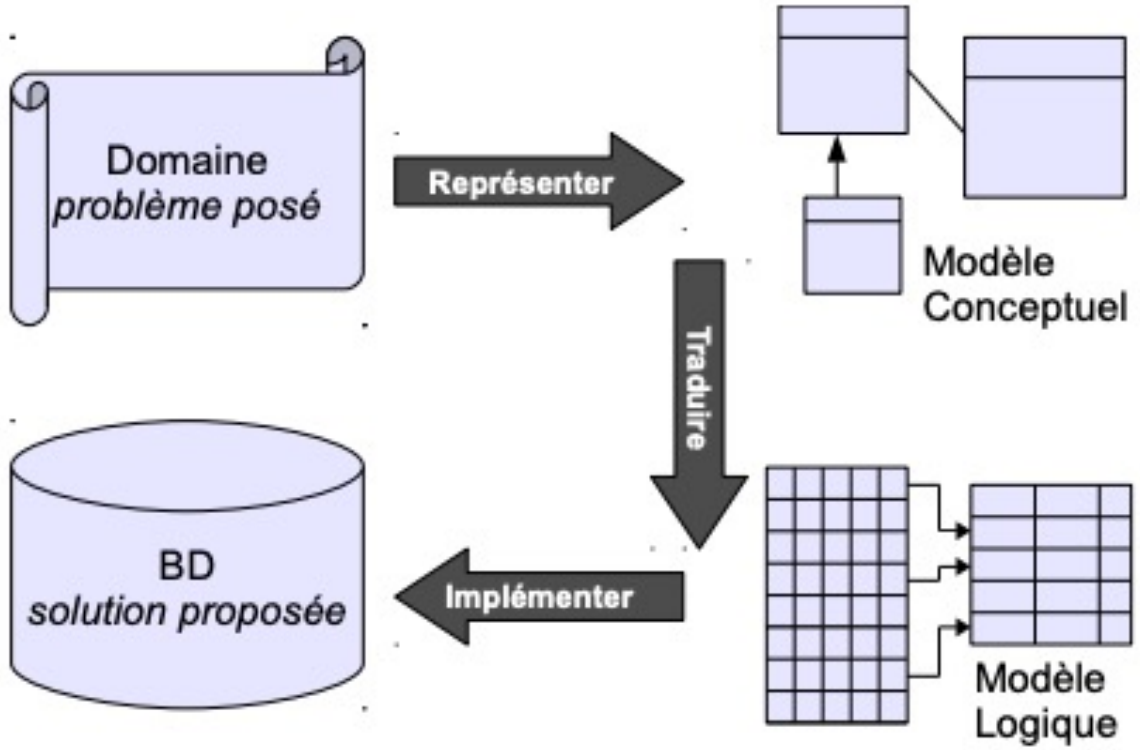
Base de données

Conception

Quatre étapes dans la conception d'une base de données :

- **L'analyse**: elle consiste à étudier le problème, consigner les besoins, les choix, les contraintes
- **La modélisation conceptuelle**: elle permet de décrire le problème posé, de façon non-formelle (en générale graphique), en prenant des hypothèses de simplification
- **La modélisation logique:** elle permet de décrire une solution, en prenant une orientation informatique générale: SQL ? NoSQL ? Fichier ? etc.
- **L'implémentation:** elle correspond aux choix techniques et à leur mise en œuvre (programmation, optimisation...)

Base de données



Base de données

Création de tables

- La commande CREATE TABLE permet de créer une table en SQL. La création d'une table sert à définir les colonnes et le type de données qui seront contenus dans chacune des colonne (entier, chaîne de caractères, date, valeur binaire …).

La syntaxe générale pour créer une table est la suivante :

```
CREATE TABLE nom_de_la_table
(
    colonne1 type_donnees,
    colonne2 type_donnees,
    colonne3 type_donnees,
    colonne4 type_donnees
)
```

Dans cette requête, 4 colonnes ont été définies. Le mot-clé « type_donnees » sera à remplacer par un mot-clé pour définir le type de données (INT, DATE, TEXT …). Pour chaque colonne, il est également possible de définir des options telles que (liste non-exhaustive) :

- **NOT NULL :** empêche d'enregistrer une valeur nulle pour une colonne.
- **DEFAULT :** attribuer une valeur par défaut si aucune données n'est indiquée pour cette colonne lors de l'ajout d'une ligne dans la table.
- **PRIMARY KEY :** indiquer si cette colonne est considérée comme clé primaire pour un index.

Base de données

Création de tables: exemple

- **id** : identifiant unique qui est utilisé comme clé primaire et qui n'est pas nulle
- **nom** : nom de l'utilisateur dans une colonne de type VARCHAR avec un maximum de 100 caractères
- **prenom** : idem mais pour le prénom
- **email** : adresse email enregistré sous 255 caractères au maximum
- **date_naissance** : date de naissance enregistré au format AAAA-MM-JJ (exemple : 1973-11- 17)
- **pays** : nom du pays de l'utilisateur sous 255 caractères au maximum
- **ville** : idem pour la ville
- **code_postal** : 5 caractères du code postal
- **nombre_achat** : nombre d'achat de cet utilisateur sur le site

```
CREATE TABLE utilisateur
(
    id INT PRIMARY KEY NOT NULL,
    nom VARCHAR(100),
    prenom VARCHAR(100),
    email VARCHAR(255),
    date_naissance DATE,
    pays VARCHAR(255),
    ville VARCHAR(255),
    code_postal VARCHAR(5),
    nombre_achat INT
)
```

Base de données

Sélection de données

- L'utilisation la plus courante de SQL consiste à lire des données issues de la base de données. Cela s'effectue grâce à la commande SELECT, qui retourne des enregistrements dans un tableau de résultats. Cette commande peut sélectionner une ou plusieurs colonnes d'une table.

Commande basique

L'utilisation basique de cette commande s'effectue de la manière suivante :

```
SELECT nom_du_champ
FROM nom_du_tableau
```

Cette requête va sélectionner (SELECT) le champ « nom_du_champ » provenant (FROM) du tableau appelé « nom_du_tableau ».

Base de données

Sélection de données (1/3)

Imaginons une base de données appelée « client » qui contient des informations sur les clients d'une entreprise.

Table « client » :

identifiant	prenom	nom	ville
1	Pierre	Dupond	Paris
2	Sabrina	Durand	Nantes
3	Julien	Martin	Lyon
4	David	Bernard	Marseille
5	Marie	Leroy	Grenoble

Si l'ont veut avoir la liste de toutes les villes des clients, il suffit d'effectuer la requête suivante :

```
SELECT ville
FROM client
```

Résultat :

ville
Paris
Nantes
Lyon
Marseille
Grenoble

Sélection de données (2/3)

Obtenir plusieurs colonnes

Avec la même table client il est possible de lire plusieurs colonnes à la fois. Il suffit tout simplement de séparer les noms des champs souhaités par une virgule. Pour obtenir les prénoms et les noms des clients il faut alors faire la requête suivante:

```
SELECT prenom, nom
FROM client
```

Résultat :

prenom	nom
Pierre	Dupond
Sabrina	Durand
Julien	Martin
David	Bernard
Marie	Leroy

Sélection de données (3/3)

Obtenir toutes les colonnes d'un tableau

Il est possible de retourner automatiquement toutes les colonnes d'un tableau sans avoir à connaître le nom de toutes les colonnes. Au lieu de lister toutes les colonnes, il faut simplement utiliser le caractère « * » (étoile). C'est un joker qui permet de sélectionner toutes les colonnes. Il s'utilise de la manière suivante :

```
SELECT * FROM client
```

Cette requête retourne exactement les mêmes colonnes qu'il y a dans la base de données. Dans notre cas, le résultat sera donc :

identifiant	prenom	nom	ville
1	Pierre	Dupond	Paris
2	Sabrina	Durand	Nantes
3	Julien	Martin	Lyon
4	David	Bernard	Marseille
5	Marie	Leroy	Grenoble

Base de données

Sélection de données - Compléments

SELECT est une commande relativement commune. Il existe plusieurs clauses qui permettent de mieux gérer les données que l'on souhaite lire:

- Joindre un autre tableau aux résultats
- Filtrer pour ne sélectionner que certains enregistrements
- Classer les résultats
- Grouper les résultats pour faire uniquement des statistiques (note moyenne, prix le plus élevé …)

Sélection de données - Compléments

Une requête SELECT peut devenir assez longue. Voici toutes les instructions possibles :

```
SELECT *
FROM table
WHERE condition
GROUP BY expression
HAVING condition
{ UNION | INTERSECT | EXCEPT }
ORDER BY expression
LIMIT count
OFFSET start
```

Sélection de données – Clause WHERE

La commande WHERE dans une requête SQL permet d'extraire les lignes d'une base de données qui respectent une condition. Cela permet d'obtenir uniquement les informations désirées.

Syntaxe

La commande WHERE s'utilise en complément à une requête utilisant SELECT. La façon la plus simple de l'utiliser est la suivante :

```
SELECT nom_colonnes
FROM nom_table
WHERE condition
```

Sélection de données – Clause WHERE

Exemple

Imaginons une base de données appelée « client » qui contient le nom des clients, le nombre de commandes qu'ils ont effectués et leur ville :

id	nom	nbr_commande	ville
1	Paul	3	paris
2	Maurice	0	rennes
3	Joséphine	1	toulouse
4	Gérard	7	paris

Pour obtenir seulement la liste des clients qui habitent à Paris, il faut effectuer la requête suivante :

```
SELECT *
FROM client
WHERE ville = 'paris'
```

Résultat :

id	nom	nbr_commande	nbr_commande
1	Paul	3	paris
4	Gérard	7	paris

Base de données

Sélection de données – Opérateurs

Il existe plusieurs opérateurs de comparaison:

Opérateur	Description
=	Égale
<>	Pas égale
!=	Pas égale
>	Supérieur à
<	Inférieur à
>=	Supérieur ou égale à
<=	Inférieur ou égale à
IN	Liste de plusieurs valeurs possibles
BETWEEN	Valeur comprise dans un intervalle donnée (utile pour les nombres ou dates)
LIKE	Recherche en spécifiant le début, milieu ou fin d'un mot.
IS NULL	Valeur est nulle
IS NOT NULL	Valeur n'est pas nulle

Sélection de données – Opérateurs logiques

Une requête SQL peut être restreinte à l'aide de la condition WHERE. Les opérateurs logiques AND et OR peuvent être utilisées au sein de la commande WHERE pour combiner des conditions.

L'opérateur AND permet de s'assurer que la condition1 ET la condition2 sont vrai :

```
SELECT nom_colonnes
FROM nom_table
WHERE condition1 AND condition2
```

L'opérateur OR vérifie quant à lui que la condition1 OU la condition2 est vrai :

```
SELECT nom_colonnes FROM nom_table
 WHERE condition1 OR condition2
```

Ces opérateurs peuvent être combinés à l'infini et mélangés. L'exemple ci-dessous filtre les résultats de la table « nom_table » si condition1 ET condition2 OU condition3 est vrai :

```
SELECT nom_colonnes FROM nom_table
 WHERE condition1 AND (condition2 OR condition3)
```

Attention : il faut penser à utiliser des parenthèses lorsque c'est nécessaire. Cela permet d'éviter les erreurs car et ça améliore la lecture d'une requête par un humain.

Sélection de données – Opérateurs logiques

Exemple:

Pour illustrer les prochaines commandes, nous allons considérer la table « produit » suivante :

id	nom	categorie	stock	prix
1	ordinateur	informatique	5	950
2	clavier	informatique	32	35
3	souris	informatique	16	30
4	crayon	fourniture	147	2

Opérateur AND

L'opérateur AND permet de joindre plusieurs conditions dans une requête. En gardant la même table que précédemment, pour filtrer uniquement les produits informatique qui sont presque en rupture de stock (moins de 20 produits disponible) il faut exécuter la requête suivante :

```
SELECT * FROM produit
WHERE categorie = 'informatique' AND stock < 20
```

Insertion de données – Une ligne

L'insertion de données dans une table s'effectue à l'aide de la commande INSERT INTO. Cette commande permet au choix d'inclure une seule ligne à la base existante ou plusieurs lignes d'un coup.

Pour insérer des données dans une base, il y a 2 syntaxes principales

- Insérer une ligne en indiquant les informations pour chaque colonne existante (en respectant l'ordre)
- Insérer une ligne en spécifiant les colonnes que vous souhaitez compléter. Il est possible d'insérer une ligne en renseigner seulement une partie des colonnes

Insertion de données – Une ligne

Insérer une ligne en spécifiant toutes les colonnes

La syntaxe pour remplir une ligne avec cette méthode est la suivante :

```
INSERT INTO table
VALUES ('valeur 1', 'valeur 2', ...)
```

Cette syntaxe possède les avantages et inconvénients suivants :

- Obliger de remplir toutes les données, tout en respectant l'ordre des colonnes
- Il n'y a pas le nom de colonne, donc les fautes de frappe sont limitées. Par ailleurs, les colonnes peuvent être renommées sans avoir à changer la requête
- L'ordre des colonnes doit resté identique sinon certaines valeurs prennent le risque d'être complétée dans la mauvaise colonne

Insérer une ligne en spécifiant seulement les colonnes souhaitées

Cette deuxième solution est très similaire, excepté qu'il faut indiquer le nom des colonnes avant « VALUES ». La syntaxe est la suivante :

```
INSERT INTO table
(nom_colonne_1, nom_colonne_2, ...
VALUES ('valeur 1', 'valeur 2', ...)
```

A noter : il est possible de ne pas renseigner toutes les colonnes. De plus, l'ordre des colonnes n'est pas important.

Insertion de données – Plusieurs lignes

Il est possible d'ajouter plusieurs lignes à un tableau avec une seule requête. Pour ce faire, il convient d'utiliser la syntaxe suivante :

```
INSERT INTO client (prenom, nom, ville, age)
 VALUES
 ('Rébecca', 'Armand', 'Saint-Didier-des-Bois', 24),
 ('Aimée', 'Hebert', 'Marigny-le-Châtel', 36),
 ('Marielle', 'Ribeiro', 'Maillères', 27),
 ('Hilaire', 'Savary', 'Conie-Molitard', 58);
```

Mise à jour de données

La commande UPDATE permet d'effectuer des modifications sur des lignes existantes. Très souvent cette commande est utilisée avec WHERE pour spécifier sur quelles lignes doivent porter la ou les modifications.

Syntaxe

La syntaxe basique d'une requête utilisant UPDATE est la suivante :

```
UPDATE table
SET nom_colonne_1 = 'nouvelle valeur'
WHERE condition
```

Cette syntaxe permet d'attribuer une nouvelle valeur à la colonne nom_colonne_1 pour les lignes qui respectent la condition stipulé avec WHERE. Il est aussi possible d'attribuer la même valeur à la colonne nom_colonne_1 pour toutes les lignes d'une table si la condition WHERE n'était pas utilisée.

A noter, pour spécifier en une seule fois plusieurs modification, il faut séparer les attributions de valeur par des virgules. Ainsi la syntaxe deviendrait la suivante :

```
UPDATE table
SET colonne_1 = 'valeur 1', colonne_2 = 'valeur 2', colonne_3 = 'valeur 3'
WHERE condition
```

Suppression de données

La commande DELETE en SQL permet de supprimer des lignes dans une table. En utilisant cette commande associée à WHERE il est possible de sélectionner les lignes concernées qui seront supprimées.

Attention : avant d'essayer de supprimer des lignes, il est recommandé d'effectuer une sauvegarde de la base de données, ou tout du moins de la table concernée par la suppression. Ainsi, s'il y a une mauvaise manipulation il est toujours possible de restaurer les données. Certaines BDD permettent de faire des actions d'annulation (ROLLBACK) si le mode transactionnel est disponible

Syntaxe

La syntaxe pour supprimer des lignes est la suivante :

```
DELETE FROM table
WHERE condition
```

Attention : s'il n'y a pas de condition WHERE alors toutes les lignes seront supprimées et la table sera alors vide.

Base de données - NoSQL

NoSQL ?

- Arrivée dans les années 2000 via **une remise en question du modèle relationnel par les GAFAM** brassant de plus en plus de quantités de données, volumes non compatibles avec les SGBD existants (millions de ligne à milliard de lignes !)
- Désigne une famille de systèmes de gestion de base de données (SGBD) qui s'écarte du modèle classique des bases relationnelles
- Nouveaux concepts: **données non structurées, architecture en grappe ou distribuée**
- Ces systèmes peuvent aussi gérer des données hautement structurées

Base de données - NoSQL

NoSQL

- Les quatre principaux types de bases de données NoSQL sont les suivants :

Clé-valeur

Clé-valeur stocke des paires de clés et de valeurs à l'aide d'une table de hachage. Les types clé-valeur sont particulièrement adaptés lorsqu'une clé est connue et que la valeur associée à la clé est inconnue.

Document

Les bases de données de documents étendent le concept de base de données clé-valeur en organisant des documents entiers dans des groupes appelés collections. Elles prennent en charge les paires clé-valeur imbriquées et autorisent les requêtes sur tous les attributs d'un document.

En colonnes

Les bases de données en colonnes, en colonnes larges ou en familles de colonnes stockent efficacement les données et interrogent les lignes de données éparses, et offrent la possibilité d'interroger les colonnes spécifiques d'une base de données.

Graphe

Les bases de données graphes utilisent un modèle basé sur les nœuds et les bords pour représenter les données interconnectées (relations entre membres d'un réseau social, par exemple), et offrent un stockage et une navigation facilités en présence de relations complexes.

Base de données - NoSQL

Quelques solutions

Base de données - NoSQL

Avantages

- Problème des bases de données relationnelles : **difficiles à distribuer sur plusieurs serveurs, usage du « vertical scaling »** (i.e., on augmenter les performances du serveur)
- **NoSQL utilise « l'horizontal scaling »,** une base de données peut être **distribuée** sur une "pool" de serveurs ce qui permet de mutualiser les performances. L'avantage de cette méthode est qu'elle permet d'adapter l'architecture en fonction de la charge en distribuant sur plus ou moins de serveurs suivant les cas.

Mais…

- Analogie avec les boites de vitesses: le NoSQL est l'équivalent d'une boîte de vitesse manuelle, elle permet d'obtenir de meilleure performance à condition de savoir quand et comment passer les vitesses. Les bases de données relationnelles, comme les boites automatiques permettent de ne pas avoir à se soucier de ce qui se passe derrière en automatisant les choses.

Base de données - NoSQL

Que faire ?

- Globalement, **s'orienter vers du SQL** : vous n'avez pas des milliers de serveurs pour gérer votre base de données, vous n'avez pas des millions de pages vues à la seconde
- Attention à l'idée « utiliser une base de données NoSQL pour sauvegarder rapidement des données sans avoir à réfléchir au préalable à la structure des données » ! **Si les données sont sauvegardées de manière trop anarchique, il sera très difficile de manipuler et organiser nos données par la suite**

Exemple d'écueil NoSQL

- But: stocker les commentaires de nos articles au sein d'un objet

{ "title": "Le seigneur des anneaux", "note": 4,
"reviews": [
{ "username": "John doe", "content": "Pourquoi ne pas avoir utilisé l'aigle ?" },
{ "username": "John doe", "content": "Mon précieux !" }] }

- Pratique et efficace dans un premier temps, sans avoir à faire des liaisons complexes. Par contre, si dans le futur nous souhaitons récupérer les derniers commentaires de tous les livres: KO. Notre organisation va être handicapante nous obligeant alors à repenser notre base ou créer des requêtes plus complexes et ainsi moins performantes !

Base de données – JSON

Kesako ?

- **(*JavaScript Object Notation*)** est un format d'échange de données.
- Format accepté par de **nombreux langages de programmation**, particulièrement utile pour les applications basées sur JavaScript ou API WEB.
- Permet de représenter des nombres, des booléens, des chaînes de caractères, la valeur null, des tableaux (séquences ordonnées de valeurs) et des objets (correspondances chaînes-valeurs) composés de ces valeurs (ou d'autres tableaux ou objets)
- Ne permet pas, nativement, de représenter des données plus complexes comme des fonctions, des expressions rationnelles ou des dates

Exemple

- {"widget": { "debug": "on", "window": { "title":
"Sample Konfabulator Widget",
"name": "main_window",
"width": 500,
"height": 500 },
"image": { "src": "Images/Sun.png", "name":
"sun1",
"hOffset": 250,
"vOffset": 250,
"alignment": "center" },
"text": { "data": "Click Here", "size": 36, "style":
"bold",
"name": "text1",
"hOffset": 250,
"vOffset": 100,
"alignment": "center", "onMouseUp":
"sun1.opacity = (sun1.opacity / 100) * 90;" } }}

Base de données – CSV

Kesako ?

- Un fichier CSV est un fichier texte, par opposition aux formats dits « binaires ».
- Chaque ligne du texte correspond à une ligne du tableau et les virgules correspondent aux séparations entre les colonnes.
- Les portions de texte séparées par une virgule correspondent ainsi aux contenus des cellules du tableau.

Exemple

Fichier au format .csv	Représentation tabulaire

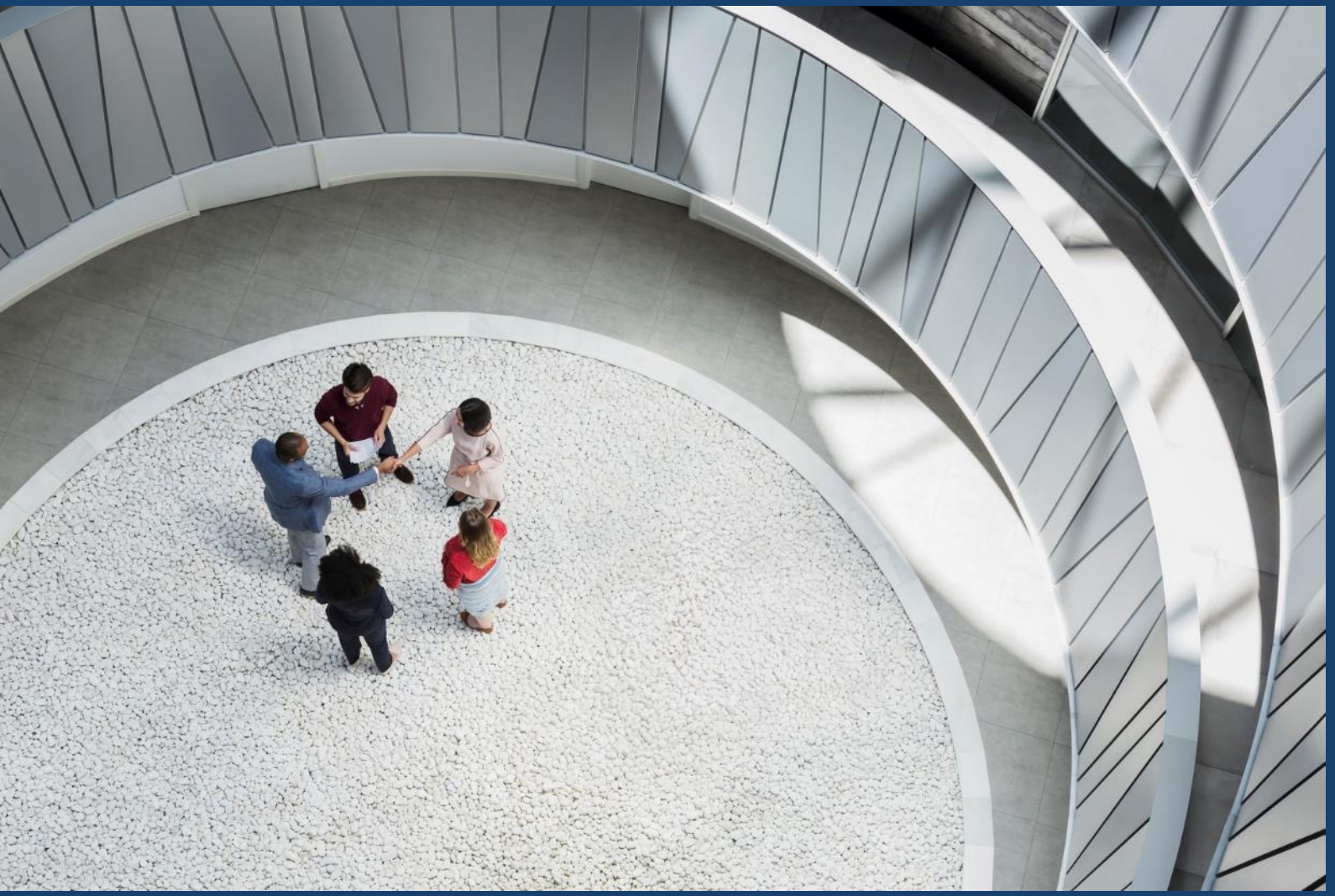
```
Sexe,Prénom,Année de naissance
M,Alphonse,1932
F,Béatrice,1964
F,Charlotte,1988
```

Sexe	Prénom	Année de naissance
M	Alphonse	1932
F	Béatrice	1964
F	Charlotte	1988

Base de données

Conclusion

- Il est important de choisir une technologie par rapport aux besoins du projet que l'on développe et non pas en fonction d'une éventuelle tendance technologique.
- La base de données constitue en général la fondation d'une application, et un mauvais choix technique peut s'avérer fatal pour l'évolution du projet.



IA

- Définitions
- Projet
- Exemples

Définitions - IA

Kesako ?

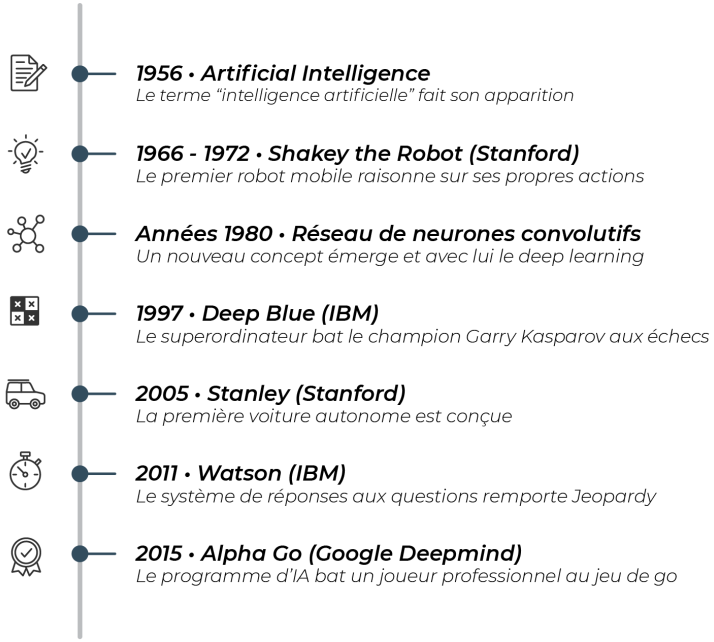
- "L’intelligence artificielle, c’est **toute technologie informatique qui permet de résoudre des problèmes complexes qu'on aurait cru réservés à l'intelligence humaine."**

  Cédric Villani

Donc en réalité, il n'y a pas "une" intelligence artificielle, il y a différentes technologies qui font partie du champ d’étude de l’intelligence artificielle.

Définitions - IA

Histoire



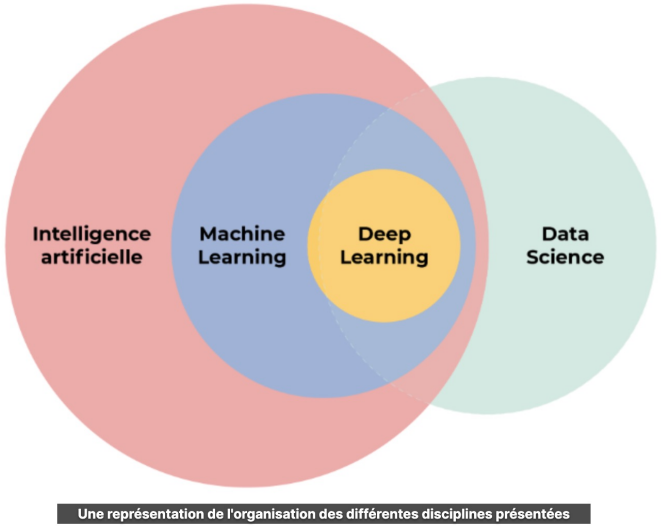
Big brother

- **L'intelligence artificielle est déjà présente dans nos quotidiens** : de nos applications de réseaux sociaux à nos choix d’itinéraires, en passant par nos choix musicaux ou de vidéo.
- Ce champ d’étude a encore beaucoup à nous proposer, et de nombreuses utilisations de l’IA vont voir le jour dans les prochaines années.

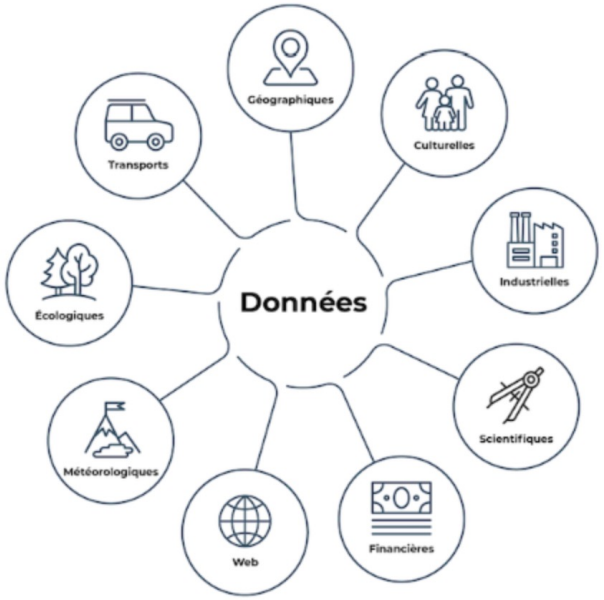
Définitions - IA

Concepts autour de l'IA

- Données, le Big Data, Deep Learning, machine learning



Une représentation de l'organisation des différentes disciplines présentées



Définitions - IA

Data, données ?

- Les "data", les "données", la "protection des données", la "magie de la data", le "vol de données", prendre des "décisions basées sur la data"....

- **Data = données**, les data sont simplement la traduction anglophone des données

Définition

- Ce sont ces informations qui sont enregistrées pour être utilisées par les programmes informatiques.
- Exemple: un texte enregistré sur votre ordinateur, un mémo vocal enregistré avec votre smartphone, la dernière photo de votre appareil photo, etc

Définitions - IA

Echelle ?

- Tout le monde produit des données. Chaque minute :

Google est sollicité près de 4 millions de fois ;

4,5 millions de vidéos sont visionnées sur YouTube ;

188 millions d'emails sont échangés.

- Toutes ces données forment le concept de **Big Data ou "données massives".**
- L'élément-clé le **volume** de données qui est considérable.

- Nature des données du Big Data: variées ! Celle du monde de l'Internet mais également celles de capteurs dans le monde "physique »: capteurs, balises, etc.

Définitions - IA

Finalités

- Souvent collectées et utilisées par des organisations. Objectifs: améliorer votre expérience en ligne ou offrir des services personnalisés
- Nouvelle discipline: la **Data Science, ou science des données**.
- **Exemple:** une chaîne de vêtements avec plusieurs boutiques en France. Nombreuses données, notamment l'ensemble des ventes réalisées par ses différents points de vente. La Data science : analyse les ventes, identifier les collections qui sont le plus susceptibles de se vendre à l'avenir.

Compétences

- Connaissances en mathématiques et en statistiques pour analyser les chiffres ;
- Compétences en informatique pour être en mesure de traiter des quantités importantes d'informations ;
- Compétences dans le secteur dans lequel elle intervient.
- Exemple: après son analyse des ventes, mettre en place des outils pour anticiper automatiquement les produits qui seront les plus vendus dans les prochains mois: une IA !

Définitions - IA

Machine Learning, apprentissage automatique

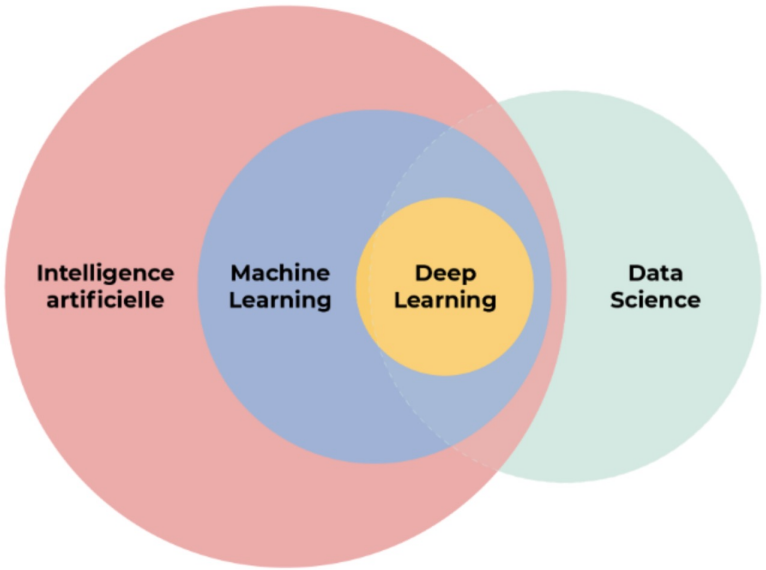
- Sous-ensemble de l'intelligence artificielle
- Permet à un programme informatique d'effectuer une tâche pour laquelle il n'est pas programmé explicitement: il est programmé pour apprendre à la faire
- Injection dans le programme de nombreuses données; il apprend à partir de ces données.
- Exemple: votre boîte email pour classer automatiquement un email en spam.

Deep Learning ou apprentissage profond

- Sous-champs d'étude du Machine Learning
- Repose sur la construction de réseaux de neurones artificiels
- Ces réseaux, composés de milliers, voire millions de neurones, sont inspirés du cerveau humain.
- S'applique souvent sur des quantités de données beaucoup plus importantes que le Machine Learning.
- Programme apprend de cette masse d'exemples et obtient dans certains cas de bien meilleurs résultats que les disciplines traditionnelles d'intelligence artificielle.

Définitions - IA

En résumé



Une représentation de l'organisation des différentes disciplines présentées

Définitions - IA

Mythes et réalité

Les programmes d'IA sont supérieurement intelligents aux humains

Les IA ne sont pas si *intelligentes !* Elles sont surtout très **spécialisées,** et ainsi **performantes** sur certaines tâches très **précises**.

"Les intelligences artificielles les plus abouties ont moins de sens commun qu'un rat !"

Exemple

"Rose a quitté le bâtiment avec sa valise"

- L'humain dispose d'un **sens commun.** Ainsi, par cette simple phrase :

1. vous savez que Rose est une personne et non une fleur ;
2. vous comprenez que Rose n'est plus dans le bâtiment ;
3. vous faites même l'hypothèse qu'elle s'apprête à voyager si elle est munie d'une valise, etc.

Cette capacité d'user de sens commun, les machines ne savent pas encore le faire.

Définitions - IA

Mythes et réalité

L'IA fonctionne comme le cerveau humain

L'ambition de l'intelligence artificielle: résoudre des problèmes complexes, qu'on pourrait penser réservés aux humains. Par exemple: la perception visuelle ou de la reconnaissance du langage.

Exemple

Intelligence peut être mesurée sous différents angles: la capacité à faire des opérations mathématiques rapidement ou d'avoir une mémoire considérable.

-> Les intelligences artificielles nous dépassent largement !

Mais

D'autres compétences qu'elles n'auront vraisemblablement pas avant longtemps: être capable de ressentir des émotions, de faire preuve d'empathie ou d'humour !

Définitions - IA

Mythes et réalité

Les programmes d'IA sont supérieurement intelligents aux humains

Exemple

Des robots sont-ils capables de joie et de tristesse ?

-> En apparence seulement: on peut doter les machines de toutes sortes d'émotions, mais il faut garder à l'esprit qu'elles ne font que les simuler ! Au, mieux, reproduire ce qu'elles ont appris ! Mais rien n'est spontané… Cf chatGPT et son apprentissage basé sur l'absorption de texte type wiki, etc

Définitions - IA

Travers

- Contrôles des données (RGPD) Bien choisir/maitriser à qui vous les partagez et paramétrez vos services selon vos usages !
- Ne collecter que le nécessaire !
- Les deepfakes participent à la désinformation. Pour contrer cela, rien de tel qu'un esprit critique et un peu de recherches pour croiser vos sources
- L'IA utilise beaucoup d'énergie. L'optimisation de ses coûts écologiques est un sujet important à garder en tête pour le futur

Exemples IA

- chatGPT
- Midjourney
- Dall-E
- Lensa
- Bard
- Etc

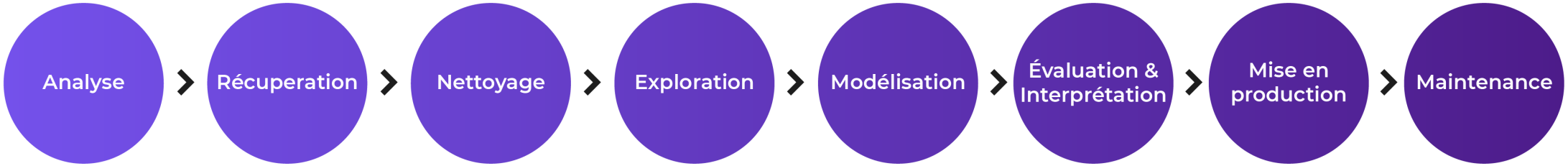
Définitions - IA

Conclusion

- L'évolution technologique apportée par l'IA implique une **transition** sur le marché du travail, avec des métiers qui disparaissent et des nouveaux qui sont créés.
- L'IA va surtout nous **aider** dans certaines tâches de nos métiers à être plus efficaces, plus précis ou plus rapides.
- Les êtres humains ont encore de belles années de travail devant eux et vont pouvoir s'adapter à ces changements technologiques grâce à la **formation tout au long de la vie**.

Projet - IA

Un projet IA, plusieurs étapes



Projet - IA

Acteurs

- **Experts métiers** : des spécialistes du secteur (industrie, commerciale) dont la connaissance sera cruciale pour développer une solution pertinente
- Experts en **projets numériques**, notamment un architecte logiciel et des développeurs
- **Spécialistes de l'IA** : un expert IA, des Data Scientists
- Une **gouvernance** : un DPO (*Délégué à la Protection des Données*), un représentant du service juridique (pour la gestion légale des données), etc

1. Cadrer

- **un cadrage business approprié** (ou ROI): quoi, pour qui, pour quoi ? Quel budget ?
- **une évaluation de l'impact** : éthique, social, sécurité des personnes.
- **la gouvernance des données** : quelles données, accessibles par qui, comment, quels droits d'utilisation… ?
- **le design de la solution** : quelle architecture ? Quel niveau de sécurité des données ?
- **l'industrialisation de la solution** : *monitoring* des résultats, gestion des utilisateurs, communication, *change management*...

Projet - IA

2. Récupérer

- Collecter le plus de données possibles, même si vous n'êtes pas sûr de les utiliser, pour tous les vecteurs possibles: données publiques, capteurs, IoT, API, etc.

3. Nettoyer

- **S'assurer que données sont bien fiables:**

- regarder s'il y a des données manquantes. .

- vous assurer qu'il n'y a pas de données aberrantes.

- Travail effectué par le Data Scientist. Exemple: choisir de remplacer les données manquantes ou erronées au moyen d'outils statistiques. On parle ici d'***imputation statistique***.
- À l'issue de cette étape, vous avez donc des données de qualité à votre disposition !

Projet - IA

4. Explorer

- Observer les données sous toutes leurs facettes: c'est l'**exploration de données** ou **fouille de données** (en anglais, *Data Mining*)
- Data Scientist travaille avec des experts métiers et des experts des sources de données afin de mieux comprendre les données
- **Objectifs:** vérifier vos hypothèses ou intuitions qui pourront être validées ou contredites !

5. Modéliser

- En deux phases:

1. A**pprentissage:** entraîner votre modèle avec des exemples basées **sur des données des périodes passées**
2. **Prédiction** : votre système est prêt: prédire le futur !

Projet - IA

6. Evaluer

Evaluer le modèle, c'est-à-dire confirmer qu'il est pertinent et fournit des prévisions de qualité… ou bien l'entrainer à nouveau !

7. Industrialiser

- Ça y est, votre système d'intelligence artificielle est prêt à être mis en place. Il va permettre à l'usine de **piloter** au mieux sa consommation d'énergie. À la clé, l'usine espère réaliser des économies d'énergie.
- Pour s'en assurer, votre équipe projet fera le point après une période de test. Le système d'IA est-il pertinent ? En particulier, amène-t-il une plus-value business ?
- Un tel système d'IA, une fois mis en route, doit faire l'objet d'un suivi. Vous devez vous assurer quotidiennement qu'il produit des résultats pertinents. De temps en temps, vous devez réaliser une maintenance du système. C'est l'occasion de réajuster ses paramètres pour s'assurer qu'il fournit des résultats toujours pertinents.

Machine learning – Focus « modèle »

Modèle

- Un **modèle** est une représentation mathématique d'un problème donné.
- La modélisation des données se compose de deux phases : **l'apprentissage** et la **prédiction.**

Cas d'un agent immobilier : faire estimer un bien au prix de vente le plus conforme au marché immobilier. Pour réaliser cette évaluation, deux étapes :

1. **il recueille des données** sur les caractéristiques clés du bien immobilier (par exemple : l'emplacement géographique, la superficie, l'état général, etc.) ;
2. **il procède ensuite à une évaluation** fondée sur des données publiques disponibles, ainsi que sur sa propre expertise immobilière.

Au fil du temps, l'agent va devenir de plus en plus apte à fournir une évaluation de bien: il a développé un **modèle** d'évaluation des prix.

Machine learning – Focus modèle

Pour l'IA, comment modéliser ?

- **Apprentissage** (ou entraînement): étape qui permet de construire le modèle, i.e., fournir à l'algorithme de nombreux exemples à analyser pour qu'il puisse apprendre par l'expérience.

Il existe trois façons d'apprendre pour un programme d'IA, i.e., de former un modèle : **l'apprentissage supervisé, non supervisé** et **essai/erreur.**

Méthode 1 : l'apprentissage supervisé

Exemple: apprendre à estimer le prix d'une maison. Fourniture de nombreux exemples de ventes de maisons, en donnant les caractéristiques de chaque maison ainsi que son prix de vente.

Ainsi, pour chaque vente, nous aurons :

- les **caractéristiques** : les caractéristiques que nous souhaitons pour estimer le prix de n'importe quelle maison, une fois le système entraîné: la surface, le nombre de chambres, la présence d'un balcon, etc.
- les **étiquettes** : cible que nous souhaitons prédire. En entraînement, l'algorithme a accès à cette information. Une fois le système prêt, l'objectif est de le prédire à partir des caractéristiques d'une nouvelle maison.

Machine learning – focus modèle

- En deux phases:

1. Au départ, lorsque l'algorithme considère les premiers exemples, il produira des réponses qui ne seront *a priori* pas très pertinentes.

2. Au fur et à mesure qu'il intégrera de nouveaux cas, il s'adaptera, se transformera, jusqu'à devenir prêt à estimer le prix de maisons qu'il n'a jamais observées !

Méthode 2 : l'apprentissage non supervisé

- Dans le cas présent, l'algorithme n'a **pas accès aux étiquettes !**
- On ne sait pas à l'avance ce que l'on va trouver !
- Exemple: fournir une liste de maisons et demander de faire 3 groupes de maisons, sans aucune supervision de notre part !
- Une fois trois groupes formalisés: appel à des experts qui trouvent le nom des étiquettes !

Machine learning – focus modèle

Méthode 3 : l'apprentissage par essai/erreur

- Aussi appelé apprentissage par **renforcement**,
- Un **"agent"** (l'algorithme) qui interagit avec un **"environnement"** (l'ensemble des "caractéristiques").
- Objectif: trouver par tâtonnements successifs (essai ou erreur) la solution optimale à un problème donné. On dit que cet algorithme est **auto-adaptatif** : il est en apprentissage constant.
- Ecueil: l'algorithme fait parfois des millions d'essais avant de devenir un as dans une discipline…. Donc, entraînement parfois très long !

- Algorithme souvent utilisé pour les jeux: jeu de Go
- Autre exemple: confier à un algorithme la conception du plan d'un immeuble. On précise toutes les contraintes à prendre en compte, ainsi que les objectifs à optimiser:

on souhaite avoir 3 salles de réunion, chacune d'une certaine dimension, avec un certain nombre de fenêtres, et idéalement à proximité les unes des autres.

L'algorithme va chercher le plan idéal par tâtonnements.

Deep learning – focus modèle

Kesako ?

- Le Deep Learning: **réseau de neurones artificiels convolutifs.**
- Par l'exemple: le domaine de la **reconnaissance d'image** et plus précisément le tri de photos
- Objectif: construire un **programme d'intelligence artificielle** pour effectuer ce tri à notre place. Sa mission : étiqueter, pour chaque photo, le ou les sujets présents avec leur(s) nom(s).

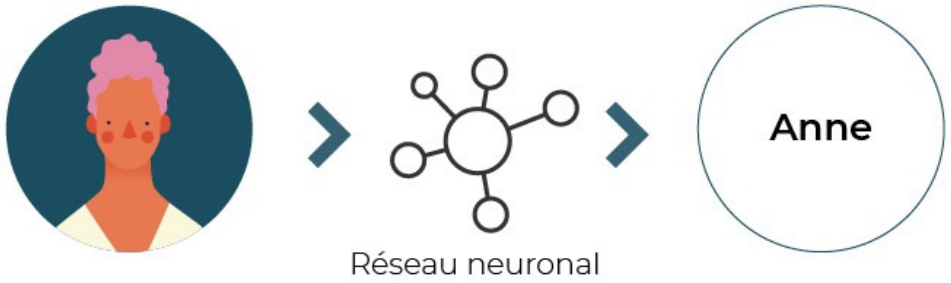
Comment ?

- Entraîner notre intelligence artificielle à identifier les membres de notre famille sur des clichés
- Rassembler une série de clichés pour chaque individu (étiquette, cible)

Deep learning - focus modèle

Comment ?

- Utiliser un **réseau de neurones artificiels** !
- Càd ? C'est comme une boîte avec :
- une **entrée** : les données fournies (une photo) ;
- une **sortie** : le résultat attendu (le prénom du sujet présent sur la photo), et
- entre les deux : un **réseau neuronal,** autrement dit des couches successives de neurones artificiels.



Deep learning – focus modèle

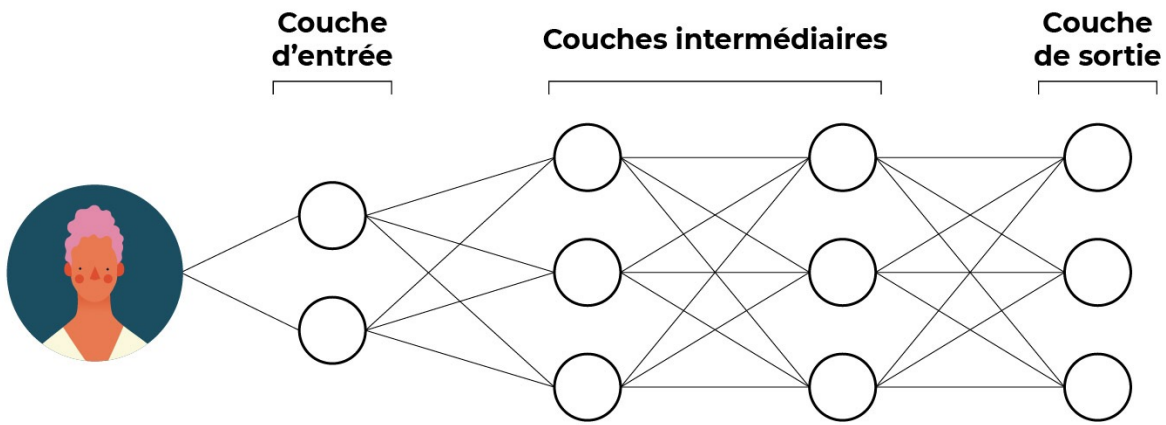
Méthode ?

- **Supervisé** ou **non supervisé,** ici ?

Deep learning – focus modèle

Supervisé !

- Problème typique d'apprentissage **supervisé:** entraîner un algorithme à partir d'exemples annotés.
- Comment ? Le réseau est composé de nombreux neurones, qui sont groupés en plusieurs couches de neurones. De base, trois couches: la **couche d'entrée,** les **couches intermédiaires** et la **couche de sortie.**



Deep learning – focus modèle

Couches

La couche d'entrée

- Elle reçoit l'information de notre image. Notre cliché est composé de millions de points appelés **pixels.** Chaque pixel alimente un neurone de la couche d'entrée.
- En résumé: si l'image comporte 1 million de pixels, la première couche sera composée d'1 million de neurones.

Les couches intermédiaires

- Peuvent être nombreuses (de quelques dizaines à plusieurs centaines)
- Les neurones d'une couche sont connectés aux neurones de la couche suivante. Les neurones d'une couche interagissent avec ceux de la couche suivante en réalisant des **opérations mathématiques** élémentaires comme l'addition, la soustraction, la multiplication et la division.
- En fonction du résultat des opérations mathématiques, chaque neurone sera **activé** ou non. En réalité, on utilise ici une fonction mathématique appelée **"fonction d'activation"**. Elle prend en entrée les données de la couche précédente et renvoie 1 (activation du neurone) ou 0 (non-activation).

Deep learning – focus modèle

Couche

La couche de sortie

- La dernière couche porte le **résultat final**
- Dans notre exemple, il s'agit du nom de la personne présente sur le portrait.

Y'a plus qu'à !

- L'algorithme doit **s'entraîner,** en prenant chaque photo et en la faisant passer par ses différentes couches de neurones jusqu'à obtenir un résultat. Au début, il va répondre au hasard, car il n'a aucune expérience !

Deep learning – focus modèle

Limitation

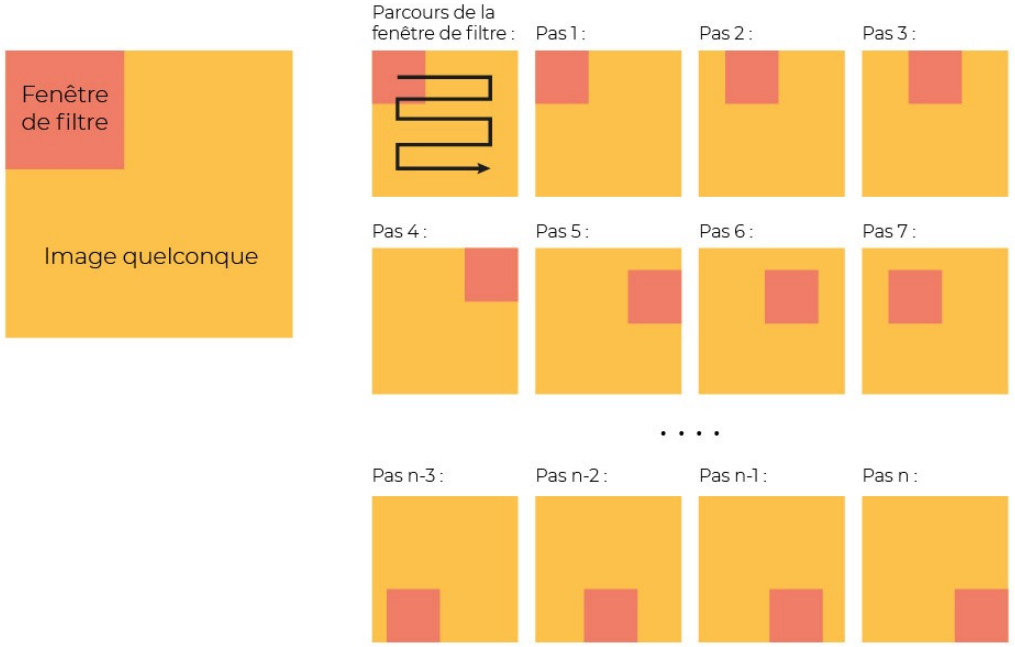
- On ne peut catégoriser que les clichés contenant des portraits qui sont parfaitement centrés !
- Si nous lui fournissons des clichés où les portraits ne sont pas bien centrés, ses performances sont moindres. Pire, si nous lui donnons une photo de groupe, il ne pourra pas étiqueter plusieurs personnes car il n'a pas été entraîné pour ce besoin!

-> Nous avons besoin de muscler notre outil pour lui permettre d'étiqueter plusieurs sujets sur une même image.

Aller plus loin…

- Utiliser un réseau de neurones particulier appelé **réseau de neurones convolutifs** (en anglais : *convolutional neural networks*, souvent abrégé en *CNN*).
- **Convolution** ? Un **filtrage** de notre image. Plutôt que de traiter l'image en un bloc, nous allons la diviser en différents carrés que nous allons analyser séparément.
- Ainsi, nous allons nous déplacer progressivement sur les différentes zones de l'image, en effectuant ce qu'on appelle des **pas.**

Deep learning – focus modèle



Conclusion

- En alliant les réseaux de neurones artificiels et la convolution, les scientifiques ont mis en place **l'apprentissage profond,** ou *Deep Learning*.
- Mieux que le machine learning ? Oui, pour ses **résultats** et ses **capacités** futures :
  - capacité à exploiter de très grands gisements de données
  - amélioration continue

En résumé: le futur est là !!!

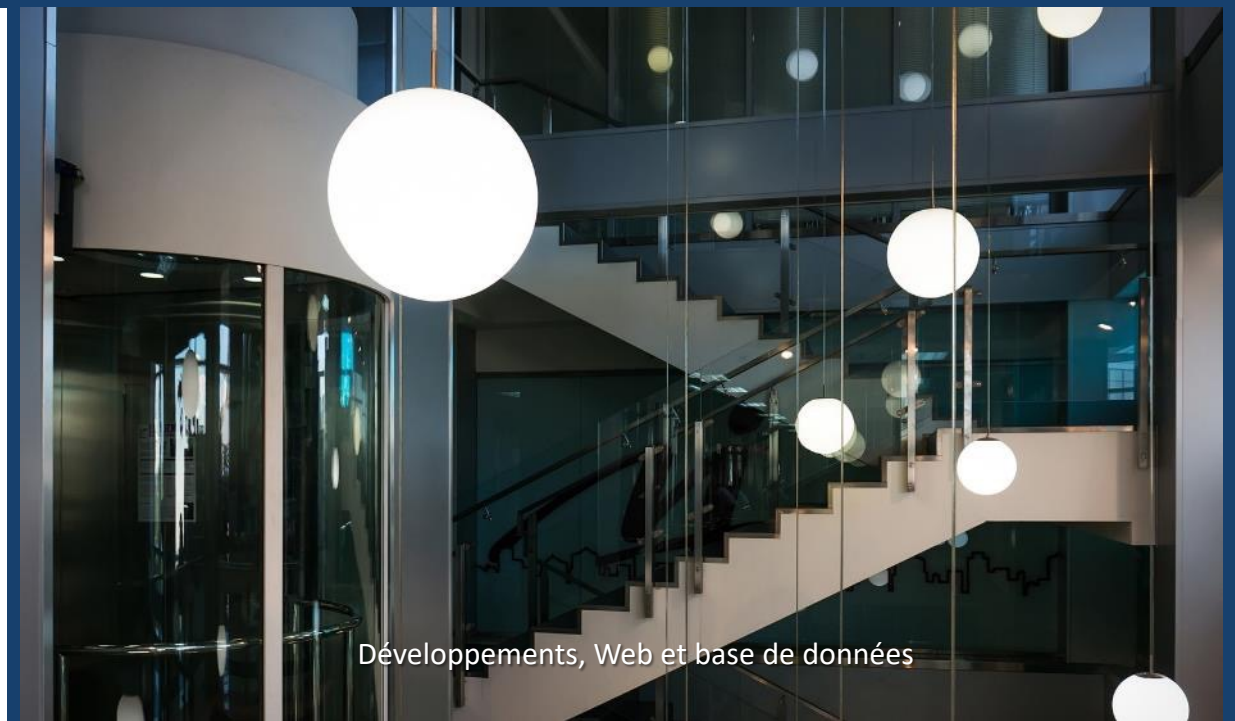
Exemple avec Pytorch

Quid ?

- Librairie python pour créer des réseaux de neurones
- Des datasets pour entrainer un réseau: `CIFAR10`

Chien, chat, etc.. ?

Merci d’avoir tenu le coup !



- Nicolas Perrier
- nperrier@artymon.com