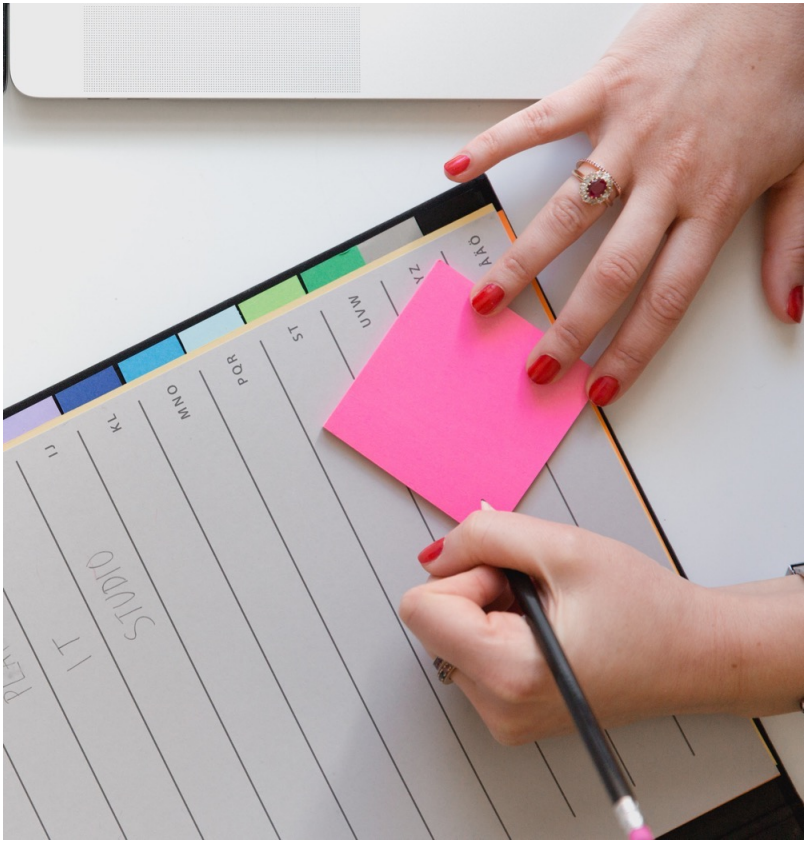
Formation

Sécuriser les applications web

SE24-174141 - Rév 3

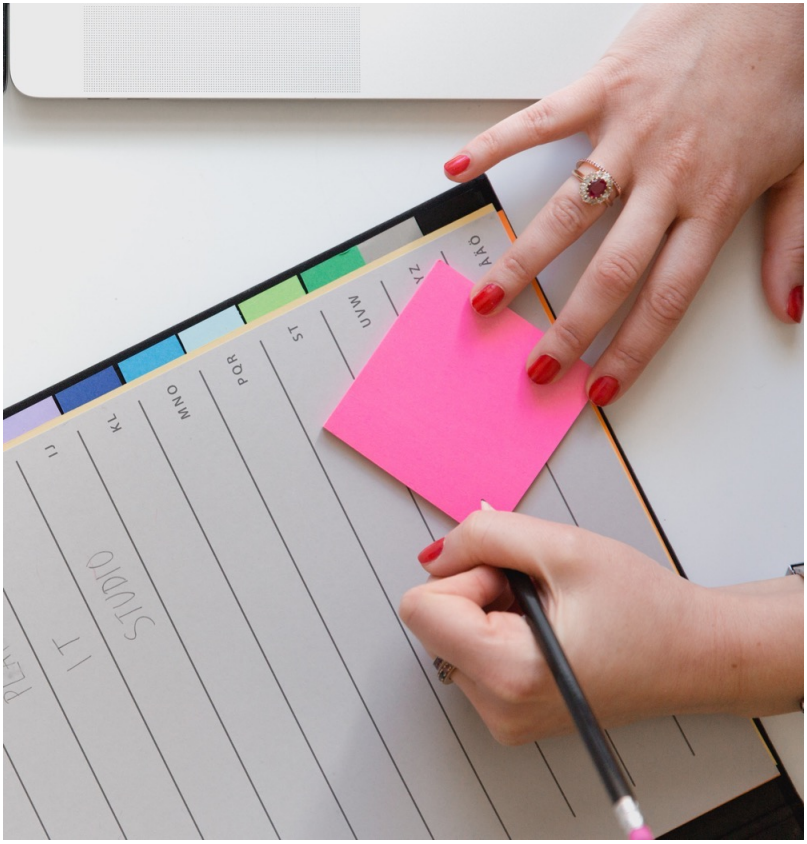
m2iformation.fr

Fonctionnement de la session de formation :



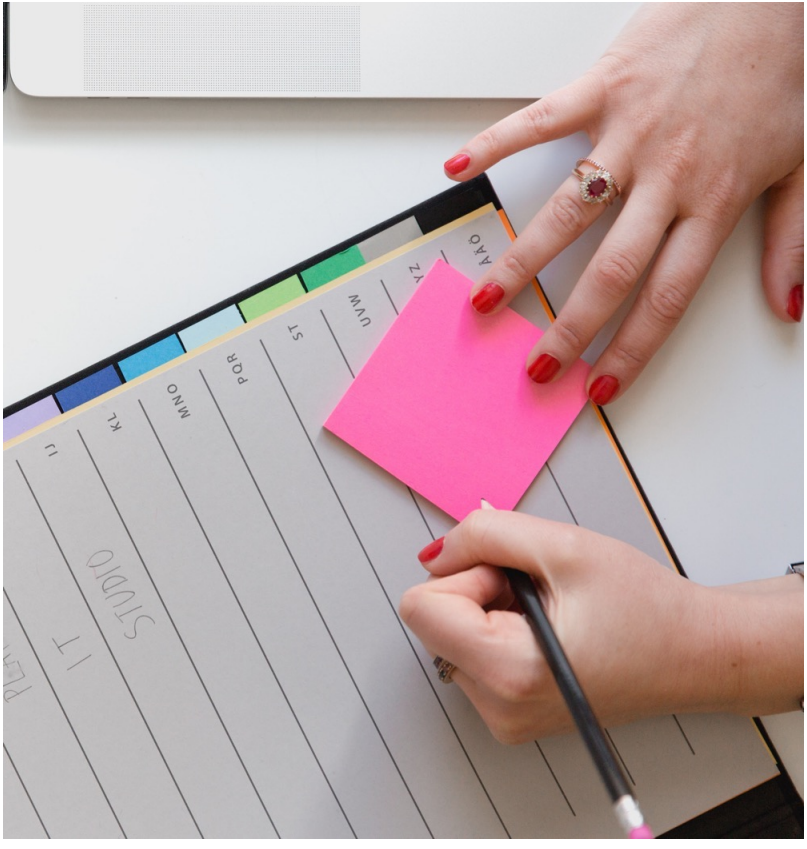
- Posez des questions
- Interrompez le formateur si un point reste incompris
- Vos expériences personnelles concernant le sujet peuvent être intéressantes et donner lieu à un complément d'information
- Prenez en note les informations complémentaires fournies
- Le formateur reste le maître des horloges

Après cette formation, vous serez en mesure de :



- Intégrer la sécurité dès le début du cycle de développement (DevSecOps)
- Utiliser les techniques de sécurisation des applications Web
- Identifier et mettre en place des contre-mesures contre les vulnérabilités courantes.

Après cette formation, vous serez en mesure de :



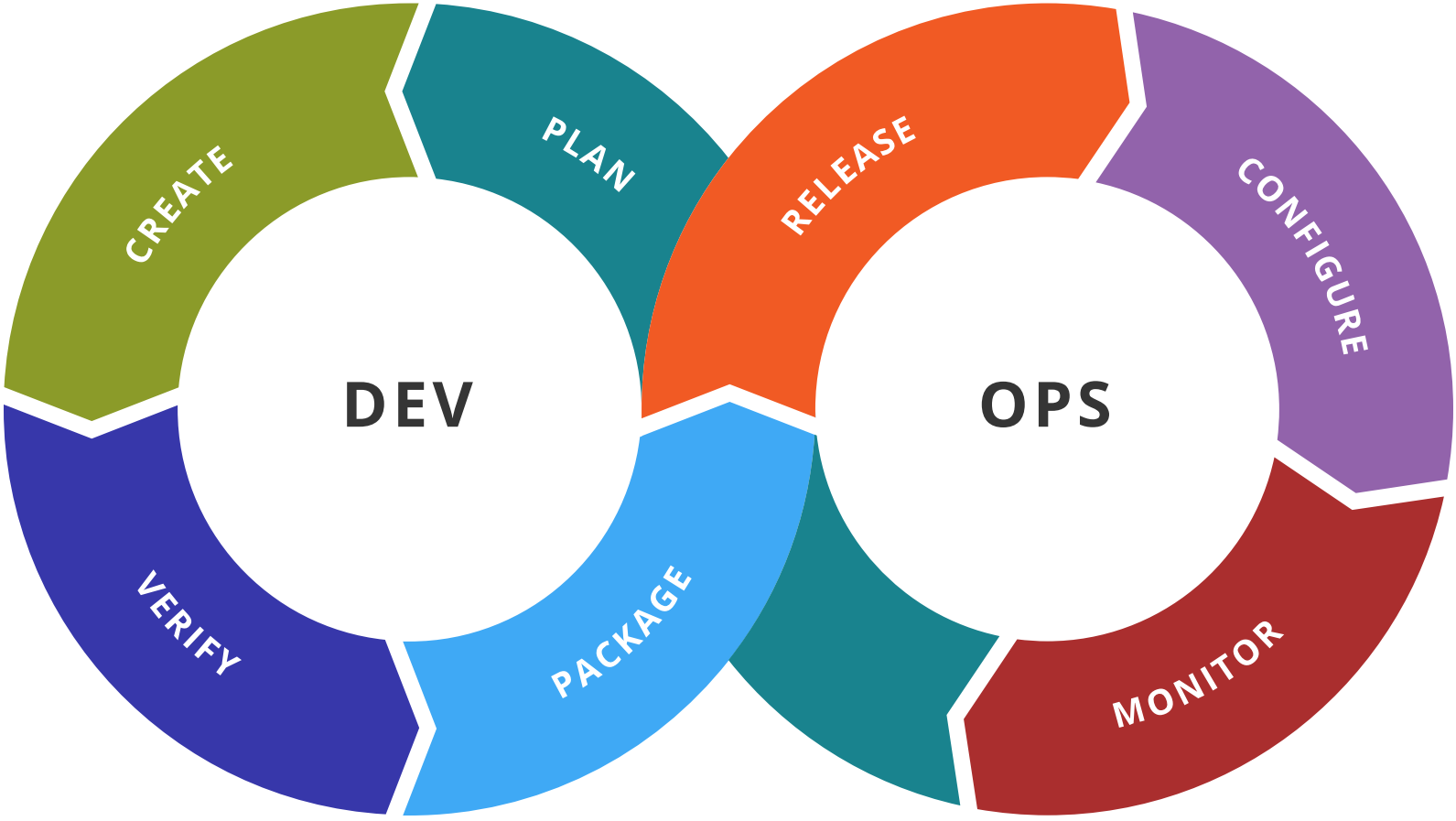
- Décrire les fondements du DevSecOps
- Intégrer la sécurité dans un pipeline CI/CD (intégration et développement continu)
- Identifier les vulnérabilités spécifiques aux technologies de conteneurisation
- Effectuer des tests d'intrusion relatifs aux technologies de conteneurisation
- Sécuriser les technologies de conteneurisation

Présentation des stagiaires

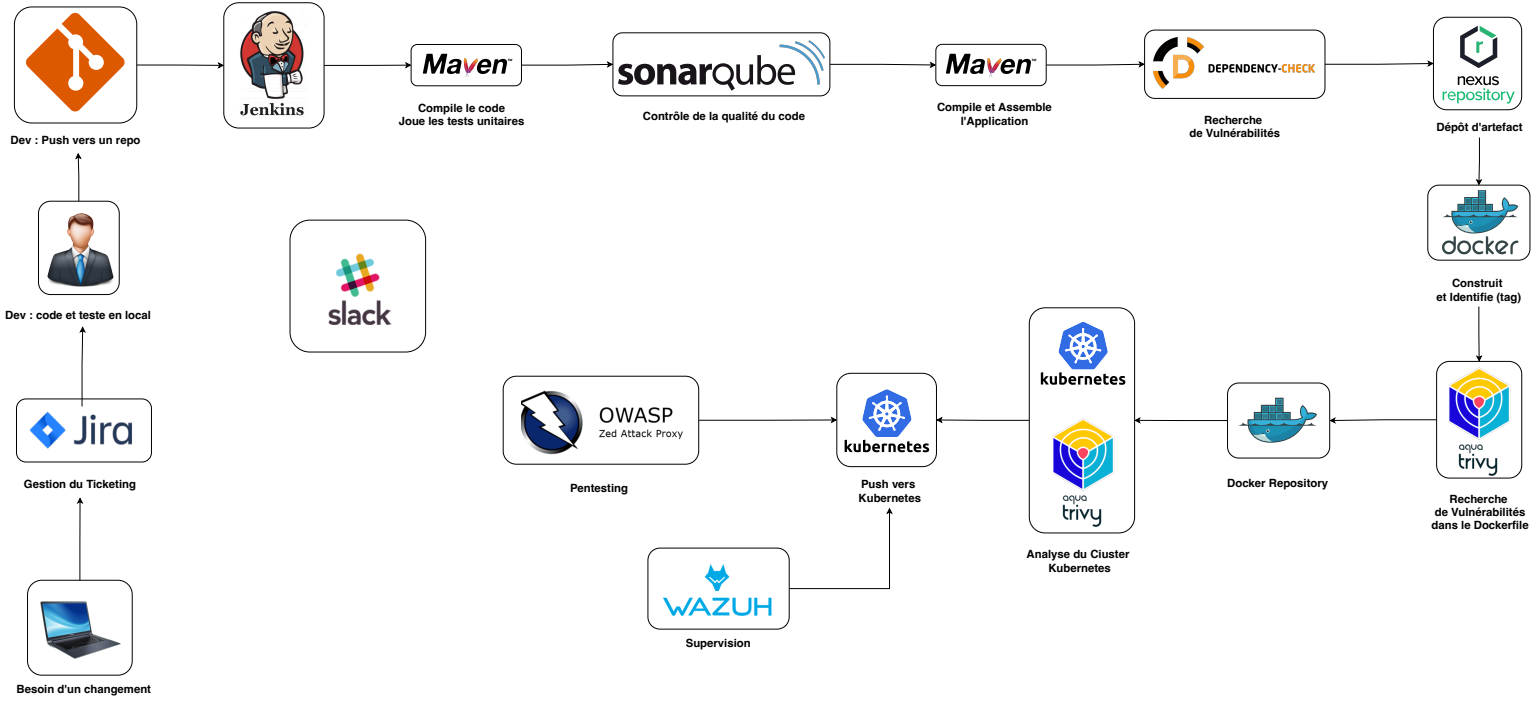
- Nom
- Niveau en DevOps et Cybersécurité
- Attente de cette formation

SÉCURISER LES APPLICATIONS WEB

Schéma classique de DevOps :



Être capable d'expliquer ce schéma :



CID (Confidentialité, Intégrité et Disponibilité)

Ces trois principes sont au cœur de la sécurité informatique et sont essentiels pour comprendre et mettre en œuvre des pratiques de sécurité efficaces.

- **Confidentialité**
- **Intégrité**
- **Disponibilité**

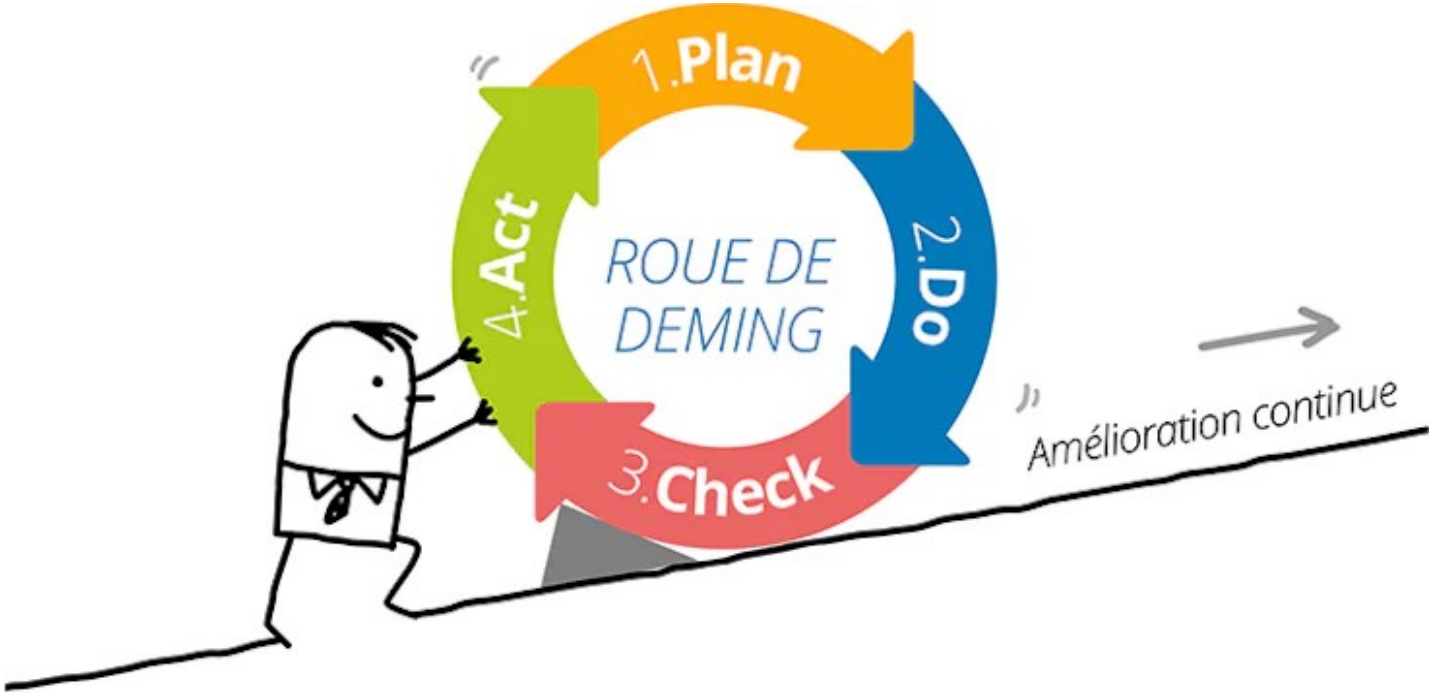
Les 3 états de la données

Les trois états des données se réfèrent aux différentes manières dont les données peuvent être stockées, traitées ou transmises au sein d'un système informatique.

- **Données au repos (Data at Rest)**
- **Données en Mouvement (Data in Transit)**
- **Données en Usage (Data in Use)**

Principes de base de Qualité

Amélioration continue

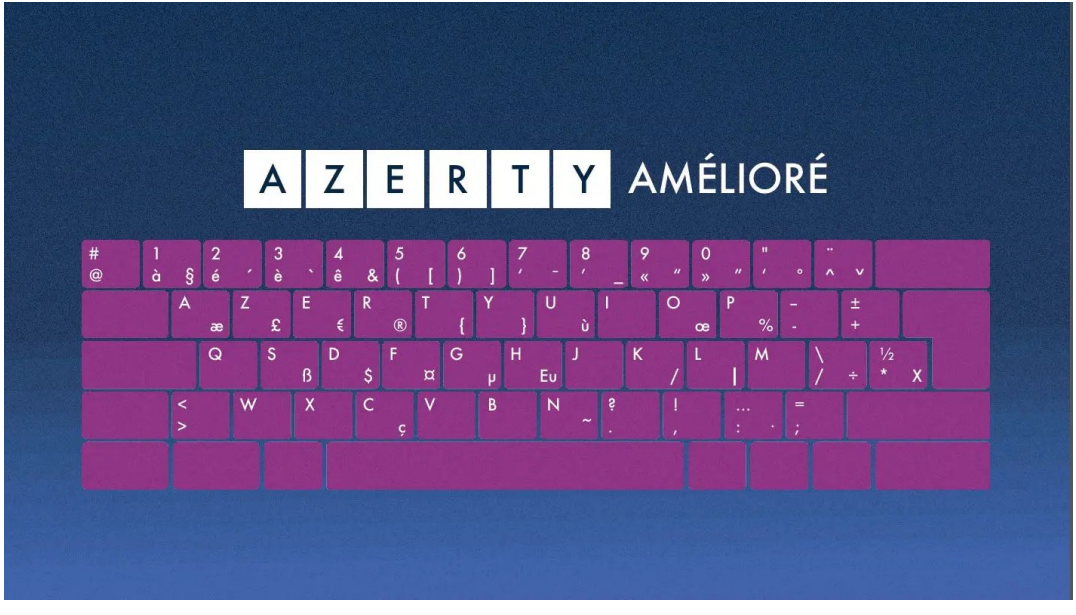


INTRODUCTION

Préambule

Être un développeur

- Savoir taper sur un clavier sans le regarder
- Préférer la disposition AFNOR du clavier pour sa localisation plus logique des paires de signes.



Préambule

Être un développeur

- Au-delà du branchement conditionnel et de la boucle, connaître les algorithmes (tri, recherche) et structures de données (tableau, vecteur, liste, arbre, graphe) classiques
- Savoir déterminer le "grand O" d'un algorithme
- Connaître plusieurs langages différents. Être capable de passer de l'un à l'autre

roadmap.sh

- Réutiliser du code déjà écrit et validé depuis un certain temps : librairies, framework
- Utiliser pour cela les dépôts standards pour chaque langage: pypi, npm et autres
- Avoir de bonnes habitudes de veille technologique afin de se maintenir à jour

Un exemple de développement : Rendering Text

Préambule

Être un développeur web 'full-stack'

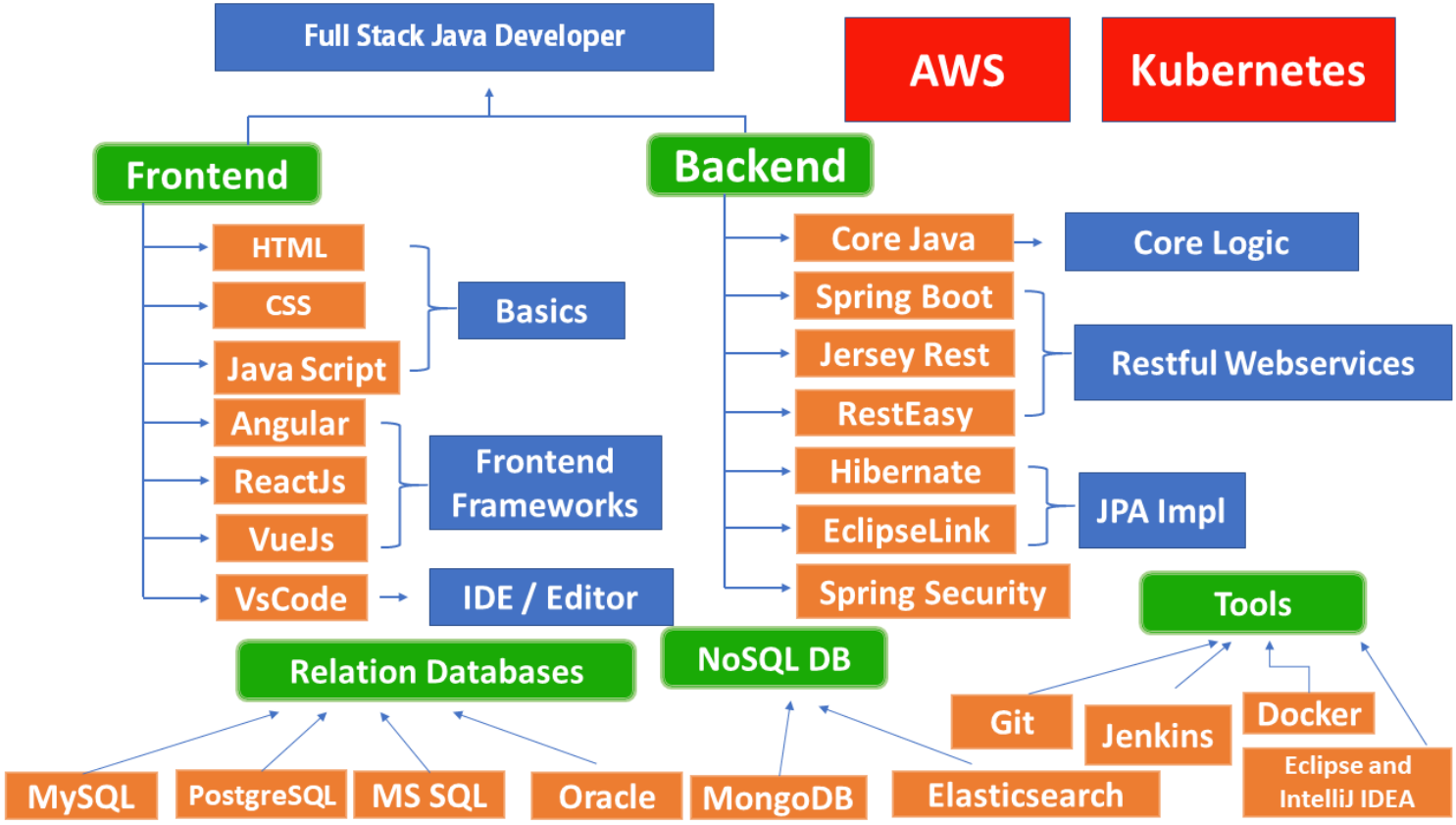
Être capable d'intervenir à toutes les étapes de conception et de maintien d'une application web.

- Base de données SQL ou NoSQL
- Couche traitement (java, .net)
- Reverse proxy (apache, nginx, traefik)
- Browser (HTTP, HTML, javascript et CSS)

Un exemple de stack MEAN :
- MongoDB
- Express
- AngularJS
- NodeJS

Mais aussi MERN, MEVN, LAMP (historique) ou autres.

Préambule


Full Stack Java Developer
AWS
Kubernetes
Frontend
Backend
HTML
CSS
Java Script
Basics
Angular
ReactJs
VueJs
Frontend Frameworks
VsCode
IDE / Editor
Core Java
Core Logic
Spring Boot
Jersey Rest
RestEasy
Restful Webservices
Hibernate
EclipseLink
JPA Impl
Spring Security
Tools
Relation Databases
NoSQL DB
Git
Jenkins
Docker
Eclipse and IntelliJ IDEA
MySQL
PostgreSQL
MS SQL
Oracle
MongoDB
Elasticsearch

Préambule

Un développeur conscient des problématiques de cybersécurité

- Utiliser un VCS (Version Control System) pour sauvegarder vos développement
- Problématique de la sauvegarde (backup), et règle du 321
- Vos communications se font à l'aide de openSSH ou d'un VPN (Wireguard)
- Utilisation d'un IDE moderne (VSCode ou Neovim) pour valider vos développements à l'aide de linters, formatters et des protocoles LSP et DAP.
- Intégration dans un processus agile
- Utilisation des container (Docker / Kubernetes) et des VM (VMWare / Proxmox)
- Connaissance du cloud

Préambule

Un développeur conscient des problématiques de cybersécurité

Un code simple possède le minimum de dépendance :
- Il possède une faible "surface de vulnérabilité"
- Il sera facile à maintenir, puisque modifier une partie du code aura un minimum d'effet non-anticipé sur une autre partie du code
- Ce qui nous mène aux notion d'urbanisme et de micro-service (API).

Refactoring Guru

Les préoccupations standards :
- **CRUD** (Create, Read, Update and Delete)

La simplification :
- **MVC** (Model, Vue, Controler)

- **KISS** (Keep It Stupid Simple)
- **DRY** (Don't Repeat Yourself)
- **YAGNI** (You Aren't Gonna Need It)
- **SOLID**

"The best part is no part. The best process is no process. It weighs nothing. Costs nothing. Can’t go wrong"
Elon Musk
"Avant de sortir, jetez un dernier coup d’oeil dans le miroir et enlevez un accessoire"
Coco Chanel

Tests unitaires

Les test unitaires servent à établir lors de chaque évolution du code que les fonctionnalités de base n'ont pas été impactées par ce changement.

C'est utile, car cela facilite la validation des modifications au niveau le plus bas, permettant de passer aux test de plus haut niveau tout de suite.

Le style de développement agile TDD (Test Driven Development) consiste justement à écrire d'abord le test qui valide l'évolution du code souhaitée.

Il échoue forcément au début.

Puis, on écrit le code qui effectue l'action souhaitée. Finissant ainsi par valider le test.

Des outils ou librairies standards dans chaque langages existent et sont en permanence utilisées pour ces actions.

Les formats de fichiers que vous pouvez rencontrer

Une description des différents formats de fichiers que vous pouvez rencontrer se trouve dans la série des fichiers format*md

Ces fichiers sont au format markdown.

- markdown (keenwrite / retext)
- ini
- toml
- yaml
- csv
- xml
- json
- latex
- pdf (une demo d'application web dédiée à PDF : StirlingPDF)

Tous les formats d'images, de vidéos.

- svg

Nuage de mots

Lexique Général :

• Collaboration	• Ticketing
• Gestionnaire de code • Pipeline / CI/CD	• Monitoring • Supervision • Alerting
• Build / Run	• Infogérance

Nuage de mots

Lexique du Cloud Engineer :

Cloud • Cloud • On-premise / On-site • IaaS / PaaS / Saas	**Cloud provider** • AWS • MSAzure • GCP • Scaleway • OVHCloud • S3ens • NumSpot • OutScale • Digital Ocean
Containérisation • Docker / Podman • Docker Swarm • Kubernetes / K8s • Openshift	**IaC** • Terraform • Ansible • Cloudformation • Packer • Bash
CI/CD • Gitlab-CI • Github-Actions • Azure DevOps • Jenkins	

Nuage de mots

Lexique du Développeur :

Général : • Frontend • Backend • Fullstack • Base de données • Langage de programmation • Framework • MVP (Minimal Viable Product) • POC (Proof Of Concept) • UI/UX design	**Langages de programmation :** • Python : Flask, Django • Dart, Flutter • Javascript : AngularJS, VueJS, ReactJS, NodeJS • PHP : Laravel, Symfony • Java
Outils de gestion code (Forge) : • git • Github • Gitlab • BitBucket	

Nuage de mots

Lexique d'architecture :

Général :
- Compute
- Network
- Storage
- SQL
- NoSQL
- MicroService
- Container
- VM

Stockage :
- PostgreSQL
- MongoDB
- GlusterFS
- NFS

Nuage de mots

Lexique de sécurité :

Acronymes : • SCA (Software Composition Analysis) • SAST (Static Application Security Testing) • DAST (Dynamic Application Security Testing) • RAST (Runtime Application Security Testing) • SIEM (Security Information and Event Management) • CSPM (Cloud Security Posture Management) • SOAR (Security Orchestration, Automation and Response) • IAST (Interactive Application Security Testing) • IAC (Secure Infrastructure as Code) • Compliance as Code • Vulnérabilité Management	**Outils de sécurité :** • Snyk • BridgeCrew / Chekov
Outils de Compliance : • Vanta • Drata • OpenSCAP	

Nuage de mots

Lexique de QoS :

Général : • Disponibilité • Scalabilité • Résilience • Durabilité • User Experience	**SRE (Site Reliability engineering) :** • SLA (Service Level Agreement) • SLO (Service Level Objectif) • SLI (Service Level Indicator)
Vocabulaire supervision : • ITSM • Incident • CPU • RAM	**Outils de monitoring / supervision :** • ELK • Grafana • Prometheus • Alert Manager • Splunk • Wazuh • Datadog • Uptime Kuma
Outils de qualité : • SonarQube • osv-scanner	

AVANT

Docker est un outil de containérisation permettant de déployer des services, d'en maintenir la fonctionnalité.

Le "dockerfile" est le format de fichier décrivant le contenu d'un docker.

Des fichiers décrivent plus avant ceci.

Un ITSM : GLPI

GLPI est un outil de ticketing, et bien plus, adapté pour gérer le support, et les évolutions sur des logiciels et les matériels qui les supportent.

Plus de détails dans le fichier GLPI.md

DÉVELOPPEMENT WEB

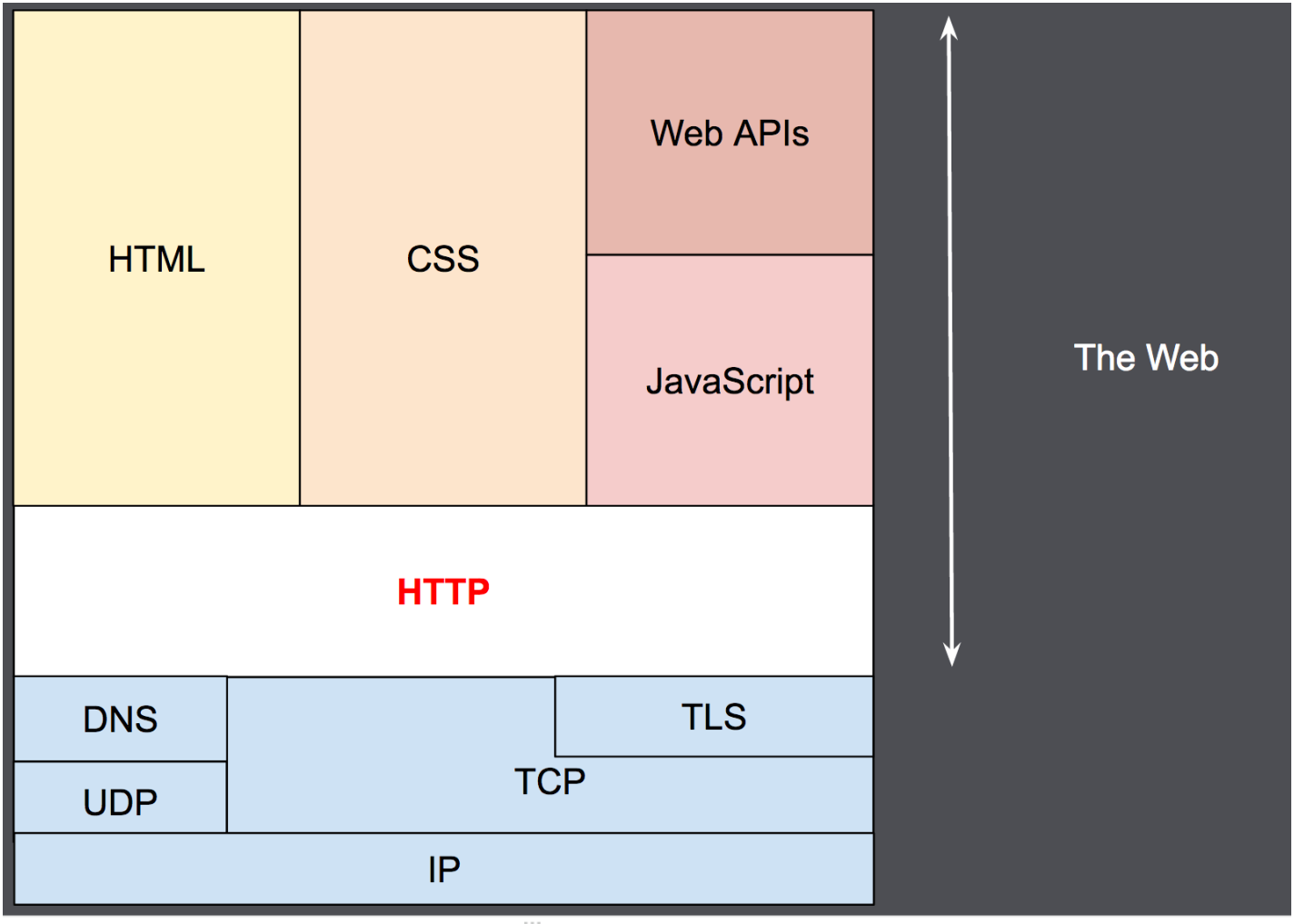
Le développement web consiste à utiliser successivement une série de protocoles qui, pris ensemble, construisent le web tel que nous le connaissons :

- **NTP**
- **IP**
- **TCP / UDP**
- **DNS**
- **Certificats**
- **HTTP**
- **HTML**
- **CSS**
- **javascript**

Tous ne sont pas directement de la responsabilité du développeur, mais les connaître facilite la vie dudit développeur.

- **NTP** : Network Time Protocole
- **IP (et ARP)** : Pour trouver la prise ethernet qui doit nous répondre
- **TCP / UDP** : Pour lui parler. Avec une session, ou sans
- **DNS** : Pour donner un nom que les humains peuvent comprendre

Développement web



Les certificats sont les outils qui servent à sécuriser les communications HTTP qui passent en clair en leur absence.

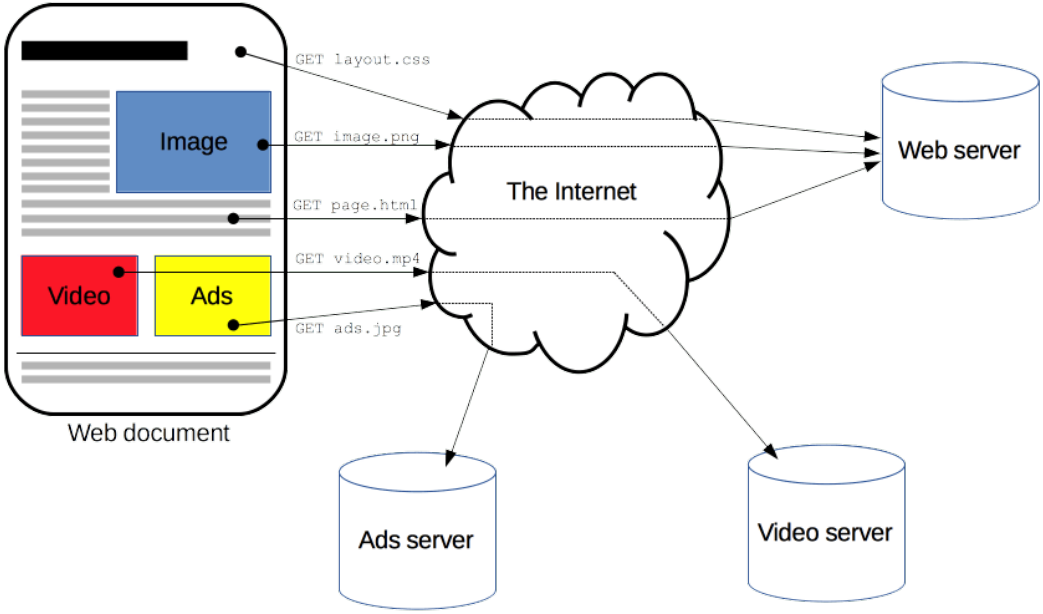
Ils respectent un certain nombre de régles pour être délivrés, renouvelés ou annulés.

Pour découvrir les informations sur les certificats :

- lire le fichier certificats.md
- regarder dans son browser
- ssllabs, pour analyser la qualité de la configuration des certificats d'un site web
- sslconfig, pour la configuration de reverse proxy

HTTP (HyperText Transfer Protocol) est le protocole de base du web. C'est ce qui fait transiter les informations entre le serveur et le client.

Regardons le fichier : http.md



Les APIs sont des sites web qui n'ont pas d'interface pour les humains. Ils se contentent de présenter une interface texte sur lesquelles des machines peuvent demander de l'information.

Plus d'information dans les fichiers markdown.

Des exemples d'usage d'API :

graou
marinetraffic
flightera
météo
doctolib

API gouv.fr

Des fausses API, pour apprendre à les utiliser :

12 APIs that you as a developer will love it

HTML

Le HTML (HyperText Markup Language) est un langage basé sur xml composé de deux parties head et body dans un conteneur html global :

```
<!doctype html>
<html lang="fr">
<head>
  <meta charset="utf-8">
  <title>Titre de la page</title>
  <link rel="stylesheet" href="style.css">
  <script src="script.js"></script>
</head>
<body>
  ...
  <!-- Le reste du contenu -->
  ...
</body>
</html>
```

Fonction 'inspecter' du browser

check : Markup Validation Service

Le CSS (Cascade Style Sheet) est un langage de description des propriétés de taille, de couleur des éléments HTML basé sur le concept de boite.

Voici quelques exemples de ce que l'on peut faire :

- Haikei
- Color hunt
- Realtime Colors
- Neumorphism
- Fancy Border Radius
- CSS Grid Generator
- Hamburgers
- CSS Buttons
- Glow Generator
- Omatsu
- W3.CSS
- FreeCSS exemple

Une collection d'éléments déjà construits : tailwindcss

Une description plus complète : Description

Javascript

Javascript est un langage interprété présent par défaut dans les browser (sous la forme de V8 pour les browser de la famille des chrome). Il est objet orienté et non-typé.

Il a un frère jumeau typescript, qui lui est typé. Il n'est utilisé que pour le développement. Il doit être traduit ("transpilé") en javascript pour être utilisé.

De la documentation :

Vous ne connaissez pas JS

Des exemples :

Rubik's Cube (des demos du MDN)
W3.javascript

Un framework connu : nodejs

Un exemple de site web :

Hugo

ou l'application python

Déploiement en trois tiers

Une application web (API ou pas) est classiquement déployée en trois tiers (suivant, en cela, les trois états de la donnée) :

- **Couche Stockage** : Une base de donnée, un fichier, de la mémoire
- **Couche Traitement** : Les données, extraites de la couche stockage, sont mises en forme avant d'être passées à la couche suivante
- **Couche Présentation** : Les pages web, le javascript et le CSS sont mis à disposition du browser distant pour qu'il les affiche au client. C'est la couche qui pose le plus de risques.

Le principe à comprendre ici, est qu'il ne peut pas avoir de communication qui saute une ligne. Rien ne passe directement de la couche présentation au stockage, ou réciproquement.

Gestion des identités

La gestion des identités est l'une des préoccupations principales de la sécurité de l'information, puisque c'est par ce canal que les autorisations de réaliser des actions sont données à des entités sur le réseau.

Active directory (Entra)
Keycloak

GESTIONNAIRE DE CODE

Gestionnaire de code

Création de la notion de gestionnaire de code :

Pour partager le code d'un programme, et ses évolutions, à l'époque où la communication se faisait par modem à 1200 bps, il était impossible de partager des mégaoctets de données.

Il a donc fallu créer une façon de ne transmettre que l'indispensable : Ce qui a changé.

Les programmes diff et patch ont donc était créés pour ce faire.

Le fichier diff+patch.md montre une façon basique de les utiliser.

C'est encore l'outil de base pour transmettre les évolutions du kernel linux : patchwork

Un outil linux connu est meld

Création de la notion de gestionnaire de code :

Après un certain nombre de tentatives (subversion, bitkeeper), la solution actuelle, écrite par Linus Torvald en un weekend s'appelle git.

Le fichier git+creation.md montre la création d'un nouveau repository (repo) git.

gitg est un exemple de logiciel qui montre les évolutions d'un repo git.

il est standard dans ubuntu.

sudo apt install gitg.

Les forges pour gérer du code

La forge

Le premier site web de gestion de repo de données s'appelle : Sourceforge

Une forge logiciel regroupe les services nécessaire autour de la gestion du code source.

La gestion d'un repo git, un système de ticketing et les autres.

Il en existe un certain nombre :

- github
- gitlab
- gitea
- bitbucket (de la famille Atlassian - Jira, Confluence, Bamboo, …)

CI/CD

Un système de **CI/CD (Intégration Continue / Déploiement Continu)** est un logiciel qui :

- Surveille les entrées dans un dépôt de sources
- Pour le compiler (le transpiler, ou n'importe quelle autre action préalable)
- Passer ses tests unitaires
- En évaluer la sécurité sous le maximum d'angle
- En évaluer la conformité normative et réglementaire
- Pour, finalement, le déployer dans l'environnement suivant

Nous allons utiliser Jenkins pour regarder.