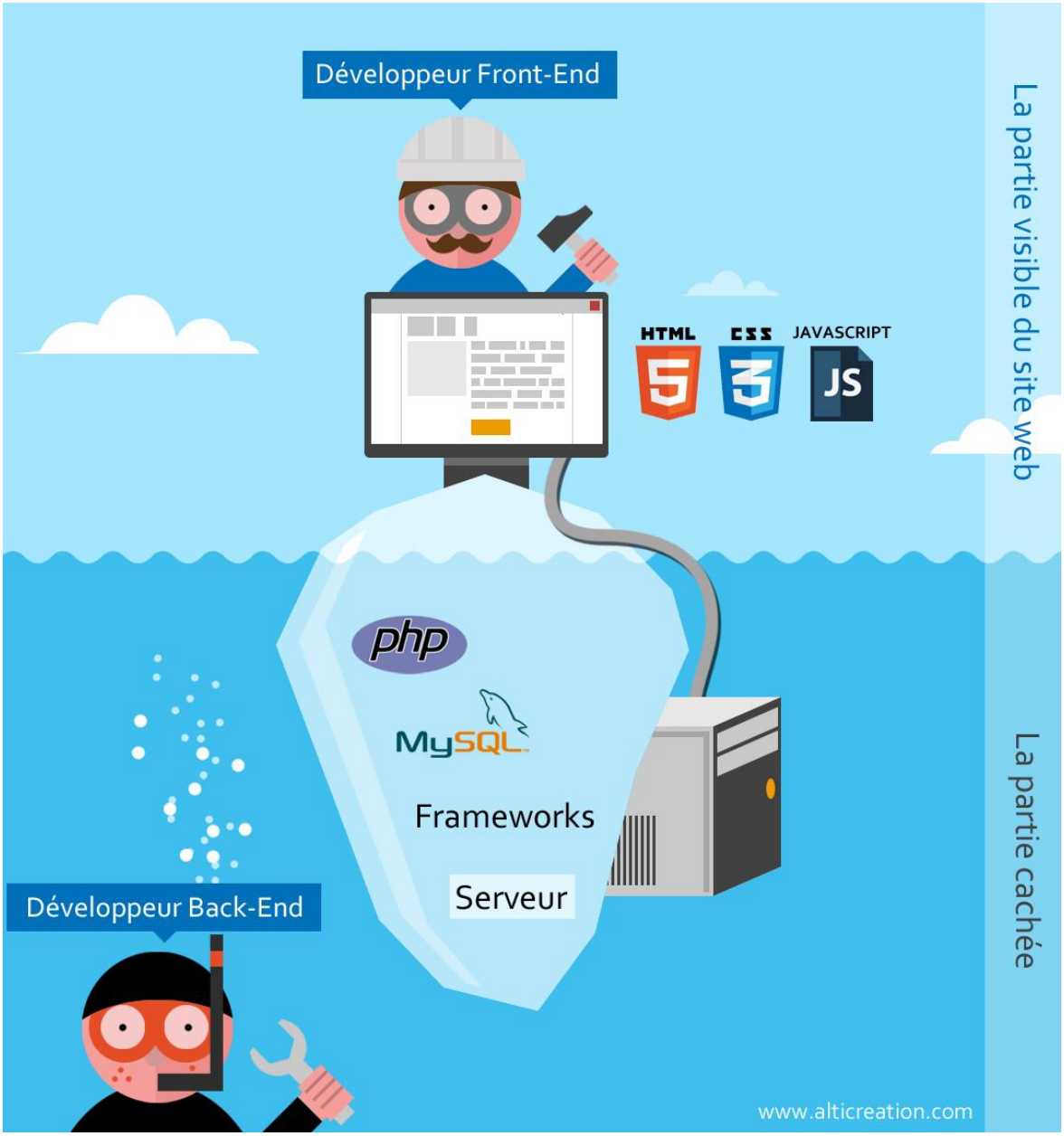
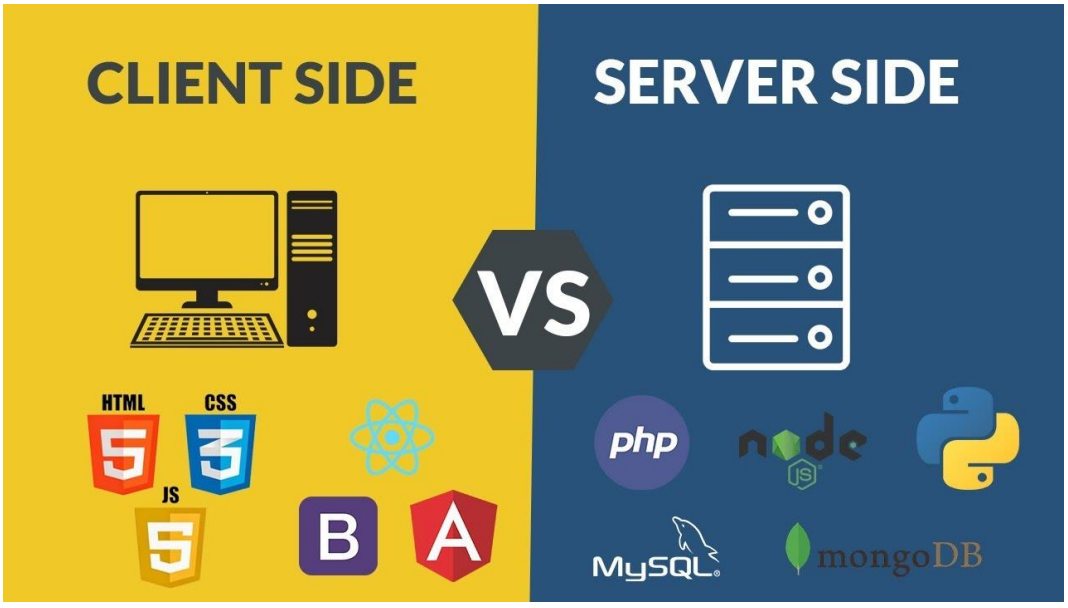

Développeur Front-End
HTML
CSS
JAVASCRIPT
La partie visible du site web
php
MySQL
Frameworks
Serveur
La partie cachée
Développeur Back-End
www.alticreation.com


CLIENT SIDE
SERVER SIDE
VS
HTML
CSS
JS
B
A
php
MySQL
mongoDB

HTML - HyperText Markup Language. Langage par balise, site statique type minitel

C'est moche !!

Je dois répéter mon style partout !

CSS - Les feuilles de style en cascade

Ça manque d'interaction et de mouvement !!

JavaScript - Langage de programmation de scripts coté client browser

Comment je rends mon site dynamique ?

Comment je sauvegarde des données ?

Comment j'interagis avec mes utilisateurs ?

PHP – Langage Hypertext Preprocessor pour générer du HTML coté serveur et avoir une application unique pour chaque client, permet d'avoir du rendu sur mesure en fonction de la demande

MySQL - (*Structured Query Language)*, langage de requête structurée

Je dois toujours réécrire le même code !

Quand ça devient compliqué c'est le bordel !

Reprendre le code d'un autre c'est chiant !

Framework - ensemble cohérent de composants logiciels structurels qui sert à créer les fondations ainsi que les grandes lignes de tout ou partie d'un logiciel et qui impose un cadre de travail respectant les bonnes pratiques de programmations (aussi bien back que front).

Git / SVN – Travailler à plusieurs, se partager les taches

Ça a déjà été fait pourquoi m'enmerder ?

Comment je peux réutiliser le code d'un autre ?

Composer, npm – Les Librairies d'applications, mettre à profit le savoir collectif.

Des problèmes complexes !

De la communication entre mes programmes !

Comment mieux coder ?

Design pattern – Développement Orienté objet – Interface - apprendre à gérer la complexité

C'est chiant faut recharger la page !

J'ai besoin de plus de rapidité !

Les websockets – connexion directe et gestion d'évènements avec le client

Le chemin de croix d'un développeur Full Stack

--- Les bases

1. Apprendre le HTML5, connaitre les balises disponibles, utiliser les bonnes balises aux bons endroits. Respecter les règles du w3c.
2. Apprendre le CSS, maitriser le responsive design en fonction de la taille de l'écran du client. Comprendre l'intérêt du SASS pour mettre des variables dans votre CSS et simplifier vos fiches de style. Réunir et compresser vos fichiers css pour la production.
3. Apprendre le javascript, maitriser les interactions côté client et dynamiser vos applications, savoir interroger une API. Réunir et compresser vos fichiers js pour la production.
4. Apprendre un des langages serveurs PHP/Python/Java/.Net/C pour créer du contenu dynamique, communiquer avec l'utilisateur, sauvegarder des données, gérer les sessions et organiser la logique de votre application.
5. Apprendre à créer et à requêter des bases de données (SQL)

--- La professionnalisation

6. Apprendre à utiliser des Frameworks pour gagner en productivité et utiliser les bonnes pratiques de la programmation. Au moins un Framework back (Laravel, Symfony, Django) et un Framework front (React, VueJs, Angular, Jquery)
7. Apprendre à installer des librairies et les adapter à notre cas d'usage en lisant les documentations. Réutiliser le travail d'autres développeurs et ne pas réinventer ce que d'autre ont déjà fait mieux que vous.
8. Apprendre la programmation orientée objet pour simplifier et rendre le code plus facile à réutiliser et à partager au sein d'un projet et pour vos futures applications.
9. Apprendre les designs patterns qui répondent au mieux aux problèmes les plus courants.
10. Apprendre à se servir d'un logiciel de versioning pour travailler en groupe et revenir facilement en arrière (Git/SVN)
11. Mettre une application en production et optimiser le SEO et le cache pour améliorer les performances
12. Créer des tests unitaires et fonctionnels sur les points sensibles de l'application

--- La spécialisation

13. Identifier le ou les domaines de compétences ou on est le plus à l'aise et entrer dans les détails des frameworks ou des librairies pour maitriser le cœur du savoir et participer aux projets open source.
14. Faire de la veille active sur les nouveautés et continuer à se former pour être à jour dans ses domaines de compétences

Conseils pratiques :

PRATIQUER, PRATIQUER, PRATIQUER. Créer des projets et recommencer en tentant d'améliorer à chaque fois. Demander conseil à des développeurs senior demandez-leur de relire votre code et de vous proposer des solutions plus performantes. N'hésitez pas à lire le code des autres sur les projets open source de github pour découvrir d'autre méthodologie de codage.

Les bases et les docs officielles

1. **HTML**

   Tout ce qui à savoir sur le HTML (les normes, les bonnes pratiques)
   https://www.w3schools.com/html/default.asp
   Toutes les références (balises)
   https://www.w3schools.com/tags/default.asp
   Les browsers support
   https://www.w3schools.com/tags/ref_html_browsersupport.asp

2. **CSS**

   Tout ce qui à savoir en CSS
   https://www.w3schools.com/css/default.asp
   Toutes les références
   https://www.w3schools.com/cssref/index.php
   Les browsers support (important)
   https://www.w3schools.com/cssref/css3_browsersupport.php

3. **Javascript**

   Tout ce qui à savoir en javascript
   https://www.w3schools.com/js/default.asp
   Toutes les références
   https://www.w3schools.com/jsref/default.asp

4. **SQL**

   Tout ce qui à savoir sur le SQL
   https://www.w3schools.com/sql/default.asp
   Toutes les références
   https://www.w3schools.com/sql/sql_ref_keywords.asp

5. **PHP**

   Tout ce qui à savoir sur le PHP
   https://www.w3schools.com/php/default.asp
   Toutes les références
   https://www.w3schools.com/php/php_ref_overview.asp

L'ensemble des références

https://www.w3schools.com/references/index.php

Des modèles de tous les principaux types de composant utile sur les app

https://www.w3schools.com/howto/

Vous pourrez également trouver des Quiz et des exercices corrigés sur chacun des langages sur le site w3c pour mettre à l'épreuve votre savoir et jauger vos compétences et votre progression.

https://www.w3schools.com/exercises/index.php
https://www.w3schools.com/quiztest/default.asp

Sur LinkedIn vous avez des tests sur les différents langages et les différents skills pour valider vos compétences sur le profil. C'est intéressant pour valider vos compétences sur les profils

https://www.linkedin.com/skill-assessments/hub/quizzes/

Le Backend

Un projet évolutif pour monter en compétence : le combat de personnages

1. Avec vos compétences limitées en PHP et en sauvegardant les datas en session pour éviter d'utiliser une base de données créer un système de combat ressemblant à l'exemple vous pouvez reprendre ou améliorer mon exemple ou créer la vôtre avec le framework css que vous préférez http://anime-sanctuary.net/battle/ . L'adversaire est un robot vous devrez programmer son attaque pour qu'il riposte à chaque attaque. Si cela vous amuse vous pouvez animer les combats en javascript en faisant bouger les images ou en ajoutant des impacts mais ne perdez pas trop de temps là-dessus ce n'est pas le but de l'exercice.
2. Plutôt que de sauver les datas en session, branchez une base de données et modifiez votre code sans changer le rendu existant. Garder ainsi une trace de tous les personnages créés, proposez une liste des combattants déjà existant ou la possibilité d'en créer un nouveau et d'uploader une photo pour l'avatar et de la redimensionner pour qu'elle s'adapte au format. Sauvegarder également tous les combats. Rajouter un graphique avec les statistiques de victoire pour faire un classement des meilleurs combattants.
3. Plutôt que de travailler en procédurale refactoriser votre code pour travailler avec des objets liés à la base de données et inventer de nouvelles possibilités lors du combat pour le complexifier. Par exemple la possibilité de se soigner en utilisant le mana, ou de charger un sort, ou rajouter des % de chances d'esquiver un coup ou de faire un coup critique ou tout ce que vous pouvez imaginer vous devrez également améliorer le robot pour l'adversaire qu'il puisse lui aussi utiliser ces nouvelles compétences.
4. Plutôt que de recharger la page à chaque attaque utilisez l'ajax ou le fetch pour créer des combats plus dynamique avec une mini API en back qui vous répondra du html ou du json
5. Recommencer le projet avec le Framework Symfony quand vous le maitriserez

L'intérêt de l'exercice et de montrer qu'il existe une multitude de façon de résoudre un problème en fonction de l'étendue de ses connaissances en utilisant des méthodes plus ou moins lourdes et complexes. Un projet et souvent amener à évoluer au cours du temps, modifier, améliorer et maintenir un projet nécessite d'avoir une méthodologie de programmation flexible et robuste penser de manière à être facilement adaptable. Bien qu'il puisse paraitre complexe et fastidieux dans un premier temps de professionnaliser son développement les avantages en termes de productivité et de flexibilité sont rapidement perceptible. En améliorant le programme et en augmentant la qualité du code vous prendrez conscience des avantages de chaque évolution et pourrez décider en fonction de la complexité d'un projet de choisir une réponse adaptée au temps et au budget disponible.