Notes React/Projets Web

React -> partie visuelle du projet Web / interation etc… bibliothèques qui permet de gérer les interfaces utilisateurs

Dans le web, quelque soit le language ou le framework, c’est globalement la même chose

Permet de faire du responsive, temps de chargement, Navigation intuituve et plaisante et Mobile Like

Serveur Apache : serveur web // php -> langage typé

QUAND ON effectue requête http ->> Serveur PHP va renvoyer code HTML de la page ->> temps de chargement tout pourri

Simuler de la spontanéité de l’interface web

Tout ce code est chargé initialement, du code effectué coté client

Chargement initiale -> requête http

Js est exécuté dans naviguateur, logique n’est plus executée coté client

Js permet de manipuler la structure hmtl d’une page web (DOM)

On peut modifier contenu page sans exécuter requête http

Chrome -> moteur V8 permet d’executer du js

Node.js permet d’executer du Js en prod sur la machine

Ça passe d’un language serveur, ce n’est plus un simple langage de scripting

HTML/CSS interprété

Chocolatey et NVM permet de changer très rapidement de version si nos projets utilisent des vieilles version

Npm -> gestionnaire de paquet qui permet d’exe lib js sur Node (permet d’éviter de tout refaire)

En react tout est séparé, + clair, plus parlant alors qu’en JS il faut tout créer à la main c’es long et lourd

On retrouve des éléments html avec des éléments de javascript JSX

React contrôle totalité DOM (Structure HTML) / front d’un site contrôle des vue

Les alternatives : Angular // Vue JS

VueJS -> + rapide / - doc

Angular FrameWork -> PROPRE architecture de développement à respecter – pour dev cross platform – projet très lourd fonctionne bien librairie qui va gérer beaucoup de donné très bien

Pour petit projet React et Vue + plus rapide si petit projet

React et vue on va devoir installer tout un tas de bibliothèques externes pour faire ce qu’on veut

Angular il est prêt pour lancer une appli en prod direct san rien installer en plus

React de + en + utiliser avec typescript (type js) et Vue pas mal associer à symphony et PHP

Ces trois framework -> Finalité c’est de produire de l’html/css/js sur des gros projets quand on dit tel framework est plus rapide c’est qu’il produit du code hmtl/css/js plus clair et plus opti

Npm js org

aggreed -> Meilleur bibliothèque pour gérer les tableaux au monde

Rappel JS Bin :

-Version JS ES-6-7-8-9 ….

-Tester son code dans différent naviguateur

Bibliothèque Here -> Maps

Jamais mot var

« Développer.mozilla.org »

Dès que var il va le mettre tout en haut il va l’exe avant toute les autres ce qui peut poser problèmes -> Hoisting

A la place on va utiliser const et let

Const -> on ne peut plus réassigner on ne peut plus utiliser le =

Let -> on peut réassigner la variable

« ` » permet d’écrire des string sur plusieurs ligne

Création fonction :

function helloworld(name)

{

console.log("Hello", name);

}

const helloworldCarla = () => {

}

const RetourneMonNom = (nom) => {return nom};

const RetourneMonNom = (nom) => {return nom};

const UnNom = RetourneMonNom("Adrien");

console.log()

//export anonyme

export default RetourneMonNom; // Dispo pour tous les autres fichiers js du projet

//export nommé

export const myFunction = () => {};

//import anonyme (qui veut dire un export anonyme avant)

import NameOfDefaultImport from './'

//import nommé

import {myFunction} from './'

export nommé permet d’exporter des fonctions précises grâce à leur nom ( import préalables requis)

//import \* as MyAlias from './../

//myAlias.myFunction

Alias permet dans un projet où il y a beaucoup de fonctions avec même nom, d’être sûr d’utiliser les bonnes fonctions ( plus lisibles, + clair)

Class en JS :

class Person {

constructor(){

this.name = "Maximilien";

}

}

On peut même faire juste this.name pour déclarer variable

Extends pour héritage

Mot clé this à une portée différente que C#

class Person {

name = "toto";

sayHello = () => {

this.name;

}

}

// Si on avait mit Name dans un constructor normal this .name n’était pas la variable name crée au-dessus

const AnArray = [1, 2, 3];

const newArray = [...AnArray, 4, 5];

console.log(newArray);

Spread operator permet de mettre les valeurs d’un tableau dans un autre

Attention le javascript permet de tout faire on peut mettre un tableau dans un tableau de int car tout les variables sont considérés comme « any » pour « anything » on peut régler ça grâce au typescript

const toto = {

bonjour: "LaFrance",

bienvenue: "Chez Moi"

}

const tonton = {

...toto,

bienvenue: " Là - bas"

}

console.log("tonton", tonton);

si on enlève le spread de toto il colle directement l’objet

Destructuring

const array = [1, 2, 3]

const [a, b, c] = array

console.log(a)

console.log(c)

const Toto = {

bonjour: "La France",

bienvenue: " Chez Moi"

}

const {bonjour} = Toto

console.log(bonjour)

Sommet arbre App (Dans app il faut un élément unique au sommet de l’arbre, ne rajouter une deuxième div)