 307 – Réaliser des pages Web interactives

Rapport personnel

Version 1 du 19 juin 2021

Danilo Anzile

|  |  |
| --- | --- |
|  | Module du 17.05.2021 au 15.06.2021 |

Table des matières

[1 Introduction 1](#_Toc75009942)

[2 Tests technologiques 1](#_Toc75009943)

[2.1 Exercice 1 1](#_Toc75009944)

[2.1.1 Index.html 1](#_Toc75009945)

[2.1.2 IndexCtrl.js 2](#_Toc75009946)

[2.1.3 Teste technologique 2](#_Toc75009947)

[a) A quoi sert le « onload » ? 2](#_Toc75009948)

[b) « onclick » 2](#_Toc75009949)

[c) Interagir avec le html depuis le js 2](#_Toc75009950)

[2.2 Exerice 2 3](#_Toc75009951)

[2.2.1 Index.html 3](#_Toc75009952)

[2.2.2 indexCtrl.js 4](#_Toc75009953)

[2.2.3 Main.css 4](#_Toc75009954)

[2.2.4 Questions : 5](#_Toc75009955)

[2.2.5 Teste technologique 6](#_Toc75009956)

[a) Windows.confirm() 6](#_Toc75009957)

[b) Let 6](#_Toc75009958)

[2.3 Exercice 3 6](#_Toc75009959)

[2.3.1 Index.html 6](#_Toc75009960)

[2.3.2 Main.css 7](#_Toc75009961)

[2.3.3 Login\_Get.php 8](#_Toc75009962)

[2.3.4 Login\_POST.php 9](#_Toc75009963)

[2.3.5 Teste technologique 9](#_Toc75009964)

[a) Php 9](#_Toc75009965)

[2.4 Exercice 4 9](#_Toc75009966)

[2.5 Exercice 5 10](#_Toc75009967)

[2.5.1 Questions 10](#_Toc75009968)

[2.6 Exercice 6 11](#_Toc75009969)

[2.6.1 Index.html 11](#_Toc75009970)

[2.6.2 indexCtrl.js 12](#_Toc75009971)

[2.6.3 main.css 13](#_Toc75009972)

[2.6.4 Teste technologique 14](#_Toc75009973)

[a) Switch 14](#_Toc75009974)

[b) Date() 14](#_Toc75009975)

[2.7 Exercice 7 14](#_Toc75009976)

[2.7.1 Index.html 14](#_Toc75009977)

[2.7.2 indexCtrl.js 15](#_Toc75009978)

[2.7.3 Teste technologique 16](#_Toc75009979)

[a) Fori 16](#_Toc75009980)

[b) While 16](#_Toc75009981)

[c) DoWhile 16](#_Toc75009982)

[2.8 Exercice 8 17](#_Toc75009983)

[2.8.1 Index.html 17](#_Toc75009984)

[2.8.2 IndexCtrl.js 17](#_Toc75009985)

[2.8.3 Teste technologique 18](#_Toc75009986)

[a) Const 18](#_Toc75009987)

[b) Json 18](#_Toc75009988)

[2.9 Exercice 9 19](#_Toc75009989)

[2.9.1 Index.html 19](#_Toc75009990)

[2.9.2 indexCtrl.js 19](#_Toc75009991)

[2.10 Exercice 10 22](#_Toc75009992)

[2.10.1 Index.html 22](#_Toc75009993)

[2.10.2 IndexCtrl.js 23](#_Toc75009994)

[2.10.3 Worker.js 25](#_Toc75009995)

[2.10.4 Personne.js 26](#_Toc75009996)

[2.11 Exercice 11 26](#_Toc75009997)

[2.11.1 Index.html 27](#_Toc75009998)

[2.11.2 indexCtrl.js 28](#_Toc75009999)

[2.11.3 Personne.js 30](#_Toc75010000)

[2.11.4 Worker.js 30](#_Toc75010001)

[2.11.5 Teste technologique 31](#_Toc75010002)

[a) Class 31](#_Toc75010003)

[b) This 31](#_Toc75010004)

[2.12 Exercice 12 31](#_Toc75010005)

[2.12.1 Index.html 31](#_Toc75010006)

[2.12.2 IndexCtrl.js 32](#_Toc75010007)

[2.12.3 Teste technologique 33](#_Toc75010008)

[a) Closures 33](#_Toc75010009)

[2.13 Exercice 13 33](#_Toc75010010)

[2.13.1 Index.Html 33](#_Toc75010011)

[2.13.2 indexCtrl.js 34](#_Toc75010012)

[2.13.3 personne.js 36](#_Toc75010013)

[2.13.4 worker.js 36](#_Toc75010014)

[2.14 Exercice 14 37](#_Toc75010015)

[2.14.1 Qu’est-ce que c’est ? 37](#_Toc75010016)

[2.14.2 Pourquoi choisir jQuery ? 37](#_Toc75010017)

[2.14.3 Qui l’utilise ? 37](#_Toc75010018)

[2.14.4 Popularité 37](#_Toc75010019)

[2.14.5 Facile à prendre en main 37](#_Toc75010020)

[2.14.6 L’objet jQuery 38](#_Toc75010021)

[2.14.7 L’événement « ready » 38](#_Toc75010022)

[2.14.8 Traverser les nœuds du DOM 38](#_Toc75010023)

[2.14.9 Enchaînement 38](#_Toc75010024)

[2.14.10 Enchaînement transversal 39](#_Toc75010025)

[2.14.11 Manipulation avec .after() 39](#_Toc75010026)

[2.14.12 Manipulation avec .append() 39](#_Toc75010027)

[2.14.13 Manipulation avec .css() 39](#_Toc75010028)

[2.14.14 Sélecteurs HTML 39](#_Toc75010029)

[2.14.15 Evénement .submit() 40](#_Toc75010030)

[2.14.16 Evénement .click() 40](#_Toc75010031)

[2.14.17 Evénement .hover() 40](#_Toc75010032)

[2.14.18 Evénements en live 41](#_Toc75010033)

[2.14.19 Effet .slideToggle() 41](#_Toc75010034)

[2.14.20 Effet .animate() 41](#_Toc75010035)

[2.14.21 Effet .hide() et .show() 41](#_Toc75010036)

[2.14.22 Ajax – lire XML 41](#_Toc75010037)

[2.14.23 Ajax – lire JSON 42](#_Toc75010038)

[2.14.24 Ajax – .load() 42](#_Toc75010039)

[2.14.25 jQuery CDN 42](#_Toc75010040)

[2.15 Exercice 15 42](#_Toc75010041)

[2.15.1 jQuery 42](#_Toc75010042)

[2.15.2 JavaScript 42](#_Toc75010043)

[2.15.3 Teste technologique 43](#_Toc75010044)

[2.16 Exercice 18 43](#_Toc75010045)

[2.16.1 Index.js 43](#_Toc75010046)

[2.16.2 httpServ.js 44](#_Toc75010047)

[2.16.3 Teste technologique 45](#_Toc75010048)

[2.17 Exercice 19 45](#_Toc75010049)

[2.17.1 indexCtrl.js 45](#_Toc75010050)

[2.17.2 httpServ.js 46](#_Toc75010051)

[2.17.3 Teste technologique 47](#_Toc75010052)

[2.18 Exercice 21 47](#_Toc75010053)

[2.18.1 Street.html 47](#_Toc75010054)

[2.18.2 streetCtrl.js 48](#_Toc75010055)

[2.18.3 Teste technologique 49](#_Toc75010056)

[3 LWG 49](#_Toc75010057)

[3.1 Analyse 49](#_Toc75010058)

[3.2 Conception 50](#_Toc75010059)

[3.2.1 Explication supplémentaire 50](#_Toc75010060)

[3.3 Implémentation 50](#_Toc75010061)

[3.3.1 Index.php 50](#_Toc75010062)

[3.3.2 Lwg-history.php 50](#_Toc75010063)

[3.3.3 Events.php 50](#_Toc75010064)

[3.3.4 Gallery.php 50](#_Toc75010065)

[3.3.5 Contact.php 51](#_Toc75010066)

[3.3.6 Head.php 51](#_Toc75010067)

[3.3.7 Nav-bar.php 51](#_Toc75010068)

[3.3.8 ScriptEndPage.php 51](#_Toc75010069)

[3.3.9 Defines.php 51](#_Toc75010070)

[3.3.10 Instagram\_basic\_display\_api.php 51](#_Toc75010071)

[3.3.11 instagramAPI.php 51](#_Toc75010072)

[3.3.12 mainCtrl.js 51](#_Toc75010073)

[3.3.13 Database.php 51](#_Toc75010074)

[3.4 Hébergement 51](#_Toc75010075)

[3.5 Fonctionnement 52](#_Toc75010076)

[3.6 API 52](#_Toc75010077)

[3.6.1 Ce que je vais utiliser de cette API 52](#_Toc75010078)

[3.6.2 Lien vers documentations API 52](#_Toc75010079)

[3.6.3 Recuperation du Token LongLife 52](#_Toc75010080)

[3.7 Le site principal 53](#_Toc75010081)

[3.7.1 Page d’accueil 53](#_Toc75010082)

[3.7.2 Page History 53](#_Toc75010083)

[3.7.3 Page Events 53](#_Toc75010084)

[3.7.4 Page Gallery 53](#_Toc75010085)

[3.7.5 Page de contact 54](#_Toc75010086)

[4 Intranet.LWG 55](#_Toc75010087)

[4.1 Analyse 55](#_Toc75010088)

[4.2 Conception 56](#_Toc75010089)

[4.3 Implémentation 56](#_Toc75010090)

[4.3.1 Index.php 56](#_Toc75010091)

[4.3.2 Database.php 56](#_Toc75010092)

[4.3.3 Accueil.php 56](#_Toc75010093)

[4.3.4 Head.php 56](#_Toc75010094)

[4.3.5 scriptBasDePage.php 56](#_Toc75010095)

[4.3.6 mainCtrl.js 56](#_Toc75010096)

[4.4 Hébergement 57](#_Toc75010097)

[4.5 Fonctionnement 57](#_Toc75010098)

[4.6 Le site 57](#_Toc75010099)

[4.6.1 Page de login 57](#_Toc75010100)

[5 Conclusion 57](#_Toc75010101)

# Introduction

Durant ce module nous travaillerons sur ces divers points :

* Rappel sur les bases des techniques du Web côté client (HTML, CSS et autres)
* Tester un environnement de travail efficace pour le développement web
* Comprendre les bases du développement web côté client (intelligence locale) en étudiant un langage de script et une ou l’autre bibliothèque adaptée
* Comprendre les bases de la consommation de services web existants et tester en pratique par des requêtes HTTP
* Comprendre les bases d’un découpage MVC d’une application web côté client et tester en pratique
* Mettre en forme des données JSON ou XML provenant d’un serveur.
* Comprendre la problématique de l’adaptation des applications Web interactives à des écrans de diverses natures et mettre en pratique

Créer et améliorer un projet global *« accompagné »*, basé sur les techniques vues pendant le module

# Tests technologiques

## Exercice 1

### Index.html

<!DOCTYPE html>

<html>

  <!--

   But :

   Auteur : Danilo Anzile

   Date :   17.05.2021 / V1.0

   -->

  <!-- entête de la page -->

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link rel="icon" href="#" />

    <link rel="stylesheet" href="css/main.css" />

    <script type="text/javascript" src="js/indexCtrl.js" async></script>

    <title>Ex01 — Exercice de base</title>

  </head>

  <!-- corps de la page avec enregistrement d'écouteur dans le JS-->

  <body onload="initCtrl()">

    <div id="container">

      <p>Cliquez sur le bouton qui contient votre prénom (js) !</p>

      <button id="testez">Teste-moi, Danilo</button>

      <p id="info">&nbsp;</p>

    </div>

  </body>

  <!-- corps de la page avec enregistrement d'écouteur dans le HTML de la page-->

  <!-- Ajoutez l'écouteur testez() directement sur le bouton-->

  <!-- Jouez avec les commentaires pour executer un <body> après l'autre-->

  <body>

    <div id="container">

      <p>Cliquez sur le bouton qui contient votre prénom (html) !</p>

      <button id="testez" onclick="testez()">Teste-moi, Danilo</button>

      <p id="info">&nbsp;</p>

    </div>

  </body>

</html>

### IndexCtrl.js

/\*

  But :

  Auteur : Danilo Anzile

  Date :   17.05.2021 / V1.0

\*/

function initCtrl() {

  // Ecouteur du bouton "Testez-moi..."

  document.getElementById("testez").addEventListener("click", testez);

}

function testez() {

   document.getElementById("info").innerHTML =

      "C'est <b>Danilo Anzile</b> qui a pressé le bouton !";

}

### Teste technologique

#### A quoi sert le « onload » ?

Il permet de charger le .js lorsque la page est chargée.

#### « onclick »

Permet d’effectuer une action lorsque nous cliquons sur un bouton.

#### Interagir avec le html depuis le js

  document.getElementById("testez").addEventListener("click", testez);

Permet de récupérer l’élément qui a comme id « Testez » et lui rajoute un écouteur.

document.getElementById("info").innerHTML =

      "C'est <b>Danilo Anzile</b> qui a pressé le bouton !";

Permet d’insérer du texte dans ou l’id est « info ».

## Exerice 2

### Index.html

<!DOCTYPE html>

<html>

<!--

   But :

   Auteur : Danilo Anzile

   Date :   17.05.2021 / V1.0

   -->

<!-- entête de la page -->

<head>

  <meta charset="UTF-8" />

  <meta name="viewport" content="width=device-width, initial-scale=1.0" />

  <link rel="icon" href="#" />

  <link rel="stylesheet" href="css/main.css" />

  <script type="text/javascript" src="js/indexCtrl.js" async></script>

  <title>Ex02 — Exercice de login</title>

</head>

<!-- corps de la page -->

<body>

  <div id="container">

    <!-- début du formulaire -->

    <form class="user-form">

      <fieldset>

        <legend>Identification:</legend>

        <div class="field">

          <label for="username">Nom d'utilisateur:</label>

          <input type="text" size="30" id="username" placeholder="un nom svp" autofocus />

        </div>

        <div class="field">

          <label for="password">Mot de passe:</label>

          <input type="password" size="30" id="password" placeholder="un mot de passe svp" />

        </div>

        <input type="button" id="valider" value="Valider" onclick="validerUtilisateur();" />

      </fieldset>

    </form>

    <!-- fin du formulaire -->

  </div>

</body>

</html>

### indexCtrl.js

/\*

  But :

  Auteur : Danilo Anzile

  Date :   17.05.2021 / V1.0

\*/

function validerUtilisateur() {

  // récupère les objets InputText du formulaire HTML

  // let userComp = document.getElementById("username");

  // let passwordComp =  document.getElementById("password");

  // récupère les valeurs de ces objets

  // let username = userComp.value;

  // let password = passwordComp.value;

  // Plus court

  let username = document.getElementById("username").value;

  let password = document.getElementById("password").value;

  // affichage pour débogguer avec console.log

  // message possible  utilisateur: titi  mot de passe: emf

  console.log("utilisateur:" + username + "mot de passe:" + password);

  // validation du user et password par l'affichage d'une popup

  if (username.toLowerCase() === "admin" && password === "emf123") {

    window.confirm("Validation OK.")

  } else {

    window.confirm("Utilisateur ou mot de passe incorrect !!!")

  }

}

### Main.css

/\*

  But :

  Auteur : Danilo Anzile

  Date :   17.05.2021 / V1.0

\*/

body {

  background-color: grey;

}

#container {

  background-color: white;

  width: 90%;

  padding: 1em;

  border: 1px solid black;

  margin: auto;

  font-family: Verdana, Arial, serif;

}

.user-form {

}

.user-form .field {

  padding: 0.5em;

  background: lightgray;

}

.user-form .field label {

  display: inline-block;

  width: 20%;

}

.user-form .field input {

  display: inline-block;

}

#valider {

  margin: 1%;

  border-radius: 1cm;

  background-color: lightgreen;

  width: 10%;

}

### Questions :

* À quoi sert l’attribut *« placeholder »* ?

Permet de renseigner un texte par défaut dans le champ du formulaire.

* À quoi sert l’attribut *« autofocus »* ?

C’est un attribut booléen qui spécifie qu’un input élément doit automatiquement obtenir le focus lors du chargement de la page.

* Y a-t-il déjà du JavaScript dans ce code ?

Oui

* Quelle est la différence entre le bouton de l'ex01 et celui-ci ?

Celui-ci, on ne lui met pas un écouteur sur le bouton, on lui dit directement d’appeler la méthodes qu’il faut.

### Teste technologique

#### Windows.confirm()

Permet d’afficher un popup.

#### Let

Permet d’initier une variable. Il est mieux d’utilisé let que var car il gère mieux les divers types de base.

## Exercice 3

### Index.html

<!DOCTYPE html>

<html>

  <!--

   But :

   Auteur : Danilo Anzile

   Date :   17.05.2021 / V1.0

   -->

  <!-- entête de la page -->

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link rel="icon" href="#" />

    <link rel="stylesheet" href="css/main.css" />

    <script type="text/javascript" src="js/indexCtrl.js" async></script>

    <title>Ex03 — Exercice de login</title>

  </head>

  <!-- corps de la page -->

  <body>

    <div id="container">

      <!-- début du formulaire -->

      <form

        class="user-form"

        action="login\_POST.php"

        method="POST"

      >

        <fieldset>

          <legend>Identification:</legend>

          <div class="field">

            <label for="username">Nom d'utilisateur:</label>

            <input

              type="text"

              size="30"

              id="username"

              placeholder="un nom svp"

              autofocus

              name="username"

            />

          </div>

          <div class="field">

            <label for="password">Mot de passe:</label>

            <input

              name="password"

              type="password"

              size="30"

              id="password"

              placeholder="un mot de passe svp"

            />

          </div>

          <div class="button">

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

          </div>

        </fieldset>

      </form>

      <!-- fin du formulaire -->

    </div>

  </body>

</html>

### Main.css

/\*

  But :

  Auteur : Danilo Anzile

  Date :   17.05.2021 / V1.0

\*/

body {

    background-color: grey;

  }

  #container {

    background-color: white;

    width: 90%;

    padding: 1em;

    border: 1px solid black;

    margin: auto;

    font-family: Verdana, Arial, serif;

  }

  .user-form {

  }

  .user-form .field {

    padding: 0.5em;

    background: lightgray;

  }

  .user-form .field label {

    display: inline-block;

    width: 20%;

  }

  .user-form .field input {

    display: inline-block;

  }

  .user-form .button {

    margin-top: 1%;

    margin-left: 40%;

  }

  .user-form input[type=submit]{

    color: white;

    background-color: lightgray;

    width: 20%;

  }

### Login\_Get.php

<?php

// test si on a reçu une donnée de formulaire nommée "username"

if (isset($\_GET['username'])) {

  // récupération des données transmises dans des variables locales

  $username = strtolower($\_GET['username']);

  $password = $\_GET['password'];

  // affichage des infos reçues

  echo "username: ".$username."</br>";

  echo "password: ".$password."</br>";

  // test username et mot de passe

  if (($username == "admin") && ($password == "emf123")) {

    echo "<script>alert('Validation OK');</script>";

  } else {

    echo "<script>".

            "alert('Utilisateur ou mot de passe incorrect !!!');".

         "</script>";

  }

}

?>

### Login\_POST.php

<?php

// test si on a reçu une donnée de formulaire nommée "username"

if (isset($\_POST['username'])) {

  // récupération des données transmises dans des variables locales

  $username = strtolower($\_POST['username']);

  $password = $\_POST['password'];

  // affichage des infos reçues

  echo "username: ".$username."</br>";

  echo "password: ".$password."</br>";

  // test username et mot de passe

  if (($username == "admin") && ($password == "emf123")) {

    echo "<script>alert('Validation OK');</script>";

  } else {

    echo "<script>".

            "alert('Utilisateur ou mot de passe incorrect !!!');".

         "</script>";

  }

}

?>

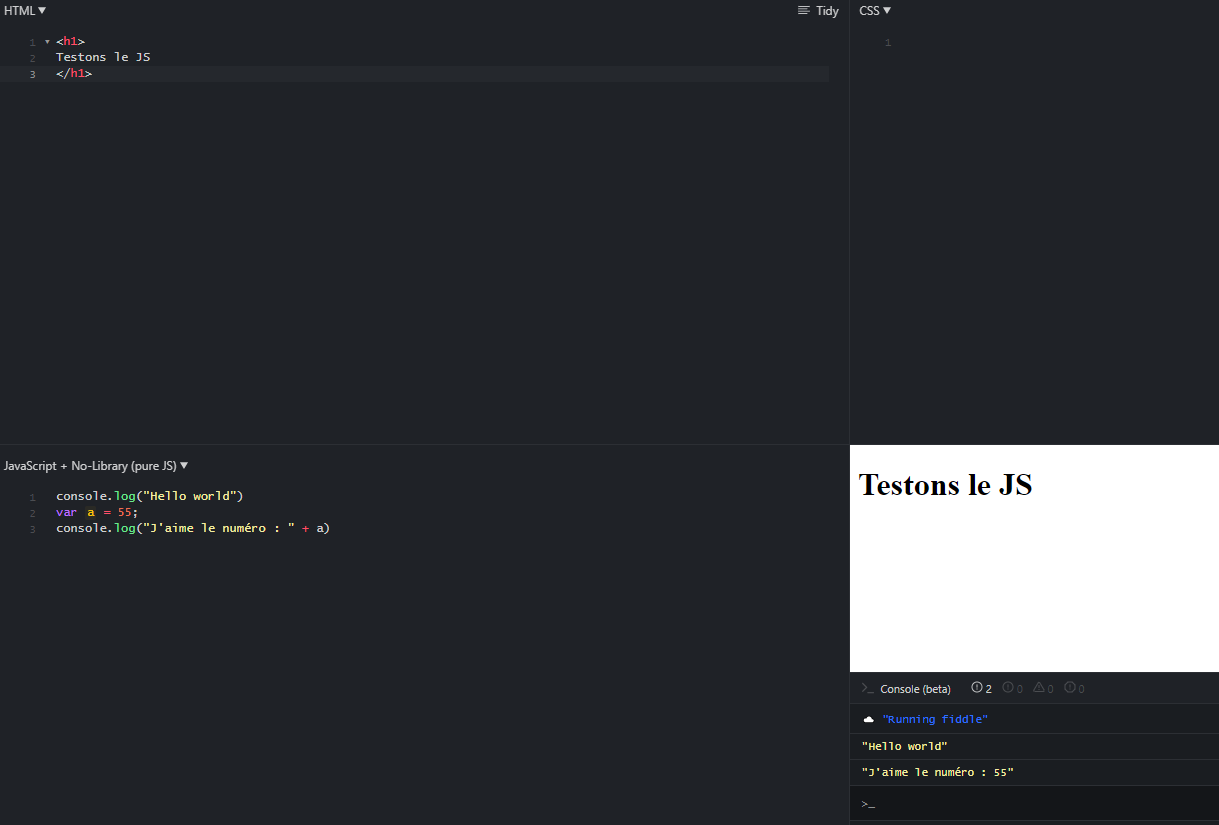
### Teste technologique

#### Php

C’est un langage qui est utilisé du côté serveur. Cela veut dire qu’il nous permet d’envoyé une page html en base a ce dont l’utilisateur a besoin et de ses droits.

## Exercice 4

Durant cet exercice, nous devions juste découvrir le site <http://jsfiddle.net/>



## Exercice 5

### Questions

1. Effacer le contenu de la console avec *« console.clear() »* ;

console.clear();

1. Créer une variable nommée *« a »*;

var a;

1. Afficher le contenu de *« a »*avec console.log;

console.log(a);

1. Stocker la valeur 15 dans cette variable ;

a = 15;

1. Afficher le contenu de cette variable dans la console sous la forme *« Ma variable a = ? »*

console.log("Ma variable a =" + a);

1. Créer une variable nommée *« b »* et lui assigner directement la valeur 9 ;

var b = 9;

1. Afficher le contenu de cette variable dans la console sous la forme *« Ma variable b = ? »*

console.log("Ma variable b =" + b);

1. Faire l’addition de ces 2 variables en affichant directement le résultat dans la console sous cette forme : « 15 + 9 = ? » ; (essayer d’utiliser un littéral avec `…${…}…`)

console.log(`${a} + ${b} = ${a + b}`);

1. Compléter en faisant de même pour une soustraction, une multiplication et une division des deux variables ;

console.log(`${a} - ${b} = ${a - b}`);

console.log(`${a} \* ${b} = ${a \* b}`);

console.log(`${a} / ${b} = ${a / b}`);

1. Stocker *« Bonjour »* dans la variable a ;

a = "Bonjour";

1. Stocker *« les amis »* dans la variable b ;

b = "les amis";

1. Afficher *« bonjour les amis »* dans la console en concaténant les variables ;

console.log(a + " " + b);

1. Stocker *« true »* dans la variable a ;

a = true;

1. Stocker *« false »* dans la variable b ;

b = false;

1. Effectuer une opération AND entre les 2 variables et afficher le résultat sous cette forme *« true AND false = ? » ;*

console.log(`${a} && ${b} = ${a && b}`);

1. Effectuer une opération OR entre les 2 variables et afficher le résultat sous cette forme *« true OR false = ? » ;*

console.log(`${a} OR ${b} = ${a || b}`);

1. Stocker la date du jour dans la variable a avec new Date();

a = new Date();

1. Calculer une nouvelle date dans la variable b qui est 61 jours avant la date courante (utilisation getDate, setDate)

b = new Date();

b.setTime(a.getTime() - 61 \* 24 \* 3600000);

1. Afficher les dates suivantes en vous aidant de *« toLocaleString() et toLocaleDateString() »* :

**12/05/2020 à 17:05:57**

**12/05/2020**

**29/02/2020 à 16:56:06 // il y a 61 jours**

**12.05.2020 // format Suisse romand**

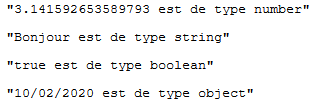
a.toLocaleString();

a.toLocaleDateString();

a.setTime(a.getTime() - 24 \* 60 \* 60 \* 1000 \* 61);

a.toLocaleString();

a.toLocaleDateString("fr-ch");

1. Stocker la valeur de Math.PI dans a, *« bonjour »* dans b, créer et assigner true dans c, créer et assigner la date courante dans d, puis afficher le type pour les 4 variables :  
   

console.log(a + " est de type " + typeof(a));

console.log(b + " est de type " + typeof(b));

console.log(c + " est de type " + typeof(c));

console.log(d.toLocaleDateString() + " est de type " + typeof(d));

## Exercice 6

### Index.html

<!DOCTYPE html>

<html>

  <!--

   But :

   Auteur : Danilo Anzile

   Date :   18.05.2021 / V1.0

   -->

  <!-- entête de la page -->

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link rel="icon" href="#" />

    <link rel="stylesheet" href="css/main.css" />

    <script type="text/javascript" src="js/indexCtrl.js" async></script>

    <title>Ex01 — Exercice de base</title>

  </head>

  <!-- corps de la page avec enregistrement d'écouteur dans le JS-->

  <body onload="initCtrl()">

    <div id="container">

      <p>

        Cliquez sur le bouton ci-dessous pour découvrir quel jour nous sommes

      </p>

      <button onclick="afficherJourSemaine();" id="testez">Teste-moi</button>

      <button onclick="afficherJourSemaineTab();" id="testez-tab">Teste-moi</button>

      <p id="info">&nbsp;</p>

      <p id="infoTab">&nbsp;</p>

    </div>

  </body>

</html>

### indexCtrl.js

/\*

  But :

  Auteur : Danilo Anzile

  Date :   18.05.2021 / V1.0

\*/

function initCtrl() {

  // Ecouteur du bouton "Testez-moi..."

  document.getElementById("testez").addEventListener("click", testez);

  document.getElementById("testez-tab").addEventListener("click", testez);

}

function afficherJourSemaine() {

  let dateCourante = new Date();

  let intJourDeLaSemaine = dateCourante.getDay();

  let jourDeLaSemaine;

  switch (intJourDeLaSemaine) {

    case 0:

      jourDeLaSemaine = "dimanche";

      break;

    case 1:

      jourDeLaSemaine = "lundi";

      break;

    case 2:

      jourDeLaSemaine = "mardi";

      break;

    case 3:

      jourDeLaSemaine = "mercredi";

      break;

    case 4:

      jourDeLaSemaine = "jeudi";

      break;

    case 5:

      jourDeLaSemaine = "vendredi";

      break;

    case 6:

      jourDeLaSemaine = "samedi";

      break;

  }

  document.getElementById("info").innerHTML = "On est " +jourDeLaSemaine;

}

function afficherJourSemaineTab() {

  let dateCourante = new Date();

  let intJourDeLaSemaine = dateCourante.getDay();

  let jourDeLaSemaine = ['dimanche','lundi','mardi','mercredi','jeudi','vendredi','samedi'];

  let resultat;

  for (let i = 0; i<jourDeLaSemaine.length; i++){

    if (i === intJourDeLaSemaine){

      resultat = jourDeLaSemaine[i];

    }

  }

  document.getElementById("infoTab").innerHTML = "On est " + resultat + " [tab]"

}

### main.css

/\*

  But :

  Auteur : Danilo Anzile

  Date :   17.05.2021 / V1.0

\*/

body {

  background-color: grey;

}

#container {

  background-color: white;

  width: 90%;

  padding: 1em;

  border: 1px solid black;

  margin: auto;

  font-family: Verdana, Arial, serif;

}

#testez {

  color: black;

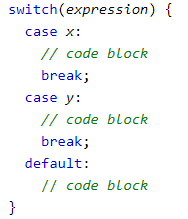
  background-color: white;

  border-radius: 0.1cm;

}

### Teste technologique

#### Switch



https://www.w3schools.com/js/js\_switch.asp

#### Date()

C’est une classe qui nous permet de travailler avec des dates.

## Exercice 7

### Index.html

<!DOCTYPE html>

<html>

  <!--

   But :

   Auteur : Danilo Anzile

   Date :   18.05.2021 / V1.0

   -->

  <!-- entête de la page -->

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link rel="icon" href="#" />

    <link rel="stylesheet" href="css/main.css" />

    <script type="text/javascript" src="js/indexCtrl.js" async></script>

    <title>Ex07 — Exercice de base</title>

  </head>

  <!-- corps de la page avec enregistrement d'écouteur dans le JS-->

  <body onload="initCtrl()">

    <div id="container">

      <p>

        Cliquez sur le bouton ci-dessous pour découvrir quel jour nous sommes

      </p>

      <button onclick="forTest();" id="forTest">For</button>

      <button onclick="whileTest();" id="whileTest">While</button>

      <button onclick="doWhileTest();" id="doWhileTest">Do While</button>

      <p id="for-Test">&nbsp;</p>

      <p id="while-Test">&nbsp;</p>

      <p id="doWhile-Test">&nbsp;</p>

    </div>

  </body>

</html>

### indexCtrl.js

/\*

  But :

  Auteur : Danilo Anzile

  Date :   18.05.2021 / V1.0

\*/

function initCtrl() {

  // Ecouteur des bouton

  document.getElementById("forTest").addEventListener("click", forTest);

  document.getElementById("whileTest").addEventListener("click", whileTest);

  document.getElementById("doWhileTest").addEventListener("click", doWhileTesT);

}

function forTest() {

  document.getElementById("for-Test").innerHTML =

    "for (let i=0; i<5; i++){...} </br>";

  for (let i = 0; i < 5; i++) {

    document.getElementById("for-Test").innerHTML += "i = " + i + "</br>";

  }

  document.getElementById("for-Test").innerHTML +=

    "--> à utiliser si o sait que l'on veut itérer x fois (x connu avant de commencer la boucle) ";

}

function whileTest() {

  document.getElementById("while-Test").innerHTML = "while(i<5){...};</br>";

  let i = 0;

  while (i < 5) {

    document.getElementById("while-Test").innerHTML += "i = " + i + "</br>";

    i++;

  }

  document.getElementById("while-Test").innerHTML +=

    "--> à utiliser si on ne sait pas le nombre d'itérations au démarrage de la boucle";

}

function doWhileTesT() {

  document.getElementById("doWhile-Test").innerHTML = "do{...} while(i<5);</br>";

  let i = 0;

  do {

    document.getElementById("doWhile-Test").innerHTML += "i = " + i + "</br>";

    i++;

  } while (i < 5);

  document.getElementById("doWhile-Test").innerHTML +=

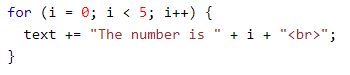
    "--> à utiliser si on ne sait pas le nombre d'itérations au démarrage de la boucle mais avec un passage obligatoire";

}

### Teste technologique

#### Fori

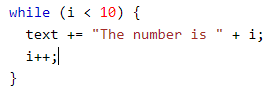
Comme en Java, la boucle fori permet d’incrémenter un nombre et d’effectuer une action tant que le nombre est respecter la condition du milieu.



https://www.w3schools.com/js/js\_loop\_for.asp

#### While

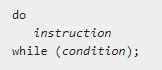
Effectue une action tant que la condition entre parenthèse est vraie.



https://www.w3schools.com/js/js\_loop\_while.asp

#### DoWhile

Effectue une première fois l’opération puis contrôle que la condition est vraie. De toute façon les instructions sont faites au moins 1 fois.



https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Statements/do...while

## Exercice 8

### Index.html

<!DOCTYPE html>

<html>

  <!--

   But :

   Auteur : Danilo Anzile

   Date :   18.05.2021 / V1.0

   -->

  <!-- entête de la page -->

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link rel="icon" href="#" />

    <link rel="stylesheet" href="css/main.css" />

    <script type="text/javascript" src="js/indexCtrl.js" async></script>

    <title>Ex08</title>

  </head>

  <!-- corps de la page avec enregistrement d'écouteur dans le JS-->

  <body onload="initCtrl()">

    <div id="container">

      <p>Cliquez sur le bouton pour parcourir un tableau JSON</p>

      <button onclick="parcourirUnTableauJSON() ;" id="btn">

        Parcourir un tableau JSON

      </button>

      <p id="info">&nbsp;</p>

    </div>

  </body>

</html>

### IndexCtrl.js

/\*

  But :

  Auteur : Danilo Anzile

  Date :   18.05.2021 / V1.0

\*/

function parcourirUnTableauJSON() {

  const json = {

      personnes: [

        {prenom: "John", nom: "Doe", age: 44},

        {prenom: "Anna", nom: "Smith", age: 32},

        {prenom: "Peter", nom: "Jones", age: 29}

      ]

    };

    let txt = "";

  for(let i = 0; i< json.personnes.length; i++){

    let personne = json.personnes[i];

    txt += i + ". " + personne.prenom +" "+ personne.nom +" "+  personne.age +"</br>";

  }

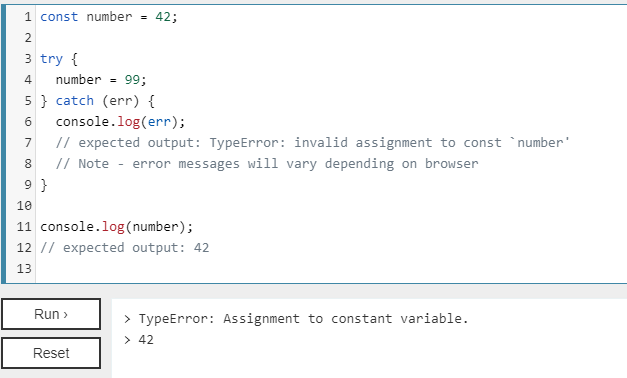
  document.getElementById("info").innerHTML= txt;

}

### Teste technologique

#### Const

Cette déclaration permet de créer une constante qui est accessible uniquement en lecture. Une valeur d’une constante ne peut pas être modifié.



https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Statements/const

#### Json

C’est un format de données textuelles dérivé de la notation des objets venu de javascript. Il permet de représenter des données structurées comme le permet XML.

Dans notre exercice, nous avons stocker des personnes qui chaque une a un nom, un prénom et un âge.

## Exercice 9

### Index.html

<!DOCTYPE html>

<html>

  <!-- entête de la page -->

   <head>

      <meta charset="UTF-8" />

      <meta name="viewport" content="width=device-width, initial-scale=1.0" />

      <link rel="icon" href="#"/>

      <link rel= "stylesheet" href= "css/main.css" >

      <script type="text/javascript" src="js/indexCtrl.js"></script>

      <title>Ex09 — POO - demo objet</title>

   </head>

   <!-- corps de la page -->

   <body>

    <div id="container">

      <p>Cliquez sur le bouton pour créer découvrir la création d'objets</p>

      <button onclick="creerDesObjetsJSON();">simplement JSON</button>

      <button onclick="creerDesObjetsObject();">avec Object</button>

      <button onclick="creerDesObjetsFonction();">avec objet fonction</button>

      <button onclick="creerDesObjetsClassePrototype();">avec fonction prototype</button>

      <button onclick="creerDesObjetsClasse();">avec classe</button>

      <p id="info">&nbsp;</p>

    </div>

  </body>

</html>

### indexCtrl.js

//

//  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* JSON \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//

function creerDesObjetsJSON() {

  console.log("---------------JSON-------------------------");

  // créer un objet en JSON

  let p1 = {

    prenom: "Jules",

    nom: "Tartampion",

    age: 25,

  };

  console.log(p1);

  // ajouter deux méthodes

  p1.toString = function () {

    return this.prenom + " " + this.nom;

  };

  p1["toStringAge"] = function () {

    return this.prenom + " " + this.nom + " (" + this.age + ")";

  };

  console.log(p1);

  // let txt = p1.toString() + "<br>" + p1.toStringAge();

  let txt = p1 + "<br>" + p1.toStringAge();

  document.getElementById("info").innerHTML = txt;

}

//

//  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Object \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//

function creerDesObjetsObject() {

  console.log("-----------------Object----------------------");

  // créer un objet avec la fonction Object

  const p1 = new Object();

  p1.prenom = "Juliette";

  p1.nom = "Tartampion";

  p1.age = 23;

  console.log(p1);

  // ajouter deux méthodes

  p1.toString = function () {

    return this.prenom + " " + this.nom;

  };

  p1["toStringAge"] = function () {

    return this.prenom + " " + this.nom + " (" + this.age + ")";

  };

  console.log(p1);

  //        var txt = p1.toString() + "<br>" + p1.toStringAge();

  let txt = p1 + "<br>" + p1.toStringAge();

  document.getElementById("info").innerHTML = txt;

}

//

//  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Objet fonction \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//

function Personne(prenom, nom, age) {

  this.prenom = prenom;

  this.nom = nom;

  this.age = age;

  this.toString = function () {

    return this.prenom + " " + this.nom + " (" + this.age + ")";

  };

}

function creerDesObjetsFonction() {

  console.log("-----------------objet fonction------------------");

  let p1 = new Personne("JuJu", "Tartampion", 40);

  console.log(p1);

  let p2 = new Personne("Julie", "Tartampion", 32);

  console.log(p2);

  let txt = p1 + "<br>" + p2;

  document.getElementById("info").innerHTML = txt;

}

//

//  \*\*\*\*\*\*\*\*\*\*\*\*Function pseudo-classe => fonction-prototype \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//

function Professeur(prenom, nom, age) {

  this.prenom = prenom;

  this.nom = nom;

  this.age = age;

}

Professeur.prototype.toString = function () {

  return this.prenom + " " + this.nom + " (" + this.age + ")";

};

function creerDesObjetsClassePrototype() {

  console.log("-----------------prototype----------------------");

  let p1 = new Professeur("Jules", "Tartampion", 25);

  console.log(p1);

  let p2 = new Professeur("Juliette", "Tartampion", 24);

  console.log(p2);

  let txt = p1 + "<br>" + p2;

  document.getElementById("info").innerHTML = txt;

}

//

//  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Class \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//

class Eleve {

  constructor(prenom, nom, age) {

    this.prenom = prenom;

    this.nom = nom;

    this.age = age;

  }

  toString() {

    return this.prenom + " " + this.nom + " (" + this.age + ")";

  }

}

function creerDesObjetsClasse() {

  console.log("-----------------Class----------------------");

  let p1 = new Eleve("Julien", "Tartampion", 18);

  console.log(p1);

  let p2 = new Eleve("Julia", "Tartampion", 22);

  console.log(p2);

  let txt = p1 + "<br>" + p2;

  document.getElementById("info").innerHTML = txt;

}

## Exercice 10

### Index.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link rel="icon" href="#" />

    <link rel="stylesheet" href="css/main.css" />

    <title>Ex10 - POO - prototype</title>

    <script src="js/personne.js"></script>

    <script src="js/worker.js"></script>

    <script src="js/indexCtrl.js"></script>

  </head>

  <body>

    <div id="container">

      <form class="user-form">

        <fieldset>

          <legend>

            Veuillez introduire une nouvelle personne ou sélectionner une

            existante&nbsp;:

          </legend>

          <div class="field">

            <label for="prenom">Prénom:</label>

            <input

              type="text"

              size="20"

              id="prenom"

              placeholder="prénom"

              required="required"

              autofocus

            />

          </div>

          <div class="field">

            <label for="nom">Nom:</label>

            <input

              type="text"

              size="20"

              id="nom"

              placeholder="nom"

              required="required"

            />

          </div>

          <div class="field">

            <label for="nom">Âge:</label>

            <input

              type="text"

              size="20"

              id="age"

              placeholder="âge"

              required="required"

            />

          </div>

          <div class="button">

            <input type="button" value="Ajouter" onclick="ajouter();" />

            <input type="button" value="Supprimer" onclick="supprimer();" />

            <input type="reset" value="Nettoyer" />

          </div>

        </fieldset>

      </form>

      <div id="info"></div>

    </div>

  </body>

</html>

### IndexCtrl.js

/\*

  But :     Contrôleur

  Auteur : Danilo Anzile

  Date :   31.05.2021 / V1.0

\*/

/\*

 \* 1. DOM PRET : DEMARRAGE DE L'APPLICATION

 \*/

document.onreadystatechange = function () {

  if (document.readyState === "complete") {

    \_afficherPersonnes();

  }

};

/\*

 \* 2. METHODES PRIVEES DE LECTURE/ECRITURE DANS LA VUE

 \*/

// affiche la liste des données au bas de la vue (avec du HTML généré)

function \_afficherPersonnes() {

  let txt = "<ul>";

  for (let i = 0; i < personnes.length; i++) {

    txt +="<li>" +'<a href="#" onclick="selectionnerPersonne(' +i +')">' + personnes[i] +"</a>" +"</li>";

  }

  txt += "</ul>";

  document.getElementById("info").innerHTML = txt;

}

// affiche les infos d'une personne dans le formulaire

function \_afficherInfosPersonne(p) {

  document.getElementById("prenom").value = p.prenom;

  document.getElementById("nom").value = p.nom;

  document.getElementById("age").value = p.age;

}

// lit le contenu des masques de saisie pour en faire une personne

function \_lireInfosPersonne() {

  let p = null;

  let prenom = document.getElementById("prenom").value;

  let nom = document.getElementById("nom").value;

  let age = document.getElementById("age").value;

  p = new Personne(prenom, nom, age);

  return p;

}

/\*

 \* 3. METHODES PUBLIQUES NECESSAIRES A LA VUE

 \*/

// appelée depuis la vue pour afficher les données de la personne sélectionnné

function selectionnerPersonne(i) {

  \_afficherInfosPersonne(personnes[i]);

}

// appelée depuis la vue pour ajouter une personne

function ajouter() {

  let p = \_lireInfosPersonne();

  ajouterPersonne(p);

  \_afficherPersonnes();

}

// appelée depuis la vue pour supprimer une personne

function supprimer() {

  let p = \_lireInfosPersonne();

  supprimerPersonne(p);

  \_afficherPersonnes();

}

### Worker.js

/\*

  But :    Worker

  Auteur : Danilo Anzile

  Date :   31.05.2021 / V1.0

\*/

// définition du modèle de données, créé au chargement du js dans le index.html

const personnes = [

    new Personne("John", "Doe", 44),

    new Personne("Anna", "Smith", 32),

    new Personne("Peter", "Jones", 29)

  ];

  // premier tri de la liste de personnes

  personnes.sort();

  // fonction privée pour retrouver l'index d'une personne dans le tableau, -1 autrement

  // il faut comparer avec toString()

  function \_trouverPersonne(p) {

    let idx = -1;

    for (let i=0;i<personnes.length;i++){

        if(p === personnes[i]){

            idx = i;

            break;

        }

    }

    return idx;

  }

  // ajouter une personne dans la liste des personnes si pas trouvée

  function ajouterPersonne(p) {

    let idx = \_trouverPersonne(p);

    if(idx === -1){

        personnes.push(p);

        personnes.sort;

    }

  }

  // supprimer une personne dans la liste des personnes si trouvée

  function supprimerPersonne(p) {

    let idx = \_trouverPersonne(p);

    if(idx >= -1){

        personnes.splice(idx, 1);

    }

  }

### Personne.js

/\*

  But :    Bean Personne

  Auteur : Danilo Anzile

  Date :   31.05.2021 / V1.0

\*/

function Personne(prenom,nom,age){

    this.prenom = prenom;

    this.nom = nom;

    this.age = age;

}

Personne.prototype.toString = function () {

    return this.nom + " " + this.prenom + " (" + this.age + ")";

  };

## Exercice 11

### Index.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link rel="icon" href="#" />

    <link rel="stylesheet" href="css/main.css" />

    <title>Ex10 - POO - prototype</title>

    <script src="js/personne.js"></script>

    <script src="js/worker.js"></script>

    <script src="js/indexCtrl.js"></script>

  </head>

  <body>

    <div id="container">

      <form class="user-form">

        <fieldset>

          <legend>

            Veuillez introduire une nouvelle personne ou sélectionner une

            existante&nbsp;:

          </legend>

          <div class="field">

            <label for="prenom">Prénom:</label>

            <input

              type="text"

              size="20"

              id="prenom"

              placeholder="prénom"

              required="required"

              autofocus

            />

          </div>

          <div class="field">

            <label for="nom">Nom:</label>

            <input

              type="text"

              size="20"

              id="nom"

              placeholder="nom"

              required="required"

            />

          </div>

          <div class="field">

            <label for="nom">Âge:</label>

            <input

              type="text"

              size="20"

              id="age"

              placeholder="âge"

              required="required"

            />

          </div>

          <div class="button">

            <input type="button" value="Ajouter" onclick="ajouter();" />

            <input type="button" value="Supprimer" onclick="supprimer();" />

            <input type="reset" value="Nettoyer" />

          </div>

        </fieldset>

      </form>

      <div id="info"></div>

    </div>

  </body>

</html>

### indexCtrl.js

/\*

  But :     Contrôleur

  Auteur : Danilo Anzile

  Date :   31.05.2021 / V1.0

\*/

/\*

 \* 1. DOM PRET : DEMARRAGE DE L'APPLICATION

 \*/

document.onreadystatechange = function () {

  if (document.readyState === "complete") {

    window.ctrl = new Ctrl(); // ou ctrl = new Ctrl();

    ctrl.afficherPersonnes();

  }

};

class Ctrl {

  constructor() {

    this.wrk = new Worker();

    document.onreadystatechange = function () {

      if (document.readyState === "complete") {

        \_afficherPersonnes();

      }

    };

  }

  /\*

   \* 2. METHODES PRIVEES DE LECTURE/ECRITURE DANS LA VUE

   \*/

  // affiche la liste des données au bas de la vue (avec du HTML généré)

  \_afficherPersonnes() {

    let txt = "<ul>";

    for (let i = 0; i < personnes.length; i++) {

      txt +=

        "<li>" +

        '<a href="#" onclick="window.ctrl.selectionnerPersonne(' +

        i +

        ')">' +

        personnes[i] +

        "</a>" +

        "</li>";

    }

    txt += "</ul>";

    document.getElementById("info").innerHTML = txt;

  }

  // affiche les infos d'une personne dans le formulaire

  \_afficherInfosPersonne(p) {

    document.getElementById("prenom").value = p.prenom;

    document.getElementById("nom").value = p.nom;

    document.getElementById("age").value = p.age;

  }

  // lit le contenu des masques de saisie pour en faire une personne

  \_lireInfosPersonne() {

    let p = null;

    let prenom = document.getElementById("prenom").value;

    let nom = document.getElementById("nom").value;

    let age = document.getElementById("age").value;

    p = new Personne(prenom, nom, age);

    return p;

  }

  /\*

   \* 3. METHODES PUBLIQUES NECESSAIRES A LA VUE

   \*/

  // appelée depuis la vue pour afficher les données de la personne sélectionnné

  selectionnerPersonne(i) {

    this.\_afficherInfosPersonne(personnes[i]);

  }

  // appelée depuis la vue pour ajouter une personne

  ajouter() {

    let p = this.\_lireInfosPersonne();

    this.ajouterPersonne(p);

    this.\_afficherPersonnes();

  }

  // appelée depuis la vue pour supprimer une personne

  supprimer() {

    let p = this.\_lireInfosPersonne();

    this.supprimerPersonne(p);

    this.\_afficherPersonnes();

  }

}

### Personne.js

/\*

  But :    Bean Personne

  Auteur : Danilo Anzile

  Date :   31.05.2021 / V1.0

\*/

class Personne{

  constructor(prenom,nom,age){

    this.prenom = prenom;

    this.nom = nom;

    this.age = age;

  }

}

Personne.prototype.toString = function () {

  return this.nom + " " + this.prenom + " (" + this.age + ")";

};

### Worker.js

/\*

  But :    Worker

  Auteur : Danilo Anzile

  Date :   31.05.2021 / V1.0

\*/

// définition du modèle de données, créé au chargement du js dans le index.html

const personnes = [

  new Personne("John", "Doe", 44),

  new Personne("Anna", "Smith", 32),

  new Personne("Peter", "Jones", 29),

];

class Worker {

constructor() {

    personnes.sort();

}

\_trouverPersonne(p) {

  let idx = -1;

  for (let i = 0; i < personnes.length; i++) {

    if (personnes[i].toString() === p.toString()) {

      idx = i;

      break;

    }

  }

  return idx;

}

ajouterPersonne(p) {

  let idx = this.\_trouverPersonne(p);

  if (idx == -1) {

    personnes.push(p);

  }

}

supprimerPersonne(p) {

  let idx = this.\_trouverPersonne(p);

  if (idx != -1) {

    personnes.splice(idx, 1);

  }

}

}

### Teste technologique

#### Class

Dans JavaScript, nous pouvons aussi créer des classes comme on peut avoir l’habitude avec le Java.

#### This

Lorsque nous appelons une méthode qui se trouve dans la même classe, nous devons utiliser le this.<nomDeLaMéthode> pour l’utiliser.

## Exercice 12

### Index.html

<!DOCTYPE html>

<html>

    <!--

   But :

   Auteur : Danilo Anzile

   Date :   31.05.2021 / V1.0

   -->

  <!-- entête de la page -->

   <head>

      <meta charset="UTF-8" />

      <meta name="viewport" content="width=device-width, initial-scale=1.0" />

      <link rel="icon" href="#"/>

      <link rel= "stylesheet" href= "css/main.css" >

      <script type="text/javascript" src="js/indexCtrl.js" async></script>

      <title>Ex12 — Closure</title>

   </head>

   <!-- corps de la page -->

   <body>

    <div id="container">

      <p>Cliquez pour incrémenter les compteurs</p>

      <button id="button1" onclick="actionCompteur(1);">Compteur No 1</button>

      <button id="button2" onclick="actionCompteur(2);">Compteur No 2</button>

      <button id="button3" onclick="actionCompteur(3);">Compteur No 3</button>

    </div>

  </body>

</html>

### IndexCtrl.js

/\*

  But :

  Auteur : prénom + nom

  Date :   jj.mm.aaaa / V1.0

\*/

function compteur(){

  let count=0;

  return function(){

    return count++;

  };

}

let fct = [compteur(), compteur(), compteur()];

function actionCompteur(idx){

  let val = fct[idx-1].call();

  document.getElementById("button"+idx).innerHTML="Compteur No "+idx+": <b>"+val+"</b>";

}

### Teste technologique

#### Closures

C’est une fonction interne qui va se rappeler et du coup va pouvoir continuer a utilisé des variables définies dans sa fonction parente même après la fin de l’exécution.

## Exercice 13

### Index.Html

<!DOCTYPE html>

<html>

  <!-- entête de la page -->

   <head>

      <meta charset="UTF-8" />

      <meta name="viewport" content="width=device-width, initial-scale=1.0" />

      <link rel="icon" href="#"/>

      <link rel= "stylesheet" href= "css/main.css" >

      <script type="text/javascript" src="js/personne.js"></script>

      <script type="text/javascript" src="js/worker.js"></script>

      <script type="text/javascript" src="js/indexCtrl.js"></script>

      <title>Ex13 — POO - classe - IIFE</title>

   </head>

   <!-- corps de la page -->

   <body>

    <div id="container">

      <form class="user-form">

        <fieldset>

          <legend>Veuillez introduire une nouvelle personne

                  ou sélectionner une existante&nbsp;:</legend>

          <div class="field">

            <label for="prenom">Prénom:</label>

            <input type="text" size="20" id="prenom"

                   placeholder="prénom" required="required" autofocus>

          </div>

          <div class="field">

            <label for="nom">Nom:</label>

            <input type="text" size="20" id="nom"

                   placeholder="nom" required="required">

          </div>

          <div class="field">

            <label for="nom">Âge:</label>

            <input type="text" size="20" id="age"

                   placeholder="âge" required="required">

          </div>

          <div class="button">

            <input type="button" value="Ajouter" onclick="ctrl.ajouter();">

            <input type="button" value="Supprimer" onclick="ctrl.supprimer();">

            <input type="reset" value="Nettoyer">

          </div>

        </fieldset>

      </form>

      <div id="info"></div>

    </div>

  </body>

</html>

### indexCtrl.js

/\*

 \* 1. DOM PRET : DEMARRAGE DE L'APPLICATION

 \*/

// (() => {

  document.onreadystatechange = function () {

  if (document.readyState === "complete") {

    ctrl = new Ctrl();

    ctrl.afficherPersonnes();

  }

};

// })();

class Ctrl {

  constructor() {

    this.wrk = new Worker();

  }

  /\*

   \* 2. METHODES PRIVEES DE LECTURE/ECRITURE DANS LA VUE

   \*/

  // affiche la liste des personnes au bas de la vue (avec du HTML généré)

  afficherPersonnes() {

    let txt = "<ul>";

    for (let i = 0; i < this.wrk.personnes.length; i++) {

      txt +=

        '<li><a href="#" onclick="ctrl.selectionnerPersonne(' +

        i +

        ');">' +

        this.wrk.personnes[i] +

        "</a></li>";

    }

    // this.wrk.personnes.forEach((personne, index) => {

    //   txt += '<li onclick="ctrl.selectionnerPersonne(' + index + ')">'+personne+'</li>';

    // });

    txt += "</ul>";

    document.getElementById("info").innerHTML = txt;

  }

  // affiche les infos de la personne

  afficherInfosPersonne(p) {

    document.getElementById("prenom").value = p.prenom;

    document.getElementById("nom").value = p.nom;

    document.getElementById("age").value = p.age;

  }

  // lit le contenu des masques de saisie pour en faire une personne

  \_lireInfosPersonne() {

    let p = null;

    let prenom = document.getElementById("prenom").value;

    let nom = document.getElementById("nom").value;

    let age = parseInt(document.getElementById("age").value);

    if (prenom.length > 0 && nom.length > 0 && age > 0) {

      p = new Personne(prenom, nom, age);

    }

    return p;

  }

  /\*

   \* 3. METHODES PUBLIQUES NECESSAIRES A LA VUE

   \*/

  // appelée depuis la vue pour afficher les données de la personne sélectionnée

  selectionnerPersonne(i) {

    this.afficherInfosPersonne(this.wrk.personnes[i]);

  }

  // appelée depuis la vue pour ajouter une personne

  ajouter() {

    let p = this.\_lireInfosPersonne();

    this.wrk.ajouterPersonne(p);

    this.afficherPersonnes();

  }

  // appelée depuis la vue pour supprimer une personne

  supprimer() {

    let p = this.\_lireInfosPersonne();

    this.wrk.supprimerPersonne(p);

    this.afficherPersonnes();

  }

}

### personne.js

class Personne {

  constructor(prenom, nom, age) {

    this.prenom = prenom;

    this.nom = nom;

    this.age = age;

  }

  toString() {

    return this.nom + " " + this.prenom + " (" + this.age + ")";

  }

}

### worker.js

/\*

 \* Worker (partie métier de l'application)

 \*/

class Worker {

  constructor() {

    this.personnes = [

      new Personne("John", "Doe", 44),

      new Personne("Anna", "Smith", 32),

      new Personne("Peter", "Jones", 29),

    ];

    this.personnes.sort();

  }

  // fonction privée pour retrouver l'index d'une personne dans le tableau, -1 autrement

  \_trouverPersonne(p) {

    let idx = -1;

    for (let i = 0; i < this.personnes.length; i++) {

      if (p.toString() === this.personnes[i].toString()) {

        idx = i;

        break;

      }

    }

    return idx;

  }

  // ajouter une personne dans la liste des personnes

  ajouterPersonne(p) {

    let idx = this.\_trouverPersonne(p);

    if (idx < 0) {

      this.personnes.push(p);

      this.personnes.sort();

    }

  }

  // supprimer une personne dans la liste des personnes

  supprimerPersonne(p) {

    let idx = this.\_trouverPersonne(p);

    if (idx >= 0) {

      this.personnes.splice(idx, 1);

    }

  }

}

## Exercice 14

### Qu’est-ce que c’est ?

C'est une librairie open source qui simplifie l'interaction entre du HTML et du JavaScript. Son été créée en 2006 par John Resig.

### Pourquoi choisir jQuery ?

* Bonne documentation
* Grande communauté de développeurs.
* Serie de plugins intéressants.
* Léger (~25kb)
* Grande compatibilité avec les navigateurs.

### Qui l’utilise ?

* Microsoft
* Google
* Mozilla
* IBM
* Amazone
* HP
* Intel
* Wordpress
* Django
* …

### Popularité

C’est un complément qui est utilisé par au moins 25 % de touts les sites selon les sources prise chez Google Trends.

### Facile à prendre en main

Il utilise une syntaxe facile et rapide.

### L’objet jQuery

$("div").addClass("special");

Permet de trouve un ou plusieurs éléments d’une page HTML.

* $ 🡪 C’est l’objet JQuery. Soit une fonction JavaScript.
* ‘div’ 🡪 Recherche effectuer avec un sélecteur CSS.
* $(‘div’) 🡪 Retourne un set JQuery de 0 à N nœuds du DOM.
* addClass(…) 🡪 Permet d’ajouter une classe HTML à tous les nœuds qui ont été trouvé.

Il fonctionne un peu comme le CSS, si un nœud n’est pas trouvé, il ne plantera pas.

### L’événement « ready »

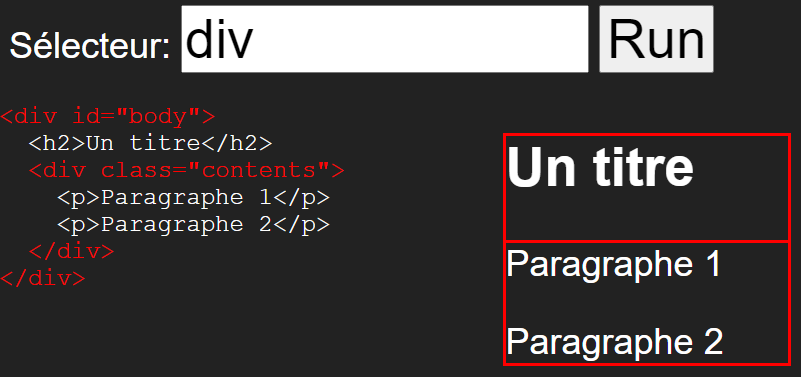
$(document).ready(function(){

    // Votre code jQuery doit se trouver ici

});

Pour manipuler une page HTML, nous devons attendre que celle-ci soit charger soit « ready » avant de l’utiliser. Cet événement « ready » part à l’instant même ou le DOM est prêt à travailler.

**Exemple de sélecteur dans jQuery :**

****

### Traverser les nœuds du DOM

Avec le jQuery, nous avons toutes une suite de méthode qui nous permettent de parcourir le DOM. Exemple :

* Parent()
* Next()
* Prev()
* Children()
* Siblings()
* Etc…

Voir aussi ici pour d'autres méthodes : http://www.w3schools.com/jquery/jquery\_traversing.asp

### Enchaînement

$("div").hide();

$("div").hide().css("color","blue");

$("div").hide().css("color","blue").remove();

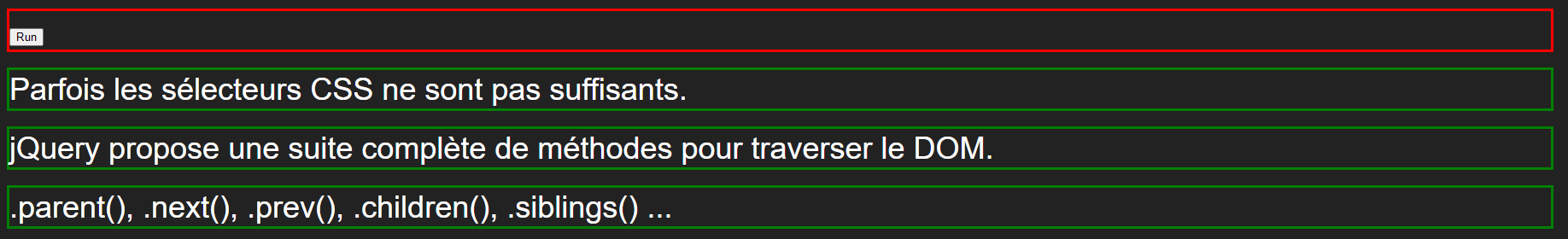
### Enchaînement transversal

$("button")

.parent().css("border", "3px solid red")

.siblings().css("border", "3px solid green");

Résultat :



### Manipulation avec .after()

$("a[target=\_blank]")

.after("<img src='images/open.png'/>");

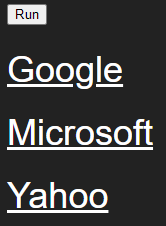
Elle permet de rajouter un élément après une balise <a>, dans ce cas une image.

### Manipulation avec .append()

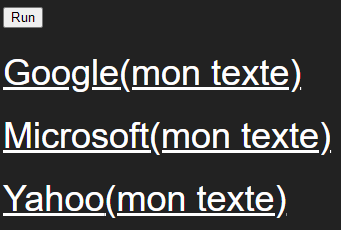
$("a[target=\_blank]")

.append("(mon texte)");

Avant :



Après :



### Manipulation avec .css()

$("li a").css({

color: "red",

fontWeight: "bold"

});

Il permet de changer le style des éléments choisit.

### Sélecteurs HTML

$("<li><a></a></li>")

.find("a")

.attr("href", "http://www.emf.ch/")

.html("EMF")

.end()

.appendTo("ul");

Résultat :



### Evénement .submit()

$("form").submit(function(){

if ( $("#name").val() === "" ) {

$("span.help").show();

return false;

}

});

Si le formulaire est vide, il affiche un message.



### Evénement .click()

$("a.menu").click(function(){

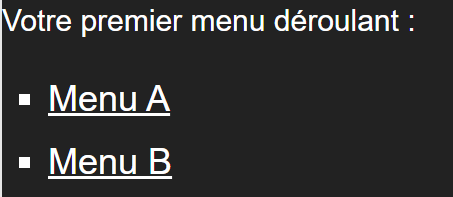
$(this).next().toggle();

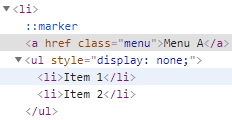
return false;

});

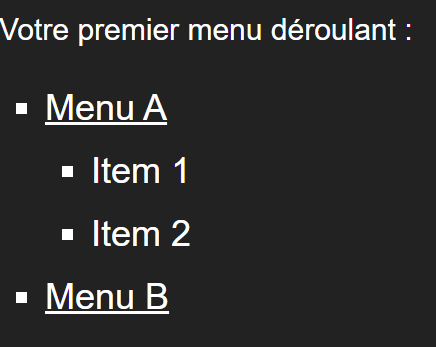
Cette méthode nous permet de faire nos premiers menus déroulants.

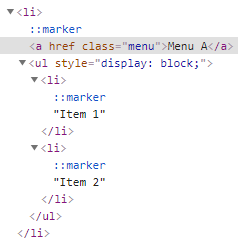
Avant :





Après :





### Evénement .hover()

$("li").hover(function(){

$(this).animate({marginLeft: 38, marginRight: 0});

}, function(){

$(this).animate({marginLeft: 18, marginRight: 18});

});

Cette méthode permet de détecter lorsqu’un utilisateur survole un élément avec la souris et applique un effet de déplacement avec une animation.

### Evénements en live

$("a.menu").live("hover", function(){

$(this).next().toggle(200);

return false;

});

Ce bout de code, permet d’ouvrir le menu déroulant lorsque nous passant notre souris sur l’élément menu. Il le fait avec une petite animation.

### Effet .slideToggle()

$("a.menu").click(function(){

$(this).next().slideToggle("slow");

return false;

});

Lorsque nous cliquons sur l’élément menu, le menu déroulant s’ouvre vers le bas avec une petite animation.

### Effet .animate()

$("div.block").animate({

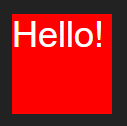
fontSize: "2em",

width: "+=20%",

backgroundColor: "green"

});

Avant:



Après:



### Effet .hide() et .show()

$("div.block").hide("slow", function(){

$(this).show("slow");

});

Fait disparaitre la div l’entement puis la fait réapparaitre lentement.

### Ajax – lire XML

$.ajax({

url: "file.xml",

success: function( xml ) {

$(xml).find("tab").each(function(){

$("ul").append(

"<li>" + $(this).text() + "</li>");

});

}

});

### Ajax – lire JSON

$.getJSON("file.json", function( obj ) {

for ( var prop in obj ) {

$("ul").append(

"<li>" + prop + ": " + obj[prop] + "</li>");

}

});

### Ajax – .load()

$("div.load").load("file.html");

### jQuery CDN

<script src='http://code.jquery.com/jquery.js'></script>

## Exercice 15

### jQuery

// JQuery

$().ready(function () {

  ctrl = new Ctrl();

  let couleur = $("#couleurs").val();

  console.log(couleur);

  ctrl.changerCouleur(couleur);

});

class Ctrl {

  constructor() {}

  changerCouleur(couleur) {

    $("#container").fadeOut(1500, function () {

      $("#container").css("background-color", couleur);

      if (couleur === "red" || couleur === "blue" || couleur === "green") {

        $("#container").css("color", "white");

      } else {

        $("#container").css("color", "black");

      }

      $("#container").slideToggle(1000);

      // $("#container").fadeIn(1000);

    });

  }

}

### JavaScript

(() => {

  document.onreadystatechange = function () {

    if (document.readyState === "complete") {

      window.ctrl = new Ctrl();

      let couleur = document.getElementById("couleurs").value;

      console.log(couleur);

      ctrl.changerCouleur(couleur);

    }

  }

})();

class Ctrl {

  constructor(){

  }

  changerCouleur(couleur) {

    let conteneur = document.getElementById("container");

    conteneur.style.background=couleur;

    if (couleur === "red" || couleur === "blue" || couleur === "green") {

      conteneur.style.color = "white";

    } else {

      conteneur.style.color = "black";

    }

  }

}

### Teste technologique

Durant cet exercice, nous avons fait la même chose pour les deux documents sauf qu’un c’était avec du pur JS tandis que l’autre c’était en utilisant du jQuery.

## Exercice 18

### Index.js

$().ready(function () {

  ctrl = new Ctrl();

  if (httpServ) {

     httpServ.centraliserErreurHttp(ctrl.afficherErreurHttp);

  }

});

class Ctrl {

  constructor() {

  //  this.httpServ = new HttpServ();

  }

  celsius2Fahrenheit(){

    let degres = parseFloat($("#celsius1").val());

    httpServ.convertirCelsiusVersFahrenheit(degres, this.okCelsius2Fahrenheit);

  }

  afficherErreurHttp(msg) {

    //    console.error(msg);

        alert(msg);

  }

  okCelsius2Fahrenheit(data, textStatus, jqXHR) {

    let temp = $(data).find("double").text();

    let affiche = (temp !== "NaN") ? temp : "";

    $("#fahrenheit1").val(affiche);

  }

}

### httpServ.js

var httpServ = (function () {

  function \_centraliserErreurHttp(httpErrorCallbackFn) {

    $.ajaxSetup({

      error: function (jqXHR, exception) {

        let msg;

        if (jqXHR.status === 0) {

          msg = "Pas d'accès à la ressource serveur demandée !";

        } else if (jqXHR.status === 404) {

          msg = "Page demandée non trouvée [404] !";

        } else if (jqXHR.status === 500) {

          msg = "Erreur interne sur le serveur [500] !";

        } else if (exception === 'parsererror') {

          msg = "Erreur de parcours dans le JSON !";

        } else if (exception === 'timeout') {

          msg = "Erreur de délai dépassé [Time out] !";

        } else if (exception === 'abort') {

          msg = "Requête Ajax stoppée !";

        } else {

          msg = "Erreur inconnue : \n" + jqXHR.responseText;

        }

        httpErrorCallbackFn(msg);

      }

    });

  }

  function \_envoyerRequete(nomService, valeur, successCallback) {

    let url = "http://mettrauxpa.emf-informatique.ch/convert\_temp\_p\_xml.php";

    let param = "Temperature=" + valeur;

    // suivant le type de copnversion, on modifie la requête

    if (nomService === "CelsiusToFahrenheit") {

      param += "&FromUnit=C&ToUnit=F";

    } else {

      param += "&FromUnit=F&ToUnit=C";

    }

    // envoi de la requête

    $.ajax(url, {

      type: "POST",

      contentType: "application/x-www-form-urlencoded; charset=UTF-8",

      data: param,

      success: successCallback

    });

  }

  function \_convertirCF(valeur, successCallback) {

    \_envoyerRequete("CelsiusToFahrenheit", valeur, successCallback);

  }

  function \_convertirFC(valeur, successCallback) {

    \_envoyerRequete("FahrenheitToCelsius", valeur, successCallback);

  }

  // définition des noms de méthodes publiques (à gauche)

  return {

    centraliserErreurHttp: \_centraliserErreurHttp,

    convertirCelsiusVersFahrenheit: \_convertirCF,

    convertirFahrenheitVersCelsius: \_convertirFC

  };

})();

### Teste technologique

Durant cet exercice, nous avons dû réussir a communiquer avec un serveur qui nous permettait de transformer une valeur °C en °F.

## Exercice 19

### indexCtrl.js

$().ready(function () {

  ctrl = new Ctrl();

});

class Ctrl {

  constructor() {

    this.httpServ = new HttpServ();

    this.httpServ.centraliserErreurHttp(this.afficherErreurHttp);

  }

  afficherErreurHttp(msg){

    alert(msg);

  }

  celsius2Fahrenheit() {

    let degres = parseFloat($("#celsius1").val());

    this.httpServ.celcius2Fahrenheit(degres, this.OKCelsius2Fahrenheit);

  }

  OKCelsius2Fahrenheit(data) {

    let temp = $(data).find("temperature").text();

    let affiche = temp !== "NaN" ? temp : "";

    $("#fahrenheit1").val(affiche);

  }

  fahrenheit2Celcius() {

    let degres = parseFloat($("#fahrenheit2").val());

    this.httpServ.fahrenheit2Celcius(degres, this.OKFahrenheit2Celcius);

  }

  OKFahrenheit2Celcius(data) {

    let temp = data.temperature;

    let affiche = temp !== "NaN" ? temp : "";

    $("#celsius2").val(affiche);

  }

}

### httpServ.js

class HttpServ {

  constructor() {}

  centraliserErreurHttp(httpErrorCallbackFn) {

    $.ajaxSetup({

      error: function (xhr, exception) {

        let msg;

        if (xhr.status === 0) {

          msg = "Pas d'accès à la ressource serveur demandée !";

        } else if (xhr.status === 404) {

          msg = "Page demandée non trouvée [404] !";

        } else if (xhr.status === 500) {

          msg = "Erreur interne sur le serveur [500] !";

        } else if (exception === "parsererror") {

          msg = "Erreur de parcours dans le JSON !";

        } else if (exception === "timeout") {

          msg = "Erreur de délai dépassé [Time out] !";

        } else if (exception === "abort") {

          msg = "Requête Ajax stoppée !";

        } else {

          msg = "Erreur inconnue : \n" + xhr.responseText;

        }

        httpErrorCallbackFn(msg);

      },

    });

  }

  celcius2Fahrenheit(degres, successCallback) {

    let url = "http://mettrauxpa.emf-informatique.ch/convert\_temp\_p\_xml.php";

    let param = "Temperature=" + degres + "&FromUnit=C&ToUnit=F";

    // envoi de la requête

    $.ajax(url, {

      type: "POST",

      contentType: "application/x-www-form-urlencoded; charset=UTF-8",

      data: param,

      success: successCallback

    });

  }

  fahrenheit2Celcius(degres, successCallback) {

    let url = "http://mettrauxpa.emf-informatique.ch/convert\_temp\_g\_json.php";

    let param = "Temperature=" + degres + "&FromUnit=F&ToUnit=C";

    // envoi de la requête

    $.ajax(url, {

      type: "GET",

      data: param,

      success: successCallback

    });

  }

}

### Teste technologique

Le même principe que l’exercice 18 mais ici il fallait le faire dans les deux censes, de °C en °F ou de °F en °C.

## Exercice 21

### Street.html

<html>

  <!-- entête de la page -->

  <head>

    <meta content="text/html; charset=UTF-8" http-equiv="Content-Type" />

    <link

      rel="stylesheet"

      href="https://unpkg.com/leaflet@1.6.0/dist/leaflet.css"

      integrity="sha512-xwE/Az9zrjBIphAcBb3F6JVqxf46+CDLwfLMHloNu6KEQCAWi6HcDUbeOfBIptF7tcCzusKFjFw2yuvEpDL9wQ=="

      crossorigin=""

    />

    <script

      src="https://unpkg.com/leaflet@1.6.0/dist/leaflet.js"

      integrity="sha512-gZwIG9x3wUXg2hdXF6+rVkLF/0Vi9U8D2Ntg4Ga5I5BZpVkVxlJWbSQtXPSiUTtC0TjtGOmxa1AJPuV0CPthew=="

      crossorigin=""

    ></script>

    <link rel="stylesheet" href="css/main.css" />

    <script type="text/javascript" src="js/streetCtrl.js"></script>

    <!-- Titre du script HTML -->

    <title>EX21 - Géolocalisation</title>

  </head>

  <!-- Section BODY -->

  <body onload="initialiserCarte()">

    <h1>Rappelez-vous où était située l'École des Métiers de Fribourg avec Open StreetMap</h1>

    <div id="mapid"></div>

  </body>

</html>

### streetCtrl.js

function initialiserCarte() {

  const RedMarkerIcon = L.icon({ iconUrl: 'images/redmarkericon.png', iconSize: [18, 30], iconAnchor: [9, 30], popupAnchor: [0, -20] });

  const GreenMarkerIcon = L.icon({ iconUrl: 'images/greenmarkericon.png', iconSize: [18, 30], iconAnchor: [9, 30], popupAnchor: [0, -20], });

  const mapid = L.map("mapid").setView([46.8, 7.15], 15);

  L.tileLayer("https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png", {

    attribution:

      '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors',

  }).addTo(mapid);

  L.marker([46.7938995, 7.1563006])

    .addTo(mapid)

    .bindPopup("Je travaille <b>ici<b>")

    .openPopup();

  L.marker([46.803154, 7.151173])

    .addTo(mapid)

    .bindPopup("Gare de Fribourg");

  L.marker([46.804821, 7.154811], { icon: RedMarkerIcon })

    .addTo(mapid)

    .bindPopup("Mon restaurant favori");

  L.marker([46.805699, 7.164400],{ icon: GreenMarkerIcon })

    .addTo(mapid)

    .bindPopup("Celui du prof !<br>Restaurant du Cygne");

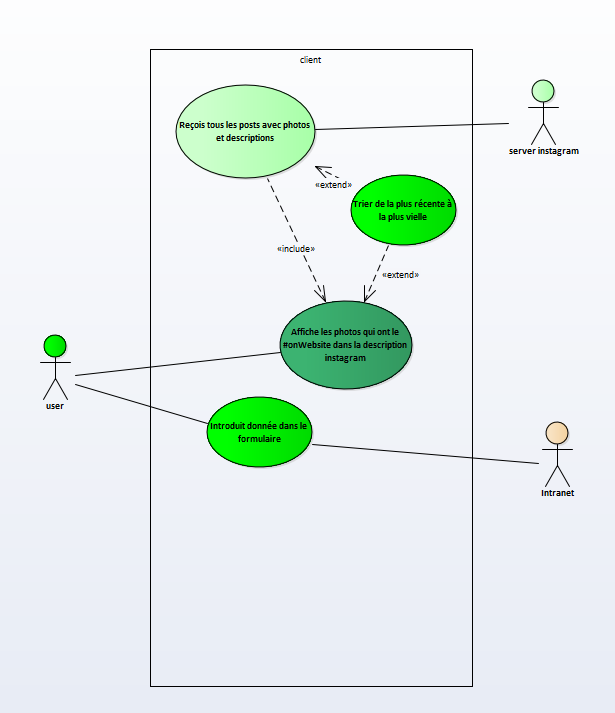
}

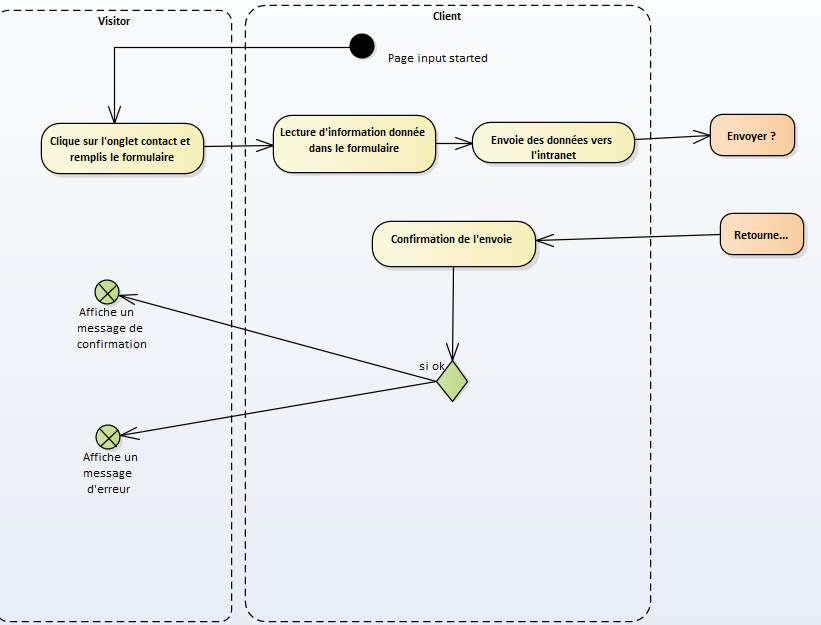
### Teste technologique

Durant cet exercice, nous avons dû générer une map openstreet et ensuite y travailler dessus en lui rajoutant des points de référence comme l’emplacement de l’EMF.

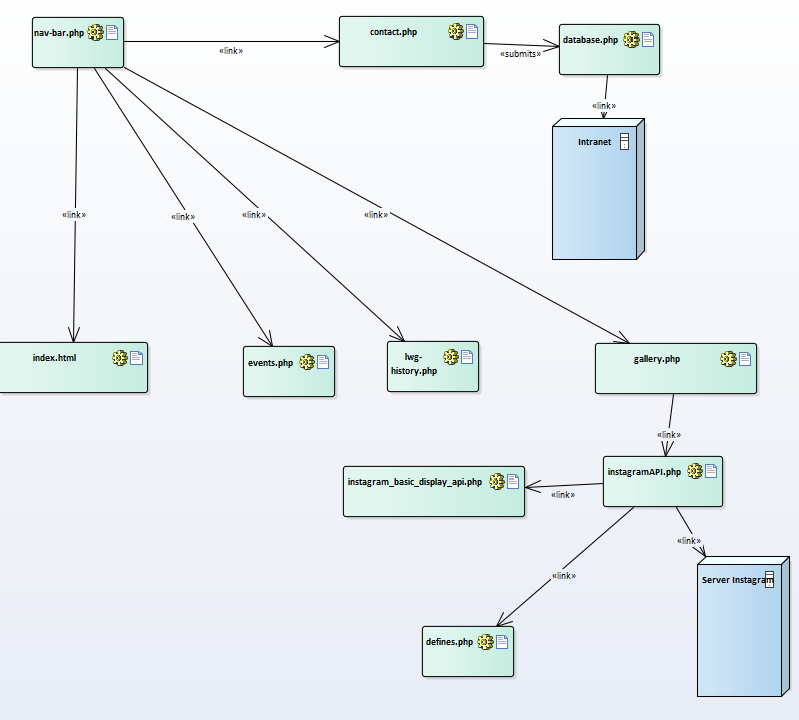
# LWG

## Analyse





## Conception



### Explication supplémentaire

Chaque page est reliée à un css qui est nommé avec le même nom que la page.

Pour éviter d’avoir du code en doublons, j’ai fait 3 documents qui sont appelée à chaque page qui est charger. Ses documents sont :

* Head.php
* Footer.php
* scriptEndPage.php

## Implémentation

### Index.php

Cette page est l’accueil du site. Elle est constituée de :

* La barre de navigation
* Une image d’accueil
* Du bas de page

### Lwg-history.php

Cette page explique un peu l’histoire de l’entreprise et leur fonctionnement. Elle aussi est constituée de la barre de navigation et le bas de page.

### Events.php

Actuellement cette page n’est pas très remplie à cause de la situation sanitaire actuelle. Dans le futur, il y aura la liste de tous les évènements qui pourront être gérer depuis l’intranet.

### Gallery.php

Cette page est la plus complété et complexe du site. On y retrouve en premier une belle montre en noir et blanc. En suite dans la première section, nous retrouvons toutes les images qui se trouve sur Instagram et qui on dans la description le #OnWebsite. La deuxième section est actuellement constituée de 3 images qui sont choisi aléatoirement mise en noir et blanc puis remis en couleur lorsque nous les survolant. Et pour conclure nous retrouvons de nouveau le bas de page.

### Contact.php

C’est une page assez simple qui est constitué d’un formulaire de contact. Chaque formulaire envoyé va se retrouver dans la base de données qui permettra de gérer ses demandes sur l’intranet.

### Head.php

Ici nous retrouvons tout les liens interne au site. Comme les pages css ou l’icone du site. Cette page est appelée sur toute les pages principaux ce qui permet d’éviter d’avoir du code en doublons.

### Nav-bar.php

Nous y retrouvons la barre de navigation qui se trouve aussi sur tous les pages principaux.

### ScriptEndPage.php

On y retrouve l’appelle toutes les scripts utilisées sur le site.

### Defines.php

On y retrouve les identifiants de connections pour l’api Instagram.

### Instagram\_basic\_display\_api.php

C’est la classe créer qui permet d’appeler l’api. On peut y interagir avec en lui demandant tout les médias de l’utilisateur choisi ainsi que les métas donnés de l’utilisateur.

### instagramAPI.php

Ici nous appelons la classe Instagram et nous lui demandons de récupérer toutes les images du compte qui on le #OnWebsite dans la description.

### mainCtrl.js

Nous retrouvons principalement 2 méthodes.

* La première est celle qui se trouve dans la page gallery qui permet de choisir 3 images aléatoire à chaque chargement de page.
* La deuxième est celle pour la barre de navigation pour qu’elle soit responsive sur mobile.

### Database.php

Ce fichier est utilisé lorsque l’utilisateur envoie un formulaire de contact. Il permet de se connecter a la base de données et d’envoyé le formulaire.

## Hébergement

Pour l’hébergement, le site principal se trouve à l’adresse : <luxurywatchgallery.anziled.emf-informatique.ch>.

L’intranet se situe à l’adresse : <intranet.luxurywatchgallery.anziled.emf-informatique.ch>.

## Fonctionnement

L’utilisateur doit juste se balader sur le site et s’il veut, il peut contacter les membres de l’entreprise via le formulaire. L’objectif final du site sera que les membres de l’entreprises, puisse directement modifier les pages du site depuis l’intranet. Comme rajouter de nouveau évènement ou encore rajouter de nouvelles images.

## API

J’ai choisi d’utilisé l’API de Instagram qui me permettra de récupérer plusieurs informations d’un profil. L’objectif sera de récupérer les post de <https://instagram.com/luxury_watch_gallery.ch> pour en faire une gallérie d’image.

### Ce que je vais utiliser de cette API

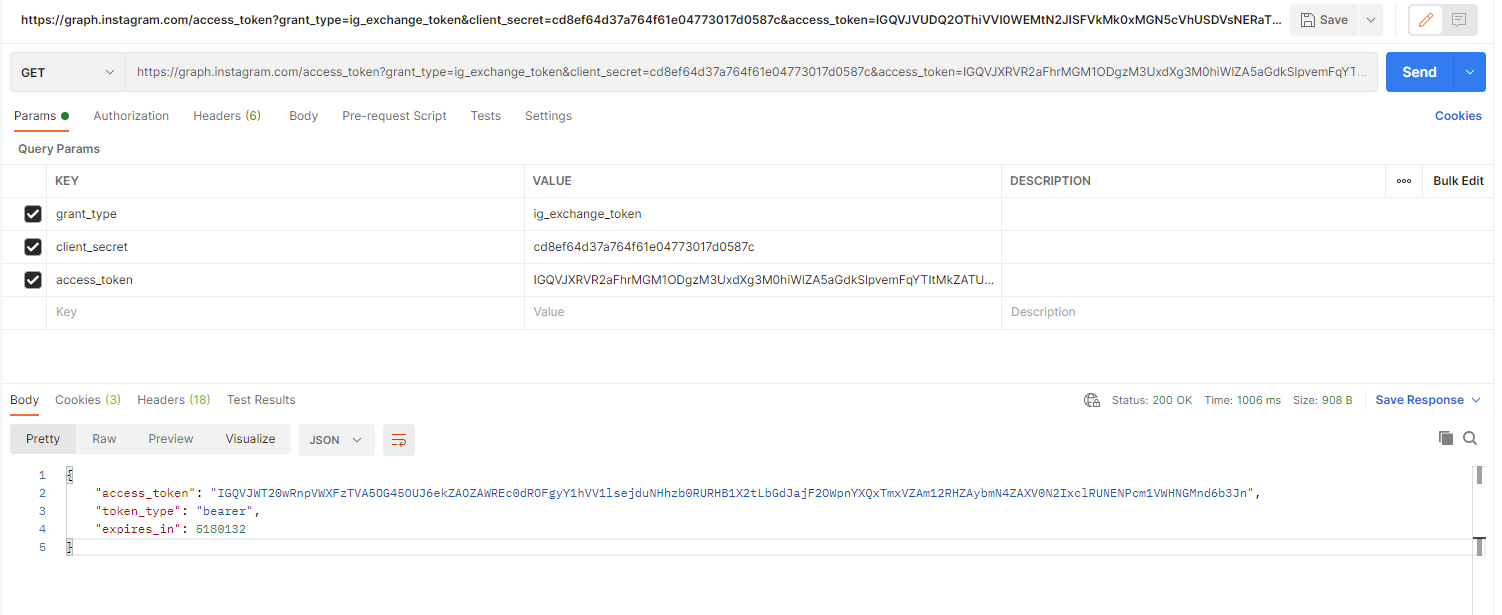
Je vais récupérer chaque image qui est sur le profil qui contient dans la description un #onWebsite. Avec des images, je vais pouvoir faire un portfolio structurer avec comme base ces photos.

### Lien vers documentations API

<https://developers.facebook.com/docs/instagram-api/>

### Recuperation du Token LongLife

Pour ne pas devoir à chaque fois me loguer sur le site, il me fallait un token de longue vie. Pour ce faire, j’ai dû d’abord récupérer le token de courte vie, puis me rendre sur PostMan, pour faire une requête au serveur d’Instagram.



Malheureusement, ce token a une durée de vie de 60 jours. Ce qui veut dire que je dois effectuer cette manipulation manuellement chaque deux mois environs pour garder le site à jour.

Méthode Get :

https://graph.instagram.com/access\_token?grant\_type=ig\_exchange\_token&client\_secret=cd8ef64d37a764f61e04773017d0587c&access\_token=IGQVJXRVR2aFhrMGM1ODgzM3UxdXg3M0hiWlZA5aGdkSlpvemFqYTItMkZATU0NmS0Y3LV9BWTc2eWlQNnR1dHhDZAjkzR1JLaE9RZAkJBUHEwd2E1NzhZAdkFTMG5tb2M0TlJlU2VBRUlaMnNKOTVqN1NCZADh5VDNSSVFQaTRv

Lien vers PostMan : <https://web.postman.co/>

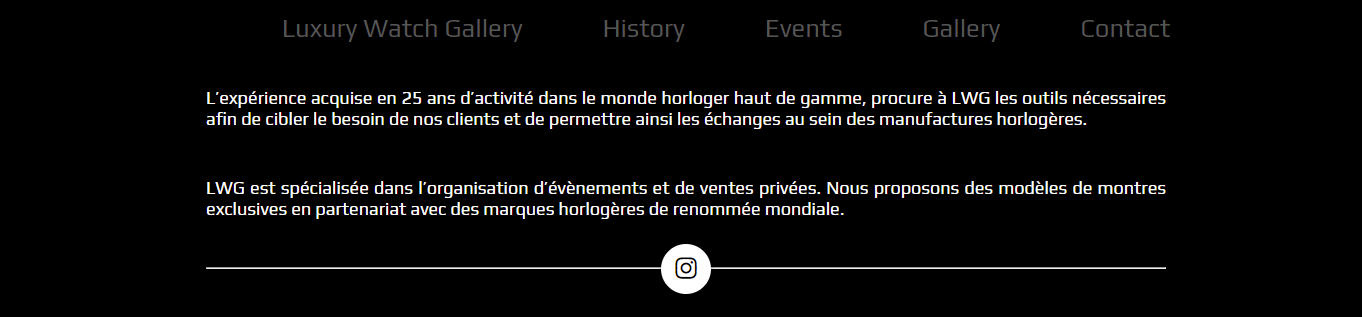
## Le site principal

Accessible actuellement depuis l’url : <luxurywatchgallery.anziled.emf-informatique.ch>

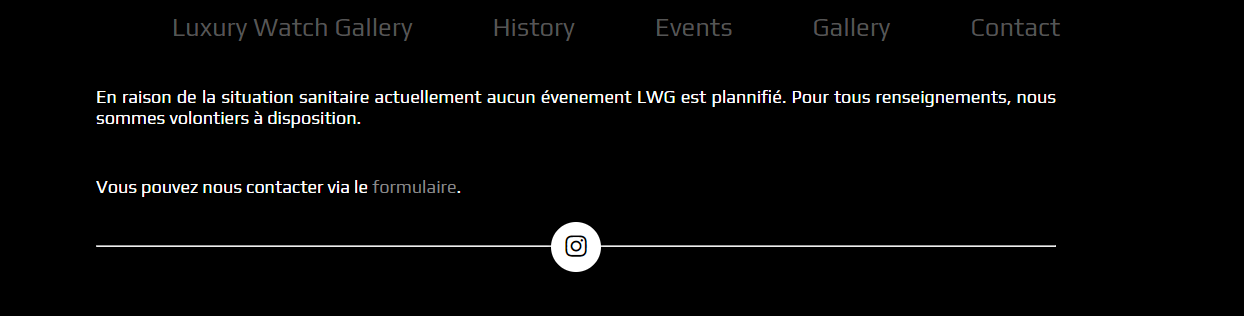
### Page d’accueil



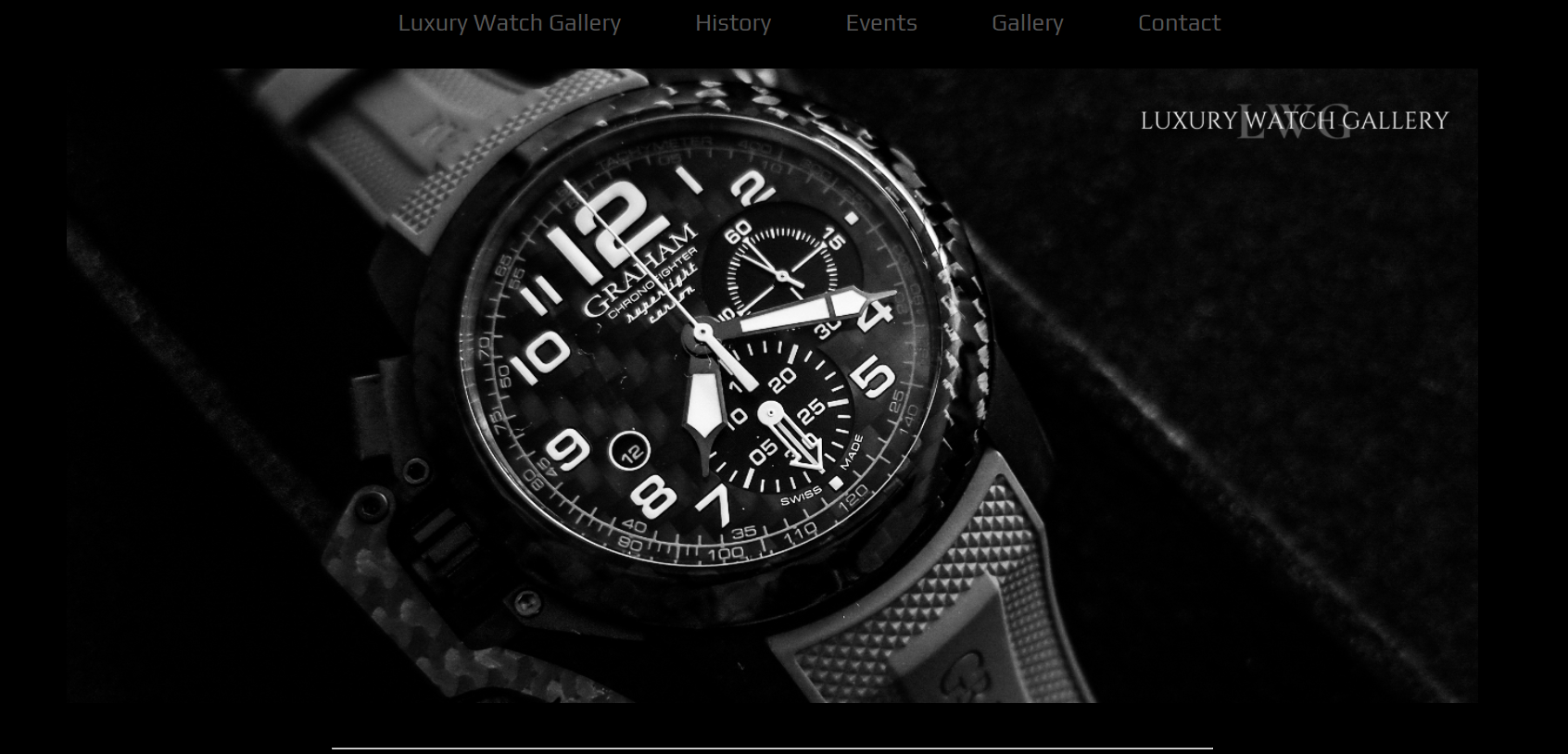
### Page History



### Page Events



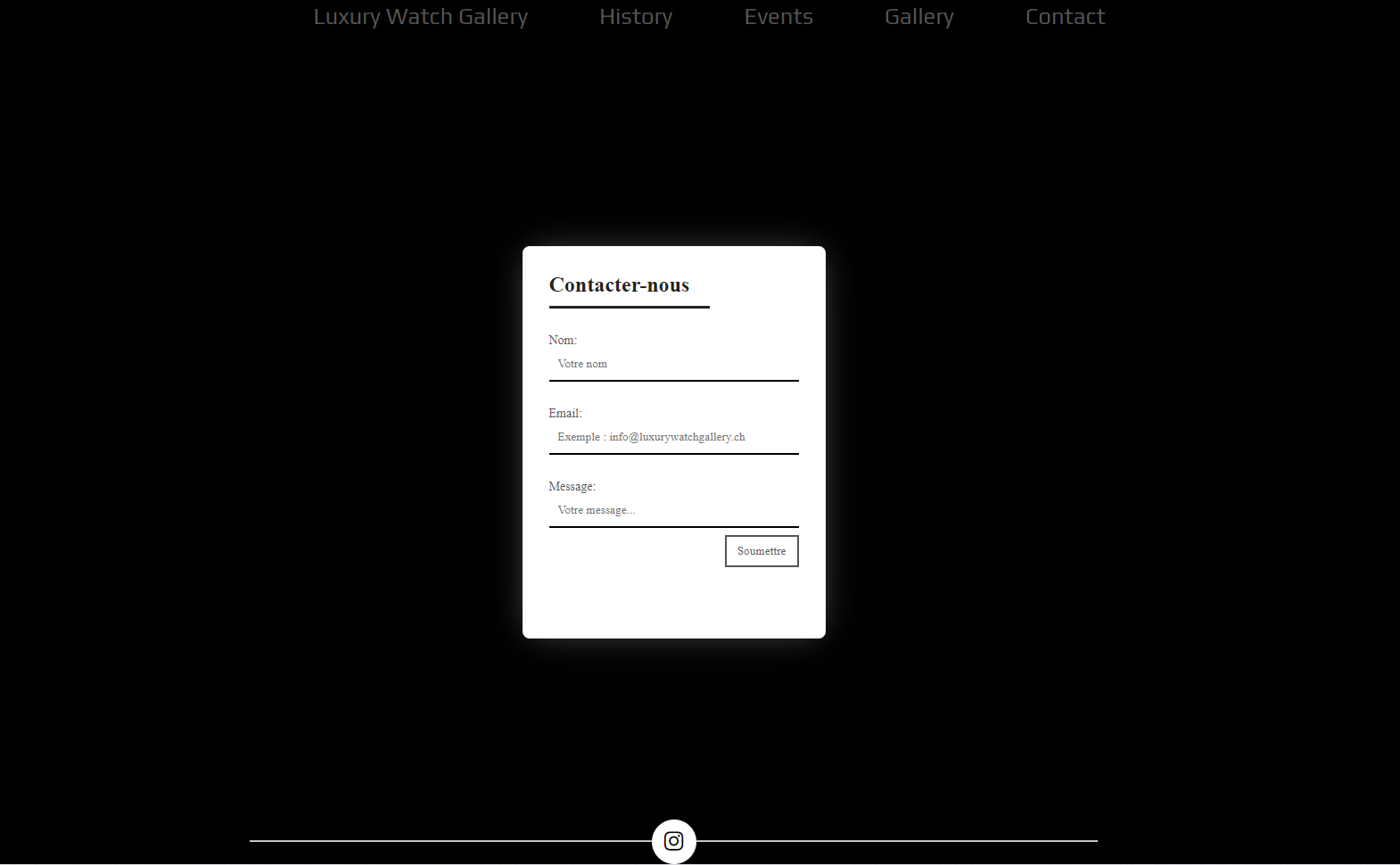
### Page Gallery





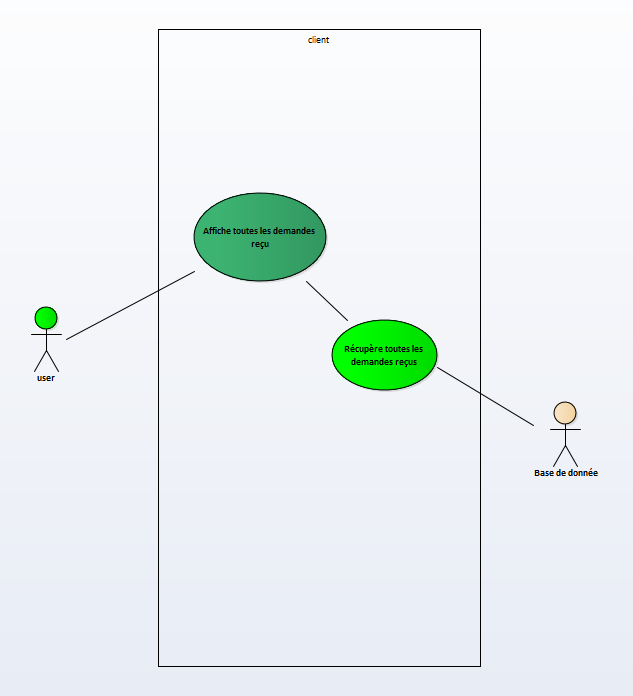
### Page de contact

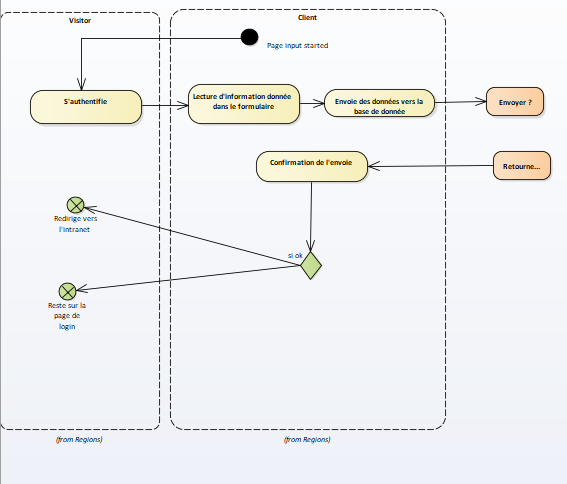
Pour celle-ci, elle sert au client de contacter l’entreprise. Le site va collecter les informations entrées dans une base de données puis tout les demandes seront traité sur l’intranet qui est accessible par un login.



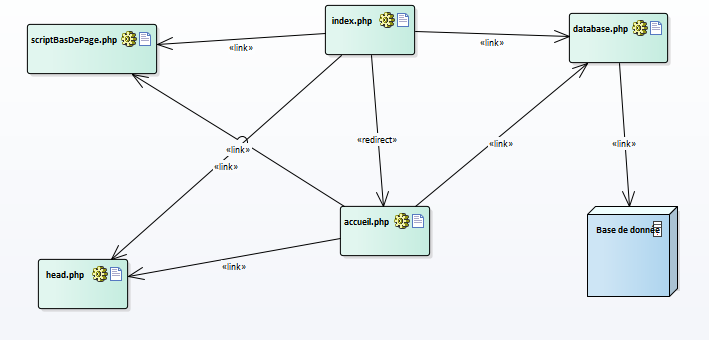
# Intranet.LWG

## Analyse





## Conception



## Implémentation

### Index.php

C’est la page de login qui nous redirige vers l’intranet lorsque nous nous connectons.

### Database.php

C’est le fichier qui nous permet de nous connecter a la base de données et de récupère nos identifiants ainsi que la liste des demandes reçu.

### Accueil.php

Actuellement la page est un peu vide mais dans le futur nous pourrons gère diverses fonctionnalités comme rajouter des évènements sur le site principal ou rajouter des images dans la gallery et gère les demandes reçues.

### Head.php

L’entête de la page html.

### scriptBasDePage.php

Toutes les scripts utilisées dans le site.

### mainCtrl.js

Le fichier qui contient les fonctions javascripts du site.

## Hébergement

L’intranet se situe à l’adresse : <intranet.luxurywatchgallery.anziled.emf-informatique.ch>.

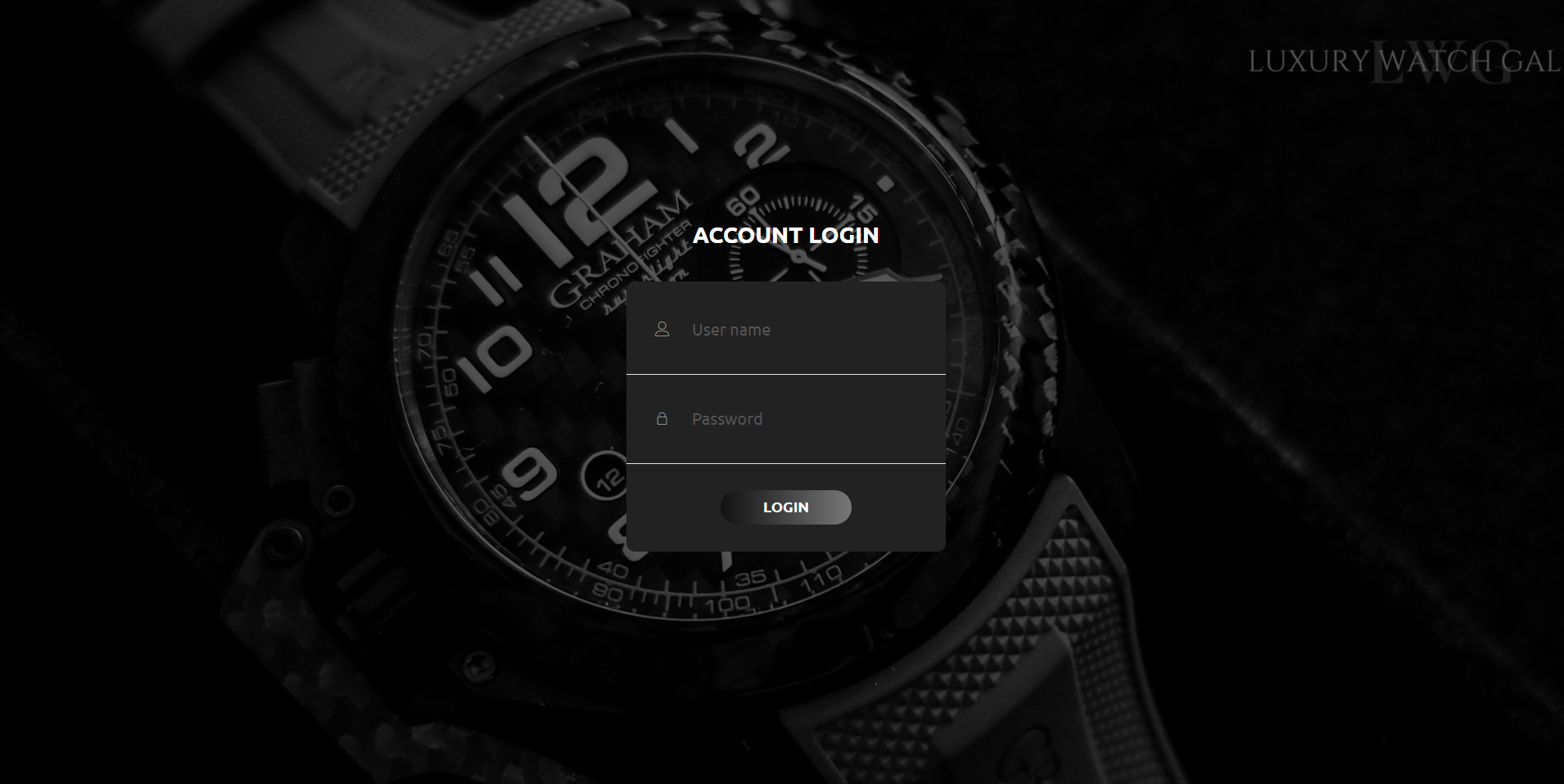
## Fonctionnement

Permet de gérer le site principal d’une façon plus simple et sans savoir coder. (Objectif final.)

## Le site

Accessible actuellement depuis l’url : [https://intranet.luxurywatchgallery.anziled.emf-informatique.ch](https://intranet.luxurywatchgallery.anziled.emf-informatique.ch/index.php)

### Page de login



Utilisateur temporaire : TempUsr

Mdp : Emf12345/?

# Conclusion

J’ai vraiment beaucoup aimé ce module, je pense même que c’est le meilleur module qu’on a eu durant la formation actuellement. Il m’a permis de faire un projet pour le quelle je suis fier du rendu final et m’a beaucoup appris sur le fonctionnement web. L’unique point négatif et que nous avons pris trop de temps au début du module pour faire des exercices.