

Travaux dirigés & pratiques : PHP & JS

Table des matières

I Période 8 : PHP	3
1 TD PHP	4
1.1 TD PHP Bases (©J. Engels)	5
1.2 TD PHP Structures de contrôle (©J. Engels)	6
1.3 TD PHP Chaînes de caractères (©J. Engels)	7
1.4 TD PHP Tableaux (©J. Engels)	9
1.5 TD PHP Formulaires (©J. Engels)	12
1.6 TD PHP Fonctions (©J. Engels)	13
2 TP PHP	14
2.1 TP PHP 0 - Environnement de travail	15
2.2 TP PHP 1 - Premières pages	17
2.3 TP PHP 2 - Génération HTML	19
2.4 TP PHP 3 - Manipulation de tableaux	21
2.5 TP PHP 4 - Formulaires	23
2.6 TP PHP 5 - Fonctions, Lecture de fichiers	24
2.7 TP PHP 6 - Création de fichiers	26
2.8 TP PHP 7 - Manipulation CSV/JSON	27
II Période 9 : JS	28
3 TD Javascript	29
3.1 TD JS Bases (©J.M. Richer)	30
4 TP Javascript	31
4.1 TP JS 0 - Outils de développement	32
4.2 TP JS 1 - Premiers pas en JS	36
4.3 TP JS 2 - Chaines et Tableaux	38
4.4 TP JS 3 - Objets	40
4.5 TP JS 4 - Premiers pas avec le DOM	44
4.6 TP JS 5 - Création d'éléments	46
4.7 TP JS 6 - Manipulation statique du DOM	47
4.8 TP JS 7 - Manipulation statique du DOM	49
III Période 10 : JS / PHP	52
5 TD JS / PHP	52
5.1 TD JS Evénements (©J.M. Richer, A. Rossi)	53

6 TP JS / PHP	56
6.1 TP JS 8 - Evènements et formulaires	57
6.2 TP JS 9 - Evènements et formulaires	59
6.3 TP JS 10 - Evènements et formulaires	61
6.4 TP JS 11 - Validation JS de formulaires et bases de données	64
6.5 TP PHP 8 - Base de données (sportifs)	66
6.6 TP PHP 9 - Base de données (crédit bancaire)	69
6.7 TP PHP 10 - Révisions PHP/Javascript (catalogue musical)	72

Première partie

Période 8 : PHP

1 TD PHP

1.1 TD PHP Bases (©J. Engels)

Exercice 1. Parmi les variables suivantes, lesquelles ont un nom valide :

```
mavar, $mavar, $var5, $_mavar, $_5var, $__élément1, $hotel4*
```

Exercice 2. Donner les valeurs de \$x, \$y, \$z à la fin du script suivant :

```
<?php
$x="PostgreSQL";
$y="MySQL";
$z=&$x;
$x="PHP 5";
$y=&$x;
?>
```

Exercice 3. Lire les valeurs des variables du script de l'exercice 2 à l'aide du tableau \$GLOBALS.

Exercice 4. Déterminer le numéro de version de PHP, le nom du système d'exploitation de votre serveur ainsi que la langue du navigateur du poste client.

Exercice 5. Donner la valeur de chacune des variables pendant et à la fin du script suivant et vérifier l'évolution du type de ces variables :

```
$x="PHP7";
$a[ ]=&$x;
$y=" 7e version de PHP";
$z=$y*10;
$x.= $y;
$y*=$z;
$a[0]="MySQL";
```

Exercice 6. Donner la valeur de chacune des variables à la fin du script :

```
$x="7 personnes";
$y=(integer) $x;
$x="9E3";
$z=(double) $x;
```

Exercice 7. Donner la valeur booléenne de chacune des variables :

```
$a="0";
$b="TRUE";
$c=FALSE;
$d=($a OR $b);
$e=($a AND $c);
$f=($a XOR $b);
```

1.2 TD PHP Structures de contrôle (©J. Engels)

Exercice 1. Rédiger une expression conditionnelle pour tester si un nombre est à la fois un multiple de 3 et de 5.

Exercice 2. Écrire une expression conditionnelle utilisant les variables `$age` et `$sexe` dans une instruction `if` pour sélectionner une personne de sexe féminin dont l'âge est compris entre 21 et 40 ans et afficher un message de bienvenue approprié.

Exercice 3. Effectuer une suite de tirages de nombres aléatoires jusqu'à obtenir une suite composée d'un nombre pair suivi de deux nombres impairs. Exemple de résultat obtenu :

- 194,285,494
- 435,759,162
- 237,292,768
- 366,533,397

Résultat obtenu en 4 coups.

Exercice 4. Créer et afficher des numéros d'immatriculation automobile (pour Paris, par exemple) en commençant au numéro 100 PHP 75. Si on réalise le script complet, il affiche plusieurs millions de numéros de 100 PHP 75 à 999 ZZZ 75. L'exécution est donc très longue et risque de bloquer le serveur. Pour effectuer un test, les valeurs des chiffres seront limitées entre 100 et 120.

Effectuer ensuite la même procédure en mettant en réserve les numéros dont le premier groupe de chiffres est un multiple de 100. Stocker ces numéros particuliers dans un tableau. Pour ne conserver que les nombres multiples de 100 nous remplaçons l'instruction `echo` par le stockage des numéros dans un tableau. Il est affiché à la fin de toutes les boucles à l'aide de la fonction `print_r()`. On constate alors qu'il contient déjà 65150 éléments !

Exercice 5. Choisir un nombre de trois chiffres. Effectuer ensuite des tirages aléatoires et compter le nombre de tirages nécessaire pour obtenir le nombre initial. Arrêter les tirages et afficher le nombre de coups réalisés. Réaliser ce script d'abord avec l'instruction `while` puis avec l'instruction `for`. Notez qu'il est rare d'obtenir le nombre cherché en moins de 100 tirages.

Exercice 6. Créer un tableau dont les indices varient de 11 à 36 et dont les valeurs sont des lettres de A à Z. Lire ensuite ce tableau avec une boucle `for` puis une boucle `foreach` et afficher les indices et les valeurs (la fonction `chr(n)` retourne le caractère dont le code ASCII vaut `n`).

Exercice 7. Utiliser une boucle `while` pour déterminer le premier entier obtenu par tirage aléatoire qui soit un multiple d'un nombre donné. Écrire la variante utilisant la boucle `do...while`.

Exercice 8. Rechercher le **plus grand commun diviseur** de deux variables initialisées dans votre script. Gérer au moyen d'une exception le cas où l'une des variables n'est pas entière.

1.3 TD PHP Chaînes de caractères (©J. Engels)

Exercice 1. Transformez une chaîne écrite dans des cassettes différentes afin que chaque mot ait une initiale en majuscule. Appliquez à la chaîne :

```
"TransFOrmeZ unE ChaîNE écRITe dans des cASses diFFéreNTes
afIN qUe chAQue MOT ait une iNITiale en MAJUSCULE"
```

Le résultat est le suivant :

```
Transformez Une Chaîne Écrite Dans Des Cassettes Différentes
Afin Que Chaque Mot Ait Une Initiale En Majuscule
```

Exercice 2. En utilisant la fonction `strlen()` écrivez une boucle qui affiche chaque lettre de la chaîne "PHP MySQL" sur une ligne différente.

Exercice 3. Formattez l'affichage d'une suite de chaînes contenant des noms et prénoms en respectant les critères suivants : un prénom et un nom par ligne affichés sur 20 caractères ; toutes les initiales des mots doivent se superposer verticalement. Pour obtenir de bons résultats il est préférable d'utiliser une police à espacement fixe en incluant le texte de chaque ligne dans un élément HTML `<tt>`. Le résultat envisagé est le suivant :

```
Azerky_____ Sophia_____
Bazertudoh_____ Jean-Michel_____
```

Utilisez ensuite la fonction `vsprintf()` et passer les chaînes à afficher dans un tableau multidimensionnel.

Exercice 4. Utilisez les fonctions adéquates afin que la chaîne `<form action="script.php">` soit affichée telle quelle et non comme du code HTML. Le code HTML de l'affichage est le suivant :

```
&lt;form action="script.php"&gt;
```

Exercice 5. À partir de deux chaînes quelconques contenues dans des variables, effectuez une comparaison entre elles pour pouvoir les afficher en ordre alphabétique naturel. Utilisez la fonction `strtolower()` avant d'opérer la comparaison, sinon tous les caractères de A à Z sont avant les caractères a à z.

Exercice 6. Effectuez une censure sur des textes (par exemple sur "Zut je me suis trompé") en n'affichant pas ceux qui contiennent le mot zut.

Exercice 7. Créez une fonction de validation d'une adresse HTTP utilisant `preg_match()`. Les adresses doivent se conformer au modèle suivant : (1) Commencer par « www »; (2) Suivi par des caractères alphanumériques puis éventuellement un point ou un tiret suivi d'un deuxième groupe de caractères alphanumériques; (3) Se terminer par un point suivi de l'extension qui peut avoir de 2 à 4 caractères alphanumériques. Le modèle est insensible à la casse. Exemples de résultats à obtenir :

```
www.univ-angers.fr est valide
www.info.ua.fr est valide
www.49\_info.ua.fr est invalide
WWW.info.ua.fr est valide
www*info.ua.fr est invalide
www.info;ua.fr est invalide
www.info.ua.france est invalide
www.info.ua.ulb.fr est invalide
```

Exercice 8. Créez une expression régulière pour valider un âge inférieur à 100 ans. Le résultat obtenu sur la chaîne "84" est le suivant :

```
84 est un âge valide
```

Exercice 9. Dans la chaîne "PHP 5 \n est meilleur \n que ASP\n et JSP \n réunis", remplacez les caractères \n par
 en utilisant deux méthodes différentes (une fonction ou une expression régulière) :

PHP
est meilleur
que ASP
et JSP
réunis

1.4 TD PHP Tableaux (©J. Engels)

Exercice 1. Écrivez un tableau multidimensionnel associatif dont les clés sont des noms de personne et les valeurs des tableaux indicés contenant le prénom, la ville de résidence et l'âge de la personne. Afficher le tableau avec la fonction `print_r()`.

Exemple :

```
Array (
    [Dupont] => Array ( [0] => Paul [1] => Paris [2] => 27 )
    [Schmoll] => Array ( [0] => Kirk [1] => Berlin [2] => 35 )
    [Smith] => Array ( [0] => Stan [1] => Londres [2] => 45 )
)
```

Exercice 2. Écrivez un tableau multidimensionnel associatif dont les clés sont des noms de personne et les valeurs des tableaux associatifs dont les clés sont le prénom, la ville de résidence et l'âge de la personne avec une série de valeurs associées.

Exemple :

```
Array (
    [Dupont] => Array ( [prenom] => Paul [ville] => Paris [age] => 27 )
    [Schmoll] => Array ( [prenom] => Kirk [ville] => Berlin [age] => 35 )
    [Smith] => Array ( [prenom] => Stan [ville] => Londres [age] => 45 )
)
```

Exercice 3. Utilisez une boucle `foreach` pour lire les tableaux des exercices 1 et 2.

Lecture du tableau de l'exercice 1 :

```
Elément Dupont :
élément 0 :Paul
élément 1 :Paris
élément 2 :27
Elément Schmoll :
élément 0 :Kirk
élément 1 :Berlin
élément 2 :35
Elément Smith :
élément 0 :Stan
élément 1 :Londres
élément 2 :45
```

Lecture du tableau de l'exercice 2 :

```
Element Dupont :
prenom :Paul
ville :Paris
age :27
Element Schmoll :
prenom :Kirk
ville :Berlin
age :35
Element Smith :
prenom :Stan
ville :Londres
age :45
```

Exercice 4. Utilisez une boucle `while` pour lire les tableaux des exercices 1 et 2 avec la fonction `each()`.

Lecture du tableau de l'exercice 1 :

```
Personne: Dupont
clé 0 valeur Paul
clé 1 valeur Paris
clé 2 valeur 27
Personne: Schmoll
clé 0 valeur Kirk
clé 1 valeur Berlin
clé 2 valeur 35
Personne: Smith
clé 0 valeur Stan
clé 1 valeur Londres
clé 2 valeur 45
```

Lecture du tableau de l'exercice 2 :

```
Personne: Dupont
prenom:Paul
ville:Paris
age:27
Personne: Schmoll
prenom:Kirk
ville:Berlin
age:35
Personne: Smith
prenom:Stan
ville:Londres
age:45
```

Exercice 5. Créez un tableau contenant une liste d'adresses de sites recommandés :

```
$tab=array(
"PHP"=>"http://www.php.net",
"MySQL"=>"http://www.mysql.org",
"SQLite"=>"http://www.sqlite.org");
```

Puis créez un lien aléatoire vers le premier site de la liste après avoir trié le tableau en ordre aléatoire en utilisant la fonction `array_rand()` qui retourne la clé de l'élément pris au hasard (et non pas `shuffle()`).

Exercice 6. Créez un tableau d'entiers variant de 1 à 63, puis à partir de celui ci un autre tableau de nombres variant de 0 à 6.3. Créez ensuite un tableau associatif dont les clés X varient de 0 à 6.3 et dont les valeurs sont `sin(X)`. Affichez le tableau de valeurs dans un tableau HTML.

Le résultat affiché est le tableau HTML suivant :

```
Tableau de valeurs de la fonction sinus
X sin( X )
0 0
0.1 0.0998334166468
0.2 0.198669330795
0.3 0.295520206661
...
5.9 -0.37387666483
6 -0.279415498199
6.1 -0.182162504272
6.2 -0.0830894028175
6.3 0.0168139004843
```

Exercice 7. Créez un tableau contenant une liste d'adresses e-mail. Extrayez le nom de serveur de ces données, puis réalisez des statistiques sur les occurrences de chaque fournisseur d'accès.

Exemple de résultat :

Fournisseur d'accès : free.com = 16.67 %
Fournisseur d'accès : fiscali.fr = 50 %
Fournisseur d'accès : waladoo.fr = 33.33 %

1.5 TD PHP Formulaires (©J. Engels)

Exercice 1. Créer un formulaire comprenant un groupe de champs ayant pour titre "Adresse client". Le groupe doit permettre la saisie du nom, du prénom, de l'adresse, de la ville et du code postal. Les données sont ensuite traitées par un fichier PHP séparé récupérant les données et les affichant dans un tableau HTML. Utiliser la fonction `stripslashes()` pour supprimer le caractère d'échappement \ qui est automatiquement ajouté dans les chaînes saisies.

Exemple de résultat obtenu :

```
Confirmation de vos coordonnées
nom : Lagarde
prenom : Richard
adresse : 556 Rue de l'Odéon
ville : Paris
code : 75006
```

Exercice 2. Améliorer le script précédent en vérifiant l'existence des données et en affichant une boîte d'alerte JavaScript si l'une des données est manquante.

Exercice 3. Le fichier suivant peut-il être enregistré avec l'extension .php ou .htm? Où se fait le traitement des données?

```
<!DOCTYPE html>
<html>
<head>
<title> Insertion des données </title>
</head>
<body>
<form method="post" action="ajout.php" >
//Suite du formulaire
</form>
</body>
</html>
```

Exercice 4. Comment faire pour que les données soient traitées par le même fichier que celui qui contient le formulaire ? Proposer deux solutions.

1.6 TD PHP Fonctions (©J. Engels)

Exercice 1. Créez une fonction PHP qui affiche une boîte d'alerte à partir de la fonction JavaScript dont la syntaxe est `alert ("chaine_de_caractères")`. Cette fonction peut être appelée avec comme paramètre le texte du message à afficher. Elle est particulièrement utile pour afficher des messages d'erreur de manière élégante, sans que ces derniers restent écrits dans la page. La fonction retourne la valeur TRUE. Ceci n'est pas obligatoire mais peut permettre un contrôle d'exécution.

Exercice 2. Écrivez une fonction de lecture de tableaux multidimensionnels en vous inspirant de l'exemple 7.3. L'affichage se fait sous forme de tableau HTML dont les titres sont les clés des tableaux.

Exercice 3. Écrivez une fonction qui retourne la somme de la série de terme général $u_n = x^{2n} + \frac{1}{n!}$. Les paramètres de la fonction sont n pour le nombre d'itérations et d pour le nombre de décimales affichées pour le résultat.

Exercice 4. Écrivez une fonction dont le paramètre passé par référence est un tableau de chaînes de caractères et qui transforme chacun des éléments du tableau de manière que le premier caractère soit en majuscule et les autres en minuscules, quelle que soit la casse initiale des éléments, même si elle est mixte.

Exercice 5. À partir de la fonction sinus de PHP, écrivez une fonction qui donne le sinus d'un angle donné en radian, en degré ou en grade. Les paramètres sont la mesure de l'angle et l'unité est symbolisée par une lettre. Le deuxième paramètre doit avoir une valeur par défaut correspondant aux radians.

Exercice 6. Réaliser la programmation des coefficients du binôme (ou triangle de Pascal). Pour mémoire, il s'agit de la suite suivante :

```
1  
1 1  
1 2 1  
1 3 3 1  
1 4 6 4 1  
...
```

En remarquant que la première colonne et la diagonale valent toujours 1 et que chaque autre élément est égal à la somme de celui qui est au dessus et de celui qui se trouve sur la diagonale gauche (exemples $3=2+1$ ou $6=3+3$).

Exercice 7. Créez une fonction anonyme qui retourne la division entière de deux nombres.

2 TP PHP

2.1 TP PHP 0 - Environnement de travail

Vous travaillerez sur les postes AIO avec un serveur web **LAMP** (Linux Apache MySQL PHP) que vous lancerez à chaque séance dans un conteneur **Docker**. Ce serveur hébergera, de manière temporaire, vos ressources web (fichiers HTML/CSS, images, etc), vos fichiers PHP et vos données sous MySQL. Nous détaillons ici **l'utilisation** de ces conteneurs. Si vous souhaitez récupérer et installer l'image Docker chez vous, consultez [Wiki LERIA](#).

Docker

- Assurez-vous qu'aucune instance de Firefox n'est en cours d'exécution avant de lancer le conteneur Docker.
- Pour lancer un conteneur Docker, exécutez la commande `_php_8` dans un terminal et attendez que le terminal vous rende la main après avoir affiché les deux comptes d'accès `root` et `phpmyadmin` à l'application web phpMyAdmin (voir plus bas).
- Pour vérifier que le serveur Apache fonctionne, lancez Firefox et chargez l'URL¹ <http://localhost>. Le serveur doit afficher une page listant le contenu de son répertoire racine comprenant le sous-répertoire `Mes_projets_web` et le script `phpinfo.php`.
- Pour vérifier que le module PHP fonctionne sous Apache, chargez <http://localhost/phpinfo.php>. Le script `phpinfo.php` doit générer une page affichant les caractéristiques de l'installation Apache/PHP.
- Pour arrêter et supprimer le conteneur Docker, exécutez la commande `exit`. Si vous supprimez la console de lancement d'un conteneur sans avoir arrêté le conteneur au préalable avec `exit`, il vous faudra exécuter la commande `_clean` avant de pouvoir relancer un nouveau conteneur.

Répertoire de travail & Sauvegarde

Quel que soit l'éditeur utilisé, vous devrez placer tous vos fichiers et répertoires (PHP, JS, HTML, ...) dans le répertoire `Mes_projets_web` situé dans votre répertoire personnel, répertoire auquel accède le serveur LAMP pour servir vos pages web :

- Chemin local : `$HOME/Mes_projets_web`
- Chemin dans le conteneur : `/home/php_dev/Mes_projets_web`
- URL : http://localhost/Mes_projets_web/

Attention : vos fichiers seront détruits une fois le conteneur arrêté, et notamment à chaque déconnexion en fin de séance.

Veuillez donc à les sauvegarder, en utilisant par exemple

- `scp` ou FileZilla pour les transférer dans votre répertoire personnel sur le serveur `starwars` du département informatique ;
- ou `git` pour cloner un dépôt git distant dans `Mes_projets_web` et y sauvegarder vos modifications en fin de séance.

1. Le port 80 de localhost est redirigé (par configuration) vers le port 80 du conteneur sur lequel écoute le serveur Apache donc inutile de le consigner dans l'URL.

Xdebug

PHP est configuré avec **Xdebug** qui permet de relayer avertissements et erreurs PHP en HTML à l'utilisateur/développeur. Testez Xdebug en chargeant le script suivant :

```
<?php
echo "rapport d'erreurs préconfiguré : ".error_reporting()."<br/>";
error_reporting(~E_WARNING);
$a = 2 / 0;
echo 'division par 0<br/>';
error_reporting(~E_NOTICE);
echo $b;
echo 'variable indéfinie<br/>';
error_reporting(~E_ALL);
require "nowhere.file";
echo "ne s'affichera pas<br/>";
?>
```

Commentez ensuite les différents appels à la fonction **error_reporting** utilisant les **constantes prédéfinies** E_WARNING, E_NOTICE et E_ALL et analysez le résultat. Effectuez les mêmes opérations en utilisant PHP en ligne de commande.

phpMyAdmin

phpMyAdmin est une interface web aux SGBD MySQL écrite en PHP. phpMyAdmin est préinstallée dans le conteneur Docker et accessible à l'adresse : <http://localhost/phpmyadmin>.² Vous pourrez vous y connecter avec le compte `root` et le mot de passe qui vous est communiqué par le shell de lancement du conteneur.^{3,4} Toute base de données MySQL que vous créerez devra être exportée sous forme de script MySQL afin d'être sauvegardée et réimportée la session suivante. Export et import se font simplement à partir de l'interface phpMyAdmin. Alternativement, vous pouvez utiliser les commandes suivantes :

- Pour exporter une base de données appelée dbExample :

```
mysqldump -u root -p dbExample > dbExample.sql
```

- Pour importer la base dans le SGBD MySQL :

```
mysql -u root -p < dbExample.sql
```

2. Si vous rencontrez des problèmes d'authentification avec le compte root sur phpmMyAdmin, arrêtez Firefox, relancez le conteneur puis Firefox de nouveau.

3. Attention, les mots de passe sont renouvelés d'une session à l'autre.

4. Ne pas utiliser le compte `phpmyadmin`.

2.2 TP PHP 1 - Premières pages

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le [manuel PHP](#) et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Hello world !

Créez une page web de type “Hello world !” à l'aide d'un script PHP. Le fichier HTML généré doit importer un fichier CSS qui appliquera une couleur de fond de votre choix. L'onglet de la page web affiché par le navigateur sera dénommé TP-PHP-1.1 - Hello.



FIGURE 1 – Première page

Exercice 2. Gabarits et inclusion de fichiers

Implantez le fichier `main.php` dans un sous-répertoire pour produire une page semblable à celle illustrée en Figure 2.

A screenshot of a web browser window. The title bar says "TP-PHP-1.2". The address bar shows "localhost:8080/l2midw/tp-php-1/correctio...". The main content area is divided into sections: "EN-TETE", "CORPS", and "PIED". The "CORPS" section contains the text "Fichier /Users/david.lesaint/Documents/esg/l2mi-dw/wsp/tp-php-1/corrections/exo2/header.php". Below it, the "PIED" section contains the text "Fichier /Users/david.lesaint/Documents/esg/l2mi-dw/wsp/tp-php-1/corrections/exo2/footer.php". At the bottom of the page, there is a link "Le site officiel de PHP". The browser interface includes various icons for file operations like copy, paste, and save.

FIGURE 2 – Page web assemblant en-tête, corps et pied de page

Le fichier `main.php` assemble trois fragments du document HTML produits respectivement par les fichiers `header.php`, `body.php` et `footer.php`. Chacun de ces fichiers affiche différentes constantes et variables qui sont initialisées dans `main.inc.php`:

- Une variable dénotant le numéro du TP apparaissant dans le titre et l'en-tête de la page.
- Une variable dénotant le numéro de l'exercice apparaissant dans le titre et l'en-tête de la page.
- Une constante dénotant la méthode HTTP utilisée pour accéder à `main.php` (cf. superglobale `$_SERVER`) et apparaissant dans le corps de la page.
- Une constante dénotant le fichier en cours d'exécution (cf. superglobale `$_SERVER`) et apparaissant dans le corps de la page.
- Une variable dénotant l'URL (`http://www.php.net`) de l'hyperlien du pied de page.
- Une variable dénotant le contenu texte ("Le site officiel de PHP") de l'hyperlien du pied de page.

Le nom de chaque fichier inclus doit apparaître dans le fragment HTML qu'il génère (cf. constante magique `FILE`). Les caractères é et ° doivent s'afficher correctement quel que soit l'encodage utilisé dans votre navigateur (à modifier via le menu Affichage→Encodage du texte).

2.3 TP PHP 2 - Génération HTML

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le [manuel PHP](#) et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Génération du tableau HTML de comparaison par l'opérateur ==

Le tableau suivant affiche la comparaison par la relation == (opérateur binaire booléen en PHP) des littéraux de l'ensemble { TRUE, FALSE, 1, 0, -1, "1", "0", "-1", NULL, [], "" }. Par exemple, `1=="1"` est vrai (dénoté par TRUE) alors que `1==[]` est faux (dénoté par la chaîne vide).

==	TRUE	FALSE	1	0	-1	"1"	"0"	"-1"	NULL	[]	""
TRUE	TRUE		TRUE		TRUE	TRUE		TRUE			
FALSE	TRUE		TRUE				TRUE		TRUE	TRUE	TRUE
1	TRUE		TRUE			TRUE					
0		TRUE		TRUE			TRUE		TRUE		TRUE
-1	TRUE				TRUE			TRUE			
"1"	TRUE		TRUE			TRUE					
"0"		TRUE		TRUE			TRUE				
"-1"	TRUE				TRUE			TRUE			
NULL		TRUE		TRUE					TRUE	TRUE	TRUE
[]		TRUE							TRUE	TRUE	
""	TRUE		TRUE						TRUE		TRUE

FIGURE 3 – Tableau de comparaison avec ==.

Générez ce tableau HTML à l'aide d'un script PHP en créant un tableau associatif des littéraux à tester sur lequel vous itérerez pour générer les en-têtes de lignes et de colonnes et les cellules "résultats".

Exercice 2. Génération de tableau HTML avec mise en forme conditionnelle

Le fichier **tableau.php** est à compléter pour obtenir la page (partiellement) illustrée en Figure 4. Il contient un tableau \$employees modélisant des employés qu'il s'agit de transformer en tableau HTML à 2 colonnes, le bloc apparaissant à droite étant déjà codé en HTML. L'extrait ci-dessous fournit le code HTML attendu pour les 3 premiers employés :

```
<tr class="bassalaire">
    <td title="7 voyelles, 872 euros">Geraldine Meyer</td>
    <td><input type="number" min="0" max="100" value="95" disabled /></td>
</tr>
<tr class="bassalaire">
    <td title="4 voyelles, 4230 euros">Idona Glenn</td>
    <td><input type="number" min="0" max="100" value="25" /></td>
</tr>
<tr class="hautsalaire">
    <td title="5 voyelles, 9776 euros">Martena Hyde</td>
    <td><input type="number" min="0" max="100" value="70" disabled /></td>
</tr>
```

Chaque ligne HTML correspond à un employé et doit satisfaire aux contraintes suivantes :

- La ligne est classée hautsalaire si le salaire de l'employé est > 5000 et classée bassalaire sinon.
- La première cellule contient le nom de l'employé et une infobulle qui affichera au survol de la souris le nombre de voyelles comptabilisées dans le nom de l'employé ainsi que son salaire.
- La seconde cellule contient un champ numérique affichant l'âge de l'employé : ce champ est désactivé si l'âge de l'employé est > 50.

Geraldine Meyer	95
Idona Glenn	25
Martena Hyde	70
Colette Mcmillan	37
Raya Cook	45
Warren Hendrix	62
Lionel Best	53
Louis Brown	41
Ginger Wolf	77
Sade May	65

Haut salaire > 5000
Bas salaire <= 5000
Désactivé si âge > 50

FIGURE 4 – Tableau d'employés (fichier **tableau.php**).**Exercice 3. Manipulation de tableaux : filtrage, tri, parcours**

L'objectif est ici de calculer des statistiques sur le tableau de données défini dans le fichier **tabEmployees.php** par la variable `$employees`. Vous ne modifierez que le fichier **statistiques.php**. Pour répondre, n'hésitez pas à utiliser les fonctions PHP de manipulation de tableaux (filtrage, tri, parcours, ...). Les 4 statistiques à calculer dans les variables définies au début du fichier sont les suivantes :

- `$ageMoyen` : âge moyen de tous les employés présents dans le tableau `$employees`.
- `$salaireMoyen` : salaire moyen de tous les employés de plus de 55 ans présents dans le tableau `$employees`. Le calcul devra être fait sur un tableau filtré au préalable (contenant uniquement les employés de plus de 55 ans).
- `$tenRichest` : extraire les 10 employés qui ont le meilleur salaire à partir du tableau `$employees`. Ces 10 employés seront stockés dans `$tenRichest` au format "Prénom Nom (salaire)". Par exemple : `$tenRichest = ["Oren Dudley (9985 €)", "Jerome Flynn (9881 €)", ...]`

2.4 TP PHP 3 - Manipulation de tableaux

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le [manuel PHP](#) et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Manipulation de chaînes et tableaux

Complétez le fichier `voyelles.php` afin de créer un tableau associatif qui comptabilise les occurrences des voyelles a, e, i, o, u apparaissant dans le fichier texte `Latin-Lipsum.txt`. Afin de récupérer sous forme de chaîne de caractères le contenu de ce fichier, utilisez la fonction PHP `file_get_contents`. L'affichage du tableau en fin de script doit aboutir au résultat suivant (l'ordre des éléments n'a pas d'importance) :

```
Array
(
    [o] => 147
    [e] => 286
    [i] => 274
    [u] => 191
    [a] => 207
)
```

Appuyez-vous sur les fonctions PHP de manipulation de tableaux et de conversion de chaînes, p.ex., `str_split`, `array_filter`, `in_array`, `array_walk`, `array_key_exists`.

Exercice 2. Tri et transformation de tableaux

Le fichier `comparaison.php` contient un tableau `$langues` comptabilisant des voyelles pour 5 textes écrits dans des langues différentes. On peut comparer et ordonner ces textes en nombre d'occurrences croissant pour une voyelle donnée. Par exemple, `gallois` précède `français` pour la voyelle o. En cas d'égalité, on peut utiliser une nouvelle voyelle pour tenter de discriminer 2 textes. Par exemple, `samoan` précède `anglais` pour l'ordre (o, e) (248 o pour chacun mais 308 e pour le premier contre 335 e pour le second). Plus généralement, on peut trier ces textes sur la base de n'importe quel ordre total sur les voyelles répertoriées. Complétez `comparaison.php` pour réaliser ce tri sur la base de l'ordre des voyelles donné par la variable `$voyelles`. L'affichage attendu est illustré sur la page suivante. Utilisez la fonction `print_r` pour l'affichage des tableaux et appuyez-vous sur les fonctions PHP de tri et de transformation de tableaux, p.ex., `sort`, `usort`, `array_map`.

```
Tri selon l'ordre o,e,i,u,a:
```

```
Array
(
    [0] => gallois
    [1] => français
    [2] => zoulou
    [3] => samoan
    [4] => anglais
)
```

```
Tri selon l'ordre a,e,i,o,u:
```

```
Array
(
    [0] => anglais
    [1] => français
    [2] => gallois
    [3] => zoulou
    [4] => samoan
)
```

2.5 TP PHP 4 - Formulaires

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le [manuel PHP](#) et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Calculatrice

Complétez le fichier **calculatrice.php** pour afficher une mini-calculatrice proposant les opérations d'addition, soustraction, division, et exponentiation sur deux entiers. Les arguments saisis doivent rester affichés après calcul et toute soumission de formulaire incomplet doit être gérée. La Figure 5 illustre le résultat d'un calcul.

FIGURE 5 – Calculatrice

Exercice 2. Agent utilisateur (*User Agent*)

Créez un formulaire de saisie d'adresse e-mail contenant un champ caché destiné à récupérer le type du navigateur de l'utilisateur. Le code PHP affiche l'adresse mail et le nom du navigateur dans la même page après vérification de l'existence des données. La Figure 6 illustre la réponse à une saisie.

Votre mail et votre navigateur	
mail :	a@b.c
navigateur :	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:57.0) Gecko/20100101 Firefox/57.0

FIGURE 6 – Saisie d'adresse e-mail

2.6 TP PHP 5 - Fonctions, Lecture de fichiers

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le [manuel PHP](#) et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Génération de formulaire

Complétez le corps des fonctions du fichier **form_generation.inc.php** permettant de créer le code HTML de deux types de champs de formulaire :

- les champs texte monoligne,
- les boutons de soumission.

Ajoutez une fonction permettant de générer un bouton radio à partir d'un libellé, du nom du champ et de sa valeur. Développez ensuite deux fonctions supplémentaires permettant de générer les balises d'ouverture et de clôture d'un formulaire HTML avec les attributs requis. Complétez enfin le fichier **form_generation.php** qui utilisera ces fonctions pour créer le formulaire illustré ci-dessous. Ce formulaire enverra les données par méthode HTTP POST au fichier **traiter.php**. La Figure 7 illustre la génération d'un formulaire.

FIGURE 7 – Calculatrice

Exercice 2. Conversion de fichier CSV en tableau HTML

Le fichier **villes.csv** modélise différentes villes au format CSV (le point-virgule fait office de séparateur) en donnant pour chacune son nom, sa région, sa population et son statut de préfecture : “1” s'il s'agit d'une préfecture, “0” sinon. Complétez le fichier **population.php** de sorte :

- qu'il lise ce fichier avec une fonction `importerCSV` que vous développerez,
- puis affiche la page illustrée en Figure 8.

Utilisez la fonction `fgetcsv` pour lire et décomposer chaque ligne du fichier.

Adaptez ensuite votre fichier **population.php** de sorte que les villes soient triées par nom de région décroissant, et, pour les villes d'une même région, par population décroissante. Le résultat attendu est illustré en Figure 9.

Nom	Region	Population	Préfecture
Nantes	Pays de la Loire	282047	oui
Avignon	Provence-Alpes-Côte d'Azur	89592	oui
Marseille	Provence-Alpes-Côte d'Azur	850602	oui
Toulon	Provence-Alpes-Côte d'Azur	165514	oui
Saumur	Pays de la Loire	27486	non

FIGURE 8 – Population

Nom	Region	Population	Préfecture
Marseille	Provence-Alpes-Côte d'Azur	850602	oui
Toulon	Provence-Alpes-Côte d'Azur	165514	oui
Avignon	Provence-Alpes-Côte d'Azur	89592	oui
Nantes	Pays de la Loire	282047	oui
Saumur	Pays de la Loire	27486	non

FIGURE 9 – Population triée

Adaptez votre fichier **population.php** afin d'insérer un lien hypertexte sur le texte Nom. Chaque clic sur l'hyperlien permet de commuter entre deux pages :

- Après un nombre pair de clics (notamment au chargement de la page), les villes apparaissent triées selon l'ordre défini précédemment et le tableau a un fond rouge.
- Après un nombre impair de clics, les villes apparaissent en ordre inverse et le tableau a un fond orange.

Le comportement attendu est illustré en Figures 10 et 11.

Nom	Region	Population	Préfecture
Marseille	Provence-Alpes-Côte d'Azur	850602	oui
Toulon	Provence-Alpes-Côte d'Azur	165514	oui
Avignon	Provence-Alpes-Côte d'Azur	89592	oui
Nantes	Pays de la Loire	282047	oui
Saumur	Pays de la Loire	27486	non

FIGURE 10 – Au chargement ou après un nombre pair de clics sur l’hyperlien

Nom	Region	Population	Préfecture
Saumur	Pays de la Loire	27486	non
Nantes	Pays de la Loire	282047	oui
Avignon	Provence-Alpes-Côte d'Azur	89592	oui
Toulon	Provence-Alpes-Côte d'Azur	165514	oui
Marseille	Provence-Alpes-Côte d'Azur	850602	oui

FIGURE 11 – Après un nombre impair de clics sur l’hyperlien

2.7 TP PHP 6 - Crédation de fichiers

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le [manuel PHP](#) et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Génération d'un fichier de journalisation

Créez un fichier qui enregistre la date de chaque connexion et affiche le log des connexions en retour. Exemple d'affichage après plusieurs connexions :

```
24/Jan 09:01:33
24/Jan 09:01:16
24/Jan 09:01:18
24/Jan 09:01:19
24/Jan 09:01:20
24/Jan 09:01:21
24/Jan 09:01:22
```

Utilisez les fonctions :

- `fopen` (en mode lecture ou ajout), `fclose` et `flock` pour verrouiller/déverrouiller l'accès au fichier.
- `time` pour obtenir le [timestamp UNIX](#).
- `date` pour formater l'affichage d'un timestamp, par ex. `date ("d/M H:i:s", $t)` produit l'affichage ci-dessus pour chaque timestamp `$t`.

Exercice 2. Génération et mise à jour de fichier CSV par formulaire

Créez un fichier permettant d'afficher une liste d'étudiants et de s'y inscrire en fournissant nom, prénom et groupe (I1,...I6). Le comportement attendu est illustré en Figures 12 et 13.

Inscrivez-vous ou affichez la liste des étudiants

Nom : Didact

Prénom : Otto

Groupe : I1

Otto Didact : vous êtes inscrits dans le groupe I1

FIGURE 12 – Inscription d'étudiant

ID	NOM	PRENOM	GROUPE
1	De Ronsard	Pierre	I1
2	Du Bellay	Joachim	I1
3	Hugo	Victor	I2
4	Baudelaire	Charles	I6
5	Rimbaud	Arthur	I5
6	Verlaine	Paul	I4
7	Appolinaire	Guillaume	I2
8	Eluard	Paul	I5
9	Aragon	Louis	I2
10	Prévert	Jacques	I6

FIGURE 13 – Listing d'étudiants

Le script créera et mettra à jour un fichier au format CSV contenant les étudiants. Utilisez la fonction `fgetcsv` pour lire et décomposer chaque ligne du fichier. Pour chaque étudiant du listing, insérez un lien hypertexte dans le champ nom permettant de lui envoyer un mail (`...href="mailto:X.Y@etud.univ-angers.fr"...`). Veillez à ce que l'inscription d'un étudiant de nom `<h1 style="color:red">Trompe</h1>` et prénom Ronald se passe correctement.

2.8 TP PHP 7 - Manipulation CSV/JSON

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le [manuel PHP](#) et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Création de fichier CSV

Le fichier **langages.php** contient 5 tableaux, chacun comptabilisant des voyelles d'un texte écrit dans une langue spécifique. Ces tableaux sont regroupés dans un tableau associatif dont les clés identifient chaque langue. Complétez la fonction `creerFichier($f, $v, $l)` qui crée un fichier au format CSV, de nom `$f`, dont la première ligne affiche les voyelles du tableau `$v`, et qui transcrit les différents tableaux de `$l`, comme illustré ci-dessous :

```
LANGUE,o,e,i,u,a  
anglais,248,355,222,107,198  
français,178,435,240,212,201  
gallois,152,264,221,107,266  
samoan,248,308,334,208,656  
zoulou,203,298,311,274,411
```

Cette fonction doit pouvoir être réutilisable avec d'autres jeux de données (autres langues et voyelles).

Exercice 2. Conversion de fichier JSON en tableau PHP

Transformez le contenu du fichier JSON **country-by-capital-city.json** en un tableau associatif dont la clé de chaque élément correspond au nom d'un pays et sa valeur à la capitale du pays. Le tableau sera trié en ordre alphabétique décroissant des noms de pays. Utilisez la fonction PHP dédiée à la conversion de chaînes de caractères au format JSON.

Exercice 3. Lecture et écriture de fichiers CSV et manipulation de tableau PHP

Le fichier **countries.csv** est au format CSV et utilise le point-virgule comme séparateur de champs. Chaque ligne correspond à un pays et indique son continent suivi de son type de gouvernement. Complétez le fichier **csv.php** pour répondre aux questions qui suivent.

1. Extrayez de **countries.csv** les types de gouvernements sous forme de tableau PHP sans se soucier des doublons.
2. Eliminez les doublons du tableau, le trier alphabétiquement, le consigner dans le fichier CSV **governments.csv** (fichier à une seule colonne) et l'afficher avec la fonction PHP `var_export`.

Deuxième partie
Période 9 : JS

3 TD Javascript

3.1 TD JS Bases ([©J.M. Richer](#))

Décompressez l'archive déposée sur Moodle pour ce TD. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#).

Exercice 1. Conversion entier-hexadécimal-chaîne

Implémentez une fonction JS `toHex(n)` qui prend en paramètre un nombre entier positif ou nul et le convertit en sa représentation hexadécimale sous forme de chaîne de caractères.

Exercice 2. Occurrences de mots

Implémentez une fonction JS `wordsOcc(s)` qui prend en paramètre une chaîne de caractères `s` et détermine le nombre d'occurrences de chaque mot de `s`. Utilisez les méthodes `String.prototype.split` et `Array.prototype.sort`.

Exercice 3. Temps de parcours

Implémentez une fonction JS `tempsDeParcours(st1, st2)` qui prend en paramètre deux chaînes de caractères qui référencent des stations de la ligne de tramway A d'Angers et donne le temps de parcours en minutes entre ces deux stations. On code les stations sous la forme suivante :

```
tram = new Array();
tram.push(new Array("roseraie", "5:51"));
tram.push(new Array("jeanxxiii", "5:54"));
tram.push(new Array("les gares", "6:01"));
tram.push(new Array("foch", "6:04"));
tram.push(new Array("ralliement", "6:07"));
tram.push(new Array("saint serge", "6:11"));
tram.push(new Array("chu", "6:14"));
tram.push(new Array("jean moulin", "6:18"));
tram.push(new Array("verneau", "6:22"));
tram.push(new Array("bois du roy", "6:27"));
tram.push(new Array("saint gilles", "6:30"));
tram.push(new Array("ardenne", "6:33"));
station1 = "chu";
station2 = "ardenne";
minutes = tempsDeParcours(tram, station1, station2);
document.write("temps de parcours de " + station1 + " à " + station2 + " = " + minutes + "min");
```

Exercice 4. Nombres premiers

Créez une page web [nombres-premiers.html](#) pour afficher les nombres premiers dans l'intervalle 1 à 50. Vous implémenterez une fonction JS `is_prime(n)` qui renvoie `true` si `n` est premier et `false` dans le cas contraire. Le résultat attendu est illustré en Figure 14. Utilisez la méthode `Array.prototype.join` pour l'affichage.

Nombres premiers

Voici la liste des nombres premiers entre 1 et 50 : 1, 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47

FIGURE 14 – Nombres premiers dans [1, 50].

4 TP Javascript

4.1 TP JS 0 - Outils de développement

Ce document présente différentes méthodes et outils pour développer, tester et publier vos pages Web.

Publier ses pages Web sur le serveur du département informatique

Vous disposez d'un compte sur le serveur Linux starwars du département informatique. L'accès se fait avec l'identifiant LDAP et mot de passe que vous utilisez habituellement pour vous connecter à l'intranet de l'université. Vous pouvez vous y connecter à distance en utilisant le protocole ssh sur le port 2019⁵. Par exemple, pour le compte otto.didakt :

```
ssh -p 2019 otto.didakt@leria-etud.univ-angers.fr
```

Dans votre répertoire personnel sur starwars figure un dossier nommé **public_html**. Tout fichier placé dans ce dossier et dont les permissions excèdent 004 (-r) est publiquement accessible sur le Web (cf. [cloud pédagogique](#)). Précisément, toute requête HTTP(S) pour accéder à l'une de ces ressources est traitée par un serveur web qui y répond en renvoyant la ressource si les permissions le permettent.

L'URL d'accès au dossier **public_html** de l'étudiant otto.didakt est la suivante (ne pas oublier le tilde) :

```
https://leria-etud.univ-angers.fr/~otto.didakt
```

Les URL d'accès aux fichiers de chemins relatifs **f.html** et **rep/g.html** dans **public_html** seront⁶ :

```
https://leria-etud.univ-angers.fr/~otto.didakt/f.html
https://leria-etud.univ-angers.fr/~otto.didakt/rep/g.html
```

Transférer des fichiers sur le serveur du département informatique

La méthode recommandée pour transférer des fichiers dans son dossier **public_html** sur le serveur starwars est de procéder par montage sshfs. En supposant que vous êtes connectés à un poste AOI de salle PC, exécutez les commandes suivantes :

```
$ mkdir ~/public_html_starwars
$ sshfs otto.didakt@leria-etud.univ-angers.fr:/home/otto.didakt/public_html/ ~/
  public_html_starwars/ -p 2019
$ echo "Mon premier test" > ~/public_html_starwars/test1.html
```

Puis chargez la page que vous venez de créer à cette [adresse](#). Vous pouvez dorénavant modifier ce fichier à volonté sur votre AIO à partir de n'importe quel éditeur. Notez que le montage est à ré-établir à chaque session.

Alternativement, utilisez une application telle [Filezilla](#) : créez un "site" en choisissant le protocole SFTP et en renseignant votre identifiant de connexion, mot de passe, numéro de port 2019 et URI <sftp://leria-etud.univ-angers.fr> (Figure 15). Une fois connecté, glissez-déposez vos fichiers de votre dossier local vers le dossier **public_html**. Vous pouvez modifier les permissions d'un fichier/dossier par clic droit sur l'élément.

Alternativement, utilisez **scp** en ligne de commande pour transférer vos fichiers (utilisez **-P** et non pas **-p**) puis fixez les permissions récursivement sur **public_html** avec **ssh**. Par exemple,

```
$ scp -P 2019 nocovid.html otto.didakt@leria-etud.univ-angers.fr:~/public_html/
$ ssh -p 2019 otto.didakt@leria-etud.univ-angers.fr "chmod -R 755 public_html"
$ ssh -p 2019 otto.didakt@leria-etud.univ-angers.fr "ls public_html"
```

5. Consultez le [Wiki](#) du département informatique.

6. Il est parfois nécessaire de rafraîchir la page dans le navigateur pour qu'elle soit correctement chargée !

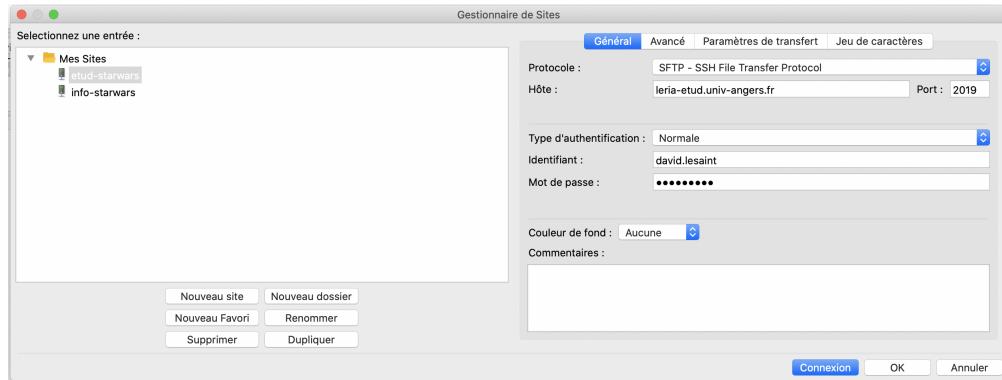


FIGURE 15 – Transfert avec Filezilla

Les EDI pour développer en HTML/CSS/JS/PHP

Un environnement de développement intégré (EDI ou IDE en anglais) fournit l’outillage nécessaire au développement logiciel (éditeur, interpréteur/compilateur, débogueur ...) pour donner plus de confort et améliorer la productivité. Certains EDI sont dédiés à un langage de programmation ou une technologie particulière, d’autres permettent d’intégrer différents langages par le biais de plugins ou d’extensions. Parmi les fonctionnalités qu’offrent les EDI pour le développement web autour du HTML, du CSS et des langages de script côtés client et serveur (Javascript, PHP ...), citons :

- la coloration syntaxique du code source en fonction du langage utilisé,
- l’auto-complétion en cours d’édition,
- le formattage automatique du code (indentation ...),
- la génération de code automatique (cf. [Emmet.io](#))
- l’affichage des erreurs dans le code source avant interprétation,
- un terminal d’exécution,
- un navigateur intégré,
- outils de débogage et profilage,
- le déploiement automatique de site sur un serveur, etc.

Quelques exemples d’EDI⁷ :

- [Atom](#) : licence open source - développé par [GitHub](#)
- [Brackets](#) : licence open source - axé développement web
- [Eclipse](#) : licence open source - développé par la fondation Eclipse
- [Geany](#) : licence open source
- [NotePad++](#) : licence open source
- [Visual Studio Code \(VSCode\)](#) : licence open source - développé par [Microsoft](#)
- [VSCodium](#) : licence open source - fork de VSCode
- [PHPStorm/Webstorm/PyCharm/...](#) : outils payants ou en version gratuite développés par JetBrains
- [Sublime Text](#) : licence propriétaire avec période d’utilisation gratuite

Nous présentons ci-dessous les étapes d’installation de VSCodium qui vous permettra notamment d’éditer à distance vos fichiers placés sur le serveur starwars ou d’y transférer des fichiers.

Installation

VSCodium est multi-plateforme et vous pouvez donc l’installer que vous utilisez Linux, Mac ou Windows. Consultez cette [page](#).

7. En tant qu’étudiant, vous avez aussi accès à de nombreux outils gratuits avec le [GitHub Student Developer Pack](#).

Utilisation

L'EDI offre des raccourcis claviers pour diverses opérations (cf. Figure 16). Pour un aperçu des raccourcis, tapez F1 puis Help : Keyboard Shortcuts Reference.

 Visual Studio Code Keyboard shortcuts for Windows	Ctrl+M Toggle Tab moves focus Search and replace Ctrl+F Find Ctrl+H Replace F3 / Shift+F3 Find next/previous Alt+Enter Select all occurrences of find match Ctrl+D Add selection to next Find match Ctrl+K Ctrl+D Move last selection to next Find match Alt+K R / W Toggle case-sensitive / regex / whole word Multi-cursor and selection Alt+Click Insert cursor Ctrl+Alt+ 1 / 1 Insert cursor above / below Ctrl+U Undo last cursor operation Shift+Alt+1 Insert cursor at end of each line selected Ctrl+L Select current line Ctrl+Shift+L Select all occurrences of current selection Ctrl+F2 Select all occurrences of current word Shift+Alt+→ Expand selection Shift+Alt+ (drag mouse) Shrink selection Shift+Alt+ (drag mouse) Column (box) selection Ctrl+Shift+Alt+ (arrow key) Column (box) selection Ctrl+Shift+Alt+ (arrow key) Column (box) selection page up/down Rich languages editing Ctrl+Space Trigger suggestion Ctrl+Shift+Space Trigger parameter hints Shift+Alt+F Format document Ctrl+K Ctrl+F Format selection F12 Go to Definition Alt+F12 Peek Definition Ctrl+K F12 Open Definition to the side Ctrl+. Quick Fix Shift+F12 Show References F2 Rename Symbol Ctrl+K Ctrl+X Trim trailing whitespace Ctrl+K M Change file language	File management Ctrl+N New File... Ctrl+O Open File... Ctrl+S Save Ctrl+Shift+S Save As... Ctrl+K S Save All Ctrl+F4 Close Ctrl+K Ctrl+W Close All Ctrl+Shift+T Reopen closed editor Ctrl+K Enter Keep preview mode editor open Ctrl+Tab Open next Ctrl+Shift+Tab Open previous Ctrl+K P Copy path of active file Ctrl+K R Reveal active file in Explorer Ctrl+K O Show active file in new window/instance Display F11 Toggle full screen Shift+Alt+0 Toggle editor layout (horizontal/vertical) Ctrl+ + Zoom in/out Ctrl+ - Zoom out Ctrl+B Toggle Sidebar visibility Ctrl+Shift+E Show Explorer / Toggle focus Ctrl+Shift+F Show Search Ctrl+Shift+G Show Source Control Ctrl+Shift+D Show Debug Ctrl+Shift+X Show Extensions Ctrl+Shift+H Replace in files Ctrl+Shift+J Toggle Search details Ctrl+Shift+U Show Output panel Ctrl+Shift+V Open Markdown preview Ctrl+K V Open Markdown preview to the side Ctrl+K Z Zen Mode (Esc Esc to exit) Debug F9 Toggle breakpoint F10 Start/Continue Shift+F5 Stop F11 / Shift+F11 Step into/out F10 Step over Ctrl+K Ctrl+I Show hover Integrated terminal Ctrl+ Create integrated terminal Ctrl+Shift+ Create new terminal Ctrl+C Copy selection Ctrl+V Paste into active terminal Ctrl+I / Scroll up/down Shift+PgUp / PgDn Scroll page up/down Ctrl+Home / End Scroll to top/bottom <small>Other operating systems' keyboard shortcuts and additional unassigned shortcuts available at aka.ms/vscodkeybindings</small>
---	--	---

FIGURE 16 – Liste des raccourcis clavier de VSCode pour Windows

Installer son propre serveur web

Pour installer un serveur web *AMP (Apache, MySQL, PHP) sur votre ordinateur, machine virtuelle ou conteneur, consultez les sites suivants selon le système d'exploitation :

- **XAMPP** pour Linux, Mac, Windows.
- **LAMP** pour Linux.
- **WAMP** pour Windows.
- **MAMP** pour Mac.

Tester avec les outils du navigateur

Pensez à ouvrir systématiquement le panneau "outils de développement" de Firefox pour chacune des pages que vous chargez (Figure 17).

La console relaie notamment les erreurs d'imports de fichiers (CSS, images, etc.) et d'exécution du code JS. L'onglet Réseau est utile pour examiner les échanges HTTP entre navigateur et serveurs (p. ex. données de formulaires). L'inspecteur permet de visualiser et modifier dynamiquement le code HTML de la page (Figure 18). L'éditeur de style CSS permet de modifier les règles CSS (Figure 19).

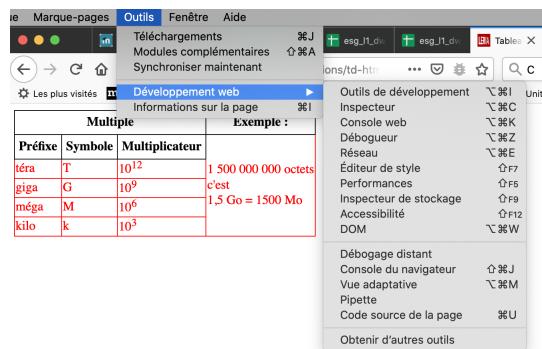


FIGURE 17 – Accéder aux outils de développement de Firefox

The screenshot shows the Firefox developer tools interface with two panels open: 'Inspecteur' (Inspector) on the left and 'Éditeur de style' (Style Editor) on the right. The Inspector panel displays the HTML structure of a table with rows for prefixes like tera, giga, méga, and kilo. The Style Editor panel shows the CSS code for the table's styling, including color, border, and padding properties. A red box highlights the 'Multiples' table in both panels.

Préfixe	Symbol	Multiplicateur	Exemple :
tera	T	10^{12}	1 500 000 000 octets
giga	G	10^9	c'est
méga	M	10^6	1,5 Go = 1500 Mo
kilo	k	10^3	

```

<table>
  <thead>
    <tr>
      <th>Préfixe</th>
      <th>Symbol</th>
      <th>Multiplicateur</th>
      <th>Exemple :</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>tera</td>
      <td>T</td>
      <td>10<sup>12</sup></td>
      <td>1 500 000 000 octets</td>
    </tr>
    <tr>
      <td>giga</td>
      <td>G</td>
      <td>10<sup>9</sup></td>
      <td>c'est</td>
    </tr>
    <tr>
      <td>méga</td>
      <td>M</td>
      <td>10<sup>6</sup></td>
      <td>1,5 Go = 1500 Mo</td>
    </tr>
    <tr>
      <td>kilo</td>
      <td>k</td>
      <td>10<sup>3</sup></td>
      <td></td>
    </tr>
  </tbody>
</table>

```

FIGURE 18 – L'inspecteur HTML

Jouer dans les “bacs-à-code” (code playgrounds)

De nombreux sites internet proposent un EDI multi-fenêtré pour le développement Web côté client. Ces sites permettent de développer le code HTML, CSS et Javascript en ligne et génèrent le rendu automatiquement. Ils ne nécessitent aucune installation d’outil ou de plug-in pour être utilisables. A noter que les outils de Firefox ne sont d’aucune utilité dans ce contexte. Quelques exemples : [Liveweave](#), [JS Fiddle](#), [CodePen](#), [JS Bin](#), [CSSDeck](#), [Dabblet](#), [CodeSandbox](#).

La solution du pauvre

Une approche non recommandée consiste à charger directement dans le navigateur un fichier HTML qui est enregistré sur votre ordinateur (l’adresse affichée dans la barre de navigation débute par `file://`). Cette méthode présente de sérieux risques de sécurité et ne demande aucun échange HTTP avec un serveur web.

4.2 TP JS 1 - Premiers pas en JS

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Console, alertes, boîtes d'invite

Lancez Firefox, chargez le fichier `console.html`, ouvrez la console du navigateur et mettez-vous en mode d'édition multi-lignes. Exécutez ensuite dans la console les instructions qui vous sont demandées pour chaque question (en les conservant toutes dans l'éditeur).

1. Créez une constante `x` stockant la chaîne de caractères "15" et l'affichez dans la console.
2. Re-exécutez vos instructions. Qu'observez-vous ? Rechargez la page. Re-exécutez vos instructions. Qu'observez-vous ?
3. Affectez la valeur `true` à `x`. Qu'observez-vous ?
4. Commentez cette dernière instruction.
5. Transformez votre constante `x` en une variable.
6. Affichez l'invite "Saisir une valeur" dont le champ de saisie affiche la valeur de `x` par défaut. Stockez la réponse du visiteur dans une nouvelle variable `y` dont vous afficherez et le type et la valeur dans la console. Testez les 3 cas possibles : Annuler, OK en conservant la valeur par défaut, OK en changeant la valeur par défaut. L'affichage attendu est illustré en Figure 20.



FIGURE 20 – Affichage de l'invite

Exercice 2. Débogueur

1. Lancez Visual Studio. Copiez-collez les instructions de l'exercice précédent dans le fichier fourni `./debugger/debugger.js`. L'importez dans `debugger/debugger.html` et chargez ce dernier dans le navigateur.
2. Faites en sorte que le paragraphe encodé dans le fichier HTML s'affiche après que le visiteur ait renseigné sa réponse dans l'invite, et non pas avant.
3. Placez-vous dans le débogueur de Firefox et y affichez le fichier `debugger.js`. Placez un point d'arrêt sur la première ligne. Rechargez la page puis exécutez le script pas à pas en cliquant sur la deuxième flèche (Passez la fonction) du panneau droit du débogueur. Observez les valeurs que prennent vos variables `x` et `y` à chaque pas (en les survolant à la souris, ou bien en consultant la rubrique Portées, objet Window, qui s'affiche automatiquement dans le panneau droit).
4. Supprimez le point d'arrêt sur la première ligne et en placez-un sur l'instruction affichant l'alerte contenant `x`. Rechargez la page. Au point d'arrêt, modifiez la valeur de `x` dans la console et poursuivez l'exécution en pas à pas. Qu'observez-vous ?
5. Dans la console, tapez `window.` puis cherchez `x` parmi les propriétés suggérées, complétez en choisissant `x` et exécutez. Qu'en déduisez-vous ? Modifiez la valeur de `y` via l'objet `window` dans la console.

Exercice 3. Conversion chaîne-entier

prompt.html importe **prompt.js** et affiche le résultatat d'un appel à la fonction `prompt_positive_number()`.

1. Implémentez cette fonction qui demande un nombre au visiteur et convertit la chaîne saisie en nombre entier en utilisant `parseInt`. La fonction doit vérifier qu'il s'agit bien d'un nombre et qu'il est positif avant de le renvoyer. Elle réitère la demande dans le cas contraire.

2. Ajoutez un paramètre d'entrée à la fonction dénotant la base de représentation des entiers utilisable par le visiteur (2, 8, 10, etc.). Modifiez le corps de la fonction pour que la conversion puisse fonctionner selon la base passée en argument (p. ex. le visiteur devra saisir un nombre octal si la base vaut 8). Etendez **prompt.html** pour afficher le résultatat d'un appel en base 2 dans un titre de deuxième niveau. Testez. Etendez **prompt.html** pour afficher le résultatat d'un appel en base 8 dans un titre de troisième niveau. Testez. L'affichage attendu est illustré en Figure 21.

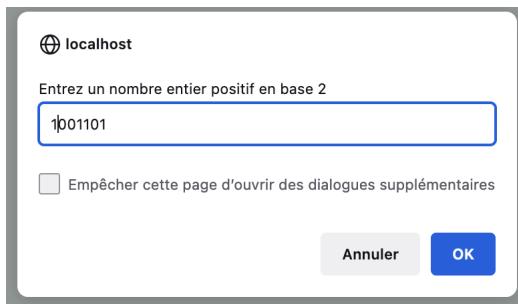


FIGURE 21 – Affichage de l'invite

Exercice 4. Chaînes

toNumber.html importe **toNumber.js** et affiche le résultatat de différents appels à la fonction `toNumber()`. Cette dernière attend en entrée une chaîne de caractères et renvoie le nombre entier (positif ou négatif) correspondant. Elle doit donc vérifier que la chaîne communiquée correspond à la représentation décimale d'un entier. Dans le cas contraire, elle renvoie la valeur `Nan`.

1. Implémentez cette fonction sans utiliser `parseInt` et en vous aidant de l'opérateur `typeof` et des méthodes `String.prototype.charAt` et `String.prototype.charCodeAt`. L'affichage attendu est illustré en Figure 22.

toNumber

```
toNumber('234') 234 (number)
toNumber('-11') -11 (number)
toNumber('-3.1415') NaN (number)
toNumber('ab') NaN (number)
parseInt('123ab') 123 (number)
parseInt('[0]') NaN (number)
parseInt(/\^123/) NaN (number)
```

FIGURE 22 – Conversion de différentes chaînes

4.3 TP JS 2 - Chaines et Tableaux

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Occurrences de sous-chaînes

Le fichier **nbOcc.html** importe **nbOcc.js** et affiche le résultat d'appels à la fonction `nbOcc()`. Cette dernière prend en paramètre une chaîne de caractères `s` et une sous-chaîne `ss` et détermine le nombre d'occurrences de `ss` dans `s`. Implémentez cette fonction en vous aidant de fonctions telles `indexOf`, `search` ou `substring`. L'affichage attendu est illustré en Figure 23.

nbOcc

doigts dans `Les doigts boudinés de l'ogresse, les doigts acérés de l'ogresse, ses doigts qu'il fallait à tout prix satisfaire.` = 3

notre dans `Le nombre faisait notre force et notre faiblesse; notre force, parce que nous étions une armée; notre faiblesse, car nous étions désunis.` = 4

FIGURE 23 – Recherche d'occurrences de sous-chaînes

Exercice 2. Parcours de tableaux

Le fichier **matrix.html** importe **matrix.js** et affiche le résultat d'appels à la fonction `sMatrix()`. Implémentez cette fonction qui prend en entrée une "matrice" d'entiers (un tableau de tableaux) et l'affiche sous forme matricielle dans une alerte. Utilisez impérativement la méthode d'itération `forEach`. L'affichage attendu est illustré en Figure 24.

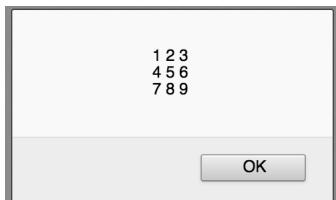


FIGURE 24 – Matrice d'entiers dans une alerte

Exercice 3. Manipulation de tableaux

Le fichier `voyelles.html` importe `voyelles.js` et définit deux chaînes, l'une `s` contenant un texte “*“Lorem ipsum...”*”, l'autre `voyelles` contenant la liste des voyelles. L'affichage attendu est illustré en Figure 25.

1. Transformez `s` en un tableau de caractères `tab` soit à l'aide d'une méthode de `String`, soit à l'aide d'une méthode de `Array`.
2. Eliminez de `tab` tout caractère qui n'est pas une voyelle en vous servant de `voyelles` et stockez le tableau résultat dans `tabv`.
3. Créez à partir de `tabv` un tableau associatif (objet `Map`) qui associera à chaque voyelle le nombre de ses occurrences dans `tabv`.
4. Déterminez la voyelle ayant le plus grand nombre d'occurrences et stockez la dans `maxv`.

Voyelles

1. `tab = L,o,r,e,m, i,p,s,u,m, ,d,o,l,o,r, s,i,t, a,m,e,t,,, c,o,n,s,e,c,t,e,t,u,r, ,a,d,i,p,j,s,c,i,n,g, e,l,i,t,, ,S,e,d, ,n,o,n, r,i,s,u,s,, ,S,u,s,p,e,n,d,i,s,s,e, ,l,e,c,t,u,s, ,t,o,r,t,o,r,, ,d,i,g,n,i,s,s,i,m, ,s,i,t, ,a,m,e,t,,, ,a,d,i,p,j,s,c,i,n,g, ,n,e,c,, ,u,l,t,r,i,c,i,e,s, ,s,e,d,,, ,d,o,l,o,r,,`
2. `tabv = o,e,i,u,o,o,i,a,e,o,e,e,u,a,i,i,e,i,e,o,i,u,u,e,i,e,e,u,o,o,i,i,i,a,e,a,i,i,e,u,i,i,e,e,o,o`
3. `map = a,4,e,13,i,17,o,9,u,6,y,0`
4. `maxv = i=17`



FIGURE 25 – Page web et sortie console attendues

4.4 TP JS 3 - Objets

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Rappel sur objets et prototypes en JS

```
// constructeur
function Foo(y) {
    // propriété en propre pour chaque objet construit avec Foo
    this.y = y;
}
// propriété héritée par tout objet construit avec Foo
Foo.prototype.x = 10;
// méthode héritée par tout objet construit avec Foo
Foo.prototype.calculate = function (z) {
    return this.x + this.y + z;
};
var b = new Foo(20);
var c = new Foo(30);
b.calculate(30); // 60
c.calculate(40); // 80
console.log(
    b.__proto__ === Foo.prototype, // true
    c.__proto__ === Foo.prototype, // true
    b.constructor === Foo, // true
    c.constructor === Foo, // true
    Foo.prototype.constructor === Foo, // true
    b.calculate === b.__proto__.calculate, // true
    b.__proto__.calculate === Foo.prototype.calculate // true
);
```

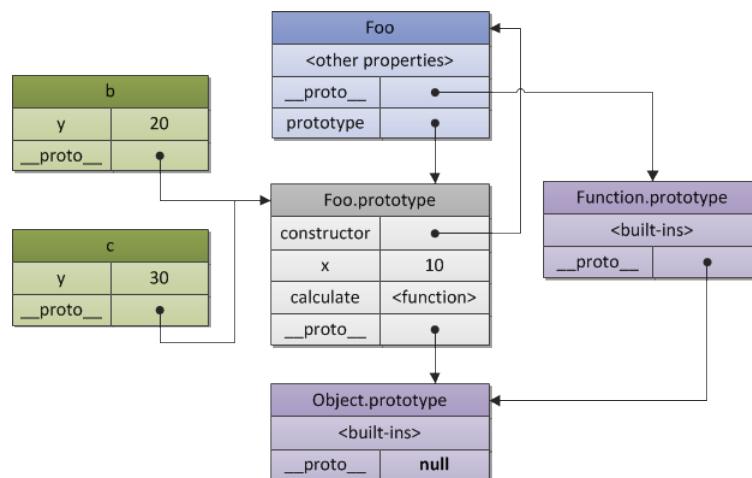


FIGURE 26 – Liens entre objets et prototypes

Exercice 1. Construction d'objets

Complétez **dalton.html** pour y importer en mode différé les fichiers **dalton1.js**, **dalton2.js** et **dalton3.js** dans cet ordre. Complétez ensuite **dalton1.js** pour répondre aux questions de cet exercice.

1. Créez un constructeur Dalton modélisant un “Dalton” par son prénom.
2. Créez un Dalton averell de prénom “Averell”.
3. Créez une fonction `log` prenant un objet en paramètre et l'affichant à la console ainsi que sa propriété prénom. Appelez la avec `averell`.
4. Créez un Dalton avec `Object.create` et fixez son prénom à “Jack”. Affichez cet objet avec `log`.
5. Créez un objet littéral `joe` ayant “Joe” pour valeur de la propriété `prénom`. Affectez un objet anonyme de type Dalton à son prototype. L'afficher avec `log`.
6. Créez un objet `william` à partir de la chaîne au format JSON '`{ "prénom" : "William" }`' en utilisant la méthode `JSON.parse`. Modifiez son prototype comme ci-dessus. L'afficher avec `log`.
7. '`{ "prénom" : "William" }","{ "prénom" : William }`' et '`{ "prénom" : "William", }`' sont-elles des chaînes au format JSON ?
8. Comparez la propriété `prototype` du constructeur Dalton avec le prototype objet de chacun des 4 objets Dalton en y accédant avec `Object.getPrototypeOf()`. Remontez la chaîne de prototype si nécessaire pour obtenir l'égalité.
9. Invoquez la méthode `hasOwnProperty` sur l'objet `william` pour tester si la propriété `prénom` est une propriété (en) propre de cet objet. Invoquez `Object.getOwnPropertyNames` avec le paramètre `william` pour récupérer ses propriétés propres. Invoquez `Object.keys` avec le paramètre `william` pour récupérer ses propriétés propres et énumérables (itérables). Itérez sur toutes les propriétés énumérables (propres ou héritées) de l'objet `william` avec une boucle `for ... in`.

Exercice 2. Prototypes

Complétez **dalton2.js** pour répondre aux questions de cet exercice.

1. Ajoutez aux prototypes des objets construits avec `Dalton` une propriété `nom` égale à “Dalton”.
2. Ajoutez aux prototypes des objets construits avec `Dalton` une méthode `afficher` qui est sans arguments et affiche leurs nom et prénom dans la console.
3. Créez un tableau `daltons` contenant les 4 objets `DALTON`. Invoquez la méthode `afficher` sur ses éléments en itérant sur `daltons` avec une boucle `for ... of`. Même question mais en utilisant la méthode `map` (héritées par les objets `Array`).
4. Invoquez la méthode `hasOwnProperty` sur l'objet `william` pour vérifier que `nom` est une propriété héritée par cet objet. Itérez sur l'objet `william` avec une boucle `for ... in` pour afficher clé et valeur de ses propriétés énumérables. Itérez sur le tableau `daltons` avec une boucle `for ... in` pour afficher clé et valeur de ses propriétés énumérables. Que représentent les clés affichées ?
5. Détruissez la propriété `nom` du constructeur `DALTON` et réaffichez les objets `DALTON` du tableau.
6. Affichez parmi les 4 objets `DALTON` du tableau ceux dont le prénom commence par la lettre `J`. Chainez les méthodes de tableaux `filter` et `map` à cet effet.

Exercice 3.

Complétez **dalton3.js** pour répondre aux questions de cet exercice.

1. Créez un constructeur `Famille` prenant une chaîne `nom` en paramètre, et définissant pour “ses” objets une propriété `nom` égale à cette chaîne si elle définie ou à la chaîne vide sinon, et une propriété `membres` égale au tableau vide.
2. Construisez un objet `DALTON` de nom `Dalton` avec `Famille`.
3. Ajoutez aux prototypes des objets construits avec `Famille` une méthode `ajouter` prenant un objet en paramètre et ajoutant sa propriété `prénom` en fin du tableau `membres`.
4. Invoquez cette méthode sur `DALTON` pour y ajouter `averell`.
5. Invoquez cette méthode sur `DALTON` pour y ajouter `jack` en utilisant la méthode `call` héritées par les objets `Function`.

6. Invoquez cette méthode sur DALTON sur chacun des éléments du tableau [joe, william] en utilisant la méthode map héritées par les objets Array.
7. Ajoutez aux prototypes des objets créés avec Famille une méthode afficher affichant leur nom et les prénoms de leurs membres. L'invoquer sur DALTON.

Exercice 4.

Complétez personne.html pour y importer en mode différé les fichiers personne.js et professeur.js dans cet ordre. Complétez ensuite personne.js pour répondre aux questions de cet exercice.

1. Créez un constructeur Personne modélisant une personne par son identité (nom et prénom), son âge et ses centres d'intérêts. Ce constructeur fournira deux méthodes produisant des alertes telles que celles illustrées en Figure 27 et 28.

- salutation qui affiche une alerte du type Bonjour, je m'appelle X;
- bio qui affiche une alerte du type Nom : X, Prénom : Y, Age : nn, Centres d'intérêts : I1, I2, ... et In.



FIGURE 27 – Salutations

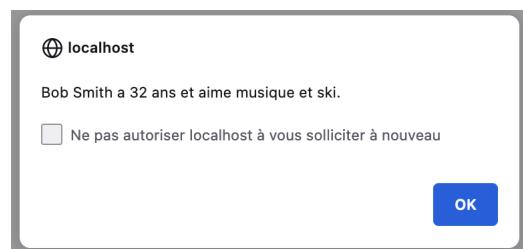


FIGURE 28 – Bio

2. Créez un objet avec ce constructeur, en examinez le contenu et le prototype.
3. Ajoutez une méthode aurevoir à la propriété prototype du constructeur produisant une alerte telle qu'illustrée en Figure 29. Testez cette méthode sur l'objet précédent et en réexaminez le prototype.



FIGURE 29 – Adios

Exercice 5. Héritage prototypique

Complétez `professeur.js` pour répondre aux questions de cet exercice.

1. On modélisera un professeur comme une personne enseignant une matière. Créez un constructeur `Professeur` qui, par héritage prototypique, hérite de `Personne` et ajoute une propriété spécifique `matière`. L'affichage attendu est illustré en Figure 30.

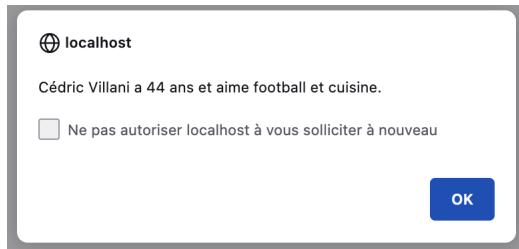


FIGURE 30 – Bio

2. Ajoutez au prototype de `Professeur` une méthode `saluer` qui affiche nom et matière enseignée. L'affichage attendu est illustré en Figure 31.

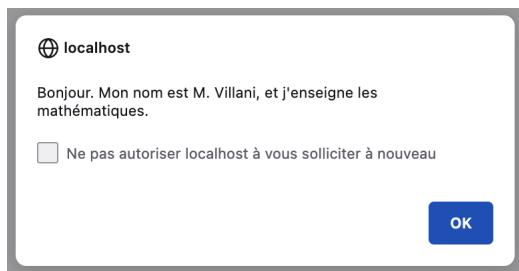


FIGURE 31 – Salutations

4.5 TP JS 4 - Premiers pas avec le DOM

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Survol de l'API DOM HTML

Chargez le fichier **survol.html** dans Firefox. Dézoomez de sorte que toute la page apparaisse dans la zone d'affichage. Ouvrez la console en mode d'édition **multi-lignes** puis y chargez (CTRL-O) le fichier **survol.js**. Décommentez tour à tour les instructions pour en analyser les résultats dans la console et la page web.

```

document.getElementById("TAB");
h23 = document.getElementById("TAB");
h23 === document.querySelector("#TAB");
h23.__proto__;
h23.__proto__ === Object.getPrototypeOf(h23);
h23.nodeType;
h23.nodeName;

div = h23.parentNode;
h23 === div.querySelector("h2:nth-child(17)");
h23 === div.querySelector("h2:nth-of-type(3)");
h23 === div.querySelectorAll("h2")[2];
"BODY" === h23.parentNode.parentNode.tagName;
h23.hasChildNodes();
h23.firstChild;
h23.firstElementChild === null;
p = h23.nextElementSibling;
p === div.querySelector("dl+h2+p");

form = div.lastElementChild.previousElementSibling;
form.getAttributeNames();
true === form.hasAttributes();
"post" === form.getAttribute("method");
form.action === form.getAttribute("method");

div.textContent;
div.innerText;

document.querySelector("h1").textContent = "Nouveau titre";
document.querySelector("pre").style.textAlign = "right";
P = document.querySelectorAll("p");
P.item(0).style.fontFamily = "Cursive";
P.item(2).classList;
P.item(2).classList.remove("important");
P.item(1).classList.add("important");

thds = document.querySelectorAll("th,td");
thds.forEach((e) => { e.style.visibility = "hidden"; });
Array.from(thds).map((e) => { e.style.display = "none"; });

form.querySelector("input[type='text']").value = "Tchatgémolié";
(select = form.querySelector("select[name='h_adr_ville']")).options;
form.querySelectorAll("input[type='radio']").item(2).checked = true;

```

Exercice 2. Endommager

Chargez le fichier **endom.html** (identique au fichier de l'exercice précédent). Exécutez vos instructions dans la console pour répondre aux questions.

1. Remplacez le texte “Formulaires” par ce que vous voulez.
2. Fixez la largeur du premier tableau à 80% de celle de son conteneur.
3. Coloriez en `lightcyan` le fond des éléments de la liste de définition.
4. Remplacez le contenu texte des items des listes par `Hello`.
5. Supprimez le second tableau.
6. Supprimez la troisième ligne du premier tableau.
7. Supprimez la deuxième colonne du premier tableau.
8. Remplissez le champ du code postal avec la chaîne `49000`.
9. Cochez les cases `garage` et `piscine`.

Exercice 3. Dédommager

Le fichier **dedom.html** importe **dedom.js**. Complétez ce dernier en répondant aux questions qui suivent pour obtenir la page illustrée en Figure. 32.

1. Supprimez le formulaire et le titre `Elections` en utilisant la méthode `previousElementSibling`.
2. Centrez le texte de toutes les cellules du tableau.
3. Faîtes apparaître les bordures des cellules du tableau en violet et avec une épaisseur `1px`.
4. Retirez de l'affichage la section `Introduction` et son paragraphe.
5. Appliquez la police de caractères `Cursive` aux items de la première liste si le contenu commence par la lettre `D` sinon afficher l'icône de code `26A0`.

Pays de la Loire

Sommaire

1. 
2. [Deux départements](#)
3. [Démographie](#)
4. 

Deux départements

1. Loire-Atlantique
 - Nantes
 - Préfecture de la région
 - Sixième ville de France
 - Saint-Nazaire
2. Maine-et-Loire
 - Angers
 - Chef-lieu du département
 - Deuxième ville la plus peuplée de la région
 - Cholet

Démographie (en milliers d'habitants)

Année	Départements					Total
	Loire-Atlantique	Maine-et-Loire	Mayenne	Sarthe	Vendée	
2013	1 328	800	307	568	655	3 661
2014	1 343	804	308	570	662	3 689

FIGURE 32 – Pays de la Loire

4.6 TP JS 5 - Création d'éléments

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Image, ancre, paragraphe

Le fichier `drapeau.html` importe `drapeau.js`. Complétez ce dernier pour y créer image, lien et paragraphe tel qu'illustré en Figure 33.

1. Implémentez une fonction `createSimpleNode(tag, options, text)` qui crée et renvoie un objet modélisant l'élément HTML de balise `tag`, d'attributs `options` et de contenu `text` (attributs et noeud texte pouvant être vides).

2. Utilisez cette fonction pour créer l'image, l'[lien](#), le paragraphe et les retours à la ligne nécessaires.



[Un site](#)

Du texte

FIGURE 33 – Crédit à la création d'éléments

Exercice 2. Liste

Le fichier `liste.html` importe `liste.js`. Complétez ce dernier afin de dupliquer la liste existante dans l'élément d'identifiant `output` sans utiliser la méthode de clonage `cloneNode`.

Exercice 3. Tableaux

Le fichier `couleurs.html` importe `couleurs.js`. Complétez ce dernier pour afficher 3 matrices 8×8 de couleurs au chargement de la page (voir Figure 34). Utilisez le format RGB pour les [codes couleurs](#) à cet effet.

Couleurs

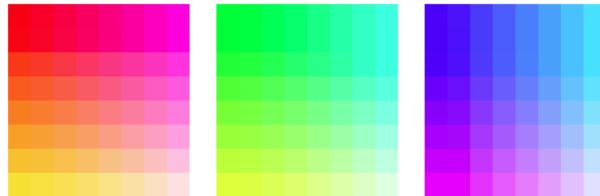


FIGURE 34 – 3 tableaux de couleurs en dégradé

4.7 TP JS 6 - Manipulation statique du DOM

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Boules de fort

Le fichier **bdf.html** importe le fichier **bdf.js** qui définit :

- La variable `societes` qui est un tableau de tableaux, chacun stockant le nom d'une société de boules de fort, son numéro de téléphone, code postal, commune et président (ou présidente). Chaque président est identifié par son prénom suivi de son nom.
- La variable `prenoms_feminins` qui est un tableau de prénoms féminins.
- La variable `departements` qui est un tableau de numéros de départements français.
- La méthode de tableaux `supprimerDoublons()` qui renvoie une copie du tableau sur lequel elle est invoquée en supprimant les doublons.

Complétez **bdf.js** pour obtenir la page illustrée en Figure 35.

The screenshot shows a web application interface on the left and a developer tools console on the right.

Left Side (Search Interface):

Choisir une société :

SOCIÉTÉ	PRÉSIDENT(E)
LA PAIX (37500) Laurence SOREAU	LA PAIX (49540) François CHENE
LA RENAISSANCE (49320) Roselyne EGLANTIER	LA RENAISSANCE (49320) Roselyne EGLANTIER
LE PRIEURE (49730) André MABILLEAU	LE PRIEURE (49730) André MABILLEAU
LES LABOUREURS (49150) Gildas MARGAS	LES LABOUREURS (49150) Gildas MARGAS
L'UNION (49650) Stéphane DEGRAVE	L'UNION (49650) Stéphane DEGRAVE
L'AMBIBOULE (37340) Michel DERRIDDER	L'AMBIBOULE (37340) Michel DERRIDDER
LE ROSIER (49700) Daniel COUTOIS	LE ROSIER (49700) Daniel COUTOIS
STAND LAMORICIERE (44150) Marie-Jo BAUDOUIN	STAND LAMORICIERE (44150) Marie-Jo BAUDOUIN
L'AMICALE (49800) Jean-Luc MOUTEAU	L'AMICALE (49800) Jean-Luc MOUTEAU
LES GROIES (49800) Émile HERVÉ	LES GROIES (49800) Émile HERVÉ
ASPTT L'ORMEAU (49100) Jean-Claude BLANCHARD	ASPTT L'ORMEAU (49100) Jean-Claude BLANCHARD
FRATERNELLE JEAN MACE (49100) Albert ROGER	FRATERNELLE JEAN MACE (49100) Albert ROGER
JEANNE D'ARC (49000) Gisèle JASNault	JEANNE D'ARC (49000) Gisèle JASNault
JULES FERRY (49100) Christian GARREAU	JULES FERRY (49100) Christian GARREAU
A LA PENSEE (49000) Serge LEGROS	A LA PENSEE (49000) Serge LEGROS
LE BON CONSEIL (49100) Vincent CHARRUAU	LE BON CONSEIL (49100) Vincent CHARRUAU
LE CHAMP D'HONNEUR (49000) Jean-Jacques SIMON	LE CHAMP D'HONNEUR (49000) Jean-Jacques SIMON
LES AMIS RÉUNIS (49100) Marie-Claire FRESNAIE	LES AMIS RÉUNIS (49100) Marie-Claire FRESNAIE
NOTRE DAME (49000) Hubert PEHU	NOTRE DAME (49000) Hubert PEHU
SAINT JACQUES (49100) Marc DUTOUR	SAINT JACQUES (49100) Marc DUTOUR
SAINT LÉONARD (49000) Martial LOREAU	SAINT LÉONARD (49000) Martial LOREAU
LA RENAISSANCE (49490) Jean-Paul AVRIL	LA RENAISSANCE (49490) Jean-Paul AVRIL

Right Side (Developer Console):

Console tab open, showing network requests and logs:

- GET http://localhost/l2/cc-21-js/corrections/exercice-2/bdf.html [HTTP/1.1 200 OK 46ms]
- GET http://localhost/l2/cc-21-js/corrections/exercice-2/bdf.js [HTTP/1.1 200 OK 45ms]
- #SOCIETES SANS TELEPHONE = 114
bdf.js:1303:9
- #SOCIETES PAR DEPARTEMENT =
bdf.js:1315:9
- Object { 37: 19, 41: 1, 44: 3, 49: 262, 53: 1, 72: 32 }
bdf.js:1316:9
- PRENOMS TRIES =
Alain,Albert,André,Annie,Auguste,Benoist,Bernard,Bruno,Catherine,Chantal,Christiane,Christophe,Clair,Claude,Cyril,Daniel,Denis,Didier,Dominique,Elaine,Elisabeth,Emile,Emmanuel,Eric,Fabienne,Fabrice,Fernand,Florent,Franck,François,Françoise,Frédéric,Gabriel,Gaëlle,Georges,Gervais,Ghislain,Gilbert,Gildas,Gilles,Gisèle,Guillaume,Guy,Guyalaine,Gérard,Henri,Hervé,Hubert,Jacky,Jacques,Jamais,Jany,Jean,Jean-Claude,Jean-Emile,Jean-François,Jean-Jacques,Jean-Louis,Jean-Luc,Jean-Marie,Jean-Maurice,Jean-Michel,Jean-Noël,Jean-Paul,Jean-Pierre,Jean-Robert,Joseph,Joël,Jérôme,Kamel,Laurence,Laurine,Louis,Loïc,Lucien,Ludo,Marc,Marcel,Marie-Claire,Marie-Jo,Martial,Martine,Maurice,Michel,Minouche,Nicolas,Nicole,Norbert,Noël,Olivier,Pascal,Patrice,Patrick,Paul,Philippe,Pierre,Pierre-Yves,Raymond,Rémy,René,Renée,Robert,Roger,Roland,Roselyne,Régis,Serge,Stéphane,Suzanne,Sylvie,Teddy,Thierry,Tiphaine,Tony,Venceslas,Vincent,Virginie,Yannick,Yolande,Ivan,Yves,Émile,Éric
bdf.js:1323:9
- GET http://localhost/favicon.ico [HTTP/1.1 200 OK 2ms]

FIGURE 35 – Page web et affichage en console attendus

1. Calculez le nombre de sociétés dont le numéro de téléphone est la chaîne `xx-xx-xx-xx-xx-xx`. Stockez le résultat dans la variable `sans_telephone`.
2. Construisez l'objet `distribution` qui associe à chaque numéro de département figurant dans le tableau `departements` le nombre de sociétés de ce département (les deux premiers chiffres du code postal d'une société correspondent à son numéro de département).
3. Extrayez du tableau `societes` les prénoms des présidents (ou présidentes), supprimez les doublons et triez les en ordre alphabétique. Stockez le résultat dans le tableau `prenoms`.
4. Le fichier HTML contient un tableau à 3 colonnes libellées “SOCIETE”, “CP” et “PRESIDENT(E)”. Complétez-le en ajoutant une ligne par société avec les champs attendus. Si le président d'une société a l'un des prénoms féminins enregistrés dans le tableau `prenoms_feminins`, attribuez à la case HTML correspondante la classe `presidente`

qui colorera automatiquement le fond de la cellule en rose via les règles CSS. Utilisez la propriété `classList` et les méthodes `insertRow` et `insertCell`.

5. Le code JS fourni pour cette question construit un objet avec le constructeur `Societe` pour chaque société et invoque sur cet objet la méthode `toHTML` en lui passant la `data-list HTML` d'identifiant `societes`. Une fois ce code exécuté, le champ texte de la page web suggère différentes options au visiteur dès qu'il saisit des caractères (voir Figure 35). Implémentez le constructeur `Societe` qui stocke pour chaque objet créé le nom de la société, son code postal et son président. Ajoutez ensuite, sans modifier le constructeur, la méthode `toHTML(datalist)` qui, lorsqu'elle est invoquée sur un objet `Societe` avec une `data-list HTML` en argument :

- Ajoute à cette `data-list` un sous-élément de balise `option`.
- Initialise l'attribut `HTML value` de cette option avec le texte concaténant le nom, code postal et nom de président stockés dans l'objet.

Exercice 2. TUX

Le fichier `tux.html` importe `tux.js`. Complétez ce dernier pour obtenir la page illustrée en Figure 37.



Sondage

Connaissiez-vous TUX ? Oui Non
E-mail :

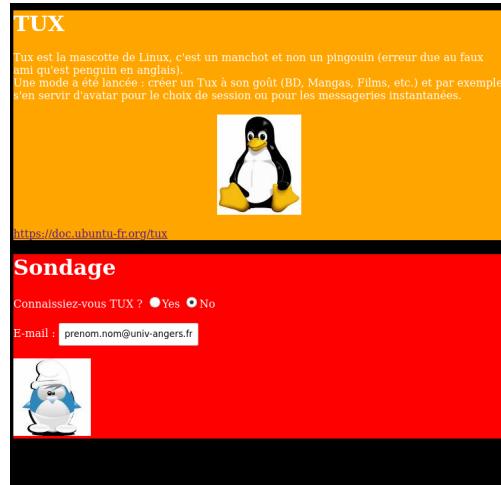


FIGURE 36 – Page avant exécution JS

FIGURE 37 – Page après exécution JS

1. Modifiez la couleur d'arrière plan du corps du document (`<body>`) en noir. Modifiez les couleurs d'arrière plan des divisions (`<div>`) d'identifiant `intro` et `sondage`, respectivement, en orange et en rouge. Modifiez la couleur des éléments de chaque division du document en blanc.
2. Centrez toutes les images de la division d'identifiant `intro`. Pour ce faire, vous pouvez utiliser la classe `center` définie dans `tux.html`.
3. Dans la division `sondage` :
 - Tous les champs de saisie d'un email doivent prendre comme valeur `nom.prenom@univ-angers.fr`
 - Toutes les cases de valeur `non` doivent être cochées.
 - Les labels de toutes les cases de valeur `oui` et `non` doivent être, respectivement, `Yes` et `No`.
4. Ajoutez pour chaque image une description textuelle (`alt`) indiquant où se situe l'image (arborescence DOM jusqu'à l'élément `body`). Par exemple pour la première image : `alt="Image - P (parIntro) - DIV (intro)"` indique que l'image est dans le paragraphe (P) d'identifiant `parIntro` qui est dans la division (DIV) d'identifiant `intro`.

4.8 TP JS 7 - Manipulation statique du DOM

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Triangle de Pascal

L'objectif est de produire un fichier `pascal.html` qui génère le triangle de Pascal sous forme d'un tableau de dimension $(N + 1) \times (N + 1)$ où $0 \leq N \leq 7$ est un paramètre choisi par le visiteur. Pour tout k et n vérifiant $0 \leq k \leq n \leq N$, la cellule située sur la ligne de rang n et la colonne de rang k a pour valeur le coefficient binomial $C(n, k)$. Les Figures 38 et 39 illustrent le résultat attendu pour $N = 0$ et $N = 7$, respectivement. Par exemple, la cellule $(2, 1)$ située sur la 3^{ème} ligne et la 2^{nde} colonne a pour valeur $C(2, 1) = 2$ ainsi qu'illustré en figure 39. La valeur d'une cellule est obtenue en sommant celle du dessus (valeur 0 si elle est vide ou n'existe pas) et celle située à gauche de cette dernière (valeur 0 si elle est vide ou n'existe pas).

Le fichier `pascal.html` importe le fichier `pascal.js` et contient un champ de type `number` suivi d'un bloc `div`. Deux gestionnaire d'évènements sont implémentés à la fin du fichier `pascal.js`. Le premier se déclenche au chargement de la page, accède à la valeur N prédefinie du champ libellé `N`, et appelle avec cette valeur la fonction `creerTableau` qui se chargera de générer le tableau HTML correspondant au tableau $(N+1) \times (N+1)$ des coefficients $C(n, k)$. Le second se déclenche à chaque changement de valeur du champ `N`, supprime le tableau existant et rappelle `creerTableau` pour régénérer le tableau en fonction de la nouvelle valeur de `N`.

Implémentez la fonction `creerTableau` en trois étapes :

1. Génération d'un tableau HTML à N lignes et N cellules par ligne à insérer dans le bloc `div`.
2. Remplissage des cellules du tableau HTML située sous la diagonale en appliquant les règles de calcul suivantes :

- $C(n, 0) = 1$ pour $0 \leq n \leq N$.
- $C(n, n) = 1$ pour $0 \leq n \leq N$.
- $C(n, k) = C(n - 1, k - 1) + C(n - 1, k)$ pour $0 \leq k \leq n \leq N$.

3. Ajout de l'abréviation HTML de titre `C(n, k)` à la cellule (n, k) pour tout $0 \leq k \leq n \leq N$. Chaque abréviation apparaîtra au survol de sa cellule ainsi qu'illustré en Figure 39 pour la cellule $(2, 1)$ de valeur 2.

`N = 0`

1

`N = 7`

1							
1	1						
1	2	1					
1	3	3	1				
1	4	6	4	1			
1	5	10	10	5	1		
1	6	15	20	15	6	1	
1	7	21	35	35	21	7	1

FIGURE 38 – $C(n, k)$ pour $0 \leq k \leq n \leq 0$.

FIGURE 39 – $C(n, k)$ pour $0 \leq k \leq n \leq 7$.

Exercice 2. Gouvernements

Le fichier **gouvernements.html** affiche un tableau HTML. L'objectif est de formatter ce tableau en fonction d'évènements utilisateur. Vous compléterez à cet effet le fichier **gouvernements.js**.

1. Créez un tableau qui stocke chaque triplet d'entiers apparaissant sur une ligne du tableau HTML dans un tableau JS à 3 éléments. Ce tableau ne doit contenir ni les en-têtes, ni les noms de continents.

2. Créez un tableau qui stocke chaque ligne du tableau HTML, sauf la ligne d'en-têtes, sous la forme d'un objet ayant 4 propriétés dénommées `continent`, `republic`, `monarchy` et `other`.

3. Implémentez la fonction `effacerFormatte()` qui applique à toute cellule du tableau HTML contenant un entier la mise en forme suivante :

- son fond est blanc,
- le texte est en noir, et
- la graisse (weight) des chiffres est normale.

4. Implémentez la fonction `formaterMinMax(ncol)` qui prend en entrée un numéro de colonne du tableau HTML (`ncol`) et applique aux 2 cellules contenant les valeurs minimale et maximale de la colonne la mise en forme suivante :

- le texte est en gras,
- le texte est de couleur rouge pour la valeur minimale et vert clair pour la valeur maximale,
- le fond est de couleur rose pour la valeur minimale et vert pour la valeur maximale.

Appuyez-vous sur l'un ou l'autre des tableaux produits lors des questions précédentes pour l'implémentation. La Figure 40 illustre le résultat obtenu sur la colonne Other.

5. Remplacez le texte des en-têtes du tableau HTML `Republic`, `Monarchy` et `Other` par des boutons (éléments `button` ou `input` de type `button`). Chaque bouton affiche le même texte que l'en-tête qu'il remplace. Le gestionnaire d'événement pré-enregistré sur chaque bouton se chargera d'appeler les fonctions `effacerFormatte` et `indiquerMinMax` pour la colonne concernée à chaque clic visiteur.

Exercice 3. Horaires

Le fichier **horaires.html** importe **horaires.js**. Complétez ce dernier pour obtenir la page illustrée en Figure 42.

1. Ajoutez une ligne d'en-tête au tableau HTML dont les cellules seront une copie des items de la liste : le noeud texte du i-ème item sera copié dans celui de la i-ème cellule. Supprimez la liste une fois la copie effectuée.

2. Le fichier **horaires.js** définit deux tableaux dénommés `t_labels` et `c_labels`. Accédez aux valeurs du premier pour créer le contenu texte des 4 éléments HTML `label`. Accédez aux valeurs du second pour définir la couleur de fond CSS de ces 4 éléments. Par exemple, `A éviter` et `orange` seront respectivement le contenu texte et la couleur de fond du troisième label.

3. Modifiez les 4 boutons radio en affectant à leur attribut HTML `name` la même valeur `preference`. Affectez à l'attribut HTML `value` du i-ème bouton le i-ème élément du tableau `c_labels`. Par exemple, `red` sera la valeur du second bouton.

4. Ajoutez un bouton HTML de type `submit`, de nom `ok` et de valeur `OK` à la suite du bouton existant.

5. Ajoutez un élément HTML de type `form` et sans attributs contenant un clone complet de l'élément `fieldset`.

The screenshot shows a table of government types by continent and its representation in the JavaScript console.

Continent	Republic	Monarchy	Other
Africa	48	3	5
Antarctica	0	0	1
Asia	31	12	6
Europe	29	10	6
North America	13	10	14
Oceania	7	7	14
South America	12	0	2

Console output:

```

console.table()
gouvernements.js:16:13
    (index)      0          1          2
    0            48         3          5
    1              0         0          1
    2            31         12         6
    3            29         10         6
    4            13         10        14
    5              7         7        14
    6            12         0          2
  
```



```

console.table()
gouvernements.js:86:13
    (index)  continent   republic   monarchy   other
    0       Africa      48         3          5
    1     Antarctica   0          0          1
    2       Asia        31         12         6
    3       Europe      29         10         6
    4   North America  13         10        14
    5     Oceania      7          7        14
    6   South America  12         0          2
  
```

FIGURE 40 – Page web et console après clic sur l'entête de la colonne "Other"

Horaires

- 08:00
- 09:00
- 10:00
- 11:00
- 14:00
- 15:00
- 16:00
- 17:00
- 18:00
- 19:00

Palette

Remplir

FIGURE 41 – Page sans exécution JS

08:00	09:00	10:00	11:00	14:00	15:00	16:00	17:00	18:00	19:00

Palette

Neutre Interdit A éviter Préféré

Remplir OK

FIGURE 42 – Page avec exécution JS

Troisième partie

Période 10 : JS / PHP

5 TD JS / PHP

5.1 TD JS Evènements (©J.M. Richer, A. Rossi)

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#).

Exercice 1. Somme

Le fichier **somme.html** importe le fichier **somme.js**. Complétez ce dernier pour calculer automatiquement la somme des valeurs saisies dans les 2 premiers champs et l'afficher dans le troisième champ. Le résultat attendu est illustré en Figure 43.

Somme

+ =

FIGURE 43 – Calcul de somme

Exercice 2. Conversion de températures

Le fichier **temperature.html** importe le fichier **temperature.js**. Complétez ce dernier pour convertir en degrés Fahrenheit une température exprimée en degrés Celsius et inversement. On rappelle les formules de conversion :

$$\begin{aligned} C &= (F - 32) * 5/9 \\ F &= C * 9/5 + 32 \end{aligned}$$

Le résultat attendu est illustré en Figure 44.

Température

FIGURE 44 – Conversion de °C en °F

Exercice 3. Moyenne

Le fichier **moyenne.html** importe le fichier **moyenne.js**. Complétez ce dernier pour calculer la moyenne d'une série de notes décimales ou entières saisies dans le champ texte. On tolérera les espaces multiples entre nombres consécutifs. Utilisez les méthodes `trim`, `replace`, `split`, `forEach` et `parseFloat` pour traiter la chaîne saisie. Le résultat attendu est illustré en Figure 45.

Moyenne

FIGURE 45 – Calcul de moyenne

Exercice 4. PGCD

Le fichier **pgcd.html** importe le fichier **pgcd.js**. Complétez ce dernier pour calculer le plus grand commun diviseur de deux entiers en utilisant l'algorithme d'Euclide rappelé en Figure 46. Le résultat attendu est illustré en Figure 47.

```

fonction pgcd(a, b)
debut
    si (a <= 0) ou (b <= 0) alors retourne 0
    faire
        si (a < b) alors échanger a et b
        r = a modulo b
        a = b
        b = r
    tant que r != 0
    retourne a
fin
  
```

FIGURE 46 – Algorithme du calcul du PGCD

PGCD - algorithme d'Euclide

a = b =

FIGURE 47 – Calcul du PGCD de 2 entiers

Exercice 5. Calcul mental

Le fichier **calcul-mental.html** importe le fichier **calcul-mental.js**. Complétez ce dernier pour afficher et masquer à tour de rôle la réponse à la question pré-affichée à chaque clic sur le bouton. Le résultat attendu est illustré en Figures 48 et 49.

Calcul mental

Combien font six fois sept ?

Quarante-deux

Calcul mental

Combien font six fois sept ?

FIGURE 48 – Affichage de réponse à chaque clic “impair”

FIGURE 49 – Affichage de question à chaque clic “paire”

Exercice 6. Somme des premiers carrés

Le fichier **somme-carres.html** importe le fichier **somme-carres.js**. Complétez ce dernier pour afficher la liste des carrés des 10 premiers entiers naturels au clic sur le bouton. Le résultat attendu est illustré en Figures 50 et 51.

afficher les carrés
$$\begin{aligned}1^2 &= 1 \\2^2 &= 5 \\3^2 &= 14 \\4^2 &= 30 \\5^2 &= 55 \\6^2 &= 91 \\7^2 &= 140 \\8^2 &= 204 \\9^2 &= 285 \\10^2 &= 385\end{aligned}$$

FIGURE 50 – Affichage initial de la page

FIGURE 51 – Affichage après clic sur le bouton

6 TP JS / PHP

6.1 TP JS 8 - Evènements et formulaires

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Masquage d'éléments

Le fichier **toggle.html** importe **toggle.js** et contient un élément `aside` et un bouton. La page doit, à chaque clic sur le bouton, masquer l'élément `aside` s'il est affiché, ou l'afficher s'il est masqué. Complétez **toggle.js** pour obtenir le comportement souhaité tel qu'illustré en Figures 52 et 53.

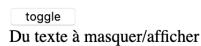


FIGURE 52 – Affichage aux clics “impairs”



FIGURE 53 – Masquage aux clics “pairs”

Exercice 2. Liste éditable

Le fichier **liste-editable.html** importe **liste-editable.js**. Complétez ce dernier de sorte qu'en cliquant sur un item de la liste déroulante, une invite s'affiche et permette de modifier le texte de l'item. Le comportement attendu est illustré en Figures 54, 55 et 56.

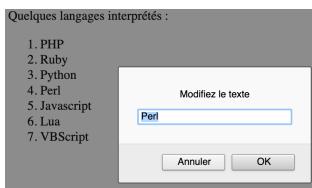


FIGURE 54 – Liste éditable (1)

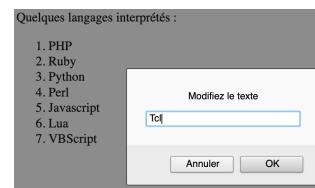


FIGURE 55 – Liste éditable (2)

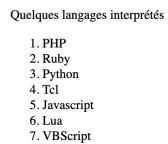


FIGURE 56 – Liste éditable (3)

Exercice 3. Cochage de cases

Le fichier **cocher.html** importe le fichier **cocher.js** et contient :

- une liste ordonnée (`ol`) dont chaque item (`li`) contient une case à cocher ;
- deux boutons pour cocher et décocher toutes les cases d'un coup.

Complétez **cocher.js** pour obtenir le comportement attendu tel qu'illustré en Figures 57 et 58.

Exercice 4. Vérification d'adresse email

Le fichier **verifier-email.html** importe le fichier **verifier-email.js** et contient deux champs texte donnant la possibilité au visiteur de saisir son adresse e-mail dans le premier, et de la re-saisir dans le second pour confirmation. Complétez **verifier-email.js** pour comparer les deux adresses de sorte que :

- la vérification se déclenche quand le visiteur vient d'entrer une lettre dans le second champ (événement `onkeyup`) ;

1. <input checked="" type="checkbox"/> A
2. <input checked="" type="checkbox"/> B
3. <input checked="" type="checkbox"/> C

1. <input type="checkbox"/> A
2. <input type="checkbox"/> B
3. <input type="checkbox"/> C

FIGURE 57 – Cochage des cases au clic sur Cocher tout

FIGURE 58 – Décochage des cases au clic sur Décocher tout

- la vérification se déclenche lors de la perte du focus sur le second champ (évènement `onblur`).

Exploitez les deux règles CSS pour mettre les deux champs en vert s'ils contiennent la même adresse, ou en rouge sinon. Le comportement attendu est illustré en Figures 59 et 60.

E-Mail :

Confirmez :

FIGURE 59 – Adresse e-mail validée

E-Mail :

Confirmez :

FIGURE 60 – Adresse e-mail invalidée

Exercice 5. Nombre mystère

Le fichier **nombre-mystere.html** importe **nombre-mystere.js**. La page doit permettre au visiteur de deviner un nombre entier choisi aléatoirement dans l'intervalle [1,30]. A chaque clic sur le bouton, on informe le visiteur en cas de succès (nombre deviné) ou on le guide en cas d'échec. Le nombre de tentatives est limitée à 5 et on affiche le nombre à deviner le cas échéant. Complétez **nombre-mystere.js** pour obtenir le comportement attendu tel qu'illustré en Utilisez les méthodes `Math.floor` et `Math.random` pour la génération aléatoire des nombres entiers. Le comportement attendu est illustré en Figures 61, 62, 63 et 64.

20

FIGURE 61 – Première tentative

20

C'est moins que 20

FIGURE 62 – Affichage (échec) après validation

10

C'est plus que 10

13

Bravo! Le nombre mystère est 13. Nombre de tentative(s): 5

FIGURE 63 – Affichage (échec) après une seconde tentative

FIGURE 64 – Affichage de la réponse et du nombre de tentatives en cas de succès

6.2 TP JS 9 - Evènements et formulaires

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Liste dynamique

Le fichier `liste-dynamique.html` importe `liste-dynamique.js`. La page doit permettre de créer une liste item par item ou bien de la supprimer intégralement. A chaque clic sur le bouton d'ajout, on affiche une invite pour que le visiteur saisisse le texte du prochain item puis on actualise la liste affichée en conséquence. A chaque clic sur le bouton de suppression, on supprime la liste intégralement. Complétez `liste-dynamique.js` pour obtenir le comportement attendu tel qu'illustré en Figures 65, 66 et 67.

FIGURE 65 – Etat initial ou résultant de la suppression de la liste

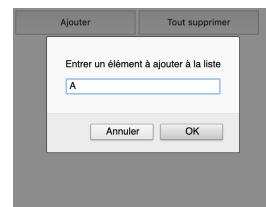


FIGURE 66 – Saisie d'un second élément de contenu A

FIGURE 67 – Résultat de l'ajout

Exercice 2. Génération de questions-réponses

Le fichier `questions-reponses.html` importe `questions-reponses.js` pour générer des questions et leurs réponses. Au premier clic sur le bouton, on affiche une question du type Combien font x fois y ? où x et y sont des entiers tirés aléatoirement dans l'intervalle [1,10]. Au clic suivant, on affiche la réponse puis on revient à l'état de départ pour pouvoir enchaîner sur d'autres questions. Complétez `questions-reponses.js` pour obtenir le comportement attendu tel qu'illustré en Figures 68, 69 et 70.

Génération de questions et de réponses

[Créer une question](#)

FIGURE 68 – Etat initial

Génération de questions et de réponses

[Montrer la réponse](#)

Combien font 6 fois 8 ?

FIGURE 69 – Affichage d'une question à chaque clic “impair”

Génération de questions et de réponses

[Créer une question](#)

Combien font 6 fois 8 ? 48

FIGURE 70 – Affichage de la réponse à chaque clic “pair”

Exercice 3. Calcul d'intégrales

Le fichier `integrale.html` importe `integrale.js` pour calculer l'intégrale de fonctions par approximation en utilisant les trois méthodes suivantes :

- Méthode des rectangles à droite.
- Méthode des trapèzes.
- Méthode de Simpson.

Le formulaire de la page doit permettre de sélectionner par menu déroulant la fonction $f(x)$, l'intervalle de calcul $[x_1, x_2]$ et la granularité utilisée en nombre d'intervalles (voir Figure 71). Complétez `integrale.js` pour obtenir le comportement attendu en répondant aux questions qui suivent.

1. Implémentez l'écouteur du menu déroulant afin de mettre à jour la fonction à intégrer selon l'option choisie par le visiteur.
2. Implémentez la fonction `segmentation(x1, x2, n)` qui renvoie le tableau des $n + 1$ valeurs segmentant l'intervalle $[x_1, x_2]$ en n intervalles contigus et de mêmes tailles.
3. Implémentez la fonction `integrale_rectangle_droite(f, x1, x2, n)` qui renvoie l'approximation de l'intégrale de la fonction f sur $[x_1, x_2]$ par la méthode des rectangles à droite basée sur une décomposition en n intervalles. Cette méthode diffère simplement de la méthode des rectangles à gauche (déjà implémentée) en sommant sur les n premières valeurs de la segmentation de $[x_1, x_2]$ plutôt que les n dernières. Utilisez de préférence la méthode `Array.prototype.reduce` pour calculer la somme r .
4. Associez un écouteur aux clics sur le bouton “calculer” de sorte à remplir les 5 champs (colorés en rouge) avec les valeurs résultant des calculs.
5. Associez un écouteur aux clics sur le bouton “effacer” de sorte à effacer le contenu des 5 champs de calcul.

Calcul d'intégrales

Calcul de l'intégrale $f(x) = \boxed{x * x}$

x_1 $y_1=f(x_1)$

x_2 $y_2=f(x_2)$

intervalles

Résultat méthode des rectangles à droite

Résultat méthode des trapèzes

Résultat méthode de Simpson

FIGURE 71 –

6.3 TP JS 10 - Evènements et formulaires

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Génération de matrices

La page produite par le fichier **matrices.html** doit permettre de générer et manipuler des matrices binaires (0/1) ou entières. **matrices.html** importe deux fichiers JS :

- **utils.js** qui contient différentes fonctions utilitaires,
- **matrices.js** qui contient les gestionnaires d'évènements de la page faisant appel à ces fonctions.

Les attendus sont les suivants. Au chargement de la page, une matrice aléatoire s'affiche : ses dimensions et son type découlent des valeurs par défaut des champs numériques et du bouton précoché (cad. une matrice binaire 4×4). Une copie de cette matrice est aussi affichée à sa droite (voir Figure 72). On peut en changer le type en cochant l'un des boutons radios (Figure 73). On peut aussi permuter deux de ses lignes tirées aléatoirement (Figure 74) ou la transposer (Figure 75) en pressant les boutons correspondants. On peut enfin réduire ses dimensions (Figure 76) ou les augmenter (Figure 77) en modifiant les champs numériques. Ces opérations peuvent être effectuées à tout moment et, hormis les opérations de transposition et permutation qui n'affectent que la matrice de droite, elles conduisent à une régénération complète de la matrice et de sa copie.

Consultez le fichier **utils.js** puis complétez les fonctions du fichier **matrices.js** en vous conformant aux commentaires pour fournir les fonctionnalités attendues.

Exercice 2. Date en post-it

Le fichier **postit.html** importe **postit.js** et permet d'afficher une horloge sous forme de postit. Le postit s'affiche au chargement et la date (jour, heure, minute et seconde) est réactualisée toutes les demi-secondes à l'aide de la méthode globale `setTimeout` : l'instruction `setTimeout (f, d)` ; appelle la fonction `f` après `d` millisecondes. Complétez **postit.js** pour obtenir le comportement attendu tel qu'illustré en Figure 78.

Matrices 0/1 oui non

Lignes

Colonnes

	1	2	3	4
1	1	1	0	1
2	1	1	1	0
3	0	1	0	0
4	1	1	0	1

	1	2	3	4
1	1	1	0	1
2	1	1	1	0
3	0	1	0	0
4	1	1	0	1

Matrices 0/1 oui non

Lignes

Colonnes

	1	2	3	4
1	0	14	3	9
2	13	0	10	10
3	13	13	8	13
4	12	8	8	2

	1	2	3	4
1	0	14	3	9
2	13	0	10	10
3	13	13	8	13
4	12	8	8	2

FIGURE 72 – Génération aléatoire de matrices binaires 4x4 au chargement ou régénération (bouton oui)

FIGURE 73 – Génération aléatoire de matrices entières (bouton non)

Matrices 0/1 oui non

Lignes

Colonnes

	1	2	3	4
1	0	14	3	9
2	13	0	10	10
3	13	13	8	13
4	12	8	8	2

	1	2	3	4
3	13	13	8	13
2	13	0	10	10
1	0	14	3	9
4	12	8	8	2

FIGURE 74 – Permutation de 2 lignes tirées aléatoirement (bouton Permuter)

Matrices 0/1 oui non

Lignes

Colonnes

	1	2	3	4
1	0	14	3	9
2	13	0	10	10
3	13	13	8	13
4	12	8	8	2

	3	2	1	4
1	13	13	0	12
2	13	0	14	8
3	8	10	3	8
4	13	10	9	2

FIGURE 75 – Transposition (bouton Transposer)

Matrices 0/1 oui non

Lignes

Colonnes

	1	2	3	4
1	1	0	14	20
2	13	24	11	2
3	13	21	0	3
4	14	11	13	3
5	13	16	11	15
6	19	23	11	6

	1	2	3	4
1	1	0	14	20
2	13	24	11	2
3	13	21	0	3
4	14	11	13	3
5	13	16	11	15
6	19	23	11	6

Matrices 0/1 oui non

Lignes

Colonnes

	1	2
1	3	3
2	3	4
3	9	2
4	11	12
5	4	8
6	9	12

	1	2
1	3	3
2	3	4
3	9	2
4	11	12
5	4	8
6	9	12

FIGURE 76 – Extension du nombre de lignes (champ Lignes)

FIGURE 77 – Réduction du nombre de colonnes (champ Colonnes)

Affichage du jour et de l'heure dans un post-it

Vendredi
16:22:23

Un paragraphe dans une page web...

FIGURE 78 – Horloge cadencée à 0,5s

6.4 TP JS 11 - Validation JS de formulaires et bases de données

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Validation JS de formulaire

Le fichier **formulaire.html** importe les fichiers **formulaire.css** et **formulaire.js** et seul ce dernier est à compléter. L'objectif est de valider la saisie d'un formulaire en JavaScript avant envoi. Le comportement attendu est illustré par les figures suivantes :

- Figure 79 correspond au chargement de la page ou à sa réinitialisation.
- Figure 80 décrit l'actualisation de la page en cas de saisie incorrecte pour tous les champs.
- Figure 81 décrit l'actualisation de la page en cas de saisie incorrecte sur certains champs.
- Figure 82 décrit l'actualisation de la page en cas de saisie correcte.

Genre : Homme Femme

Nom :

Prénom :

Âge :

Pseudo :

Mot de passe :

Mot de passe (confirmation) :

Pays : Sélectionnez votre pays de résidence

Je désire recevoir la newsletter chaque mois.

[M'inscrire](#) [Réinitialiser le formulaire](#)

Genre : Homme Femme Vous devez sélectionnez votre genre

Nom : Un nom ne peut pas faire moins de 2 caractères (uniquement des lettres)

Prénom : Un prénom ne peut pas faire moins de 2 caractères (uniquement des lettres)

Âge : L'âge doit être compris entre 5 et 140

Pseudo : Le pseudo ne peut pas faire moins de 4 caractères

Mot de passe : Le mot de passe ne doit pas faire moins de 6 caractères

Mot de passe (confirmation) : Le mot de passe de confirmation doit être identique à celui d'origine

Pays : Sélectionnez votre pays de résidence Vous devez sélectionner votre pays de résidence

Je désire recevoir la newsletter chaque mois.

[M'inscrire](#) [Réinitialiser le formulaire](#)

FIGURE 79 – Vue par défaut ou réinitialisée

FIGURE 80 – Saisie intégralement incorrecte

Genre : Homme Femme

Nom : Alpha

Prénom : T Un prénom ne peut pas faire moins de 2 caractères

Âge : 13

Pseudo : charlie

Mot de passe :

Mot de passe (confirmation) : Le mot de passe de confirmation doit être identique à celui d'origine

Pays : Angleterre

Je désire recevoir la newsletter chaque mois.

[M'inscrire](#) [Réinitialiser le formulaire](#)

Genre : Homme Femme

Nom : Alpha

Prénom : Tango

Âge : 13

Pseudo : charlie

Mot de passe :

Mot de passe (confirmation) :

Pays : Angleterre

Je désire recevoir la newsletter chaque mois.

Le formulaire est bien rempli. OK

[M'inscrire](#) [Réinitialiser le formulaire](#)

FIGURE 81 – Saisie partiellement correcte

FIGURE 82 – Saisie correcte

Toute saisie incorrecte pour un champ affiche une aide (tooltip) sur les contraintes à respecter et encadre le champ en rouge. La Figure 80 explicite ces contraintes pour chacun des champs. Toute saisie correcte fait disparaître l'aide et encadre le champ en vert.

Complétez le fichier **formulaire.js** pour obtenir le comportement attendu :

- L'encadrement en vert (resp. rouge) d'un champ s'obtient en lui affectant la classe CSS **correct** (resp. **incorrect**). Les règles CSS pour ces classes sont prédefinies dans le fichier **formulaire.css**.

- Les aides sont des éléments `span` de classe `tooltip` qui suivent immédiatement le champ associé (ou dernier champ associé dans le cas des 2 boutons radio). Leur affichage/masquage se fait en manipulant la propriété CSS `display`.
- Un écouteur d'évènements est à associer à chaque champ, y compris les boutons de soumission et de réinitialisation. Il doit vérifier la saisie et actualiser la page en conséquence en (re)classant les éléments concernés. Les évènements déclencheurs sont les suivants :
 - `onkeyup` sur un champ texte ou mot de passe,
 - `onclick` sur un bouton radio,
 - `onmouseup` sur le menu déroulant,
 - `onsubmit` et `onreset` sur le formulaire.

Exercice 2. Insertion en base de données

Toute soumission du formulaire après validation invoque le fichier `inserer-personne.php`. Complétez ce dernier de sorte à insérer les données saisies dans une base de données dénommée `12_personnes` puis afficher le tableau des inscrits comme illustré en Figures 83 et 84.

GENRE	NOM	PRENOM	AGE	LOGIN	MOTDEPASSE	PAYS	NEWSLETTER
Homme	Morricone	Ennio	89	morri	mndp123	ETATS-UNIS	1
Homme	Coppola	Francis Ford	79	ffcop	motdepasse	ANGLETERRE	0

FIGURE 84 – Résultat de la soumission

FIGURE 83 – Soumission correcte

Créez tout d'abord la base sous PhpMyAdmin avec l'interclassement `utf8mb4_bin`. Importez-y le fichier `12_personnes.sql` qui créera la table `Personne`. Le fichier `connexpdo.inc.php` fournit une fonction de connexion à la base via l'interface PDO. Il est à adapter avec votre login et mot de passe MySQL et peut être exploité par les fichiers `inserer-personne.php` et `afficher-tableau.php`.

Veillez aux points suivants :

- Du fait de la syntaxe utilisée pour nommer les champs du formulaire (p. ex. `person[age]`, `person[pwd1]`), PHP construira automatiquement un tableau associant à chaque “clé” de champ (p. ex. `age`) la valeur saisie par le visiteur, tableau qui sera stocké dans la variable `$_POST['person']`.
- Le champ correspondant à la confirmation du mot de passe (`pwd2`) est redondant.
- La valeur de la case à cocher (`news`) n'est pas communiquée si la case n'a pas été cochée : le choix doit tout de même être consigné dans la base.
- Toutes les valeurs du tableau `$_POST['person']` sont des chaînes de caractères qu'il faut transtyper dans le cas des champs `age` et `newsletter`.

6.5 TP PHP 8 - Base de données (sportifs)

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le [manuel PHP](#).

Exercice 1. Création de base de données avec PhpMyAdmin

On considère une base de données comportant trois tables schématisées ci-dessous où les clés primaires apparaissent en gras et en italique.

TABLE 1: Structure de la table personne

Colonne	Type	Null	Valeur par défaut
<i>id_personne</i>	int(11)	Non	
nom	varchar(20)	Non	
prenom	varchar(20)	Oui	NULL
depart	int(2)	Non	
mail	varchar(50)	Non	

TABLE 2: Structure de la table pratique

Colonne	Type	Null	Valeur par défaut
<i>id_personne</i>	int(11)	Non	
<i>id_sport</i>	int(11)	Non	
niveau	tinyint(4)	Oui	NULL

TABLE 3: Structure de la table sport

Colonne	Type	Null	Valeur par défaut
<i>id_sport</i>	int(11)	Non	
design	varchar(30)	Non	

Créez cette base avec phpMyAdmin en procédant comme suit :

- (1) Créez la base de nom `12_sportifs` en choisissant l'interclassement `utf8mb4_bin`.
- (2) Sélectionnez la base et importez le script `12_sportifs.sql` (via l'onglet `Importer`) afin de créer tables et enregistrements.
- (3) Ajoutez un enregistrement dans chacune des tables.

Exercice 2. Fonction de connexion à base de données via l'API PDO

Créez un fichier de connexion `connexpdo.inc.php` à SGBD en utilisant l'API PDO. Ce fichier doit contenir une fonction qui prend le nom d'une base de données en entrée (`string`), s'y connecte avec le bon DSN, et renvoie l'objet PDO créé pour la connexion à cette base. Cette fonction sera appelée dans tous les fichiers PHP exécutant des requêtes SQL.

Exercice 3. Rapport d'erreurs du SGBD en PHP

Afin que PDO relaie sous forme d'exceptions PHP les erreurs SQL (connexion, syntaxe des requêtes, etc.) , ajoutez l'instruction

```
$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

dans votre fonction de connexion sitôt l'objet PDO créé. Le fichier **test-connexion.php** provoque volontairement une erreur de connexion. La Figure 85 illustre un traitement possible d'erreur SQL par affichage dans la console et boîte d'alerte. Exécutez ce script et utilisez autant que faire se peut les différentes fonctions d'affichage fournies dans le fichier **js.php**.



FIGURE 85 – Gestion PHP de l'échec d'une tentative de connexion à la base de données

Exercice 4. Extraction d'enregistrements et affichage sous forme de tableau trié

Créez un fichier **afficher-personnes.php** permettant d'afficher le contenu de la table **personne** dans un tableau HTML. Les personnes doivent être affichées dans l'ordre alphabétique de leurs noms. Le tri est à effectuer soit en PHP sur les enregistrements extraits, soit par la requête SQL programmée (auquel cas le résultat dépendra du jeu choisi pour l'interclassement). La Figure 86 illustre la page à produire.

id_personne	nom	prenom	depart	mail
5	Anywhere	Out	99	of@the.world
3	Azerty		13	azerty@no.clue
1	Bart	Jean	59	jean.bart@en.guerre
4	Do	You	84	know@what
7	Jacques		49	jacques@a.dit
8	Mel	E	49	m.a@i.l
2	Surcouf	Robert	66	surcouf@catch.me

FIGURE 86 – Tableau trié des sportifs

Exercice 5. Sélection et affichage d'enregistrement via lien hypertexte

Modifiez le fichier **afficher-personnes.php** en insérant un lien hypertexte sur chaque nom tel qu'ilustré en Figure 87. Chaque lien cliqué doit afficher le tableau des sports pratiqués par la personne tel qu'ilustré en Figure ???. Développez un fichier cible **afficher-personne.php** pour les liens qui combineront jointure et sélection pour extraire et afficher les données requises.

id_personne	nom	prenom	depart	mail
5	Anywhere	Out	99	of@the.world
3	Azerty		13	azerty@no.clue
1	Bart	Jean	59	jean.bart@en.guerre
4	Do	You	84	know@what
7	Jacques		49	jacques@a.dit
9	Loeb	Sébastien	67	loeb@mille.lacs
8	Mel	E	49	m.a@i.l
2	Surcouf	Robert	66	surcouf@catch.me

Nom	Prénom	Sport
Surcouf	Robert	Tennis
Surcouf	Robert	Tennis de table

FIGURE 88 – Sports pratiqués par un corsaire

FIGURE 87 – Tableau avec hyperliens

Exercice 6. Insertion d'enregistrement par formulaire

Créez un fichier **inserer-personnes.php** permettant d'insérer une nouvelle personne dans la base tel qu'ilustré en Figure 89. En cas de succès, affichez l'identifiant attribué à cette personne dans une boîte d'alerte tel qu'ilustré en Figure 90.

Vos coordonnées

Nom :	<input type="text" value="Loeb"/>
Prénom :	<input type="text" value="Sébastien"/>
Département:	<input type="text" value="67"/> <input type="button" value=""/>
Mail :	<input type="text" value="loeb@mille.lacs"/>
<input type="button" value="Envoyer"/>	

Formulaire à compléter!



FIGURE 89 – Inscription d'un sportif

FIGURE 90 – Succès de l'inscription

6.6 TP PHP 9 - Base de données (crédit bancaire)

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le [manuel PHP](#).

On propose un mini-site permettant de choisir son crédit bancaire pour financer achat de véhicule ou travaux d'habitat. Le choix se fait en trois étapes successives :

- (1) Le visiteur indique sur la page d'accueil **acceuil.php** la nature de ses dépenses - Auto ou Habitat - ainsi que le montant souhaité. La Figure 91 illustre une demande de 10k€ pour des travaux. La Figure 92 illustre une demande de 5k€ pour l'achat d'un véhicule.
- (2) La page suivante **credits.php** offre au visiteur deux alternatives pour finaliser son crédit :
 1. Soit il fixe lui-même le montant de ses mensualités sur la base d'un taux affiché (qui servira à calculer la durée du crédit en page suivante). La Figure 93 illustre le choix de mensualités à 2000€ pour un taux proposé de 2,9%.
 2. Soit il choisit la durée de son crédit parmi plusieurs possibles, chacune étant associée à un taux spécifique (à partir duquel on calcule les mensualités qui sont affichées). La Figure 94 illustre le choix d'une durée de 60 mois à laquelle s'applique un taux mensuel de 3,65% et qui aboutit à des mensualités de 206,53€.
- (3) La dernière page **formule.php** récapitule la demande du visiteur tel qu'illustré par les Figures 95 et 96 pour les 2 scénarios présentés.

L'objectif des exercices est de compléter les scripts **credits.php** et **formule.php** pour obtenir le résultat souhaité.

FIGURE 91 – Travaux d'habitat (10k€)

FIGURE 92 – Achat de véhicule (5k€)

Durée	12 mois	24 mois	36 mois	48 mois	60 mois
Mensualité (€)	998.63	584.13	447.84	384.89	345.99
Taux mensuel (%)	2.90	2.90	2.85	2.85	2.80

FIGURE 93 – Choix d'une mensualité (2000€)

Exercice 1. Création de la base de données et connexion

Les taux mensuels proposés par la banque dépendent de la durée du crédit (en mois) et du type de financement (automobile ou habitat). Ces données sont stockées dans une base à laquelle devra accéder le script **credits.php**. Créez cette base sous phpMyAdmin en la nommant **12mi_tp_php_credit** et en choisissant l'interclassement **utf8mb4_bin**. Importez-y le script **12mi_tp_php_credit.sql**. L'import crée la table **CREDIT** qui stocke les offres de crédits. CREDIT est dotée des colonnes suivantes :

- ID : l'identifiant du crédit (`int, AUTO_INCREMENT, PRIMARY_KEY`)

Vous souhaitez financer un véhicule pour un montant de 5000€

Pour le remboursement, choisissez :

soit une mensualité de EUR à 3.75%

soit une durée de crédit proposée dans ce tableau

Durée	12 mois	24 mois	36 mois	48 mois	60 mois
Mensualité (€)	525.06	319.59	253.55	224.2	206.53
Taux mensuel (%)	3.75	3.75	3.70	3.70	3.65

[Suivant](#) [Effacer](#)

FIGURE 94 – Choix d'une durée (60 mois)

Les caractéristiques de votre crédit pour financer des travaux :

Emprunt : 10000€
 Durée : 6 mois
 Taux mensuel : 2.9%
 Mensualité : 2000€
 Coût du crédit : 2000€

[Retour à l'accueil](#)

FIGURE 95 – Récapitulatif - scénario 1

Les caractéristiques de votre crédit pour financer un véhicule :

Emprunt : 5000€
 Durée : 60 mois
 Taux mensuel : 3.65%
 Mensualité : 206.53€
 Coût du crédit : 7391.8€

[Retour à l'accueil](#)

FIGURE 96 – Récapitulatif - scénario 2

- PRET : le type de prêt (type enum à deux valeurs "auto" et "habitat")
- DUREE : la durée du crédit en mois (int)
- TAUX : le taux mensuel du crédit donné en pourcentage, par ex. 3,75 (float).

Adaptez le script **connexpdo.inc.php** qui sera utilisé par **credits.php** pour vous y connecter.

Exercice 2. Affichage des taux et calcul des mensualités

Complétez le fichier **credits.php** pour produire l'affichage attendu tel qu'illustré en Figure 93 pour un prêt “habitat” et en Figure 94 pour un prêt “auto”. Les taux du tableau devront être récupérés dans la base de données selon le type de prêt demandé par le visiteur (Auto ou Habitat). Les mensualités affichées dans le tableau sont calculées en fonction du taux proposé et du montant demandé avec la formule suivante :

$$\text{mensualite} = (\text{montant} \times \text{taux} \times (1 + \text{taux})^{\text{duree}}) / ((1 + \text{taux})^{\text{duree}} - 1) \quad (1)$$

où *mensualite*, *duree* et *taux* dénotent respectivement la mensualité en euros, la durée du crédit en nombre de mois, et le taux mensuel (réel dans l'intervalle [0,1] et non pas un pourcentage). Le taux qui est proposé si le visiteur fixe lui-même ses mensualités (par ex. 2,9% en Figure 93) est le maximum des taux proposés par la banque pour ce type de prêt (cad. le maximum des taux affichés dans le tableau). On affichera les taux et les mensualités avec une précision de 2 chiffres après la virgule.⁸

8. Penser à utiliser les fonctions `pow`, `max` et `number_format` dans cet exercice.

Exercice 3.

Le script **formule.php** doit produire un récapitulatif de la demande tel qu'illustré en Figures 95 et 96. Il doit notamment afficher le coût du crédit qui vaut $duree * mensualite - montant$. Deux cas se présentent : soit le visiteur a fixé sa mensualité, soit il a choisi sa durée de crédit. Dans le premier cas, **formule.php** doit déduire la durée du crédit à partir de la mensualité, du taux et du montant en “inversant” la formule précédente :

$$duree = \log(mensualite / (mensualite - (montant \times taux))) / \log(1 + taux) \quad (2)$$

Le nombre de mois obtenu sera arrondi à l'entier supérieur avant affichage.⁹ Dans le second cas, **formule.php** doit recalculer la mensualité avec la formule précédente à partir de la durée, du taux et du montant.

Dans les deux cas, le calcul effectué par **formule.php** se base sur le montant du prêt que le visiteur a choisi sur la page d'accueil et qui a été communiqué par formulaire à **credits.php**. De même, **formule.php** se base sur le type de prêt choisi afin de personnaliser l'en-tête du récapitulatif (... pour financer ...). Utilisez le mécanisme des sessions PHP afin que **credits.php** puisse “partager” ces deux données avec **formule.php**. Vous pouvez aussi utiliser ce mécanisme pour échanger les autres données (les durées, leurs taux, ...) qui ont été extraites de la base de données plutôt que de les réextraire de la base.

9. Utiliser les fonction PHP `log` et `ceil` pour ce calcul.

6.7 TP PHP 10 - Révisions PHP/JS (catalogue musical)

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le [manuel PHP](#) et pour visualiser ce qui est attendu, ce [démonstrateur](#).

On se propose de développer un mini-site web permettant de consulter et d'enrichir un catalogue musical. Le catalogue répertorie des albums et les titres de leurs chansons. Il est stocké dans une base de données dénommée `12_music` qui contient 2 tables dont la structure est illustrée en Figure 97. Importez le fichier `catalogue.sql` sous PhpMyAdmin pour créer les tables de cette base et leurs enregistrements.

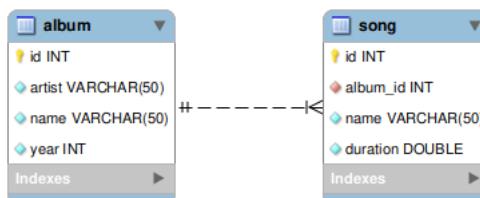


FIGURE 97 – Modèle de la base de données `12_music`

Exercice 1. PHP : Bases de données

Cet exercice porte sur l'accès en PHP à la base `12_music` pour y insérer et en extraire albums et titres. Le fichier `connexion.php` fournit une fonction de connexion à la base que vous devez adapter à votre environnement.

1. Complétez le fichier `catalogue.php` afin d'extraire de la base de données tous les albums et tous les titres que vous stockerez dans 2 tableaux associatifs identiques à ceux définis dans le fichier `./utils/data.php`. Ne modifiez pas la partie HTML du fichier : aucun affichage n'est requis dans cet exercice.
2. Le fichier `ajout.php` affiche au chargement un formulaire d'ajout d'album. Complétez le fichier afin d'insérer les données saisies par l'utilisateur dans la base de données (cf. Figure 98). Si l'insertion échoue, affichez un paragraphe HTML en bas du formulaire tel qu'illustré en Figure 99 : la classe (HTML) du paragraphe doit être égale à `error` pour obtenir la coloration rouge du message d'erreur.

Catalogue Musical : Gestionnaire

Ajouter un album

Artiste	P.J. Harvey
Album	Dry
Année	1992

Catalogue Musical : Gestionnaire

Ajouter un album

Artiste	<input type="text"/>
Album	<input type="text"/>
Année	<input type="text"/>

L'album n'a pas été ajouté.

FIGURE 98 – Formulaire d'ajout d'album

FIGURE 99 – Echec en cas d'ajout d'album

Exercice 2. PHP : Génération HTML

Le chargement du fichier **catalogue.php** affiche le formulaire illustré en Figure 100. Chaque clic sur le bouton “Voir le catalogue” recharge le fichier et doit afficher les albums sous forme de tableau et les titres de chansons sous forme de liste tel qu’illustré en Figure 101. Complétez le fichier **catalogue.php** pour obtenir ce comportement. Si vous n’avez pas traité la question 1 de l’exercice précédent, importez dans **catalogue.php** le fichier **/utils/data.php** qui définit les deux tableaux PHP contenant toutes ces données.

1. Générez et placez le tableau des albums dans la section `<div id="albums">` du code HTML produit. Le tableau comporte 4 colonnes : artiste, album, année et actions (voir Figure 101). Pour chaque album, les données des 3 premières colonnes proviennent de la base (ou du tableau défini dans **data.php**). La quatrième colonne contient un bouton d’attribut HTML `onclick=afficherAlbumDetails(id, name)`¹⁰ où `id` et `name` dénotent l’identifiant et le nom de l’album stockés dans la base (ou dans le tableau défini dans **data.php**).
2. Générez sous forme de liste ordonnée tous les titres de chansons, liste qui sera placée dans la section `<div id="songs">` du code HTML produit (voir Figure 101). Attribuez à chaque élément de la liste la classe HTML dont la valeur est l’identifiant de l’album contenant la chanson.

artiste	album	année	actions
Metallica	...And Justice For All	1988	Détails
Metallica	Black Album	1991	Détails

FIGURE 100 – Formulaire

artiste	album	année	actions
Metallica	...And Justice For All	1988	Détails
Metallica	Black Album	1991	Détails

- 1. One(7.25)
- 2. Blackened(6.42)
- 3. Enter Sandman(5.3)
- 4. Sad But True(5.29)
- 5. Master of Puppets(8.35)
- 6. Battery(5.13)
- 7. Dialectic Chaos(2.26)
- 8. Endgame(5.57)
- 9. Peace Sells(4.09)

FIGURE 101 – Affichage du catalogue

Exercice 3. JS : Manipulation du DOM

Le fichier **exo3.html** importe **catalogue.js** que vous complétez pour répondre aux questions qui suivent.

1. Au chargement du fichier HTML, cachez tous les items de la liste ordonnée contenant les titres des chansons (ne pas les supprimer).
2. Un clic sur le bouton “Détails” associé à un album affiche à droite du tableau le nom de l’album et sa liste de chansons (voir Figure 102). La fonction déclenchée par le clic se dénomme `afficherAlbumDetails` et appelle successivement `afficherTitre` et `afficherPlaylist`. Complétez la fonction `afficherTitre(albumName)` pour qu’elle remplisse le texte de l’élément de balise `<h4>` de la section `songs` avec le nom de l’album choisi.
3. Complétez la fonction `afficherPlaylist(idAlbum)` pour qu’elle affiche les items de la liste associés aux titres de l’album choisi et qu’elle masque les autres.

"Black Album"

- 1. Enter Sandman(5.3)
- 2. Sad But True(5.29)

FIGURE 102 – Nom et titres d’album

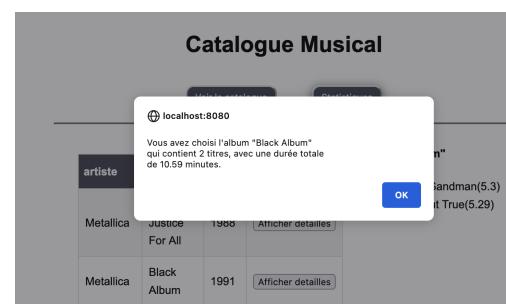


FIGURE 103 – Statistiques d’un album

10. `onclick` met en place un écouteur sur l’élément HTML de manière identique à la méthode JS `addEventListener`.

Exercice 4. JS : Manipulation de tableaux

Le fichier `exo4.html` encode les albums sous forme de tableau HTML et les titres de chansons sous forme de liste ordonnée. Au chargement, seuls le nom et les titres du premier album sont affichés, les autres titres étant masqués (voir code source). Le fichier contient des instructions JS obfusquées permettant de réactualiser nom et titres de chansons si l'on clique sur le bouton “Détails” d'un autre album (tel que demandé dans l'exercice précédent). `exo4.html` importe par ailleurs le fichier `stats.js` qui doit permettre d'afficher diverses statistiques sur l'album choisi lorsque le visiteur clique sur le bouton “Statistiques” (voir) Figure 103. Complétez `stats.js` pour obtenir le comportement attendu en répondant aux questions qui suivent.

1. Associez un écouteur au bouton “Statistiques” qui appellera la fonction `lancerStats` lorsqu'il est cliqué.
Implémentez pas à pas la fonction `lancerStats` :
2. Stockez dans une variable dénommée `titres` de type `Array` tous les éléments `` de la liste `` figurant dans la section `songs`.
3. Filtrez le tableau précédent pour ne conserver que les items de la liste qui sont affichés (items dont propriété CSS `display` vaut `list-item`). Vous stockerez le résultat dans une variable `albumTitres` de type `Array`.
4. Modifiez le tableau `albumTitres` pour n'y stocker que le texte associé à chaque chanson affichée.
5. Extrayez du tableau `albumTitres` la durée de chaque titre (nombre apparaissant entre parenthèses après le titre d'une chanson) et stockez le résultat dans une variable `dureeTitres` de type `Array`.
6. Stockez dans une variable dénommée `dureeTotale` de type `Float` la durée totale de l'album.
7. Complétez la fonction `lancerStats()` pour lancer une alerte qui contient : le nom de l'album, son nombre de chansons et sa durée totale tel qu'illustré en Figure 103.