Designer Web

ECF Intégration JQuery & PHP

4H

DEROULEMENT DE L'EPREUVE PRATIQUE

Cette épreuve pratique a pour but de mesurer votre capacité à manipuler les langages de programmation JQuery et PHP.

Critères d'évaluation

Vous devez réaliser les tâches demandées EXACTEMENT telles qu'elles sont exprimées.
Tout élément qui ne serait pas identique au cahier des charge sera interprété comme une non-conformité et ne sera pas comptabilisé lors de la correction. Tout ce qui n'est pas précisé est laissé à votre libre interprétation.
Pour réussir cette épreuve, vous devez répondre à au moins 60% des exigences exprimées.

Mise en place de l'épreuve

Récupérez auprès du surveillant le dossier nommé ecfPHP et placez-le à la racine de votre serveur web, dans C:\xampp\htdocs.
Renommez le dossier ecfPHP du nom que vous souhaitez.
Dans ce dossier vous placerez un fichier texte portant vos noms et prénoms (exemple : jean_dubois.txt). Ce fichier servira à vous identifier lors des corrections.

Vous avez terminé / fin de l'épreuve :
1 - Recopiez TOUT le dossier de travail
2 - Placez la copie sur le bureau
3 - Renommez le dossier avec votre prénom
4 - Faites venir le surveillant afin qu'il récupère votre dossier

Durée

Vous disposez de **4h** pour mener à bien les tâches décrites dans ce document.

Documentation

Cette épreuve est "à livre ouvert" : vous avez accès à Internet, aux supports de cours ainsi qu'à vos notes personnelles. Les livres personnels ne sont PAS autorisés sauf si chacun en possède un exemplaire.
Aucune aide externe ou interne n'est autorisée.

L'examinateur ne répondra à aucune question d'ordre technique ou fonctionnel sauf pour apporter un correctif à l'épreuve.
Bonne chance à toutes & tous.

CAHIER DES CHARGES

Aspects généraux

Vous devez intégrer du code JQuery et du code PHP à un site HTML/CSS déjà codé, notamment :

- Transformer les codes HTML récurrents en fonctions PHP
- Créer des fonctions PHP afin de contrôler l'affichage de blocs de code
- Analyser le contenu PHP reçu d'un formulaire
- Valider/invalider des champs à la volée à l'aide de JQuery
- Charger dynamiquement du contenu externe dans la page en cours

Eléments fournis

Le dossier **ecfPHP** est pré-alimenté avec un site web déjà fonctionnel, comportant 5 pages : **index, apropos, contact, implantations, email.**

La librairie JQuery est également fournie et intégrée : **jquery.tool.min.js.**

Une page **includes/functions.php** existe également, et contient du code déjà testé par le développeur :

- La fonction **menu()** qui se charge d'afficher le menu principal

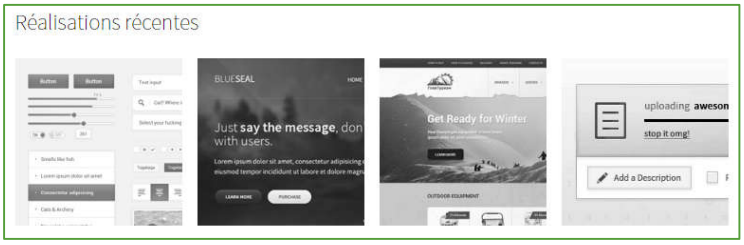
Un fichier de données Excel (**implantations.xls**) contenant la liste des pays où le client est présent.

1.1 Réalisations attendues

1.1.1 Etape 1 - DECOUPAGE PHP (20 points)

Exporter les codes communs d'entête et de pied de page dans deux fichiers externes placés dans le dossier **includes** et les réincorporer à chacune des 5 pages du site.

1.1.2 Etape 2 - FONCTIONS PHP (20 points)



Créer en une fonction qui, grâce à un paramètre fourni, permet de générer autant de blocs de réalisations récentes en page d'accueil qu'on lui en demande.

Par exemple, si on lance **crea_realiz(33),** la fonction créera 33 blocs réalisations tandis que **crea_realiz(12)** n'en créera que…12 bravo !

La fonction devra vérifier que le nombre demandé n'est pas supérieur à 64 car seules 64 images de portfolio (**portfolio1.jpg** à **portfolio64.jpg**) sont fournies dans le dossier **assets/img**.

Si le nombre demandé dépasse 64, alors la fonction n'en affichera que 64, dans le cas contraire, elle affichera effectivement le nombre de blocs de réalisations demandés.

Vous lancerez la fonction en demandant 16 images, puis 32 et enfin 70.

1.1.3 Etape 3 - TABLEAUX PHP (30 points)

Dans le fichier Excel **implantations.xls,** vous trouverez une liste brute d'extensions et de pays que vous utiliserez pour créer les blocs de la page **implantations.php**.

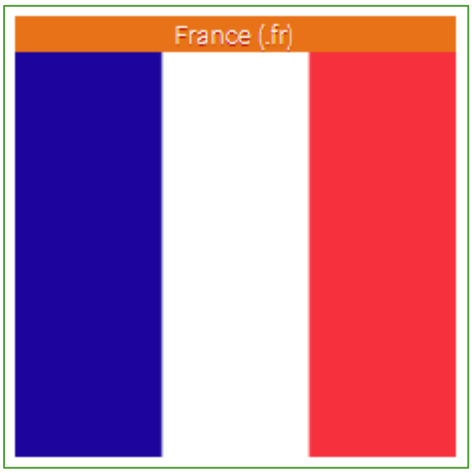
L'idée est de **transformer la liste en un tableau de données PHP,** puis de **traiter ce tableau** directement dans la page **implantations.php** afin de générer les blocs.

Chaque pays de la liste devra générer un bloc, sur le même modèle que le seul pays déjà affiché (mais bien entendu avec ses données spécifiques).

Observez bien le code existant (au besoin en chantant la Marseillaise) pour comprendre comment l'exploiter.

Les images de drapeaux sont dans **assets/drapeaux.**

Etat normal



Etat survolé

1.1.4 Etape 4 - THIS IS JQUERY ! (40 points)

Vous devez modifier le formulaire en page **contact** afin que les champs indiquent en temps réel s'ils sont correctement remplis ou non.

Les champs obligatoires sont facilement reconnaissables : ils portent la classe... "*obligatoire*"

Vous donnerez les classes "*ko*" et "*ok*" (déjà codées en CSS) aux champs de texte selon qu'ils sont ou pas valides.
Vous modifierez aussi les images associées : *ko.png* et *ok.png*, toutes deux situées dans **assets/img**/.

Chaque champ obligatoire porte donc la classe "obligatoire" et, selon qu'il est valide ou pas, la classe "ok" ou "ko".

La validité d'un champ dépend du nombre de caractères qu'il contient.

Ce nombre minimal est stocké dans un attribut personnalisé "**data-longmini**" dans chaque champ.

> **Un champ obligatoire est donc valide (classe ok) lorsqu'il contient autant ou plus de caractères qu'indiqué dans son attribut *data-longmini*. En dessous de cette valeur, le champ doit être considéré invalide (classe ko).**

Le test de validité de l'adresse email (présence d'arobase, de point, d'extension) ne fait partie du périmètre : laissez donc cette vérification de côté.

Après un brainstorm de plusieurs semaines et concertation des plus brillants cerveaux de la planète, un algorithme de validation des champs a été défini.

C'est celui que vous devez coder/traduire afin de valider le formulaire en temps réel.

Dès qu'une touche est tapée dans un des champs obligatoires :

- récupérer le nombre de caractères de ce champ-ci
- récupérer son nombre de caractères minimum (attribut data-longmini)
- **comparer le nombre de caractères entrés avec le nombre minimum pour ce champ.**

Si le champ est valide, alors :

- lui donner la classe **ok** en retirant la classe **ko**
- afficher l'image "**ok**" dans l'emplacement de validation associé au champ (remonter au parent du champ et chercher un enfant img)

Sinon :

- lui donner la classe **ko** en retirant la classe **ok** et afficher l'image "ko" dans l'emplacement de validation associé au champ (remonter au parent du champ et chercher un enfant img)

Dans tous les cas : compter le nombre d'éléments portant la classe **ko**.

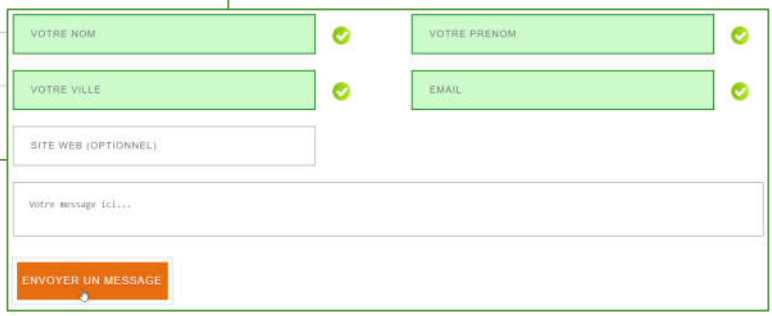
Si ce nombre est 0, alors aucun champ obligatoire n'est invalide et on peut donc activer le bouton de soumission, sinon il faut désactiver le bouton de soumission.

Le bouton de validation est initialement désactivé et ne devrait être activé que lorsque les 4 champs obligatoires portent tous la classe "ok", autrement-dit quand tous les champs sont valides.

Ainsi il n'est pas possible de soumettre le formulaire tant que tous les champs ne sont pas valides.

Tous les champs ne sont pas valides : le bouton submit est désactivé

Tous les champs sont valides : le bouton submit est actif.



Rappel : *disabled* est une propriété, pas un attribut.
Rechercher au besoin "*jdn désactiver input jquery*" sur Google...

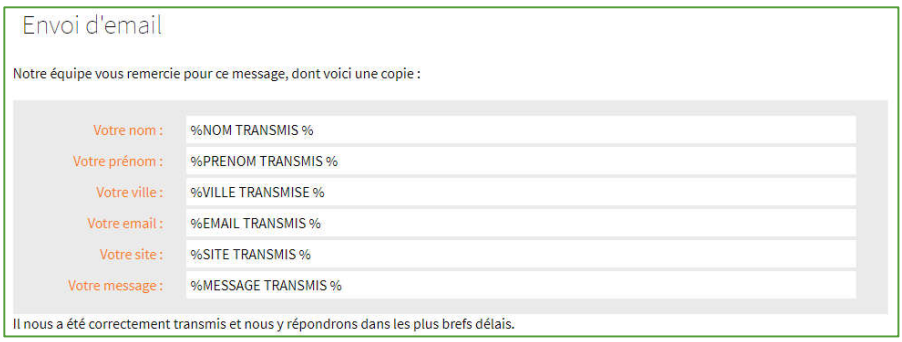
Note : ceux qui ne parviendraient pas à coder cet affichage/masquage du bouton submit, devront le court-circuiter et rendre le bouton systématiquement opérationnel afin de pouvoir tout de même soumettre le formulaire et réaliser l'étape 5.

1.1.5 Etape 5 - RECUPERATION DE DONNEES EN PHP (20 points)

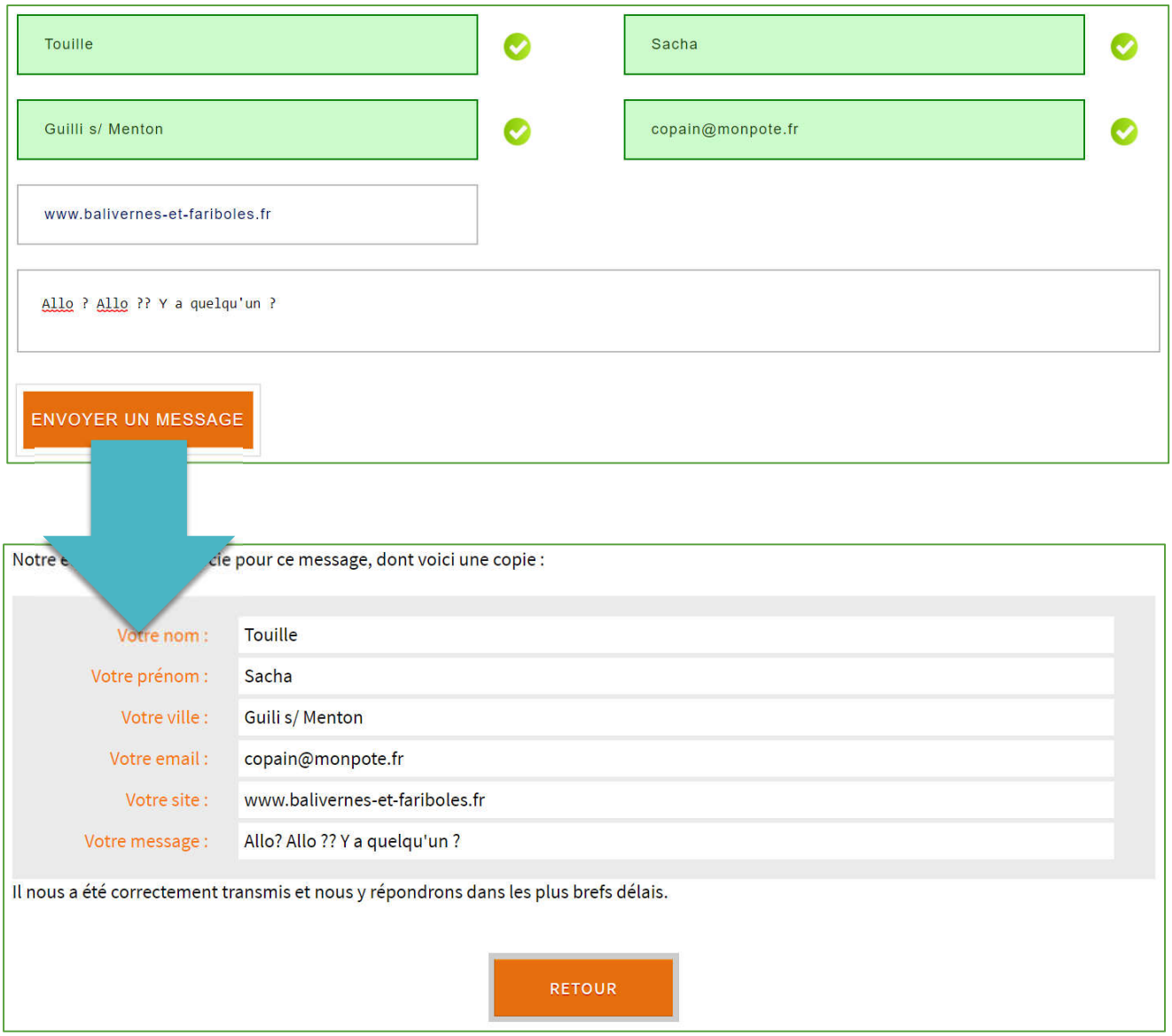
Le formulaire, une fois validé doit envoyer les données à la page **email.php**.
La page **email.php** doit afficher un récapitulatif des données reçues.

Le texte et la mise en forme sont déjà fournis dans la page **email.php.**

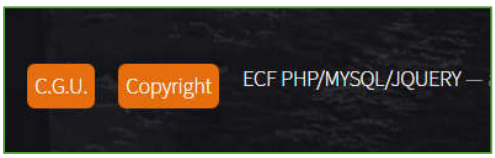
Par défaut, **email.php** affiche des données "%bidon%" : vous ferez en sorte de les remplacer par les véritables données transmises par le formulaire.



Aucun test ni envoi des données n'est demandé, il ne s'agit que d'afficher ce qui est reçu.



1.1.6 Etape 6 - JQUERY & AJAX (20 points)



Chaque page comporte en entête un ensemble de div cachées (#couverture, #popup, #popup_contenu et #close) déjà stylées en CSS ainsi que 2 liens en bas de page : **cgu** et **copyright**.

Deux fichiers **cgu.txt** et **copyright.txt** sont également fournis à la racine du site.

Vous utiliserez vos connaissances en JQuery et en AJAX pour coder les fonctionnalités suivantes :

- un clic sur **C.G.U.** charge le contenu du fichier **cgu.txt** dans la div *#popup_contenu* et rend la div *#couverture* visible

- un clic sur **Copyright** charge le contenu du fichier **copyright.txt** dans la div *#popup_contenu* et rend la div *#couverture* visible

- un clic sur *#close* masque la div *#couverture*.

Total 150 points

FIN DE L'EPREUVE