Documentation technique du site climat-2030.org

Au **07/07/2023**

Site : climat-2030.org

Hébergeur : OVHcloud

Connexion FTP :

Hôte : ftp.cluster030.hosting.ovh.net

Utilisateur : bgooodi-aurelien

Mot de passe : 23marsDonuts

Port : 22

Dossier courant : /www/climat-2030/

Attention :

Le site utilise la réécriture d'URL. Tous les fichiers passent par /configuration.php (hormis ceux de la partie administrateur). Toutes modification de lien ou de nom de fichier doit être répercutée dans le tableau MAPPING dans /configuration.php (sauf pour la partie administrateur).

Pour plus de détails, voir les commentaires dans les fichiers en question.

SOMMAIRE :

2 – Côté administrateur :

2.1 - isAdmin.php
2.2 - linkdb.php
2.3 - login.php
2.4 - my-account.php
2.5 - customers.php
2.6 - prices.php
2.7 - users.php
2.8 - contents/
2.9 - contents/part/
2.10 - content.php
2.11 - images/

III - Base de données

IV - Recommandation

V - Notes

I - Fichier JavaScript :

1 - Côté utilisateur :

1.1 - autoscroll.js :

Utilité : /fr/club-foret.php ; /en/forest-club.php (partie AF)

Le fichier autoscroll.js contient deux fonctions. La première : autoScroll() permet au carrousel de passer automatiquement d'une slide à l'autre. La deuxième fonction : stopAutoScroll() permet l'arrêt de la fonction autoScroll() lorsque l'utilisateur appuis sur un input[type=radio] pour sélectionner une slide.

Pour modifier le timer de la fonction autoScroll(), il suffit d'augmenter ou de diminuer la valeur de setInterval() : setInterval(function () { … } , ***valeur à modifier***).

1.2 - carousel.js :

Utilité : /fr/club-foret.php ; /en/forest-club.php (partie Étudiants)

Le fichier carousel.js contient trois fonctions. La première updateCarousel() permet d'ajouter la classe active à l'élément actif et la retirer aux autres. Elle est appelée par les fonctions suivantes. Les deux autres fonctions : nextSlide() et previousSlide() permettent de passer à la slide suivante ou précédente lorsque l'utilisateur clique sur les flèches. Elles appellent la fonction updateCarousel().

1.3 – contact.js :

Utilité : /fr/contact.php ; /en/contact.php

Le fichier contact.js contient six fonctions. La première : afficherChamps() permet d'afficher ou non les inputs de la page contact en fonction de ce qui est sélectionné. La seconde : compterCaracteres() compte les caractères et l'affiche (caractères max pour le message).

Pour modifier le nombre mac de caractères : modifier l'attribut du textarea maxlength= ''***X***''.

La troisième fonction : returnPrix() renvoie l'éco-participation en fonction du nombre d'élève et du prix du cours. Elle est utilisée dans les fonctions suivantes. Les deux dernières fonctions : estimationPrix() et estimationPrixEn() retourne une estimation du prix de la formation pour un certain nombre d'élèves et un prix du cours (éco-participation * nombre d'élève / 12). L'une sert pour la version française et l'autre, pour la version anglaise.

1.4 – countdown.js :

Utilité : /fr/index.php ; /en/index.php

Le fichier countdown.js contient deux fonctions. La première : countdown() permet d'afficher le temps qu'il reste avant la deadline (1er janvier 2030 à minuit).

Pour modifier la deadline : modifier la constante deadline = new Date('***aaa-mm-jjTh-min-sec***'')

La seconde : countdownPopUp() permet de zoomer sur le compteur lorsque l'utilisateur appuie sur 'en combien de temps ?'.

Pour modifier la taille du cadre après le zoom, modifier le pourcentage de la taille de l'écran : width = `${hauteurEcran * ***XX*** / 100}px` ;

height = `${hauteurEcran * ***XX*** / 100}px` ;

Pour modifier la durée de la transition, modifier : transitionDuration = '***X.X***s' ;

Pour modifier les taille des chiffres et des unités : fontSize1 = '***X.XX***rem' (unité) ;
fontsize2 = '***X.XX***rem' (Chiffres).

1.5 - search.js / enSearch.js :

Utilité : /fr/php/header.php ; /en/php/header.php

Les fichiers search.js et enSearch.js permettent de rechercher parmi les pages existantes sur le site. search.js est dédié aux pages en français et enSearch.js est dédié au page en anglais.

Pour rajouter ou enlever des pages : deux étapes :

1 : les ajouter / enlever dans words (qui liste les éléments qui seront proposés)

2 : les ajouter / enlever dans le tableau pageUrls (qui associe le nom de la page avec celui du fichier) ***Attention :*** Les nom des pages dans words et la clé dans pageUrls doivent être parfaitement identiques.

1.6 - searchActive.js :

Utilité : /fr/php/header.php ; /en/php/header.php

Le fichier searchActive.js permet d'afficher la barre de recherche lorsqu'on clique sur l'icône.

1.7 - main.js :

Utilité : /fr/php/header.php ; /en/php/header.php + toutes les pages

Le fichier main.js contient 3 fonctions. La première, utilisant jQuery est utilisé pour l'apparition de la flèche en bas à droite de l'écran permettant de remonter en haut de la page.

Deux paramètres sont modifiables : à partir de quand la flèche apparaît : if ($(this).scrollTop() > ***XXX***) et de combien elle se décale de la droite : $('.scrollUp').css('right','***XX***px').

Les 2 dernières fonctions : changerLeft(x) ; changerLeft2(x) sont dédiées à la navBar en mode responsive. ChangerLeft(x) permet de faire apparaitre la navBar de ***x*** px et de bloquer l'overflow et de calculer la largeur de la bande qu'il reste une fois la navBar déployée entre le bord droit de la navBar et le bord droit de l'écran et d'appliquer cette largeur à la div qui se positionne par-dessus le contenu et permet de renvoyer la navBar vers la gauche lors du click dans cette zone. ChangerLeft2(x) permet de rétablir l'overflow et de revoyer la navBar sur la gauche de ***x*** px et de faire disparaitre la div qui se positionnent par-dessus le contenu et permet de renvoyer la navBar vers la gauche lors du click dans cette zone.

2 - Côté Administrateur :

2.1 - add.js :

Utilité : /admin/tables /users.php ; /admin/tables /customers.php

Le fichier add.js contient la fonction ajouter() qui permet de faire apparaitre / disparaitre la ligne d'ajout.

2.2 - delete.js :

Utilité : /admin/tables /users.php ; /admin/tables /customers.php

Le fichier delete.js contient deux fonctions. La première : effacerCustomers(id) qui permet d'afficher une fenêtre Pop-up de validation afin d'éviter les erreurs. Elle prend en argument l'identifiant du customers à supprimer. De même pour la seconde fonction : effacerUsers(id) qui permet elle aussi l'affichage d'une pop-up de vérification afin d'éviter les erreurs. Elle prend en argument l'identifiants du User à supprimer.

Il est possible de modifier le message qui apparait : if (window.confirm(***'Êtes-vous sûr de vouloir supprimer la ligne '*** + id + ***' ?'***)) {

Il est possible de modifier le chemin du fichier vers lequel la fonction renvoie : window.location.href = ***'/admin/tables/customers.php?action=delete&id='*** + id;

2.3 – filtres.js :

Utilité : /admin/tables/users.php ; /admin/tables /customers.php

Le fichier filtrs.js contient trois fonctions. La première : filtres() permet d'afficher le formulaire de filtres en ajoutant / supprimant la classe 'invisible'. La seconde : afficherSort() permet d'afficher les inputs de tri si le trie est sélectionné. De même pour la troisième fonction : afficherSearch() qui permet d'afficher les inputs de recherche si la recherche est sélectionnée.

2.4 – info.js :

Utilité : /admin/tables/users.php ; /admin/tables/customers.php ; /admin/tables/images/ ; /admin/tables/contents/

Le fichier info.js permet d'afficher la div contenant les informations liées aux tables lors du survol du 'i' d'information.

2.5 – main.js :

Utilité : toutes les page dans /admin/

Le fichier main.js contient une fonction utilisant jQuery utilisé pour l'apparition de la flèche en bas à droite de l'écran permettant de remonter en haut de la page.

Deux paramètres sont modifiables : à partir de quand la flèche apparat : if ($(this).scrollTop() > ***XXX***) et de combien elle se décale de la droite : $('.scrollUp').css('right','***XX***px').

2.6 – miseEnForme.js :

Utilité : /admin/tables/contents/

Le fichier miseEnForme.js contient plusieurs fonctions. La première : miseEnForme() permet d'afficher ou non la div de mise en forme lors du clic sur le bouton 'mise en forme'. Elle ajoute / supprime la classe 'invisible'. La partie suivante permet la sélection des boutons de mise en forme puis de générer le texte à copier. La fonction copyToClipboard() permet de copier le texte généré précédemment dans le presse-papier et de l'afficher. La fonction startDrag() permet de détecter lorsque la div est 'prise' par la souris, la fonction drag() permet de la bouger et la fonction stopDrag() permet d'arrêter le processus.

2.7 – prices.js :

Utilité : /admin/tables/princes/php

Le fichier prices.js contient les fonctions qui génèrent les graphiques. On passe en argument le tableau dans lequel est stocké les données. Les lignes sont initialisées vide puis, remplie grâce à la double boucle for. Le graphique est ensuite créé grâce à la librairie chart.js. Dans datasets on indique le label (nom de la colonne), les données (générées ci-dessus et stockées dans des variables (allant de row1 pour la première puis, data1 à data8)). Ensuite, les options : le responsive et l'attribut 'stacked' sont activés (stacked permet d'avoir plusieurs courbes sur un même graphique).

2.8 – repereId.js :

Utilité : /admin/tables/contents

Le fichier repereId.js permet de remonter tous les repères de chaque contenu ce qui permet de revenir sur le contenu après modification sans qu'il soit sous la navBar.

Il est possible de modifier la hauteur avec laquelle est reporté le repère : element.style.top = (currentTop - ***XXX***) + 'px';

2.9 – update.js :

Utilité : /admin/tables/customers.php ; /admin/tables/users.php

Le fichier update.js permet d'afficher ou non la ligne de modification en lui ajoutant / supprimant la classe 'invisible'.

2.10 – updateContents.js :

Utilité : /admin/tables/contents/

Le fichier updateContents.js permet de remplacer le contenue par un textarea pour le modifier. Il ajoute / supprimer la classe ‘invisible’.

2.11 – updatePrices.js :

Utilité : /admin/tables/prices.php

Le fichier updatePrices.js permet de remplacer le tableau des prix par un formulaire permettant de modifier les prix. Il ajoute / supprimer la classe ‘invisible’.

II - Fichier PHP :

1 - Côté utilisateur :

1.0 - Intégration des textes et des images :

Étant donné que les contenus du site doivent pouvoir être modifiés régulièrement, tous les texte et les chemins des images sont stockés dans la base de données : la table `contents` pour les textes et la table `images` pour les images.

Ces données sont passées dans trois tableaux multidimensionnels : deux pour les textes (un contenant les texte de la base de données avec le système de balisage propre au site et l'autre avec les balises remplacées par des balises HTML) et le troisième pour les images.

Pour plus d'informations concernant les tableaux est leur construction : se référer au 2.10 - content. PHP

Le tableau contenant les textes est construit de la manière suivante :

```
contenuView (ou contenu) = [
	pageX = [
		langue1 = [
			nomContenu = [
				contenu = contenuX,
				id = idX
			]
		],
		langue2 = [
			nomContenu = [
				contenu = contenuY,
				id = idY
			]
		]
	],
	pageY = [
		langue1 = [
			nomContenu = [
				contenu = contenuX,
				id = idX
			]
		],
		langue2 = [
			nomContenu = [
				contenu = contenuY,
				id = idY
			]
		]
	]
]
```

L'appel d'un contenu donne : $contenuView[page][langue][nomContenu][contenu]

Le tableau contenant les chemins des images est construit de la manière suivante :

```
images = [
        pageX = [
                nomImage = [
                        image = cheminX,
                        id = idX
                ]
        ]
        pageY = [
                nomImage = [
                        image = cheminY,
                        id = idY
                ]
        ]
]
```

L'appel d'un contenu donne : $image[page][nomImage][image]

1.1 - configuration.php :

---- RÉÉCRITURE D'URL ----

Le fichier configuration.php contient un tableau et deux fonctions. Le tableau MAPPING associe l'élément écrit dans l'URL (après le domaine) et le chemin de la page auquel il correspond. La première fonction : getVarFromUrl() permet de d'associer l'élément de l'URL avec le chemin de la page auquel il correspond. Si rien ne correspond, le chemin de la page 404 est renvoyée. La seconde fonction : retrieveVar() permet de récupérer l'élément de l'URL (après le domaine).

1.2 - /index.php :

---- RÉÉCRITURE D'URL ----

Le fichier /index.php permet d'afficher le fichier de la page demandée (chemin renvoyé par la fonction getVarFromUrl() dans configuration.php).

À NOTER : Toutes les pages du site sont chargées à partir de /index.php ce qui signifie que TOUS les fichier inclus dans les pages (feuille de styles, fichier de script et include PHP) doivent avoir des liens absolus si possible ou relatif EN PARTANT DU FICHIER /index.php !

1.3 - header.php :

Le fichier header.php contient le header (en français et en anglais et inclus dans chaque page). Il contient un tableau associatif qui associe chaque page en français avec la page correspondante en anglais. Une vérification est effectuée : si la page n'existe pas dans le tableau, la page 404 est renvoyée. Sinon la page correspondante en anglais et en français. Ce code permet de passer de l'anglais au français et inversement tout en restant sur la même page.

1.4 - footer.php :

Le fichier footer.php contient le footer (en français et en anglais et inclus dans chaque pages). Il contient un tableau associatif qui associe chaque numéro de mois au mois en question. Il récupère la date courante (dans les variables $jour, $mois et $annee et concaténé dans la variable $date). Cela permet d'afficher la date courante dans le compteur à impact (div.global-counter). Ensuite, le nombre de ligne de la table `customers` est récupérer pour afficher dans le compteur a impact (div.global-counter).

1.5 - contact.php :

Le fichier contact.php contient le code permettant d'envoyer un e-mail lors de la validation du formulaire de la page contact. Tout d'abord, une vérification est effectuée : si la variable 'action' existe dans la variable $_POST et si elle est égale à 'mail', alors, toutes information passées dans $_POST sont passées dans des variables. Certaines variables sont modifiées par la suite à l'aide de conditions.

Ensuite, le contenu du mail est créé : il y a trois possibilités :si le choix est une question, si le choix est un devis et qu'il concerne le club pédagogique et si le choix concerne les autres devis.

Puis, le destinataire, l'objet et les headers du mail sont renseignés dans des variables, le mail est envoyé et une redirection est effectuée (un message est passé lors que la redirection en GET pour afficher si le mail est envoyé ou non (messages contenus dans le tableau associatif $tab_message).

2 - Côté administrateur :

2.1 - isAdmin.php :

Le fichier isAdmin.php permettant de vérifier si l'utilisateur connecté dispose des privilèges d'administrateurs. Elle est utilisée sur toutes les pages de la partie administrateur. Dans le cas où l'utilisateur n'est pas administrateur, il est redirigé vers la page de login.

2.2 - linkdb.php :

Le fichier linkdb.php contient les constantes permettant de se connecter à la base de données. Ensuite, on tente une connexion (try / catch).

2.3 - login.php :

Le fichier login.php contient le nécessaire pour se connecter, se déconnecter et afficher des messages d'erreur : pour se connecter, on vérifie tout d'abord si $_POST est vide ou non. Dans le cas ou elle n'est pas vide, on vérifie l'existence de 'mail' et 'password'. S'ils existent, on vérifie qu'ils correspondent à un jeu d'identifiants présent dans la base de données. Si c'est le cas, une session est lancée et les données liées à l'utilisateur sont envoyé dans la variable $_SESSION sinon, un message d'erreur est renvoyé.

Ensuite, pour la déconnexion, on vérifie que $_GET n'est pas vide. Si elle n'est pas vide et 'destroy' existe dans $_GET, la session est détruite à l'aide de la fonction session_destroy().

Enfin, il y a la génération de message d'erreur. Elle créer le message d'erreur si la valeur passée dans $_GET correspond.

2.4 - my-account.php :

Le fichier my-account.php contient le nécessaire pour modifier le mot de passe de l'administrateur connecté. Il vérifie d'abord que les données soient passé dans $_POST puis, compare les trois mots de passe (l'ancien, le nouveau et la vérification du nouveau) si les deux nouveaux mots de passe correspondent et que l'ancien est valide, le mot de passe est modifié. Un message est affiché permettant de savoir si la manœuvre a réussi ou non.

2.5 - customers.php :

Le fichier customers.php contient le nécessaire pour supprimer, ajouter, modifier, trier et rechercher des clients. L'action à réaliser est passer dans la variable $_GET['action'].

Supprimer : lorsque $_GET['action'] == 'delete'. L'id du client à supprimer est aussi passé dans $_GET. Si l'id == 'all', la table est vidée.

Ajouter : lorsque $_GET['action'] == 'add'. Toutes les données sont passées dans $_GET. Elles sont ensuite mises dans des variables (et passe par la fonction htmlspecialchars() qui permet d'éviter les attaques XSS). Les données sont ensuite insérées dans la base de données.

Modifier : lorsque $_GET['action'] == 'update'. Toutes les données sont passées dans $_GET. Elles sont ensuite mises dans des variables (et passe par la fonction htmlspecialchars() qui permet d'éviter les attaques XSS). Les données sont ensuite modifiées dans la base de données.

Trier et **Rechercher** : lorsque $_GET['action'] == 'sortSearch'. Les paramètres du tri et de la recherche sont passés dans $_GET. Ils sont ensuite mis dans des variables. Une vérification est ensuite effectuée : si les paramètres de recherche sont vides, un tri est effectué ; si les paramètres de tri sont vides, une recherche est effectuée ; si aucuns paramètres n'est vide, un tri ET une recherche sont effectués.

La requête SQL est ensuite exécutée.

2.6 - prices.php :

Le fichier prices.php contient le nécessaire pour modifier le tableau contenant les prix. Lorsque $_GET['action'] == 'update', on parcoure le tableau et on récupère toutes les données qui sont injectées une à une dans la base de données.

2.7 - users.php :

Le fichier users.php contient le nécessaire pour supprimer, ajouter, modifier, trier et rechercher des utilisateurs. L'action à réaliser est passer dans la variable $_GET['action'].

Supprimer : lorsque $_GET['action'] == 'delete'. L'id de l'utilisateur à supprimer est aussi passé dans $_GET. Si l'id == 'all', la table est vidée.

Ajouter : lorsque $_GET['action'] == 'add'. Toutes les données sont passées dans $_GET. Elles sont ensuite mises dans des variables (et passe par la fonction htmlspecialchars() qui permet d'éviter les attaques XSS). Les données sont ensuite insérées dans la base de données.

Modifier : lorsque $_GET['action'] == 'update'. Toutes les données sont passées dans $_GET. Elles sont ensuite mises dans des variables (et passe par la fonction htmlspecialchars() qui permet d'éviter les attaques XSS). Les données sont ensuite modifiées dans la base de données.

Trier et **Rechercher** : lorsque $_GET['action'] == 'sortSearch'. Les paramètres du tri et de la recherche sont passés dans $_GET. Ils sont ensuite mis dans des variables. Une vérification est ensuite effectuée : si les paramètres de recherche sont vides, un tri est effectué ; si les paramètres de tri sont vides, une recherche est effectuée ; si aucuns paramètres n'est vide, un tri ET une recherche sont effectués.

La requête SQL est ensuite exécutée.

2.8 - contents/ :

Le dossier contents/ contient toutes les pages permettant de modifier le contenu du site. Toutes les pages ont une structure identique : les fonction PHP sont contenues dans chaque page (dans contents/). Les contenus sont dans content/part/ (et porte le même nom que la page) et sont inclus dans les pages (dans contents/).

Chaque page contient le nécessaire pour modifier les contenus : on vérifie l'existence des paramètres 'action', 'id_contents' et 'content_contents'. Si c'est vérifier, on procède à la modification du contenue (grâce à son id).

On redirige ensuite sur la page au niveau de l'élément modifier (grâce à son id passer dans l'url (climat-2030.org/…#id).

2.9 - contents/part/ :

Dans le dossier contents/part/, il y a, comme dit ci-dessus, les contenus des pages (et content.php, repeat.php et miseEnForme.php qui ne sont pas abordés ici) permettant la modification de ces dit contenus. Pour chaque contenu, il y a quatre variables définies : $tempsContenu, $tempContenuView, $tempId et $language. La première contient le contenu à afficher (avec les balises HTML), la seconde contient le contenu modifiable (directement celui de la base de données – avec le système de balisage propre au site), la troisième contient l'id du contenu et la quatrième contient la langue du contenu ('fr' ou 'en' ou vide si le contenu est multi langue).

Ensuite, repeat.php est inclus : c'est la partie qui affiche le texte et qui gère, à l'aide de updatContents.js l'affichage ou non du textarea permettant la modification du contenu. Il gère aussi la couleur de l'icône modifier (rouge en anglais, bleu en français et blanc pour le contenu multi langue)

2.10 - content.php :

Le fichier content.php contient 6 fonctions. La première : content() qui permet la création du tableau contenant les contenus avec les balises propre au site.

La deuxième : contentView() qui permet la création du tableau contenant les contenus avec les balises HTML.

Forme des tableaux :

```
contenuView (ou contenu) = [
	pageX = [
		langue1 = [
			nomContenu = [
				contenu = contenuX,
				id = idX
			]
		],
		langue2 = [
			nomContenu = [
				contenu = contenuY,
				id = idY
			]
		]
	],
	pageY = [
		langue1 = [
			nomContenu = [
				contenu = contenuX,
				id = idX
			]
		],
		langue2 = [
			nomContenu = [
				contenu = contenuY,
				id = idY
			]
		]
	]
]
```

La troisième fonction : texteView() est utiliser par la fonction contentView() et permet d'extraire les balises propres au site : les balise sont de la forme suivante : '{{xx-yy-zz}}' ; '{{/}}' : elles commencent par '{{' et finissent par '}}'. À l'intérieur, chaque paramètre sont séparés par '-'. À l'aide d'expression régulière, la partie entre '{{' et '}}' est extraite et traitée par la quatrième fonction : RegExReplace() qui, dans un premier temps sépare chaque paramètre et les stockes dans un tableau. Ensuite, chaque paramètre est remplacé par les classe HTML correspondante (stockés dans le tableau associatif $replace). La balise HTML est renvoyée et la balise propre au site est enfin remplacé par la balise HTML grâce à la troisième fonction : texteView().

La cinquième fonction : images() permet la création du tableau contenant les chemins des images.

Forme du tableau :

```
images = [
        pageX = [
                nomImage = [
                        image = cheminX,
                        id = idX
                ]
        ]
        pageY = [
                nomImage = [
                        image = cheminY,
                        id = idY
                ]
        ]
]
```

Pour finir, la sixième fonction : prices() permet de générer le tableau multidimensionnel contenant les prix.

Forme du tableau :

```
prix = [
        row = [
                col = [
                        prix = prixX,
                        id = idX
                ]
        ]
        row = [
                col = [
                        prix = prixY,
                        id = idY
                ]
        ]
]
```

2.11 - images/ :

Le dossier images/ contient toutes les pages permettant de modifier les images du site. Toutes les pages ont une structure identique : les fonction PHP sont contenues dans chaque page (dans images/).

Chaque page contient le nécessaire pour modifier les images: on vérifie l'existence du paramètre 'action'. S'il existe et s'il est égale à 'updateImage' alors on récupère le chemin de l'image à modifier, le nouveau chemin et, on supprime l'ancienne image, on la remplace par la nouvelle et on modifie le chemin dans la base de données. Si une erreur se produit, on affiche un message d'erreur sinon, on redirige sur la page.

III - Base de données :

La base de données contient 5 tables :

- contents : stockage des contenus
- images : Stockage des chemins des images
- customers : stockage des informations liées aux clients
- prices : stockage des prix
- users : stockage des utilisateurs du site

À noter : Le site à subit plusieurs modifications notamment sur les noms de fichier. Ces modifications ont été répercutées sur le site mais pas sur la base de données. Toujours garder en tête que ‘académie du climat’ est devenu ‘club pédagogique’ et que ‘social club’ est devenu ‘club climat’.

IV – Recommandation :

Dans un but d'amélioration du site, voici quelques points qui pourraient être modifiés :

- Lors de l'arrivée sur la page contact après le clic sur un bouton d'inscription présent à l'intérieur d'une page, il pourrait être intéressant de pré-remplir le formulaire.

- Toujours dans le formulaire de contact, ajouter un calendrier lors du choix 'club-pédagogique' pour planifier directement les formations.

V - Notes :

Liens utiles :

- Compresseur d'image (pour éviter que les images chargent trop lentement) : cliquez ici

- Enlever le fond (pour intégrer des images sur des parties comportant déjà un fond) : cliquez ici

À noter : Lors de modifications de style (fichier CSS) ou de script (fichier JavaScript) pensez à vider le cache du navigateur si les changements ne s'appliquent pas.

Manœuvre pour Chrome :

- Sur PC : clic droit (n'importe où sur la page) → inspecter → clic gauche puis droit sur le bouton actualiser → 'vider le cache puis effectuer une actualisation forcée'

- Sur téléphone : historique → effacer les données de navigation → effacer les cookies, images et fichiers en cache.