Rapport Javascript

L'objectif de ce rapport est de présenter et expliquer les différents scripts utilisé sur le site web personnel.

Le changement de style (script.js)

Afin de changer de thème j'ai utilisé l'évènement onclick de mon lien situé dans le menu :

```
<li><a href="#" onclick="themeManager()">Changer de thème</a> </li>
```

C'est donc la fonction themeManager() qui sera appelé en premier lorsque l'utilisateur souhaite changer le thème.

```
function themeManager() {
    var currentTheme = localStorage.getItem("currentstyle");
    if(currentTheme == "style.css")
        changeCSS("dark.css");
    else
        changeCSS("style.css");
}
```

Dans la fonction ci-dessus nous récupérons d'abord le thème courant qui est stocké dans le localStorage (une alternative au cookie) à la clef "currentstyle". Ensuite celons la valeur contenu dans la variable currentTheme je change le thème en conséquence grâce à la fonction changeCSS().

```
function changeCSS(cssFile) {
    var currentTheme = cssFile;
    document.getElementsByTagName("link")
        .item(1).setAttribute("href", cssFile);
    localStorage.setItem("currentstyle", currentTheme);
}
```

Dans la fonction ci-dessus je récupère le deuxième élément link dans mon html et je change le lien par le paramètre de la fonction. On prend le deuxième élément link dans la html car le premier contient le css de base commun aux deux thèmes. Ensuite je mets à jour le contenu de la clef currentstyle dans le localStorage.

Cependant ces deux fonctions ne suffisent pas sauvegarder le thème choisi par l'utilisateur (notamment perdu en changeant de page). Pour cela j'ai fait appel à la fonction window.onload qui est appelée lorsque la page à fini de se charger.

```
window.onload = function (){
    var currentstyle = localStorage.getItem("currentstyle");

    if(currentstyle == null)
        localStorage.setItem("currentstyle", "style.css");
    else
        changeCSS(currentstyle);

    // La suite de la fonction
}
```

Tous d'habord je récupère le thème courant du localStorage si il n'existe pas (ce qui est fort probable lors de la première navigation sur le site) alors je mets à jours le localStorage avec le thème par défaut. Sinon s'il existe une valeur je change le css par la bonne valeur.

L'animation des barres de progression (progressbar.js)

Sur la page de CV il y a des barres de progression avec une animation de chargement pour atteindre un bon pourcentage pour cela lors du chargement de la page (window.onload) j'appelle la fonction loadProgress qui a pour objectif de lancer l'animation de tous les div qui ont la classe "progress-bar".

```
function loadProgress() {
   var cpt = 0;
   var barList = document.getElementsByClassName("progress-bar");
   while(cpt < barList.length) {
      var bar = barList[cpt];
      var value = bar.getAttribute("value");
      if(value)
         progressBar(bar, value);
      cpt++;
   }
}
```

Dans cette fonction je récupère tous d'abord tous les éléments par classe (renvoyé sous forme de tableau) et ensuite de fait une boucle jusqu'à la taille de ce tableau. A chaque itération de boucle je vais récupérer l'objet ayant la bonne classe et je récupère la valeur de l'attribut "value", ensuite si il a bien cette attribut je fais a la fonction progressBar() qui fait créer l'animation sur l'objet pris en paramètre jusqu'à la valeur récupéré.

Ci-dessous un exemple de bar de progression représenté sous format HTML.

```
<div value="80" class="progress-bar grey">80%</div>
```

Dans la fonction ci-dessous progressBar il y a deux grande étape

La première consiste à s'assurer que la largeur de la barre est bien à 0% et que son texte est bien à égal à « 0% » aussi (récupéré grâce à innerHTML, on aurait pu prendre aussi innerText)

```
function progressBar(bar, finalPurcent) {
    bar.style.width = 0 + "%";
    bar.innerHTML = "0%";
    var purcent = 0;
    // Le reste de la fonction
}
```

La deuxième parti consiste à créer l'animation pour cela j'ai utilisé la fonction setInterval() qui me permet d'appeler une fonction tous les x milliseconde (ici tous les 10ms on exécutera la fonction interne frame()). Dans la fonction frame on se contente juste de mettre à jours la largeur de la barre et son texte, on vérifie ensuite si le pourcentage de la barre est bien celle que l'on souhaitais si oui alors on arrête l'animation grâce à la fonction clearInterval().

```
function progressBar(bar, finalPurcent) {
    // Le début de la fonction
    var id = setInterval(frame, 10);

    function frame() {
        purcent+=0.5;
        bar.style.width = purcent + "%";
        bar.innerHTML = purcent.toFixed(0) + "%";
        if(purcent >= finalPurcent)
            clearInterval(id);
    }
}
```

La vérification du formulaire

La vérification du nom

```
<input type="text" id="name" name="name" onblur="checkName(this);" />
```

Lorsque l'utilisateur a fini de rentrer le contenu du champ "name" alors la fonction checkName est appelée.

```
function checkName() {
    var name = document.getElementById("name");
    if(name.value == "") {
        errorField(name, "Ce champs ne peut pas être vide");
    } else {
        validField(name);
        checkForm();
    }
}
```

Ici la fonction est relativement simple on regarde si la valeur du champ "name" est vide si c'est le cas alors on fait appel à la fonction errorField() afin d'afficher un message à l'utilisateur. Sinon on montre que le champ est valide grâce à la fonction validField() puis on vérifie que le formulaire est entièrement valide afin d'activer le bouton d'envoi. Les fonctions seront étudiées par la suite.

La vérification de l'email

```
<input type="text" id="email" name="email" onblur="checkMail(this);">
```

La vérification de l'email repose sur le même principe que précédemment sauf que c'est la fonction checkMail qui est appelée.

```
function checkMail() {
    var email = document.getElementById("email");

    if(email.value == "") {
        errorField(email, "Ce champs ne peut pas être vide");
    }
    else {
        var regex = /^[a-z0-9._-]+@[a-z0-9._-]+\.[a-z]{2,6}$/;
        if(!regex.test(email.value)) {
            errorField(email, "Format d'email incorrect (ex :
toto@toto.fr)")
        } else {
            validField(email);
            checkForm();
        }
    }
}
```

Dans cette fonction on vérifie si le champ est vide. Si ce n'est pas le cas je vérifie si l'adresse est syntaxiquement correct grâce la regex contenu dans la variable de même nom et celons le résultat je valide ou pas le champ mail.

La vérification de la date

```
<input type="text" id="birthday" name="birthday"
        placeholder="JJ/MM/AAAA" onblur="checkBirthday(this);">
```

Pour la vérification de la date de naissance c'est la fonction checkBirthday() qui va être appelée.

```
function checkBirthday() {
    var birthday = document.getElementById("birthday");
    if(birthday.value == "") {
        errorField(birthday, "Ce champs ne peut pas être vide");
    }
    else {
        var regex = /^[0-9]{2}\/[0-9]{2}\/[0-9]{4}$/;
        if(!regex.test(birthday.value)) {
            errorField(birthday, "Le format de la date de naissance est
invalide (JJ/MM/YYYY)");
            return;
        }

        var date = birthday.value.split("/");
        var day = parseInt(date[0], 10);
        var month = parseInt(date[1], 10);
        var year = parseInt(date[2], 10);

        if(year < 1000 || year > 3000 || month == 0 || month > 12) {
            errorField(birthday, "La date est invalide");
            return;
        }

        var monthLenght = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31];

        if(year%400 == 0 || (year%100 != 0 && year%4 == 0))
            monthLenght[1] = 29;

        if(day <= 0 || day > monthLenght[month - 1]) {
            errorField(birthday, "La date est invalide");
        }
        else {
            validField(birthday);
            checkForm();
        }
    }
}
```

Pour la première partie jusqu'à la regex c'est le même système que pour le champ email, sauf que si la date est valide syntaxiquement alors je récupère les valeurs du jour, mois et années afin de vérifier si elles sont cohérentes (par exemple qu'on n'a pas un mois de 99 jours). Pour cela je vérifie si l'année et supérieur à 1000 et inférieur à 3000. Je vérifie ensuite si le mois est compris entre 1 et 12 sinon je renvoie une erreur. Ensuite j'ai créé un tableau qui contient le nombre de jours par mois (pour une année non bissextile). Je calcul ensuite l'utilisateur est né une année bissextile si oui alors je mets à jours le nombre de jours dans le tableau correspondant au moins de février. Après cela je vérifie si le jour un supérieur à 0 et qu'il est inférieur ou égal au maximum de jours dans le mois. Si c'est le cas alors je valide le champ.

La vérification du champ de texte

```
<textarea name="message" id="message" onblur="checkMessage(this);">
</textarea>
```

Pour le champ message c'est la fonction checkMessage() qui est appelé.

```
function checkMessage() {
    var message = document.getElementById("message");
    if(message.value == "") {
        errorField(message, "Le message ne peut pas être vide");
    }
    else {
        validField(message);
        checkForm();
    }
}
```

Cette fonction fonctionne exactement pareil que pour le champ "name".

La vérification du formulaire complet

```
function checkForm() {
    var submit = document.getElementById("submit");
    var message = document.getElementById("message");
    var birthday = document.getElementById("birthday");
    var email = document.getElementById("email");
    var name = document.getElementById("name");

    if(!hasClass(message, "inputValid") || !hasClass(birthday,
"inputValid")
        || !hasClass(name, "inputValid") || !hasClass(email, "inputValid"))
        submit.display = true;
    else
        submit.disabled = false;
}
```

Un champ du formulaire est valide si celui-ci possède la classe "inputValid" si c'est le cas pour tous les champs du formulaire alors j'active le bouton d'envoi. La fonction hasClass() renvoie un booléen vrai si l'élément en paramètre possède la classe envoyé en paramètre. Elle sera décrite plus bas dans le rapport.

Les indicateurs d'erreur

```
function errorField(field, message) {
    removeClass(field, "border-sucess");
    removeClass(field, "inputValid");
    removeClass(field.previousElementSibling.previousElementSibling,
"correct");

    addClass(field, "border-danger");
    addClass(field.previousElementSibling.previousElementSibling, "error");

    showHint(field.previousElementSibling, message);
}
```

Dans cette fonction j'enlève tous d'abord les classes (avec la fonction removeClass()) "inputValid", "border-sucess" et "correct" et j'ajoute les classes (avec la fonction addClass())"border-danger" et "error". Ensuite je fais appel à la fonction showHint() afin d'afficher un message à l'utilisateur sous forme d'infobulle. (field.previousElementSibling = L'élément au-dessus de l'élément field)

Ce qui donne l'effet visuel suivant (après les animations d'apparition) :

Les indicateurs que un champ est correct

L'affichage des indications qu'un champ est valide est l'exact contraire que si c'était incorrect si qui donne le code suivant :

```
function validField(field) {
    removeClass(field, "border-danger");
    removeClass(field.previousElementSibling.previousElementSibling,
"error");

    addClass(field, "border-sucess");
    addClass(field, "inputValid");
    addClass(field.previousElementSibling.previousElementSibling,
"correct");

    hideHint(field.previousElementSibling);
}
```

La fonction hideHint() a pour objectif de faire une animation pour cacher l'infobulle. Ce qui donne le rendu visuel suivant :

L'apparition et la disparition des infobulles

```
function showHint(hint, text) {
    hint.firstElementChild.innerText = text;
    hint.style.opacity = "0.0";
    var opacity = parseFloat(hint.style.opacity);
    var id = setInterval(frame, 50);

    function frame() {
        opacity = opacity + 0.2;

        hint.style.opacity = opacity;

        if(opacity >= 1.0) {
            clearInterval(id);
        }
    }
}
```

```
function hideHint(hint) {
    var opacity = parseFloat(hint.style.opacity);
    var id = setInterval(frame, 50);

    function frame() {
        opacity = opacity - 0.2;

        hint.style.opacity = opacity;

        if(opacity <= 0.0) {
            clearInterval(id);
        }
    }
}
```

Ces deux fonctions sont semblables à la fonction d'animation des barres de progressions sauf que celle-ci modifie l'opacité des infobulles. showHint() et hideHint() sont exactement opposées.

Les fonctions utiles

```
function hasClass(elem, value) {
    return (' ' + elem.className + ' ').indexOf(' ' + value + ' ') > -1;
}
```

Pour cette fonction je fais une recherche parmi la liste des classes si y a la classe si c'est le cas alors la fonction renverra true, sinon elle renverra false.

```
function removeClass(elem, value) {
    elem.className = elem.className.replace(new RegExp('(?:^|\\s)'+ value
+'(?!\\S)'),'');
}
```

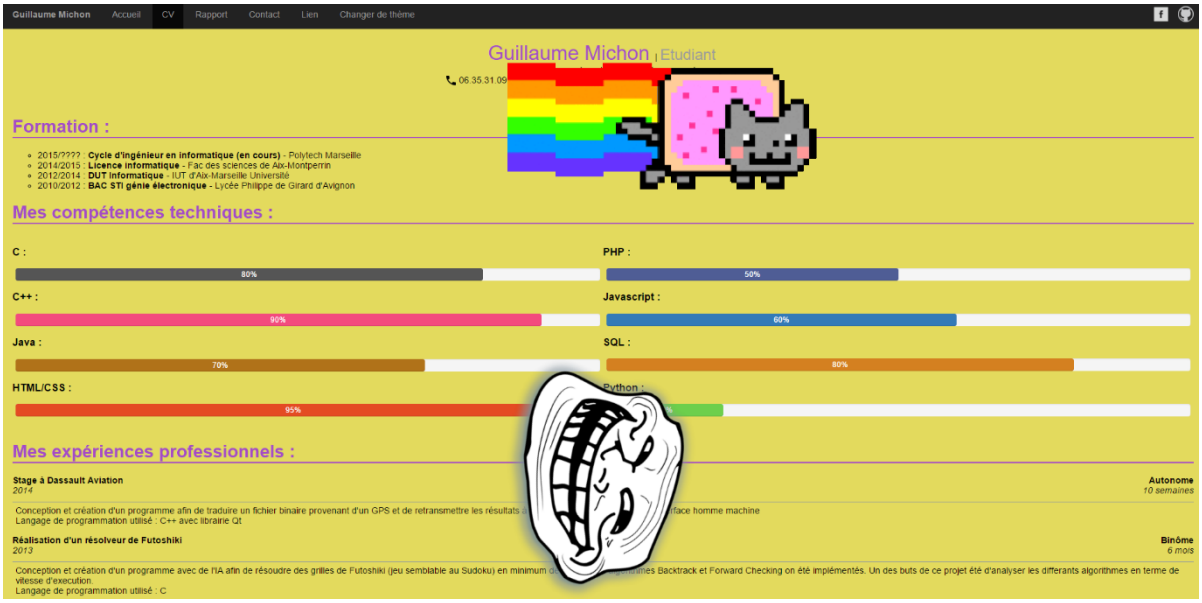
Pour cette fonction je fais une recherche de la classe à retirer parmi la liste des classes et je la remplace par du vide.

```
function addClass(elem, value) {
    if(elem.className.indexOf(value) == -1)
        elem.className = elem.className + " " + value;
}
```

Pour cette fonction je vérifie si l'élément a déjà la classe que l'on souhaite rajouté. Si ce n'est pas le cas alors je récupère la liste des classes et je concatène la nouvelle classe contenu dans la variable value.

Bonus : L'easter egg

Lancement de l'easter egg



La lancement des animations de l'easter egg ce fait lorsque l'utilisateur appuis sur la touche T pour pouvoir détecter l'événement d'appuis de touche au clavier j'ai rajouté la ligne suivante dans le window.onload(). Ce qui appellera la fonction easterEgg() à chaque appuis de touche au clavier.

```
document.addEventListener("keydown", easterEgg, false);
```

Dans cette fonction je vérifie le code de la touche si elle correspond à la touche T. Si c'est le cas je vérifie si l'easter egg est déjà en marche (si la div avec l'id "music" n'est pas vide) et lance les fonctions d'affichage ou nettoyage de l'easter egg.

```
function easterEgg(e) {
    if(e.keyCode == 84) { // Touche T
        if(document.getElementById('music').childNodes.length > 0) {
            stopSound();
            catchNyanCat();
            catchTroll();
            cleanBackground();
        } else {
            playSound();
            freeNyanCat();
            freeTroll();
            epicBackground();
        }
    }
}
```

Afin de pouvoir stopper les animations n'importe où j'ai déclaré les variables suivantes :

```
var frameNyan;
var frameTroll;
var frameBack;
```

La fond qui clignote

La fonction epicBackground() fait en sorte de changer de changer la couleur de fond du contenu du site. Pour commencer on sauvegarde la couleur courante du fond d'écran (dans le cas si dans un des thèmes on change la couleur). Et ensuite grâce à un booléen on détermine quelle couleur de fond est en cours (grâce à la variable nommée "epic") et on met à jours le fond.

```
function epicBackground() {
    localStorage.setItem("backgroundValue",
document.body.style.background);
    var epic = false;
    frameBack = setInterval(frame, 200);

    function frame() {
        epic = !epic;
        if(epic == true) {
            document.body.style.background = "#237691";
        }
        else
            document.body.style.background = "#E3DA5D";
    }
}
```

Pour le nettoyage (avec la fonction cleanBackground()) on récupère dans le localStorage la couleur et on remet la bonne couleur. Ensuite on stop l'animation du changement de couleur du fond.

```
function cleanBackground() {
    document.body.style.background =
localStorage.getItem("backgroundValue");
    clearInterval(frameBack);
}
```

La musique du Nyan Cat

Pour lancer la musique je mets à jours le HTML de la div "music" avec le contenu nécessaire grâce à la balise <object>.

```
function playSound() {
    document.getElementById('music').innerHTML =
        '<object type="audio/mpeg" width="0" height="0"
data="nyancatSong.mp3">' +
        '<param name="filename" value="nyancatSong.mp3" />' +
        '<param name="autostart" value="true" />' +
        '<param name="loop" value="true" /></object>';
}
```

Pour l'arrêt de la musique (avec la fonction stopSound()) on supprime le contenu de la div "music".

```
function stopSound() {
    document.getElementById('music').innerHTML = '';
}
```

Le Nyan Cat et le troll

Il y a 3 sens d'arrivé du l'image du Nyan Cat pour cela j'ai utilisé un Math.random pour avoir un nombre aléatoire entre 1 et 3 celons la valeur je lance ensuite la fonction correspondant au sens.

```
function freeNyanCat() {
    var nyan = document.getElementById("nyanCat");
    var rand  = Math.floor(Math.random() * 3) + 1;

    switch(rand) {
        case 1:
            freeNyanCatLeft(nyan);
            break;
        case 2:
            freeNyanCatDiagonal(nyan);
            break;
        case 3:
            freeNyanCatUp(nyan);
            break;
    }
}
```

Du fait que la plupart des fonctions qui suivents pour le Nyan Cat et le Troll sont semblable par conséquence je vais décrire que l'animation de l'arrivé du Nyan Cat par la gauche (fonction freeNyanCatLeft()). Dans la première partie de la fonction ci-dessous on récupère la largeur de la résolution de l'écran stocké dans la variable "width" et j'initialise l'image pour le positionner sur la gauche de l'écran avec une rotation de 0 (prévention des autres sens possible d'apparition des Nyan Cat). Ensuite je lance l'animation qui rajoute 3px à la position de l'image, on vérifie que la position de l'image a dépassée l'écran à droite. Si c'est le cas alors j'arrête l'animation courante et je relance la fonction freeNyanCat() afin de faire réapparaitre un autre Nyan Cat dans un autre sens peut être.

```
function freeNyanCatLeft(nyan) {
    var width = window.innerWidth
        || document.documentElement.clientWidth
        || document.body.clientWidth;

    nyan.style.left = "-1000px";
    nyan.style.top = "100px";
    nyan.style.display = "inline";
    nyan.style.marginLeft = "-350px";
    nyan.style.webkitTransform = 'rotate('+0+'deg)';
    nyan.style.mozTransform    = 'rotate('+0+'deg)';
    nyan.style.msTransform     = 'rotate('+0+'deg)';
    nyan.style.oTransform      = 'rotate('+0+'deg)';
    nyan.style.transform       = 'rotate('+0+'deg)';

    frameNyan = setInterval(frame, 1);
    var left = -1000;

    function frame() {
        left = left + 3;
        nyan.style.left = left + "px";
        if(left >= width + 350) {
            nyan.style.left = -1000 + "px";
            left = -1000;
            clearInterval(frameNyan);
            freeNyanCat();
        }
    }
}
```

Pour le nettoyage de ces deux animations j'appelle les deux fonctions ci-dessous :

```
function catchNyanCat() {
    var nyan = document.getElementById("nyanCat");
    nyan.style.display = "none";
    clearInterval(frameNyan);
}
```

```
function catchTroll() {
    var troll = document.getElementById("trol");
    troll.style.display = "none";
    clearInterval(frameTroll);
}
```