TP – Découvrir les événements

Vous allez pouvoir créer une page HTML avec du JavaScript afin de découvrir ce qu’il en retourne. Vous manipulerez les formulaires en JavaScript en gérant ce qui se passe lorsque l’on interagit avec. Vous verrez également ce qu’il est possible de faire.

Le code sera déposé dans le répertoire /JS/S5/TP.

# Créer les fichiers HTML et JS

Créez la page index.html contenant le code suivant.<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Mes Sincères Salutations</title>

<script src="salutations.js" defer></script>

<script src="execution.js" defer></script>

</head>

<body>

<form>

<fieldset>

<legend>Identité</legend>

<input type="text" name="prenom" id="prenom" placeholder="saisissez votre prénom">

<input type="text" name="nom" id="nom" placeholder="saisissez votre nom">

<input type="text" name="titre" id="titre" placeholder="saisissez votre titre / surnom">

<select name="race" id="race">

<option value="human" selected>Humain(e)</option>

<option value="elf">Elfe</option>

<option value="ork">Orque</option>

</select>

<input type="radio" name="rank" id="roturier" value="roturier" checked

>roture

<input type="radio" name="rank" id="noble" value="noble"> noblesse

<input type="radio" name="rank" id="royal" value="royal"> Royauté

</fieldset>

<fieldset>

<legend>Classe(s) du personnage</legend>

<input type="checkbox" name="priest" id="priest" class="classe"> Prêtre

<input type="checkbox" name="warrior" id="warrior" class="classe"> Guerrier

<input type="checkbox" name="rogue" id="rogue" class="classe"> > Voleur

<input type="checkbox" name="mage" id="mage"> class="classe"> Mage

</fieldset>

<button id="btn\_salutation">Dit bonjour</button>

</form>

</body>

</html>

Ouvrez ensuite cette page dans votre navigateur et observez ce qui s’affiche.

<fieldset> - Crée un cadre qui entoure une série de champs et éléments de formulaire.

<legend> - Crée un texte qui donne un intitulé au fieldset parent.

<input type="text" – Crée un champ de saisie permettant de saisir du texte. L’attribut placeholder permet de définir un texte explicatif dans le formulaire.

<select – Permet de créer une liste déroulante d’option, une seule pouvant être choisie.

<option – Défini chacune des options présentées dans la liste déroulante, le mot-clé « selected » permet de forcer la sélection d’une option : par défaut la première de la liste est sélectionnée. L’attribut « value » permettra de déterminer l’option choisie.

<input type="radio" – Permet de créer une liste d’items sélectionnable, présentés les uns à côté des autres, sachant qu’un seul peut être sélectionné : l’attribut « name » permet de définir les différents groupes de bouton-radios ainsi liés. L’attribut « value » permettra de déterminer l’option choisie.

<input type="checkbox"" – Permet de créer une liste d’items sélectionnable, présentés les uns à côté des autres, sachant que tous peuvent être sélectionné : L’attribut « value » permettra de déterminer les options choisies, chaque checkbox ayant un identifiant unique. Attention qu’il n’y a pas de valeur si non sélectionné.

<button – Un bouton cliquable. S’il est inclus dans entre deux balises <FORM></FORM> il envoie le formulaire par défaut (ce qui recharge la page ou charge une autre page).

# Créer notre premier événement

Nous allons annuler l’envoi du formulaire lors du clic sur le bouton et faire une « pop-up » avec ALERT lors du clic.

Il nous faut une fonction saluer(), qui va être ajoutée dans le fichier salutations.js

function saluer(event)

{

// bloquer l'événement par défaut, on peut récupérer les informations de l'événement

// grâce au paramètre event qui est rempli automatiquement par le listener.

event.preventDefault();

alert("AH UN CLICK SUR LE BOUTON !");

}

Il ne se passe toujours rien lors du click ? Normal, il faut abonner le bouton à la fonction saluer() quand il y a click.

Dans le fichier execution.js

let bouton = document.querySelector('#btn\_salutation');

bouton.addEventListener('click',saluer);

**Maintenant, vérifions !**

* **Le clic sur le bouton « Dit bonjour » fait apparaître un message en pop-up : « AH UN CLICK SUR LE BOUTON ! »**

# Saluer avec ou sans le prénom et nom du personnage

Nous allons enrichir le texte du message avec le prénom et nom du personnage. Bien entendu, on se doute que l’on peut cliquer sur le bouton SANS avoir rempli ces informations ! Il va falloir accéder à la valeur et vérifier si cela est rempli !

Dans le fichier salutations.js

function saluer(event)

{

// bloquer l'événement par défaut, on peut récupérer les informations de l'événement

// grâce au paramètre event qui est rempli automatiquement par le listener.

event.preventDefault();

// récupération de la valeur des champs de saisie.

let prenom = document.querySelector('#prenom').value;

let nom = document.querySelector('#nom').value;

// construction du message

let message = `Bonjour ${prenom} ${nom}`;

//alert("AH UN CLICK SUR LE BOUTON !");

alert(message);

}

Nous ne vérifions pas s’il y a un nom ou prénom. Quoi faire si cela est manquant ?

Pour se faciliter la vie, s’il manque un des deux, ce n’est pas grave, le code affichera une chaîne de caractère vide. S’il manque les deux, nous allons mettre dans le message « illustre inconnu(e) ».

Dans le fichier salutations.js

function saluer(event)

{

// bloquer l'événement par défaut, on peut récupérer les informations de l'événement

// grâce au paramètre event qui est rempli automatiquement par le listener.

event.preventDefault();

// récupération de la valeur des champs de saisie.

let prenom = document.querySelector('#prenom').value;

let nom = document.querySelector('#nom').value;

// construction du message

let message="";

if (prenom.length==0 && nom.length==0)

message ="Bonjour, illustre inconnu(e)";

else

message = `Bonjour ${prenom} ${nom}`;

//alert("AH UN CLICK SUR LE BOUTON !");

alert(message);

}

**Pour vous exercer**: affichez également le titre à la suite du prénom entre guillemets.

**Maintenant, vérifions !**

* **Le clic sur le bouton « Dit bonjour » fait apparaître un message en pop-up : « Bonjour illustre inconnu(e) ! » si aucun prénom et nom n’ont été saisis**
* **Le clic sur le bouton « Dit bonjour » fait apparaître un message en pop-up : « Bonjour Matt Martigan ! » si les nom et prénom ont été saisis.**

# Afficher un texte spécifique suivant la race

Vous avez remarqué que l’on définissait la race du personnage. Peut-être que certain(e) d’entre vous se sentent les oreilles pointues ?

Suivant la race, nous allons définir un petit texte d’ambiance. Et oui, l’ordinateur sera probablement séduit par un ou une elfe, et carrément repoussé par l’apparence d’un ork !

Dans le fichier salutations.js nous allons créer une fonction par race.

function emoteHumain()

{

return "l'ordinateur observe la personne qui approche.";

}

function emoteElfe()

{

return "l'ordinateur observe la personne, totalement subjugé par la grâce elfique.";

}

function emoteOrque()

{

return "l'ordinateur observe la personne difforme et repoussante, plissant le nez.";

}

Oui, définitivement, cet ordinateur est très partial.

Maintenant, suivant la race identifiée, nous allons appeler l’une ou l’autre des fonctions. Cela pourrait être fait avec des IF/Else imbriqué, mais utilisons plutôt la structure SWITCH.

Dans le fichier salutations.js :

function saluer(event) {

…

alert(message);

let race = document.querySelector('#race').value;

let emote = "";

switch (race) {

case 'elf':

emote = emoteElfe();

break;

case 'ork':

emote = emoteOrque();

break;

default:

emote = emoteHumain();

break;

}

alert(emote);

}

**Maintenant, vérifions !**

* **Le clic sur le bouton « Dit bonjour » fait apparaître un message en pop-up avec les noms et prénom et ensuite un message suivant la race sélectionnée.**

# S’entrainer : affichez un texte spécifique suivant le rang social

Si le personnage est issu de la roture : « Le garde fait un signe de tête. »

Si le personnage est issu de la noblesse : « Le garde s’incline respectueusement. »

Si le garde est issu de la royauté : « Le garde, ainsi que toutes les personnes présentes s’inclinent profondément. »

Pour récupérer la valeur choisie :

let rang = document.querySelector('input[name=rank]:checked').value;

Et oui, on sélectionne l’input qui a pour nom « rank » et qui est selectionné !

Créez bien trois fonctions et utilisez également un switch. Ajoutez le résultat dans la variable « emote ».

**Maintenant, vérifions !**

* **Le clic sur le bouton « Dit bonjour » fait apparaître un message en pop-up avec les noms et prénom et ensuite un message suivant la race sélectionnée et le rang social.**

# Afficher les informations sur la classe du personnage

Un personnage peut avoir entre 1 et 3 classes maximum. Nous allons afficher le nombre de classes sélectionnées et leur liste sous forme d’une liste non ordonnée.

Nous ajouterons ultérieurement un test pour empêcher l’utilisateur de cliquer sur le bouton « Dit bonjour » s’il n’y a pas au moins une classe sélectionnée et au maximum trois.

Pour compter le nombre de classe, ajoutez la fonction suivante dans salutations.js :

function getNombreClasse()

{

let classes= document.querySelectorAll('.classe:checked');

return classes.length;

}

C’est là ou l’on voit le chouette du querySelectorAll : on peut véritablement ciblé de manière très efficace !

Pour retourner la liste des classes, on ajoute une fonction également : notre fonction saluer() commence à être drôlement longue ! Ca devient un brin môche !

Ajoutons listerLesClasses() dans salutations.js :

function listerLesClasses()

{

let classes= document.querySelectorAll('.classe:checked');

let message = "<ul>";

classes.forEach(classe => {

message+=`<li>${classe.name}</li>`;

});

return message;

}

Dans la fonction saluer(), ajoutez les lignes suivantes :

let paragrapheClasses = document.querySelector('#classes');

paragrapheClasses.innerHTML = listerLesClasses();

Dans le document HTML, ajoutez le div suivant :

<div id="classes"></div>

**Maintenant, vérifions !**

* **Le clic sur le bouton « Dit bonjour » fait apparaître un message en pop-up avec les noms et prénom et ensuite un message suivant la race sélectionnée.**
* **Sous forme d’un message alert, vous devez voir également le nombre des classes.**
* **Dans le document HTML, vous devez voir apparaître la liste des classes, si des classes ont été sélectionnées !**

# Mise au propre

Enlevons maintenant tous les affichages sous forme d’alertes et ajoutons des div pour mettre une belle mise en forme :

<div id="classes"></div>

<div id="error"></div>

<div id="message">

<div id="emotes"></div>

<div id="salutations"></div>

</div>

Pour faire disparaitre les alertes, créez une nouvelle fonction : saluerSansAlertes() et attribuez-la au clic. Cela permet de garder votre ancien code. On va également faire ce que l’on appelle du « refactoring ». Le but est que chaque fonction soit courte et claire, ce qui n’est pas le cas de saluer() !

Visual Code nous aide !

Sélectionnez le code et click droit pour avoir le menu contextuel

Une image contenant texte, capture d’écran, moniteur, écran

Description générée automatiquement

Ensuite choisissez l’option « extraire vers function dans la portée globale et donnez lui un nom (genererMessageSalutation).

Une image contenant texte

Description générée automatiquement

Et voila !

De la même manière générez les fonctions genererEmoteRaciale et genererEmoteSociale. Attention que le fait que les deux manipulent la variable « emote », il faut corriger un peu. Créez pour le coup deux variables : emoteRaciale et emoteSociale.

Il n’y a plus qu’à viser les paragraphes et les mettre à jour.

Voici ce que doit donner votre fonction saluerSansAlertes()

function saluerSansAlertes(event) {

// bloquer l'événement par défaut, on peut récupérer les informations de l'événement

// grâce au paramètre event qui est rempli automatiquement par le listener.

event.preventDefault();

// génération du message et des emotes

let message = genererMessageSalutation();

let emoteRaciale = genererEmoteRaciale();

let emoteSociale = genererEmoteSociale();

let classes = listerLesClasses();

document.querySelector('#emotes').innerHTML=`<ul><li>${emoteRaciale}</li><li>${emoteSociale}</li></ul>`;

document.querySelector('#salutations').innerHTML= message;

document.querySelector('#classes').innerHTML = classes;

}

Voici les autres fonctions

function genererEmoteSociale() {

let rang = document.querySelector('input[name=rank]:checked').value;

let emote = "";

switch (rang) {

case 'royal':

emote = emoteRoyaute();

break;

case 'noble':

emote = emoteNoblesse();

break;

default:

emote = emoteRoture();

break;

}

return emote;

}

function genererEmoteRaciale() {

let race = document.querySelector('#race').value;

let emote = "";

switch (race) {

case 'elf':

emote = emoteElfe();break;

case 'ork':

emote = emoteOrque();break;

default:

emote = emoteHumain();break;

}

return emote;

}

function genererMessageSalutation() {

let prenom = document.querySelector('#prenom').value;

let nom = document.querySelector('#nom').value;

// construction du message

let message = "";

if (prenom.length == 0 && nom.length == 0)

message = "Bonjour, illustre inconnu(e)";

else

message = `Bonjour ${prenom} ${nom}`;

return message;

}

Tout est davantage joli n’est-ce pas ?

Vous avez remarqué le div « message » ? On va le cacher par défaut : il ne sera visible QUE s’il y a entre une et trois classes sélectionnées…

# N’afficher le message que s’il y a au moins une classe !

Ajoutez un fichier css et incluez le code suivant dedans :

#message

{

display: none;

}

Par défaut, notre message est caché !

Il nous faut ajouter la gestion du message d’erreur et le test en fonction du nombre de classe.

function saluerSansAlertes(event) {

// bloquer l'événement par défaut, on peut récupérer les informations de l'événement

// grace au paramère event qui est rempli automatiquement par le listener.

event.preventDefault();

// on affiche tout par défaut, et il n'y a pas d'erreur !

document.querySelector('#error').innerHTML="";

document.querySelector('#message').style.display='block';

// vérifier les trois classes

// test de garde !

if ((getNombreClasse()==0)||getNombreClasse()>3)

{

document.querySelector('#message').style.display='none';

document.querySelector('#classes').innerHTML="";

document.querySelector('#error').innerHTML = `Vous ne pouvez pas avoir ${getNombreClasse()} classe(s): entre 1 et 3 maximum.`;

return;

}

// génération du message et des emotes

let message = genererMessageSalutation();

let emoteRaciale = genererEmoteRaciale();

let emoteSociale = genererEmoteSociale();

let classes = listerLesClasses();

document.querySelector('#emotes').innerHTML=`<ul><li>${emoteRaciale}</li><li>${emoteSociale}</li></ul>`;

document.querySelector('#salutations').innerHTML= message;

document.querySelector('#classes').innerHTML = classes;

}

Il faut noter le « return » dans le IF : cela met fin à l’exécution de la fonction saluerSansAlertes().

Et voilà ! Maintenant, vous pouvez vous faire plaisir et rendre cette page davantage jolie !