Laboratoire: JQuery ET AJAX

# Objectif(s)

* Jquery :
  + Événements
  + Class
  + Ajax

# Créer la branche de la fonctionnalité

Clonage du repository

1. Dans Gitkraken, clonez le repository
   * Soit via URL
   * Soit via GitHub.com
2. Ouvrez le repository (OK)

Création d’une branche pour la fonctionnalité

1. Créez une branche appelée **FCT \_JQuery**
2. Assurez-vous d'être dans la branche que vous avez créée dans LOCAL

Dans ce laboratoire, vous aurez a décider vous-mêmes quand faire les *Comment, commit, push.*

# Ajouter de l’animation

## Weapons/index

L’objectif est de surligner le *Weapon* sur lequel la souris passe dans la vue Weapon/Index.

Une image contenant texte, capture d’écran, logiciel, Page web

Description générée automatiquement

1. Ouvrez le fichier Root/CSS/site.css
2. Ajoutez une classe CSS pour le surlignage appelée highlight :
   * Couleur de fond : DarkViolet
   * Opacity : 0.5
   * Couleur du text : white
3. Essayez par vous-mêmes en vous rappelant vos notions de web serveur, sinon, utilisez le code ci-dessous.

/\* Surligner le weapon dans l'index\*/

.highlight {

background: DarkViolet;

opacity: 0.5;

color: white;

}

1. Ouvrez la vue **Weapon/Index**
2. Ajoutez un **script** qui ajoute la classe **highlight** quand la souris passe dessus le **<tr>** ayant la classeWeaponList et l’enlève quand la souris sort.
3. Ajoutez la fonction principale document ready en jQuery

<script>

$(document).ready(function () {

});

</script>

1. Il y a plusieurs façons de réaliser l’objectif. Ajoutez la fonction pour attacher l’événement hover qui ajoute et enlève la classe créée précédement.

<script>

$(document).ready(function () {

$('tr').hover(function(){

$(this).toggleClass("highlight");

});

});

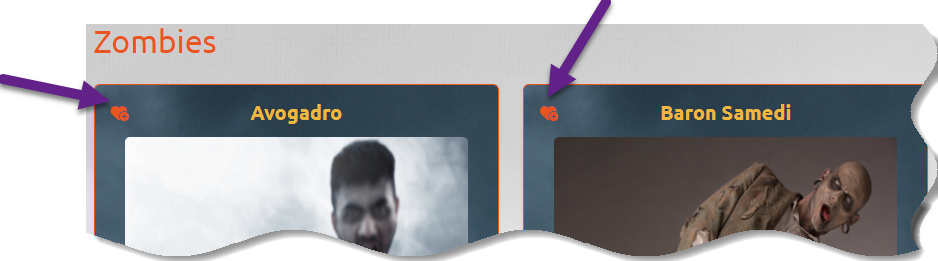
</script>

# Ajouter des call ajax

## Ajouter un favori vue partielle ZombieCard

L’objectif est d’identifier un Zombie favori en cliquant sur une icône

1. Ouvrez la classe **Zombie**
2. Ajoutez une propriété bool appelé IsFavorite
3. Ouvrez la vue partielle **\_ZombieCard**
4. Ajoutez lien identifié par une icône Font Awesome orange (text-primary),ayant la classe **Favorite**

****

<div class="col-lg-4 col-md-6 pb-4">

<div class="card border-primary rounded shadow-sm" style="background: url('/images/imagesLayout/fond.png');">

<div class="card-body px-3 pt-3 pb-1 row">

<div class="position-absolute">

<**a** class="Favorite text-primary"><i class="fa-solid fa-heart-circle-plus"></i></**a**>

</div>

<p class="card-title text-center text-warning h5"><b>@Model.Name</b></p>

</div>

1. Modifiez le code afin que la classe text-dark soit appliquée si IsFavorite est false et que ce soit text-primary si la valeur de IsFavorite est true

<a class="Favorite @(Model.IsFavorite ? "text-primary": "text-dark")"><i class="fa-solid fa-heart-circle-plus"></i></a>

1. Ajoutez le code nécessaire qui met à jour le Zombie en favori via un call Ajax afin de ne mettre à jour que la vue partielle. Regardez les notes de cours pour trouver comment réaliser cette étape. Les grandes lignes :
   1. Ajouter un attribut avec la valeur de l’id du zombie à l’élément <**a>** qui contient une class **Favorite**
   2. Ajouter une action **Favorite** à votre contrôleur ZombieController qui va modifier l’état de IsFavorite d’un zombie et retourner un **JsonResult**
   3. Écrire un **script** qui fait un appel **AJAX** à l’action que vous venez d’ajouter lorsque l’on clique sur un élément qui contient la class **Favorite.** Utilisez l’attribut que vous avez ajouté pour fournir l’Id du zombie sur lequel l’utilisateur a cliqué.

## Ajouter une vue modale sur Zombie/DEtails

L’objectif est d’afficher une vue partielle modale en cliquant sur le bouton Details de la vue \_**ZombieCard**

1. Créez une vue partielle Affichant le détail d’un zombie **\_ZombieDetails**
   * La vue partielle doit afficher toutes les informations sauf l’image et IsFavorite.

@model Zombie

<div>

<h4>Zombie</h4>

<hr />

<dl class="row">

<dt class = "col-sm-2">

@Html.DisplayNameFor(model => model.Name)

</dt>

<dd class = "col-sm-10">

@Html.DisplayFor(model => model.Name)

</dd>

<dt class = "col-sm-2">

@Html.DisplayNameFor(model => model.ShortDesc)

</dt>

<dd class = "col-sm-10">

@Html.DisplayFor(model => model.ShortDesc)

</dd>

<dt class = "col-sm-2">

@Html.DisplayNameFor(model => model.Point)

</dt>

<dd class = "col-sm-10">

@Html.DisplayFor(model => model.Point)

</dd>

<dt class = "col-sm-2">

@Html.DisplayNameFor(model => model.Force)

</dt>

<dd class = "col-sm-10">

@Html.DisplayFor(model => model.Force)

</dd>

<dt class = "col-sm-2">

@Html.DisplayNameFor(model => model.ZombieType)

</dt>

<dd class = "col-sm-10">

@Html.DisplayFor(model => model.ZombieType.TypeName)

</dd>

</dl>

</div>

1. Ouvrez la vue \_**ZombieCard**
2. Ajoutez le code nécessaire afin d’appeler la vue partielle affichant des détails pour le zombie concerné en modal. Utilisez un call Ajax pour afficher l’information.
3. Essayez par vous-mêmes en consultant le PowerPoint et le laboratoire précédent. Les grandes lignes :
   * Ajouter une action Details qui dans **ZombieController** qui retoune la vue partielle \_**ZombieDetails** d’un zombie (N’oubliez pas qu’il est possible de la tester)
   * Ajouter une class **detailBtn** sur le bouton **details** de \_**ZombieCard** pour nous permettre de facilement écrire notre **jQuery**
   * Ajouter le div modal dans la vue Zombie/Index
   * Ajouter un script pour faire un appel AJAX lorsque l’on clique sur un bouton détails et afficher le contenu retourné dans le modal

## Commentaires et validation (Commit) des changements dans le code

1. Faites un dernier **Commit**
2. "Poussez" **Push** les modifications sur le repo en ligne **Remote**: cliquez sur 
3. Merge la branche **FCT \_ JQuery** dans la branche **Main** et faites un **dernier push.**

**Fin du laboratoire!**