Cours № 1 : JQuery

Pr. Mohamed LACHGAR, ENSA - UCD 10/02/2023

1 Séance 1

1.1 Introduction à JQuery

1.1.1 Qu'est-ce que jQuery et pourquoi l'utiliser ?

jQuery est une bibliothèque JavaScript gratuite, open-source et très populaire qui permet de simplifier la création d'applications web interactives. Elle permet de manipuler facilement le contenu HTML, de réagir aux événements de l'utilisateur, de créer des animations, d'envoyer des requêtes Ajax pour récupérer des données du serveur et bien plus encore. jQuery est largement utilisé dans le développement web car il permet d'écrire du code JavaScript beaucoup plus rapidement et efficacement que si l'on devait utiliser du code pur JavaScript.

Il existe plusieurs raisons pour lesquelles on peut utiliser jQuery dans un projet web. Tout d'abord, jQuery permet de simplifier la syntaxe JavaScript et de rendre le code plus facile à comprendre et à maintenir. En outre, jQuery est compatible avec de nombreux navigateurs différents, ce qui facilite le développement d'applications web multiplateformes. Enfin, jQuery dispose d'une grande communauté de développeurs qui ont créé de nombreuses extensions et plugins qui peuvent être utilisés pour étendre les fonctionnalités de jQuery et simplifier encore plus le développement web.

1.1.2 Les différentes versions de jQuery

Il y a plusieurs versions de jQuery, chaque version étant une amélioration de la précédente, avec de nouvelles fonctionnalités et des corrections de bugs. Voici un aperçu des principales versions de jQuery :

- jQuery 1.x : Cette version est considérée comme la plus ancienne, mais elle est encore largement utilisée dans de nombreux projets. La dernière version de la série 1.x est jQuery 1.12.4. Cette version prend en charge les anciens navigateurs tels que Internet Explorer 6, 7 et 8.
- jQuery 2.x : Cette version est une mise à jour de jQuery 1.x et elle a été introduite pour améliorer les performances de jQuery. Elle abandonne le support des anciens navigateurs et ne fonctionne qu'avec les navigateurs modernes. La dernière version de la série 2.x est jQuery 2.2.4.
- jQuery 3.x : Cette version est la plus récente et elle est compatible avec tous les navigateurs modernes. Elle prend également en charge les dernières normes web telles que HTML5 et CSS3. La dernière version de la série 3.x est jQuery 3.6.0.

Il est important de noter que certaines fonctions de jQuery ont été supprimées dans la série 3.x, car elles étaient obsolètes ou peu utilisées. Il est donc important de vérifier la documentation de chaque version de jQuery pour s'assurer que les fonctions nécessaires sont prises en charge.

1.1.3 Téléchargement et intégration de jQuery dans une page web

Pour télécharger et intégrer jQuery dans une page web, suivez les étapes suivantes :

1. Allez sur le site officiel de jQuery à l'adresse suivante : https://jquery.com/
2. Cliquez sur le bouton "Download" pour télécharger le fichier compressé contenant la dernière version de jQuery.
3. Extrayez le fichier .zip téléchargé et copiez le fichier jquery.js dans le dossier de votre projet.
4. Ajoutez une balise <script> dans l'en-tête de votre page web, entre les balises <head> et </head>.
5. Spécifiez l'emplacement du fichier jquery.js en utilisant l'attribut src de la balise <script>, comme suit :

```
1     <head>
2         <script src="chemin/vers/jquery.js"></script>
3     </head>
```

Une fois que jQuery est intégré dans votre page web, vous pouvez utiliser la bibliothèque en appelant la fonction jQuery() ou sa forme courte $() pour sélectionner des éléments HTML et appliquer des manipulations sur ces éléments. Par exemple, vous pouvez utiliser la méthode .css() pour modifier les styles CSS d'un élément HTML sélectionné, comme suit :

```
1     <!DOCTYPE html>
2     <html>
3     <head>
4         <script src="chemin/vers/jquery.js"></script>
5         <script>
6             $(document).ready(function(){
7                 $("p").css("color", "red");
8             });
9         </script>
10    </head>
11    <body>
12        <p>Ceci est un paragraphe.</p>
13        <p>Ceci est un autre paragraphe.</p>
14    </body>
```

```
15      </html>
```

Dans cet exemple, la méthode .css() est utilisée pour changer la couleur du texte de tous les éléments <p> de la page en rouge. La fonction $(document).ready() est utilisée pour s'assurer que le code JavaScript ne s'exécute qu'après que la page a fini de se charger.

1.2 Sélection d'éléments avec jQuery

1.2.1 Les sélecteurs jQuery

Les sélecteurs jQuery sont utilisés pour sélectionner des éléments HTML dans une page web, afin de pouvoir les manipuler à l'aide de fonctions jQuery. Voici les sélecteurs les plus couramment utilisés en jQuery :

- Selecteurs de balises : sélectionne tous les éléments HTML qui correspondent à une balise spécifique. Par exemple, le sélecteur "p" sélectionne tous les éléments <p> de la page.
- Sélecteurs d'ID : sélectionne un élément HTML qui a un attribut id spécifique. Par exemple, le sélecteur "monID" sélectionne l'élément avec l'attribut id="monID".
- Sélecteurs de classe : sélectionne tous les éléments HTML qui ont une classe spécifique. Par exemple, le sélecteur ".maClasse" sélectionne tous les éléments qui ont la classe "maClasse".
- Sélecteurs d'attribut : sélectionne tous les éléments HTML qui ont un attribut spécifique. Par exemple, le sélecteur "[type='text']" sélectionne tous les éléments qui ont un attribut "type" égal à "text".
- Sélecteurs de relation : sélectionne des éléments HTML en fonction de leur relation avec d'autres éléments. Par exemple, le sélecteur "p > a" sélectionne tous les éléments <a> qui sont des enfants directs d'éléments <p>.
- Sélecteurs de filtre : sélectionne des éléments HTML en fonction de certains critères. Par exemple, le sélecteur ":first" sélectionne le premier élément HTML correspondant au sélecteur.

Il est également possible de combiner plusieurs sélecteurs pour créer des sélecteurs plus spécifiques. Par exemple, le sélecteur "ul li" sélectionne tous les éléments <li> qui sont des enfants directs d'éléments <ul>.

Il est important de noter que les sélecteurs jQuery sont très similaires aux sélecteurs CSS. Si vous êtes familier avec les sélecteurs CSS, vous trouverez donc facile de comprendre et d'utiliser les sélecteurs jQuery.

1.2.2 La méthode $() de jQuery

La méthode $() est une fonction jQuery qui permet de sélectionner des éléments HTML dans une page web et de les manipuler à l'aide de fonctions jQuery. La syntaxe de la méthode $() est la suivante :

```
1    $(sélecteur)
```

Le sélecteur passé en argument est utilisé pour sélectionner les éléments HTML de la page. Par exemple, le sélecteur "p" sélectionne tous les éléments <p> de la page. Le résultat de la sélection est un objet jQuery, qui contient une liste d'éléments HTML correspondant au sélecteur.

Vous pouvez ensuite utiliser différentes fonctions jQuery pour manipuler les éléments sélectionnés. Par exemple, la méthode .text() peut être utilisée pour récupérer ou modifier le contenu texte d'un élément HTML. Voici un exemple :

```
1    <!DOCTYPE html>
2    <html>
3    <head>
4        <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script↩
             >
5        <script>
6            $(document).ready(function(){
7                $("p").text("Bonjour jQuery !");
8            });
9        </script>
10   </head>
11   <body>
12
13       <p>Ceci est un paragraphe.</p>
14       <p>Ceci est un autre paragraphe.</p>
15
16   </body>
17   </html>
```

Dans cet exemple, la méthode $() est utilisée pour sélectionner tous les éléments <p> de la page, puis la méthode .text() est utilisée pour modifier le contenu texte des éléments sélectionnés en "Bonjour jQuery !". Le résultat sera que les deux éléments <p> de la page afficheront le texte "Bonjour jQuery !".

Il est important de noter que la méthode $() est la méthode principale de jQuery, mais elle peut également être appelée en utilisant la syntaxe jQuery(). Par exemple, les deux lignes de code suivantes sont équivalentes :

```
1    $(sélecteur)
```

```
jQuery(sélecteur)
```

1.2.3 La manipulation de CSS avec jQuery

jQuery permet de manipuler les styles CSS des éléments HTML sélectionnés. Pour cela, il est possible d'utiliser les méthodes .css() et .addClass() de jQuery.

La méthode .css() permet de récupérer ou de modifier une propriété CSS spécifique d'un élément HTML. La syntaxe de la méthode est la suivante :

```
$(élément).css(nom_de_la_propriété)
```

Ou

```
$(élément).css(nom_de_la_propriété, valeur_de_la_propriété)
```

Par exemple, pour récupérer la valeur de la propriété "background-color" d'un élément HTML, vous pouvez utiliser la méthode .css() comme ceci :

```
var bgColor = $("div").css("background-color");
```

Pour modifier la valeur de cette propriété, vous pouvez utiliser la même méthode avec une valeur différente :

```
$("div").css("background-color", "red");
```

La méthode .addClass() permet d'ajouter une ou plusieurs classes CSS à un élément HTML. La syntaxe de la méthode est la suivante :

```
$(élément).addClass(nom_de_la_classe)
```

Ou

```
$(élément).addClass(nom_de_la_classe_1 nom_de_la_classe_2 ...)
```

Par exemple, pour ajouter la classe "ma-classe" à un élément HTML, vous pouvez utiliser la méthode .addClass() comme ceci :

```
$("div").addClass("ma-classe");
```

Il est également possible de combiner les méthodes .css() et .addClass() pour ajouter une classe

CSS avec un style particulier à un élément HTML. Par exemple, pour ajouter la classe "fond-rouge" à un élément HTML avec un fond rouge, vous pouvez utiliser les méthodes comme ceci :

```
1    $("div").addClass("fond-rouge").css("background-color", "red");
```

Ainsi, la combinaison de ces deux méthodes permet de modifier les styles CSS des éléments HTML sélectionnés et de créer des animations ou des effets visuels dynamiques sur une page web.

1.3 Manipulation d'éléments avec jQuery

1.3.1 Les méthodes .html(), .text() et .val()

Les méthodes .html(), .text() et .val() sont des méthodes de jQuery qui permettent de récupérer ou de modifier le contenu des éléments HTML sélectionnés.

La méthode .html() permet de récupérer ou de modifier le contenu HTML d'un élément. La syntaxe de la méthode est la suivante :

```
1    $(élément).html()
```

Ou

```
1    $(élément).html(nouveau_contenu_html)
```

Par exemple, pour récupérer le contenu HTML d'un élément HTML, vous pouvez utiliser la méthode .html() comme ceci :

```
1    var contenuHTML = $("div").html();
```

Pour modifier le contenu HTML de cet élément, vous pouvez utiliser la même méthode avec une chaîne de caractères contenant le nouveau contenu HTML :

```
1    $("div").html("<p>Mon nouveau contenu HTML</p>");
```

La méthode .text() permet de récupérer ou de modifier le contenu texte d'un élément. La syntaxe de la méthode est la suivante :

```
1    $(élément).text()
```

Ou

```
$(élément).text(nouveau_contenu_texte)
```

Par exemple, pour récupérer le contenu texte d'un élément HTML, vous pouvez utiliser la méthode .text() comme ceci :

```
var contenuTexte = $("div").text();
```

Pour modifier le contenu texte de cet élément, vous pouvez utiliser la même méthode avec une chaîne de caractères contenant le nouveau contenu texte :

```
$("div").text("Mon nouveau contenu texte");
```

La méthode .val() permet de récupérer ou de modifier la valeur d'un champ de formulaire, comme un champ de texte ou une case à cocher. La syntaxe de la méthode est la suivante :

```
$(élément).val()
```

Ou

```
$(élément).val(nouvelle_valeur)
```

Par exemple, pour récupérer la valeur d'un champ de texte, vous pouvez utiliser la méthode .val() comme ceci :

```
var valeurChampTexte = $("input[type=text]").val();
```

Pour modifier la valeur de ce champ, vous pouvez utiliser la même méthode avec une chaîne de caractères contenant la nouvelle valeur :

```
$("input[type=text]").val("Nouvelle valeur du champ");
```

Ces méthodes sont très utiles pour manipuler le contenu des éléments HTML sélectionnés et pour créer des formulaires interactifs sur une page web.

1.3.2 La manipulation d'attributs avec jQuery

jQuery permet de manipuler les attributs des éléments HTML sélectionnés, en utilisant les méthodes .attr(), .removeAttr() et .prop().

La méthode .attr() permet de récupérer ou de modifier la valeur d'un attribut HTML d'un élément. La syntaxe de la méthode est la suivante :

```
$(élément).attr(nom_de_l_attribut)
```

Ou

```
$(élément).attr(nom_de_l_attribut, nouvelle_valeur)
```

Par exemple, pour récupérer la valeur de l'attribut "src" d'une image, vous pouvez utiliser la méthode .attr() comme ceci :

```
var srcImage = $("img").attr("src");
```

Pour modifier la valeur de cet attribut, vous pouvez utiliser la même méthode avec une chaîne de caractères contenant la nouvelle valeur :

```
$("img").attr("src", "nouvelle_image.png");
```

La méthode .removeAttr() permet de supprimer un attribut HTML d'un élément. La syntaxe de la méthode est la suivante :

```
$(élément).removeAttr(nom_de_l_attribut)
```

Par exemple, pour supprimer l'attribut "alt" d'une image, vous pouvez utiliser la méthode .removeAttr() comme ceci :

```
$("img").removeAttr("alt");
```

La méthode .prop() permet de récupérer ou de modifier la valeur d'une propriété HTML d'un élément, comme par exemple la propriété "checked" pour une case à cocher. La syntaxe de la méthode est la suivante :

```
$(élément).prop(nom_de_la_propriete)
```

Ou

```
$(élément).prop(nom_de_la_propriete, nouvelle_valeur)
```

Par exemple, pour récupérer la valeur de la propriété "checked" d'une case à cocher, vous pouvez utiliser la méthode .prop() comme ceci :

```
var caseCochee = $("input[type=checkbox]").prop("checked");
```

Pour modifier la valeur de cette propriété, vous pouvez utiliser la même méthode avec une valeur booléenne :

```
1    $("input[type=checkbox]").prop("checked", true);
```

Ces méthodes permettent de manipuler les attributs et les propriétés des éléments HTML sélectionnés avec jQuery, et de créer des interactions dynamiques sur une page web.

1.3.3 L'ajout et la suppression d'éléments avec jQuery

jQuery permet d'ajouter et de supprimer des éléments HTML de manière dynamique, en utilisant les méthodes .append(), .prepend(), .after(), .before() et .remove().

La méthode .append() permet d'ajouter un élément HTML en tant que dernier enfant d'un élément sélectionné. La syntaxe de la méthode est la suivante :

```
1    $(élément_parent).append(nouvel_element)
```

où nouvel_element peut être un élément HTML, un texte, ou une chaîne de caractères contenant du code HTML.

Par exemple, pour ajouter une nouvelle balise <p> à la fin d'un élément , vous pouvez utiliser la méthode .append() comme ceci :

```
1    $("div").append("<p>Contenu de la nouvelle balise</p>");
```

La méthode .prepend() fonctionne de manière similaire, mais ajoute l'élément en tant que premier enfant de l'élément sélectionné.

La méthode .after() permet d'ajouter un élément HTML juste après l'élément sélectionné, tandis que la méthode .before() ajoute l'élément juste avant l'élément sélectionné.

La syntaxe de ces deux méthodes est la suivante :

```
1    $(élément_selectionné).after(nouvel_element)
2    $(élément_selectionné).before(nouvel_element)
```

La méthode .remove() permet de supprimer un élément HTML sélectionné, ainsi que tous ses enfants. La syntaxe de la méthode est la suivante :

```
1    $(élément_selectionné).remove()
```

Par exemple, pour supprimer une balise <p> avec la classe "ma_classe", vous pouvez utiliser la méthode .remove() comme ceci :

```
1    $("p.ma_classe").remove();
```

Ces méthodes permettent de créer et de supprimer des éléments HTML de manière dynamique, en réponse à des interactions utilisateur ou à des événements sur une page web.

1.4 TP : Création d'une page web simple avec utilisation de sélecteurs jQuery et manipulation de l'HTML et des CSS

Dans ce TP, nous allons créer une page web simple avec HTML, CSS et jQuery. Nous allons utiliser des sélecteurs jQuery pour cibler des éléments spécifiques de la page, et manipuler l'HTML et les CSS avec jQuery pour créer des interactions dynamiques.

- Partie 1 : Création de la page web
    1. Créez un fichier HTML vide et ajoutez les balises de base (doctype, html, head, body).
    2. Dans la balise head, ajoutez un lien vers le fichier CSS externe que nous allons créer.
    3. Dans la balise body, ajoutez un en-tête (balise h1) avec un titre pour la page, et un paragraphe (balise p) avec une courte description.
    4. Ajoutez un formulaire avec un champ de texte et un bouton de soumission.
    5. Ajoutez une liste à puces (balise ul) avec au moins 3 éléments de liste.
- Partie 2 : Création du fichier CSS
    1. Créez un fichier CSS externe vide et ajoutez-le au fichier HTML.
    2. Ajoutez des règles CSS pour modifier l'apparence des éléments HTML de base (corps de la page, en-tête, paragraphe, liste à puces, formulaire)
- Partie 3 : Utilisation des sélecteurs jQuery pour cibler des éléments spécifiques
    1. Ajoutez un lien vers le fichier jQuery CDN dans la balise head du fichier HTML.
    2. Utilisez la méthode .css() de jQuery pour modifier le style CSS de l'en-tête (balise h1) et du paragraphe (balise p).
    3. Utilisez la méthode .hide() de jQuery pour cacher le formulaire lorsqu'il est soumis.
    4. Utilisez la méthode .click() de jQuery pour ajouter une interaction au bouton de soumission du formulaire. Lorsque le bouton est cliqué, affichez une alerte avec le texte du champ de texte.
- Partie 4 : Manipulation de l'HTML avec jQuery
    1. Utilisez la méthode .append() de jQuery pour ajouter un nouvel élément à la liste à puces (balise ul). Le nouvel élément doit contenir un lien vers une page web externe de votre choix.
    2. Utilisez la méthode .text() de jQuery pour modifier le texte de l'en-tête (balise h1) et du paragraphe (balise p).

3. Utilisez la méthode .attr() de jQuery pour modifier l'attribut "href" du lien que vous avez ajouté à la liste à puces (balise a).

- Partie 5 : Manipulation de l'CSS avec jQuery

1. Utilisez la méthode .addClass() de jQuery pour ajouter une classe CSS à l'en-tête (balise h1).
2. Utilisez la méthode .removeClass() de jQuery pour supprimer une classe CSS de l'en-tête (balise h1).
3. Utilisez la méthode .toggleClass() de jQuery pour alterner entre l'ajout et la suppression d'une classe CSS à l'en-tête (balise h1) lorsque le bouton de soumission du formulaire est cliqué.

Indications :

- Partie 1 : Création de la page web

```
<!DOCTYPE html>
    <html>
    <head>
      <title>Ma page web</title>
    </head>
    <body>
      <form>
        <input type="text" name="input-text">
        <button type="submit">Envoyer</button>
      </form>
      <ul>
        <li> l ément 1</li>
        <li> l ément 2</li>
        <li> l ément 3</li>
      </ul>
    </body>
    </html>
```

- Partie 2 : Création du fichier CSS

```
/* Utilisation des sélecteurs CSS pour cibler les éléments HTML ↩
   */
form {
  margin-bottom: 20px;
}

input[type="text"] {
```

```
7       width: 200px;
8       padding: 5px;
9     }
10
11    button {
12      background-color: blue;
13      color: white;
14      border: none;
15      padding: 10px;
16      cursor: pointer;
17    }
18
19    ul {
20      list-style: none;
21      margin: 0;
22      padding: 0;
23    }
24
25    li {
26      margin-bottom: 10px;
27    }
```

- Partie 3 : Utilisation des sélecteurs jQuery pour cibler des éléments spécifiques

```
1     $(document).ready(function() {
2       // Cibler l élément input de type texte dans le formulaire
3       $('form input[type="text"]').val('Saisissez votre texte ici');
4
5       // Cibler le premier élément de la liste à puces et ajouter une↩
            classe CSS
6       $('ul li:first').addClass('premier-element');
7
8       // Cibler tous les éléments de la liste à puces et changer leur↩
            couleur de texte
9       $('ul li').css('color', 'green');
10    });
```

- Partie 4 : Manipulation de l'HTML et des CSS avec jQuery

```
1     $(document).ready(function() {
2       // Ajouter un nouveau paragraphe apr s le formulaire
3       $('form').after('<p>Voici un nouveau paragraphe ajouté avec ↩
            jQuery !</p>');
4
5       // Modifier le texte du bouton de soumission du formulaire
```

```
6       $('form button').text('Cliquez ici pour envoyer');
7
8       // Modifier la couleur de fond du premier élément de la liste à↩
            puces
9       $('ul li:first').css('background-color', 'yellow');
10    });
```

- Partie 5 : Manipulation de l'CSS avec jQuery

```
1     $(document).ready(function() {
2       // Cacher la liste à puces au chargement de la page
3       $('ul').hide();
4
5       // Ajouter un événement "click" sur le bouton de soumission
6       $('button').click(function() {
7         // Afficher la liste à puces avec une animation de fondu
8         $('ul').fadeIn('slow');
9       });
10    });
```

Dans cet exemple, nous utilisons la méthode hide() de jQuery pour cacher la liste à puces au chargement de la page. Ensuite, nous ajoutons un événement "click" sur le bouton de soumission en utilisant la méthode click() de jQuery.

Dans la fonction de rappel de l'événement "click", nous utilisons la méthode fadeIn() de jQuery pour afficher la liste à puces avec une animation de fondu. La chaîne de caractères "slow" est passée comme argument pour définir la durée de l'animation.

2 Séance 2

2.1 Événements avec jQuery

2.1.1 Les événements courants tels que click, hover, submit

Les événements avec jQuery permettent de déclencher des actions en réponse à des actions de l'utilisateur sur une page web, comme un clic de souris ou une saisie de clavier.

Voici quelques exemples d'événements couramment utilisés avec jQuery :

- click: déclenché lorsqu'un élément est cliqué
- submit: déclenché lorsqu'un formulaire est soumis
- focus: déclenché lorsqu'un élément reçoit le focus
- blur: déclenché lorsqu'un élément perd le focus

- keyup: déclenché lorsqu'une touche de clavier est relâchée

Pour ajouter un événement à un élément HTML avec jQuery, vous pouvez utiliser la méthode on() de jQuery. Voici un exemple pour ajouter un événement "click" à un bouton :

```
1    $('button').on('click', function() {
2      // Code exécuté lorsque le bouton est cliqué
3    });
```

Dans cet exemple, nous utilisons la méthode on() pour ajouter un événement "click" à tous les boutons de la page. La fonction de rappel passée comme deuxième argument sera exécutée lorsque le bouton est cliqué.

Il est également possible de combiner plusieurs événements en utilisant un espace comme séparateur. Par exemple, pour ajouter un événement "focus" et un événement "blur" à un champ de saisie, vous pouvez utiliser le code suivant :

```
1    $('input').on('focus blur', function() {
2      // Code exécuté lorsque le champ de saisie re oit ou perd le focus
3    });
```

Enfin, il est possible d'arrêter la propagation d'un événement en utilisant la méthode stopPropagation() et d'empêcher le comportement par défaut d'un événement en utilisant la méthode preventDefault(). Par exemple, pour empêcher le formulaire de se soumettre et recharger la page lorsqu'un utilisateur clique sur le bouton de soumission, vous pouvez utiliser le code suivant :

```
1    $('form').on('submit', function(event) {
2      event.preventDefault(); // Empêcher le formulaire de se soumettre
3    });
```

2.1.2 La méthode .on() de jQuery

La méthode .on() est une méthode de liaison d'événement de jQuery qui permet de lier un ou plusieurs événements à un ou plusieurs sélecteurs, et de spécifier une fonction de rappel qui sera exécutée lorsque l'événement est déclenché. La méthode .on() est préférable aux méthodes .bind(), .live(), et .delegate() qui ont été dépréciées depuis jQuery 3.0.

Voici la syntaxe de base de la méthode .on() :

```
1    $(selector).on(event, childSelector, data, function)
```

- selector : le sélecteur qui cible les éléments auxquels les événements seront liés.

- event : le ou les événements à écouter, séparés par un espace.
- childSelector : (optionnel) le sélecteur qui cible les éléments enfants de selector qui déclencheront l'événement.
- data : (optionnel) des données à transmettre à la fonction de rappel sous forme d'objet.
- function : la fonction de rappel qui sera exécutée lorsque l'événement est déclenché.

Voici un exemple d'utilisation de la méthode .on() pour lier un événement "click" à tous les boutons de la page :

```
1    $(document).ready(function() {
2      $('button').on('click', function() {
3        // Code exécuté lorsque le bouton est cliqué
4      });
5    });
```

Dans cet exemple, nous utilisons la méthode .on() pour lier un événement "click" à tous les boutons de la page. La fonction de rappel passée comme deuxième argument sera exécutée lorsque le bouton est cliqué.

La méthode .on() peut également être utilisée pour lier plusieurs événements en même temps en séparant les événements par un espace. Par exemple, pour lier les événements "mouseenter" et "mouseleave" à tous les liens de la page, vous pouvez utiliser le code suivant :

```
1    $(document).ready(function() {
2      $('a').on('mouseenter mouseleave', function() {
3        // Code exécuté lorsque le lien est survolé
4      });
5    });
```

Enfin, la méthode .on() permet également d'utiliser des sélecteurs de descendant pour cibler des éléments enfants spécifiques. Par exemple, pour lier un événement "click" à tous les boutons situés dans un élément ayant la classe "conteneur", vous pouvez utiliser le code suivant :

```
1    $(document).ready(function() {
2      $('.conteneur').on('click', 'button', function() {
3        // Code exécuté lorsque le bouton est cliqué
4      });
5    });
```

Dans cet exemple, nous utilisons la méthode .on() avec un sélecteur de descendant pour lier un événement "click" à tous les boutons situés dans un élément ayant la classe "conteneur". La fonction de rappel sera exécutée uniquement lorsque le bouton est cliqué.

2.1.3 La délégation d'événements

La délégation d'événements est une technique qui permet de lier un événement à un élément parent qui existe déjà dans le DOM, plutôt qu'à des éléments enfants qui peuvent être ajoutés ou supprimés dynamiquement. Cette technique est utile lorsque vous travaillez avec des éléments générés dynamiquement ou lorsque vous avez un grand nombre d'éléments enfants sur lesquels vous devez lier un événement.

La méthode .on() de jQuery peut être utilisée pour mettre en place une délégation d'événements en spécifiant un sélecteur de descendant en deuxième argument. Voici la syntaxe de base de la délégation d'événements avec la méthode .on() :

```
1    $(parentSelector).on(event, childSelector, function)
```

- parentSelector : le sélecteur qui cible l'élément parent auquel les événements seront liés.
- event : le ou les événements à écouter, séparés par un espace.
- childSelector : le sélecteur qui cible les éléments enfants de parentSelector qui déclencheront l'événement.
- function : la fonction de rappel qui sera exécutée lorsque l'événement est déclenché.

Voici un exemple d'utilisation de la délégation d'événements pour lier un événement "click" à tous les éléments de liste de la page, même s'ils sont générés dynamiquement :

```
1    $(document).ready(function() {
2      $('ul').on('click', 'li', function() {
3        // Code exécuté lorsque l élément de liste est cliqué
4      });
5    });
```

Dans cet exemple, nous utilisons la méthode .on() avec un sélecteur de descendant pour lier un événement "click" à tous les éléments de liste qui sont des descendants de l'élément ul. La fonction de rappel passée en troisième argument sera exécutée lorsque l'élément de liste est cliqué, même s'il a été généré dynamiquement.

La délégation d'événements peut également être utilisée avec plusieurs événements en même temps, en spécifiant les événements séparés par un espace. Par exemple, pour lier les événements "mouseenter" et "mouseleave" à tous les liens de la page, même s'ils sont générés dynamiquement, vous pouvez utiliser le code suivant :

```
1    $(document).ready(function() {
2      $('body').on('mouseenter mouseleave', 'a', function() {
3        // Code exécuté lorsque le lien est survolé
4      });
```

```
5     });
```

Dans cet exemple, nous utilisons la méthode .on() avec un sélecteur de descendant pour lier les événements "mouseenter" et "mouseleave" à tous les liens qui sont des descendants de l'élément body. La fonction de rappel sera exécutée lorsque le lien est survolé, même s'il a été généré dynamiquement.

En utilisant la délégation d'événements, vous pouvez éviter d'avoir à lier des événements à chaque élément enfant individuellement, ce qui peut améliorer les performances et simplifier votre code.

2.2 Effets avec jQuery

2.2.1 Les effets de base tels que hide(), show() et toggle()

Les méthodes de base hide(), show() et toggle() sont des méthodes d'effets de base de jQuery qui permettent de masquer ou d'afficher des éléments sur une page web de manière animée.

Voici une description de chacune de ces méthodes :

- hide() : Cette méthode permet de masquer un élément de la page. Elle prend en charge plusieurs options, telles que la durée de l'animation, la fonction de rappel à exécuter une fois l'animation terminée et l'équation d'animation utilisée. Voici un exemple :

```
1     $(document).ready(function() {
2       $('button').click(function() {
3         $('#myDiv').hide('slow', function() {
4           alert('L\'élément est maintenant masqué !');
5         });
6       });
7     });
```

Dans cet exemple, lorsque l'utilisateur clique sur le bouton, l'élément myDiv est masqué avec une animation lente ('slow'). Une fois que l'animation est terminée, la fonction de rappel est exécutée et une boîte de dialogue est affichée avec le message "L'élément est maintenant masqué !".

- show() : Cette méthode permet de montrer un élément de la page qui a été masqué avec la méthode hide(). Elle prend en charge les mêmes options que la méthode hide(). Voici un exemple :

```
1     $(document).ready(function() {
2       $('button').click(function() {
3         $('#myDiv').show('fast', function() {
4           alert('L\'élément est maintenant affiché !');
5         });
```

```
6       });
7     });
```

Dans cet exemple, lorsque l'utilisateur clique sur le bouton, l'élément myDiv est affiché avec une animation rapide ('fast'). Une fois que l'animation est terminée, la fonction de rappel est exécutée et une boîte de dialogue est affichée avec le message "L'élément est maintenant affiché !".

- toggle() : Cette méthode permet de basculer entre les états masqué et affiché d'un élément de la page. Elle prend en charge les mêmes options que les méthodes hide() et show(). Voici un exemple :

```
1     $(document).ready(function() {
2       $('button').click(function() {
3         $('#myDiv').toggle(1000);
4       });
5     });
```

Dans cet exemple, lorsque l'utilisateur clique sur le bouton, l'élément myDiv est masqué s'il est actuellement affiché, et affiché s'il est actuellement masqué. L'animation prend une seconde (1000 millisecondes) pour se terminer.

En utilisant ces méthodes, vous pouvez facilement ajouter des effets animés à votre page web, ce qui peut rendre votre site plus attractif et plus engageant pour les utilisateurs.

2.2.2 Les effets d'animation avec animate()

Les effets d'animation avec la méthode animate() de jQuery permettent de modifier les propriétés CSS d'un élément HTML de manière progressive et fluide, plutôt que de manière immédiate. Vous pouvez utiliser cette méthode pour créer des transitions d'éléments, des effets de défilement, des animations de curseurs, etc. La méthode animate() prend en charge la plupart des propriétés CSS et peut être personnalisée pour répondre à vos besoins spécifiques.

Voici un exemple simple pour vous montrer comment utiliser la méthode animate() :

HTML :

```
1     <div id="box"></div>
```

CSS :

```
1     #box {
2       width: 100px;
3       height: 100px;
4       background-color: red;
5       position: relative;
```

```
6     }
```

JavaScript avec jQuery :

```
1     $(document).ready(function() {
2       $("#box").click(function() {
3         $(this).animate({
4           left: '250px',
5           opacity: '0.5',
6           height: '+=150px',
7           width: '+=150px'
8         }, 2000);
9       });
10    });
```

Dans cet exemple, nous avons utilisé la méthode animate() pour animer la boîte rouge créée en HTML et CSS. Lorsque l'utilisateur clique sur la boîte, la méthode animate() est déclenchée. Les propriétés CSS spécifiées dans l'objet passé à animate() sont modifiées progressivement sur une période de 2 secondes. Les propriétés sont les suivantes :

- left est modifié pour déplacer la boîte de sa position actuelle de 250 pixels vers la droite.
- opacity est modifié pour changer l'opacité de la boîte à 0,5.
- height et width sont modifiées pour agrandir la boîte de 150 pixels dans les deux dimensions.

Vous pouvez personnaliser la durée de l'animation en spécifiant un nombre de millisecondes comme deuxième argument de la méthode animate(). Dans cet exemple, la durée de l'animation est de 2 secondes, soit 2000 millisecondes.

Notez que les propriétés passées à la méthode animate() doivent être des propriétés CSS qui peuvent être animées. Par exemple, les propriétés telles que display ou visibility ne peuvent pas être animées avec animate().

2.2.3 Les effets de fondu avec fadeIn() et fadeOut()

Les fonctions fadeIn() et fadeOut() sont des fonctions de la bibliothèque jQuery qui permettent d'appliquer des effets de fondu à un élément HTML.

La fonction fadeIn() permet de faire apparaître progressivement un élément HTML en le faisant passer de l'opacité 0 à l'opacité 1. Cela crée un effet de fondu en entrée, où l'élément HTML apparaît progressivement sur la page.

La fonction fadeOut() fait l'inverse de fadeIn() en faisant disparaître progressivement un élément HTML en le faisant passer de l'opacité 1 à l'opacité 0. Cela crée un effet de fondu en sortie, où l'élément HTML disparaît progressivement de la page.

Ces effets de fondu sont souvent utilisés pour améliorer l'expérience utilisateur en créant des transitions plus fluides entre les éléments de la page. Par exemple, ils peuvent être utilisés pour faire apparaître des menus déroulants ou pour afficher des messages d'erreur de manière plus subtile.

Voici un exemple simple d'utilisation de fadeIn() et fadeOut() avec jQuery :

HTML :

```
<div id="monElement">Bonjour !</div>
<button id="boutonDisparition">Faire dispara tre</button>
<button id="boutonApparition">Faire appara tre</button>
```

JavaScript avec jQuery :

```
$(document).ready(function() {
  // Cacher l élément au chargement de la page
  $('#monElement').hide();

  // Faire appara tre l élément avec fadeIn()
  $('#boutonApparition').click(function() {
    $('#monElement').fadeIn();
  });

  // Faire dispara tre l élément avec fadeOut()
  $('#boutonDisparition').click(function() {
    $('#monElement').fadeOut();
  });
});
```

Dans cet exemple, le div avec l'id "monElement" est initialement caché grâce à la fonction hide(). Lorsque l'utilisateur clique sur le bouton "Faire apparaître", l'élément HTML est affiché avec un effet de fondu grâce à la fonction fadeIn(). De même, lorsque l'utilisateur clique sur le bouton "Faire disparaître", l'élément HTML est caché avec un effet de fondu grâce à la fonction fadeOut().

2.3 Utilisation de plugins jQuery

2.3.1 Les plugins jQuery couramment utilisés

jQuery est une bibliothèque JavaScript populaire qui facilite la manipulation du DOM et l'interaction avec les événements de la page web. En plus de ses fonctionnalités de base, il existe de nombreux plugins jQuery qui étendent les fonctionnalités de la bibliothèque. Voici quelques exemples de plugins jQuery couramment utilisés :

1. jQuery UI : une bibliothèque qui fournit des composants

2. Bootstrap : une bibliothèque de composants d'interface utilisateur très populaire qui utilise jQuery, ainsi que CSS et HTML, pour créer des pages web réactives et mobiles.
3. Slick : un plugin jQuery pour créer des carrousels et des diaporamas élégants et personnalisables.
4. Magnific Popup : un plugin jQuery pour créer des fenêtres modales (popup) réactives avec des effets de transition.
5. DataTables : un plugin jQuery pour créer des tableaux interactifs, triables et filtrables.
6. Select2 : un plugin jQuery pour créer des listes déroulantes personnalisables avec des fonctionnalités avancées telles que la recherche, le filtrage et le chargement dynamique de contenu.
7. Waypoints : un plugin jQuery pour déclencher des actions lorsque l'utilisateur fait défiler la page, par exemple pour animer des éléments ou charger dynamiquement du contenu.
8. Masonry : un plugin jQuery pour créer des mises en page en grille dynamiques et réactives avec des éléments disposés en fonction de leur taille et de leur position.

Il existe de nombreux autres plugins jQuery disponibles, chacun ayant ses propres fonctionnalités et avantages.

2.3.2 L'installation et l'utilisation de plugins jQuery

L'installation et l'utilisation de plugins jQuery dépendent du plugin en question, mais la plupart des plugins suivent un processus d'installation et d'utilisation similaire. Voici les étapes générales pour installer et utiliser un plugin jQuery :

1. Télécharger le fichier JavaScript du plugin : le fichier JavaScript du plugin doit être téléchargé et enregistré dans un répertoire accessible depuis votre page web. Le fichier peut être obtenu à partir du site web du plugin ou d'un référentiel de code source tel que GitHub.
2. Inclure le fichier JavaScript du plugin dans votre page web : utilisez la balise script pour inclure le fichier JavaScript du plugin dans votre page web. Assurez-vous que le chemin d'accès au fichier JavaScript est correct.
3. Inclure la bibliothèque jQuery dans votre page web : si vous n'avez pas déjà inclus la bibliothèque jQuery dans votre page web, vous devrez le faire avant d'utiliser le plugin jQuery.
4. Initialiser le plugin jQuery : pour utiliser le plugin, vous devrez l'initialiser en appelant une fonction spécifique fournie par le plugin. Cette fonction peut être appelée à partir de votre fichier JavaScript personnalisé ou en ligne directement dans votre balise HTML.

Voici un exemple de code pour installer et utiliser un plugin jQuery, en utilisant le plugin Slick pour créer un diaporama :

HTML :

```
<div class="slick-slider">
  <div><img src="image1.jpg"></div>
  <div><img src="image2.jpg"></div>
  <div><img src="image3.jpg"></div>
</div>
```

JavaScript :

```
$(document).ready(function() {
  // Inclure le fichier JavaScript de Slick
  <script src="path/to/slick.js"></script>

  // Initialiser le plugin Slick
  $('.slick-slider').slick();
});
```

Dans cet exemple, le fichier JavaScript de Slick est inclus dans la page web à l'aide de la balise script, puis le plugin est initialisé en appelant la fonction slick() sur l'élément HTML avec la classe "slick-slider". Ce code créerait un diaporama réactif en utilisant les images fournies, et serait personnalisable avec les options de configuration disponibles dans la documentation de Slick.

2.4 TP2 : Ajout d'événements à une page web à l'aide de jQuery, utilisation d'effets pour améliorer l'interface utilisateur, utilisation d'un plugin jQuery pour ajouter une fonctionnalité à la page web

Dans cette activité pratique, vous allez créer une page web qui permet aux utilisateurs de s'abonner à une newsletter, en utilisant jQuery pour ajouter des événements et des effets d'interface utilisateur, et en utilisant un plugin jQuery pour valider les entrées utilisateur.

Étapes à suivre :

1. Créer une page web simple contenant un formulaire d'abonnement à une newsletter. Le formulaire devrait comporter les champs suivants : nom, adresse e-mail et bouton d'envoi. Assurez-vous que le formulaire a un ID et des classes CSS pour faciliter la sélection avec jQuery.

2. Ajouter jQuery à votre page web. Vous pouvez inclure jQuery en utilisant la balise script et un lien vers une version hébergée de la bibliothèque jQuery, ou en téléchargeant une copie de jQuery et en l'incluant localement dans votre page web.

3. Utiliser jQuery pour ajouter des événements d'interface utilisateur à votre formulaire. Par exemple, vous pouvez ajouter un événement pour valider les champs de formulaire lorsque l'utilisateur clique sur le bouton d'envoi, ou pour afficher un message d'erreur si l'utilisateur entre une adresse e-mail invalide.

4. Utiliser les effets jQuery pour améliorer l'interface utilisateur de votre formulaire. Par exemple, vous pouvez ajouter un effet de fondu pour masquer un message d'erreur après que l'utilisateur ait corrigé une entrée de formulaire invalide, ou pour ajouter un effet de transition lorsque l'utilisateur envoie le formulaire.

5. Utiliser un plugin jQuery pour valider les entrées de formulaire de l'utilisateur. Il existe de nombreux plugins jQuery disponibles pour la validation de formulaire, tels que jQuery Validation ou Parsley. Choisissez un plugin qui répond à vos besoins, téléchargez-le et incluez-le dans votre page web.

6. Utiliser le plugin pour ajouter des règles de validation à vos champs de formulaire. Par exemple, vous pouvez ajouter une règle pour exiger que l'utilisateur entre une adresse e-mail valide, ou pour exiger que tous les champs soient remplis avant que l'utilisateur puisse soumettre le formulaire.

7. Tester votre formulaire en entrant des données valides et invalides pour vous assurer que les événements, les effets et la validation fonctionnent correctement.

8. Mettre en ligne votre page web pour permettre aux utilisateurs de s'abonner à votre newsletter.

Voici un exemple de code pour chaque étape de l'activité pratique :

1. Création d'un formulaire d'abonnement à la newsletter avec des ID et des classes pour faciliter la sélection avec jQuery :

```
<form id="newsletter-form">
  <label for="name">Nom :</label>
  <input type="text" id="name" class="form-input" required>

  <label for="email">Adresse e-mail :</label>
  <input type="email" id="email" class="form-input" required>

  <button type="submit" id="submit-btn">S'abonner</button>
</form>
```

2. Inclusion de la bibliothèque jQuery dans la page web :

```
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
```

3. Ajout d'un événement pour valider les champs de formulaire lors de la soumission et afficher un message d'erreur si l'utilisateur entre une adresse e-mail invalide :

```
$(document).ready(function() {
  $('#newsletter-form').submit(function(event) {
    event.preventDefault();
    var name = $('#name').val();
    var email = $('#email').val();
    var emailRegex = /\S+@\S+\.\S+/;

    if (name === '' || email === '' || !emailRegex.test(email)) {
      $('#error-message').text('Veuillez remplir tous les champs avec des informations valides.').show();
    } else {
      // Envoi du formulaire à un serveur pour traitement
    }
  });
});
```

4. Ajout d'un effet de fondu pour masquer un message d'erreur après que l'utilisateur ait corrigé une entrée de formulaire invalide :

```
$(document).ready(function() {
  $('#newsletter-form').submit(function(event) {
    event.preventDefault();
    var name = $('#name').val();
    var email = $('#email').val();
    var emailRegex = /\S+@\S+\.\S+/;

    if (name === '' || email === '' || !emailRegex.test(email)) {
      $('#error-message').text('Veuillez remplir tous les champs avec des informations valides.').fadeIn();
    } else {
      $('#error-message').fadeOut();
      // Envoi du formulaire à un serveur pour traitement
    }
  });
});
```

5. Inclusion d'un plugin jQuery de validation de formulaire, par exemple jQuery Validation :

```
<script src="https://cdn.jsdelivr.net/jquery.validation/1.16.0/jquery.validate.min.js"></script>
```

6. Ajout de règles de validation à chaque champ de formulaire en utilisant le plugin jQuery Validation :

```
$(document).ready(function() {
  $('#newsletter-form').validate({
    rules: {
      name: {
        required: true,
      },
      email: {
        required: true,
        email: true,
      },
    },
    messages: {
      name: {
        required: 'Veuillez entrer votre nom.',
      },
      email: {
        required: 'Veuillez entrer votre adresse e-mail.',
        email: 'Veuillez entrer une adresse e-mail valide.',
      },
    },
    submitHandler: function(form) {
      // Envoi du formulaire à un serveur pour traitement
    }
  });
});
```

7. Test du formulaire en entrant des données valides et invalides pour s'assurer que les événements, les effets et la validation fonctionnent correctement.

8. Mise en ligne de la page web pour permettre aux utilisateurs de s'abonner à la newsletter en toute sécurité et de manière conviviale.

9. Code CSS :

```
<style>
label {
  display: block;
  margin-bottom: 10px;
}

.form-input {
  display: block;
  width: 100%;
  padding: 5px;
  margin-bottom: 20px;
```

```
12        border: 1px solid #ccc;
13      }
14
15      #submit-btn {
16        display: block;
17        margin-top: 20px;
18        padding: 10px 20px;
19        border: none;
20        background-color: #0077cc;
21        color: #fff;
22        font-size: 16px;
23        cursor: pointer;
24      }
25
26      #error-message {
27        display: none;
28        margin-top: 10px;
29        padding: 10px;
30        border: 1px solid #cc0000;
31        background-color: #ffe6e6;
32        color: #cc0000;
33        font-size: 14px;
34        font-weight: bold;
35      }
36    </style>
```

3 Challenge : Jeux de Tic Tac Toe (Morpion)

Dans ce projet, vous allez créer un jeu de Tic Tac Toe en utilisant la bibliothèque jQuery et les animations.

Règles du jeu

Le Tic Tac Toe est un jeu de stratégie pour deux joueurs. Les joueurs placent chacun leur tour leur symbole (X ou O) sur une grille de 3x3 cases. Le joueur qui arrive à aligner trois de ses symboles horizontalement, verticalement ou en diagonale gagne la partie.

Fonctionnalités

- Au début du jeu, les joueurs choisissent leur symbole : X ou O.
- Le joueur qui commence est choisi aléatoirement.
- Les joueurs jouent à tour de rôle jusqu'à ce que l'un des joueurs gagne ou que la grille soit remplie.

- Lorsqu'un joueur gagne, une animation s'affiche pour indiquer les cases gagnantes.
- Si la partie se termine par un match nul, une animation s'affiche pour indiquer la fin de la partie.
- Il est possible de rejouer une partie à la fin.

Contraintes techniques

- Le jeu doit être développé en utilisant jQuery.
- Les animations doivent être réalisées en utilisant la bibliothèque jQuery UI.
- Le code doit être structuré et facilement lisible.
- Le design du jeu est libre, mais il doit être agréable et fonctionnel.

Bonus

Si vous avez terminé les fonctionnalités de base, voici quelques idées pour ajouter des fonctionnalités supplémentaires :

- Implémenter un mode de jeu contre l'ordinateur.
- Ajouter une fonctionnalité de chat pour que les joueurs puissent discuter pendant le jeu.
- Implémenter un système de points ou de classement pour les joueurs.
- Ajouter une fonctionnalité de mise en attente pour les parties multijoueurs en ligne.

Modalités de livraison

Le projet doit être livré sous forme de code source sur un dépôt GitHub public. Le code doit être accompagné d'un fichier README.md décrivant l'utilisation et l'installation du jeu.

Indications

Génération de la grille

Pour mettre en place la grille avec les divs en utilisant jQuery, voici les étapes à suivre :

1. Créer une div principale qui va contenir la grille. Cette div peut avoir une classe grid pour faciliter le style CSS.

```
1     <div class="grid"></div>
```

2. Utiliser une boucle pour générer les cases de la grille. Dans cet exemple, on utilise une boucle for pour générer 9 cases. Chaque case est une div qui a une classe cell et un identifiant unique cell-x-y où x et y sont les coordonnées de la case.

```
// Générer les cases de la grille
for (let x = 0; x < 3; x++) {
  for (let y = 0; y < 3; y++) {
    let cell = $('<div>').addClass('cell').attr('id', 'cell-' + x↩
        + '-' + y);
    $('.grid').append(cell);
  }
}
```

3. Appliquer un style CSS à la grille et aux cases pour les afficher correctement. Voici un exemple de style CSS qui permet d'afficher la grille en utilisant une disposition en grille de 3x3 et une bordure pour chaque case :

```
.grid {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  grid-template-rows: repeat(3, 1fr);
  gap: 5px;
  border: 1px solid black;
  padding: 5px;
}

.cell {
  border: 1px solid black;
  width: 100%;
  height: 100%;
}
```

4. Pour tester que la grille s'affiche correctement, vous pouvez ajouter le code suivant dans votre fichier JavaScript :

```
// Tester la grille
$('#cell-0-0').text('X');
$('#cell-1-1').text('O');
$('#cell-2-2').text('X');
```

Cela va placer les symboles "X" et "O" dans les cases (0,0), (1,1) et (2,2) de la grille.

Avec ces étapes, vous devriez être en mesure de créer une grille en utilisant jQuery et les divs.

Structure de données

Pour stocker l'état de jeu d'un Tic Tac Toe, on peut utiliser une matrice ou un tableau à deux dimensions. Chaque élément du tableau correspond à l'état d'une case de la grille.

Dans cet exemple, on utilisera un tableau à deux dimensions de taille 3x3 pour représenter la grille. Chaque élément du tableau est initialisé à la valeur null, indiquant que la case est vide.

```
1     let gameState = [
2       [null, null, null],
3       [null, null, null],
4       [null, null, null]
5     ];
```

Lorsqu'un joueur joue un coup, on met à jour la valeur correspondante dans le tableau. Par exemple, si le joueur joue un coup à la case (0, 1) et choisit le symbole "X", on met à jour le tableau comme suit :

```
1     gameState[0][1] = 'X';
```

On peut ensuite utiliser ce tableau pour déterminer si un joueur a gagné ou s'il y a match nul.

Cette structure de données est simple et efficace pour stocker l'état de jeu d'un Tic Tac Toe. Elle permet de représenter facilement la grille et de mettre à jour l'état du jeu à chaque coup joué.