JS – JQUERY

Légende :

Pour approfondir *Exemple* *Notion* *Exercice*

MANIPULER LE DOM

Nous allons voir dans ce chapitre les bases pour manipuler le DOM. Nous partons du principe que vous êtes déjà familier avec les bases d'HTML, CSS, JS, et que savez ce qu'est le DOM.

Plan du cours :

Sélectionner

La première étape pour interagir avec des élément du DOM consiste à pouvoir les sélectionner.
Nous allons voir comment faire de façon comparative entre les CSS, le Javascript natif, et Jquery.

En CSS

Savoir comment sélectionner des éléments du DOM en CSS est primordial car on va pouvoir utiliser la même logique en Javascript et en Jquery.

En CSS on peut sélectionner un ou plusieurs élément(s) du DOM via les façons suivantes :

Sélection	Syntaxe	Comportement
Tout	*{...}	Sélectionne tout
Par balise	p{....}	Sélectionne tous les
Par classe	p.exemple{....}	Sélectionne tous les dont la classe est « exemple »
Par classes	p.exemple.autre-exemple	Sélectionne tous les dont la classe est « exemple » ET « autre-exemple »
Par identifiant	p#exemple	Sélectionne l'unique dont l'id est « exemple »
Par pattern de chemin	footer p	Sélectionne les qui satisfont au pattern de chemin « descendant d'un <footer> »
Par enfant direct	footer > p	Sélectionne les qui sont un enfant direct d'un <footer>
Par premier enfant	p:first-child	Sélectionne les qui sont le premier fils de leur élément parent parmi tous les enfants <footer> sélectionné pas sélectionné </footer>

		<footer> pas sélectionné pas sélectionné </footer>
Par dernier enfant	p:last-child	Sélectionne les qui sont le dernier fils de leur élément parent parmi tous les enfants
Par premier enfant d'un type	p:first-of-type	Sélectionne les qui sont le premier fils de leur élément parent parmi tous les enfants de type <footer> pas sélectionné sélectionné </footer>

Par dernier enfant d'un type	p:last-of-type	Sélectionne les qui sont le dernier fils de leur élément parent parmi tous les enfants de type
		
		<footer>
		   pas sélectionné
		  pas sélectionné
		  sélectionné
		   pas sélectionné
		</footer>

Par enfant unique	p:only-child	Sélectionne les l'unique enfant de leur parent
Par absence d'enfant	p:empty	Sélectionne les qui n'ont pas d'enfant
Par numéro	p:nth-child(n)	Sélectionne les qui sont le $n^{ème}$ enfant de leur parent parmi tous les enfants
Par numéro parmi les éléments d'un type	p:nth-of-type(n)	Sélectionne les qui sont le $n^{ème}$ enfant de leur parent parmi les enfants de type
Par enfant de numéro pair	p:nth-child(even)	Sélectionne les qui sont les enfant de numéro pair parmi tous les enfants

Par enfant de numéro impair	p:nth-child(odd)	Sélectionne les <p> qui sont les enfant de numéro impair parmi tous les enfants
Par enfant de d'un type numéro pair	p:nth-of-type(even)	Sélectionne les <p> qui sont les enfant de numéro pair parmi tous les enfants de type <p>
Par enfant d'un type de numéro impair	p:nth-of-type(odd)	Sélectionne les <p> qui sont les enfant de numéro impair parmi tous les enfants de type <p>
Par numéro d'enfant suivant un pattern	p:nth-child(2n)	Sélectionne les <p> qui sont satisfont le pattern « le numéro est un multiple de 2 » parmi tous les enfants
Par numéro d'enfant suivant un pattern (autre cas)	p:nth-child(3n + 1)	Sélectionne les <p> qui sont satisfont le pattern « le numéro est un multiple de 3 + 1 » parmi tous les enfants
Par numéro d'enfant d'un type suivant un pattern	p:nth-of-type(2n)	Sélectionne les <p> qui sont satisfont le pattern « le numéro est un multiple de 2 » parmi tous les enfants de type <p>
Par élément suivant	h3 + p	Sélectionne le premier <p> parmi les <p> qui suivent directement un <h3>
Par éléments suivants	h3 ~ p	Sélectionne tous les <p> qui suivent un <h3> (directement ou non)
Par exclusion	p:not(:nth-child(n))	Sélectionne les <p> qui ne sont pas le $n^{ème}$ enfant de leur parent parmi tous les enfants
Par événement de survol	p:hover	Sélectionne les <p> qui sont survolés par la souris
Par événement de focus	input:focus	Sélectionne les <input> qui ont le focus
Par événement de click pour un lien	a:active	Sélectionne les <a> sur lesquels on clic
Par état	a:link	Sélectionne les <a> dont l'état est neutre
	a:visited	Sélectionne les <a> dont sur

		lesquels on a déjà cliqué
Par présence d'attribut	a[title]	Sélectionne les <a> dont l'attribut « title » est défini
Par valeur d'attribut	a[title=''exemple'']	Sélectionne les <a> dont l'attribut « title » vaut « exemple »
Par attribut contenant une valeur	a[title*=''exemple'']	Sélectionne les <a> dont l'attribut « title » contient « exemple »
Par attribut commençant par une valeur	a[title^=''exemple'']	Sélectionne les <a> dont l'attribut « title » commence par « exemple »
Par attribut finissant par une valeur	a[title$=''exemple'']	Sélectionne les <a> dont l'attribut « title » finit par « exemple »
Par une des valeurs d'un attribut multivaleur	a[title~=''exemple'']	Sélectionne les <a> dont l'attribut « title » contient plusieurs valeur séparées par un espace dont une vaut « exemple »

En Javascript

Maintenant, comment sélectionner des éléments du dom en JS ?

Sélection	Syntaxe	Comportement
Tout	document.getElementsByTagName('*')	Sélectionne tout
Par balise	document.getElementsByTagName('p')	Sélectionne tous les <p>
Par classe	document.getElementsByClassName('exemple')	Sélectionne tous les éléments dont la classe est « exemple », au format d'une liste HTMLCollection
Par identifiant	document.getElementById('exemple')	Sélectionne l'unique élément dont l'id est « exemple »
Par name	document.getElementByName('exemple')	Sélectionne le(s) élément(s) dont le name est « exemple », au

		format d'une liste HTMLCollection
Par CSS pour un élément	document.querySelector('footer p:first-child')	Sélectionne le premier élément qui satisfait à la règle CSS passée en paramètre, donc ici le premier des les <p> qui sont le premier enfant de leur parent et descendant d'une balise footer
Par CSS pour plusieurs élément	document.querySelectorAll('footer p:first-child')	Sélectionne tous les éléments qui satisfont à la règle CSS passée en paramètre, donc ici tous les <p> qui sont le premier enfant de leur parent et descendant d'une balise footer, au format d'une liste HTMLCollection

Comme vous pouvez le constater, les méthodes « querySelector » et « querySelectorAll » permettent de sélectionner des éléments du DOM via la syntaxe de sélection utilisée en CSS, d'où l'intérêt de bien la connaître.

En JQuery

Maintenant, comment sélectionner des éléments du dom en Jquery?

Sélection	Syntaxe	Comportement
Tout	$('*')	Sélectionne tout
Par balise	$('p')	Sélectionne tous les <p>
Par classe	$('.exemple')	Sélectionne tous les éléments dont la classe est « exemple », au format d'une liste HTMLCollection

Par identifiant	$('#exemple')	Sélectionne l'unique élément dont l'id est « exemple »
Par CSS pour plusieurs élément	$('footer p:first-child')	Sélectionne tous les éléments qui satisfont à la règle CSS passée en paramètre, donc ici tous les <p> qui sont le premier enfant de leur parent et descendant d'une balise footer, au format d'une liste HTMLCollection

Là aussi, on peut utiliser les sélecteurs CSS qui sont très puissants, donc il est quasiment inutile d'aller plus loin. Cependant, sachez que Jquery implémente des méthodes équivalentes à quasiment tous les sélecteurs CSS, vous pouvez si vous êtes curieux les trouver ici :

https://api.jquery.com/category/selectors/

Notez certains sélecteurs qui sont uniques à Jquery, tels que «:animated », «:contains() »

Sélection	**Syntaxe**	**Comportement**
Tout	$(''*'')	Sélectionne tout
Par balise	$(''p'')	Sélectionne tous les <p>
Par classe	$(''.exemple'')	Sélectionne tous les éléments dont la classe est « exemple », au format d'une liste HTMLCollection
Par identifiant	$(''#exemple'')	Sélectionne l'unique élément dont l'id est « exemple »
Par CSS pour plusieurs élément	$('footer p:first-child')	Sélectionne tous les éléments qui satisfont à la règle CSS passée en paramètre, donc ici tous les <p> qui sont le premier enfant de leur parent et descendant d'une

		balise footer, au format d'une liste HTMLCollection
Par pattern de chemin	footer p	Sélectionne les qui satisfont au pattern de chemin « descendant d'un <footer> »
Par enfant direct	footer > p	Sélectionne les qui sont un enfant direct d'un <footer>
Par premier enfant	p:first-child	Sélectionne les qui sont le premier fils de leur élément parent parmi tous les enfants <footer> sélectionné pas sélectionné </footer> <footer> pas sélectionné pas sélectionné </footer>
Par dernier enfant	p:last-child	Sélectionne les qui sont le dernier fils de leur élément parent parmi tous les enfants
Par premier enfant d'un type	p:first-of-type	Sélectionne les qui sont le premier fils de leur élément parent parmi tous les enfants de type <footer> pas sélectionné sélectionné </footer>
Par dernier enfant d'un type	p:last-of-type	Sélectionne les qui sont le dernier fils de leur élément parent parmi tous les enfants de type <footer> pas sélectionné pas sélectionné sélectionné pas sélectionné

		</footer>
Par enfant unique	p:only-child	Sélectionne les <p> l'unique enfant de leur parent
Par absence d'enfant	p:empty	Sélectionne les <p> qui n'ont pas d'enfant
Par numéro	p:nth-child(n)	Sélectionne les <p> qui sont le $n^{ème}$ enfant de leur parent parmi tous les enfants
Par numéro parmi les éléments d'un type	p:nth-of-type(n)	Sélectionne les <p> qui sont le $n^{ème}$ enfant de leur parent parmi les enfants de type <p>
Par enfant de numéro pair	p:nth-child(even)	Sélectionne les <p> qui sont les enfant de numéro pair parmi tous les enfants
Par enfant de numéro impair	p:nth-child(odd)	Sélectionne les <p> qui sont les enfant de numéro impair parmi tous les enfants
Par enfant de d'un type numéro pair	p:nth-of-type(even)	Sélectionne les <p> qui sont les enfant de numéro pair parmi tous les enfants de type <p>
Par enfant d'un type de numéro impair	p:nth-of-type(odd)	Sélectionne les <p> qui sont les enfant de numéro impair parmi tous les enfants de type <p>
Par numéro d'enfant suivant un pattern	p:nth-child(2n)	Sélectionne les <p> qui sont satisfont le pattern « le numéro est un multiple de 2 » parmi tous les enfants
Par numéro d'enfant suivant un pattern (autre cas)	p:nth-child(3n + 1)	Sélectionne les <p> qui sont satisfont le pattern « le numéro est un multiple de 3 + 1 » parmi tous les enfants
Par numéro d'enfant d'un type suivant un pattern	p:nth-of-type(2n)	Sélectionne les <p> qui sont satisfont le pattern « le numéro est un multiple de 2 » parmi tous les enfants de type <p>
Par élément suivant	h3 + p	Sélectionne le premier <p> parmi les <p> qui suivent directement un <h3>
Par éléments suivants	h3 ~ p	Sélectionne tous les <p> qui

		suivent un <h3> (directement ou non)
Par exclusion	p:not(:nth-child(n))	Sélectionne les <p> qui ne sont pas le n^{ème} enfant de leur parent parmi tous les enfants
Par événement de survol	p:hover	Sélectionne les <p> qui sont survolés par la souris
Par événement de focus	input:focus	Sélectionne les <input> qui ont le focus
Par événement de click pour un lien	a:active	Sélectionne les <a> sur lesquels on clic
Par état	a:link	Sélectionne les <a> dont l'état est neutre
	a:visited	Sélectionne les <a> dont sur lesquels on a déjà cliqué
Par présence d'attribut	a[title]	Sélectionne les <a> dont l'attribut « title » est défini
Par valeur d'attribut	a[title=''exemple'']	Sélectionne les <a> dont l'attribut « title » vaut « exemple »
Par attribut contenant une valeur	a[title*=''exemple'']	Sélectionne les <a> dont l'attribut « title » contient « exemple »
Par attribut commençant par une valeur	a[title^=''exemple'']	Sélectionne les <a> dont l'attribut « title » commence par « exemple »
Par attribut finissant par une valeur	a[title$=''exemple'']	Sélectionne les <a> dont l'attribut « title » finit par « exemple »
Par une des valeurs d'un attribut multivaleur	a[title~=''exemple'']	Sélectionne les <a> dont l'attribut « title » contient plusieurs valeur séparées par un espace dont une vaut « exemple »

Sélection	**Syntaxe**	**Comportement**
Tout	$('*')	Sélectionne tout
Par balise	$('p')	Sélectionne tous les <p>
Par classe	$('.exemple')	Sélectionne tous les éléments dont la classe est « exemple », au

		format d'une liste HTMLCollection
Par identifiant	$('#exemple')	Sélectionne l'unique élément dont l'id est « exemple »
Par CSS pour plusieurs élément	$('footer p:first-child')	Sélectionne tous les éléments qui satisfont à la règle CSS passée en paramètre, donc ici tous les <p> qui sont le premier enfant de leur parent et descendant d'une balise footer, au format d'une liste HTMLCollection

Modifier

La seconde étape maintenant que nous savons comment sélectionner des éléments du DOM consiste à voir comment les modifier.

En Javascript

Il y a deux moyens de modifier la valeur d'un attribut en JS : soit en appelant la propriété de l'élément, soit en en utilisant la méthode setAttribute() et en lui passant en paramètre le nom de l'attribut

Sélection	Syntaxe	Comportement
Modifier la valeur d'un attribut	element.setAttribute('name', 'value')	Assigne la valeur 'value' à l'attribut 'name' de l'élément
Modifier la valeur de l'innerHTML	element.innerHTML = 'value' équivalent de element.setAttribute('innerHTML', 'value')	Assigne la valeur 'value' à l'attribut 'innerHTML' de l'élément
Modifier la valeur de l'attribut className	element.className = 'value' équivalent de element.setAttribute('className', 'value')	Assigne la valeur 'value' à l'attribut 'className' de l'élément
Modifier la valeur du style de l'élément	element.style.cssProperty = 'value'	Assigne la valeur 'value' à la propriété 'cssProperty' des styles de l'élément. Exemple : ele.style.borderWidth = 'solid' Notez qu'une propriété CSS qui s'écrit avec un tiret s'écrit sans tiret en camelCase
mModifier la valeur d'un input	element.value = 'value'	Assigne la valeur 'value' à la propriété 'value' d'un input

Notez que grâce à la méthode setAttribute, il est possible de modifier tous les attributs simplement...

En Jquery

Il y a deux moyens de modifier la valeur d'un attribut en JS : soit en appelant la méthode Jquery ciblant propriété de l'élément, soit en en utilisant la méthode attr() et en lui passant en paramètre le nom de l'attribut

Sélection	Syntaxe	Comportement
Modifier la valeur d'un attribut	$(selecteur).attr('name', 'value')	Assigne la valeur 'value' à l'attribut 'name' de l'élément
Modifier la valeur de l'innerHTML	$(selecteur).html('value')	Assigne la valeur 'value' à l'attribut 'innerHTML' de l'élément
Modifier la valeur de l'attribut className	$(selecteur).addClass('value')	Ajoute la valeur 'value' à la liste des classes de l'élément sélectionné
Modifier la valeur du style de l'élément	$(selecteur).css('css property', 'value')	Assigne la valeur 'value' à la propriété 'css property' des styles de l'élément. Exemple : $('#element').css('border-width', 'solid')
Modifier la valeur d'un input	$(selecteur).val('value')	Assigne la valeur 'value' à la propriété 'value' d'un input

Créer

En Javascript

Pour créer un élément en javascript, trois étapes sont nécessaires :

Créer l'élément

On créé l'élément via la méthode document.createElement('tagName')

```
let newP = document.createElement('p') ;
```

Ici on a une variable newP qui contient un élément de type « p ».

Notez que ce p n'a aucune valeur pour aucune de ses propriétés, et qu'il n'existe pas dans le DOM. Il flotte dans les limbes de la page web tel une âme perdue pour le moment...

Valoriser l'élément

On donne les valeurs voulues à l'élément

```
newP.setAttribute('innerHTML', 'Bonjour à tous') ;
newP.setAttribute('class', 'bienvenu') ;
```

Ici on a notre newP a désormais un contenu et une classe.

Insérer l'élément dans le DOM

Afin que notre élément puisse enfin exister dans la page et cesse d'errer comme une âme damnée, nous devons l'insérer dans le document.

Pour cela, nous pouvons soit l'insérer comme dernier enfant d'un élément parent via la méthode appendChild(), soit avant un autre élément via la méthode insertBefore()

```
let body = document.getElementsByTagName('body')[0] ;
body.appendChild(newP) ;
```

Ici on a ajouté notre newP dans le body, comme dernier élément de celui ci

ou

```
let suivant = document.getElementById('exemple');
suivant.parentNode.insertBefore(newP,  suivant) ;
```

Ici on a inséré notre newP avant un élément dont l'id est « exemple »

Ce qui nous donne mis ensemble :

```
let newP = document.createElement('p') ;

newP.setAttribute('innerHTML', 'Bonjour à tous') ;
newP.setAttribute('class', 'bienvenu') ;

let body = document.getElementsByTagName('body')[0] ;
body.appendChild(newP) ;
```

Alternative bourrin

Il existe une alternative bourrin qui consiste à ne pas utiliser la logique objet, mais à modifier la valeur du innerHTML du conteneur de l'élément à créer. Le navigateur se débrouille ensuite pour interpréter tout ça :

```
let body = document.getElementsByTagName('body')[0] ;
 body.innerHTML = body.innerHTML + '<p class=''bienvenu''>Bonjour à tous</p> ;
```

```

```

C'est plus rapide à écrire mais pas à exécuter.

Par ailleurs on ne peut pas interagir simplement avec les objets créés (dans l'hypothèse où on souhaiterait lui attacher un événement par exemple)

En JQuery

Pour créer un élément en Jquery, trois étapes sont nécessaires, mais grâce à la syntaxe de Jquery elles peuvent facilement être réunies :

Créer et valoriser l'élément

```
let newP = $('<p></p>') ;
```

Ici on a une variable newP qui contient un élément de type « p ».

Valoriser l'élément

```
newP.html('Bonjour à tous') ;
newP.addClass('bienvenu') ;
```

Ou si on veut faire les deux étapes en même temps :

```
let newP = $('<p class=''bienvenu''>Bonjour à tous</p>') ;
```

Insérer l'élément dans le DOM

```
$(''#conteneur'').append(newP) ;
```

Ici on a ajouté notre newP dans le body, comme dernier élément de celui ci

ou si on veut tout faire en même temps :

```
$(''#conteneur'').append('<p class=''bienvenu''>Bonjour à tous</p>') ;
```

Notez que la syntaxe suivante est également possible :

```
$('<p class=''bienvenu''>Bonjour à tous</p>').appendTo(''#conteneur'') ;
```

Ou encore :

```
$('<p></p>', {
        'class' : 'bienvenu',
        'html': 'Bonjour à tous'
   }).appendTo('#conteneur') ;
```

Par ailleurs, Jquery dispose d'une méthode prepend, qui insère au début du conteneur, contrairement à append qui insère à la fin :

```
$(''#conteneur'').prepend('<p class=''bienvenu''>Bonjour à tous</p>') ;
```

Et il existe également deux méthodes pour insérer avant ou après un élément :

```
$('<p class=''bienvenu''>Bonjour à tous</p>').insertBefore($(''#element'')) ;
$('<p class=''bienvenu''>Bonjour à tous</p>').insertAfter($(''#element'')) ;
```

```
```

Notez que ces méthodes permettent des insertions multiples si vous utilisez un sélecteur de classe au lieu d'un identifiant :

```
$('<p class=''bienvenu''>Bonjour à tous</p>').insertBefore($(''.exemple'')) ;
```

Va insérer un paragraphe avant chaque élément de classe « exemple »

Enfin, on peut chaîner les créations / insertions :

```
$('body').append(
                $('<p></p>')
                .addClass('bienvenu')
                .append(
                     $('<span></span>')
                      .html('Bonjour à tous')
                )
        );
```

Supprimer

En Javascript

Pour supprimer un élément en Javascript on utilise la méthode removeChild appliquée sur le parent avec l'élément à supprimer en paramètre :

```
let parent = document.getElementById('Conteneur') ;
let enfant = docu.getElementById('Element') ;
parent.removeChild(enfant) ;
```

En JQuery

Pour supprimer un élément en JQuery on a deux méthodes : empty et remove

Empty supprime tous les enfants d'un élément :

```
$('#parent').empty() ;
```

Remove supprime un élément (et donc ses enfants, tout comme removeChild en JS) :

```
$('#element').remove() ;
```

Par ailleurs remove peut s'appliquer sur une liste d'élément :

```
$('p').remove() ;
```

Supprime tous les <p>

Enfin remove peut recevoir un paramètre pour filtrer le ou les éléments à supprimer :

```
$('p').remove('.exemple') ;
```

Supprime tous les <p> dont la classe est exemple

EXERCICE

Créer une page en html qui intègre un lien vers un fichier javascript.

Dans ce fichier javascript, une fois la page chargée effectuez les actions suivantes :

1- créer un tableau qui contient les éléments suivants en pur javascript et insérez le dans le body :

Bouton, texte : « Effacer »	Input type text, placeholder = 0		
Bouton, texte : « % »	Bouton, texte : « x^2 »	Bouton, texte : « √ »	Bouton, texte : « ÷ »
Bouton, texte : « 7 »	Bouton, texte : « 8 »	Bouton, texte : « 9 »	Bouton, texte : « X »
Bouton, texte : « 4 »	Bouton, texte : « 5 »	Bouton, texte : « 6 »	Bouton, texte : « - »
Bouton, texte : « 1 »	Bouton, texte : « 2 »	Bouton, texte : « 3»	Bouton, texte : « + »
Bouton, texte : « +/- »	Bouton, texte : « 0 »	Bouton, texte : « ,»	Bouton, texte : « = »

2- faites évoluer votre code si ce n'est déjà fait pour que ces actions soient effectuées dans l'instanciation d'un objet « calculatrice ». Faites en sorte que cet objet puisse recevoir un paramètre : l'id du conteneur, et que votre objet insère la structure html dans ce conteneur s'il existe dans la page. Puis dans votre page, via un script instanciez un objet calculatrice dans le conteneur de votre choix.

3- reproduisez la même chose en JQuery