

# Phalcon-jquery : JQuery

Principales fonctionnalités de la librairie Phalcon-JQuery (hors JQuery UI et Twitter Bootstrap)

## -- Association d'évènements

### -- Evénements connus

#### -- Exemple

Click sur un bouton d'id **#btn** :

```
public function clickAction(){
    $this->click("#btn","console.log('click sur #btn');");
    $this->query->compile($this->view);
}
```

Vue associée :

```
<input type="button" value="Test click" id="btn">
{{script_foot}}
```

### -- Utilisation du paramètre event du callback

Le code passé au callback peut utiliser le paramètre **event** passé par JQuery, et **\$(this)**, la cible de l'évènement (target) :

**Exemple** : appui sur une touche, récupération du code de la touche frappée et de l'id de l'élément ayant généré l'évènement.

```
public function clickAction(){
    $this->jquery->keydown("#btn","console.log('code de touche appuyée : ' +
event.which + ' sur #' + $(this).attr('id'));");
    $this->jquery->compile($this->view);
}
```

**Exemple** : affichage/masquage d'un élément sur changement de la valeur d'une checkbox :

```
public function checkedAction(){
    $this->jquery->change("#ck",$this->jquery->toggle("#fs",$(this).checked()));
    $this->jquery->compile($this->view);
}
```

Vue associée :

```
<input type="checkbox" id="ck" checked>
<fieldset id="fs">
<legend>Elément à afficher/masquer</legend>
</fieldset>
{{script_foot}}
```

### -- Liste des méthodes JQuery liées aux évènements

Méthode	Evènement
blur(selector,callback)	Sur perte du focus par l'élément désigné par <b>selector</b>
change(selector,callback)	Sur changement de valeur d'un <b>input</b> , <b>select</b> ou <b>textarea</b> (déclenché généralement sur perte focus)
click(selector,callback)	Sur click de l'élément désigné par <b>selector</b>
dblclick(selector,callback)	Sur double-click de l'élément désigné par <b>selector</b>
error(selector,callback)	Sur erreur de chargement sur <b>selector</b>
focus(selector,callback)	Sur réception du focus par l'élément désigné par <b>selector</b>
hover(selector,callback)	Sur déplacement de la souris au dessus de l'élément désigné par <b>selector</b>
keydown(selector,callback)	Sur touche clavier enfoncée lorsque l'élément désigné par <b>selector</b> a le focus
keypress(selector,callback)	Sur touche clavier enfoncée lorsque l'élément désigné par <b>selector</b> a le focus
keyup(selector,callback)	Sur touche clavier relâchée lorsque l'élément désigné par <b>selector</b> a le focus
load(selector,callback)	Sur chargement de l'élément désigné par <b>selector</b>
mousedown(selector,callback)	Sur bouton souris enfoncé lorsque l'élément désigné par <b>selector</b>
mouseout(selector,callback)	Sur sortie de la souris de l'élément désigné par <b>selector</b>
mouseover(selector,callback)	Sur passage souris au dessus de l'élément désigné par <b>selector</b>
mouseup(selector,callback)	Sur bouton souris relâché lorsque l'élément désigné par <b>selector</b>
ready(callback)	Sur chargement du document terminé
scroll(selector,callback)	Sur défilement dans/sur l'élément désigné par <b>selector</b>
trigger(selector,event)	Déclenche l'évènement <b>event</b> sur l'élément désigné par <b>selector</b>
unload(selector,callback)	Sur déchargement de l'élément désigné par <b>selector</b>

### -- Evènements non listés

Pour les évènements inconnus de JQuery ou ne disposant pas de méthode associée dans la librairie phalcon-jquery, on utilise les méthodes :

- `execAndBindTo(selector,event,callback)` permettant d'exécuter le **callback** sur le déclenchement de **event** sur l'élément désigné par **selector**.
- `dojQueryAndBindTo($element,$event,$elementToModify,$jqueryCall,$param="", $function="")` permettant d'exécuter la méthode jquery **jqueryCall** sur **elementToModify**, en lui passant **param** sur le déclenchement de **event** sur l'élément désigné par **element**.

### -- Requêtes Ajax

#### -- Requêtes immédiates

Il s'agit des requêtes dont l'exécution n'est pas différée (non associée à un évènement par exemple).

Les requêtes immédiates peuvent permettre le chargement d'une page en cascade : élément par élément.

## --GET

**Exemple :** Chargement de **exemple/getResponse** dans la div **#divResponse**

```
public function getAction(){
    $this->jquery->get("exemple/getResponse", "#divResponse");
    echo "<div id='divResponse'>Chargement...</div>";
    echo $this->jquery->compile();
}
public function getResponseAction(){
    echo "Chargement terminé";
    $this->view->disable();
}
```

## -- POST

**Exemple :** Post de valeurs vers **exemple/postResponse** dans la div **#divPostResponse**

```
public function postAction(){
    $this->jquery->post("testsbs/postResponse", "#divPostResponse", "{nom: 'DOE', prenom: 'John'}");
    echo "<div id='divPostResponse'>Chargement...</div>";
    echo $this->jquery->compile();
}
public function postResponseAction(){
    var_dump($_POST);
    $this->view->disable();
}
```

## -- JSON

Les requêtes JSON permettent d'alimenter en données une vue, à partir d'un retour d'objet JSON par le serveur.

**Exemple :** Le get JSON vers la page **exemple/jsonResponse** alimente tous les éléments DOM **nom** et **prenom** avec l'élément JSON retourné.

```
public function jsonAction(){
    $this->jquery->json("testsbs/jsonResponse");
    echo "<div id='nom'>Chargement...</div>";
    echo "<input type='text' id='prenom' value='chargement...'>";
    echo $this->jquery->compile();
}
public function jsonResponseAction(){
    echo '{"nom": "DOE", "prenom": "John"}';
    $this->view->disable();
}
```

```
}
```

## -- Association de requêtes à des événements

Pour faire plus court, les exemples suivants ne comportent pas de vues, et les éléments HTML sont directement produits dans le contrôleur (à éviter...)

### -- getAndBindTo

Permet d'associer l'exécution d'un Get à un événement déclenché sur un élément :

Exemple : Sur click du bouton **#btnGet**, on veut effectuer un get vers **exemple/getResponse**, et afficher le résultat dans la div **#divResponse**.

```
public function getAction(){
$this->jquery->getAndBindTo("#btn","click","testsbs/getResponse","#divResponse");
    echo "<input type='button' id='btn' value='doGet'>";
    echo "<div id='divResponse'>En attente de click...</div>";
    echo $this->jquery->compile();
}
public function getResponseAction(){
    echo "Chargement terminé";
    $this->view->disable();
}
```

### -- postFormAndBindTo

Permet d'associer le post d'un formulaire à un événement déclenché sur un élément :

Exemple : Sur click du bouton **#btnPost**, on veut poster le formulaire **frmLogin** vers **exemple/postResponse**, et afficher le résultat dans la div **#divResponse**.

```
public function postResponseAction(){
    var_dump($_POST);
    $this->view->disable();
}

public function postFormAction(){
$this->jquery->postFormAndBindTo("#btn","click","testsbs/postResponse","frmLogin","#divPostResponse");
    echo "<form id='frmLogin' name='frmLogin' onsubmit='return false;'><input
type='text' name='login' placeholder='Login... '>
    <input type='password' name='password'
placeholder='password... '></form>";
    echo "<input type='button' id='btnPost' value='doPost'>";
    echo "<div id='divPostResponse'>Avant validation</div>";
    echo $this->jquery->compile();
}
```

Il est également possible d'utiliser :

Méthode	Rôle
postAndBindTo(element,event,url[,params,responseElement])	Post les paramètres <b>params</b> vers url sur l' <b>event</b> produit sur <b>element</b>

## -- Manipulation du DOM

Méthode	Rôle
append(to,element)	ajoute l'élément DOM <b>element</b> à la fin de chaque élément désigné par <b>to</b>
prepend(to,element)	ajoute l'élément DOM <b>element</b> au début de chaque élément désigné par <b>to</b>
attr(element,attributeName [,value,immediatly])	Lit/modifie l'attribut <b>attributeName</b> de <b>element</b>
html(element[,value,immediatly])	Lit le contenu html / Affecte <b>value</b> au contenu html de <b>element</b>

## -- Utilitaires

Méthode	Rôle
show(element)	Affiche l'élément DOM <b>element</b>
hide(element)	Masque l'élément DOM <b>element</b>
toggle(element)	Affiche/masque l'élément DOM <b>element</b>
animate(element[,params,speed,extra])	Effectue une animation sur <b>element</b>
addClass(element,class[,immediatly])	Ajoute la classe <b>class*</b> à l'élément DOM <b>element</b>   <b>toggleClass(element,class[,immediatly])</b>   Ajoute/supprime la classe <b>class</b> à l'élément DOM <b>element**</b>

From:

<http://slamwiki2.kobject.net/> - SlamWiki 2.1

Permanent link:

<http://slamwiki2.kobject.net/slam4/php/phalcon/jquery/usage?rev=1426642701>

Last update: **2019/08/31 14:42**

