2025/11/04 02:01 1/7 Phalcon-jquery : JQuery

# **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.

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

Last update: 2019/08/31 14:42

#### 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 relachée 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 relaché 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

#### **Exécuter du Javascript**

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

• execAndBindTo(selector, event, callback) permettant d'exécuter le **callback** sur le déclenchement de **event** sur l'élément désigné par **selector**.

### **Exemple**

Sur dropdown (event bootstrap) du bouton, un message est affiché dans les logs de la console

public function dropdownAction(){

2025/11/04 02:01 3/7 Phalcon-jquery : JQuery

#### **Exécuter du JQuery**

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.

#### **Exemple**

Sur dropdown (event bootstrap) du bouton, un message est affiché dans l'élément #divMessage

# -- 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("exemple/postResponse","#divPostResponse","{nom:'DOE',prenom:'J
ohn'}");
        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("exemple/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();
}
```

# -- JSON Array

Il peut être également intéressant de réceptionner des données JSON sous forme de tableau, et d'afficher ces données dans une vue, en utilisant un mask (présent dans la page) :

```
public function jsonArrayAction(){
     $this->jquery->jsonArray("#mask","exemple/jsonArrayResponse");
    echo $this->jquery->compile();
}
```

La vue contenant le mask :

Les éléments à remplacer dans le mask doivent comporter l'attribut data-id

```
<div id='mask' style='display: none'><button data-id='nom' class='btn btn-
```

2025/11/04 02:01 5/7 Phalcon-jquery : JQuery

```
primary'>Chargement.../button> 
<input type='text' data-id='prenom' value='chargement...' class='form-
control'></div>
```

Une réponse JSON pour tester :

```
public function jsonArrayResponseAction(){
        echo
'[{"nom":"DOE","prenom":"John"},{"nom":"SMITH","prenom":"Robert"},{"nom":"GATES","p
renom":"Bill"}]';
        $this->view->disable();
}
```

# -- Association de requêtes à des évènements

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 évènement 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 évènement 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'</pre>
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>

Ou de combiner les méthodes différées avec les méthodes évènementielles :

#### Requête Ajax Get sur click d'un bouton :

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

# -- 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(selector,attributeName [,value,immediatly])	Lit/modifie l'attribut attributeName des éléments correspondant à selector
html(selector[,value,immediatly])	Lit le contenu html / Affecte <b>value</b> au contenu html des éléments correspondant à <b>selector</b>

# -- Utilitaires

Méthode	Rôle
show(selector[,speed,callback])	Affiche le ou les éléments DOM correspondant au selecteur <b>selector</b>
hide(selector[,speed,callback])	Masque le  ou les éléments DOM correspondant au selecteur <b>selector</b>
fadeIn(selector[,speed,callback])	Effectue une transition sur le ou les éléments DOM correspondant au selecteur <b>selector</b> pour le(s) rendre opaque

2025/11/04 02:01 7/7 Phalcon-jquery : JQuery

Méthode	Rôle
	Effectue une transition sur le ou les éléments DOM correspondant au selecteur <b>selector</b> pour le(s) rendre transparent
TOUGHERSEIECTORI	Affiche/masque le  ou les éléments DOM correspondant au selecteur selector
animate(selector[,params,speed,extra])	Effectue une animation sur le ou les éléments DOM correspondant au selecteur <b>selector</b>
addClass(selector,class[,immediatly])	Ajoute la classe <b>class</b> sur les éléments DOM correspondant au selecteur <b>selector</b>
toggleClass(selector,class[,immediatly])	Ajoute/supprime la classe <b>class</b> sur le  ou les éléments DOM correspondant au selecteur <b>selector</b>

From:

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

Permanent link:

http://slamwiki2.kobject.net/slam4/php/phalcon/jquery/usage?rev=1427037597

Last update: 2019/08/31 14:42

