Guide du développeur

Utilisation de DataTable

Edouard COMTET

Février 2014

Guide du développeur

Utilisation de DataTable

1. Introduction

Lors de la création d'une partie d'administration, la plupart des des développeurs doivent régulièrement mettre en place des tables d'administration.

Le but de ce framework est de vous permettre, en quelques lignes de code, de générer une table HTML/CSS/Jquery en lien avec votre base de données. La modification de la table sera, de manière automatique, répercuté dans la base de données.

Ce document est destiné aux développeurs qui devront utiliser cet outil. Il vous expliquera quelle fonction utilisé et qu'elles sont les fonctionnalités paramétrables.

Dans la suite de ce document, on considèrera que le framework est installé dans un dossier du nom *DataTabl*e et que le fichier utilisant ce dernier est à la racine et se nomme *index.php*.

Ce doucement est rattaché à un deuxième qui permet de comprendre l'architecture du framework et comment le modifier.

Ce framewok se base sur DataTable, disponible à l'adresse http://datatables.net/.

2. Installation

L'installation du framework se décompose en 3 étapes simplistes:

- Copier l'intégralité des fichiers dans le répertoire d'installation
- Il est nécessaire que le dossier *DataTable/temp* soit accessible en lecture et en écriture par le serveur WEB (celui qui exécutera les pages PHP).
- Initialiser les variables de connexions disponible au début du fichier *DataTable/php/Table.php*.

Pour le paramètres de langues, se référer à la section langue, pour ajouter un driver, se référer à l'autre document.

```
// Default parameters
protected static $BDD_URL = "127.0.0.1";
protected static $BDD_LOGIN = "root";
protected static $BDD_PASSWORD = "";
protected static $BDD_DATABASE = "pfe";
protected static $BDD_DRIVER = "mysql";
protected static $LANGUAGE = "en";
protected static $ENCODING = "utf8";
```

Ceux sont ces paramètres qui seront utilisé par défaut pour la connexion à la base de données et le rendu. Il est important de ne pas mettre de constante dans ces variables. Pour pouvoir assurer la transition entre javascript et php, l'objet Table est sérialisé. L'utilisation de constante comme avec l'instruction "define" n'assure pas la désérialisation.

3. Création d'un objet Table

1. Création de la page HTML

Pour créer une nouvelle page avec un Table dedans, il faut d'abords créer une page HTML. Voici un exemple:

```
<html>
	<head>
		<title>Test DataTable</title>

		<link href="./DataTable/css/bootstrap.css" rel="stylesheet">
		<link href="./DataTable/css/bootstrap-theme.css" rel="stylesheet">
		<link href="./DataTable/css/dataTables.css" rel="stylesheet">
	</head>
	<body>
		<div class="container">
			<div class="row">
				<h1>Test du PFE avec affichage en bootstrap</h1>
			</div>
			<div class="row">
				<?php
					// CODE PHP ICI
				?>
			</div>
		</div>

		<script type="text/javascript" src="./DataTable/js/jquery.js"></script>
		<script type="text/javascript" src="./DataTable/js/bootstrap.js"></script>
		<script type="text/javascript" src="./DataTable/js/dataTables.js"></script>
		<script type="text/javascript" src="./DataTable/js/table.js"></script>
	</body>
</html>
```

Il est nécessaire d'inclure la librairie bootstrap (autant au niveau javascript que CSS), la libraire DataTable de base ainsi que le fichier css du framework et son équivalent javascript.

2. Mise en place du PHP

La structure de base étant en place, la création d'un objet Table sera un jeu d'enfant. Dans la partie PHP, vous pouvez y inscrire ceci:

```
require_once('./DataTable/php/include.php');

$table = new DataTable\php\Table();
$table->init("user");

$table->show("./DataTable/");
```

Cette partie de code permet de créer un objet Table, de le connecter à la table *user* et de l'afficher. La fonction *show* prends un seul paramètre: le chemin où est installé le framework depuis le fichier d'utilisation. Il faut le terminer par un /.

3. Personnalisation des options

Il est possible de modifier les paramètres de connexion et de langue grâce à la fonction init.
Si il n'y a pas de paramètre passé ou qu'ils sont vide, les valeurs par défaut vu plus haut seront utilisées.
Nous étudions plus tard comment modifier le style de pagination, l'activation ou la désactivation du tri, l'autorisation de la modification, de la suppression et de

```
public function init(
       $name,
       $url = “",
       $login = “",
       $password = “",
       $database = “",
       $driver = “",
       $language = “",
       $encoding = “”
);
```

l'ajout.

Dans le deuxième document, il est également expliqué comment ajouter une langue ou un driver et la personnalisation du CSS.

La partie suivante permet d'ajouter des colonnes et des filtres dans le framework.

4. Ajout des colonnes et des filtres

Pour ajouter des colonnes et des filtres, il existe des fonctions s'appliquent sur un objet table et qui gèreront l'ensemble.

1. Ajout d'un filtre

Un filtre permet de n'afficher qu'une certaine partie des données d'une table, par exemple si on ne souhaite afficher que les utilisateurs qui sont connecté on fera

```
$table->addFilter("connected", "= TRUE");
```

Cette fonction prends donc en paramètres la colonne sur lequel on fera le tri et la condition à prendre en compte.

2. Ajout d'une colonne

Il est possible d'ajoute différents types de colonnes à un objet Table qui seront, suivant les paramètres, affichés ou non, modifiables ou non, … Voici la liste intégrale des colonnes disponibles:

- Col: Une colonne classique. La fonction prends en paramètres un libellé affiché à l'utilisateur et le nom de la colonne dans la table.
- ColIndex: Cette colonne sera utilisé comme index lors de la modification, cependant il est impossible de modifier une colonne index. Il faudra la supprimer et l'ajouter. La fonction prends en paramètre un libellé pour la colonne, le nom de la colonne dans la table, un booléen pour savoir si la colonne doit être affiché ou non et un deuxième pour préciser s'il faut proposer la colonne lors de l'ajout ou non.
- ColHidden: Cette colonne n'est jamais affiché à l'utilisateur mais lors de l'ajout, la valeur par défaut de la colonne sera ajouté. La fonction prends en paramètre un libellé, non utilisé, le nom de la colonne dans la table et la valeur par défaut (non vide).
- ColLinked: Cette colonne est lié à une autre table (notion de clef étrangère). La liaison sera donc faite automatiquement avec la seconde table et affichera la valeur de la seconde table. La fonction prends en paramètre un libellé affiché, le nom de la colonne dans la table, le nom de la table où faire liaison,

l'index de la seconde table (une seule colonne) et la colonne qui sera affiché à l'utilisateur (unique).

- ColIndexLinked: Une colonne qui servira d'index et qui est lié à une autre table. Elle prends en paramètre les mêmes paramètres que la fonction ColLinked. Il n'est pas nécessaire de proposer de l'afficher ou non ou si on peut l'ajouter ou pas car sinon il sera plus simple d'utiliser ColIndex. Sur ce type de colonne il est également possible d'ajouter un filtre avec la fonction getCol et addFilter.

```
$table->getCol("user_group")->addFilter("label", "= 'Admin'");
```

3. Affinement des autres options

1. Fonctions CRUD

Lors de la création d'une fonction, il existe principalement 4 possibilités: la création (Create), la lecture (Read), la modification (Update) et la suppression (Delete). Ce framework permet de distinguer ces 4 fonctions possibilités avec ces fonctions:

```
// Operation available
public static $CREATE = 0x01;
public static $READ = 0x02;
public static $UPDATE = 0x04;
public static $DELETE = 0x08;

public function canCreate($create);
public function canRead($read);
public function canUpdate($update);
public function canDelete($delete);

public function can($flags)
```

La dernière fonction utilise les 4 premiers drapeaux. On pourra par exemple donner la fonction de création et de lecture avec:

```
$table->can(DataTable\php\Table::$CREATE | DataTable\php\Table::$READ);
```

2. Pagination & tri

De la même manière, on peut activer la pagination ou la désactiver avec les drapeaux et la fonction suivante:

```
// Style of pagination
public static $PAGINATION_FULL_NUMBERS = "full_numbers";
public static $PAGINATION_NO = “";

public function setPaginateStyle($paginate)
```

Et la fonction pour activer ou désactiver le tri:

```
public function canSort($sort);
```

Elle prends en paramètre un booléen.

3. Langue

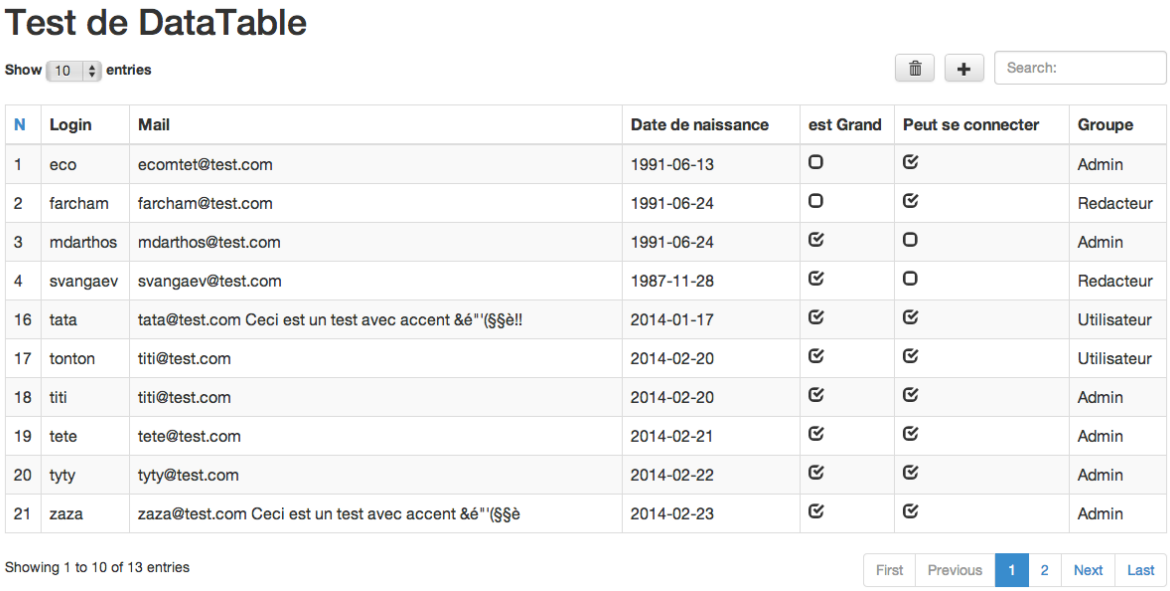
Il est également possible de personnaliser le framework en modifiant la langue avec les langues disponibles et la fonction init vu précédemment.

```
// Language available
public static $FRANCAIS = "fr";
public static $ENGLISH = "en";
```

Il est également possible de rajouter une langue en créant un nouveau fichier dans *DataTable/translations/*. Le nom du fichier se composer de cette manière nomDeLaLangue.json et le fichier se structure de cette manière:

```
{
	"sProcessing":     "Traitement...",
	"sSearch":         "Rechercher",
	"sLengthMenu":     "Afficher _MENU_ éléments",
	"sInfoThousands":  " ",
	"sInfo":           "Affichage des éléments _START_ à _END_ sur _TOTAL_ éléments",
	"sInfoEmpty":      "Aucun éléments",
	"sInfoFiltered":   "(filtré de _MAX_ éléments au total)",
	"sInfoPostFix":    "",
	"sLoadingRecords": "Chargement...",
	"sZeroRecords":    "Aucun élément à afficher",
	"sEmptyTable":     "Aucune donnée disponible dans le tableau",
	"sDragDrop":    "Glissez-Déposez une ligne pour la supprimer",
	"oPaginate": {
		"sFirst":      "Premier",
		"sPrevious":   "Précédent",
		"sNext":       "Suivant",
		"sLast":       "Dernier"
	},
	"oAria": {
		"sSortAscending":  ": Activer pour trier la colonne par ordre croissant",
		"sSortDescending": ": Activer pour trier la colonne par ordre décroissant"
	},
	"eFunctionUnkown" : "Fonction inconnue",
	"eMissingPatameters": "Paramètres non trouvés",
	"eUnserialize": "Impossible de désérialiser",
	"eUnkonwn": "Erreur inconnu",
	"eAction": "Impossible d'effectuer cette action",
	"eQuery": "Erreur dans la requête",
	"eFindCol": "Impossible de trouver la colonne",
	"eFile": "Impossible de trouver le fichier",
	"eFail": "Erreur",
	"eNone": "Succés"
}
```

4. Rendu de l'interface

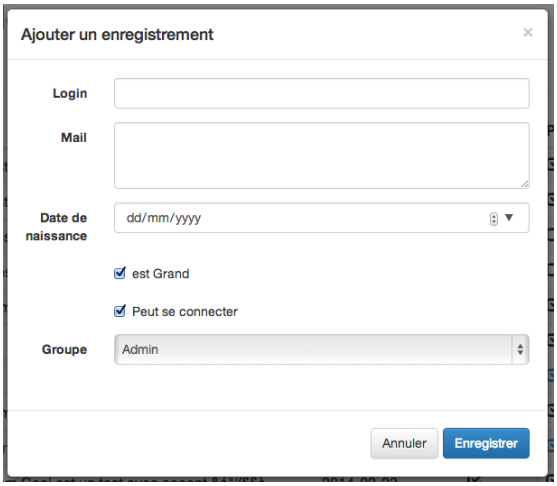


1. Modifier

Pour modifier une case, il suffit de double cliquer dessus. Un clique autre part sauvegarde la modification, une pression sur la touche échappe annule la modification.
En cas d'erreur, elle sera affiché sur la case en cours de modification.

2. Ajouter

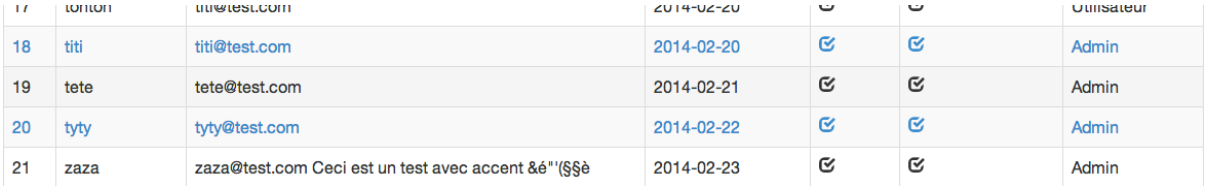
Une icône avec plus permet d'ajouter une nouvelle ligne. Une nouvelle fenêtre s'ouvre et vous permet de remplir les cases. En cliquant sur enregistrer, la nouvelle ligne s'ajoute ou une erreur est affiché sur l'icône plus.



3. Supprimer

Il existe deux manières de supprimer:

- En glissant déposant une ligne sur l'icône de poubelle.
- En sélectionnant des lignes et en cliquant sur la poubelle. Il est possible de sélectionner plusieurs lignes.

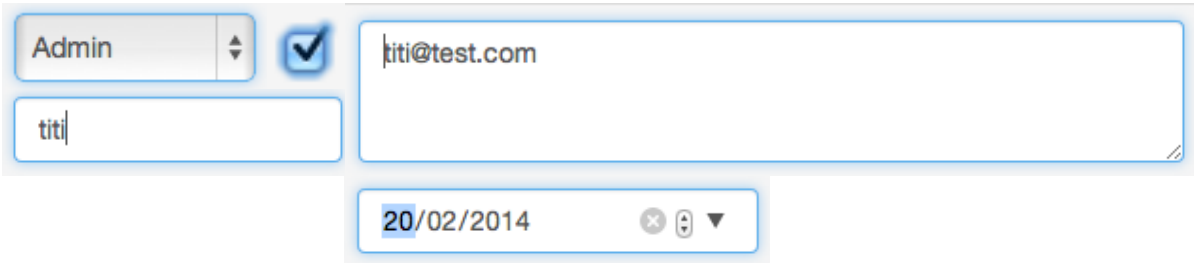


18	titi	titi@test.com	2014-02-20			Admin
19	tete	tete@test.com	2014-02-21			Admin
20	tyty	tyty@test.com	2014-02-22			Admin
21	zaza	zaza@test.com Ceci est un test avec accent &é"'(§§è	2014-02-23			Admin

En cas d'erreurs, elle sera affiché sur l'icône de la poubelle.

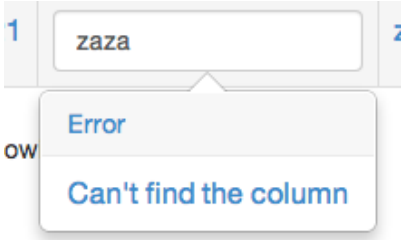
4. Introspection sur le type des champs

Ce framework détecte également automatiquement le type des champs de la base. Pour cela, une requête est exécutée et récupère le type de chaque colonne. L'affichage se transforme suivant le type.



5. Affichage des erreurs

Les erreurs s'affichent sous forme de bulle comme ceci:



6. Outils de détection des erreurs

Ce projet, encore en phase de test, peut présenter des erreurs. Il est possible de les signaler ici: https://github.com/doudou34/DataTable/issues. Une correction sera apportée le plus rapidement possible.

Il peut aussi s'agir d'une erreur de votre part suite à un manque de documentation, pour vous aider, voici deux méthodes.

Les erreurs classiques, en PHP. Pour que PHP affiche les erreurs il est nécessaire de mettre la variable display_errors à "On". Deux manières, directement en PHP, ici http://php.net/manual/fr/errorfunc.configuration.php, ou dans le fichier php.ini. Pour trouver ce fichier, la fonction phpinfo() vous aidera.

Les erreurs en javascript peuvent être détecté sous Chrome avec la console. Pour cela faites un clique droit, inspecter l'éléments. Vous devriez voir apparaitre une fenêtre en bas:

L'onglet console vous permettre de détecter les erreurs javascript. Un deuxième onglet, Réseau, vous permettra de détecter les erreurs AJAX en récupérant les fichiers demandé par AJAX, notamment le fichier ajax.php qui sert de transfert entre le PHP et le javascript dans le framework.