TableManager 1.1.0: plugin jQuery per la gestione delle tabelle

Documentazione

COS'È TABLEMANAGER?

TableManager è un plugin jQuery pensato per sviluppatori web e web designer che hanno bisogno di lavorare con le tabelle html. Questo plugin è stato progettato e creato per facilitare la gestione delle tabelle, spesso usate, ad esempio, nei back-end dei siti oppure nel mostrare a video i risultati di qualche query. Grazie ad esso infatti sarà molto semplice rendere le tabelle ordinabili per colonna, ma anche filtrarle tramite un *input text* e dividerle in pagine.

La sua installazione è molto semplice ma è consigliata una conoscenza di base di Html e JavaScript per il suo utilizzo completo. In ogni caso è stato creato un **tutorial** proprio per facilitarne l'installazione e la sua personalizzazione.

Le sue macro-funzioni sono:

- Ordinare le colonne
- Filtrare la tabella
- Gestire numeri di righe e paginazione

Oltre a queste funzioni sarà possibile anche correggere eventuali errori di ordinamento della tabella dovuto a una colonna che contiene date, disabilitare colonne dall'ordinamento, effettuare un debug…

NOVITÀ

1) Nuova classe aggiunta alla tabella: tablePagination, tableFilterBy;
2) Nuova classe per l'elemento <th>: disableFilterBy;
3) Nuova opzione per disabilitare il filtro su una o più colonne: disableFilterBy: [];
4) Nuovo data-attribute per mostrare le righe in una singola tabella: data-showrows="[10,20,100]".

UTILIZZO DEL PLUGIN

Utilizzo di base

Con una semplice riga di codice sarà possibile rendere le colonne della propria tabella ordinabili secondo il criterio "numerico-alfabetico", ordinando quindi prima i numeri poi le lettere.

```
$('#mia-tabella').tablemanager();
```

In questo modo dunque abbiamo inizializzato il plugin per ottenere la sua funzione di base di ordinamento.

Opzioni disponibili

Prima visualizzazione: firstSort

Si può decidere quale sarà l'ordinamento della prima visualizzazione, per esempio voglio che appena stampata a video la tabella risulti ordinata per la colonna 2 e a seguire per la colonna 3. Sarà anche possibile scegliere se le colonne devono essere ordinate in modo ascendente ("0" o "asc") o discendente ("1" o "desc").

Di seguito l'esempio di ordinamento iniziale per la colonna 2 in modo ascendente e la colonna 3 in modo discendente.

```
$('#mia-tabella').tablemanager({
     firstSort: [[2,0],[3, 'desc']]
});
```

Disabilitare una colonna: disable

Se si vuole disabilitare una o più colonne sarà possibile farlo con l'opzione "disable". Si può anche decidere di disabilitare l'ultima colonna, pur senza conoscere l'esatto numero di colonne nella tabella, in questi casi basterà scrivere "-1" o "last". Nell'esempio che segue disabilitiamo la prima e l'ultima colonna:

```
$('#mia-tabella').tablemanager({
     disable: [1,"last"]
});
```

Per disabilitare singolarmente una colonna sarà possibile anche agire direttamente sulla tabella stessa. Questo grazie a due opzioni:

- la classe disableSort
- l'attributo data-tablemanager

Per disabilitare una colonna basterà aggiungere al <th> corrispondente alla colonna di cui disabilitare l'ordinamento la classe "disableSort":

```
<tr>
     <th class="disableSort">Number</th>
     <th>First Name</th>
     <th>Last Name</th>
     <th>Date</th>
     <th>Points</th>
     <th>Controls</th>
</tr>
```

In alternativa, se si preferisce utilizzare l'attributo data- al posto della classe, basterà aggiungere al <th> il data-attribute "data-tablemanager= 'disable'":

```
<tr>
     <th data-tablemanager="disable">Number</th>
     <th>First Name</th>
     <th>Last Name</th>
     <th>Date</th>
     <th>Points</th>
     <th>Controls</th>
</tr>
```

In questo modo potrò gestire la singola colonna della singola tabella e renderla non ordinabile.

Colonna in formato data: dateFormat

Per correggere gli errori di ordinamento di una colonna che contiene una data si può dichiarare il formato di data della colonna attraverso l'opzione "dateFormat". Attraverso quest'opzione è possibile dichiarare il numero di colonna da formattare e il formato da noi utilizzato al momento della stampa a video. E' facile infatti che le date che noi riportiamo in tabella siano formattate in modo simile a gg-mm-aaaa, queste verrebbero normalmente ordinate quindi per giorno, mese, anno, il che vorrebbe dire che in ordine ascendente la data 20-03-1978 verrebbe dopo la data 12-03-2011, pur essendo ovviamente la seconda la data più recente. L'opzione dateFormat ordina la colonna nel modo corretto (per anno, mese, giorno).

Per esempio, io voglio dichiarare che la terza colonna è una data in formato gg-mm-aaaa e la quarta è sempre una data, ma in formato gg/mm/aaaa:

```
$('#mia-tabella').tablemanager({
     dateFormat[[3,"dd-mm-yyyy"],[4,"dd/mm/yyyy"]]
});
```

Un'alternativa che permetterebbe di gestire la singola colonna della singola tabella per dichiarare che contiene una data ed esplicitarne il formato è quello di usare il data-attribute "data-tablemanager".

L'attributo data-tablemanager deve essere formattato in questo modo:

data-tablemanager = '{"dateFormat":"dd/mm/yyyy"}'

E' **importante** ricordare che le parentesi graffe vanno incluse tra apici ', mentre i singoli attributi tra virgolette ". In caso contrario la lettura di formattazione della colonna come data non funzionerà.

Esempio:

```
<tr>
     <th class="disableSort">Number</th>
     <th>First Name</th>
     <th>Last Name</th>
     <th data-tablemanager='{"dateFormat":"dd-mm-yyyy"}'>Date</th>
     <th>Points</th>
     <th>Controls</th>
</tr>
```

Aggiungere filtro alla tabella: appendFilterby

E' possibile aggiungere un filtro sopra la tabella che permette di mostrare solo le righe che contengono la stringa immessa nel campo apposito. L'utente potrà selezionare dalla tendina la colonna da usare come filtro e il campo input per scrivere ciò che desidera trovare all'interno della tabella. Per utilizzare questo filtro basterà aggiungere l'opzione appendFilterby (che può avere solo valori true o false) e dargli valore true.

```
$('#mia-tabella').tablemanager({
```

```
    appendFilterby: true
});
```

Un altro modo per aggiungere un filtro alla tabella è quello di aggiungere la classe **tableFilterBy** alla tabella.

E' anche possibile disabilitare il filtro su una o più colonne tramite opzioni. Nell'esempio seguente disabilito il filtro sulla prima e l'ultima colonna.

```
$('#mia-tabella').tablemanager({
    disableFilterBy: [1, "last"]
});
```

Si può disabilitare anche il filtro sulla singola colonna tramite la classe "disableFilterBy" attribuita al <th>.

```
<tr>
    <th>Number</th>
    <th>First Name</th>
    <th>Last Name</th>
    <th>Date</th>
    <th>Points</th>
    <th class="disableFilterBy">Controls</th>
</tr>
```

Tramite una classe attribuita alla tabella è possibile disabilitare l'intera tabella dall'uso del filtro se necessario. Per farlo do la classe all'elemento <table>.

```
<table id="mia-tabella" class="disableFilterBy">
```

Aggiungere un'opzione per far scegliere quante righe mostrare della tabella: showrows

L'opzione showrows permette di aggiungere sopra alla tabella un menu a tendina con diverse opzioni di numeri di righe da mostrare. In tal modo l'utente potrà scegliere se mostrare 10, 20, 100… righe della tabella. L'opzione si aspetta un array di valori, che saranno poi i numeri di righe che la tabella mostrerà dopo la scelta dal menu a tendina. Nell'esempio il menu a tendina permetterà di mostrare i seguenti numeri di righe: 5, 10, 20, 50, 100.

Solitamente questa opzione è accompagnata all'opzione "pagination".

```
$('#mia-tabella').tablemanager({
    showrows: [5,10,20,50,100]
});
```

Un modo alternativo per aggiungere uno strumento showrows è quello di usare un data-attribute per la singola tabella: data-showrows. I valori contenuti nell'attributo dovranno essere un array: [num, num, num].

Nell'esempio permetto di mostrare 5, 20, 100 righe.

```
<table id="mia-tabella" data-showrows="[5,20,100]">
```

Aggiungere la paginazione: pagination

Attraverso questa opzione sarà possibile aggiungere i controlli per dividere la tabella in pagine e per navigare tra esse. Creerà dei pulsanti per mostrare ultima e prima pagina, precedente e successiva e la pagina numero n.

Solitamente questa opzione è accompagnata all'opzione "showrows".

```
$('#mia-tabella').tablemanager({
     pagination: true
});
```

Un altro modo per aggiungere un filtro alla tabella è quello di aggiungere la classe **tablePagination** alla tabella.

Tradurre le stringhe di default: vocabulary

Quest'opzione permette di tradurre (o cambiare) le stringhe che il sistema mostra di default in inglese. Le opzioni che mostrano testo traducibile sono:

- appendFilterby (voc_filter_by, voc_type_here_filter)
- showrows (voc_show_rows)

Nell'esempio traduco tutte le stringhe traducibili delle opzioni sopra riportate.

```
$('#mia-tabella').tablemanager({
     vocabulary: {
                    voc_filter_by: 'Filtra per',
                    voc_type_here_filter: 'Scrivi qui per
filtrare...',
                    voc_show_rows: 'Mostra righe'
               }
});
```

Modalità di sviluppo: debug

L'opzione debug è molto utile per capire cosa non va o cosa abbiamo sbagliato durante la scrittura delle opzioni e la loro messa a punto. Lo strumento di debug non mostrerà niente a video, ma mostrerà gli errori

nella console del browser (tramite un classico console.log). E' una variabile booleana (true o false) e naturalmente per attivare il debug basterà scrivere:

```
$('#mia-tabella').tablemanager({
     debug: true
});
```

Utilizzo completo

E' importante ricordare che ogni opzione deve essere separata da una virgola, ciò vale soprattutto se si vogliono utilizzare più opzioni contemporaneamente.

Qui sotto potete trovare l'esempio di utilizzo con tutte le opzioni possibili:

```
$('#mia-tabella').tablemanager({
     firstSort: [[3,0],[2,0],[1,'desc']],
     disable: ["last"],
     appendFilterby: true,
     dateFormat: [[4,"dd-mm-yyyy"]],
     debug: false,
     vocabulary: {
          voc_filter_by: 'Filtra per',
          voc_type_here_filter: 'Scrivi qui per filtrare...',
          voc_show_rows: 'Mostra righe'
     },
     pagination: true,
     showrows: [5,10,50,100],
     disableFilterBy: [1, "last"]
});
```

Spieghiamolo un attimo.

- Inizialmente la tabella verrà ordinata con il seguente ordine di colonne: 3 ascendente, 2 ascendente, 1 discendente;
- l'ultima colonna è disabilitata;
- attivo il filtro per colonna;
- la quarta colonna contiene date con il formato gg-mm-aaaa;
- modalità di debug inattiva;
- le stringhe sono tradotte con il testo tra virgolette;
- abilito la paginazione;
- abilito il mostra righe con le seguenti opzioni: 5, 10, 50, 100;
- disabilito il filtro sulla prima colonna e sull'ultima.

Per maggiori informazioni seguire il **Tutorial**.

OPZIONI, CLASSI E ATTRIBUTI

Tabella opzioni e valori

Opzione	**Uso e valori**
firstSort	[[*colonna*,*ordinamento*],[*colonna*,*ordinamento*]] *colonna* = numero intero *ordinamento* = 0 o 'asc', 1 o 'desc'
disable	[*colonna*,*colonna*] *colonna* = numero intero o 'last'
dateFormat	[[*colonna*,'*formato-data*']] *colonna* = numero intero '*formato-data*' = dd-mm-yyyy, mm-dd-yyyy, dd/mm/yyyy, mm/dd/yyyy
appendFilterby	*Booleano* true o false
pagination	*Booleano* true o false
showrows	[*valore*,*valore*,*valore*] *valore* = numeri interi

vocabulary	{
	stringa : '*traduzione*',
	stringa : '*traduzione*'
	}
	
	stringa = *voc_filter_by* (Prima della tendina di scelta della colonna per cui filtrare), *voc_type_here_filter* (Placeholder dell'input del filtro), *voc_show_rows* (Prima della tendina di scelta di numeri di righe da mostrare)
	traduzione = stringa di testo

disableFilterBy	[*colonna*,*colonna*] *colonna* = numero intero o 'last'
debug	*Booleano* true o false

Classi e data-attributi utili

Classe/Attributo	**Uso e/o valori**
disableSort	Da aggiungere alla colonna da disabilitare singolarmente.
tablePagination	Da aggiungere alla tabella a cui assegnare la paginazione.
tableFilterBy	Da aggiungere alla tabella a cui assegnare il filtro.

disableFilterBy	Da aggiungere alla colonna su cui disabilitare il filtro.
data-tablemanager="disable"	Da aggiungere alla colonna da disabilitare singolarmente.
data-tablemanager= '{"dateFormat":"formato della data"}'	Da aggiungere alla colonna contenente date. *dateFormat*: non è modificabile. *formato della data*: indica il formato della data che si è utilizzato nella colonna. I formati ammessi sono: ddmmyyyy, mmddyyyy, dd-mm-yyyy, mm-dd-yyyy, dd/mm/yyyy, mm/dd/yyyy
data-showrows="[int,int,int]"	Da aggiungere alla tabella per indicare le righe da mostrare nella tendina.

Tutorial

In questo semplice tutorial vedremo passo per passo come installare il plugin nel nostro progetto e come farlo funzionare, quindi vedremo come personalizzarlo affinché renda al meglio.

In questo tutorial supporremo di avere una tabella con id “mia-tabella”.

```
<table id="mia-tabella">
     <thead>
          <tr>
               <th>Number</th>
               <th>First Name</th>
               <th>Last Name</th>
               <th>Date</th>
               <th>Points</th>
               <th>Controls</th>
          </tr>
     </thead>
     <tbody>
          … Righe …
     </tbody>
</table>
```

E’ fondamentale che la tabella sia formattata correttamente, come quella riportata qui sopra:

l’elemento table deve contenere thead (dove andranno le righe con le intestazioni o th) e tbody dove si trova il contenuto vero e proprio della tabella (td).

Requisiti minimi

- jQuery versione da 1.8 in poi (non testato con jQuery 2)
- tableManager.js

Installazione

1. Per prima cosa andrà installata (se non si è già fatto) la libreria jQuery nel nostro progetto. Per farlo basterà aggiungere la seguente stringa prima della fine del </body>:
   ```
   <script
   src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.2/jquer
   y.min.js"></script>
   ```
   Questa riga vi permetterà di utilizzare jQuery senza avere il file nello stesso spazio del vostro progetto. In alternativa dovrete andare a scaricare l’ultima versione dal sito ufficiale: https://jquery.com/ . Una volta scaricato inserite il file .js o il .min.js in una cartella a vostra scelta all’interno del vostro progetto o del vostro sito (ad esempio in una cartella chiamata “js”) quindi aggiungete la seguente stringa prima della fine del </body>:
   ```
   <script src="percorso-fino-al-file/js/jquery.min.js"></script>
   ```
   Solitamente i file javascript si includono alla fine dei documenti per rendere più veloce il caricamento della pagina.
   Per approfondire l’installazione potrete dare un’occhiata alla documentazione ufficiale jQuery.
2. Ora, se non lo avete già fatto, scaricate il **plugin tableManager** da qui: http://www.stonewebdesign.it/table-manager-plugin-jquery-per-la-gestione-delle-tabelle
   Prendete il file tableManager.js e inseritelo in una cartella del vostro sito o progetto (ad esempio una cartella chiamata “js”). A questo punto inserite la seguente stringa **sotto** quella di inclusione di

jQuery. Attenzione! E' fondamentale che essa sia dichiarata sotto quella di jQuery altrimenti il browser restituirà degli errori.

```
<script src="percorso-fino-al-file/js/tableManager.js"></script>
```

Per capire meglio come includere un file .js date un'occhiata alla guida di Html.it: http://www.html.it/pag/16048/i-percorsi-assoluti-e-relativi/

Utilizzo di base

Una volta terminata l'installazione del plugin si può passare al suo utilizzo e alla sua inizializzazione.

Nella nostra pagina, in cui compare la tabella, dovremo scrivere il seguente codice javascript, **sotto** l'inclusione del file tableManager.js, e prima della fine del tag </body>:

```
<script type="text/javascript">
$('#mia-tabella').tablemanager();
</script>
```

E' **fondamentale** che questo codice javascript sia scritto sotto il file tableManager.js.

Analizziamo la stringa:

`<script type="text/javascript">` apro il codice javascript all'interno del file html o php;

`$('#mia-tabella').tablemanager();` dichiaro che l'elemento con id = "mia-tabella" utilizzerà la funzione principale tablemanager(). In tal modo inizializzo il plugin.

`</script>` chiudo il tag script.

Questo codice va bene naturalmente per inizializzare il nostro plugin all'interno di un documento html o php. Se invece preferisco avere un file js separato non dovrò far altro che creare un file javascript, ad esempio script.js, e inserirlo nel mio progetto, per esempio nella classica cartella denominata "js". A quel punto dovrò includere nel mio documento html o php il mio file esterno javascript:

```
<script src="percorso-fino-al-file/js/script.js"></script>
```

E invece nel mio file script.js personalizzato andrò ad inserire semplicemente la seguente stringa:

```
$('#mia-tabella').tablemanager();
```

Questa spiegazione vale se si vuole fare un utilizzo di base del plugin, senza scegliere opzioni e senza personalizzazioni. Per le opzioni e le personalizzazioni disponibili seguire il tutorial **Utilizzo completo**.

Utilizzo completo

Per utilizzare al meglio questo plugin per la gestione delle tabelle html sono state messe a disposizione dello sviluppatore alcune opzioni per correggere possibili errori, organizzare, personalizzare la propria tabella. Per la lista delle opzioni disponibili leggere la documentazione completa.

Una volta inizializzato il plugin attraverso il tutorial **Utilizzo di base** si potrà decidere di voler ordinare di default una tabella al momento del caricamento della pagina. Qui entra in gioco l'opzione "firstSort". Per

aggiungere opzioni alla funzione di base “tablemanager” bisogna inserirle all’interno delle sue parentesi tonde e comprese tra due ulteriori parentesi graffe (vedi esempio). Quindi scrivere:

```
$('#mia-tabella').tablemanager({
     firstSort: [[3,0],[2,'desc']]
});
```

Le parentesi graffe { e } servono per delimitare il gruppo di opzioni che si vogliono dare alla tabella.

firstSort: [[3,0],[2,'desc']] → in questo modo diciamo al plugin: appena la tabella viene caricata ordina per colonna numero 3 in modo ascendente, se ci sono valori uguali ordina per colonna 2 in modo discendente.

Per aggiungere altre opzioni queste devono essere separate da virgole.

Aggiungiamo un’opzione che disabiliti la prima e l’ultima colonna:

```
$('#mia-tabella').tablemanager({
     firstSort: [[3,0],[2,'desc']],
     disable: [1,'last']
});
```

A questo punto ci accorgiamo che la colonna 4 è una data (separata magari da “/”), quindi formattiamola per far sì che l’ordinamento prenda in considerazione la data nell’ordine corretto.

```
$('#mia-tabella').tablemanager({
     firstSort: [[3,0],[2,'desc']],
     disable: [1,'last'],
     dateFormat[[4,'dd/mm/yyyy']]
});
```

Ora che tutti gli ordinamenti funzionano passiamo al resto. Aggiungiamo un’opzione per filtrare e una per mostrare un certo numero di righe e dividere la tabella in pagine.

```
$('#mia-tabella').tablemanager({
     firstSort: [[3,0],[2,'desc']],
     disable: [1,'last'],
     dateFormat[[4,'dd/mm/yyyy']],
     appendFilterby: true,
     pagination: true,
     showrows: [5,10,50,100]
});
```

appendFilterby → dandogli valore true attiviamo la funzione “Filtra per colonna”.

pagination → con valore true attiviamo la paginazione.

showrows → diamo un gruppo di valori che saranno le righe da mostrare.

A questo punto decidiamo che magari vogliamo tradurre in italiano le stringhe di testo presenti. Questo si può fare tramite l'opzione vocabulary.

```
$('#mia-tabella').tablemanager({
     firstSort: [[3,0],[2,'desc']],
     disable: [1,'last'],
     dateFormat[[4,'dd/mm/yyyy']],
     appendFilterby: true,
     pagination: true,
     showrows: [5,10,50,100],
     vocabulary: {
          voc_filter_by: 'Filtra per',
          voc_type_here_filter: 'Scrivi qui per filtrare...',
          voc_show_rows: 'Mostra righe'
     }
});
```

In tal modo abbiamo tradotto le stringhe del filtro e del "Mostra righe".

Se qualcosa non funziona un'opzione molto comoda, soprattutto per i più esperti, può essere quella di debug.

```
$('#mia-tabella').tablemanager({
     firstSort: [[3,0],[2,'desc']],
     disable: [1,'last'],
     dateFormat[[4,'dd/mm/yyyy']],
     appendFilterby: true,
     pagination: true,
     showrows: [5,10,50,100],
     vocabulary: {
          voc_filter_by: 'Filtra per',
          voc_type_here_filter: 'Scrivi qui per filtrare...',
          voc_show_rows: 'Mostra righe'
     },
     debug: true
});
```

Attivando la funzione di debug il plugin potrà aiutarti a trovare errori o incompatibilità tramite la console del tuo browser.

Personalizzazione: Stile e CSS

La personalizzazione dello stile è stato reso semplice grazie all'aggiunta di id e classi specifiche per gli elementi toccati dal plugin.

Qui di seguito le classi che vengono assegnate agli elementi con rispettiva spiegazione:

sorterHeader → è la classe di base della testata della tabella che permette l'ordinamento. Questa classe quindi è presente solo se la colonna è stata resa ordinabile.

disableSort → indica che quella colonna non è ordinabile perché l'abbiamo in qualche modo disabilitata.

sortingAsc → indica che la colonna è ordinata in ordine ascendente.

sortingDesc → indica che la colonna è ordinata in ordine discendente.

for_filter_by → classe del filtro che contiene select e input per filtrare la tabella.

for_numrows → classe dello strumento che mostra un certo numero di righe.

pagesControllers → racchiude i controlli delle pagine della tabella.

pagecontroller → comprende tutti i pulsanti utili per la paginazione.

currentPage → indica la pagina corrente.

pagecontroller-f → indica il pulsante che porta alla prima pagina.

pagecontroller-p → indica il pulsante che porta alla pagina precedente.

pagecontroller-num → indica il pulsante che porta alla pagina numerata.

pagecontroller-n → indica il pulsante che porta alla pagina successiva.

pagecontroller-l → indica il pulsante che porta all'ultima pagina.

Qui invece sono riportati gli id disponibili, utili per la personalizzazione:

for_filter_by → filtro che contiene select e input per filtrare la tabella.

for_numrows → strumento che mostra un certo numero di righe.

pagesControllers → racchiude i controlli delle pagine della tabella.

Esempio classico di utilizzo delle classi per la personalizzazione.

se voglio aggiungere delle icone (Font Awesome per esempio) ai miei elementi <th>, intanto scarico Font Awesome e lo installo all'interno del mio progetto, quindi includo i suoi file. Quando è pronto per l'utilizzo posso scrivere per esempio questo:

```
.tablemanager th.sorterHeader {
    cursor: pointer;
}
.tablemanager th.sorterHeader:after {
    content: " \f0dc";
```

```
        font-family: "FontAwesome";
    }
    /*Style sort desc*/
    .tablemanager th.sortingDesc:after {
        content: " \f0dd";
        font-family: "FontAwesome";
    }
    /*Style sort asc*/
    .tablemanager th.sortingAsc:after {
        content: " \f0de";
        font-family: "FontAwesome";
    }
```

In questo modo quindi ho fatto in modo che al di sopra delle colonne ordinabili esca fuori la manina (cursor: pointer). Quindi ho messo le classiche freccine su e giù per indicare visivamente che quella colonna è ordinabile, la freccia in giù se l'ordinamento è discendente e in su se è ascendente.