Главная › Блоги › Lertion's блог

[Обучение Ext JS] :: Часть 3. Создание форм в Ext

Фев 18 2010**ComboBox**
Под *ComboBox*, или, как это называется в HTML, *SELECT* имеют в виду выпадающее меню (очень полезный элемент формы!). С его помощью пользователь не надо нажимать много клавиш на клавиатуре (какая забота!). У *ComboBox* в **Ext** есть очень много способов применения, и еще больше настроек.
Сначала, давайте сделаем комбо, используя локальные данные. Для осуществления этого нам понадобиться создать хранилище данных. Существуют разные виды хранилищ,каждое из которых предназначено для определенной ситуации. Тем не менее, для этого случая, мы воспользуемся самым простым хранилищем::

```
var genres = new Ext.data.SimpleStore({
   fields: ['id', 'genre'],
   data : [['1','Comedy'],['2','Drama'],['3','Action']]
});
```

Подобно другим полям нашей формы мы добавим их в настройки *items*. Несколько других настроек необходимы когда мы запускаем и настраиваем комбинированное окно. *Хранилище (store)* очевидный источник данных для нашего окна. Еще нам понадобятся: *mode*, который определяет откуда поступают данные - из локального или удаленного источника и *displayField*, определяющий какую колонку данных следует выводить на экран:

```
{
  xtype: 'combo',
  name: 'genre',
  fieldLabel: 'Genre',
  mode: 'local',
  store: genres,
  displayField:'genre',
  width: 120
}
```

Это даст нам комбинированное окно, которое использует локальные данные (что неплохо для маленьких списков, или тех списков которые редко изменяются). Что происходит когда мы обращаемся к списку?

Комбинированное окно под управлением базы данных
Самое большое изменение находится на серверной стороне — получить ваши данные и отформатировать их для **JSON**, чтобы комбинированное окно могло начать их использовать. Каким бы не был язык сервера нам потребуется библиотека **JSON** для 'шифрования' данных. Если мы пользуемся **PHP** 5.1 или выше, то библиотека уже встроена.

ВАЖНО: *для того чтобы проверить какая версия* ***PHP*** *у вас установлена можно выполнить команду в терминальном окне или запустить простой код. Если у вас есть доступ к командной строке запустите php –v для проверки версии. Или, в противном случае запустите скрипт содержащий строку <?php phpinfo(); ?>.*

Вот что нам потребуется для создания нашей базы данных **JSON** используя **PHP** 5.1 или выше:

Исходный код примера:

Исходный код примера:

```
<?php
// connection to database goes here
$result = mysql_query('SELECT id, genre_name FROM genres');
If (mysql_num_rows($result) > 0) {
   while ($obj = mysql_fetch_object($result)) {
      $arr[] = $obj;
   }
}
Echo '{rows:'.json_encode($arr).'}';
?>
```

Когда мы используем удаленный источник, то должно произойти еще насколько вещей. прежде всего хранилище должно знать в каком формате находятся данные. Мы задаем это используя устройство считывания данных—в нашем случае это **JSON Reader**.

Исходный код примера:

Исходный код примера:

```
var genres = new Ext.data.Store({
   reader: new Ext.data.JsonReader({
      fields: ['id', 'genre_name'],
      root: 'rows'
   }),
   proxy: new Ext.data.HttpProxy({
      url: 'data/genres.php'
   })
});
```

Первое значение устройства считывания данных, это объект содержащий настройки для нашего считывателя. Оно уточняет какие поля следует смотреть и где находятся корневой элемент. Список полей это просто массив названий поля. Обратим внимание на то, что мы выпустили sort_order. Это поле будет недоступно для нашего набора данных. Нашим корневым каталогом является элемент, содержащий наш массив данных. В данном случае это ряды, но вы с легкостью можете заменить их на все. что вам заблагорассудится:

Исходный код примера:

Исходный код примера:

```
{rows:[
   {
      "id":"1",
      "genre_name":"Comedy",
      "sort_order":"0"
   },{
     "id":"2",
     "genre_name":"Drama",
     "sort_order":"1"
  },{
     // snip...//
  }]
}
```

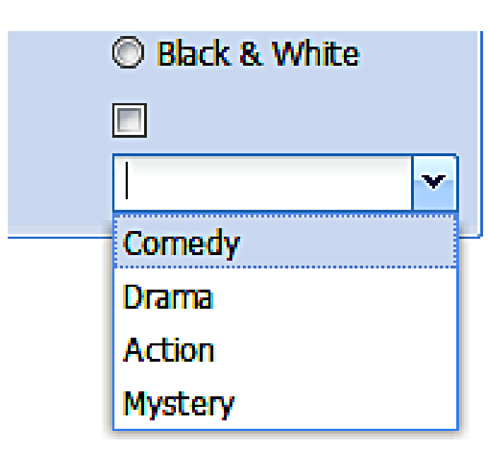
У нас так же есть настройки для прокси (обычно ведь HTTP-Proxy забирает данные того же домена). Это самый распространенный способ, но существует еще ScriptTagProxy который может быть использован для получения данных с другого домена. Все что нужно, это вписать в прокси URL откуда следует брать данные

ВАЖНО: *Когда мы задаем 'proxy',мы используем* ***AJAX****. для этого необходимо чтобы у вас был запущен веб-сервер, в противном случае* ***AJAX*** *работать не будет. исполнение кода из файловой системы в веб-браузере не сработает.*

Давайте отправим запрос на загрузку функции в конец, таким образом данные будут загружены в наше комбинированное окно прежде чем пользователь начнет с ним что-то делать.

```
genres.load();
```

Это даст нам комбинированное окно заполненное из базы данных и выглядящее следующим образом:



Другой способ предварительно загружать данные из хранилища - присвоить autoLoad значение true:

Исходный код примера:

Исходный код примера:

```
var genres = new Ext.data.Store({
   reader: new Ext.data.JsonReader({
      fields: ['id', 'genre_name'],
      root: 'rows'
   }),
   proxy: new Ext.data.HttpProxy({
      url: 'data/genres.php'
   }),
   autoLoad: true
});
```

TextArea и HTMLEditor
Мы хотим добавить поле ввода текста в нашу форму и для этого у **Ext** есть несколько способов. Мы можем использовать стандартные textarea, с которыми знакомы еще по **HTML**, или прибегнуть к помощи поля HTMLEditor, которое даст нам богатейшие возможности по редактированию текста:

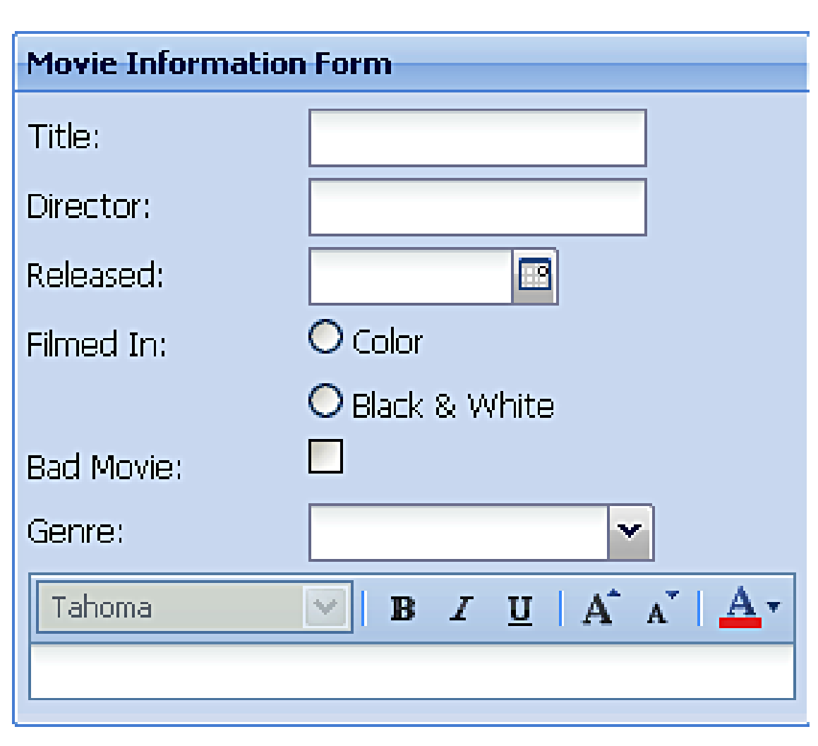
- *textarea:* Схож с обычным полем textarea **HTML**
- *htmleditor:* Хороший текстовый редактор с кнопками для часто выполняемых задач.

Если мы выставили значение hideLabel на true и почистили разделители, то получим
textarea, которая занимает всю ширину панели нашей формы. Да и выглядит мило:

```
{
  xtype: 'textarea',
  name: 'description',
  hideLabel: true,
  labelSeparator: '',
  height: 100,
  anchor: '100%'
}
```

Изменяя xtype так как показано ниже мы получим относительно простой **HTML** редактор со встроенными установками для шрифта, его размера, цвета, курсив и всего прочего. Это первый компонент **Ext** использованный нами, который требует предварительной инициализации компонента QuickTips.

```
{
  xtype: 'htmleditor',
  name: 'description',
  hideLabel: true,
  labelSeparator: '',
  height: 100,
  anchor: '100%'
}
```



Отслеживание событий поля формы
при помощи Ext довольно легко отслеживать действия пользователей, будь то нажатия на элементы формы, или клавиш.
Общепринятым считается отслеживать нажатие клавиши Enter, и последующее применение формы. Взглянем как это осуществить:

Исходный код примера:

Исходный код примера:

```
{
 xtype: 'textfield',
 fieldLabel: 'Title',
 name: 'title',
 allowBlank: false,
 listeners: {
   specialkey: function(f,e){
     if (e.getKey() == e.ENTER) {
      movie_form.getForm().submit();
     }
   }
 }
}
```

Specialkey вызывается всякий раз, когда нажимается клавиша имещая отношение к навигации. Этот слушатель так же вызывается всякий раз при нажатии клавиш со стрелками, Tab, Esc, и так далее. Поэтому, прежде чем двигаться дальше, нам надо проверить и убедиться что он срабатывает именно на клавишу Enter.
Теперь форма будет отправлена только при нажатии Enter.

СобытияComboBox
Опыт показывает что комбинированному окну надо присоединять события. Возьмем наше окно с жанрами и добавим к нему слушатель, который будет запускаться когда пункт в списке будет выбран пользователем.
Сначала добавим пустой предмет в наш список и назовем его New Genre:

Исходный код примера:

Исходный код примера:

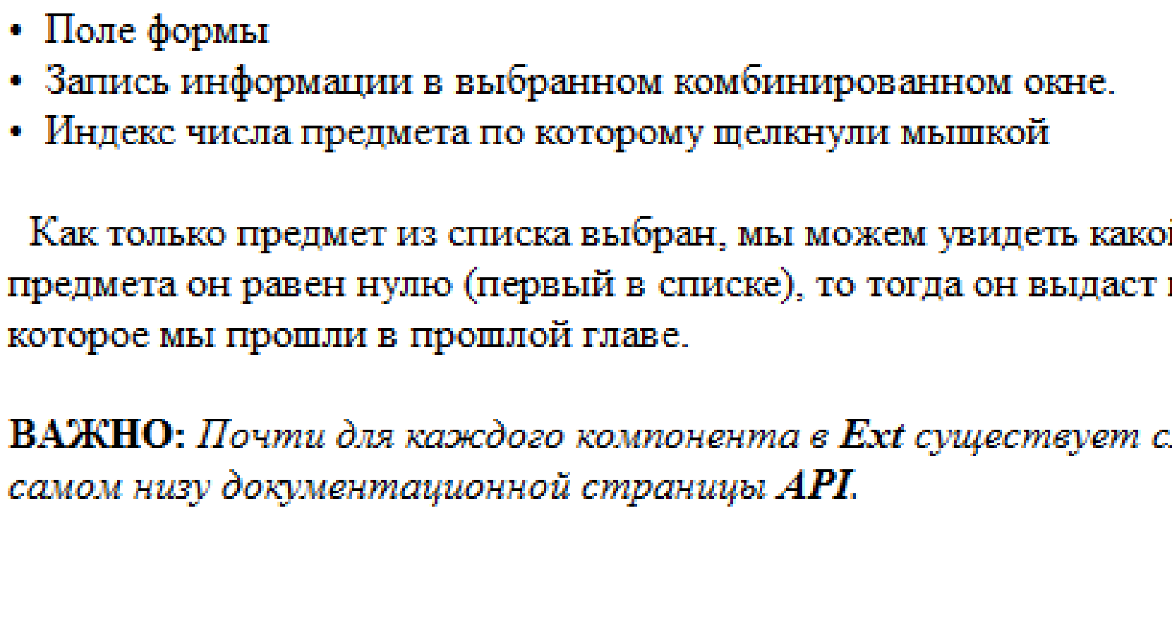
```
var genres = new Ext.data.SimpleStore({
   fields: ['id', 'genre'],
   data : [
      ['0','New Genre'],
      ['1','Comedy'],
      ['2','Drama'],
      ['3','Action']
   ]
});
```

Затем добавляем слушатель в наше окно:

Исходный код примера:

Исходный код примера:

```
 xtype: 'combo',
 name: 'genre',
 fieldLabel: 'Genre',
 mode: 'local',
 store: genres,
 displayField:'genre',
 width: 130,
 listeners: {
   select: function(f,r,i){
     if (i == 0){
       Ext.Msg.prompt('New Genre','Name',Ext.emptyFn);
     }
   }
 }
}
```



Genre: New Genre
Name: Romantic Comedy

Слушатель ждет события, а потом запускает заранее прописанную функцию. Каждый слушатель имеет свой набор переменных которые передаются функции (для большей информации по ним смотрите **API**).
Для выбранного события наша функция проверяет три вещи:

- Поле формы
- Запись информации в выбранном комбинированном окне.
- Индекс числа предмета по которому щелкнули мышкой

Как только предмет из списка выбран, мы можем увидеть какой именно. Третий параметр нашего слушателя это индекс. Если у предмета он равен нулю (первый в списке), то тогда он выдаст пользователю окно для ввода нового жанра, используя диалоговое окно, которое мы прошли в прошлой главе.

ВАЖНО: *Почти для каждого компонента в* ***Ext*** *существует слушатель. Список событий для каждого компонента можно найти в самом низу документационной страницы* ***API****.*

- « первая
- ‹ предыдущая
- 1
- 2
- 3
- 4
- следующая ›
- последняя »

- English

Голос
20