[Обучение Ext JS] :: Часть 6. Редактор таблиц

Фев 22 2010

Добавление или удаление информации в хранилище данных

Мы собираемся создать две кнопки позволяющие нам изменять содержимое хранилища данных — добавлять или удалять ряды. Давайте поместим эти кнопки на верхнюю панель инструментов (tbar) таблицы:

Исходный код примера:

Исходный код примера:

```
var grid = new Ext.grid.EditorGridPanel({
 // more config options clipped //,
 tbar: [{
   text: 'Remove Movie'
 }]
}
```

Удаление рядов таблицы из хранилища данных

Также, мы добавим на панель кнопку удаления. При нажатии на нее, пользователю будет выводиться диалоговое окно с названием фильма. Нажатием на кнопку "Yes" можно удалить выбранный ряд из базы.

Исходный код примера:

Исходный код примера:

```
{
 text: 'Remove Movie',
 icon: 'images/table_delete.png',
 cls: 'x-btn-text-icon',
 handler: function() {
   var sm = grid.getSelectionModel();
   var sel = sm.getSelected();
   if (sm.hasSelection()){
     Ext.Msg.show({
       title: 'Remove Movie',
       buttons: Ext.MessageBox.YESNOCANCEL,
       msg: 'Remove '+sel.data.title+'?',
       fn: function(btn){
         if (btn == 'yes'){
           grid.getStore().remove(sel);
         }
       }
     });
   };
 }
}
```

Title	Director	Released	Genre	Tagline
Office Space	Mike Judge	02/19/1999	Comedy	Work Sucks
Super Troopers		02/15/2002	Comedy	Altered State Police
American Beauty				... Look Closer
The Big Lebowski				The "Dude"
Fight Club	David Fincher	10/15/1999	Action	How much can you

Remove Movie

Remove The Big Lebowski?

Давайте рассмотрим происходящее поподробнее. Мы задали несколько переменных, которые будут использованы в случае выбора. Собственно сами переменные:

- sm: Мы получаем модель выбора из нашей таблицы
- sel: Модель выбора используется для возвращения ряда который был выбран
- grid.getStore().remove(sel): Отправляет хранилищу данных команду на удаление ряда. После удаления таблица будет обновлена

Локальное хранилище данных, которое находится в памяти браузера было обновлено. Но какой смысл в удалении, если потом ничего нельзя добавить? Спокойствие, только спокойствие!

Добавление ряда в таблицу

Для добавления ряда нам придется слегка извратиться. Нам нужно получить определение того, как выглядит информация для того, чтобы создать новый ряд. В чем-то похоже на создание считывателя.

Исходный код примера:

Исходный код примера:

```
var ds_model = Ext.data.Record.create([
   'id',
   'coverthumb',
   'title',
   'director',
   {name: 'released', type: 'date', dateFormat: 'Y-m-d'},
   'genre',
   'tagline',
   {name: 'price', type: 'float'},
   {name: 'available', type: 'bool'}
]);
```

Как только мы прописали это определение информации, мы с относительной легкостью можем вставить новый ряд. Для этого действия можно добавить на панель инструментов кнопку:

Исходный код примера:

Исходный код примера:

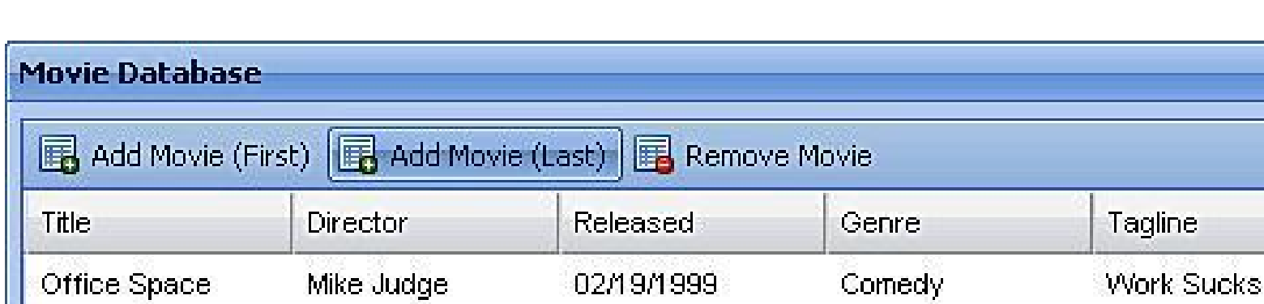
```
{
 text: 'Add Movie',
 icon: 'images/table_add.png',
 cls: 'x-btn-text-icon',
 handler: function() {
   grid.getStore().insert(
     0,
     new ds_model({
       title:'New Movie',
       director:'',
       genre:0,
       tagline:''
     })
   );
   grid.startEditing(0,0);
 }
}
```

Первый аргумент функции insert является местом, куда нужно вставить запись. У меня стоит 0 (ноль), таким образом запись появится на самом верху. Если же мы хотим чтобы новые записи появлялись внизу, надо всего-лишь получить количество рядов из нашего хранилища. Поскольку индексация номеров начинается с ноля, то приращение не нужно, поскольку количество рядов всегда будет на один больше чем последний элемент в индексе.

Исходный код примера:

Исходный код примера:

```
grid.getStore().insert(
   grid.getStore().getCount(),
   new ds_model({
     title:'New Movie',
     director:'',
     genre:0,
     tagline:''
   })
);
grid.startEditing(grid.getStore().getCount()-1,0);
```



Title	Director	Released	Genre	Tagline
Office Space	Mike Judge	02/19/1999	Comedy	Work Sucks
Super Troopers	Jay Chandrasekhar	02/15/2002	Comedy	Altered State Police
American Beauty	Sam Mendes	10/01/1999	Drama	... Look Closer
The Big Lebowski	Joel Coen	03/06/1998	Comedy	The "Dude"
Fight Club	David Fincher	10/15/1999	Action	How much can you
New Movie			New Genre	

Теперь вернемся к добавлению ряда. Второй аргумент является новым определением записи, которое может быть отправлено с несколькими значениями по умолчанию.

```
new ds_model({
  title:'New Movie',
  director:'',
  genre:0,
  tagline:''
})
```

После добавления нового ряда, мы вызываем функцию, которая будет включать редактор ячеек. Для его включения, функции необходим индекс строки и столбца.

`grid.startEditing(0,0);`

Это позволяет пользователю начать вводить название фильма сразу же после нажатия на кнопку Add Movie.

Title	Director	Released	Genre	Tagline
New Movie			New Genre	
Office Space	Mike Judge	02/19/1999	Comedy	Work Sucks
Super Troopers	Jay Chandrasekhar	02/15/2002	Comedy	Altered State Police
American Beauty	Sam Mendes	10/01/1999	Drama	... Look Closer
The Big Lebowski	Joel Coen	03/06/1998	Comedy	The "Dude"
Fight Club	David Fincher	10/15/1999	Action	How much can you

Сохранение изменений на сервер

Все, что мы делали до этого, относится к изменениям в локальном хранилище (которое, как мы, конечно, помним, находится в памяти браузера, ага). И довольно часто у нас будет появляться странное желание сохранить все внесенные изменения на сервере и изменить базу данных, файловую систему или что-то другое, но несомненно важное.
Это раздел приоткроет завесу тайны над наиболее часто используемыми требованиями таблиц, которые нужны для обновления серверной информации.

- Обновление записи
- Создание новой записи
- Удаление записи

Отправка обновлений на сервер

Ранее мы настроили и установили слушатель для пост-редакционных событий. Вот эти самые события мы и будем использовать для того,чтобы отправить изменения серверу. Ячейку за ячейкой.
Для этого нам надо знать три вещи:

- field: какое поле было изменено
- Value: каково новое значение поля
- record.id: К какому ряду принадлежит это поле

Это даст нам достаточное количество информации для того чтобы сделать внятное обновление базы данных. С сервером мы общаемся методом запроса соединения (используем **AJAX**!).

Исходный код примера:

Исходный код примера:

```
listeners: {
  afteredit: function(e){
    var conn = new Ext.data.Connection();
    conn.request({
       url: 'movie-update.php',
         params: {
         action: 'update',
         id: e.record.id,
         field: e.field,
         value: e.value
       },
       success: function(resp,opt) {
         e.commit();
       },
         failure: function(resp,opt) {
         e.reject();
       }
    });
  }
}
```

Это отправит запрос с четырьмя параметрами в форме сообщения заголовка к скрипту movie-update.php. Эти параметры, которые мы передаем как объекты настроек в соединение, отправляются через заголовки нашему серверному скрипту.
Скрипт movie-update.php должен быть сделан так, чтобы распознавать действие 'update' и считывать его id, поле, значение данных и переходить к обновлению файловой системы или базы данных (или того что вам там еще надо обновить).
Вот это доступно когда используем пост-редакционное событие:

Property	Property
grid	Ссылка на текущую таблицу
record	объект с информацией из редактируемого ряда
field	Название редактируемого поля
value	Новое, вводимое в поле, значение
originalValue	Изначальное значение в поле
row	Iиндекс редактируемого ряда — помогает его находить
column	Индекс редактируемых столбцов

Удаление данных с сервера

Когда мы хотим удалить данные с сервера, то весь процесс очень напоминает то, что мы только что с вами проделали. При удалении так же осуществляется обращение к скрипту с указанием требуемого действия.
Для удаления мы добавим еще одну кнопку на нашу панель

Исходный код примера:

Исходный код примера:

```
{
 text: 'Remove Movie',
 icon: 'images/table_delete.png',
 cls: 'x-btn-text-icon',
 handler: function() {
  var sm = grid.getSelectionModel();
  var sel = sm.getSelected();
  if (sm.hasSelection()){
   Ext.Msg.show({
     title: 'Remove Movie',
     buttons: Ext.MessageBox.YESNOCANCEL,
     msg: 'Remove '+sel.data.title+'?',
     fn: function(btn){
      if (btn == 'yes'){
       var conn = new Ext.data.Connection();
       conn.request({
         url: 'movie-update.php',
         params: {
            action: 'delete',
            id: e.record.id
         },
         success: function(resp,opt) {
            grid.getStore().remove(sel);
         },
         failure: function(resp,opt) {
            Ext.Msg.alert('Error',
            'Unable to delete movie');
         }
       });
      }
     }
   });
  };
 }
}
```

Как и в случае с редактированием, мы сделаем запрос к серверу для удаления ряда. Скрипт movie-update.php увидит что это действие удаления и сделает свое черное дело.

Сохранение новых рядов на сервере

Сейчас мы добавим еще одну кнопку отвечающую за добавление нового ряда. Она отправляет, содержащий необходимые параметры, запрос серверу и считывает id вставки из ответа сервера. Используя этот id мы можем добавить ряд которому сервер присвоит уникальный идентификатор.

Исходный код примера:

Исходный код примера:

```
{
 text: 'Add Movie',
 icon: 'images/table_add.png',
 cls: 'x-btn-text-icon',
 handler: function() {
  var conn = new Ext.data.Connection();
  conn.request({
   url: 'movies-update.php',
   params: {
     action: 'insert',
     title:'New Movie'
   },
   success: function(resp,opt) {
     var insert_id = Ext.util.JSON.decode(
       resp.responseText
     ).insert_id;
     grid.getStore().insert(0,
       new ds_model({
        id:insert_id,
        title:'New Movie',
        director:'',
        genre:0,
        tagline:''
       })
     );
     grid.startEditing(0,0);
   },
   failure: function(resp,opt) {
     Ext.Msg.alert('Error','Unable to add movie');
   }
  });
 }
}
```

Как и в случае с редактированием и удалением мы отправляем запрос серверу на вставку нового ряда. На этот раз мы рассмотрим ответ повнимательнее, ведь нам нужно найти id вставки (уникальный идентификатор ряда)чтобы отправить его назад в таблицу для того,чтобы когда мы начнем редактировать ряд можно было легко сохранить изменения.

Исходный код примера:

Исходный код примера:

```
success: function(resp,opt) {
 var insert_id = Ext.util.JSON.decode(
   resp.responseText
 ).insert_id;
 grid.getStore().insert(0,
   new ds_model({
     id:insert_id,
     title:'New Movie',
     director:'',
     genre:0,
     tagline:''
   })
 );
 grid.startEditing(0,0);
}
```

У функции нашего обработчика success есть несколько аргументов. Первый является объектом ответа, содержащий текст из нашего скрипта movie-update.php. Поскольку этот ответ приходит в формате **JSON**, мы переведем его в готовый к использованию объект и выцепим значение insert id.

```
var insert_id = Ext.util.JSON.decode(
  resp.responseText
).insert_id;
```

после вставки этого ряда в наше хранилище, то сможем пользоваться полученным insert id.

Заключение

Таблица **Ext JS** по функционалу является одной из самых продвинутых частей оформления. С резервированием пакетов Ext.data таблица может комплексно тянуть информацию с сервера. Поддержка этого уже встроена в класс таблиц. Благодаря массиву возможных настроек у нас есть возможность вывести информацию самыми разнообразными формами и подготовить к взаимодействию с пользователями (
В этой главе мы познакомились с тем, как поддержка данных таблицей предлагает знакомый многим разработчика способ управления информацией. Метод поправь-и-добавь разрешает неплохо управлять данными отправляемыми на сервер при использовании валидационной политик (это не считая способа отклонять изменения). Кроме изменения изначальной информации мы познакомили с тем, какие у таблицы есть функциональные возможности по добавлению и удалению рядов данных.
Так же мы показали как можно использовать стандартные поля форм **Ext JS** (например ComboBox) и с их помощью дать пользователю очень функциональный интерфейс. С такой сильной поддержкой ввода данных таблица является очень мощным инструментом создания приложений.
В следующей главе мы покажем как можно встроить элементы (например таблицу) в другие части экрана приложения. Разумеется с использованием того функционала структуры **Ext JS**.

- « первая
- ‹ предыдущая
- 1
- 2

- English

Голос
21