

[Обучение Ext JS] :: Часть 5. Информация в таблице

Фев 21 2010

Таблица, без сомнения один из наиболее широко используемых элементов **Ext**. У всех есть информация, которую надо предоставить конечному пользователю в удобоваримой форме. Динамическая таблица (она же просто таблица) (a.k.a.: grid) просто идеальна для такой задачи. Сама по себе эта идея довольно полезна. А **Ext** берет эту идею и делает ее гибче и просто потрясающей.

В этой главе мы:




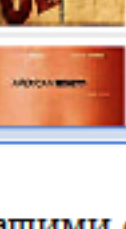
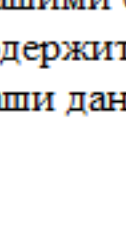
- Будем использовать GridPanel для вывода структурированных данных в виде удобной таблицы
- Получать данные с сервера (или базы данных) и отображать их в таблице
- Поработаем с событиями таблицы и посмотрим ее особенности
- Использовать некоторые продвинутые способы форматирования данных для таблиц
- Разбивать данные в таблице.

Мы рассмотрим как отличить ряд от столбца, но что более важно, мы научимся делать наши таблицы красивыми. Этого можно добиться добавляя самостоятельно обработанные ячейки, которые содержат картинки и изменить стиль, в зависимости от значений данных. Совершая все это, мы превращаем таблицу в поистине ценный инструмент, выходя далеко за ее пределы.

Что же такое таблица

Таблицы **Ext** схожи с обычными динамическими таблицами; Они также состоят из двух важных частей:

- Столбцы
- Ряды

Movie Database				
Title	Released	Genre	Price	
 The Big Lebowski Joel Coen The "Dude"	03/06/1998	Comedy	\$21.95	
 Super Troopers Jay Chandrasekhar Altered State Police	02/15/2002	Comedy	\$14.95	
 Office Space Mike Judge Work Sucks	02/19/1999	Comedy	\$19.95	
 Fight Club David Fincher How much can you know about yourself...	10/15/1999	Action	\$19.95	
 American Beauty Sam Mendes	10/01/1999	Drama	\$19.95	

Нашими столбцами будут "Название", "Дата выхода в прокат" (Released), "Жанр" и "Цена". Каждый из рядов содержит фильмы. Например, The Big Lebowski, Super Troopers, и так далее. На самом деле эти ряды и есть наши данные. Каждый ряд в таблице представляет собой запись в базе данных

Отображение структурированных данных при помощи GridPanel

Для этого нам потребуется два элемента **Ext**:

- Хранилище, которое будет действовать как база данных "в памяти", и отслеживать информацию которую мы хотим отобразить
- Панель таблицы показывающая хранимые данные в хранилище.

Перед началом, создадим, углубимся немного в терминологию, которую будем использовать поскольку команды показаться сложными (или непонятными):

- Столбцы: Относятся ко всем данным и будут содержать информацию касающуюся только только того, что следует вывести на экран. В **Ext JS**, эта информация является частью Column Model.
- Поля: Это также относится ко всем данным, но касается точных значений данных. В **Ext JS** используется в считывателе для загрузки данных.

Устанавливаем хранилище данных

Первое что нам предстоит сделать - установить наши данные, которые будут помещены в хранилище. Типы хранилища, которые доступны в **Ext** дают нам отличную возможность работать с любыми форматами, например **XML** и **JSON**, а кроме того с этими данными спокойно могут работать любые приложения **Ext**. Вне зависимости от того в каком формате у нас находятся данные (**JSON**, **XML**, массив или даже какой-то собственноручно созданный тип), они все доступны благодаря хранилищу.

По умолчанию в **Ext** доступны следующие виды хранилищ:

- Простое (Массив)
- **XML**
- **JSON**

Свое хранилище следует создавать тогда, когда ваши данные не вписываются ни в одну из вышеупомянутых категорий. На форуме **Ext JS** можно найти считыватели для таких форматов как **CSV** и **ColdFusion**.

Добавление данных в хранилище

Во время нашей первой попытки мы создадим таблицу которая будет использовать простой локальный массив данных. Все данные, использованные ниже, взяты из очень маленькой базы данных с моими любимыми фильмами, и похожа на настоящую базу (с которой мы тоже поработаем, но чуть позже). Хранилищу данных нужно две вещи. Собственно данные и ее описание. Для последнего нам нужны поля. Считыватель используется для того чтобы считывать (извините тавтологию) данные из массива, где мы и определяем поля данных содержащихся в массиве.считыватель работает как переводчик видов. Он знает как перевести строку данных в ряд данных для того чтобы **Ext JS** мог его использовать. Поместите следующий код внутри функции OnReady:

Исходный код примера:

Исходный код примера:

```
var store = new Ext.data.Store({
    data: [
        [
            1,
            "Office Space",
            "Mike Judge",
            "1999-02-19",
            1,
            "Work Sucks",
            "19.95",
            1
        ],[
            3,
            "Super Troopers",
            "Jay Chandrasekhar",
            "2002-02-15",
            1,
            "Altered State Police",
            "14.95",
            1
        ]
    ],
    //...more rows of data removed for readability...//
    reader: new Ext.data.ArrayReader({id:'id'}, [
        'id',
        'title',
        'director',
        {name: 'released', type: 'date', dateFormat: 'Y-m-d'},
        'genre',
        'tagline',
        'price',
        'available'
    ])
});
```

Если мы попытаемся посмотреть на этот код в браузере, то ничего не увидим. Это все потому, что хранилище данных - всего лишь способ загрузки и отслеживания информации. Мы загрузили базу в память браузера и теперь надо ее как-то показать пользователю.

Определение ваших данных в хранилище

Считывателю необходимо знать какие поля следует считать данными, поэтому мы должны их определить.

Поля определяются при помощи массива объектов — или, в том случае если данные надо прочитать буквально, строка уточняет название поля. В нашем примере всем полям кроме одного можно определить простыми названиями. Например поле "название" может быть определено вот таким объектом:

```
{name: 'title'}
```

Тем не менее в нашем случае, поскольку мы всего-лишь считываем данные как строку мы можем просто передать название поля и избавить себя от лишней работы:

```
'title'
```

Поле с "Дата выхода в прокат" (released) отличается, потому что мы хотим относиться к его содержанию как к типу данных. Для каждого типа формата поля может быть много настроек определяющих формат данных более подробно. Вместе с типами данных надо определить и строку содержащую dateFormat. если вы использовали **PHP**, то эти строки покажутся вам знакомыми, поскольку **Ext** использует тот же самый формат.

Указание типов данных

Ext JS has many ways to properly read in particular data types. They are shown here:

Тип поля	Описание	Информация
string	Информация строки	
int	Number	Использует функцию JavaScript parseInt function
float	Номер плавающей запятой	Использует функцию JavaScript parseFloat
boolean	информация True/False	
date	Информация о дате	Требуется настройка dateFormat

Ниже я привел несколько типов данных которыми я часто пользуюсь:

Тип поля	Описание	Информация
date	Информация содержащая дату	Вам так же следует уточнить dateFormat. Это скажет Ext как превращать данные в даты Y-m-d значит 'полный 4-х значный, с номером года, месяца и дня' dateFormat в Ext такая же как в PHP , и потому вот вам полезная ссылка о разных фрматах дат: http://www.php.net/date
int	Числовые данные	Относится к переменным как к целым числам. Это полезно, когда вы собираетесь сделать сравнение вашей информации для выполнения других действий, например для добавления двух столбцов вместе.
boolean или bool	Информация типа True/False	Заботится о чтении различных логических переменных, напремер перевода строк в действительные логические (булевы) значения, или перевод туда же нуля и единицы.

Если бы нам пришлось отображать данные так как увидит их пользователь. то получили что-то похожее на это:

Movie Database			
Released	Price	Avail	
Fri Feb 19 1999 00:00:00 GMT-0700 (Mountain Standard Time)	19.95	true	
Fri Feb 15 2002 00:00:00 GMT-0700 (Mountain Standard Time)	14.95	true	
Fri Oct 01 1999 00:00:00 GMT-0600 (Mountain Daylight Time)	19.95	true	
Fri Mar 06 1998 00:00:00 GMT-0700 (Mountain Standard Time)	21.9	true	
Fri Oct 15 1999 00:00:00 GMT-0600 (Mountain Daylight Time)	19.95	true	

Выглядит страшно, и сейчас объясню что произошло:

- Дата выхода в прокат (released) была правильно установленным типом даты, и переведена из значений наших данных. Это сделал страшный **JavaScript** формата даты. К счастью **Ext** может сделать это красивее.
- Столбец "Цена" была выставлена как плавающее значение. Замечу что нет необходимости выставять десятичную точность.
- Столбец Avail был переведен в логическое значение true, даже если исходные данные таковыми не были.

Поэтому я считаю полезным уточнять тип считываемых данных и применять специальные возможности.

- 1
- 2
- 3
- [следующая >](#)
- [последняя >](#)

-  [English](#)