

Обучение: Таблица PHP/SQL часть 3

Июнь 19 2010
 Настраиваем таблицу Ext

Итак... Время, чтобы показать наши результаты в красивой таблице Ext... Перед тем как продолжить, Вы должны быть в курсе двух вещей: DataStores и ColumnModels

Займемся нашим файлом mainscript.js :

```

var PresidentsDataStore;      // this will be our datastore
var PresidentsColumnModel;    // this will be our columnmodel
var PresidentListingEditorGrid;
var PresidentListingWindow;
```

```

Ext.onReady(function(){
    Ext.QuickTips.init();
```

DataStores

DataStores - это массивы, которые хранят данные. Самое важное то, что они могут получить свои данные от любого вида источника, от статических массивов XML до... динамических источников JSON!! Таблица должна быть присоединена к DataStore. Если Вы хотите изменить содержание таблицы, Вы должны изменить содержание datastore и затем вызвать refresh() метод.

Нам нужен DataStore, который должен будет соединиться с нашим файлом database.php, отправить запрос 'LISTING' и затем прочитать результаты. Затем мы даем соответствующие названия, которые 'mapped' к данным во множестве. Вот как мы получаем построение основного DataStore :

```

PresidentsDataStore = new Ext.data.Store({
    id: 'PresidentsDataStore',
    proxy: new Ext.data.HttpProxy({
        url: 'database.php',    // File to connect to
        method: 'POST'
    }),
    baseParams: {task: "LISTING"}, // this parameter asks for listing
    reader: new Ext.data.JsonReader({
        // we tell the datastore where to get his data from
        root: 'results',
        totalProperty: 'total',
        id: 'id'
    },[
        {name: 'IDpresident', type: 'int', mapping: 'IDpresident'},
        {name: 'FirstName', type: 'string', mapping: 'firstname'},
        {name: 'LastName', type: 'string', mapping: 'lastname'},
        {name: 'IDparty', type: 'int', mapping: 'IDparty'},
        {name: 'PartyName', type: 'string', mapping: 'name'},
        {name: 'TookOffice', type: 'date', mapping: 'tookoffice'},
        {name: 'LeftOffice', type: 'date', mapping: 'leftoffice'},
        {name: 'Income', type: 'float', mapping: 'income'}
    ]),
    sortInfo: {field: 'IDpresident', direction: "ASC"}
});
```

Вот... Не так уж и сложно.

ColumnModels

Теперь, мы должны определить ColumnModel для нашей таблицы. Проще говоря, ColumnModel объясняет Ext способ, которым определены наши колонки. Хотим ли мы вывести числа? строки? хотим ли мы чтобы они были редактируемыми? Фактически, ColumnModels может сделать намного больше вещей, о которых мы можем говорить в этом обучение. Вы можете сделать практически все с отдельной ячейкой если Вы персонализируете ее...

Теперь, мы просто покажем основные колонки, как вы можете видеть со следующем кодом:

```

PresidentsColumnModel = new Ext.grid.ColumnModel(
[
    {
        header: '#',
        readOnly: true,
        dataIndex: 'IDpresident', // this is where the mapped name is important!
        width: 50,
        hidden: false
    },
    {
        header: 'First Name',
        dataIndex: 'FirstName',
        width: 150,
        editor: new Ext.form.TextField({ // rules about editing
            allowBlank: false,
            maxLength: 20,
            maskRe: /^[a-zA-Z0-9\s]+$/ // alphanumeric + spaces allowed
        })
    },
    {
        header: 'Last Name',
        dataIndex: 'LastName',
        width: 150,
        editor: new Ext.form.TextField({
            allowBlank: false,
            maxLength: 20,
            maskRe: /^[a-zA-Z0-9\s]+$/
        })
    },
    {
        header: 'ID party',
        readOnly: true,
        dataIndex: 'IDparty',
        width: 50,
        hidden: true // we don't necessarily want to see this...
    },
    {
        header: 'Party',
        dataIndex: 'PartyName',
        width: 150,
        readOnly: true
    },
    {
        header: "Income",
        dataIndex: 'Income',
        width: 150,
        renderer: function(v){ return '$ ' + v; },
        // we tell Ext how to display the number
        editor: new Ext.form.NumberField({
            allowBlank: false,
            decimalSeparator&nbsp;: '.',
            allowDecimals: true,
            allowNegative: false,
            blankText: '0',
            maxLength: 11
        })
    }
]);
PresidentsColumnModel.defaultSortable= true;
```

Часть которую вы очень ждали...

ОК! Теперь, когда мы настроили источник данных исходя из того откуда считываются наши данные, настроили способ показать данные в таблице, мы готовы показать таблицу!! Следующая часть кода - действительно важная часть. Удостоверьтесь, что Вы не забыли load() DataStore :

```

PresidentListingEditorGrid = new Ext.grid.EditorGridPanel({
    id: 'PresidentListingEditorGrid',
    store: PresidentsDataStore,    // the datastore is defined here
    cm: PresidentsColumnModel,    // the columnmodel is defined here
    enableColLock: false,
    clicksToEdit: 1,
    selModel: new Ext.grid.RowSelectionModel({singleSelect: false})
});
```

```

PresidentListingWindow = new Ext.Window({
    id: 'PresidentListingWindow',
    title: 'The Presidents of the USA',
    closable: true,
    width: 700,
    height: 350,
    plain: true,
    layout: 'fit',
    items: PresidentListingEditorGrid // We'll just put the grid in for now...
});
```

```

PresidentsDataStore.load();    // Load the data
PresidentListingWindow.show();  // Display our window
```

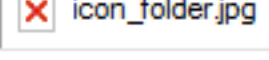
```

});
```

Что чтоит запомнить из данного раздела...

- Таблица требует как datastore так icolumnmodel чтобы корректно отображаться.
- Datastore необходимо перезагрузить если в базе данных сделаны изменения.

Вы можете увидеть результаты части 3 нашего обучения [здесь](#).

Вы можете загрузить исходники кода здесь :  [Загрузить zip файл](#)