[Обучение Ext JS] :: Часть 11. Drag-and-Drop

Мар 16 2010**Доводим drag-drop до ума**
Представьте что у вашего приложения есть экран с четырьмя отдельными контейнерами и каждый указан одновременно как DragZone и DropZone. Без drag-drop групп, поведение по умолчанию в этом случае будет таковым, что пользователь сможет переносить из любого из этих областей в любую другую область.
В большинстве случаев вы захотите ограничить такие действия. Давайте скажем что первый и второй контейнеры представляют сегодняшние и завтрашние назначения для врача, а третий и четвертый, то же самое, но для медсестры. Встречи врача не могут быть перенесены во встречи медсестры, и наоборот. Таким образом нам надо ввести ограничения.
Сделать это можно очень просто,указав ddGroup для "врачебных" контейнеров, и ddGroup для "сестринских". Это будет означать что при попытке перенести медсестры в контейнер доктора ничего не будет. Объектам медсестры не позволено взаимодействовать с предметами за пределами их собственной группы.
Подобное преимущество становится очень важным, когда вы создаете приложение, которое очень активно использует drag-and-drop. Нам не надо писать код для проверки правильности цели отпускания, поскольку группы позаботятся об этом за нас.

Все в деталях
Мы рассмотрели главные классы в предела пакета Ext.dd. Значит настало время искать интересные опции, методы, свойства и события которые помогут улучшить поведение классов drag-and-drop.

Настройке
"Большая четверка" Ext.dd — DragSource, DragZone, DropTarget, и DropZone — заметны в структуре **Ext JS**, являясь важными классами при этом не имеющих большого количества опций настроек. Это потому, что не смотря на их важность они относительно просто устанавливаются: назначить связанный узел и пошло-поехало. А потому мы уже рассмотрели одну из опций связанную с поддержкой классов —ddGroup— и есть еще пара более общих опций.
Опции dropAllowed и dropNotAllowed являются строками которые указывают классам **CSS** отправится в источник переноса когда все условия верны. Например, предмет парит над неправильным местом, тогда будет использован класс, указываемый dropNotAllowed.
У dropTarget и DropZone есть общая опция overClass. Это позволяет вам менять класс **CSS** примененный к элементу цели отпускания когда источник переноса перемещается на него. По умолчанию оно пустое, и это был бы полезный метод улучшения графической команды вызова, предоставляемой заместителем статуса.

Все под контролем
Поддержка drag-and-drop в **Ext JS** структурирована и немного отличается от остальной структуры. Обычно, чтобы добавить собственное поведение, вы обрабатываете события которые запускает класс, как это было в TreePanel и других компонентах. В случае с Ext.dd можно обработать очень мало событий. Вместо этого нам дается набор абстрактных методов для создания подмены.

ВАЖНО: ***Ext JS*** *описывает эти методы как "абстрактные". Этот концепт редко встречается в* ***Javascript*** *поскольку очень плохо поддерживается. Здесь термин просто относится к пустой разработке, которая должна быть заменена разработчиком для того чтобы достигнуть желаемой функциональности.*

Мы уже рассмотрели как надо для завершения процесса отпускния переписывать методы notifyDrop и onNodeDrop для DropZone, но существует множество других абстрактных методов за которые мы можем ухватиться. На самом деле, даже для действия отпускания существует набор похожих методов, которые можно использовать в определенных случаях:

- notifyDrop
- onContainerDrop
- onNodeDrop

Говоря о них, все они могут применяться по одному и тому же сценарии: обработка действия происходит при отпускании предмета на цель. Фокус заключается в том, что эти трое связаны. В DropZone, notifyDrop не совсем абстрактно, вместо этого оно просто указывает взывать onNodeDrop или onContainerDrop. именно эти два метода надо переписать если вы хотите выполнить действие отпускания.

Управляя движением
Пакет Ext.dd включает в себя класс DragDropMgr, который в основном предназначен для использования как внутренний помошник для структуры. Тем не менее есть несколько интересных моментов, которые можно заметить в этом классе. Давайте быстро рассмотрим способ, которым этот менеджер можно вставить в приложение.

Глобальный свойства
Поскольку DragDropMgr используется для отслеживания всех происходящих событий drag-and-drop, мы можем использовать его для глобальных изменений того, каким образом обрабатываются эти события. Например свойство clickPixelThresh может быть изменено, установив минимальное число пикселей, которые надо пройти курсору мышки прежде чем начнется перенос. Это полезно в тех случаях, когда ваши цели переноса выполняют какие-либо действия по щелчку, и вы хотите, увеличив это значение, застраховаться от нечаянных переносов.
Схожим образом, clickTimeThresh, обычно установленный на 1000 миллисекунд или 1 секунду, указывает время задержки до начала переноса. Его можно уменьшить, чтобы процесс выглядел более отзывчивым, или увеличить, если он мешает другим действиям.
Можно также настрить способ которым обрабатывается действие отпускания. По умолчанию режим drag-and-drop установлен на POINT, при котором положение курсора мыши используется для обозначения того, перетаскивается ли объект в пределы цели отпускания. мы можем изменить это поведение на режим INTERSECT:

```
Ext.dd.DragDropMgr.mode = Ext.dd.DragDropMgr.INTERSECT;
```

Это значит, что края источника переноса и цели отпускания используются для установки взаимодействия. Если два предмета перекрывают их, то тогда цель сброса считается верной. Это может быть полезно, если предметы которые вы хотите переносить будут большими и для пользователя будет проще заставить предметы коснуться краями.
Эти свойства только в теории могут использоваться так, но все же добавим их в наш инструментарий.

Управление прокруткой
Одна из хороших особенностей головоломки drag-and-drop заключется в том, что прокрутка контейнера, при попытке перенести предмет в область за границы экрана происходит автоматически. Ext.dd.ScrollManager значит, что вы можете или переносить предметы за пределы тела документа, или заставляет элемент с полосами прокрутки смещаются по мере того, как вы переносите элементы внутри него. Настройка этого довольно простая:

```
Ext.dd.ScrollManager.register('myContainer');
```

Этим кодом мы устанавливаем элемент myContainer для управление прокруткой. Обратите внимание, что прокрутка происходит короткими рывками. Это можно настроить при помощи регистрации элемента ddScrollConfig.

```
var el = Ext.get('myContainer');
  el.ddScrollConfig = {};
Ext.dd.ScrollManager.register(el);
```

DdScrollConfig состоит из объекта настройки, содержащий несколько опций. Рывки прокрутки анимированы по умолчанию и могут быть отключены установкой анимации на false, или же можно изменить их продолжительность на другое значение (по умолчанию 0,4 секунды):

```
el.ddScrollConfig = {
  animDuration: 0.2 // anim takes 0.2 seconds to complete
};
```

Можно увеличить частоту рывков установив частоту в миллисекундах, и изменить количество пикселей на рывок, используя опцию шага. Также можно управлять шириной зоны, которая задействует прокрутку. Опции vthresh и hthresh отвечают за вертикальную и горизонтальную зоны соответственно. Размеры для них задаются в пикселях.
Хотя Ext.dd.ScrollManager выполняет простые задачи и с минимум кода, нельзя недооценивать полезность этого класса. Установка функционала вручную будет изматывающей, а это настолько важная часть drag-and-drop, что без ScrollManage нам бы пришлось писать код для каждого приложения. Таки образом, хотя Ext.dd.ScrollManager и является довольно простым служебным классом, он сильно облегчает работу с drag-and-drop.