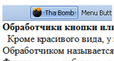
[Обучение Ext JS] :: Часть 4. Кнопки, меню и панели инструментов

Фев 20 2010**Кнопки иконок**

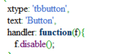
Стандартная кнопка может работать как кнопка иконки, вроде тех что используются в текстовых редакторах чтобы выделить текст курсивом или жирным шрифтом. Иконка кнопки создается в два этапа. Сначала, надо определить какое изображение будет использоваться в качестве иконки, а затем присвоить кнопке соответствующий класс.

```
{
 xtype: 'tbbutton',
 cls: 'x-btn-icon',
 icon: 'images/bomb.png'
}
```



Это с легкостью может быть иконкой за пределами текста, просто при помощи изменения класса стиля и добавления настроек текста.

```
{
 xtype: 'tbbutton',
 cls: 'x-btn-text-icon',
 icon: 'images/bomb.png',
 text: 'Tha Bomb'
}
```



Обработчики кнопки или, "нажми меня!"

Кроме красивого вида, у кнопки так же должна быть реакция на действия пользователей. Для этого мы используем обработчики. Обработчиком называется функция, запускающаяся при нажатии на кнопку .
Функцию мы добавляем в настройках обработчика:

```
{
 xtype: 'tbbutton',
 text: 'Button',
 handler: function(){
    Ext.Msg.alert('Boo', 'Here I am');
 }
}
```

По этому коду, в случае нажатия кнопки, будет выдаваться окно предупреждения. Иногда нам требуется чтобы вид кнопки, при нажатии на нее менялся. Таким образом, каждый обработчик кнопки оправляет ссылку на самого себя. Первое значение нашего обработчика является ссылкой на элемент запускающий событие.

```
{
 xtype: 'tbbutton',
 text: 'Button',
 handler: function(f){
   f.disable();
 }
}
```

Мы можем взять эту отсылку на кнопку, отсылку на саму себя, и получить доступ ко всем настройкам и функциям кнопки. В этом случае мы вызвали функцию недоступности. С ней кнопка становится серой и недоступной для действий пользователя.
Это весело, но почему бы не попробовать что-то более полезное?

Загрузка содержимого при нажатии на элемент меню

Давайте сделаем что-нибудь полезное с нажатием на кнопку. В данном случае, мы собираемся добавить возможность настройки, для каждого элемента меню, который будет использоваться для того, чтобы определять, какой файл содержимого будет загружаться на нашей странице.:

Исходный код примера:

Исходный код примера:

```
{
 xtype: 'tbsplit',
 text: 'Help',
 menu: [{
    text: 'Genre',
   helpfile: 'genre',
    handler: Movies.showHelp
 },{
    text: 'Director',
   helpfile: 'director',
    handler: Movies.showHelp
 },{
    text: 'Title',
   helpfile: 'title',
    handler: Movies.showHelp
 }]
}
```

Заметили настройку helpfile, которую мы добавили в каждый элемент? Это сделано затем, что нам необходимо хранить переменные, уникальные для каждого элемента меню. Это возможно, благодаря тому, что свойства настроек могут быть чем угодно и их можно создавать на ходу. В этом случае, мы используем свойство настройки как переменную, содержащую имя того файла, который мы хотим загрузить.

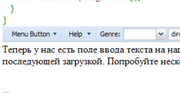
Еще мы создаем набор функция для обработки нажатий на кнопки элементов меню. Эти функции организованы в класс Movies.

Исходный код примера:

Исходный код примера:

```
var Movies = function() {
  return {
    showHelp : function(btn){
      var helpbody = Ext.get('helpbody');
      if (!helpbody) {
        Ext.DomHelper.append(Ext.getBody(), {
            tag:'div',
            id:'helpbody'
          });
      }
      Movies.doLoad(btn.helpfile);
    },
    doLoad : function(file){
      Ext.get('helpbody').load({
        url: 'html/' + file + '.txt'
      });
    }
  };
}();
```

Пока что, я не хочу углубляться в путаницу этого класса, но, все что он делает, это обрабатывает нажатия на кнопки меню. Этот класс будет загружать текст в файл нашей страниц через запрос **AJAX**. В зависимости от того, какой элемент был нажат, тот текст и будет загружаться. Таким образом, как только этот класс займет свое место, мы сможем открыть страницу в браузере и щелкая по кнопкам загрузить относящийся к ним файл помощи.



Дальше, мы попробуем сделать тоже самое, но используя поле ввода текста.

Поля формы из панели инструментов

Как в большинстве других предметов **Ext**, на панели инструментов можно разместить практически любой элемент. Поля формы и комбинированные окна, пожалуй, самые полезные.

```
{
 xtype: 'textfield'
}
```

Тем же самым способом, которым мы пользовались в прошлой главе, мы добавим поле формы в массив элементов. Это добавит на панель нашу форму. Теперь, заставим ее делать что-нибудь полезное, например выполнять ту же самую работу что и меню помощи, но более динамично.

```
{
 xtype: 'textfield',
 listeners: {
   specialkey: Movies.doSearch
 }
}
```

Этот слушатель добавлен непосредственно в настройки поля формы. Для этого мы используем слушатель specialkey, который уже встречался нам в предыдущей главе. Это тот же самый элемент, который отслеживал нажатия таких клавиш, в том числе Enter и Delete. Мы также добавим обработчик функции к созданному нами ранее классу Movies:

```
doSearch : function(frm,evt){
  if (evt.getKey() == evt.ENTER) {
    Movies.doLoad(frm.getValue());
  }
}
```


Теперь у нас есть поле ввода текста на нашей панели инструментов, которое позволяет нам вписать название текстового файла с его последующей загрузкой. Попробуйте несколько примеров из нашего меню, например director или title.

Панели инструментов в окнах, сетках и табло

У всех панелей инструментов с которыми мы работали есть настрока элементов. И если вы захотите разместить одну из панелей в другом элементе Ext, таком как табло или окно, мы можем просо взять содержимое настроек и поместить их в один из двух, существующих по умолчанию, контейнеров для панельных элементов.

У таких панельных элементов как окно и сетка, существуют верхние и нижние панели инструментов:

- *tbar:* Верхняя панель инструментов
- *bbar:* Нижняя панель инструментов

Если мы хотим разместить панель инструментов вверху окна, то можно сделать это заполнив настройки tbar массивом элементов панели:

Исходный код примера:

Исходный код примера:

```
new Ext.Window({
    title: 'Help',
    id: 'helpwin',
    width: 300,
    height: 300,
    tbar: [{
        text: 'Close',
        handler: function(){
            Ext.getCmp('helpwin').close();
        }
    },{
        text: 'Disable',
        handler: function(t){
            t.disable();
        }
    }],
    autoLoad: 'html/' + btn.helpfile + '.txt'
}).show();
```

В **Ext** существует свои панель инструментов для страничных сеток и содержащая все кнопки для перемещения по страницам с результатами. Подробнее мы рассмотрим это в главе посвященной собственно сеткам.

Заключение

В этой главе нам выпала возможность поиграть с различными способами создания элементов панели инструментов, включая использование настроек объекта и их ярлыков. Большое количество опций, доступных для панелей делает их чрезвычайно полезными элементами для всего, начиная от простейшей кнопки и заканчивая сложными сочетаниями кнопок, меню и полей формы. А работать с кнопками, полями формы и меню легко, используя встроенные обработчики.