15
Ext JS 4 Next Steps

The Ext JS 4 library is one of the most powerful libraries for building Rich Internet Applications. Through this book we have learned how the library works and what capabilities it has. Now that we have a full working application and have learned how to use the library, we must know that there are a lot of other things around this awesome library. In this chapter we will see what other concepts may help us to improve our development process when using Ext JS 4.

In this chapter we will cover the following topics:

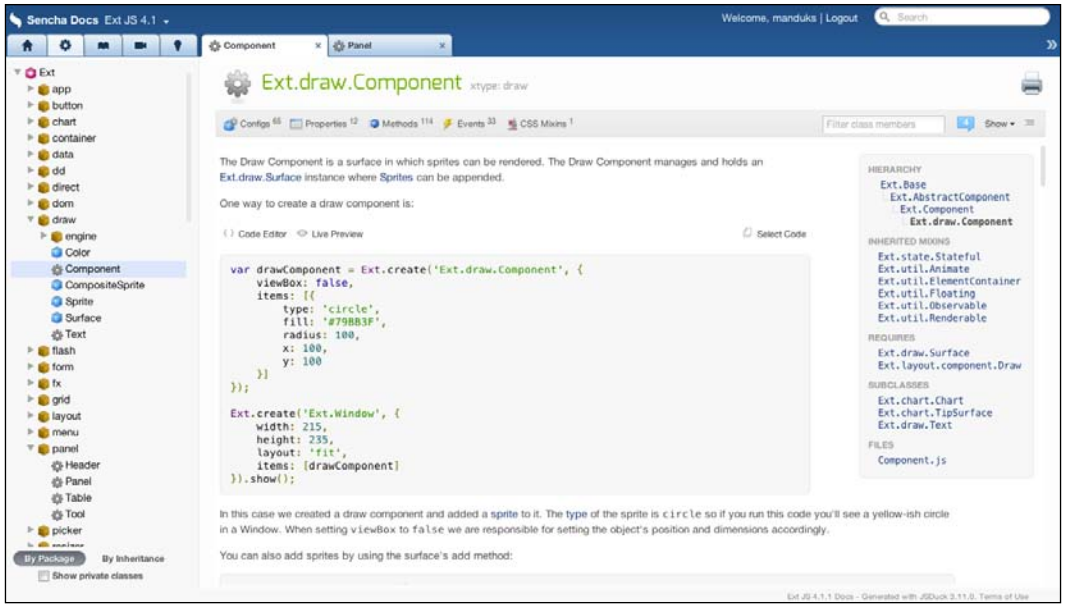
- The Ext JS 4 API documentation
- Using JSDuck
- Want to go mobile?
- The Sencha forums

The Ext JS 4 API documentation

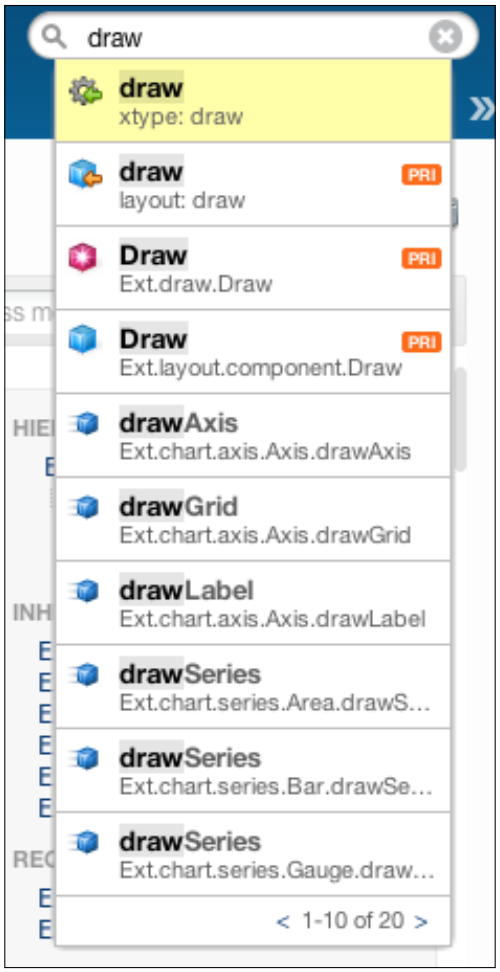
Ext JS 4 has some of the best documentation; it is easy to use and very powerful. The Sencha team has always put so much effort when delivering their frameworks' documentation, but the Ext JS 4 and Sencha Touch 2 APIs are a state of art in documentation.

When we are new to Ext JS 4, one of the most important things we have to know is how the API works. We don't need to memorize each class and class methods, we just need to know where to find them.

The following screenshot shows the **Sencha Docs** structure:

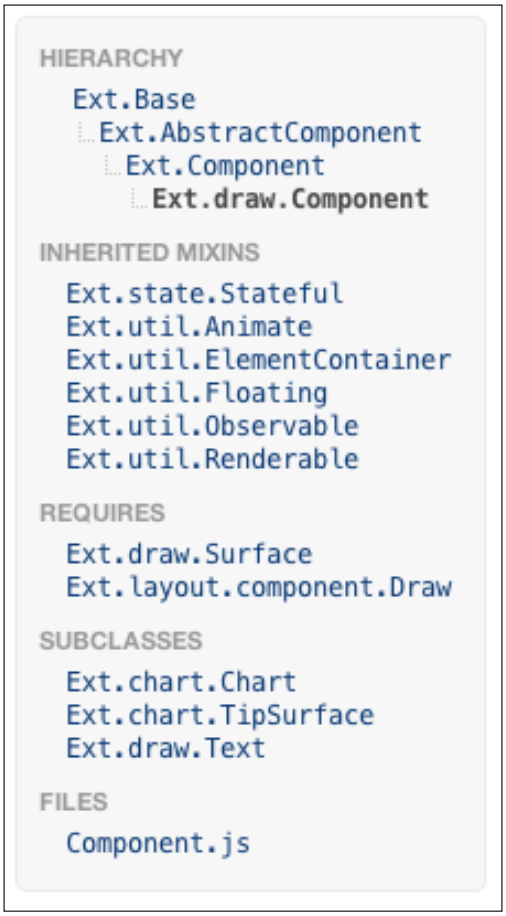


In the Sencha Docs we have five main regions. On the left-hand side we have the package tree list with all the packages and classes Ext JS 4 has in its library. In the top right-hand corner we have the search component where we can search almost anything like classes, methods, guides, xtypes, and more. The following screenshot shows how the search component works:



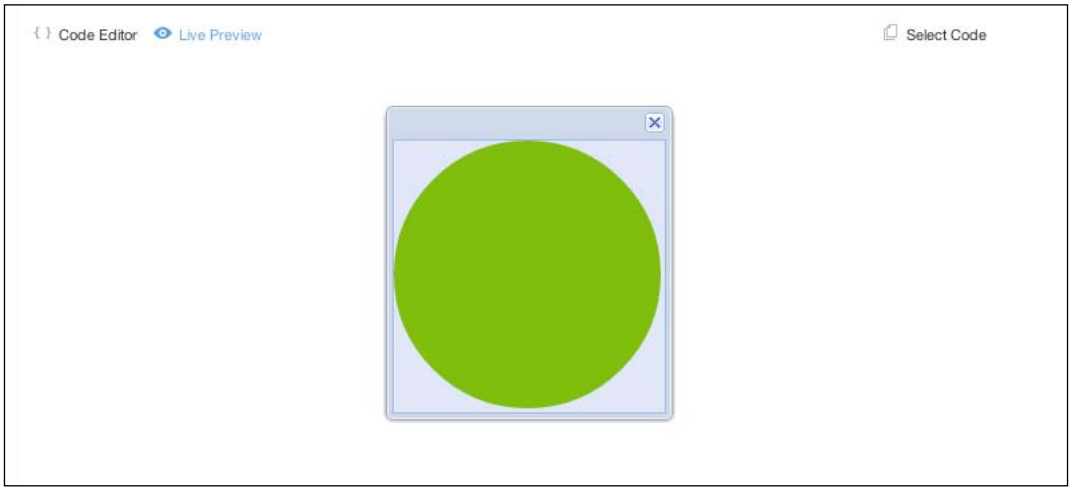
Once we are in the class definition there are three main regions we need to understand. First we have the class members' toolbar, where we can see each class property, possible configurations, methods, events, and the CSS Mixins used on SASS. In this region we also have a class members filter. The following screenshot shows the class members' toolbar:

The second region is to the right-hand side of the class definition. We have the class relations container, where we can see the class hierarchy, inherited mixins, required subclasses, and the filename. In this container we can access any class definition the class has a relation with quickly. The following screenshot shows the class relations container:

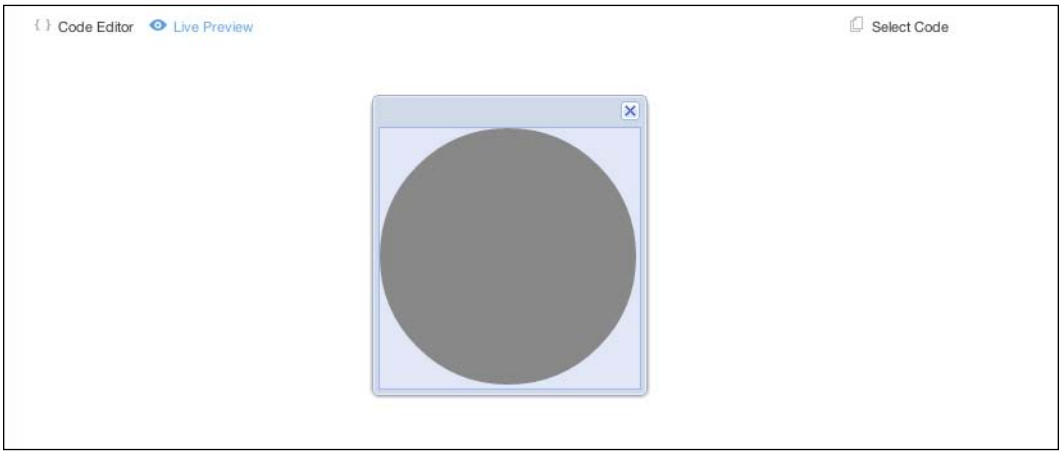


The final region is where we can read the code examples, here we can see a live preview of the code and also can edit the code and play around with the examples. The following screenshot show this region:

The previous screenshot shows the code editor region, where we can edit the code and see a live preview in action just by clicking on the **Live Preview** button. The following screenshot shows the **Live Preview** button in action:



Now we are just going to edit the circle color in the code editor and set it to gray, the following screenshot shows the output:



Now that we have some understanding about how the documentation works, we are going to see how to create our own documentation.

Using JSDuck

JSDuck is the API documentation generator for Secha frameworks; this project is hosted on GitHub (`https://github.com/senchalabs/jsduck`). JSDuck is a Ruby-based project so we need to install Ruby and then install the JSDuck gem with the following command:

```
$[sudo] gem install jsduck
```

When we have finished installing the JSDuck gem we are ready to generate our project documentation. First we need to go to our project files folder and document each class we have defined. The following code example explains how we must comment our code:

```
/**
 * @class MyApp.view.categories.CategoriesForm
 * @extends Ext.form.Panel
 * @author Armando Gonzalez <iam@armando.mx>
 * <p>The categories form panel</p>
 */
Ext.define('MyApp.view.categories.CategoriesForm', {
  extend    : 'Ext.form.Panel',
  alias       : 'widget.categories.form',
```

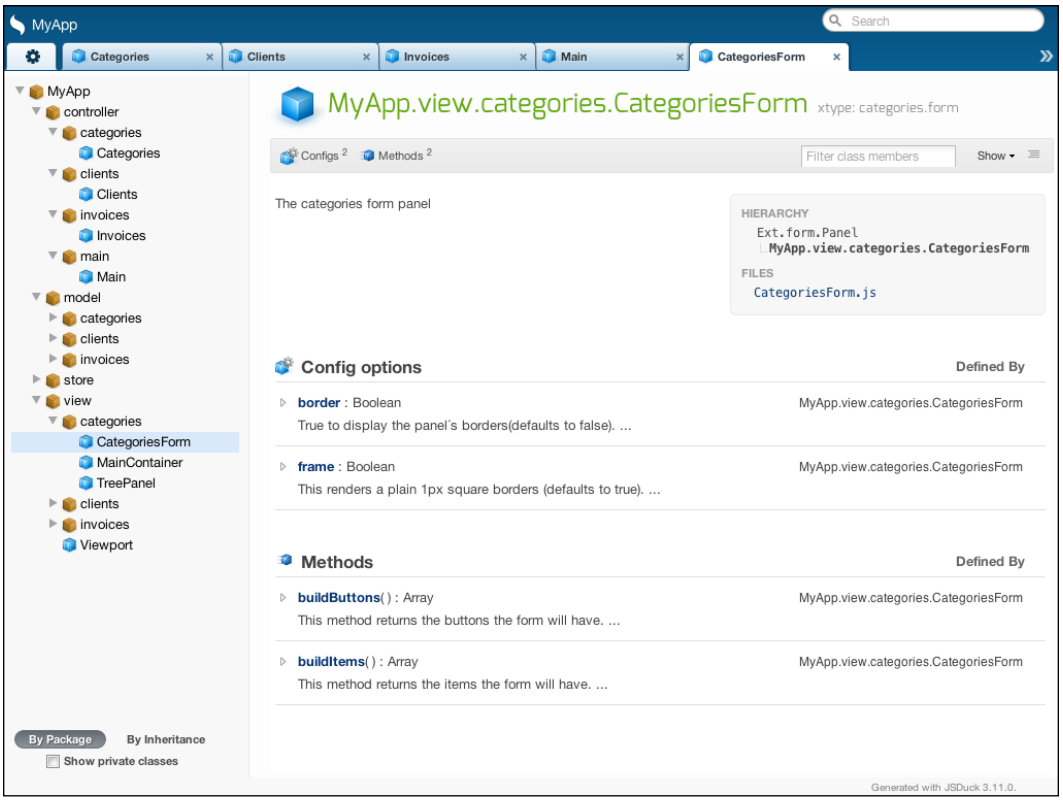
```
    /**
     * @cfg {Boolean} border True to display the panel´s borders(defaults
  to <tt>false</tt>).
     */
    border    : false,
    /**
     * @cfg {Boolean} frame This renders a plain 1px square borders
  (defaults to <tt>true</tt>).
     */
    frame    : true,

    // private
    initComponent   : function(){
      var me = this;
      me.items = me.buildItems();
      me.buttons = me.buildButtons();
      me.callParent();
    },

    /**
     * This method returns the items the form will have.
     * @return {Array} The fields which this Form contains.
     */
    buildItems : function(){
      return [{
        fieldLabel  : 'Name',
        name    : 'name',
        xtype    : 'textfield',
      }];
    },

    /**
     * This method returns the buttons the form will have.
     * @return {Array} The Buttons this Form contains.
     */
    buildButtons:function(){
      return [{
      text: 'Save',
          itemId: 'savebtn'
            }];
    }
  });
```

It is very important to add comments to each class in our projects. It may look tedious, but it will show its value during the developing process.

Once we have documented each class in our application we just need to run a simple command to our project folder:

```
$ jsduck app -o documentation --title=MyApp
```

The following screenshot shows our project documentation in action:



Making our project documentation is very easy. We just need to get in the habit of always documenting our code and JSDuck will do the rest. For more information about JSDuck you can check the JSDuck guide at `https://github.com/senchalabs/jsduck/wiki/Guide`.

Want to go mobile?

These days it is very common to have a mobile version of our applications and the Sencha team knows it, that is why they have made the Sencha Touch 2 framework.

Sencha Touch 2 is a high-performance HTML5 mobile application framework. This framework was designed for building fast and impressive applications that work on IOS, Android, BlackBerry, and Kindle Fire.

The Sencha Touch 2 framework is very similar to Ext JS 4, both have a similar syntax, architecture, and performance. So for users, who code in Ext JS 4, it is very easy to start coding Sencha Touch 2 applications.

Through this book we have learned how to code our applications using Ext JS 4. But we have to go mobile at some time, that is why we are going to see how easy it is to start a mobile application using the Sencha Touch 2 framework.

First we are going to download the Sencha Touch 2 framework from `http://www.sencha.com/products/touch/download/`. Let's go and see the getting started guide at `http://docs.sencha.com/touch/2-0/#!/guide/getting_started` and do all the steps that are mentioned there.

Once we have our getting started example running we are ready to start listing invoices in our basic invoices mobile application.

First we need to define the invoices' model:

```
/**
 * @class InvoicesMobile.model.Invoice
 * @extends Ext.data.Model
 * @author Armando Gonzalez <iam@armando.mx>
 * The invoice model definition
 */
Ext.define('InvoicesMobile.model.Invoice', {
  extend: 'Ext.data.Model',
  config :{
    fields  :   [
      {name:'name', type:'string'},
      {name:'date', type:'date', dateFormat: 'm/d/Y'}
    ],
  proxy  : {
  type  : 'ajax',
  url  : 'serverside/invoices/list.json',
        reader  : {
          type  : 'json',
          rootProperty  : 'data'
        }
      }
  }
});
```

This model definition is very similar to the previous invoices' definition we had. After this we need to define the store:

```
/**
 * @class InvoicesMobile.store.Invoices
 * @extends Ext.data.Store
```

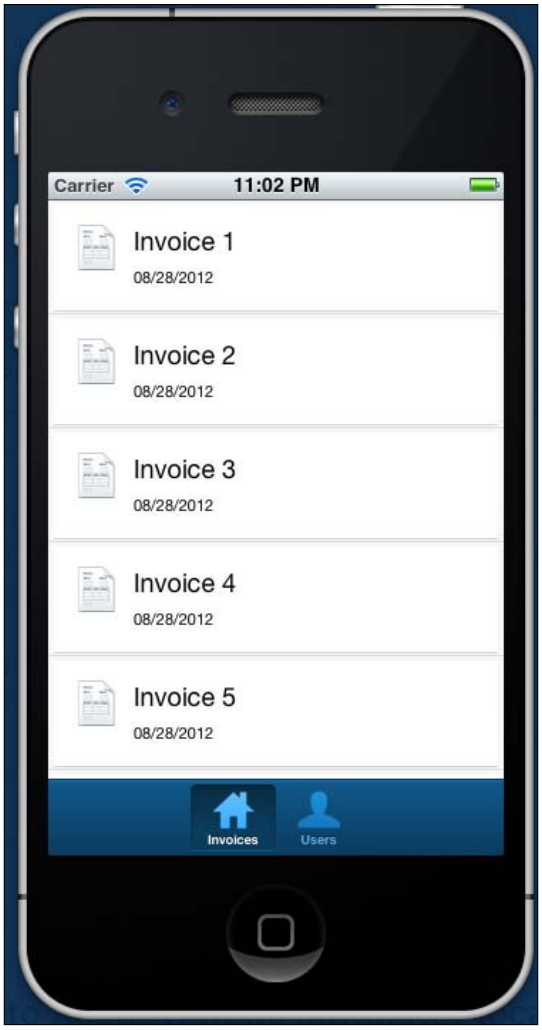
```
 * @author Armando Gonzalez
 * This is the definition of our Invoices store
 */
Ext.define('InvoicesMobile.store.Invoices', {
  extend    : 'Ext.data.Store',
  requires  : 'InvoicesMobile.model.Invoice',

  config :{
    model    : 'InvoicesMobile.model.Invoice',
    autoLoad  : true
  }
});
```

The store definition is pretty similar too, now we are going to define the list that will render our invoices:

```
/**
 * @class InvoicesMobile.view.invoices.List
 * @extends Ext.List
 * @author Armando Gonzalez
 * This is the definition of our Invoices list
 */
Ext.define('InvoicesMobile.view.invoices.List', {
  extend: 'Ext.List',
  xtype:'invoiceslist',

  requires :['Ext.plugin.PullRefresh','Ext.plugin.ListPaging'],

config: {
  store:'Invoices',
    masked:{
      xtype: 'loadmask',
      message: 'loading ...'
    },
    scrollable: {
      direction: 'vertical',
      directionLock: true
    },
    plugins: [
             'pullrefresh',
             {
  type: 'listpaging',
  autoPaging: true
             }
        ],
    itemTpl:[
        '<tpl for =".">',
          '<div class="invoice">',
            '<div class="img">',
              '<imgsrc="resources/images/invoice64.png">',
```

```
                '</div>',
                '<div class="content">',
                  '{name}</br>',
                  '<spam>{[Ext.util.Format.date(values.date)]}</spam>',
                '</div>',
                '</div>',
            '</tpl>'].join(''),
        }
    });
```

The list's definition has some extra plugins added, so it can paginate and reload the store with the `pullrefresh` action. With this basic model, store, and list's definitions we are ready to see our basic invoices mobile application in action:



The previous screenshot is the output of the invoices mobile version. As we have seen once we know the basics of the Ext JS 4 framework we can start developing a mobile version of our applications using the Sencha Touch 2 framework.

The Sencha forums

Another good place to keep learning are the Sencha forums (`http://www.sencha.com/forum`). The Sencha libraries have a huge growing community, where we can find answers for our questions, and help others answer their questions.

Recently Sencha launched another great idea, the Sencha Try project, hosted on `http://try.sencha.com/`. Here you can find and download fully working examples that community members upload to GitHub (`https://github.com/`) and are exposed in the Sencha Try site.

Summary

Through this book we have learned the power, usability, and awesome architecture Ext JS 4 has. This library is very powerful and can boost our developing process. Learning to use the framework is quite easy, but we need to know the basis of how to design a good application architecture.

Now we are ready to use, extend, and configure the main classes of the library and are ready to start delivering powerful web applications using the Ext JS 4 framework.