The new Sencha Ext JS 5 library offers hundreds of components and APIs to build robust applications and fulfills the critical needs of customers all around the world. The new version 5 is packed with new themes and the MVVM architecture that allows you to connect the model layer to the view and automatically update the model when the view is modified and vice versa.

This book covers many new features and components of Ext JS 5. At the beginning, learn the core concepts of Sencha Ext JS, components, data models, and mapping. This book also teaches you event-driven development, forms and grids, charts and themes, and third-party plugins. Later on in the book, you'll learn the implementations of the Tree panel, the MVC pattern, and a completely new feature called MVVM. By working sequentially through each chapter and following the step-by-step guides, you will be able to create a basic application.

Who this book is written for
If you are a JavaScript developer who now wants to use the Ext JS framework, this is the book for you. This guide is useful to you whether you’re new to Ext JS 5 or are a seasoned expert. Experience in HTML, CSS, and JavaScript is required in order to understand and get the most out of this book.

What you will learn from this book
- Create applications with powerful web components designed for an easy end user experience
- Discover how Ext JS works and how to make good use of this tool
- Understand the basics of web applications using Ext JS
- Use the new features of version 5 in your applications
- Build agile and responsive web applications using Ext JS
- Understand the MVVM application model to connect the model layer to the view
- Easily customize sizing, spacing, colors, fonts, and so on in the new touch-friendly themes

Create powerful web applications with the new and improved Ext JS 5 library.
In this package, you will find:

- The author's biography
- A preview chapter from the book, Chapter 1 'An Introduction to Ext JS 5'
- A synopsis of the book’s content
- More information on Learning Ext JS Fourth Edition
About the Authors

Carlos A. Méndez is a freelance developer and graphic designer living in México, with expertise in web development since 2000 and Windows development since 1998. He has also worked with Ext JS since version 2.x up to the present day. Since 1998, he has developed and designed administrative applications for accounting, payroll, inventory, human resource control, restaurants, hotels, and much more—applications that are in production and up to date.

Trying to explore creativity with a deep passion, Carlos has created many Ext JS components and VB components for private companies and was also involved in graphic design, such as illustrations and small animations used for interactive presentations by some companies in México. He always has a passion for creating and learning new things on the Web.

He is the founder and development manager of the company administrationonline.com, which is focused on administrative applications. Carlos is involved in many projects around the world and also provides support and maintenance to many Ext JS projects ranging from 2.x to 5.x.
Crysfe1 Villa is a software engineer with more than 8 years of experience with JavaScript. He started his career as a web developer working with HTML and basic JavaScript in the late 1990s but then started focusing on server-side technologies, such as PHP and Java J2EE.

Before he started working with the Ext JS library, he loved to work with MooTools, but in late 2007, he started learning about an awesome new library that was emerging as an open source project. At that time, version 2.0 of the Ext JS library had just been released, and Crysfe1 started using this new library for medium-to-large projects in the agency that he used to work for.

In early 2010, he started working as a freelancer. He began training teams on Ext JS for private companies; writing a blog with tutorials, tips, and tricks; developing custom components on Ext JS for his clients; and working on open source projects to share his knowledge with the world.

More recently, Crysfe1 has been getting into new technologies such as Angular JS and React Native. If you want to find out more about his work, you can follow him on Twitter (@crysfe1) or download his open source projects from GitHub (crysfe1).
Over the past few years, Ext JS has become a popular and powerful JavaScript framework for desktop application development. For an Ext JS developer, the learning curve is not very easy/fast and I have seen cases where developers learning this framework find it to be a slow process. While writing this book I was thinking about the easiest and most comprehensible points so that you can understand the basics, just as I would have liked to learn about the framework if I was in your place.

This book is intended for developers who have the desire to learn and begin using this framework for their applications, and also for developers who have not started using the current version. It is written as an easy-to-follow guide that will help you understand the basics and fundamentals of the framework. If you have experience with previous versions of the framework, this book may clear many of your doubts about upgrading and how things happen in version 5.x.

This book covers all of the basic information you need to know to start development with this nice and powerful framework.

What this book covers

Chapter 1, An Introduction to Ext JS 5, covers an explanation of how to start by getting the framework (downloading the file) and setting up the basic requirements you need in order to begin coding. This chapter also provides an explanation of how the framework is structured, how to set up some required tools, and gives a quick peek at the product, Sencha Architect.

Chapter 2, The Core Concepts, is about the framework’s class system, and tells you how to use object-oriented programming with Ext JS. Also, this chapter explains how to extend classes, how to inherit properties, and the use of the Loader system in order to define and require dependencies in a dynamic way.
Chapter 3, *Components and Layouts*, explains how components work, how they are created, their life cycle, and how to take advantage of all this. Here, you also learn about types of containers and the layout system, which will help you create amazing UIs with little effort.

Chapter 4, *It's All about the Data*, explains how the framework handles and manipulates data to display information using data-aware widgets or components.

Chapter 5, *Buttons and Toolbars*, shows you how to make use of component events; listen to events; (mainly) create buttons, toolbars, and menus; and set the most basic configurations for these components.

Chapter 6, *Doing It with Forms*, talks about the form component, the available fields that we can use in our forms, and how to collect and submit data.

Chapter 7, *Give Me the Grid*, covers the basics of the most popular component, the Grid panel, in the framework, how to implement it, its column model, and custom data renderers for displaying data. We also see how to listen to events in the Grid panel and look at some plugins and features (specific capabilities) that can be implemented in the grid.

Chapter 8, *DataViews and Templates*, explains how to make use of DataViews and templates to create data-aware views, implement a nice organization of our data, and set styles and custom logic for the representation of data.

Chapter 9, *The Tree Panel*, covers the use of the tree panel component and its implementation. It also explains how to create stores and data for this component.

Chapter 10, *Architecture*, is one of the most important chapters in the book. It shows how to create an application using the MVC and MVVM patterns. This is done in order to create applications that can be scalable and easy to maintain. The MVVM pattern, which is a powerful pattern for reducing code, is introduced in version 5.

Chapter 11, *The Look and Feel*, demonstrates how to create new themes inside the framework and applications by giving our applications a new look and some color changes (themes). Also, you learn how to create specific component-style UIs using Compass and Sass.

Chapter 12, *Responsive Configurations and Tablet Support*, explains how we can use touch screen themes and how to set responsive configurations in components in order to make those components responsive-aware.
Chapter 13, *From Drawing to Charting*, talks about the basics of drawing and chart creation. We see how to create charts by the use of the SVG/VML engines. This chapter also explains how to add the Chart package to applications and the theme engine introduced in version 5.

Chapter 14, *Finishing the Application*, covers how to prepare our application for the production environment and deployment, covering the most essential parts for final production.

Chapter 15, *What's Next?*, shows you where to get more feedback and resources such as forums, other useful resources to get information tutorials from, and so on. This chapter also gives a sneak peek into some useful plugins (commercial and free).
An Introduction to Ext JS 5

When learning a new technology such as Ext JS, some developers face a hard time to begin with, so this book will give you the best possible way to start to understand this technology more than any other source. We have to go from the library documentation to blogs and forums looking for answers, trying to figure out how the library and all the components work together. Even though there are tutorials in the official learning center, it would be great to have a guide to learn the library from the basics to a more advanced level; this is the main goal of this book.

Ext JS is a state of the art framework to create Rich Internet Applications (RIAs). The framework allows us to create cross-browser applications with a powerful set of components and widgets. The idea behind the framework is to create user-friendly applications in rapid development cycles, facilitate teamwork (MVC or MVVM), and also have a long-term maintainability.

Ext JS is not just a library of widgets anymore; the brand new version is a framework full of new exciting features for us to play with. Some of these features are the new class system, the loader, the new application package, which defines a standard way to code our applications, and much more awesome stuff.

The company behind the Ext JS library is Sencha Inc. They work on great products that are based on web standards. Some of the most famous products that Sencha also have are Sencha Touch and Sencha Architect.

In this chapter, we will cover the basic concepts of the framework of version 5. You’ll learn how to set up the library or SDK and create our first program, get to know the available tools to write our code, and take a look at some of the new features in Ext JS 5.

- Considering Ext JS for your next project
- Getting started with Ext JS—our first program
An Introduction to Ext JS 5

- Tools and editors
- What's new in Ext JS 5?

Considering Ext JS for your next project
Ext JS is a great library to create RIAs that require a lot of interactivity with the user. If you need complex components to manage your information, then Ext is your best option because it contains a lot of widgets such as the grid, forms, trees, panels, and a great data package and class system.

Ext JS is best suited for enterprise or intranet applications; it's a great tool to develop an entire CRM or ERP software solution. One of the more appealing examples is the Desktop sample (http://dev.sencha.com/ext/5.1.0/examples/desktop/index.html). It really looks and feels like a native application running in the browser. In some cases, this is an advantage because the users already know how to interact with the components and we can improve the user experience.

Ext JS 5 came out with a great tool to create themes and templates in a very simple way. The framework for creating themes is built on top of Compass and Sass, so we can modify some variables and properties and in a few minutes we can have a custom template for our Ext JS applications. If we want something more complex or unique, we can modify the original template to suit our needs. This might be more time-consuming depending on our experience with Compass and Sass.

Compass and Sass are extensions for CSS. We can use expressions, conditions, variables, mixins, and many more awesome things to generate well-formatted CSS. You can learn more about Compass on their website at http://compass-style.org/.

The new class system allows us to define classes incredibly easily. We can develop our application using the object-oriented programming paradigm and take advantage of the single and multiple inheritances. This is a great advantage because we can implement any of the available patterns such as MVC, MVVM, Observable, or any other. This will allow us to have a good code structure, which leads us to have easy access for maintenance.

Another thing to keep in mind is the growing community around the library; there are lots of people around the world that are working with Ext JS right now. You can even join the meeting groups that have local reunions frequently to share knowledge and experiences; I recommend you to look for a group in your city or create one.
The new loader system is a great way to load our modules or classes on demand. We can load only the modules and applications that the user needs just in time. This functionality allows us to bootstrap our application faster by loading only the minimal code for our application to work.

One more thing to keep in mind is the ability to prepare our code for deployment. We can compress and obfuscate our code for a production environment using the Sencha Cmd, a tool that we can run on our terminal to automatically analyze all the dependencies of our code and create packages.

Documentation is very important and Ext JS has great documentation, which is very descriptive with a lot of examples, videos, and sample code so that we can see it in action right on the documentation pages, and we can also read the comments from the community.

Getting started with Ext JS
So, let's begin with Ext JS! The first thing we should do is download the framework from the official website, http://www.sencha.com/products/extjs/. The version available at the time of writing this book is 5.1.1.

There are three types of license:

- **The open source license**: If you are creating or want to develop an open source application compatible under the GNU GPL license v3 (http://www.gnu.org/copyleft/gpl.html).

- **The commercial license**: You need to buy this if you are planning/wanting to develop a closed source project and want to keep the source code as your own property. Usually used by corporations, banks, or enterprises.

- **The commercial OEM**: If you want to use Ext JS to create your own commercial SDK or web application builder, or use it as frontend for some embedded device, then this comes into the picture. As this type of license can vary, it's customized for each customer.

You can see more detailed information about this subject at http://www.sencha.com/products/extjs/licensing.
An Introduction to Ext JS 5

Downloading Ext JS

If you download Ext JS directly from http://www.sencha.com/products/download/, this will be a 30-day trial version of Ext JS and you will also be required to enter some personal information in order to get the trial. To obtain the GPL version, you can get it from http://www.sencha.com/legal/GPL/. We can also use the available Content Delivery Network (CDN), as shown in the following table, so that we don't need to store the library on our own computer or server:

<table>
<thead>
<tr>
<th>Theme</th>
<th>Links</th>
</tr>
</thead>
</table>
| Classic| • CSS file: http://cdn.sencha.com/ext/trial/5.1.1/packages/ext-theme-classic/build/resources/ext-theme-classic-all.css  
    • JavaScript file: http://cdn.sencha.com/ext/trial/5.1.1/build/ext-all.js   |
| Neptune| • CSS file: http://cdn.sencha.com/ext/trial/5.1.1/packages/ext-theme-neptune/build/resources/ext-theme-neptune-all.css  
    • JavaScript file: http://cdn.sencha.com/ext/trial/5.1.1/build/ext-all.js  
    • Theme JS Overrides: http://cdn.sencha.com/ext/trial/5.1.1/packages/ext-theme-neptune/build/ext-theme-neptune.js  |
| Crisp  | • CSS file: http://cdn.sencha.com/ext/trial/5.1.1/packages/ext-theme-crisp/build/resources/ext-theme-crisp-all.css  
    • JavaScript file: http://cdn.sencha.com/ext/trial/5.1.1/build/ext-all.js  
    • Theme JS Overrides: http://cdn.sencha.com/ext/trial/5.1.1/packages/ext-theme-crisp/build/ext-theme-crisp.js  |
Setting up and installing Ext JS 5

After you download the Ext JS Library (ZIP file), extract the contents to a working folder. For the first time, you will probably get overwhelmed by the size of the ZIP file and by the number of files and folders, but don't worry, the purpose of each file and the content of each folder will be explained shortly.

Sencha Cmd

Besides the Ext JS library, we need to download the Sencha Cmd (command tool). This tool is intended to be a cornerstone for building applications, creating workspaces, and new themes, and the ability to minify and deploy our applications to a production environment.

Download this tool at http://www.sencha.com/products/sencha-cmd/ and also check that the following requirements are met in order for Sencha Cmd to work properly:

- JRE Sencha Cmd requires Java Runtime Environment version 1.7 to support all functionality, however, most features will work with 1.6 (the minimum supported version).
- Ruby differs by OS:
  - Mac OS: Ruby is preinstalled. You can test whether Ruby is installed with the Ruby -v command.
  - Linux-based OS: Use sudo apt-get install ruby 2.0.0 to download Ruby.

Run the Sencha Cmd setup, follow the instructions, and after installing Sencha Cmd, we need to verify the installation. Proceed to open the command line and type the following command:

```
sencha
```

On Windows environments, it's recommended that you restart the system after installation in order to get the proper environment variables applied.
After typing the command Sencha, we should see the following output:

![Command Output]

**Why so many files and folders?**

This is a natural question when you look at the downloaded files and folders for the first time, but every file and folder is there for a purpose and now you're going to learn it:

- The **build** folder contains compiled files of the SDK and is ready to be used. This folder is very useful to start with in Ext JS without the need to use Sencha Cmd. From version 5, this folder also contains examples and ready-to-use Ext JS themes located in packages (folder).

- The **examples** folder contains the source code of the examples. These examples are built to show what we can do with the library. However, one significant change in version 5 is that this folder needs to be compiled using Sencha Cmd in order to be deployed/compiled into the build folder.

- The **overrides** folder contains JavaScript files used to add extra functionality and behavior to components and widgets and they are also used when an application or code is compiled.

- The **packages** folder is where the styles and images are located; we can also find the Sass files to create our custom theme in here. Sass is an extension of CSS3 to improve the language; we can use variables, mixins, conditionals, expressions, and more with Sass. From version 5 onward, this folder also contains more folders, which are Locales, Ext JS Core, Charts, Aria, and many more.
• The src folder contains the source code files that are part of the framework. Each file represents a class/object so we can read it easily, and every folder corresponds to the namespace assigned to the class. For example, the Ext.grid.Panel class is in a file called Panel.js, which in a folder called grid (src/grid/Panel.js).

• The welcome folder contains the styles and images that are shown when we open the index.html file in the root folder.

If you look at the root folder, you can also see other JavaScript files. Basically, they are the compressed, debug, and development versions of the library.

• The bootstrap-* .js files contain information about the framework; these files are used by ext*.js files in order to load the required files (the src folder or packages folder).

• The ext-all.js file loads the complete library with all the components, utilities, and classes.

• The ext-all-debug.js file is the same as the ext-all.js file. The difference is that this file will show console logs and we can use this file to debug our application.

• The ext.js file is the core and foundation layer for Ext JS. If we use this file, we're not loading the whole library; this file contains only the class system, the loader, and a few other classes. We can use the Ext.Loader class to load just the required classes and not the entire framework.

Folders that changed in version 5 from previous versions

Developers that use previous versions of Ext JS may find the new folder structure confusing, and may notice that some of folders disappeared in version 5. The significant changes to folders are listed below. The builds folder no longer exists; instead, we should use the build folder.

• The locale folder has been moved to the packages/ext-locale folder. In version 5, Locales have a more complex folder structure and we also now have the ext-locale-language.js file and ext-locale-language-debug.js file. By default, the components are displayed in English, but you can translate them to any other language.

• The jsbuilder folder was removed, now in version 5 we will use Sencha Cmd to build and compress our source code.
• The ext*-dev.js file was removed in version 5, as according to Sencha, there was much confusion about the use of these files. On Sencha Touch these files were merged and following the same pattern as Sencha Touch on Ext JS. *-dev.js files and ext*-debug.js files were merged into one.

• The resources folder was removed, so now we need to use the packages folder.

• The docs folder was removed, so as of version 5, developers should check the documentation and guides at http://docs.sencha.com/. Also, there is an alternative to download the offline documentation selection (offline docs) from the link in the documentation menu:

Now that you have a basic understanding of the downloaded files and folders, we can advance to the next step of "getting started."

Looking at the whole picture
Before we start writing code, you need to learn and understand a few concepts first. Ext JS is divided into three layers, as shown in the following screenshot. The purpose of these layers is to share code with Sencha Touch, a framework to create mobile web applications.
In the **Ext Foundation** layer, the `Ext` object is created, as well as some useful utilities and the class system that allows us to extend classes, override methods and properties, add mixins and configurations to classes, and many more things.

To understand more about mixins, see [http://docs.sencha.com/extjs/5.1.1-apidocs/#!/api/Ext.Mixin](http://docs.sencha.com/extjs/5.1.1-apidocs/#!/api/Ext.Mixin).

The **Ext Core** layer contains the classes that manage the **Document Object Model (DOM)**, the setting and firing of events, support for Ajax requests, and classes to search the DOM using CSS selectors. Also now, part of Ext.Core is the data package (classes related to data such as fields, store, and so on). As of version 5.1, Ext JS Core will have common shared code with Sencha Touch (the next major release).

Finally, the **Ext JS 5** layer contains all the components, widgets, and many more features that you're going to be learning about in this book.

### Our first program

We need to set up our workspace to write all the examples of this book. Let's create a folder named `learning-ext-5`. For now, we don't need a web server to host our examples, but in the following chapters we are going to use Ajax; therefore, it's a good idea to use your favorite web server to host our code from these first examples.

In our new folder, we are going to create folders that contain the examples for each chapter in this book. At this point, we have a folder called `chapter_01` that corresponds to this chapter and other called `extjs-5.1.1` that contains the Ext JS framework. Both folders are located on the same level.
Inside the chapter_01 folder, we're going to create a file called myfirstapp.html, where we need to import the Ext library and create a JavaScript file called app.js that will contain our JavaScript code:

Open the myfirstapp.html file in your favorite editor and type the following code:

```html
<!doctype html>
<html>
<head>
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta charset="utf-8">
<title>My first application</title>
<!-- Importing the stylesheet (theme neptune) -->
<link rel="stylesheet" type="text/css" href="../ext-5.1.1/build/packages/ext-theme-neptune/build/resources/ext-theme-neptune-all.css">
<!-- Importing the Extjs library -->
<script src="../ext-5.1.1/build/ext-all.js"></script>
<!-- Importing overrides Js code special for theme neptune -->
<script src="../ext-5.1.1/build/packages/ext-theme-neptune/build/ext-theme-neptune.js"></script>
<!-- Importing our application -->
<script type="text/javascript" src="app.js"></script>
```
Writing the Ext JS code

The previous code shows how to import the library for a development environment. First, we import the stylesheet that is located at `ext-5.1.1/build/packages/ext-theme-neptune/build/resources/ext-theme-neptune-all.css`. The second step is to import the whole library from `ext-5.1.1/build/ext-all.js`. The third step is to import a JavaScript file that contains overrides so the theme can work properly (specific adjustments on this theme).

Now we're ready to write our code in the `app.js` file.

Before we can start creating widgets, we need to wait until the DOM is ready to be used and Ext JS is loaded and parsed. Ext JS provides a function called `Ext.onReady`, which executes a callback automatically when all the nodes in the tree can be accessed. Let's write the following code in our `app.js` file:

```javascript
Ext.onReady(function(){
    alert("This is my first Extjs app !");
});
```

An alternative to this code can also be:

```javascript
Ext.application({
    name : 'MyFirstApplication',
    launch : function() {
        Ext.Msg.alert("Hello"," my first Ext JS app");
    }
});
```

One of the advantages of using Ext JS is that the library only uses one single object in the global scope called `Ext` to allocate all the classes and objects within the framework.
An Introduction to Ext JS 5

If you open the HTML file in your favorite browser, you will see something like the following screenshot:

Feel free to use your favorite browser to work through the examples in this book. I recommend you use Google Chrome because it has more advanced developer tools and it's a fast browser. If you are a Firefox fan, you can download the Firebug plugin; it's a powerful tool that we can use for debugging in Firefox.

If for some reason we can't see the alert message in our browser, it's because we haven't defined the correct path to the `ext-all.js` file. If you look at the JavaScript console, we'll probably see the following error:

```
Uncaught ReferenceError: Ext is not defined
```

This means that the `ext-all.js` file is not imported correctly. We need to make sure everything is correct with the path and refresh the browser again.
Adding interaction to the program

Now that we know how to execute code when the DOM is ready, let's send an alert message from the Ext library. Using the Ext.MessageBox alias Ext.Msg object, we can create different types of messages such as an alert, confirmation, prompt, progress bar, or even a custom message:

Ext.onReady(function() {
  //alert("This is my first Extjs app !");
  Ext.Msg.alert("Alert","This is my first Ext JS app !");
});

The output for the preceding lines of code is shown in the following screenshot:

If you're not getting any errors in the JavaScript console but still you can't see the message on the screen, as seen in the preceding screenshot, make sure you have inserted the stylesheet correctly.

In this case, we're using the alert method of the Ext.Msg object. The first parameter is the title of the message and the second parameter is the content of the message. That was easy, right? Now let's create a confirmation dialog box:

Ext.onReady(function() {
  Ext.Msg.alert("Alert","This is my first Ext JS app !");
  Ext.Msg.confirm("Confirm","Do you like Ext JS 5?");
});
We use the `confirm` method to request two possible answers from the user. The first parameter is the title of the dialog box and the second parameter is the question or message we want to show to the user:

Before the confirmation dialog box appeared, there was an alert that didn't show up. One important thing to keep in mind is that the messages and alerts from the Ext library don't block the JavaScript loop, unlike the native browser dialog box. This means that if we add another alert or custom message after the `confirm` method is called, we will not see the confirmation dialog box anymore.

So far, we have shown a confirmation dialog box requesting two possible answers from the user, but how can we know the user's response in order to do something according to the answer? There's a third parameter in the confirmation dialog box, which is a callback function that will be executed when the user clicks on one of the two answers:

```javascript
Ext.onReady(function()
    Ext.Msg.alert("Alert","This is my first Ext JS app !");
    Ext.Msg.confirm("Confirm","Do you like Ext JS 5?",
        function(btn){
            if (btn === "yes") {
                Ext.Msg.alert("Great!","This is great!");
            } else {
                Ext.Msg.alert("Really?","That's too bad.");
            }
        });
});
```
The callback function is executed after the user clicks on the **Yes** or **No** button or closes the confirmation dialog box. The function receives the value of the clicked button as a parameter, which is **Yes** or **No**; we can do whatever we want inside the callback function. In this case, we're sending a message depending on the given answer. Let's refresh our browser and test our small program to watch our changes. Confirmations are usually asked when a user wants to delete something, or maybe when they want to trigger a long process, basically anything that has only two options.

## Tools and editors

Before we go any further, it's important we use some tools in order to be more productive when building our code and our applications. There are many editors in the market we can use to write code. Let's review some of them that will be useful in this book.

### XAMPP or WAMP

XAMPP is an open source distribution of Apache that contains MySQL, PHP, and Perl, and is easy to install and easy to use. XAMPP can provide us with a local web development environment that is easy to handle without the need to test on a public server or hosting.

XAMPP is available for Windows (32-bit), Linux, and OS X versions and you can download XAMPP at [https://www.apachefriends.org/index.html](https://www.apachefriends.org/index.html).

WAMP is another free package containing Apache, MySQL, and PHP, especially designed for the Windows OS. You can get it at [http://www.wampserver.com/en/](http://www.wampserver.com/en/). WAMP comes in 32-bit and 64-bit versions according to your Windows OS.

In order to test the code or review some Ext JS examples, we will need a web server (Apache) or IIS to get the proper functionality and AJAX responses that the examples and code require.
Aptana

The Aptana editor is an IDE from Appcelerator. It's based on Eclipse but optimized for web applications. It's an open source project and free of charge.

Among other IDEs, Aptana contains an autocomplete functionality for JavaScript and Ext JS, a JavaScript validator, a CSS and HTML validator, a JavaScript debugger, Bundles, and more:

Aptana is a great tool when working with Python, Ruby, or PHP as the backend of our projects. It contains tools to work with those out-of-the-box languages and also contains tools to deploy your application in the cloud using Heroku or Engine Yard.
Sencha Architect

The Sencha Architect desktop application is a tool that will help you design and develop an application faster than coding it by hand. The idea is to drag and drop the components into a canvas, and then add the functionality. The Sencha Architect desktop application is a product from Sencha Inc. that aims to help developers define components with a few clicks. We can create an Ext JS or Sencha Touch project. We can get a free trial from the official website of Sencha. We can also buy the license there.

The current Sencha Architect version is 3.1.0. This works for Ext JS versions 4.x and 5.x. Previous versions of Sencha architect, such as 2.x will not work for Ext JS 5.

One of the advantages that Architect has is that you can drag-and-drop components into the work zone and this tool will create (generate) the basic code of the components and/or the part(s) you are working with.
Also, you can switch to the code view and see the results so far (some parts of the Ext JS classes/components can be edited but not all of them). You can see the code view in the following screenshot:

The intention of this book is not to teach you about Sencha Architect, it's for you to understand and learn Ext JS and create applications without the need to use Sencha Architect. Later, if you feel you need to use Sencha Architect or want to give a try, it may be easier once you know the basics and essentials that are covered in this book.
What's new in Ext JS 5?

Ext JS 5 introduces a great number of new features, and most of them will be covered in the upcoming chapters when we have a closer look, but for the moment we will briefly mention some of the significant additions in version 5:

- **Tablet support and new themes**: This has introduced the ability to create apps compatible with touch-screen devices (touch-screen laptops, PCs, and tablets). The Crisp theme is introduced and is based on the Neptune theme. Also, there are new themes for tablet support, which are Neptune touch and Crisp touch.

- **New application architecture – MVVM**: Adding a new alternative to MVC Sencha called MVVM (which stands for Model-View-ViewModel), this new architecture has data binding and two-way data binding, allowing us to decrease much of the extra code that some of us were doing in past versions. This new architecture introduces:
  - Data binding
  - View controllers
  - View models

- **Routing**: Routing provides deep linking of application functionality and allows us to perform certain actions or methods in our application by translating the URL. This gives us the ability to control the application state, which means that we can go to a specific part or a direct link to our application. Also, it can handle multiple actions in the URL.

- **Responsive configurations**: Now we have the ability to set the `responsiveConfig` property (new property) to some components, which will be a configuration object that represents conditions and criteria on which the configurations set will be applied, if the rule meets these configurations. As an example:

```javascript
responsiveConfig: {
  'width > 800': { region: 'west' },
  'width <= 800': { region: 'north' }
}
```

- **Data package improvements**: Some good changes came in version 5 relating to data handling and data manipulation. These changes allowed developers an easier journey in their projects, and some of the new things are:
  - Common Data (the Ext JS Data class, `Ext.Data`, is now part of the core package)
An Introduction to Ext JS 5

- Many-to-many associations
- Chained stores
- Custom field types

- **Event system**: The event logic was changed, and is now a single listener attached at the very top of the DOM hierarchy. So this means when a DOM element fires an event, it bubbles to the top of the hierarchy before it's handled. So Ext JS intercepts this and checks the relevant listeners you added to the component or store. This reduces the number of interactions on the DOM and also gives us the ability to enable gestures.

- **Sencha Charts**: Charts can work on both Ext JS and Sencha Touch, and have enhanced performance on tablet devices. Legacy Ext JS 4 charts were converted into a separate package to minimize the conversion/upgrade. In version 5, charts have new features such as:
  - Candlestick and OHLC series
  - Pan, zoom, and crosshair interactions
  - Floating axes
  - Multiple axes
  - SVG and HTML Canvas support
  - Better performance
  - Greater customization
  - Chart themes

- **Tab Panels**: Tab panels have more options to control configurations such as icon alignment and text rotation. Thanks to new flexible Sass mixins, we can easily control presentation options.

- **Grids**: This component, which has been present since version 2.x, is one of the most popular components, and we may call it one of the cornerstones of this framework. In version 5, it got some awesome new features:
  - Components in Cells
  - Buffered updates
  - Cell updaters
  - **Grid filters** (The popular "UX" (user extension) has been rewritten and integrated into the framework. Also filters can be saved in the component state.)
  - Rendering optimizations
• **Widgets**: This is a lightweight component, which is a middle ground between `Ext.Component` and the Cell renderer.

• **Breadcrumb bars**: This new component displays the data of a store (a specific data store for the tree component) in a toolbar form. This new control can be a space saver on small screens or tablets.

• **Form package improvements**: Ext JS 5 introduces some new controls and significant changes on others:
  
  - **Tagfield**: This is a new control to select multiple values.
  
  - **Segmented buttons**: These are buttons with presentation such as multiple selections on mobile interfaces.

  - **Goodbye to TriggerField**: In version 5, TriggerField is deprecated and now the way to create triggers is by using the Text field and implementing the triggers on the TextField configuration. (TriggerField in version 4 is a text field with a configured button(s) on the right side.)

  ![Form with TriggerField](image)

  - **Field and Form layouts**: Layouts were refactored using HTML and CSS, so there is improvement as the performance is now better.

• **New SASS Mixins**([http://sass-lang.com/](http://sass-lang.com/)): Several components that were not able to be custom-themed now have the ability to be styled in multiple ways in a single theme or application. These components are:

  - `Ext.menu.Menu`
  
  - `Ext.form.Labelable`
  
  - `Ext.form.FieldSet`
  
  - `Ext.form.CheckboxGroup`
  
  - `Ext.form.field.Text`
  
  - `Ext.form.field.Spinner`
  
  - `Ext.form.field.Display`
  
  - `Ext.form.field.Checkbox`
• **The Sencha Core package**: The core package contains code shared between Ext JS and Sencha Touch and in the future, this core will be part of the next major release of Sencha Touch. The Core includes:
  - Class system
  - Data
  - Events
  - Element
  - Utilities
  - Feature/environment detection

**Summary**

In this chapter, you learned how to set up and install Ext JS 5 and Sencha Cmd too. You also learned about the content of the folders included in the Ext JS 5 SDK, and the differences between folders in version 4 and version 5. We mentioned some useful tools that will be essential for this book; however, many developers may not like them or may not feel comfortable with these tools, so feel free to use your favorite tools and editors.

Through the course of this book, you are going to learn about the use of Ext JS 5, and we will cover the most important parts, features, and components, and many more classes and tools. At the end of this book, we will complete a small real-world application so you can get an idea of how to implement Ext JS for your next and future projects.

In the next chapter, we will focus on the Ext JS 5 core concepts and DOM manipulation.
Where to buy this book

You can buy Learning Ext JS Fourth Edition from the Packt Publishing website.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

Click here for ordering and shipping details.