



Sencha, Inc.

<u>www.sencha.com</u> 525 University Avenue, Suite 23 <u>Palo Alto, CA 94301</u>

Copyright © 2010, Sencha, Inc. All Rights Reserved.

Contents

What is Sencha Designer?	
Additional Information	
Building a Web App UI with Designer	
Navigating Designer	
Shortcuts	
Anatomy of an Designer UI	
Laying Out UI Components with Designer	. 3
Layout Options	
auto	4
absolute	. 4
accordion	4
anchor	. 4
border	. 5
card	. 5
column	. 6
fit	. 6
form	. 7
hbox	. 7
table	. 8
vbox	. 9
Adding Components	
Positioning Components	
Configuring Components	
Using Templates	
Connecting to Data	
Exporting a Project	
Attaching Event Handlers to UI Components	

What is Sencha Designer?

Designer is a graphical user interface builder for Ext JS Web applications. Its easy-to-use drag-anddrop environment enables you to quickly prototype your application's interface components, connect the interface components to your data, and export well-formed, object-oriented code for each component.

Programmers and non-programmers alike can use Designer to collaborate on an application's design, which helps get projects started faster and enables faster iteration. With Designer, you can:

- Quickly and easily build complex forms.
- Change component layouts and swap control types with the click of a button.
- Focus on writing implementation code, rather than boilerplate UI code.

Additional Information

For more information about Designer and Ext JS:

- Watch the <u>Designer Demo</u> for a quick introduction to Designer.
- For information about the latest Designer release and updates, see the <u>Designer Changelog</u>.
- If you're new to Ext JS, see the component and container model information in the <u>Ext JS</u> <u>Overview</u>.
- For the details about any Ext JS class or method, see the Ext JS API Reference.

Getting Started with Designer Building a Web App UI with Designer

Designer is designed to be used in conjunction with your existing development environment and tools. It's not a replacement for Eclipse or your favorite text editor. The code generated by Designer can be imported into your existing IDE, and you can edit the UI implementation files outside of Designer from within your IDE or with the editor of your choice.

When using Designer, you:

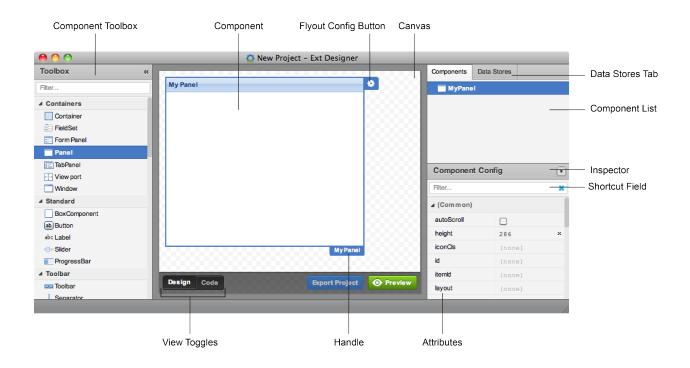
- 1. Lay out your UI components on the Designer canvas.
- 2. Configure the components.
- 3. Connect to your data stores.
- 4. Export your project.
- 5. Implement your event handling and custom methods in the generated .js file.

Important! When you export your project for the first time, two Javascript files are created for each top-level component in your project. The file with the .ui.js extension contains the base class for the UI component. You extend this base class in the file with the .js extension to implement your event handlers and custom functions. DO NOT modify the .ui.js file directly, it will be overwritten whenever you modify and export your project.

You can iterate through this process until you are satisfied with your UI. As long as you only make changes to the .js file, you can regenerate the UI code as many times as you want.

Navigating Designer

When you launch Designer, it automatically displays a new project with a blank canvas.



- Toolbox—contains all of the components you can use to build your UI. These correspond to standard Ext JS classes. For more information about each class, see the Ext JS API Reference. You can drag and drop components from the toolbox onto the canvas.
- Canvas—provides a design space for you to assemble your UI. You can resize and rearrange components you've added to the canvas and edit component titles and labels. (You can only reposition absolute-positioned components, for more information see <u>Laying Out UI</u> <u>Components with Designer</u>.)
- Components—lists all of the components you've added to your project. From the Components tab, you can select, rearrange, duplicate, transform, and delete components you've added to the canvas. You can view and modify the selected component's settings in the Component Config pane.
- Data Stores—shows the data sources you've added to your project. From the Data Stores tab, you can add new JSON, Array, XML, and Direct data sources, add and remove a source's data fields, and select, duplicate, or delete existing sources. You can view and modify the selected data store's settings in the Component Config pane.
- Component Config—view and modify settings for the selected component or data store.

As you add components to the canvas, you can see what they will look like in a web browser by clicking the Preview button below the canvas. You can see what the generated Javascript code looks like by toggling between the Design and Code views. You save the generated code to an external file by clicking the Export button. Note that you'll need to save your new project before you can export it.

Shortcuts

Designer provides a number of navigation and configuration shortcuts. You can:

- Double-click components in the Toolbox to add them to the canvas.
- Tab between inline editable fields on the canvas.
- Locate particular attributes in the Component Config inspector by typing in the Shortcut Field.
- Set attribute values using the Shortcut Field by entering the name of the attribute followed by a colon and the value you want to set. For example, *title: Car Listings*.

Anatomy of an Designer UI

When you lay out UI components with Designer, you drag a container such as a Window or FormPanel onto the canvas and add components to the container. By adding additional top-level containers to your project, you can lay out the different parts of your UI as separate entities. When you export your project, each top-level container is represented by a class with the code for that class in a separate file. This gives you flexibility in how you assemble the elements on your web pages, makes it possible to reuse common components, and makes it easier to maintain your implementation code.

Laying Out UI Components with Designer

Designer leverages the powerful layout capabilities of Ext JS to simplify the creation of complex forms and make it easy to switch between alternate layout options.

Layout Options

When you set the layout on a container, it controls how Ext JS lays out the components within that container. You can switch between layout options by clicking a Container's flyout config button and selecting a different layout.

ExtJS provides a number of basic container layouts. Some support specific, commonly-used presentation models such as accordions and cards, while others provide more general-purpose models that can be used for a variety of applications.

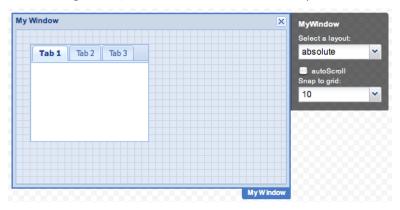
auto

If no other layout is set for a container, it defaults to the auto layout. For general purpose containers such as a Panel, this means child components will be rendered sequentially. Note that some containers are automatically configured to use a particular layout. For example, FormPanel defaults to the form layout and TabPanel defaults to the card layout.

absolute

Arranges components using explicit x/y positions relative to the container. This enables you to explicitly reposition and resize components within the container. While this gives you fine-grained control over the layout, keep in mind that absolute-positioned components remain fixed even if their parent container is resized.

When you use the absolute layout, Ext Designer displays a grid within the container. By default, components are snapped to the grid as you reposition them. You can change the grid size or disable the grid by clicking on the container's flyout config button. The grid is only displayed as a layout guide in the Design view, it is not visible when the component is rendered.



accordion

Arranges panel components in a vertical stack where only one panel is expanded at a time. Only panels (including FormPanel and TabPanel) can be added to a container that uses the accordion layout.

anchor

Arranges components relative to the sides of the container. You can specify the width and height of child components as a percentage of the container or specify offsets from the right and bottom edges of the container. If the container is resized, the relative percentages or offsets are maintained.

My Grid Column Co		Select a layout:
Column Co		
	olumn Column	anchor 🗸
cell ce	ell cell	autoScroll
cell ce	ell cell	
cell ce	ell cell	

border

Arranges panel components in a multi-pane layout. Panels are arranged in the container by assigning them to one of five regions: North, South, East, West, or Center. A container that uses the border layout has to have a child assigned to the Center region. The center panel is automatically sized to fit the available space. You can resize the North, South, East, and West panes on the canvas by clicking and dragging the right or bottom edge of the panel.

You can make any of the panels in a border layout collapsible by enabling the collapsible attribute. When rendered, the child panels automatically resize when the container is resized.

North			MyPanel Select a layout:
West	Center	East	border V autoScroll
South		Nu: Doco1/0	
		My Panel 12	

card

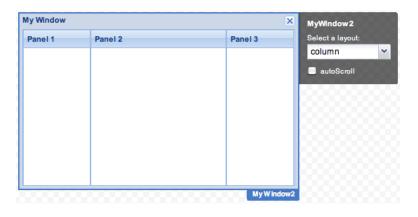
Arranges multiple child components but only one component is visible at a time. This layout can be used to step through a series of components and is commonly used to create wizards.

To specify the component that you want to make visible, you need to call setActiveItem. Typically, you attach this behavior to a navigation UI such as Previous and Next buttons in the footer of the container.

Getting Started with Designer Card 1 MyPanei18 Select a layout v card autoScroll ve Item ٨. 0 - Card 1 0 - Card 1 1 - Card 2 2 - Card 3 Prev Next My Panel18

column

Arranges components in a multi-column layout. The width of each column can be specified either as a percentage (column width) or an absolute pixel width (width). The column height varies based on the contents. You can enable autoscroll so it's possible to scroll to view column contents that exceed the container height.



fit

Expands a single child component to fill the available space. For example, you might use this to create a dialog box that contains a single TabPanel. If the container is a type of panel component, you can also add a Toolbar to it.

My Window X Tab 1 Tab 2 Tab 3	MyWindow1 Select a layout:
Hello!	fit 🗸
Cancel Save	
My Window1	

form

Arranges a collection of labelled form fields. A FormPanel uses the form layout by default.

/ Form			MyForm		
My Fields			Select a form	layout:	~
Label:			auto	Scroll	×.
Label:					2
My Fields					
My Fields — Label:					
-					

hbox

Arranges the child components horizontally. Setting the alignment of the container to stretch causes the child components to fill the available vertical space. Setting the flex attribute of the child components controls the proportion of the horizontal space each component fills.

align: stretch —	My Form Participant Name: Birthdate: Phone (hm): Phone (cell): Email Address:	Emergency Contact Name: Relationship: Phone (hm): Phone (cell): Email Address:		MyForm Select a layout: hbox autoScroll Alignment: stretch	×
L	flex: 1	 flex: 1	My Form		

table

Arranges components in an HTML table. You specify the number of columns in the table and can create complex layouts by specifying the rowspan and colspan attributes on the child components.

My Panel	MyPanel21			
Cell 1	Cell 2	Rowspan 2	Cell 4	Select a layout: table 🗸
Colspan 2			Cell 6	
	000000	000000	My Panel21	

vbox

Arranges the child components vertically. Setting the alignment of the container to stretch causes the child components to fill the available horizontal space. Setting the flex attribute of the child components controls the proportion of the vertical space each component fills.

	My Form	MyForm
	Participant Name:	Select a layout: VDOX
	Birthdate:	autoŠcroli Alignment:
flex: 1—	Phone (hm): Phone (cell): Email Address:	stretch 🗸
flex: 1—	Emergency Contact Name : Relationship: Phone (hm): Phone (cell): Email Address:	
	MyFom	

Adding Components

To assemble your application's UI, you drag components from the Toolbox onto the canvas. Designer ensures components are nested properly and won't let you add incompatible components to a container. For example, you can't drop a Window or Viewport component into a Container.

For example, to assemble the Car Listings UI shown in the <u>Sencha Designer demo</u>:

1. Drag a Panel container onto the canvas. This is the top-level component for the Car Listing application.

00	🙆 Ne	w Project - Ext Designer				
Toolbox «		15		Components I	Data Stores	
Filter	My Panel		•	MyPanel	0	
Containers			100			
Container			100			
FieldSet			1.22			
E Form Panel			100			
Panel			1.0	Component	Cartin	0
TabPanel			1.52	Component	Comig	•
View port				1		×
Window			1.00	> (Common)		
⊿ Standard			1.01	⊿ (Designer)		
BoxComponent						
ab Button	00000000000	0000000000	MyPanel	jsClass	MyPanel	x
abe Label				userXType	(none)	
- Slider			888888	⊿ Ext.BoxComp	ponent	
ProgressBar			000000	autoHeight		
a Toolbar			000000	autoWidth		
📼 Toolbar			8888888	height	250	x
Separator			000000	pageX	(none)	
Spacer			555555	and the second		
abc Text Item	Design Code	Export Project	O Preview	pageY	(none)	
→ Fill	0000	Export Froject		width	400	x
Paging Toolbar						

2. Drag a Grid Panel into the Panel container. The grid panel will display the available car listings and enable the user to select a listing to view.

Toolbox	«					Components D	Data Stores	
Filter	My Panel				123	WyPanel		
I Paging Toolbar	My Grid				•	🔻 📰 My Grid	1	
ButtonGroup	Column	Column	Colu	mn		Colur	mn	
⊿ Menu	cell	cell	cell		My Grid	Colur		
Menu					my chu	Colur	m	
Menu Item					1.00	Component (Config	
⊿ Tree					1.50		Joing	
- Async Tree Node					120	Filter		•
Async Tree Loader					100	⊳ (Common)		
💬 Tree Panel					193	⊿ (Designer)		
⊿ Grid					0.0			
III Grid Panel		0000	00000	0.000.0000	000000	autoRef	(none)	
Editor Grid Panel	000000					▲ Ext.BoxComp	onent	
Grid Column	1000000					autoHeight		
Boolean Column	1000000					height	(none)	
1 Number Column	1000000					pageX	(none)	
Date Column	1000000					pageY	(none)	
Form Fields	1000000					width	(none)	
Checkbox	-			-		x	(none)	
Date Field	Design Cod	le		Export Project	• Preview	У	(none)	

Getting Started with Designer 3. Drag another Panel into the Panel container. This panel will display the car details for the listing selected in the Grid Panel.

000		01	New Project -	Ext Designer				
Toolbox	«					Components D	Data Stores	
Filter	My Panel					WyPanel		
▲ Containers	My Grid				5	T My Grid		
Container	Column	Column	Colum	n		Colu		
E FieldSet	cell	cell	cell			Colu		
E Form Panel	My Panel				•			
Panel					My Panel	My Pan	iel	
🛅 TabPanel					- B	0	0 1	-
View port						Component (Config	
Window						Filter		*
▲ Standard					1.5	⊳ (Common)		1
BoxComponent								
ab Button						⊿ (Designer)		_
abe Label	000000					autoRef	(none)	
- Slider	1000000					⊿ Ext.BoxComp	onent	
ProgressBar						autoHeight		
⊿ Toolbar	- 000000					autoWidth		
Separator	100000					height	(none)	
Spacer	1000000					pageX	(none)	
abc Text Item		00000	100000	00000	000000	pageY	(none)	
→ Fill	Design Cod	de		Export Project	O Preview		(none)	
Paging Toolbar	Contraction and the Address of the A							

Positioning Components

By default, components are laid out using relative positioning. The best way to control the position of the elements on the canvas is to set the layout options on the containers and adjust the attributes that adjust the relative positions of each component.

Note: If you choose the Absolute layout option, you can drag components around on the canvas to reposition them. (Generally, this isn't what you want to do. It's normally better to rely on the Ext JS layout manager to control the relative positions of the components.)

For example, to configure the layout of the components in the Car Listings UI:

1. Click the flyout config button on the top-level panel and set the layout to vbox. This will arrange the grid and subpanel vertically. From this menu, you can also set the alignment and auto-scroll attributes.

000		Nev	w Project - Ex	t Designer					
Toolbox «							Components	Data Stores	
Filter	My Panel					MyPa	nel MyPane	1	
⊿ Containers	My Grid					Selec	t a layout: My G		
Container	Column	Column	Column			auto)	∼ m	
FieldSet	cell	cell	cell			auto		m	
Form Panel	My Panel					1	olute	-	
Panel						1.000	ordion	31	
🛅 TabPanel						anc			-
						card		Config	
Window						colu			×
▲ Standard						fit			1
BoxComponent						form			
ab Button		00000			MyPanel	hbo			
abc Label	000000					tabl	9	MyPanel	×
- Slider	00000000					vbox	<	(none)	
ProgressBar	0000000						▲ Ext.BoxCon	nponent	
4 Toolbar	000000					- I	autoHeight	D	
📼 Toolbar	0000000					21	autoWidth	0	
Separator	1000000					20	height	250	x
Spacer	10000000								^
abc Text Item	Design Code		E	port Project			pageX	(none)	
→ Fill				porcerojeci	- TIEVIE		pageY	(none)	
Paoino Toolbar									

2. Set the alignment for the top-level panel to Stretch. This will cause the subcomponents to stretch to fill the available space.

000		O Ne	ew Project - Ext Designer			
Toolbox	"			Components	Data Stores	
Filter	My Panel			MyPanel MyPai		
▲ Containers	My Grid			Select a layout: My		
Container	Column	Column	Column	vbox	¥ m	
E FieldSet	cell	cell	cell	autoScroll		
E Form Panel	My Panel				Column	
Panel				stretch	<u>~</u> 1	
🛅 TabPanel				top		
View port				stretch	Config	•
Window				stretchmax		
				b (Commo	2)	1
BoxComponent						
ab Button				MyPanel / (Designe	:)	
abe Label	000000			jsClass	MyPanel	×
- Slider	10000000			userXType	(none)	
ProgressBar				⊿ (Layout)		
⊿ Toolbar	- 000000			align	(none)	
I Toolbar	10000000			pack	(none)	
Separator Spacer				padding	(none)	
abc Text tem		00000		scrollOffse		
→ Fill	Design Code		Export Project	Preview		
Paging Toolbar				⊿ Ext.BoxCo	mponent	

3. Select the Grid Panel and set the flex attribute to 1 in the Component Config inspector.

Tip: You can type the name or first few characters of an attribute in the text field at the top of the Component Config inspector to quickly navigate to a particular attribute.

000		🐴 Nev	w Project - Ext Designe	r			
Toolbox	¢		k		Components	Data Stores	
Filter	My Panel				WyPanel		
▲ Containers	My Grid			•	🔻 🎹 My Gri	id	
Container	Column	Column	Column	100	Colu		
E FieldSet	cell	cell	cell	192	Colu		
Form Panel	cell	cell	cell	0.0	Col		
Panel	cell	cell	cell	100	My Par	nel	
🛅 TabPanel				100		0. 5	-
View port					Component	Config	•
Window					f		*
▲ Standard				P	▲ Ext.Panel		
BoxComponent	My Panel			My Grid			
ab Button					floating		
abc Label	000000			0000000	footer		
=[]= Slider	0000000			8999999	frame		
ProgressBar	100000			0000000	A Ext.layout.VE	BoxLayout	
⊿ Toolbar					flex	1	×
📼 Toolbar	000000			0000000			
Separator	10000000						
G Spacer	1000000			0000000			
abe Text Item							
→ Fill	Design Cod	ie	Export Projec	t O Preview			
Paoino Toolbar							
							1

The panel inherits the flex attribute from Ext.layout.VBoxLayout because the layout of the container is set to vbox. Setting the flex attribute of each of the components in the container to 1 will cause the components to take up the same amount of vertical space when the container is resized. (Similarly, if you wanted the subpanel to take up 2/3 of the vertical space, you could set the flex value of the panel to 2, and the flex of the grid to 1.)

4. Select the subpanel and set the flex attribute to 1.

Toolbox	"					Components	Data Stores	
Filter	My Panel				10	W MyPanel		
▲ Containers	My Grid				100	🔻 🔜 My Grid	t	
Container	Column	Column	Column		100	Cole		
FieldSet	cell	cell	cell		n	T Col		
Form Panel	cell	cell	cell		4	Coli		
Panel	cell	cell	cell		×	My Pa	nel	
TabPanel	My Panel				•	Component	Config	0
Window					- 8	f		1
4 Standard					1.0	A Ext.Panel		
BoxComponent					193			
ab Button		00000		1000	My Panel	footer		
abe Label	666666				00000	frame		
- Slider	10000000					Ext.layout.VE	BoxLayout	
ProgressBar						flex	1	×
⊿ Toolbar								
I Toolbar	000000							
Separator	1000000							
G Spacer	10000000							
abc Text Item	Design Code		Erno	rt Project	• Preview			
→ Fill	-ooign Cour		Expo	re riojaci				

Configuring Components

You can edit component attributes such as titles and labels directly. Simply double-click the text you want to modify and start typing. The Component Config inspector enables you to set all possible attributes for the selected component.

For example, you can directly set the title and column heading attributes for the Car Listings application:

1. Double-click the title bar of the top-level Panel to edit its title and change *My Panel* to *Car Listings*. This is the same as setting the title attribute in the Component Config inspector.

00		🙆 Nev	w Project - Ext Design	ier			
Toolbox	«			▶	Components E	Data Stores	
Filter	Car Listings			•	V MyPanel		
▲ Containers	My Grid			193	T My Grid		
Container	Column	Column	Column		Colur		
E FieldSet	cell	cell	cell	0	Colur		
Form Panel	cell	cell	cell	Y	Colur		
Panel	cell	cell	cell	×	My Pane	əl	
TabPanel	My Panel				Component (Config	(F)
View port				122	Component o	ooning	
Window	- 1 K			1.0	Filter		*
▲ Standard	- 10				⊳ (Common)		1
BoxComponent ab Button					⊿ (Designer)		
ab Button	100000	-5555555		MyPanel	jsClass	MyPanel	x
= Slider	100000				userXType	(none)	
ProgressBar	000000					(none)	
4 Toolbar	1000000				⊿ (Layout)		
🚥 Toolbar	- pococo				align	stretch	х
Separator	000000				pack	(none)	
G Spacer	1000000				padding	(none)	
abc Text Item				_	scrollOffset	(none)	
→ Fill	Design Co	de	Export Proje	oct O Preview	▲ Ext.BoxComp	onent	
📼 Paging Toolbar							_

2. Double-click the column headings in the grid to set them to Manufacturer, Model, and Price. This is the same as setting the header attribute through the inspector.

00		🙆 Ne	w Project - Ex Designer				
Toolbox	"				Components [Data Stores	
Filter	Car Listings				WPanel		
▲ Containers	My Grid				T My Grid		
Container	Manufacturer	Model	Column			ufacturer	
FieldSet	cell	cell	cell	0	Mod		
Form Panel	cell	cell	cell	V	Colu		
Panel	cell	cell	cell	•	My Pane	el	
🛅 TabPanel	My Panel				0	Orafia	-
View port					Component	Connig	•
Window				1.0	Filter		
▲ Standard	- 1			1.00	⊳ (Common)		1
BoxComponent				100	⊿ (Designer)		
ab Button							
abe Label	000000				autoRef	(none)	
=[]= Slider	6000000				A Ext.grid.Colu	mn	
ProgressBar	- 0000000				align	(none)	
✓ Toolbar ■ Toolbar	- 000000				CSS	(none)	
Separator	1000000				dataIndex	string	x
Spacer	6666666				fixed		
abc Text Item			00000000000	000000	header	Column	×
→ Fill	Design Code		Export Project	• Preview	hidden	0	
Paging Toolbar	And the second second second second					0	_

You can set the rest of the component attributes in the Component Config inspector:

1. Remove the title bars from the grid and subpanel by selecting each component and clicking the delete icon (x) to the right of the title attribute in the inspector. Now, the only title bar visible is the Car Listings title.

000		O Ne	ew Project - Ext Designer				
Toolbox «					Components I	Data Stores	
Filter	Car Listings				V MyPanel		
▲ Containers	My Grid			•	🔻 📰 My Gri	d	
Container	Manufacturer	Model	Price	100		ufacturer	
E FieldSet	cell	cell	cell	n	Mod		
E Form Panel	cell	cell	cell	4	T Price		
Panel	cell	cell	cell	×	My Pan	el	
🛅 TabPanel	My Panel			My Grid	Component	Config	
View port				1004			
Window				100	Filter		*
▲ Standard				100	⊿ (Common)		
BoxComponent				223	height	(none)	
ab Button abc Label		10000			iconCls	(none)	
In the second se	0000000			5555556	id	(none)	
ProgressBar	0000000			000000	itemid	(none)	
4 Toolbar	C. C	-			store	(none)	x
Contraction	Design Code		Export Project	Preview	title	My Grid	Ť
							1

2. Select the Car Listings panel and set the jsClass attribute to CarMasterDetail. This will be the name of the component in the generated code.

0 0		0	New Project - I	Ext Designer				
Toolbox	*					Components D	lata Stores	
Filter	Car Listings				•	🔻 🗖 CarMaster	Detail	
▲ Containers	Manufacturer	Model	Price		55555	T MyGrid1		
Container	cell	cell	cell		10000	📰 Manu	facturer	
FieldSet	cell	cell	cell		0000	Mode	el	
Form Panel	cell	cell	cell		55555	T Price		
Panel					10000	MyPanel	3	
🛅 TabPanel					10000	Component C	Config	5
View port					55555		, in the second s	- C
Window					0000	Filter		
Standard					0000	⊿ (Designer)		
BoxComponent					55555	jsClass	CarMasterDetail	
ab Button		00000		CarMasterDetal	2000	userXType	(none)	
abe Label	000000				0000	⊿ (Layout)		
- Slider	10000000				88888	align	stretch	x
ProgressBar		100000	000000	000000000	0000	pack	(none)	
Toolbar	Design Code			Export Project	Preview			
🚥 Toolbar						padding	(none)	

Note: You can toggle between the design and code views with the buttons below the canvas.

3. Enable the <u>frame</u> attribute of the Car Listings panel. Instead of the plain 1px square borders, this renders the panel with additional styling, including rounded corners.

Getting Started with Designer

Toolbox	«					Components Data	Stores	
Filter	Car Listings				•	CarMasterDe	tail	
Containers	Manufacturer	Model	Price		10000	WyGrid1	(SD)	
Container	cell	cell	cell		10000	Manufa	cturer	
FieldSet	cell	cell	cell		0.000	Model		
Form Panel	cell	cell	cell		5555	T Price		
Tanel					10000	MyPanel3		
🛅 TabPanel					0.000	Component Co	nfig	
View port					5555			3
Window					10000	Filter		
Standard					0.000	footer		
BoxComponent					10000	frame	1	
ab Button				CarMasterDet	al	header	N	
abe Label	10000000					headerAsText	M	
-O-Slider	1000000					hideCollapseTool		
ProgressBar Toolbar		00000	000000		_	html	(none)	
Ioolbar Ioolbar	Design Code			Export Project	O Preview	iconCls	(none)	

4. Configure auto references for the components so you can directly reference them in your code without worrying about how they are nested. Set the autoRef for the Grid Panel to *grid* and the autoRef for the subpanel to *detail*.

00		0	New Project - I	xt Designer				
Toolbox	«					Components	Data Stores	
Filter	Car Listings				- 2000	T CarMast	erDetail	
▲ Containers	Manufacturer	Model	Price		0.000	T MyGi		
Container	cell	cell	cell		100000		lanufacturer	
E FieldSet	cell	cell	cell		10000	II M		
E Form Panel	cell	cell	cell		0000	I P		
Tenel					0000	MyP	anel3	
TabPanel					0	Componer	nt Config	
View port					10000			
Window					100000	auto		*
▲ Standard					0000	⊿ (Common)	l.	
BoxComponent					55555	autoScroll	0	
ab Button			0 0 0 0 0 0	MyPa	nel3	⊿ (Designer)		
abe Label	10000000				000000	autoRef	detail	
- Slider	10000000				000000			
ProgressBar		00000	000000	000000	000000	▲ Ext.BoxCor	nponent	
4 Toolbar	Design Code			Export Project		autoHeight		
💷 Toolbar						autoWidth		

5. To add some space padding around the contents of the subpanel, select the panel, type p to jump to the padding attribute, and set the value to 10. (This is the typical CSS padding attribute.)

00		0	New Project -	Ext Designer				
Toolbox	«					Components	Data Stores	
Filter	Car Listings					CarMaste	rDetail	
▲ Containers	Manufacturer	Model	Price			V MyGrid		
Container	cell	cell	cell				nufacturer	
E FieldSet	cell	cell	cell			II Mo		
E Form Panel	cell	cell	cell			T Pric		
Panel						MyPar	nel3	
TabPanel					•	Component	Config	
View port								
Window					10000	p		×
4 Standard					10000	⊿ Ext.BoxCom	ponent	
BoxComponent					199999	pageX	(none)	
ab Button				MyPa	inel3	pageY	(none)	
abe Label	0.000000					⊿ Ext.Panel		
	1000000					padding	10	
ProgressBar		00000	00000	1000000	000000	padding		
4 Toolbar	Design Code			Export Project	O Preview			
📼 Toolbar	Contraction of the second							

Using Templates

You can use <u>templates</u> to dynamically display information from a data store in a panel component. A template is an HTML fragment that can contain variables that reference fields in a data store. Templates also support auto-filling of arrays, conditional processing, math functions, and custom functions.

Variables are enclosed in curly braces. For example, {manufacturer} references the data field called *manufacturer*. You can also specify formatting functions to control how the data is displayed. For example, {price:usMoney} uses the usMoney format to prepend a dollar sign and format the number as dollars and cents. See <u>Ext.util.Format f</u>or the full range of available formatting functions.

The Car Listings application uses a template to display the detail information for the selected listing. The image and wiki URL are pulled in from data fields in the cars.json data store. (See <u>Connecting to</u> <u>Data</u> for information about how to attach a data store.)

To configure the template:

1. Click the flyout config button on the subpanel and then click Edit Template to add a template for the detail information. The body of the component becomes an editable text area.

Toolbox	"				Components	Data Stores	
Filter	Car Listings				CarMaste		
Containers	Manufacturer	Model	Price		🔻 🛄 MyGri	id1	
Container	cell	cell	cell	12000		anufacturer	
FieldSet	cell	cell	cell	10000	II Ma		
E Form Panel	cell	cell	cell		🔲 Pri		
Panel				10000	MyPa	inel3	
TabPanel				MyPanel3	Component	t Config	
View port				Select a layo			
Window				auto	~		
Standard				autoScrol	⊿ (Common)		
BoxComponent					autoScroll	0	
ab Button				Done Editing	height	(none)	
abe Label	0.000000			100000000000000000000000000000000000000	iconCls	(none)	
-0= Slider	1000000			00000000	id	(none)	
ProgressBar		00000	000000000000	0000000	itemld	(none)	
Toolbar	Design Code		Export Projec	t O Preview			
💷 Toolbar	Constanting of the local division of the loc				layout	(none)	

2. Enter the HTML mark-up for the template:

```
Getting Started with Designer
<img src="cars/{img}" style="float: right" />
Manufacturer: {manufacturer}<br/>br/>
Model: <a href="{wiki}" target="_blank">{model}</a><br/>Price: {price:usMoney}<br/>
```

3. When you're finished editing the template, click **Done Editing**.

Connecting to Data

You can attach data stores and bind them to the components in your UI from within Designer.

For example, the listing information displayed by the Car Listings application is read from a JSON data store called cars.json. To connect the data store and pull in the manufacturer, model, price, wiki, and image data:

- 1. Add a data store for the cars data:
 - a. Select the Data Stores tab.
 - b. Select Add Json Store from the Data Stores toolbar.

00	20		New Project - Ex	d Designer			20
Toolbox	«					Components D	Data Stores
Filter	Car Listings				10000	Add Xml Store	- 🦣 Load data Remove
Containers	Manufacturer	Model	Price			Json Store	.lm
Container	cell	cell	cell		15555	Array Store	
FieldSet	cell	cell	cell		10000	Xml Store	
E Form Panel	cell	cell	cell		0000		
Panel					2000	Direct Store	
TabPanel	<img arc="c</td><td>ars/{img}" s<="" td=""/> <td>tyle="float: r</td> <td>ight" /></td> <td>0</td> <td>Component C</td> <td>Config</td>	tyle="float: r	ight" />	0	Component C	Config	
🕂 View port	Manufacture	r: {manufact	urer} 	k">{model} <b< td=""><td></td><td></td><td>3</td></b<>			3
Window	Price: {price			x >(moder)()a/c	, 1	Filter	
4 Standard					1.0.0.0.0	⊿ (Common)	
Standard BoxComponent						⊿ (Common) autoScroll	
BoxComponent ab Button				MyPanel			(none)
BoxComponent ab Button abc Label				My Pane 2		autoScroll	
BoxComponent ab Button abc Label -D- Slider				MyPanel3		autoScroll height iconCls	(none) (none)
BoxComponent ab Button abc Label 				MyPanelS		autoScroll height iconCls id	(none) (none) (none)
ab Button abc Label 	Design Code				O Preview	autoScroll height iconCls	(none) (none)

c. Select the newly-created store and set the jsClass attribute to name the store CarStore.

Toolbox	«				Components D	Data Stores	
Filter	•	🛃 Add Json Store - 🔩 Load data Remove					
Containers	Manufacturer	Model	Price	10000	CarStore		
Container	cell	cell	cell	120000			
FieldSet	cell	cell	cell	0000			
Form Panel	cell	cell	cell	0000			
Panel				10000			
TabPanel	<img s<="" src="c</td><td>ars/{img}" td=""/> <td>tyle="float: right" /></td> <td>0000</td> <td>Component C</td> <td>Config</td> <td>1</td>	tyle="float: right" />	0000	Component C	Config	1	
		r: {manufact		Filter			
View port		ref="{wiki}"	target=" blank">{model}<	br	Filter		
	Model: <a h:<="" td=""><td>ref="{wiki}"</td><td>target=" blank">{model}</td> <	ref="{wiki}"	target=" blank">{model}	br	Filter ⊿ (Common)		
Window	Model: <a h:<="" td=""><td>ref="{wiki}"</td><td>target=" blank">{model}</td> <	ref="{wiki}"	target=" blank">{model}	br		(none)	
Window Standard BoxComponent ab Button	Model: <a h:<="" td=""><td>ref="{wiki}"</td><td>target=" blank">{model}</td> <	ref="{wiki}"	target=" blank">{model}		⊿ (Common)	(none)	3
Window Standard BoxComponent ab Button abc Label	Model: <a h:<="" td=""><td>ref="{wiki}"</td><td>target="_blank">{model}</td> < br/>	ref="{wiki}"	target="_blank">{model}		⊿ (Common) root url		•
Window Standard BoxComponent BoxComponent BoxComponent BotLabelSilder	Model: <a h:<="" td=""><td>ref="{wiki}"</td><td>target="_blank">{model}</td> < br/>	ref="{wiki}"	target="_blank">{model}		 ⊿ (Common) root url ⊿ (Designer) 	(none)	*
Window Standard BoxComponent ab Button abc Label	Model: <a h:<="" td=""><td>ref="{wiki}"</td><td>target="_blank">{model}</td> < br/>	ref="{wiki}"	target="_blank">{model}		⊿ (Common) root url		3

d. Set the storeId attribute to the same name. (The storeId is the name Designer displays in

the list of available stores.)

2. Right-click the data store and select **Add Fields > 5 fields** to add data fields to the CarStore for each field defined in cars.json.

00		0	New Project - I	Ext Designer					
Toolbox	*					Components	Data S	stores	
Filter	Car Listings	Car Listings				🛃 Add Json Store 🗕 🦣 Load data Remove			nove »
▲ Containers	Manufacturer	Model	Price			ि) Car			
Container	cell	cell	cell		0000	Í	Select		
E FieldSet	cell	cell	cell		10000		Duplic	ate	
E Form Panel	cell	cell	cell		10000		C Delete	and the second sec	
🧰 Panel	1.8				100000		Delete		
TabPanel View port Window	port Manufacturer: {manufacturer} Model: <a "="" href="">href="" {wiki}" target=" black">[model] br					Compo 1 field	Quick	add 🕨	•
4 Standard					0.000	2 fields	n)		-
BoxComponent						3 fields		(none)	
ab Button				CarMasterDe	tal	4 fields		(none)	
abe Label	0000000			0000000		5 fields	r)		
- Slider	10000000					jsClass	1		×
ProgressBar		00000	000000	0000000				CarStore	^
▲ Toolbar	Design Code			Export Project	Preview	userXType		(none)	

- 3. Configure the CarStore:
 - a. Set the url attribute to the relative path where the store will reside, cars/cars.json. This path is relative to the URL prefix specified in the Project Settings. To change the URL prefix, select Edit Preferences from the Edit Menu.

Toolbox «					Components D	ata Stores	
Filter	Grid				Add Json Store	- 🔩 Load data Remo	ove
Containers	Manufacturer	Model		_	🔻 🕅 carStore		
Container FieldSet Form Panel Panel					manufac model price w iki	turer	
🛅 TabPanel	8				Component C	onfig	
View port					Filter		2
4 Standard					⊿ (Common)		
BoxComponent	Manufacturer: {ma				root	data	х
ab Button abc Label	Price: {price:usM	wiki}" target="_bla: oney} 	nk">{model}	 	url	cars/cars.json	
- Slider					⊿ (Designer)		
ProgressBar	1				jsClass	carStore	x
Toolbar	Design Code	000000000	Description of	O Preview	userXType	(none)	
📼 Toolbar	Losign Code		Export Project	Preview	⊿ Ext.data.Json	Store	

- b. Set the root attribute to *data*.
- c. Configure the data store to load automatically by enabling the autoLoad attribute. (Otherwise, you won't see any data when you view the index.html file.)
- d. Set the name attribute of each data field in the store: manufacturer, model, price, wiki, and img.

Getting Started with Designer

Toolbox	«				Components	Data Stores
Filter	Grid			- 21	Add Json St	tore 🗕 🆏 Load data Remove
Containers Container Container Form Panel Form Panel TabPanel Window Standard	Narufacturer	Model		Price O	Componen Filter	facturer I
BoxComponent Button abc Label -0= Slider ProgressBar	Manufacturer: {manu	ki}" target="_blank">{model} <td>> </td> <td>My Grid</td> <td>name (Designer) autoRef Ext.data.Dat</td> <td>(none)</td>	> 	My Grid	name (Designer) autoRef Ext.data.Dat	(none)
Toolbar	Design Code		Export Project	O Preview	dateFormat	(none)
📼 Toolbar					mapping	(none)

4. Click the flyout config button on the grid component and select CarStore to bind the grid component to the store.

000	0	demo.xds (/Users/debadair/Desktop,	/demo/) - Ext Design	er			
Toolbox	"				Components D	Data Stores	
Filter	Grid				Add Json Store	e 🗸 🍇 Load data Ren	move »
4 Containers	Manufacturer	Model		Price	My Grid - To carStore	oad error	
Container					Select a store: 🧾 manufac	turer	
E FieldSet					CarStore ¥		
Form Panel					CarStore	l	
Panel					📕 w iki		
🛅 TabPanel				- I K	Component C	Config	
View port				- B		0	
Window					Filter		*
4 Standard				My Grid	⊿ (Common)		
BoxComponent	Manufacturer: {ma				root	data	x
ab Button	Model: <a """"""""""""""""""""""""""""""""""<="" href="" td=""><td><pre>{wiki}" target="_blank">{model}</pre></td> {onev} br/>	<pre>{wiki}" target="_blank">{model}</pre>	 br/>		url	cars/cars.j	ison×
abe Label					⊿ (Designer)		
- Slider							
ProgressBar					jsClass	carStore	×
4 Toolbar	Design Code		Export Project	O Previet	userXType	(none)	
💷 Toolbar			Baport Project	C III C III	A Ext.data.Json	Store	

- 5. Link the columns to the appropriate data fields:
 - a. Select a column in the Components list.
 - b. Set the dataIndex attribute to the name of the data field.

Toolbox	**				Components D	ata Stores		
Filter	Car Listings			CarMasterDetail				
Containers	Manufacturer	Model	Price	1.55	T MyGrid			
Container				100	Manu Manu			
FieldSet				55	Mode	-		
E Form Panel				201	III Price			
Panel				1.00	MyPanel	1		
🛅 TabPanel	<img s<="" src="ca</td><td>ars/{img}" td=""/> <td>tyle="float: right" /></td> <td>100</td> <td></td> <td></td> <td></td>	tyle="float: right" />	100					
View port		r: {manufact	urer} target=" blank">{model	s/a>shr	Component C	onfig		
Window	Price: {pri	ce:usMoney}<	br/>					
▲ Standard				100	da		*	
BoxComponent				555	⊿ Ext.grid.Colun	ın		
ab Button					dataIndex	price	×	
abe Label				100000		2.2.00		
- Slider	Design Code		Export Project	O Preview				
ProgressBar	No. of Concession, Name							

Data from the store is immediately displayed in the grid.

Exporting a Project

Exporting a project generates the Javascript files for your application. When you export, two Javascript files are created for each top-level component in your UI:

- .ui.js contains the base class that defines the component. For example, CarMasterDetailUi. You extend this base class to implement your event handler code and custom methods, you do not modify this file directly. The .ui.js files are overwritten whenever you re-export your project.
- .js is a starter file for your implementation. The .js files are generated the first time you
 export you project. You edit this file to add your event handler code and custom methods.

Important! DO NOT modify the .ui.js file generated by Designer, it will be overwritten whenever you modify and export your project.

Along with the Javascript files, Designer generates an xds_index.html file that loads the javascript and displays your app.

To export your project:

- 1. Save your changes. (You have to save your project before you'll be able to export it.)
- 2. Click the Export button below the canvas. The project will be saved to the Export Path specific in the Project Settings. (To change the location, select Edit Preferences from the Edit menu.)

Attaching Event Handlers to UI Components

You can import the files Designer generates into the editor or IDE of your choice. To add event handlers, you need to edit the .js files.

For example, to add an event handler to the Car Listings app that displays the appropriate image and wiki information when a row is selected in the grid:

- 1. Edit CarMasterDetail.js.
- 2. Create a selection model for the grid:

```
var sm = this.grid.getSelectionModel();
```

The default selection model for a grid is a <u>RowSelectionModel</u>. Whenever a row in the grid is selected, a <u>rowselect</u> event is fired. This event includes the SelectionModel, rowIndex and the Record that provides the data for the selected row.

Add an event handler to call a custom onRowSelect function when a row in the grid is selected:

```
sm.on('rowselect', this.onGridRowSelect, this);
```

4. Implement onRowSelect to update the the row with the data from the data store:

```
onGridRowSelect: function(sm, rowIdx, r) {
    this.detail.update(r.data);
}
```

For more information about working with Ext JS grids, see the API documentation.