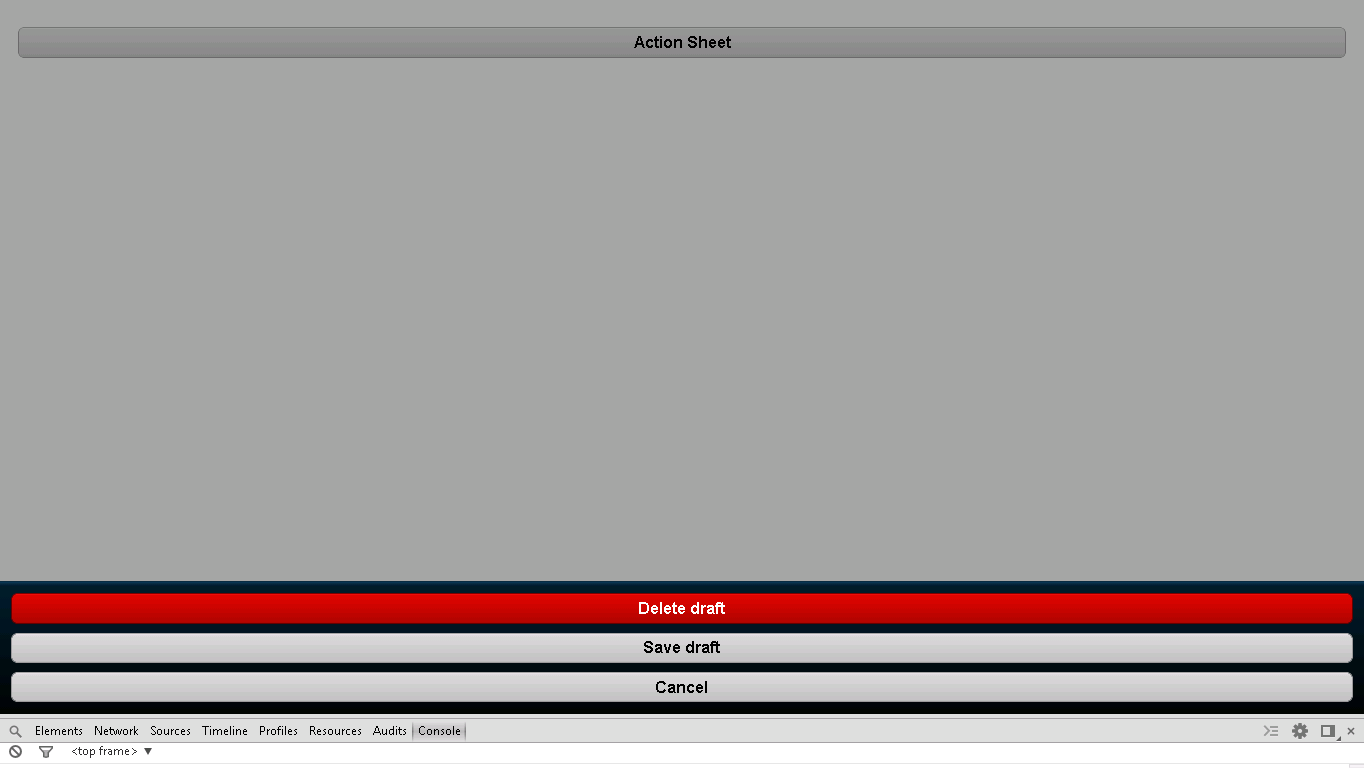
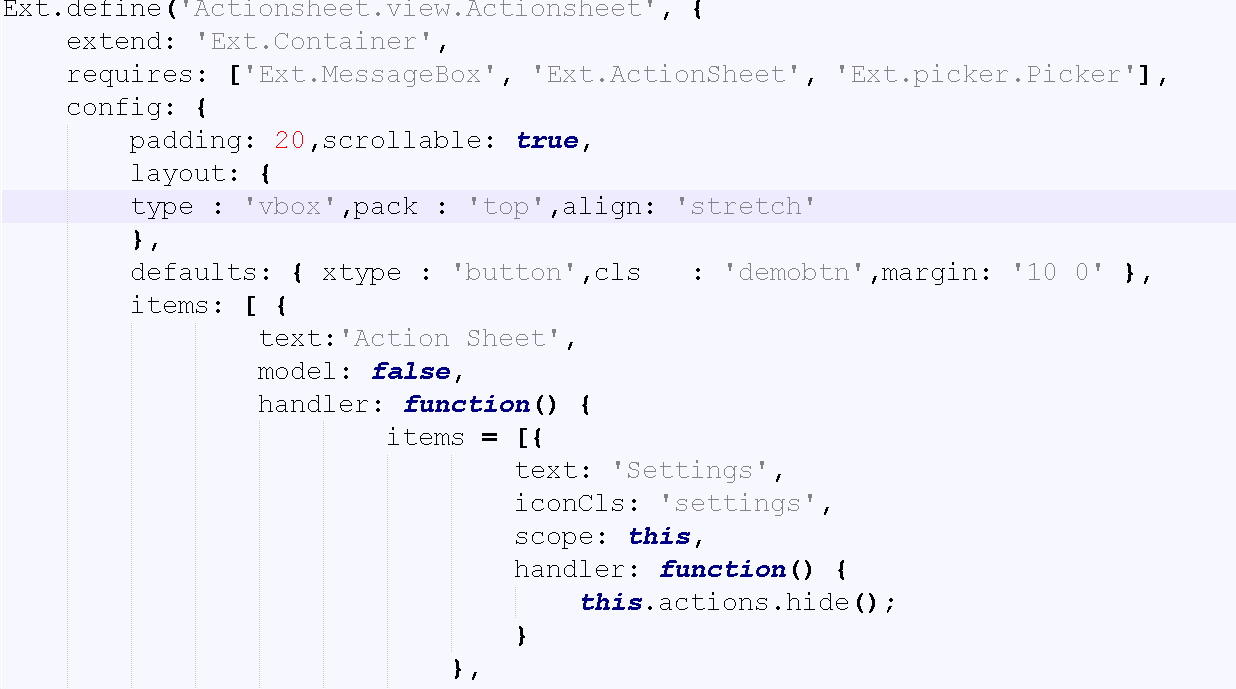
**Action Sheet**

Following snapshot shows Action sheet to delete and save draft.



**Snapshot 1.1**

**Code**

****

**Snapshot 1.2**

* Here, Handler function is used in which items are added,

**handler: function() {**

**items = [{** :-> Here, iconCls is added default for setting which relates to delete item in next list of Items[], within it again handler function is added to hide this particular item.

**text: 'Settings',**

**iconCls: 'settings',**

**scope: this,**

**handler: function() {**

**this.actions.hide();**

**}**

**},**

****

**Snapshot 1.3**

**{**

**text: 'New Item',** :-> Here, iconCls is added default for compose which relates to save item in next list of Items [], within it again handler function is added to hide this particular item.

**iconCls: 'compose',**

**scope: this,**

**handler: function() {**

**this.actions.hide();**

**}**

**},**

**{**

**xtype: 'button',** :-> This is simple button is created for cancle function.

**text: 'Star',** :-> Here, iconCls is added default for star which relates to cancle item in next list of Items [], within it again handler function is added to hide this particular item.

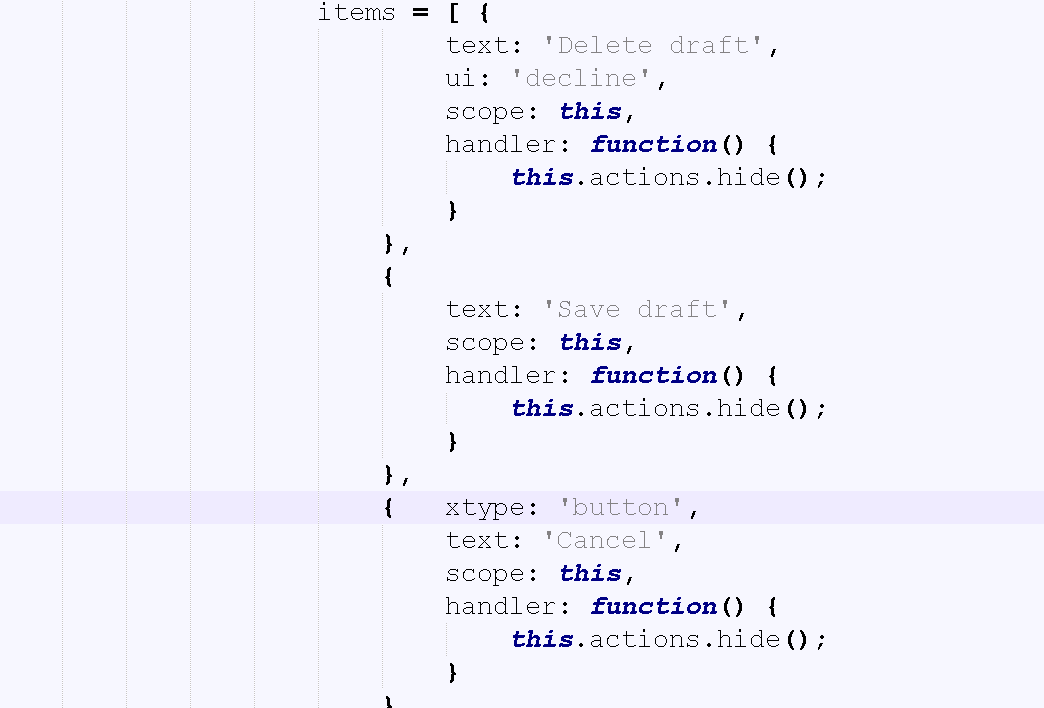
**iconCls: 'star',**

**scope: this,**

**handler: function() {**

**this.actions.hide();**

**}**

****

**Snapshot 1.4**

**items = [** :-> These items are just ui presentation of above items Delete drafts, save draft, cancel to setting, star, and cancel button respectively.

**{ text: 'Delete draft',**

**ui: 'decline',**

**scope: this,**

**handler: function() {**

**this.actions.hide();**

**}},**

**{**

**text: 'Save draft',**

**scope: this,**

**handler: function() {**

**this.actions.hide();**

**}},**

**{ xtype: 'button',**

**text: 'Cancel',**

**scope: this,**

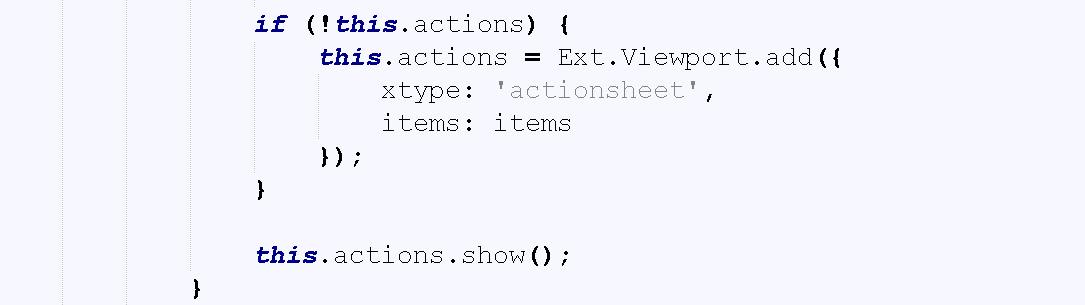
**handler: function() {**

**this.actions.hide();**

**}**

**}**

**];**

****

**Snapshot 1.5**

**if (!this.actions)** :-> This function is called at every time when action sheet is button is pressed, to add items.

**{**

**this.actions = Ext.Viewport.add({**

**xtype: 'actionsheet',**

**items: items**

**});**

**}**

**this.actions.show();**

**}**

**FAQ**

1. For what viewport is used?

Ans: Ext.Viewport is a instance created when you use {@link Ext#setup}. Because {@link Ext.Viewport} extends from

{@link Ext.Container}, it has as {@link #layout} (which defaults to {@link Ext.layout.Card}). This means you

can add items to it at any time, from anywhere in your code. The {@link Ext.Viewport} {@link #cfg-fullscreen}

configuration is `true` by default, so it will take up your whole screen.

**Checkpoints**

1. Check carefully with your indentations as if you will miss any of the braces by mistake may reflect in compilation error.
2. See that your .js file starts with Ext.Define block and check that ends at the end of the program.
3. More than one items are separated by comma (,) separator and main program block of define is ended by semi colon (;)
4. Any value is assigned using colon (:) for example ui: 'confirm' and value is written inside single coats.
5. Items block starts with square brackets [] , which contains multiple members and also sub items.
6. Don’t forget to add entries of your files in app.js.

**Source Code**

****