YUI Library: Dialog & SimpleDialog

2007-12-4 **v2.4**

Simple Use Case: YAHOO.widget.Dialog

Markup (optional, using HTML form in standard module format):

```
 <div id="myDialog">
   <div class="bd">
     <form name="dlgForm" method="POST" action="
    post.php">
     <label for="firstname">First Name:</label>
    <input type="text" name="firstname" />
   </form></div>
</div>
```

Script:

```
//create the dialog:
var myDialog = new YAHOO.widget.Dialog("myDialog");
//set dialog to use form post on submit action:
myDialog.cfg.queueProperty("postmethod", "form");
//set up button handler:
var handleSubmit = function() {
   this.submit(); }; //default submit action
//set up button, link to handler
var myButtons = [ { text:"Submit",
   handler:handleSubmit, isDefault:true } ]);
//put buttons in configuration queue for processing
myDialog.cfg.queueProperty("buttons", myButtons);
mDialog.render(); //render dialog to page
myDialog.show(); //make dialog visible
```

Creates, renders and shows a panel using existing markup and all default Dialog settings.

Constructor: YAHOO.widget.Dialog & SimpleDialog

`YAHOO.widget.Dialog(str elId[, obj config]);`

Arguments:

(1) **Element ID:** HTML ID of the element being used to create the Dialog or SimpleDialog. If this element doesn't exist, it will be created.

(2) **Configuration Object:** JS object defining configuration properties for the Dialog. See Configuration section for full list.

The `postmethod` Property: Dialog & SimpleDialog

postmethod:	Characteristics:
`"none"`	Button handlers do all form processing.
`"form"`	Button handlers called, then form posted to url designated in form's target attribute.
`"async"`	Button handlers called, then form sent to url designated in form's target attribute using asynchronous XMLHttpRequest (via Connection Manager).

Key Interesting Moments in Dialog & SimpleDialog

See online docs for a complete list of Custom Events associated with Container controls.

Event	Arguments
beforeSubmitEvent	None.
cancelEvent	None.
submitEvent	None.

All events above are YUI Custom Events (see Event Utility docs); subscribe to these events using their subscribe method: `myDlg.hideEvent.subscribe(fnMyHandler);`.

Dialog/SimpleDialog Configuration Options

See online docs for complete list of Container options; see Simple Use Case (top left) for config. syntax.

Option (type)	Default	Description
text	null	Sets body text of SimpleDialog *(SimpleDialog only)*.
icon	"none"	Sets url for graphical icon. Six icons are provided: ICON_BLOCK, ICON_WARN, ICON_HELP, ICON_INFO, ICON_ALARM, and ICON_TIP. *(SimpleDialog only.)*
postmethod (s)	varies	Designates handling of form data; see box at bottom left. Default is "none" for SimpleDialog and "async" for Dialog.
buttons (a)	null	Array of button objects. Button objects contain three members: `text` label for button, `handler` function to process button click, and `isDefault` boolean specifying whether this is the default action on form submit.

See cheat sheet for Panel for additional configuration options; see online documentation for full list.

Solutions

Use `validate()` to **check form data** prior to submitting:

```
fnCheckEmail = function() {
  if (myDialog.getData().email.indexOf("@") > -1)
    {return true;} else {return false;} };
myDialog.validate = fnCheckEmail;
```

Set "success" handler for asynchronous post:

```
fnSuccess = function(o) {//function body};
myDialog.callback.success = fnSuccess;
```

Dependencies

Dialog requires the full Container package, the Yahoo Object, Dom Collection, and Event Utility. Animation, Button, Connection Manager and Drag And Drop are optional (though required for specific features).

YAHOO.widget.Dialog & SimpleDialog: Key Properties

body (el)
form (el)
callback (o) Connection Manager callback object for async transactions.
element (el) containing header, body & footer
footer (el)
header (el)
id (s) of the element

YAHOO.widget.Dialog & SimpleDialog: Methods

appendToBody(el *element*)
appendToFooter(el *element*)
appendToHeader(el *element*)
cancel() Executes cancel then hide().
getData() Returns obect of name/value pairs representing form data.
hide()
render([el *element*]) Argument required for Dialogs not built from existing markup. Dialog will not be in the DOM or visible until render is called.
setBody(str or el *content*)
setFooter(str or el *content*)
setHeader(str or el *content*)
submit() Executes submit followed by hide().
show()
getButtons()

YUI Library: Panel

2007-12-4 **v2.4**

Simple Use Case: YAHOO.widget.Panel

Markup (optional, using standard module format):

```
<div id="myPanel">
  <div class="hd">Header content.</div>
  <div class="bd">Body content.</div>
  <div class="ft">Footer content.</div>
</div>
```

Script:

```
var oPanel = new YAHOO.widget.Panel("myPanel");
oPanel.render();
oPanel.show();
```

Creates, renders and shows a panel using existing markup and all default Panel settings.

Constructor: YAHOO.widget.Panel

```
YAHOO.widget.Panel(str elId[, obj config]);
```

Arguments:

(1) **Element ID:** HTML ID of the element being used to create the Panel. If this element doesn't exist, it will be created.

(2) **Configuration Object:** JS object defining configuration properties for the panel. See Configuration section for full list.

Solutions

There are three ways to **configure options on your Panel**:

```
// 1. In the constructor, via an object literal:
var myPanel = new YAHOO.widget.Panel("myPanel", {
   visible:false });
// 2. Via "queueProperty", prior to rendering:
myPanel.cfg.queueProperty("visible",false);
// 3. Via "setProperty" after rendering:
myPanel.cfg.setProperty("visible",false);
```

Align the top left corner of your Panel with the bottom right corner of an element whose HTML ID is "contextEl":

```
myPanel.cfg.setProperty("context", ["contextEl",
   "tl", "br"]);
```

Subscribe to a Panel Custom Event, listening for changes to the Panel's postion, alerting its new position after move:

```
alertMove = function(type, args) {
  alert(args[0] + ", " + args[1]);
}
myPanel.subscribe("move", alertMove);
```

Key Interesting Moments in Panel

See online docs for a complete list of Panel's Custom Events.

Event	Arguments
beforeRenderEvent	None.
renderEvent	None.
beforeShowEvent	None.
showEvent	None.
beforeHideEvent	None.
hideEvent	None.
beforeMoveEvent	X, Y to which the Panel will be moved.
moveEvent	X, Y to which the Panel was moved.
hideMaskEvent	None.
showMaskEvent	None.
changeContentEvent	None.
changeBodyEvent	String or element representing new body content (**Note:** there are corresponding Header and Footer change events, too).

All Panel events are YUI Custom Events (see Event Utility docs); subscribe to these events using their subscribe method: `myPanel.hideEvent.subscribe(fnMyHandler);`.

Key Panel Configuration Options

See online docs for complete list of Panel options; see Solutions (bottom left) for how to set your options.

Option (type)	Default	Description
close (b)	null	Display close icon.
draggable (b)	null	Make the Panel draggable.
modal (b)	null	Use a modal mask behind Panel when Panel is visible.
visible (b)	true	Sets the "display" style property to "block" (true) or "none" (false).
x, y, and xy (int, int, ar)	null	These properties can be used to set the Panel's "top" and/or "left" styles.
context (ar)	null	Anchors Panel to a context element; format: `[el contextEl, s panelCorner, s contextCorner]` with corners defined as "tr" for "top right" and so on.
fixedcenter (b)	false	Automatically center Panel in viewport?
width (s)	null	Sets "width" style property.
height (s)	null	Sets "height" style property.
zindex (int)	null	Sets "z-index" style property.
constrainto viewport (b)	false	When true, prevents the Panel from being dragged out of the viewport.
underlay (s)	"shadow"	Type of underlay: "shadow", "none", or "matte".
effect (obj)	null	Object defining effect (FADE or SLIDE) to use in showing and hiding Panel: `{effect: YAHOO.widget.ContainerEffect.FADE, duration:1}`

YAHOO.widget.Panel: Properties

body (el)
element (el) containing header, body & footer
footer (el)
header (el)
id (s) of the element

YAHOO.widget.Panel: Methods

appendToBody(el *element*)
appendToFooter(el *element*)
appendToHeader(el *element*)
hide()
render([el *element*]) Argument required for Panels not built from existing markup. Panel will not be in the DOM or visible until render is called
setBody(str or el *content*)
setFooter(str or el *content*)
setHeader(str or el *content*)
show()
bringToTop()

Dependencies

Panel requires the full Container package, the YAHOO object, Event, and Dom. Animation, and Drag and Drop are optional. **Note:** Panels use Drag and Drop by default — a simple Panel with default configuration options will not be draggable without it.