

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:

- 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.
- 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 (<i>SimpleDialog only</i>).
icon	"none"	Sets url for graphical icon. Six icons are provided: ICON_BLOCK, ICON_WARN, ICON_HELP, ICON_INFO, ICON_ALARM, and ICON_TIP. (<i>SimpleDialog only</i> .)
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: <code>text</code> label for button, <code>handler</code> function to process button click, and <code>isDefault</code> 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 object 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()**

