Dojo Widgets (Dijit)

Topics

- What are Widgets, What is Dijit?
- How to use widgets
  - > Declaratively or programmatically
- Form widgets
  - > Input form validation
- Layout widgets
- Command control widgets
- User assistance and feedback widgets
- Advanced editing and display widgets
- Themes and Design

What is a Widget?
What is Dijit?

What is a Widget?

- Is a UI element such as a button, text box, scroll bar, calendar, tree etc
  - Widgets are a lot easier to deal with than DOM APIs
- Can be a composite element
- Can have custom style
- Event handlers can be registered
  - Event handlers might call backend service
- Can handle browser incompatibilities

What is Dijit?

- Dojo Widget Library
  - A widget system layered on top of dojo.
- Fully theme'able
  - Come with three default themes
  - You can override the theme by container or by element to add nuance and flair.
- Fully internationalized
- Fully accessible

Widgets in Dijit

- Form, Validation, Specialized Input
  - > CheckBox, ComboBox, DateTextBox, InlineEditBox, Slider, Textarea, TextBox, ValidationTextBox, etc
- Layout
  - > AccordionContainer, Border Container, ContentPane, etc.
- Command control
  - > Button, DropDownButton, ComboButton, Menu, Toolbar
- User assistance and feedback
  - > Dialog, TooltipDialog, PregressBar, TitlePane, Tooltip
- Advanced editing and display
  - > Editor, Grid, Tree

How to Use Widgets

How to Use Widgets

- Declaratively or Programmatically
  - Declaratively by using special attributes inside of regular HTML tags (markup)
  - Programmatically through JavaScript.
- You have the same options either way

Using Widgets Declaratively (Markup)

- Be sure to `dojo.require` your widget.
- Be sure to include the Dojo parser!
- Widgets are created as soon as the DOM is created (if possible) before any other `addOnLoad` callbacks.
- Create your markup, and add the following attribute to the main node:

```
<div dojoType="path.to.widget"></div>
```

...where "*path.to.widget*" is the fully qualified name of the widget constructor

Example: Creating a Progress Bar Declaratively

```
<head>
  <script src=”dojotoolkit/dojo/dojo.js”
    djConfig=”parseOnLoad: true”>
  </script>
  <script>
    dojo.require(“dojo.parser”);
    dojo.require(“dijit.ProgressBar”);
  </script>
</head>
<body>
  <div style="width: 400px;" maximum="200" id="setTestBar"
    progress="20" dojoType="dijit.ProgressBar">
  </div>
</body>
```

Creating Widgets Programatically

- DOM should be available before creating widgets programmatically.
  - Unless you pass it a new parentless node (or no node)
  - **`dojo.addOnLoad`**
- Call the widget constructor directly:

```
var myWidget = new dijit.form.Button({
    // properties
    iconClass:"someCSSclass"
}, "myButtonNode");
```

Constructor Parameters

- All widget constructors take two arguments:
  - A properties object
  - A node reference
- Node reference is optional, but is considered good practice.
  - A `div` will be automatically created.
- Properties object is a set of named arguments you can use to set various properties on a widget.

Example: Creating a Progress Bar Programmatically

```
<head>
  <script src=”dojotoolkit/dojo/dojo.js”></script>
  <script>
    dojo.require(“dijit.ProgressBar”);

    dojo.addOnLoad(function(){
      var instance = new dijit.ProgressBar({
        maximum: 200,
        progress: 20
      }, “setTestBar”);
    });
  </script>
</head>
<body>
  <div style="width: 400px;" id="setTestBar”></div>
</body>
```

Dijit Form Widgets

Form Widgets – Common Characteristics

- All widgets beginning with “dijit.form”
- The form widgets can be used in a FORM tag, in a dijit.form.Form widget, or outside of a form.

Form Widgets – Common Characteristics

- Common Attributes
  - disabled, intermediateChange, tabIndex
- Common Methods
  - focus – set the focus on this widget
  - getValue – get the value of the widget
  - setValue – set the value of the widget
  - reset – resets the widget to its initial value
  - undo – restore the value to the last value passed to onChange
  - setAttribute - Controls all sorts of attributes for widgets like disabled, readonly, etc.
- Common Extension Points
  - onChange – callback when value is changed

Form Widget – convenience methods

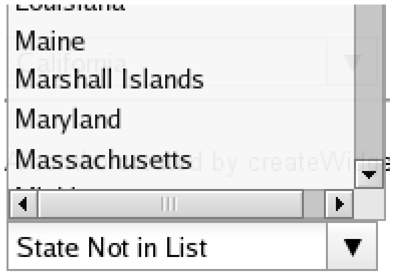
- Convenience methods added to regular HTML form
  - *getValues* - generate JSON structure from form values get widget values
  - *setValues* - fill in form values from a JSON structure generate map from name --> [list of widgets with that name]
  - *isValid* - Return true if every widget's isValid method returns true.
  - *submit* - programmatically submit form

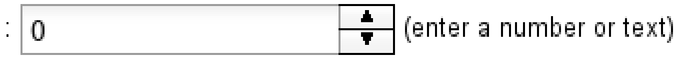
Form Widget – Extension Points

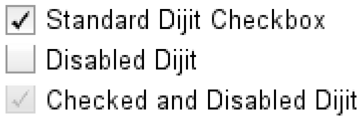
- Extension points
  - execute - User defined function to do stuff when the user hits the submit button

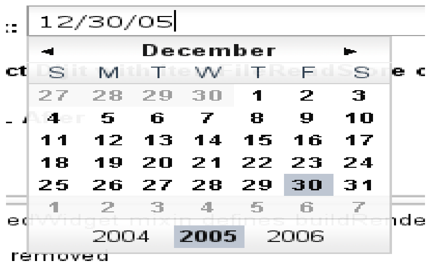
Form Widgets

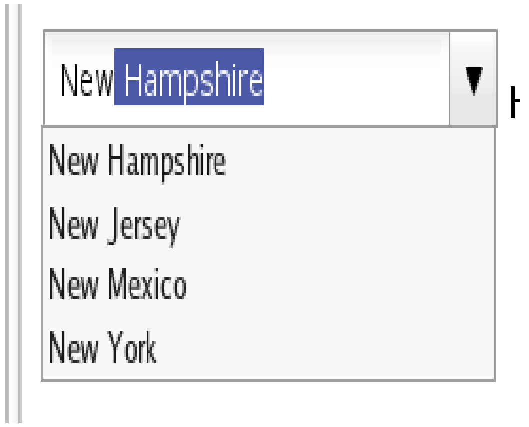
- CheckBox, RadioButton, ComboBox, DateTextBox, FilterSelect, NumberSpinner, NumberTextBox, Slider, TextArea, TextBox, etc

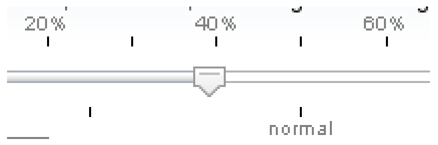






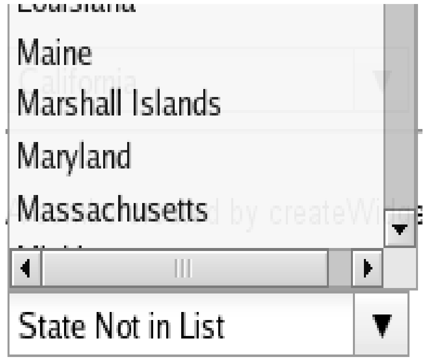






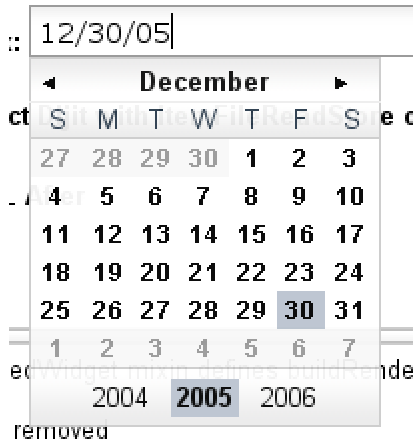
dijit.form.ComboBox Example

```
<head>
  ...
  <script type="text/javascript">
    dojo.require("dijit.form.ComboBox");
    function setVal1(value) {
      alert("Selected "+value);
    }
  </script>
</head>

<body class="tundra">
  <select name="state1"
      dojoType="dijit.form.ComboBox"
      autocomplete="true"
      value="California"
      onChange="setVal1">
      <option selected="selected">California</option>
      <option>Connecticut</option>
      <option >Illinois</option>
      <option >New York</option>
  </select>
</body>
```



dijit.form.DateTextBox Example

```
<head>
    ...
    <script type="text/javascript">
        dojo.require("dijit.form.DateTextBox");
    </script>
</head>

<body class="tundra">
    <h1>DateTextBox</h1>
    <input type="text" name="date1" value="2009-09-30"
           dojoType="dijit.form.DateTextBox"
           required="true" />
</body>
```



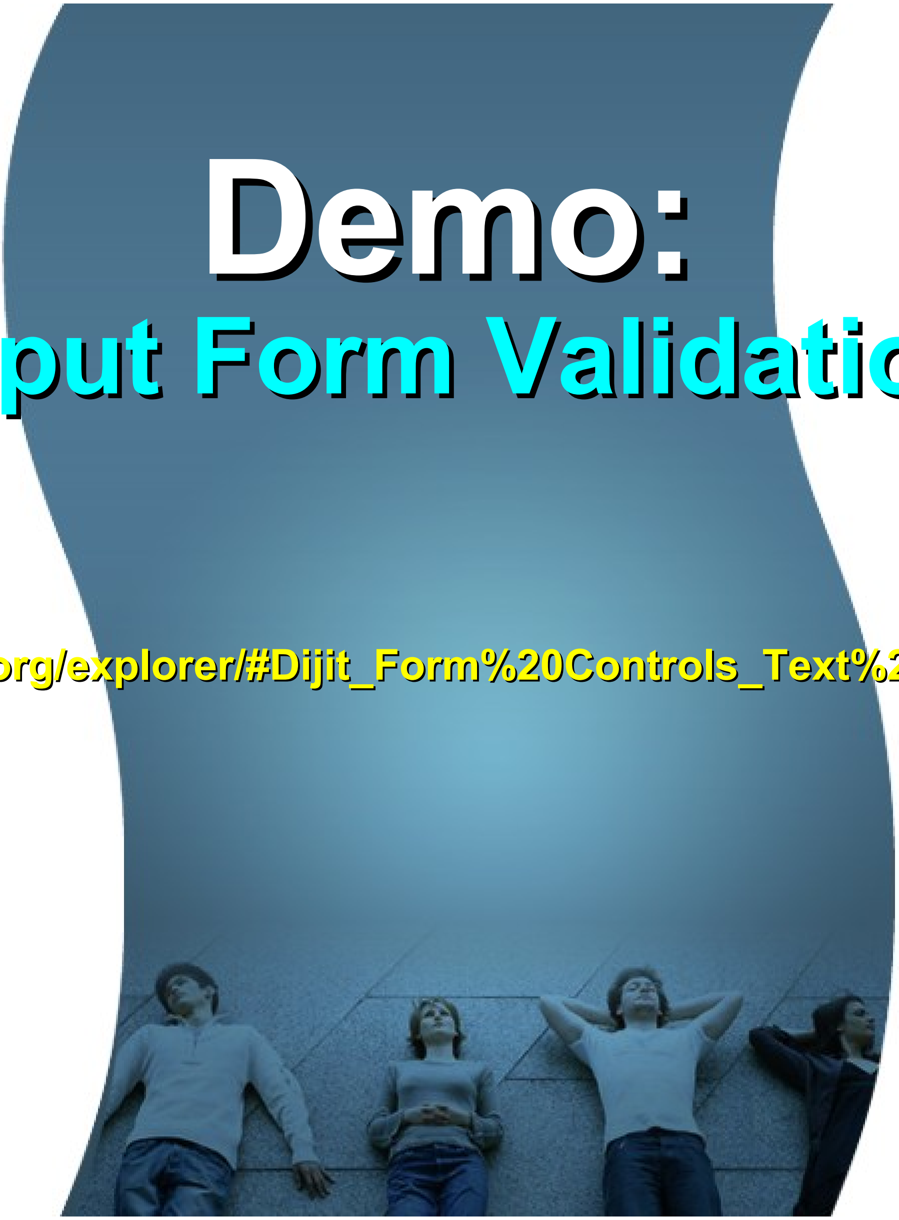
Validation

Example: Validation

```
dojo.require("dijit.form.ValidationTextBox");

<input type="text" name="phone"
       id="phone"
       value="someTestString"
       dojoType="dijit.form.ValidationTextBox"
       regExp="[\w]+"
       required="true"
       invalidMessage="Invalid Non-Space Text.">
```

Demo:
Input Form Validation

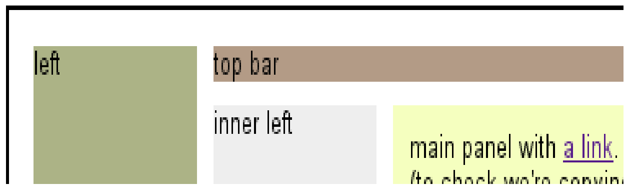
http://dojocampus.org/explorer/#Dijit_Form%20Controls_Text%20Boxes_Validation

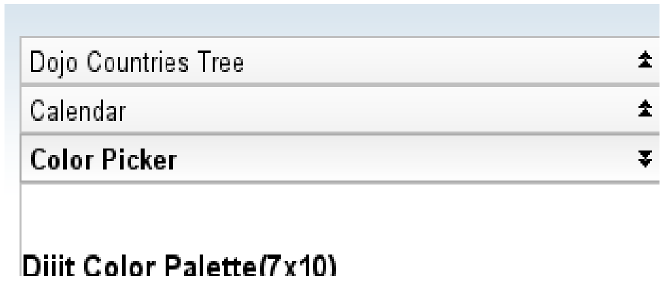


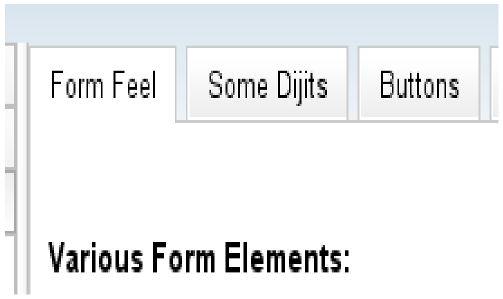
Dijit Layout Widgets

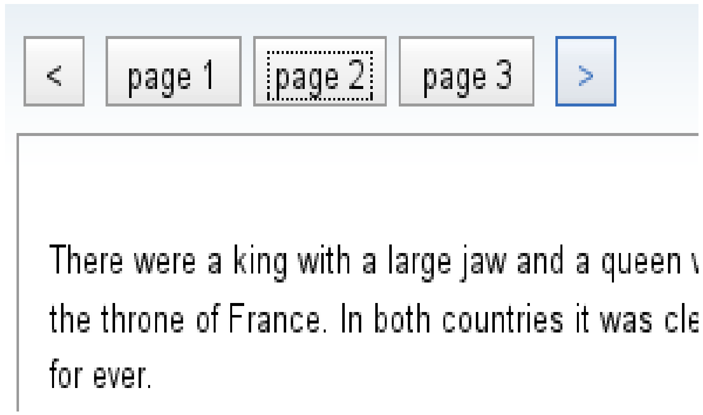
Dijit Layout Widgets

- Accordion Container, Content Pane, Layout Container, Split Container, Stack Container, Tab Container









There were a king with a large jaw and a queen
the throne of France. In both countries it was cle
for ever.

dijit.layout.SplitContainer Example

```
<script type="text/javascript">
  dojo.require("dijit.layout.SplitContainer");
  dojo.require("dijit.layout.ContentPane");
</script>

<div dojoType="dijit.layout.SplitContainer"
  orientation="horizontal"
  sizerWidth="7"
  activeSizing="false"
  style="border: 1px solid #bfbfbf; float: left; width: 400px; height: 300px;">
  <div dojoType="dijit.layout.ContentPane" sizeMin="20" sizeShare="20">
    this box has two horizontal panes
  </div>
  <div dojoType="dijit.layout.ContentPane" sizeMin="50" sizeShare="50">
    without active resizing, a smaller sizer, different starting sizes and minimum sizes
  </div>
</div>
```

Demo:

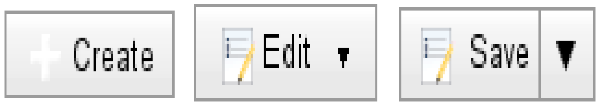
Layout Widgets

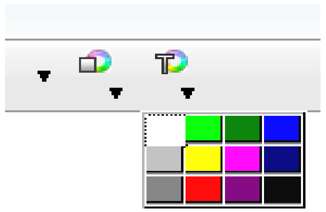
http://dojocampus.org/explorer/#Dijit_Layout

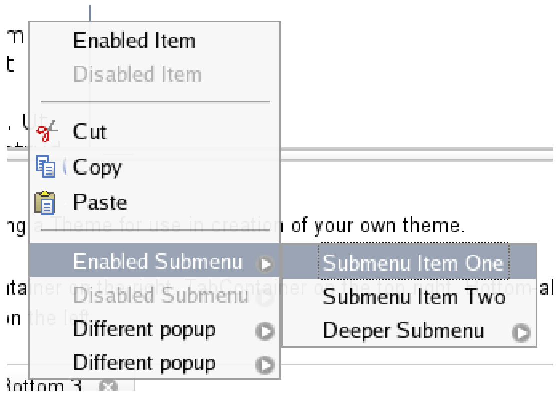
Dijit Command Control Widgets

Dijit Command Control Widgets

- Button, ComboButton, DropDownButton
- Menu, Toolbar







dijit.Menu Example

```
<script language="JavaScript" type="text/javascript">
  dojo.require("dijit.Menu");
  dojo.require("dojo.parser");  // scan page for widgets and instantiate them
  function doNothing() {
    alert('not actually doing anything, just a test!');
  }
</script>

<div dojoType="dijit.Menu" id="popup1campus" contextMenuForWindow="false" style="display: none;"
targetNodeIds="btn1">
  <div dojoType="dijit.MenuItem" onClick="alert('Hello world');">Enabled Item</div>
  <div dojoType="dijit.MenuItem" disabled="true">Disabled Item</div>
  <div dojoType="dijit.MenuSeparator"></div>
  <div dojoType="dijit.MenuItem" iconClass="dijitEditorIcon dijitEditorIconCut"
    onClick="doNothing();">Cut</div>
  <div dojoType="dijit.MenuItem" iconClass="dijitEditorIcon dijitEditorIconCopy"
    onClick="doNothing();">Copy</div>
  <div dojoType="dijit.MenuItem" iconClass="dijitEditorIcon dijitEditorIconPaste"
    onClick="doNothing();">Paste</div>
</div>

<div id="btn1">Right click Me To Show Menu</div>
```

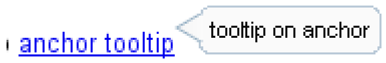
Demo:
Menu Widgets

http://dojocampus.org/explorer/#Dijit_Menu

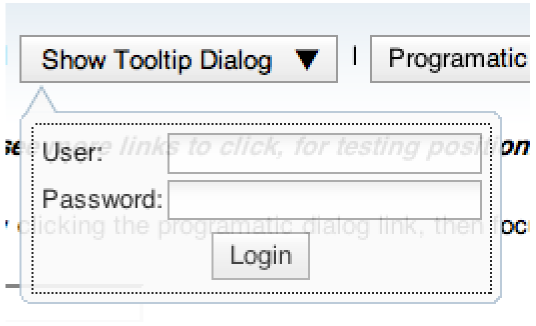
Dijit User Assistance & Feedback Widgets

Dijit User Assistance & Feedback Widgets

- Dialog, TooltipDialog, ProgressBar, TitlePane, Tooltip


anchor tooltip

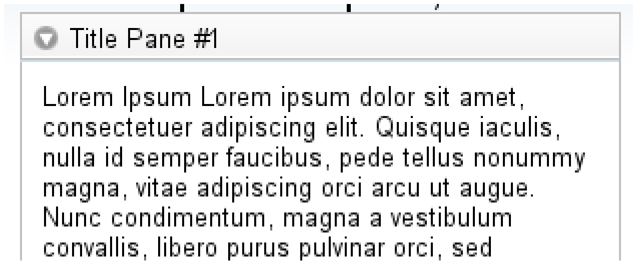
tooltip on anchor


Show Tooltip Dialog ▼ | Programatic

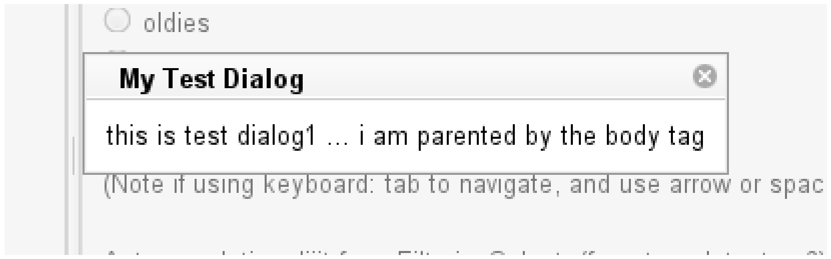
User:

Password:

Login


Title Pane #1

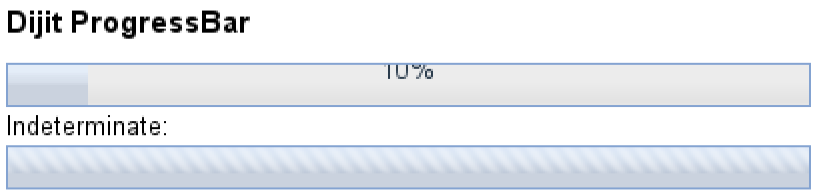
Lorem Ipsum Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Quisque iaculis, nulla id semper faucibus, pede tellus nonummy magna, vitae adipiscing orci arcu ut augue. Nunc condimentum, magna a vestibulum convallis, libero purus pulvinar orci, sed


oldies

My Test Dialog

this is test dialog1 ... i am parented by the body tag

(Note if using keyboard: tab to navigate, and use arrow or spac


Dijit ProgressBar

10%

Indeterminate:

dijit.Dialog Example

```
<body>
  ...
   <button id="buttonOne" dojoType="dijit.form.Button" type="button">
     Click me to display a dialog!
     <script type="dojo/method" event="onClick" args="evt">
        // Show the Dialog:
        dijit.byId("dialogOne").show();
     </script>
  </button>

  <div id="dialogOne" dojoType="dijit.Dialog" title="My Dialog Title">
     <div dojoType="dijit.layout.TabContainer" style="width: 200px; height: 300px;">
        <div dojoType="dijit.layout.ContentPane" title="foo">
           Content of Tab "foo"
        </div>
        <div dojoType="dijit.layout.ContentPane" title="bar">
           Hi, I'm Tab "bar"
        </div>
     </div>
  </div>
</body>
```

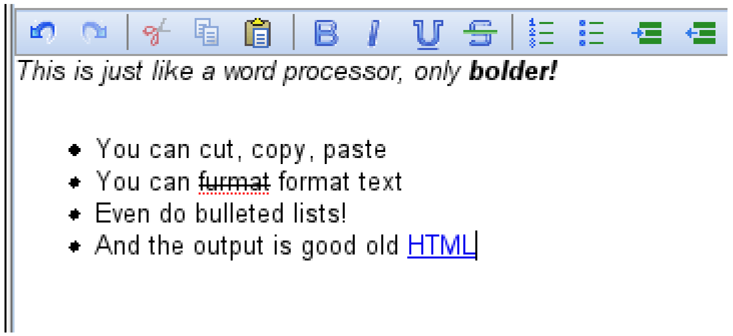
Demo:
Dialog Widgets

http://dojocampus.org/explorer/#Dijit_Dialog

Dijit Advanced Editing & Display

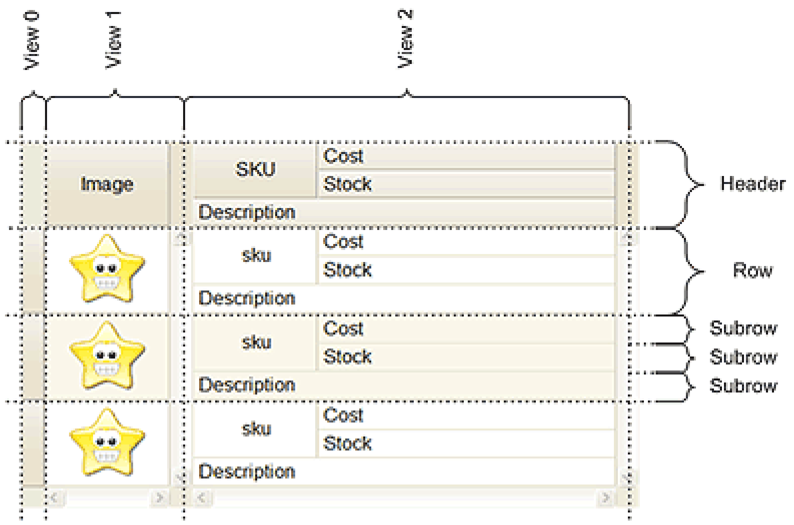
Dijit Advanced Editing & Display Widgets

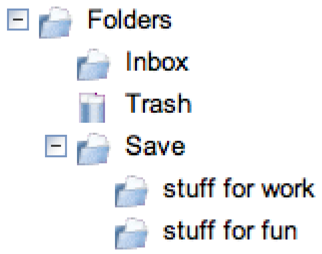
- Editor, Grid, Tree, ColorPalette, InlineEditBox

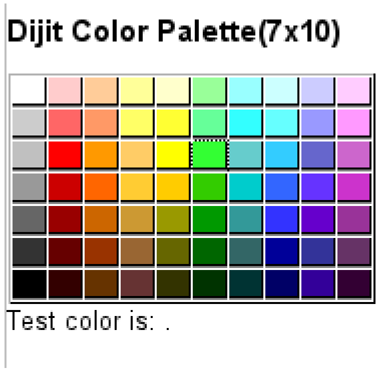


This is just like a word processor, only ***bolder!***

- You can cut, copy, paste
- You can furmat format text
- Even do bulleted lists!
- And the output is good old HTML







dijit.Tree Example (Code)

```
<script language="JavaScript" type="text/javascript">
  dojo.require("dojo.data.ItemFileReadStore");
  dojo.require("dijit.Tree");
  dojo.require("dojo.parser");  // scan page for widgets and instantiate them
</script>

<div dojoType="dojo.data.ItemFileReadStore" jsId="continentStore"
  url="featureexplorer/Dojo/countries.json"></div>

<h3>Tree with hardcoded root node (not corresponding to any item in the store)</h3>
<div dojoType="dijit.Tree" id="tree1" store="continentStore"
   query="{type:'continent'}" labelAttr="name" label="Continents">
  <script type="dojo/method" event="onClick" args="item">
    if(item){
      alert("Execute of node " + continentStore.getLabel(item)
        +", population=" + continentStore.getValue(item, "population"));
    }else{
      alert("Execute on root node");
    }
  </script>
</div>
Click <a href="featureexplorer/Dojo/countries.json" target="_new">here</a> to see the data used to
populate the tree.
```

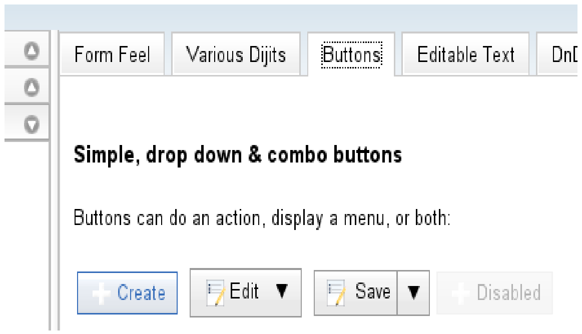
dijit.Tree Example (JSON)

```
{ identifier: 'name',
  label: 'name',
  items: [
     { name:'Africa', type:'continent',
         children:[{_reference:'Egypt'}, {_reference:'Kenya'}, {_reference:'Sudan'}] },
     { name:'Egypt', type:'country' },
     { name:'Kenya', type:'country',
         children:[{_reference:'Nairobi'}, {_reference:'Mombasa'}] },
     { name:'Nairobi', type:'city' },
     { name:'Mombasa', type:'city' },
     { name:'Sudan', type:'country',
         children:{_reference:'Khartoum'} },
     { name:'Khartoum', type:'city' },
     { name:'Asia', type:'continent',
         children:[{_reference:'China'}, {_reference:'India'}, {_reference:'Russia'},
                   {_reference:'Mongolia'}] },
     { name:'China', type:'country' },
     { name:'India', type:'country' },
     { name:'Russia', type:'country' },
     { name:'Mongolia', type:'country' },
     { name:'Australia', type:'continent', population:'21 million',
         children:{_reference:'Commonwealth of Australia'}},
     { name:'Commonwealth of Australia', type:'country', population:'21 million'},
     …
]}
```

Themes and Design

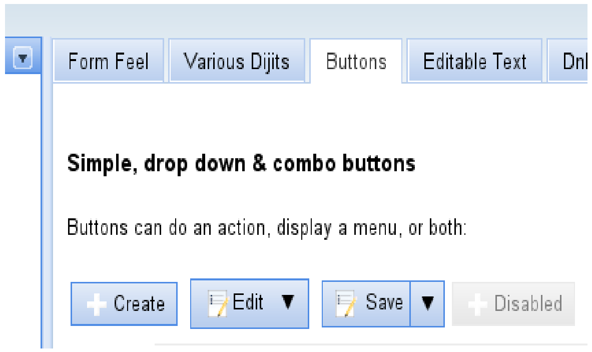
Themes

- Dijit Themes lend a consistent look and feel to widgets
- Built-in themes in Dijit
  > Tundra (Most commonly used)
  > Soria
  > Noir



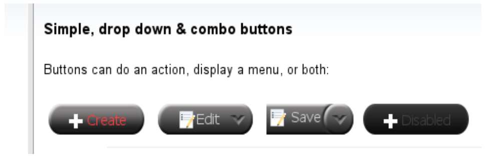
Simple, drop down & combo buttons

Buttons can do an action, display a menu, or both:



Simple, drop down & combo buttons

Buttons can do an action, display a menu, or both:



Simple, drop down & combo buttons

Buttons can do an action, display a menu, or both:

How to Specify a Theme

```
 <head>
    ....
   <style type="text/css">
      @import "/dojo-release-1.5.0/dijit/themes/tundra/tundra.css";
      @import "/dojo-release-1.5.0/dojo/resources/dojo.css";
   </style>
   ...
</head>
<body class="tundra">
   <button dojoType="dijit.form.Button" onclick="create_button">
      Click me to create a new button programatically!!
   </button>
   <div id="button-placeholder"> </div>
</body>
```

Advanced Topics of Dojo Toolkit

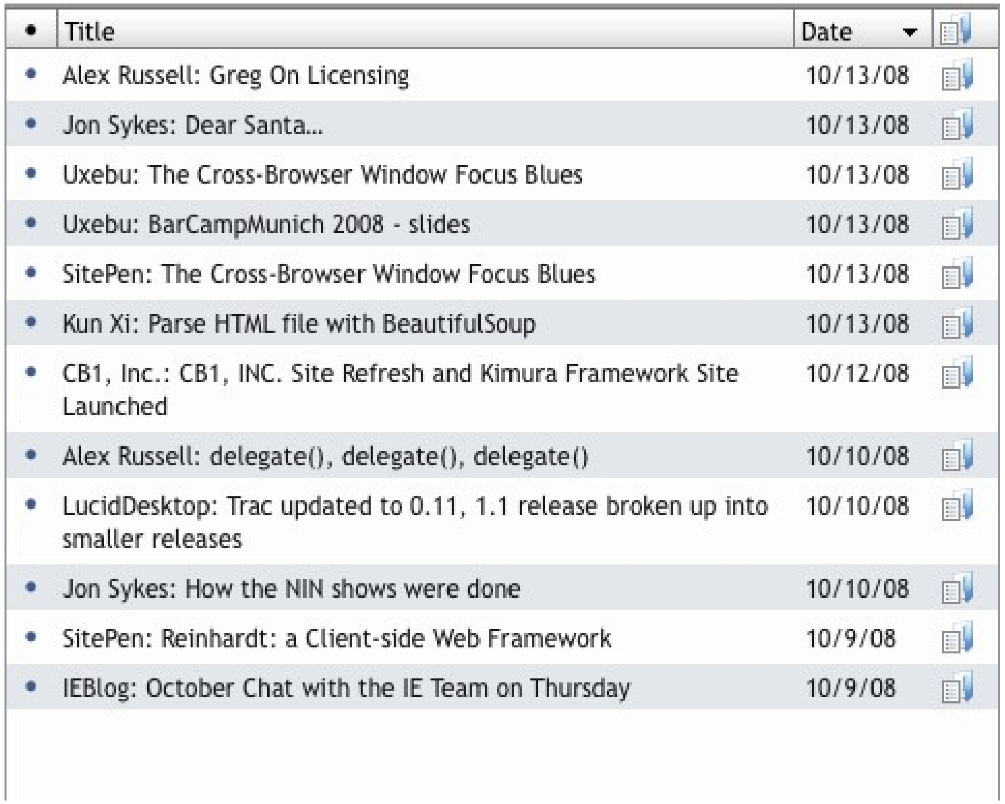
Topics

- dojox.grid
- dojox.charting
- dojox.cometd
- more

dojox.grid

More in DojoX: The Grid

- Has all the features HTML tables are missing
- Often found in:
  - > RSS readers
  - > Mail readers
- Anything geared toward browsing through large amounts of data



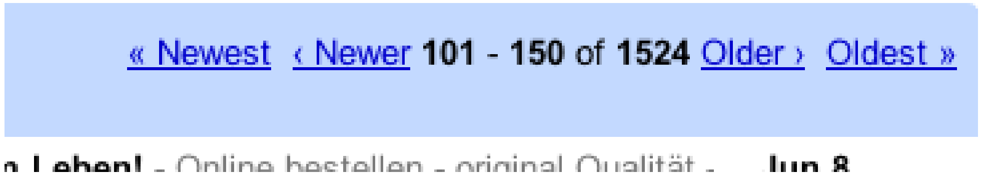
●	Title	Date	
●	Alex Russell: Greg On Licensing	10/13/08	
●	Jon Sykes: Dear Santa...	10/13/08	
●	Uxebu: The Cross-Browser Window Focus Blues	10/13/08	
●	Uxebu: BarCampMunich 2008 - slides	10/13/08	
●	SitePen: The Cross-Browser Window Focus Blues	10/13/08	
●	Kun Xi: Parse HTML file with BeautifulSoup	10/13/08	
●	CB1, Inc.: CB1, INC. Site Refresh and Kimura Framework Site Launched	10/12/08	
●	Alex Russell: delegate(), delegate(), delegate()	10/10/08	
●	LucidDesktop: Trac updated to 0.11, 1.1 release broken up into smaller releases	10/10/08	
●	Jon Sykes: How the NIN shows were done	10/10/08	
●	SitePen: Reinhardt: a Client-side Web Framework	10/9/08	
●	IEBlog: October Chat with the IE Team on Thursday	10/9/08	

What the Grid Is

- Spreadsheet-like “supertable”
- Efficient mechanisms for viewing and editing data
- Designed for large data sets
- Very customizable

The Grid Fixes Pagination

- Working through large number of items is hard.
- Benefit to pagination is that only data for the current data set is loaded.
- The Dojo Grid implements the same functionality as pagination through the position of its scroll bar.



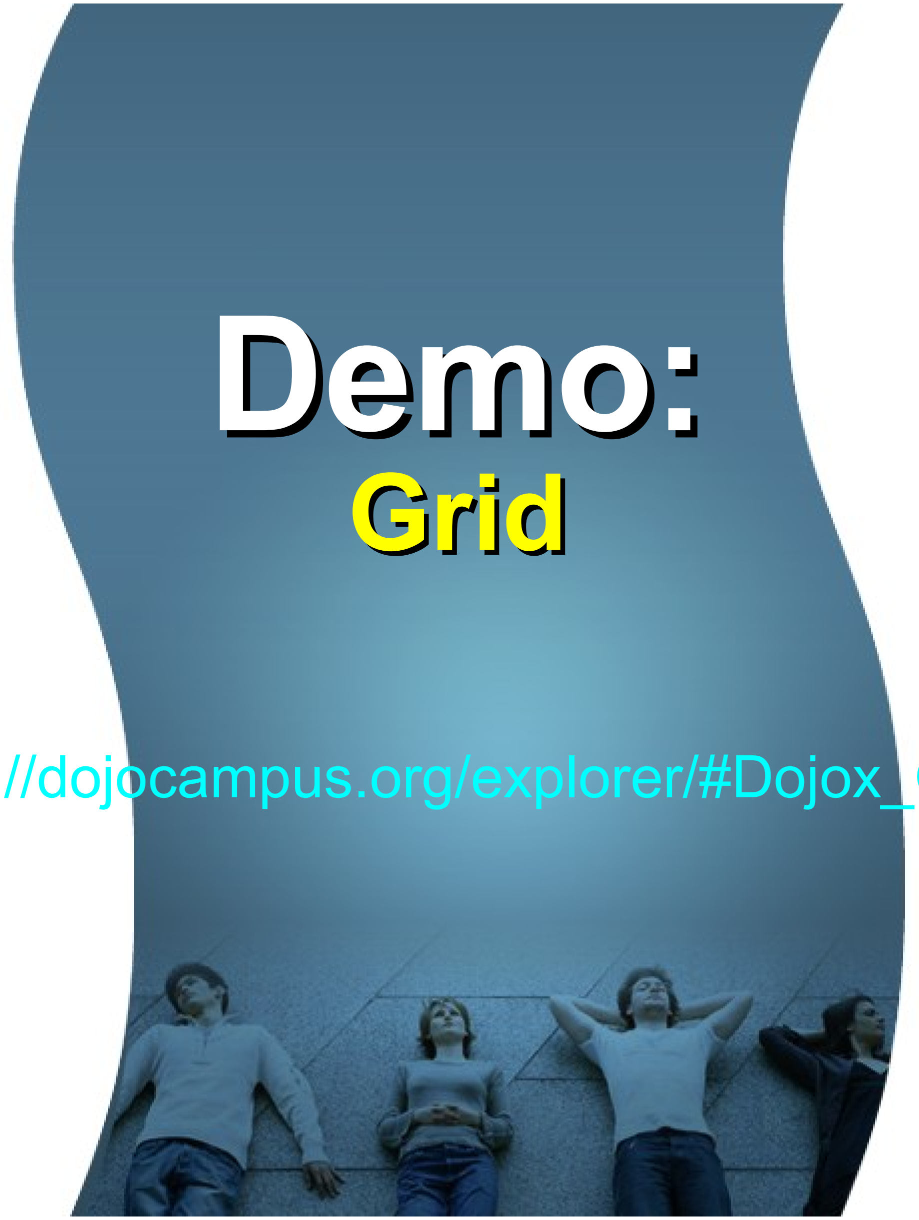
« Newest ‹ Newer **101 - 150 of 1524** Older › Oldest »

Ease of Data Display in the Grid

- Capabilities are in line with the advanced (sometimes underutilized) features of the HTML table.
  - Grouping
  - Spanning columns and rows
- Cell and row dimensions can be tweaked.
- A cell can combine other cell properties and change its appearance.
  - Easy to conditionally style complicated data within cells
    - Sometimes hard to implement in desktop grids

Working With the Grid

- Creation, mutation, and filtering of grid data is simple.
  - Thanks to a Dojo Data back end
- Grid uses various events to alert you of everything it’s doing.



Demo:

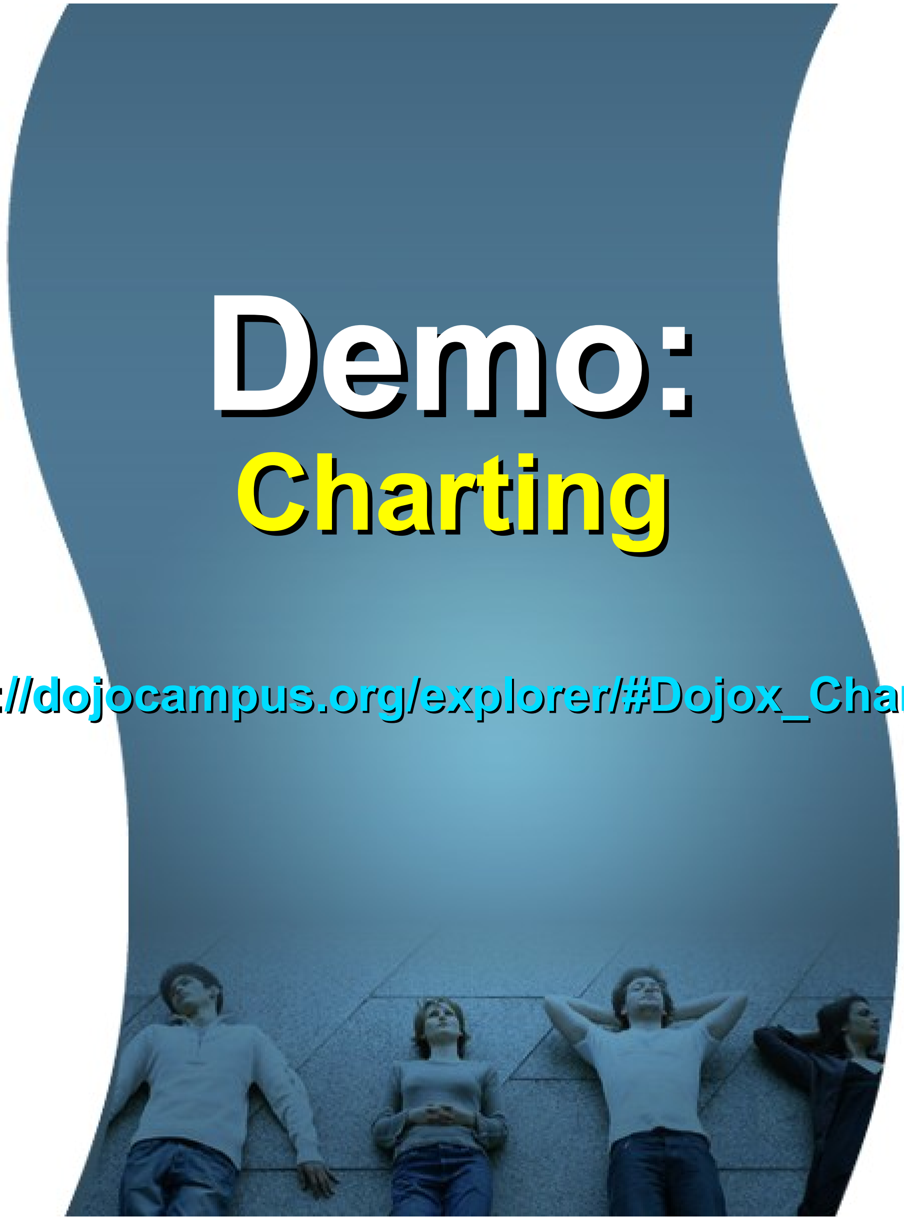
Grid

http://dojocampus.org/explorer/#Dojox_Grid

dojox.charting

More in DojoX: Charting

- Extremely rich charting infrastructure
- Multiple plots
- Axis labels
- Actions
- Highly stylized



Demo:

Charting

http://dojocampus.org/explorer/#Dojox_Charting

dojox.cometd

What is CometD?

- CometD is a scalable HTTP-based event routing bus that uses a Ajax Push technology pattern known as Comet
- CometD is a Dojo Foundation project to provide implementations of the Bayeux protocol in javascript, java, perl, python and other languages.

Demo:
CometD (Chat, Slide)

Steps for Building Comet Application

1. Configure Cometd servlet in the web.xml
2. Initialize (Connect to Cometd server)
3. Subscribe a channel
4. Write callback function
5. Publish to the channel
6. Unsubscribe the channel
7. Disconnect

Step1: Configure Cometd Servlet in your Application's web.xml

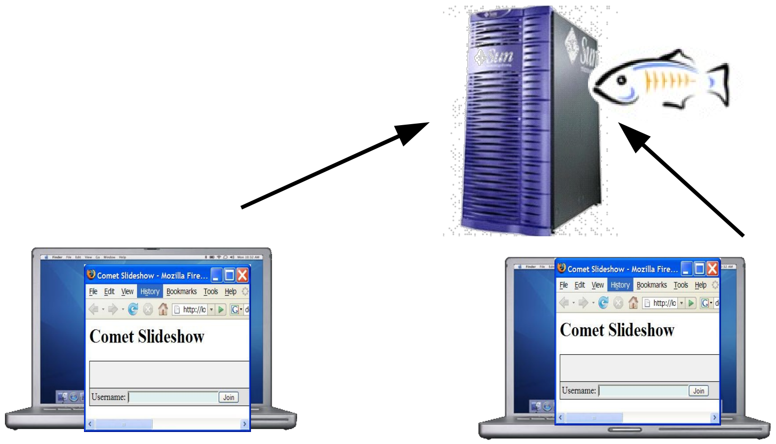
```
<servlet>
    <servlet-name>Grizzly Cometd Servlet</servlet-name>
    <servlet-class>
        com.sun.grizzly.cometd.servlet.CometdServlet
    </servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>Grizzly Cometd Servlet</servlet-name>
    <url-pattern>/cometd/*</url-pattern>
</servlet-mapping>
```

Step2 : Initialize (Connect to Cometd Server)

```
dojo.require("dojox.cometd");

// Initialize a connection to the given Cometd server:
// the GlassFish Grizzly Bayeux servlet
dojox.cometd.init("cometd");
```

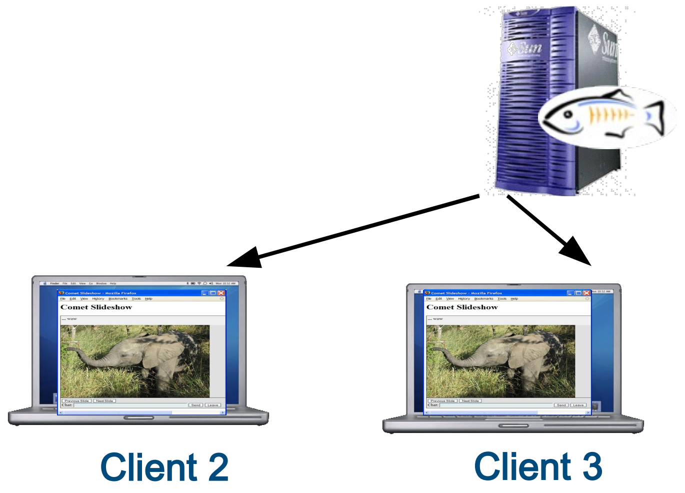
Step3 : Subscribe a Channel

```
// Subscribe a channel with callback function.
// Every time a message is published, the callback function
// (of all clients who subscribed the channel) gets invoked.
dojox.cometd.subscribe(“channel”, "callback receiver",
                                          "callbackFunction");
```


Comet Slideshow - Mozilla Fire...
File Edit View History Bookmarks Tools Help
Comet Slideshow
Username:
Join
Comet Slideshow - Mozilla Fire...
File Edit View History Bookmarks Tools Help
Comet Slideshow
Username:
Join

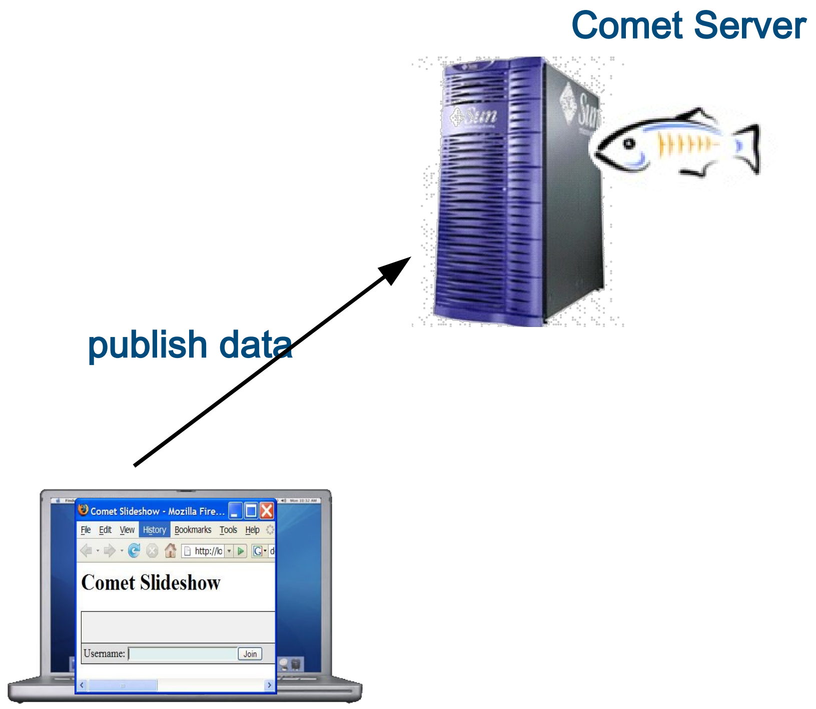
Step4 : Write callback function

```
// The callback function gets invoked when the client receives
// a "published" message from a Cometd server. Typically you
// update the contents on the page.
callbackHandler: function(msg) {
    alert("msg.data.testMessage = " + msg.data.testMessage)
}
```



Step5 : Publish data onto the channel

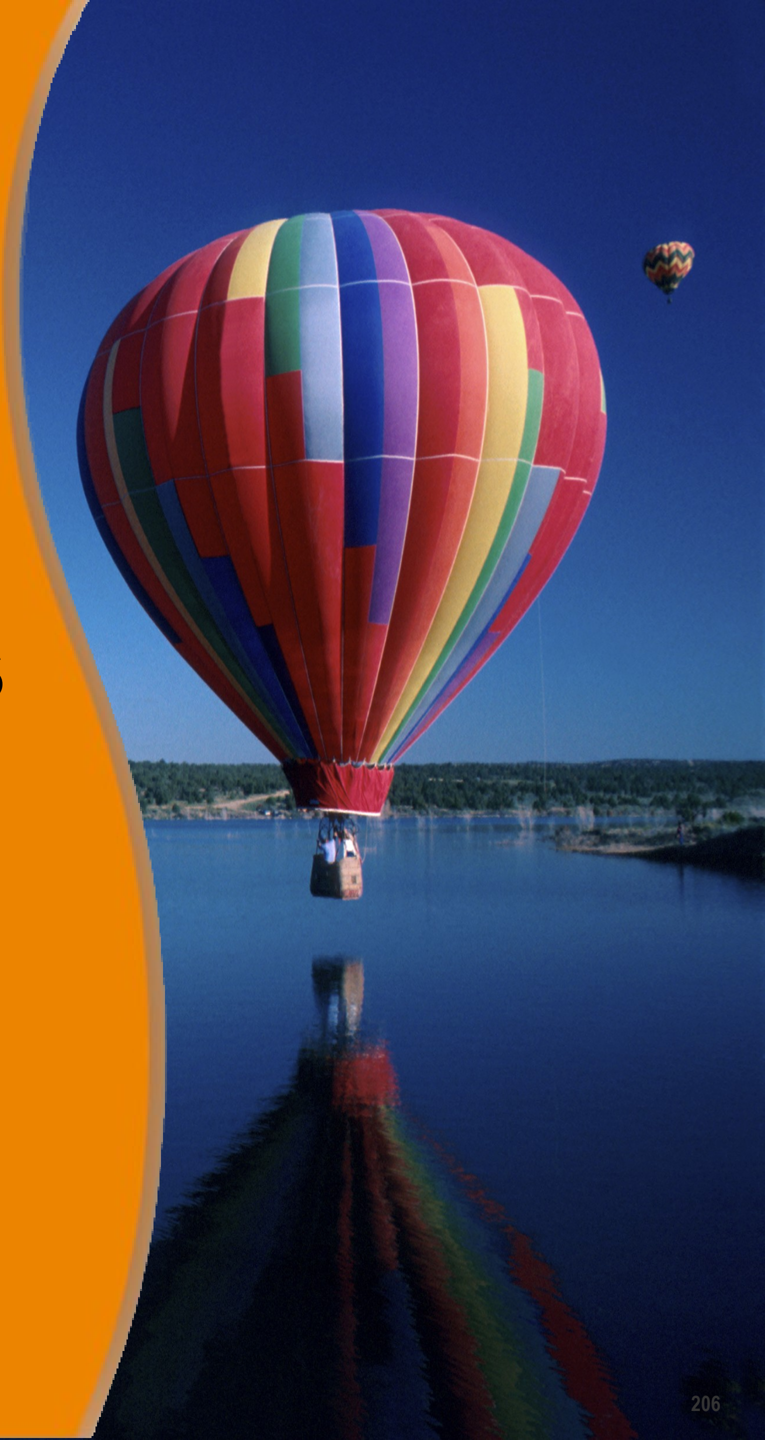
```
// Publish data (in JSON format) onto the channel.
dojox.cometd.publish("channel", {jsonname: jsonvalue});
```


Comet Server
publish data
Comet Slideshow - Mozilla Fire...
File Edit View History Bookmarks Tools Help
Comet Slideshow
Username:
Join

Step6 : Unsubscribe, Disconnect

```
// Unsubscribe the channel
dojox.cometd.unsubscribe(“channel”, myObject, "callbackHandler");

// Disconnect from the Cometd server
dojox.cometd.disconnect();
```

Dojo Utilities & Tools

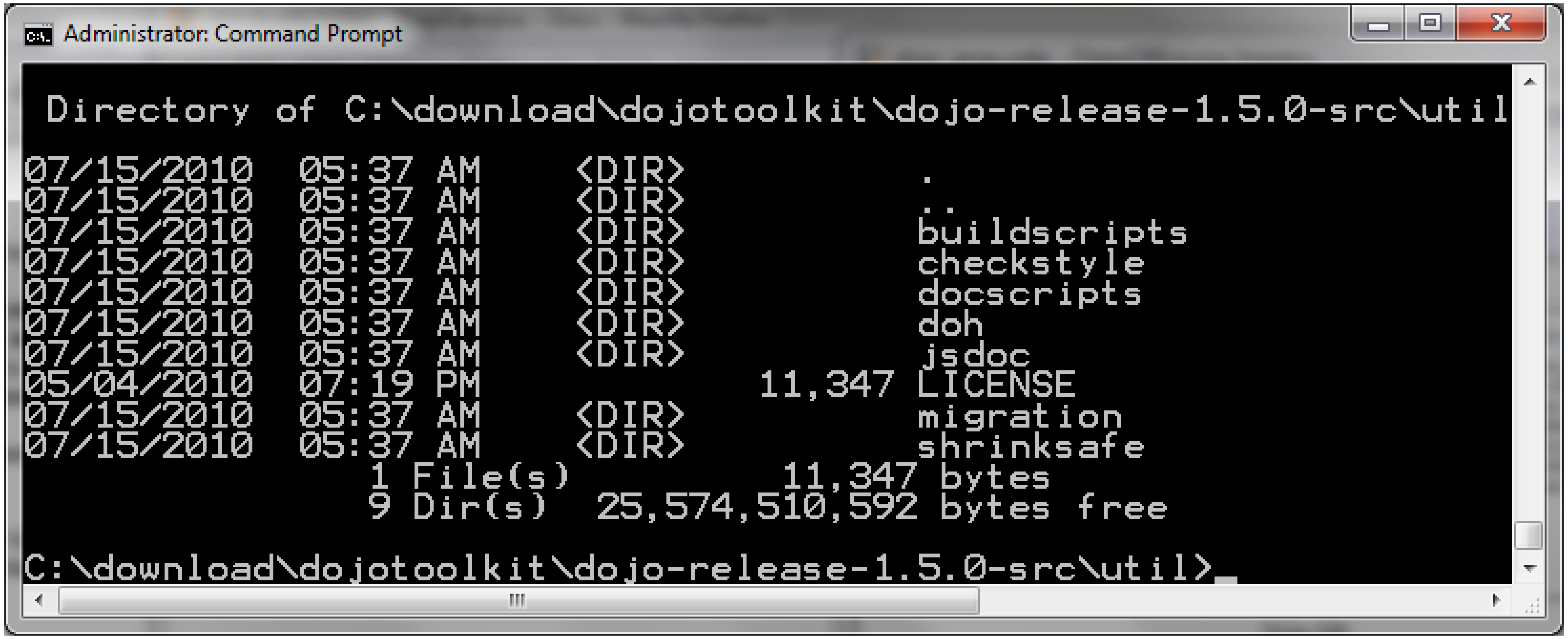


Dojo Utilities & Tools

- Testing via DOH
  - Functional testing
  - Performance testing
- Build system
- ShrinkSafe
- Debugging via Firebug

Where do you get these tools?

- Comes with Dojo toolkit source package

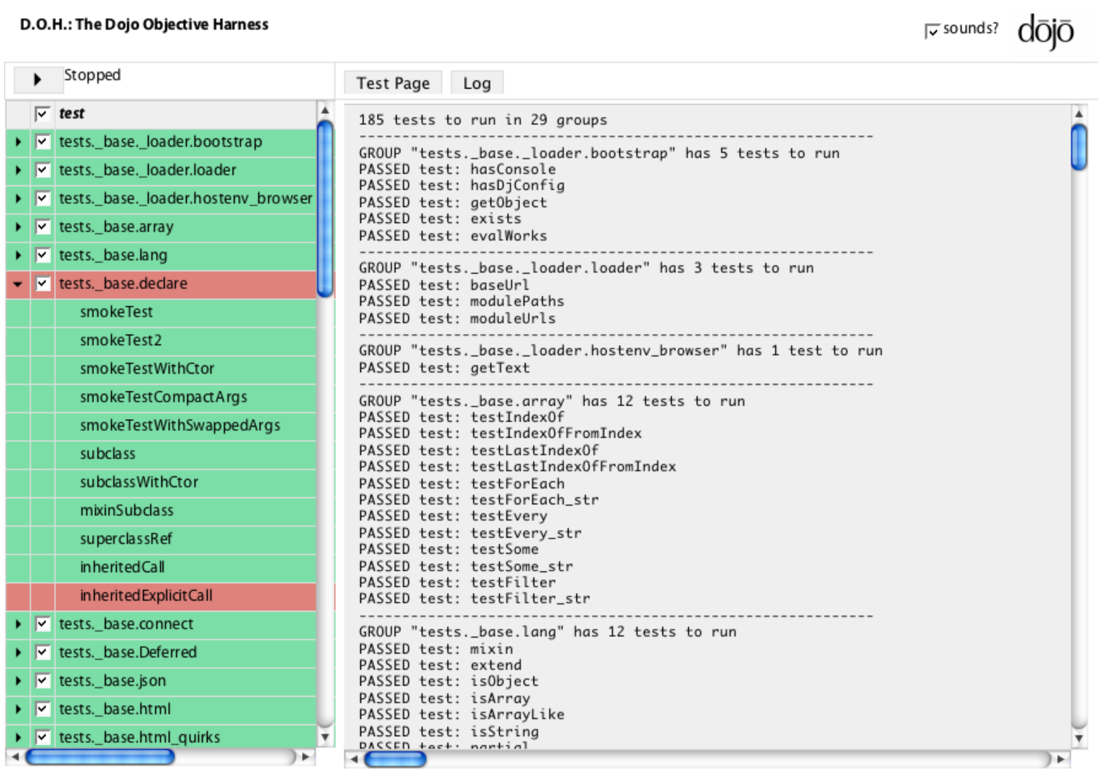


```
 Directory of C:\download\dojotoolkit\dojo-release-1.5.0-src\util

07/15/2010  05:37 AM    <DIR>          .
07/15/2010  05:37 AM    <DIR>          ..
07/15/2010  05:37 AM    <DIR>          buildscripts
07/15/2010  05:37 AM    <DIR>          checkstyle
07/15/2010  05:37 AM    <DIR>          docscripts
07/15/2010  05:37 AM    <DIR>          doh
07/15/2010  05:37 AM    <DIR>          jsdoc
05/04/2010  07:19 PM            11,347 LICENSE
07/15/2010  05:37 AM    <DIR>          migration
07/15/2010  05:37 AM    <DIR>          shrinksafe
               1 File(s)         11,347 bytes
               9 Dir(s)  25,574,510,592 bytes free

C:\download\dojotoolkit\dojo-release-1.5.0-src\util>_
```

Testing via Dojo Objective Harness (DOH)

The Dojo Objective Harness (DOH)

- Dojo Objective Harness ("d.o.h")
  - > New unit test harness
  - > Command-line or within the browser
- Unit test JavaScript and aggregate tests into a single interface



test
- tests._base._loader.bootstrap
- tests._base._loader.loader
- tests._base._loader.hostenv_browser
- tests._base.array
- tests._base.lang
- tests._base.declare
  - smokeTest
  - smokeTest2
  - smokeTestWithCtor
  - smokeTestCompactArgs
  - smokeTestWithSwappedArgs
  - subclass
  - subclassWithCtor
  - mixinSubclass
  - superclassRef
  - inheritedCall
  - inheritedExplicitCall
- tests._base.connect
- tests._base.Deferred
- tests._base.json
- tests._base.html
- tests._base.html_quirks

```
185 tests to run in 29 groups
------------------------------------------------------------
GROUP "tests._base._loader.bootstrap" has 5 tests to run
PASSED test: hasConsole
PASSED test: hasDjConfig
PASSED test: getObject
PASSED test: exists
PASSED test: evalWorks
------------------------------------------------------------
GROUP "tests._base._loader.loader" has 3 tests to run
PASSED test: baseUrl
PASSED test: modulePaths
PASSED test: moduleUrls
------------------------------------------------------------
GROUP "tests._base._loader.hostenv_browser" has 1 test to run
PASSED test: getText
------------------------------------------------------------
GROUP "tests._base.array" has 12 tests to run
PASSED test: testIndexOf
PASSED test: testIndexOfFromIndex
PASSED test: testLastIndexOf
PASSED test: testLastIndexOfFromIndex
PASSED test: testForEach
PASSED test: testForEach_str
PASSED test: testEvery
PASSED test: testEvery_str
PASSED test: testSome
PASSED test: testSome_str
PASSED test: testFilter
PASSED test: testFilter_str
------------------------------------------------------------
GROUP "tests._base.lang" has 12 tests to run
PASSED test: mixin
PASSED test: extend
PASSED test: isObject
PASSED test: isArray
PASSED test: isArrayLike
PASSED test: isString
```

Unit Tests Supported

- Basic DOH infrastructure is for code-only tests
  - DOH Robot adds UI testing in DojoX.
- Support for setup and tear-down.
- Build a test around a page.
- Include all of the different tests styles in a test suite.

DOH Test Assertions

- `doh.assertTrue(x), doh.t(x)`
- `doh.assertFalse(x), doh.f(x)`
- `doh.assertEqual(x, y), doh.is(x, y)`

Build System

What is and Why Use Dojo Build System?

- To achieve the improved performance and better user experience by
  - Reducing the number of files sent over the wire
  - Reducing the JavaScript file sizes

Dojo Build System 4-Step Process

- First, it groups together modules into "layers".
  - A layer, which is one big .js file, loads faster than the individual .js modules that comprise it
- Second, it "interns" external non-JavaScript files.
  - This is most important for Dijit templates, which are kept in a separate HTML file. Interning pulls the entire file in and assigns it to a string.
- Third, it smooshes the layer down with ShrinkSafe.
  - ShrinkSafe removes unneeded whitespace and comments, and compacts variable names down to smaller ones. This file downloads and parses faster than the original.

Dojo Build System 4-Step Process

- Finally, it copies all non-layered scripts to the appropriate places.
  - While this doesn't speed anything up, it ensures that all Dojo modules can be loaded, even if not present in a layer.
  - If you use a particular module only once or twice, keeping it out of the layers makes those layers load faster.

ShrinkSafe

What is ShrinkSafe?

- ShrinkSafe is a standalone Java-based JavaScript compressor which utilizes Rhino to parse code and safely shorten the results.
- ShrinkSafe renames local references to short names prefixed with an underscore.
  - This saves bytes on the wire and also provides some obfuscation of the code.
- It also eliminates whitespace and comments when generating the new code.
- Global references and property names remain unchanged such that external references to the compressed code should be safe.

Example: ShrinkSafe

- Compresses bigcode.js to smallercode.js

```
java -jar shrinksafe.jar bigcode.js > smallercode.js
```

- Compresses three files into one

```
java -jar shrinksafe.jar file1.js file2.js file3.js >
combined.js
```

Debugging via Firebug or Firebug Lite

Firebug or Firebug Lite

- Firebug, an open source debugging extension for Firefox, is "essential" for JavaScript, HTML and CSS debugging
- If you use Internet Explorer or Safari, you can use the Firebug Lite library, bundled with Dojo.
  - This gives you some of the logging and command line features of Firebug.
  - It's not a full emulation, but it's a fairly good alternative and is fully API-compatible
  - To use Firebug Lite, you must include the isDebug config parameter - no effect to Firefox/Firebug

```
<script type="text/javascript" src="http://path/dojo.js"
    djConfig="parseOnLoad: true, isDebug: true"></script>
```

Demo:

Firebug

Resources

Resources

- Web sites
  > http://dojotoolkit.org/
  > http://dojocampus.org/
- Demo sites
  > http://dojocampus.org/explorer/
  > http://demos.dojotoolkit.org/demos/
- Quick start guides
  > http://dojotoolkit.org/reference-guide/quickstart/
  > http://docs.dojocampus.org/quickstart

Resources

- API document
  - > http://api.dojotoolkit.org/
- Forums
  - > http://mail.dojotoolkit.org/mailman/listinfo/dojo-interest

Thank you!

Sang Shin
Founder and Chief Instructor
http://www.JPassion.com
"Learn with JPassion!"