Fluid

Designing software
that works – for everyone

DHTML Techniques for an Inclusive Web 2.0
Or
How to Write JavaScript That Doesn't Suck

Colin Clark, Fluid Project Technical Lead, Adaptive Technology Resource Centre, University of Toronto

Michelle D'Souza, Software Developer, Adaptive Technology Resource Centre, University of Toronto

Eli Cochran, UI Developer, UC Berkeley

Techniques We'll Talk About

Fluid
Designing software that works – for everyone

- Play nice with others: mashups and portals
  - Don't override built-in types
  - Namespacing and uniqueness
  - Addressing the DOM
- Accessibility
  - Keyboard Handling
  - Supporting assistive technologies
  - State of the standards
- Using toolkits

USER EXPERIENCE

Mashups

Fluid
Designing software
that works – for everyone

- What's a mashup?
  - A combination of data and markup from different sources
  - Aggregating content from Web feeds, services, etc.
- Chunks of markup and JavaScript that:
  - Share the same DOM
  - Are mutually ignorant of the others' existence
- The same technical challenges as in a portal

USER EXPERIENCE

Fluid

Designing software
that works – for everyone

Playing Nice With Others

Don't Modify Built-in Types

Fluid
Designing software
that works – for everyone

- JavaScript is wickedly dynamic… but use it carefully
- Our changes can effect other programs.
- In JavaScript, you can easily augment an object with simple assignment.

```
myObject.myMethod = function() { };
```

USER EXPERIENCE

Namespace Everything

Fluid
Designing software
that works – for everyone

- Namespaces will help avoid collisions.
- Encapsulate functionality nicely.
- Provides some documentation.
- How do we do this?
- JavaScript has a global object which holds top level functions and global variables.
  - Create an object in the global space and put everything in it.

USER EXPERIENCE

Sample Code: Defining Namespaces

Fluid
Designing software that works – for everyone

```
var myNameSpace = myNameSpace || { };

myNameSpace.foo = function () { alert("bar"); };

myNameSpace.foo();
```

USER EXPERIENCE

...Even Your Markup

Fluid
Designing software
that works – for everyone

- In a portal we don't control the whole page
- Markup can show up in multiple places on the page
- Unique ids are the key to addressing particular elements

Sample Markup: Semantic IDs

Fluid
Designing software
that works – for everyone

Example:

tool.context.widget.element

becomes

announcement.list.navToolBar.edit

USER EXPERIENCE

Sample Code: Finding By ID

Fluid
Designing software
that works – for everyone

```
// Find an element using an explicit ID
var elm =
document.getElementById(‘
    announcement.list.navToolBar.new ‘);
```

USER EXPERIENCE

Fluid

Designing software
that works – for everyone

Accessibility

DHTML & Accessibility

Fluid
Designing software
that works – for everyone

- Just when we thought we had Web accessibility in hand...
  - Not enough information: opaque user interface markup
  - Non-mouse usage is often overlooked completely
  - Dynamically updated information can be challenging

Assistive Technologies

Fluid
Designing software that works – for everyone

- Used by people with disabilities to perceive and control the user interface.
  - Screen reader
  - Screen magnifier
  - On-screen keyboard
- Most assistive technologies use built-in operating system APIs for reflecting the user interface:
  - Windows: MSAA/IAccessible2
  - Linux: ATK
  - Mac: Universal Access for Carbon and Cocoa

USER EXPERIENCE

Opaque Markup

Fluid
Designing software
that works – for everyone

- HTML has limited semantics:
  - Forms, links, buttons, lists, tables
- Dynamic UIs are built from generic HTML tags
  - For example, and
  - No <slider> or <menu> tags available
- Assistive technologies attempt to read the underlying document markup
- Problem: how do assistive technologies represent DHTML interfaces to the user?

USER EXPERIENCE

Example of Opaque Markup:

Fluid
Designing software
that works – for everyone

- A DHTML menu bar without semantics:

```
<ol id=”menubar”>
        <li id="editMenu">Edit
            <ol>
                <li>Cut</li>
                <li>Copy</li>
                <li>Paste</li>
            </ol>
        </li>
</ol>
```

USER EXPERIENCE

Opaque Markup: Solution

Fluid
Designing software
that works – for everyone

- Provide additional semantics or metadata that describe the role, function, and states of DHTML user interfaces. How?
- **ARIA (Accessible Rich Internet Application)**

http://www.w3.org/TR/aria-roadmap/
http://www.w3.org/TR/aria-role/
http://www.w3.org/TR/aria-state/

- Working standard from the W3C, led by Fluid partner Rich Schwerdtfeger

USER EXPERIENCE

ARIA

Fluid
Designing software
that works – for everyone

- Attributes added to your HTML markup that describe the function and states of your UI components
- These map to all your familiar types of UI widgets:
  - Dialog
  - Slider
  - Progress Bar
  - Tab Panel
  - Menu bar

USER EXPERIENCE

Sample Code: ARIA Roles

Fluid
Designing software
that works – for everyone

- A DHTML menu bar with ARIA semantics:

```
<ol id=”menubar” role=”wairole:menubar”>
    <li id="editMenu" role="wairole:menuitem“
                      haspopup=“true”>Edit
        <ol>
            <li role="wairole:menuitem">Cut</li>
            <li role="wairole:menuitem">Copy</li>
            <li role="wairole:menuitem">Paste</li>
        </ol>
    </li>
</ol>
```

USER EXPERIENCE

The Value of ARIA

Fluid
Designing software that works – for everyone

- DHTML accessibility is a short-term problem
- Long-term, it has the potential to make web accessibility much better
- Assistive technology developers have had a decade to get desktop GUI accessibility right
- By mapping rich-client interfaces with ARIA, web interfaces can leverage this support

USER EXPERIENCE

Non-mouse accessibility

Fluid
Designing software
that works – for everyone

- Most rich Web interactions *require* the mouse.
- Standard tabbing strategy in browsers is tedious
- Keyboard bindings will enable lots of non-mouse control strategies, including:
  - On-screen keyboard
  - Single switch
  - Voice control

USER EXPERIENCE

Tabbing and tabindex

Fluid
Designing software
that works – for everyone

- Browsers used to only allow you to use tab to focus form elements and links
- There is an HTML attribute called “tabindex” that allows you to tell the browser how to handle tabbing
- Strategy:
  - allow the user to tab to user interface widgets
  - use the arrow keys allow selection within
  - Add JavaScript handlers for arrow keys

USER EXPERIENCE

Sample Markup: Tabindex

Fluid
Designing software
that works – for everyone

```
<ol id=”menubar” tabindex=”0”>
    <li id="editMenu">Edit
        <ol>
            <li><a href=”/cut” tabindex=”-1”>Cut</a></li>
            <li><a href=”/copy” tabindex=”-1”>Copy</li>
            <li><a href=”/paste” tabindex=”-1”>Paste</li>
        </ol>
    </li>
</ol>
```

USER EXPERIENCE

Sample Code: Keyboard Handlers

Fluid
Designing software that works – for everyone

```
jQuery(elmRef).keydown( function(event) {
        switch(event.keyCode){
                case 40: // 40 = Arrow Down
                        // highlight the next element
                        jQuery(elmRef).removeClass('highlight');
                        var nextElm = jQuery(elmRef).next();
                        jQuery(nextElm).addClass('highlight');
                case 38:  // 38 = Arrow Up
                        // highlight the prev element
});
```

USER EXPERIENCE

DHTML Accessibility Advice

Fluid
Designing software that works – for everyone

- Out of date accessibility standards and legislation
  - Technology-specific standards go out of date easily
  - Current standards impede innovation
- Strategy:
  - Embrace JavaScript
  - Use emerging standards: ARIA, tabindex, etc.
  - Degrade gracefully
  - Think about the use case for accessibility
  - Start with accessibility, don’t add it at the end

USER EXPERIENCE

Accessibility Meta Concepts

Fluid
Designing software
that works – for everyone

1. Label everything
- Design for variable font and screen sizes
- It has to work with the keyboard

USER EXPERIENCE

Fluid

Designing software
that works – for everyone

JavaScript Toolkits

JavaScript is Painful

Fluid
Designing software
that works – for everyone

- Four points of pain:
  - Browser bugs and inconsistencies
  - DOM traversal and selection
  - Event management
  - AJAX

USER EXPERIENCE

Why use a JS Framework?

Fluid
Designing software
that works – for everyone

- Leverage someone else's hell
- Someone else wrote it...
- ...and someone's already tested it
- The framework handles the fundamentals

USER EXPERIENCE

Summary

Fluid
Designing software
that works – for everyone

- Key techniques:
  - Don't poke around with the built in types
  - Namespace everything
  - Be careful of vacuuming up the DOM
  - Make it work with the keyboard
  - Add ARIA roles and states
- Toolkits will save you time. We like:
  - jQuery for just about everything: “jQuery **is** the DOM”
  - Dojo for black-box, accessible widgets (for now)

USER EXPERIENCE

Join us for a JavaScript BOF

Fluid

Designing software
that works – for everyone

- JavaScript Birds of a Feather
- 3:40 pm in the Laguna room

- More time to talk about JavaScript
- Bring your ideas and questions!

USER EXPERIENCE