The Inclusive Web

Hands on with HTML5 and jQuery

Justin Obara & Colin Clark
Inclusive Design Research Centre
fluidproject.org
A bit about us, quickly
What is accessibility?
Rethinking Disability
Rethinking Disability
Rethinking Disability

A mismatch between the user and the user interface
Disability is a usability issue

C:\>DIR A:

Not ready reading drive A
Abort, Retry, Fail?_
Disability is contextual
Designing for Context
Disability is environmental
Accessibility is... the ability of the system to accommodate the needs of the user
the web today
jQuery Hits the Spot
jQuery Hits the Spot

• Browser inconsistencies and bugs
jQuery Hits the Spot

- Browser inconsistencies and bugs
- Complexity of the DOM
jQuery Hits the Spot

- Browser inconsistencies and bugs
- Complexity of the DOM
- Handling events and asynchrony
jQuery Hits the Spot

- Browser inconsistencies and bugs
- Complexity of the DOM
- Handling events and asynchrony
- Communicating with the server
Toolkits can help!

- **Browser Abstraction**
- **Complexity of the DOM**
- **Handling events and asynchrony**
- **Communicating with the server**
Toolkits can help!

- Browser abstraction
- A simple, unified API for the DOM
- Handling events and asynchrony
- Communicating with the server
Toolkits can help!

- Browser abstraction
- A simple, unified API for the DOM
- Easy, functional events system
- Communicating with the server
Toolkits can help!

- Browser abstraction
- A simple, unified API for the DOM
- Easy, functional events system
- Built-in AJAX, XML, and JSON
function stripeListElements() {
    // get the items from the list
    var myItems = document.getElementsByTagName("li");
    // skip line 0 as it's the header row
    for(var i = 0; i < myItems.length; i++) {
        if ((i % 2) === 0) {
            myItems[i].className = "striped";
        }
    }
}
With jQuery

jQuery("li");
With jQuery

jQuery("li:even");
With jQuery

jQuery("li:even").addClass("striped");
Accessible systems are...

- Flexible
- Separable
- Modifiable
Graceful Degradation
It's just a couple of classes!
Hide the fancy stuff, show the basics by default.
The Code

// Use JavaScript to hide basic markup.
$("head") .append("<style type='text/css'>
  .fl-progEnhance-basic { display: none; }
  .fl-progEnhance-enhanced { display: block; }
</style>");

Use JavaScript to flip the styles around!
how assistive technology works
keyboard navigation & aria
Opaque Markup

<!-- This is a Tabs widget. -->
<!-- How would you know, looking only at the markup? -->

<ol>
    <li id="ch1Tab">
        <a href="#ch1Panel">Chapter 1</a>
    </li>
    <li id="ch2Tab">
        <a href="#ch2Panel">Chapter 2</a>
    </li>
    <li id="quizTab">
        <a href="#quizPanel">Quiz</a>
    </li>
</ol>

<div>
    <div id="ch1Panel">Chapter 1 Stuff</div>
    <div id="ch2Panel">Chapter 2 Stuff</div>
    <div id="quizPanel">Quiz Stuff</div>
</div>
Chapter 1
Prolegomena

What is logic?

Logic is the study of the consistency of beliefs. For beliefs to be consistent it must be possible for them to obtain at the same time. For example, it is illogical to believe that the sky is completely blue and that the sky is completely red because the sky being entirely blue is inconsistent with its being entirely red, i.e. it is not possible for the sky to be entirely red at the same time as its being entirely blue.

Logic is also a study of "logical consequence", i.e. what follows by necessity from something else. By studying inconsistency of beliefs, philosophers are able to study the validity of arguments, as will be shown later. Methods of finding whether certain arguments are valid is described later.

The symbolisation of these sentences, known as formalisation, simplifies and quickens this process. It also enables the philosopher to clarify ideas using an unambiguous language in which to represent thoughts. The sophistication of the language used enables greater insights into the significance of these thoughts (and a cursory analysis of more logical languages is described in Other Logics).
ARIA fills the gap
Roles, States, Properties

• **Roles** describe widgets not present in HTML 4
  
  slider, menubar, tab, dialog

• **Properties** describe characteristics:
  
  draggable, hasPopup, required

• **States** describe what’s happening:
  
  busy, disabled, selected, hidden
Using ARIA

<!-- Now *these* are Tabs! -->
<ol role="tablist">
   <li id="ch1Tab" role="tab">
      <a href="#ch1Panel">Chapter 1</a>
   </li>
   <li id="ch2Tab" role="tab">
      <a href="#ch2Panel">Chapter 2</a>
   </li>
   <li id="quizTab" role="tab">
      <a href="#quizPanel">Quiz</a>
   </li>
</ol>

<div>
   <div id="ch1Panel" role="tabpanel" aria-labelledby="ch1Tab">Chapter 1 Stuff</div>
   <div id="ch2Panel" role="tabpanel" aria-labelledby="ch2Tab">Chapter 2 Stuff</div>
   <div id="quizPanel" role="tabpanel" aria-labelledby="quizTab">Quiz Stuff</div>
</div>
Adding ARIA in code

// Identify the container as a list of tabs.
tabContainer.attr("role", "tablist");

// Give each tab the "tab" role.
tabs.attr("role", "tab");

// Give each panel the appropriate role,
panels.attr("role", "tabpanel");
panels.each(function (idx, panel) {
    var tabForPanel = that.tabs.eq(idx);
    // Relate the panel to the tab that labels it.
    $(panel).attr("aria-labelledby", tabForPanel[0].id);
});
Keyboard Navigation

- Everything that works with the mouse should work with the keyboard
- ... but not always in the same way
- Support familiar conventions

http://dev.aol.com/dhtml_style_guide
Keyboard Conventions

• **Tab** key focuses the control or widget

• **Arrow keys** select an item

• **Enter** or **Spacebar** activate an item

Tab is handled by the browser. For the rest, you need to write code. A lot of code.
Chapter 1
Prolegomena

What is logic?

Logic is the study of the consistency of beliefs. For beliefs to be consistent it must be possible for them to obtain at the same time. For example, it is illogical to believe that the sky is completely blue and that the sky is completely red because the sky being entirely blue is inconsistent with its being entirely red, i.e. it is not possible for the sky to be entirely red at the same time as its being entirely blue.

Logic is also a study of "logical consequence", i.e. what follows by necessity from something else. By studying inconsistency of beliefs, philosophers are able to study the validity of arguments, as will be shown later. Methods of finding whether certain arguments are valid is described later.

The symbolisation of these sentences, known as formalisation, simplifies and quickens this process. It also enables the philosopher to clarify ideas using an unambiguous language in which to represent thoughts. The sophistication of the language used enables greater insights into the significance of these thoughts (and a cursory analysis of more logical languages is described in Other Logics).
Tabindex examples

<!-- Tab container should be focusable -->
<ol id="animalTabs" tabindex="0">
  <!-- Individual Tabs shouldn’t be focusable -->
  <!-- We’ll focus them with JavaScript instead -->
  <li id="tab1">
    <a href="#cats" tabindex="-1">Cats</a>
  </li>
  <li id="tab2">
    <a href="#cats" tabindex="-1">Dogs</a>
  </li>
  <li id="tab3">
    <a href="#cats" tabindex="-1">Alligators</a>
  </li>
</ol>
Making Things Tabbable

- Tabindex varies subtly across browsers
- jQuery.attr() normalizes it as of 1.3
- For all the gory details:

// Make the tablist accessible with the Tab key.
$tabContainer.attr("tabindex", "0");
// And take the anchors out of the Tab order.
$("a", tabs).attr("tabindex", "-1");
Adding the Arrow Keys

// Make each tab accessible with the left and right arrow keys.

// Make each tab accessible with the left and right arrow keys.
tabContainer.fluid("selectable", {
    selectableSelector: that.options.selectors.tabs,
    direction: fluid.a11y.orientation.HORIZONTAL,
    onSelect: function (tab) {
        $(tab).addClass(that.options.styles.highlighted);
    },

    onUnselect: function (tab) {
        $(tab).removeClass(that.options.styles.highlighted);
    }
});
Making Them Activatable

// Make each tab activatable with Spacebar and Enter.
tabs.fluid("activatable", function (evt) {
    // Your handler code here. Maybe the same as .click()?
});
Documentation

• Tutorial:
  http://wiki.fluidproject.org/display/fluid/Keyboard+Accessibility+Tutorial

• API Reference:
  http://wiki.fluidproject.org/display/fluid/Keyboard+Accessibility+Plugin+API
the web tomorrow
“you have to use flash for that”

“the web can’t do that!”

“you need an app for that!”
augmented reality
mobile
Beyond the buzzword...

- Media, drawing, animation, and interactivity
  <audio>, <video>, <canvas>
- New widgets—you don’t have to roll your own
  <progress>, <menu>
- Richer semantics for forms and documents
  <article>, <nav>, <input type="date">
Other cool stuff...

- CSS3
  transition, transform, gradient

- Working with files
  File API, FormData, XHR Level 2

- Coming soon
  Device, Text to Speech!
What about accessibility?
Making use of semantics
What’s coming

- **Headings**
  Based on nesting within sections

- **Continued enhancements from semantics**
  e.g. improved AT awareness for navigation `<nav>`

- **Native widgets**
Canvas Accessibility
Canvas Accessibility

```javascript
// if highlight line is on, paint the highlight color
if ((settings & settings.isSettingOn('highlightline')) &&
    (currentline == ed.cursorManager.getCursorPosition().row)) {
    ctx.fillStyle = theme.highlightCurrentLineColor;
    ctx.fillRect(x + (Math.abs(this.xoffset)), y, linewidth, this.lineHeight);
    // if not on highlight, see if we need to paint the zebra
    } else if ((currentline % 2) == 0) {
    ctx.fillStyle = theme.zebraStripeColor;
    ctx.fillRect(x + (Math.abs(this.xoffset)), y, linewidth, this.lineHeight);
}

x += this.LINE_INSETS.left;

cy = y + (this.lineHeight - this.LINE_INSETS.bottom);

// paint the selection bar if the line has selections
var selections = this.selectionHelper.getRowSelectionPositions(currentline);
if (selections) {
    tx = x + (selections.startCol * this.charWidth);
    tw = (selections.endCol == -1) ? (lastColumn - firstColumn) * this.charWidth : (selections.endCol - selections.startCol) * this.charWidth;
    ctx.fillStyle = theme.editorSelectedTextBackgroundColor;
    ctx.fillRect(tx, y, tw, this.lineHeight);
}

var lineMetadata = this.model.getRowMetadata(currentLine);
var lineText = lineMetadata.lineToText;
var searchIndices = lineMetadata.searchIndices;
```
Canvas Accessibility

1. Shadow DOM
2. Focus indicators
   ... not quite yet.

In the meantime...

1. Build alternatives
2. Degrade gracefully
The Bottom Line

• HTML5 is coming—experiment with it now
• Lots of great potential for improving access
• Assistive technologies are slow on the uptake
• Some features are going to be a challenge (Canvas)
building cool stuff
an HTML5 uploader
Features

• Degrades gracefully
• Uploads multiple files at once
• Keyboard navigable
• Uses hot new HTML5 features:

  FormData  XMLHttpRequest Level 2  <progress> (almost!)
Dive right in: markup

<input type="file" multiple=""
    id="d-uploader-filesControl"
    class="d-uploader-filesControl fl-progEnhance-basic" />

Wednesday, June 1, 2011
Getting the files

```javascript
filesControl.change(function () {
    that.events.onAdd.fire(filesControl[0].files);
});
```
demo.uploader.sendRequest = function (file, url, events) {
    var formData = new FormData();
    formData.append("file", file);

    // Create a new XHR.
    var xhr = new XMLHttpRequest();
    xhr.open("POST", url, true);

    // Register success and error listeners.
    xhr.onreadystatechange = function () {
        if (status === 200) {
            events.onSuccess.fire(file);
        } else {
            events.onError.fire(file);
        }
    }

    // Listen for progress events as the file is uploading.
    xhr.upload.onprogress = function (progressEvent) {
        events.onProgress.fire(file, progressEvent.loaded, progressEvent.total);
    }

    // Send off the request to the server.
    xhr.send(formData);
};
HTML5 Inputs

<input type="tel"> <!-- phone number -->
<input type="email"> <!-- e-mail address -->
<input type="date"> <!-- date -->
<input type="search"> <!-- search field -->

<!-- number field -->
<input type="number" min="0" max="10" step="1" value="1">

<!-- Like an autocomplete widget -->
<input list="dlist">
<datalist id="dlist"><option value="HTML5"></datalist>
<label for="name">Name</label>
<input type="text" id="name" placeholder="My name is ..." required autofocus />
Geolocation

// test if geolocation api is supported
if (!navigator.geolocation) {
    // success callback is passed a location object
    // coords property holds coordinate information
    // Firefox also has an address property
    navigator.geolocation.getCurrentPosition(success, error);
}
Geolocation: Location Object

// test if geolocation api is supported
if (!!!navigator.geolocation) {
    // success callback is passed a location object
    navigator.geolocation.getCurrentPosition(success, error);
}
What’s Infusion?

- Application framework built on top of jQuery
- UI components you can reuse and adapt
- Lightweight CSS framework for styling
- Accessibility tools and plugins for jQuery
- Open architecture: everything is configurable
Great UX is hard work

- Your code gets unruly as it grows
- UIs are hard to reuse or repurpose
- Design change requires big code change
- Accessibility is confusing
- Combining different code/libraries doesn’t always work
No Black Boxes

Open Architecture:
Unlock your markup
Let developers *and users* in
A widget isn’t just one thing
Question the rules

No Black Boxes
Transparent Apps

- M is where it’s at
- Events inside and out
- Assistive technology inside the Web, not bolted on
CSS Frameworks

“If you’re going to use a framework, it should be yours; one that you’ve created. You can look at existing frameworks for ideas and hack at it. But the professionals in this room are not well served by picking up a framework and using it as-is.”

- Eric Meyer
Fluid Skinning System

- FSS is built to be hacked on
- Provides a core set of building blocks
- Reset, text, layouts, themes
- Namespaced: no conflicts with your stuff
- Themes for better legibility & readability

http://wiki.fluidproject.org/x/96M7
Photo Credits


Curb cut, Great PA-NJ, http://www.flickr.com/photos/50393252@N02/4822063888/

Stethoscope, Han-Oh Chung, http://www.flickr.com/photos/chickenlump/2038512161/

Texting while walking, Mobile Monday Amsterdam, http://www.flickr.com/photos/momoams/2926622070/

MOMA WiFi, http://www.flickr.com/photos/89554035@N00/2445178036

