The Desktop Advantage

- Large selection of GUI components
- Rich event model
- Low response time

HTML Event Models

- HTML 4 Event Model
  - HTML 4.01 Specification - http://www.w3.org/TR/REC-html40/interact/scripts.html#h-18.2.3
  - Limited but widely supported
- Standard Event Model
  - DOM Level 2 HTML Specification - http://www.w3.org/TR/DOM-Level-2-Events/events.html
- Browser specific event models

Events and Event Handler

- Events
  - onfocus, onblur, onkeypress, onkeydown, onkeyup, onclick, ondblclick, onmousedown, onmouseup, onmousemove, onmouseout...
- Specify event handler
  - <element event="code"/>
  - For example:
    <button onclick="clickHandler();">click</button>

Example: Event Handling

- j1.html
  - Uses X Library from http://cross-browser.com/
  - Handles events
  - Modifies the HTML document
JavaScript

Interpreted language
Originally developed by Netscape
Syntax is similar to Java

Core JavaScript

Mainly covers language syntax, which is similar to Java
Some “un-Java-like” language features
- Object creation
- Functions as first-class citizens

Object Creation – Approach 1

```javascript
var car = new Object();
car.make = 'Honda';
car.model = 'Civic';
car.year = 2001;

var owner = new Object();
owner.name = 'Chengyu';
car.owner = owner;
```

- A JavaScript object consists of a set of properties which can be added dynamically

Object Creation – Approach 2

```javascript
var car = {
    make: 'Honda',
    model: 'Civic',
    year: 2001,
    owner: {
        name: 'Chengyu'
    }
};
```

- Object literal

Functions as First-class Citizens

- In JavaScript, functions are considered objects like other object types
  - Assigned to variables
  - Assigned as a property of an object
  - Passed as a parameter
  - Returned as a function result
  - Function literals (i.e. functions without names)

Function Examples

```javascript
function foo() {
    alert('foo');
}

bar = function() {
    alert('bar');
}

setTimeout( bar, 5000 );
setTimeout( function() {
    alert('foobar');
}, 5000 );
```

- Regular function creation
- Function literal
- Function as parameter
- Function literal as parameter
Client-Side JavaScript

- Embed JavaScript in HTML
  - `<script>
    *type="text/javascript"
    *language="JavaScript"
    *src="path_to_script_file"
  </script>`
- Run inside a browser

Processing an HTML Document

```html
<html>
<head>
  <title>JavaScript Example</title>
</head>
<body>
  <h1>JavaScript Example</h1>
  <p>Some content.</p>
</body>
</html>
```

- As a text file – very difficult
- As an object

Document Object Model (DOM)

- Representing documents as objects so they can be processed more easily by a programming language

DOM Representation

```plaintext
Document
  - HTMLDocument
    - Text
    - Comment
  - Element
  - HTMLElement
```

Manipulate a Document

- Find Elements
- Modify Elements
- Create Elements
Find Elements

- `document.getElementById()`
- `document.getElementsByTagName()`
- `document.getElementsByName()`

Modify Elements ...

- HTML Element properties and methods
  - IE
    - `innerHTML`
    - `innerText`
    - `insertAdjacentHTML()`
    - `insertAdjacentText()`
  - Netscape/Mozilla
    - `innerHTML`
  - Element-specific

... Modify Elements

- `node`
  - `setAttribute()`, `removeAttribute()`
  - `appendChild()`, `removeChild()`
  - `insertBefore()`, `replaceChild()`

Create Elements

- `document`
  - `createElement()`
  - `createTextNode()`

Example: Document Manipulation

- `j2.html`
  - Read and display the text input
  - Display "Hello <name>"??
  - Add text input to table??

Communicate with Server

- The request-response model is still a limiting factor in user interactivity
- Solution: XMLHttpRequest
  - A JavaScript object
    - Send request and receive response
  - Response can be handled *asynchronously*
    - Do not need to wait for the response
Understand Asynchronous

- **Synchronous**
  - `send( request );`
  - `// wait for response`
  - `process( response );`
  - `// do other things`

- **Asynchronous**
  - `send( request );`
  - `// don’t wait for response`
  - `process( response );`
  - `// do other things`

  What’s the problem??
  What’s the solution??

About the Example

- `clickHandler()`
- `newXMLHttpRequest()`
- `updateDocument()`
- `getReadyStateHandler()`

(XMLHttpRequest - Properties)

- `onreadystatechange`
- `readyState`
  - 0 – uninitialized
  - 1 – loading
  - 2 – loaded
  - 3 – interactive
  - 4 – complete
- `status`
- `statusText`

(XMLHttpRequest - Methods)

- `abort()`
- `getAllResponseHeaders()`
- `getResponseHeader( header )`
- `open( method, url, asyncFlag, username, password )`
  - asyncFlag, username, password are optional
- `send( messageBody )`
- `setRequestHeader( name, value )`

So What is Ajax?

- **Asynchronous JavaScript and XML**
  - JavaScript + XMLHttpRequest
- **Characteristics of Ajax**
  - Non-blocking – the server response is handled asynchronously with a callback function
  - Partial page update using JavaScript

An XMLHttpRequest Example

- `a1.html`
  - A client script sends an XMLHttpRequest
  - A servlet responds with a random number
  - When the message arrives on the client, a callback function is invoked to update the document
More About AJAX

- XMLHttpRequest used to be an IE specific feature that received little attention
- It's all started by Google Maps
- The beginning of “Web 2.0”

Problems of Plain JavaScript + XMLHttpRequest

- Each browser has their own JavaScript implementation
  - Code that works on some browsers may not work on others
- Lack of pre-made GUI components
- Implementing Ajax operations is quite tedious

One Library to Rule Them All – jQuery

- jQuery - http://jquery.com/
- jQuery UI - http://jqueryui.com/
- The increasing market share of jQuery
  - http://trends.builtwith.com/javascript/JQuery
  - http://trends.builtwith.com/javascript/JQuery_tv

Key Elements of an Ajax Operation

<table>
<thead>
<tr>
<th>Client</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event</td>
<td>Event handler</td>
</tr>
<tr>
<td>Event handler</td>
<td>Create a XMLHttpRequest</td>
</tr>
<tr>
<td></td>
<td>Attach a callback function</td>
</tr>
<tr>
<td></td>
<td>Send the request</td>
</tr>
<tr>
<td>Callback function</td>
<td>Process the response</td>
</tr>
<tr>
<td></td>
<td>Update the HTML Page</td>
</tr>
</tbody>
</table>

JavaScript/Ajax Frameworks and Libraries

  - Cross-browser compatibility
    - New JavaScript API, e.g. X Lib, JQuery
    - New language, e.g. ZK, Taconite
  - Pre-made, Ajax-enabled GUI component
  - Simplify the implementation of Ajax operations

A jQuery Example

- a2.html
  - The document ready handler
    $(function() { ... })
    - Similar to window.onload but better
  - Selectors $(‘#clickBtn’) and $(‘#number’)
  - Events click()
  - Ajax call $.ajax()
Readings

- jQuery in Action by Bear Bibeault and Yehuda Katz

What’s in the Future? – RIA vs. Ajax and HTML5

- Rich Internet Application (RIA) platforms
  - Flex, Silverlight, JavaFX
- vs. Ajax and HTML5
  - Proprietary
  - Require browser plugins
  - Rich GUI features
  - Good development tool support