Ajax – Design and Usability

William Hudson
william.hudson@syntagm.co.uk
www.syntagm.co.uk/design

Ajax – Design and Usability

- About Ajax
  - Ajax in context
  - How Ajax works
  - How Ajax is different
  - How Ajax is similar

- Computer-Human Interaction
  - Change blindness
  - Attentional gambling
  - Flow
  - Feedback

- Design and usability guidelines
Activity 1

- Agree/elect/appoint a spokesperson/note taker
- Discuss the issues raised on the worksheets
- At the end of the discussion the spokesperson will give a verbal summary (3 – 5 minutes) to the whole group

Activity 1

Worksheet issues

1. Introductions – name, company, job role
2. What does Ajax mean to you?
3. Describe some of your most pressing user experience problems. Do you think Ajax will help?
4. Do feel any pressure to use AJAX? What form does it take?
AJAX

- Asynchronous
- JavaScript
- And XML

Ajax in Context

AJAX

- Asynchronous
- JavaScript
- And XML – not needed – client and server can use any convenient data format
Ajax in Context

- Asynchronous
- **JavaScript** – not really, it could be any scripting language such as VBScript

Ajax in Context

- Asynchronous – only relative to page loading, to users it looks synchronous
So what is Ajax?

- Ajax ≈ Remote Scripting
- It’s an approach for making server requests without reloading a page
  - Relies on non-standard server request facilities (XMLHttpRequests for example) supported by most browsers
  - Uses Document Object Model (DOM) scripting to make changes to the current page, for example:
    - Show popup windows (div elements)
    - Change the contents of page elements
    - Hide, delete or move page elements
Ajax Example: Google Maps

Google Maps allows near-real-time scrolling and zooming

How Ajax Works

A standard page request without scripting
How Ajax Works

An asynchronous request within a page

How Ajax is Different

Without Ajax (in general)
- Request is for a URL
- Anything that is returned must be a complete page with appropriate headers

With Ajax
- Client-side scripts use non-standard XMLHttpRequest to make a request for data
- The data does not have to be XML!
- Request can complete at any time but user is not left waiting for a new page
- Updates achieved through the Document Object Model
Technical Benefits of Ajax

- Unit of information reduced from a complete page (or frame) – can be as little as a single character
  - Faster than returning a whole page
  - Data-oriented requests means greater code reuse – especially server-side (in form validation, for example)
  - Overall approach can be much more like a desktop application

A major benefit of Ajax is reduced granularity

Ajax Example: Google Suggest

1) Just the ‘a’ is sent to the server

As you type, Google

2) Only the visible list is returned (but in no obvious order)
Technical Drawbacks of Ajax

- The asynchronous data requests rely on technology that is not standard (yet)
  - Different code needed on different browsers
  - Requests may not complete in a timely fashion (or at all)
  - Pages laden with code will take longer to load
  - Network overheads may make short transfers slow in proportion

How Ajax is Similar

<table>
<thead>
<tr>
<th></th>
<th>Installation</th>
<th>Speedy Start</th>
<th>Granularity</th>
<th>Portability</th>
<th>Maintenance</th>
<th>Sophistication</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML Only</td>
<td>*****</td>
<td>***</td>
<td>*</td>
<td>****</td>
<td>****</td>
<td>*</td>
</tr>
<tr>
<td>Simple Client-Side Scripting</td>
<td>*****</td>
<td>***</td>
<td>*</td>
<td>****</td>
<td>****</td>
<td>**</td>
</tr>
<tr>
<td>DOM Scripting (with iFrames)</td>
<td>*****</td>
<td>***</td>
<td>***</td>
<td>****</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>Flash</td>
<td>***</td>
<td>**</td>
<td>***</td>
<td>***</td>
<td>****</td>
<td>****</td>
</tr>
<tr>
<td>Java Applets</td>
<td>**</td>
<td>*</td>
<td>****</td>
<td>**</td>
<td>****</td>
<td>****</td>
</tr>
<tr>
<td>AJAX</td>
<td>*****</td>
<td>**</td>
<td>*****</td>
<td>***</td>
<td>**</td>
<td>****</td>
</tr>
<tr>
<td>Desktop Applications</td>
<td>*</td>
<td>****</td>
<td>****</td>
<td>*</td>
<td>*</td>
<td>*****</td>
</tr>
</tbody>
</table>

More asterisks are better!
How Ajax is Similar

- Installation
  - Plug-in or installation required (*=full installation)
- Speedy Start
  - How quickly users can start to interact (*=long wait)
- Granularity
  - The smallest unit of data updates (*=page)
- Portability
  - Portability across platforms (*=none)
- Maintenance
  - Difficulty of making and distributing changes (*=very)
- Sophistication
  - Richness of the user experience (*=limited)

Ajax User Experience

- User experience of Ajax can be better than the traditional page-at-a-time model (but not necessarily)
- We need to consider some Computer-Human Interaction (CHI) issues first:
  - Change blindness
  - Attentional gambling
  - Flow
  - Feedback
CHI – Change Blindness

Change blindness means that we cannot easily see the differences between two images when they are separated by a blank field.

(Diagram courtesy of Ron Rensink)

Can you spot the difference in the two scenes?
Only 44% of observers noticed the gorilla while counting the number of ball passes between teams. The gorilla was on screen for 5 seconds! The paper (called “Gorillas in our midst”) won an IgNoble prize in 2004. See http://www.wjh.harvard.edu/~cfc/Simons1999.pdf
CHI – Change Blindness

CHI – Attentional Gambling

- Having performed an action, users must gamble on where to look next
CHI – Attentional Gambling

- Usually, having clicked in a left-navigation panel, users will focus their attention on the content area.

Users gamble on looking at A but lose since B is where they needed to look.

CHI – Visual Perception Summary

- Change blindness – changes will not be noticed, especially if...
  - A page reload is involved
  - Users are distracted
- Attentional gambling – users will not attend to content or navigation...
  - If it isn’t where they expected it to be
  - If they are distracted
- Appropriate use of popups and animation can assist both problems but beware of
  - Distracting users and
  - Flow…
CHI – Flow

- Mihaly Csikszentmihalyi’s theory of flow describes optimal psychological experience
  - Living in the moment
  - Lack of distraction (for CHI a “transparent interface”)
  - Appropriate pacing of activity
  - Appropriate degree of difficulty

- Coupled with flow, we need to make sure that users feel
  - Their effort is rewarded
  - They are getting closer to their goal (information foraging theory)

CHI – The Flow Curve

Adapted from Kathy Sierra’s Featuritis Curve
CHI – Feedback

- Feedback needs to be close to its cause in three aspects:
  - Time
  - Space
  - Meaning
- The time and space aspects are also related to attentional gambling (when and where users are expecting to look)

Activity 2

- Agree/elect/appoint a different spokesperson/note taker
- Discuss the issues raised on the worksheets and supplied handouts
- At the end of the discussion the spokesperson will give a verbal summary of the discussions (3 – 5 minutes) to the whole group
Activity 2

- Worksheet issues: Discuss the good and bad points of Ajax relative to
  1. Change blindness
  2. Attentional gambling
  3. Flow
  4. Feedback

(See the separate handout for a reminder of these topics)

Ajax and Change Blindness

- Biggest contributor to change blindness is reloading the page
- Ajax can fix this, but there is also:
  - Attentional gambling (are users looking in the right place?) and
  - “splash blindness” to contend with…
“Splash Blindness”

See “Change blindness as a result of mudsplashes” in Nature, March 1999

Ajax and Attentional Gambling

- Updating pages without reloading is only effective if
  - users are looking in the right place or
  - the change attracts attention by being large, colourful or animated
Ajax and Attentional Gambling

When nothing appears to happen, users focus on the progress bar – the update at the top of the page would be invisible, even with Ajax.

Order line shows popup detail when moused-over, but users still need to notice that it’s there.
Ajax and Flow

Of course, Ajax (or just DOM scripting) would let you do something like this:

But flow requires an optimum amount of feedback:

More Feedback ≠ Better
Ajax and Feedback

- For server-based validation Ajax can provide much more timely feedback than the page-at-a-time model
- However, timely client-side validation requires only DOM scripting

Confusingly, this whole page was reloaded when the delivery method changed, just so the delivery cost could be updated
Design and Usability Guidelines

1. Do not focus on the technology: Use Ajax to address usability and user experience problems with the page-at-a-time model.

Ajax used to “spice up” the user experience

On the same page changing quantity has no effect

Priority should have been given to fixing the serious usability problem caused by requiring a page update – instead a popup was added
Design and Usability Guidelines

Updating an expired credit cards takes forever on most web sites – on Amazon.co.uk it takes two clicks but with no Ajax in sight

2. Do not be over-optimistic about update speeds. The recommendation for thin-client computing is a minimum of 100 MB/S (about 100 times better than many broadband connections)
   - Consider Ajax only for moderate volumes of code and data
Design and Usability Guidelines

3. Help users to gamble with their attention effectively – make updates:
   - visually obvious or
   - where users’ attention will be focussed
Design and Usability Guidelines

4. Do not cause more problems than you solve with Ajax:
   - Make sure the browser back button still works
   - Deal with delayed or missing server responses by implementing timers
   - Optimize flow by avoiding excessive feedback, especially message boxes that have to be dismissed by users
Design and Usability Guidelines

5. Remember that most users do not really care about the technology

Microsoft’s Outlook Web Access is a spectacular example of Ajax technology but most ordinary users wonder why it isn’t more like the desktop version