Wookies and Widgets

Scott Wilson (University of Bolton)

Scott.bradley.wilson@gmail.com

Twitter: scottbw

University
of Bolton

Widgets?

Mini portable applications, generally designed to work in a small view area rather than full-screen

Examples include games, clocks, feed displays, etc

Lots of competing models for this: iPhone SDK, Android SDK, Konfabulator, Apple Dashboard, Opera Widgets, Nokia Widgets, Google OpenSocial etc

University of Bolton

W3C Widget Specs

- Open standard for widgets

- You create an HTML file, add as many CSS, JavaScript and image files as you need, add a "config.xml" file with some basic info, zip the lot, change the extension to .wgt and it works. Just like that.

University
of Bolton

Widget Specs & Status

- Packaging and Configuration (CR)
  - 100% conformance
- The Widget Interface (CR)
  - 90% conformance
- WARP (Draft)
- Widget URI scheme (Draft)
- Widget Signatures (Draft)

University
of Bolton

W3C DAP & BONDI

- Camera
- Contacts
- Device status and settings
- File system
- Media gallery

Widget development

- More like mobile app development that traditional web development
- W3C Mobile Web Best Practices a useful guide

University
of Bolton

Widget market

- Right now W3C Widgets is at an early stage of adoption
- However it can be seen that within a short space of time, W3C widgets may have the widest global customer reach of any "app" technology
- So its worth investing in!

University
of Bolton

QuickTime™ and a
decompressor
are needed to see this picture.

italic *"Wookie" is not a clever acronym. so if you spell it WOOKIE you're shouting!)*

University
of Bolton

Apache Wookie (incubating)

- Entered incubator July '09
- originally developed in TenCompetence (Framework 6 IP)
  - Funded projects tend to build prototypes, not communities
  - Apache Incubator is to build community around code
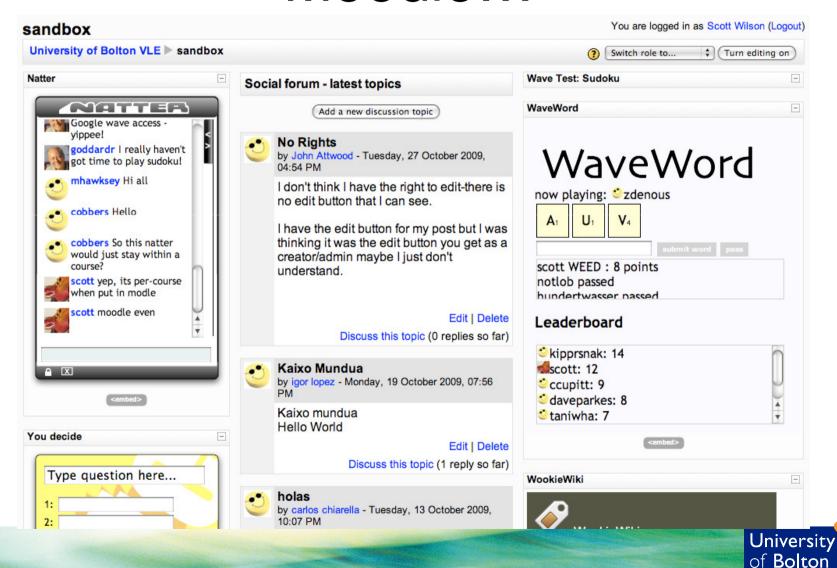- http://incubator.apache.org/wookie/

Wookie is...

- A Widget runtime for websites
- Implements
  - W3C Widgets P&C
  - W3C Widgets Interface
  - Wave Gadget API
  - (any API you want...)
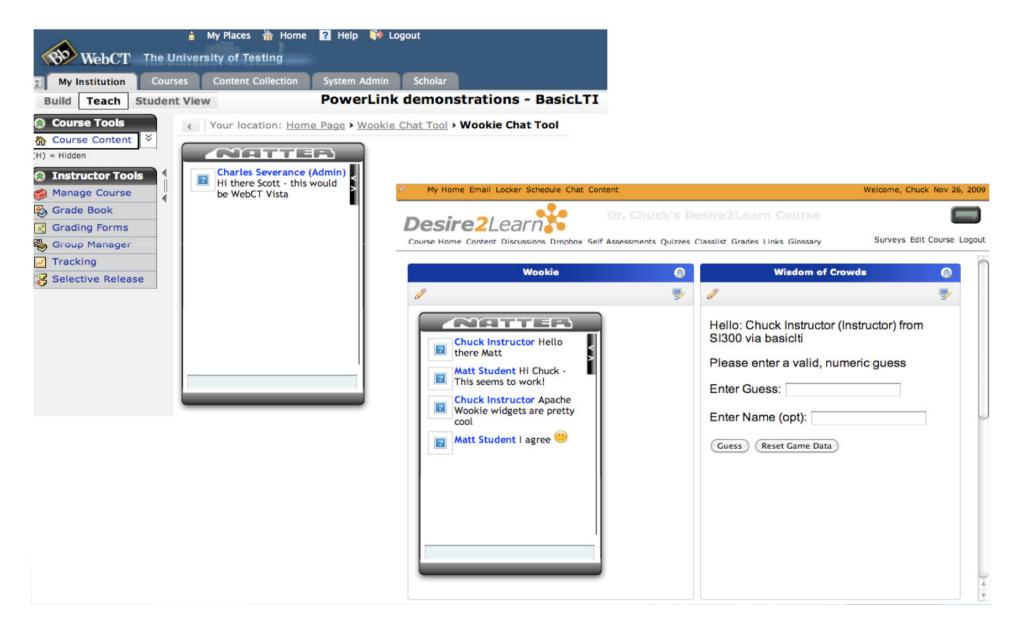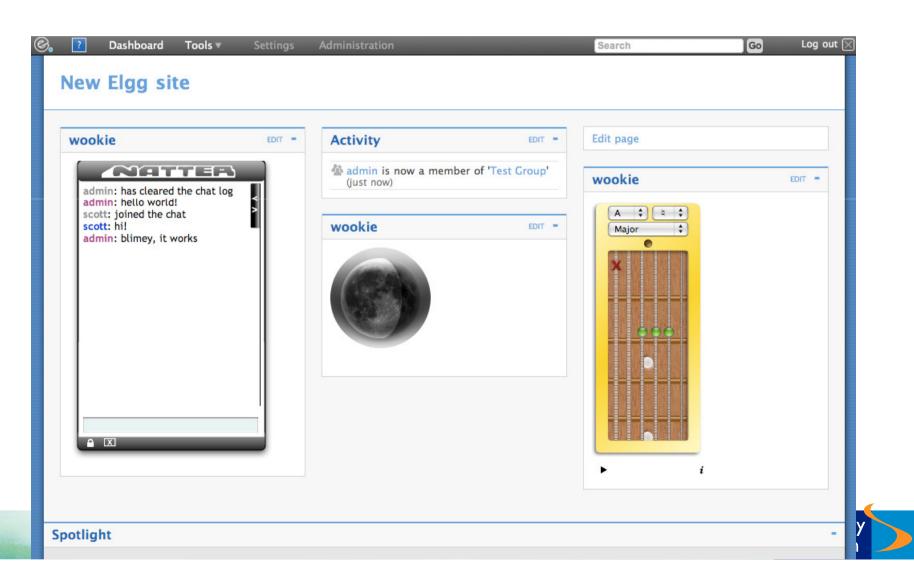- Integrates into other apps via plugins

Moodle…

LAMS...

QuickTime™ and a
decompressor
are needed to see this picture.

WebCT, Blackboard, D2L, Sakai…

Elgg… etc

How Wookie Works

- Servlet, runs in Tomcat or Jetty
- Provides REST API for applications to access widgets using a plug-in
- Provides JavaScript API objects for widgets at runtime
- Admin UI for installing and managing W3C Widgets
- Server-side storage
- State and Participant push events using Comet (implemented with DWR)

University
of Bolton

Wookie REST API

- GET /widgets *gets the list of widgets installed*
- POST /widgetinstances *instantiates a widget*
- POST /participants *adds a participant to a widget instance*
- Lots more at
  http://incubator.apache.org/wookie/wookie-rest-api.html

Typical lifecycle

- GET /widgets -> render as a gallery for the user to pick which one to add
- POST /widgetinstances -> create an instance of the widget for the user, returns the URI and widget height/width
- POST /participants -> add user as a participant in the instance
- Create iframe with src, height and width set

Widget runtime APIs

```
<feature name="http://wave.google.com"
  required="true"/>
```

- W3C Widget Object : preferences, metadata
- Google Wave Gadget API : state, participants
- *Bondi camera API, W3C DAP*
- *SCORM CMI API*
- *… anything else!*

University
of Bolton

Things developers can do with Wookie

- Create **widgets** Wookie can serve up
- Create **plugins** to connect Wookie with other platforms
- Create **features** that add more runtime capabilities
- Contribute to improving the **server code itself**

University
of Bolton

A few widget best practices

Build widgets to W3C, not for Wookie alone

- – Don't rely on special internal Wookie features (e.g. DWR) if it can be avoided
- – Try to provide fallbacks for installed features which may not be available for other widget containers (e.g. Wave)

University
of Bolton

Consider mobile users too

W3C Widgets can also run in MS Windows Mobile, Opera, Aplix and other platforms - if you design them with mobile use in mind

- **Don't** navigate or use popups - always use widget.openUrl()
  - Will use device native best practice and user prefs (e.g. switch to mobile browser app, open new tab, prompt user etc)
- **Don't** rely on cookies - use widget.preferences
  - User may access from different devices; use preferences instead to store user personalisation or tracking info.
- **Do** keep your widgets as small as possible
- **Do** follow the W3C mobile web guidelines where it makes sense

University of Bolton

Security and privacy

- While you **can** store user credentials in widget.preferences, this is not necessarily a very secure place to put them

- In future oAuth may be added to Widgets and Wookie, which will make life a lot easier

- Widgets are bound by same-origin policy; until CORS is implemented, you need to use the server-side proxy offered by wookie for AJAX calls

University of Bolton