asynchronous calls & ajax

Daniel Jackson
basic HTTP

when you enter a URL or click on a link
› browser sends HTTP request to server
› server responds with big string (a page) containing more links

HTTP verbs
› request types: GET, POST (and some others)
› both can pass key/value map to server
› GET can take query string in URL
   http://books.com/search?author="Dickens"
› POST can take form in body of request
a blocking protocol

HTTP is “blocking” or “synchronous”
› browser freezes while waiting for response from server

what’s wrong with this?
› small changes still need new page
› excessive bandwidth
› lack of responsiveness
› can’t provide dynamic feel
Ajax

an extension to HTTP
› XMLHttpRequest
› Microsoft 1999

this request is “asynchronous”
› browser doesn’t freeze
› can issue other requests before response
what is Ajax?

“asynchronous JavaScript and XML”
› instead of HTML, sends XML in both directions
› libraries let you send and receive data in JSON

now a dominant idiom
› a natural extension of client-side events
› think of server call as registering a handler
› “when server responds, call f”

what’s nice
› good for users: dynamic & responsive
› often natural for programmers
› good support in libraries & frameworks

what’s not
› tricky if you care about ordering
what you can do with Ajax

push updates to page
› status, stock quotes, weather, time

interact with user over input
› drill downs (eg, state>city), autocomplete

persist user data on server
› Google docs, stickies

execute server-side commands on same page
› vote up/down on posts, email portal
event model

- single thread
- event loop: run timed event if expired or event at front of Q
- events added by: user actions, server responses, events
particulars

two key facts
› each event runs to completion before next event
› response to Ajax request is *not* always next

good news
› no need to worry about mutual exclusion

bad news
› long-running event freezes the UI
› timeouts may run late
› when AJAX response comes, context may have changed
JQuery’s Ajax API

$.getScript(url, callback)
› get script at url, run it, then execute callback

e.load(url)
› get content at url, and insert into element e

$.getJSON(url, data, callback)
› pass data to server at url
› server sends JSON back; parse and pass to callback

$.get(url, data, callback, type)
› type determines preprocessing of response
› $.post similar, but does HTTP post instead

$.ajax(…)
› most flexible, lowest level method
using network inspector

- example from Safari
using network inspector

› can see here that request was get
encoding data for transit

XML
› parsing built into browser (XHR)
› comes back as DOM: not convenient

JSON
› Javascript object literals
› JQuery uses parser, not eval (why?)

examples from http://en.wikipedia.org/wiki/JSON