CS1120 Guest Lecture: Web applications

Kinga Dobolyi

November 10, 2009

Last lecture

- Lazy evaluation
  - Don't need to evaluate arguments until they're needed

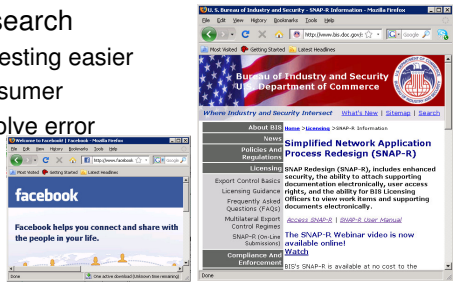
This lecture

- Introduction to web applications
  - Why they are interesting
  - History
  - Developing a web application
    - Challenges
    - Testing
  - Errors in web applications

Why I care about web applications

- Built them professionally in 2004
- Topic of my dissertation research
  - How to make testing easier
  - How does consumer perception involve error severity
- Ubiquitous

Severity Level	Description
0	The fault was not noticeable by customers/users on the website/application
1	Customers/users would return this website/application again
2	Customers/users probably would return this website/application again
3	Customers/users would not return to this website/application again
4	Customers/users would complain about this website/application

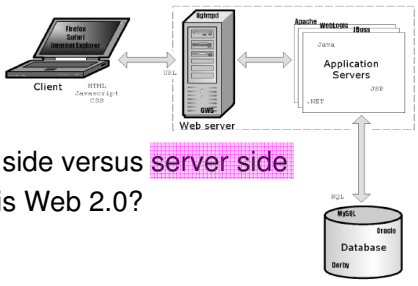


Why you care about web applications

- Upcoming project
- You may want to build them in the future
  - A way to reach billions of people
- They combine many interesting components and functionality
  - Databases
  - Session state
  - Graphical user interfaces
  - Security
  - E-commerce

What is a web application?

- Web applications started as static pages
- Client-server model delivers static and/or dynamic pages

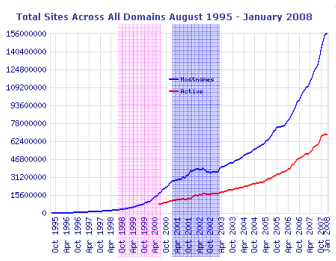


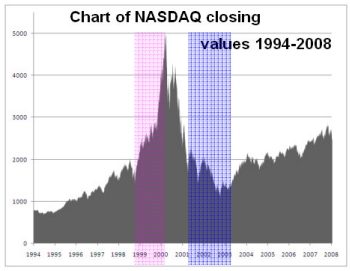
  - Client side versus server side
  - What is Web 2.0?

Web applications are everywhere

- 73% of the people in the US used the Internet in 2008
- Internet-based transaction orders total several trillions of dollars annually
- Information sent in HTML; accessible by any browser

Growth of web applications





Websites are in a golden age

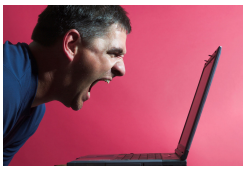
- increasing exponentially every year

Growth of web applications

Position Oct 2009	Position Oct 2008	Delta in Position	Programming Language	Ratings Oct 2009	Delta Oct 2008	Status
1	1	=	Java	18.718%	-2.23%	A
2	2	=	C	16.891%	+1.33%	A
3	5	↑↑	PHP	10.390%	+1.78%	A
4	3	↓	C++	9.911%	-1.04%	A
5	4	↓	(Visual) Basic	8.729%	-1.08%	A
6	8	↑↑	C#	4.433%	+0.67%	A
7	6	↓	Python	3.914%	-0.65%	A
8	7	↓	Perl	3.776%	-0.64%	A
9	11	↑↑	JavaScript	3.033%	+0.36%	A
10	10	=	Ruby	2.458%	-0.40%	A
11	9	↓↓	Delphi	2.140%	-1.15%	A
12	13	↑	PL/SQL	1.020%	0.00%	A
13	49	↑↑↑↑↑↑↑↑↑	Objective-C	0.902%	+0.82%	A--
14	14	=	SAS	0.805%	+0.21%	A
15	16	↑	Pascal	0.669%	+0.15%	A-
16	20	↑↑↑↑	ABAP	0.661%	+0.22%	A-
17	19	↑↑	Lisp/Scheme	0.605%	+0.12%	B
18	22	↑↑↑↑	MATLAB	0.577%	+0.17%	B
19	12	↓↓↓↓↓↓↓	D	0.570%	-0.76%	B
20	15	↓↓↓↓↓	Lua	0.527%	-0.02%	B

Growth of web applications

- Increasing exponentially
- Dynamic environment:
  - have short delivery times, high developer turnover rates, and rapidly-evolving user needs that translate into an enormous pressure to change
  - Developed informally

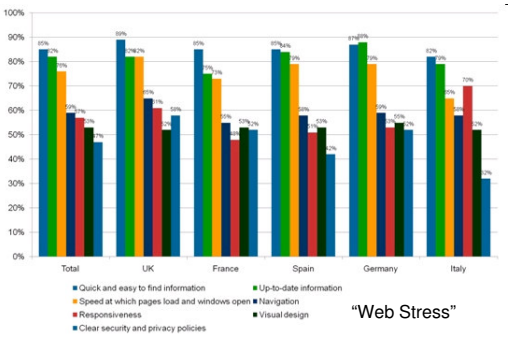


What happens during informal development?

- User-visible failures are endemic to top-performing web applications:
  - about 70% of such sites are subject to user-visible failures
  - one hour of downtime at Amazon.com has been estimated to cost the company $1.5 million dollars
  - Customer loyalty is low

How can we make web applications more reliable?

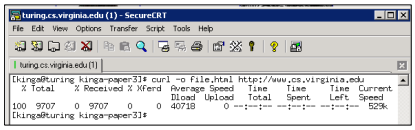
- Design with care
- Testing!



"Web Stress"

Problems with testing web applications

- Perceived return on investment is low
- Testing can be resource intensive
  - Example: one of my benchmarks took 50 hours to complete
  - Runtime = (# user sessions) * [database clean + (#urls)*(`curl`)]
    - Average 30 urls per session
    - `curl` takes 5 seconds on average per url
    - Database clean takes 1 minute
    - Run entire test rig more than once a day



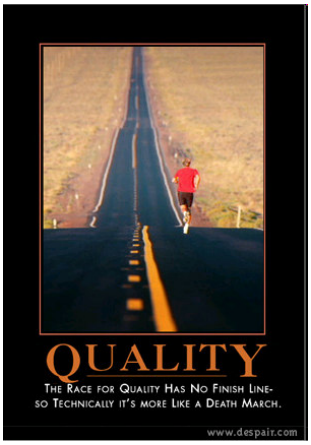
Problems with testing web applications

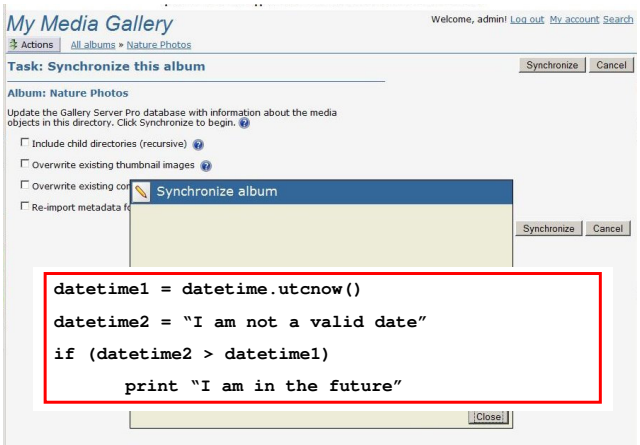
- Even this very small test suite takes almost 3 hours to run!
- How do you reduce the runtime in this case?
  - Reduce the number of user sessions? Urls? Time to refresh database? Time for `curl`?
  - Can't change database or `curl` runtime

Problems with testing web applications

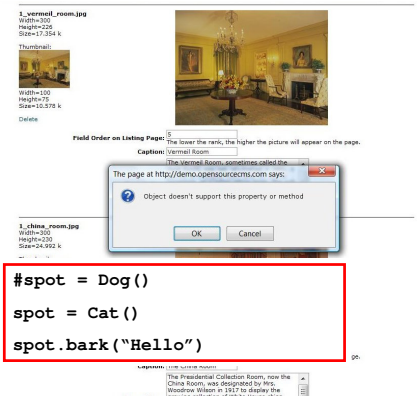
- Since we can't change the algorithm, and can't change the database access time or `curl`, what do we do?
- We can reduce our runtime by changing what we expect to get from testing
  - Can't test everything
  - Can you think of ways we can have some lower bound on what we do know?
    - That is, run fewer tests but still be useful?

Errors in Web applications

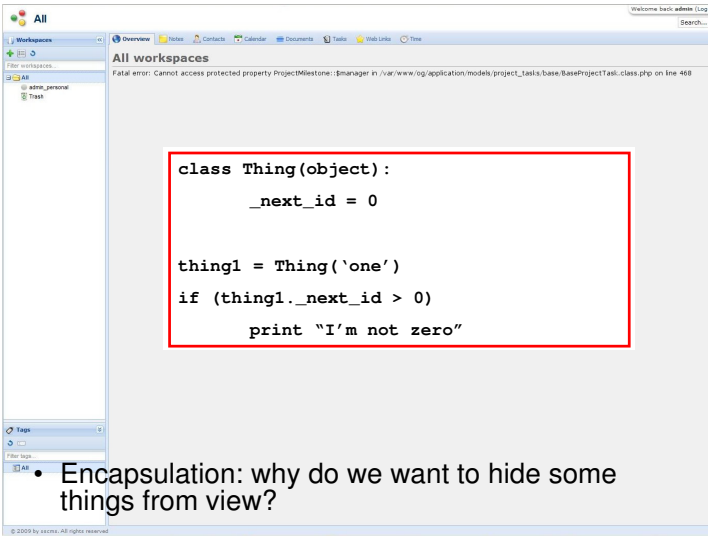




- Inheritance: String is not a subclass of DateTime

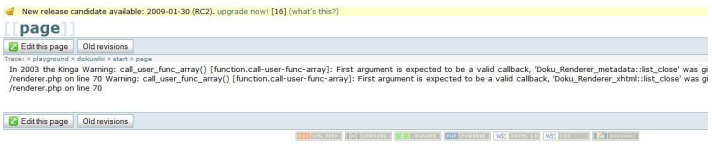


- Objects: an object packages state and procedures



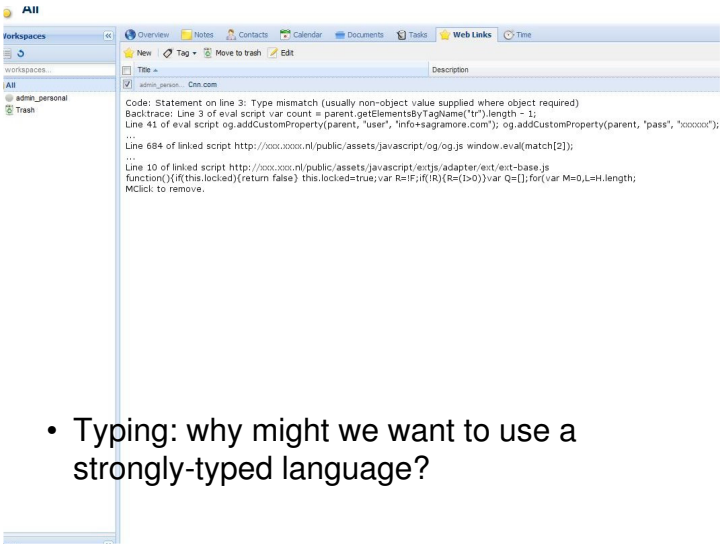
```
class Thing(object):
        _next_id = 0

thing1 = Thing('one')
if (thing1._next_id > 0)
        print "I'm not zero"
```

- Encapsulation: why do we want to hide some things from view?



```
# statement-based loop
for e in lst:
        func(e)
# passing func as an argument
map(func,lst)
```

- Functional programming: can pass a function as an argument



- Typing: why might we want to use a strongly-typed language?

- Security: want to avoid passing string arguments that may be interpreted as code and run on the host machine
  - Example: a line of code that deletes a table in the database

Summary

- Web applications
  - Are growing exponentially and used widely
  - Combine interesting technologies
  - Need to be tested like all software

Information

- The Django tutorial is posted on the course site now
  - Go through this by Sunday
- Professor Evans won't have normally scheduled office hours today
  - normal office hours Tuesday morning (10:30-11:30am)