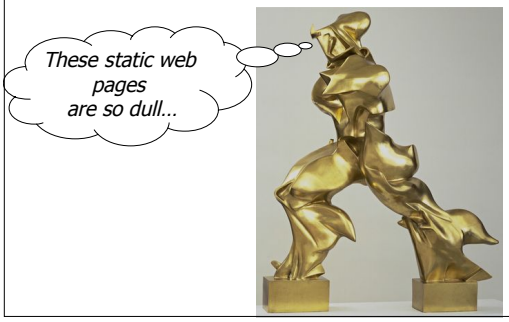
Lecture 2
Dynamism!



HTML

- HyperText Markup Language
- Invented by Tim Berners-Lee in 1990
- Set of tags for rendering page

```
<html>
<body>
The content of the body element
  is displayed in your browser.
</body>
</html>
```

- Tags form a hierarchy - important later

2

Dynamic Content

- So far, HTTP servers are fileservers
  - And browsers are HTML renderers
- Think of the things that are impossible with simple static pages
  - Web search
  - Database lookups
  - Current time
  - # visitors to page
  - Everything

3

Part I

- The Server

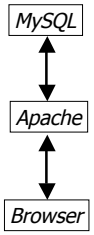
4

Dynamic Server Content

- In the old days (1997?), almost all requests were just disk loads
- Computing the page dynamically was a ***mind-blowing idea;*** today it's assumed
  - Server-Side Includes (SSI) - directives interpreted by the web server itself
  - Common Gateway Interface (CGI) - code executed as a separate process
  - Scripting Languages - PHP, ASP, JSP, Ruby
  - Application Servers - J2EE, .NET, Tomcat

5

2 (or 3)-Tier model



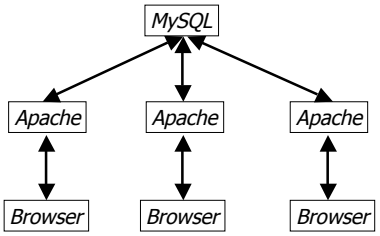
6

N-tier model

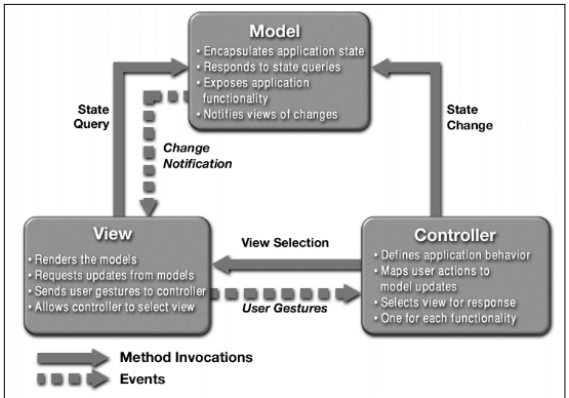
- We've separated persistent storage from user interaction
- Other separations?
  - Manageability (what if db goes down?)
  - Security & Mining (logging all user steps)
  - Efficiency (a single dataset in memory?)
- Web apps often break down pieces of code functionality into machines

7

In Real Life…

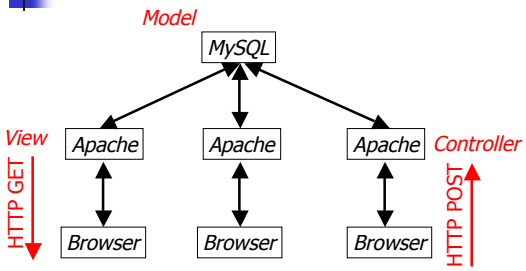


8



9

MVC in deployment



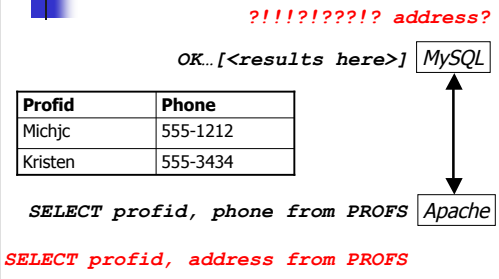
10

More MVC

- What if…
  - Data stored distributed geographically?
  - Want to take advantage of Wiimote?
  - Browser over phone?

11

Deployed MVC



12

Object-Relational Mapping

- Used by Rails (around for a long time)
  - Ruby is the language
  - Rails is the framework, built just for Ruby lovers
- Addresses the "impedance mismatch" between relational and object worlds
  - Map object classes to database tables
    - Table=>class
    - Column=>attribute
    - Row=>object instance

13

Dynamic Server Recap

- Many techniques for mixing content and code, but it's probably PHP
- Dynamic pages rendered to user at query-time
- MVC popular tool for database design

14

Part II

- The Client

15

Dynamic Client, too

- All rendered pages used to be static
- Plain HTML not as bad as statically-generated pages, but certainly limiting
  - No in-browser chat
  - No browser-based field validation
  - No grabbable maps
  - No deferred data loading in Gmail
  - No dumb hit-the-monkey ads

16

Dynamic Client Content

- In early days, many engineers obsessed with "rich browser client"
  - No good client-side language, clients very slow
  - Browser experience very different from desktop
- That all changed with:
  - Adobe Flash
  - ActiveX
  - Java applets
  - JavaScript
  - The <blink> tag
- Actually, all of these *except* blink
- Today, heavily AJAX (Asynchronous JavaScript and XML)

17

Worst Idea In This Class

- JavaScript Invented by Netscape, called LiveScript
- Java was hot, also starting to be built into browsers, *so Netscape renamed LiveScript to JavaScript*
  - They have nothing to do with each other
  - Except they are both pieces of mid-90s browser tech endorsed by Netscape
  - Also, they have extremely similar names
  - Until 2003 or so, very confusing

18

JavaScript Execution

- Usually, but not always, inside browser
  - Sometimes at HTTP server, instead of PHP!
- JS code runs when triggered by `<script>`
- Has read/write access to the page's DOM, or Document Object Model
  - The DOM defines HTML of onscreen page
  - Changing the DOM changes what's on screen
  - JS doesn't have to hang the page while executing (hence, the "asynchronous" part)

19

JS DOM Example

```
<html>
<body>

<script type="text/javascript">
document.write("Hello World!");
</script>

</body>
</html>
```

20

JS DOM Example 2

```
<html>
<body>

<script type="text/javascript">
document.write("<h1>Hello
  World!</h1>");
</script>

</body>
</html>
```

21

Browser Security

- JS is potential security nightmare!
  - Sandboxed (no disk, syscalls, infinite loops)
  - Same-origin policy (defined by domain, app protocol, and port)
  - Very similar to Java applets

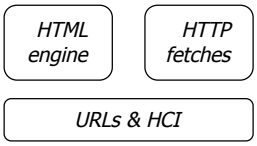
22

Attacks

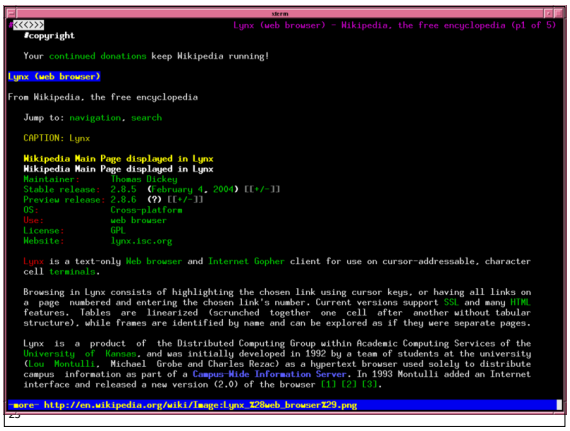
- Cross-site scripting
  - User visits `aaa.com`
  - Site `aaa.com` computes nefarious URL, convinces user to click on it
  - URL is something like:
    - http://bbb.com/hello?param1=*!*!@,<h1>Your-account-is-empty-sucker</h1>
  - Site bbb.com, with modification, is displayed to user, with inserted-HTML. Could have stolen info
  - Many have been found against Google
- This is called "tag injection"
  - What is "SQL injection"?
- What if the URL is on someone's Facebook wall?

23

Browser Internals

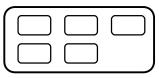


24



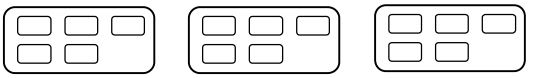
25

Browser Internals



26

Browser Internals



http://cia.gov *http://facebook.com* *http://webmd.com*

- *How many threads?*
- *How much memory separation?*
- *What if there are >30 apps?*

27

Chrome

- Introduced by Google last year
- Introduces process separation to browser architecture
  - One app dies, others survive

28

Grand Unified GET

- What are all the moving parts involved in GETting a Facebook home page?
- <draw on board>

29