# AnyEvent Asynchronous Programming in Perl

# HELLO MY NAME IS Josh Barratt



### LINKFARM

http://github.com/jbarratt
http://serialized.net
@jbarratt

What is "Event Driven Programming"?

#### Think GUIs

Sit and wait for something to happen

"Main Loop" -- nothing else to do

# Synchronous Code

One Server (process) per Client

Think apache

### Event-driven code

One Server, many clients

Think nginx

#### Under the hood

epoll, kqueue

Kernel tracks the "clients", not our process

O(1) vs O(n), callbacks instead of lists

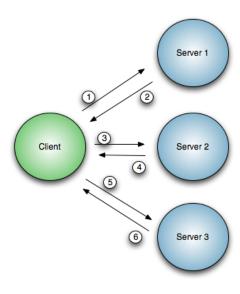
#### When to use it?

I/O Wait >> "Think Time"

Awesome for network, I/O heavy work

# Basic Sync Operation

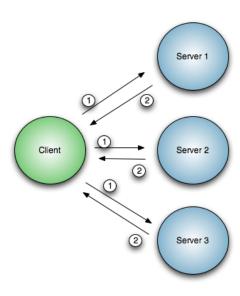
One server at a time



Sync Runtime
SUM(S1, S2, S3, ... SN)

# Basic Async

Ask All Servers
Process all answers



Async Runtime
MAX(S1, S2, S3 ... SN)

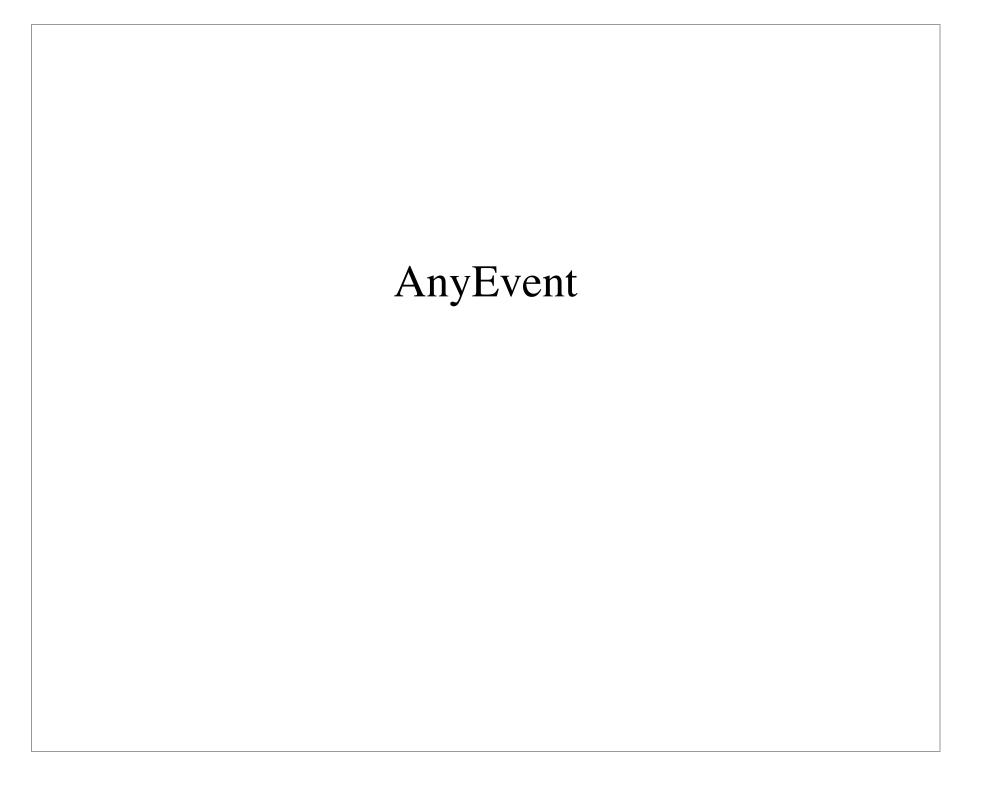
#### **TMTOWTDI**

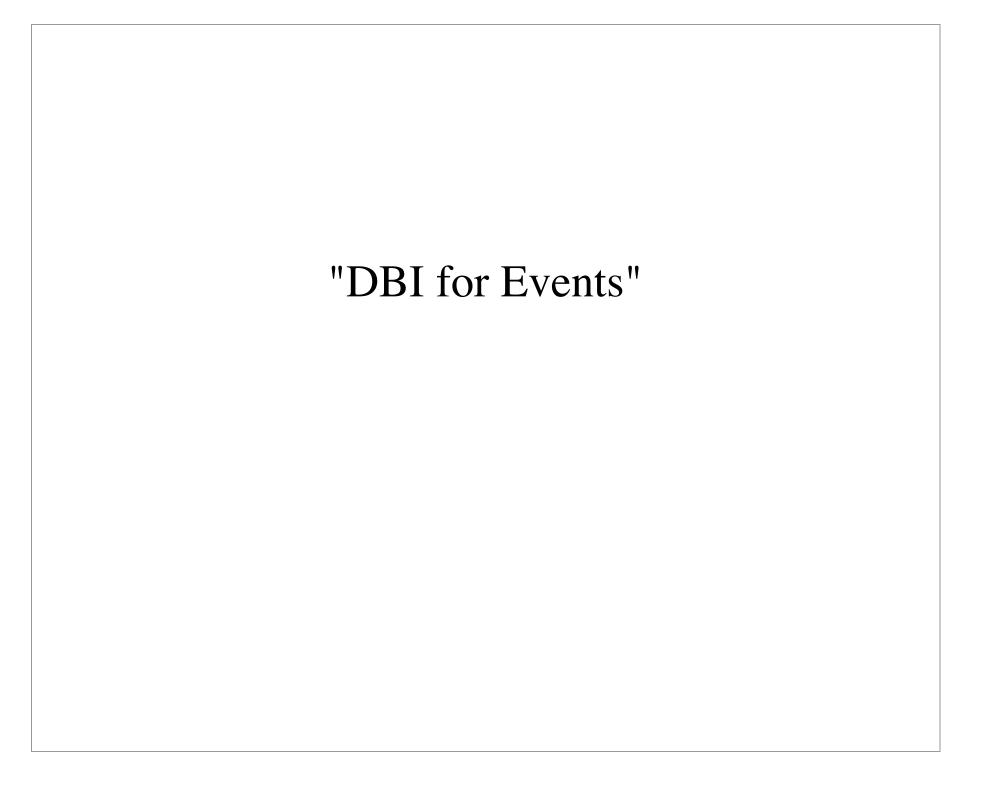
Other strategies for this workload

Threads (Coro)

Lots of others: c10k

(c10k techniques still work; c1M on modern hardware)





#### TMTOWTDI Redux

POE, IO::Async, Danga::Socket, IO::Lambda, Event, Event::Lib, EV, Glib, Qt, Tk

Remember the "main loop"?
You only get one.

# AnyEvent

Use any loaded event loop

No loop? Start an **EV** 

Very fast, very memory efficient

Perl wrapper on libev

One Module, 2 APIs	

# AnyEvent

The original

More Verbose

Uses methods vs functions

(nominally) Slower

```
$w = AnyEvent->timer (
    after => <fractional_seconds>,
    interval => <fractional_seconds>,
    cb => <callback>,
);
```

#### AE

Functions = static type checking

Less verbose

Faster (2x)

Incomplete (barely.)

# I will use AE today. (Fits on slides better.)

## **AnyEvent Basics**

Fancy Event Tracking system

i/o, timers, signals, processes (fork, etc)

API = how to create/manage these

## Concept: Condition Variables

"Condition that's initially false"

Merge Points, Sync Points

**Basic Messaging** 

AnyEvent->condvar or AE::cv

# ENUF TALK MOAR CODE

# Async DNS lookups

All of the 4 char domains are taken

How many of the 6 chars are?

#### Synchronous Code

#### AnyEvent Code

#### AnyEvent Code

#### Results

\$ ./01\_dns.pl
Synchronous code did 6.3983 lookups/second
Async code: 179.2940 lookups/second.
Overall, for 6 character domains, 0.50% resolved.

Simple Transactions with begin and end

# For a given name, which TLD's are resolving?

Want to do a bunch of DNS lookups

Transaction "complete" when all are done

begin @ callback creation

end when work is done

```
my @tld = qw(com net org mp ly cc co info
     biz mobi name pro);
my $cv = AE::cv;
for my $tld (@tld) {
    $cv->begin;
my $zone = "$name.$tld";
    AnyEvent::DNS::a $zone, sub {
    my $ip = shift || "NONE";
          Say "$zone => $ip";
          $cv->end;
$cv->wait;
Say "Transaction complete!";
```

```
my @tld = qw(com net org mp ly cc co info
      biz mobi name pro);
my $cv = AE::cv;
for my $tld (@tld) {
     $cv->begin; # outstanding++
my $zone = "$name.$tld"; # create cb
AnyEvent::DNS::a $zone, sub {
    my $ip = shift || "NONE";
            Say "$zone => $ip";
            $cv->end;
$cv->wait;
Say "Transaction complete!";
```

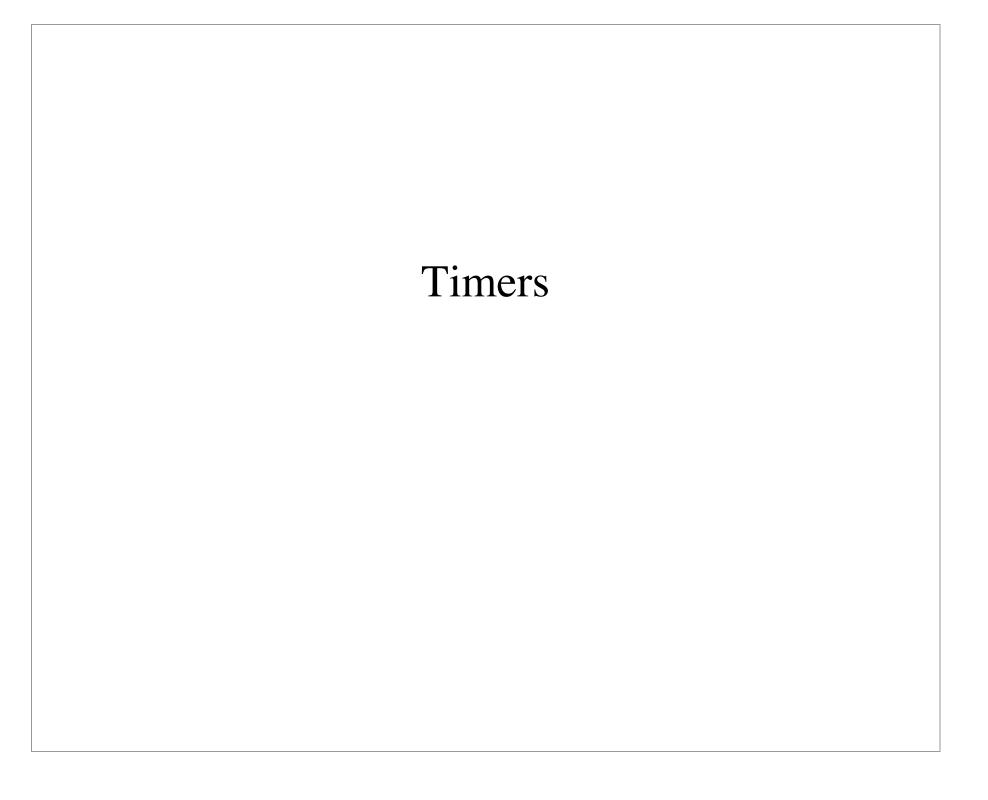
```
my @tld = qw(com net org mp ly cc co info
     biz mobi name pro);
my $cv = AE::cv;
for my $tld (@tld) {
     $cv->begin;
my $zone = "$name.$tld";
     AnyEvent::DNS::a $zone, sub {
    my $ip = shift || "NONE";
    say "$zone => $ip"; # in closure
           $cv->end; # outstanding--
     };
$cv->wait;
Say "Transaction complete!";
```

```
my @tld = qw(com net org mp ly cc co info
    biz mobi name pro);
my $cv = AE::cv;
for my $tld (@tld) {
    $cv->begin;
    my $zone = "$name.$tld";
    AnyEvent::DNS::a $zone, sub {
    my $ip = shift || "NONE";
         Say "$zone => $ip";
         $cv->end;
$cv->wait; # block until outstanding = 0
SQY "Transaction complete!";
```

\$ ./02\_transaction.pl anyevent
anyevent.mobi => NONE
anyevent.org => 207.46.222.11
anyevent.cc => NONE
anyevent.pro => NONE
anyevent.name => NONE
anyevent.co => NONE
anyevent.co => NONE
anyevent.com => 97.74.144.133
anyevent.mp => 199.34.127.242
anyevent.ly => NONE
anyevent.ly => NONE
anyevent.net => 193.93.174.26
anyevent.info => 213.171.192.98
anyevent.biz => 213.171.195.53
Transaction complete!

#### This works for any batch work

#### First Example Refactor



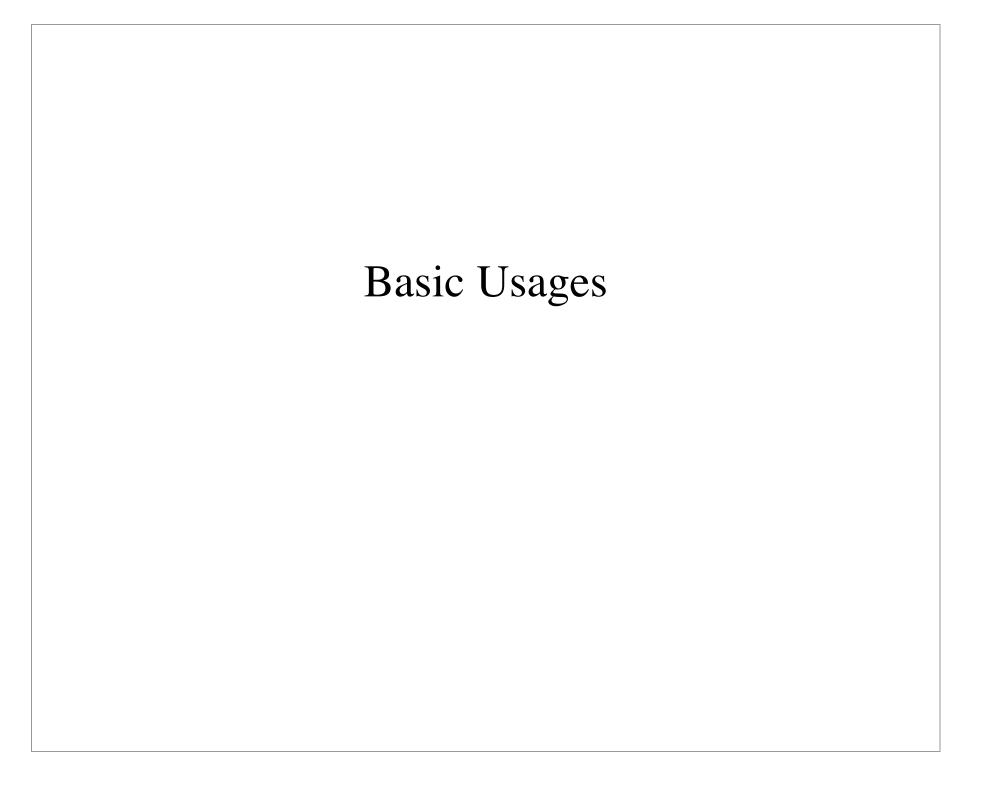
```
my $cv = AE::cv;
my $spoke = 0;
my $t1 = AE::timer 0, 2,
sub {
    say "2 second timer";
if($spoke++ > 10) { $cv->send; }
my $t2; $t2 = AE::timer 0, .5,
sub {
    Say "1/2 second timer";
    if($spoke++ > 5) { undef $t2; }
$cv->recv;
```

\$ ./04\_timer.pl
2 second timer
1/2 second timer
1/2 second timer
1/2 second timer
1/2 second timer
2 second timer

Why does the 1/2 second stop?

```
# Turn 1 statement into two
# my $t2 = AE::timer becomes...
my $t2; $t2 = AE::timer 0, .5,
sub {
    Say "1/2 second timer";
    if($spoke++ > 5) {
        # undef'ing AnyEvent objects
        # deletes the event
        undef $t2;
        # If the callback needs to stop it
        # Needs to be able to refer to it
        # declare in a different statement
```

Very common (but confusing) idiom



#### i/o

#### Timers

#### Signals and Condvars

```
# $w = AE::signal $signame, $cb
my $w = AE::signal
   TERM => sub { warn 'TERM received'; };
$cv = AE::cv;
# $cv = AE::cv { BLOCK }
```

#### Time and Idle

```
# Return the current time
say "At the tone, " . AE::time;

my $w = AE::idle {
          warn 'Event loop is idle';
};
```

#### Other Powertools

AnyEvent::DNS

AnyEvent::HTTP

AnyEvent::Handle

AnyEvent::Socket

```
# From AnyEvent::Socket
tcp_connect localhost => 22, sub {
   my $fh = shift
      or die "unable to connect: $!";
   # do something
};
```

# Common Async Gotchas

RAM Tradeoff

All those active conns need to live somewhere

How many reqs do you send out @ once?

"Quality Problem"

# Example: DNS lookups (again)

2 Letter domains are rare

Especially 2 letters + real word

Who has one but isn't using it?

#### What's wrong here?

#### RAM ASPLODE

\$ top PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND 6141 jbarratt 18 -2 668m 441m 796 D 4.4 88.5 0:14.58 05\_handle.pl 23098 jbarratt 18 -2 28588 1280 472 R 0.5 0.3 0:21.20 screen 5323 jbarratt 18 -2 29176 1256 1252 S 0.0 0.2 0:13.26 vi Reading Disk faster than DNS

Queue DNS lookups

Connections and Callbacks

Normal (Read, then lookup, then read) avoids

Solution: Manage Outstanding Work

```
my $t; $t = AE::timer 0, 1,
 sub {
     if($outstanding <= MAX OUTSTANDING/2) {</pre>
          Say "NEW Watcher on \$fh at byte " . tell($fh);
          my $w; $w = AE::io $fh, 0, sub {
              my $name = <\fin>; $name = 1c(\finame);
              if($outstanding' > MAX_OUTSTANDING)'{
                   undef $w;
                   Say "UNDEF watcher";
              \frac{1}{2} name =~ \frac{S}{[^a-z]}/g;
              return unless length($name) == 2;
              my $zone = "$name.com";
              $outstanding++;
              AnyEvent::DNS::a $zone, sub {
                   my $ip = shift;
                   $outstanding--;
                   if(!defined($ip)) {
                       $cv->send($zone);
};
};
```

\$ ./06\_throttle\_io.pl
NEW Watcher on \$fh at byte 0
UNDEF watcher
NEW Watcher on \$fh at byte 1794
... [SNIP]...
UNDEF watcher
NEW Watcher on \$fh at byte 111833
UNDEF watcher
NEW Watcher on \$fh at byte 125110
UNDEF watcher
First unresolvable .com in input: ds.com

#### Don't Reinvent the Wheel

Many Core modules have this "for free"

AnyEvent::DNS::resolver->max\_outstanding(...)

\$AnyEvent::HTTP::ACTIVE

# Simple HTTP example

How much of the "oldschool" jargon is still in play?

ESR's Jargon File

How many hits/word on google?

Queue control "for free" from AE::HTTP

```
http_get $jargon, sub {
   my ($body) = shift;
     for my $term (split(/\n/, $body)) {
          Say "searching for $term";
          $cv->begin;
          http_get "https://ajax.googleapis.com/ajax/" .
               "services/search/web?v=1.0&q=" .
                uri_escape($term),
              sub {
                   my ($body) = shift;
                   my $data = from_json($body);
                   my $results
                        = $data->{responseData}
                            [cursor]{estimatedResultCount};
                   if(defined($results)) {
    $terms{$term} = $results;
                        SQY "\t$term => $results";
                    $cv->end;
```

# Still going strong...

for free: 1,230,000,000

comment out: 879,000,000

download: 354,000,000

for values of: 343,000,000

by hand: 309,000,000

face time: 238,000,000

### **RIP**

VAXectomy: 35

cruftsmanship: 82

featurectomy: 84

Godzillagram: 85

pseudosuit: 103

CrApTeX: 126

# The larger ecosystem

If you fling packets at it, AnyEvent it!

CPAN FTW, lots already done

# Databases DBI

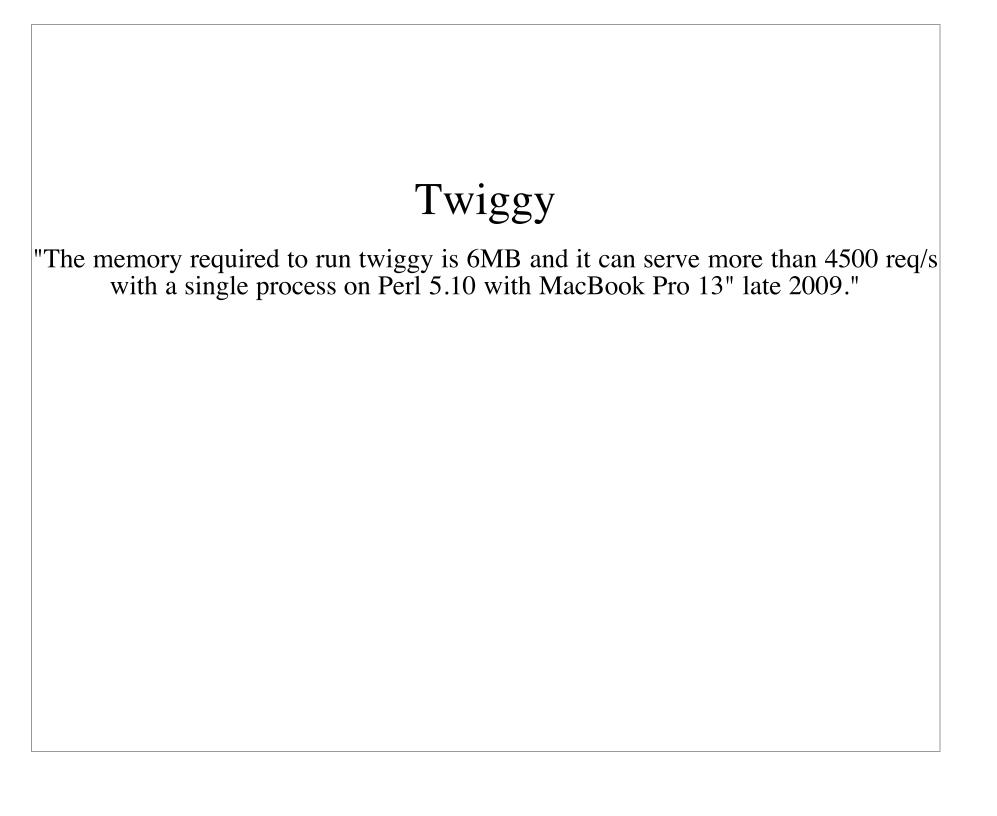
Redis CouchDB Riak Memcached MongoDB

(Mongo Meetup: <a href="http://j.mp/drX4Ri">http://j.mp/drX4Ri</a>)

# Messaging AnyMQ Stomp AMPQ RabbitMQ

# Networking FastPing IRC SNMP XMPP

# WebServers Tatsumaki Twiggy



## Async is perfect for webapps

Most of the time is waiting for input (FS, DB, Memcached)

Let your process be "working" (for lots of clients) instead of "waiting" (for one)

FREE THE RAMZ

Things To Keep In Mind when using AnyEvent	

# Never wait the old way A single sync wait will stall the event loop

### Protect your RAM

What can queue, in what conditions?

What callbacks are you creating and how long will they live?

What needs to be limited?

What can you undef (shut down) and when?

## Testing == HARD(er)

say and print

"integration" testing

Factor 'work' out of 'events'

Local state in modules

```
http_get google_search_uri($term),
    sub {
        my($body) = shift;
        handle_search_res($body);
        $cv->end;
};
```

