

A Guide to **STYLECI**

Who am I?

Graham Campbell

<https://github.com/GrahamCampbell>

<https://twitter.com/GrahamJCampbell>

<https://gjcampbell.co.uk/>

Who am I?

Graham Campbell

<https://github.com/GrahamCampbell>

<https://twitter.com/GrahamJCampbell>

<https://gjcampbell.co.uk/>

Current:

- Student at the **University of York**
 - BSc Computer Science and Mathematics

Disclaimer: The views and opinions expressed are my own.

StyleCI and Cachet are not associated with the University of York, Cambridge Consultants, or Bugsnag.

StyleCI and Cachet are the intellectual property of Alt Three Services Limited.

Who am I?

Graham Campbell

<https://github.com/GrahamCampbell>

<https://twitter.com/GrahamJCampbell>

<https://gjcampbell.co.uk/>

Current:

- Student at the **University of York**
 - BSc Computer Science and Mathematics
- Placement at **Cambridge Consultants**
 - Software Technologies, ICE

Disclaimer: The views and opinions expressed are my own.

StyleCI and Cachet are not associated with the University of York, Cambridge Consultants, or Bugsnag.

StyleCI and Cachet are the intellectual property of Alt Three Services Limited.

Who am I?

Graham Campbell

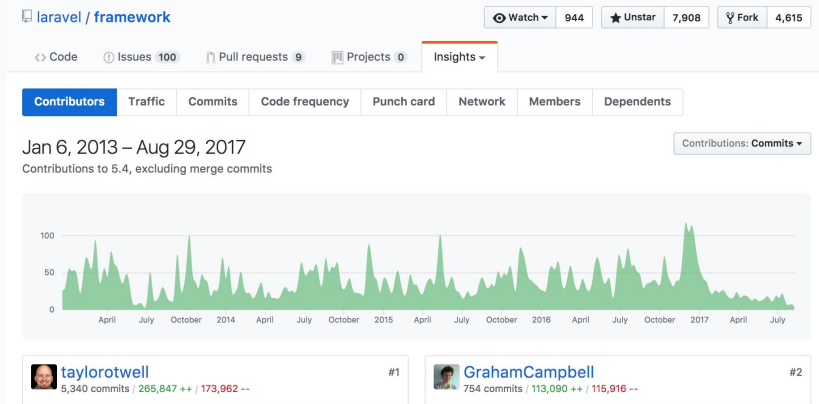
<https://github.com/GrahamCampbell>

<https://twitter.com/GrahamJCampbell>

<https://gjcampbell.co.uk/>

Current:

- Student at the **University of York**
 - BSc Computer Science and Mathematics
- Placement at **Cambridge Consultants**
 - Software Technologies, ICE
- **Laravel Framework** Core Team
 - Contributing since August 2013



Who am I?

Graham Campbell

<https://github.com/GrahamCampbell>

<https://twitter.com/GrahamJCampbell>

<https://gjcampbell.co.uk/>

Current:

- Student at the **University of York**
 - BSc Computer Science and Mathematics
- Placement at **Cambridge Consultants**
 - Software Technologies, ICE
- **Laravel Framework** Core Team
 - Contributing since August 2013
- **StyleCI** Founder and **Cachet** Core



Disclaimer: The views and opinions expressed are my own.

StyleCI and Cachet are not associated with the University of York, Cambridge Consultants, or Bugsnag.

StyleCI and Cachet are the intellectual property of Alt Three Services Limited.

Who am I?

Graham Campbell

<https://github.com/GrahamCampbell>

<https://twitter.com/GrahamJCampbell>

<https://gjcampbell.co.uk/>

Current:

- Student at the **University of York**
 - BSc Computer Science and Mathematics
- Placement at **Cambridge Consultants**
 - Software Technologies, ICE
- **Laravel Framework** Core Team
 - Contributing since August 2013
- **StyleCI** Founder and **Cachet** Core

Previous:

- **Bugsnag** UK Office
 - Primarily worked on their PHP notifiers

Disclaimer: The views and opinions expressed are my own.

StyleCI and Cachet are not associated with the University of York, Cambridge Consultants, or Bugsnag.

StyleCI and Cachet are the intellectual property of Alt Three Services Limited.

The Problem

Standardizing code style across your workplace or OSS can be hard.

The Solution

Standardizing code style across your workplace or OSS can be hard.

December 2014


StyleCI created for my
personal OSS projects.

The Solution

- **Easy to get started.**
 - One-click enable from StyleCI.
 - Configure from the browser or a *.styleci.yml*.

The Solution

- **Easy to get started.**
 - One-click enable from StyleCI.
 - Configure from the browser or a *.styleci.yml*.
- **Hours are saved!**
 - No more code style reviews as part of code reviews.
 - StyleCI can send fixes as PR on demand.
 - StyleCI can be configured to do this automatically!
 - StyleCI can be configured to automatically merge its own PRs.
 - StyleCI can also be configured to directly commit the fixes!

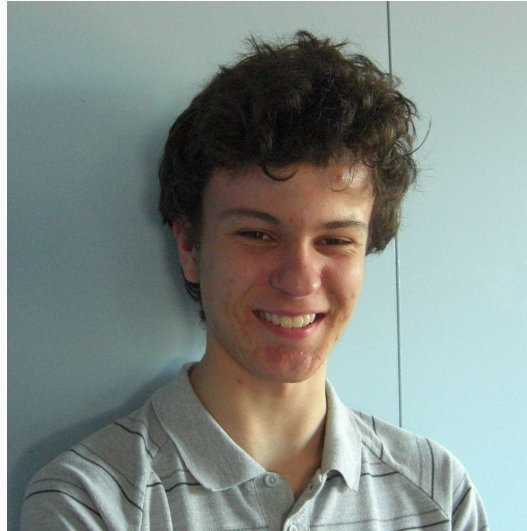
A laptop screen is shown in the background, displaying a line graph with a blue line and a pie chart. The text is overlaid on the screen in a large, white, sans-serif font. The line graph has a legend with 'New Visitor' and 'Returning Visitor'. The pie chart is divided into three segments: a large blue one, a smaller green one, and a very small red one. The laptop keyboard is visible at the bottom of the frame.

Automate code style:
Don't let code style
take over your code
reviews.

The Team



James Brooks



Graham Campbell



Joe Cohen

History

December 2014

StyleCI created for my personal OSS projects.

July 2015

Launched private repo support and new website.

July 2016

New PR analysis system.

Q4
2014

Q1
2015

Q2
2015

Q3
2015

Q4
2015

Q1
2016

Q2
2016

Q3
2016

Q4
2016

Q1
2017

Q2
2017

Q3
2017

March 2015

Opened to everyone for public repos!

June 2016

Added direct push support.

December 2016

Added Slack notifications.

DEMO TIME



So, how does all this work?

Architectural Problems

- StyleCI has to interface with webhooks as well as web users.
 - In fact, 43% of our requests last month were from GitHub webhooks!

Architectural Problems

- StyleCI has to interface with webhooks as well as web users.
 - In fact, 43% of our requests last month were from GitHub webhooks!
- StyleCI has process commands/events in a good order.
 - Double hitting repo enables, double hitting plan change forms, lots of commit events.

Architectural Problems

- StyleCI has to interface with webhooks as well as web users.
 - In fact, 43% of our requests last month were from GitHub webhooks!
- StyleCI has process commands/events in a good order.
 - Double hitting repo enables, double hitting plan change forms, lots of commit events.
- StyleCI has to be correct with it's fixing.
 - We don't want to break people's code!

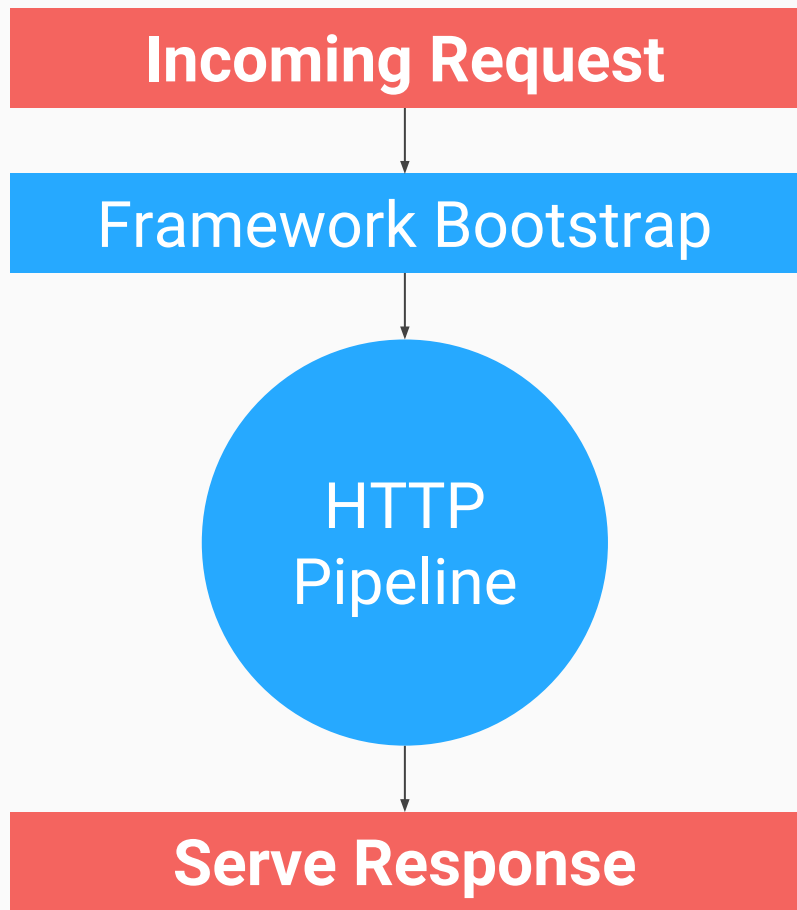
Architectural Problems

- StyleCI has to interface with webhooks as well as web users.
 - In fact, 43% of our requests last month were from GitHub webhooks!
- StyleCI has process commands/events in a good order.
 - Double hitting repo enables, double hitting plan change forms, lots of commit events.
- StyleCI has to be correct with it's fixing.
 - We don't want to break people's code!
- StyleCI has to be high performance.
 - It has to be able to cope with daily usage.
 - It has to return the results as fast as possible.

StyleCI is Built Using Laravel 5!

HTTP Execution

Laravel's Request Execution



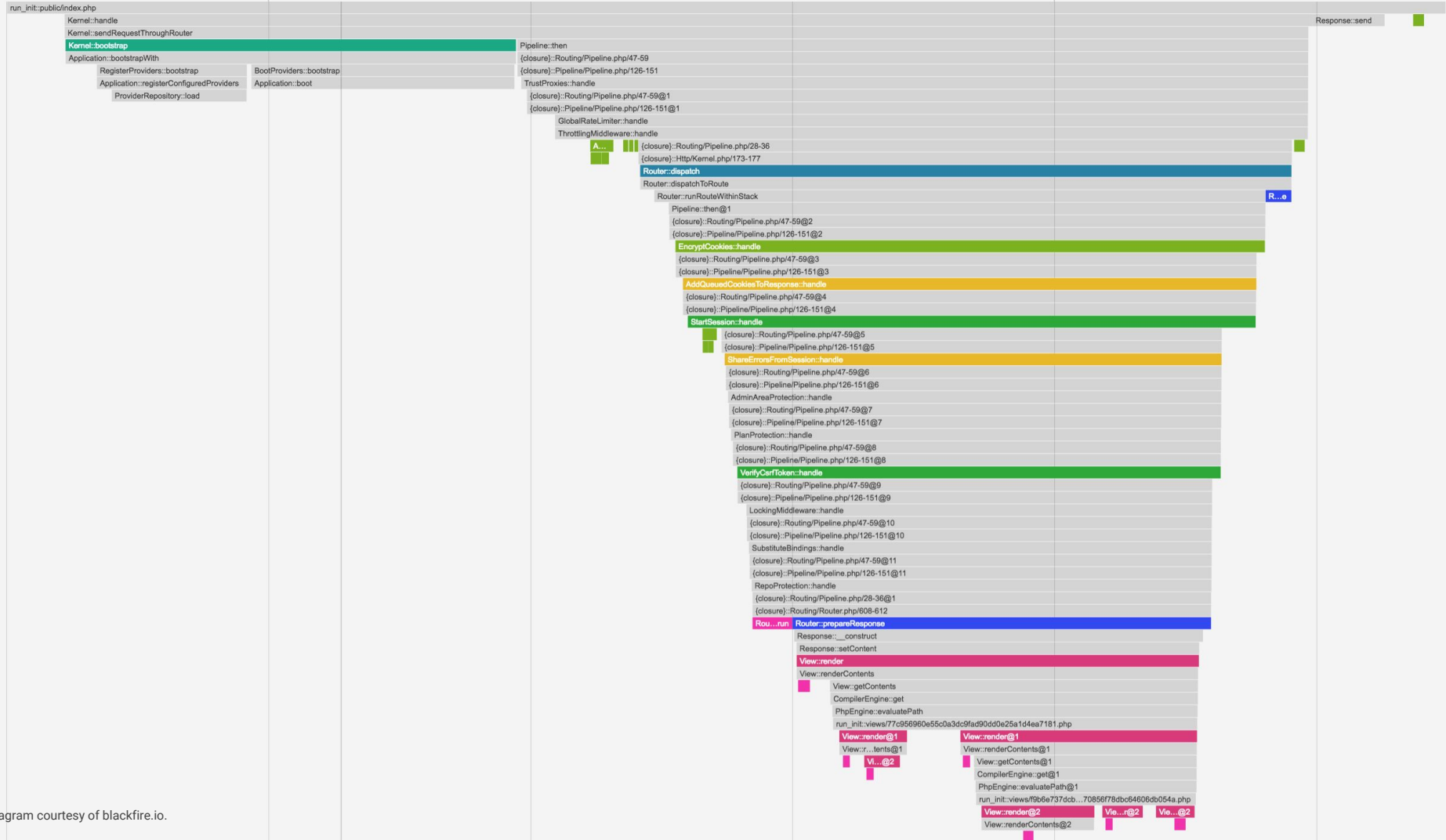
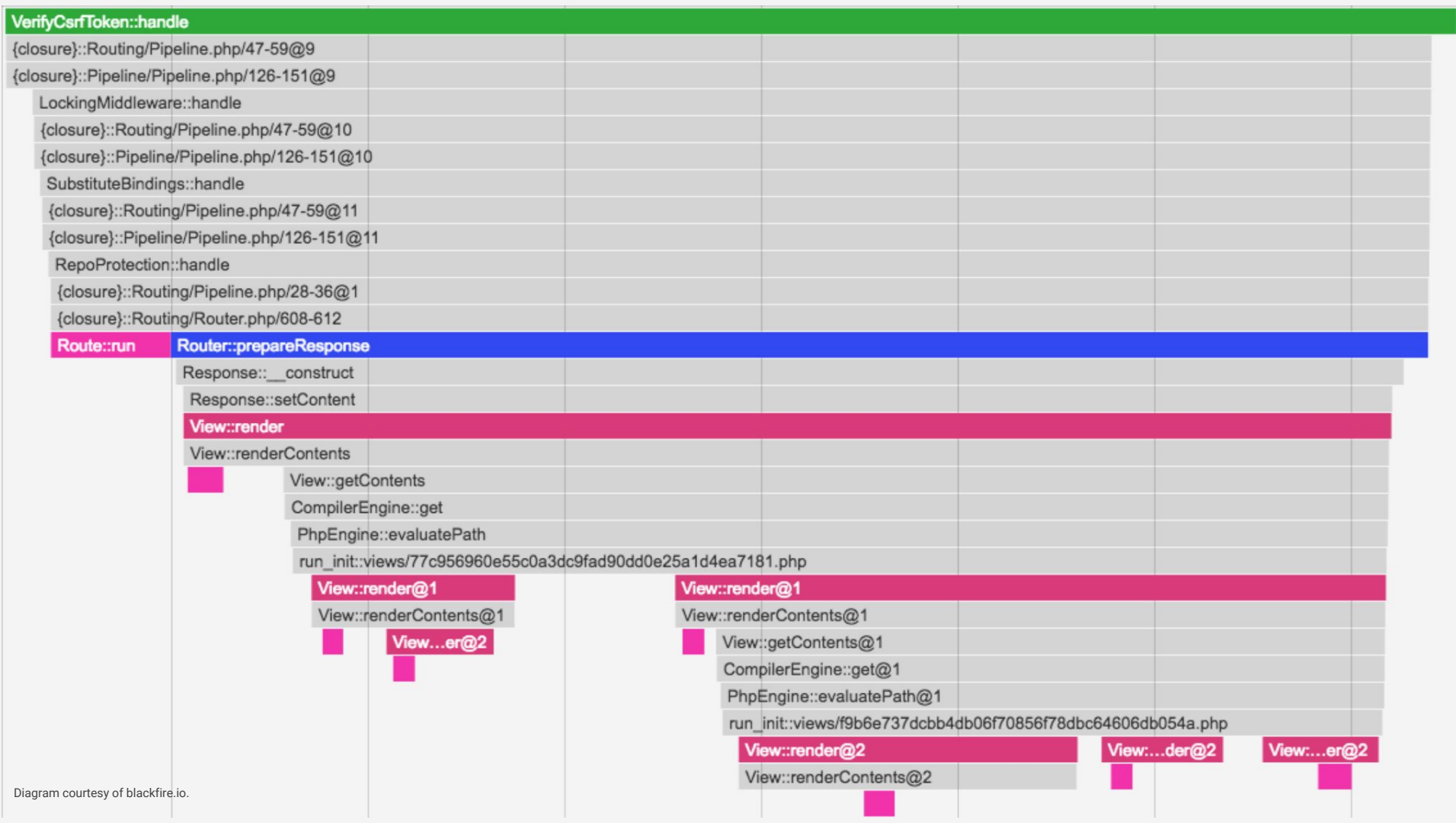


Diagram courtesy of blackfire.io.



Diagram courtesy of blackfire.io.



Again, again!



Diagram courtesy of blackfire.io.

Command Bus!

Commands

- **We have commands and handlers**
- Some commands are async
- Commands never return data
- Commands may omit events
- Commands may throw exceptions

```
<?php
```

```
declare(strict_types=1);
```

```
/*  
 * This file is part of Alt Three StyleCI.  
 *  
 * (c) Alt Three Services Limited  
 *  
 * For the full copyright and license information, please view the LICENSE  
 * file that was distributed with this source code.  
 */
```

```
namespace AltThree\StyleCI\Bus\Commands\Repo;
```

```
use AltThree\StyleCI\Models\Repo;
```

```
/**  
 * This is the disable repo command class.  
 *  
 * @author Graham Campbell <graham@alt-three.com>  
 */
```

```
final class DisableRepoCommand
```

```
{  
    /**  
     * The repo to disable.  
     *  
     * @var \AltThree\StyleCI\Models\Repo  
     */  
    public $repo;  
  
    /**  
     * Create a new disable repo command instance.  
     *  
     * @param \AltThree\StyleCI\Models\Repo $repo  
     *  
     * @return void  
     */  
    public function __construct(Repo $repo)  
    {  
        $this->repo = $repo;  
    }  
}
```

```
<?php
```

```
declare(strict_types=1);
```

```
/*  
 * This file is part of Alt Three StyleCI.  
 *  
 * (c) Alt Three Services Limited  
 *  
 * For the full copyright and license information, please view the LICENSE  
 * file that was distributed with this source code.  
 */
```

```
namespace AltThree\StyleCI\Bus\Handlers\Commands\Repo;
```

```
use AltThree\StyleCI\Bus\Commands\Repo\DisableRepoCommand;
```

```
use AltThree\StyleCI\Bus\Events\Repo\RepoWasDisabledEvent;
```

```
/**  
 * This is the disable repo command handler class.  
 *  
 * @author Graham Campbell <graham@alt-three.com>  
 */
```

```
class DisableRepoCommandHandler
```

```
{  
    /**  
     * Handle the disable repo command.  
     *  
     * @param \AltThree\StyleCI\Bus\Commands\Repo\DisableRepoCommand $command  
     *  
     * @return void  
     */  
    public function handle(DisableRepoCommand $command)  
    {  
        $repo = $command->repo;  
  
        event(new RepoWasDisabledEvent($repo));  
  
        $repo->delete();  
    }  
}
```

Events

- **We have events and handlers**
- All handlers are synchronous
- Some handlers are “crashable”


```
<?php
```

```
declare(strict_types=1);
```

```
/*
```

```
* This file is part of Alt Three StyleCI.
```

```
*
```

```
* (c) Alt Three Services Limited
```

```
*
```

```
* For the full copyright and license information, please view the LICENSE
```

```
* file that was distributed with this source code.
```

```
*/
```

```
namespace AltThree\StyleCI\Bus\Handlers\Events;
```

```
/**
```

```
* This is the crashable handler interface.
```

```
*
```

```
* @author Graham Campbell <graham@alt-three.com>
```

```
*/
```

```
interface CrashableHandlerInterface
```

```
{
```

```
    //
```

```
}
```

```

namespace AltThree\StyleCI\Bus\Handlers\Events\Analysis;

use AltThree\StyleCI\Bus\Events\Analysis\AnalysisEventInterface;
use AltThree\StyleCI\Bus\Events\Analysis\AnalysisWasQueuedEvent;
use AltThree\StyleCI\Bus\Handlers\Events\CrashableHandlerInterface;
use AltThree\StyleCI\Integrations\Contracts\Metrics;

/**
 * This is the analysis metrics handler class.
 *
 * @author Graham Campbell <graham@alt-three.com>
 */
class AnalysisMetricsHandler implements CrashableHandlerInterface
{
    /**
     * The metrics instance.
     *
     * @var \AltThree\StyleCI\Integrations\Contracts\Metrics
     */
    protected $metrics;

    /**
     * Create a new analysis metrics handler instance.
     *
     * @param \AltThree\StyleCI\Integrations\Contracts\Metrics $metrics
     *
     * @return void
     */
    public function __construct(Metrics $metrics)
    {
        $this->metrics = $metrics;
    }

    /**
     * Handle the analysis event.
     *
     * @param \AltThree\StyleCI\Bus\Events\Analysis\AnalysisWasQueuedEvent|\AltThree\StyleCI\Bus\Events\Analysis\DirectPushingFixesWasQueued\SendingPullRequestWasQueuedEvent|\AltThree\StyleCI\Bus\Events\Analysis\PullRequest\SendingPullRequestWasQueuedEvent $event
     *
     * @return void
     */
    public function handle(AnalysisEventInterface $event)
    {
        if ($event instanceof AnalysisWasQueuedEvent) {
            $this->metrics->recordAnalysis();
        } else {
            $this->metrics->recordPush();
        }
    }
}

```

```

/**
 * This is the crash wrapping handler class.
 *
 * @author Graham Campbell <graham@alt-three.com>
 */
class CrashWrappingHandler
{
    /**
     * The crashable handler instance.
     *
     * @var \AltThree\StyleCI\Bus\Handlers\Events\CrashableHandlerInterface
     */
    protected $handler;

    /**
     * The logger instance.
     *
     * @var \Psr\Log\LoggerInterface
     */
    protected $logger;

    /**
     * Create a new crash wrapping handler instance.
     *
     * @param \AltThree\StyleCI\Bus\Handlers\Events\CrashableHandlerInterface $handler
     * @param \Psr\Log\LoggerInterface $logger
     *
     * @return void
     */
    public function __construct(CrashableHandlerInterface $handler, LoggerInterface $logger)
    {
        $this->handler = $handler;
        $this->logger = $logger;
    }

    /**
     * Handle the event and deal with any crashes.
     *
     * @param \AltThree\StyleCI\Bus\Events\EventInterface $event
     *
     * @return void
     */
    public function handle(EventInterface $event)
    {
        try {
            $this->handler->handle($event);
        } catch (Throwable $e) {
            $this->logger->warning($e);
        }
    }
}

```

```

/**
 * Register the application's event listeners.
 *
 * @param \Illuminate\Contracts\Events\Dispatcher $dispatcher
 *
 * @return void
 */
public function boot(Dispatcher $dispatcher)
{
    foreach ($this->listen as $event => $listeners) {
        foreach ($listeners as $listener) {
            $dispatcher->listen($event, $this->getListener($listener));
        }

        $dispatcher->listen($event, $this->getListener(ActionStorageHandler::class));
    }
}

/**
 * Create a class based listener using the IoC container.
 *
 * Note that the should queue interface is totally ignored here.
 *
 * @param string $listener
 *
 * @return \Closure
 */
public function getListener(string $listener)
{
    return function ($event) use ($listener) {
        $handler = $this->app->make($listener);

        if ($handler instanceof CrashableHandlerInterface) {
            $handler = new CrashWrappingHandler($handler, $this->app->make(LoggerInterface::class));
        }

        return $handler->handle($event);
    };
}
}

```

Github\Exception\ApiLimitExceedException · AltThree\StyleCI\Bus\Jobs\Analysis\RunAnalysisJob

You have reached GitHub hourly limit! Actual limit is: 5000

3 days ago · Sep 4th, 14:31:46 UTC

production

STACKTRACE

BREADCRUMBS

APP

DEVICE

JOB

ADD TAB +

Github\Exception\ApiLimitExceedException · You have reached GitHub hourly limit! Actual limit is: 5000

FULL TRACE

RAW

app/Integrations/GitHub/Status.php:80 · AltThree\StyleCI\Integrations\GitHub\Status::push



app/Integrations/Wrapper/Status.php:54 · AltThree\StyleCI\Integrations\Wrapper\Status::push



app/Bus/Handlers/Events/Analysis/AnalysisStatusHandler.php:55 · AltThree\StyleCI\Bus\Handlers\Events\Analysis\AnalysisStatusHandler::handle



app/Bus/Handlers/Events/CrashWrappingHandler.php:65 · AltThree\StyleCI\Bus\Handlers\Events\CrashWrappingHandler::handle



app/Foundation/Providers/EventServiceProvider.php:394 · AltThree\StyleCI\Foundation\Providers\EventServiceProvider::AltThree\StyleCI\Foundation\Providers\{closure}



app/Bus/Handlers/Jobs/Analysis/RunAnalysisJobHandler.php:155 · AltThree\StyleCI\Bus\Handlers\Jobs\Analysis\RunAnalysisJobHandler::handle



app/Bus/Middleware/LockingMiddleware.php:70 · AltThree\StyleCI\Bus\Middleware\LockingMiddleware::AltThree\StyleCI\Bus\Middleware\{closure}



app/Bus/Middleware/LockingMiddleware.php:77 · AltThree\StyleCI\Bus\Middleware\LockingMiddleware::handle



app/Bus/Middleware/ThrottlingMiddleware.php:70 · AltThree\StyleCI\Bus\Middleware\ThrottlingMiddleware::handle



The backend...

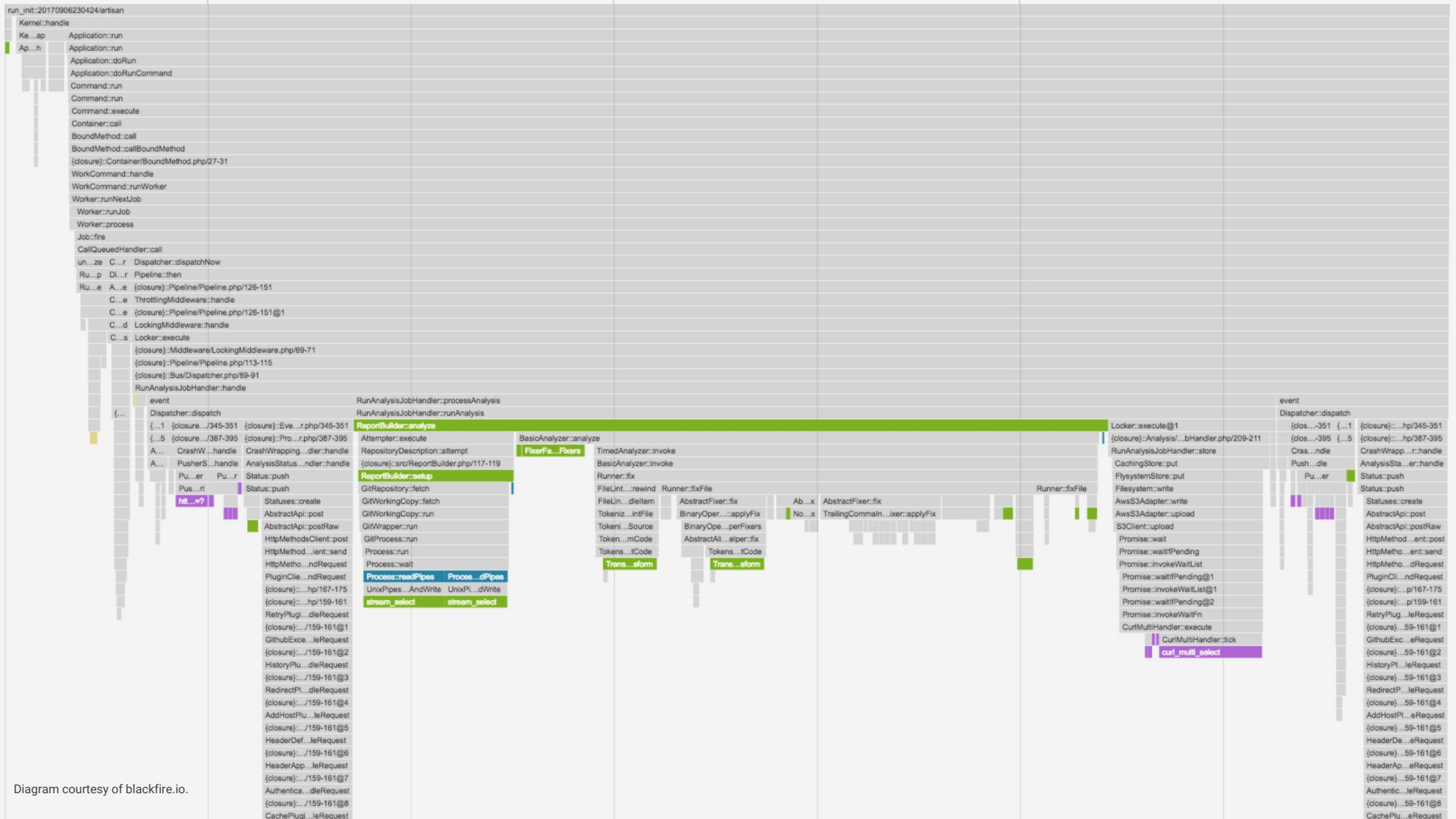


Diagram courtesy of blackfire.io.

WorkCommand::handle
WorkCommand::runWorker
Worker::runNextJob
Worker::runJob
Worker::process
Job::fire
CallQueuedHandler::call
Dispatcher::dispatchNow
Pipeline::then
(closure):Pipeline/Pipeline.php/126-151
ThrottlingMiddleware::handle
(closure):Pipeline/Pipeline.php/126-151@1
LockingMiddleware::handle
Locker::execute
(closure):Middleware/LockingMiddleware.php/69-71
(closure):Pipeline/Pipeline.php/113-115
(closure):Bus/Dispatcher.php/89-91
RunAnalysisJobHandler::handle
Model::save
event
Dispatcher::dispatch
(closure):Events/...tcher.php/345-351
(closure):Provide...vider.php/387-395
AnalysisLoggingHandler::handle
AnalysisLoggingH...ler::getContext
Ana...ay Rep...ay Mo...t
Re...is Mo...e
Mo...t Mo...e
M...e Mo...d
M...e
M...d

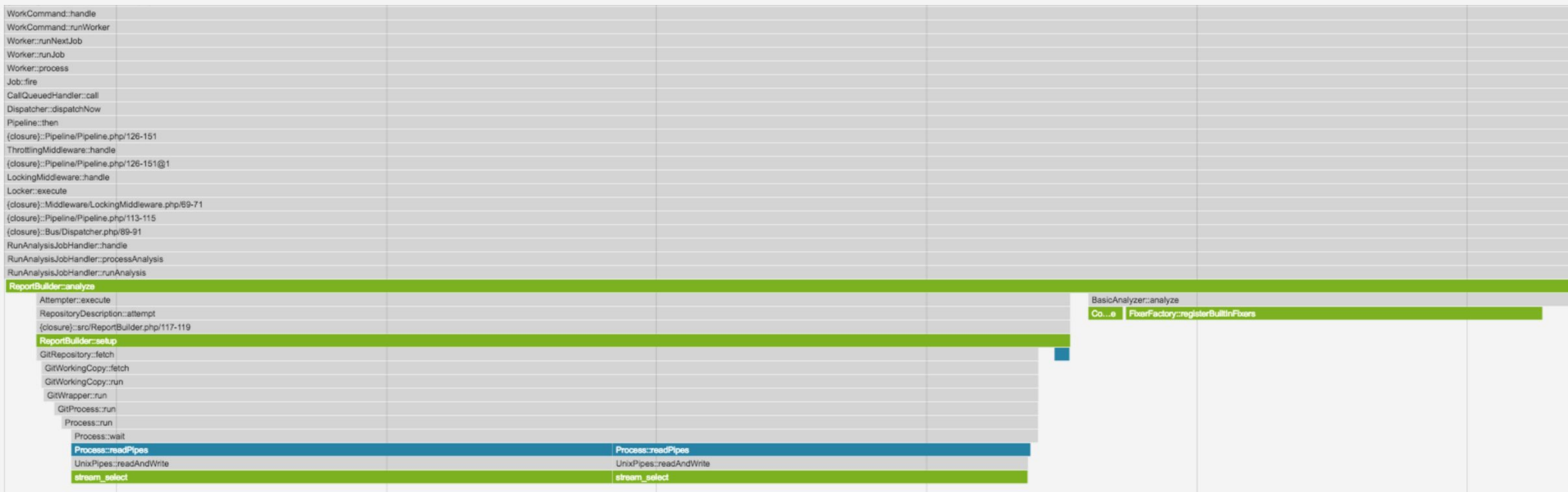
(closure):Events/Dispatcher.php/345-351
(closure):Providers/EventServiceProvider.php/387-395
Ap...ke CrashWrappingHandler::handle
Co...ke PusherStatusHandler::handle
Co...ve Pusher::trigger
Con...id Pusher::exec_curl
https://api.pusherapp.com...uth_version?&body_md5=?
ht...w?

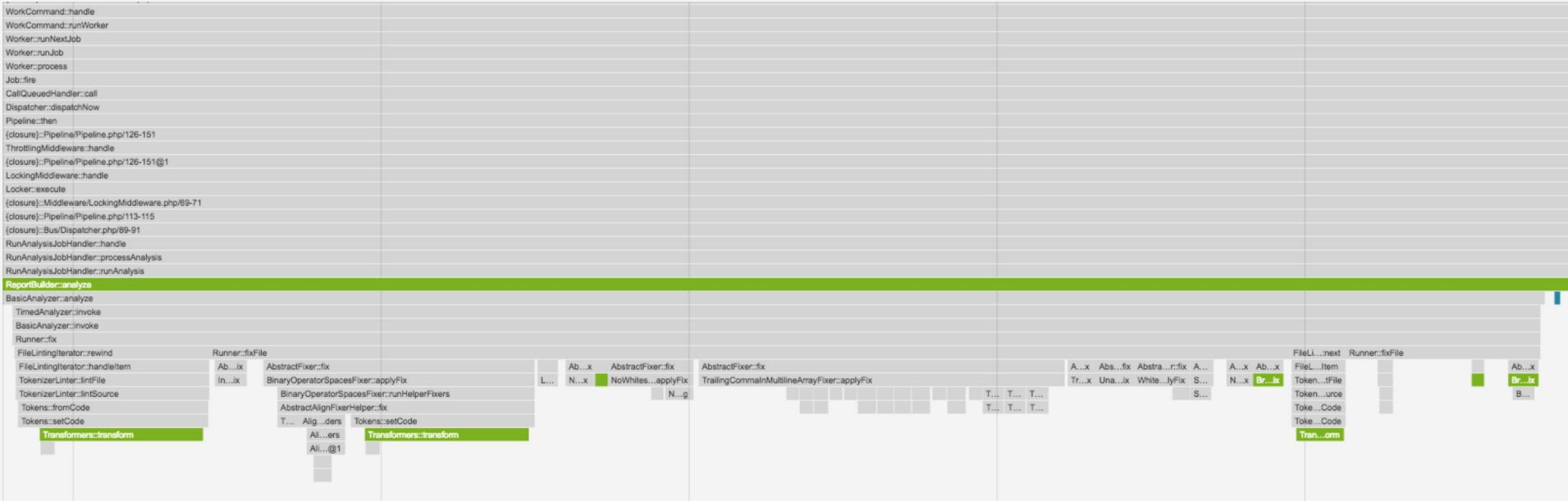
Pu...er PusherStatusHandler::trigger
Pus...ri Us...rs Re...y Pu...r Pu...er Pu...er ht...w?
Pu...d Pu...ri Pu...d
ht...w? ht...w? ht...w?

(closure):Events/Dispatcher.php/345-351
(closure):Providers/EventServiceProvider.php/387-395
CrashWrappingHandler::handle
AnalysisStatusHandler::handle
Status::push
Status::push
ClientFactory::make
GitHubFactory::make
GitHubF...Builder

Statuses::create
AbstractApi::post
AbstractApi::postRaw
HttpMethodsClient::post
HttpMethodsClient::send
HttpMethodsClient::sendRequest
PluginClient::sendRequest
(closure):src/PluginClient.php/167-175
(closure):src/PluginClient.php/159-161
RetryPlugin::handleRequest
(closure):src/PluginClient.php/159-161@1
GithubExceptionHandler::handleRequest
(closure):src/PluginClient.php/159-161@2
HistoryPlugin::handleRequest
(closure):src/PluginClient.php/159-161@3
RedirectPlugin::handleRequest
(closure):src/PluginClient.php/159-161@4
AddHostPlugin::handleRequest
(closure):src/PluginClient.php/159-161@5
HeaderDefaultsPlugin::handleRequest
(closure):src/PluginClient.php/159-161@6
HeaderAppendPlugin::handleRequest
(closure):src/PluginClient.php/159-161@7
Authentication::handleRequest
(closure):src/PluginClient.php/159-161@8
CachePlugin::handleRequest
(closure):src/PluginClient.php/79-85
Client::sendRequest
Promise::wait
Promise::wait
Promise::waitIfPending
Promise::invokeWaitList
Promise::waitIfPending@1
Promise::invokeWaitFn
CurlMultiHandler::execute
CurlM...tick c...l c...l CurlMultiHandler::tick
c...l ou...c curl_multi_select

Diagram courtesy of blackfire.io.





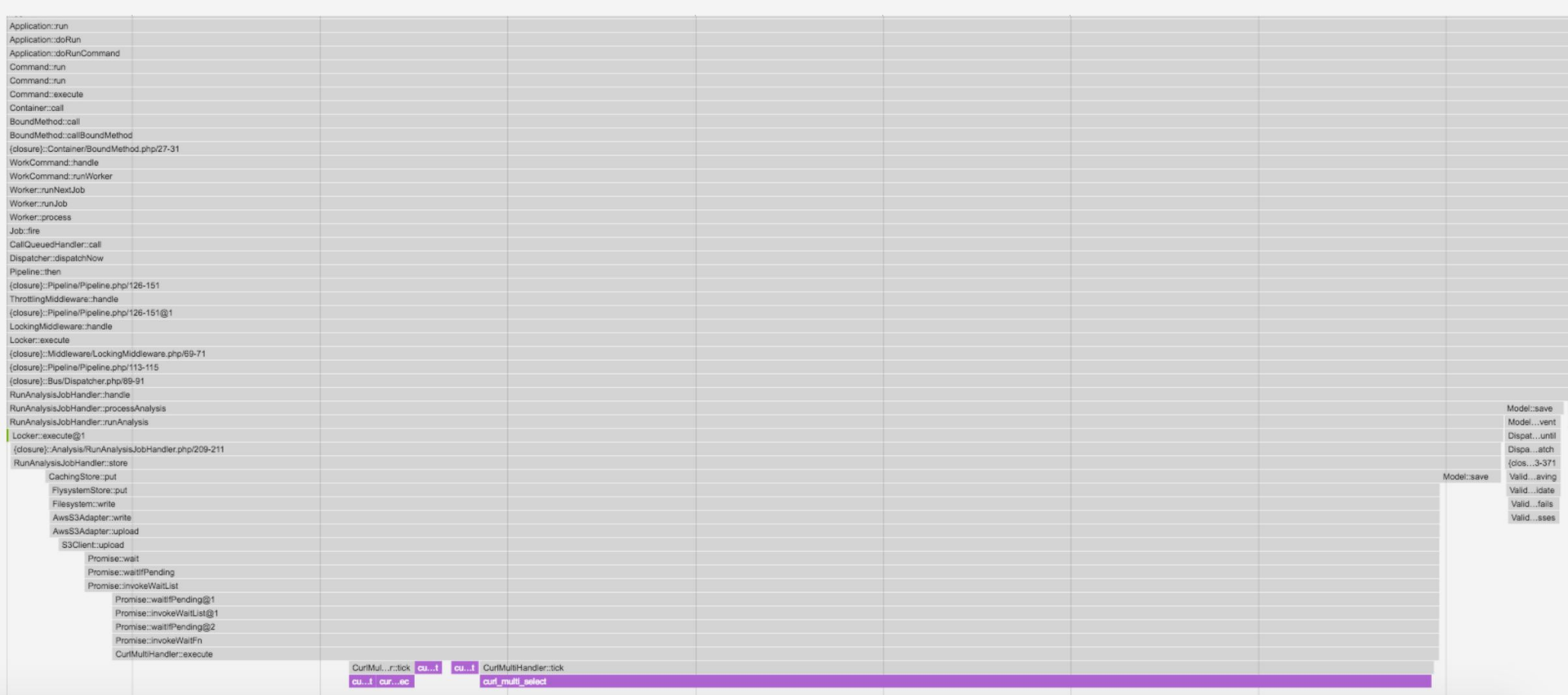


Diagram courtesy of blackfire.io.

Caching

```
19  /**
20   * This is the store class.
21   *
22   * @author Graham Campbell <graham@alt-three.com>
23   */
24  class Store
25  {
26      /**
27       * The special dead value identifier.
28       *
29       * Users should not set this value by hand, as it will kill their entry.
30       *
31       * @var string
32       */
33      const DEAD = 'DEAD-ENTRY';
34
35      /**
36       * The number of seconds death lasts for.
37       *
38       * @var int
39       */
40      const TIMEOUT = 10;
41
42      /**
43       * The locker prefix.
44       *
45       * @var int
46       */
47      const PREFIX = 'store-';
48
49      /**
50       * The cache instance.
51       *
52       * @var \Illuminate\Contracts\Cache\Repository
53       */
54      protected $cache;
55
56      /**
57       * The locker instance.
58       *
59       * @var \AltThree\Locker\Locker
60       */
61      protected $locker;
```

```
63     /**
64      * Create a new store instance.
65      *
66      * @param \Illuminate\Contracts\Cache\Repository $cache
67      * @param \AltThree\Locker\Locker $locker
68      *
69      * @return void
70      */
71     public function __construct(Repository $cache, Locker $locker)
72     {
73         $this->cache = $cache;
74         $this->locker = $locker;
75     }
76
77     /**
78      * Get the value at the given key, if not dead.
79      *
80      * If dead, or unset, we will return null.
81      *
82      * @param string $key The key to use.
83      *
84      * @return mixed
85      */
86     public function get(string $key)
87     {
88         $value = $this->cache->get($key);
89
90         if ($value === static::DEAD) {
91             return;
92         }
93
94         return $value;
95     }
```

```

97  /**
98   * Save the value at the given key, if not dead.
99   *
100  * Stores are performed atomically with other saves, and kills.
101  *
102  * @param string    $key    The key to use.
103  * @param mixed     $value  The value to store.
104  * @param int|float $life   The lifetime in minutes.
105  *
106  * @throws \AltThree\Locker\Exceptions\UnableToAcquireLockException
107  *
108  * @return void
109  */
110 public function save(string $key, $value, $life)
111 {
112     $this->locker->execute(function () use ($key, $value, $life) {
113         if ($this->cache->get($key) === static::DEAD) {
114             return;
115         }
116
117         $this->cache->put($key, $value, $life);
118     }, static::PREFIX.$key, static::TIMEOUT * 1000);
119 }

```



```
121  /**
122   * Kill the given key.
123   *
124   * Kills are performed atomically with other kills and saves. On
125   * resurrection, after the timeout values re-initialize to being unset.
126   *
127   * @param string $key The key to use.
128   *
129   * @throws \AltThree\Locker\Exceptions\UnableToAcquireLockException
130   *
131   * @return void
132   */
133  public function kill(string $key)
134  {
135      $this->locker->execute(function () use ($key) {
136          $this->cache->put($key, static::DEAD, static::TIMEOUT / 60);
137          }, static::PREFIX.$key, static::TIMEOUT * 1000);
138  }
139  }
```

```

24 class Bunker
25 {
26     /**
27      * The store instance.
28      *
29      * @var \AltThree\Store\Store
30      */
31     protected $store;
32
33     /**
34      * Create a new bunker instance.
35      *
36      * @param \AltThree\Store\Store $store
37      *
38      * @return void
39      */
40     public function __construct(Store $store)
41     {
42         $this->store = $store;
43     }
44
45     /**
46      * Get the bunker item for the given key.
47      *
48      * @param string $key The key to use.
49      * @param int|float $life The lifetime in minutes.
50      *
51      * @throws \InvalidArgumentException
52      *
53      * @return \AltThree\Store\Bunker\Item
54      */
55     public function for(string $key, $life)
56     {
57         if (!$key) {
58             throw new InvalidArgumentException('The bunker key must be non-empty.');
```

```

62     /**
63      * Get the item value, if not dead.
64      *
65      * If dead, or unset, we will return null.
66      *
67      * @return mixed
68      */
69     public function get()
70     {
71         return $this->store->get($this->key);
72     }
73
74     /**
75      * Save the item value, if not dead.
76      *
77      * Stores are performed atomically with other saves, and kills.
78      *
79      * @param mixed $value The value to store.
80      *
81      * @throws \AltThree\Locker\Exceptions\UnableToAcquireLockException
82      *
83      * @return void
84      */
85     public function save($value)
86     {
87         $this->store->save($this->key, $value, $this->life);
88     }
89
90     /**
91      * Kill the item.
92      *
93      * Kills are performed atomically with other kills, and saves. On
94      * resurrection, after the timeout, values re-initialize to being unset.
95      *
96      * @throws \AltThree\Locker\Exceptions\UnableToAcquireLockException
97      *
98      * @return void
99      */
100    public function kill()
101    {
102        $this->store->kill($this->key);
103    }
104 }

```

1.1 Million Analyses Performed!

*When I checked last night, this figure stood at 1,135,053.

What are you waiting for?

Sign up today for free, and
get unlimited public repos.

Paid plans are also available for private repos.

<https://styleci.io/>

