

API Creation to Iteration Without the Frustration

Steve Rice

software engineer

frontend, backend, APIs

API Captain at **PagerDuty** in San Francisco

Platform Team

What is PagerDuty?

- incident resolution platform
- triage events
- notify responders
- manage incident response
- SaaS product
- ~250 employees, started with half that
- Rails, MySQL, Scala, Cassandra, Kafka, etc.
- Backbone, Ember.js, Android + iOS

The screenshot shows a web browser window titled "PagerDuty Developer". The main content area features a title "Get started with the PagerDuty API" and three steps for beginners. A "Get Started" button is present. To the right is a sidebar with links to various API endpoints like REST API, Incidents, and Reports. At the bottom, there are sections for Questions, Release Notes, and Integration.

{ PD: "developer" }

Get started with the PagerDuty API

Step 1 [Sign up](#). Our trial accounts are free to try.

Step 2 Play around. Send yourself a few alerts. Get a feel for the place.

Step 3 Come back here. [Dig into our REST APIs](#) or [check out our Events API](#).

[Get Started](#)

Looking for Our (Non API) Integration Guides?
Hop over to our [monitoring system integration guides](#) for information on how to directly integrate PagerDuty with [Nagios](#), [New Relic](#), [Pingdom](#), [Server Density](#), [Splunk](#), and a bunch more. If your monitoring system isn't directly supported, you can come back here and use our API to get events into PagerDuty.

REST API

- [General Info](#)
- [Authentication](#)
- [Errors](#)
- [Types](#)
- [Pagination](#)
- [Incidents](#)
- [Alerts](#)
- [Reports](#)
- [Schedules](#)
- [Overrides](#)
- [Users](#)
- [Contact Methods](#)
- [Notification Rules](#)
- [Log Entries](#)
- [Services](#)
- [Email Filters](#)
- [Maintenance Windows](#)

Integration

- [Integration API](#)
- [Trigger](#)
- [Acknowledge](#)
- [Resolve](#)

Questions?

We are here to help!
If you have any questions or need help setting up your PagerDuty account, please don't hesitate to contact us.

Release Notes

[December 7, 2012](#) • [Alerts API now has filter parameter properly documented.](#)

[December 6, 2012](#) • [User#create has requester_id properly documented.](#)

<https://web.archive.org/web/20130116145245/http://developer.pagerduty.com/>

```
# This controller is for communication between the main app  
# server and other processes in the system. The idea here is  
# that this shouldn't be exposed to the outside world.
```

- Commits on May 3, 2011
 -  **Refactoring the api controllers into v1 and beta APIs, and adding the...** [...](#) [!\[\]\(3d496ca5740a387f002644c845f4275b_img.jpg\)](#) [3690bca](#) [!\[\]\(78a029b04ee0ee05998c29299c47b06c_img.jpg\)](#)
... concept of 'experimental' functionality
jlaban committed on May 3, 2011

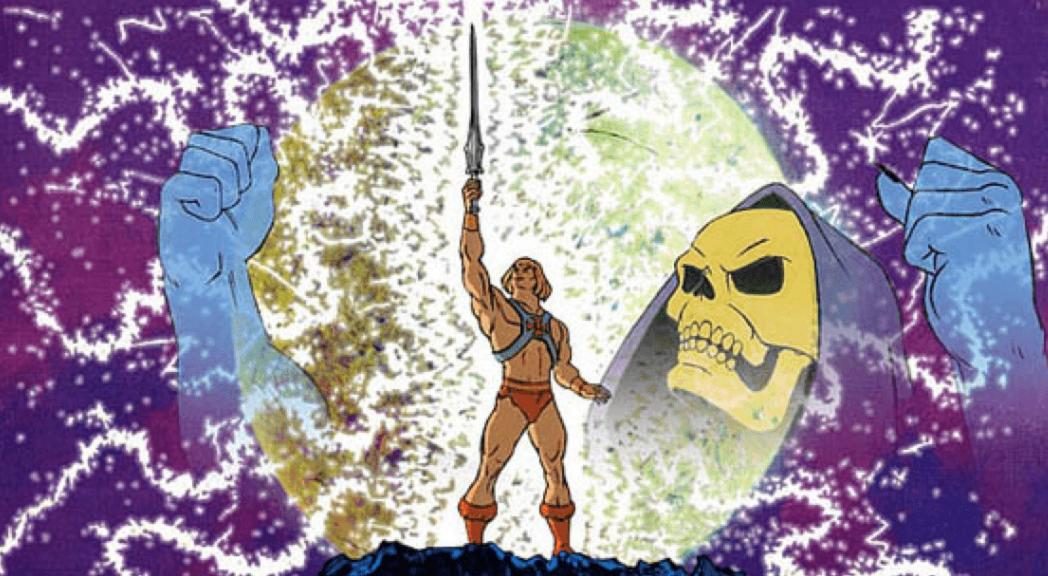
You Have The Power! | PagerDuty Blog

PAGERDUTY

Home Feature Tour FAQ **Blog** Contact Us Jobs Docs Pricing & Signup LOGIN

PagerDuty Blog

SEP 6 12 You Have The Power!
by Ian Enders



Getting excited about APIs can sometimes be a stretch. An API is not a piece of flair you can adorn across your chest, glistening and glowing in eternal splendor. Front-end customer facing features tend to bring more attention. They are more apparent. More visual. More magical. You see their benefit immediately and for an eye regularly trained on the product they are pretty hard to miss. The product team at PagerDuty is making a conscious effort to move past these (mis)conceptions and we have pushed forward to a world where our APIs are exciting. Or at the very least *can be*.

PAGERDUTY

Phone & SMS alerts for your monitoring tools

Sign up for free

Follow

RSS Twitter

Search

Recent Posts

Improved Scout Integration with PagerDuty
Introducing the PagerDuty iPhone app

<https://web.archive.org/web/20130312085114/http://blog.pagerduty.com/2012/09/you-have-the-power/>

Types
Pagination
Alerts
Escalation Policies
Escalation Rules
On-Call
Incidents
Notes
Log Entries
Maintenance Windows
Reports
Schedules
Overrides
Services
Email Filters
Users
Contact Methods
Notification Rules
On-Call
Teams
Webhooks
Integration

Update an existing Email Filter.

Resource URL

```
PUT https://<subdomain>.pagerduty.com/api/v1/services/:service_id/email_filters/:id
```

Parameters

| Name | Type | Required | Description |
|------------------|--------|----------|---|
| subject_mode | String | No | One of <code>always</code> , <code>match</code> , <code>no-match</code> , which, respectively, means to not filter the email trigger by subject, filter it if the email subject matches the given regex, or filter if it doesn't match the given regex. Defaults to <code>always</code> . |
| subject_regex | String | No | The regex to be used when <code>subject_mode</code> is <code>match</code> or <code>no-match</code> . It is a required parameter on such cases. |
| body_mode | String | No | One of <code>always</code> , <code>match</code> , <code>no-match</code> , which, respectively, means to not filter the email trigger by body, filter it if the body email matches the given regex, or filter if it doesn't match the given regex. Defaults to <code>always</code> . |
| body_regex | String | No | The regex to be used when <code>body_mode</code> is <code>match</code> or <code>no-match</code> . It is a required parameter on such cases. |
| from_email_mode | String | No | One of <code>always</code> , <code>match</code> , <code>no-match</code> , which, respectively, means to not filter the email trigger by its from address, filter it if the email from address matches the given regex, or filter if it doesn't match the given regex. Defaults to <code>always</code> . |
| from_email_regex | String | No | The regex to be used when <code>from_email_mode</code> is <code>match</code> or <code>no-match</code> . It is a required parameter on such cases. |

Example



Research

A photograph of several turtles resting on a brown log in a body of water. The turtles have dark shells with distinct yellow and red patterned markings on their heads and necks. They are positioned at various angles, some facing left and some facing right, creating a sense of depth and community.

Stand on the Shoulders
of Giants

Conferences and Talks

<https://www.heavybit.com/library/video/move-fast-dont-break-api/>
<http://amberonrails.com/move-fast-dont-break-your-api/>

A preview of the new Dropbox API v2

Leah Culver | April 8, 2015

  0  4  9

[EDIT June 3, 2015] This post has been updated to reflect the latest API v2 syntax.

Developer Blogs

We've been talking a lot recently about the changes we're making to the API while it's still in beta, which is why we've had to skip the first developer blog post since the API went live. We have some exciting news to share, though, so we're taking a break from the API blog to talk about some structural changes underway, and we'd like to know what you think!

<https://blogs.dropbox.com/developers/2015/04/a-preview-of-the-new-dropbox-api-v2/>

Overall, we've simplified our use of HTTP. For example, most endpoints always use HTTP POST, including those that return structured data. Requests take JSON in the body and responses return JSON in the body.

We will continue to use other HTTP features in specific cases where they provide concrete benefits. For example, endpoints that return bulk binary data (e.g. file contents) support HTTP GET and [ETag-based caching](#). This allows browsers and HTTP client libraries to transparently cache that data for you.

We've also simplified our use of [HTTP status codes](#) for errors. For errors that are common to all API calls, we do the same thing as before: 400 for bad request, 401 for auth failure, 429 for rate limiting, etc. But if a request fails for some call-specific reason, v1 might have returned any of 403, 404, 406,

Protocols and Hypermedia Types

- HAL: http://stateless.co/hal_specification.html
- OData: <http://www.odata.org>
- Collection+JSON: <https://github.com/collection-json/spec>
- JSON-API: <http://jsonapi.org>
- JSON-LD: <http://json-ld.org>
- GData: <https://developers.google.com/gdata/>
- GraphQL: <http://graphql.org>
- Siren: <https://github.com/kevinswiber/siren>

<https://sookiocheff.com/post/api/on-choosing-a-hypermedia-format/>

Plan

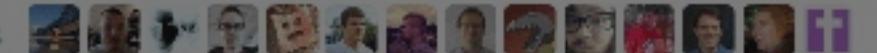


Make decisions.

 gmjosack Fix bug in datetime encoding

05326d5 on Aug 30

14 contributors



Executable File | 762 lines (562 sloc) | 23.3 KB

Raw Blame History



```
396 class Incident(Container):
397     def __init__(self, *args, **kwargs):
398         Container.__init__(self, *args, **kwargs)
399         self.log_entries = LogEntries(self.pagerduty, self)
400         self.notes = Notes(self.pagerduty, self)
401
402     def _do_action(self, verb, requester_id, **kwargs):
403         path = '{}/{}/{}'.format(self.collection_name, self.id, verb)
404         data = {'verb': verb, 'requester_id': requester_id}
405         data.update(kwargs)
406         return self.pagerduty.request('PUT', path, data=_json_dumper(data))
407
408     def has_subject(self):                      https://github.com/dropbox/pygerduty
409         return hasattr(self.trigger_summary_data, 'subject')
410
411     def resolve(self, requester_id):
412         self._do_action('resolve', requester_id=requester_id)
413
414     def acknowledge(self, requester_id):
415         self._do_action('acknowledge', requester_id=requester_id)
416
417     def snooze(self, requester_id, duration):
418         self._do_action('snooze', requester_id=requester_id, duration=duration)
419
420     def get_trigger_log_entry(self, **kwargs):
421         match = TRIGGER_LOG_ENTRY_RE.search(self.trigger_details_html_url)
422         return self.log_entries.show(match.group('log_entry_id'), **kwargs)
```

Go to the Source

Pour over logs

- which endpoints aren't being used?
- which endpoints are being used in ways they shouldn't?
- which customers are using the APIs?

An example of a relationship is [Notification Rules](#), whose primary meaning is as a link between a user and a contact method. No individual notification rule has a unique, name-worthy meaning, and so they are handled (read and written) [only as a collection](#). There is no compelling reason to update a single notification rule in place rather than replace the entire object with a new one. While this might seem like a significant loss of functionality, data shows that a trivially small number of clients outside of the PagerDuty web app are using it:

- [over the last 30 days](#), only 11 requests were made to [PUT notification_rules/id](#) using non-cookie authentication, and all by the subdomain `redacted`. While the mobile team experimented with it in the latest "Edit Notification Rules" update in 3.6, they've confirmed that that endpoint isn't being used.

Similarly, the [escalation rules](#) relationship sees little in-place use:

- [over the last 30 days](#), only one request was made to [PUT escalation_rules/id](#) using non-cookie authentication.

This document outlines the vision and scope for v2 of the API.

Ordered lists represent prioritized order. We recannot accomplish everything at once, and thus everything needs to be prioritized sooner or later. I choose sooner.

This document does not attempt to present a comprehensive record of all the changes that will be made, as those will be captured by [stories in JIRA](#). Instead, it establishes the goals, principles, and patterns that we'll use to make the decisions about what needs to change.

It is a living document and will evolve as we discover new edge cases and outliers that weren't previously considered.

Goals

What are we trying to accomplish with API v2?

These are the high-level goals, in priority order:

1. [iteration](#)
 - leave well enough alone
2. [adoption](#)
 - prove the effectiveness of our versioning strategy
3. [consistency](#)
 - provide consistency across endpoints
4. [performance](#)
 - fix design decisions that have led to performance problems
 - see [Deprecated API Functionality](#)
 - enable greater cacheability
5. [semantics](#)
 - better use of generic HTTP libraries and patterns

Pick your Battles

- Don't change what works
- Prioritize
- Keep bullets away from feet

header- and key-based versioning

Accept: application/vnd.pagerduty+json;version=2

Create API Key

X

Description:

API version:

Which version should I choose? ▾

API v2.0 is designed to make it easier for new integrations to communicate with PagerDuty. Most existing PagerDuty integrations will require an API v1.0 key. If you have questions on which API version to use, please [refer to our documentation](#) or contact support@pagerduty.com.

- v2 Current ([documentation](#))
- v1 Legacy ([documentation](#))
- Read-only API Key ?

canonical API subdomain

initech.pagerduty.com

aememonitoring.pagerduty.com

evilcorp.pagerduty.com

api.pagerduty.com

use standards

- respect HTTP semantics
- ISO 8601
- IANA tzinfo

deprecate basic auth



performance tweak

```
{  
  "incidents": [ {  
    "id": "P1R8NC6",  
    "type": "incident",  
    "summary": "[#7893] There are 2 Charizard nearby!",  
    "self": "https://api.pagerduty.com/incidents/P1R8NC6",  
  } ],  
  "limit": 1,  
  "offset": 0,  
  "total": null, //don't care about this  
  "more": true //more resources available  
}
```

consistency

The screenshot shows a web browser window displaying the PagerDuty Developer API documentation. The title bar reads "Resource References · PagerDuty". The header includes the "pagerdutyDeveloper" logo, navigation links for "Guides" and "API Reference", and a search bar with a magnifying glass icon, along with "Sign Up" and "Migrate to v2" buttons.

The left sidebar contains a "BASICS" section with links to "Getting Started", "Migrate to REST API v2", and "API v2 FAQ". Below it is a "REST API" section with links to "Overview", "Authentication", "Versioning", "Rate Limiting", "Endpoints", "Types", "Filtering", "Sorting", "Pagination", "Resource Schemas", "Resource References", "Wrapped Entities", "Includes", "Errors", and "Integrations".

The main content area is titled "Resource References". It begins with a paragraph explaining that references are used to represent pointers to resources, consisting of five fields: id, type, summary, self, and html_url. It then discusses how references are used to represent relationships between resources, noting that modifying the reference or references in a field identified by id and type will update the relationship. It also states that every PagerDuty resource can be uniquely identified by its id and type, ignoring the distinction between type and type_reference.

BASICS

- Getting Started
- Migrate to REST API v2
- API v2 FAQ

REST API

- Overview
- Authentication
- Versioning
- Rate Limiting
- Endpoints
- Types
- Filtering
- Sorting
- Pagination
- Resource Schemas
- Resource References**
- Wrapped Entities
- Includes
- Errors
- Integrations

EVENTS API

- Overview
- Send an Event**
- Trigger an Incident in JavaScript
- Integrations

Resource References

A key component of the PagerDuty REST API is the use of `references` to represent a pointer to a resource. A reference consists of five fields:

- `id` (required, readOnly)
- `type` (required)
- `summary` (readOnly)
- `self` (readOnly)
- `html_url` (readOnly)

References are used anywhere that the identification of an object is important, but the object itself cannot be modified. In this regard, you can think of a reference like an alias — a different object type that represents a resource. They also provide an essential subset of information about the resource that tells a client how to uniquely identify it, provides some information about what it represents, and offers locations where more information can be found about it.

References will be used to represent relationships between resources. If a resource has a relationship with other types of entities, they may be displayed as part of the resource's schema and contain references to the resources of that type that make up the other end of the relationship. Modifying the reference or references in this field — identified only by `id` and `type` — will update the relationship between resources. Modifying any other fields of the reference has no effect on the resource it represents.

`id` and `type`

Every PagerDuty resource can be uniquely identified by the combination of its `id` and `type`, ignoring the distinction between a given `type` and the corresponding `type_reference`. The `type` field indicates what kind of resource it is, which is [equivalent to specify its schema](#).

Change the Plan

Clarify: include[] includes resources throughout the object graph · PagerDuty/captains-log@e67771b

Showing 1 changed file with 3 additions and 1 deletion.

Unified Split

4 proposals/api-v11-vision.md

View

An `include` array must be used to specify any and all related resources that should be used in a response

- `include` must be an array
- `include` must only be used for including resources, not other entities nor extra fields/metadata
- `include` is one level only, there are no nested includes (e.g. `include[] = incidents.trigger_log_entry`)
- an `include` includes resources at all levels
 - there is no syntax for specifying includes at a specific level (e.g. `include[] = incidents.trigger_log_entry` or `include[] = services/incidents/log_entries`)
 - the API will prevent infinite recursion, e.g. `include[] = users` on a `/teams` endpoint will show full `users` inside that team, and each user will have an array of `teams` references. However, those `teams` references will not show full `users`.
- `include` cannot have any default values; i.e. no resources can be explicitly included by default.
 - the exception to this is that when POSTing our PUTting a resource containing nested resources that would ordinarily require `include`-ing, those resources will be included in the response
- `include` must be populated solely by either the pluralized name of a resource or polymorphic relationship

For example, the v1.0 schedules API allows including a user's next `oncall` via:

0 comments on commit e67771b

Lock conversation



I'm largely in favor of this change if you can get it past Ken, Paul or Andrew. Product justification: deduping is the most awesome thing in the world, if you really really want 100 incidents, you can send distinct incident_keys.

- show quoted text -



Ken Rose

4/15/14



★ Other recipients: da...@pagerduty.com, st...@pagerduty.com, ow...@pagerduty.com

Changing the integration API to perform deduping by default is a breaking change. I hate to be so negative, but I can't approve that.

Deduping is currently opt-in. If you as a client want your events deduped, the onus is on you to provide an incident_key. More importantly though, if you as a client do *not* want your events deduped, you just don't set incident_key. If we change the integrations API to dedup automatically, then deduping becomes opt-out. Any client that previously relied on getting a new incident now has to change their code to have something akin to "incident_key = random()". That "has to change their code" part... that's bad.

Additionally, from [https://github.com/PagerDuty/pagerduty-go-client/pull/10#issuecomment-1000000000](#)
| If this field (incident_key) isn't provided, PagerDuty will automatically generate a new incident with a unique key.

Process is Paramount

There it is, documented, in all of its unholy glory. It sucks, it probably wasn't the ideal choice, but it's part of the API contract now and so is sacred.

"But this is a terrible default" I hear you all say.

You're right, but we can't change it, because the API contract is sacred.

"OK, let's make incident_key a required field then, that'll fix this".

That sounds great! Except that's a breaking change too and the API contract is sacred.

"So we can't do anything for the "2010-04-15" API?".

Bingo. The sacredness of the API contract really gets in your way sometimes, doesn't it?

Now, all that said, I understand the scenario that brought this up, but I'd like to think it's a rare one that's solvable through support. I'd be interested in seeing how many customers do not send incident_key and what their use cases are.

– kenrose

P.S. - I am in favour of this behaviour if we ever bump the version of the integration API past "2010-04-15"

- show quoted text -

★ John Laban Unfortunately, I totally agree with Ken. (You guys didn't actually think we could make this change, did you?) HOWEVER, what if we take some inspiration from PD email

4/15/14

- your prototype is not making any breaking changes to existing contracted APIs

What constitutes approval?

An API is not approved until it's been merged into the Captain's Log.

Wild Clients

Any client that PagerDuty cannot, at a whim, revoke or modify without customer impact, is a "**wild client**". These clients are outside PagerDuty's control. These include:

- [Mobile apps](#) that have been shipped to a public app store
- Clients or client libraries written by third parties based on our [published documentation](#)
- Client libraries that have been written and released by PagerDuty

Managed Clients

Any client whose distribution is controlled entirely by PagerDuty is a "**managed client**". This control includes the ability to stop users of the client from being able to use it at any time.

Clients that *are* under PagerDuty's control include:

- the [Backbone-based Web app](#)
- the [Ember-based Web UI app](#)
- the [mobile-web app](#)
- [Watchdog](#)
- [Mobile apps](#) that are only available on an internal testing platform ([Crashlytics Beta](#), [Deploygate](#))

However, you should still expect to modify managed clients if you're making a change to an API change they depend upon — otherwise the client will break!

[INS-579] Make drilldown consistent with time by patslat · Pull Request #3821 · PagerDuty/web

Conversation 7 Commits 6 Files changed 8

 patslat commented on Mar 21, 2015 • edited

Account timezone wasn't being used with legacy reports. Now it is. x-axis alignment doesn't really make sense so I fixed that too. Allows incident presenters to effectively set `time_zone`.

[PagerDuty/captains-log#9](#)

 pd-marge added the **API Review** label on Mar 21, 2015

 patslat commented on the diff on Mar 22, 2015

app/api/v1/report_legacy_adapter.rb [View full changes](#)

| ... | ... | @@ -0,0 +1,16 @@ |
|-----|-----|---|
| | | 1 +module Api |
| | | 2 + module V1 |
| | | 3 + class ReportLegacyAdapter < ApiDuty::Adapter |
| | | 4 + include ::ApiDuty::Adapter::DateAndTime::Generators |
| | | 5 + |

<https://github.com/PagerDuty/flagger>

Document

An aerial photograph of a bridge spanning a deep, narrow canyon. The bridge consists of several concrete piers supporting a road deck. The canyon walls are steep and rocky. A river flows through the bottom of the canyon. In the background, there are more hills and mountains. The sky is clear and blue.

Keep it maintainable

```
450      </td>
451      <td>
452          <a href="/documentation/rest/types#object">Object</a>
453      </td>
454      <td>
455          The PagerDuty <a href="/documentation/rest/services/">service</a> that the incident belongs to. The service will contain fields of i
456      </td>
457      </tr>
458      <tr>
459          <td>
460              escalation_policy
461          </td>
462          <td>
463              <a href="/documentation/rest/types#object">Object</a>
464          </td>
465          <td>
466              The <a href="/documentation/rest/escalation_policies/">escalation policy</a> that the incident belongs to. The policy will contain f
467          </td>
468      </tr>
469      <tr>
470          <td>
471              teams
472          </td>
473          <td>
474              <a href="/documentation/rest/types#array">Array</a>
475          </td>
476          <td>
477              <p>
478                  The <a href="/documentation/rest/teams/">teams</a> involved in the incident's lifecycle. The teams will contain fields of their ow
479              </p>
480              <p>
481                  <strong>NOTE:</strong> You must explicitly include <code>teams</code> in the <code>fields</code> parameter in order to include it
```

Domain-specific technologies



```
4283
4284
4285
4286
4287
4288
4289
4290
4291
4292
4293
4294
4295
4296
4297
4298
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312
4313
4314
"parameters": [
    "name": "team",
    "in": "body",
    "description": "The team to be created.",
    "schema": {
        "type": "object",
        "properties": {
            "team": {
                "$ref": "#/definitions/Team"
            }
        },
        "required": ["team"]
    }
],
"responses": {
    "201": {
        "description": "The team that was created.",
        "schema": {
            "type": "object",
            "properties": {
                "team": {
                    "$ref": "#/definitions/Team"
                }
            },
            "required": ["team"]
        },
        "examples": {
            "application/json": {
                "team": {
                    "id": "PQ9K7I8",
                    "type": "team",
                    "summary": "Engineering"
                }
            }
        }
    }
}
```

Consider the Humans

The screenshot shows the Stripe API Reference documentation page. The header includes the PagerDuty logo, the title "Stripe API Reference", and a navigation bar with links for curl, Ruby, Python, PHP, Java, Node, and Go. The main content area has a sidebar with sections for Introduction, Topics, Core Resources, and Authentication. The main content area features two large columns: "API Reference" and "API libraries". The "API Reference" column contains detailed descriptions of the Stripe API's RESTful nature, predictable URLs, and built-in HTTP features like authentication and verbs. It also discusses cross-origin resource sharing, JSON responses, and API libraries. The "API libraries" column provides information on official and community-supported libraries available in various languages. A callout box highlights the API endpoint as `https://api.stripe.com`.

Stripe API Reference

curl Ruby Python PHP Java Node Go

API Reference

The Stripe API is organized around [REST](#). Our API has predictable, resource-oriented URLs, and uses HTTP response codes to indicate API errors. We use built-in HTTP features, like HTTP authentication and HTTP verbs, which are understood by off-the-shelf HTTP clients. We support [cross-origin resource sharing](#), allowing you to interact securely with our API from a client-side web application (though you should never expose your secret API key in any public website's client-side code). [JSON](#) is returned by all API responses, including errors, although our [API libraries](#) convert responses to appropriate language-specific objects.

To make the API as explorable as possible, accounts have test mode and live mode API keys. There is no "switch" for changing between modes, just use the appropriate key to perform a live or test transaction. Requests made with test mode credentials never hit the banking networks and incur no cost.

API libraries

Official libraries for the Stripe API are [available in several languages](#). Community-supported libraries are also available for [additional languages](#).

API Endpoint

`https://api.stripe.com`

INTRODUCTION

Introduction

TOPICS

Authentication

Errors

Expanding Obj...

Idempotent Req...

Metadata

Pagination

Request IDs

Versioning

CORE RESOURCES

Balance

Charges

Customers

Disputes

Events

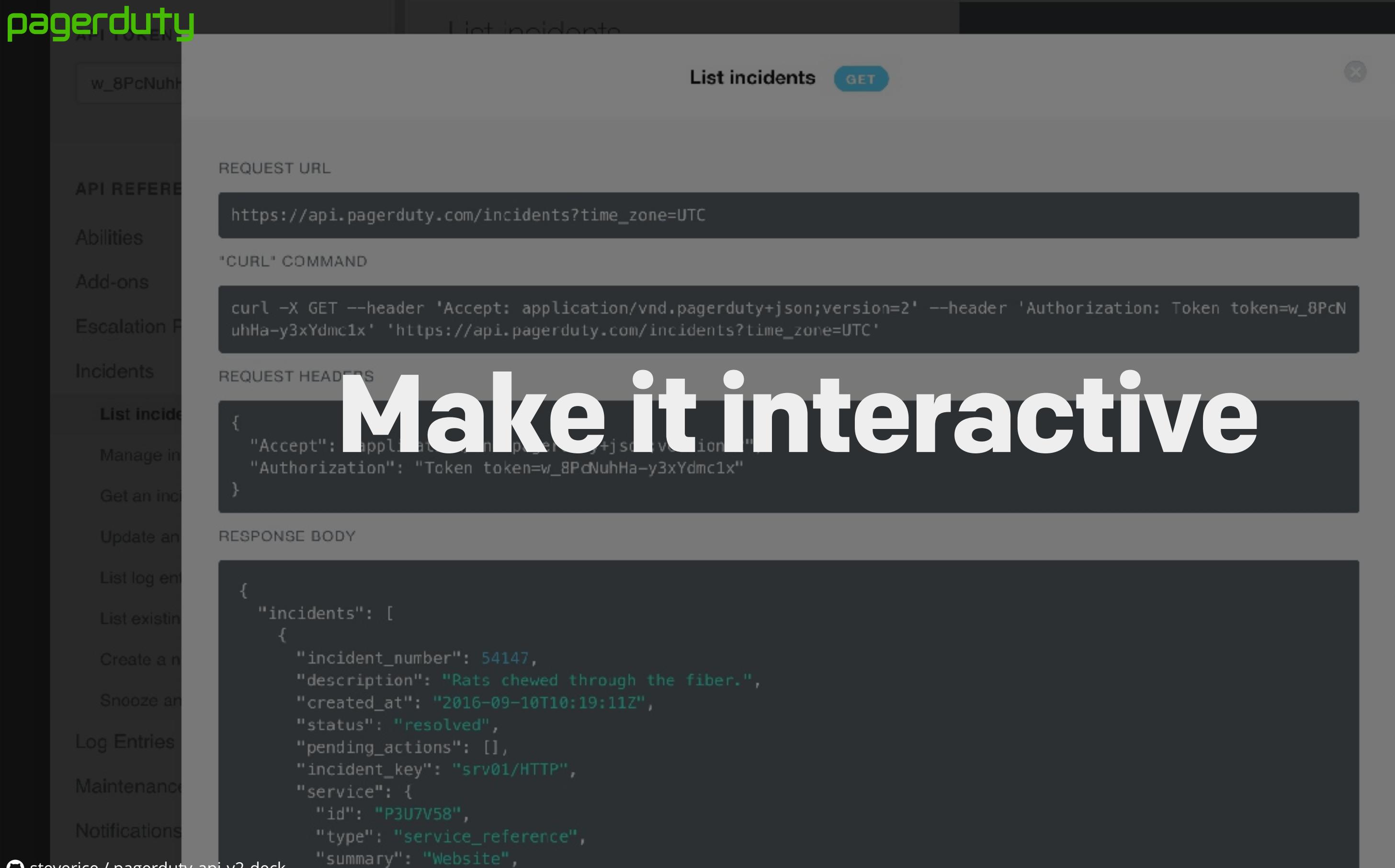
File Uploads

Refunds

Tokens

Authentication

<https://stripe.com/docs/api>



Make it interactive

Develop

Keep it Simple

```
class Api::V2::EscalationPolicyAdapter::Base < ApiDuty::Adapter
  root_attribute :escalation_policy

  validate :name, validator: :is_string
  validate :escalation_rules, validator: :is_array
  validate :description, validator: :is_string
  validate :num_loops, validator: :is_integer
  validate :teams, validator: :is_array
  validate :teams, validator: make_validator_each(
    make_validator_object_resource(%w(team).freeze)
  )

  passthrough :name, :description
  generate :num_repeats, from: :num_loops do |num|
    num.to_i + 1
  end
```

Keep it Simple

```
module Api
  module V2
    class SchedulesController < Api::V1::SchedulesController
    end
  end
end
```

Be pragmatic

DRY is not a god

```
def unwrapped_incident_json
  # V2+: incident object is wrapped.
  json[:incident]
end

def assert_assigned_to_user(json, user)
  # V2+: assigned_to_user is removed
  assert_not_include json.keys, 'assigned_to_user'
end

def assert_reassigned_to_user(json, user)
  # V2+: assigned_to_user is removed
  assert_not_include json.keys, 'assigned_to_user'
end

def assert_incident_count(expected_json)
```

Test

Dogfood.
Everything.

| API-64 first cut requesting v1.1 · PagerDuty/web@7b96b87 | |
|--|---|
| | 323 + sync: (method, model, options) -> 324 + options (options = {}) 325 + options.headers (options.headers = {}) 326 + options.headers.Accept = 327 + Pd.Helpers.RequestHeader.getAcceptHeader() 328 + return Backbone.Collection.prototype.sync.call(this, method, 329 model, options) 330 + 331 + 332 + |
| 320 window.AppRoutes = {} 321 mixin ParseUrlParams, class window.Pd.Router extends Backbone.Router 322 | 330 window.AppRoutes = {} 331 mixin ParseUrlParams, class window.Pd.Router extends Backbone.Router 332 |
| ✖ | |
| 8 app/assets/javascripts/shared/helpers/request_header.coffee | View |
| ... @@ -0,0 +1,8 @@ | 1 +window.Pd = {} 2 +window.Pd.Helpers = {} 3 +window.Pd.Helpers.RequestHeader = 4 + getAcceptHeader: -> 5 + 'application/vnd.pagerduty+json;version=1.1a' 6 + 7 + setApiAcceptHeader: (request) -> 8 + request.setRequestHeader('Accept', Pd.Helpers.RequestHeader.getAcceptHeader()) |
| 2 app/assets/javascripts/users/models/users.coffee | View |
| ✖ @@ -42,11 +42,13 @@ class window.User extends Pd.Model | |
| 42 \$.ajax 43 type: 'PUT' 44 url: "/api/v1/teams/#{team.id}/users/#{@id}" 45 | 42 \$.ajax 43 type: 'PUT' 44 url: "/api/v1/teams/#{team.id}/users/#{@id}" 45 + beforeSend: Pd.Helpers.RequestHeader.setApiAcceptHeader 46 |

Automate



Trust, but Verify

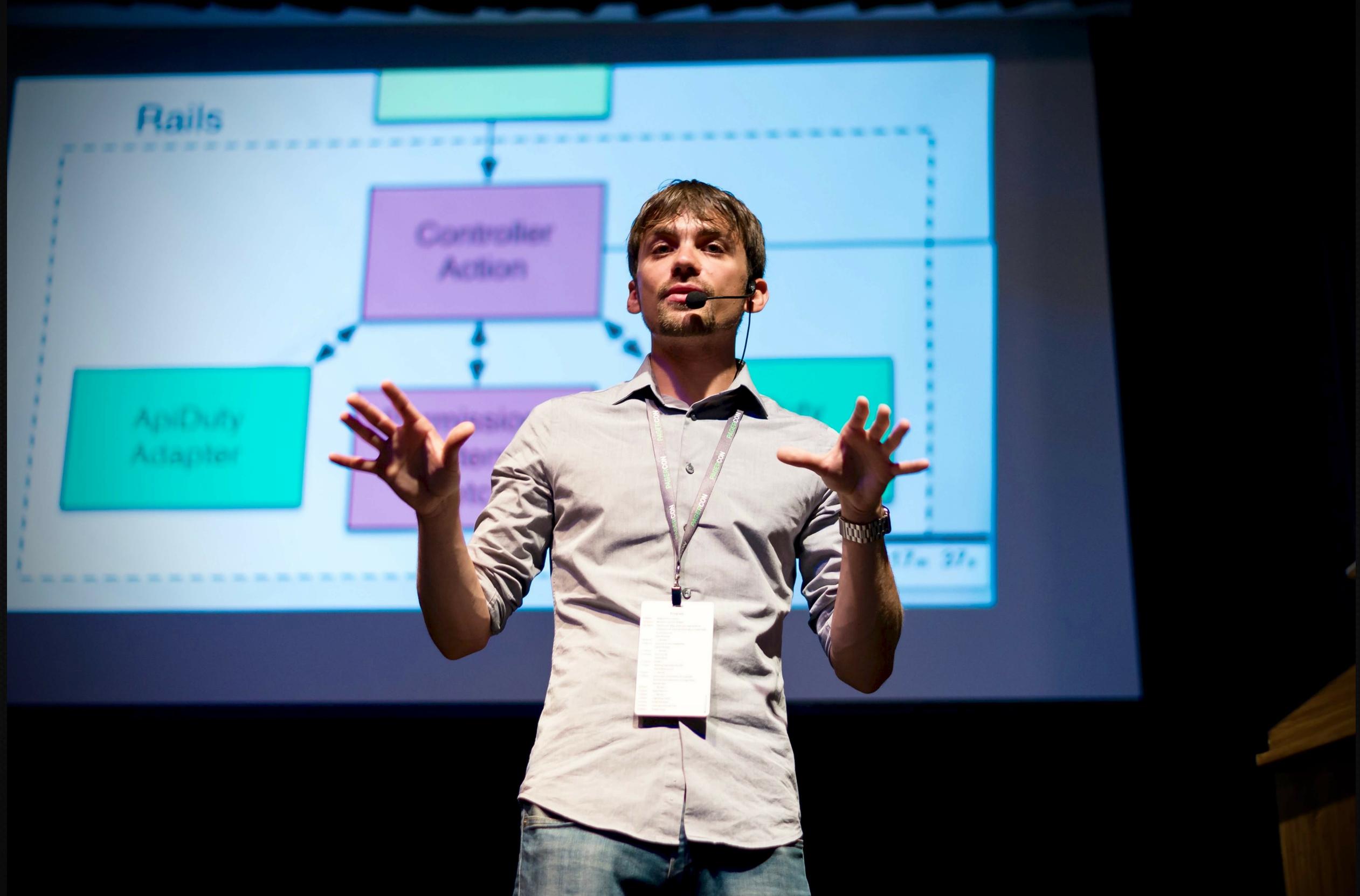
```
def self.compare_json(old, new, consider_ordering = false)
  return true if old == new

  old, new = *[old, new].map(&JSON.method(:parse))

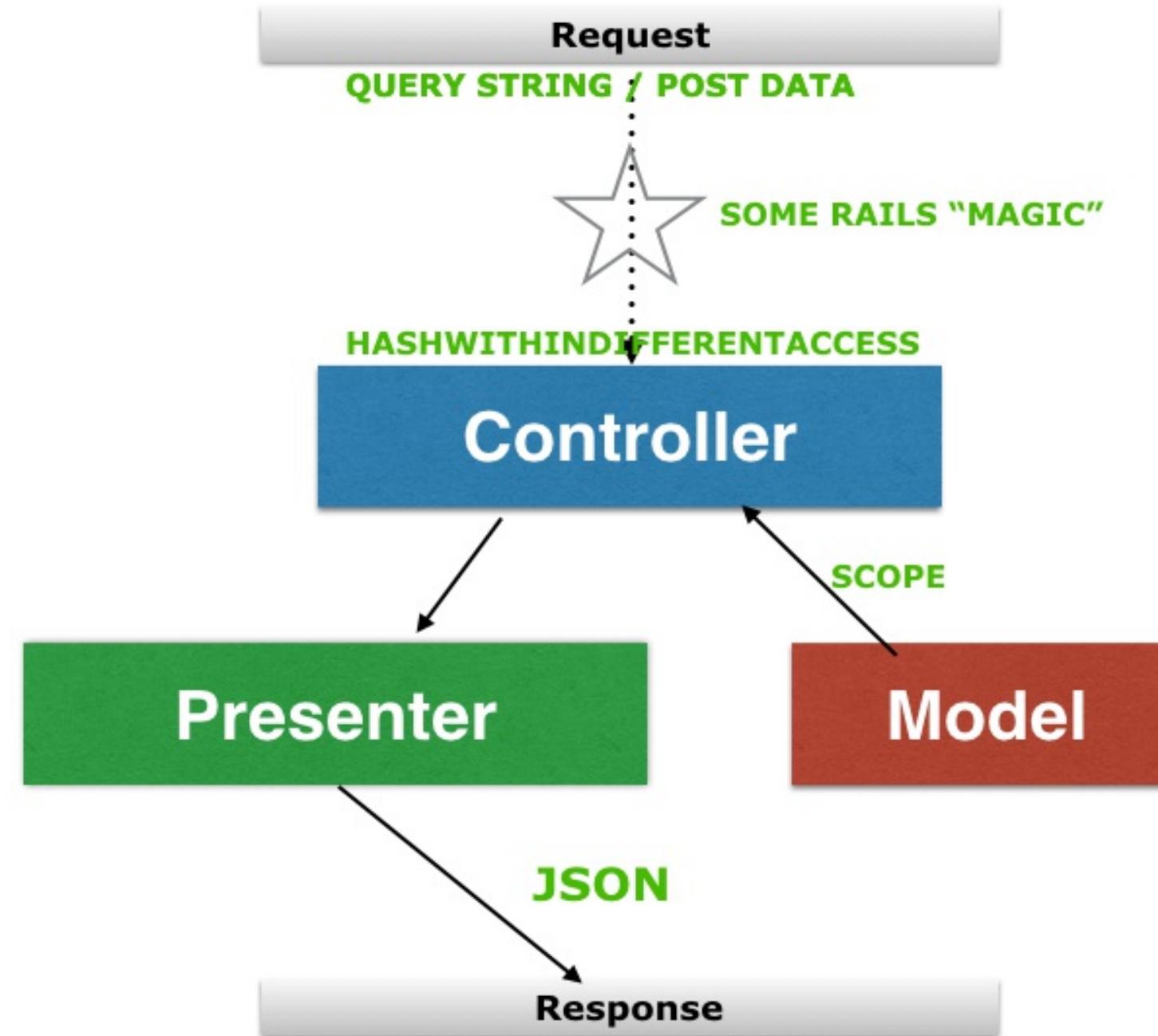
  unless consider_ordering
    # Recursively sort hash keys so the diff ignores ordering
    original = self.sort_recursively(original)
    modified = self.sort_recursively(modified)

    # Quick test, in case the sorted hashes are equal
    return true if original == modified
  end
  return false
end
```

Connect



Why Adapt?





Hello,

Thanks for your continued interest and use of the PagerDuty API.

We're writing to let you know that as of November 4th, 2015, we will be discontinuing the ability to authenticate against the API using HTTP Basic Authentication (PagerDuty username and password), which has [been deprecated since mid-2014](#).

We are doing this to improve security of PagerDuty accounts by limiting the spread of PagerDuty user passwords, which are difficult to audit and revoke should they become compromised.

Additionally, the need to authenticate a user via HTTP Basic Authentication imposes a performance penalty on every API request due to the robust hashing techniques we use to store and validate user passwords. This is done by design to make brute forcing passwords computationally infeasible.

Switching away from basic authentication will improve the security of your users' passwords and make your API requests faster.

We've identified your PagerDuty account as one that is likely using HTTP Basic Authentication for automated HTTP requests against our API. The following PagerDuty users on your account are making recurring Basic Authentication requests:

`${USER_NAME} ${USER_EMAIL}`

If you don't recognize any of these users as being used with a PagerDuty integration or don't believe you are using HTTP Basic Authentication, please let us know and we can investigate further to help identify your integration.

If you are using a third-party client, tool, or library to connect to PagerDuty, check with the vendor/maintainer for an updated version that supports API Token Authentication. If you maintain such an integration internally, your development team can refer to our authentication documentation at <https://developer.pagerduty.com/documentation/rest/authentication> to learn more about using API Tokens.

Should you have any questions, don't hesitate to reach out. We recommend making the necessary changes well in advance of the November 4th deadline to provide ample time for testing and prevent any interruption in service.

Instead, we've focused on refining, standardizing, and simplifying the patterns that appear across our API. We've gotten a lot of great feedback over the years and are addressing a lot of the challenges developers have faced. And we're providing developers with better tools to build on and interact with the API, starting with a brand new [developer portal](#) and the ability to play with API v2 right from a web browser.

Here are just a few of the highlights:

- We've simplified our API URLs. Instead of needing to know an account's subdomain, just point all your requests to `api.pagerduty.com` and we'll figure out what account you wanted from the authentication. No more `/api/v1` root path either — it's as simple as `/incidents` or `/schedules`!
- We've standardized all of our resource types so that a user is always a user and a service is always a service, no matter where they appear in the API.
- We're launching a new [/oncalls API](#) that makes accessing information about who is on call, when, and for what more straightforward than it's ever been.
- We're adhering even more closely to [REST architectural constraints](#). HTTP verbs work as you'd expect, and URIs are provided for every resource.
- We've given every resource a `summary` that describes what it is. It's your one-stop-shop for getting text to name and identify a resource.
- Across the board, we've taken steps to simplify or eliminate things in the v1 API that weren't strictly necessary, weren't being used, or caused developer confusion.

Our [v1 to v2 migration guide](#) has all the details about what sorts of things have changed, and our new [reference documentation](#) backed by a comprehensive [Open API Initiative Specification](#) contains complete specifics about all of our supported endpoints.

The [PagerDuty Events API](#) is versioned separately and is unaffected by these changes. All of your PagerDuty integrations will be able to continue sending events to PagerDuty as they always have.

Market

The screenshot shows the 'PagerDuty Developer - Home' page. The left sidebar contains a navigation menu with sections like REST API Documentation, General Information, Alerts, Escalation Policies, Incidents, Log Entries, Maintenance Windows, Reports, Schedules, Services, and Users. Each section has a list of sub-links. The main content area features a large heading 'PagerDuty API' and three steps for getting started: Step 1 (Create an account), Step 2 (Extend PagerDuty), and Step 3 (Share your work). Below these steps are sections for 'Want to build something without coding?' (using Zapier and webhooks) and 'Looking for existing integration guides?' (with links to monitoring system integration guides).

PagerDuty API

Step 1

If you don't have one already, [sign up for a PagerDuty account](#). Our trial accounts are free to try. Play around. Send yourself a few alerts.

Step 2

Extend PagerDuty to fit your operational needs:

- Build an integration with your monitoring tool using our [Integration API](#)
- Trigger actions when an incident is created / modified with [webhooks](#)
- Integrate PagerDuty and your internal systems with our extensive REST API
- Extend an existing [tool](#), build on a [code sample](#) or use one of our customer contributed [libraries](#)

Step 3

Share what you've made! Email us at developers@pagerduty.com to let us know how you're building out your operation performance platform.

Want to build something without coding?

With [Zapier](#) and [PagerDuty webhooks](#), you can automate workflow with hundreds of apps.

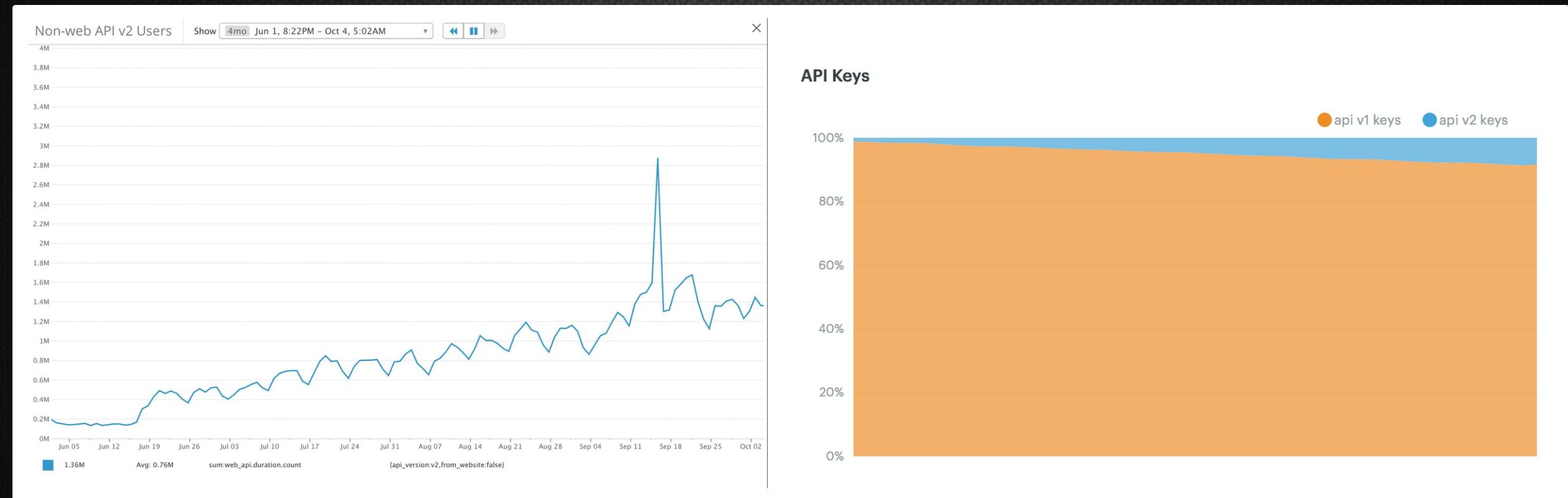
Looking for existing integration guides?

Hop over to our [monitoring system integration guides](#) for information on how to directly integrate PagerDuty with [Nagios](#), [New Relic](#), [Pingdom](#), [Server Density](#), [Splunk](#), and a bunch more. If your monitoring system isn't directly supported, you can come back here and use our API to get events into PagerDuty.

PagerDuty Developer - Home

<https://v1.developer.pagerduty.com/>

Demonstrate
webinars
sales demos
live demos
customer success



What's next?

Deprecation

Timeline: API V2 Rollout and transition from V1 - Platform - Confluence

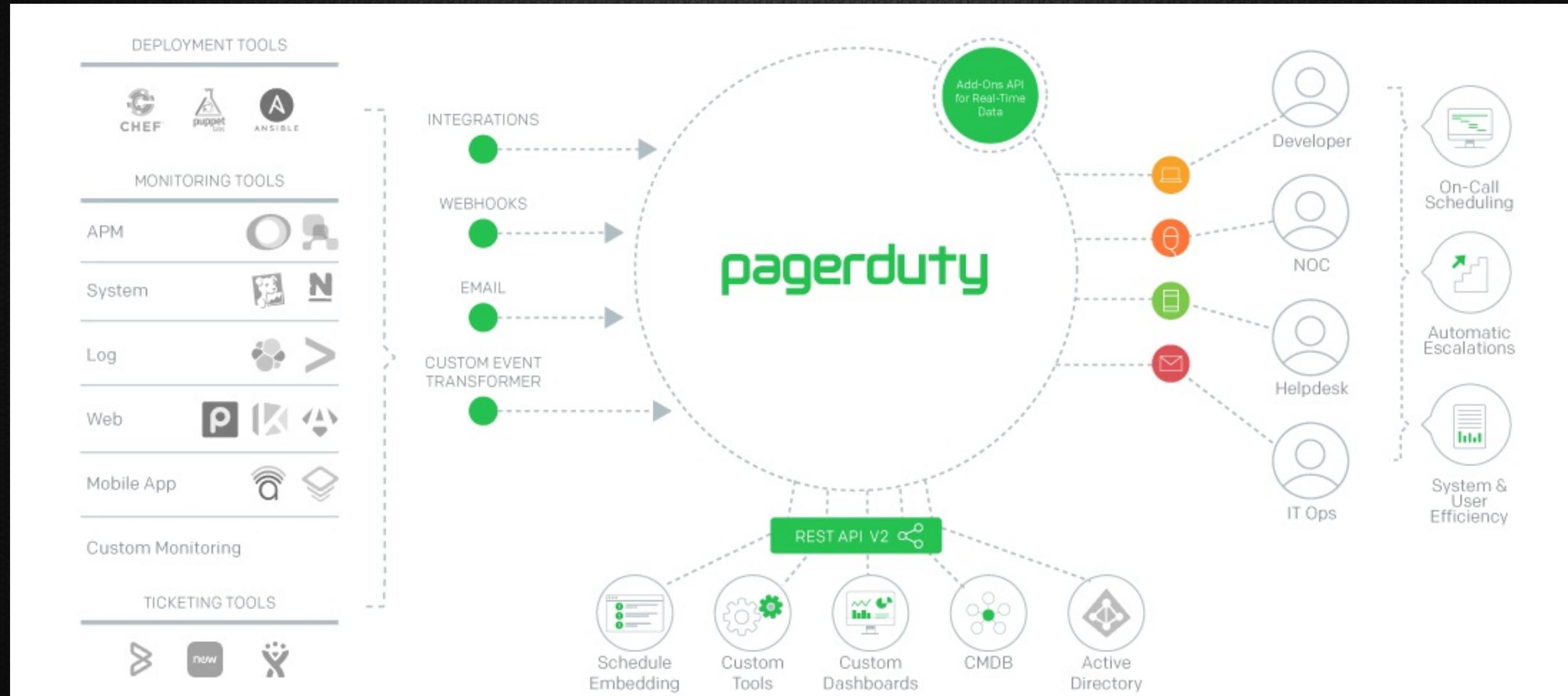
The screenshot shows a Confluence page titled "Timeline: API V2 Rollout and transition from V1 - Platform - Confluence". The page has a dark header with the pagerduty logo, a search bar, and a user icon. On the left is a sidebar with icons for dashboard, spaces, people, and settings. The main content is a table with the following data:

| Topic | Task | Owner | Status | Start Dt | End Dt |
|-----------------------|---|-------------------------|-------------|-----------|------------------|
| | IOS app updates submitted to Apple and released | Mobile | Completed | 6/15/2016 | depends on Apple |
| | Android app updates released | Mobile | Completed | | |
| Training for v2 | C.S Training material is created | Customer Support (Lisa) | Completed | | before 6/15 |
| Launch Activities | Announce v2 as GA & point out all the new aspects (add-ons, meta data) | Marketing / Hayes | Completed | | Around 6/15 |
| | v2 docs go to developer.pd.com / Move community to new docs | Platform | Completed | | Around 6/15 |
| | Default API for new integrations becomes V2 | Platform | Completed | | 6/15 |
| Sunsetting Activities | Announce sunset plan for v1 | ?? | Not started | | |
| | No longer allow the creation of V1 keys | Platform | Not started | | |
| | Email account owners with an API key & anyone who has created an active API key | Customer Support? | Not started | | |
| Sunset v1 | Support no longer answers questions about v1 | Customer Support | Not started | | |
| | Warning email to any active v1 API users | Customer Support | Not started | | |
| Disable | Disable API v1 | Platform | Not started | | |

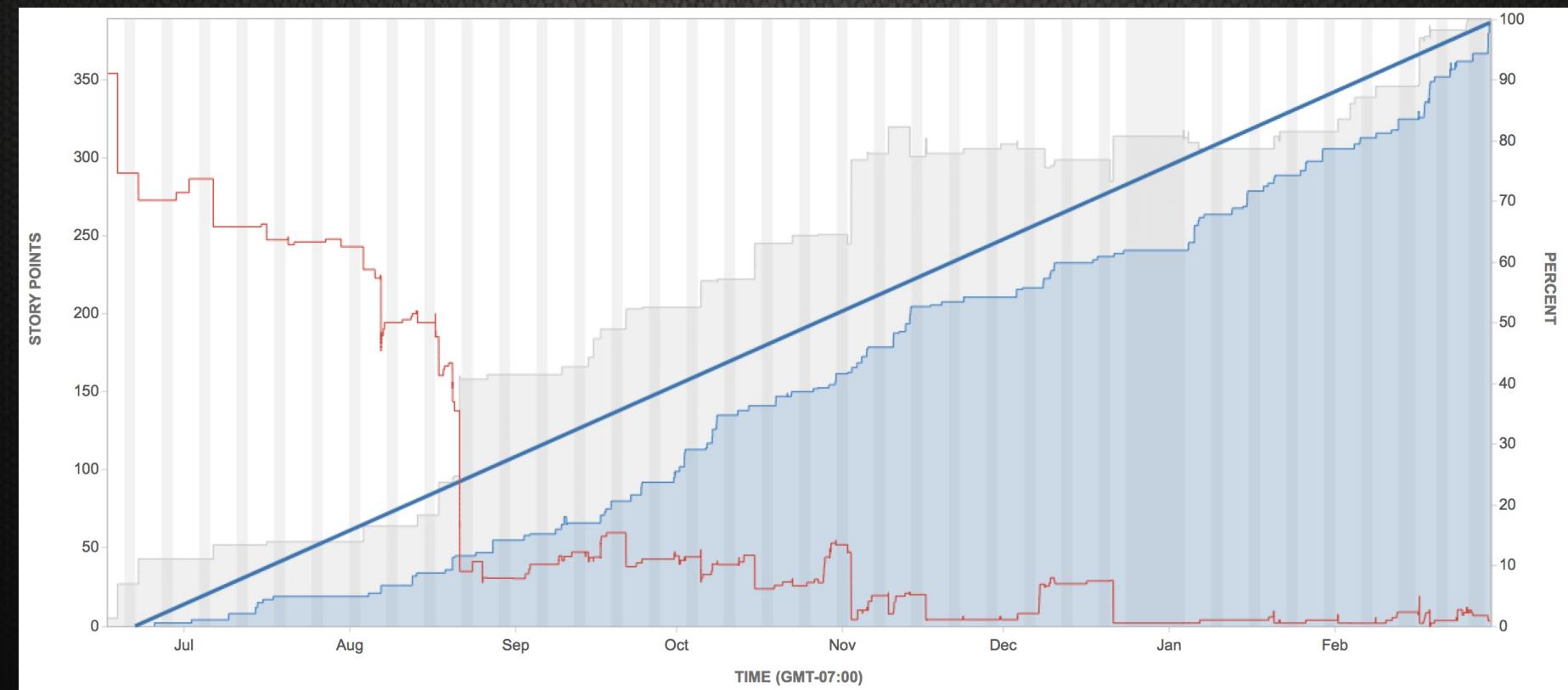
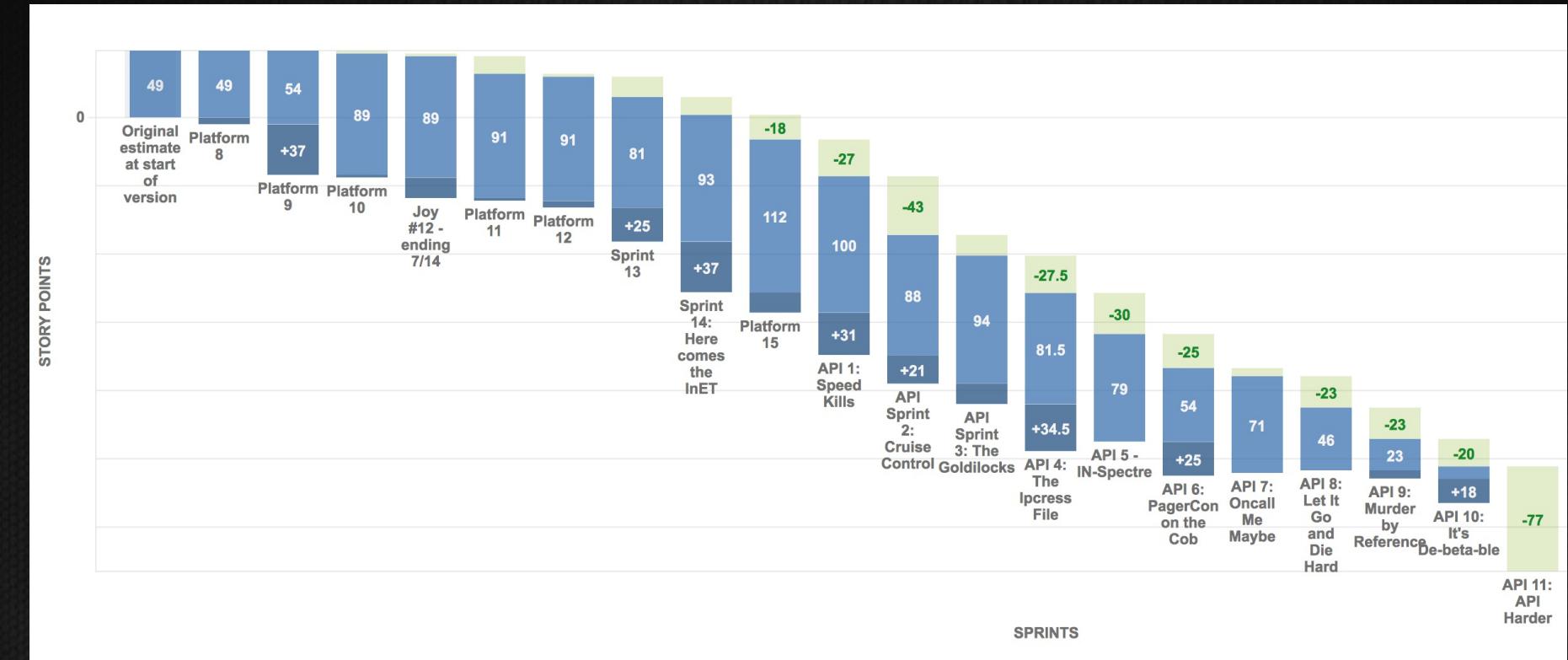
BEST FRIENDS FOREVER CO.

BFF

The Platform Strategy



<https://www.pagerduty.com/features/platform-extensibility/>



It takes a village



A woman with long brown hair, wearing a yellow patterned dress, stands on a rocky beach at night. She is looking towards a large, white, textured object that resembles a dead whale or seal lying on the sand. The scene is dimly lit by the moon and stars.

Fin?

developer.pagerduty.com

Appendix

Image Credits

- "Steve Rice" By Steve Rice © 2016 (Own work), all rights reserved
- "Stand on the Shoulders of Giants" By Samir Eberlin [CC0], via Wikimedia Commons —
<https://commons.wikimedia.org/wiki/File:Turtles.on.a.stone.in.brazil.jpg>
- "Make Decisions." By SeppVei (Own work) [CC0], via Wikimedia Commons —
https://commons.wikimedia.org/wiki/File:Bicycle_shed.JPG
- "Be Pragmatic" By Liveon001 © Travis K. Witt (Own work) [CC BY-SA 3.0 (<http://creativecommons.org/licenses/by-sa/3.0>) or GFDL (<http://www.gnu.org/copyleft/fdl.html>)], via Wikimedia Commons —
https://commons.wikimedia.org/wiki/File:Farfalle_Pasta.JPG
- "Dogfood. Everything." Original source not found. Used without permission.
- "Keep it maintainable" By David Jones from Isle of Wight, United Kingdom [CC BY 2.0], via Wikimedia Commons —
https://commons.wikimedia.org/wiki/File:Mike_O%27Callaghan-Pat_Tillman_Bridge_construction.jpg
- "Pour over logs" by Timothy Marsee [CC BY 2.0], via Flickr —
<https://www.flickr.com/photos/tmarsee530/9899684046>
- "BFF" By Guillaume Paumier (Own work) [CC BY-SA 3.0 (<http://creativecommons.org/licenses/by-sa/3.0>)], via Wikimedia Commons —
https://commons.wikimedia.org/wiki/File:Best_Frends_Forever_-_Golden_Gate_bridge_guard_rail_166.jpg
- "Fin?" Copyright 1984 Warner Bros. Pictures, via Cinema Autopsy —
<https://blog.cinemaautopsy.com/2013/04/02/film-review-the-neverending-story-1984/>