

13TH FEB. ANDROID REJECT CONFERENCE 2019/2

BASED ON DROIDKAIGI 2019 APP

**FLEXIBLE, CONFIGURABLE &
SEMI-INTERACTIVE CI/CD**

Jumpei Matsuda @red_fat_daruma

- ▶ GitHub: jmatsu
- ▶ Software Architect @DeployGate
- ▶ A member of @DroidKaigi
 - ▶ Set up CI/CD of the App
 - ▶ Other Zatsuyou
- ▶ Consists of craft beer (especially Belgium) and Japanese sake.



DroidKaigi : 2019

Feb 07_(Thu) - 08_(Fri) @Bellesalle Shinjuku Grand

Thank You for Your Work and Those Who Attended!

- ▶ About 1000 people joined, over 90 speakers talked.
- ▶ We have already published session videos!
 - ▶ Check it out! <https://bit.ly/2N1DL1b>

Jumpei Matsuda  

DroidKaigi 2019 finished 4 days ago

Sorry, your **session submission has been declined.**

We are engineers so we should go with lazy methods.

In this talk, I will explain

- What can be semi- and/or fully- automated
- How to automated stuff like
- Introduce Danger
- Report differences between two apk files
- Import design assets to vector drawables or png files
- License files
- Proguard rules
- Report results of static analysis

DECLINED ~~SUCCESSFULLY~~

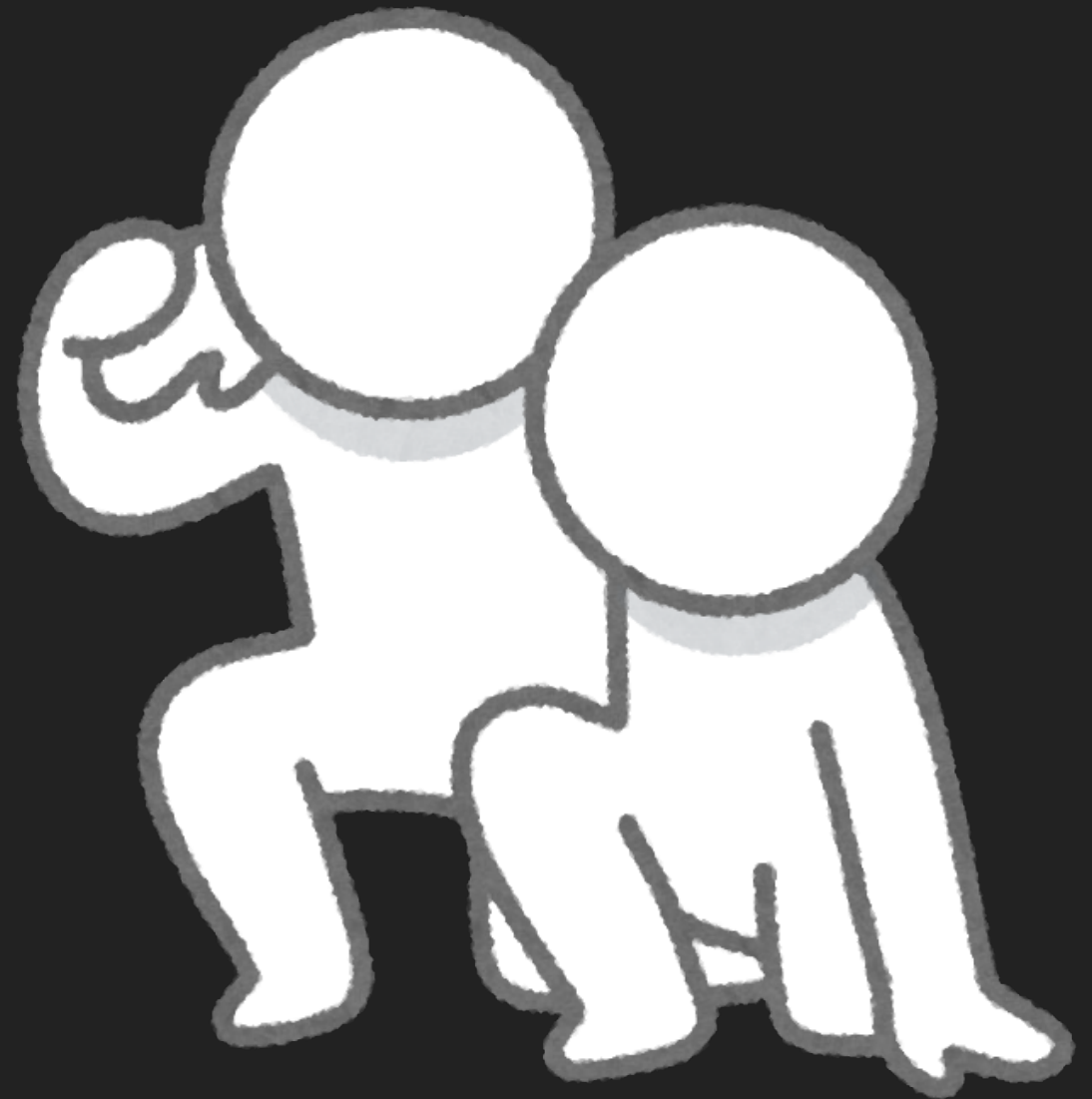
What I Wanted Talk About

- ▶ Automate ~~everything~~ many things
 - ▶ Report tests, stats
 - ▶ Import SVG assets w/ optim.
 - ▶ Import any other materials
- ▶ Keyword:
 - ▶ Be lazy, mistake-less development



Then I Noticed...

- ▶ I can play with the official app
 - ▶ Brush up my ideas
 - ▶ Apply what I want to talk about
 - ▶ ~~Great experiment field~~



Official Application



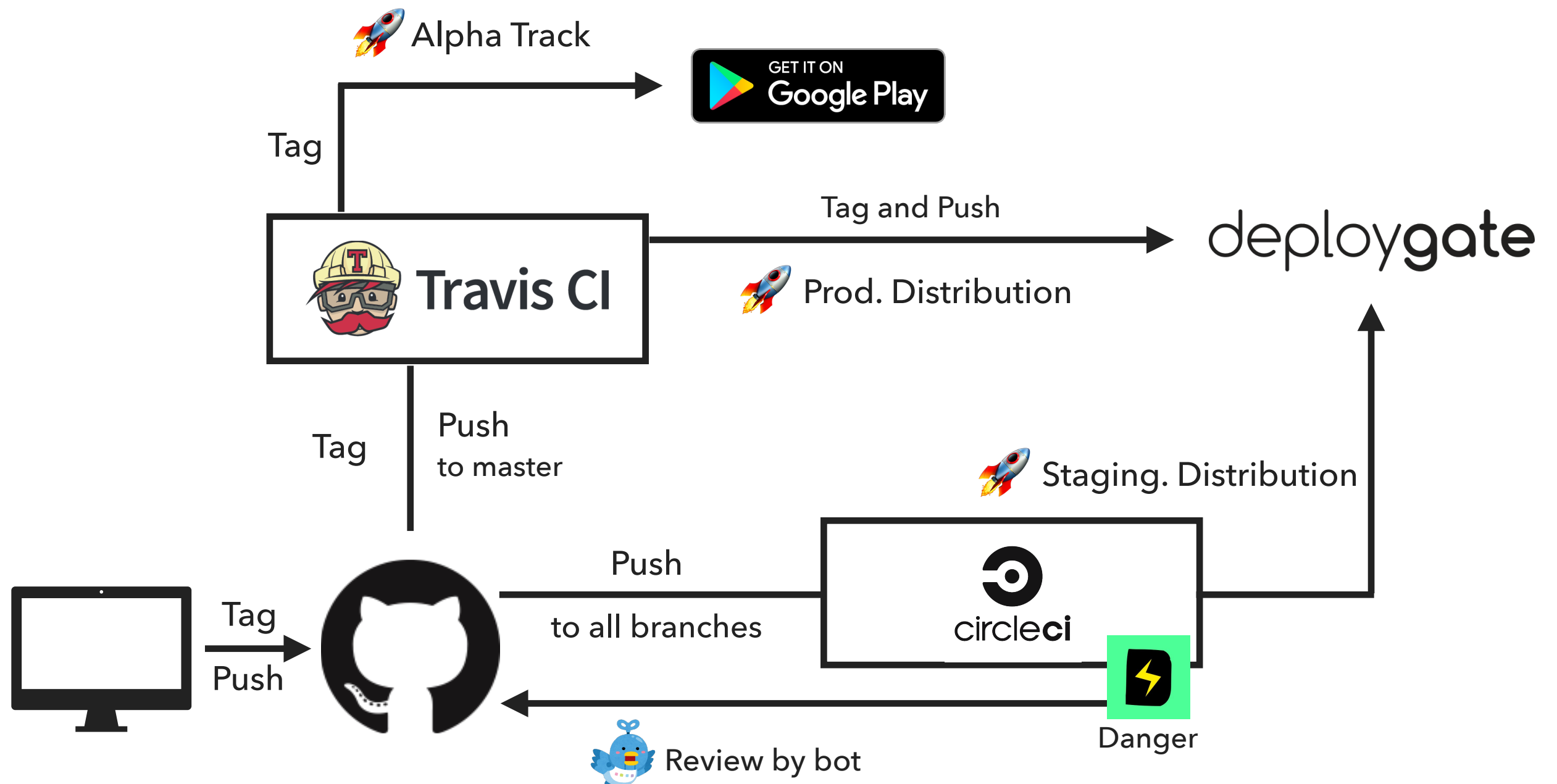
Active 1–3 Reviewers per Day



Over 145 Contributors

Trust GitHub API

No Design Sense!



Built by

- ▶ CI
 - ▶ CircleCI (Especially for PR)
 - ▶ TravisCI (Specific branch only)
- ▶ Messaging system
 - ▶ Danger
- ▶ Deployment
 - ▶ DeployGate (Distribution)
 - ▶ Google Play Store (Alpha, Prod)



deploygate



What I Will Talk About

- ▶ Desired CI/CD for DK 2019
 - ▶ Properties
 - ▶ Objectives/Solutions
 - ▶ Flexible & Configurable
 - ▶ Semi-Interactive Flow
 - ▶ Deployment Pipeline



Generally Speaking,

Desired CI/CD?

Generally Speaking,

Desired CI/CD?

- ▶ Depends on your project
- ▶ Let's break down the properties of DroidKaigi App!

Properties

- ▶ Who will be joining
- ▶ What role should be satisfied
- ▶ What are important for everyone with CI/CD

These are selected properties for DK App and this presentation.

Please keep in mind that your project may be differ.

Who Will Be Joining

- ▶ In short, ANYONE.
- ▶ No Filter!
 - ▶ Natural language
 - ▶ Programing language
 - ▶ iOS engineers joined in fact
 - ▶ From newbies, beginners to experts.

Properties

- ▶ Who will be joining
 - ▶ Anyone!
- ▶ What role should be satisfied
- ▶ What are important for everyone and CI/CD

What Role Should Be Satisfied

- ▶ Be an example for everyone
 - ▶ Be useful, reusable, and readable
- ▶ Stable but fast
 - ▶ Unstable and/or slow CI may break your heart

Properties

- ▶ Who will be joining
 - ▶ Anyone!
- ▶ What role should be satisfied
 - ▶ Be useful, re-usable, and safe examples
- ▶ What are important for everyone and CI/CD

What Are Important for Everyone and CI/CD

- ▶ Save reviewers'/reviewees' time and labour.
 - ▶ They have their own life.
- ▶ Keep contributors' motivation.
 - ▶ You would like to enjoy contributing, right?

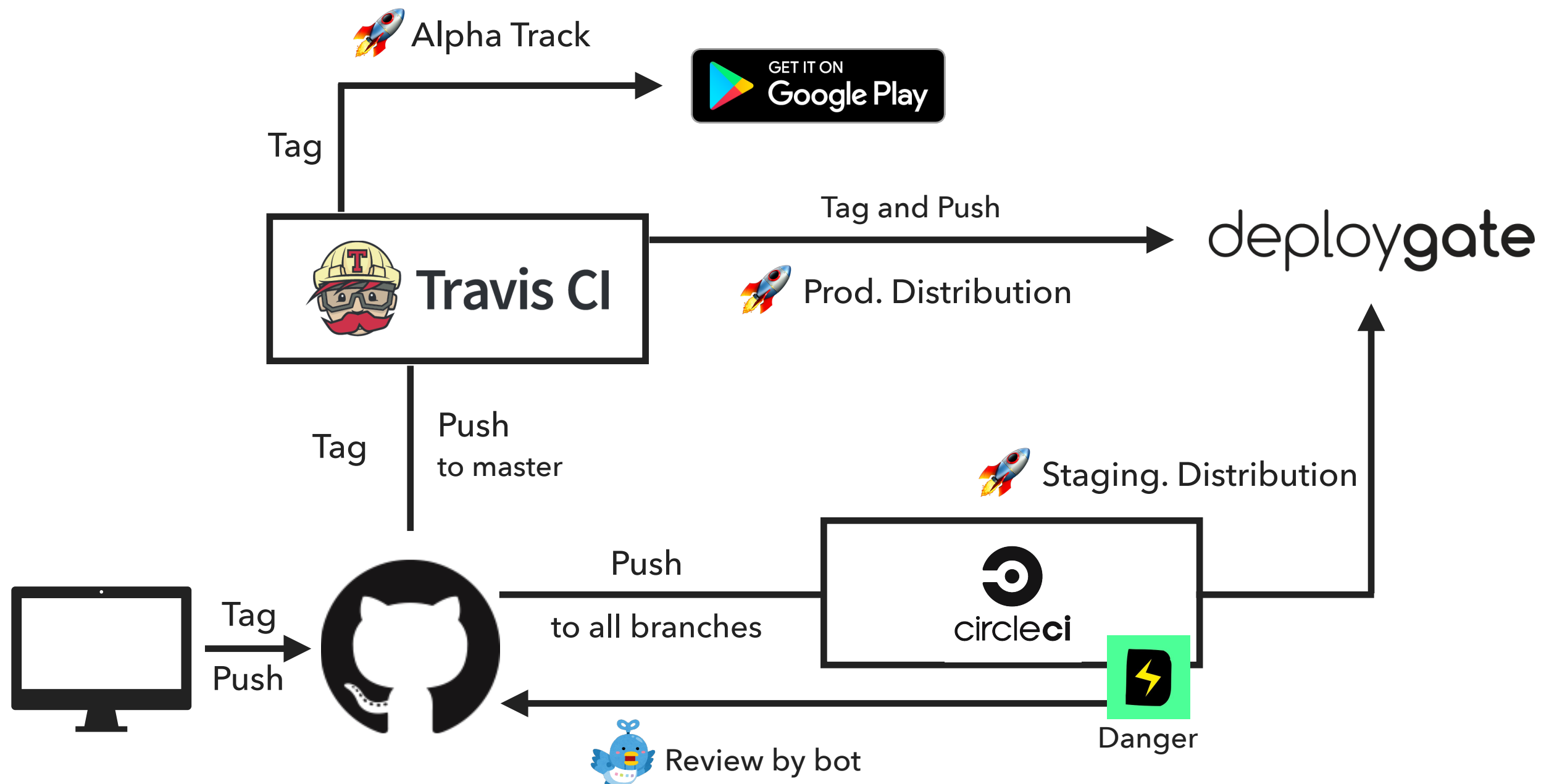
Properties

- ▶ Who will be joining
 - ▶ Anyone!
- ▶ What role should be satisfied
 - ▶ Be useful, re-usable, and safe examples
- ▶ What are important for everyone and CI/CD
 - ▶ Save reviewers' and/or contributors' time and mind

Objectives/Solutions

- ▶ Be stable
- ▶ Be environment agnostic
- ▶ Provide semi-interactive flow
 - ▶ Prompt to complete code-review preparation by contributors themselves.
 - ▶ Also solve: never make reviewers say nonsense reviews like **Please remove unused imports**.
- ▶ Give useful deployment pipeline experiences.

No Design Sense!



Objectives/Solutions

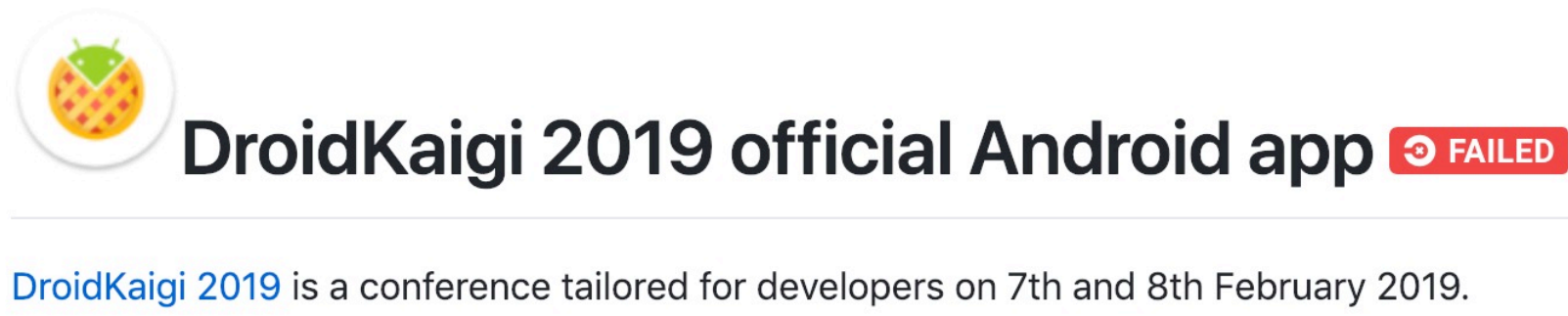
- ▶ Be stable
- ▶ Be environment-specific-less
- ▶ Provide semi-interactive flow
- ▶ Give useful deployment pipeline experiences.

Objectives/Solutions

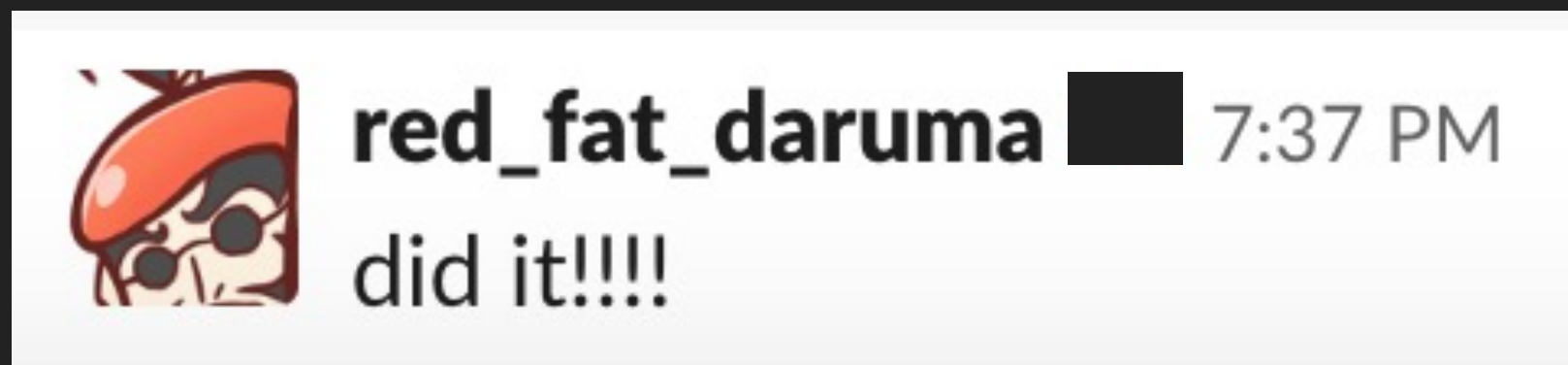
- ▶ Be stable
- ▶ Be environment-specific-less
- ▶ Provide semi-interactive flow
- ▶ Give useful deployment pipeline experiences.

Stable?

- ▶ Failed since yesterday



- ▶ I've fixed it ~~several~~ 1 and a half hour ago 😂



Anyway!

Stable Realized by Flexible

- ▶ No failure is required anyway!

Stable Realized by Flexible

- ▶ ~~No failure is required anyway!~~
- ▶ Flexible with failures.
 - ▶ What's non-flexible with failures?
 - ▶ Fail CI anyway if any task fails like ``set -e`` in the shell.
 - ▶ CI Steps are sequential, so...
 - ▶ Make CI/CD fail with depending on the commands.

Stable Realized by Flexible

- ▶ Let's categorize commands by their aspects
 - ▶ Lints
 - ▶ Reports
 - ▶ Testing
 - ▶ Assembling
 - ▶ Deployment

Stable Realized by Flexible

- ▶ Take a look at what to be assured
 - ▶ Lints → Code's quality but not product's
 - ▶ Reports
 - ▶ Testing → Product's quality (IOW, for QA)
 - ▶ Assembling
 - ▶ Deployment → For QA and DX but on demand usage

Stable Realized by Flexible

- ▶ So, how should failures be propagated?
 - ▶ Lints → Accept failures
 - ▶ Reports
 - ▶ Testing → Never accept failures
 - ▶ Assembling
 - ▶ Deployment → Depends on branches

Stable Realized by Flexible – Examples

- ▶ Separate assembling and testing into independent jobs
 - ▶ It's also a part of structuring deployment pipeline
- ▶ Use `*continue*` from gradle options
 - ▶ `./gradlew lintDebug testDebugUnitTest ktlint --continue --offline`
- ▶ Use `retry-command` to execute unstable commands
 - ▶ To download dependencies in DroidKaigi app

Objectives/Solutions

- ▶ Be stable
- ▶ Be environment agnostic
- ▶ Provide semi-interactive flow
- ▶ Give useful deployment pipeline experiences.

Environment Agnostic by Configurable

- ▶ Be independent on specific CI/CD services
- ▶ Separate implementations for each platforms

Environment Agnostic by Configurable

- ▶ Be independent on specific CI/CD services
- ▶ ~~Separate implementations for each platforms~~
- ▶ Configure inputs and dependencies before processing
 - ▶ Not a BLoC pattern but refer to it
 - ▶ Define CL inputs and injectable dependencies
 - ▶ Don't depend on any platform in the core logic

Environment Agnostic by Configurable

► Normalize common variables

```
is_travis() {
    [[ "${TRAVIS:-false}" == "true" ]]
}

is_circleci() {
    [[ "${CIRCLECI:-false}" == "true" ]]
}

is_otherwise() {
    ! is_travis && ! is_circleci
}

project_slug() {
    if is_travis; then
        echo "$TRAVIS_REPO_SLUG"
    elif is_circleci; then
        echo "$CIRCLE_PROJECT_USERNAME/$CIRCLE_PROJECT_REPONAME"
    elif [[ "${PROJECT_SLUG:-false}" != "false" ]]; then
        echo "$PROJECT_SLUG"
    fi
}
```

Environment Agnostic by Configurable

- ▶ Prepare at least two configurations (shell profiles)
 - ▶ Shared configuration & platform-specific configuration
- ▶ Shared configuration
 - ▶ Enable the custom bins: Deps \supseteq executable binaries

```
1  if [[ -z "$REPOSITORY_ROOT" ]]; then
2      export REPOSITORY_ROOT="$(git rev-parse --show-toplevel)"
3  fi
4
5  export PATH="${REPOSITORY_ROOT}/scripts:$PATH"
6  export PATH="${REPOSITORY_ROOT}/.circleci/scripts:$PATH"
```


Environment Agnostic by Configurable

- ▶ Switch config set (OO language is easier of course)
- ▶ We don't need to modify the core logic

```
use_debug_keystore() {  
    export KEY_ALIAS="$DEBUG_KEY_ALIAS"  
    export KEY_PASSWORD="$DEBUG_KEY_PASSWORD"  
    export KEY_STORE_FILE="$DEBUG_KEY_STORE_FILE"  
    export KEY_STORE_PASSWORD="$DEBUG_KEY_STORE_PASSWORD"  
}  
  
use_release_keystore() {  
    set +x  
    export KEY_ALIAS="$RELEASE_KEY_ALIAS"  
    export KEY_PASSWORD="$RELEASE_KEY_PASSWORD"  
    export KEY_STORE_FILE="$RELEASE_KEY_STORE_FILE"  
    export KEY_STORE_PASSWORD="$RELEASE_KEY_STORE_PASSWORD"  
}
```

```
java -jar bundletool.jar \  
    build-apks \  
    --bundle="$BUNDLE_FILE" \  
    --output=temp.apks \  
    --overwrite \  
    --mode=universal \  
    --ks="$KEY_STORE_FILE" \  
    --ks-key-alias="$KEY_ALIAS" \  
    --ks-pass="pass:$KEY_STORE_PASSWORD" \  
    --key-pass="pass:$KEY_PASSWORD"
```

Objectives/Solutions

- ▶ Be stable
- ▶ Be environment agnostic
- ▶ Provide semi-interactive flow
- ▶ Give useful deployment pipeline experiences.

Semi-Interactive Flow

- ▶ At least one person must be watching PRs at all times

Semi-Interactive Flow

- ▶ ~~At least one person must be watching PRs at all times~~
- ▶ Comment useful stats, reviews, & reports by bot
 - ▶ Using Danger
- ▶ Why *semi*?
 - ▶ Bot doesn't respond to you 😂 but to change-set.

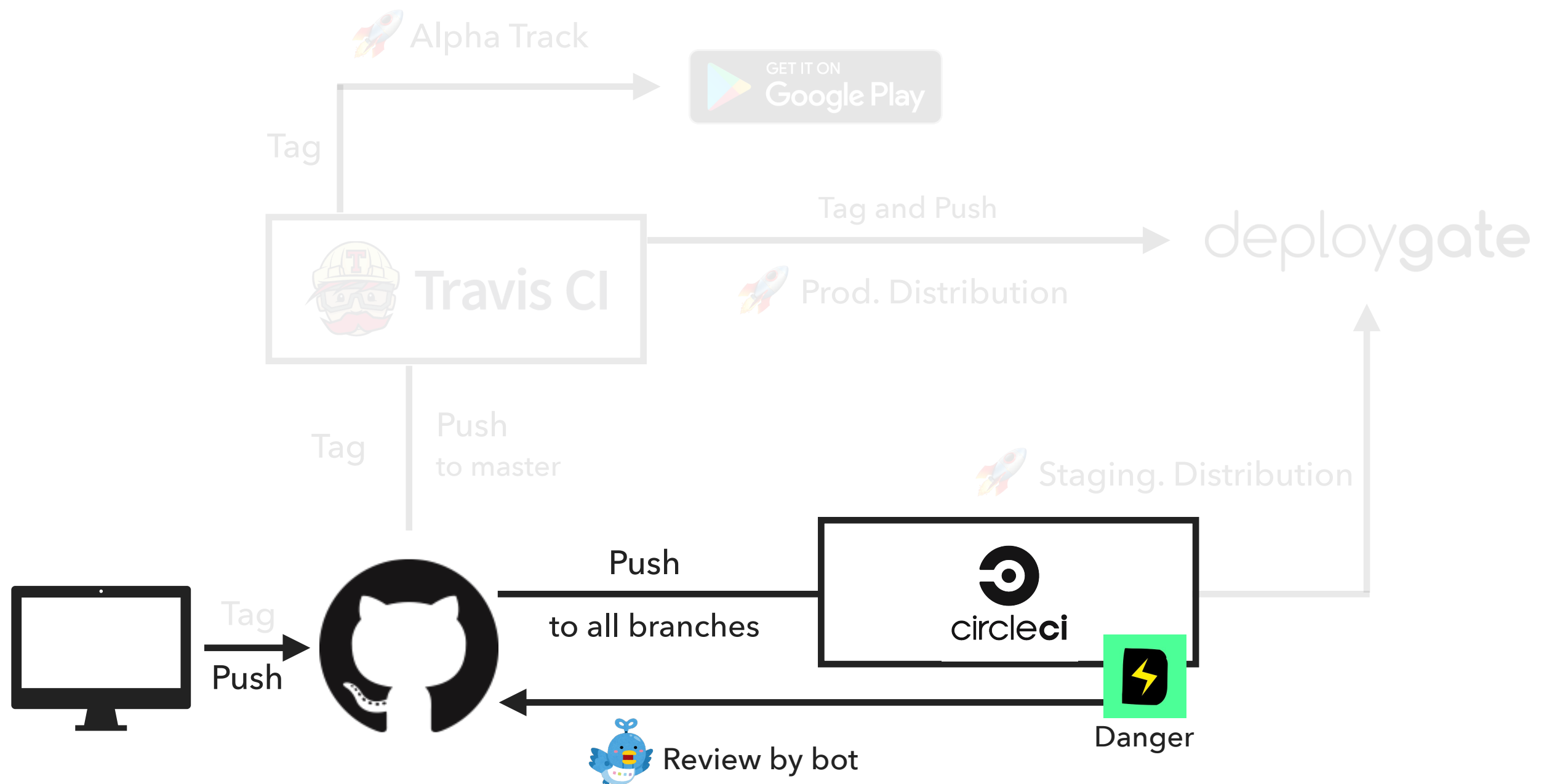
Semi-Interactive Flow

- ▶ @jmatsu-bot
- ▶ Talks through Danger, etc.
 - ▶ Checkstyle reports
 - ▶ Lint reports
 - ▶ JUnit reports
 - ▶ Deployment status




いらすとや

Semi-Interactive Flow ~PR-Aware~



Semi-Interactive Flow – orta/danger-junit

► Comment failures in JUnit style reports



jmatsu-bot commented on Jan 11

Collaborator + 😊 ...

	2 Errors
🚫	Tests have failed, see below for more information.
🚫	Please check the artifacts of CircleCI. Test reports would be available there.

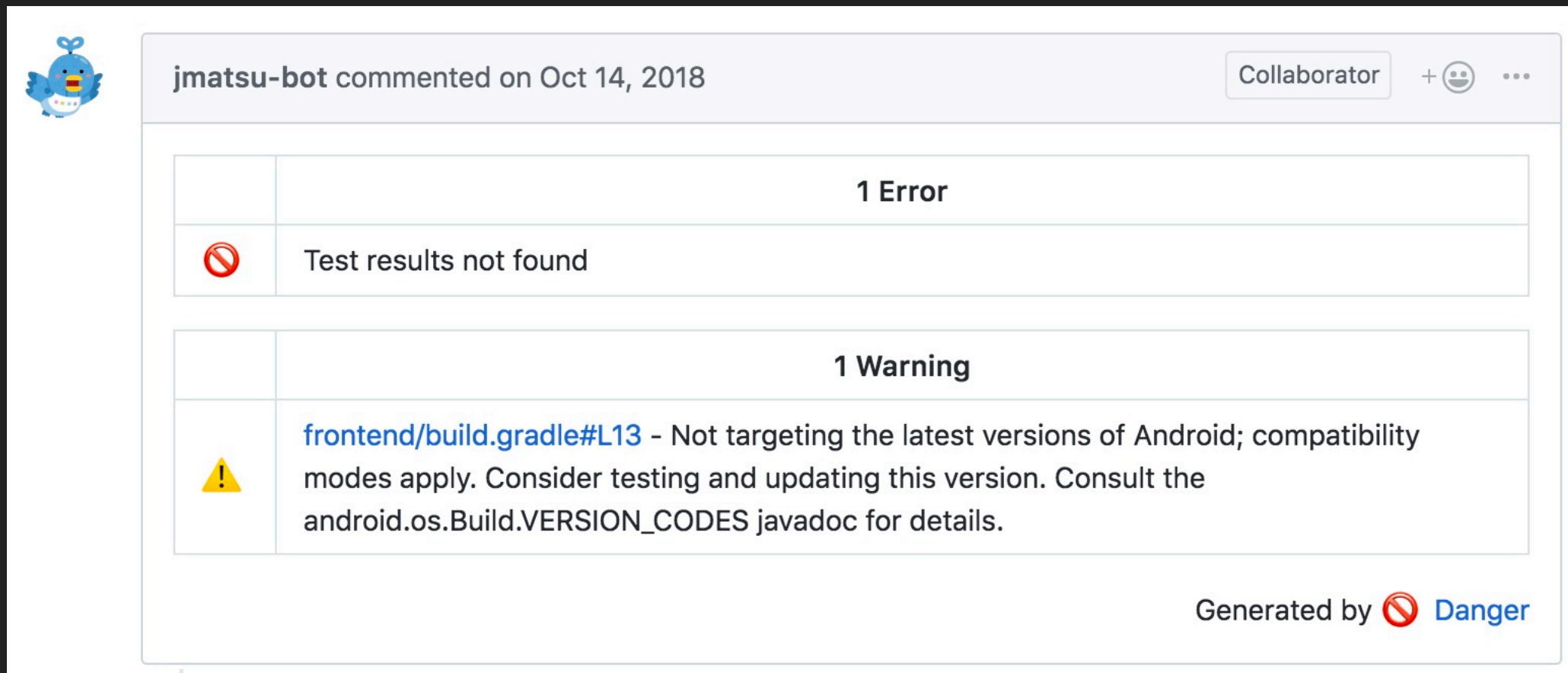
Tests:

Name	Classname	Time
loadingState	io.github.droidkaigi.confsched2019.session.ui.store.SessionContentsStoreTest	0.1s

Generated by 🚫 Danger

Semi-Interactive Flow – loadsmart/danger-android_lint

- ▶ Comment issues based on Android Lint results



The screenshot shows a GitHub comment from the bot 'jmatsu-bot' dated October 14, 2018. The comment is categorized as a 'Collaborator' and includes a '+ 🤖 ...' icon. The content of the comment is structured as follows:

1 Error	
🚫	Test results not found

1 Warning	
⚠️	frontend/build.gradle#L13 - Not targeting the latest versions of Android; compatibility modes apply. Consider testing and updating this version. Consult the android.os.Build.VERSION_CODES javadoc for details.

Generated by 🚫 Danger

Semi-Interactive Flow – jmatsu/danger-checkstyle_reports

- ▶ Comment issues based on checkstyle (, and ktlint) results

...

...

@@ -5,14 +5,18 @@ import androidx.navigation.Navigation

5

5

import com.google.android.gms.oss.licenses.OssLicensesMenuActivity

6

6

import com.xwray.groupie.Section


7

7

import io.github.droidkaigi.confsched2019.about.R


8

+ import io.github.droidkaigi.confsched2019.ext.android.changed




jmatsu-bot 9 days ago

Collaborator



Unused import




Reply...

Unresolve conversation

jmatsu marked this conversation as resolved.


Semi-Interactive Flow – jmatsu/danger-apkstats

- ▶ Report differences between the previous and the new apks

 jmatsu-bot commented 9 days ago Collaborator + 👤 ...

Apk comparision results

Property	Summary
New File Size	8366874 Bytes. (7.98 MB)
File Size Change	+8648 Bytes. (+8.45 KB)
Download Size Change	+5313 Bytes. (+5.19 KB)
New Method Reference Count	61326
Method Reference Count Change	122
New Number of dex file(s)	1
Number of dex file(s) Change	0
New Required Features	- android.hardware.wifi (requested android.permission.ACCESS_WIFI_STATE permission, and requested android.permission.CHANGE_WIFI_STATE permission)
New Permissions	- android.permission.CHANGE_WIFI_STATE - android.permission.ACCESS_WIFI_STATE

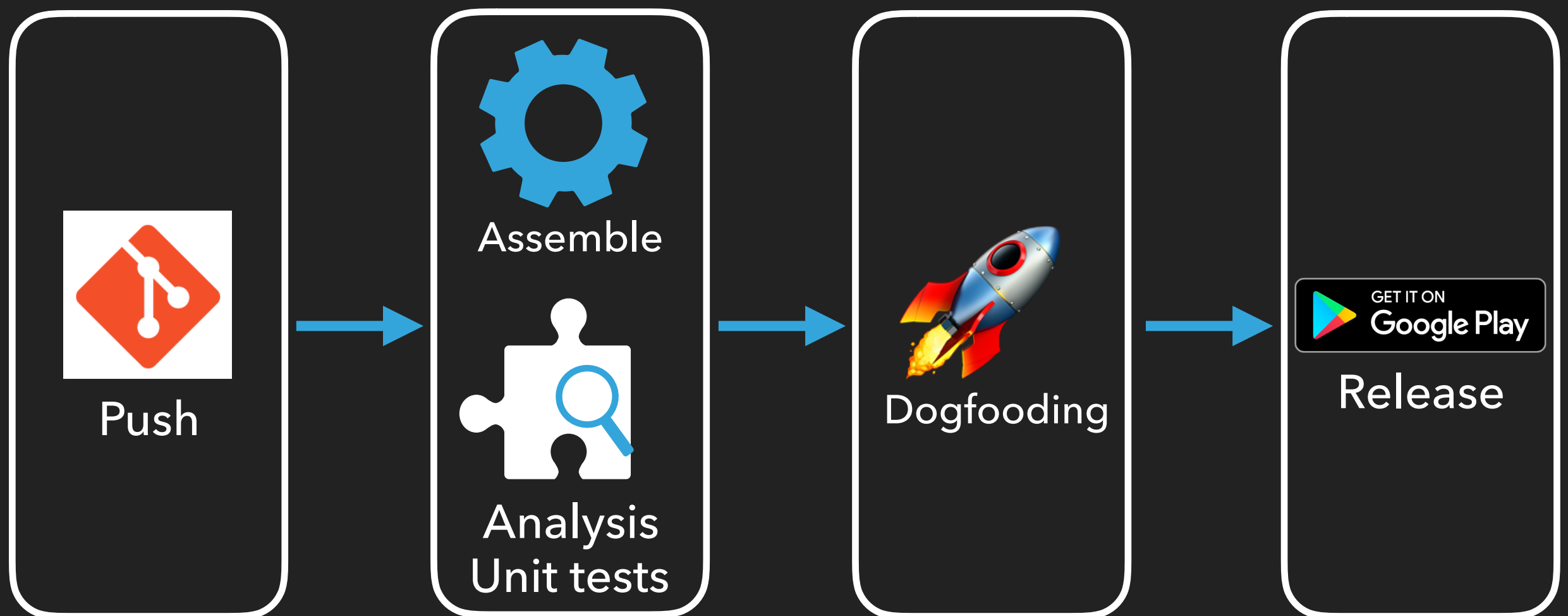
Generated by  Danger

Objectives/Solutions

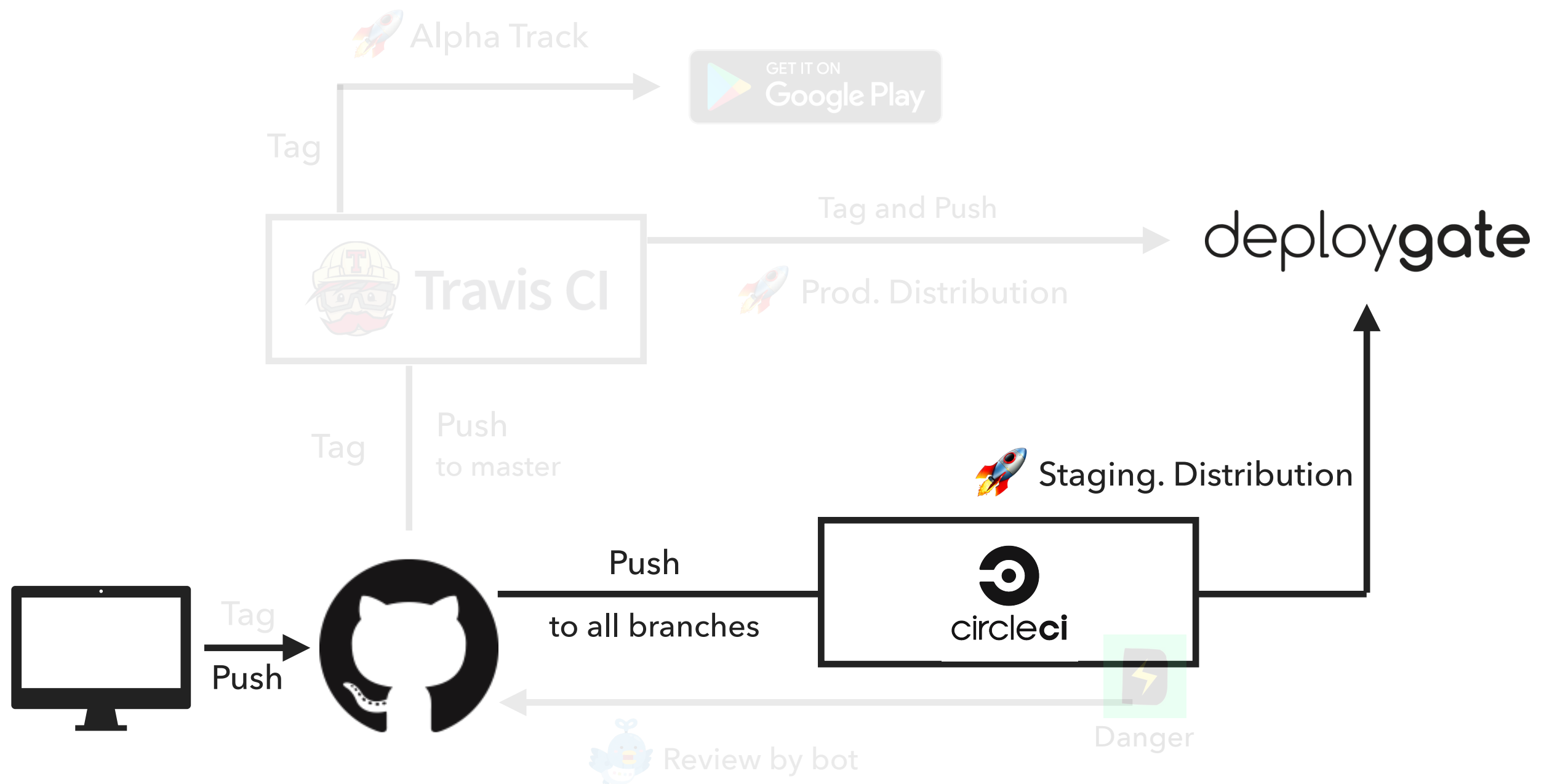
- ▶ Be stable
- ▶ Be environment agnostic
- ▶ Provide semi-interactive flow
- ▶ Give useful deployment pipeline experiences.

Deployment Pipeline

- ▶ Small and partial pipeline if say anything
 - ▶ No acceptance test but manual smoke test (dogfooding)

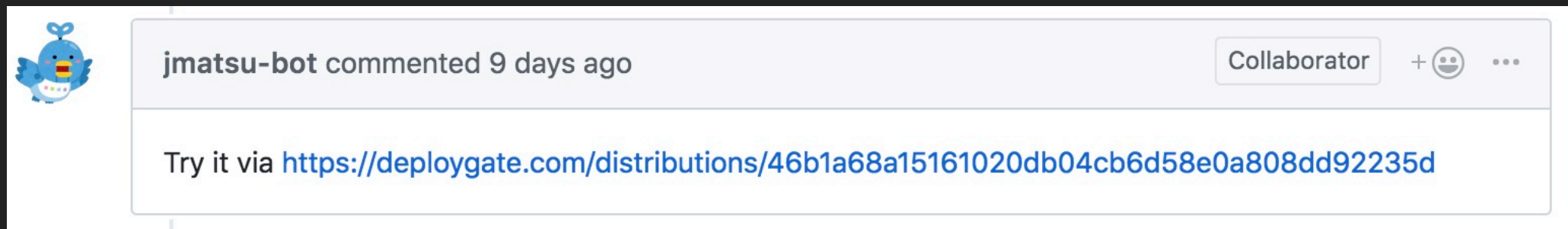


Deployment Pipeline – Dogfooding Flow

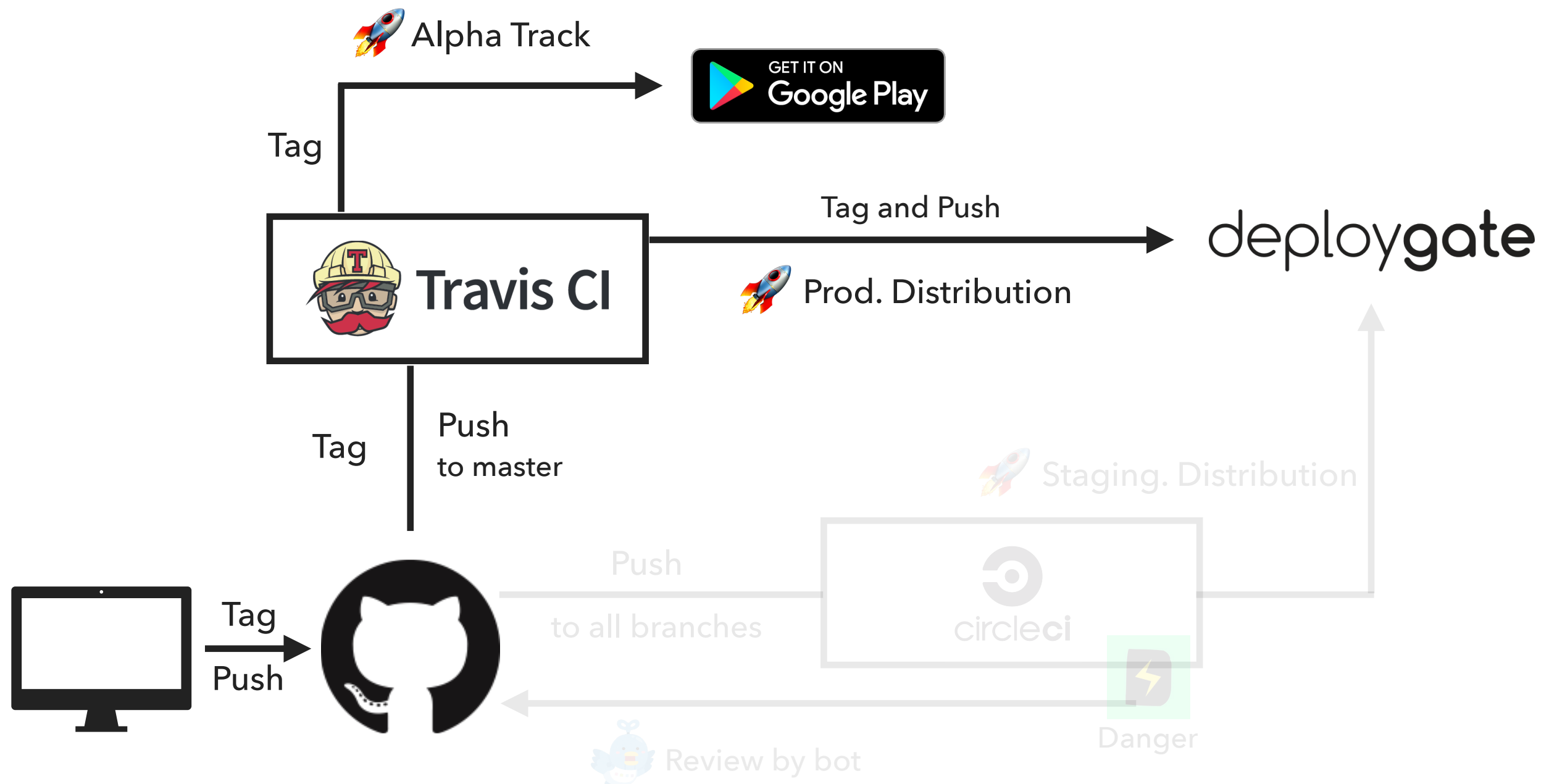


Deployment Pipeline – Dogfooding Flow

- ▶ Play with the artifact of a feature branch easily!



Deployment Pipeline – Release Flow



Deployment Pipeline – Release Flow

- ▶ Use alpha track as just a release management
- ▶ Try the production build via DeployGate
 - ▶ Didn't pin the link anywhere but it was visible on Travis :)
- ▶ Why Travis?
 - ▶ Because of secure tokens matter 😂
 - ▶ A playground to build iOS app as well

Be Flexible, Configurable, Semi-Interactive

- ▶ Be flexible with failures
 - ▶ **We** should handle CI failures instead of commands
- ▶ Be configurable by injection
 - ▶ Manipulate dependencies & platform-specific values.
- ▶ Be semi-interactive
 - ▶ Collect stats and reports, also rely on the bot.

Be Good Examples and Provide a Deployment Pipeline

- ▶ Be good examples
 - ▶ Show how to set up, how to collaborate with CI
- ▶ Give a chance to use Deployment pipeline
 - ▶ Small but enough if it's effective. (minimum req.)

Points To Be Improved

- ▶ So many many many... lets pick up several points.
- ▶ CircleCI Checks are not visible to everyone
- ▶ Use Orbs by upgrading to CircleCI 2.1
- ▶ Promote the alpha track to production safely
 - ▶ CircleCI's approval is it. If TravisCI is for contributors and CircleCI is for the release, then it made sense.

Points To Be Discussed

- ▶ Prog. Languages of commands.
 - ▶ Gradle tasks? BashScript? Ruby? kts?
- ▶ Only one contributor tweaked CI/CD configurations.
 - ▶ Did it mean *difficult*? or stable? Not sure.
- ▶ How to build; Use Gradle directly? through Fastlane?
 - ▶ If we need to build iOS app as well, Fastlane is better.

By the Way,

Improve bot name and icon #338

🔔 Open

shihochan opened this issue on Jan 10 · 2 comments



shihochan commented on Jan 10 • edited ▼



Overview (Required)

- This bot is so useful, BUT
- I think name's prefix "jmatsu-" is a little difficult to understand. Can you change more easy to understand name for DroidKaigi conference such as "droidkaigi-" ?
- I think this icon is not motivated to develop because this is free icon. I would like to propose a new icon more motivating :-)



jmatsu-bot

NOTE: Got permission from him to use this image

**I THINK THIS ICON IS NOT
MOTIVATED TO DEVELOP
BECAUSE THIS IS FREE ICON.**

Shihochan (my best friend)



Motivative Icon
Wanted

Don't Hesitate To Ask Me any Questions

- ▶ Jumpei Matsuda
 - ▶ @red_fat_daruma
- ▶ Software Architect @DeployGate
- ▶ A member of @DroidKaigi

[GitHub://DroidKaigi/conference-app-2019](https://github.com/DroidKaigi/conference-app-2019)

