7TH MAR. ANDROID TEST NIGHT #6

MAKE CI/CD MORE COMFORTABLE BEFORE/AFTER CODE TESTING

Jumpei Matsuda @red_fat_daruma

- GitHub: jmatsu
- Software Architect @DeployGate
- A member of DroidKaigi
- Consists of craft beer (especially Belgium) and Japanese sake.

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/2N1DLIb

What I Will Talk About

- Motivation
- Definition of *comfortable*
- Tips to support your testing
  - How to organize non-fragile CI
  - Save/load intermediate results of the current/previous build
  - Get advantages of DeployGate

What I Will NOT Talk About

- Technique for test code
- Testing method

What I Will Talk About

- Motivation
- Definition of *comfortable*
- Tips to support your testing
  - How to organize non-fragile CI
  - Save/load intermediate results of the current/previous build
  - Get advantages of DeployGate

Before/After Code Testing?

- Testing is an important stage of dev. process
- There are many technique for testing and we are continuing to look for better practices

- However,
  - It's a part of the whole process, and *code* is just one of testing factors
  - Things around code testing are also important, and should be comfortable

What I Will Talk About

- Motivation
- Definition of *comfortable*
- Tips to support your testing
  - How to organize non-fragile CI
  - Save/load intermediate results of the current/previous build
  - Get advantages of DeployGate

What's *Comfortable* in This Talk?

- In short, it's to accelerate our development
  - Not fragile, be stable
    - Nobody likes CI/CD which fail frequently
  - Be help for other development stages
    - Finding/fixing bugs, QA, etc.

What I Will Talk About

- Motivation
- Definition of *comfortable*
- Tips to support your testing
- How to organize non-fragile CI
- Save/load intermediate results of the current/previous build
- Get advantages of DeployGate

Not Fragile (Stable) Realized by Flexible

- No failure is allowed

Flexible ⊂ Not Fragile (Stable)

- ~~No failure is allowed~~
- Non-flexible with failures is an evil
  - BAD: Fail CI anyway if any task fails
  - BAD: Do not run a Test task if lint fails
- How to make it flexible?

Flexible ⊂ Not Fragile (Stable)

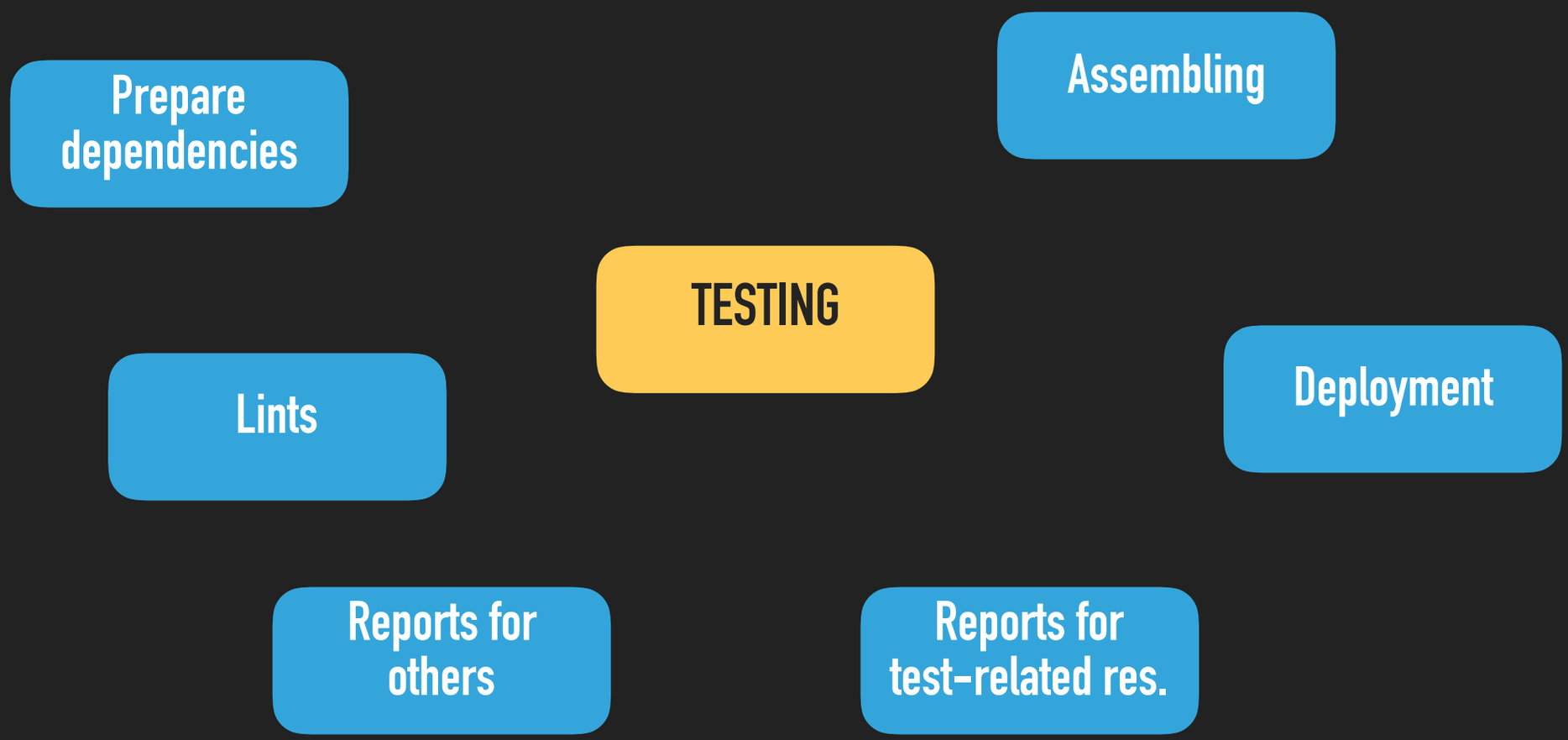
- ~~No failure is allowed~~
- Non-flexible with failures is an evil
  - BAD: Fail CI anyway if any task fails
  - BAD: Do not run a Test task if lint fails
- How to make it flexible?
  - Check command status and make CI/CD fail if needed
  - Separate jobs if possible

Flexible ⊂ Not Fragile (Stable)

- ~~No failure is allowed~~
- Non-flexible with failures is an evil
  - BAD: Fail CI anyway if any task fails
  - BAD: Do not run a Test task if lint fails
- How to make it flexible?
  - Check command status and make CI/CD fail if needed
  - Separate jobs if possible

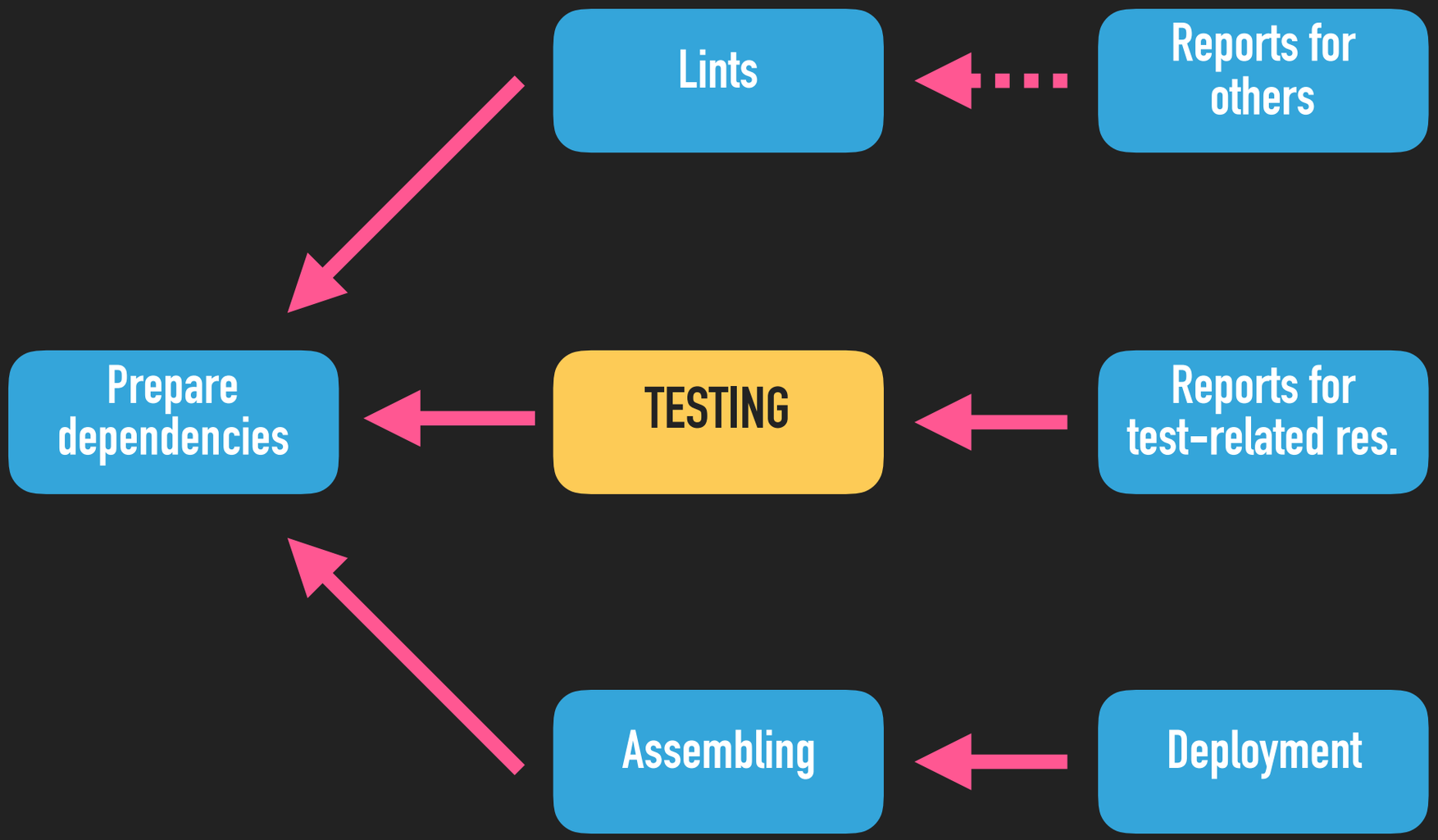
Know Rules of Command Set To Be Executed

- Basically you have multiple roles in your dep. pipeline



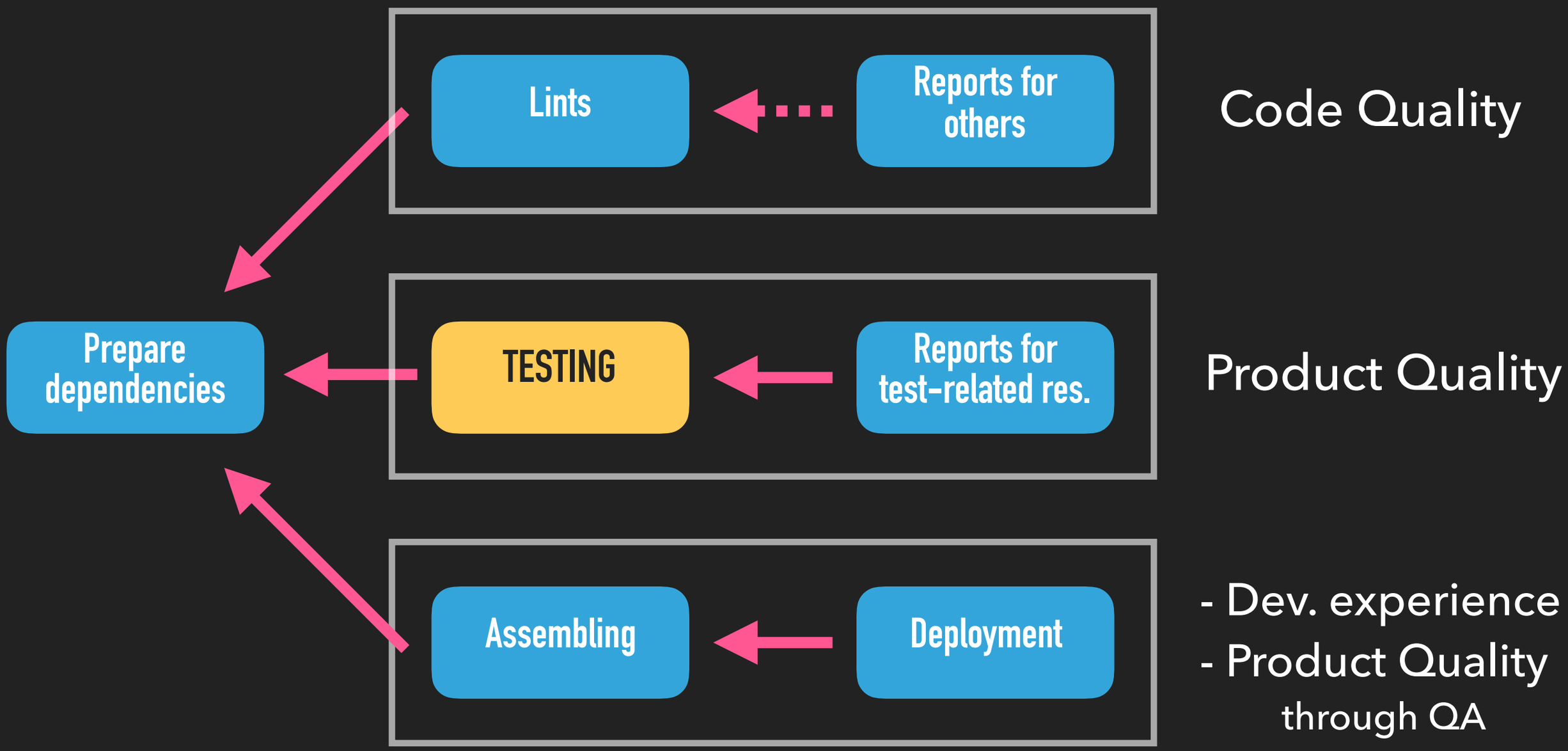
Consider *Required* Dependencies Between Command Set

- You can notice some of commands looks independent



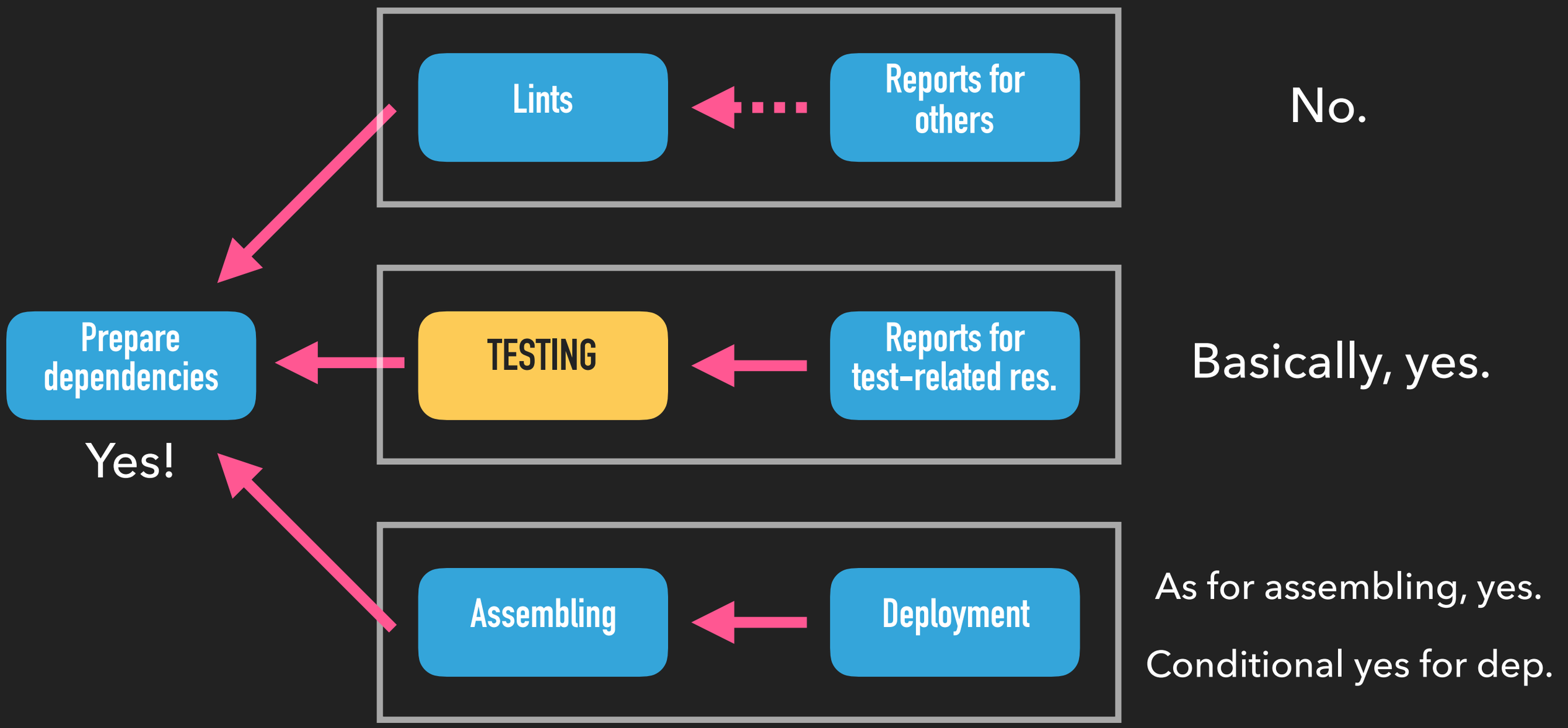
Check What Will Be Assured by Each Roles

- If you wanna assure nothing, you can eject CI/CD :p



So, Which Command Should Fail CI/CD Immediately?

- Please note that this slide and your though may differ



Other Examples

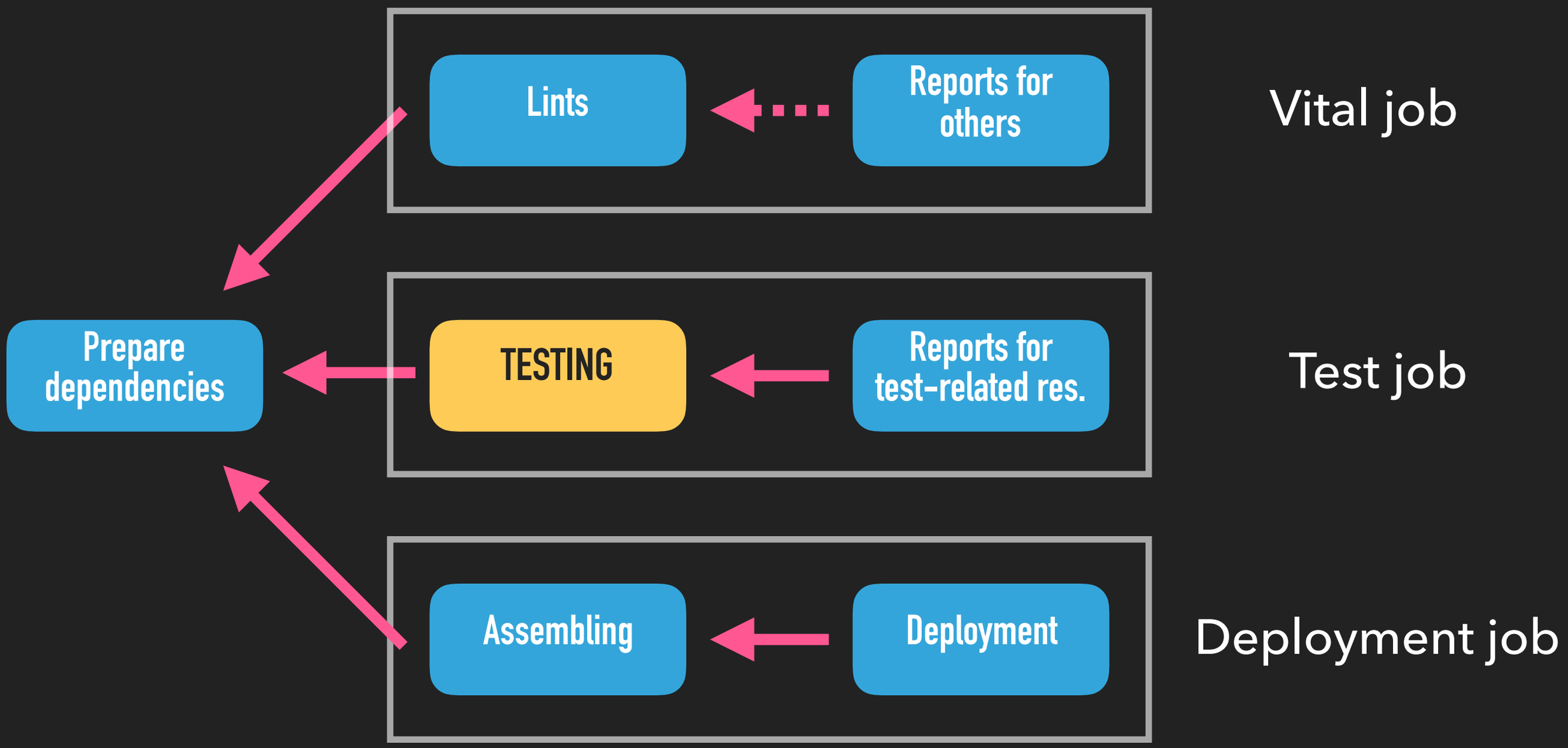
- Use *continue* option when running Gradle tasks
  - ./gradlew lintDebug testDebugUnitTest --continue --offline
- Use retry-logic to execute unstable commands
  - For example, 500 error from maven repo. should be retried
- Execute reporting system like Danger anyway
  - Basically we want to know test failures in a failure case

Flexible ⊂ Not Fragile (Stable)

- ~~No failure is allowed~~
- Non-flexible with failures is an evil
  - BAD: Fail CI anyway if any task fails
  - BAD: Do not run a Test task if lint fails
- How to make it flexible?
  - Check command status and make CI/CD fail if needed
  - Separate jobs if possible

Built *Required* Graph Will Be Your Help

- Basically, jobs depend on execution time and CI services



What I Will Talk About

- Motivation
- Definition of *comfortable*
- Tips to support your testing
  - How to organize non-fragile CI
  - Save/load intermediate results of the current/previous build
  - Get advantages of DeployGate

How To Treat Intermediate Results of CI

- Major CI services have cache systems
  - Key-value style cache, repo/branch-based cache
- They are useful for build intermediate results like artifact dependencies' cache
  - But, how about finer-grained cache...?

Cache During a Build

- Create a new file right after executing a command
  - If the file exists, the command exited with SUCCESS
  - Otherwise, the command failed

Cache Between Builds on a Single PR

- Use the CI's cache system?

Cache Between Builds on a Single PR

- Use the CI's cache system?
  - NO.
- You can use GitHub issues and HTML
  - Fortunately(?), GitHub issues and comments can parse HTML
  - It means you can use HTML comment syntax as well

```
*No issue found by ktlint!*

<!--
{ "is_successful": true, "last_build_num": 48 }
-->
```

Cache Between Builds on Multiple PRs

- Please let me know if there is a smart solution 🙊

What I Will Talk About

- Motivation
- Definition of *comfortable*
- Tips to support your testing
  - How to organize non-fragile CI
  - Save/load intermediate results of the current/previous build
- Get advantages of DeployGate

DeployGate?

- A mobile app distribution service like Fabric Beta
- ~~Annual BBQ planner in US~~
- https://deploygate.com

Advantages of DeployGate (Require Paid Plans)

- Faster download through CDN
- Distribution tracks

How To Have a Faster Download Experience

- CDN produces this advantage
- Therefore, first download from the signed URL is not fast, unfortunately

How To Have a Faster Download Experience

- CDN produces this advantage
- Therefore, first download from the signed URL is not fast, unfortunately
- Download your artifact first on CI right after deployment

```
artifact_url=$(<response> | jq -r ".results.file")
curl -SLs $artifact_url >/dev/null
```

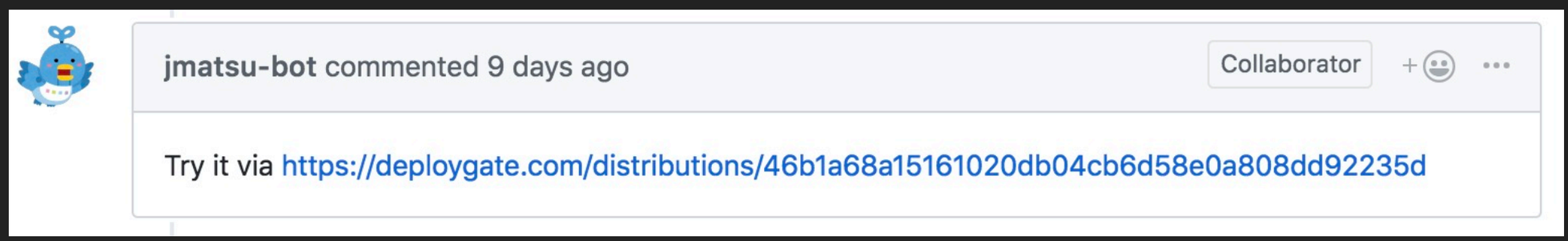
- Note that the URL will be expired, so this tip does not work if new URLs are published

Distribution Tracks

- Download pages for specific versions
  - Like named internal track on Google Play Store

Distribution Tracks

- Download pages for specific versions
  - Like named internal track on Google Play Store
- Deploy release ver., latest develop ver., feature branch ver.
  - GitHub:jmatsu/dpg will be your help
  - GitHub:DroidKaigi/conference-app-2019 is a good practice



Be Comfortable Before/After Code Testing as Well

- How to organize non-fragile CI
  - Reorganize jobs and handle exit statuses manually
  - Save/load intermediate results of the current/previous build
    - Use new files and/or html comments to save/load metadata
- Get advantages of DeployGate
  - Download a first artifact on CI, use distribution tracks

Don't Hesitate To Ask Me any Questions

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