CS304
Software Engineering

TAN, Shin Hwei

陈馨慧

Instructor: 胡春风 王大兴

Southern University of Science and Technology

UI Design

10 UI Design Rules

1. Make Everything the User Needs Readily Accessible
2. Be Consistent
3. Be Clear
4. Give Feedback
5. Use Recognition, Not Recall
6. Choose How People Will Interact First
7. Follow Design Standards
8. Elemental Hierarchy Matters
9. Keep Things Simple
10. Keep Your Users Free & In Control

Part 1: Design Interview

Invitation link:https://classroom.github.com/a/Yuzgn5gJ

Add your name and student id in README.md and answer the following question:

1. Choose one system design question: https://github.com/donnemartin/system-design-primer#system-design-interview-questions-with-solutions
2. Which question do you choose?
3. Draw the design and include the image in your README.md
4. Do your design follow the 10 UI Design Rules? Explain your answer.

Continuous Integration

What is travis CI?

- Continuous Integration (CI) is the practice of merging in small code changes frequently - rather than merging in a large change at the end of a development cycle. The goal is to build healthier software by developing and testing in smaller increments. This is where Travis CI comes in.

- As a continuous integration platform, Travis CI supports your development process by automatically building and testing code changes, providing immediate feedback on the success of the change. Travis CI can also automate other parts of your development process by managing deployments and notifications.

From: https://docs.travis-ci.com/user/for-beginners/

Travis CI for a Java project

https://dzone.com/articles/travis-ci-tutorial-java-projects

Travis CI for a Java project

- project configuration is read from a .travis.yml file at its root.
- historically, the platform was aimed at ruby, but nowadays different kind of projects in different languages can be built.
- Two parts to change:
  - Define the language
  - Define which jdk to use

```
language: java
jdk: oraclejdk8
```

Compilation

if a pom exists at the root of the project, it's automatically detected and maven (or the maven wrapper) will be used as the build tool in that case. it's a good idea to use the wrapper for it sets the maven version. barring that, there's no hint of the version used.

as written above, travis ci was originally designed for ruby projects. in ruby, dependencies are installed system-wide, not per project as for maven. hence, the lifecycle is composed of:

1. an install dependencies phase, translated by default into `./mvnw install -dskiptests=true -dmaven.javadoc.skip=true -b -v`
2. a build phase, explicitly defined by each project

this two-phase mapping is not relevant for maven projects as maven uses a lazy approach: if dependencies are not in the local repository, they will be downloaded when required. for a more idiomatic management, the first phase can be bypassed, while the second one just needs to be set:

```
1 install: true
2 script: ./mvnw clean install
```

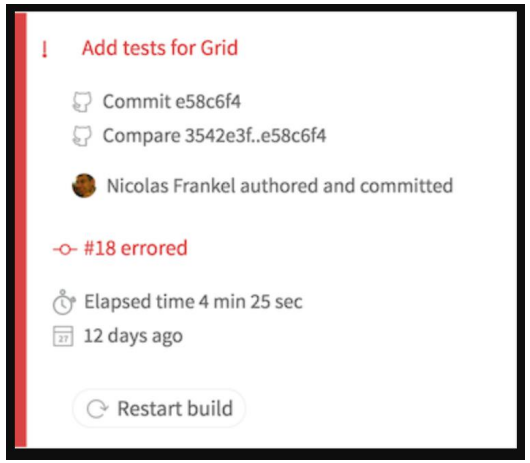
Improve Speed

- To speed up future builds, we should keep the maven local repository between different runs, as it would be the case on jenkins or a local machine. the following configuration achieves just that:

```
cache:
  directories:
  - $home/.m2
```

Tests and failing builds

- as for standard maven builds, phases are run sequentially, so calling install will first use compile and then test a single failing test, and the build will fail. a report is sent by email when it happens.



- most open source projects display their build status badge on their homepage to build trust with their users. this badge is provided by travis ci, it just needs to be hot-linked, as seen below:

Code cov

- Travis ci doesn't use jacoco reports generated during the maven build. one has to use another tool. there are several available:
  - We can use https://codecov.io/ because mockito also uses it. We just need to register using github and accept conditions.

- Codecov happily uses jacoco coverage reports, but chooses to display only lines coverage - the less meaningful metrics imho. yet, it's widespread enough, so let's display it to users via a nice badge that can be hot-linked:
- the build configuration file needs to be updated:


```
script:
  - ./mvnw clean install
  - bash <(curl -s https://codecov.io/bash)
```

- on every build, travis ci calls the online codecov shell script, which will somehow update the code coverage value based on the jacoco report generated during the previous build command.

Jobs in CI

The Job Lifecycle

Each *job* is a sequence of phases. The *main phases* are:

1. `install` - install any dependencies required
2. `script` - run the build script

Travis CI can run custom commands in the phases:

1. `before_install` - before the install phase
2. `before_script` - before the script phase
3. `after_script` - after the script phase.
4. `after_success` - when the build *succeeds* (e.g. building documentation), the result is in `TRAVIS_TEST_RESULT` environment variable
5. `after_failure` - when the build *fails* (e.g. uploading log files), the result is in `TRAVIS_TEST_RESULT` environment variable

Jobs in CI

The complete sequence of phases of a job is the lifecycle. The steps are:

1. OPTIONAL Install `apt addons`
2. OPTIONAL Install `cache components`
3. `before_install`
4. `install`
5. `before_script`
6. `script`
7. OPTIONAL `before_cache` (if and only if caching is effective)
8. `after_success` or `after_failure`
9. OPTIONAL `before_deploy` (if and only if deployment is active)
10. OPTIONAL `deploy`
11. OPTIONAL `after_deploy` (if and only if deployment is active)
12. `after_script`

Try it on a project

Step 1: Make a change and Trigger a build

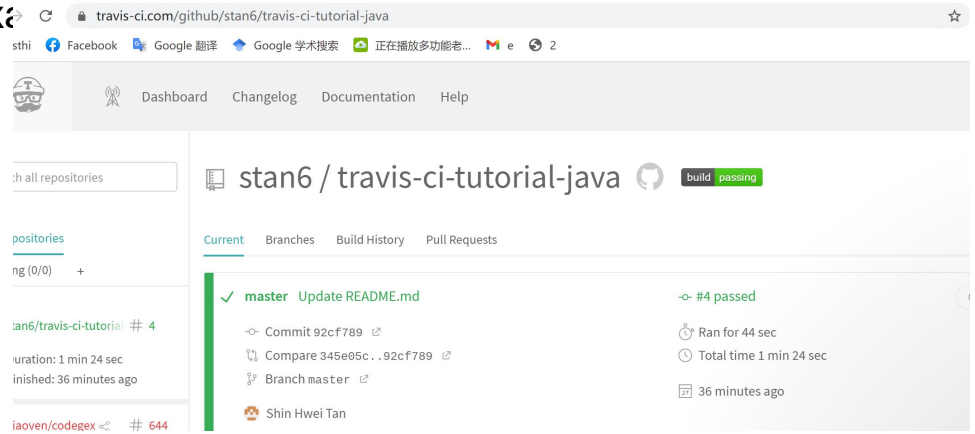
1. Fork this directory: https://github.com/joaomlneto/travis-ci-tutorial-java/fork
2. Go to Travis CI at https://travis-ci.com/ and enable the repository (**make sure that it is a public repository**)
3. Change the README.md badges (replacing in the URL joaomlneto with your-github-username) and push the changes. This should trigger a build in Travis CI!

4. Check if your build is “passing”

What to submit?

- README.md (In README.md, you only needs to add **student name and id**)

1. A screenshot with your repository showing that the travis CI build in your forked repository (travis-ci-tutorial-java) is passing (exa



2. Answer the following questions about your selected project in the README.md:

   a) Do your selected project use Continous Integration? If yes, what is the service that it uses (Travis CI, GitHub Actions, Jenkins, Bamboo)? *You can check the service used by clicking on the build:passing/failing badge*. (If no, answer this question by checking what is the service that https://github.com/google/guava used?)

   b) What are the **jobs** (e.g., install, script) defined to be run with CI for your selected project. (If your project does not use CI, answer this question by checking https://github.com/google/guava used?)

Continue working on your group project

Group Project
-Final Presentation Instruction Uploaded in Sakai
-Due 22 May 2021