Software Testing, Verification and Validation

September 23, 2022
Week #2 — Recitation #1

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Static Verification

Static Verification

Static Testing is a software testing technique which is used to check *faults* in a software application without executing its source code. It is concerned with the analysis of the static system representation (source code, documents, models, prototypes, etc.) to discover *faults*.

- Early detection of faults prior to test execution.
- Early warning about suspicious aspects of the code or design.
- Detecting dependencies and inconsistencies in software models, such as links.
- Improved maintainability of code and design.
- Prevention of faults, if lessons are learned in development.

Static Verification

The two main types of static testing techniques are:

- **Manual examinations**: Manual examinations include analysis of code done manually, also known as **Reviews.**
- **Automated analysis using tools:** Automated analysis are basically static analysis which is done using tools.

Static Verification Tools

What can static analysis do?

A form of automated testing:

- Check for violations of standards.
- Check for things which may be faulty.
- Can find unreachable code, undeclared variables, parameter type mis-matches, uncalled functions, etc.
- Static analysis tools are scalable and generally require less time to set up.

The objective of static analysis is to find faults in software source code and software models. Static analysis is performed without actually executing the software being examined by the tool. As with reviews, static analysis finds faults rather than failures.

Types of Static Analysis Tools

- Lexical: Words, strings, and regexps.
- Syntactic: Tree of program structure.
- Control flow graph.
- Data flow graph.

Here are a few examples of well-know static analysis tools:

- [Checkstyle](https://checkstyle.sourceforge.io)
- [SpotBugs](https://spotbugs.github.io)
- [PMD](https://pmd.github.io)
- [Google's Error Prone](https://errorprone.info)
- [SourceMeter](https://www.sourcemeter.com)
- [Checkmarx](https://www.checkmarx.com)

False Positive / Negative

Many static analysis tools are based on heuristics which may produce a large number of warning messages, which need to be well managed to allow the most effective use of the tool.

- Correct positive: Warning, and a true problem (👍 let's fix it!)
- Correct negative: No warning, no problem. (no action required 😎)
- False positive: Warning, but not a problem (annoying 😡)
- False negative: Problem, but no warning (dangerous 💣)

Data flow analysis

Data-flow analysis is a technique for gathering information about the possible set of values calculated at various points in a computer program. In other words, study program's variables.

```
x = y + z;     // x is defined, y and z are used

if (a > b) {   // a and b are used
  read(s);     // s is defined
}
```

Data flow analysis

```
n = 0;
read(x);
n = 1;    // anomaly: n is re-defined
          // without being used
```

Control flow analysis

In computer science, control-flow analysis (CFA) is a static-code-analysis technique for determining the control flow of a program. A control-flow graph (CFG) is a representation, using graph notation, of all paths that might be traversed through a program during its execution.

Control flow analysis can check, e.g.,

- Infinite loops
- Unreachable code

(Code) Metrics

- Lines of code
- Complexity: number of if-statements per method
- Coupling: number of classes a class depend on
- Cohesion: correlation between variables and methods
- Nesting levels: relate to how deeply nested statements are within other IF statements.
- Cyclomatic complexity is a software metric used to indicate the complexity of a program, i.e., of a flow graph. It is a quantitative measure of the number of linearly independent paths through a program’s source code. The more complex the flow graph, the greater the measure.

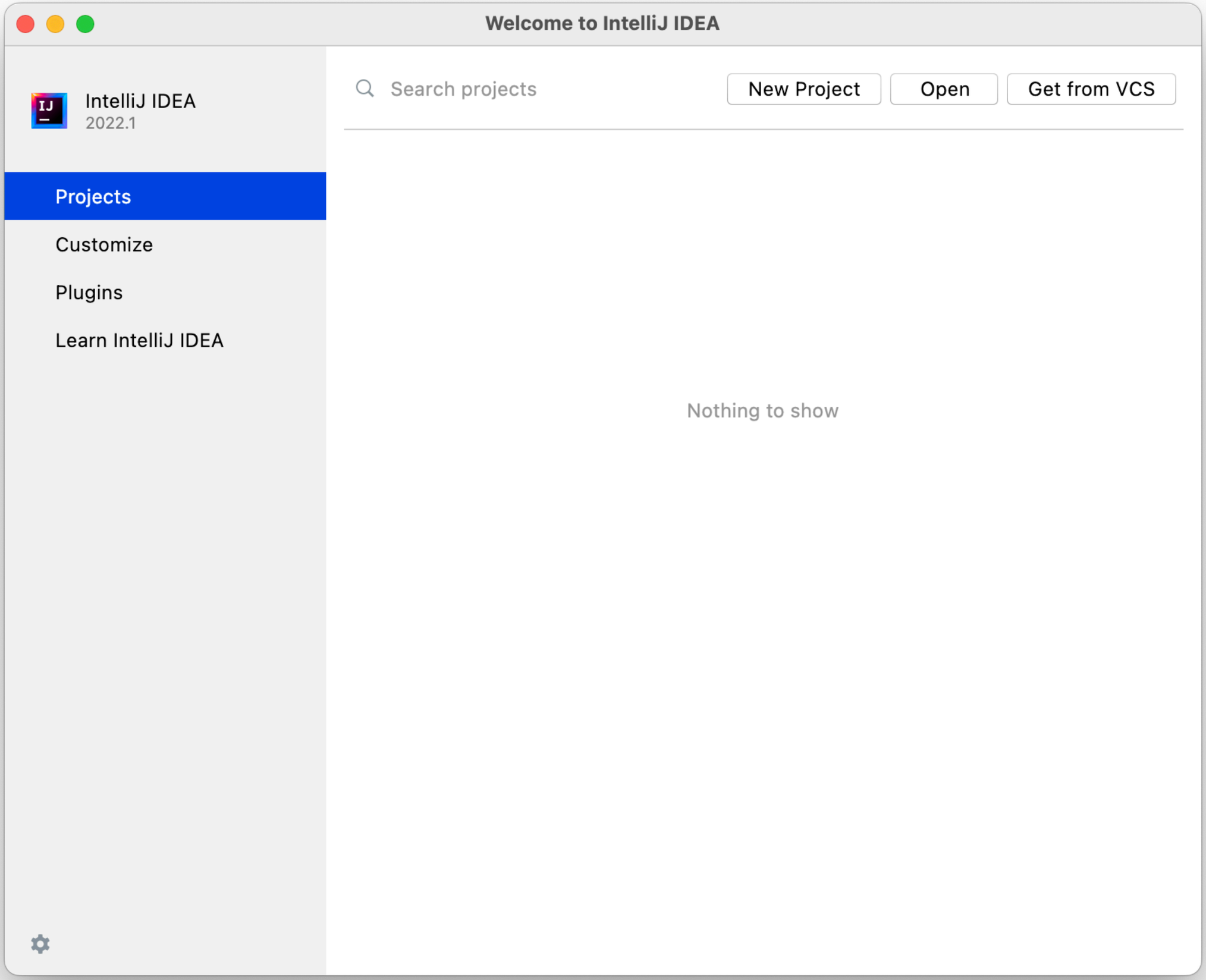
Demo

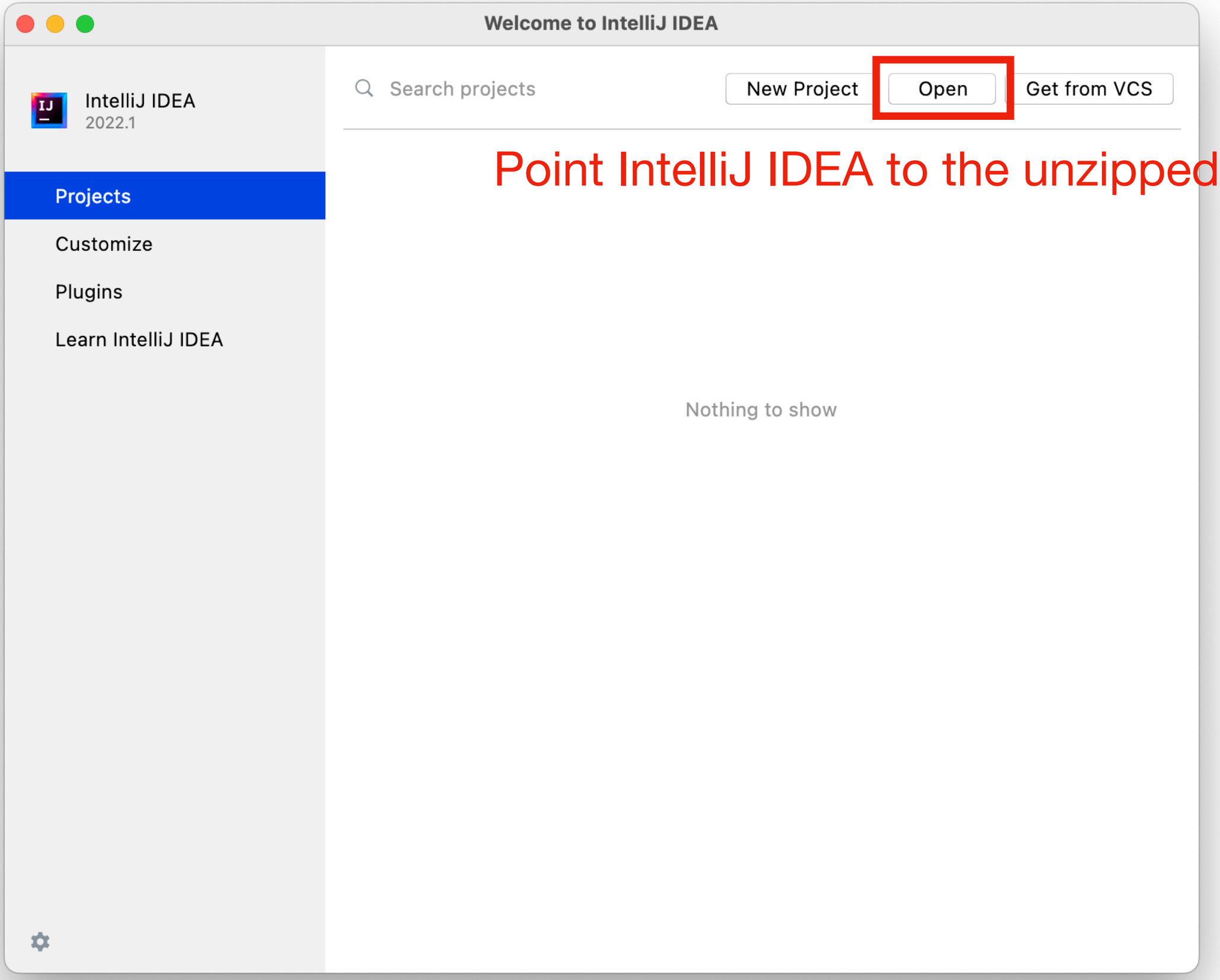
https://paginas.fe.up.pt/~jcmc/tvvs/2022-2023/recitations/recitation-1-jpacman.zip

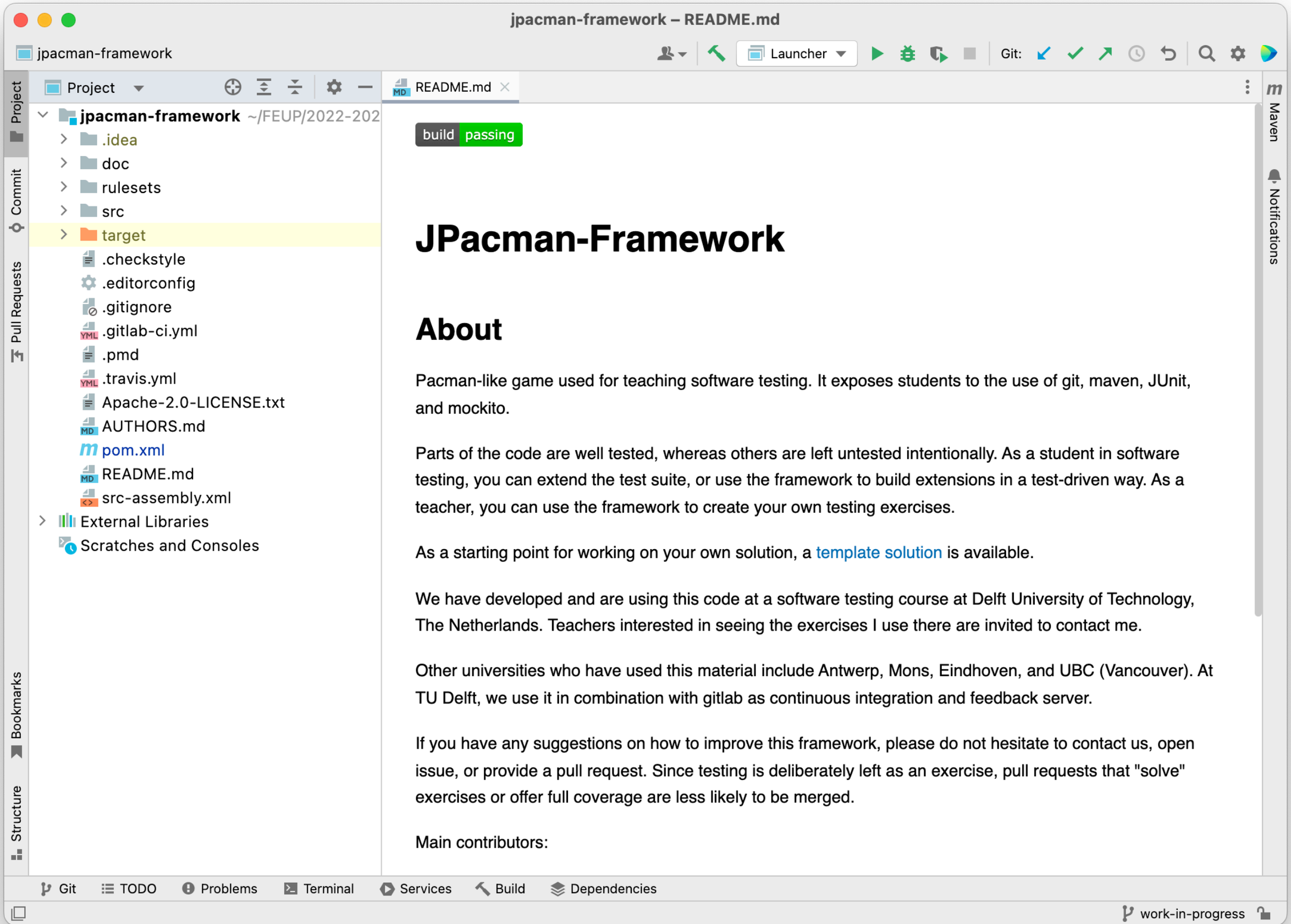
1. Get the jpacman game's source code available in here, https://paginas.fe.up.pt/~jcmc/tvvs/2022-2023/recitations/recitation-1-jpacman.zip.

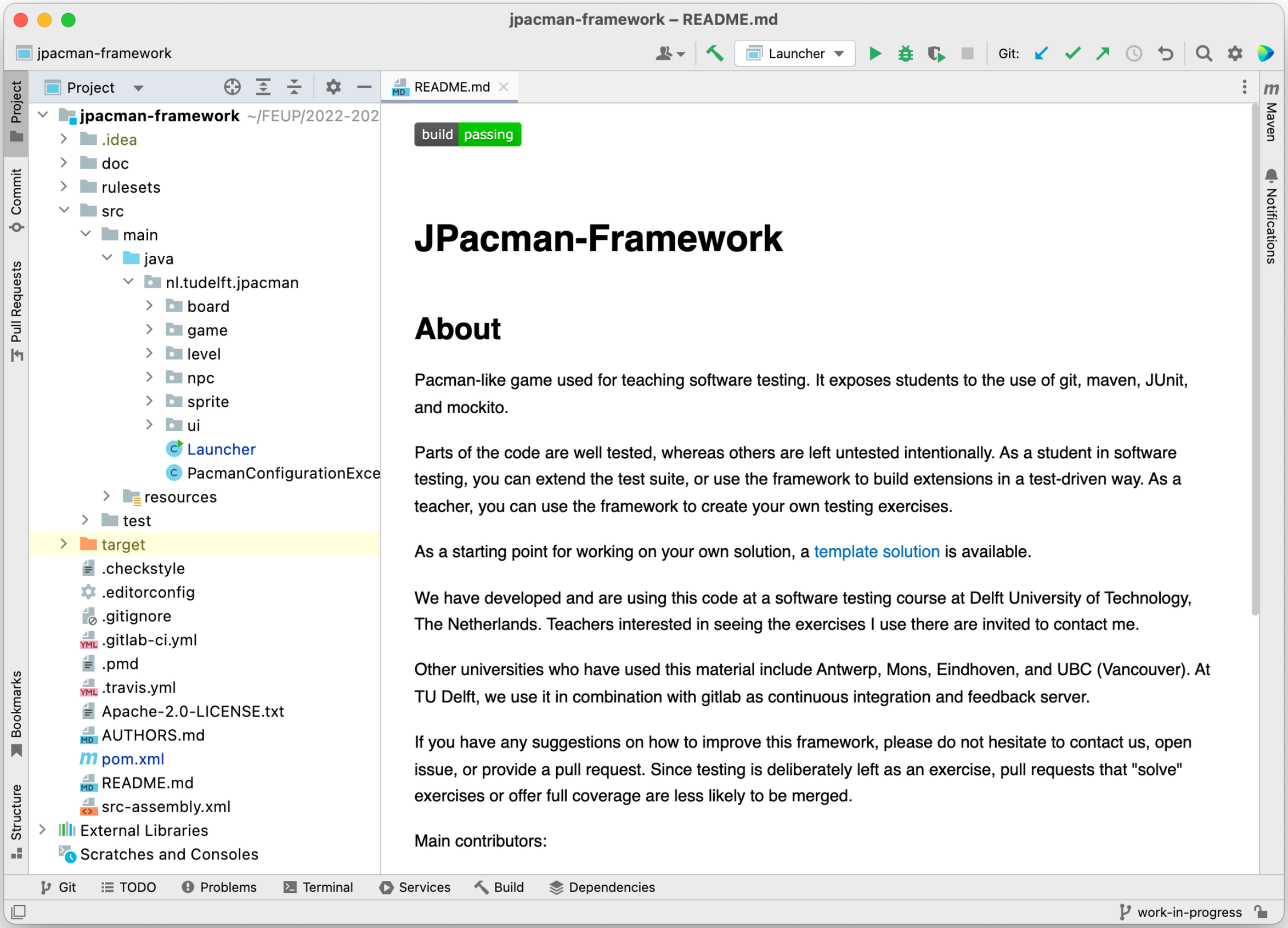
2. Unzip the given recitation-1-jpacman.zip file.

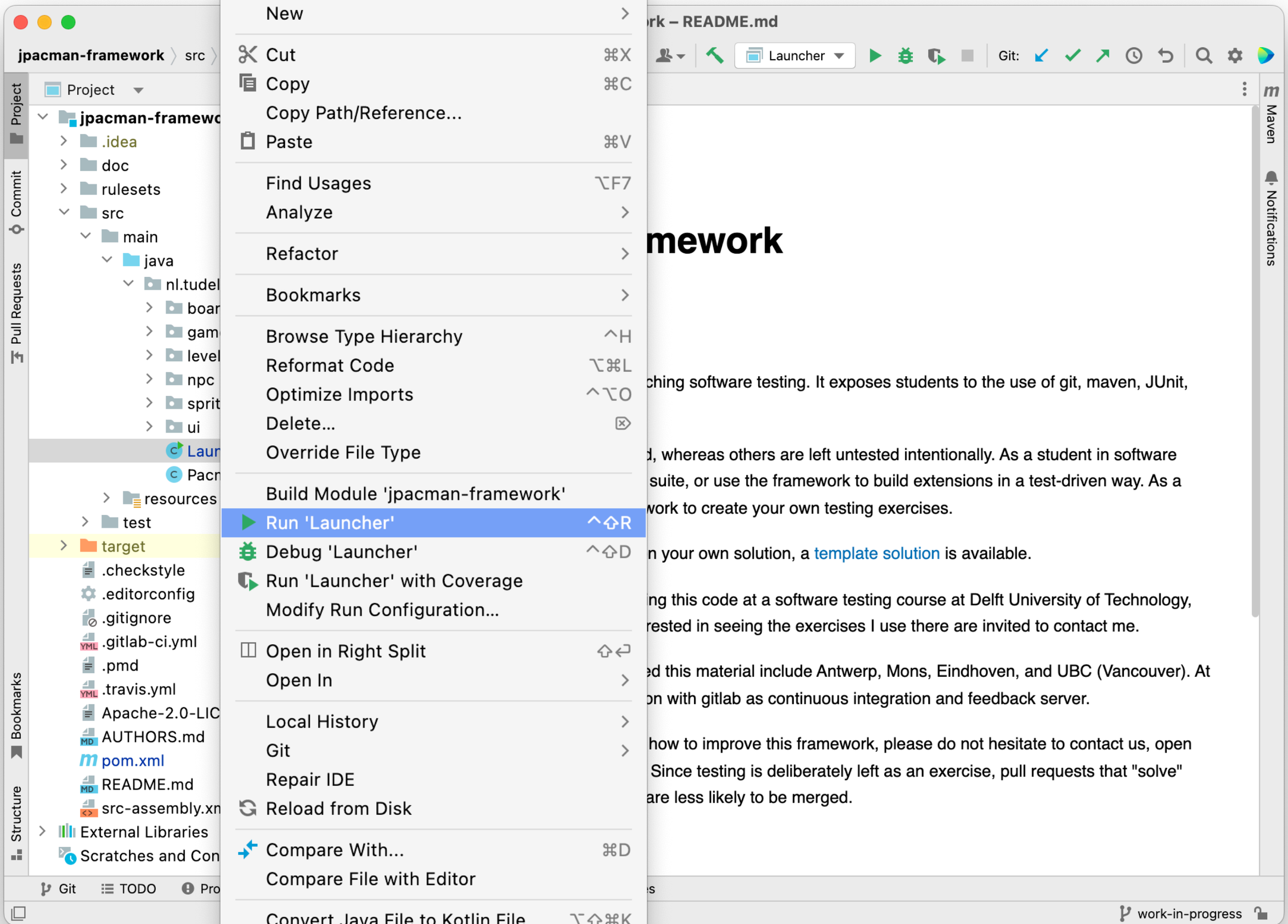
3. Open IntelliJ IDEA.

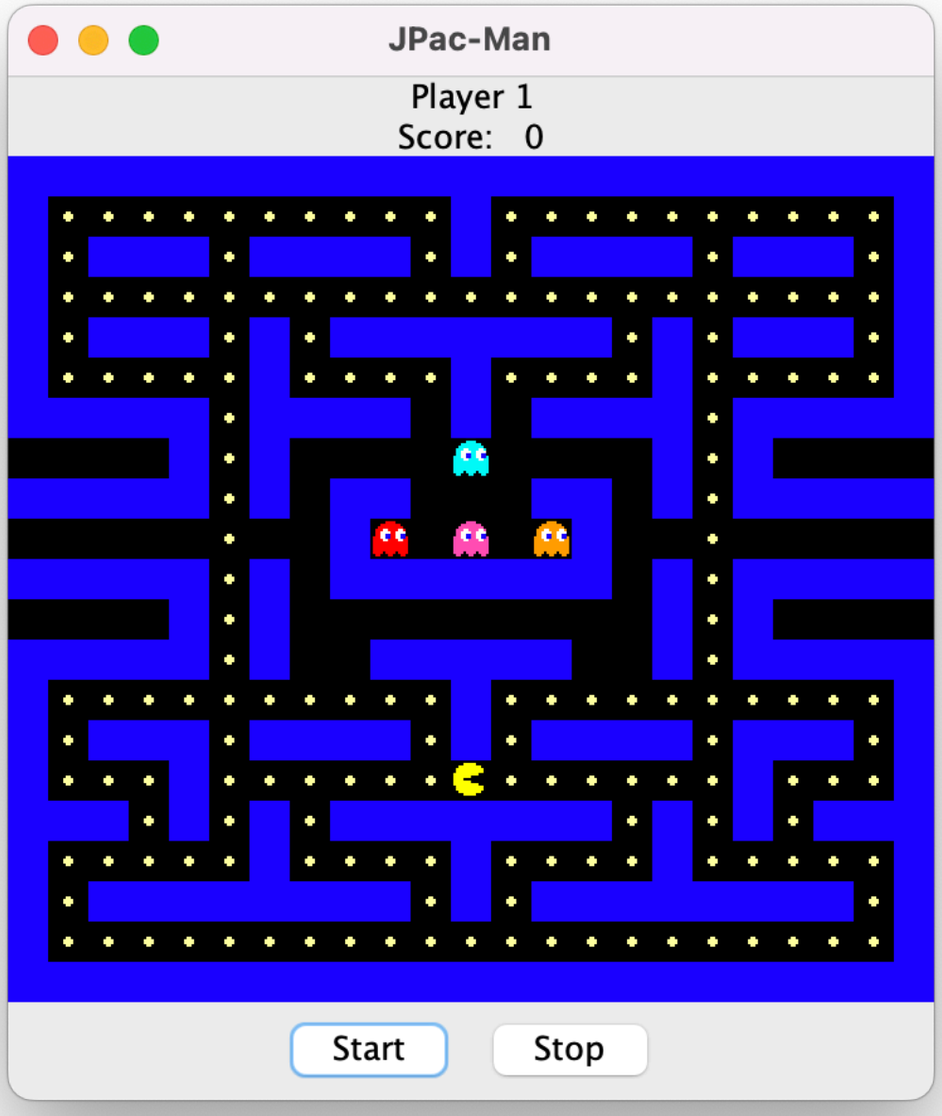

Welcome to IntelliJ IDEA
IntelliJ IDEA
2022.1
Projects
Customize
Plugins
Learn IntelliJ IDEA
Search projects
New Project
Open
Get from VCS
Nothing to show

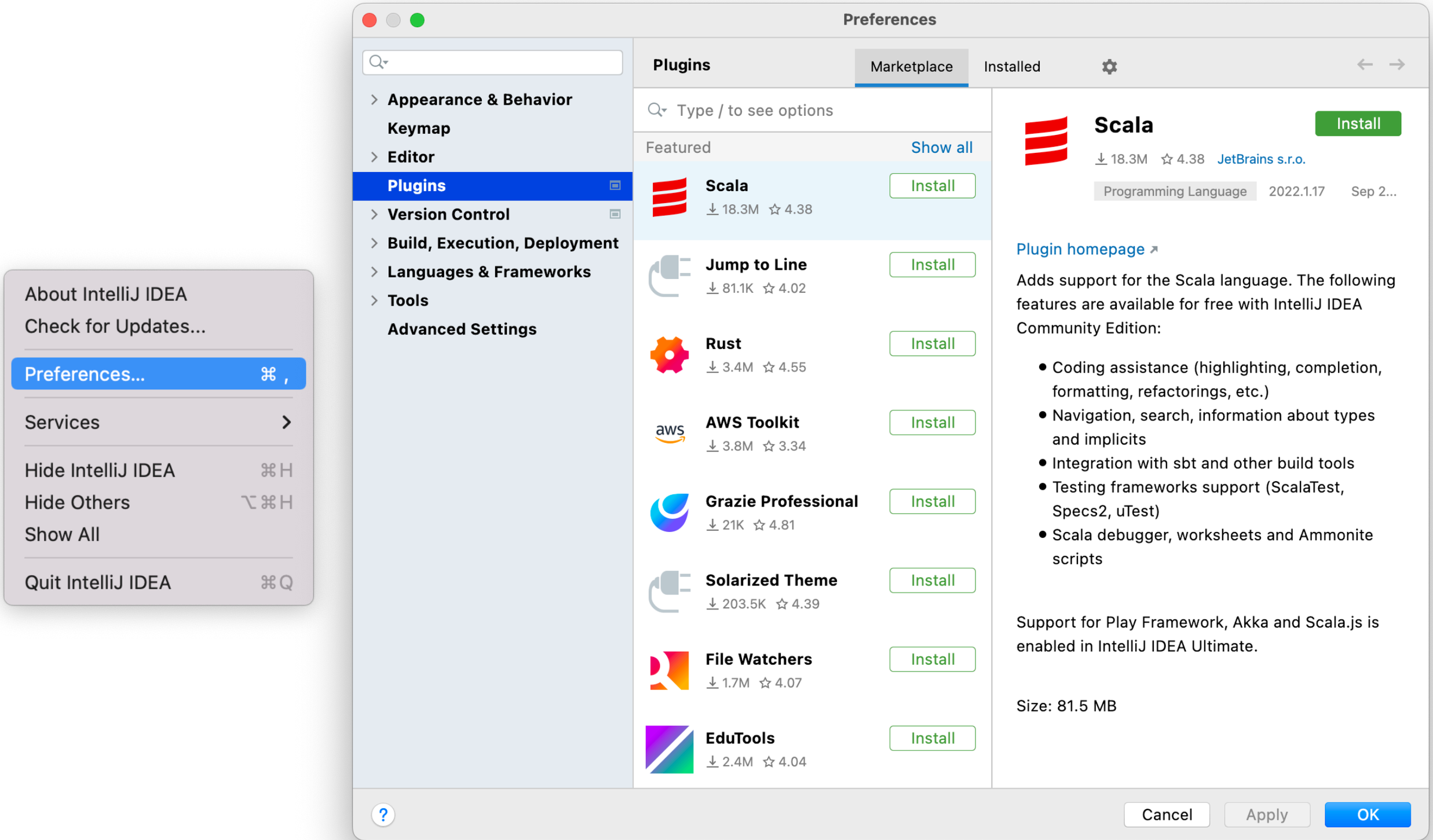

Welcome to IntelliJ IDEA
IntelliJ IDEA
2022.1
Projects
Customize
Plugins
Learn IntelliJ IDEA
Search projects
New Project
Open
Get from VCS
Point IntelliJ IDEA to the unzipped folder.
Nothing to show


jpacman-framework – README.md
jpacman-framework
Launcher
Git:
Project
README.md
jpacman-framework ~/FEUP/2022-202
.idea
doc
rulesets
src
target
.checkstyle
.editorconfig
.gitignore
.gitlab-ci.yml
.pmd
.travis.yml
Apache-2.0-LICENSE.txt
AUTHORS.md
pom.xml
README.md
src-assembly.xml
External Libraries
Scratches and Consoles
build passing
JPacman-Framework
About
Pacman-like game used for teaching software testing. It exposes students to the use of git, maven, JUnit, and mockito.
Parts of the code are well tested, whereas others are left untested intentionally. As a student in software testing, you can extend the test suite, or use the framework to build extensions in a test-driven way. As a teacher, you can use the framework to create your own testing exercises.
As a starting point for working on your own solution, a template solution is available.
We have developed and are using this code at a software testing course at Delft University of Technology, The Netherlands. Teachers interested in seeing the exercises I use there are invited to contact me.
Other universities who have used this material include Antwerp, Mons, Eindhoven, and UBC (Vancouver). At TU Delft, we use it in combination with gitlab as continuous integration and feedback server.
If you have any suggestions on how to improve this framework, please do not hesitate to contact us, open issue, or provide a pull request. Since testing is deliberately left as an exercise, pull requests that "solve" exercises or offer full coverage are less likely to be merged.
Main contributors:
Project
Commit
Pull Requests
Bookmarks
Structure
Maven
Notifications
Git
TODO
Problems
Terminal
Services
Build
Dependencies
work-in-progress


jpacman-framework – README.md
jpacman-framework
Launcher
Git:
Project
README.md
jpacman-framework ~/FEUP/2022-202
.idea
doc
rulesets
src
main
java
nl.tudelft.jpacman
board
game
level
npc
sprite
ui
Launcher
PacmanConfigurationExce
resources
test
target
.checkstyle
.editorconfig
.gitignore
.gitlab-ci.yml
.pmd
.travis.yml
Apache-2.0-LICENSE.txt
AUTHORS.md
pom.xml
README.md
src-assembly.xml
External Libraries
Scratches and Consoles
build passing
JPacman-Framework
About
Pacman-like game used for teaching software testing. It exposes students to the use of git, maven, JUnit, and mockito.
Parts of the code are well tested, whereas others are left untested intentionally. As a student in software testing, you can extend the test suite, or use the framework to build extensions in a test-driven way. As a teacher, you can use the framework to create your own testing exercises.
As a starting point for working on your own solution, a template solution is available.
We have developed and are using this code at a software testing course at Delft University of Technology, The Netherlands. Teachers interested in seeing the exercises I use there are invited to contact me.
Other universities who have used this material include Antwerp, Mons, Eindhoven, and UBC (Vancouver). At TU Delft, we use it in combination with gitlab as continuous integration and feedback server.
If you have any suggestions on how to improve this framework, please do not hesitate to contact us, open issue, or provide a pull request. Since testing is deliberately left as an exercise, pull requests that "solve" exercises or offer full coverage are less likely to be merged.
Main contributors:
Project
Commit
Pull Requests
Bookmarks
Structure
Maven
Notifications
Git
TODO
Problems
Terminal
Services
Build
Dependencies
work-in-progress


jpacman-framework
src
rk – README.md
Launcher
Git:
Project
jpacman-framewo
.idea
doc
rulesets
src
main
java
nl.tudel
boar
gam
level
npc
sprit
ui
Laun
Pacm
resources
test
target
.checkstyle
.editorconfig
.gitignore
.gitlab-ci.yml
.pmd
.travis.yml
Apache-2.0-LIC
AUTHORS.md
pom.xml
README.md
src-assembly.xm
External Libraries
Scratches and Con
mework
ching software testing. It exposes students to the use of git, maven, JUnit,
d, whereas others are left untested intentionally. As a student in software
suite, or use the framework to build extensions in a test-driven way. As a
work to create your own testing exercises.
n your own solution, a template solution is available.
ng this code at a software testing course at Delft University of Technology,
rested in seeing the exercises I use there are invited to contact me.
ed this material include Antwerp, Mons, Eindhoven, and UBC (Vancouver). At
on with gitlab as continuous integration and feedback server.
how to improve this framework, please do not hesitate to contact us, open
Since testing is deliberately left as an exercise, pull requests that "solve"
are less likely to be merged.
New
Cut ⌘X
Copy ⌘C
Copy Path/Reference...
Paste ⌘V
Find Usages ⌥F7
Analyze
Refactor
Bookmarks
Browse Type Hierarchy ^H
Reformat Code ⌥⌘L
Optimize Imports ^⌥O
Delete...
Override File Type
Build Module 'jpacman-framework'
Run 'Launcher' ^⇧R
Debug 'Launcher' ^⇧D
Run 'Launcher' with Coverage
Modify Run Configuration...
Open in Right Split ⇧⏎
Open In
Local History
Git
Repair IDE
Reload from Disk
Compare With... ⌘D
Compare File with Editor
Convert Java File to Kotlin File ⌥⇧⌘K
Create Gist...
Maven
Notifications
Bookmarks
Structure
Git
TODO
work-in-progress


JPac-Man
Player 1
Score: 0
Start
Stop


About IntelliJ IDEA
Check for Updates...
Preferences... ⌘ ,
Services
Hide IntelliJ IDEA ⌘H
Hide Others ⌥⌘H
Show All
Quit IntelliJ IDEA ⌘Q
Preferences
Appearance & Behavior
Keymap
Editor
Plugins
Version Control
Build, Execution, Deployment
Languages & Frameworks
Tools
Advanced Settings
Plugins
Marketplace
Installed
Type / to see options
Featured
Show all
Scala
18.3M ☆ 4.38
Install
Jump to Line
81.1K ☆ 4.02
Install
Rust
3.4M ☆ 4.55
Install
AWS Toolkit
3.8M ☆ 3.34
Install
Grazie Professional
21K ☆ 4.81
Install
Solarized Theme
203.5K ☆ 4.39
Install
File Watchers
1.7M ☆ 4.07
Install
EduTools
2.4M ☆ 4.04
Install
Scala
Install
18.3M ☆ 4.38 JetBrains s.r.o.
Programming Language 2022.1.17 Sep 2...
Plugin homepage
Adds support for the Scala language. The following features are available for free with IntelliJ IDEA Community Edition:
Coding assistance (highlighting, completion, formatting, refactorings, etc.)
Navigation, search, information about types and implicits
Integration with sbt and other build tools
Testing frameworks support (ScalaTest, Specs2, uTest)
Scala debugger, worksheets and Ammonite scripts
Support for Play Framework, Akka and Scala.js is enabled in IntelliJ IDEA Ultimate.
Size: 81.5 MB
Cancel
Apply
OK

Search for the plugin, in this case, **checkstyle**

About IntelliJ IDEA
Check for Updates...

Preferences... ⌘ ,

Services >

Hide IntelliJ IDEA ⌘H
Hide Others ⌥⌘H
Show All

Quit IntelliJ IDEA ⌘Q

Preferences

Appearance & Behavior
Keymap
Editor
Plugins
Version Control
Build, Execution, Deployment
Languages & Frameworks
Tools
Advanced Settings

Plugins | Marketplace | Installed

Type / to see options

Featured Show all

Scala — 18.3M ☆ 4.38 — Install
Jump to Line — 81.1K ☆ 4.02 — Install
Rust — 3.4M ☆ 4.55 — Install
AWS Toolkit — 3.8M ☆ 3.34 — Install
Grazie Professional — 21K ☆ 4.81 — Install
Solarized Theme — 203.5K ☆ 4.39 — Install
File Watchers — 1.7M ☆ 4.07 — Install
EduTools — 2.4M ☆ 4.04 — Install

Scala Install

18.3M ☆ 4.38 JetBrains s.r.o.

Programming Language 2022.1.17 Sep 2...

Plugin homepage ↗

Adds support for the Scala language. The following features are available for free with IntelliJ IDEA Community Edition:

- Coding assistance (highlighting, completion, formatting, refactorings, etc.)
- Navigation, search, information about types and implicits
- Integration with sbt and other build tools
- Testing frameworks support (ScalaTest, Specs2, uTest)
- Scala debugger, worksheets and Ammonite scripts

Support for Play Framework, Akka and Scala.js is enabled in IntelliJ IDEA Ultimate.

Size: 81.5 MB

? Cancel Apply OK

Preferences

- Appearance & Behavior
- Keymap
- Editor
- Plugins
- Version Control
- Build, Execution, Deployment
- Languages & Frameworks
- Tools
- Advanced Settings

Plugins | Marketplace | Installed

checkstyle

Search Results (9) Sort By: Relevance

CheckStyle-IDEA Install
3.8M ☆ 4.33

QAPlug - Checkstyle Install
123.5K ☆ 3.83

QAPlug Install
443.2K ☆ 3.71

QAPlug - PMD Install
143.6K ☆ 3.76

QAPlug - FindBugs Install
294.8K ☆ 3.10

CS 61B Install
48.4K ☆ 4.58

JetStyle Install
2.4K

Moderne Install
592 ☆ 4.66

CheckStyle-IDEA Install
3.8M ☆ 4.33 Jamie Shiell
Inspection 5.70.0 Aug 06, 2022

Plugin homepage ↗

This plugin provides both real-time and on-demand scanning of Java files with CheckStyle from within IDEA.

Please note this is not an official part of Checkstyle - they neither endorse nor bear responsibility for this plugin. Please see the README for full details.

▸ Change Notes

Size: 119.68 MB

Cancel Apply OK

Preferences

Appearance & Behavior
Keymap
Editor
Plugins
Version Control
Build, Execution, Deployment
Languages & Frameworks
Tools
Advanced Settings

Plugins

Marketplace | Installed

checkstyle

Search Results (9) Sort By: Relevance

Chec...
3.8M ☆ 4.33

QAPlug - Checkstyle Install
123.5K ☆ 3.83

QAPlug Install
443.2K ☆ 3.71

QAPlug - PMD Install
143.6K ☆ 3.76

QAPlug - FindBugs Install
294.8K ☆ 3.10

CS 61B Install
48.4K ☆ 4.58

JetStyle Install
2.4K

Moderne Install
592 ☆ 4.66

CheckSty
3.8M ☆ 4.33 Jamie Shiell

Inspection 5.70.0 Aug 06, 2022

Plugin homepage ↗

This plugin provides both real-time and on-demand scanning of Java files with CheckStyle from within IDEA.

Please note this is not an official part of Checkstyle - they neither endorse nor bear responsibility for this plugin. Please see the README for full details.

▸ Change Notes

Size: 119.68 MB

Cancel Apply OK

Preferences

- Appearance & Behavior
- Keymap
- Editor
- Plugins
- Version Control
- Build, Execution, Deployment
- Languages & Frameworks
- Tools
- Advanced Settings

Plugins — Marketplace — Installed

checkstyle

Search Results (9) — Sort By: Relevance

CheckStyle-IDEA — Restart IDE
3.8M ☆ 4.33

QAPlug - Checkstyle — Install
123.5K ☆ 3.83

QAPlug — Install
443.2K ☆ 3.71

QAPlug - PMD — Install
143.6K ☆ 3.76

QAPlug - FindBugs — Install
294.8K ☆ 3.10

CS 61B — Install
48.4K ☆ 4.58

JetStyle — Install
2.4K

Moderne — Install
592 ☆ 4.66

CheckStyle-IDEA — Restart IDE
3.8M ☆ 4.33 Jamie Shiell
Inspection 5.70.0 Aug 06, 2022

Plugin homepage ↗

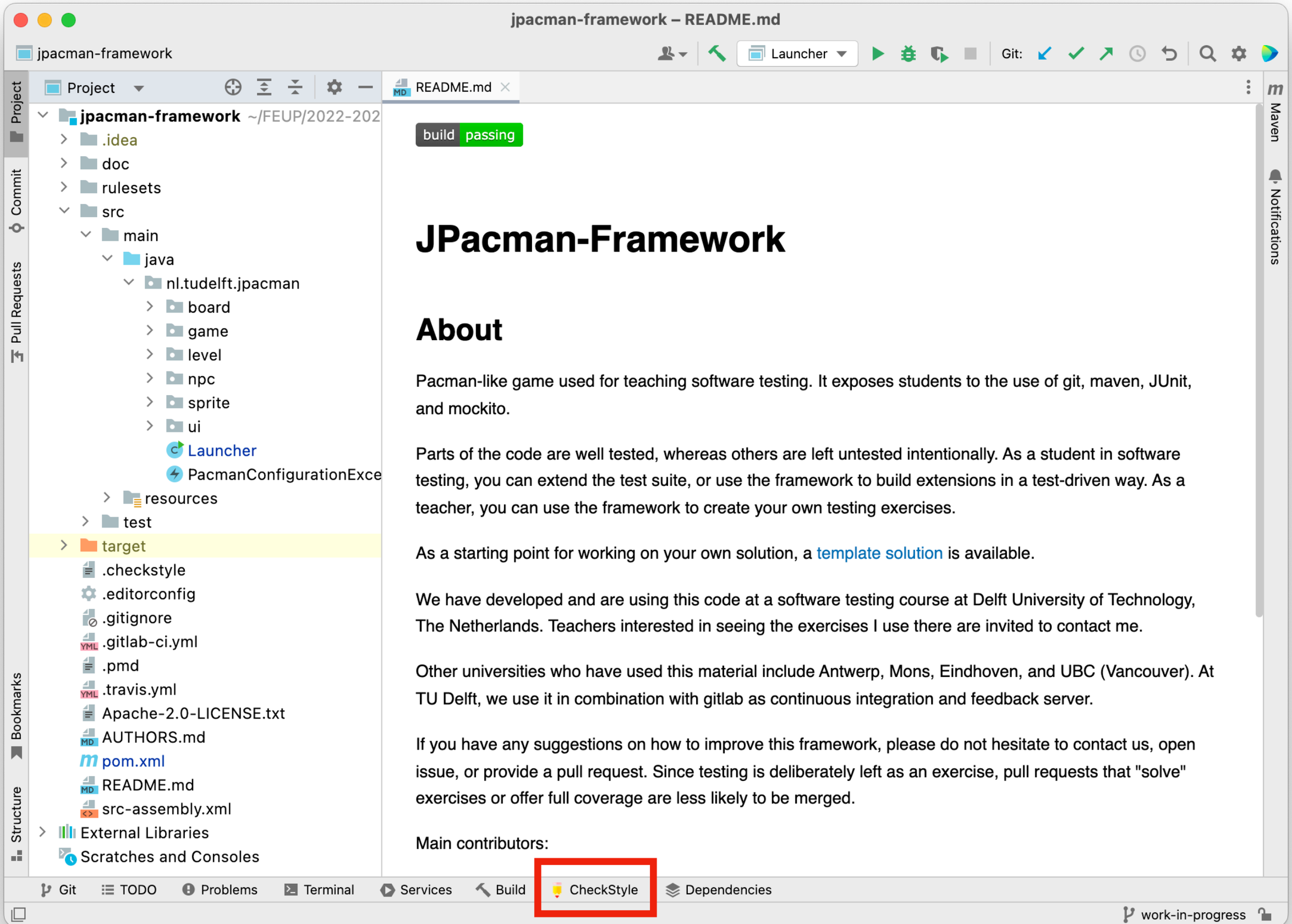
This plugin provides both real-time and on-demand scanning of Java files with CheckStyle from within IDEA.

Please note this is not an official part of Checkstyle - they neither endorse nor bear responsibility for this plugin. Please see the README for full details.

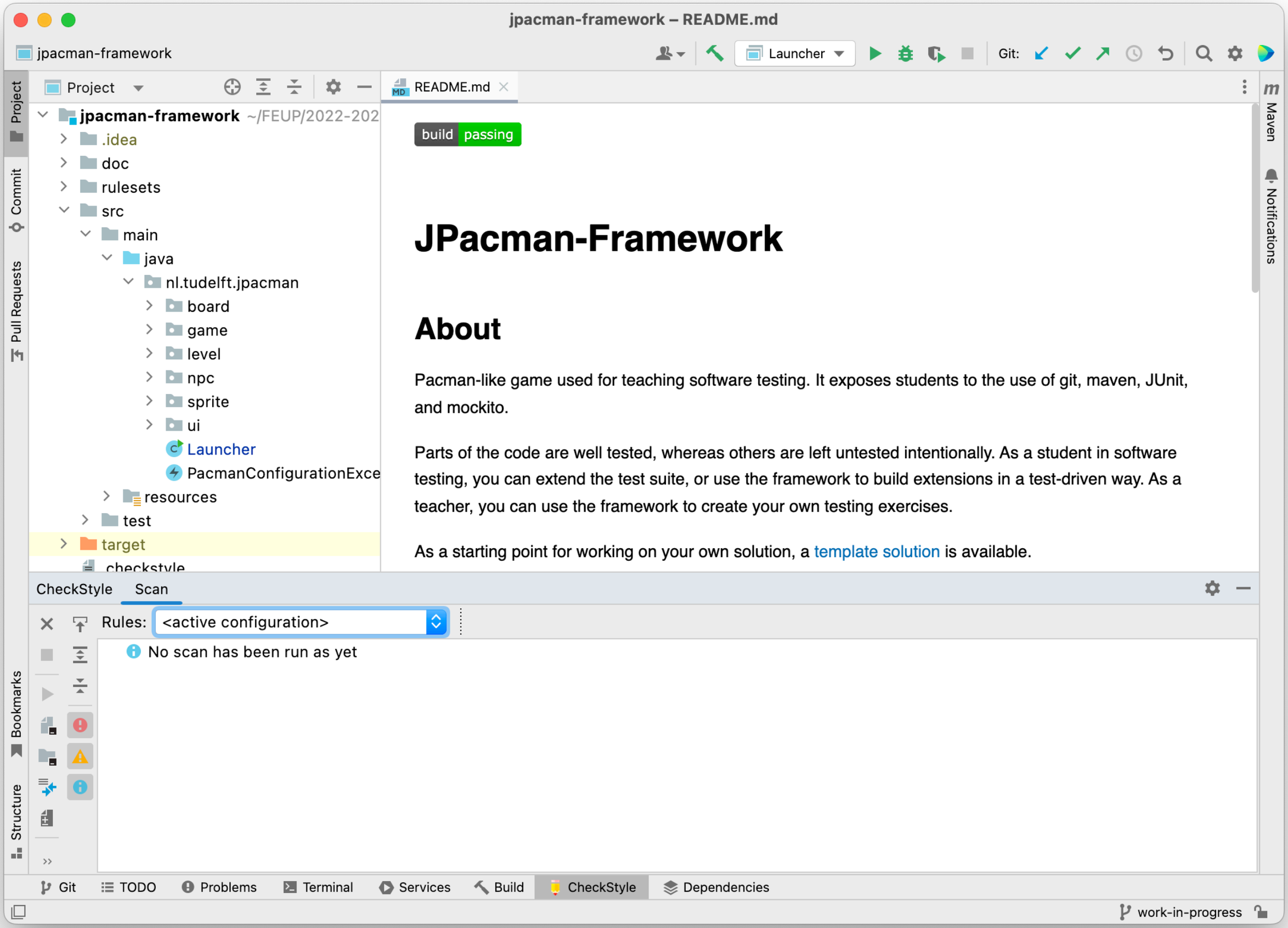
▸ Change Notes

Size: 119.68 MB

Cancel — Apply — OK



A new panel has been added to the editor


jpacman-framework – README.md
jpacman-framework
Launcher
Git:
Project
README.md
jpacman-framework ~/FEUP/2022-202
.idea
doc
rulesets
src
main
java
nl.tudelft.jpacman
board
game
level
npc
sprite
ui
Launcher
PacmanConfigurationExce
resources
test
target
checkstyle
build passing
JPacman-Framework
About
Pacman-like game used for teaching software testing. It exposes students to the use of git, maven, JUnit, and mockito.
Parts of the code are well tested, whereas others are left untested intentionally. As a student in software testing, you can extend the test suite, or use the framework to build extensions in a test-driven way. As a teacher, you can use the framework to create your own testing exercises.
As a starting point for working on your own solution, a template solution is available.
CheckStyle
Scan
Rules:
<active configuration>
No scan has been run as yet
Project
Commit
Pull Requests
Bookmarks
Structure
Maven
Notifications
Git
TODO
Problems
Terminal
Services
Build
CheckStyle
Dependencies
work-in-progress

jpacman-framework – README.md

jpacman-framework

Launcher

Git:

Project

jpacman-framework ~/FEUP/2022-202

.idea

doc

rulesets

src

main

java

nl.tudelft.jpacman

board

game

level

npc

sprite

ui

Launcher

PacmanConfigurationExce

resources

test

target

checkstyle

README.md

build passing

JPacman-Framework

About

Pacman-like game used for teaching software testing. It exposes students to the use of git, maven, JUnit, and mockito.

Parts of the code are well tested, whereas others are left untested intentionally. As a student in software testing, you can extend the test suite, or use the framework to build extensions in a test-driven way. As a teacher, you can use the framework to create your own testing exercises.

As a starting point for working on your own solution, a template solution is available.

CheckStyle Scan

Rules <active configuration>

No scan has been run as yet

Project Commit Pull Requests Bookmarks Structure Maven Notifications

Git TODO Problems Terminal Services Build CheckStyle Dependencies

work-in-progress

jpacman-framework – README.md

jpacman-framework

Launcher

Git:

Project

jpacman-framework ~/FEUP/2022-202

.idea

doc

rulesets

src

main

java

nl.tudelft.jpacman

board

game

level

npc

sprite

ui

Launcher

PacmanConfigurationExce

resources

test

target

checkstyle

README.md

build passing

JPacman-Framework

About

Pacman-like game used for teaching software testing. It exposes students to the use of git, maven, JUnit, and mockito.

Parts of the code are well tested, whereas others are left untested intentionally. As a student in software testing, you can extend the test suite, or use the framework to build extensions in a test-driven way. As a teacher, you can use the framework to create your own testing exercises.

As a starting point for working on your own solution, a template solution is available.

CheckStyle Scan

Rules: <active configuration>

<active configuration>

Sun Checks

Google Checks

Either select Sun Checks or Google Checks

Project Commit Pull Requests Bookmarks Structure

Maven Notifications

Git TODO Problems Terminal Services Build CheckStyle Dependencies

work-in-progress

jpacman-framework – README.md

jpacman-framework

Launcher

Git:

Project

jpacman-framework ~/FEUP/2022-202

.idea

doc

rulesets

src

main

java

nl.tudelft.jpacman

board

game

level

npc

sprite

ui

Launcher

PacmanConfigurationExce

resources

test

target

checkstyle

README.md

build passing

JPacman-Framework

About

Pacman-like game used for teaching software testing. It exposes students to the use of git, maven, JUnit, and mockito.

Parts of the code are well tested, whereas others are left untested intentionally. As a student in software testing, you can extend the test suite, or use the framework to build extensions in a test-driven way. As a teacher, you can use the framework to create your own testing exercises.

As a starting point for working on your own solution, a template solution is available.

CheckStyle Scan

Rules: Google Checks

No scan has been run as yet

Run the plugin

Project Commit Pull Requests Bookmarks Structure Maven Notifications

Git TODO Problems Terminal Services Build CheckStyle Dependencies

Run Checkstyle on all files in the current project

work-in-progress

jpacman-framework – README.md

jpacman-framework

Launcher

Git:

Project

jpacman-framework ~/FEUP/2022-202

.idea

doc

rulesets

src

main

java

nl.tudelft.jpacman

board

game

level

npc

sprite

ui

Launcher

PacmanConfigurationExce

resources

test

target

README.md

build passing

JPacman-Framework

About

Pacman-like game used for teaching software testing. It exposes students to the use of git, maven, JUnit, and mockito.

Parts of the code are well tested, whereas others are left untested intentionally. As a student in software testing, you can extend the test suite, or use the framework to build extensions in a test-driven way. As a teacher, you can use the framework to create your own testing exercises.

As a starting point for working on your own solution, a template solution is available.

CheckStyle Scan

Rules: Google Checks

Scanning current project...

A scan is in progress

Project Commit Pull Requests Bookmarks Structure

Maven Notifications

work-in-progress

jpacman-framework – README.md

jpacman-framework

Launcher

Git:

Project

jpacman-framework ~/FEUP/2022-202

.idea

doc

rulesets

src

main

java

nl.tudelft.jpacman

board

game

level

npc

sprite

ui

Launcher

PacmanConfigurationExce

resources

test

target

README.md

build passing

JPacman-Framework

About

Pacman-like game used for teaching software testing. It exposes students to the use of git, maven, JUnit, and mockito.

Parts of the code are well tested, whereas others are left untested intentionally. As a student in software testing, you can extend the test suite, or use the framework to build extensions in a test-driven way. As a teacher, you can use the framework to create your own testing exercises.

As a starting point for working on your own solution, a template solution is available.

CheckStyle Scan

Rules: Google Checks

Checkstyle found 1,470 item(s) in 40 file(s)

Action.java : 1 item(s)

'method def modifier' has incorrect indentation level 4, expected level should be 2. (13:5) [Indentation]

AnimatedSprite.java : 64 item(s)

'member def modifier' has incorrect indentation level 4, expected level should be 2. (16:5) [Indentation]

'member def modifier' has incorrect indentation level 4, expected level should be 2. (21:5) [Indentation]

'member def modifier' has incorrect indentation level 4, expected level should be 2. (26:5) [Indentation]

'member def modifier' has incorrect indentation level 4, expected level should be 2. (31:5) [Indentation]

'member def modifier' has incorrect indentation level 4, expected level should be 2. (36:5) [Indentation]

'member def modifier' has incorrect indentation level 4, expected level should be 2. (41:5) [Indentation]

Project

Commit

Pull Requests

Bookmarks

Structure

Maven

Notifications

work-in-progress

jpacman-framework – Action.java

an-framework 〉 src 〉 main 〉 java 〉 nl 〉 tudelft 〉 jpacman 〉 ui 〉 Action 〉 doAction

Launcher

Git:

Project

README.md

Action.java

```
3  /**
4   * An action that can be executed.
5   *
6   * @author Jeroen Roosen
7   */
8  public interface Action {
9
10     /**
11      * Executes the action.
12      */
13     void doAction();
14 }
15
```

jpacman-framework ~/FEUP/2022-202
.idea
doc
rulesets
src
main
java
nl.tudelft.jpacman
board
game
level
npc
sprite
ui
Launcher
PacmanConfigurationExce
resources
test
target

CheckStyle Scan

Rules: Google Checks

Checkstyle found 1,470 item(s) in 40 file(s)
Action.java : 1 item(s)
'method def modifier' has incorrect indentation level 4, expected level should be 2. (13:5) [Indentation]
AnimatedSprite.java : 64 item(s)
'member def modifier' has incorrect indentation level 4, expected level should be 2. (16:5) [Indentation]
'member def modifier' has incorrect indentation level 4, expected level should be 2. (21:5) [Indentation]
'member def modifier' has incorrect indentation level 4, expected level should be 2. (26:5) [Indentation]
'member def modifier' has incorrect indentation level 4, expected level should be 2. (31:5) [Indentation]
'member def modifier' has incorrect indentation level 4, expected level should be 2. (36:5) [Indentation]
'member def modifier' has incorrect indentation level 4, expected level should be 2. (41:5) [Indentation]

13:6 LF UTF-8 4 spaces work-in-progress

- Perform the exact same steps for the two other tools:
  - PMD, https://pmd.github.io
  - SpotBugs, https://spotbugs.github.io/index.html