

Reducing Entry Barriers for Online Programming Exercises

Theia in an Education Environment



Matthias Linhuber






Onboarding Developers is hard!



Onboarding Students is even harder!



Wouldn't it be great if...

Google Suche

Auf gut Glück!

Google gibt es auch auf: [English](#)





... lets see how this works

Who am I?



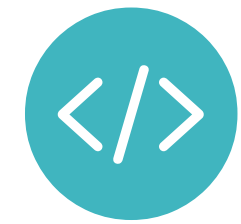
Matthias Linhuber



Doctoral Student at TUM



Educator at heart



Software and Infrastructure Architect



Research Areas

- Container-based Software Engineering
- Infrastructure Orchestration
- Scaling Education Technology



Project Team



Dennis Jandow



Yannik Schmidt



Robert Jandow



Prof. Stephan Krusche

A dark, atmospheric photograph of a historic building, likely a clock tower or bell tower, featuring two large circular clock faces with Roman numerals. The building has a balcony with a decorative railing. The text "Some Context..." is overlaid in white, bold, sans-serif font across the center of the image. The background shows a dark sky and the silhouettes of other buildings in the distance.

Some Context...

TUM - School of CIT

132

Professors

15.092

Students



1,603

Doctoral Candidates

53%

International Students

1,593

Staff Members

22%

Female Students

Courses at TUM CIT

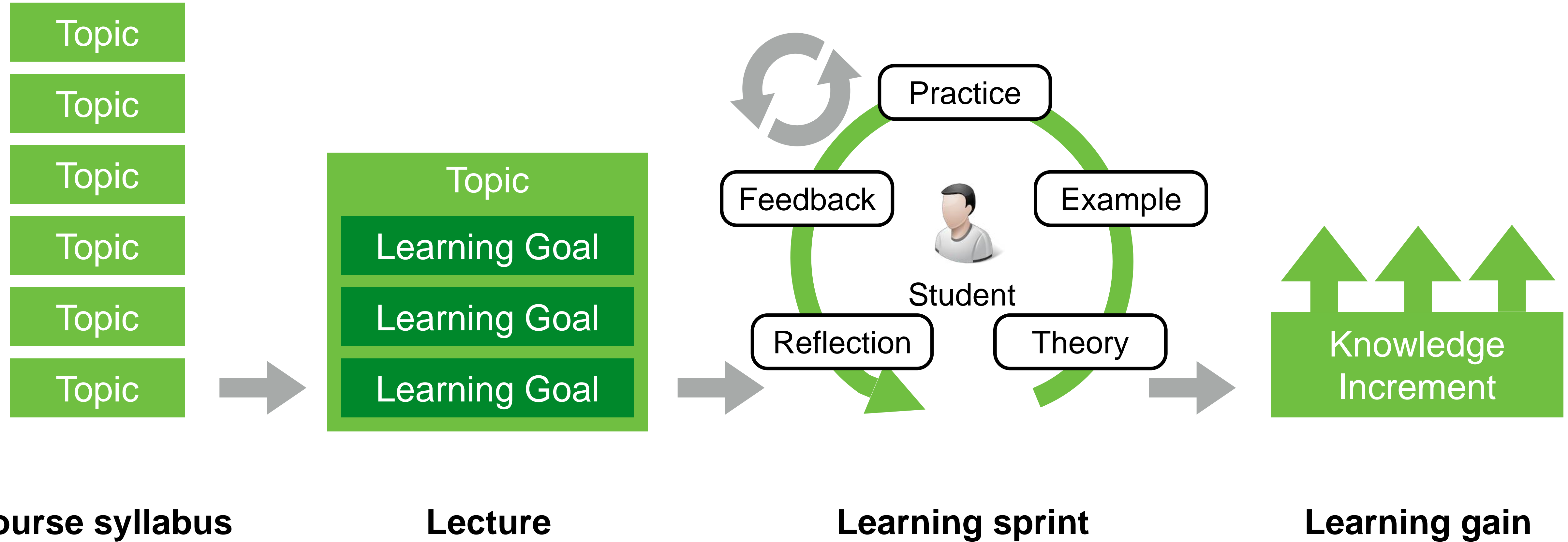


Interactive Learning

- Topic
- Topic
- Topic
- Topic
- Topic
- Topic

Course syllabus

Interactive Learning



Artemis

13

Universities

1

Release per Week

20+

Developers

LTI

compatible

80.000

Students

200+

Courses

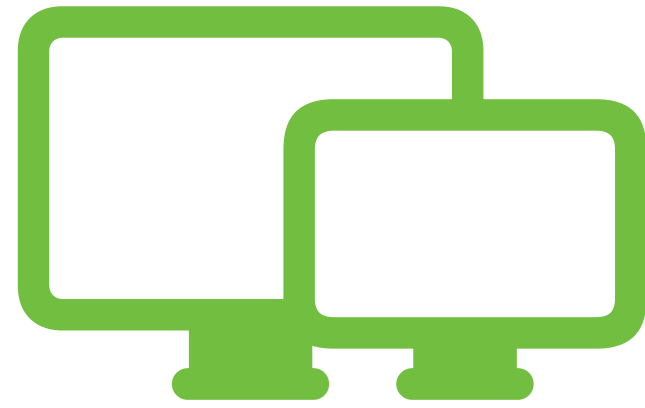
15.000

Students in 50+ exams

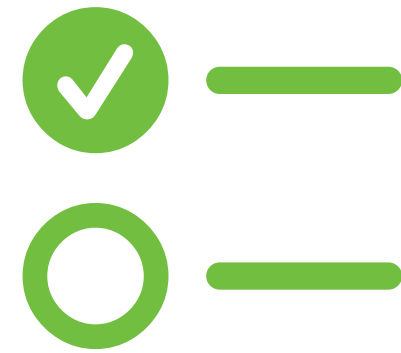
5.000

unique users per semester





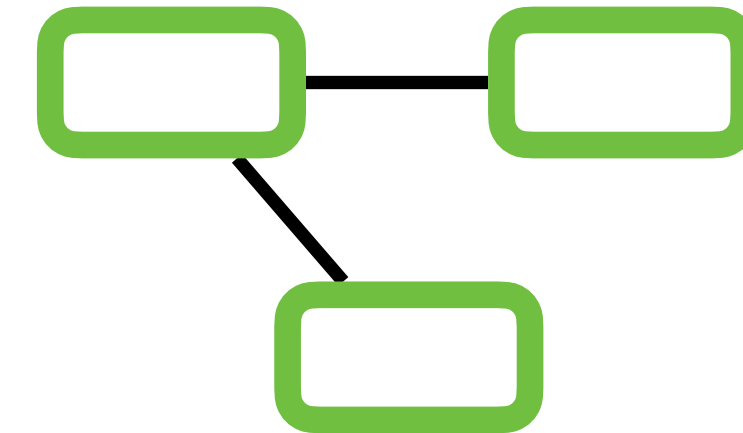
Programming



Quiz



Text



Modeling



Scalability



Usability



Instant Feedback

Artemis - Programming Exercises

The screenshot shows the Artemis interface for the 'L00E06 Observer Pattern' exercise. The top navigation bar includes 'Artemis 7.6.5', 'Course Overview', 'Course Management', a notification bell, and a moon icon. The breadcrumb trail is 'Courses > Patterns in Software Engineering (WS24/25) > Exercises > L00E06 Observer Pattern'. The left sidebar contains 'Patterns in Software Engineering (WS24/25)', 'Exercises', 'Lectures', 'Statistics', 'Communication', and 'FAQ'. The main content area displays the exercise title 'L00E06 Observer Pattern' with tags 'repetition' and 'Medium', and a 'Submission due: 4 days ago' badge. Below this are 'Points: 3' and 'Assessment: automatic'. A 'Practice' button is visible. The 'Instructor actions' section includes 'View', 'Scores', 'Participations', 'Grading', and 'Edit'. A 'Tasks' row shows ten question marks. The main text area is titled 'Observer Pattern' and contains a detailed description of the pattern, a 'Problem Statement' about a thermometer, and a note about existing source code. A code block at the bottom shows the start of an abstract class:

```
«abstract»
```


Artemis - Programming Exercises

The screenshot shows the Artemis 7.6.5 interface. The top navigation bar includes 'Artemis 7.6.5', 'Course Overview', 'Course Management', a notification bell, and a moon icon. The breadcrumb trail is 'Courses > Patterns in Software Engineering (WS24/25) > Exercises > L00E06 Observer Pattern'. The left sidebar contains 'Patterns in Software Engineering (WS24/25)', 'Exercises', 'Lectures', 'Statistics', 'Communication', and 'FAQ'. The main content area is titled 'Part 1: Connect model and views using the Observer Pattern' and lists six exercises, all marked as '100% passed'.

Artemis 7.6.5 Course Overview Course Management

Courses > Patterns in Software Engineering (WS24/25) > Exercises > L00E06 Observer Pattern

Patterns in Software Engineering (WS24/25)

Exercises Lectures Statistics Communication FAQ

Part 1: Connect model and views using the Observer Pattern

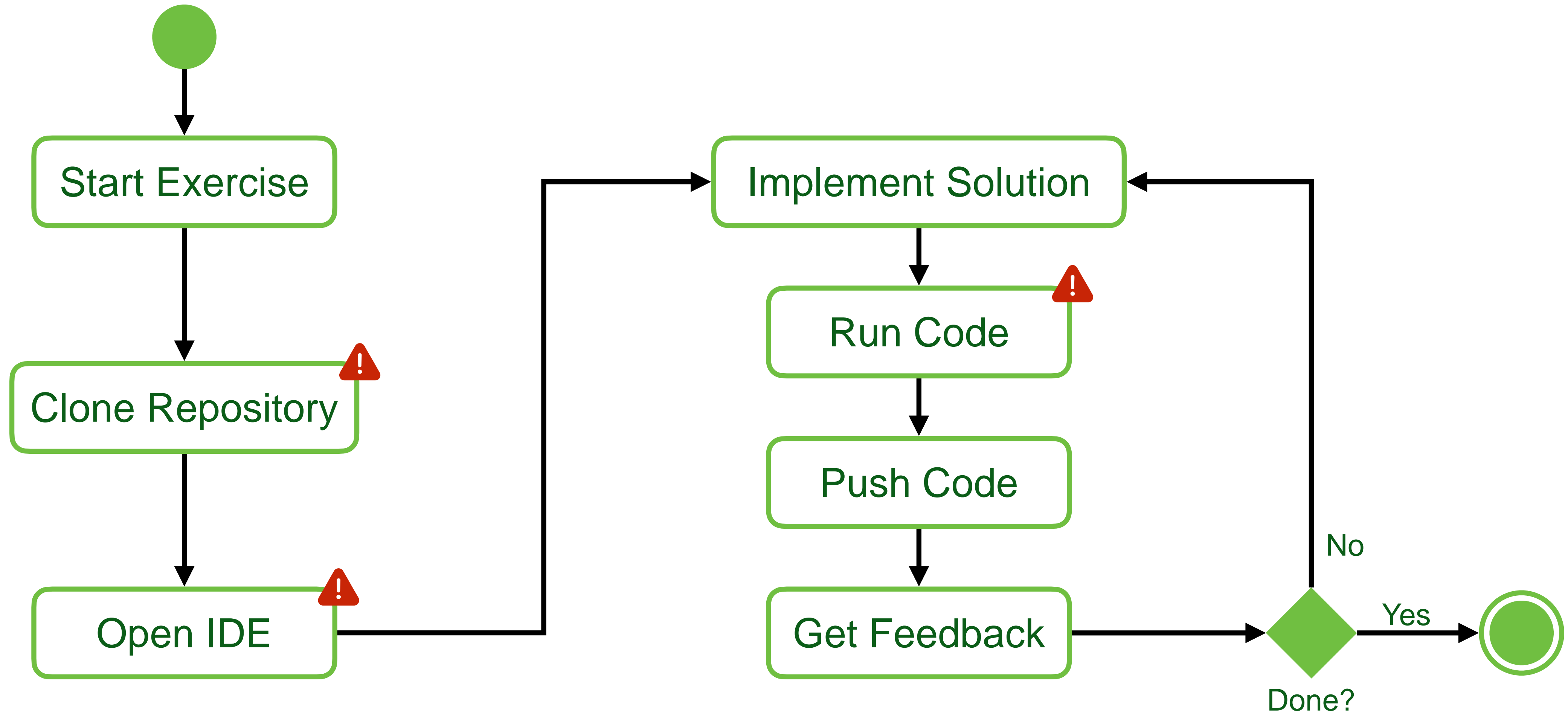
1. **Implement notifyObservers in Subject 100% passed**
Start by finalizing the implementation of the observer pattern. The `Subject` has a `notifyObservers(T)`-methods that isn't implemented yet. Iterate through all subscribed observers and inform them about the updated state by calling `onUpdate(T)`.
2. **Make TemperatureModel Observable 100% passed**
Make `TemperatureModel` observable by extending `Subject`. Notify the observers of `TemperatureModel` when the temperature changes by using the corresponding method.
3. **Let GaugeGUI implement Observer 100% passed**
Also implement the `onUpdate(Double)` method in `GaugeGUI`. The TODO-Comments in the code explain in detail how this method should work. You can use the method `intValue()` in order to convert a `Double` to an `int`.
4. **Let SliderGUI implement Observer 100% passed**
Also implement the `onUpdate(Double)` method in `SliderGUI`.
5. **Let TemperatureGUI implement Observer 100% passed**
 1. **Implement onUpdate(Double) in CelsiusGUI 100% passed**
 2. **Implement onUpdate(Double) in FahrenheitGUI 100% passed**
6. **Connect the observers with the subject 100% passed**
Connect the views `GaugeGUI`, `SliderGUI`, `TemperatureGUI` in their constructors (as observers) with the model `TemperatureModel` by invoking `addObserver(this)` on the `model` attribute.

After Part 1, your source code should follow this UML class diagram:

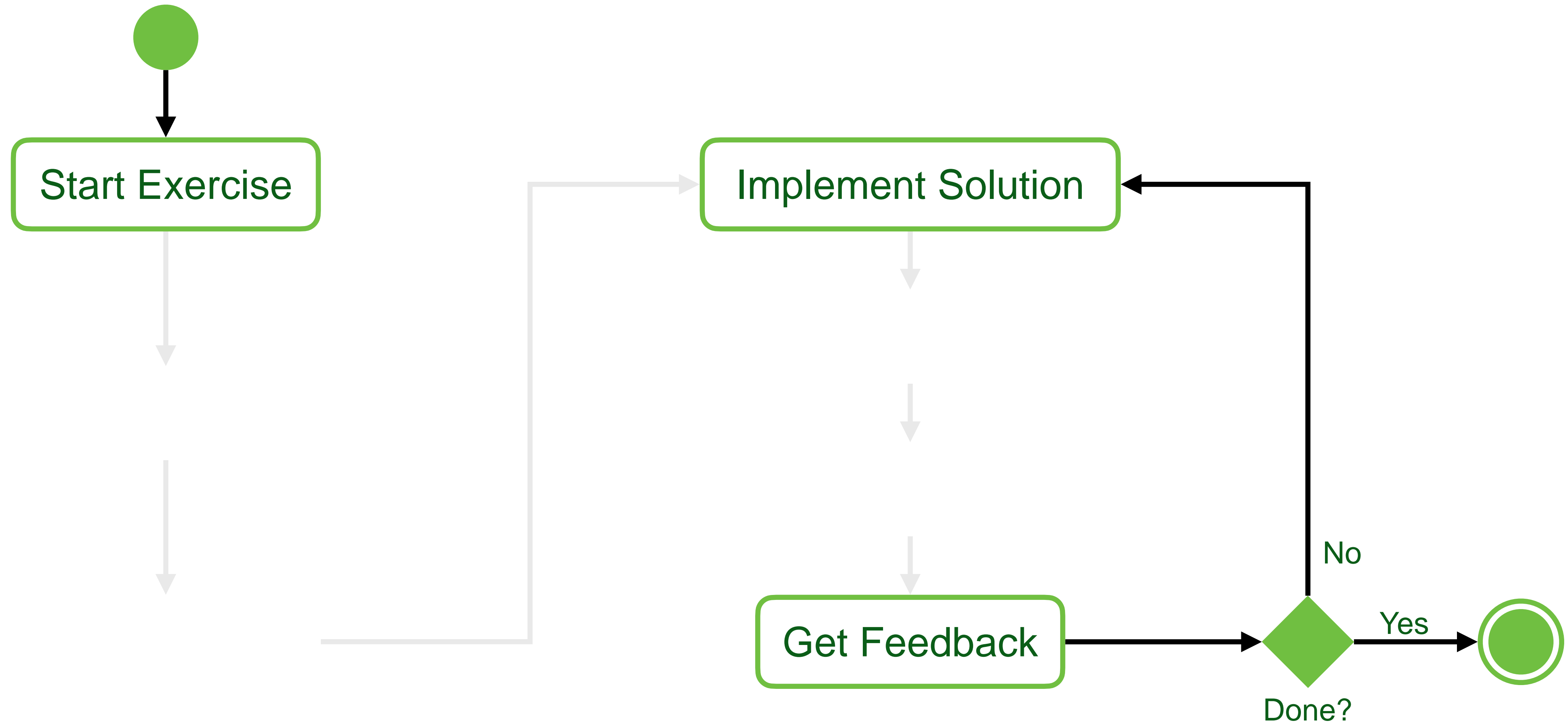
«abstract»
Subject

Con

Problem - Many things can go wrong



Problem - Many things can go wrong





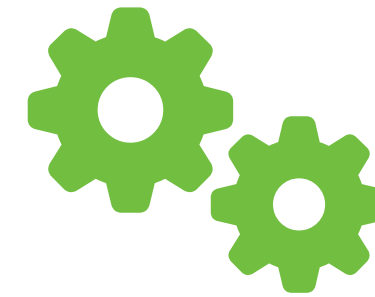
Preconfigured Online IDE

Motivation for online IDEs



Social

Interdisciplinary Students
Inclusion and Accessibility



Technology

IDE setup complexity
Language Pool



Learning

Focus on programming
Constructive Alignment

Theia Cloud



The screenshot shows the Eclipse Theia IDE interface in a browser window. The address bar displays the URL: `instance.theia.artemis.cit.tum.de/a39a2ce3-9561-449c-a02f-4666`. The interface includes a menu bar (File, Edit, Selection, View, Go, Run, Terminal, Help) and a sidebar with icons for Explorer, Search, Run and Debug, Source Control, Testing, and Extensions. The main content area displays the "Welcome" page for Eclipse Theia IDE, version 1.52.0, with a VS Code API version of 1.91.1. The page features a large "THEIA IDE" logo and a navigation menu on the right side.

Eclipse Theia IDE

Version 1.52.0
VS Code API Version: 1.91.1

What is this?

The Eclipse Theia IDE is a modern and open IDE for cloud and desktop. The Theia IDE is based on the [Theia platform](#). The IDE is available as a [downloadable desktop application](#). You can also [try the latest version of the Theia IDE online](#). The online test version is limited to 30 minutes per session and hosted via [Theia Cloud](#).

Extending/Customizing the Theia IDE

You can extend the Theia IDE at runtime by installing VS Code extensions, e.g. from the [OpenVSX registry](#), an open marketplace for VS Code extensions. Just open the extension view or browse [OpenVSX online](#). Furthermore, the Theia IDE is based on the flexible Theia platform. Therefore, the Theia IDE can serve as a **template** for building custom tools and IDEs. Browse [the documentation](#) to help you customize and build your own Eclipse Theia-based product.

Professional Support

Professional support, implementation services, consulting and training for building tools like Theia IDE and for building other tools based on Eclipse Theia is available by selected companies as listed on the [Theia support page](#).

Reporting feature requests and bugs

The features in the Eclipse Theia IDE are based on Theia and the included extensions/plugins. For bugs in

Start

- New File...
- Open
- Open Workspace

Recent

- project /home/project

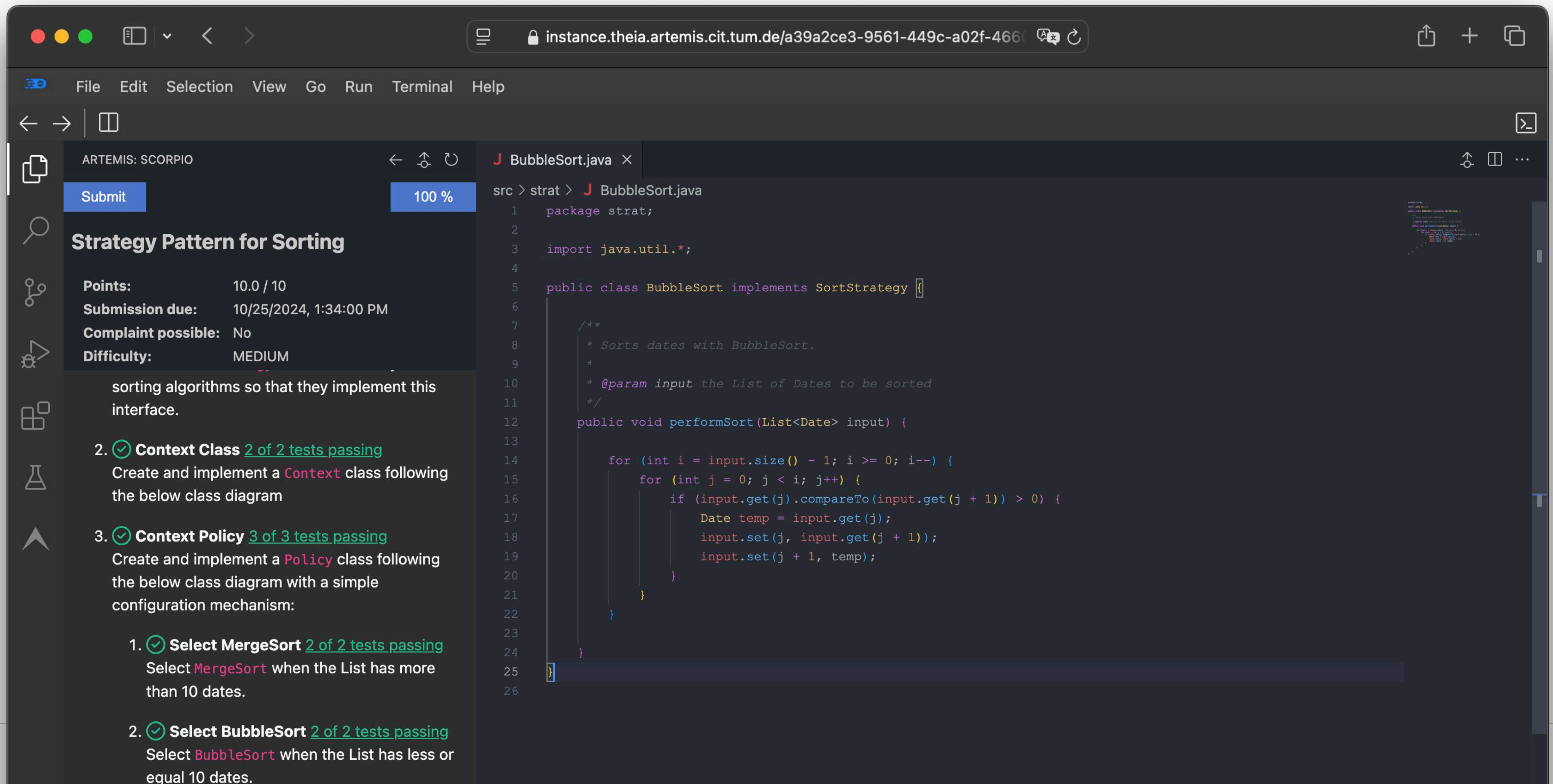
Settings

- Open Settings
- Open Keyboard Shortcuts

Help

- Documentation
- VS Code API Compatibility
- Building a New Extension
- Building a New Plugin

Theia Artemis Integration - Scorpio



The screenshot displays the Theia IDE interface. On the left, a sidebar shows the task details for "Strategy Pattern for Sorting" in the "ARTEMIS: SCORPIO" environment. The task is marked as "100%" complete. The main editor area shows the implementation of the `BubbleSort` class in `BubbleSort.java`.

Task Details:

- Submit: 100 %
- Points: 10.0 / 10
- Submission due: 10/25/2024, 1:34:00 PM
- Complaint possible: No
- Difficulty: MEDIUM

Task Description:

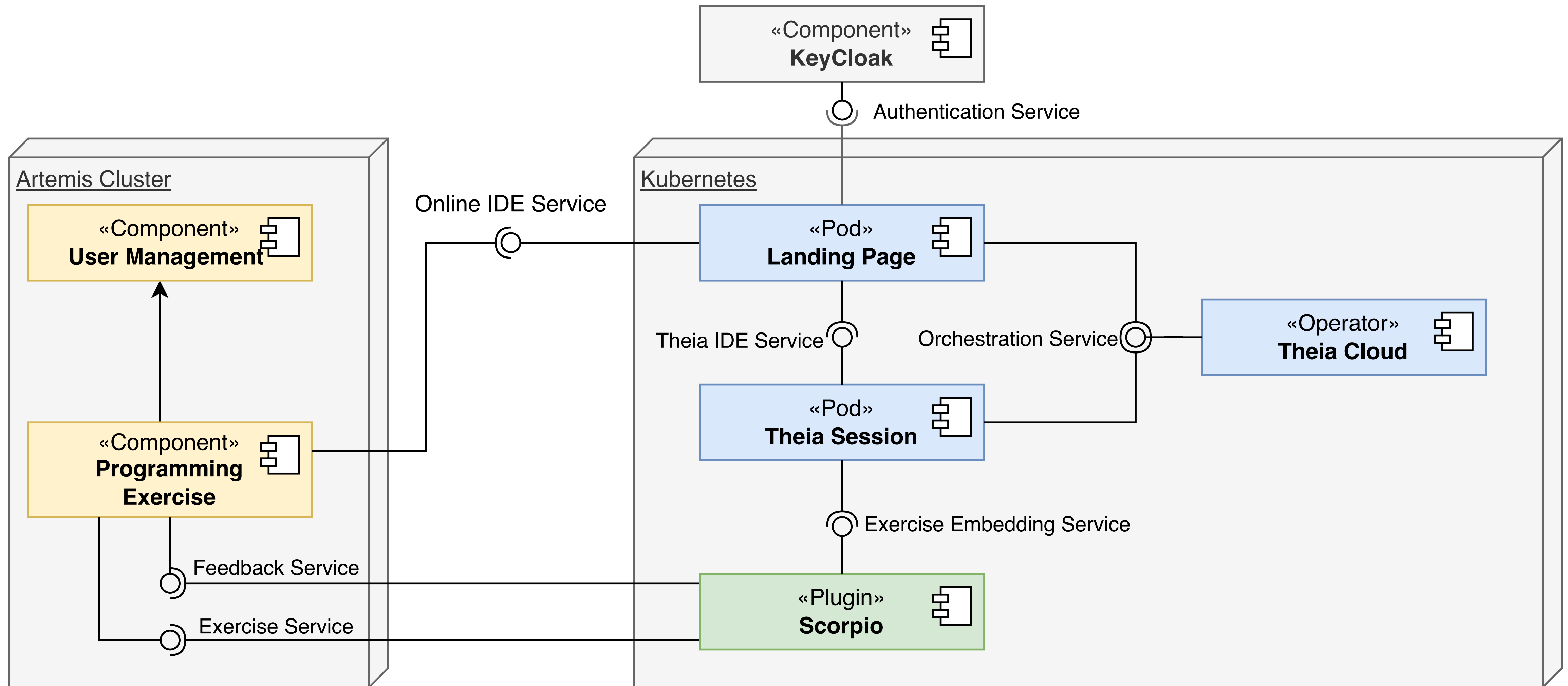
sorting algorithms so that they implement this interface.

2. **Context Class** [2 of 2 tests passing](#)
Create and implement a `Context` class following the below class diagram
3. **Context Policy** [3 of 3 tests passing](#)
Create and implement a `Policy` class following the below class diagram with a simple configuration mechanism:
 1. **Select MergeSort** [2 of 2 tests passing](#)
Select `MergeSort` when the List has more than 10 dates.
 2. **Select BubbleSort** [2 of 2 tests passing](#)
Select `BubbleSort` when the List has less or equal 10 dates.

Code Implementation:

```
src > strat > BubbleSort.java
1 package strat;
2
3 import java.util.*;
4
5 public class BubbleSort implements SortStrategy {
6
7     /**
8      * Sorts dates with BubbleSort.
9      *
10     * @param input the List of Dates to be sorted
11     */
12     public void performSort(List<Date> input) {
13
14         for (int i = input.size() - 1; i >= 0; i--) {
15             for (int j = 0; j < i; j++) {
16                 if (input.get(j).compareTo(input.get(j + 1)) > 0) {
17                     Date temp = input.get(j);
18                     input.set(j, input.get(j + 1));
19                     input.set(j + 1, temp);
20                 }
21             }
22         }
23     }
24 }
25
26
```


System Architecture





Demo

Courses
Test Server

Your current courses (2)

Sort

Search...

Course Enrollment

T Theia Java Course

Score
No statistics available

Next Exercise:
No exercise planned

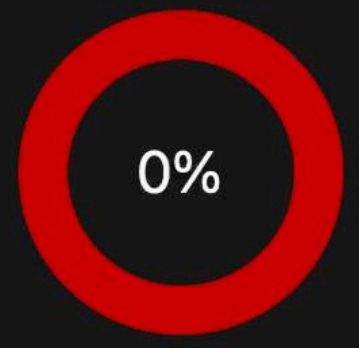
[Manage Course →](#)

T Theia Test

Score
0 / 23 Points

Next Exercise:
onlineide

[Manage Course →](#)



Looking for old courses? Click [here](#)

Theia Test

Exams

Exercises

Statistics

Communication

Exercises

Manage

Refresh

Search...

Filter

Current (1)

Merge Algorithms in Java
Dec 15, 2024 No graded result

No Date (4)

Merge Algorithms in Java

Submission due: in a month

Points: 1 Assessment: automatic

Code

No graded result

Instructor actions:

View

Scores

Participations

Grading

Edit

Tasks: ? ? ? ? ? ? ?

Sorting with the Strategy Pattern

In this exercise, we want to implement sorting algorithms and choose them based on runtime specific variables.

Part 1: Sorting

First, we need to implement two sorting algorithms, in this case MergeSort and BubbleSort.

You have the following tasks:

- Implement Bubble Sort** No results
Implement the method `performSort(List<Date>)` in the class `BubbleSort`. Make sure to follow the Bubble Sort algorithm exactly.
- Implement Merge Sort** No results
Implement the method `performSort(List<Date>)` in the class

Communication

Search for a message

Unresolved Own Reacted

Date: ↓

No messages found

+ New message

Theia Test

Exams

Exercises

Statistics

Communication

Exercises

Manage

Refresh

Search...

Filter

Current (1)

Merge Algorithms in Java
Dec 15, 2024

No Date (4)

Merge Algorithms in Java

Points: 1 Assessment: automatic

Submission due: in a month

Code

No graded result

Instructor actions: View Scores Participations Grading Edit

Tasks: ? ? ? ? ? ? ?

Sorting with the Strategy Pattern

In this exercise, we want to implement sorting algorithms and choose them based on runtime specific variables.

Part 1: Sorting

First, we need to implement two sorting algorithms, in this case MergeSort and BubbleSort.

You have the following tasks:

- Implement Bubble Sort** No results
Implement the method `performSort(List<Date>)` in the class `BubbleSort`. Make sure to follow the Bubble Sort algorithm exactly.
- Implement Merge Sort** No results
Implement the method `performSort(List<Date>)` in the class

Communication

Search for a message

Unresolved Own Reacted Date: ↓

No messages found

+ New message



EXPLORER

... Welcome ×

> OPEN EDITORS

▼ PROJECT

- > persisted /lost+found
- > theiatesttestexercise-yanni...

1

> JAVA PROJECTS

> TIMELINE

Eclipse Theia IDE

Version 1.52.0
VS Code API Version: 1.91.1

What is this?

The Eclipse Theia IDE is a modern and open IDE for cloud and desktop. The Theia IDE is based on the [Theia platform](#). The IDE is available as a [downloadable desktop application](#). You can also [try the latest version of the Theia IDE online](#). The online test version is limited to 30 minutes per session and hosted via [Theia Cloud](#).

Extending/Customizing the Theia IDE

You can extend the Theia IDE at runtime by installing VS Code extensions, e.g. from the [OpenVSX registry](#), an open marketplace for VS Code extensions. Just open the extension view or browse [OpenVSX online](#). Furthermore, the Theia IDE is based on the flexible Theia platform. Therefore, the Theia IDE can serve as a **template** for building custom tools and IDEs. Browse [the documentation](#) to help you customize and build your own Eclipse Theia-based product.

Professional Support

Professional support, implementation services, consulting and training for building tools like Theia IDE and for building other tools based on Eclipse Theia is available by selected companies as listed on the [Theia support page](#).

Reporting feature requests and bugs

The features in the Eclipse Theia IDE are based on Theia and the included extensions/plugins. For bugs in Theia please consider opening an issue in the [Theia project on Github](#). Eclipse Theia IDE only packages existing functionality into a product and installers for the product. If you believe there is a mistake in packaging, something needs to be added to the packaging or the installers do not work properly, please [open an issue on Github](#) to let us know.

Source Code

The source code of Eclipse Theia IDE is available on [Github](#).

Documentation



Start

- New File...
- Open
- Open Workspace

Recent

- project /home/project
- theiatesttestexercise-yannik.schmidt theiatesttestexercise-yannik.schmidt

Settings

- Open Settings
- Open Keyboard Shortcuts

Help

- Documentation
- VS Code API Compatibility
- Building a New Extension
- Building a New Plugin

EXPLORER

OPEN EDITORS

PROJECT

- persisted/lost+found
- theiatesttestexercise-yanni...
- .git
- .gradle
- bin
- gradle
- src/com/test
 - BubbleSort.java
 - Client.java
 - MergeSort.java
 - .gitattributes
 - .gitignore
 - build.gradle
 - gradlew
 - gradlew.bat
 - settings.gradle

JAVA PROJECTS

TIMELINE

```
theiatesttestexercise-yannik.schmidt > src > com > test > Client.java > Client > main(String[])
20 private static final int RANDOM_FLOOR = 5;
21
22 private static final int RANDOM_CEILING = 15;
23
24 private Client() {
25 }
26
27 /**
28  * Main method.
29  * Add code to demonstrate your implementation here.
30  *
31  * @param args command line arguments
32  */
33 public static void main(String[] args) throws ParseException {
34
35     // TODO: Init Context and Policy
36
37
38     // Run multiple times to simulate different sorting strategies for different Array sizes
39     for (int i = 0; i < ITERATIONS; i++) {
40         List<Date> dates = createRandomDatesList();
41
42         // TODO: Configure context
43
44         System.out.print(s:"Unsorted Array of course dates = ");
45         printDateList(dates);
46
47         // TODO: Sort dates
48
49         System.out.print(s:"Sorted Array of course dates = ");
50         printDateList(dates);
51     }
52
53
54 /**
```

Failed to detect repo: Please sign in

Login

File Edit Selection View Go Run Terminal Help

← → | □

EXPLORER

Welcome Client.java M X

> OPEN EDITORS

PROJECT

- persisted/lost+found
- theiatesttestexercice-ya...

 - .git
 - .gradle
 - bin
 - gradle
 - src/com/test
 - BubbleSort.java
 - Client.java M**
 - MergeSort.java
 - .gitattributes
 - .gitignore
 - build.gradle
 - gradlew
 - gradlew.bat
 - settings.gradle

JAVA PROJECTS

TIMELINE

```
theiatesttestexercice-yannik.schmidt > src > com > test > Client.java > Client > main(String[])
20 private static final int RANDOM_FLOOR = 5;
21
22 private static final int RANDOM_CEILING = 15;
23
24 private Client() {
25 }
26
27 /**
28  * Main method.
29  * Add code to demonstrate your implementation here.
30  *
31  * @param args command line arguments
32  */
33 Run | Debug
34 public static void main(String[] args) throws ParseException {
35
36 // TODO: Init Context and Policy
37 System.out.println(x:"Hello TheiaCon! :");
38
39 // Run multiple times to simulate different sorting strategies for different Array sizes
40 for (int i = 0; i < ITERATIONS; i++) {
41     List<Date> dates = createRandomDatesList();
42
43     // TODO: Configure context
44
45     System.out.print(s:"Unsorted Array of course dates = ");
46     printDateList(dates);
47
48     // TODO: Sort dates
49
50     System.out.print(s:"Sorted Array of course dates = ");
51     printDateList(dates);
52     System.out.println();
53     System.out.println();
54     System.out.println(x:"Hello TheiaCon! :");
55 }
```


EXPLORER

... Welcome Client.java M X

> OPEN EDITORS

PROJECT

- persisted/lost+found
- theiatesttestexercise-ya... ●
- .git
- .gradle
- bin
- gradle
- src/com/test ●
- BubbleSort.java
- Client.java M
- MergeSort.java
- .gitattributes
- .gitignore
- build.gradle
- gradlew
- gradlew.bat
- settings.gradle

JAVA PROJECTS

TIMELINE

```
theiatesttestexercise-yannik.schmidt > src > com > test > Client.java > ...
30 *
31 * @param args command line arguments
32 */
33 Run | Debug
34 public static void main(String[] args) throws ParseException {
35
36     // TODO: Init Context and Policy
37     System.out.println(x:"Hello TheiaCon! :)");
38
39     // Run multiple times to simulate different sorting strategies for different Array sizes
40     for (int i = 0; i < ITERATIONS; i++) {
41         List<Date> dates = createRandomDatesList();
42
43         // TODO: Configure context
44         System.out.print(s:"Unsorted Array of course dates = ");
45         printDateList(dates);
46
47         // TODO: Sort dates
48
49         System.out.print(s:"Sorted Array of course dates = ");
50         printDateList(dates);
51         System.out.println();
52         System.out.println();
53         System.out.println(x:"Hello TheiaCon! :)");
54     }
55 }
56
57 /**
58 * Generates a List of random Date objects with random List size between
59 * {@link #RANDOM_FLOOR} and {@link #RANDOM_CEILING}.
60 *
61 * @return a List of random Date objects
62 * @throws ParseException if date string cannot be parsed
63 */
64 private static List<Date> createRandomDatesList() throws ParseException {
```

Terminal window showing command output and logs. The output includes "Hello TheiaCon! :)", "Unsorted Array of course dates = ", "Sorted Array of course dates = ", and "Hello TheiaCon! :)".

Theia Test

Exams

Exercises

Statistics

Communication

Exercises

Manage

Refresh

Search...

Filter

Current (1)

Merge Algorithms in Java

Dec 15, 2024 7.7% (preliminary)

No Date (4)

Merge Algorithms in Java

Points: 0.1 of 1

Assessment: automatic

Submission due: in a month

Code

7.7% (preliminary) (a minute ago) GRADED

Instructor actions:

View

Scores

Participations

Grading

Edit

Recent results:

0% Build failed (preliminary) (17 minutes ago)

GRADED

7.7% (preliminary) (a minute ago)

GRADED

Show all results

Tasks:



Sorting with the Strategy Pattern

In this exercise, we want to implement sorting algorithms and choose them based on runtime specific variables.

Communication

Search for a message

Unresolved

Own

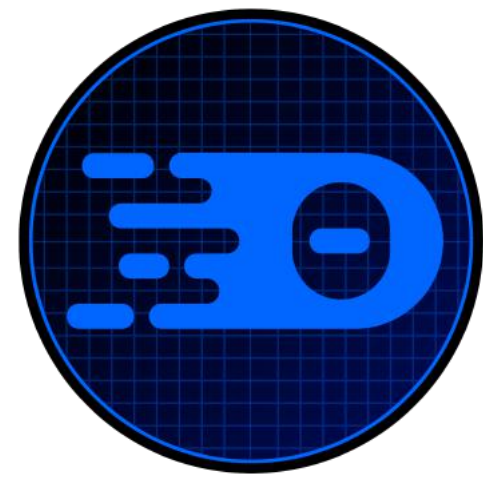
Reacted

Date: ↓

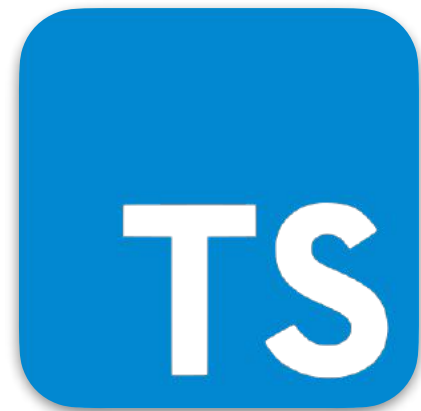
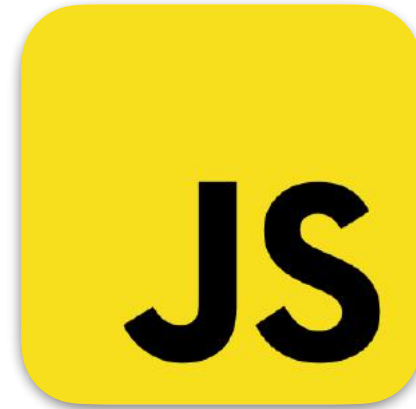
No messages found

+ New message

Contributions



Theia Blueprints



Custom Theia Landing Page

Exercise Autostart



Playground Mode



App Definitions

```
1 apiVersion: theia.cloud/v1beta10
2 kind: AppDefinition
3 metadata:
4   name: my-theia-application
5   namespace: my-namespace
6 ...
```

ARTEMIS: SCORPIO

Submit 100 %

Strategy Pattern for Sorting

Points: 10.0 / 10
Submission due: 10/25/2024, 1:34:00 PM
Complaint possible: No
Difficulty: MEDIUM

sorting algorithms so that they implement this interface.

- Select MergeSort 2 of 2 tests passing**
Select `MergeSort` when the List has more than 10 dates.
- Select BubbleSort 2 of 2 tests passing**
Select `BubbleSort` when the List has less or equal 10 dates.
- Context Class 2 of 2 tests passing**
Create and implement a `Context` class following the below class diagram
- Context Policy 3 of 3 tests passing**
Create and implement a `Policy` class following the below class diagram with a simple configuration mechanism:

4. Complete the `Client` class which demonstrates switching between two strategies at runtime.

```
graph TD
    Client -- context --> Context
    Client -- sortAlgorithm --> SortStrategy
    Context --> Policy
    Context --> BubbleSort
    Context --> MergeSort
    Policy --> SortStrategy
    BubbleSort --> SortStrategy
    MergeSort --> SortStrategy
```




**User Satisfaction
Survey**



**Prewarming and
Predictive Scaling**

Reducing Entry Barriers for Online Programming Exercises

Theia in an Education Environment



matthias.linhuber@tum.de

Matthias Linhuber