Introduction to Benchmark

Database Systems
DataLab, CS, NTHU
Spring, 2025

Outline

- VanillaBench Project
  - Introduction to VanillaBench
  - Setting Benchmark Configurations
  - Starting Up Server for Benchmarking
  - Running Benchmark Client
  - Assignment 2

Outline

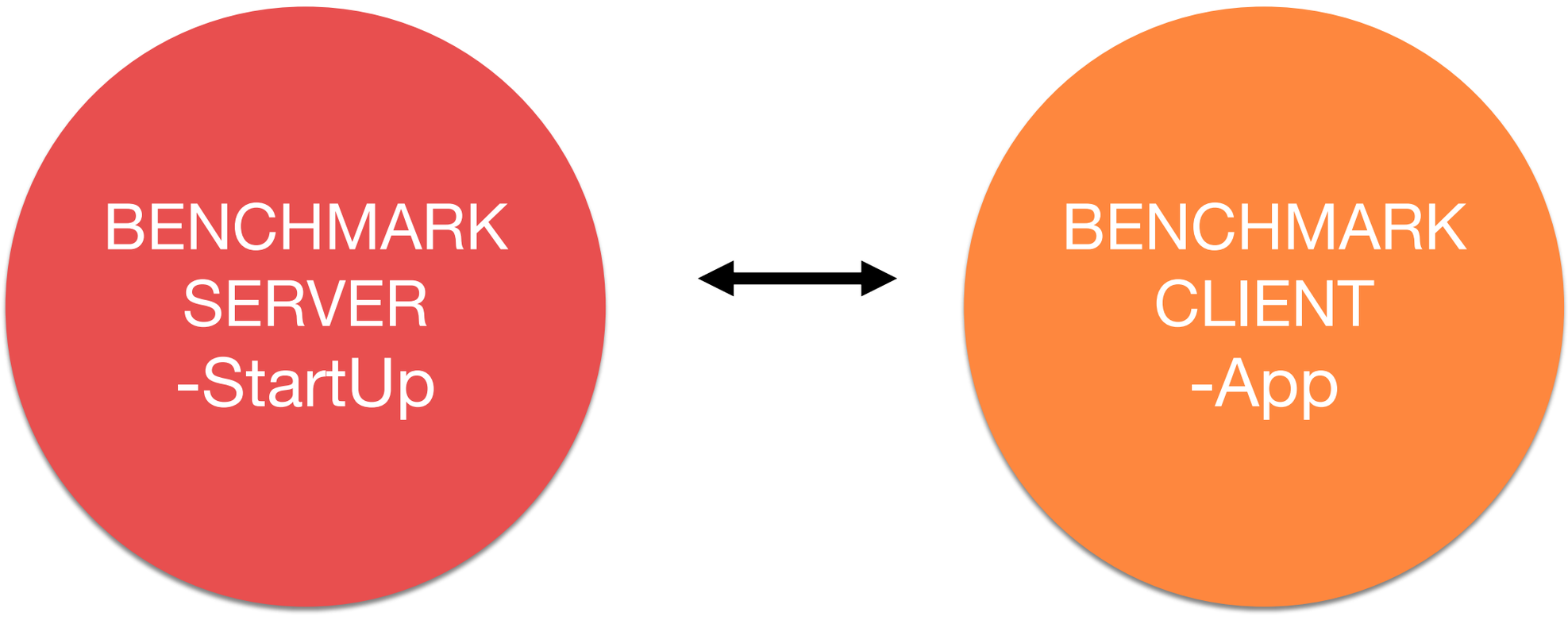
- VanillaBench Project
  - Introduction to VanillaBench
  - Setting Benchmark Configurations
  - Starting Up Server for Benchmarking
  - Running Benchmark Client
  - Assignment 2

VanillaBench

- VanillaBench is a project designed for automatically benchmarking VanillaCore
- It contains several benchmark procedures
- It also has a lot of adjustable testing parameters

[Icon Source]

Two Main Methods



Outline

- VanillaBench Project
  - Introduction to VanillaBench
  - Setting Benchmark Configurations
  - Starting Up Server for Benchmarking
  - Running Benchmark Client
  - Assignment 2

Clone the Project First

- The code of VanillaBench has been pushed to vanilladb repository

- All you need is to clone from the remote repository

```
> git clone
```

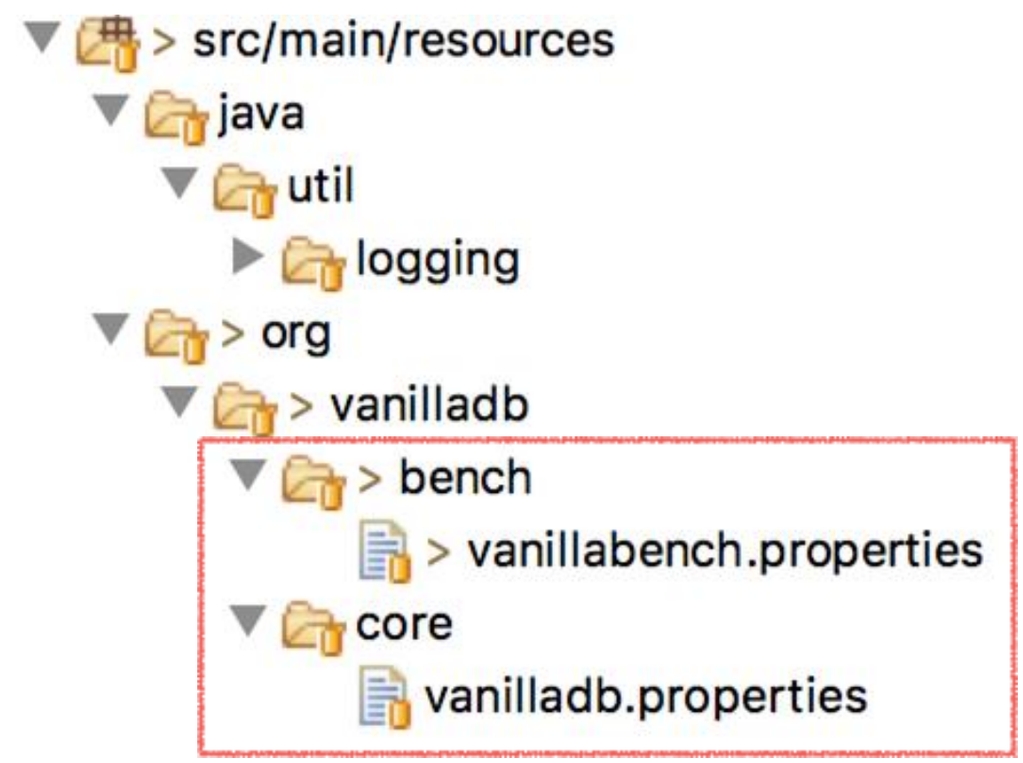
- You can clone from here:

  - https://shwu10.cs.nthu.edu.tw/courses/databases/2025-spring/db25-assignment-2

  - Fork the project first!

Benchmark Setting

- Benchmark project also has its own set of properties files

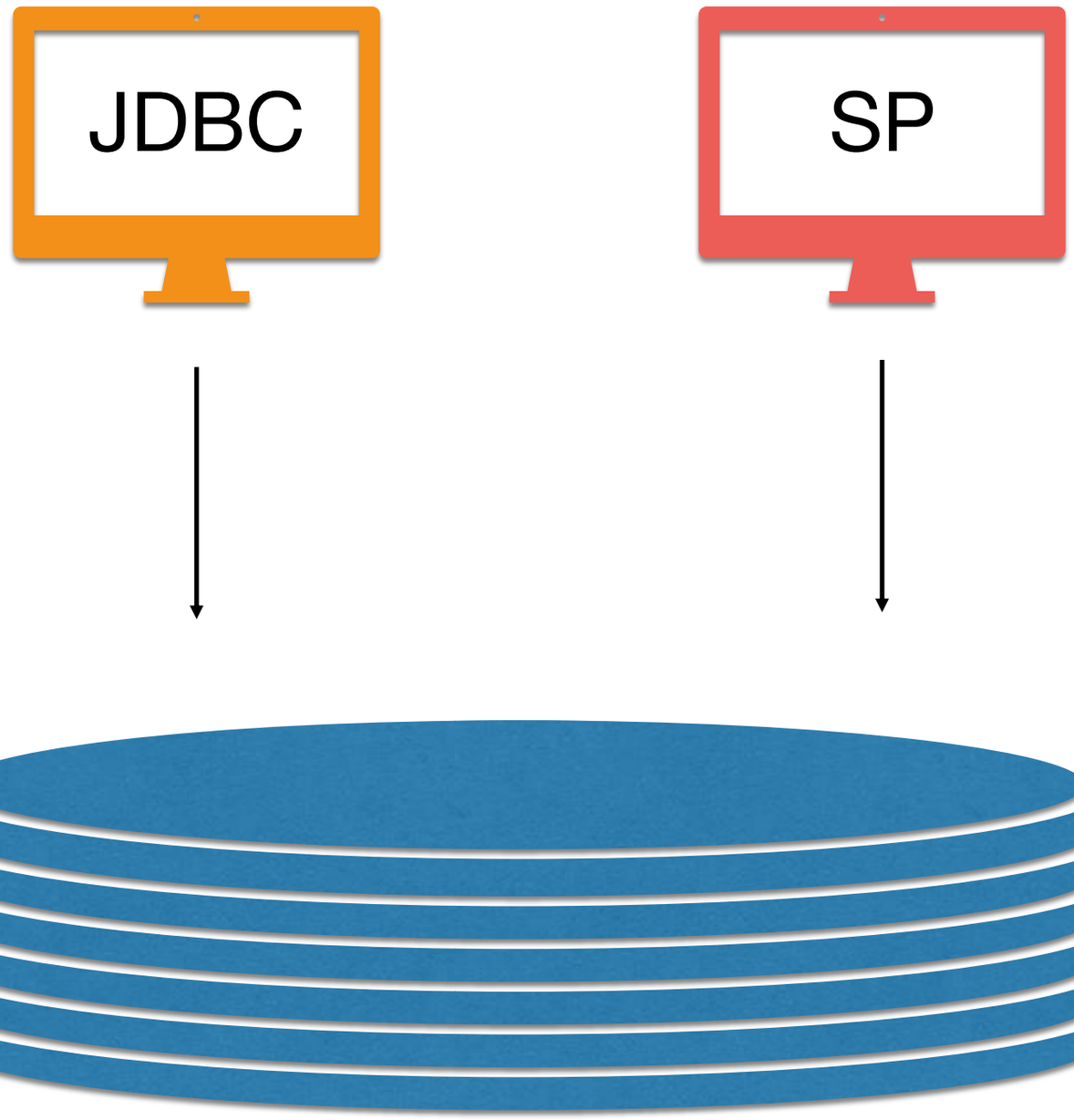


```
#
# Basic Parameters
#

# The running time for warming up before benchmarking
org.vanilladb.bench.VanillaBenchParameters.WARM_UP_INTERVAL=60000
# The running time for benchmarking
org.vanilladb.bench.VanillaBenchParameters.BENCHMARK_INTERVAL=60000
# The number of remote terminal executors for benchmarking
org.vanilladb.bench.VanillaBenchParameters.NUM_RTES=2
# The sleeping time (in milliseconds) between transactions for each RTE
# 0 = no sleeping, 100 is a generally good number for under-loaded workloads
org.vanilladb.bench.VanillaBenchParameters.RTE_SLEEP_TIME=0
# The IP of the target database server
org.vanilladb.bench.VanillaBenchParameters.SERVER_IP=127.0.0.1
# 1 = JDBC, 2 = Stored Procedures
org.vanilladb.bench.VanillaBenchParameters.CONNECTION_MODE=2
# 1 = AS2
org.vanilladb.bench.VanillaBenchParameters.BENCH_TYPE=1
# Whether it enables the built-in profiler on the server
org.vanilladb.bench.VanillaBenchParameters.PROFILING_ON_SERVER=false
# The path to the generated reports
org.vanilladb.bench.VanillaBenchParameters.REPORT_OUTPUT_DIRECTORY=
# The granularity for summarizing the performance of benchmarking
org.vanilladb.bench.VanillaBenchParameters.REPORT_TIMELINE_GRANULARITY=1000
# Whether the RTEs display the results of each transaction
org.vanilladb.bench.VanillaBenchParameters.SHOW_TXN_RESPONSE_ON_CONSOLE=false

# The number of items in the testing data set
org.vanilladb.bench.benchmarks.as2.As2BenchConstants.NUM_ITEMS=100000
# Read count
org.vanilladb.bench.benchmarks.as2.rte.As2ReadItemParamGen.TOTAL_READ_COUNT=10

```

```
#
# Basic Parameters
#

# The running time for warming up before benchmarking
org.vanilladb.bench.VanillaBenchParameters.WARM_UP_INTERVAL=60000
# The running time for benchmarking
org.vanilladb.bench.VanillaBenchParameters.BENCHMARK_INTERVAL=60000
# The number of remote terminal executors for benchmarking
org.vanilladb.bench.VanillaBenchParameters.NUM_RTES=2
# The sleeping time (in milliseconds) between transactions for each RTE
# 0 = no sleeping, 100 is a generally good number for under-loaded workloads
org.vanilladb.bench.VanillaBenchParameters.RTE_SLEEP_TIME=0
# The IP of the target database server
org.vanilladb.bench.VanillaBenchParameters.SERVER_IP=127.0.0.1
# 1 = JDBC, 2 = Stored Procedures
org.vanilladb.bench.VanillaBenchParameters.CONNECTION_MODE=2
# 1 = AS2
org.vanilladb.bench.VanillaBenchParameters.BENCH_TYPE=1
# Whether it enables the built-in profiler on the server
org.vanilladb.bench.VanillaBenchParameters.PROFILING_ON_SERVER=false
# The path to the generated reports
org.vanilladb.bench.VanillaBenchParameters.REPORT_OUTPUT_DIRECTORY=
# The granularity for summarizing the performance of benchmarking
org.vanilladb.bench.VanillaBenchParameters.REPORT_TIMELINE_GRANULARITY=1000
# Whether the RTEs display the results of each transaction
org.vanilladb.bench.VanillaBenchParameters.SHOW_TXN_RESPONSE_ON_CONSOLE=false

# The number of items in the testing data set
org.vanilladb.bench.benchmarks.as2.As2BenchConstants.NUM_ITEMS=100000
# Read count
org.vanilladb.bench.benchmarks.as2.rte.As2ReadItemParamGen.TOTAL_READ_COUNT=10

```

Use JDBC or stored procedures

JDBC / SP ?



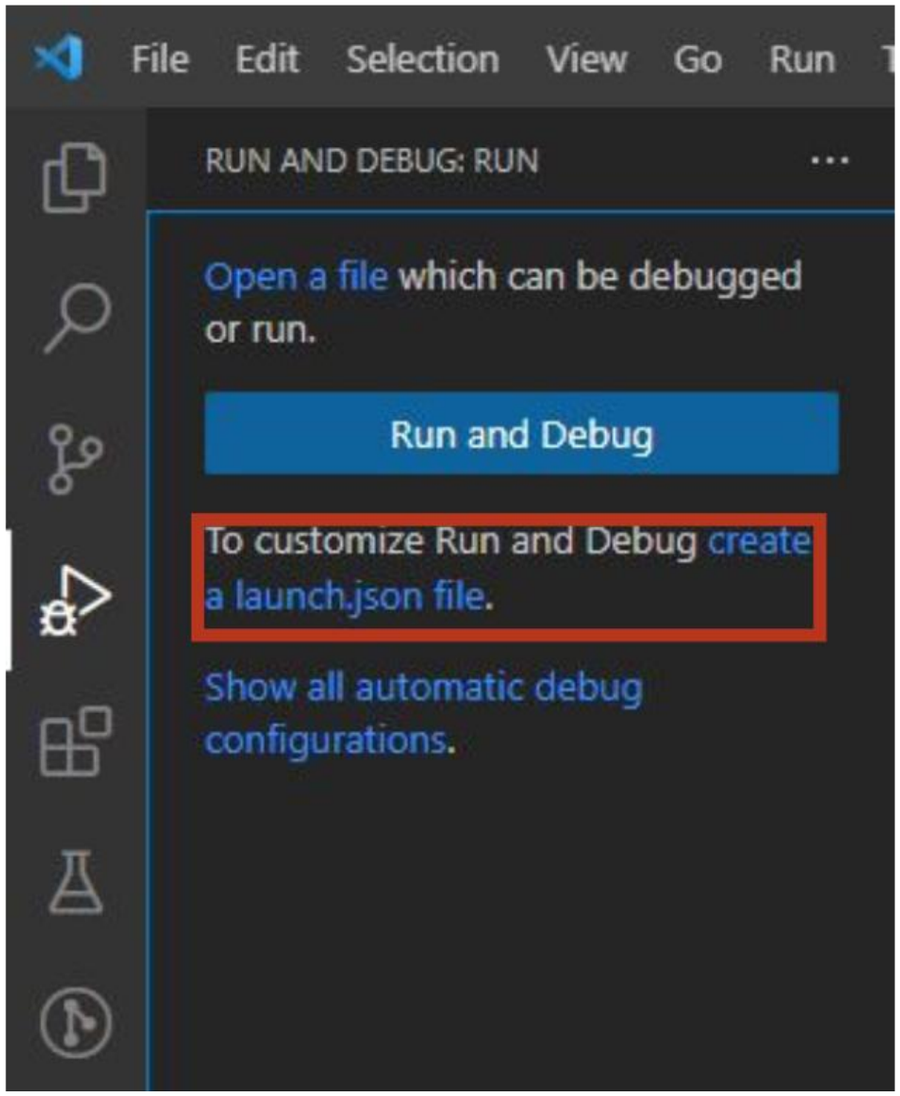
Create SP

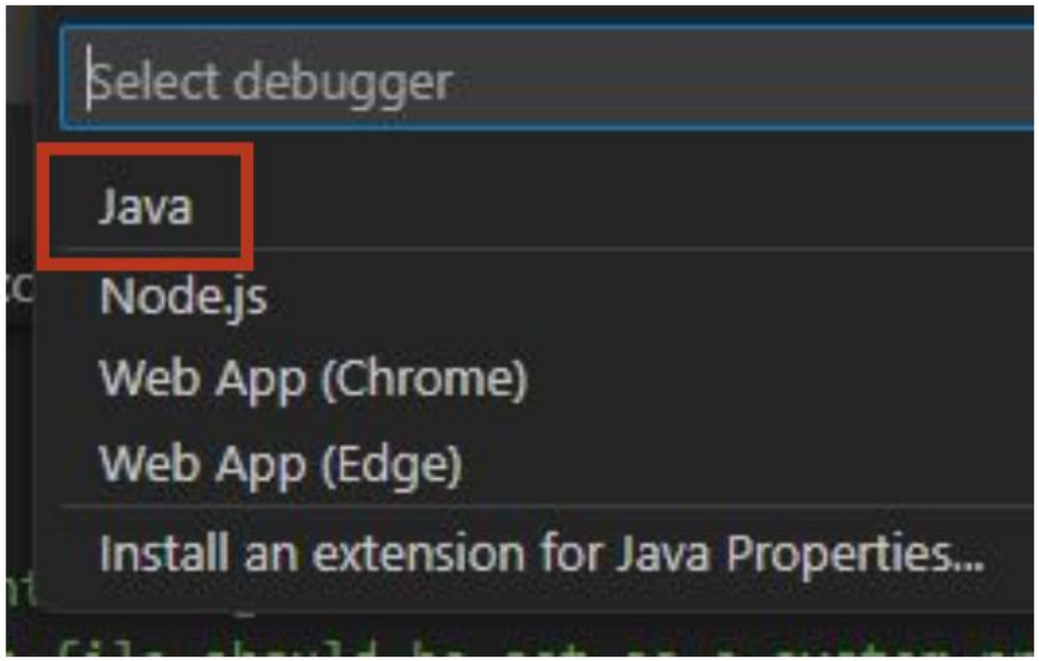
```
-- Insert user
CREATE PROCEDURE insertuser(uname VARCHAR(50), ukarma INT)
LANGUAGE SQL
AS $$
        INSERT INTO users(name, karma) VALUES (uname, ukarma);
$$;

-- Insert post
CREATE PROCEDURE insertpost(uname VARCHAR(50), post TEXT)
LANGUAGE SQL
AS $$
        INSERT INTO posts(text, "authorId")
        VALUES (post, (SELECT id FROM users WHERE name = uname));
$$;
```

```
#
# Basic Parameters
#

# The running time for warming up before benchmarking
org.vanilladb.bench.VanillaBenchParameters.WARM_UP_INTERVAL=60000
# The running time for benchmarking
org.vanilladb.bench.VanillaBenchParameters.BENCHMARK_INTERVAL=60000
# The number of remote terminal executors for benchmarking
org.vanilladb.bench.VanillaBenchParameters.NUM_RTES=2
# The sleeping time (in milliseconds) between transactions for each RTE
# 0 = no sleeping, 100 is a generally good number for under-loaded workloads
org.vanilladb.bench.VanillaBenchParameters.RTE_SLEEP_TIME=0
# The IP of the target database server
org.vanilladb.bench.VanillaBenchParameters.SERVER_IP=127.0.0.1
# 1 = JDBC, 2 = Stored Procedures
org.vanilladb.bench.VanillaBenchParameters.CONNECTION_MODE=2
# 1 = AS2
org.vanilladb.bench.VanillaBenchParameters.BENCH_TYPE=1
# Whether it enables the built-in profiler on the server
org.vanilladb.bench.VanillaBenchParameters.PROFILING_ON_SERVER=false
# The path to the generated reports
org.vanilladb.bench.VanillaBenchParameters.REPORT_OUTPUT_DIRECTORY=
# The granularity for summarizing the performance of benchmarking
org.vanilladb.bench.VanillaBenchParameters.REPORT_TIMELINE_GRANULARITY=1000
# Whether the RTEs display the results of each transaction
org.vanilladb.bench.VanillaBenchParameters.SHOW_TXN_RESPONSE_ON_CONSOLE=false

# The number of items in the testing data set
org.vanilladb.bench.benchmarks.as2.As2BenchConstants.NUM_ITEMS=100000
# Read count
org.vanilladb.bench.benchmarks.as2.rte.As2ReadItemParamGen.TOTAL_READ_COUNT=10

```

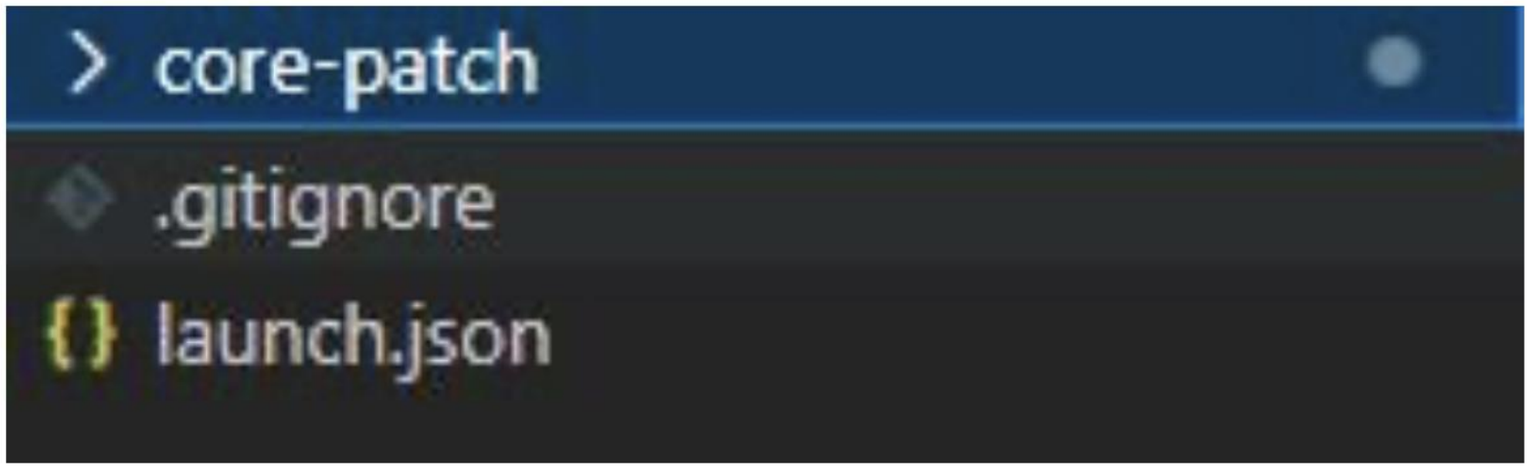
Benchmark report path

Setting Run Configuration(1/2)

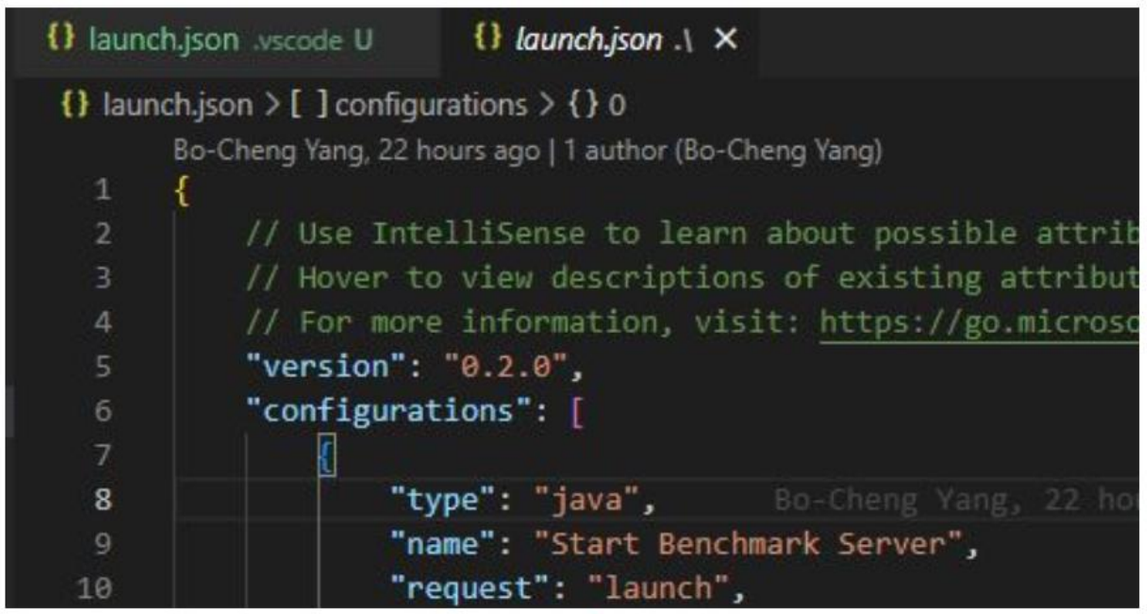




Setting Run Configuration(2/2)



open our launch.json

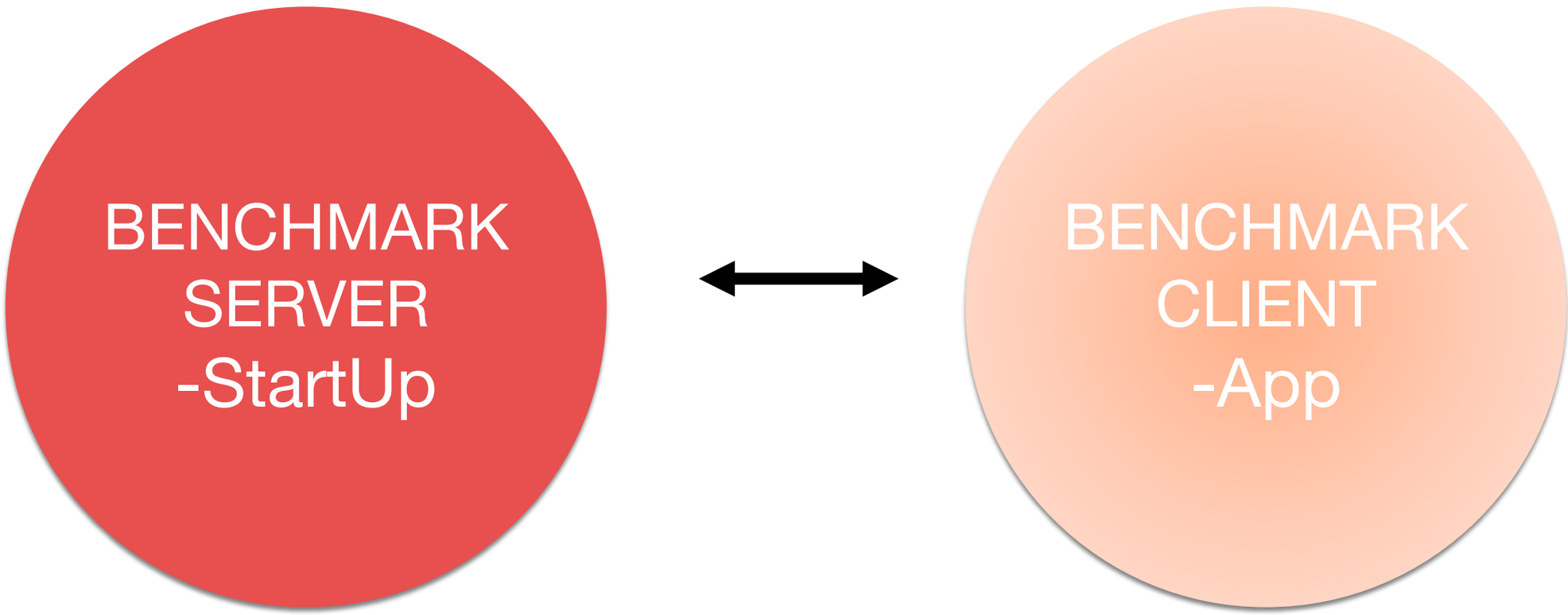


copy our content to your launch.json

Outline

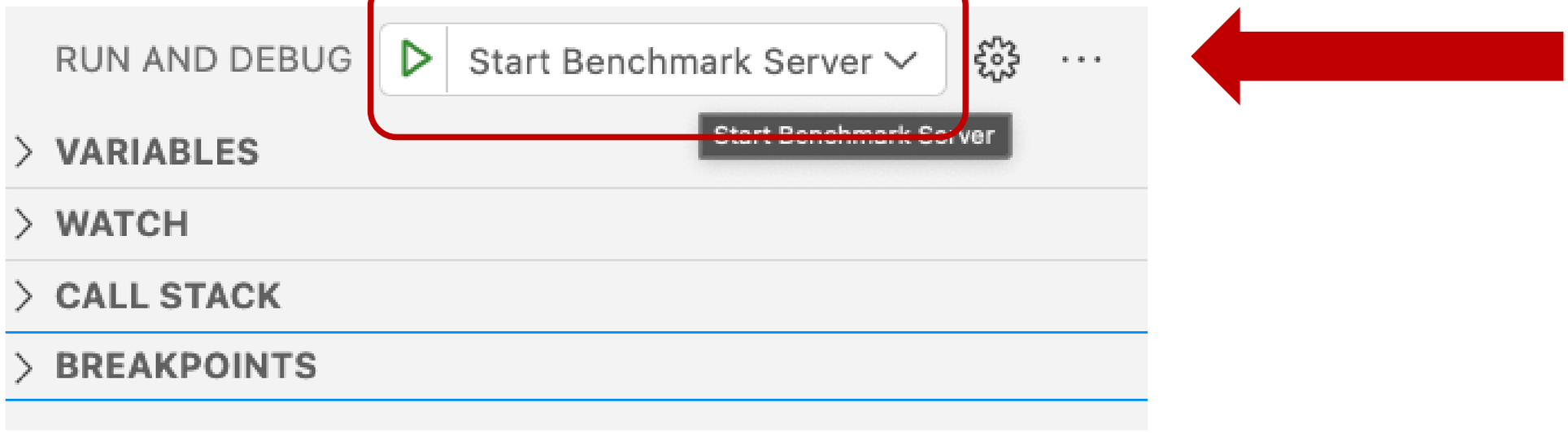
- VanillaBench Project
  - Introduction to VanillaBench
  - Setting Benchmark Configurations
- Starting Up Server for Benchmarking
- Running Benchmark Client
- Assignment 2

Two Main Methods

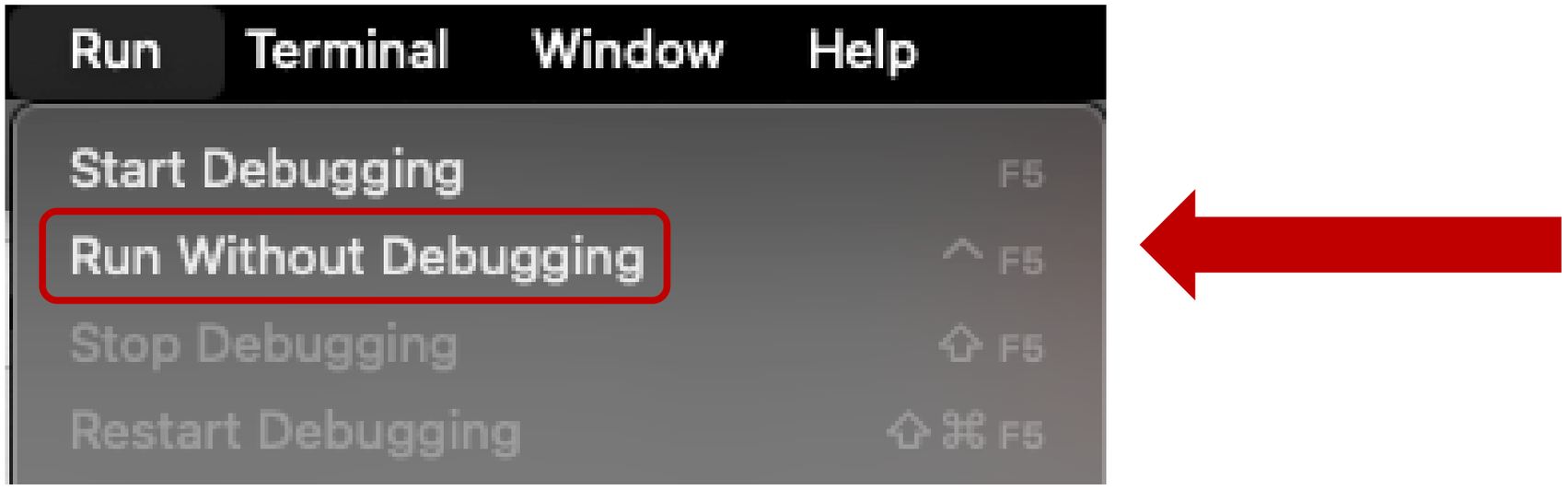


Starting Up Server

1. Select “Start Benchmark Server”



2. Click “Run without Debugging”



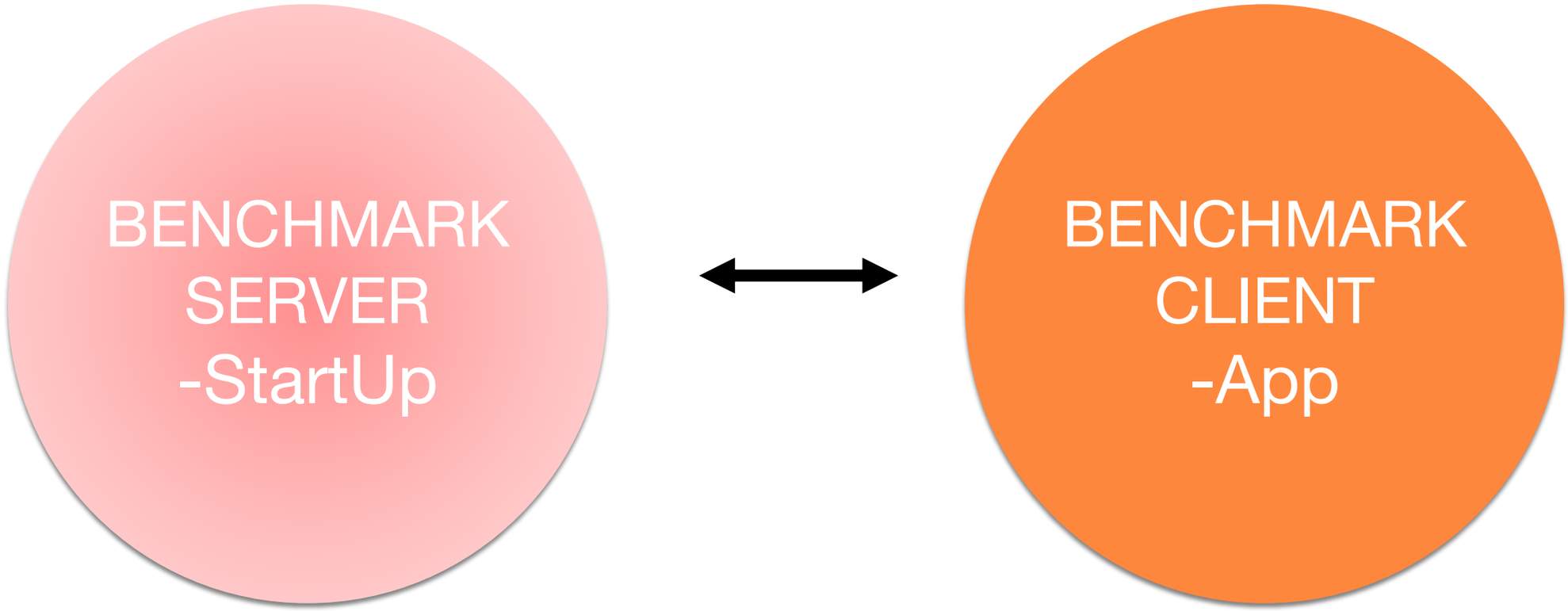
Server Messages

You should see similar messages if nothing is wrong.

Outline

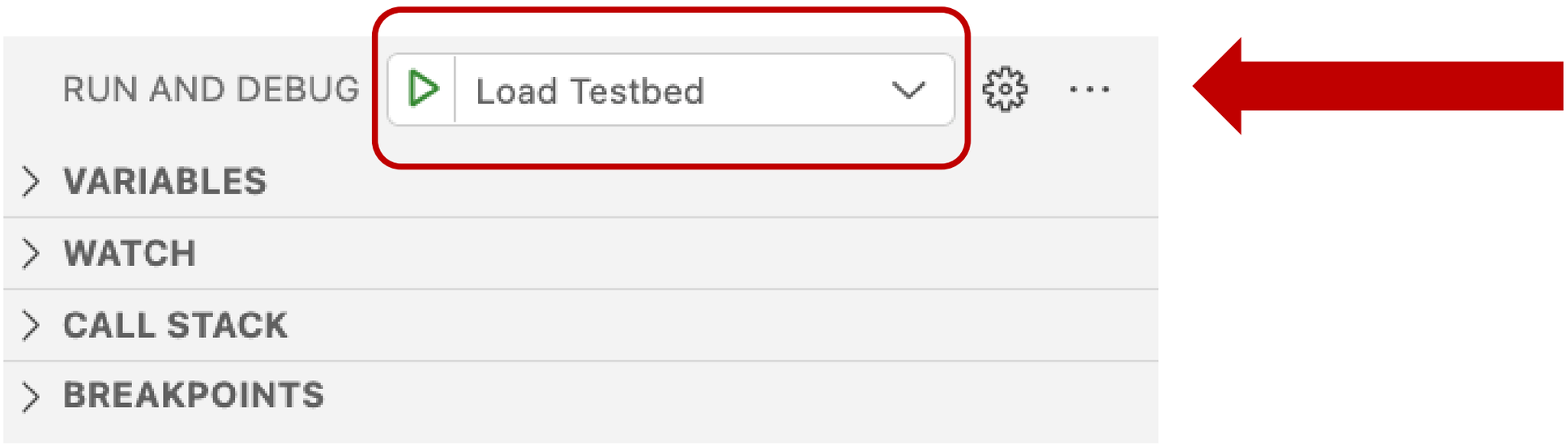
- VanillaBench Project
  - Introduction to VanillaBench
  - Setting Benchmark Configurations
  - Starting Up Server for Benchmarking
  - Running Benchmark Client
  - Assignment 2

Two Main Methods

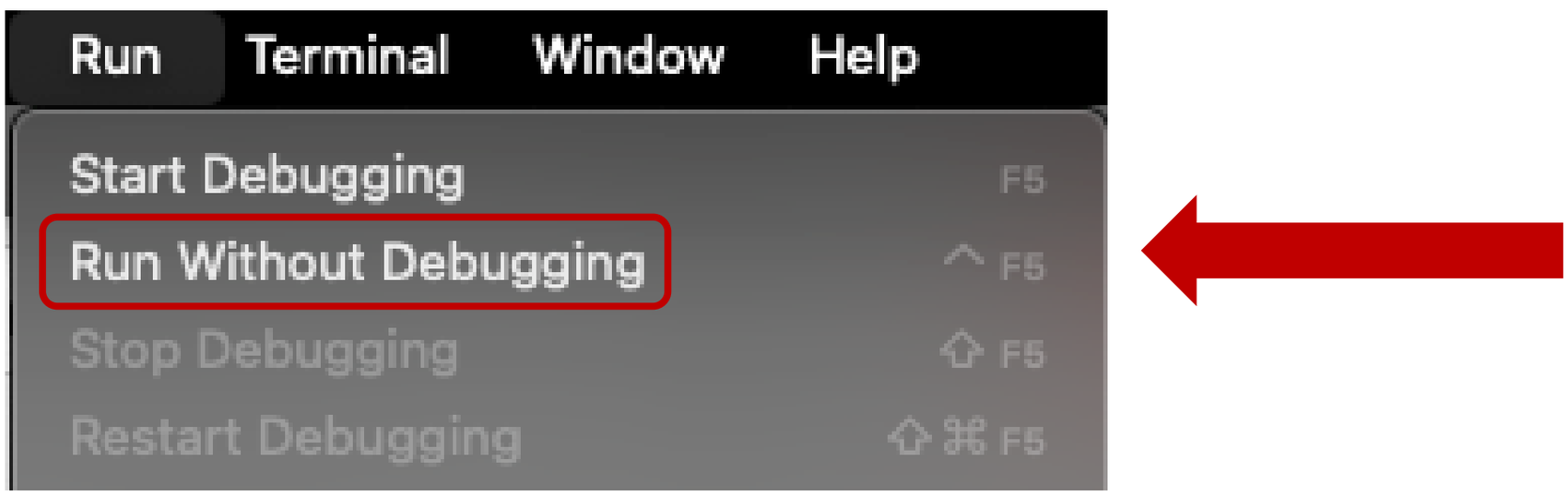


Loading Testbed

1. Select “Load Testbed”



2. Click “Run without Debugging”



Client Messages

PROBLEMS 11 PORTS OUTPUT DEBUG CONSOLE TERMINAL

```
3月 13, 2024 2:25:25 上午 org.vanilladb.bench.VanillaBenchParameters <clinit>
資訊: Using AS2 benchmarks
3月 13, 2024 2:25:25 上午 org.vanilladb.bench.VanillaBench loadTestbed
資訊: loading the testbed of the benchmark...
3月 13, 2024 2:25:25 上午 org.vanilladb.bench.VanillaBench loadTestbed
資訊: loading procedure finished.
```

The Workflow of A Client

App

Create

org.vanilladb.bench.VanillaBench

Depends on the Argument

loadTestbed()

Load Testbed

benchmark()

Start Benchmarking

The Workflow of A Client

App

Create

org.vanilladb.bench.VanillaBench

Depends on the Argument

loadTestbed()

Load Testbed

benchmark()

Start Benchmarking

Loading Testbed

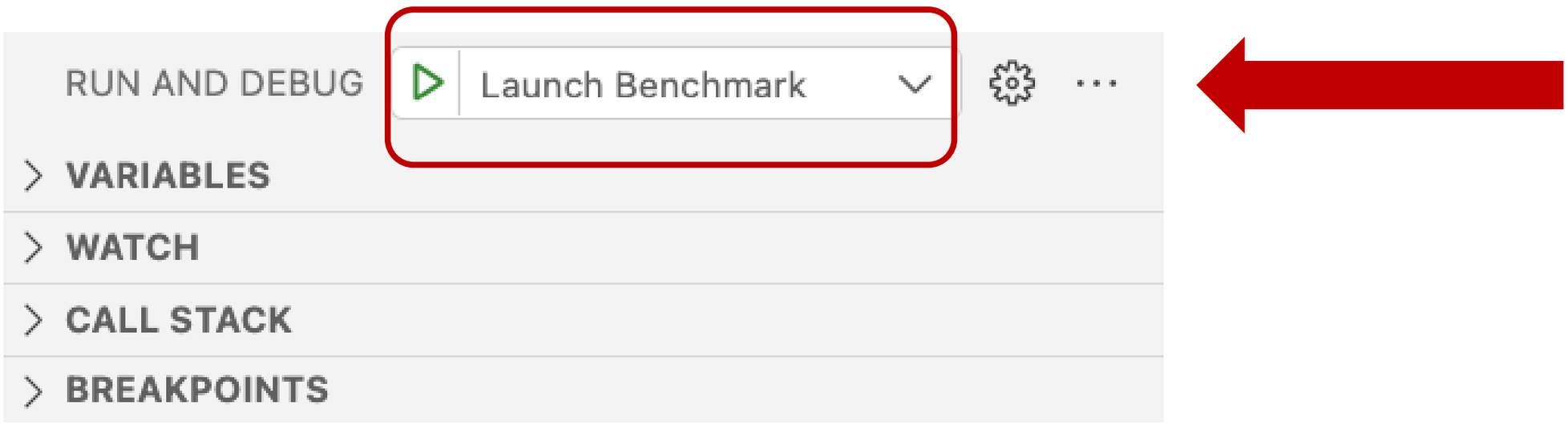
loadTestbed()

Connect to server and execute:

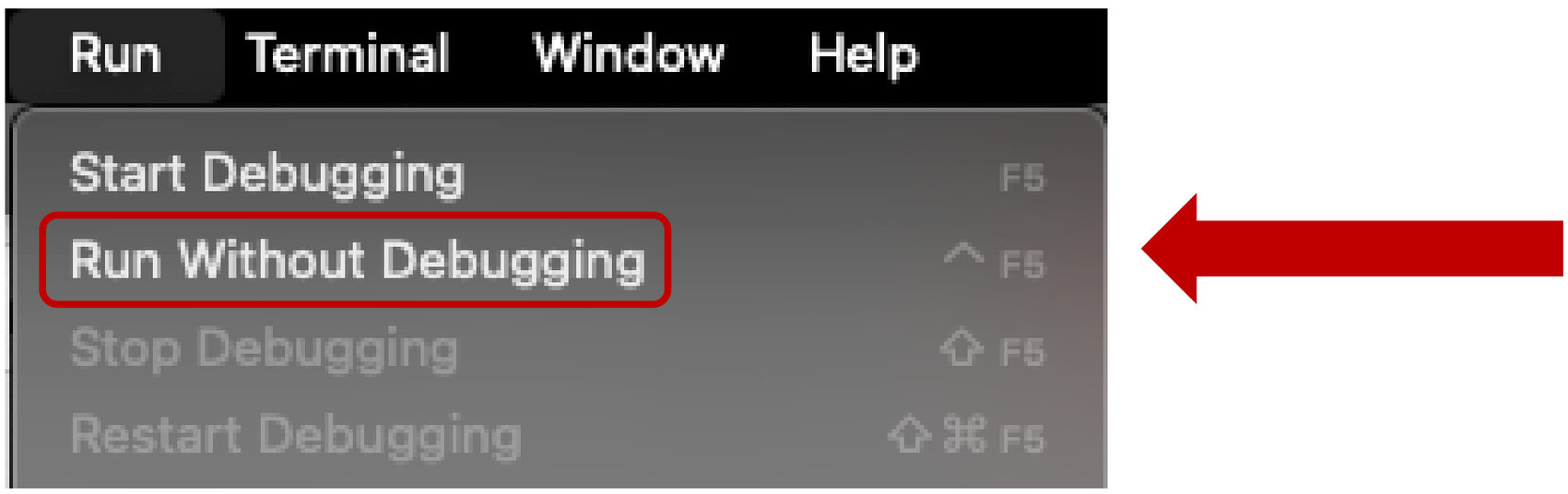
TestbedLoader

Benchmarking

1. Select “Launch Benchmark”



2. Click “Run without Debugging”

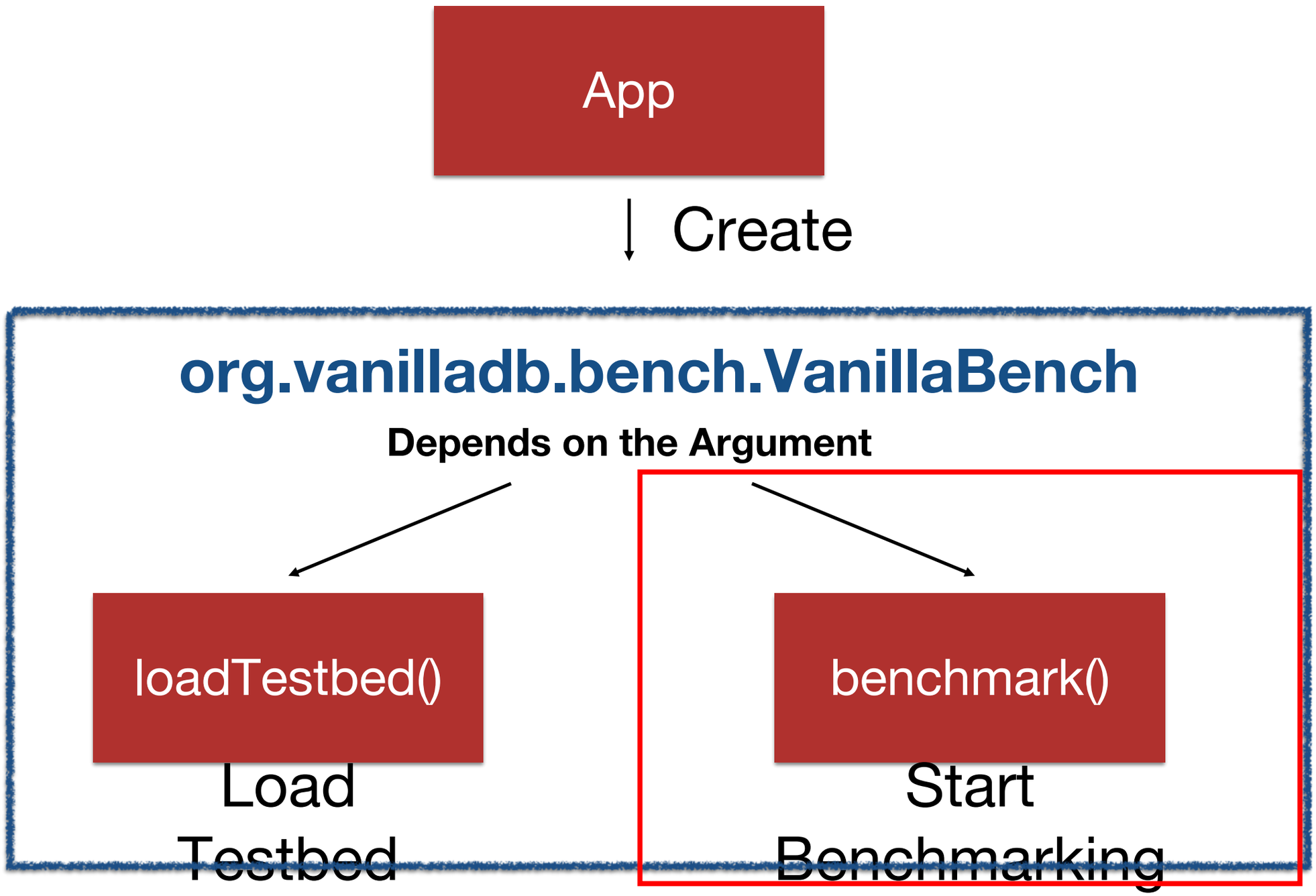


Client Messages

PROBLEMS 11 PORTS OUTPUT DEBUG CONSOLE TERMINAL

```
3月 13, 2024 3:40:21 上午 org.vanilladb.bench.VanillaBenchParameters <clinit>
資訊: Using AS2 benchmarks
3月 13, 2024 3:40:21 上午 org.vanilladb.bench.VanillaBench benchmark
資訊: checking the database on the server...
3月 13, 2024 3:40:21 上午 org.vanilladb.bench.VanillaBench benchmark
資訊: database check passed.
3月 13, 2024 3:40:21 上午 org.vanilladb.bench.VanillaBench benchmark
資訊: creating 2 emulators...
3月 13, 2024 3:40:21 上午 org.vanilladb.bench.VanillaBench benchmark
資訊: waiting for connections...
3月 13, 2024 3:40:23 上午 org.vanilladb.bench.VanillaBench benchmark
資訊: start benchmarking.
3月 13, 2024 3:41:23 上午 org.vanilladb.bench.VanillaBench benchmark
資訊: warm up period finished.
3月 13, 2024 3:41:23 上午 org.vanilladb.bench.VanillaBench benchmark
資訊: start recording results...
3月 13, 2024 3:42:23 上午 org.vanilladb.bench.VanillaBench benchmark
資訊: benchmark period finished. Stoping RTEs...
3月 13, 2024 3:42:23 上午 org.vanilladb.bench.StatisticMgr outputReport
資訊: Finish creating benchmark report.
3月 13, 2024 3:42:23 上午 org.vanilladb.bench.VanillaBench benchmark
資訊: benchmark process finished.
```

The Workflow of A Client



Starting Benchmark

benchmark()

Create

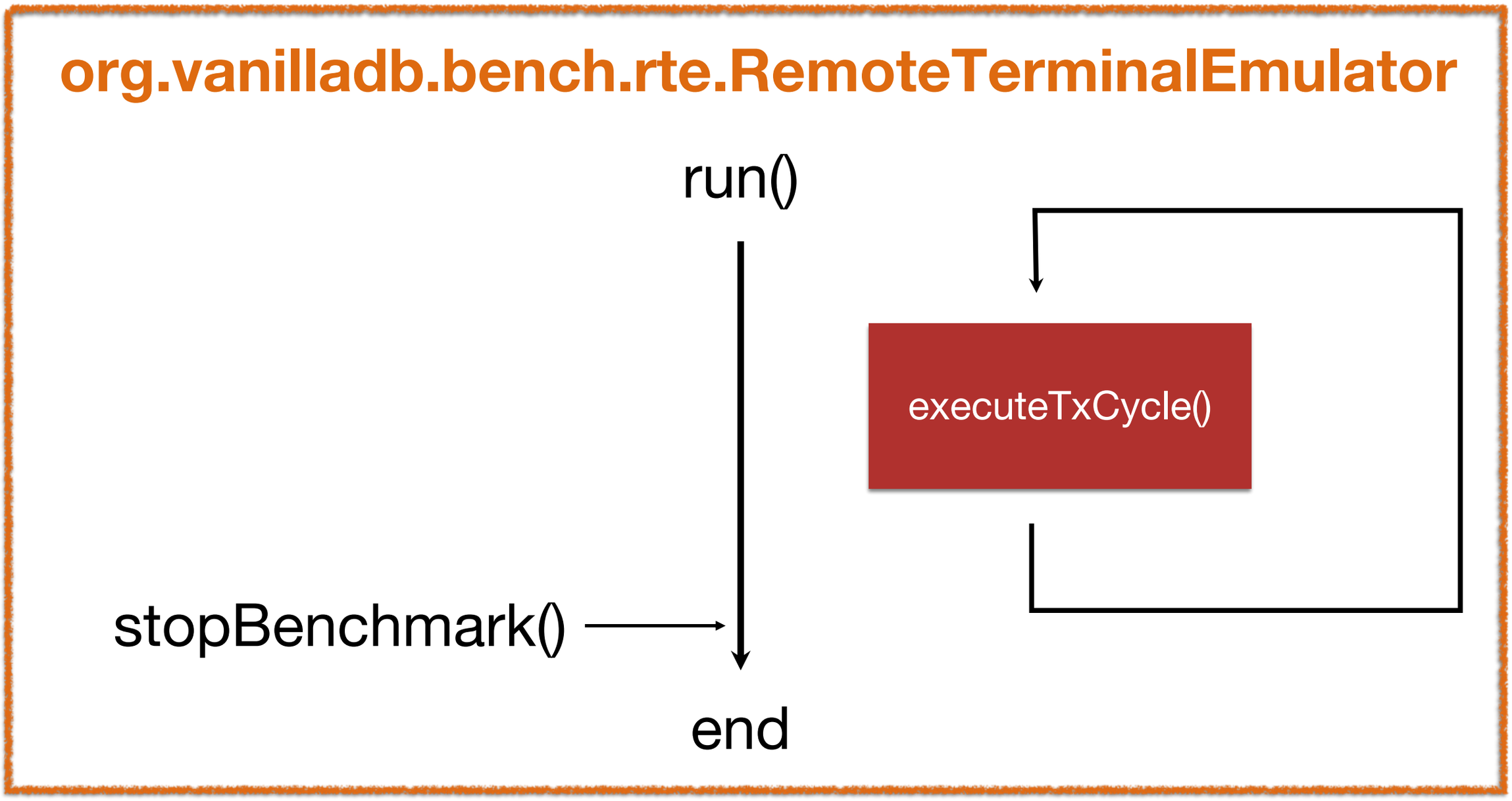
RTEs

Remote **T**erminal
Emulator

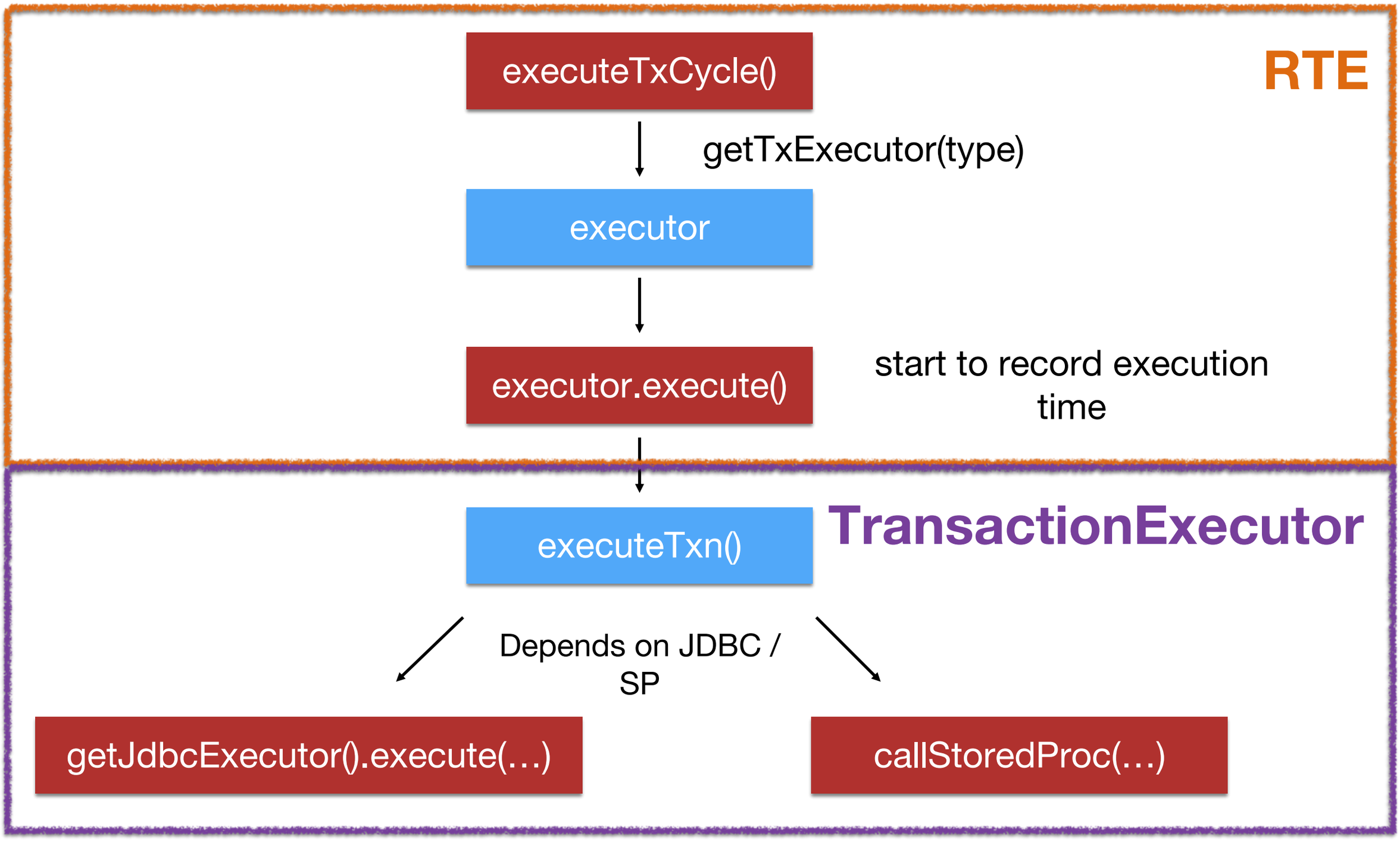
Emulates a remote terminal,
executing a sequence of transactions

Server & Client

RTE's Life Cycle



Executing a Tx



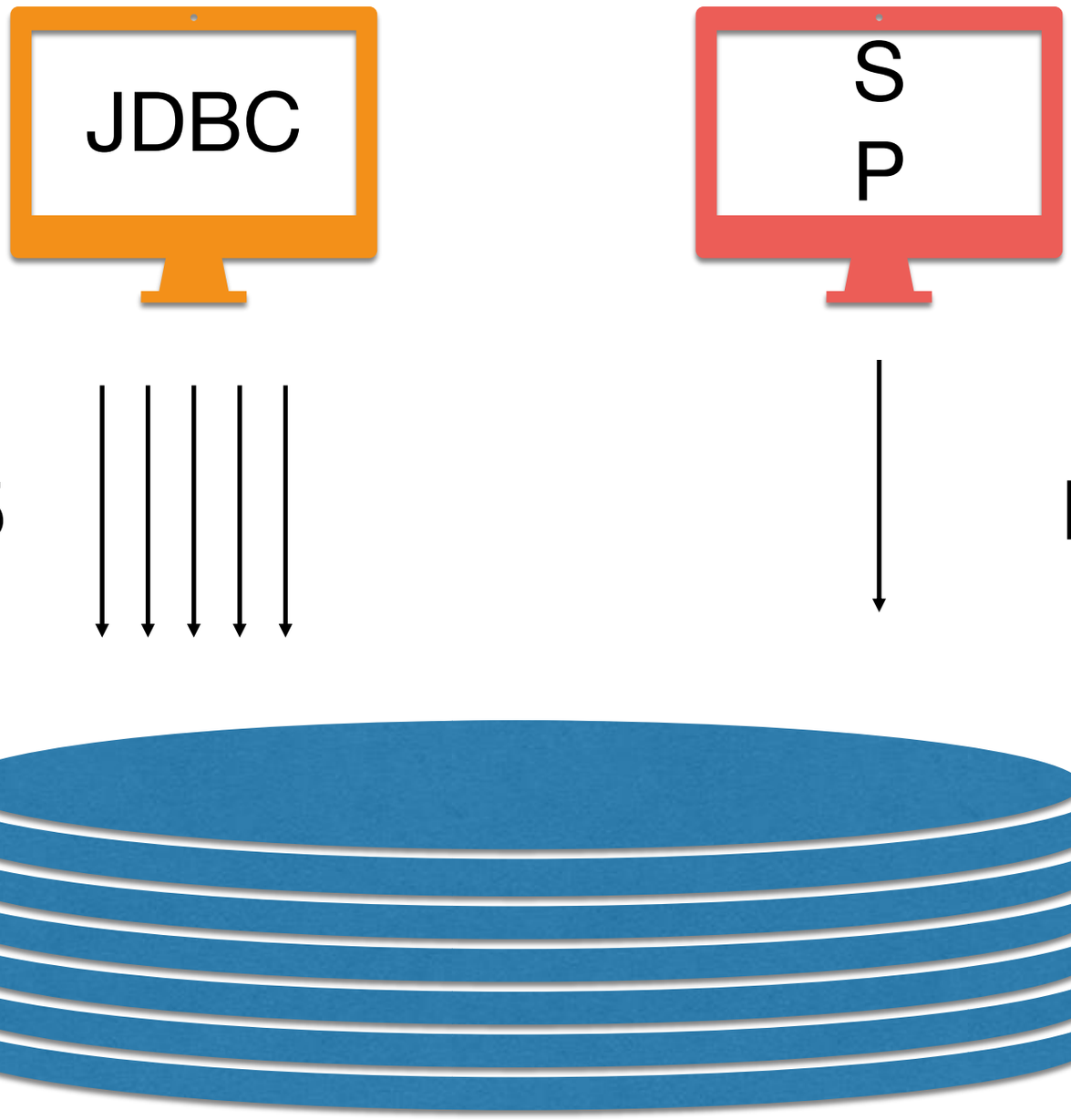
Workflow of Executing a Tx

- General steps
  - Generate parameters from TxParamGenerator
- JDBC
  - **getJdbcExecutor().execute(…)** executes a JDBC Job in **local.**
  - Job will execute each sql via JDBC connector.
- Store Procedure
  - **callStoredProc(…)** executes a stored procedure on the **remote server.**
  - Remote server will return a **SutResultSet** when the procedure is finished.

How Server Process a StoredProc call ?

- When server receives a remote procedure call, it will ask StoredProcFactory to generate the appropriate StoredProcedure
- The server will then call the StoredProcedure methods:
  - prepare(Obj…)
    - Prepares the parameters.
  - execute()
    - Executes the transaction.
    - This method will return the final result to client.

JDBC / SP ?



JDBC / SP ?

TestbedLoaderJdbcJob

CheckDatabaseJdbcJob

ReadItemTxnJdbcJob

JDBC

SP

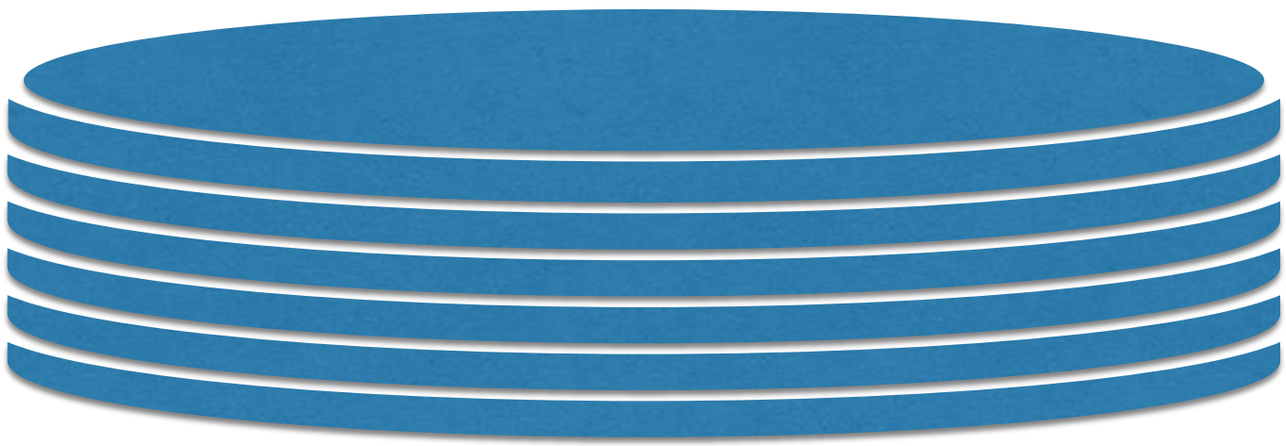
TestbedLoaderProc

CheckDatabaseProc

ReadItemTxnProc

Query

Parameter



Assignment 2

- https://shwu10.cs.nthu.edu.tw/courses/databases/2025-spring/db25-assignment-2

Q&A

- If you have any problem, you could check here first
  - https://shwu10.cs.nthu.edu.tw/courses/databases/2025-spring/faq
- If your problem is very unique, feel free to send us an email