Computer System Design & Application

计算机系统设计与应用A

陶伊达 （TAO Yida）
taoyd@sustech.edu.cn

Lecture 15

- Project Demo
- Course Review
- Grading Policy & Final Exam

Project Demo

- 何林翰、秦恺通
- 吴宇贤、汤嘉阳

Topics covered

Applications

- Data analytics and visualization
- C/S multithreaded applications
- Text scraping and processing
- Web applications & REST services

Principles

- OOP, AOP
- Functional programming
- Design principles
- JVM

Utilities

- Generic collections
- Lambdas & Stream
- Exception handling
- Files & I/O
- Annotations
- Reflection
- JUnit Testing
- Logging

Features

- GUI & JavaFX
- Networking
- Multithreading
- Web development
- Web services

Topics covered

Applications

- Data analytics and visualization
- C/S multithreaded applications
- Text scraping and processing
- Web applications & REST services

Principles

- OOP, AOP
- Functional programming
- Design principles
- JVM

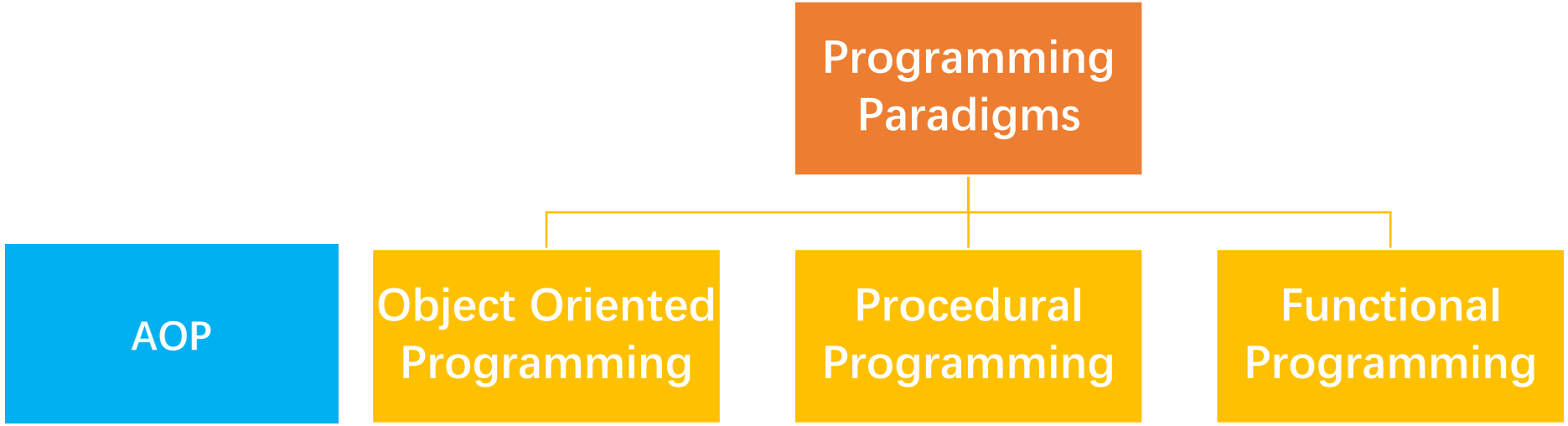
Utilities

- Generic collections
- Lambdas & Stream
- Exception handling
- Files & I/O
- Annotations
- Reflection
- JUnit Testing
- Logging

Features

- GUI & JavaFX
- Networking
- Multithreading
- Web development
- Web services

Programming Paradigms



- Business logics
- Cross-cutting concerns

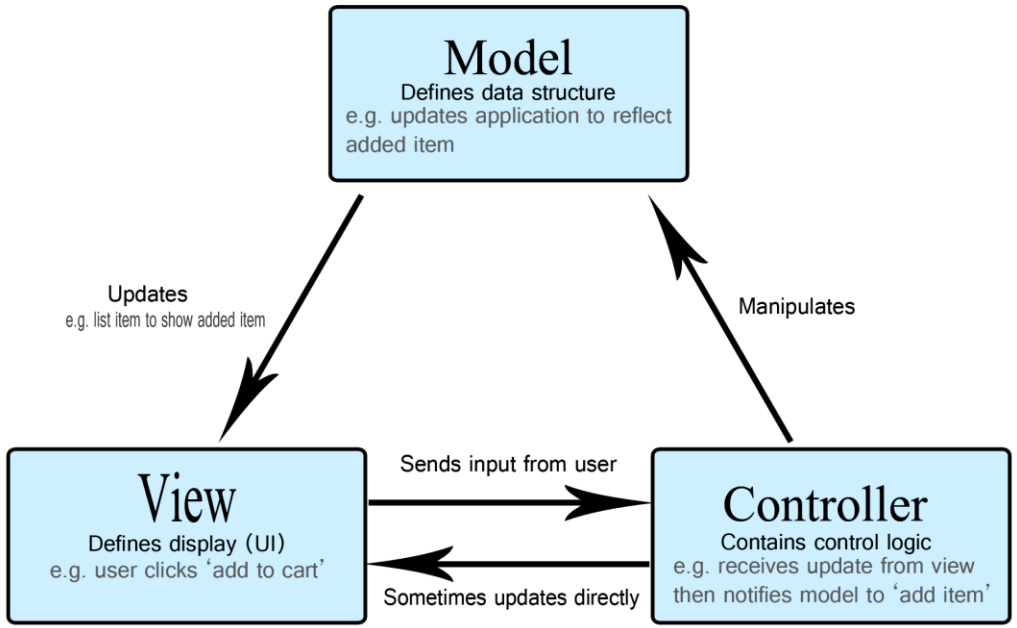
- Encapsulation
- Abstraction
- Inheritance
- Polymorphism

- First-class functions
- Higher-order functions
- No side effects

Design Principles

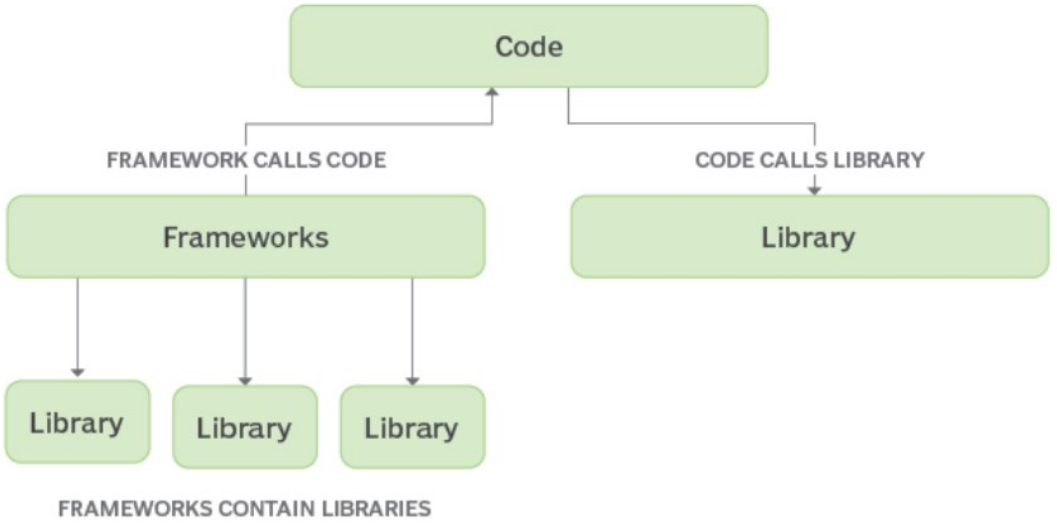
Software Design Principles

- High Cohesion (高内聚)
- Low Coupling (低耦合)
- Information Hiding (信息隐藏)

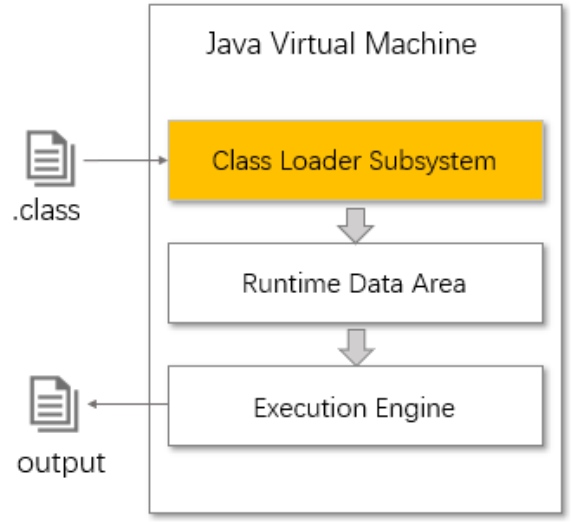


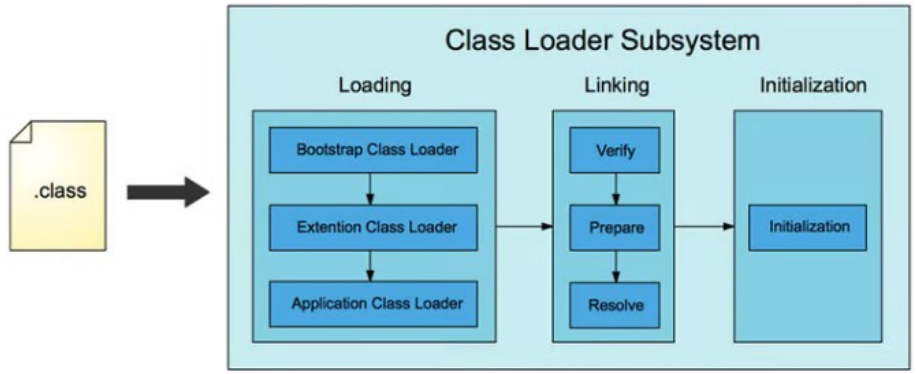
Inversion of Control

- Inversion of Control (IoC, 控制反转): a principle in SE which transfers the control of objects or portions of a program to a container or framework

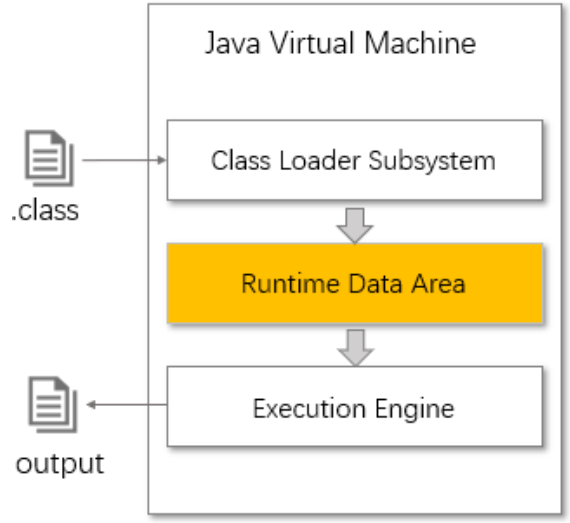


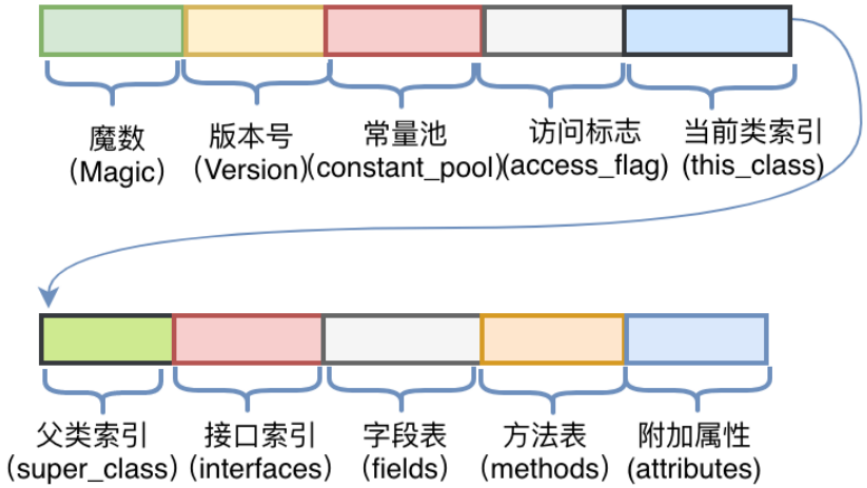
JVM Architecture



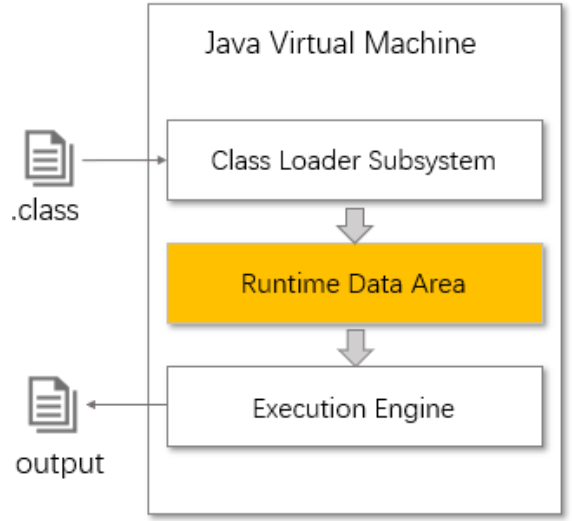


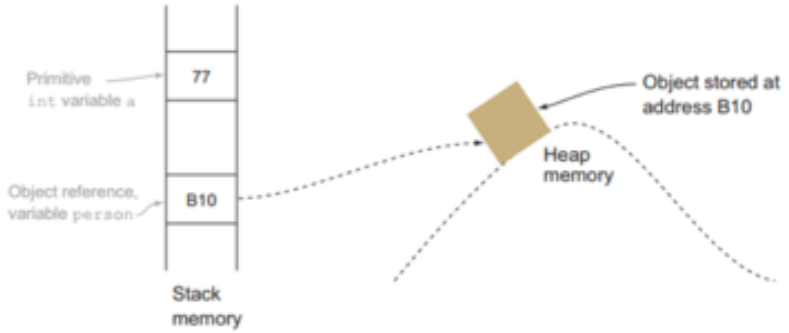
JVM Architecture



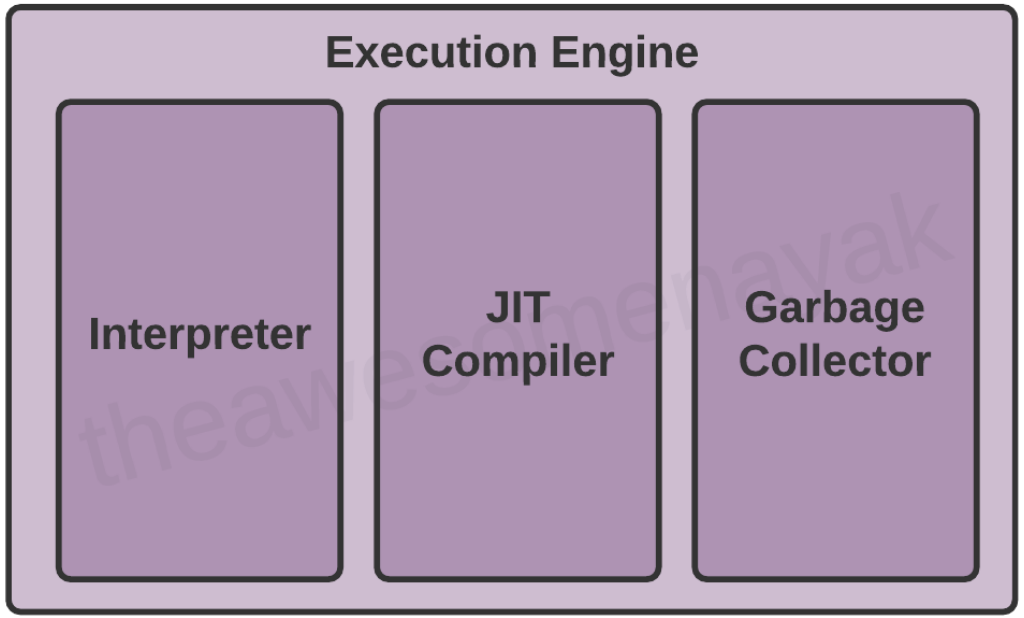


JVM Architecture





Execution Engine



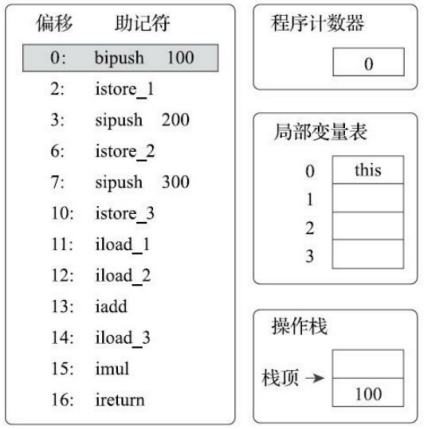
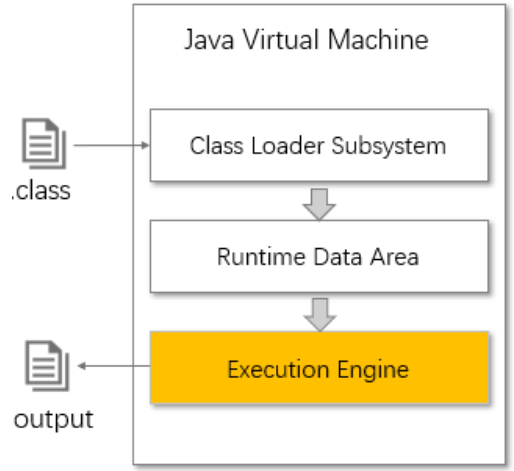
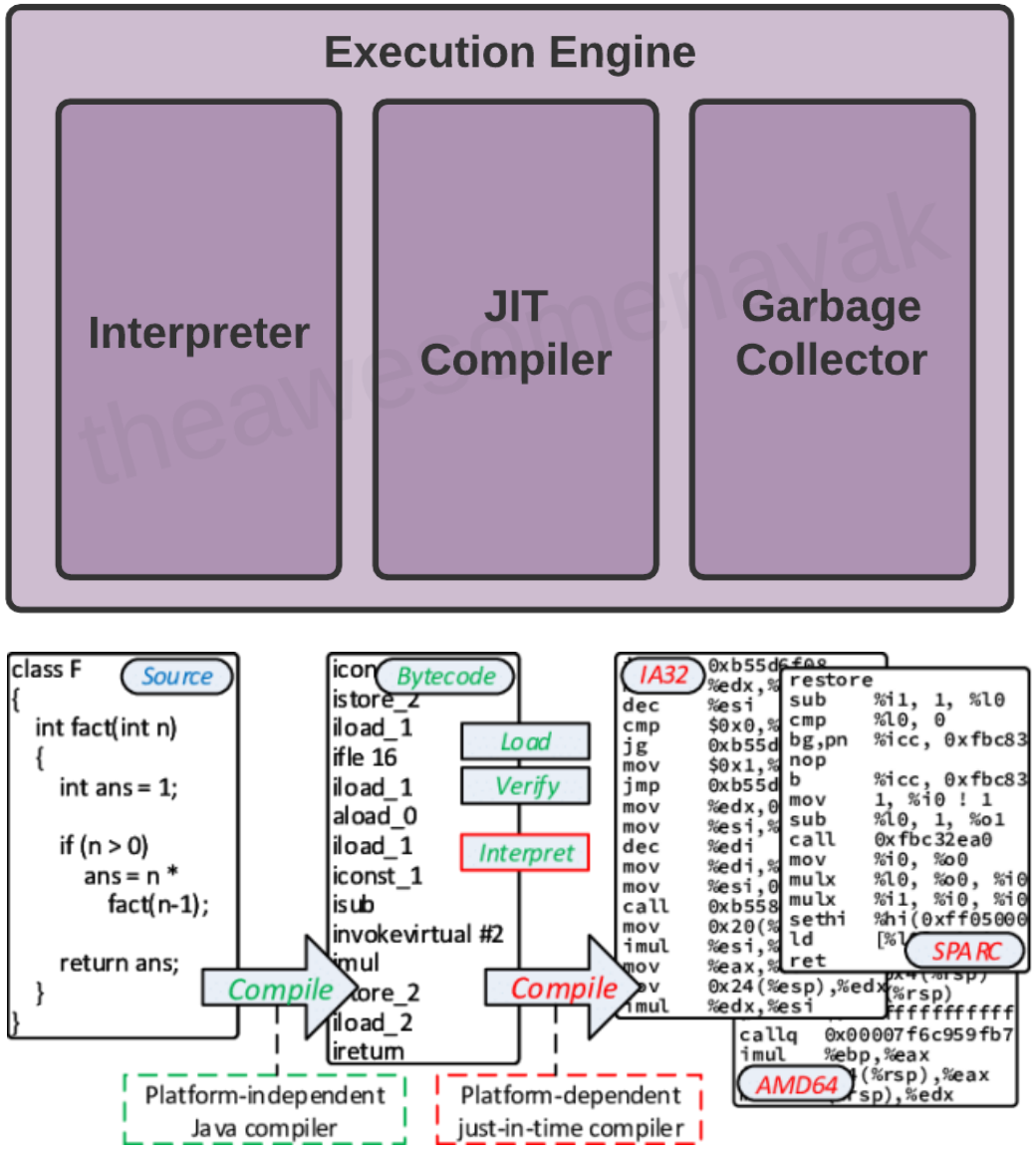


图8-5 执行偏移地址为0的指令的情况

首先，执行偏移地址为0的指令，Bipush指令的作用是将单字节的整型常量值（-128～127）推入操作数栈顶，跟随有一个参数，指明推送的常量值，这里是100。



Execution Engine



Topics covered

Applications

- Data analytics and visualization
- C/S multithreaded applications
- Text scraping and processing
- Web applications & REST services

Principles

- OOP, AOP
- Functional programming
- Design principles
- JVM

Utilities

- Generic collections
- Lambdas & Stream
- Exception handling
- Files & I/O
- Annotations
- Reflection
- JUnit Testing
- Logging

Features

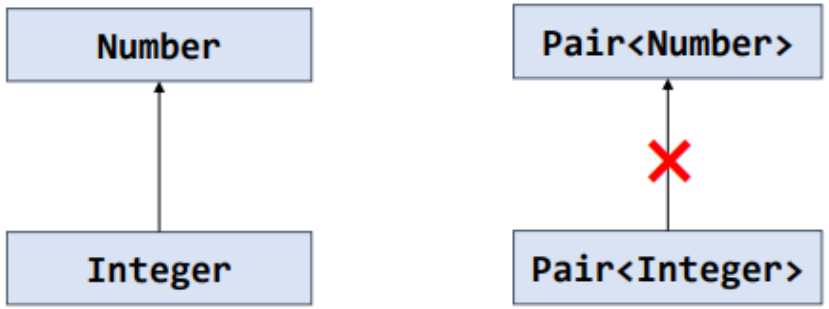
- GUI & JavaFX
- Networking
- Multithreading
- Web development
- Web services

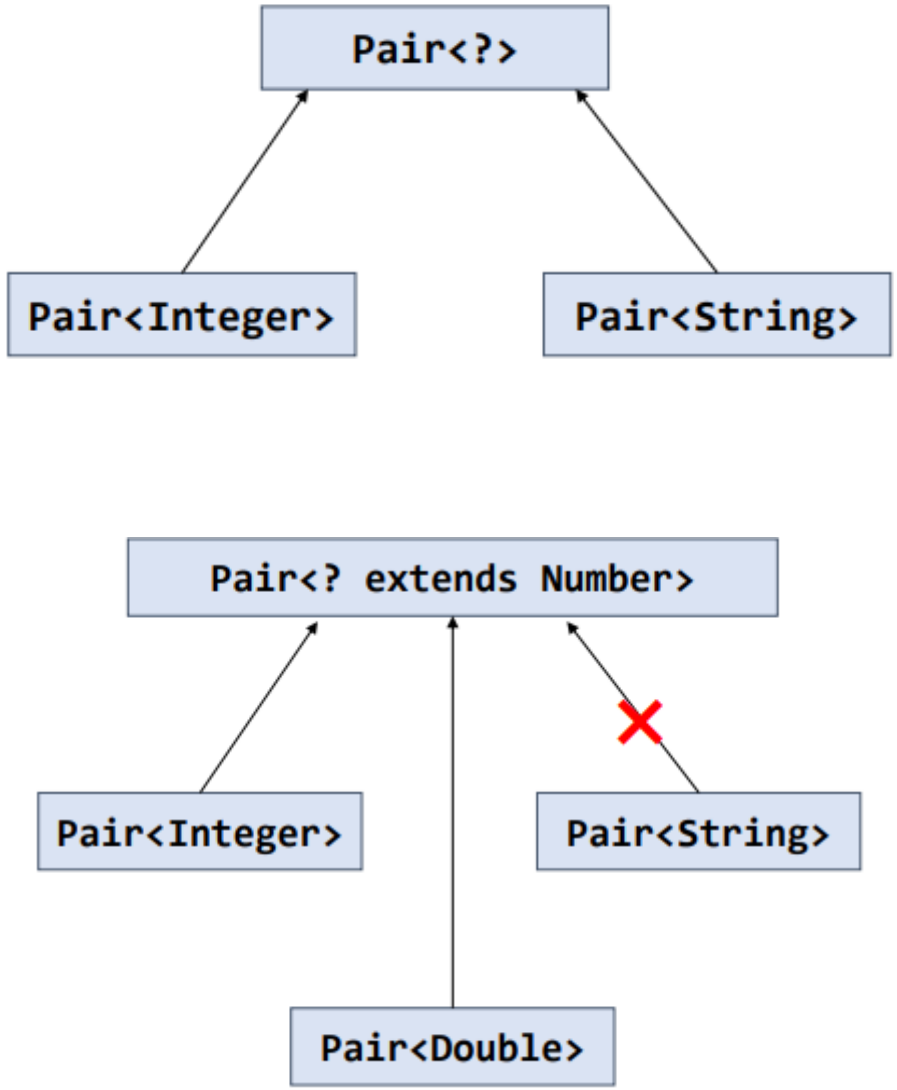
Generics

- Motivation & Benefits
- Syntax & Usages
  - Generic Classes
  - Generic Interfaces
  - Generic Methods
- Type erasure

Generics

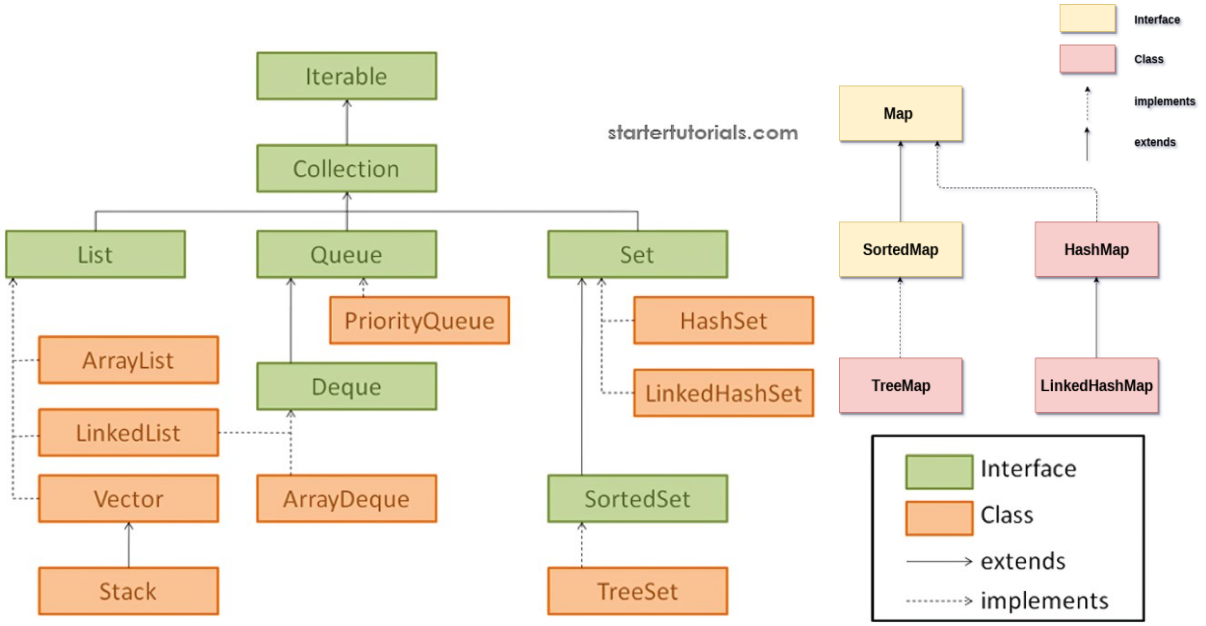
- Inheritance Rules
- Bounded Type Variables
- Wildcards





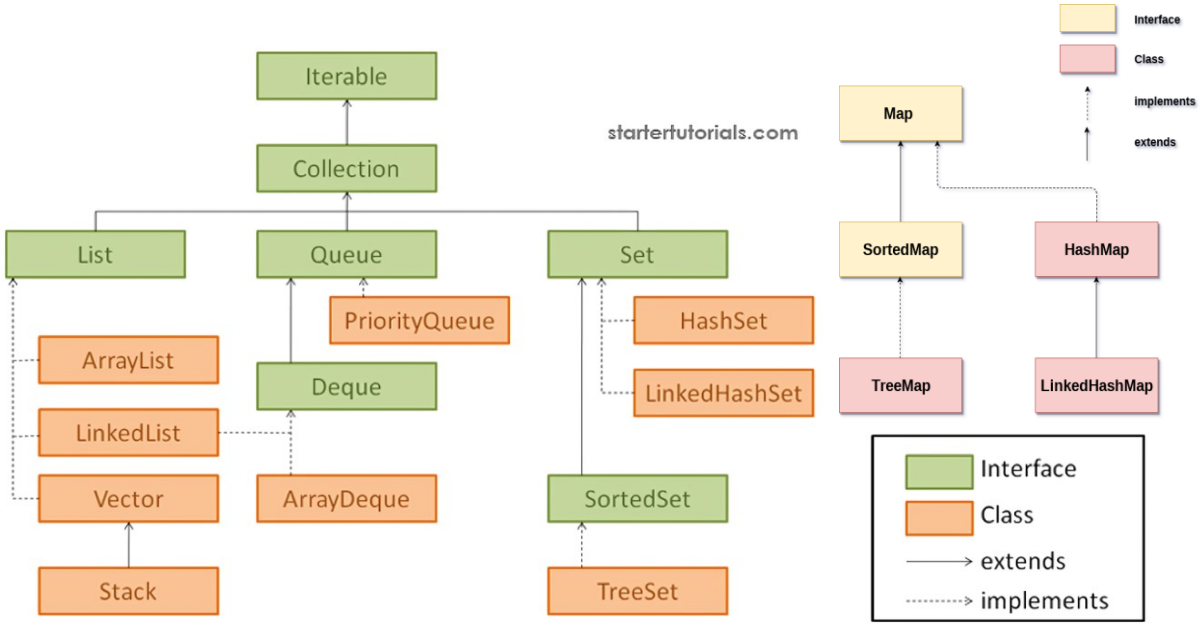
Collections

- Class hierarchy
- The Iterable<T> and Iterator<T> interfaces



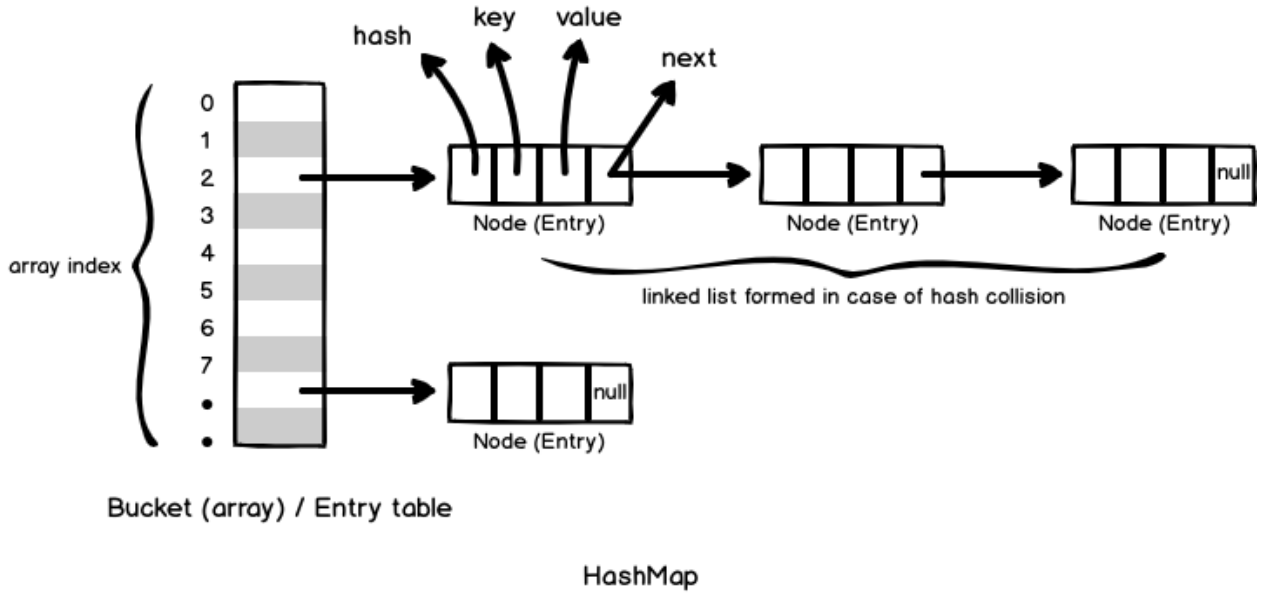
Collections

- Commonly used collection implementations & characteristics
- Comparisons between different implementations



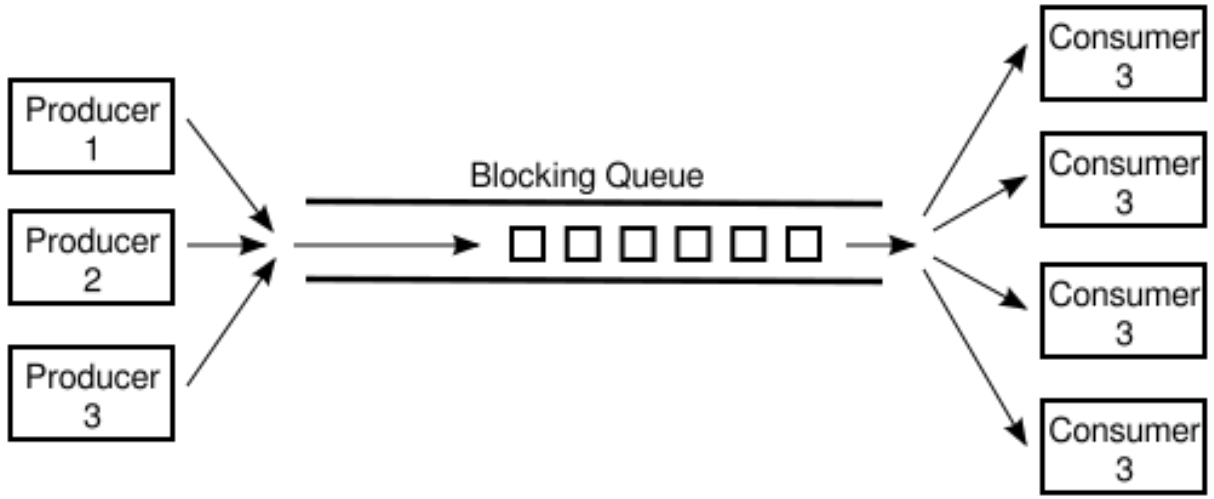
Collections

- `hashCode()` returns an integer value. By default, it converts the internal address of the object into an integer
- `equals()` checks if objects are equal. By default, `Object.equals(Object obj) { return (this == obj);}`
- If two objects are equal according to the `equals(Object)` method, then calling the `hashCode` method on each of the two objects must produce the same integer result (if you override `equals`, you must override `hashCode`.).



Collections

- Non thread-safe collections
- Thread-safe collections
  - Copy-on-Write collections
  - Compare-and-Swap collections (CAS)
  - Collections using Lock

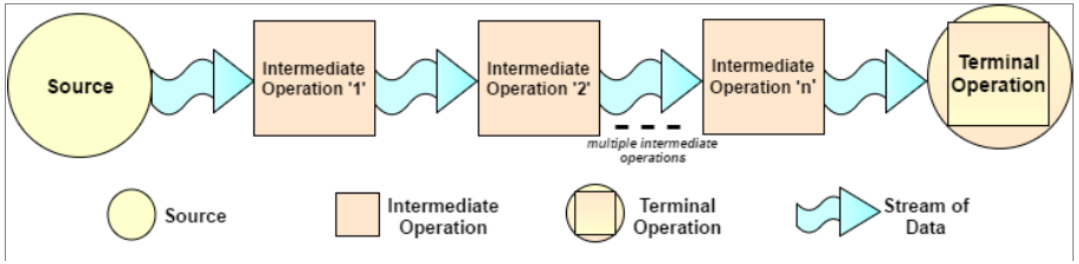


Lambda Expressions

- Lambda syntax
- Type inference
- Method references
  - Static method
  - Instance method (Bound)
  - Instance method (Unbound)
  - Constructor
- Common functional interfaces

Stream API

- How to create a stream
- Common intermediate operations
- Common terminal operations



Stream API

- Collecting results
- Write a stream pipeline/chain
- The Optional<T> class

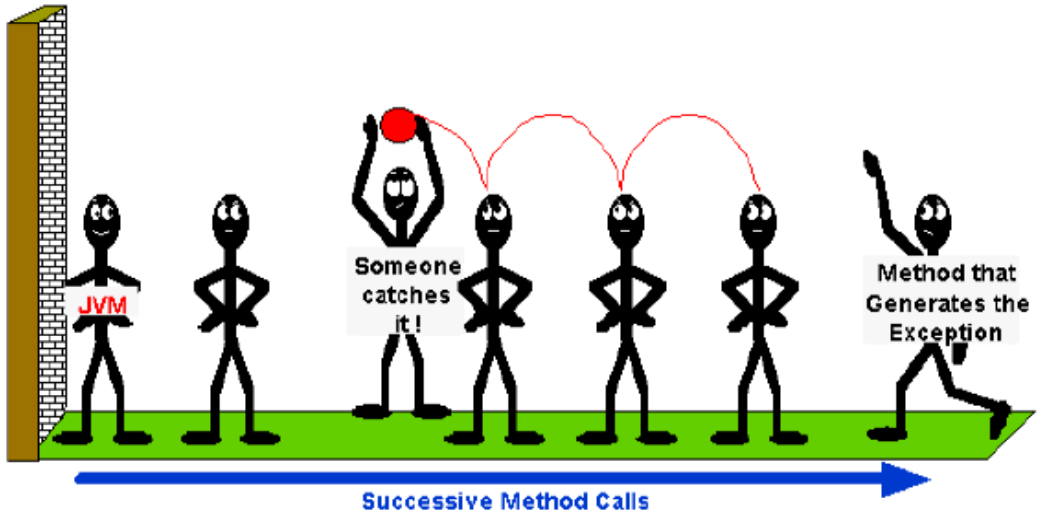
```
Stream<String> stream = Stream.of("1a", "1bb", "1c", "2a", "2a", "2bb");
Map<Character, Set<String>> group =
                    stream.collect(Collectors.groupingBy(s->s.charAt(0),
                    Collectors.mapping(s->s.substring(1), Collectors.toSet())));
```

Exception Handling

- Exception class hierarchy
- Checked vs unchecked exceptions

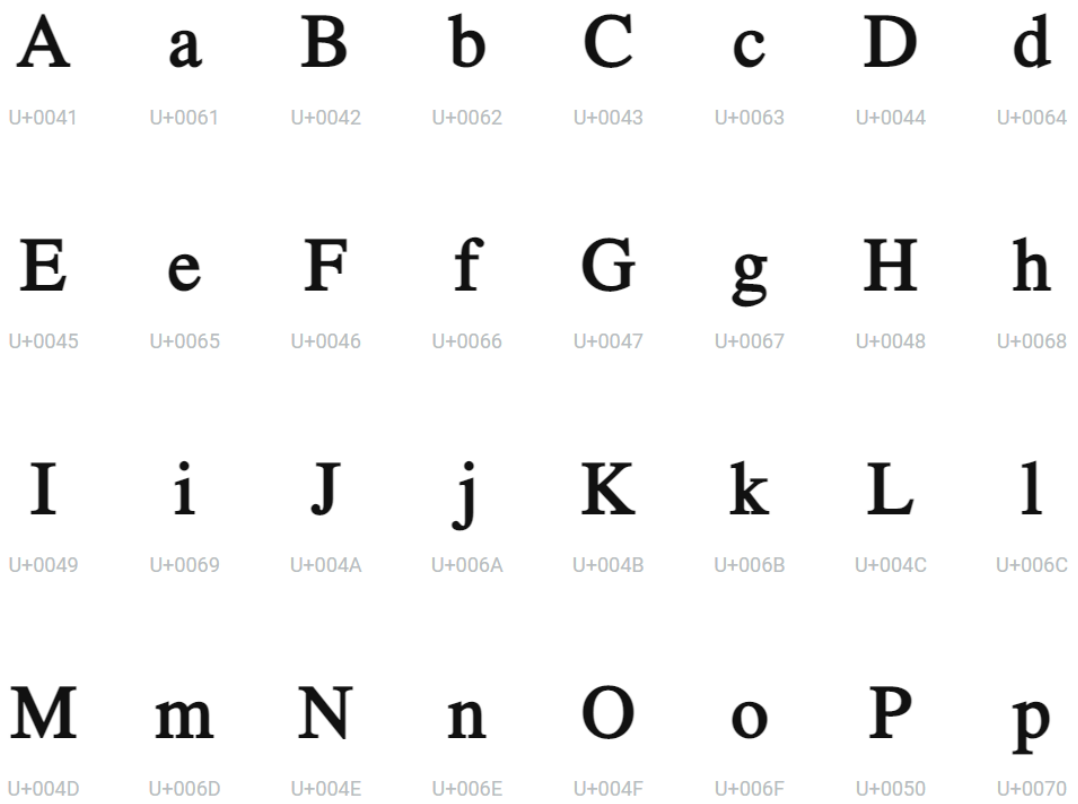
Exception Handling

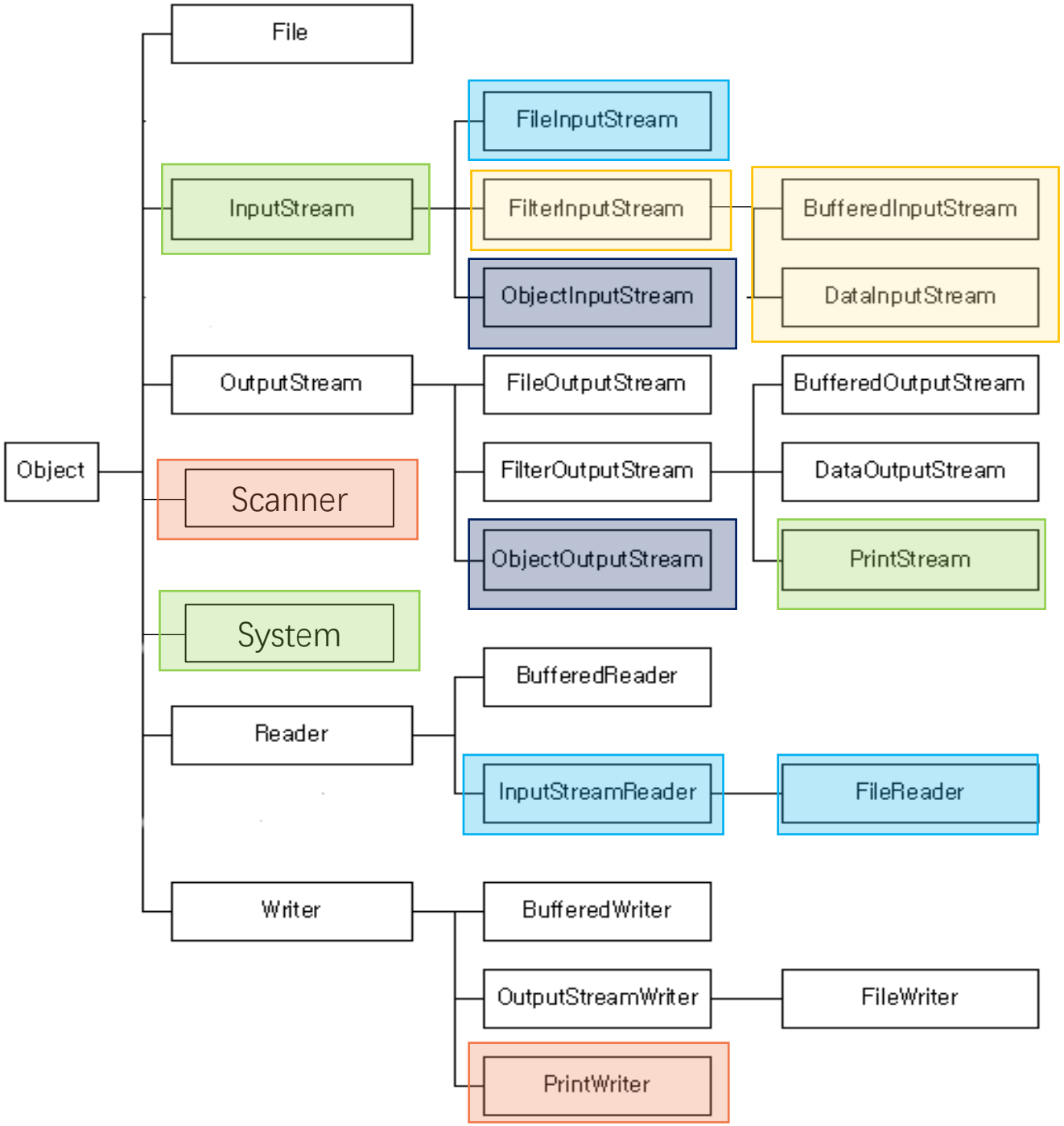
- Syntax & Control flow
- Try-catch, finally, throw



Encoding

- Basic knowledge of character encoding
- Unicode & Java char
- Common encoding schemes (e.g., UTF-8)



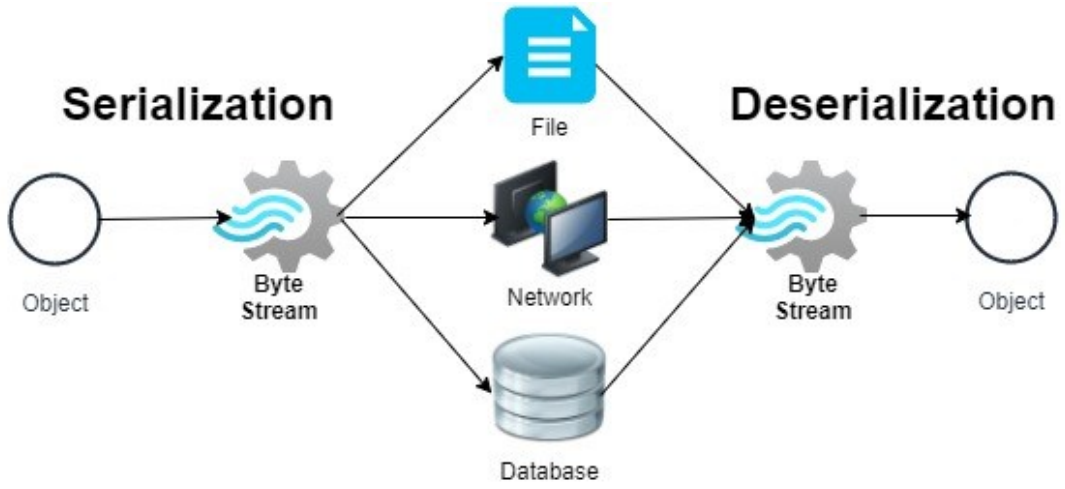


I/O

- Class hierarchy
- Byte streams vs. character streams & conversions
- Commonly used classes and methods
- I/O from command line

Serialization

- Concept
- Implementation
- Customized serialization & deserialziation

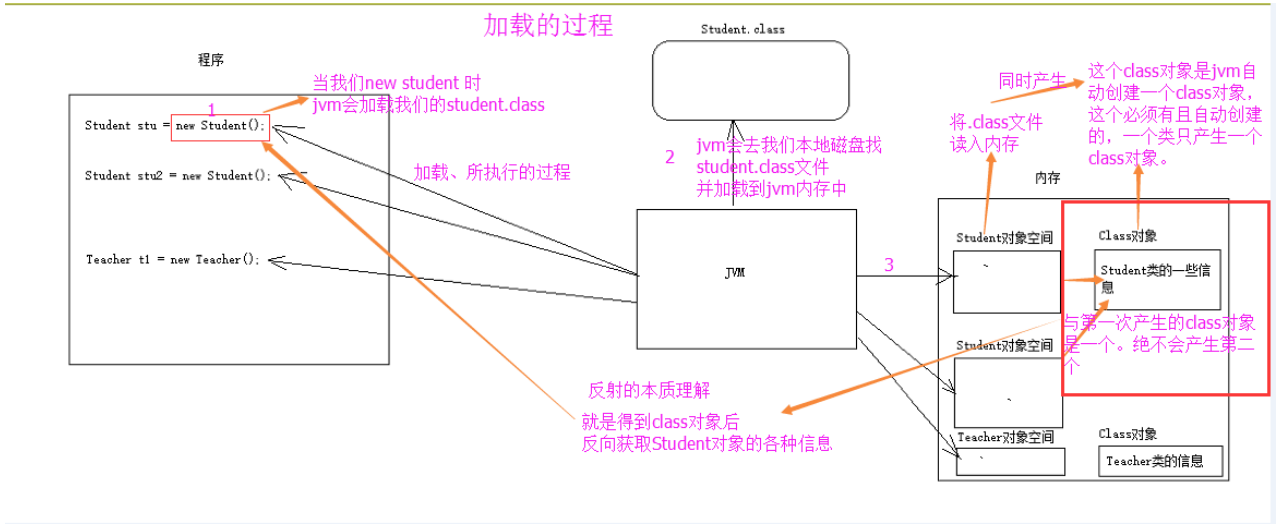


Files

- Absolute path & relative path
- Dot notations
- Basic operations for files and directories (e.g., traverse a directory recursively)

Reflection

- JVM class loading process
- Getting the `Class` object
- Examining fields and methods of a class
- Instantiating a class
- Invoking a method of an object
- Use cases

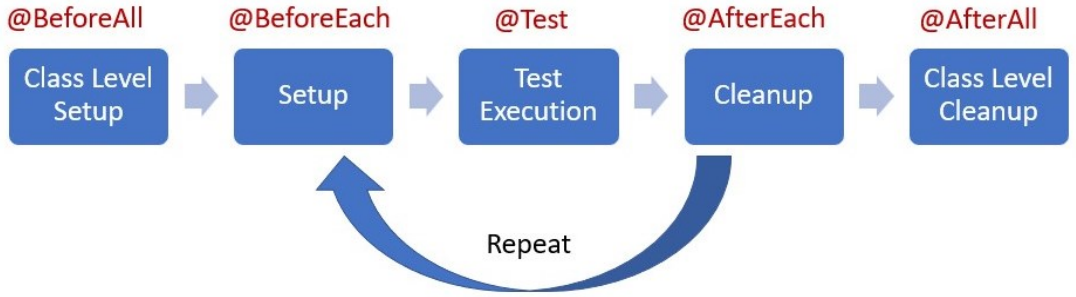


Annotations

- Built-in annotations
  - `@Deprecated`
  - `@Override`
  - `@SuppressWarnings`
  - `@FunctionalInterface`
  - `@SafeVarargs`
- Meta-annotations
  - `@Target`
  - `@Retention`
  - `@Documented`
- Customized annotations

JUnit Testing

- Test classes and test methods
- Lifecycle methods
- Test instance lifecycle
- Assertions & assumptions



Logging

- Requirements for logging
- Design of Apache Log4j
- Different log levels and meanings
- SLF4J and different logging impl

Topics covered

Applications

- Data analytics and visualization
- C/S multithreaded applications
- Text scraping and processing
- Web applications & REST services

Principles

- OOP, AOP
- Functional programming
- Design principles
- JVM

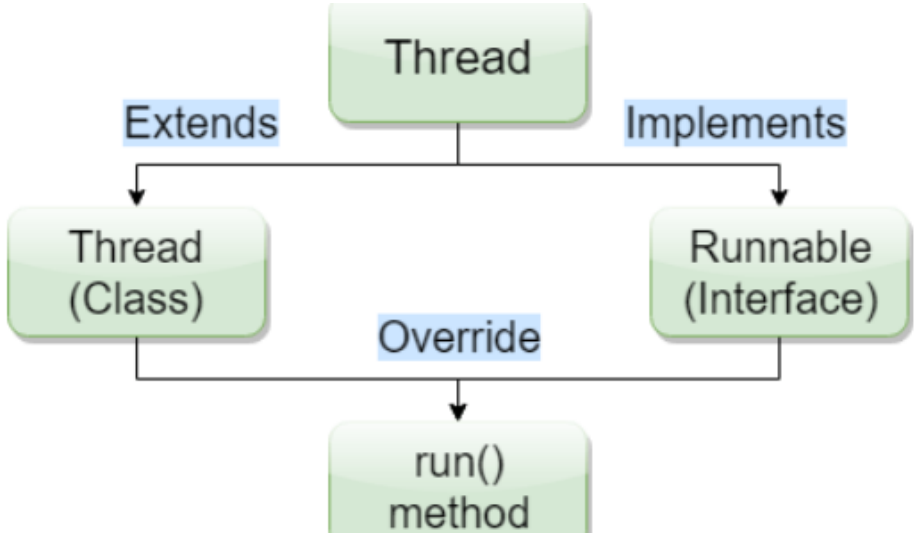
Utilities

- Generic collections
- Lambdas & Stream
- Exception handling
- Files & I/O
- Annotations
- Reflection
- JUnit Testing
- Logging

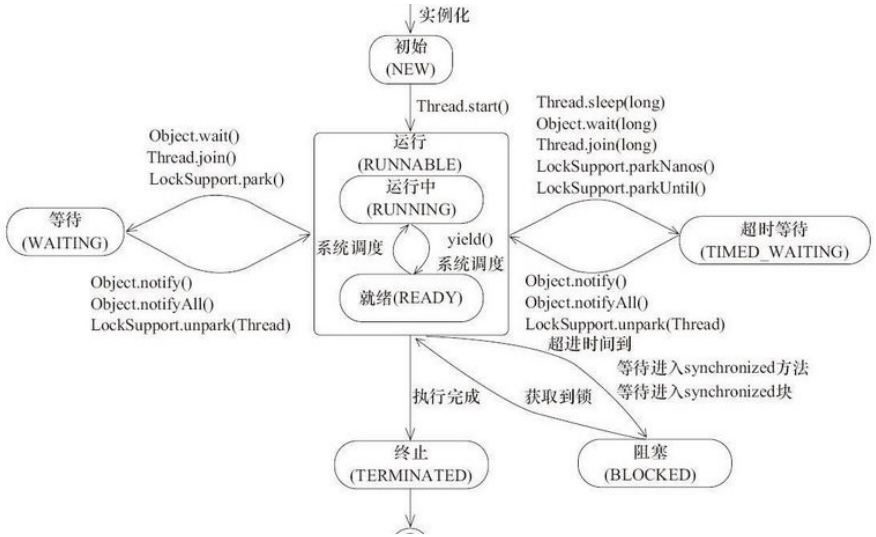
Features

- GUI & JavaFX
- Networking
- Multithreading
- Web development
- Web services

Concurrency

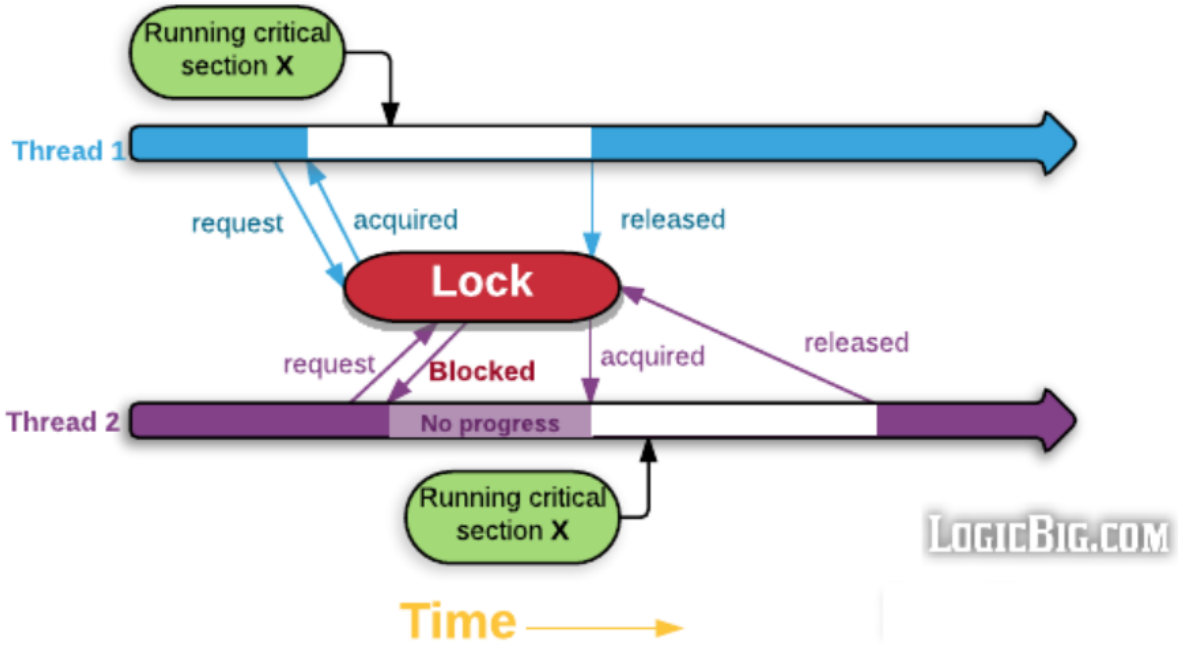


- Creating & Starting Threads
- Thread States



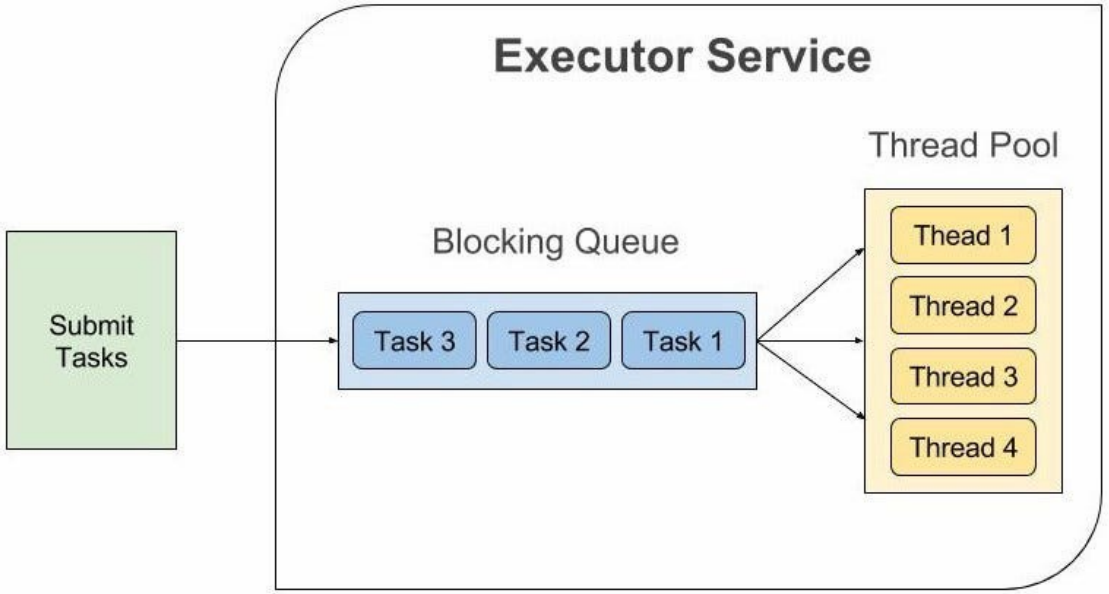
Concurrency

- Thread Safety
- Synchronization
  - The synchronized keyword
  - The Concurrency API (ReentrantLock, Condition)



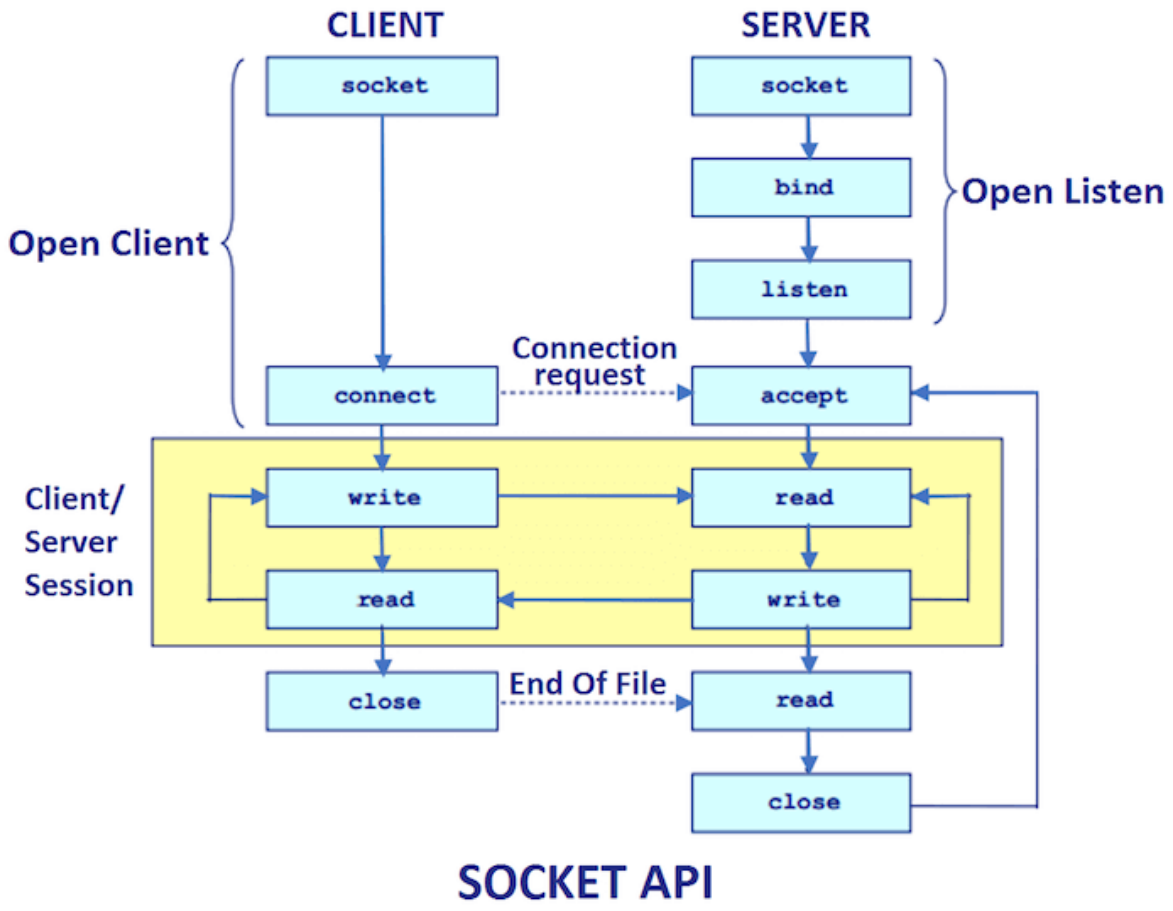
Concurrency

- Task execution
- Thread pools
- Submitting tasks
- Controlling groups of tasks



Networking

- Socket concepts
- Establishing connections
- Reading from and writing to a socket (sending requests & getting responses)



Networking

- Protocols
- Using socket and multithreading to implement basic client & server programs

Table 2 A Simple Bank Access Protocol

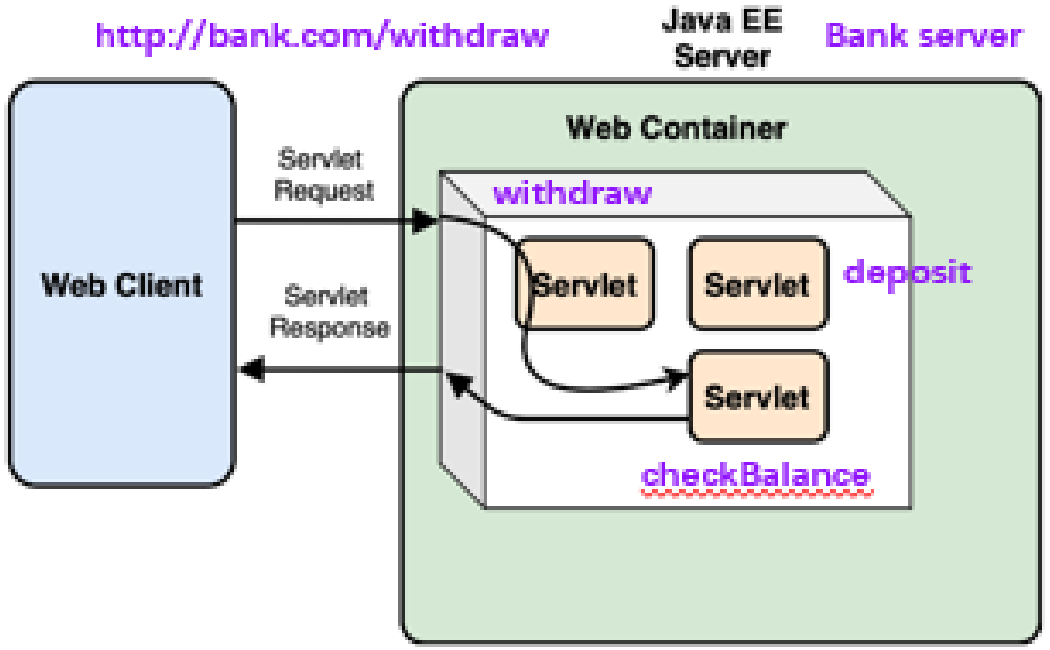
Client Request	Server Response	Description
`BALANCE n`	*n* and the balance	Get the balance of account *n*
`DEPOSIT n a`	*n* and the new balance	Deposit amount *a* into account *n*
`WITHDRAW n a`	*n* and the new balance	Withdraw amount *a* from account *n*
`QUIT`	None	Quit the connection

Java EE

- Java EE multitiered model
- Common Java EE specifications and corresponding implementations

Servlet

- Concepts
- Workflow
- Implementation
- Lifecycle
- Containers



Data Persistence

- JDBC
- ORM
- JPA
- Hibernate

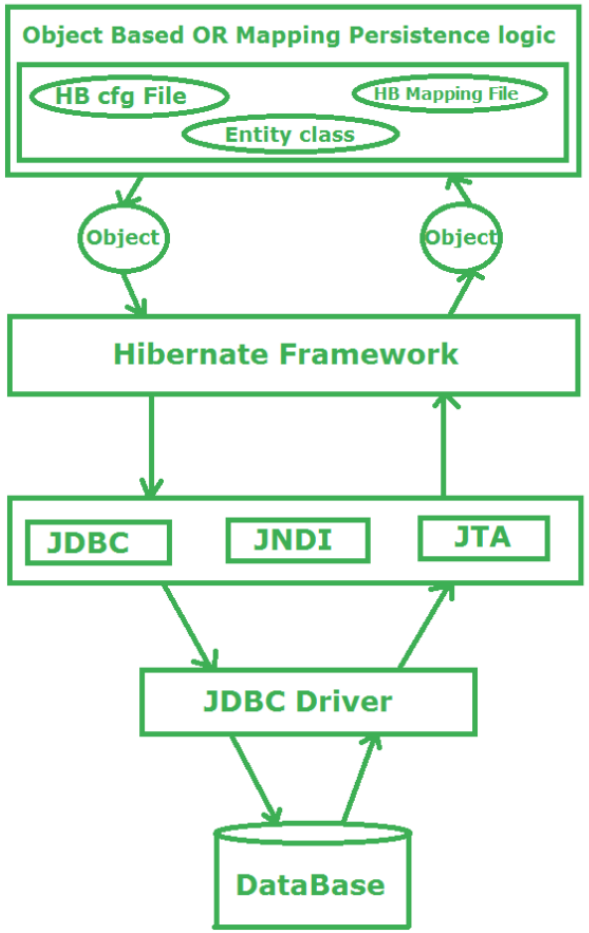
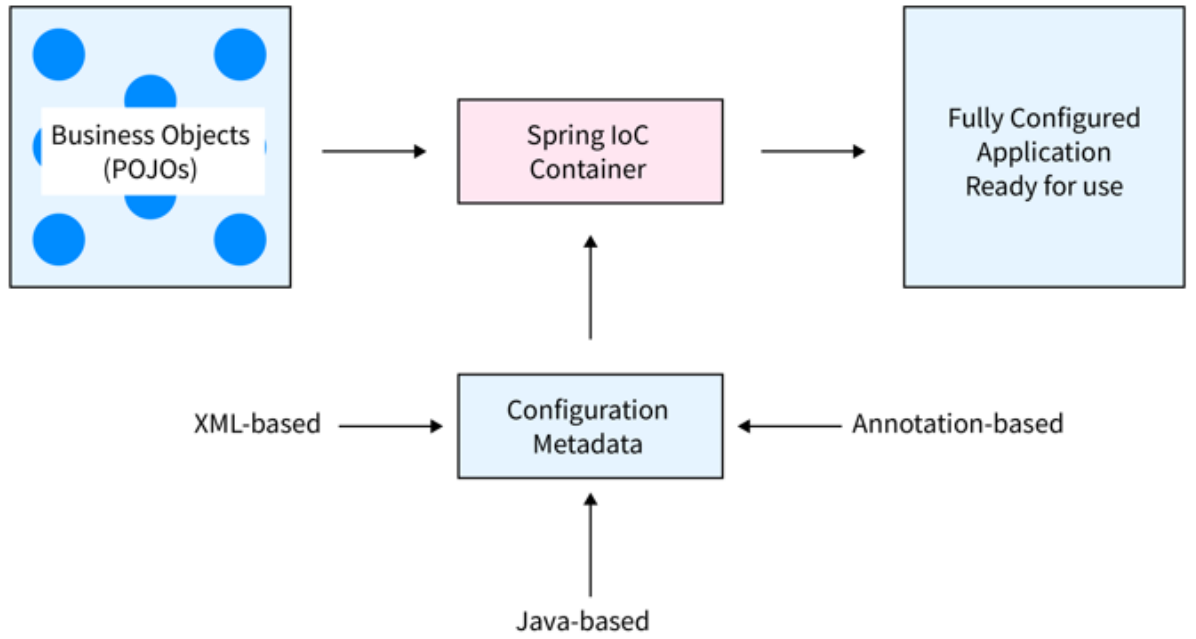


Fig: Working Flow of Hibernate framework to save/retrieve the data from the database in form of Object

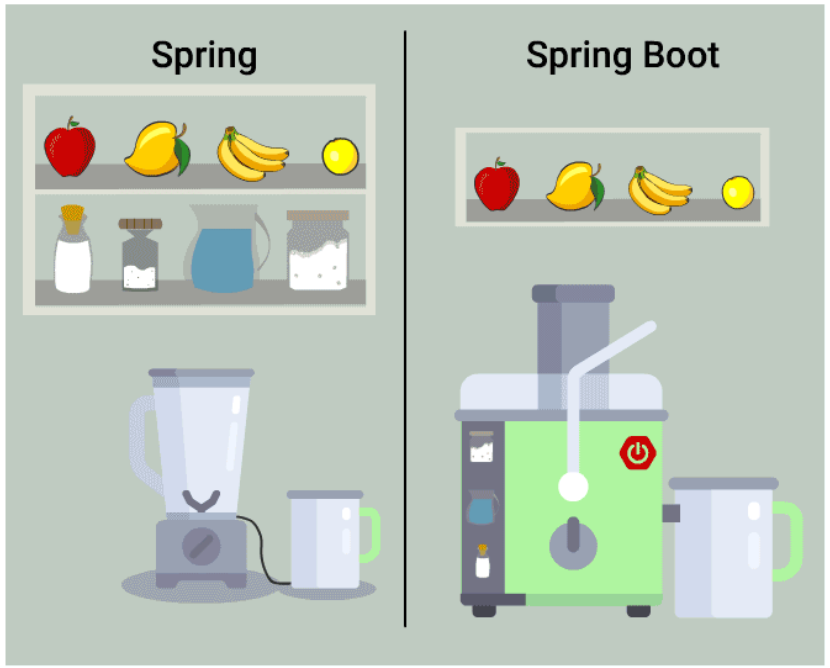
The Spring Framework

- IoC & Dependency Injection (typical annotations)
- Spring AOP concepts and terminology
- Spring MVC basic workflow



Spring Boot

- Basic concepts
- Workflow architecture
- Building a MVC web application
- Building a RESTful web service



JavaFX

- Basic concepts
  - Stage, scene, scene graph, node
  - Panes, controls, charts

Grading Policy

	Score	Description
Assignments	25%	2 assignments Assignment 1: release at week 4 and due at week 7 Assignment 2: release at week 8 and due at week 11
Project	20%	Released at around week 10 Team: 2 people +0.5 for submitting the final project at week 15 +1 (max) for presenting at week 16 lecture
Labs	15%	Attendance Lab practices (+0.1 for finishing onsite. max +1)
Quiz	10%	Quizzes, exercises, participation during lectures
Final Exam	30%	Close-book (Two pieces of A4 cheat sheets allowed) No electronic device

Finally

- Q&A
- Teaching Evaluation

1. 网页端：登录教务系统：https://tis.sustech.edu.cn/-业务办理-评教任务-2024秋季学期学生评价任务。系统按课程类型设置评价任务（理论类、实验实践类、体育类、艺术类），如页面上有多个评价任务，请逐一进入并提交评价。

2. 微信端：通过微信进入“南方科技大学”微信企业号--教学质量管理平台，在“我的任务-待评”中填写并提交本学期所选课程的所有听课评价。

操作指南请扫描下方二维码获取：

Thank You & Good Luck!