## Necessary Tools

Java Development Kit 11+

Apache Maven 3.6.0+

Git 2.24+

##Task

(1-2 stars)

##### 1. Create maven project with 5 modules (see Java 9 modules) :

- jmp-bank-api

- jmp-service-api

- jmp-cloud-bank-impl

- jmp-cloud-service-impl

- jmp-dto

##### 2. Create the fallowing classes under jmp-dto module :

- [User]

String name;

String surname;

LocalDate birthday;

- [BankCard]

String number;

User user;

- [Subscription]

String bankcard;

LocalDate startDate;

##### 3. Extend BankCard class with :

- CreditBankCard

- DebitBankCard

##### 4. Create enum :

- [BankCardType]

CREDIT

DEBIT

##### 5. Add Bank interface to jmp-bank-api with :

- BankCard createBankCard(User, BankCardType);

##### 6. Add module-info.java which :

- requires jmp-dto

- export Bank interface

##### 7. Implement Bank into jmp-cloud-bank-impl. Method should create a new class depending on the type:

##### 8. Add module-info.java which :

- requires transitive module with Bank interface

- requires jmp-dto

- exports implementation of Bank interface

##### 9. Add Service interface to jmp-service-api with :

- void subscribe(BankCard)

- Optional<Subscription> getSubscriptionByBankCardNumber(String)

**I needed to add cardNumber in Subscription class under jmp-dto**

- List<User> getAllUsers();

##### 10. Add module-info.java which :

- requires jmp-dto

- export Service interface

##### 11. Implement Service into jmp-cloud-service-impl. User Stream API, You can use Memory/DB for data storing :

##### 12. Add module-info.java which :

- requires transitive module with Service interface

- requires jmp-dto

- exports implementation of Service interface

##### 13. Use var for definition of local variables wherever it’s applicable.

##### 14. Use lambdas and Java 8 features wherever it’s applicable.

##### 15. Create application module.

##### 16. Add module-info.java which :

- use interfaces

- requires module with Bank implementation

- requires module with Service implementation

- requires jmp.dto;

##### 17. Demonstrate functionality of created classes.

(3-4 stars)

##### 18. Add a new default method for Service interface, which uses getAllUsers. Use LocalDate.now(), ChronoUnit и mapToLong.

- double getAverageUsersAge();

##### 19. Add a new static method for Service interface, which returns true, if the user is over 18 years old.

- boolean isPayableUser(User);

##### 20. Use Collectors.toUnmodifiableList() and method reference it’s applicable.

(5 stars)

##### 21. Create Exception class which will be used into orElseThrow for Optional from getSubscriptionByBankCardNumber().

##### 22. Add a new method for Service interface and implement it. Demonstrate this method ino application module:

- List<Subscription> getAllSubscriptionsByCondition(Predicate<Subscription>);

##### 23. Reimplement createBankCard with method reference (ex: CreditBankCard::new).

##### 24. Add to module-info jar for each implementation module :

- provides [interface] with [implementation]

##### 25. Add a module with Service implementation. Try to use ServiceLoader.load() for module loading.

## References

* [Java 8 new features](https://www.journaldev.com/2389/java-8-features-with-examples)
* [Java 9 new features](https://www.journaldev.com/13121/java-9-features-with-examples)
* [- modules](https://www.baeldung.com/java-9-modularity)
* [Java 10 new features](https://www.journaldev.com/20395/java-10-features)
* [- var](https://dzone.com/articles/var-work-in-progress)
* [Java 11 new features](https://www.journaldev.com/24601/java-11-features)