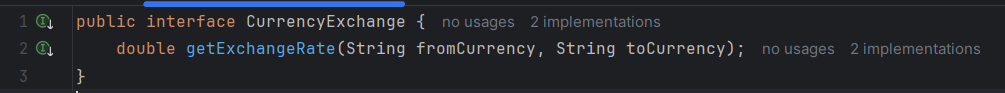
**Student: Sabyrov Sanzhar  
Group: SE-2425  
Project Report: Currency Exchange System using the Adapter Pattern  
Link to the GitHub: https://github.com/SanzharS1/SDA\_ASS\_3.git**

**Project Objective:**

**The objective of this project is to demonstrate the use of the Adapter Pattern to integrate different data sources providing currency exchange rates into a unified system. We create adapters that convert the interfaces of different APIs for retrieving exchange rates into a common interface that the client will use. This allows us to work with incompatible interfaces and integrate different systems without modifying their source code.**

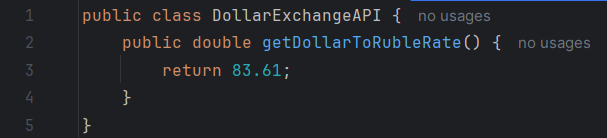
**Step-by-Step Description:**

1. **Creating the Target Interface:  
   First, we created the CurrencyExchange interface, which defines the method getExchangeRate() to retrieve exchange rates. This interface will be used by the client.**

****

**2. Creating the Adaptee Classes:  
Next, we created two adaptee classes that provide exchange rates but have incompatible interfaces:**

* **DollarExchangeAPI: returns the exchange rate from USD to RUB.**
* **EuroExchangeAPI: returns the exchange rate from EUR to RUB.**

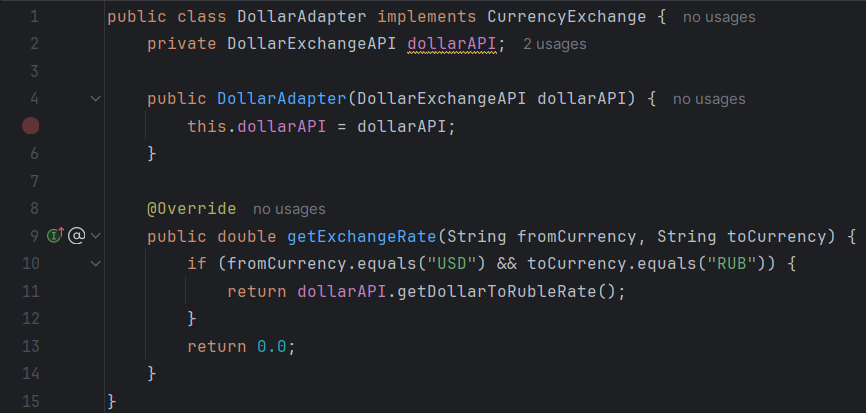
****

**Изображение выглядит как текст, Шрифт, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.**

**3. Creating the Adapters:  
We then created adapters for each of the adaptee classes. The adapters implement the CurrencyExchange interface and delegate the calls to the respective API methods.**

* **DollarAdapter: adapts the DollarExchangeAPI to the CurrencyExchange interface.**
* **EuroAdapter: adapts the EuroExchangeAPI to the CurrencyExchange interface.**

****

**Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

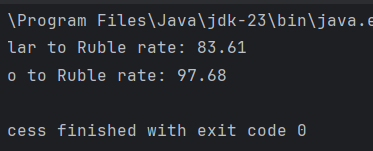
Содержимое, созданное искусственным интеллектом, может быть неверным.**

**4. Creating the Client:  
The client code works with the CurrencyExchange interface and does not know the specific APIs used for retrieving the exchange rates. The client simply calls the getExchangeRate() method on the adapters.**

**Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Содержимое, созданное искусственным интеллектом, может быть неверным.**

**Output:**

****

**The program outputs the exchange rates for USD and EUR to RUB, using the adapters that ensure compatibility between different APIs and the common CurrencyExchange interface.**

**Conclusion:**

**In this project, we demonstrated the correct usage of the Adapter Pattern to integrate various data sources (in this case, currency exchange rate APIs) into a unified system. By creating adapters for different classes, we ensured that they all conformed to the same interface, CurrencyExchange, allowing the client code to interact with the APIs seamlessly. The Adapter Pattern provided flexibility and compatibility, enabling us to work with multiple sources of data without modifying their existing code. This approach improves the system's extensibility and maintainability while allowing us to handle different data sources in a uniform way.**