27 Lecture - CS504

Important Subjective

Q: What is the Observer Pattern? **A**: The Observer Pattern is a behavioral design pattern that facilitates real-time communication between objects, where the subject notifies its registered observers about state changes. **Q: Explain the key components of the Observer Pattern.** **A**: The main components are the Subject, which maintains a list of observers, and the Observers, which register with the subject to receive updates. **Q: How does the Observer Pattern promote loose coupling?** **A**: Observers depend only on the subject's interface, reducing direct dependencies and enhancing flexibility and maintainability. **Q: What is the purpose of the Dependency Inversion Principle in the context of the Observer Pattern?** **A**: The Dependency Inversion Principle guides the pattern's implementation, ensuring that high-level modules (observers) depend on abstractions (subject) rather than concrete implementations. **Q: Describe the sequence of actions when a subject's state changes in the Observer Pattern.** **A**: The subject notifies all registered observers, and they update themselves based on the state change. **Q: How can the Observer Pattern enhance code reusability in software systems?** **A**: By decoupling the subject and observers, the same subject can notify different observers, promoting code reuse for various functionalities. **Q: Provide an example scenario where the Observer Pattern is useful.** **A**: In a stock market application, various investors (observers) need real-time updates when the stock prices (subject) change. **Q: What challenges should developers consider when using the Observer Pattern?** **A**: Developers should be mindful of potential memory leaks or performance issues when dealing with a large number of observers. **Q: How does the Observer Pattern differ from the Publish-Subscribe Pattern?** **A**: The Observer Pattern involves a one-to-many relationship, while the Publish-Subscribe Pattern enables many-to-many communication. **Q: Can a subject have different types of observers in the Observer Pattern?** **A**: Yes, a subject can have different types of observers that implement a common interface, enabling a flexible and heterogeneous observer collection.