# Visitor Design Pattern and Encapsulation

Zoran Horvat

@zoranh75 | www.codinghelmet.com

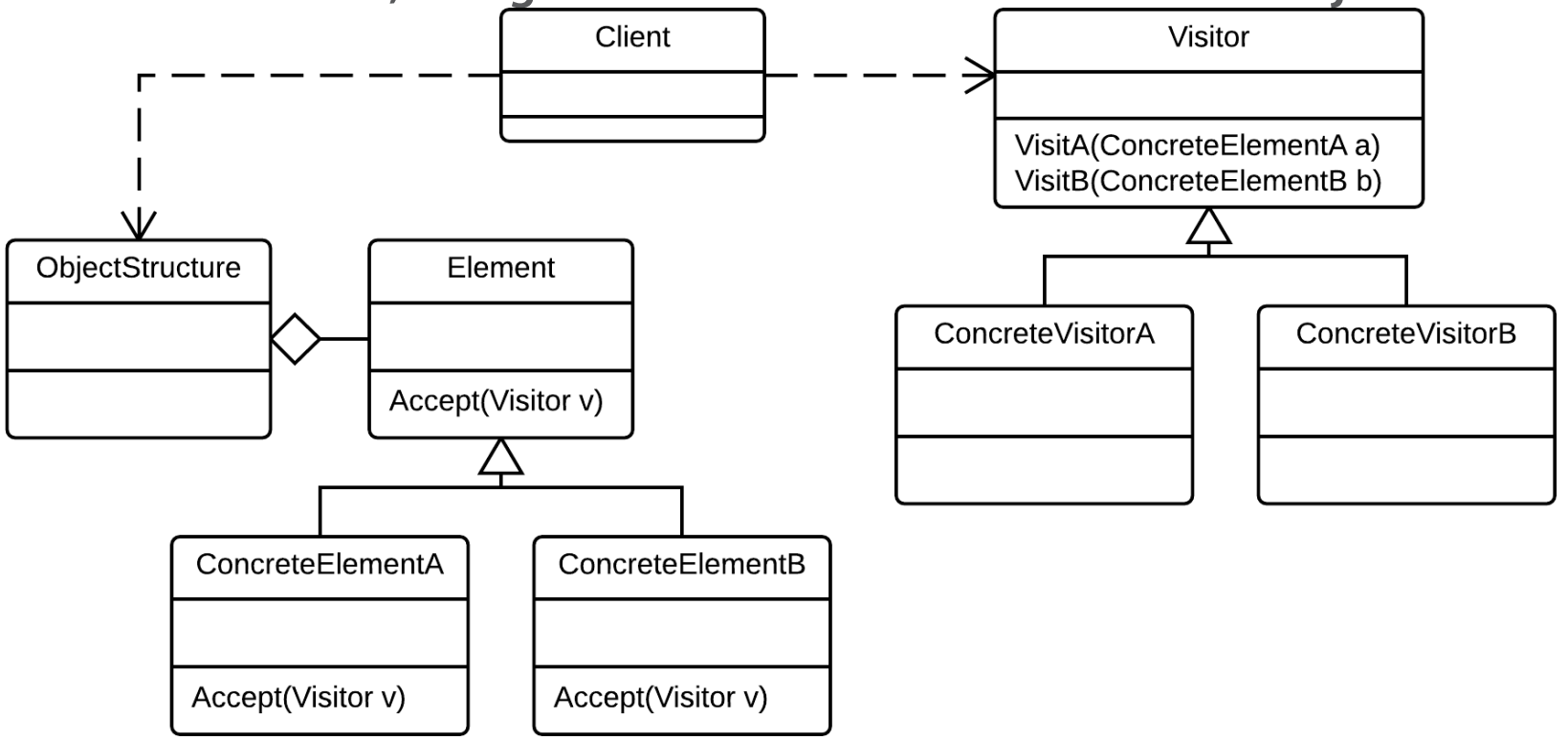
# Behavioral Design Patterns

- Widely used behavioral patterns
  - Chain of Responsibility, State, Strategy, Template Method
- Behavioral patterns built into frameworks
  - Command, Iterator, Observer
- Narrowly-scoped behavioral patterns
  - Interpreter, Mediator, Memento

# Visitor Design Pattern

*"Represents an operation to be performed on the elements of an object structure. Visitor lets you define a new operation without changing the classes of the elements on which it operates."*

**Gama et al., Design Patterns: Elements of Reusable Object-Oriented Software**



# Visitor Design Pattern

- Lets us add operations to existing hierarchy of classes
  - Existing classes do not have to be modified to support new visitor
  - Visitor must provide one method for every member of the class hierarchy
  - Adding new class to the hierarchy causes all visitors to change
- What follows in this module
  - We assume fair understanding of the Visitor design pattern
  - Examples will build on basic knowledge of this design pattern
- Note on hierarchical and linear data structures
  - It is not mandatory to have such data structure
  - Visitor design pattern is applicable to any data structure

# Summary

- Visitor design pattern and encapsulation
  - It is possible to apply the visitor even when object encapsulates its data
  - Visitor can receive raw data
  - No need to expose public accessors on the visited object
- Visitor objects are typically stateful
  - New instance of the visitor must be passed to Accept every time
  - We will address this inconvenience in the next module

# Summary

- Visitors often used as part of the result which produces result
  - Visitor could be designed to produce concrete result
  - This will be addressed in the next module
- Order of invocations
  - Never assume particular order in which Visit methods will be invoked
  - Client should be free to invoke methods in any order

# Summary

- Problem implementing business operations
  - Referenced objects are also encapsulating their data
  - In order to complete the operation, objects must work together
  - Visitor can be used to orchestrate multiple objects into a single operation