Object-Oriented Analysis and Design

DESIGN BY CONTRACT

Design by Contract

- Classes are modules
- Modules have client-supplier relationships

- Specification of classes is a formal agreement expressing rights and obligations of clients and suppliers: a **Contract**
  - DbC is used to distribute responsibility between a class and its clients

Design by Contract

- DbC is a practical methodology for evolving code together with its specification
- The contract is expressed as a set of class invariants and method pre- and post-conditions
  - These assertions have important methodological implications on the client-supplier and inheritance relationships between classes

Design by Contract

- Contracts describe an agreement between the writer of a class and the user of the class by means of assertions
  - Precondition obligates clients
    - Clients may only call methods when their preconditions are satisfied
    - Under no circumstances shall the body of a routine ever test for the routine's precondition
  - Postcondition obligates suppliers
    - Suppliers guarantee that post-conditions are satisfied on exit from a method

Design by Contract

- Invariants define correct states for objects belonging to the class
  - It is the suppliers' responsibility to maintain invariants
  - An invariants is a consistency constraint
  - { inv $\wedge$ pre } f() { inv $\wedge$ post }

Design by Contract

- A contract violation corresponds to SW errors
  - Supplier does not satisfy its specification
  - The clients expectations exceed the contract
- How is a contract expressed?
  - Pre- and post-conditions on methods
  - Class invariants

Pre- and Post-conditions

Correctness formulae

- Statement A: $\{P\}\ A\ \{Q\}$
  - Any execution of program *A*, starting in a state for which *P* holds, will terminate in a state in which *Q* holds
  - Example: $\{x<5\}\ \ x+=5\ ;\ \{x<10\}$

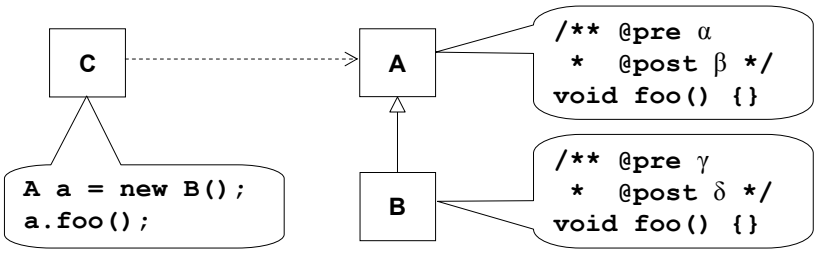
Benefits vs Obligations

	Benefit	Obligation
Client	- no need to check output values - result guaranteed to comply to postcondition (4)	satisfy precondition (1)
Supplier	- no need to check input values - input guaranteed to comply to precondition (2)	satisfy postconditions (3)

Definition

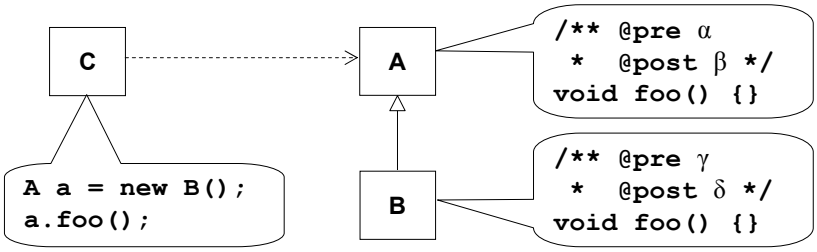
- A predicate P is stronger than a predicate *Q* if whenever P is true, Q is also true
  - that is, when P → Q holds

Behavioral Subtyping (Liskov)



- Preconditions in subclasses
  - May only be weakened (α → γ)
  - Computed as disjunctions (OR-ed)

Behavioral Subtyping (Liskov)



- Invariants and postconditions in subclasses
  - May only be strengthened ($\delta \rightarrow \beta$)
  - Computed as conjunctions (AND-ed)

Implementation of DBC

- Eiffel
  - Integral part of the language
    - require, ensure, invariant
- Java
  - External tools
  - iContract, JMassert, JML, JASS, JOSE