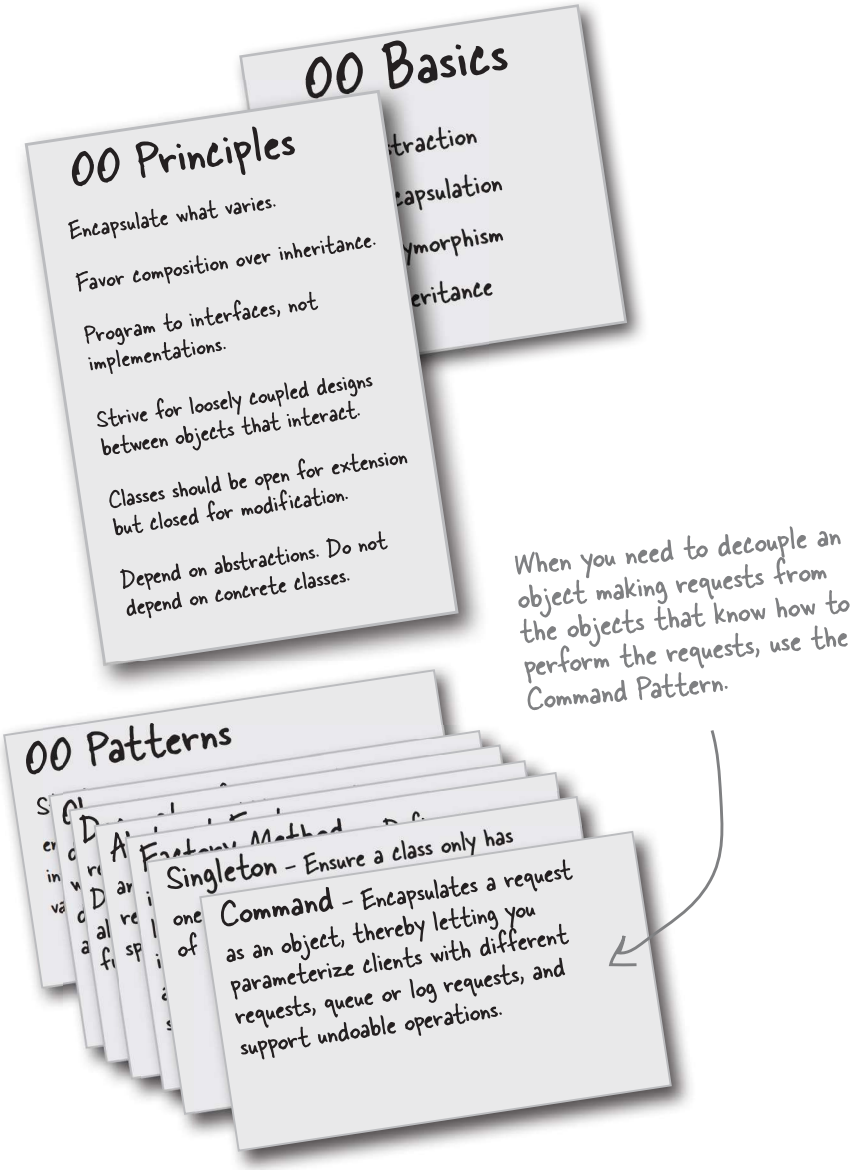
Tools for your Design Toolbox

Your toolbox is starting to get heavy! In this chapter we've added a pattern that allows us to encapsulate methods into Command objects: store them, pass them around, and invoke them when you need them.



BULLET POINTS

- The Command Pattern decouples an object making a request from the one that knows how to perform it.
- A Command object is at the center of this decoupling and encapsulates a receiver with an action (or set of actions).
- An invoker makes a request of a Command object by calling its execute() method, which invokes those actions on the receiver.
- Invokers can be parameterized with Commands, even dynamically at runtime.
- Commands may support undo by implementing an undo() method that restores the object to its previous state before the execute() method was last called.
- MacroCommands are a simple extension of the Command Pattern that allow multiple commands to be invoked. Likewise, MacroCommands can easily support undo().
- In practice, it's not uncommon for "smart" Command objects to implement the request themselves rather than delegating to a receiver.
- Commands may also be used to implement logging and transactional systems.