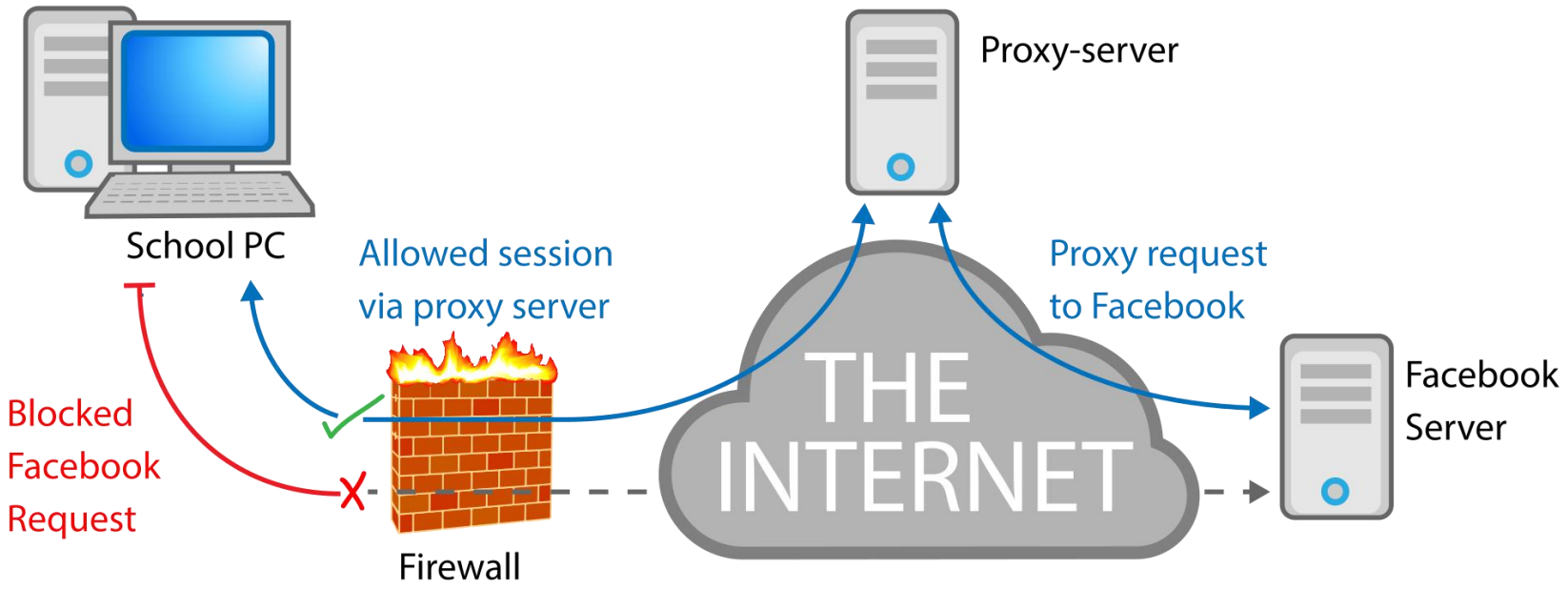
Proxy (代理, Structural Pattern)

Kai SHI

Proxy Server



Proxy Pattern

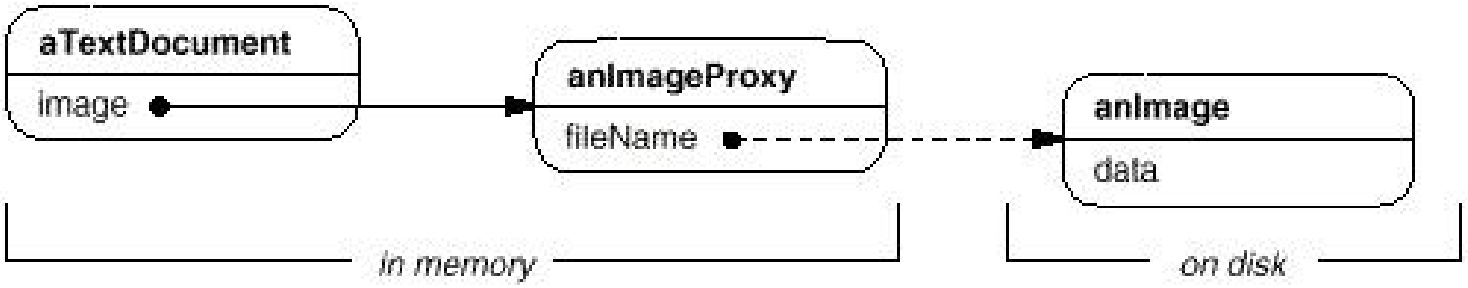
- Intent
  - Provide a surrogate (代理) or placeholder for another object to control access to it.
  - 代理模式给某一个对象提供一个代理对象，并由代理对象控制对原对象的引用。
- Also Known As
  - Surrogate (代理)

Motivation (1/2)

- One reason for controlling access to an object is to defer (推迟) the full cost of its creation and initialization until we actually need to use it.
- These constraints would suggest creating each expensive object on demand (只在必要时生成实例)
- The solution is to use another object, an proxy, that acts as a stand-in for the real object.

Motivation (2/2)

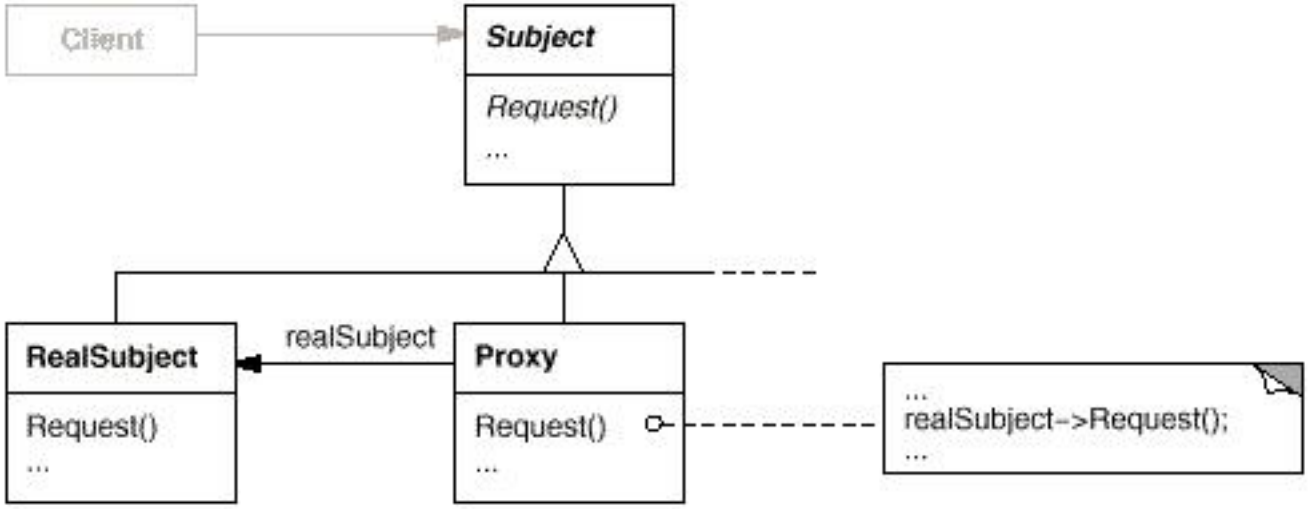
- e.g., An image proxy creates the real image only when the document editor asks it to display itself. The proxy forwards subsequent requests directly to the image.



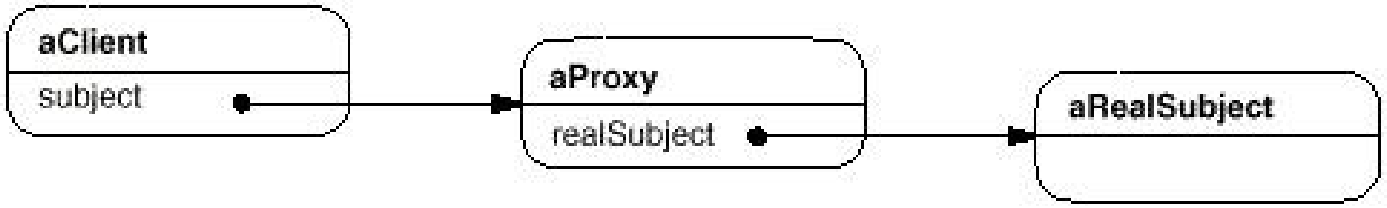
Applicability

- A remote proxy provides a local representative for an object in a different address space.
- A virtual proxy creates expensive objects on demand.
- A protection proxy controls access to the original object.
- A smart reference is a replacement for a bare pointer that performs additional actions when an object is accessed.
  - counting the number of references to the real object so that it can be freed automatically when there are no more references (smart pointers).
  - loading a persistent object (持久化对象) into memory when it's first referenced.
  - checking that the real object is locked before it's accessed to ensure that no other object can change it.

Structure



Here's a possible object diagram of a proxy structure at run-time:



Participants

- Subject
  - Defines the common interface for RealSubject and Proxy so that a Proxy can be used anywhere a RealSubject is expected.
- RealSubject
  - Defines the real object that the proxy represents.
- Proxy
  - Maintains a reference that lets the proxy access the real subject. Proxy may refer to a Subject if the RealSubject and Subject interfaces are the same.
  - Provides an interface identical to Subject's so that a proxy can by substituted for the real subject.
  - Controls access to the real subject and may be responsible for creating and deleting it.

Collaborations

- A Proxy contains the reference of a RealSubject, thus it can manipulate the a RealSubject in anytime;
- A Proxy provides an interface which is in according with the interface of a RealSubject, thus it can replacement the a RealSubject in anytime;
- A Proxy controls the reference of a RealSubject, that is, it create and destroy the a RealSubject;
- A Proxy forwards requests to a RealSubject when appropriate, depending on the kind of proxy.

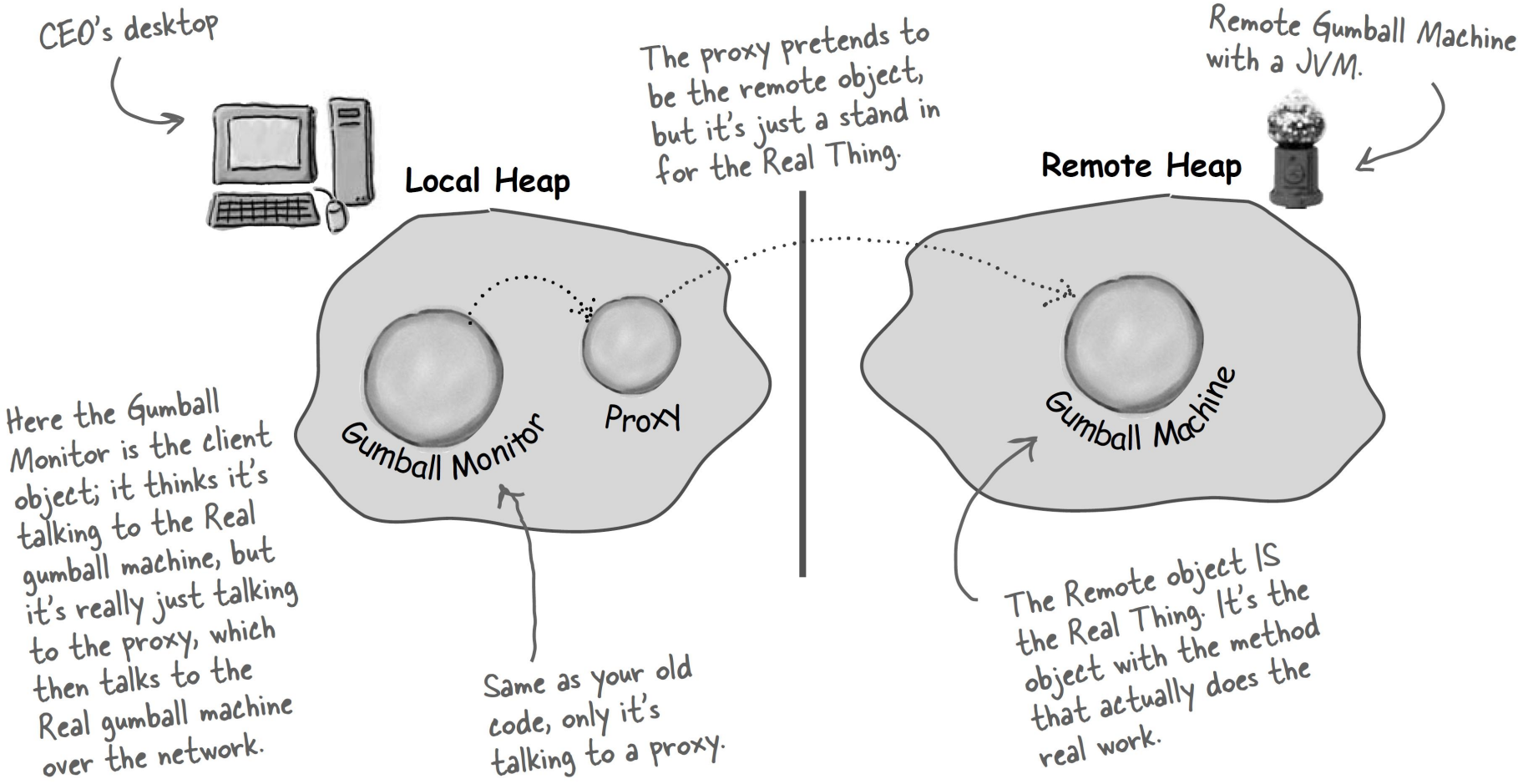
Consequences

- A remote proxy can hide the fact that an object resides in a different address space.
- A virtual proxy can perform optimizations such as creating an object on demand.
- Both protection proxies and smart references allow additional functionalities such as access control. (shown in Applicability page)
- copy-on-write
  - Copying a large and complicated object can be an expensive operation. If the copy is never modified, then there's no need to incur this cost. By using a proxy to postpone the copying process, we ensure that we pay the price of copying the object only if it's modified.

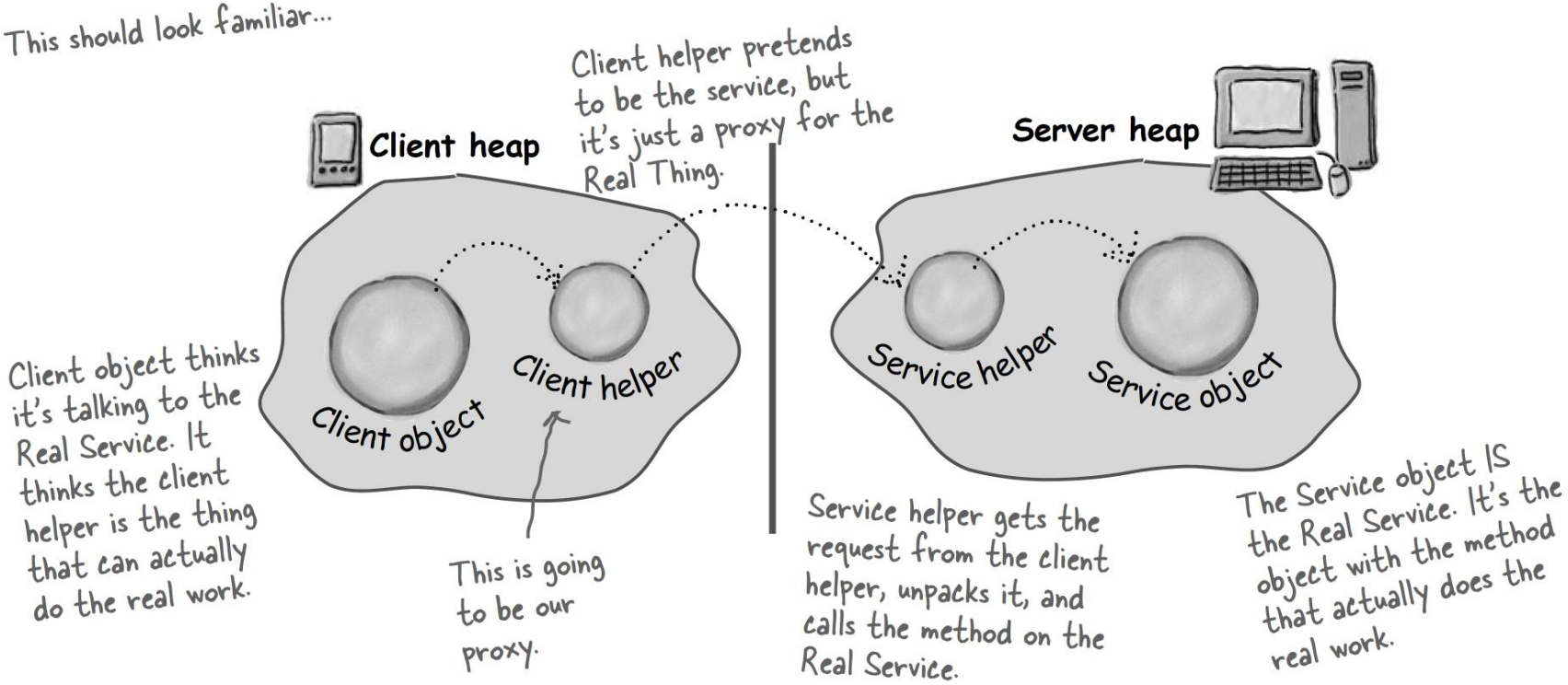
Example

- Monitor gumball machines
  - Find a way to give a report of inventory (库存) and machine state.
  - First Try
    - Code:
      - net.dp.proxy.gumballmonitor.GumballMonitor
      - net.dp.proxy.gumballmonitor.GumballMachineTestDrive
    - Problem: It is a local version, but we want an Internet version.

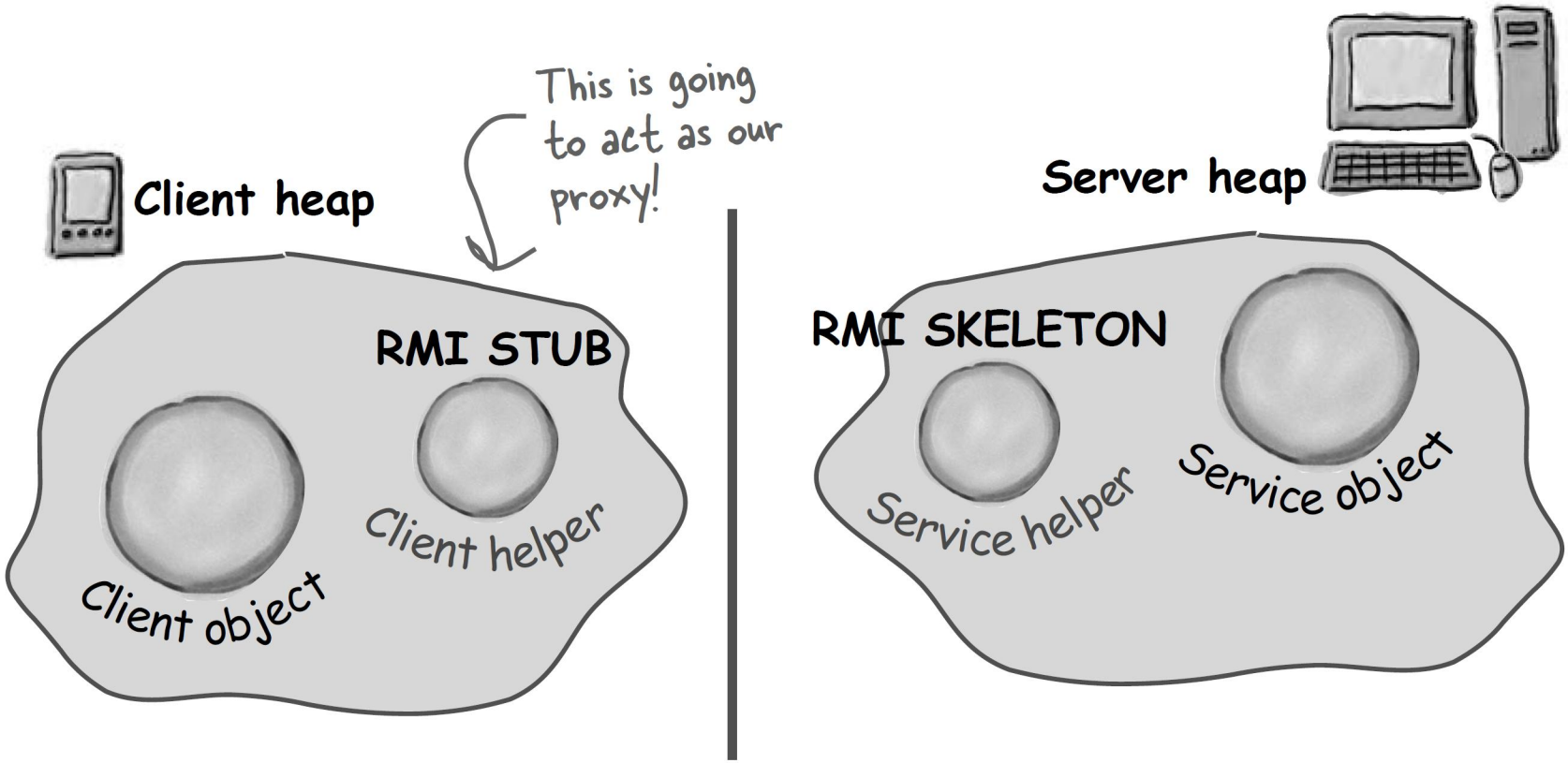
Remote Proxy: Acts as a local representative to a remote object.



Remote Methods



Java RMI: The client helper is a 'stub' and the service helper is a 'skeleton'.



After Class

- Find out how to implement an Internet version
  - Code: net.dp.proxy.gumball

Extension: Transparent proxy

- Proxy doesn't always have to know the type of real subject.
- If a Proxy class can deal with its subject solely through an abstract interface, then there's no need to make a Proxy class for each RealSubject class; the proxy can deal with all RealSubject classes uniformly.
- But if Proxy is going to instantiate, then they have to know the concrete class.