

Coping with Design Changes

- Costly and difficult
- Require the flexibility to be built-in from the very first draft of design
- How can we do this efficiently and effectively?
- Structural Patterns provide some answers
- Other answers rely on the development process which is outside of the scope of this course

Structural Patterns

- Concerning how classes are composed to form larger structures
- Let you vary some aspects of system structure independently of other aspects
- Help making a system more robust to a particular kind of structural change

There are two types of structural patterns:

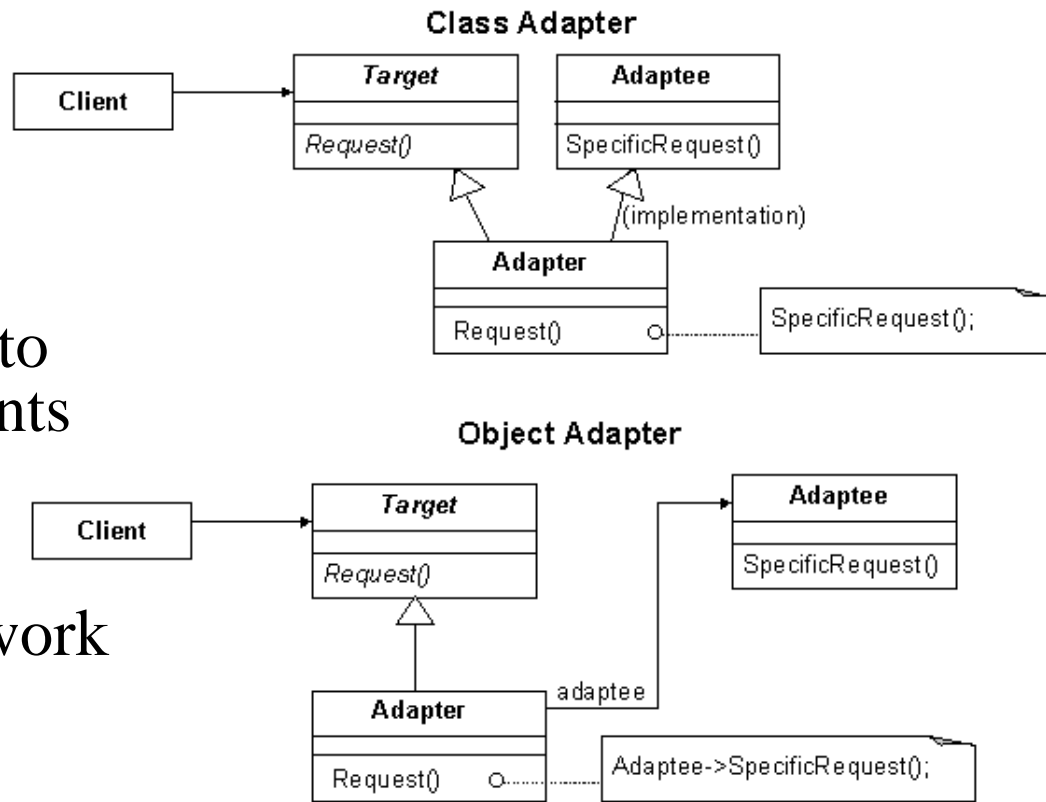
- *structural class patterns* : use class inheritance to compose interfaces or implementations
 - e.g., Class Adapter
- *structural object patterns* : use object composition to realize new functionality
 - e.g., Object Adapter

Adapter (1)

Intent: Convert the interface of a class into another interface clients expect.

Adapter lets classes work together that couldn't otherwise because of incompatible interfaces.

The adapter is responsible for functionality the adapted class doesn't provide.



Adapter (2)

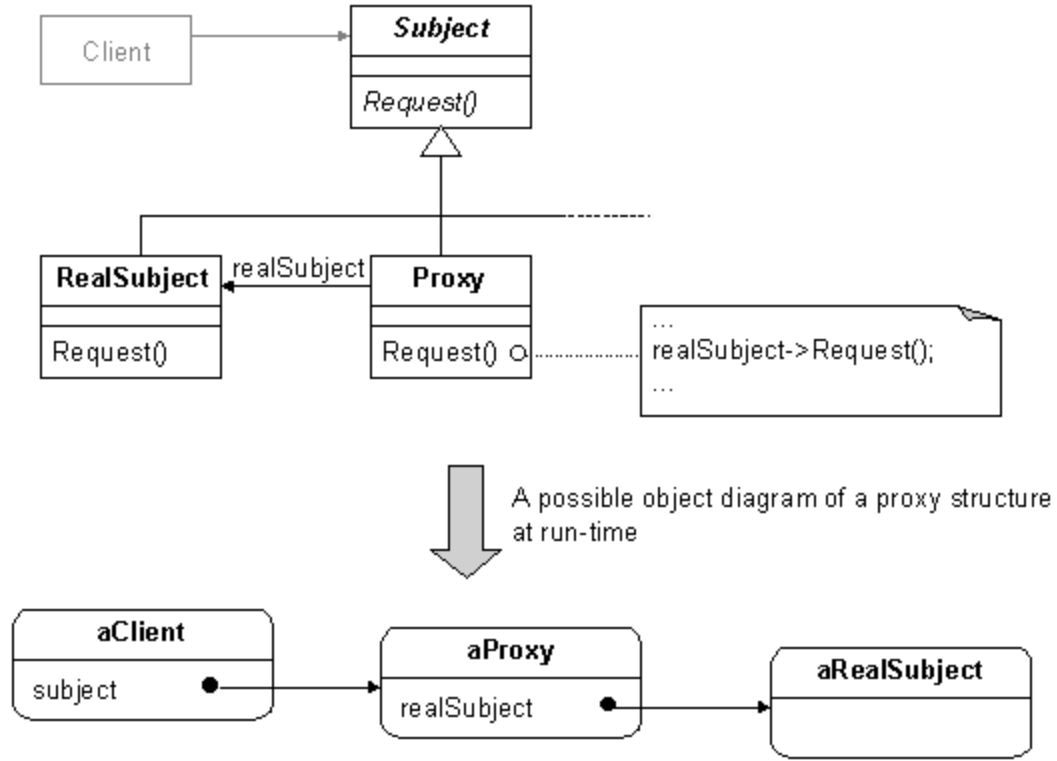
- Lets you vary interface to an object
- Applicable when
 - you want to use an existing class, and its interface does not match the one you need.
 - you want to create a reusable class that cooperates with unrelated or unforeseen classes,
 - you need to use several existing subclasses, but it is impractical to adapt their interface by subclassing every one. An object adapter can adapt the interface of its parent class.

Class Adapter vs. Object Adapter

- Inherits both from the interface and the off-the-shelf class (the Adaptee)

- Inherit from the interface, but compose an instance of the off-the-shelf class
- Particularly when multiple class inheritance is not possible (eg. Java)

Proxy (1)



Intent: Provide a surrogate or placeholder for another object to control access to it.

Proxy (2)

- Lets you vary how and where an object is accessed
- Proxy is applicable whenever there is a need for more versatile or sophisticated reference to an object than a simple pointer
 - A *remote proxy* provides a local representative for an object in a different address space
 - A *virtual proxy* creates expensive objects on demand (hides the creation on demand scheme, e.g. copy-on-write)
 - 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. Typical uses of it include:
 - counting the number of references to the real object
 - loading a persistent object into memory when it is first referenced (lazy instantiation)
 - checking that the real object is locked before it is accessed to ensure that no other object can change it

Adapter vs. Proxy

- Implements a different class (Adaptee) under the Adapter's interface (Target)

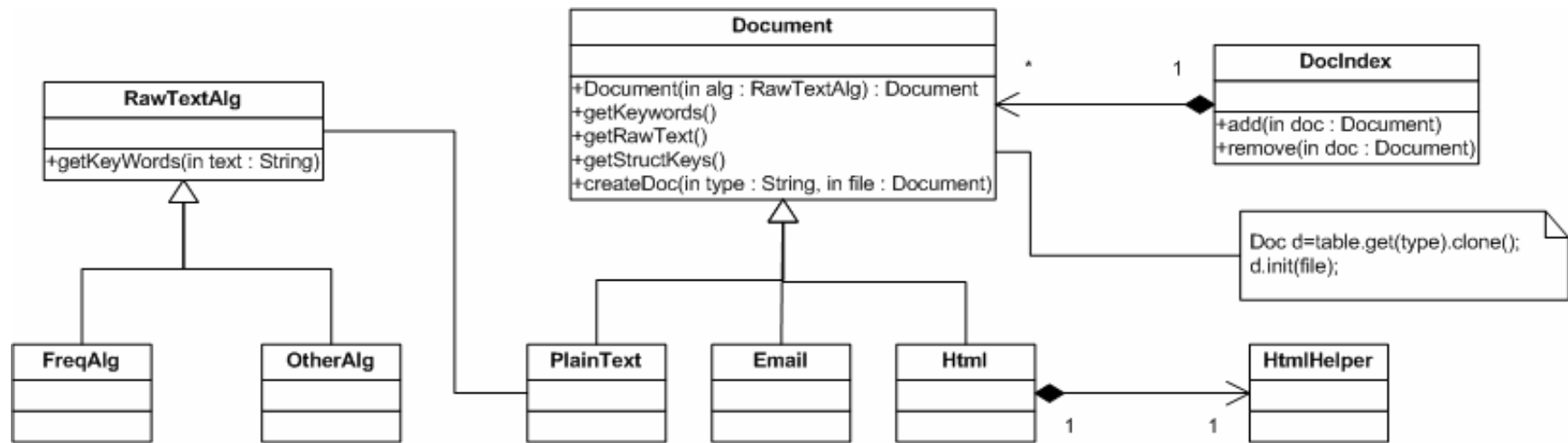
- Implements a subset of the RealSubject's functions under the same interface (Subject)

Example: Document Retrieval System

- Overview:
 - A system that enables the users to add and remove documents to and from the system, and retrieve documents based on simple keywords queries. This example is taken from 2002 fall term assignment 2.
- Key requirements:
 1. Document types include plain text, email, and HTML
 2. Queries and their results are stored
 3. Query results must be consistent with the current document collections in the system
 4. New queries can be added, and existing queries can be deleted or modified
 5. Allows dynamic use of different raw text keyword extraction algorithms
 6. Allows the use of off-the-shelf HTML keyword extraction package

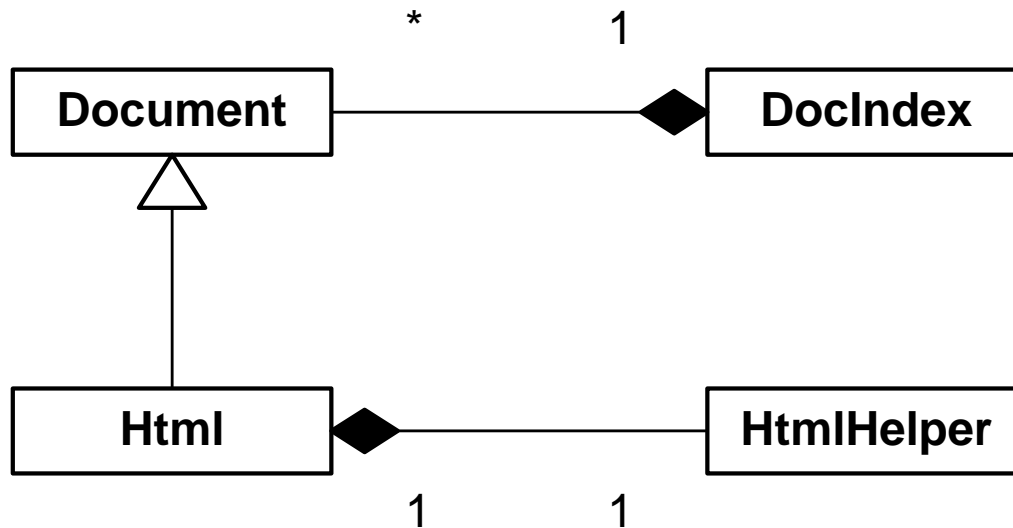
More details of this system are available at:
<http://www.cdf.utoronto.ca/~neilg/CSC407F-2002/A2>

Partial Solution: Document Retrieval System



- HtmlHelper is the accessing class to the off-the-shelf HTML keyword extraction package
- Can you make use of Proxy Pattern in this solution? Why or why not?
 - Hint: Html (Proxy), HtmlHelper (RealSubject) as a subclass of Document
 - More natural to use Adapter

Adapter Pattern Used in the Example



Class Adapter or
Object Adapter pattern?

- Participants:
 - Target : Document
 - Adapter: Html
 - Adaptee: HtmlHelper (external)
 - Client: DocIndex
- Applicability and purpose:
 - Allows the use of externally developed HTML document parsing class
 - Supports Key Req #1 and #6

Hint: object adapter

You are expected to follow this format in your assignment as much as possible.