Design Patterns give a shared vocabulary with other developers.

--------------------------------------------------------------

Encapsulate what varies

Program to interfaces and not implementation

Favor composition instead of inheritance

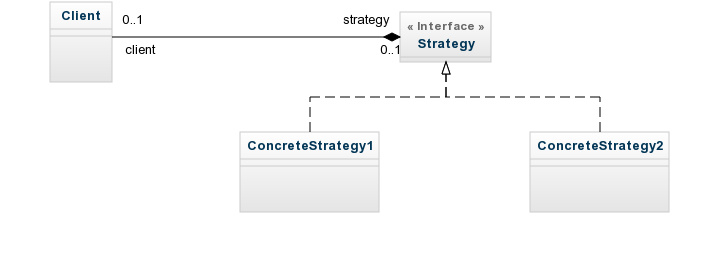
Strive for loosely coupled design between objects that interact

Classes should be open for extension but closed for modification

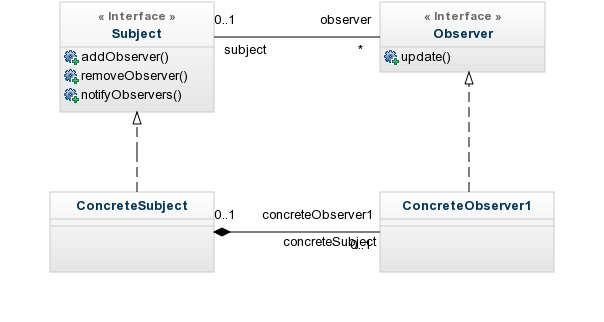
Depend on abstraction; do not depend on concrete classes

**Patterns**:

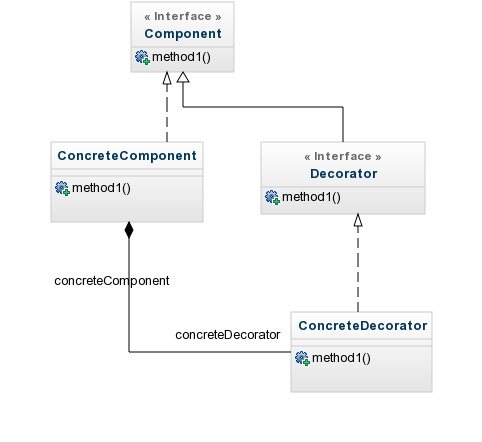
* Strategy Pattern: Defines a family of algorithms, encapsulates each one and makes them interchangeable.



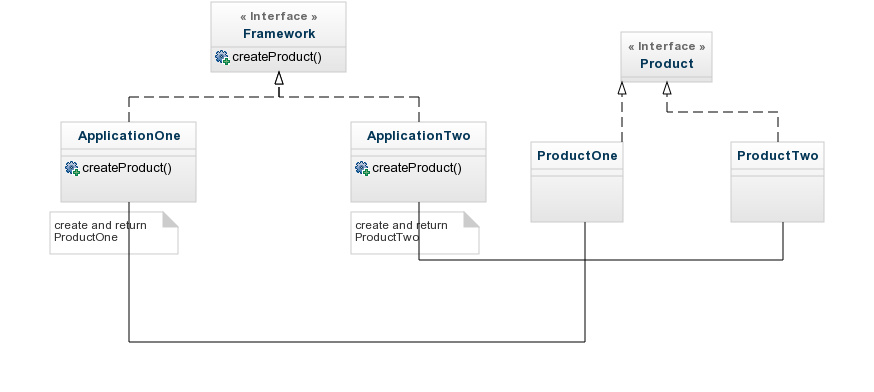
* Observer Pattern: Defines one-to-many relationships between objects, so that when one object the subject changes all its dependents (Observers) are notified of the change and updated automatically.



* Decorator Pattern: A decorator pattern attaches additional responsibilities to an object dynamically. Decorators provide a flexible alternative to subclassing for extending functionality.



* Factory method: Defines an interface for creating an object, but let subclasses decide which class to instantiate. Factory method defers the instantiation to the subclasses.



* Abstract Factory: Provides an interface for creating a families of related or dependent objects without specifying their concrete classes

[Creational patterns](http://en.wikipedia.org/wiki/Creational_pattern)

[Abstract factory](http://en.wikipedia.org/wiki/Abstract_factory_pattern) (recognizeable by creational methods returning the factory itself which in turn can be used to create another abstract/interface type)

* [javax.xml.parsers.DocumentBuilderFactory#newInstance()](http://docs.oracle.com/javase/6/docs/api/javax/xml/parsers/DocumentBuilderFactory.html#newInstance%28%29)
* [javax.xml.transform.TransformerFactory#newInstance()](http://docs.oracle.com/javase/6/docs/api/javax/xml/transform/TransformerFactory.html#newInstance%28%29)
* [javax.xml.xpath.XPathFactory#newInstance()](http://docs.oracle.com/javase/6/docs/api/javax/xml/xpath/XPathFactory.html#newInstance%28%29)

[Builder](http://en.wikipedia.org/wiki/Builder_pattern) (recognizeable by creational methods returning the instance itself)

* [java.lang.StringBuilder#append()](http://docs.oracle.com/javase/6/docs/api/java/lang/StringBuilder.html#append%28boolean%29) (unsynchronized)
* [java.lang.StringBuffer#append()](http://docs.oracle.com/javase/6/docs/api/java/lang/StringBuffer.html#append%28boolean%29) (synchronized)
* [java.nio.ByteBuffer#put()](http://docs.oracle.com/javase/6/docs/api/java/nio/ByteBuffer.html#put%28byte%29) (also on [CharBuffer](http://docs.oracle.com/javase/6/docs/api/java/nio/CharBuffer.html#put%28char%29), [ShortBuffer](http://docs.oracle.com/javase/6/docs/api/java/nio/ShortBuffer.html#put%28short%29), [IntBuffer](http://docs.oracle.com/javase/6/docs/api/java/nio/IntBuffer.html#put%28int%29), [LongBuffer](http://docs.oracle.com/javase/6/docs/api/java/nio/LongBuffer.html#put%28long%29), [FloatBuffer](http://docs.oracle.com/javase/6/docs/api/java/nio/FloatBuffer.html#put%28float%29) and [DoubleBuffer](http://docs.oracle.com/javase/6/docs/api/java/nio/DoubleBuffer.html#put%28double%29))
* [javax.swing.GroupLayout.Group#addComponent()](http://docs.oracle.com/javase/6/docs/api/javax/swing/GroupLayout.Group.html#addComponent%28java.awt.Component%29)
* All implementations of [java.lang.Appendable](http://docs.oracle.com/javase/6/docs/api/java/lang/Appendable.html)

[Factory method](http://en.wikipedia.org/wiki/Factory_method_pattern) (recognizeable by creational methods returning an implementation of an abstract/interface type)

* [java.util.Calendar#getInstance()](http://docs.oracle.com/javase/6/docs/api/java/util/Calendar.html#getInstance%28%29)
* [java.util.ResourceBundle#getBundle()](http://docs.oracle.com/javase/6/docs/api/java/util/ResourceBundle.html#getBundle%28java.lang.String%29)
* [java.text.NumberFormat#getInstance()](http://docs.oracle.com/javase/6/docs/api/java/text/NumberFormat.html#getInstance%28%29)
* [java.nio.charset.Charset#forName()](http://docs.oracle.com/javase/6/docs/api/java/nio/charset/Charset.html#forName%28java.lang.String%29)
* [java.net.URLStreamHandlerFactory#createURLStreamHandler(String)](http://docs.oracle.com/javase/6/docs/api/java/net/URLStreamHandlerFactory.html) (Returns singleton object per protocol)

[Prototype](http://en.wikipedia.org/wiki/Prototype_pattern) (recognizeable by creational methods returning a*different*instance of itself with the same properties)

* [java.lang.Object#clone()](http://docs.oracle.com/javase/6/docs/api/java/lang/Object.html#clone%28%29) (the class has to implement [java.lang.Cloneable](http://docs.oracle.com/javase/6/docs/api/java/lang/Cloneable.html))

[Singleton](http://en.wikipedia.org/wiki/Singleton_pattern) (recognizeable by creational methods returning the*same*instance (usually of itself) everytime)

* [java.lang.Runtime#getRuntime()](http://docs.oracle.com/javase/6/docs/api/java/lang/Runtime.html#getRuntime%28%29)
* [java.awt.Desktop#getDesktop()](http://docs.oracle.com/javase/6/docs/api/java/awt/Desktop.html#getDesktop%28%29)
* [java.lang.System#getSecurityManager()](http://docs.oracle.com/javase/6/docs/api/java/lang/System.html#getSecurityManager%28%29)

[Structural patterns](http://en.wikipedia.org/wiki/Structural_pattern)

[Adapter](http://en.wikipedia.org/wiki/Adapter_pattern) (recognizeable by creational methods taking an instance of*different*abstract/interface type and returning an implementation of own/another abstract/interface type which*decorates/overrides*the given instance)

* [java.util.Arrays#asList()](http://docs.oracle.com/javase/6/docs/api/java/util/Arrays.html#asList%28T...%29)
* [java.io.InputStreamReader(InputStream)](http://docs.oracle.com/javase/6/docs/api/java/io/InputStreamReader.html#InputStreamReader%28java.io.InputStream%29) (returns a Reader)
* [java.io.OutputStreamWriter(OutputStream)](http://docs.oracle.com/javase/6/docs/api/java/io/OutputStreamWriter.html#OutputStreamWriter%28java.io.OutputStream%29) (returns a Writer)
* [javax.xml.bind.annotation.adapters.XmlAdapter#marshal()](http://docs.oracle.com/javase/6/docs/api/javax/xml/bind/annotation/adapters/XmlAdapter.html#marshal%28BoundType%29) and [#unmarshal()](http://docs.oracle.com/javase/6/docs/api/javax/xml/bind/annotation/adapters/XmlAdapter.html#unmarshal%28ValueType%29)

[Bridge](http://en.wikipedia.org/wiki/Bridge_pattern) (recognizeable by creational methods taking an instance of*different*abstract/interface type and returning an implementation of own abstract/interface type which*delegates/uses*the given instance)

* None comes to mind yet. A fictive example would be new LinkedHashMap(LinkedHashSet<K>, List<V>) which returns an unmodifiable linked map which doesn't clone the items, but *uses*them. The [java.util.Collections#newSetFromMap()](http://docs.oracle.com/javase/6/docs/api/java/util/Collections.html#newSetFromMap%28java.util.Map%29) and [singletonXXX()](http://docs.oracle.com/javase/6/docs/api/java/util/Collections.html#singleton%28T%29) methods however comes close.

[Composite](http://en.wikipedia.org/wiki/Composite_pattern) (recognizeable by behavioral methods taking an instance of*same*abstract/interface type into a tree structure)

* [java.awt.Container#add(Component)](http://docs.oracle.com/javase/6/docs/api/java/awt/Container.html#add%28java.awt.Component%29) (practically all over Swing thus)
* [javax.faces.component.UIComponent#getChildren()](http://docs.oracle.com/javaee/6/api/javax/faces/component/UIComponent.html#getChildren%28%29) (practically all over JSF UI thus)

[Decorator](http://en.wikipedia.org/wiki/Decorator_pattern) (recognizeable by creational methods taking an instance of*same*abstract/interface type which adds additional behaviour)

* All subclasses of [java.io.InputStream](http://docs.oracle.com/javase/6/docs/api/java/io/InputStream.html), [OutputStream](http://docs.oracle.com/javase/6/docs/api/java/io/OutputStream.html), [Reader](http://docs.oracle.com/javase/6/docs/api/java/io/Reader.html) and [Writer](http://docs.oracle.com/javase/6/docs/api/java/io/Writer.html) have a constructor taking an instance of same type.
* [java.util.Collections](http://docs.oracle.com/javase/6/docs/api/java/util/Collections.html), the [checkedXXX()](http://docs.oracle.com/javase/6/docs/api/java/util/Collections.html#checkedCollection%28java.util.Collection,%20java.lang.Class%29), [synchronizedXXX()](http://docs.oracle.com/javase/6/docs/api/java/util/Collections.html#synchronizedCollection%28java.util.Collection%29) and [unmodifiableXXX()](http://docs.oracle.com/javase/6/docs/api/java/util/Collections.html#unmodifiableCollection%28java.util.Collection%29)methods.
* [javax.servlet.http.HttpServletRequestWrapper](http://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpServletRequestWrapper.html) and [HttpServletResponseWrapper](http://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpServletResponseWrapper.html)

[Facade](http://en.wikipedia.org/wiki/Facade_pattern) (recognizeable by behavioral methods which internally uses instances of*different*independent abstract/interface types)

* [javax.faces.context.FacesContext](http://docs.oracle.com/javaee/6/api/javax/faces/context/FacesContext.html), it internally uses among others the abstract/interface types [LifeCycle](http://docs.oracle.com/javaee/6/api/javax/faces/lifecycle/Lifecycle.html), [ViewHandler](http://docs.oracle.com/javaee/6/api/javax/faces/application/ViewHandler.html), [NavigationHandler](http://docs.oracle.com/javaee/6/api/javax/faces/application/NavigationHandler.html) and many more without that the enduser has to worry about it (which are however overrideable by injection).
* [javax.faces.context.ExternalContext](http://docs.oracle.com/javaee/6/api/javax/faces/context/ExternalContext.html), which internally uses [ServletContext](http://docs.oracle.com/javaee/6/api/javax/servlet/ServletContext.html), [HttpSession](http://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpSession.html), [HttpServletRequest](http://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpServletRequest.html), [HttpServletResponse](http://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpServletResponse.html), etc.

[Flyweight](http://en.wikipedia.org/wiki/Flyweight_pattern) (recognizeable by creational methods returning a cached instance, a bit the "multiton" idea)

* [java.lang.Integer#valueOf(int)](http://docs.oracle.com/javase/6/docs/api/java/lang/Integer.html#valueOf%28int%29) (also on [Boolean](http://docs.oracle.com/javase/6/docs/api/java/lang/Boolean.html#valueOf%28boolean%29), [Byte](http://docs.oracle.com/javase/6/docs/api/java/lang/Byte.html#valueOf%28byte%29), [Character](http://docs.oracle.com/javase/6/docs/api/java/lang/Character.html#valueOf%28char%29), [Short](http://docs.oracle.com/javase/6/docs/api/java/lang/Short.html#valueOf%28short%29), [Long](http://docs.oracle.com/javase/6/docs/api/java/lang/Long.html#valueOf%28long%29) and [BigDecimal](https://docs.oracle.com/javase/8/docs/api/java/math/BigDecimal.html" \l "valueOf-long-int-))

[Proxy](http://en.wikipedia.org/wiki/Proxy_pattern) (recognizeable by creational methods which returns an implementation of given abstract/interface type which in turn*delegates/uses*a*different*implementation of given abstract/interface type)

* [java.lang.reflect.Proxy](http://docs.oracle.com/javase/6/docs/api/java/lang/reflect/Proxy.html)
* [java.rmi.\*](http://docs.oracle.com/javase/6/docs/api/java/rmi/package-summary.html), the whole API actually.

The Wikipedia example is IMHO a bit poor, lazy loading has actually completely nothing to do with the proxy pattern at all.

[Behavioral patterns](http://en.wikipedia.org/wiki/Behavioral_pattern)

[Chain of responsibility](http://en.wikipedia.org/wiki/Chain_of_responsibility_pattern) (recognizeable by behavioral methods which (indirectly) invokes the same method in*another*implementation of*same*abstract/interface type in a queue)

* [java.util.logging.Logger#log()](http://docs.oracle.com/javase/6/docs/api/java/util/logging/Logger.html#log%28java.util.logging.Level,%20java.lang.String%29)
* [javax.servlet.Filter#doFilter()](http://docs.oracle.com/javaee/6/api/javax/servlet/Filter.html#doFilter%28javax.servlet.ServletRequest,%20javax.servlet.ServletResponse,%20javax.servlet.FilterChain%29)

[Command](http://en.wikipedia.org/wiki/Command_pattern) (recognizeable by behavioral methods in an abstract/interface type which invokes a method in an implementation of a*different*abstract/interface type which has been*encapsulated*by the command implementation during its creation)

* All implementations of [java.lang.Runnable](http://docs.oracle.com/javase/6/docs/api/java/lang/Runnable.html)
* All implementations of [javax.swing.Action](http://docs.oracle.com/javase/6/docs/api/javax/swing/Action.html)

[Interpreter](http://en.wikipedia.org/wiki/Interpreter_pattern) (recognizeable by behavioral methods returning a*structurally*different instance/type of the given instance/type; note that parsing/formatting is not part of the pattern, determining the pattern and how to apply it is)

* [java.util.Pattern](http://docs.oracle.com/javase/6/docs/api/java/util/regex/Pattern.html)
* [java.text.Normalizer](http://docs.oracle.com/javase/6/docs/api/java/text/Normalizer.html)
* All subclasses of [java.text.Format](http://docs.oracle.com/javase/6/docs/api/java/text/Format.html)
* All subclasses of [javax.el.ELResolver](http://docs.oracle.com/javaee/6/api/javax/el/ELResolver.html)

[Iterator](http://en.wikipedia.org/wiki/Iterator_pattern) (recognizeable by behavioral methods sequentially returning instances of a*different*type from a queue)

* All implementations of [java.util.Iterator](http://docs.oracle.com/javase/6/docs/api/java/util/Iterator.html) (thus among others also [java.util.Scanner](http://docs.oracle.com/javase/6/docs/api/java/util/Scanner.html)!).
* All implementations of [java.util.Enumeration](http://docs.oracle.com/javase/6/docs/api/java/util/Enumeration.html)

[Mediator](http://en.wikipedia.org/wiki/Mediator_pattern) (recognizeable by behavioral methods taking an instance of different abstract/interface type (usually using the command pattern) which delegates/uses the given instance)

* [java.util.Timer](http://docs.oracle.com/javase/6/docs/api/java/util/Timer.html) (all scheduleXXX() methods)
* [java.util.concurrent.Executor#execute()](http://docs.oracle.com/javase/6/docs/api/java/util/concurrent/Executor.html#execute%28java.lang.Runnable%29)
* [java.util.concurrent.ExecutorService](http://docs.oracle.com/javase/6/docs/api/java/util/concurrent/ExecutorService.html) (the invokeXXX() and submit() methods)
* [java.util.concurrent.ScheduledExecutorService](http://docs.oracle.com/javase/6/docs/api/java/util/concurrent/ScheduledExecutorService.html) (all scheduleXXX() methods)
* [java.lang.reflect.Method#invoke()](http://docs.oracle.com/javase/6/docs/api/java/lang/reflect/Method.html#invoke%28java.lang.Object,%20java.lang.Object...%29)

[Memento](http://en.wikipedia.org/wiki/Memento_pattern) (recognizeable by behavioral methods which internally changes the state of the*whole*instance)

* [java.util.Date](http://docs.oracle.com/javase/6/docs/api/java/util/Date.html) (the setter methods do that, Date is internally represented by a long value)
* All implementations of [java.io.Serializable](http://docs.oracle.com/javase/6/docs/api/java/io/Serializable.html)
* All implementations of [javax.faces.component.StateHolder](http://docs.oracle.com/javaee/6/api/javax/faces/component/StateHolder.html)

[Observer (or Publish/Subscribe)](http://en.wikipedia.org/wiki/Observer_pattern) (recognizeable by behavioral methods which invokes a method on an instance of*another*abstract/interface type, depending on own state)

* [java.util.Observer](http://docs.oracle.com/javase/6/docs/api/java/util/Observer.html)/[java.util.Observable](http://docs.oracle.com/javase/6/docs/api/java/util/Observable.html) (rarely used in real world though)
* All implementations of [java.util.EventListener](http://docs.oracle.com/javase/6/docs/api/java/util/EventListener.html) (practically all over Swing thus)
* [javax.servlet.http.HttpSessionBindingListener](http://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpSessionBindingListener.html)
* [javax.servlet.http.HttpSessionAttributeListener](http://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpSessionAttributeListener.html)
* [javax.faces.event.PhaseListener](http://docs.oracle.com/javaee/6/api/javax/faces/event/PhaseListener.html)

[State](http://en.wikipedia.org/wiki/State_pattern) (recognizeable by behavioral methods which changes its behaviour depending on the instance's state which can be controlled externally)

* [javax.faces.lifecycle.LifeCycle#execute()](http://docs.oracle.com/javaee/6/api/javax/faces/lifecycle/Lifecycle.html#execute%28javax.faces.context.FacesContext%29) (controlled by [FacesServlet](http://docs.oracle.com/javaee/6/api/javax/faces/webapp/FacesServlet.html), the behaviour is dependent on current phase (state) of JSF lifecycle)

[Strategy](http://en.wikipedia.org/wiki/Strategy_pattern) (recognizeable by behavioral methods in an abstract/interface type which invokes a method in an implementation of a*different*abstract/interface type which has been*passed-in*as method argument into the strategy implementation)

* [java.util.Comparator#compare()](http://docs.oracle.com/javase/6/docs/api/java/util/Comparator.html#compare%28T,%20T%29), executed by among others Collections#sort().
* [javax.servlet.http.HttpServlet](http://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpServlet.html), the service() and all doXXX() methods take HttpServletRequest and HttpServletResponse and the implementor has to process them (and not to get hold of them as instance variables!).
* [javax.servlet.Filter#doFilter()](http://docs.oracle.com/javaee/6/api/javax/servlet/Filter.html#doFilter%28javax.servlet.ServletRequest,%20javax.servlet.ServletResponse,%20javax.servlet.FilterChain%29)

[Template method](http://en.wikipedia.org/wiki/Template_method_pattern) (recognizeable by behavioral methods which already have a "default" behaviour definied by an abstract type)

* All non-abstract methods of [java.io.InputStream](http://docs.oracle.com/javase/6/docs/api/java/io/InputStream.html), [java.io.OutputStream](http://docs.oracle.com/javase/6/docs/api/java/io/OutputStream.html), [java.io.Reader](http://docs.oracle.com/javase/6/docs/api/java/io/Reader.html)and [java.io.Writer](http://docs.oracle.com/javase/6/docs/api/java/io/Writer.html).
* All non-abstract methods of [java.util.AbstractList](http://docs.oracle.com/javase/6/docs/api/java/util/AbstractList.html), [java.util.AbstractSet](http://docs.oracle.com/javase/6/docs/api/java/util/AbstractSet.html) and [java.util.AbstractMap](http://docs.oracle.com/javase/6/docs/api/java/util/AbstractMap.html).
* [javax.servlet.http.HttpServlet](http://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpServlet.html), all the doXXX() methods by default sends a HTTP 405 "Method Not Allowed" error to the response. You're free to implement none or any of them.

[Visitor](http://en.wikipedia.org/wiki/Visitor_pattern) (recognizeable by two*different*abstract/interface types which has methods definied which takes each the*other*abstract/interface type; the one actually calls the method of the other and the other executes the desired strategy on it)

* [javax.lang.model.element.AnnotationValue](http://docs.oracle.com/javase/6/docs/api/javax/lang/model/element/AnnotationValue.html) and [AnnotationValueVisitor](http://docs.oracle.com/javase/6/docs/api/javax/lang/model/element/AnnotationValueVisitor.html)
* [javax.lang.model.element.Element](http://docs.oracle.com/javase/6/docs/api/javax/lang/model/element/Element.html) and [ElementVisitor](http://docs.oracle.com/javase/6/docs/api/javax/lang/model/element/ElementVisitor.html)
* [javax.lang.model.type.TypeMirror](http://docs.oracle.com/javase/6/docs/api/javax/lang/model/type/TypeMirror.html) and [TypeVisitor](http://docs.oracle.com/javase/6/docs/api/javax/lang/model/type/TypeVisitor.html)
* [java.nio.file.FileVisitor](http://docs.oracle.com/javase/7/docs/api/java/nio/file/FileVisitor.html) and [SimpleFileVisitor](http://docs.oracle.com/javase/7/docs/api/java/nio/file/SimpleFileVisitor.html)

|  |
| --- |
| [share](http://stackoverflow.com/a/2707195) |