1. **Creational patterns**
   * Abstract factory

Recognizable by creational methods returning the factory itself which in turn can be used to create another abstract/interface type

* + - javax.xml.parsers.DocumentBuilderFactory#newInstance()
    - javax.xml.transform.TransformerFactory#newInstance()
    - javax.xml.xpath.XPathFactory#newInstance()
  + Builder

Recognizable by creational methods returning the instance itself

* + - java.lang.StringBuilder#append() (unsynchronized)
    - java.lang.StringBuffer#append() (synchronized)
    - java.nio.ByteBuffer#put() (also on CharBuffer, ShortBuffer, IntBuffer, LongBuffer, FloatBuffer and DoubleBuffer)
    - javax.swing.GroupLayout.Group#addComponent()
    - All implementations of java.lang.Appendable
  + Factory method

Recognizable by creational methods returning an implementation of an abstract/interface type

* + - java.util.Calendar#getInstance()
    - java.util.ResourceBundle#getBundle()
    - java.text.NumberFormat#getInstance()
    - java.nio.charset.Charset#forName()
    - java.net.URLStreamHandlerFactory#createURLStreamHandler(String) (Returns singleton object per protocol)
    - java.util.EnumSet#of()
    - javax.xml.bind.JAXBContext#createMarshaller() and other similar methods
  + Prototype

Recognizable by creational methods returning a different instance of itself with the same properties

* + - java.lang.Object#clone() (the class has to implement java.lang.Cloneable)
  + Singleton

Recognizable by creational methods returning the same instance (usually of itself) everytime

* + - java.lang.Runtime#getRuntime()
    - java.awt.Desktop#getDesktop()
    - java.lang.System#getSecurityManager()

1. **Structural** patterns
   * Adapter

Recognizable 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()
    - java.util.Collections#list()
    - java.util.Collections#enumeration()
    - java.io.InputStreamReader(InputStream) (returns a Reader)
    - java.io.OutputStreamWriter(OutputStream) (returns a Writer)
    - javax.xml.bind.annotation.adapters.XmlAdapter#marshal() and #unmarshal()
  + Bridge

Recognizable 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

* + - LinkedHashMap(LinkedHashSet<K>, List<V>) which returns an unmodifiable linked map which doesn't clone the items, but uses them.
    - java.util.Collections#newSetFromMap() and singletonXXX() methods however comes close.
  + Composite

Recognizable by behavioral methods taking an instance of same abstract/interface type into a tree structure

* + - java.awt.Container#add(Component) (practically all over Swing thus)
    - javax.faces.component.UIComponent#getChildren() (practically all over JSF UI thus)
  + Decorator

Recognizable by creational methods taking an instance of same abstract/interface type which adds additional behavior

* + - All subclasses of java.io.InputStream, OutputStream, Reader and Writer have a constructor taking an instance of same type.
    - java.util.Collections, the checkedXXX(), synchronizedXXX() and unmodifiableXXX() methods.
    - javax.servlet.http.HttpServletRequestWrapper and HttpServletResponseWrapper
  + Facade

Recognizable by behavioral methods which internally uses instances of different independent abstract/interface types

* + - javax.faces.context.FacesContext, it internally uses among others the abstract/interface types LifeCycle, ViewHandler, NavigationHandler and many more without that the enduser has to worry about it (which are however override able by injection).
    - javax.faces.context.ExternalContext, which internally uses ServletContext, HttpSession, HttpServletRequest, HttpServletResponse, etc.
  + Flyweight

Recognizable by creational methods returning a cached instance, a bit the "multiton" idea

* + - java.lang.Integer#valueOf(int) (also on Boolean, Byte, Character, Short, Long and BigDecimal)
  + Proxy

Recognizable 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
    - java.rmi.\*
    - javax.ejb.EJB (explanation here)
    - javax.inject.Inject (explanation here)
    - javax.persistence.PersistenceContext

1. **Behavioral patterns**
   * Chain of responsibility

Recognizable 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()
    - javax.servlet.Filter#doFilter()
  + Command

Recognizable 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
    - All implementations of javax.swing.Action
  + Interpreter

Recognizable 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
    - java.text.Normalizer
    - All subclasses of java.text.Format
    - All subclasses of javax.el.ELResolver
  + Iterator

Recognizable by behavioral methods sequentially returning instances of a different type from a queue

* + - All implementations of java.util.Iterator (thus among others also java.util.Scanner!).
    - All implementations of java.util.Enumeration
  + Mediator

Recognizable 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 (all scheduleXXX() methods)
    - java.util.concurrent.Executor#execute()
    - java.util.concurrent.ExecutorService (the invokeXXX() and submit() methods)
    - java.util.concurrent.ScheduledExecutorService (all scheduleXXX() methods)
    - java.lang.reflect.Method#invoke()
  + Memento

Recognizable by behavioral methods which internally changes the state of the whole instance

* + - java.util.Date (the setter methods do that, Date is internally represented by a long value)
    - All implementations of java.io.Serializable
    - All implementations of javax.faces.component.StateHolder
  + Observer (or Publish/Subscribe)

Recognizable by behavioral methods which invokes a method on an instance of another abstract/interface type, depending on own state

* + - java.util.Observer/java.util.Observable (rarely used in real world though)
    - All implementations of java.util.EventListener (practically all over Swing thus)
    - javax.servlet.http.HttpSessionBindingListener
    - javax.servlet.http.HttpSessionAttributeListener
    - javax.faces.event.PhaseListener
  + State

Recognizable by behavioral methods which changes its behavior depending on the instance's state which can be controlled externally

* + - javax.faces.lifecycle.LifeCycle#execute() (controlled by FacesServlet, the behaviour is dependent on current phase (state) of JSF lifecycle)
  + Strategy

Recognizable 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(), executed by among others Collections#sort().
    - javax.servlet.http.HttpServlet, the service() and all doXXX() methods take HttpServletRequest and HttpServletResponse and the implementer has to process them (and not to get hold of them as instance variables!).
    - javax.servlet.Filter#doFilter()
  + Template method

Recognizable by behavioral methods which already have a "default" behavior defined by an abstract type

* + - All non-abstract methods of java.io.InputStream, java.io.OutputStream, java.io.Reader and java.io.Writer.
    - All non-abstract methods of java.util.AbstractList, java.util.AbstractSet and java.util.AbstractMap.
    - javax.servlet.http.HttpServlet, 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

Recognizable 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 and AnnotationValueVisitor
    - javax.lang.model.element.Element and ElementVisitor
    - javax.lang.model.type.TypeMirror and TypeVisitor
    - java.nio.file.FileVisitor and SimpleFileVisitor
    - javax.faces.component.visit.VisitContext and VisitCallback