References

Design Patterns: Elements of Reusable Object-Oriented Software, Gamma, Helm, Johnson, Vlissides, 1995, pp. 117-126

The Design Patterns Smalltalk Companion, Alpert, Brown, Woolf, Addison-Wesley, 1998, pp. 77-90

Java API

Prototype-based Languages
   http://en.wikipedia.org/wiki/Prototype-based_programming
   JavaScript The Definite Guide 4'th Ed, Flanagan, O'Reilly Press, 2002
Prototype

Specify the kinds of objects to create using a prototypical instance, and create new objects by copying this prototype

Applicability

Use the Prototype pattern when

A system should be independent of how its products are created, composed, and represented; and

When the classes to instantiate are specified at run-time; or

To avoid building a class hierarchy of factories that parallels the class hierarchy of products; or

When instances of a class can have one of only a few different combinations of state.
Insurance Example

Insurance agents start with a standard policy and customize it

Two basic strategies:

Copy the original and edit the copy

Store only the differences between original and the customize version in a decorator
Copying Issues

Shallow Copy Verse Deep Copy

Original Objects

Shallow Copy

Deep Copy
Shallow Copy Verse Deep Copy

Original Objects

Deep Copy

Deeper Copy
class Door {
    public:
        Door();
        Door( const Door&);
        virtual Door* clone() const;

        virtual void Initialize( Room*, Room* );
        // stuff not shown
    private:
        Room* room1;
        Room* room2;
    }

Door::Door ( const Door& other ) //Copy constructor {
    room1 = other.room1;
    room2 = other.room2;
}

Door* Door::clone() const {
    return new Door( *this );
}
Cloning Issues - Java Clone

Shallow Copy

class Door implements Cloneable {
    private Room room1;
    private Room room2;

    public Object clone() throws CloneNotSupportedException {
        return super.clone();
    }
}

Deep Copy

public class Door implements Cloneable {
    private Room room1;
    private Room room2;

    public Object clone() throws CloneNotSupportedException {
        Door thisCloned =(Door) super.clone();
        thisCloned.room1 = (Room)room1.clone();
        thisCloned.room2 = (Room)room2.clone();
        return thisCloned;
    }
}
Prototype-based Languages

No classes

Behaviour reuse (inheritance)
   Cloning existing objects which serve as prototypes

Some Prototype-based languages

  Self
  JavaScript
  Squeak (eToys)
  Perl with Class::Prototyped module
JavaScript Example

Circle.prototype.pi = 3.14159;

function Circle_circumference() {
    return 2 * this.pi * this.r;
}
Circle.prototype.circumference = Circle_circumference;

function Circle(x, y, r) {
    this.x = x;
    this.y = y;
    this.r = r
}

var center = new Circle(0.0, 0.0, 1.0);
print(center.circumference());

Circle.prototype.area = function() { return this.pi * this.r * this.r; }
print(center.area());

center.color = "red";