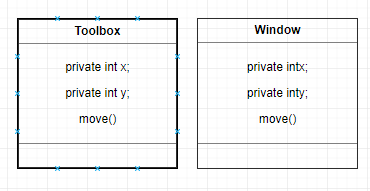
Motivation:

In this example the Toolbox and the Window class both have the same move method. In this case, it would be logical to put this method as well as the x and y properties into a new superclass that Toolbox and Window extend.

**Before Diagram:**



**Before:**

class Toolbox

{

private int x;

private int y;

public void move(int x, int y)

{

this.x = x;

this.y = y;

}

// ...

}

class Window

{

private int x;

private int y;

public void move(int x, int y)

{

this.x = x;

this.y = y;

}

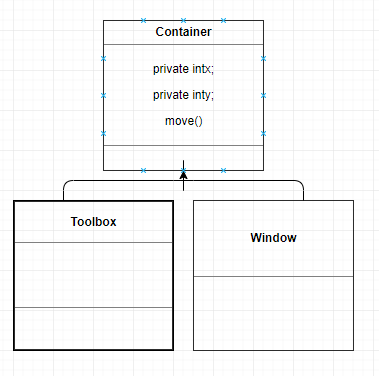
// ...

}

**Mechanics:**

The example could be easily fixed by moving the x and y properties as well as the move method to a parent class. Note that the x and y properties must be changed to protected if they are accessed from the Toolbox and Window classes.

**After Diagram:**



**After:**

class Container

{

protected int x;

protected int y;

public void move(int x, int y)

{

this.x = x;

this.y = y;

}

}

class Toolbox : Container

{

// ...

}

class Window : Container

{

// ...

}