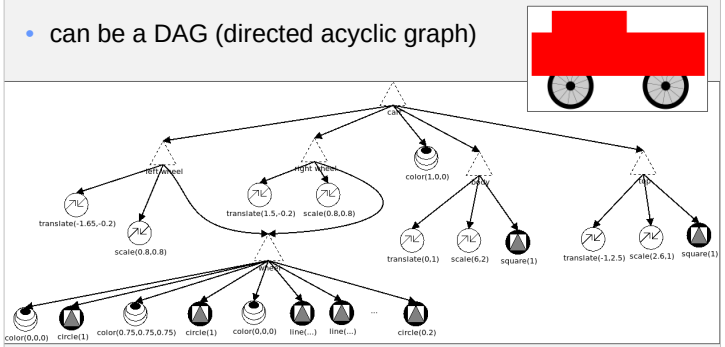
Hierarchical Scene Descriptions

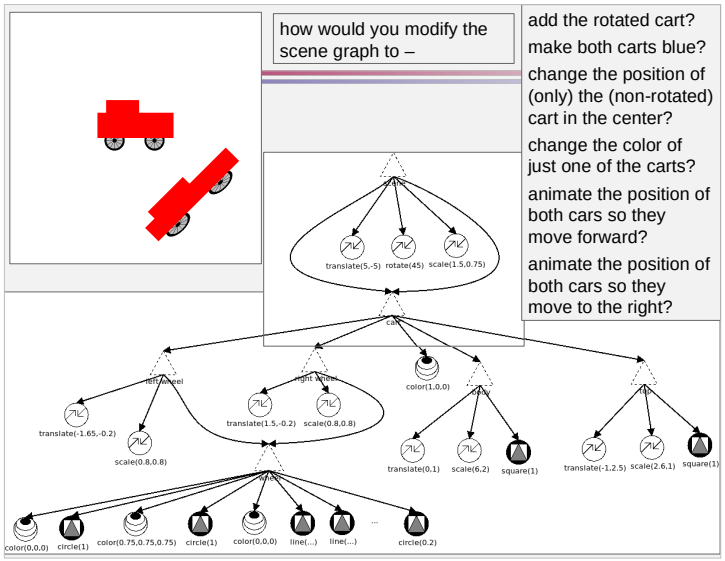
- can be a DAG (directed acyclic graph)



- preorder traversal
  - push current transform at each compound object node
  - visit children left to right
  - pop when done with subtree
- current transform, drawing properties stay in effect until changed, regardless of the level in the tree

how would you modify the scene graph to –

- add the rotated cart?
- make both carts blue?
- change the position of (only) the (non-rotated) cart in the center?
- change the color of just one of the carts?
- animate the position of both cars so they move forward?
- animate the position of both cars so they move to the right?



Scene Graphs

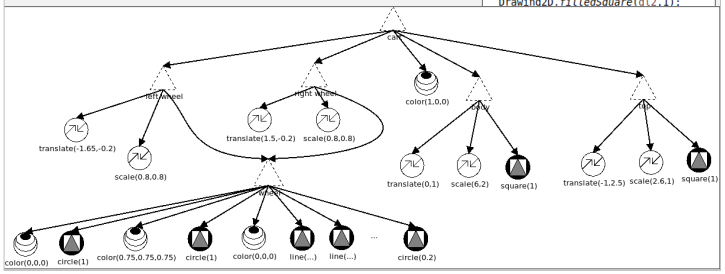
- can be *implicit* through method calls and the program call stack
- can be *explicit* with an actual data structure

```
private void drawCart ( GL2 gl2 ) {
  // left wheel
  gl2.glPushMatrix();
  gl2.glTranslatef(-1.65f,-0.2f,0f);
  gl2.glScalef(0.8f,0.8f,1f);
  drawWheel(gl2);
  gl2.glPopMatrix();

  // right wheel
  gl2.glPushMatrix();
  gl2.glTranslatef(1.5f,-0.2f,0f);
  gl2.glScalef(0.8f,0.8f,1f);
  drawWheel(gl2);
  gl2.glPopMatrix();

  gl2.glColor3f(1f,0f,0f);

  // body of cart
  gl2.glPushMatrix();
  gl2.glTranslatef(0f,1f,0f);
  gl2.glScalef(6f,2f,1f);
  Drawing2D.filledSquare(gl2,1);
```



Scene Graphs

- having an explicit representation (data structure) for scene graphs is useful
  - often more user-friendly to construct the scene graph than issue graphics library commands
  - scene graph API can be separate from a particular graphics library
  - allows for general-purpose modeling and viewing programs which can read and write scene graphs from files