Interface segregation principle

A client should never be forced to implement an interface that it doesn't use or clients shouldn't be forced to depend on methods they do not use.

e.g.

```
interface MouseAllListener
{
  void mouseClicked(MouseEvent e);
  void mouseEntered(MouseEvent e);
  void mouseExited(MouseEvent e);
  void mousePressed(MouseEvent e);
  void mouseReleased(MouseEvent e);
  void mouseDragged(MouseEvent e);
  void mouseMoved(MouseEvent e);
}
```

A programmer would like to handle events related to mouse, must implement the above interface and define all its methods.
However, if a programmer is not interested in drag or move events why he should implement those methods?
On the other hand, if a programmer wants to handle only drag or move events for him other methods don't make any sense.

Solution:

```
interface MouseListener
{
  void mouseClicked(MouseEvent e);
  void mouseEntered(MouseEvent e);
  void mouseExited(MouseEvent e);
  void mousePressed(MouseEvent e);
  void mouseReleased(MouseEvent e);
 }


  interface MouseMotionListener
  {
  void mouseDragged(MouseEvent e);
  void mouseMoved(MouseEvent e);
  }
```