```
package rolodex;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

/****
 *
 * Class OKDeleteButtonListener is attached to the OK button in the delete card
 * dialog.  It is an example of referential transparency of design.  See the
 * documentation for the <a href="OKAddButtonListener.html">
 * OKAddButtonListener </a> for discussion of the concept of referential
 * transparency in an object-oriented design.

 *

 */

public class OKDeleteButtonListener implements ActionListener {

    /**
     * Construct this with the given rolodex model and parent dialog view and
     * grandparent rolodex view.  Access to the model is for calling its delete
     * function.  Access to the parent dialog is for gathering data to be sent
     * to Rolodex.delete.  Access to the grandparent view is for reporting
     * input error messages.  See the specs of actionPerformed below for
     * further discussion.
     */
    public OKDeleteButtonListener(Rolodex r, DeleteCardDialog dcd,
            RolodexMenuUI rmui) {
        this.r = r;
        this.dcd = dcd;
        this.rmui = rmui;
    }

    /**
     * Respond to a press of the OK button by calling Rolodex.delete.  The card
     * id sent as the parameter to delete is gathered from the JTextField in
     * the parent delete dialog.
     */
    public void actionPerformed(ActionEvent e) {

        try {
            r.delete(new Id(dcd.getId()));
        }
        catch (DeleteInputError die) {
            rmui.showErrorDialog(die.getNotThereErrorMessage());
        }

        /*
         * Notify the Rolodex observers if it has changed.  Per the specs of
         * the Java Observable class, notifyObservers only performs the
         * notification if the observed object has actually changed.
         */
        r.notifyObservers();

    }

    /** The companion model */
    protected Rolodex r;

    /** The parent view */
    protected DeleteCardDialog dcd;

    /** The grandparent view */
    protected RolodexMenuUI rmui;

}
```