1 - How listener model is used.
- Background:
  - the app (demo)
  - how widgets are created and added
  - how backend calls are made

• demo transcript:
http://precog.uoregon.edu:8080/action/marker/marker-edit?zdbID=ZDB-TSCRIPT-081120-10004#proteinLookup

• attributiontable, notebox, etc, used in multiple places in one page

WIDGET CREATION:

```
private NoteListBox curatorNoteBox = new NoteListBox();
RootPanel.get(curatorNoteDiv).add(curatorNoteBox);   // adds to div
```

Widget created in TranscriptEditController

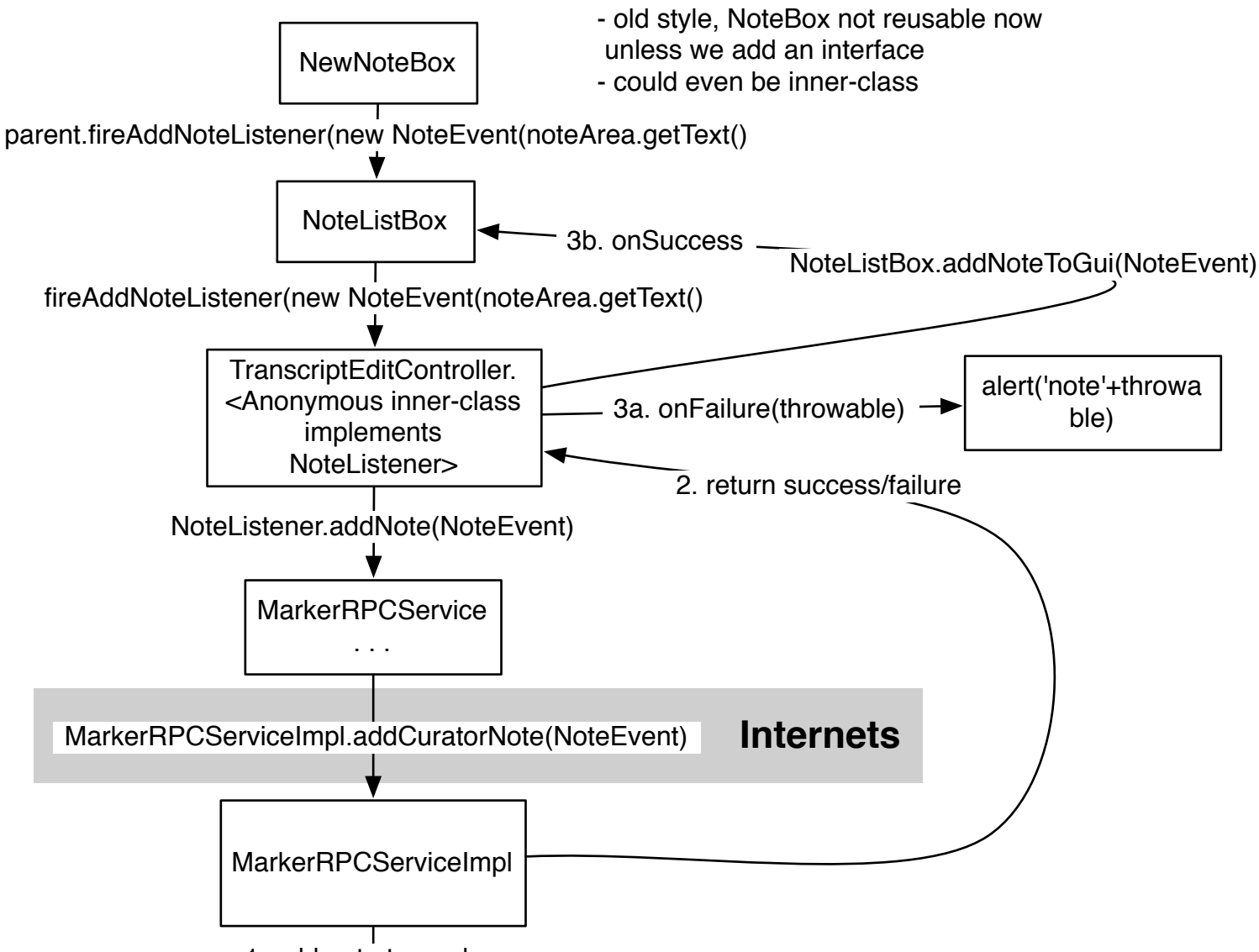
```
public class NoteListBox extends Composite {

    private List<NoteListener> noteListeners = new ArrayList<NoteListener>() ;

    public NoteListBox(boolean multipleNotes){
        this.multipleNotes = multipleNotes;
        initGui() ;
        initWidget(panel);
    }
    public void initGui(){
        panel.add(noteTable);
        addNewNoteField.add(addButton) ;
        panel.add(addNewNoteField);

        addButton.addClickListener(new ClickListener(){
            public void onClick(Widget widget) {
                if(newNoteBox ==null){
                    newNoteBox = new NewNoteBox(getNoteListBoxInstance()) ;
                }
                else{
                    newNoteBox.reset() ;
                    newNoteBox.checkButtonStatus();
                    newNoteBox.show();
                }
            }
        });
    }
```

Anonymous inner-class - button has addClickListenerMethod



Behavior set for TranscriptEditController

```
curatorNoteBox.addNoteListener(new NoteListener(){

    public void addNote(final NoteEvent noteEvent) {
        MarkerRPCService.App.getInstance().addCuratorNote(zdbID,noteEvent.getNoteData(),curatorID,new AsyncCallback(){
            public void onFailure(Throwable throwable) {
                Window.alert("failed to add note: "+ throwable);
            }

            public void onSuccess(Object o) {
                curatorNoteBox.addNoteToGui(noteEvent);
            }
        });
        ...
```

Note added to GUI on success

Adding a listener for our custom class

Make back-end RPC call (later . . .)

```
public void addNoteToGui(NoteEvent noteEvent){
    NoteLine noteLine = new
NoteLine( noteEvent.getNoteData() , this) ;
    int rowCount = noteTable.getRowCount() ;
    noteTable.insertRow( rowCount );
    noteTable.setWidget(rowCount,0,noteLine);

    if(noteTable.getRowCount()>0){
        addButton.setVisible(multipleNotes);
    }
}
```

NoteListBox

```
private List<NoteListener> noteListeners = new ArrayList<NoteListener>() ;
public void addNoteListener(NoteListener noteListener){
    noteListeners.add(noteListener) ;
}
public void fireAddNoteListener(NoteEvent noteEvent){
    for(NoteListener noteListener: noteListeners){
        noteListener.addNote(noteEvent);
    }
}
```

listeners are added and fired

NoteListener - just an interface

```
public interface NoteListener {
    public void addNote(final NoteEvent noteEvent) ;
```

RPC call in MarkerRPCServiceCall - errors are propagated back to the gUI

```
public void addCuratorNote(String markerZdbID, String note, String curatorZdbID) {
    logger.info("adding curator note: "+ markerZdbID + " - "+ note);
    Session session = HibernateUtil.currentSession() ;
    Transaction transaction = session.beginTransaction() ;
    Marker marker = markerRepository.getMarkerByID(markerZdbID) ;
    Person person= profileRepository.getPerson(curatorZdbID) ;
    markerRepository.addMarkerDataNote(marker, note,person);
    transaction.commit();
}
```