SWING:
CLASSES AND BUILDING BLOCKS - II

MENUS NEED SOME BOILER-PLATE, BUT ARE PRETTY INTUITIVE

TREE-VIEWS ARE SURPRISINGLY COMPLICATED

FILE CHOOSERS AND SCROLL BARS ARE SURPRISINGLY SIMPLE

USING MENUS

```
JMenuBar menuBar = new JMenuBar();

JMenu menu = new JMenu("File");
menu.setMnemonic(KeyEvent.VK_F);
menuBar.add(menu);

JMenuItem menuItem = new JMenuItem("Save");
menu.setMnemonic(KeyEvent.VK_S);
menu.add(menuItem);

menuItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        final JFileChooser fc = new JFileChooser();

        if (fc.showSaveDialog(editorPane) == JFileChooser.APPROVE_OPTION) {
            System.out.println(fc.getSelectedFile().getAbsolutePath());
            HTMLWriter.writeToHTML(fc.getSelectedFile().getAbsolutePath(), mapOfSummaries);
        }
    }
});

frame.setJMenuBar(menuBar);
```

CREATE A MENUBAR
CREATE MENU ITEMS AND ADD TO THE MENU BAR
(OPTIONALLY, SET KEYBOARD SHORTCUTS)

ASSIGN ACTIONS TO BE EXECUTED WHEN THOSE MENU ITEMS ARE SELECTED

SET THE MENUBAR FOR THIS FRAME

USE THE JFileChooser OBJECT FOR FILE CHOICES

```
final JFileChooser fc = new JFileChooser();

if (fc.showSaveDialog(editorPane) == JFileChooser.APPROVE_OPTION) {
    System.out.println(fc.getSelectedFile().getAbsolutePath() +
            " was what the user chose");
```

ALL OF THESE ARE ACCOMPLISHED IN THESE 2 LINES OF CODE

IF YOU THINK ABOUT IT, A FILE CHOOSER NEEDS TO CONVEY TWO BITS OF INFORMATION

1. WHETHER THE USER CANCELLED OUT OF THE FILE CHOOSE OPERATION OR NOT
2. IF THE USER DID NOT CANCEL OUT, THE FILE THAT WAS CHOSEN TO BE SAVED (OR OPENED)

FILE CHOOSERS TEND TO EITHER TO

OPEN OR TO **SAVE**

JFILECHOOSER.SHOWOPENDIALOG

JFILECHOOSER.SHOWSAVEDIALOG

THE JScrollPane CLASS IS PRETTY INTUITIVE TOO

```
final JTextArea summaryField = new JTextArea();
summaryField.setLineWrap(true);
JScrollPane summaryScrollPane = new JScrollPane(summaryField);
summaryScrollPane.setPreferredSize(new Dimension(600, 150));
JPanel summaryPanel = new JPanel();
summaryPanel.add(summaryScrollPane);
```

INSTANTIATE THE JSCROLLPANE OBJECT

SPECIFY HOW BROAD AND HOW WIDE YOU'D LIKE IT TO BE

ADD IT TO THE JPANEL INSIDE WHICH IT WILL BELONG

OPTIONALLY, YOU CAN ALSO SET "SCROLLBAR POLICIES", FOR INSTANCE WHETHER THE SCROLLBARS SHOULD ALWAYS APPEAR, OR ONLY WHEN NEEDED

TREES ARE RATHER COMPLICATED

JTree IS THE MAIN UI ELEMENT CLASS

BUT A HOST OF ASSOCIATED CLASSES NEED TO BE USED IN ADDITION

TREENODE OBJECTS, WHICH ARE CONTAINED INSIDE THE TREE

TREEMODEL OBJECTS, WHICH REPRESENT THE MODEL FOR WHICH THE JTREE IS THE VIEW

A TREEMODEL IS EXPLICITLY NEEDED FOR OPERATIONS SUCH AS UPDATING THE TREE WHEN A NEW NODE IS ADDED

TreeModelListeners

USE TREEMODELLISTENERS WHEN THE TREE DATA CHANGES AND THE TREE UI MUST BE UPDATED

AS THEIR NAME WOULD SUGGEST, "TREE-MODEL-LISTENERS" LISTEN FOR CHANGES IN THE MODEL (I.E. THE UNDERLYING DATA) OF A TREE

TreeSelectionListeners

USE TREESELECTIONLISTENERS WHEN THE USER HAS SELECTED SOME NODES IN A TREE, AND SOME OTHER UI ELEMENTS NEED TO BE UPDATED TO STAY IN SYNCH

ON THE OTHER HAND "TREE-SELECTION-LISTENERS" LISTEN FOR CHANGES IN THE VIEW (UI REPRESENTATION) OF THE TREE

CAPTURING RIGHT-CLICKS OF A MOUSE

EVERY JComponent ALLOWS PROGRAMMERS TO LISTEN IN ON MOUSE EVENTS

YOU CAN IMPLEMENT THE mouseadapter INTERFACE

AND LISTEN IN VIA THE addmouselistener METHOD, WHICH IS PRESENT IN EVERY UI ELEMENT

WITHIN THESE METHODS, IT IS POSSIBLE TO EXAMINE IF THE MOUSE EVENT WAS A POPUP TRIGGER (I.E. A RIGHT-CLICK)