Grouping Data

3-19-2012

Opening Discussion

- Minute essay comments:
  - Do I watch Battlestar Galactica?
  - Recovering from midterms.
  - Posting IcP solutions.
  - Confusing wording in the book, my fault or editors?
- World of abundance, too many people not doing anything.
- Possible format changes.

User Defined Types

- The general way we do this is to define our own types.
- For now we will just use these types to collect data together in a case class.
- This allows us to give meaning to the data with meaningful names.
  - case class TypeName(field1:Type1,field2:Type2,...)
- Example:
  - case class NBAPlayer(name:String, team:String, mins:List[Int], points:List[Int], rebounds:List[Int])

Instantiation

- Once we have defined a case class we can make new objects of that type in one of two ways.
  - TypeName(expr1,expr2,...)
  - Or
  - new TypeName(expr1,expr2,...)
- Example:
  - var td=NBAPlayer(“Tim Duncan”, ”Spurs”, List(23,13), List(15,17), List(6,8))

Usage

- We can pull out values using dot notation and the name of the field.
- Example:
  - println(td.name+” averages “+td.points.sum/td.points.length+” ppg”)
- The fields are vals so you can't change what they reference. You can only change their values if they are mutable.

Copy Method

- There is a copy method on case classes that does what the name implies.
- It can take named arguments to change certain fields in the copy.
- Example:
  - td=td.copy(mins=17::td.mins, points=15::td.points, rebounds=4::td.rebounds)

Putting it Together

- In the book I am building an example application of a grade book. I'd like to do something different in class so you see variety.
- Do you have any suggestions or do you want me to come up with something? (CPI handling, player stats, ...)

Motivation

- While text based programs still play a very big role in computing, it is mostly behind the scenes.
- You are far more used to working with Graphical User Interfaces (GUIs).
- It is time that we learn how to write GUIs in Scala.

Libraries

- There are three libraries that will wind up being relevant to our discussion.
  - java.awt – The Abstract Windowing Toolkit. Original Java GUI library.
  - javax.swing – Swing was built on top of AWT to be more flexible.
  - scala.swing – Scala code wrapped around Java Swing to aid Scala GUI programming.

Making a Window

- In order to write a GUI we need to start by popping up a window.
- For the main window of a GUI, we will make a MainFrame. For other windows there are Frame and Dialog types.
- We can set the title and size fields of the MainFrame when we create it.
- Set visible to true to bring up the window.
- Oddly, we have to prevent the script from stopping.

Active Components

- GUIs are made from components. Use scala.swing package.
  - Button(text:String)(action : => Unit).
  - new CheckBox(label:String)
    - selected:Boolean
  - new ComboBox(items:Seq[A])
    - selection.index to get the index of the current selection
  - new EditorPane(contentType:String,text:String)

More Components

- new FormattedTextField(format:String)
  - text:String that will tell you the text
- new Label(text:String)
- new ListView(items:Seq[A])
  - Use collection selection.indices to interact with the index values that are selected.
- new PasswordField or new PasswordField(text:String)
  - text:String will tell you the text

More Components

- new ProgressBar
  - min:Int, max:Int, and value:Int
- new RadioButton(text:String)
  - selected:Boolean
- new ScrollBar
  - minimum, maximum, and value are all Ints
  - Generally use ScrollPane
- new Slider
  - min, max, value
  - orientation

Still More Components

- new Table(rowData: Array[Array[Any]], columnNames: Seq[Any])
- new TextArea(text:String)
  - text:String
- new TextField(text:String)
  - text:String

Panes and Panels

- We build complex GUIs by nesting panels and panes.
  - BorderPanel
    - Can hold up to five different components in the north, south, east, west, and center positions. Add to the layout as a tuple of (Component, Position).
  - BoxPanel
    - Can hold a number of components either vertically or horizontally, each takes the space it needs. Use new BoxPanel(Orientation.Vertical). Use contents+=Button(“text”)(action).

More Panels

- FlowPanel
  - Components are laid out from left to right wrapping like text in a word processor. You can pass a variable length list of components as an argument at construction or add the components to contents.
- GridBagPanel
  - This panel is more complex.
- GridPanel
  - Holds a regular grid of components. You specify how many rows and columns the grid has at creation.

Panes

- ScrollPane
  - Holds a single component passed in as an argument at construction. Scroll bars automatic.
- SplitPane
  - Two components separated by a moveable bar.
  - new SplitPane(Orientation.Horizontal, leftComp,rightComp)
- TabbedPane
  - One component shown at a time. Tabs are always shown. Add components by adding Pages to the page object.
  - pages += new Page(“A Tab”,tabComponent)

Menus

- Windows can set the MenuBar.
- Add Menu objects to the contents of the MenuBar.
- Add MenuItems to the contents of the Menus.
  - new MenuItem(Action(“Exit”){ exit(0) })

Example GUI

- Let's spend the rest of class laying out and coding up a GUI for our data example.

Minute Essay

- Is there some type of GUI you would like to have as IcP #6.
- Assignment #2 is due on Wednesday.