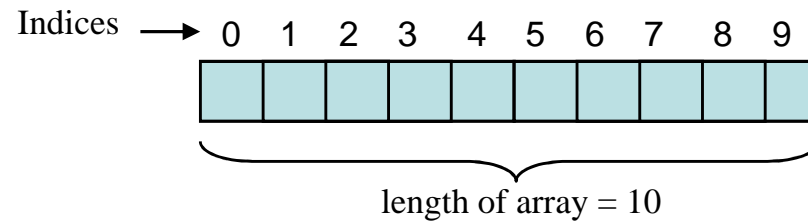

Collections

Chapter 12 - Lecture Slides

Arrays

-
- Array of ,
- array of ,
- array of ,
- array of
- All elements in an array must be the same data type
- Indices starting from
- An array of size N is indexed



Array Declaration

```
int grades[ ];  
int [ ] grades;
```

- This does NOT initialize the values!

Array Initialization

```
grades = new int[3];
```

- Or, do declaration and initialization together:

```
int [ ] grades = new int[3];  
grades[0] = 95;  
grades[1] = 87;  
grades[2] = 86;
```

- Or, using initializer lists

```
JButton controls[ ] = {  
    new JButton( "Left" ),  
    new JButton("Up"), new JButton("Right"),  
    new JButton("Down")  
};
```

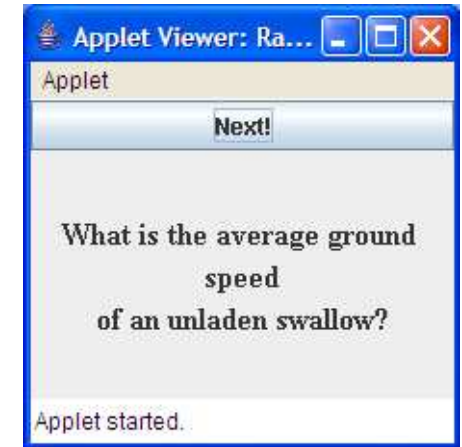
Accessing Array Elements

```
int [ ] grades = { 95, 87, 86 };
```

- First element is at index 0
 -
- Last element is at index: (length-1)
 - What does `grades.length` return?

Example Quotes

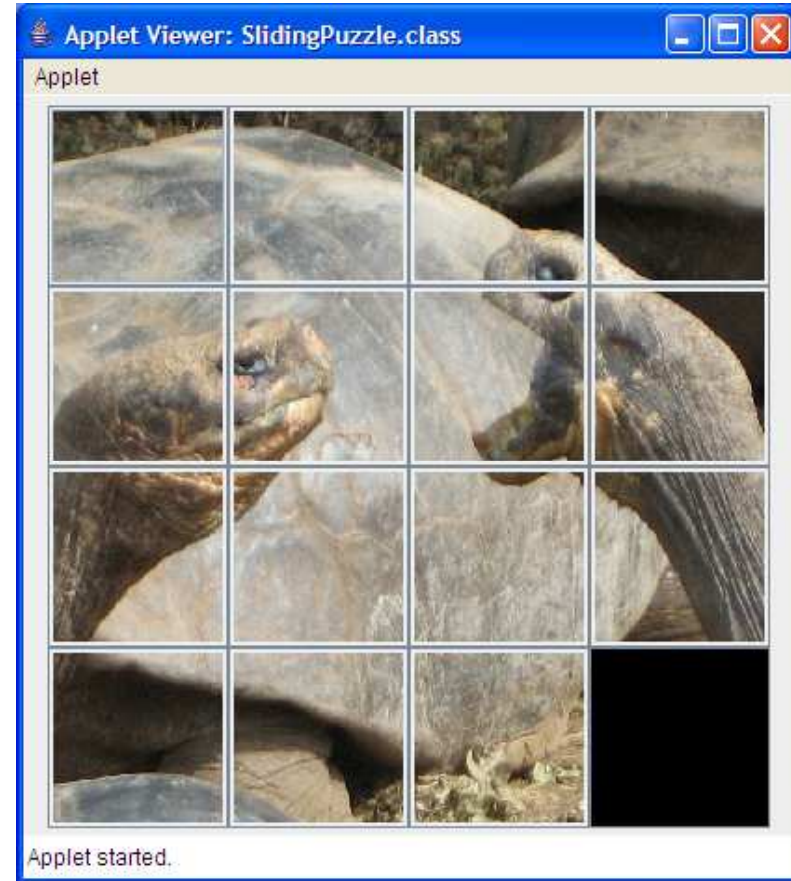
```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.*;
public class RandomQuoter extends JApplet
    implements ActionListener
{
    JButton next;
    JLabel label;
    Random random = new Random( );
    String[ ] quotes =
    {
        "What's up doc?",
        "Have you any Grey Poupon?",
        "Can you hear me now?",
        "<HTML><CENTER>What is the average ground speed "
            + "<BR> of an unladen swallow?",
        "May a platypus lay its eggs in your jockey shorts"
    };
};
```



```
public void init()
{
    setLayout( new BorderLayout( ) );
    next = new JButton( "Next!" );
    next.addActionListener( this );
    label = new JLabel( "Good morning!", JLabel.CENTER );
    label.setFont( new Font( "Serif", Font.BOLD, 16 ) );
    add( next, BorderLayout.NORTH );
    add( label, BorderLayout.CENTER );
}

public void actionPerformed( ActionEvent ae )
{
    Object obj = ae.getSource( );
    if ( obj == next )
    {
        int index = random.nextInt( quotes.length );
        label.setText( quotes[index] );
    }
}
}
```

Example – Sliding Puzzle



Array Buttons

- Create an array of buttons and add them to a GridLayout panel

```
 JButton controls[ ] = {  
     new JButton( "Left" ),  
     new JButton( "Up" ),  
     new JButton( "Right" ),  
     new JButton( "Down" )  
 };
```

```
public void init( )  
{  
    JPanel p = new JPanel( new GridLayout( controls.length,1 ) );  
    for ( int i=0; i<controls.length; i++ )  
        p.add( controls[i] );  
}
```

Array Buttons

- Create an array of buttons and add them to a GridLayout panel add events

```

    JButton controls[ ] = {
        new JButton( "Left" ), new JButton("Up"),
        new JButton("Right"), new JButton("Down")
    };
public void init( )
{
    JPanel p = new JPanel( new GridLayout( 4,1 ) );
    for ( int i=0; i<controls.length; i++ )
    {
        controls[i].addActionListener( this );
        p.add( controls[i] );
    }
}
public void actionPerformed((ActionEvent ae )
{
    Object src = ae.getSource( );
    if ( src == controls[0] )
        // do something based on first button (left) clicked
    else if ( src == controls[1] )
        // so something based on 2nd button clicked
    else
        ...
}

```

loopy arrays

- how do you find the smallest value?

set the smallest (so far) to the first element

set up the loop to go from the 2nd element to the last element in the array

check each element to see if it's smaller than the current smallest

if it is,

then make that element the smallest

loopy arrays

- how do you find the smallest value?

5	3	7	2
---	---	---	---

loopy arrays

- how do you find the smallest value?

```
int smallest = theArray[0];
for( int i=1; i<theArray.length; i++ )
{
    if( theArray[ i ] < smallest)
        smallest = theArray[i];
}
```

loopy arrays (as a method)

- how do you find the index of the smallest value?

```
public int minIndex( int theArray[ ] )
{
    int minIndex = 0;
    for( int i=1; i<theArray.length; i++ )
    {
        if( theArray[ i ] < theArray[ minIndex ] )
            minIndex = i;
    }
    return minIndex;
}
```

Multidimensional Arrays

- 2-D array – 3 students (rows)

```
int grades[ ] [ ] = new int[2][3];
```

```
grades[0][0] = 98;
```

```
grades[0][1] = 87;
```

```
grades[0][2] = 89;
```

```
grades[1][0] = 77;
```

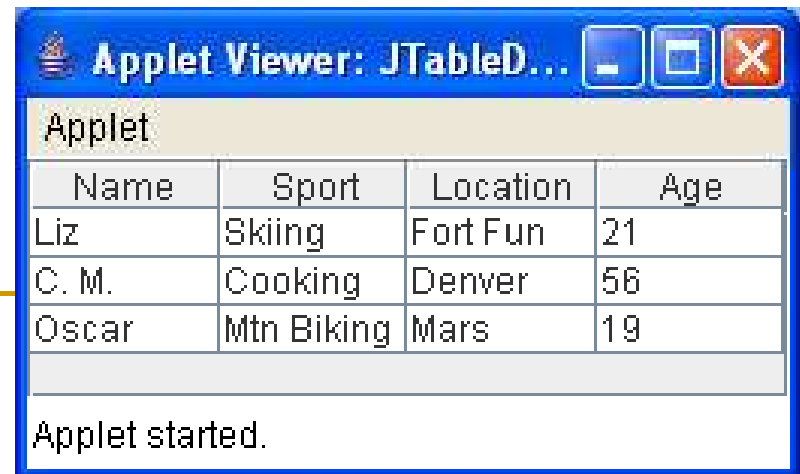
```
grades[1][1] = 79;
```

```
grades[1][2] = 68;
```


OR

```
int grades[ ] [ ] = { {98,87,89}, {77,79,68} };
```

JTable



Name	Sport	Location	Age
Liz	Skiing	Fort Fun	21
C. M.	Cooking	Denver	56
Oscar	Mtn Biking	Mars	19

Applet started.

JTable



Name	Sport	Location	Age
Liz	Skiing	Fort Fun	21
C. M.	Cooking	Denver	56
Oscar	Mtn Biking	Mars	19

Applet started.

- Rather complex widget
- Requires columns and data stored in **arrays**
 - Data goes in to a 2-dimensional array
 - Column headings go in to a single-dimensional array
- Must set the scrollable view size

```
table.setPreferredScrollableViewportSize(new Dimension(500, 70));
```

- Must put the table in a
- LOTS of additional features/complexity

JTable

- Column Headings : 1-D array

```
String[ ] colHeadings = { "Name", "Sport", "Location", "Age" };
```

- Data : 2-D array

```
Object data[ ][ ] = { { "Liz", "Skiing", "Fort Fun", "21" },  
                      { "C. M.", "Cooking", "Denver", "56" },  
                      { "Oscar", "Mtn Biking", "Mars", "19" },  
                      };
```

- To create our table:

```
JTable table;  
table = new JTable(data, colHeadings);
```

- Set viewable area

```
table.setPreferredSize(new Dimension(500, 70));
```

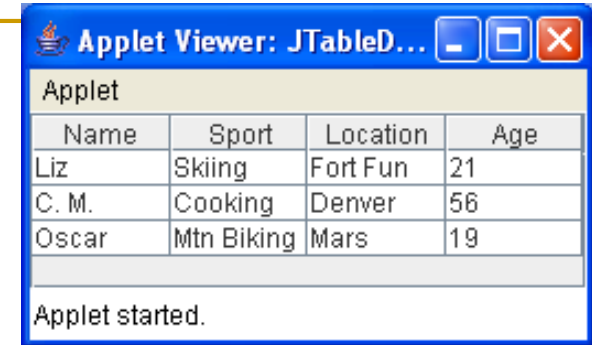
- Add to a scroll pane

```
JScrollPane scrollPane = new JScrollPane(table);  
add(scrollPane); //Add the scroll pane to this panel.
```

JTable

```
import java.awt.*;
import javax.swing.*;
public class JTableData extends JApplet
{
    String[] colHeadings = { "Name", "Sport", "Location", "Age" };
    Object data[][] = {
        { "Liz", "Skiing", "Fort Fun", "21" },
        { "C. M.", "Cooking", "Denver", "56" },
        { "Oscar", "Mtn Biking", "Mars", "19" },
    };

    JScrollPane scrollPane;
    JTable table;
    public void init( )
    {
        table = new JTable( data, colHeadings );
        table.setPreferredSize(new Dimension(400, 70));
        //Create the scroll pane and add the table to it.
        scrollPane = new JScrollPane(table);
        //Add the scroll pane to this panel.
        add(scrollPane);
    }
}
```



JTable Notes

- Can only put *objects* (String, Integer, Boolean, JButton) in the cells of a table, not *primitive* data types (int, boolean, char)
- Primitive data types must be wrapped up inside a Wrapper class
 -
 -
 -
 -

```
int age = 5;  
Integer intEx = new Integer( age );  
boolean val = false;  
Boolean bool = new Boolean( val );
```

JTable - Features

- Change the color of the grid lines

```
table.setGridColor( Color.GREEN );
```

- Change the width of each column

```
table.setAutoResizeMode( JTable.AUTO_RESIZE_OFF );  
table.getColumnModel().getColumn(0).setPreferredWidth(50);  
table.getColumnModel().getColumn(1).setPreferredWidth(200);  
table.getColumnModel().getColumn(2).setPreferredWidth(100);  
// continue for each column
```

- Lock the width of each column to a set size

```
table.getColumnModel().getColumn(0).setPreferredWidth(50);  
table.getColumnModel().getColumn(0).setMinWidth(50);  
table.getColumnModel().getColumn(0).setMaxWidth(50);  
// continue for each column
```

JTable - Features

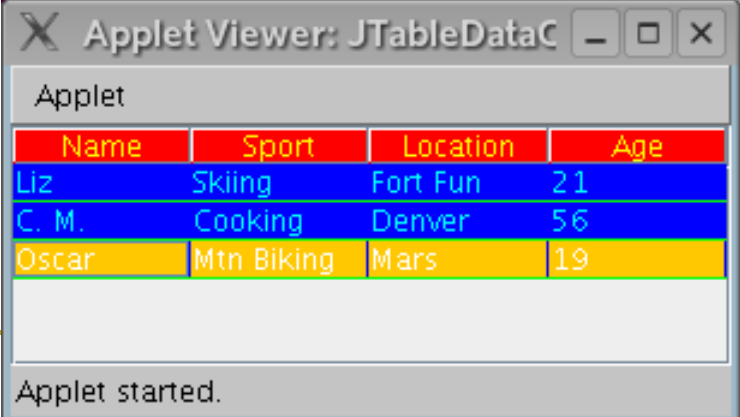
- Make the cells uneditable

```
JTable table;  
DefaultTableModel dfmodel = new DefaultTableModel(data, roster)  
    {  
        public boolean isCellEditable(int row, int column)  
        {  
            return false;  
        }  
    };  
table = new JTable(dfmodel);
```

JTable - Features

- **Change the colors of the column headings**

```
DefaultTableCellRenderer head = new DefaultTableCellRenderer( );  
head.setBackground( Color.YELLOW );  
head.setForeground( Color.BLUE );  
table.getColumnModel().getColumn(0).setHeaderRenderer(head);  
table.getColumnModel().getColumn(1).setHeaderRenderer(head);  
table.getColumnModel().getColumn(2).setHeaderRenderer(head);  
    // continue for each column
```



The screenshot shows a window titled "Applet Viewer: JTableDataC" with standard window controls. Inside the window, there is a table with four columns: "Name", "Sport", "Location", and "Age". The header row has a yellow background and blue text. The data rows have a blue background and yellow text. The data rows are: "Liz", "Skiing", "Fort Fun", "21"; "C. M.", "Cooking", "Denver", "56"; and "Oscar", "Mtn Biking", "Mars", "19". Below the table, the text "Applet started." is visible.

Name	Sport	Location	Age
Liz	Skiing	Fort Fun	21
C. M.	Cooking	Denver	56
Oscar	Mtn Biking	Mars	19

JTable

- Lots of additional features
 - Requires a fair bit of additional code
 - Adds complexity
 - Features
 - Images
 - Checkboxes
 - Sort by column when user clicks on heading
 - Enable/disable editing
 - Highlight cells/rows/columns

ArrayList

- Store a collection of *objects*
- Typecast the list to a specific type of object
 - `ArrayList <String> mylist;`
`mylist = new ArrayList<String>();`
 -
- Add to the list
 - `mylist.add("Cookie Monster");`
 -

ArrayList

- Number of elements in the list
 - `int numElements =`
- Access an element in the list with `get` and the index
 - `String name = mylist.get(0);`
 - `JButton b =`
- Remove from the list – specify index or object to remove
 - `mylist.remove(0);`
 -

Random Quote Example with ArrayList

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.*;
public class RanQuoteAL extends JApplet implements ActionListener
{
    JButton next;
    JLabel label;
    Random random = new Random();
    ArrayList<String> quotes;
    public void init()
    {
        setLayout( new BorderLayout() );
        quotes = new ArrayList<String>( );
        quotes.add( "What's up doc?" );
        quotes.add( "Have you any Grey Poupon?" );
        quotes.add( "Can you hear me now?" );
        quotes.add( "What is the speed of a swallow?" );
        quotes.add( "As you wish" );
        next = new JButton( "Next!" );
        next.addActionListener( this );
        label = new JLabel( "Good morning!", JLabel.CENTER );
        label.setFont( new Font( "Serif", Font.BOLD, 16 ) );
        add( next, BorderLayout.NORTH );
        add( label, BorderLayout.CENTER );
    }
}

public void actionPerformed( ActionEvent ae )
{
    Object obj = ae.getSource();
    if ( obj == next )
    {
        if ( quotes.size() > 0 )
        {
            int index = random.nextInt( quotes.size() );
            label.setText( quotes.get( index ) );
            quotes.remove( index );
        }
        else
            label.setText( "I have nothing more to say." );
            // ran out of quotes
    }
}
```

Summary

- Arrays
 - 1-D
 - Multi-Dimensional
- JTable
 - Basics
 - Customizations
 - Other Features
- ArrayList