

Objects From the Beginning With GUIs

Viera K. Proulx, Jeff Raab, Richard Rasala
ITiCSE 2002 Conference
Aarhus, Denmark
June 24, 2002

This work was partially supported by NSF grant DUE-9950829



Overview



- Objects first – why is it hard
- OO in Java – a survey of approaches
- Key Concepts of OO in Java
- The first four labs
- The Problem Set Framework
- Future plans
- Conclusion

Objects first – why is it hard

- Class – syntax alone
- Member data – different kind, access
- Methods – parameters, return type
- Object instances
- Syntax for calling the methods
- Using the method return value
- Extending base class; interfaces

OO in Java: other approaches

- MiniJava – selected subset
- Dr.Java – own environment – not ‘real
- BlueJ – nice workspace
- BreezyGUI – some GUI support
- Graphics Library – own set of classes
- None extend readily into real Java

Key Concepts of OO in Java

- Mental model of class
 - Member data, methods, info hiding
- Java code organization
- Type/class, object instance, reference
- Java syntax
- Algorithm logistics
- Algorithm implementation

Pedagogy

- Motivate
- One concept at a time
- Read before writing
- Learn from examples
- Think before you write
- Multiple representations
- Support exploration

The first four labs



1. Explore behavior
2. Observe and explore class/methods
3. Read and modify code
4. Implement part of the solution

Extensive reinforcement

Through code to explore

Through visual feedback

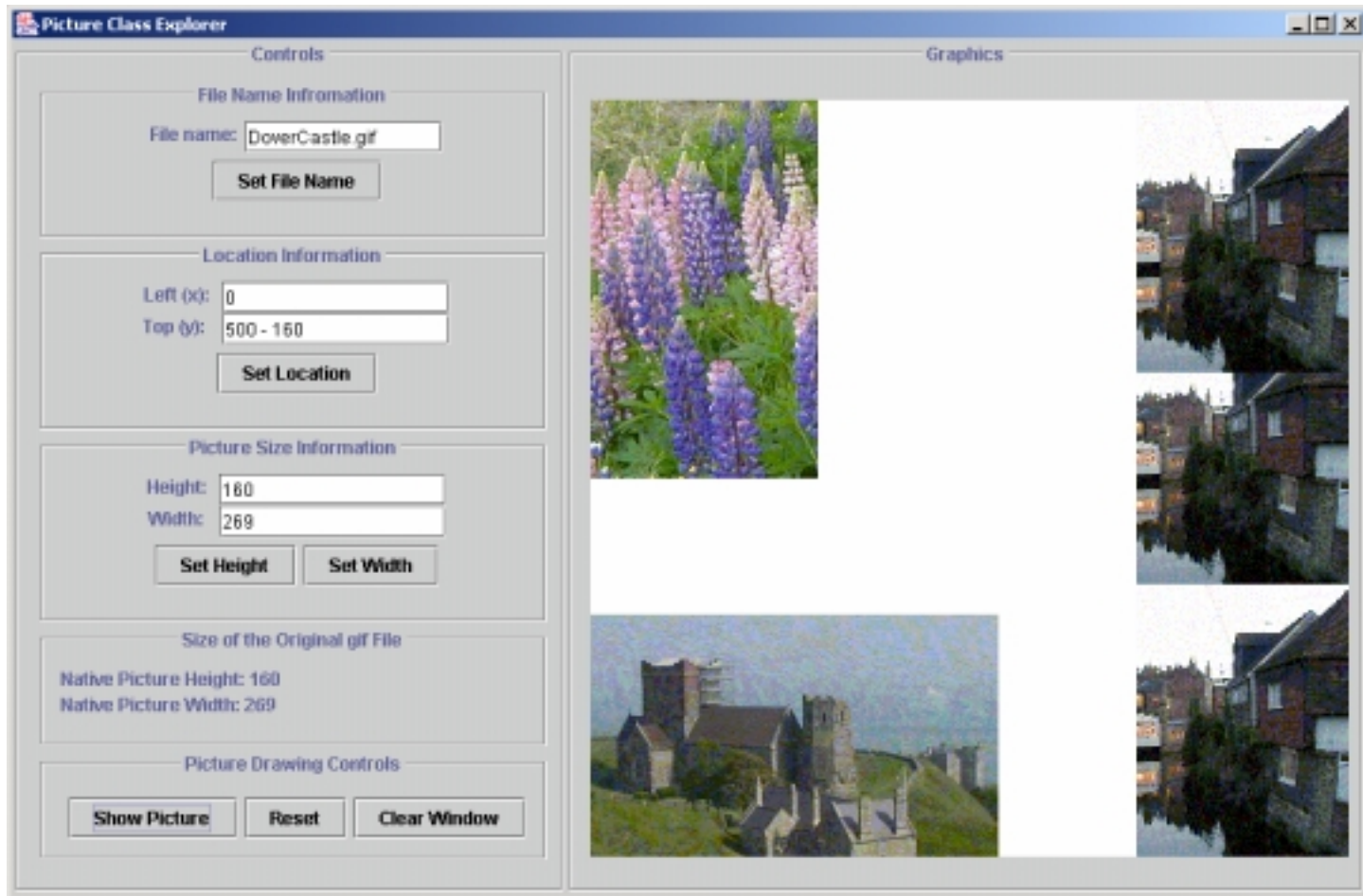
Lab 1: Picture Explorer

Explore the behavior of a picture object

- file name, location, size
- methods to set values
- method to show picture
- geometry of the graphics window

Learn to work with the IDE, snapshots...

Lab 1: Picture Explorer

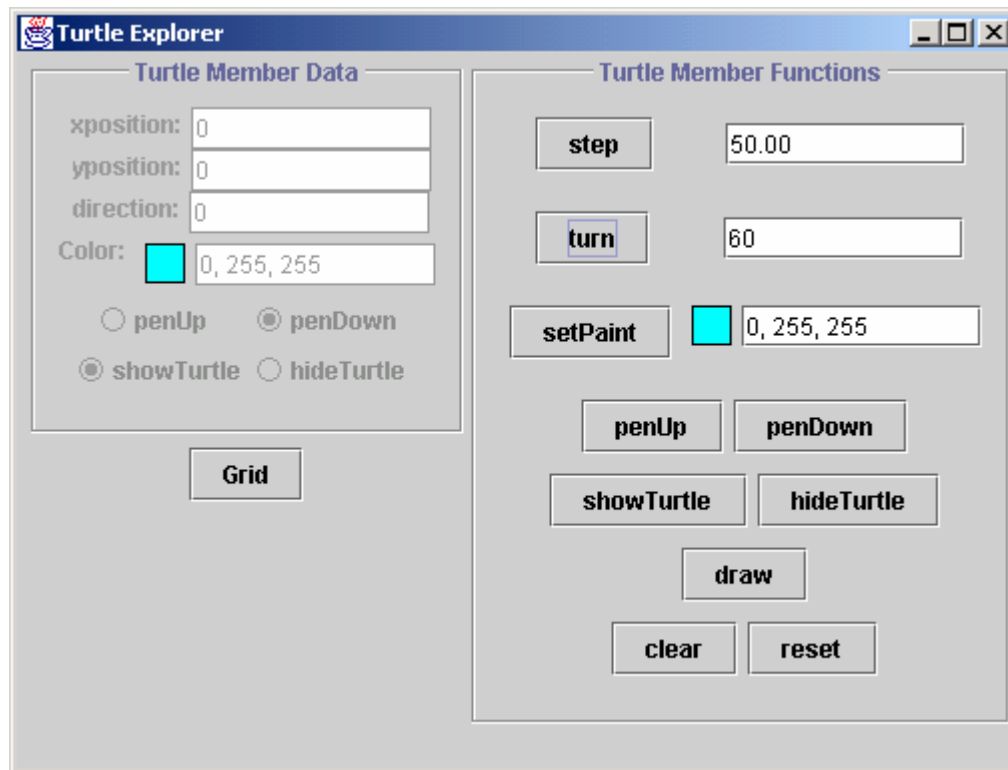


Lab 2: Turtle Explorer

Observe and explore class/methods

- member data display
- methods display - with argument inputs
- method calls perform actions
- echo method call syntax in console
- display action in graphics window
- build your own draw method body

Turtle Class Explorer



```
myTurtle.setPaint(0, 0, 255);  
myTurtle.step(50.0);  
myTurtle.turn(60.0);  
myTurtle.setPaint(255, 0, 255);  
myTurtle.step(50.0);  
myTurtle.turn(60.0);  
myTurtle.setPaint(255, 0, 0);  
myTurtle.step(50.0);  
myTurtle.turn(60.0);
```

Lab 3: Ticket Seller

- read and modify code
- implement part of the solution
- observe multiple object instances
 - preserving their state
 - performing same actions
- separation of model and view

Lab 3: Ticket Seller

The screenshot shows a window titled "Ticket Seller" with two main sections: "Sales Statistics" and "Ticket Order and Prices".

Sales Statistics

| | | |
|----------|---------|--------|
| Show: | Tarzan | |
| | Tickets | Income |
| Adult: | 12 | \$240 |
| Student: | 12 | \$144 |
| Child: | 10 | \$80 |
| Total: | 34 | \$464 |

Buttons: Tarzan, King Kong, Casablanca, Reset Sales

Ticket Order and Prices

| | | |
|--------------------|--------------------------------|------------|
| Tarzan | Tickets Available | 166 |
| | Tickets | Prices |
| Number of Adult: | <input type="text" value="4"/> | \$20 |
| Number of Student: | <input type="text" value="2"/> | \$12 |
| Number of Child: | <input type="text" value="6"/> | \$8 |
| Total: | 12 | \$152 |

Buttons: Clear, Total, Buy

Show class:

prices

sales

total tickets

has methods to
compute stats

Show objects

retain information

Lab 4: Scaled Picture Lab

- implement part of the solution
- focus on picture design
- observe several derived classes
- common behavior in abstract base class
- use existing Java classes
 - Color, Rectangle, Ellipse, Line...
- simple loops - grid of pictures

Lab 4: Scaled Picture Lab

Scaled Picture Practice

Controls

Picture Title
Title: DoverCastle.gif
Author: Viera Proulx

Location Information
Left (x): 400
Top (y): 400
Set Location

Selection
 Tree
 Engine
 Face
 Student
 Castle
 Flowers

Picture Size Information
Height: 100
Width: 100
Set Height **Set Width**

Picture Testing
Four Corners **3 X 4 Mosaic**
Diagonal **6 x 4 Mosaic**

Select Rows and Columns
Rows: 0
Columns: 0
Rows and Columns

Picture Drawing
Show Picture
Show Grid
Reset Picture
Picture Mosaic
Clear Window

Graphics

The Graphics panel displays a 3x3 grid of images. The background is blue with green trees and yellow suns. The castle image is placed in the center cell. A grid is overlaid on the bottom-right cell, showing the castle image being scaled and positioned within the grid.

The Problem Set Framework

Application that allows for rapid testing

Action button for each function given as

```
public synchronized void ButtonName();
```

Access to error-checked console

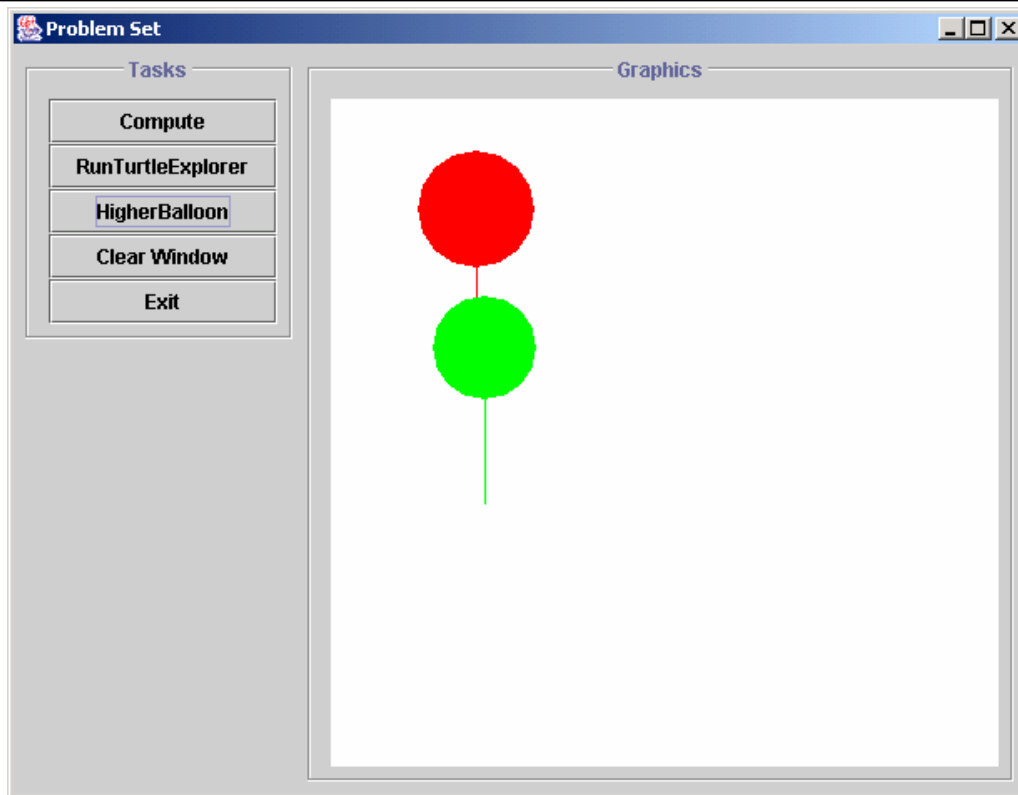
Access to buffered graphics

Access to any other class

- to create objects

- to test methods

The Problem Set Framework



test expressions
run apps
test class methods
show objects
reference model
test GUI building
regression tests

Future plans



- focus more on design recipes
- provide guidelines for program design
- follow up with
 - interacting classes
 - abstract classes
 - implementing interfaces
- illustrate/reinforce dynamic dispatch
- exploratories for these concepts

Future plans

- further development of JPT
- more examples using PSF
- JPT tutorial book
- possible textbook

Website:

<http://www.ccs.neu.edu/jpt>

Conclusion

- Objects First can be done
- Students need
 - context
 - opportunity to explore
 - support for experimentation
- Abstraction lessons should be real
- Lead/teach by example