CSC207.01 2013F, Class 12: Interfaces and Polymorphism

Overview

- Admin.
- Interfaces.
- Polymorphism.
- An example: Text blocks.

Admin

- Reading for Friday: Inheritance
- Sorry for the mixup/delay on today's readings. I'll do my best to guide you through the same material.
  - (No lab.)
- EC Opportunities
  - Convocation noon, Wednesday.
  - Learning from Alumni Thursday @ 2:15 Sam Tape and company. (3821)
  - CS Extras Thursday @ 4:30: Kim Spasaro on Linguistics Programming (3821)
  - CS Table Friday (pair programming)
  - Other?
- Are there questions on HW4?
  - The registers should hold fractions

Interfaces

- Goal in program design: Separate WHAT your code does from HOW your code achieves that goal.
  - You can change your implementation without affecting your client code
  - Your clients can't "mess up" your code.
  - You think differently about programming if you separate the two
- Example: Points in the plane
  - Get the coordinates of the point:
    - distance from the x axis and y axis (x, y)
      - `getX()`
      - `getY()`
    - distance from the origin and angleq
      - `getAngle()`
      - `getDistance()`
- Java encourages this approach

- Interfaces provide the WHAT, not the how

```
public interface Point2D {
    double getX();
    double getY();
    double getAngle();
    double getDistance();
    Point2D add(Point2D other);
    // DOT PRODUCT!
    Point2D multiply(Point2D other);
} // interface Point2D


public class XYPoint implements Point2D {
   ...
}


public class Vector2D implements Point2D {
   ...
}
```

- Two magic things happen with the "implements" keywords
  - Java forces you to implement all of the methods in Point2D.
  - If someone writes code that expects a Point2D, it will work with an XYPoint

Polymorphism

- Polymorphism is an approach to save code / avoid repetitous code.
- Consider squaring numbers
  - Scheme

    (define square (lambda (num) (* num num)))a
  - C

    int squareInt(int x) { return x*x; } double squareDouble(double x) { return x*x; }
  - Java permits overloading

    int square(int x) { return x*x; } double square(double x) { return x*x; }
  - BUt the code is repetitious!
  - Why not code with copy-paste-change?
    - Inelegant - You can do it better by writing a single procedure (we hope)
    - What if we have to change something?
    - Code bloat

- Scheme's approach is great, but not

An example: Text blocks

```
public interface TextBlock {
   int getWidth();
   int getHeight();
   String getRow(int i) throws Exception;
} // interface TextBlock

public class TextLine implements TextBlock {
   ...
}

public class VerticallyComposeTextBlock implements TextBlock {
  ...
}

TextBlock fiona = new TextLine("Hello");
TextBlock john = new TextLine("Goodbye");
TextBlock adam = new VCTB(fiona,john);
TextBlock mark = new TextLine("Mark");
TextBlock sunshine = new VCTB(adam, mark);
```