

A programming environment  
supporting a prototype-based  
introduction to OOP

# CONTEXT

---

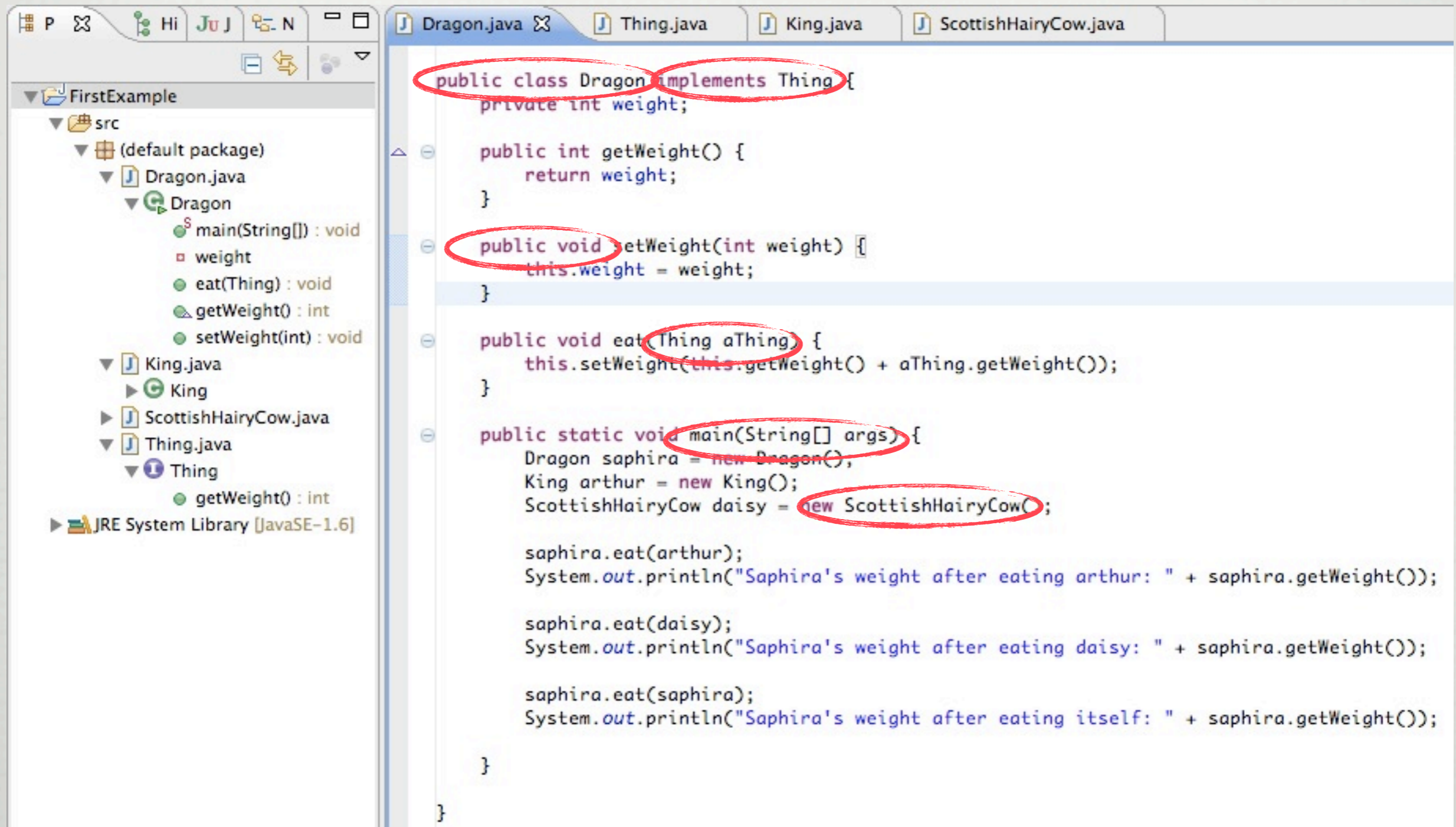
- First OOP course
- Previous knowledge in structured programming (PASCAL)
- Tight schedule

# PROBLEM

---

- Paradigm shift
- A lot of new “stuff” to learn in order to get a first working program

# EXAMPLE IN JAVA



The screenshot shows an IDE with a project named 'FirstExample'. The 'src' directory contains a 'default package' with files 'Dragon.java', 'King.java', 'ScottishHairyCow.java', and 'Thing.java'. The 'Dragon.java' file is open, showing the following code:

```
public class Dragon implements Thing {
    private int weight;

    public int getWeight() {
        return weight;
    }

    public void setWeight(int weight) {
        this.weight = weight;
    }

    public void eat(Thing aThing) {
        this.setWeight(this.getWeight() + aThing.getWeight());
    }

    public static void main(String[] args) {
        Dragon saphira = new Dragon(),
        King arthur = new King(),
        ScottishHairyCow daisy = new ScottishHairyCow();

        saphira.eat(arthur);
        System.out.println("Saphira's weight after eating arthur: " + saphira.getWeight());

        saphira.eat(daisy);
        System.out.println("Saphira's weight after eating daisy: " + saphira.getWeight());

        saphira.eat(saphira);
        System.out.println("Saphira's weight after eating itself: " + saphira.getWeight());
    }
}
```

Red circles highlight the following elements in the code:

- `public class Dragon implements Thing`
- `private int weight;`
- `public void setWeight(int weight)`
- `eat(Thing aThing)`
- `main(String[] args)`
- `new ScottishHairyCow()`

# EXAMPLE IN SMALLTALK

The image shows a screenshot of the Smalltalk IDE. The main window is titled "Dragon" and displays a class hierarchy on the left with "Dragon" selected. The right pane shows the class's methods: "eat:", "initialize", "weight", and "weight:". A red circle highlights the text "-- all -- accessing as yet unclassified" in the right pane. Below the hierarchy, the "Browse" tab is active, showing the object's subclass as "#Dragon", instance variable names as "weight", class variable names as "", pool dictionaries as "", and category as "FirstExample". A second window titled "Workspace" is open, showing several lines of Smalltalk code: "saphira := Dragon new.", "arthur := King new.", "daisy := ScottishHairyCow new.", "saphira eat: arthur.", "saphira weight.", "saphira eat: daisy.", "saphira weight.", "saphira eat: saphira.", "saphira weight.". The line "daisy := ScottishHairyCow new." is circled in red.

Dragon

FirstExample  
AST-Core-Matching  
AST-Core-Nodes  
AST-Core-Parser  
AST-Core-Pattern  
AST-Core-Tokens  
AST-Core-Visitors  
AST-Semantic  
AST-Semantic-Binding  
AST-Semantic-Exceptions  
AST-Semantic-Scope  
AST-Tests-Core  
AST-Tests-Semantic  
Announcements-Core

Dragon  
King  
ScottishHairyCow

-- all --  
accessing  
as yet unclassified

eat:  
↑ initialize  
weight  
weight:

Instance ? Class

Browse Hierarchy Variables Implementors Inheritance

Object subclass: #Dragon  
instanceVariableNames: 'weight'  
classVariableNames: ''  
poolDictionaries: ''  
category: 'FirstExample'

Workspace

```
saphira := Dragon new.  
arthur := King new.  
daisy := ScottishHairyCow new.  
saphira eat: arthur.  
saphira weight.  
saphira eat: daisy.  
saphira weight.  
saphira eat: saphira.  
saphira weight.
```

# MOTIVATION

---

Focus on:

Messages

Objects

Rich interaction between them

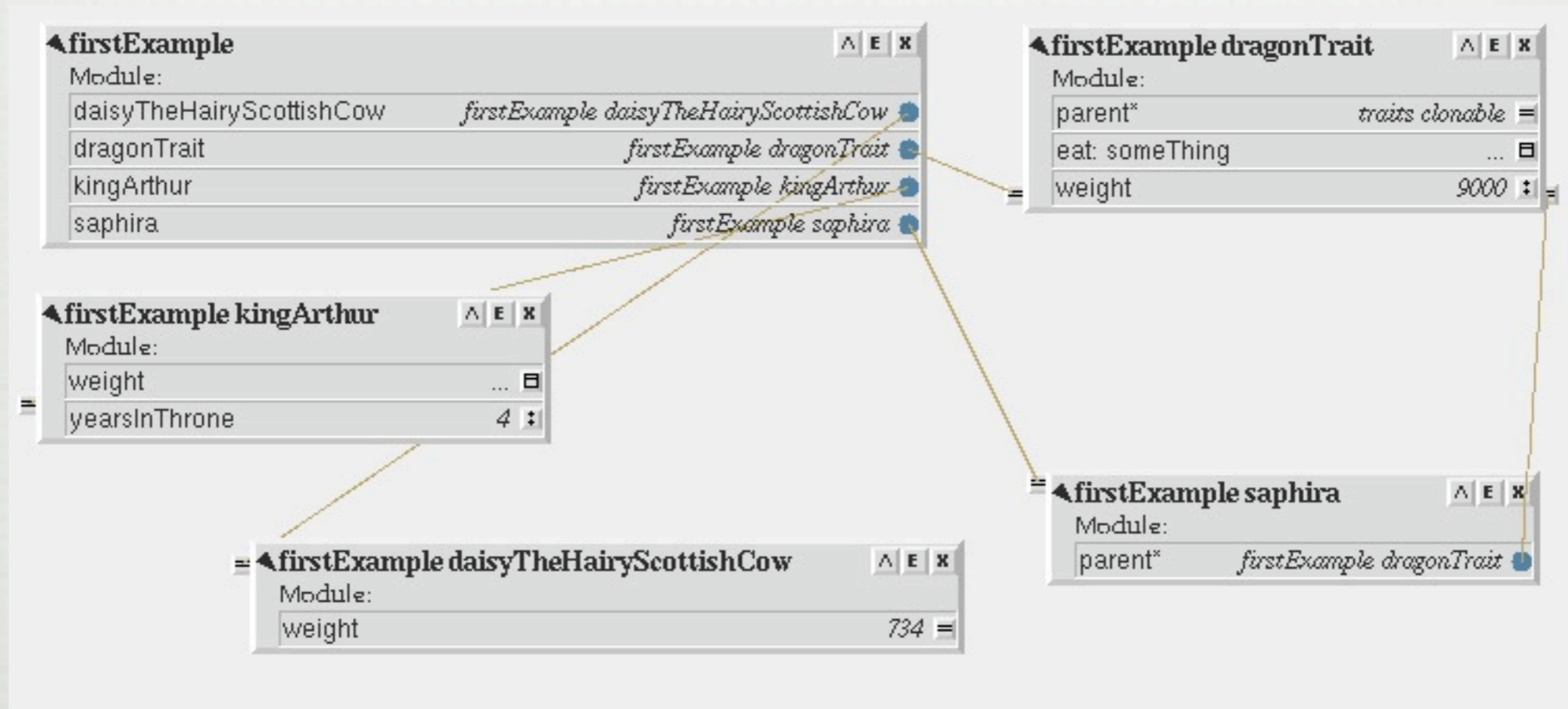
Polymorphism

# SOLUTION

---

- Start with a custom programming environment
- Delay the introduction of classes and inheritance

# WHAT ABOUT SELF?





# WHAT ABOUT SELF?

---

- Traits, prototypes, parent slots, etc.
- GUI is not familiar to the students

# FEATURES

---

- Definition and usage of objects
- Multiple object environments
- Live object diagrams
- Explicit differentiation between objects and references

# EXAMPLE IN LOOP

The image shows a screenshot of the Esug2011 - Object Browser and Esug Workspace. The Object Browser window displays a diagram of objects and their references. The diagram shows three 'anObject' nodes. One 'anObject' node is connected to 'daisyTheHairyScottish' (a cow object). Another 'anObject' node is connected to 'saphira' (a cow object). A third 'anObject' node is connected to 'kingArthur' (a cow object). The 'daisyTheHairyScottish' node has a 'weight' property set to '5000'. The Esug Workspace window shows the following execution logs:

```
saphira initialize.  
saphira eat: kingArthur.  
saphira eat: daisyTheHairyScottishCow|
```

Below the workspace, there are two buttons: 'Importar Leccion' and 'Nueva Leccion'.

# TOWARDS CLASSES

---

```
saphira >> eat: aThing  
    self weight: self weight + aThing weight.
```

```
arthur >> weight  
    ^yearsInThrone * 666
```

```
daisy >> weight  
    ^732
```

# TOWARDS CLASSES

---

```
Dragon >> eat: aThing  
        self weight: self weight + aThing weight.
```

```
King >> weight  
        ^yearsInThrone * 666
```

```
ScottishHairyCow >> weight  
        ^732
```

# EXPERIENCE

Quarter	Pass Rate
2011 Q1	84,62%
2010 Q2	68,42%
2010 Q1	69,76%
2009 Q2	80,95%
2008 Q2	66,67%
2008 Q1	74,07%
2007 Q2	73,33%
2006 Q2	75,00%



# CONCLUSIONS

---

- More time for the important concepts
- More time for more complex exercises
- Concrete to abstract learning path

# QUESTIONS

---

