**Objectives**

- Picasso Design/Parsing

---

**Project Deliverables Timeline**

<table>
<thead>
<tr>
<th>Deliverable</th>
<th>Who</th>
<th>Weight</th>
<th>Due Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preparation</td>
<td>Individual</td>
<td>8%</td>
<td>12/2</td>
</tr>
<tr>
<td>Preliminary</td>
<td>Team</td>
<td>37%</td>
<td>12/7</td>
</tr>
<tr>
<td>Implementation</td>
<td>Team</td>
<td>40%</td>
<td>You decide</td>
</tr>
<tr>
<td>Final</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis</td>
<td>Individual</td>
<td>15%</td>
<td>12/18</td>
</tr>
</tbody>
</table>

---

**Picasso Project Overview**

- Goal: Generate images from expressions
- Every pixel gets assigned a color, computed from its x and y coordinate and the given expression

- Colors are RGB values:
  - Range [-1, 1]
  - Black is [-1,-1,-1]
  - Red is [1,-1,-1]
  - Yellow is [1, 1,-1]

- [-1, -1] x 1, -1
- 1, -1 x 1, 1
- 1, 1 x [-1, 1, -1]

For all x:

\[
\text{pixels}[x][y] = \text{expression.evaluate}(x, y)
\]

---

**Examples**

\([-1, 1, -1]\) 
\(x\) 
\(x^*y\)

---

**Specification**

- User can enter expressions:
  - Interactively or from file
  - Language is defined in specification
- Lots of possible extensions

---

**Programming Language Syntax**

- What does an identifier look like in Java?
- What does an assignment statement look like in Java?
- What can be on the left hand side?
- What can be on the right hand side?
- What does a multiplication look like?
- How do we evaluate arithmetic expressions?
Programming Language Design

• Must be unambiguous
  ➢ Programming Language defines a syntax and semantics

• Interpreting programming languages
  ➢ Parse program into tokens
    • Example: \( x = 4 \times 3; \) ➞
    \(<\text{id}> \text{ <assignment> <num> <mult> <num> <endofstmt>}\>
  ➢ Verify that tokens are in a valid form
  ➢ Generate executable code

Interpreting User’s Input

• Lexical Analysis
  ➢ Recognize/create tokens
  ➢ Report errors in creating tokens

• Semantic Analysis
  ➢ Convert infix tokens into postfix
    • Report errors
  ➢ Parse tokens into expressions
    • Report errors

• Evaluation
  ➢ Evaluate expressions with respect to \( x \) and \( y \)

What We Need to Do/Represent

• Lexical Analysis
  ➢ Import an existing project from \(/\text{home/courses/cs209/handouts/picasso.tar}\)
  • Code has been updated since previous version
  • Some errors in giving testing code. Most will be fixed by end of today.
Understanding the Code

- What are the different parts of the code?

Interpreting User's Input

Java's StreamTokenizer

User's Input

Lexical Analyzer

Tokens

Expression Tree

OR

Error

Expression

parser

Compiler

Tokens

Error

Draw on canvas

Semantic Analyzer

FloorToken

FloorAnalyzer

Floor

Evaluation of expression

Understanding the Code

- Lexical analysis
  - `picasso.parser.Tokenizer`
  - `picasso.parser.tokens.TokenFactory`
  - `Output: picasso.parser.tokens.*`

Understanding the Code

- Semantic analysis
  - `picasso.parser.ExpressionTreeGenerator`
  - `picasso.parser.SemanticAnalyzer`
  - `picasso.parser.*Analyzer`
  - `Output: picasso.parser.language.expressions.*`

Understanding the Code

- Evaluation
  - Base class:
    - `picasso.parser.language.ExpressionTreeNode`
  - `Output: RGBColor`

Understanding the Code

- Create a token for the sine function
  - Same prefix as new function, e.g., `SinToken.java`
  - Needs to be added to `functions.conf`

- Create a semantic analyzer for the function with same prefix as function, e.g., `SinAnalyzer.java`
  - Analyzer class (presumably implementing `SemanticAnalyzerInterface`) returns an instance of `ExpressionTreeNode`

- Create an `ExpressionTreeNode` for function `Sine.java`
What Do You Think You’ll Need To Do About Binary Operators?

For Wednesday: Project Preparation

• Read over the Picasso (Final Project) specifications again
• 1st deliverable is a text document that answers
  ➢ What needs to be completed?
  ➢ What is your plan for completing those tasks?
  ➢ What tasks are you most interested in working on?
  ➢ ....
• Wednesday
  ➢ Discuss your plans, questions
  ➢ Discuss tools to help collaboration