Programming Syntax
Values

• 3
• 64.6
• ‘x’ ‘3’
• “Hello, world!”
• myOval
• yourVelocity
Types (and literals)

- **int**: 6 32 -5
- **long**: 6L 32L -50000000L
- **double**: 3.2 6e-12
- **float**: 3.2f 6e-12F
- **char**: ‘c’ ‘\n’
- **boolean**: true false
- **String**: “Any string\n”
Names

• Every name is born with a type:
  • `int i; GP/Graphics/Oval myOval;`

• Declaration gives a type:
  • `type-of-thing name-of-thing;`

• Definition gives type and initial value:
  • `type name = valueExpression;`

• Names refer to values.
Shoebox Names

• Some names are shoeboxes.
• Different sizes for different types.
• Exactly one value at all times.
• Value is copied to new shoebox.

• Primitive (lower case) types are shoebox types.
Label Names

• All other names are labels.
• A label may be stuck on one thing — or on nothing (null).
• Two labels may be stuck on the same thing.

• All non-primitive types are label types.
Expressions...

- ...include literals and names.
- ...do things with values:
  
  \[
  \begin{align*}
  \text{myAge} + 1 & \quad \text{mySize}.GetWidth() \\
  i &= i \% 4 & i &= 3
  \end{align*}
  \]
- ...have types and values.
- ...can be constructed out of other expressions:
  
  \[
  ((\text{myAge} + 1) == 21) \\
  \quad \text{&&} \quad (\text{mySize}.GetWidth() > 10)
  \]
Statements...

• ...do not have types or values.
• ...include declarations, definitions, and expression;
• ...include if/else, while.
• ...include return; and return expr;
The Story So Far....

values

names

c

GP.Attributes.Color()
The Story So Far....

values
expressions
names

c

new GP.Attributes.Color()
The Story So Far....

values

expressions

statements

GP.Attributes.Color c = new GP.Attributes.Color();
The Story So Far....

values

{ expressions \rightarrow statements \rightarrow rules

names

GP.Attributes.Color c = new GP.Attributes.Color();
myOval.SetColor(c);
public void SetColor (GP.Attributes.Color c) {
    myOval.SetColor(c);
}
The Story So Far....

values

names

types

expressions → statements → rules

declarations → fields

methods

GP.Graphics.Oval myOval;
The Story So Far....

class Cool
{
    GP.Graphics.Oval myOval;
    public void SetColor (GP.Attributes.Color c)
    {
        myOval.SetColor(c);
    }
}
Scope

• Names can only be known in certain places
  • Your private names, with family, friends, others
  • Fido in your house is different from Fido in CA

• In Java, each level of brackets defines place
  • Each place can have its own names

• Need name to recall value
  • But expression can be used for one time access