PYTHON REFLECTION

Curt Clifton
Rose-Hulman Institute of Technology

Check out *PythonReflection* from SVN

SELF-ISH IDEAS IN PYTHON

- Prototypes:

  - Use the copy module

  - "Fudge" method update

- Named slots instead of variables:

  - __getattribute__(), __setattr__()

http://irrepupavel.com/documents/python/instancemethod/

THE EXPRESSION PROBLEM

CONSIDER

- A set of recursive datatypes, e.g.:

  - Widgets in a UI framework

  - Expressions in an interpreter

- Functions over the datatypes, e.g.:

  - layout(), render()

  - eval(), typecheck()

STANDARD OO SOLUTION

- A hierarchy of classes

- Recursive methods

Expression

eval()
typecheck()

2   3

BinaryOp

left
right
…

If

cond
cons
alt
eval()
typecheck()

Add

eval()
typecheck()

```
def eval(self):
    lv = self.left.eval()
    rv = self.right.eval()
    return lv + rv
```
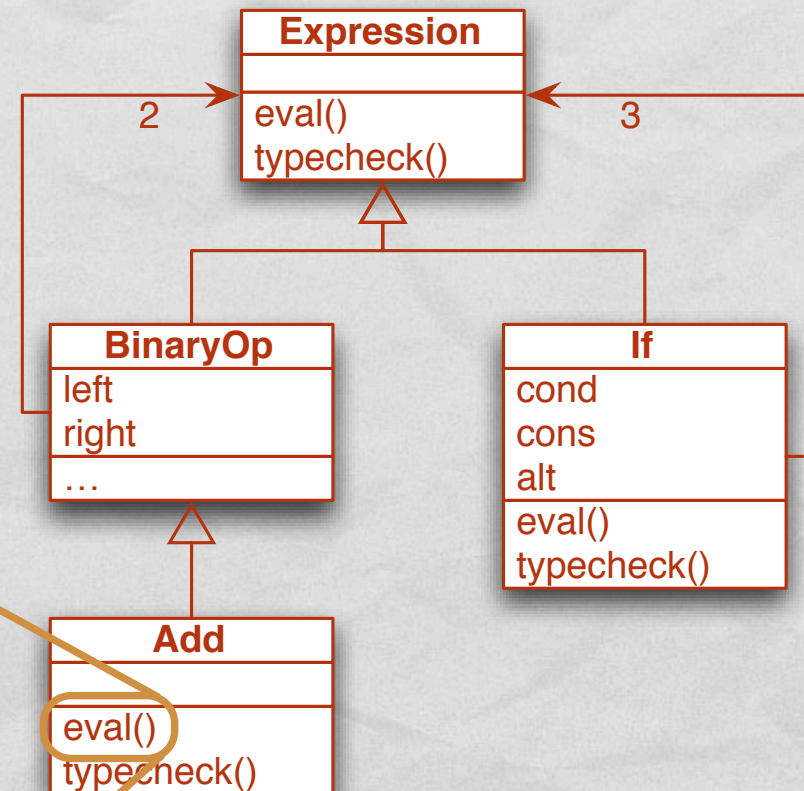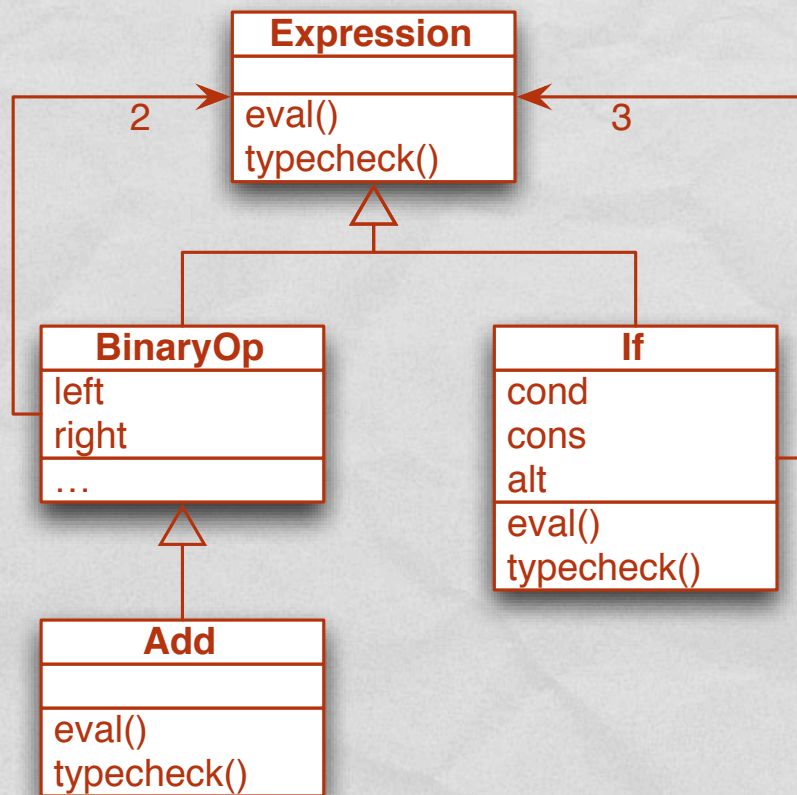
STANDARD FUNCTIONAL SOLUTION

- Records

- Recursive functions with cases
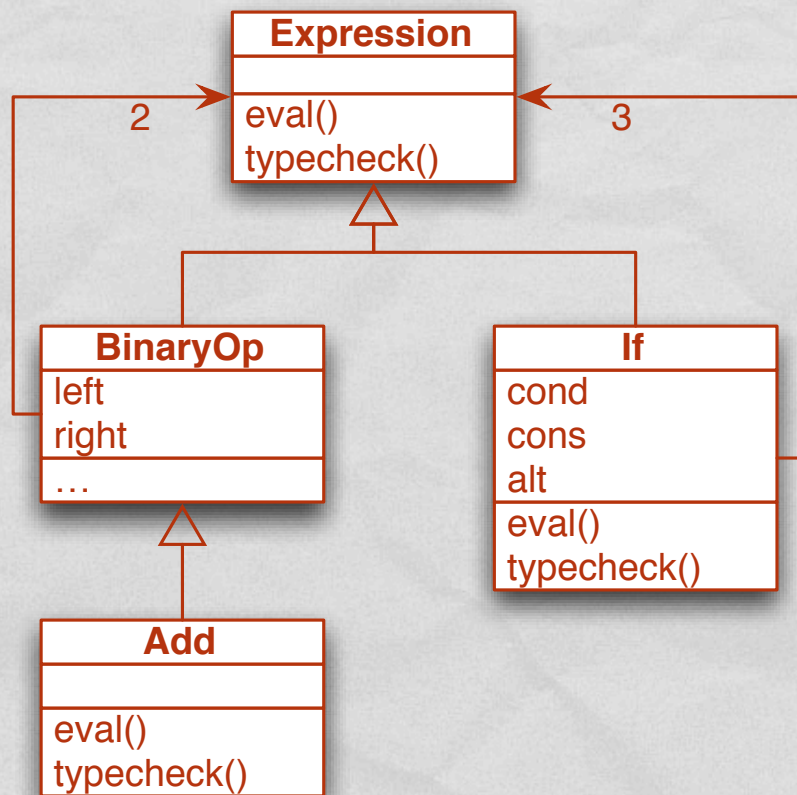
```
(define eval
  (lambda (exp)
    (cond
      ((add-exp? exp)
        (+ (eval (add-exp->left exp))
           (eval (add-exp->right exp))))
      (...))))

(define typecheck ...)
```

HOW WOULD WE ADD A "MINUS" EXPRESSION?

Expression

eval()
typecheck()

2   3

BinaryOp

left
right
…

If

cond
cons
alt
eval()
typecheck()

Add

eval()
typecheck()

```
(define eval
  (lambda (exp)
    (cond
      ((add-exp? exp)
       (+ (eval (add-exp->left exp))
          (eval (add-exp->right exp))))
      (…)))))

(define typecheck …)
```

Q1–3

HOW WOULD WE ADD A "PRETTYPRINT" OPERATION?

Expression

eval()
typecheck()

2

3

BinaryOp

left
right
...

If

cond
cons
alt
eval()
typecheck()

Add

eval()
typecheck()

```
(define eval
  (lambda (exp)
    (cond
      ((add-exp? exp)
        (+ (eval (add-exp->left exp))
           (eval (add-exp->right exp))))
      (...))))

(define typecheck ...)
```

Q4–6

OO WORK-AROUND: THE VISITOR PATTERN

- Use function objects to represent operations

- Add "accept" methods to all datatypes

- Use double-dispatch to match operations and datatypes

NEXT TIME

- Modifying Python's dispatch semantics

- Python wrap-up:

  - Think about strengths and weaknesses of Python and OOP in general