PEP 308: Conditional Events in Python

CSCI 3155: Aaron Holt, Pradyumna Kikkeri, Madison Rockwell

December 8, 2014

Python Background

- Created by Guido van Rossum - "owns" the language.
- Appeared in 1991
- Emphasized readability & fewer lines of code than C++/Java

What's a PEP?

- "Python Enhancement Proposal"
- Community input is key
- Lots of testing (alphas, betas, previews, etc.) before release

Difficulties about PEPs

- Lack of consensus
- Language still young
- Community divided on what each person thought was best

PEP 308

CSCI 3155: Aaron Holt, Pradyumna Kikkeri, Madison Rockwell

- PEP 308 introduced conditional (ternary) operators into Python
- Implemented in Python 2.5 - September 19, 2006

Before PEP 308 (Python 2.4 and older)


```
c and x or y    # incorrectly returns y if x
#is (any kind of) false

(c and [x] or [y])[0]  # reliable, but ugly and
#churns objects

(x, y)[not c]  # always evaluates both

(y, x)[c]      # only if c is
#really a bool (or otherwise 0 or 1)
```

Proposed solution number 1

- Guido van Rossum's own solution

```
<expression1> if <condition> else <expression2>
# condition evaluated first
```

Issues with Guido's solution

- Most confusing was the order
- Not the easiest to remember for C++/Java progammers

Advantages of Guido's solution

- No new key words
- Short-circuiting! Only one expression evaluated, thus optimizing performance

Strongest Contenders

Contender #1 *Implement the ternary operator like in C++ and Java*

```
<condition> ? <expression1> : <expression2>
```

- Advantages: familiar for C++/Java folks, easy to understand (left to right), short circuiting
- Rejected by Guido van Rossum, on the grounds that those *not* familiar with C++/Java would find it confusing
- Didn't want to add another role of the colon (:) operator in Python

Strongest Contenders

Contender #2 *Introduce an if-then-else syntax*

```
if <condition> then <expression1> else <expression2>
```

- Advantages: Short circuiting, left to right evaluation, easy to understand.
- Disadvantages: creation of a "then" keyword, which Guido van Rossum was hesitant to do.
- Parser could mistakenly believe that it's an if-statement

Strongest Contenders

Contender #3 *Introduce a parenthesized if-else statement*

```
(if <condition>: <expression1> else: <expression2>)
```

- Advantages: Prevents parser difficulty with parentheses, short circuited, left to right evaluation, standard python syntax
- Disadvantages: parentheses and a difference from convention, when the colon (:) is usually at the end of a line

The "do nothing" group

- A certain subset of the community wished to not change anything at all
- Continue using old conventions of using and + or logic to achieve conditionals
- Keep backwards compatibility, it almost broke with PEP 308 implementation
- In a nutshell, some list declarations are very similar to PEP 308 conditionals

Community Voting

- Guido van Rossum called for a vote
- 16 possible choices, with the winner being the one with the clear majority
- Of course, there was no clear majority

And the winner is. . . .

- Guido van Rossum, as the creator of the language, thus picked his own solution

```
<expression1> if <condition> else <expression2>
```

- Or if you really want to get adventurous

```
<expression1> if <condition1> else <expression2>
if <condition2> else <expression3>
# almost like Scala case-matching
```

Principles of a Programming Language

We're going meta here

Syntax

- Standardized the if/else expression in Python
- Adhered to Python style and standards, e.g., no parentheses around the conditional

Semantics

- Meaning of the syntax
- if, else was added to the Python language documentation
- It set a standard for *when* the standard was supposed to be used

Parsing

- New type-checking and inference
- New evaluation judgment forms
- All of this keeps the syntax unambiguous

Resolution

- Guido van Rossum accepts proposal on Friday, September 30th, 2005
- Implemented that October into the familiar if-else conditional statements that we know now

Conclusion

- First and foremost, use of conditionals was improved (duh)
- Community responded well, as negative comments started to fade away
- Python made a huge leap and was now competitive in relation to more standard languages