



The Camel and The Snake

Jon Allen

<http://perl.jonallen.info> - jj@jonallen.info



Background

- Presented the "Microsoft Scripting Games" talk at Wolverhampton LUG
- Aq and Alex both wrote Python versions of my Perl code
- The differences were very interesting
- So what can we learn?

Recap

- Scripting Games Event 3 - Instant Runoff Winner
- Election with 4 candidates, voters rank them in order of preference
- Count the first place votes
- If no overall winner, disqualify the loser
- Wash, rinse, repeat.

Event 3 - Perl solution

```
use constant FIRST => 0;
use constant LAST => -1;

open VOTES, '<', 'votes.txt' or die "Cannot open votes.txt: $!";
my @votes = map {chomp; [split /,/]} (<VOTES>);

my (@rank,%total);
do {
    undef %total;                                # Clear out previous totals
    $total{$_->[FIRST]}++ foreach @votes;        # Count first choice votes
    @rank = sort {$total{$b} <=> $total{$a}} keys %total;
    foreach my $vote (@votes) {
        $vote = [ grep {$_ ne $rank[LAST]} @$vote ];
    }
} until ($total{$rank[FIRST]} / @votes > 0.5);

printf ("The winner is %s with %f%%\n", $rank[FIRST],
        $total{$rank[FIRST]}/@votes*100);
```

Event 3 – Python solution

```
fp = open("votes.txt")
votes = [x.strip().split(",") for x in fp]

while 1:
    totalvotes = float(len(votes))
    winners     = [x[0] for x in votes]
    counts      = dict([(x,winners.count(x)) for x in winners])
    winner      = [x[0] for x in counts.items() if x[1] ==
                    max(counts.values())][0]
    loser       = [x[0] for x in counts.items() if x[1] ==
                    min(counts.values())][0]
    if (counts[winner] / totalvotes) > 0.5:
        print "%s wins! (%s votes from a total of %s)" % (winner,
            counts[winner], totalvotes)
        break
    else:
        print "No winner this round, removing %s (Votes: %s)" % (loser,
            ', '.join(["%s = %s" % x for x in counts.items()])))
        votes = [filter(lambda k: k != loser, x) for x in votes]
```

Loading the vote data

- Perl

```
open VOTES, '<', 'votes.txt' or die "Error: $!";  
my @votes = map {chomp; [split /,/]} (<VOTES>);
```

- Python

```
fp = open("votes.txt") # Throws exception on error  
votes = [x.strip().split(",") for x in fp]
```

– which can also be written as

```
votes = [x.strip().split(",") for x in open("votes.txt")]
```

How can we do this in Perl?

- Be pragmatic
- If something is better in Python, then why not just write it in Python?

How can we do this in Perl?

- Be pragmatic
- If something is better in Python, then why not just write it in Python?
- OK...

```
use Inline::Python qw/eval_python/;
```

```
my $py = '[x.strip().split(",") for x in open("v.txt")]';  
my $votes = eval_python($py, 0);
```

More Inline::Python

```
use List::Util qw/sum/;

use Inline Python => <<'END_PYTHON';
def read_judges_scores(fname):
    def record(s):
        return s[0], map(float, s[1:])
    return [record(line.strip().split(',')) for line in open(fname)]
END_PYTHON

my $records = read_judges_scores('scores.txt');

foreach my $record (@$records) {
    my $name = $record->[0];
    my $score = sum(@{$record->[1]});
    print "$name: $score\n";
}
```

Your language will be assimilated

- Inline works with many other languages

```
use Inline C =>
    'void hello() { printf("Hello, World!\n"); }';
hello;
```

```
use Inline Java => <<'END_JAVA';
    public class HelloWorld {
        public static void main() {
            System.out.println("Hello, World!");
        }
    }
END_JAVA
```

```
HelloWorld->main();
```

But isn't that cheating?

- Fair point, let's try another approach
- autobox
 - <http://search.cpan.org/dist/autobox>
 - Allows methods to be called on native types (scalars, arrays, hashes, etc)

```
$shout = "Hello, World! "->upper;
```

- This also means we can chain methods

```
30->add(20) ->subtract(8) ->print;
```

How does it work?

- If we write

```
use autobox;  
$shout = "Hello, World!"->upper;  
$shout->print;
```

- Perl will actually do

```
$shout = SCALAR::upper("Hello, World!");  
SCALAR::print($shout);
```

- We need to write the SCALAR:: functions ourselves
 - Or use autobox::Core
 - <http://search.cpan.org/dist/autobox-Core>

autobox for fun and profit

- Python

```
votes = [x.strip().split(",") for x in open("v.txt")]
```

- Perl

```
my @votes = 'v.txt'->open->getlines->strip->split(',');
```

```
@votes = (  
    [  
        'Ken Myer',  
        'Jonathan Haas',  
        'Pilar Ackerman',  
        'Syed Abbas'  
    ],  
    [ etc...  
]
```

Event 3 – fully autoboxed

- See <http://perl.jonallen.info/talks>

```
my @votes = 'votes.txt'->open->getlines->strip->split(',');

my ($rank,%total);
my $compare_votes = sub {$total{$_[1]} <=> $total{$_[0]}};

do {
    undef %total;
    $total{$_->first}++ foreach @votes;

    $rank = %total->keys->sort($compare_votes);
    @votes = map {$_->grep( sub{$_ ne $rank->last} )} @votes;
} until ($total{$rank->first} / @votes > 0.5);

printf ("The winner is %s with %f%%\n", $rank->first,
        $total{$rank->first}/@votes*100);
```

On readability...

- Python programmers will still complain that Perl is unreadable
- But Chinese is unreadable to me
- Maybe we just need to speak in their language

- What if there was a way to simply convert Perl code to Python?
 - Similar to a2p and s2p for Awk and Sed

Acme::Python

- Well of course there is...

```
use Acme::Python;
```

```
print "Hello\n";
```

Acme::Python

- Well of course there is...

```
use Acme::Python;  
Hisssssssssssssssssss  
hiss Hiss hiss Hiss hissssssss Hissss hisss  
Hiss hisss Hissss hiss Hiss hisss Hiss hiss  
Hisss hisss Hissss hiss Hisss hissss Hiss  
hiss Hissss hisssssss Hiss hissss Hiss hissss  
Hiss hissssss Hiss hisss Hiss hiss Hiss hiss  
Hiss hisss Hisss hissss Hisss hiss Hisss  
hissss Hisss hiss Hisss hiss Hissss hiss  
Hisss hissss Hissss hiss Hiss hisss Hissss  
hiss Hisss hisss Hiss hissss Hiss hisss Hisss  
hiss Hissss hissss Hiss hiss Hiss hissss
```

Fin!

Thank you for listening!

Any questions?

<http://perl.jonallen.info/talks>