Perl 6 Design Philosophy

Allison Randal

The Perl Foundation &

Onyx Neon Consulting

Design Philosophy

- Every language is unique
- Design philosophy drives the shape of the language
- Better understand Perl 6
- Better understand all languages
- "Code smells"
- "Language smells"

Simplicity

- Simple is better
- Simple is easier
  - to teach
  - to learn
  - to remember
  - to use
  - to read
  - to parse

Simplicity

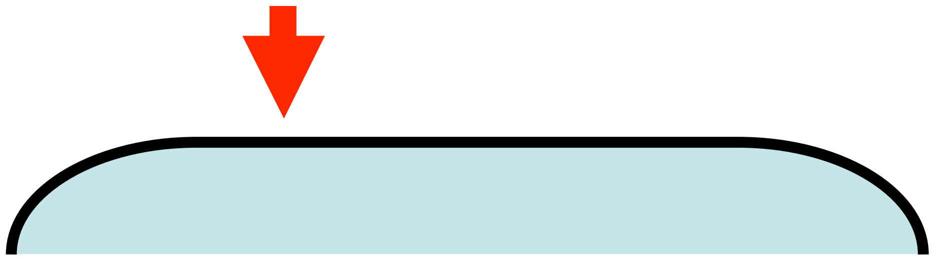
- Not all problems are simple
- "The Unlanguage"
- Choose your complexity

The Waterbed Theory

The Waterbed Theory

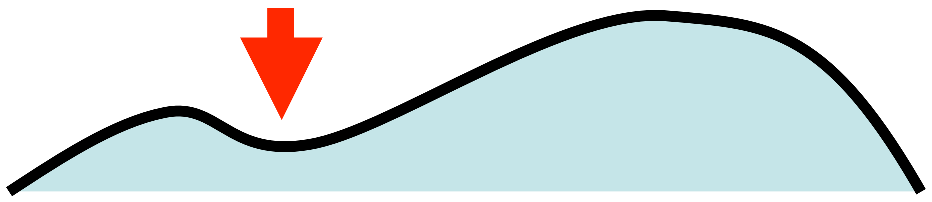
The Waterbed Theory

- Push it down...



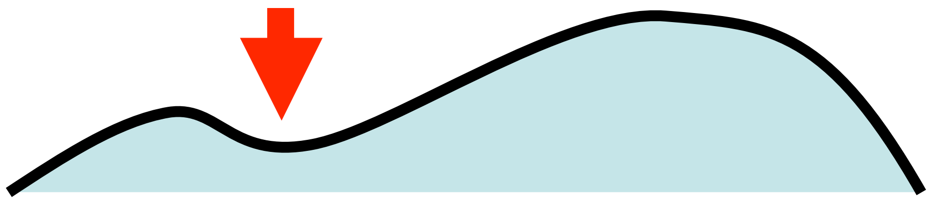
The Waterbed Theory

- Push it down...



The Waterbed Theory

- Push it down...
- ...it rises on the other side



The Waterbed Theory

- Conservation of complexity
- Many operators:

```
^%+ !@== ?/ **~* -_-$
```

- Few operators:

```
assign(a, add(multiply(3,4), 5))
```

- Find the balance

Reuse

- Repeated structures

```
while (true) {
    # do something
}
if (true) then
    # do something
end if
```

- Provides consistency
- Syntactic conventions

Distinction

- Small differences disappear
- If "cats" were "togs"
- Visual clues
  - `eval` and `try`
  - `for` and `loop`
  - `sub` and `method`
- Distinction vs. Reuse

Freedom

- Programmer freedom
- Customs, not laws
- Perl isn't a training bike
- Some things should be hard
- Freedom demands flexibility

Adaptability

- Change is natural
- Adjust to need
- Stay relevant
- Dead languages don't change
- Plan for change
- Features like:
  - modifiable parsing
  - core vs. user-defined
  - user-defined operators

Prominence

- All syntax is not created equal
- Some things stick out
- Use to your advantage
- `BEGIN` blocks (NAMED blocks)
- Modifier forms of `if`, `unless`, etc.

```
if ($blue) {
    print "True Blue.";
}

print "True Blue." if $blue;
```

End Weight

- Lengthy bits at the end

  "I gave Mary the book"

  "I gave the book to Mary"

  "I gave the book about the history of the development of peanut-based products in Indonesia to Mary."

- Easier to read, easier to parse

```
$line =~ m/^phone\s*
          (\d{3})-?
          (\d{3})-?
          (\d{4})$/ix;
```

End Weight

- Lengthy bits at the end

  "I gave Mary the book"

  "I gave the book to Mary"

  "I gave the book about the history of the development of peanut-based products in Indonesia to Mary."

- Easier to read, easier to parse

```
$line ~~ m:i/^phone\s*
         (\d<3>)-?
         (\d<3>)-?
         (\d<4>)$/;
```

DWIM

- Do What I Mean
- DWIM not always DWYM
- Use the DWIM, Luke
- Targets
  - Perl programmers
  - English speakers

    `1st, 2nd, 3rd` also `1th, 2th, 3th`

Borrowing

- Like "camouflage"
- Features for the taking
- Mutual respect
- Open thought
- Adoption with Adaptation

Perl Should Stay Perl

- What makes it Perl?
- True to designer's purpose
- Familiar
- Translatable

Long-Term Usability

- Not 2 years
- 20+ years
- Not fads or cute tricks
- Strong, dependable tools
- Not perfect, just a step

Summary

- Adaptable
- Freeing
- Simple...
- ...but complex
- Distinct...
- ...but consistent
- Borrow...
- ...but Perl-ishly

Summary

- Relevant now...
- ...and relevant then
- Programmer friendly
- More Perl

Questions?