5th State of the Onion

Larry Wall
Talk 1
An Overview of Perl

- Perl 5
- Perl 6
Talk 2
Bits and Pieces

- 33 lightning talks in half an hour
- See Apocalypse 2
- Expected apocalyptic error rate: 5%
Talk 3
Unary and Binary Operators

- Doing the Right Think
- Precedence level reform?
- User-defined operators (including Unicode)
- $\texttt{foo->bar}$ becomes $\texttt{foo.bar}$
- $\texttt{foo . $bar}$ becomes $\texttt{foo ~ $bar}$
- Scalar operators will work on array values
Talk 4
Statements and Declarations

- Switch statement--see Damian
  
  ```
  given $foo {
      when 1: { argle; }
      when "a": { bargle; }
  }
  ```

- `local` renamed to `temp`

- Optional type and property declarations
Talk 5
Pattern Matching

- Assignment to variables
- Assignment to elements of arrays or hashes
- Set operations on character classes
  
  /[[a-z]&&[^aeiou]]/
- \x modifier on by default?
- Matching on pseudo-strings
Talk 6
Subroutines

• Complete type signatures
• Discarding sub keyword on closures
• Self-identifying parameters
  \[ \text{sort} \{ \$^a \text{ cmp } \$^b \} \text{ @array} \]
• Pre-handlers and post-handlers--see Damian
Talk 7
Formats

- Kicked out of the core
- See Damian
Talk 8
References

- Pseudo-hashes must die
- Deref always assumed by [], {}, and ()
  
  **Perl 5**
  
  $foo->[bar]  $foo[$bar]
  
  **Perl 6**
  
  $foo[$bar]  @foo[$bar]

- Whitespace before hash subscript
  
  $foo .{$bar}
Talk 9
Data Structures

• Opaque objects
• Compact multidimensional arrays
• Slicing specified within the subscript
• Pairs, composed with \( \Rightarrow \)
• Ranges, composed with \( \ldots \)
• Superpositions? See Damian
Talk 10
Packages

• Split up into classes and modules
• Within class or module, subpackages
• Generalized autoloading
Talk 11

Modules

- Extended module names: version and author
- Interface modules
- Module searching
- Better metadata for automation
- Default import to lexical scope?
- Wildcard modules?
Easy declaration, accessible metadata
- Attributes declared as variables
- Attributes accessed as methods externally
- Foreign objects handled gracefully
- Optional multimethods
- `Class.bless($ref)` or `$obj.bless($ref)`
Talk 13
Overloading

• Can’t live with it, can’t shoot it.
  ```cpp
cout << foo << bar << bletch;
  ```
• Overloaded operator specified in method name
• Overloading of methods as well as operators
• Implementation via vtables
• Overloading hooks in `printf` for bignums, etc.
Talk 14
Tied Variables

• Variable type, not value type!!!
• Naturally scoped to variable lifetime
• Needs to be declared for efficiency
Talk 15
Unicode

- Compassion
- Polymorphic strings
- Abstract and concrete internal routines
- Normalization normally at the filehandle
- Type system must remember normalization
- Do the Hard Thing even if it’s Right
Talk 16
Interprocess Communication

- “No pain” installation of new protocols
- Easy mapping of high-level structured data from the network onto Perl’s data structures
- Reliable low-level primitives, safe signals
- IPv6, coming to a circus near you
Talk 17
Threads

- Basic model is I-threads
- Variables may be shared by declaration
- P-threads just means “share everything”
- Modules should be thread safe by default
- Module thread safety should be in metadata
- With or without threads, we can do better at event-based programming
Talk 18
Compiling

- A parser of our own
- Mutability
- Portability of eval to VMs
- Scoped mutations, syntactic delegation
- One-pass lexical analysis
- Immediate subroutines
Talk 19
The Command-Line Interface

- 1 RFC, proposing -r switch
- The life of a glue language--compete by cooperating with the environment
Talk 20
The Perl Debugger

- Happy to delegate this to others
- Heavy dependency on the debugging facilities of the platform (without code rewriting)
- IDEs have their own ideas
Talk 21
Internals and Externals

- More delegation--see Dan’s talk on Thursday
- Very modular
- A software CPU
- Regexen compile to normal opcodes
- Garbage collection
- Vtables
- Internal abstract APIs
Talk 22
CPAN

- Too big to download
- ISPs don’t install enough of it
- Bundles only a partial solution
- Renewed interest in SDKs
- Public rating system?
Talk 23
Security

- Tainting via new property mechanism
- Sandboxing via I-threads
- Controlling easy-to-abuse syntax
- Eternal vigilance on buffer overruns, etc.
Talk 24
Common Practices

• Eliminating (some) Common Goofs

```perl
foreach $i (1 .. 1_000_000_000) {...}
@big = 1 .. 1_000_000_000;
```

• RFC 183: “=for testing” - Embedded tests
Talk 25
Portable Perl

- Full URI support
- Full Unicode support, including dwimmery
- Easy identification of non-portable code
- Module metadata could indicate portability
Talk 26
Plain Old Documentation

• Fixing \texttt{begin/end} for commenting
• Multiple POD streams
• DATA filehandle just another POD stream
• Autodocs from module metadata (not necessarily POD)
• Big idea: equivalent of “\texttt{use}” for POD
Talk 27
Perl Culture

- Mostly self-correcting at this point
- Do your part
- Newbie friendliness
- Licensing cleanup
- Testing, testing, testing
Talk 28
Special Names

• Balance cleanup with convenience
  \$_ stays
  $( goes
• $() now means to interpolate an expression
• No more bareword filehandles
• Merge error status variables?
Talk 29
Functions

• Dealing with long return lists
• Array ops like merge, unmerge, part, flatten, reduce...
• Logical return values from index, rindex, system, wait, waitpid
• Call a method a method
• Get rid of select
Talk 30
The Standard Perl Library

- Definitions of Standard become mushy
- Cut core down to almost nothing, and force installation of a more complete SDK
- Multiple such SDKs are possible, just as with Linux distributions
- SDK editors earn trust by recommending reliable sets of modules
Talk 31
Pragmatic Modules

- More capacity to warp syntax and semantics
- But be careful with those semantics!!
- Real optimization options
Talk 32
Standard Modules

• This may be a very small chapter in the next edition of the Camel.
• Theoretically, we need just the CPAN module, and its dependencies
• Practically...
Talk 33
Diagnostic Messages

- I18n and L10n
- Exceptional exception handling
- Typed exception objects
- Unthrown exceptions: “false but interesting”

```perl
$handle = open "nonesuch" or die;
```
5th State of the Onion

Larry Wall