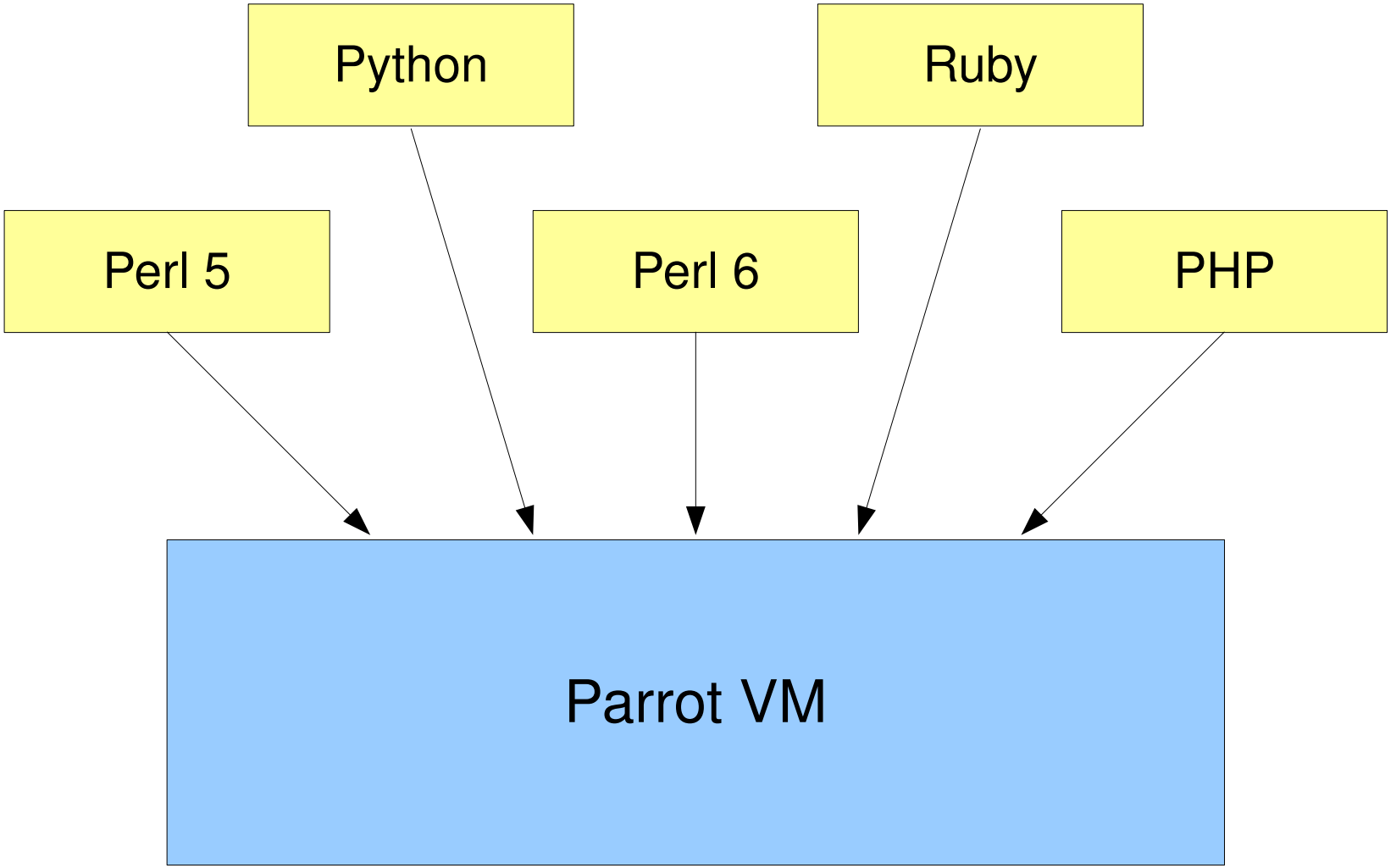
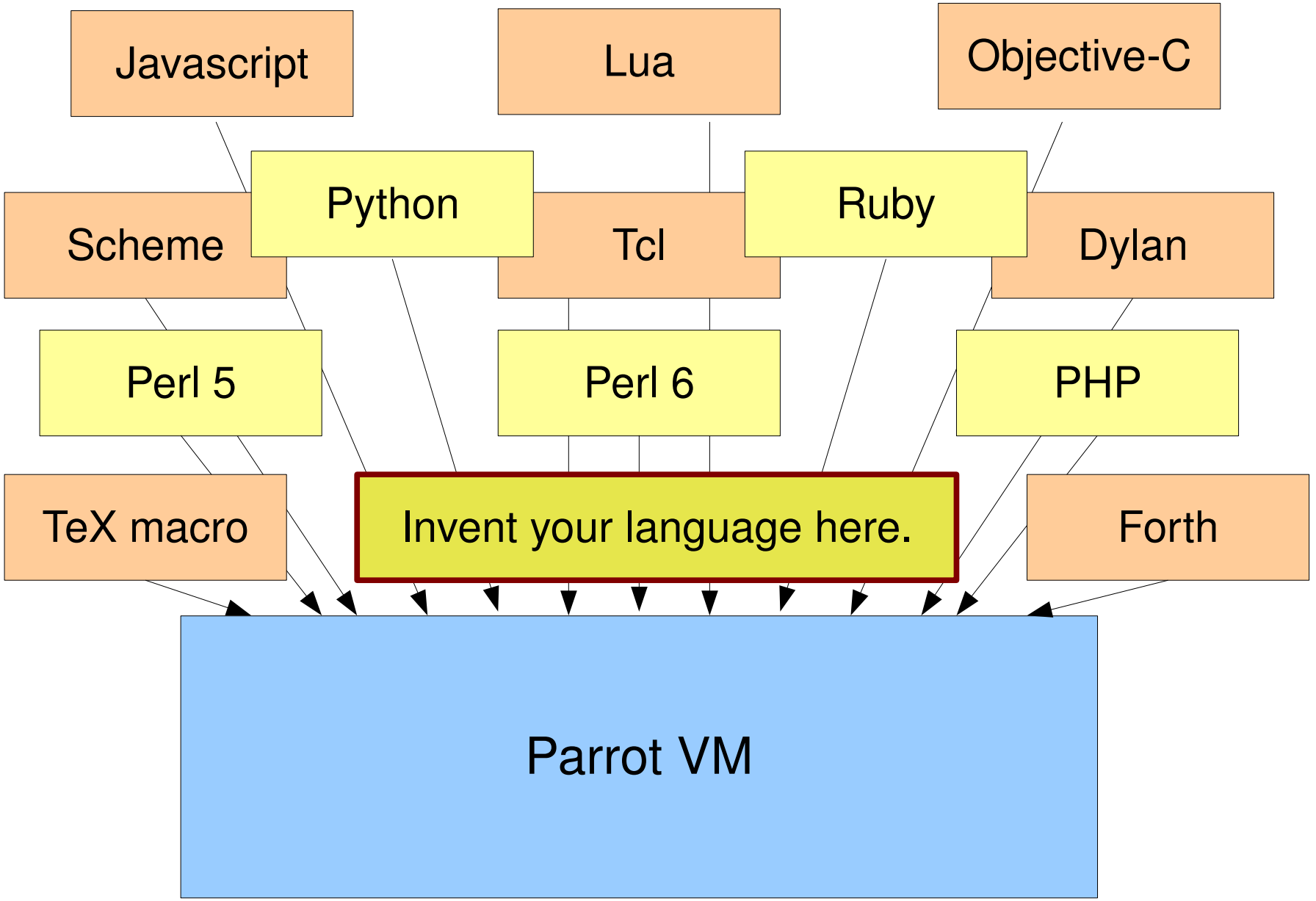
Parrot VM

Allison Randal
Parrot Foundation &
O'Reilly Media, Inc.


Python
Ruby
Perl 5
Perl 6
PHP
Parrot VM


Javascript
Lua
Objective-C
Python
Ruby
Scheme
Tcl
Dylan
Perl 5
Perl 6
PHP
TeX macro
Invent your language here.
Forth
Parrot VM

Dynamic Features

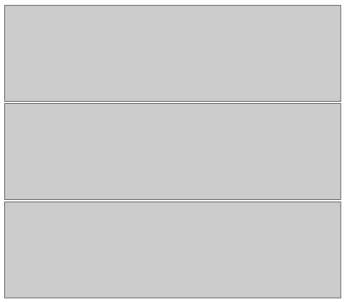
- Runtime vs. compile-time
- Extend code (eval, load)
- Define classes
- Alter type system
- Higher-order functions
- Closures, Continuations, Coroutines

Goals

- Revolution...
- ...becoming Establishment
- Powerful tools
- Portability
- Interoperability
- Language evolution

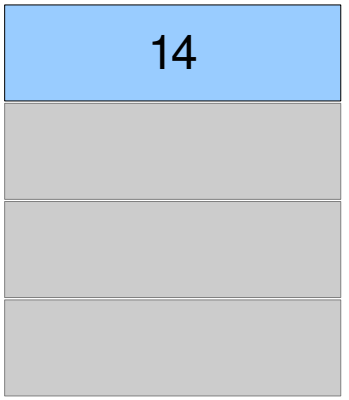
Register-based

- Stack operations



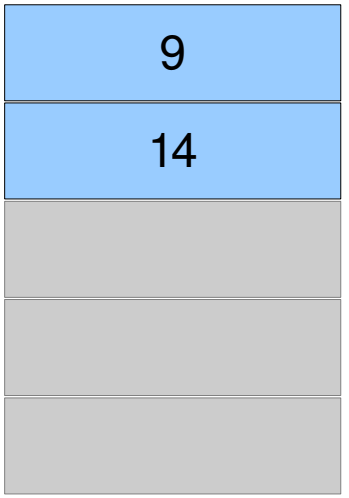
Register-based

- Stack operations



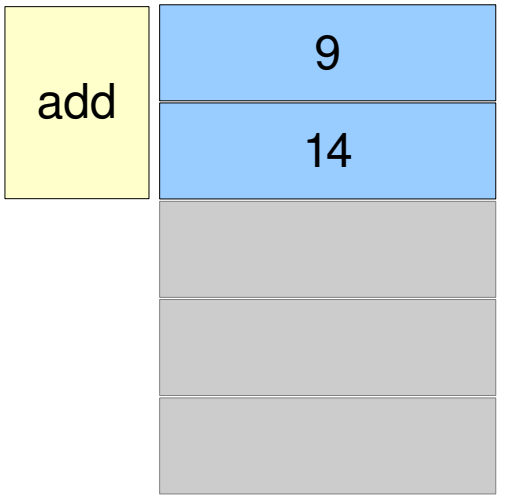
Register-based

- Stack operations



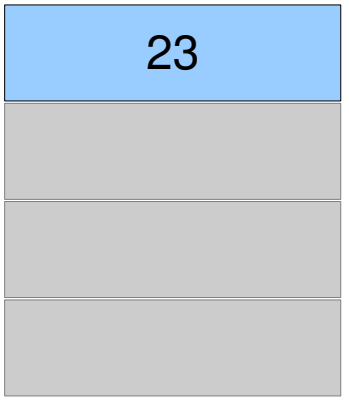
Register-based

- Stack operations



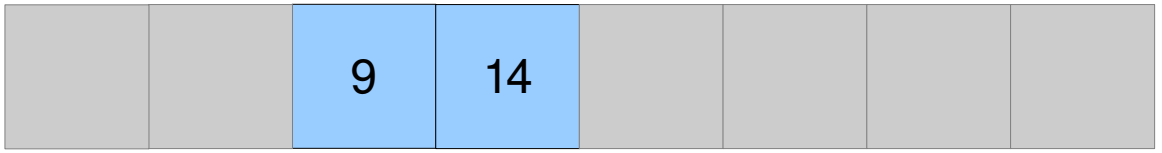
Register-based

- Stack operations



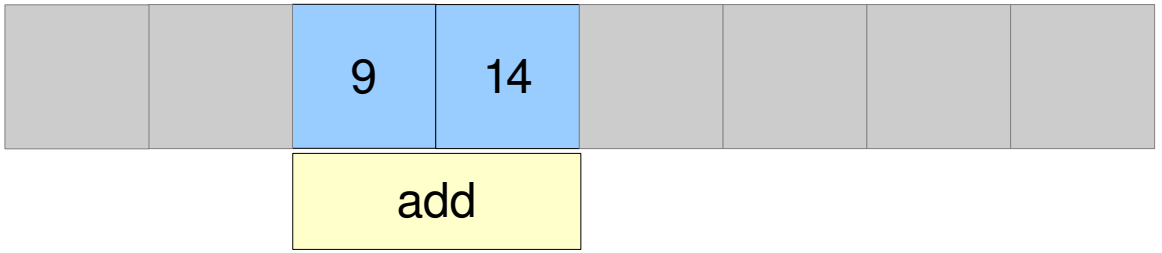
Register-based

- Stack operations
- Register operations



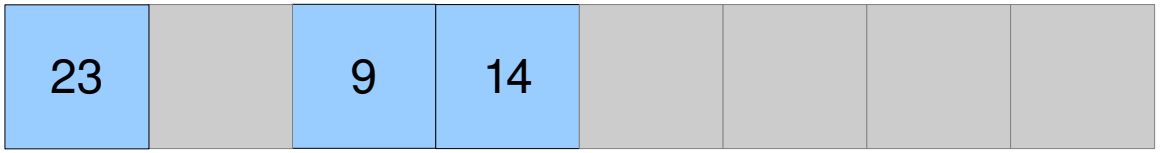
Register-based

- Stack operations
- Register operations



Register-based

- Stack operations
- Register operations



Register-based

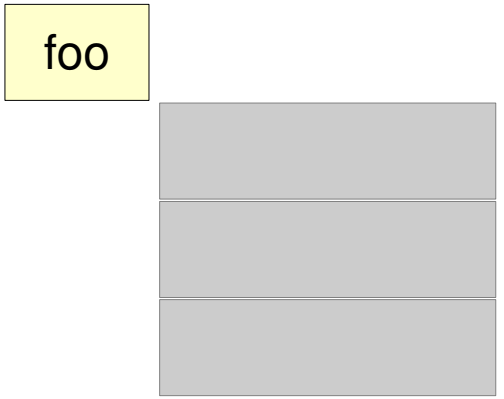
- Stack operations
- Register operations
- Fewer instructions
- Hardware registers
- Register spilling
- Flexible register sets

Continuation Passing Style

- Stack-based control flow

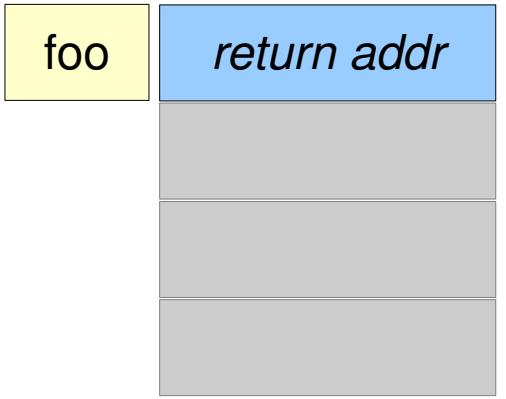
Continuation Passing Style

- Stack-based control flow



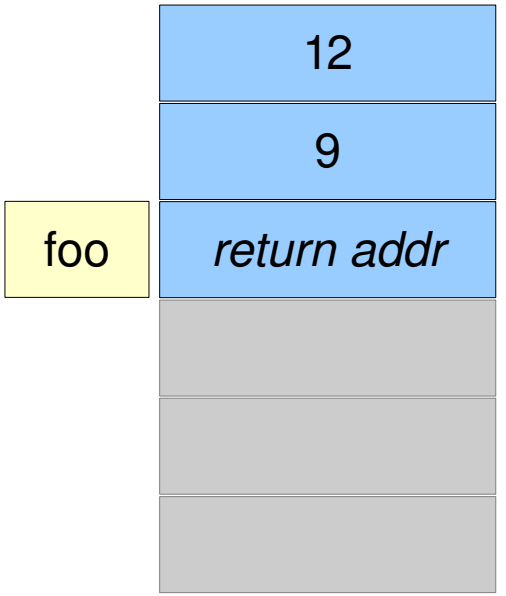
Continuation Passing Style

- Stack-based control flow



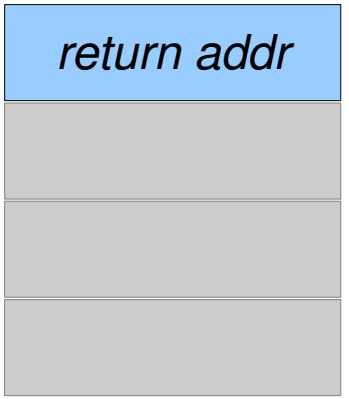
Continuation Passing Style

- Stack-based control flow



Continuation Passing Style

- Stack-based control flow

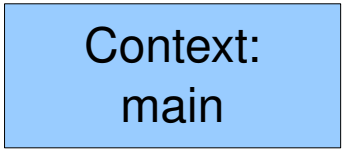


Continuation Passing Style

- Stack-based control flow

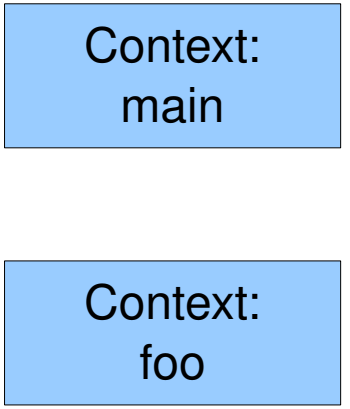
Continuation Passing Style

- Stack-based control flow
- Continuation-based control flow



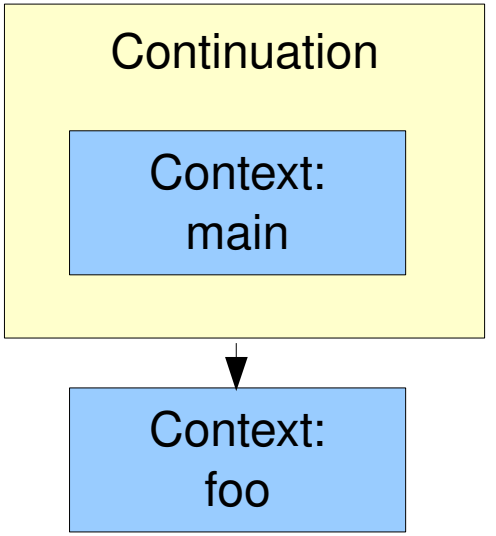
Continuation Passing Style

- Stack-based control flow
- Continuation-based control flow



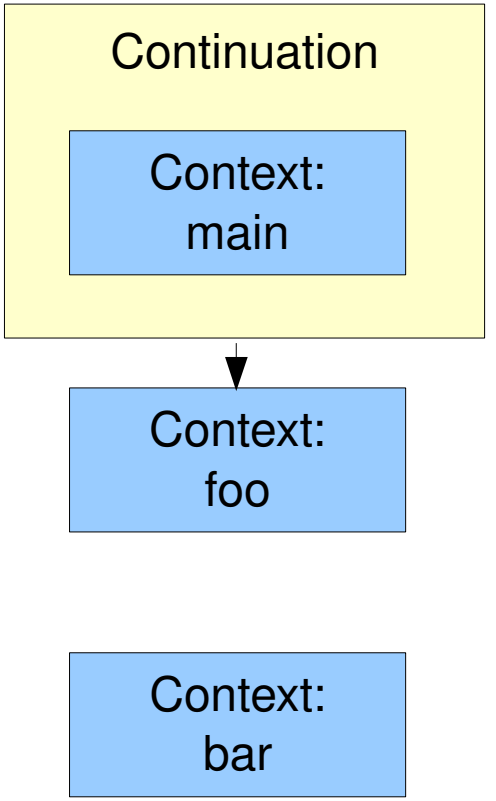
Continuation Passing Style

- Stack-based control flow
- Continuation-based control flow



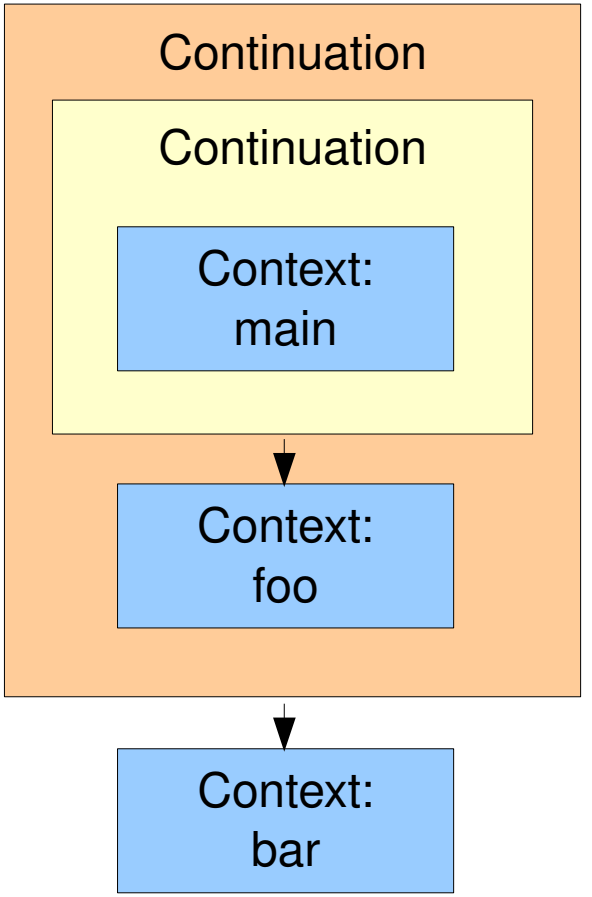
Continuation Passing Style

- Stack-based control flow
- Continuation-based control flow



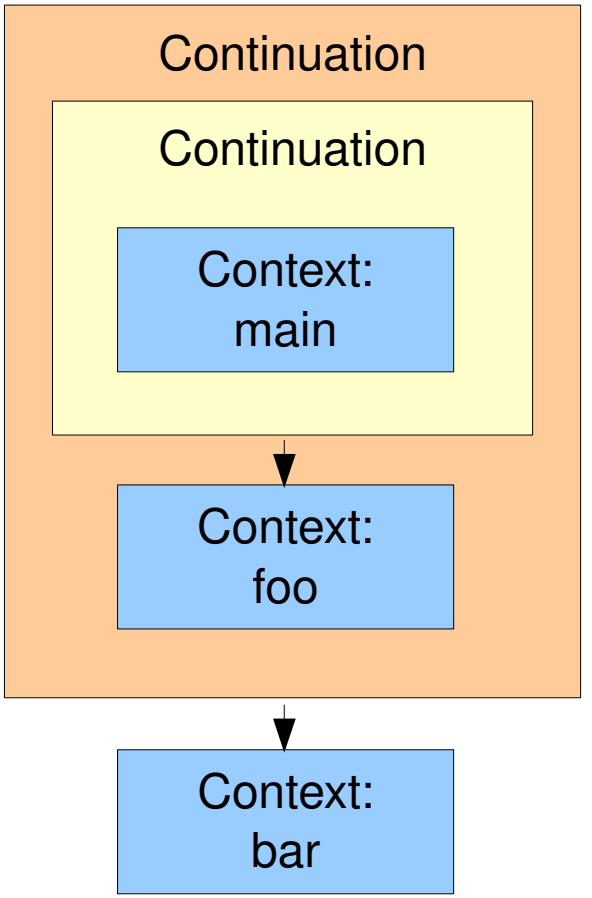
Continuation Passing Style

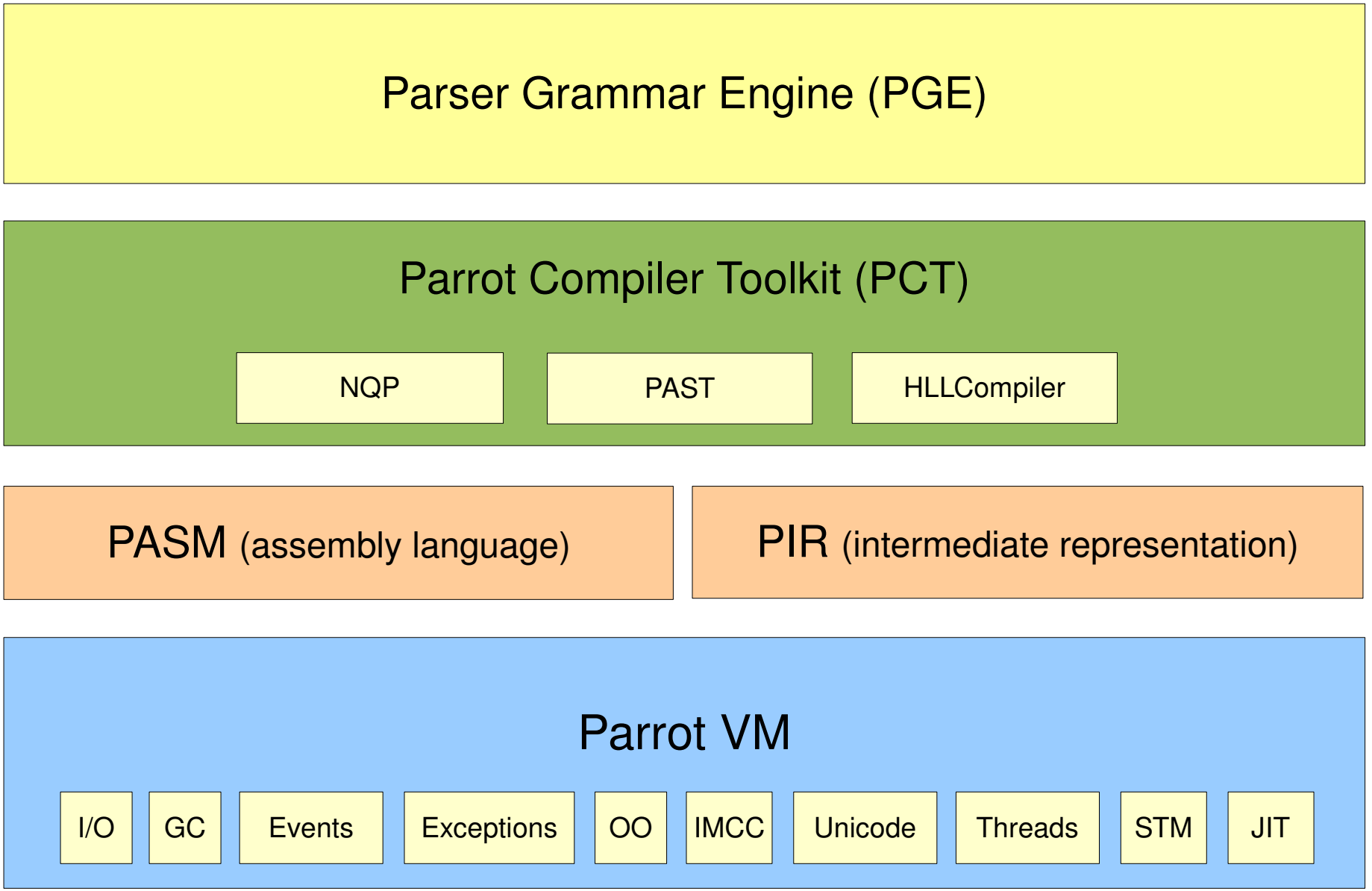
- Stack-based control flow
- Continuation-based control flow



Continuation Passing Style

- Stack-based control flow
- Continuation-based control flow
- Linked list
- Cheap continuations
- Deeply nested contexts
- Tail recursion




Parser Grammar Engine (PGE)
Parrot Compiler Toolkit (PCT)
NQP
PAST
HLLCompiler
PASM (assembly language)
PIR (intermediate representation)
Parrot VM
I/O
GC
Events
Exceptions
OO
IMCC
Unicode
Threads
STM
JIT

Languages

- Core VM = C
- PASM/PIR parser = lex/yacc (flex/bison)
- Build tools = Perl/Parrot
- Compiler tools = PIR

Squaak

- Download

```
http://www.parrot.org
```

- Build

```
$ perl Configure.pl
$ make test
```

- Language

```
$ cd examples/languages/squaak
$ make squaak
$ make test
```

Squaak

- hello.sq

```
print("Hello, World!")
```

- Run

```
$ ./squaak hello.sq
```

MyLang

- Create

```
$ perl mk_language_shell.pl MyLang
```

- Run

```
$ cd mylang
$ make test
```

- Extend

```
$ vim src/parser/grammar.pg
```

Compiler Tools Tutorial

- In the Parrot distribution:

  `examples/languages/squaak/doc`

Questions?

- Get involved at parrot-dev@lists.parrot.org or #parrot on irc.parrot.org