

# Monte: Building a programming language using RPython

Corbin Simpson

October 10, 2015 (PyDX '15)

# Hi!

Hi!

Some Pre-talk

Thanks

The Python Side

Constant Sadness

Enter Monte

RPython is not  
Python

The Tombstones  
of Terror

Typhon

Final Thoughts

I'm Corbin!

What you don't need to know:

- ✓ Monte
- ✓ RPython
- ✓ Lojban

You don't **need** to know compilers, but it will help.

# Some Pre-talk Thanks

Hi!

Some Pre-talk  
Thanks

The Python Side  
Constant Sadness  
Enter Monte

RPython is not  
Python

The Tombstones  
of Terror

Typhon

Final Thoughts

- ✓ Allen, for getting me to build Typhon
- ✓ The original implementors of E
- ✓ Em and Mike, for trailblazing
- ✓ Dan Connelly for making awesome documentation (with diagrams!)
- ✓ Everybody else in #monte and #erights

# The Python Side

Hi!  
Some Pre-talk  
Thanks  
**The Python Side**  
Constant Sadness  
Enter Monte  
RPython is not  
Python  

---

The Tombstones  
of Terror  

---

Typhon  

---

Final Thoughts

- ✓ Twisted
- ✓ PyPy

# Constant Sadness

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
**Constant Sadness**  
Enter Monte  
RPython is not  
Python  

---

The Tombstones  
of Terror  

---

Typhon  

---

Final Thoughts  

---

- ✓ People don't use Twisted
- ✓ People don't use PyPy

# Enter Monte

- ✓ Has syntactic and semantic features equivalent to always having Twisted available
- ✓ Reference implementation is built using PyPy's toolchain, **RPython**

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
**Enter Monte**

RPython is not  
Python

The Tombstones  
of Terror

Typhon

Final Thoughts

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte

## RPython is not Python

What's RPython,  
Anyway?

Why RPython?

Wait, What

RPython  
Alternatives

The Tombstones  
of Terror

---

Typhon

---

Final Thoughts

---

# RPython is not Python

# What's RPython, Anyway?

- ✓ PyPy team: “RPython is a restricted subset of Python amenable to static analysis.”
- ✓ Allen: “RPython is OCaml with a very odd syntax and a very odd standard library.”
- ✓ RPython is a toolchain that can translate carefully crafted Python packages into highly efficient low-level code.

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python  
**What's RPython,  
Anyway?**  
Why RPython?  
Wait, What  
RPython  
Alternatives  
The Tombstones  
of Terror  

---

Typhon  

---

Final Thoughts



# Why RPython?

You have to be writing an interpreter for a language; RPython is not easy to use.

If nothing else, know this: RPython turns interpreters into JIT compilers at a steep engineering discount.

- Hi!
- Some Pre-talk
- Thanks
- The Python Side
- Constant Sadness
- Enter Monte
- RPython is not Python
- What's RPython, Anyway?
- Why RPython?**
- Wait, What
- RPython
- Alternatives
- The Tombstones of Terror
- Typhon
- Final Thoughts

# Wait, What

Programs written in RPython are imported and then transformed from Python into a high-level statically-typed statically-named form (“translation”). This form is augmented:

- ✓ GC: A garbage collector, also written in RPython, is hooked in. The GC can be chosen at translation time.
- ✓ JIT: Based on JIT annotations written by hand, the program is turned into a JIT compiler which functions automatically and is correct independent of your program.

RPython finally performs some optimizations (malloc removal, generated switches, etc.) and emits an executable.

- Hi!
- Some Pre-talk
- Thanks
- The Python Side
- Constant Sadness
- Enter Monte
- RPython is not Python
- What's RPython, Anyway?
- Why RPython?
- Wait, What**
- RPython Alternatives
- The Tombstones of Terror
- Typhon
- Final Thoughts

# RPython Alternatives

**LLVM** Mediocre GC control, poor JIT technology, and writing C++ drains my soul

**GHC Haskell** Speedy and terse but mutation is hard

**Bare/NIH C++** Speedy but very difficult to write, read, and maintain

**Truffle** Very powerful but was not mature when I started

- Hi!
- Some Pre-talk
- Thanks
- The Python Side
- Constant Sadness
- Enter Monte
- RPython is not Python
- What's RPython, Anyway?
- Why RPython?
- Wait, What
- RPython Alternatives**
- The Tombstones of Terror
- Typhon
- Final Thoughts

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python

---

## The Tombstones of Terror

Introducing  
Tombstones  
Untranslated &  
Translated  
The Missing  
'Stone

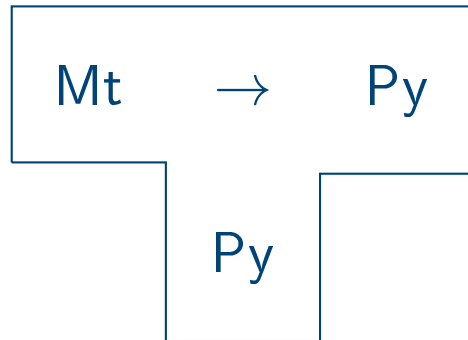
Typhon

Final Thoughts

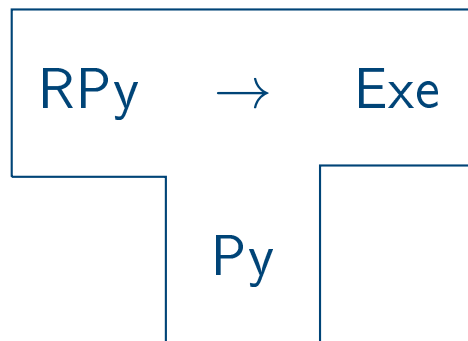
# The Tombstones of Terror

# Introducing Tombstones

The old Monte compiler:



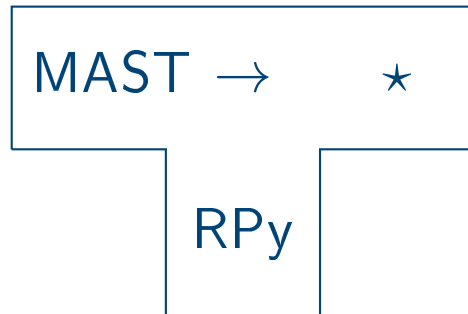
The RPython translator:



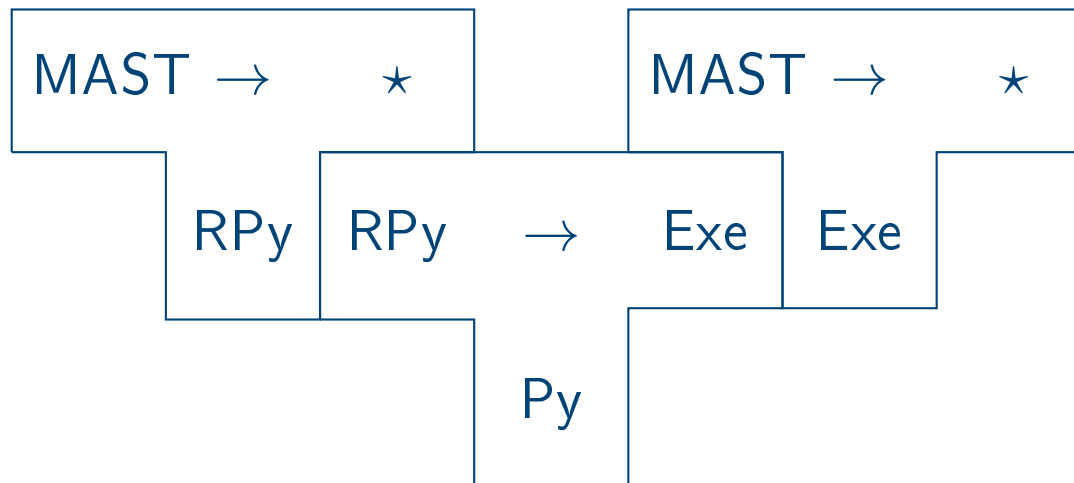
# Untranslated & Translated

- Hi!
- Some Pre-talk
- Thanks
- The Python Side
- Constant Sadness
- Enter Monte
- RPython is not Python
- The Tombstones of Terror
- Introducing Tombstones
- Untranslated & Translated**
- The Missing 'Stone
- Typhon
- Final Thoughts

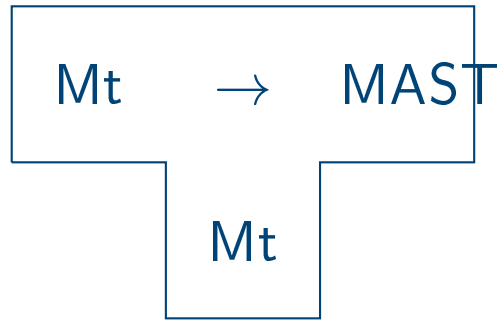
Typhon untranslated:



Typhon translated:



# The Missing 'Stone



Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python

The Tombstones  
of Terror

Introducing  
Tombstones  
Untranslated &  
Translated

The Missing  
'Stone

Typhon

Final Thoughts

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python

---

The Tombstones  
of Terror

---

## Typhon

It's Like an Onion  
AST Loader  
Bytecode Compiler  
Object Model  
JIT Annotations  
JIT Annotations,  
Cont.  
Vats  
FFI

Final Thoughts

# Typhon



# It's Like an Onion

- ✓ AST loader
- ✓ AST-to-bytecode compiler
- ✓ Object model
- ✓ JIT annotations
- ✓ Vats and libuv

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python

The Tombstones  
of Terror

Typhon  
**It's Like an Onion**

AST Loader  
Bytecode Compiler  
Object Model  
JIT Annotations  
JIT Annotations,  
Cont.  
Vats  
FFI

Final Thoughts

# AST Loader

The AST loader loads files into an in-memory AST.

- Hi!
- Some Pre-talk
- Thanks
- The Python Side
- Constant Sadness
- Enter Monte
- RPython is not Python
- The Tombstones of Terror
- Typhon
- It's Like an Onion
- AST Loader**
- Bytecode Compiler
- Object Model
- JIT Annotations
- JIT Annotations, Cont.
- Vats
- FFI
- Final Thoughts

# Bytecode Compiler

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python

---

The Tombstones  
of Terror

---

Typhon

---

It's Like an Onion  
AST Loader

**Bytecode Compiler**

Object Model

JIT Annotations  
JIT Annotations,  
Cont.

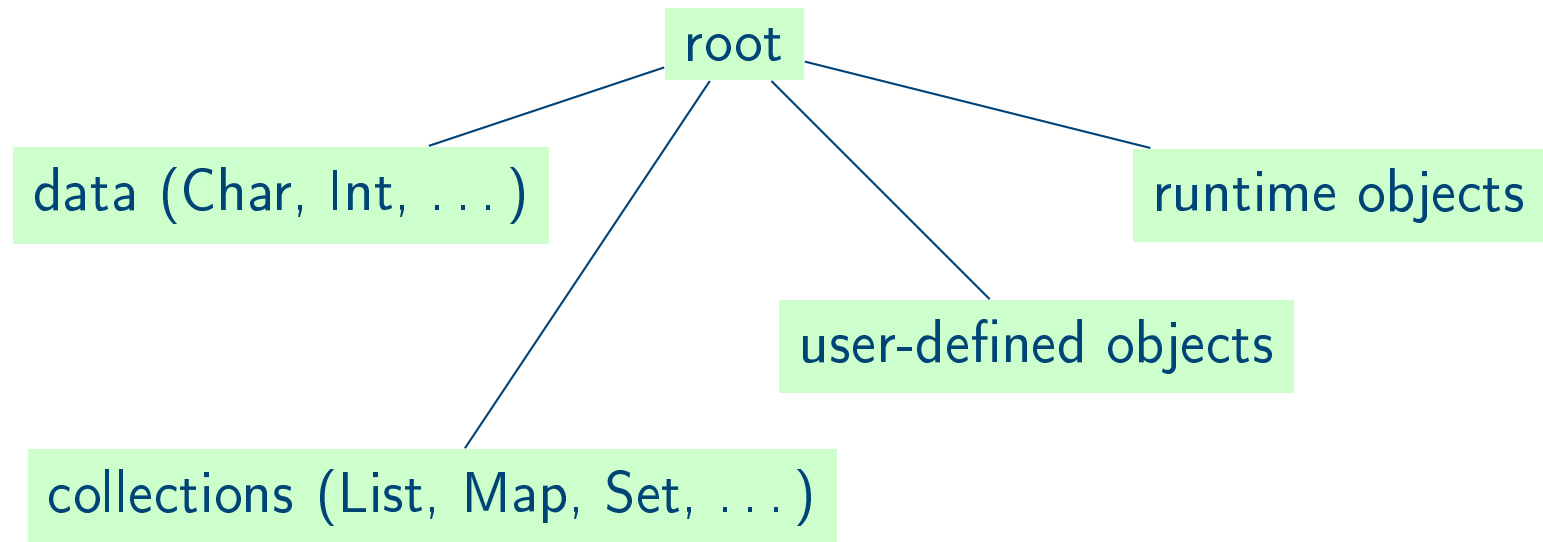
Vats

FFI

Final Thoughts

- ✓ Compiles to VM based on SmallCaps for E
- ✓ Compiler implemented as AST visitor
- ✓ Simple semantics: no loops, one test, exception mini-stack
- ✓ Compiler lowers static names to frame indices, turning dicts into lists
- ✓ Unstable design: Completely internal to Typhon (so it can be changed as needed)

# Object Model



Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte

RPython is not  
Python

The Tombstones  
of Terror

Typhon

It's Like an Onion  
AST Loader

Bytecode Compiler

**Object Model**

JIT Annotations  
JIT Annotations,  
Cont.

Vats

FFI

Final Thoughts

# JIT Annotations

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python

---

The Tombstones  
of Terror

---

Typhon

---

It's Like an Onion  
AST Loader  
Bytecode Compiler  
Object Model

**JIT Annotations**  
JIT Annotations,  
Cont.  
Vats  
FFI

---

Final Thoughts

JIT's typical usage:

1. At every JIT merge point, check whether the current code is being executed often ("hot")
2. When code is hot, trace actions **of the interpreter** from one merge point to the next
  - (a) When value or type discrimination occurs, **guard** the chosen branches
  - (b) Trace through function calls (free inlining)
3. Optimize the trace, removing superfluous operations
4. When guards fail, retrace from the failed guard and create **branches**

# JIT Annotations, Cont.

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte

RPython is not  
Python

The Tombstones  
of Terror

Typhon

It's Like an Onion  
AST Loader  
Bytecode Compiler  
Object Model

JIT Annotations  
JIT Annotations,  
Cont.

Vats

FFI

Final Thoughts

- ✓ The JIT needs to know which loops to trace. Place `jit_merge_point` annotations at the head of each (user-level) loop.
- ✓ Any interpreter-level loop causes a function to be opaque to the JIT; it will be called and not traced. To fix this, use `unroll_safe` or `look_inside_iff`, but be wary of code explosion.
- ✓ Some functions need to be opaque for performance or to avoid JIT unsafety. They are marked `dont_look_inside`.
- ✓ Some functions need to be opaque to preserve referential transparency. They are marked `elidable`. There's also `_immutable_fields_` for attributes.
- ✓ “The trick”: The JIT colors values as **red** or **green** depending on whether they are constant during the trace. `promote(red)` returns a **green** value; the JIT usually emits a guard.

# Vats

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python

---

The Tombstones  
of Terror

---

Typhon

---

It's Like an Onion  
AST Loader  
Bytecode Compiler  
Object Model  
JIT Annotations  
JIT Annotations,  
Cont.

**Vats**

FFI

Final Thoughts

- ✓ Monte's threads of control are called **vats**.
- ✓ Objects live in vats.
- ✓ Vats send messages to other vats; objects send messages to other objects. All vats are independently managed.
- ✓ Concurrency and parallelism should be a snap! Just gotta reinvent Twisted, which isn't RPython-compatible...

# FFI

- ✓ RPython has FFI for C (“rffi”)
- ✓ Violates Monte’s safety guarantees, so only used sparingly (sorry libpng)
- ✓ libuv
- ✓ libsodium, someday in the future

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python

---

The Tombstones  
of Terror

---

Typhon

---

It's Like an Onion  
AST Loader  
Bytecode Compiler  
Object Model  
JIT Annotations  
JIT Annotations,  
Cont.  
Vats

**FFI**

Final Thoughts



Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python

---

The Tombstones  
of Terror

---

Typhon

---

**Final Thoughts**  
Should you use  
RPython?  
Questions?

# Final Thoughts

# Should you use RPython?

If you're writing an interpreter, and you want it to go fast, and you don't want to write Java or C++, then yes, consider it.  
RPython's competitors are generally worth considering too.

Hi!  
Some Pre-talk  
Thanks  
The Python Side  
Constant Sadness  
Enter Monte  
RPython is not  
Python  
The Tombstones  
of Terror  
Typhon  
Final Thoughts  
Should you use  
RPython?  
Questions?

# Questions?

Thanks!

Hi!

Some Pre-talk

Thanks

The Python Side

Constant Sadness

Enter Monte

RPython is not  
Python

The Tombstones  
of Terror

Typhon

Final Thoughts

Should you use  
RPython?

Questions?