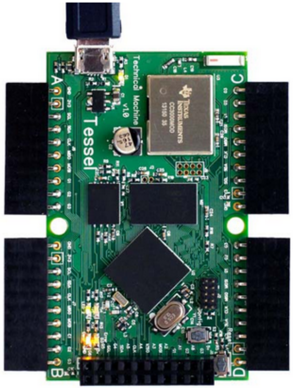
Strong and Static Typing vs Weak and Dynamic Typing

Joe Wass @joewass

October 17, 2013

tessel.io



TESSEL

Hardware development for software developers.

Pre-order now
Ships spring 2014.

Tessel is a microcontroller that runs JavaScript.
Use it to easily make physical devices that connect to the web.

Are we heading for a generation of JS-only programmers?

What I've been doing

Scala

JavaScript

Obj-C

Ruby

C#

Prolog

Java

BASIC

Python

PHP

Pascal

Go

C

Haskell

What I've been doing

Scala

JavaScript

Obj-C

Ruby

C#

Prolog

Java

BASIC

Python

PHP

Pascal

Go

C

Haskell

Error Messages from Python

```
ImproperlyConfigured: Your STATICFILES_DIRS setting
is not a tuple or list perhaps you forgot a trailing
comma?
```

```
AttributeError: ‘NoneType’ object has no attribute
‘get_file’
```

Error Messages from a Compiler (Go)

```
./main.go:61: writer.Header.Add undefined
(type func() http.Header has no field or method Add)
```

What is type safety?

Strong
There is a type system.

Weak
There isn't really a type system.

Static
Type checking at compile time.

Dynamic
Type checking at run time.

My thoughts

- You know the properties of your program.
- Why not do it **in the language**?(The alternative is comments.)
- Who is more trustworthy? You or the machine?
- Ever caught yourself writing Hungarian notation?
- Dynamic typing $\not\propto$ keyboard typing. There's more to static typing than explicit typing.

My thoughts

```
def get_key(self, key_name, headers=None, version_id=None,
            response_headers=None):
  """
  Check to see if a particular key exists within the bucket.  This
  method uses a HEAD request to check for the existance of the key.
  Returns: An instance of a Key object or None

  :type key_name: string
  :param key_name: The name of the key to retrieve

  :type response_headers: dict
  :param response_headers: A dictionary containing HTTP headers/values
   that will override any headers associated with the stored object
   in the response. See http://goo.gl/EWOPb for details.

  :rtype: :class:`boto.s3.key.Key`
  :returns: A Key object from this bucket.
  """
```

But I like it!

- Unit tests
- What if an obscure bit of your library breaks?
- Why reinvent the wheel?
- Who does TDD?
- 100% code coverage?
- Type safety is 100%
- Who refactors?

Refactoring

Can you ever really be sure that your refactor went OK?
Really?
What about those `kwargs`?
Everyone needs to refactor from time to time!

But it’s Cubmersome!

- Java interfaces are brittle!
- Java isn’t the best example!
- It leads to ridiculous design patterns!
- Have you seen the source of Django?

Go's anonymous interfaces

In Python:

```
def licenseFromRow(row):
    # ...
```

In Go:

```
func licenseFromRow(
   row interface {Scan(dest ...interface{}) error}
) (*License, error)
```

Variable declaration

In Python:

```
x = "hello"
x = 5
```

In Go:

```
x := "hello"
x = 5 // type error
```

Option types from Scala

- No time to talk about!
- But it's great!

PHP Type Juggling

- No time to talk about!
- But it's an abomination!

Fin

Thanks for listening
@joewass
afandian.com

Read this Steve Yegge rant if you're interested.
`https://plus.google.com/u/0/110981030061712822816/posts/KaSKeg4vQtz`