API Design is Hard

By Dave Halter
@davidhalter on Github
@jedidjah_ch on Twitter

Me

- Creator of Jedi and jedi-vim
- Both have ~ 2000 stars on Github
- Starting to really like clean code!
- Of course some parts of the API of Jedi suck.

Influences & Inspirations

- API Design: Lessons Learned by Raimond Hettinger
- Good API Design by Alex Martelli

Writing clean code

- Clean Code / Good Architecture
- Testing (py.test/tox)
- Documentation (sphinx)
- Code Reviews

API

- “Application Program Interface”

- Let's just talk about Python “interfaces”.
- Are there interfaces in Python?

API

- “Application Program Interface”

- Let's just talk about Python “interfaces”.
- Are there interfaces in Python?
- Yes: “abc.ABCMeta”

Bad APIs #1

- No API
- But everything has an interface

Bad APIs #2

Going for both, it shouldn't be possible to write both:

```
jedi.names(source)
jedi.Script(source).names()
```

Decide!

Bad APIs #3

- Inconsistency
- Not following standards like

```
class Foo(object):
    def getRange(self):  # Java style
        return self._range
```

But: BeautifulSoup 3 was still awesome.

Think!

- Brainstorm – Design / Performance
- Think about data types
- Don't do IO that is not readable by other languages, like pickle.
- Simple is better than complex.

  - PEP 20: The Zen of Python

Be conservative!!!

Private/Protected/Public

- _variable for protected
- __variable for private (don't use it a lot though)
- Use _ a lot!

Named Arguments

- What is this doing:

```
twitter_search('python', 3, False)
```

- Way better:

```
twitter_search('python', num_results=3, retweets=False)
```

- In Python 3:

```
def twitter_search(name, *, num_results=20, retweets=False):
```

Properties

- Use them, but only for clear defined “getters”:

```
@property
def line_nr(self):
    return 42
```

- For more compliated things:

```
def docstring(self):
    return ... # maybe in the future parametrize
```

Transitions

- Do transitions incrementally, big transitions like Python 2 → 3 are hard.
- Deprecate with Warnings & Documentation

Transitions

```
def call_name(self):
    """
    .. deprecated:: 0.8.0
       Use :attr:`.name` instead.

    The name (e.g. 'isinstance') as a string.
    """
    warnings.warn("Use name instead.", DeprecationWarning)
    [...]
```

Service Oriented Architecture

- Amazon:

  ~2002: Use service interfaces only.

  “Anyone who doesn’t do this will be fired. Thank you; have a nice day!”

Good Code

- Use what you learned in API Design for your internal API's.
- You should be able to go public with a sub-package without refactoring.

KTHXBYE

- Like writing APIs? We're looking for Python/DevOps Engineers: job16@cloudscale.ch

cloudscale.ch

- @davidhalter on Github
- @jedidjah_ch on Twitter