backcall Documentation

Release 0.2

Thomas Kluyver

Jun 09, 2020

Contents

backcall is a Python module to write backwards compatible callback APIs. That is, you can add parameters to your calls without breaking third party callback functions that don't expect those newer parameters. For an example of using it, see the Demo notebook.

It can be installed like any other Python package:

```
pip install backcall
```

To use backcall, you first define a 'callback prototype' - a function which takes all of the parameters you're going to pass. Parameters without a default are to be passed positionally. Parameters with a default, or keyword-only parameters on Python 3, are to be passed as keyword arguments. The prototype doesn't need to do anything; only the signature is used.

```
def msg_received_prototype(positional1, positional2, kw1=None, *, kw2):
    pass
```

Decorate your callback prototype with `callback_prototype()`:

You can use the new `adapt` function to prepare callbacks when they are registered with the callback API.

`prototype.`**`adapt`**`(`*callback*`)`

> Inspects the signature of `callback`. If it takes all of the arguments in `prototype`, it is returned unmodified. If it takes a subset of those arguments, a wrapper is returned which will discard the extra arguments and call `callback`. If it takes arguments not specified by the prototype, `TypeError` is raised.

CHAPTER 1

Limitations

- Callback functions can’t have any extra arguments - even if they have default values, `prototype.adapt()` wants to match each argument to something in the prototype.
- The callback API can’t specify that callbacks *must* take certain arguments. A function that takes no arguments is a valid callback anywhere.

Both of these limitations could be removed in a later version, but I want to see what’s important first.

- Callback functions must have introspectable signatures. Practically, this means functions defined in compiled code can’t be used as callbacks without wrapping them in a Python function.

CHAPTER 2

Signature introspection

backcall includes a backported copy of the Python 3.3+ `inspect.Signature` machinery to support Python 2.7. This was backported by Min RK for IPython, and copied here.

CHAPTER 3

Indices and tables

- genindex
- search

Python Module Index

Index