

## When running a complicated pipeline

- might evolve like living beings (working but not optimal solutions are not revisited (:thumbsdown:))
- once need to apply to a project, want it to be wrapt and easy to apply :question: -> enter!

## The Almighty Wrapper

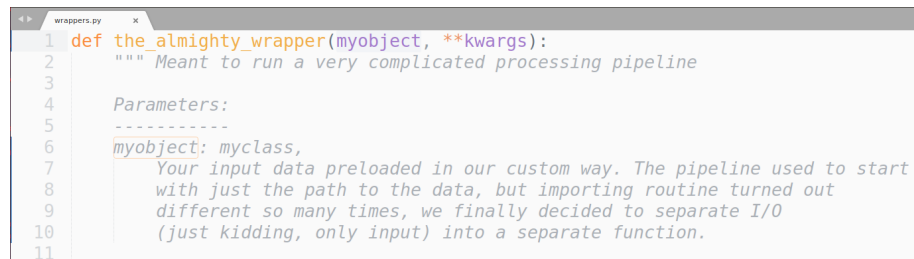


```
1 def the_almighty_wrapper(dicom_path, **kwargs):
2     """ Meant to run a very complicated processing pipeline
3
4     Parameters:
5     -----
6     dicom_path: str,
7         Path to the folder containing DICOM files of a multi-echo 3D GRE
8
```

(with the\_path in the argument)

## OK, we give up on the input

- then you apply it to a bit of a different data and it quickly becomes to much of a pain to maintain meaningfull import wrapped, -> so enter!



```
1 def the_almighty_wrapper(myobject, **kwargs):
2     """ Meant to run a very complicated processing pipeline
3
4     Parameters:
5     -----
6     myobject: myclass,
7         Your input data preloaded in our custom way. The pipeline used to start
8         with just the path to the data, but importing routine turned out
9         different so many times, we finally decided to separate I/O
10        (just kidding, only input) into a separate function.
11
```

Figure 1: title-v1

... because there has to be something unified for the rest of the pipeline to build on top :question: **IMHO:** much like in Hawkings' someone-elses-words each formula halves the number of readers...  $E = mc^2$  Each custom class incorporated in the core of the algorithms reduces the number of people contributing to the code (we all love numpy or reimplementing algos from matlab's scratch). Plus, you pay the penalty of calling API of your custom objects in algorithms.

## If myobject-abstraction leaks...

..you want to add a hook!.. :thumbsdown:

```

8      with just the path to the data, but importing routine turned out
9      different so many times, we finally decided to separate I/O
10     (just kidding, only input) into a separate function.
11
12     # Following are the options to control the workflow and special cases
13
14     echo_time: iterable, float, optional
15     (necessary for the processing) If given overwrites the echo time
16     specified in the input ``myclass`` instance
17

```

Figure 2: echoes-1

(to control corrupted object [the world we live in is cruel, some data come inconsistent])

```

567     if echo_time is None:
568         # if not given, use the one from the I/O
569         if not myobject.echo_time is None:
570             echo_time = myobject.echo_time
571         else:
572             # no good if you are here, let's see if the data came
573             # from the other route:
574             try:
575                 echo_time = myobject.header['echo_time']
576             except KeyError:
577                 print('Echo time not present')
578                 AttributeError('echo_time not set')
579

```

Figure 3: echoes-2

(But then if you add the hook, it must have priority over the object, so the flow control starts to build up, and in some other dataset there might be other source of metainformation) **IMHO:** separate preprocessing run to normalize and unify the data