BLonD Clean-up Ideas….

H. Timko

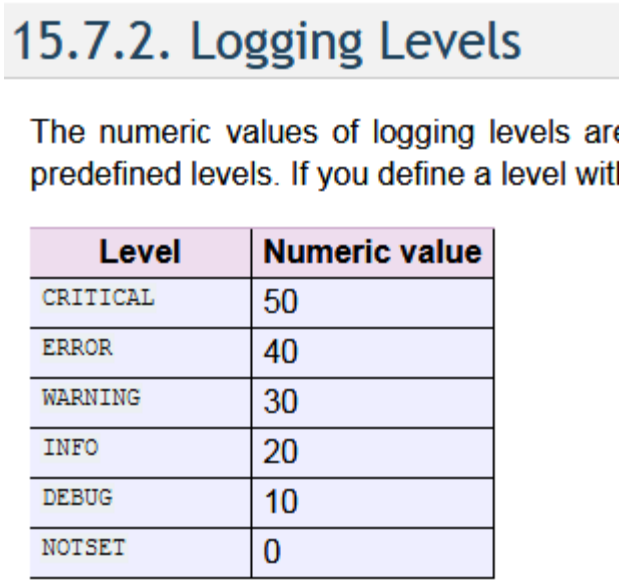
BE-RF-FB

Renaming suggestions

- GeneralParamters -> MachineParamters
- Slices -> (Beam)Profile
- TEST_CASES -> EXAMPLES, TC1 -> EX1 etc.

Logger

- Verbose mode for the full code
- Using python “logging” package
- Levels



15.7.2. Logging Levels

The numeric values of logging levels are
predefined levels. If you define a level with

Level	Numeric value
CRITICAL	50
ERROR	40
WARNING	30
INFO	20
DEBUG	10
NOTSET	0

Logger class

```
19  class Logger(object):
20      """Class to log messages coming from other classes. Messages contain
21      {Time stamp} {Class name} {Log level} {Message}. Errors, warnings and info
22      are logged into the console. To disable logging, call Logger().disable()
23
24      Parameters
25      ----------
26      debug : bool
27          Log DEBUG messages in 'debug.log'; default is False
28
29      """
30
31      def __init__(self, debug = False):
32
33          # Root logger on DEBUG level
34          root_logger = logging.getLogger()
35          root_logger.setLevel(logging.DEBUG)
36
37          # Console handler on INFO level
38          console_handler = logging.StreamHandler()
39          console_handler.setLevel(logging.INFO)
40          log_format = logging.Formatter(
41              "%(asctime)s %(name)-20s %(levelname)-9s %(message)s")
42          console_handler.setFormatter(log_format)
43          root_logger.addHandler(console_handler)
44
45          # File logger on DEBUG level
46          if debug == True:
47              file_handler = logging.FileHandler('debug.log', mode = 'w')
48              file_handler.setLevel(logging.DEBUG)
49              file_handler.setFormatter(log_format)
50              root_logger.addHandler(file_handler)
51
52          logging.info("Start logging")
53          if debug == True:
54              logging.debug("Logger in debug mode")
55
56
57      def disable(self):
58          """Disables all logging."""
59
60          logging.info("Disable logging")
61          logging.disable(level=logging.NOTSET)
```

Root logger, other loggers "report" to this one

Handler for console output: by default, on INFO level

Handler for file output: on demand, on DEBUG level

Entirely disable logging

Subloggers, INFO message

- In a class

```
23  class SPSOneTurnFeedback(object):
24      '''
25      Voltage feedback around the cavity.
26      '''
27
28      def __init__(self, RFSectionParameters, Beam, Slices):
29
30          self.rf_params = RFSectionParameters
31          self.beam = Beam
32          self.slices = Slices
33
34          # Initialise bunch-by-bunch voltage correction array
35          self.voltage = np.ones(self.slices.n_slices, dtype=float) + \
36              1j*np.zeros(self.slices.n_slices, dtype=float)
37
38          # Initialise comb filter
39          self.a_comb_filter = float(15/16)
40          self.voltage_IQ_prev = np.zeros(len(self.voltage)) #polar_to_cartesian
41
42          # Initialise cavity filter
43          self.cavity_filter_buckets = float(5) # Trev  / 4620 * 5
44
45          # Set up logging
46          self.logger = logging.getLogger(__class__.__name__)
47          self.logger.info("Class initialized")
48
49
```

Subloggers, DEBUG message

- In a module

```
16 from __future__ import division
17 import numpy as np
18 from scipy.constants import e
19
20 # Set up logging
21 import logging
22 logger = logging.getLogger(__name__)
23
24
25 def polar_to_cartesian(amplitude, phase):
26     """Convert data from polar to cartesian (I,Q) coordinates.
27
28     Parameters
29     ----------
30     amplitude : float array
31         Amplitude of signal
32     phase : float array
33         Phase of signal
34
35     Returns
36     -------
37     complex array
38         Signal with in-phase and quadrature (I,Q) components
39     """
40
41     logger.debug("Converting from polar to cartesian %.4e", np.mean(amplitude))
42
43     return amplitude*(np.cos(phase) + 1j*np.sin(phase))
44
```

Output

```
46
47  # Logger for messages on console & in file
48  #Logger().disable()
49  Logger(debug = True)
50
51  # Set up machine parameters
52  GeneralParams = GeneralParameters(N_t, C, alpha, p_s)
53  print("Machine parameters set!")
54
```

Call Logger in main.py

Console | History | Git Staging | PyUnit

```
<terminated> C:\Work\Eclipse\git\BLonD_fork2\__TEST_CASES\main_files\EX21_cavity_feedback.py
2017-08-03 11:55:52,141 root                    INFO      Start logging
Machine parameters set!
RF parameters set!
Beam set! Number of particles 100000
Time coordinates are in range -6.6914e-10 to 5.9630e-09 s
Slices set! Integral of slices is 100000
Total charges 1.6022e-08 C
1258293888.44
RF current is 4e-094 A
2017-08-03 11:55:52,279 SPSOneTurnFeedback    INFO      Class initialized

Done!
```

In console, with prints from main.py

In debug.log

```
2017-08-03 11:55:52,141 root                    INFO      Start logging
2017-08-03 11:55:52,141 root                    DEBUG     Logger in debug mode
2017-08-03 11:55:52,279 SPSOneTurnFeedback      INFO      Class initialized
2017-08-03 11:55:52,280 llrf.filters            DEBUG     Converting from polar to cartesian 1.0000e+00
```