

Data acquisition solution

[Previous](#) [Up](#) [Next](#)

Here is one possible answer:

```
import os
import ecflow

defs = ecflow.Defs()
suite = defs.add_suite("data_aquisition")
suite.add_repeat( ecflow.RepeatDay(1) )
suite.add_variable("ECF_HOME",      os.getenv("HOME") + "/course")
suite.add_variable("ECF_INCLUDE",   os.getenv("HOME") + "/course")
suite.add_variable("ECF_FILES",     os.getenv("HOME") + "/course/data")
suite.add_variable("SLEEP", "2")
for city in ( "Exeter", "Toulouse", "Offenbach", "Washington", "Tokyo", "Melbourne", "Montreal" ) :
    fcity = suite.add_family(city)
    fcity.add_task("archive")
    for obs_type in ( "observations", "fields", "images" ):
        type_fam = fcity.add_family(obs_type)
        if city in ( "Exeter", "Toulouse", "Offenbach" ): type_fam.add_time("00:00 23:00 01:00")
        if city in ( "Washington" ) : type_fam.add_time("00:00 23:00 03:00")
        if city in ( "Tokyo" ) : type_fam.add_time("12:00")
        if city in ( "Melbourne" ) : type_fam.add_day( "monday" )
        if city in ( "Montreal" ) : type_fam.add_date(1, 0, 0)

        type_fam.add_task("get")
        type_fam.add_task("process").add_trigger("get eq complete")
        type_fam.add_task("store").add_trigger("get eq complete")
```

It is also possible to automatically generate the [ecf script](#) using the python API:

```
defs.generate_scripts()
```

This enables testing of the [suite definition](#), without worrying about the [ecf script](#)'s.



When there are no [event](#)'s, [meter](#)'s or [label](#)'s in the [suite definition](#), the content of the generated scripts are identical. Hence this functionality should **only** be used as **debug** aid for the definition

[Previous](#) [Up](#) [Next](#)