DTK Package

- Parameter file for c++ & python
  - Particle mass, rho_crit, z <-->step
- Tools for sorting arrays
- Easy read/write of binary files, and read of gio files

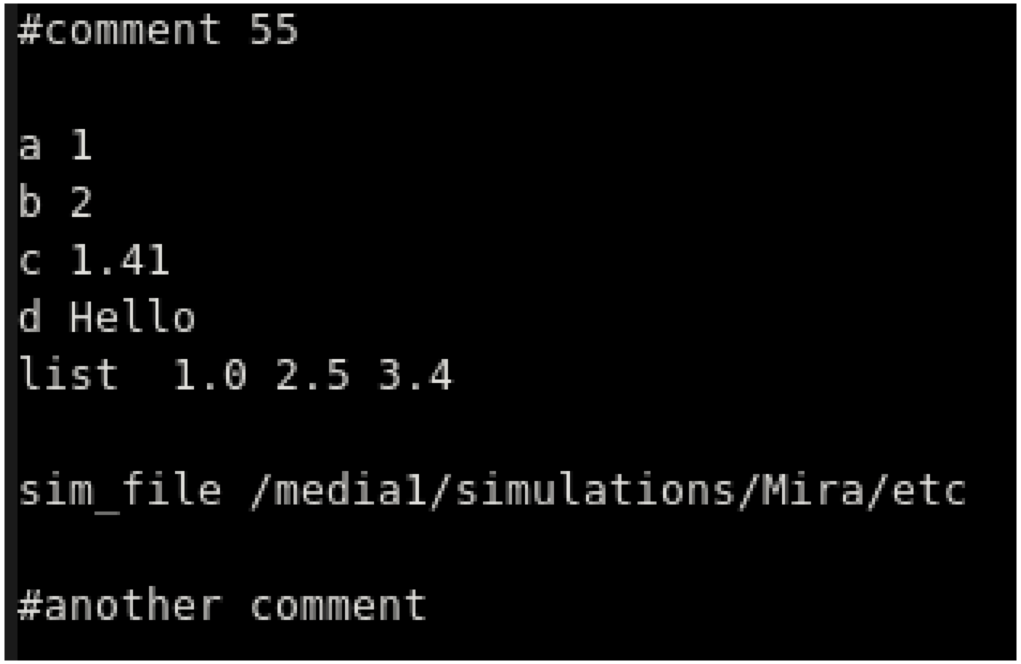
- SVN checkout from:

  https://svn.alcf.anl.gov/repos/DarkUniverse/users/dkorytov/dtk

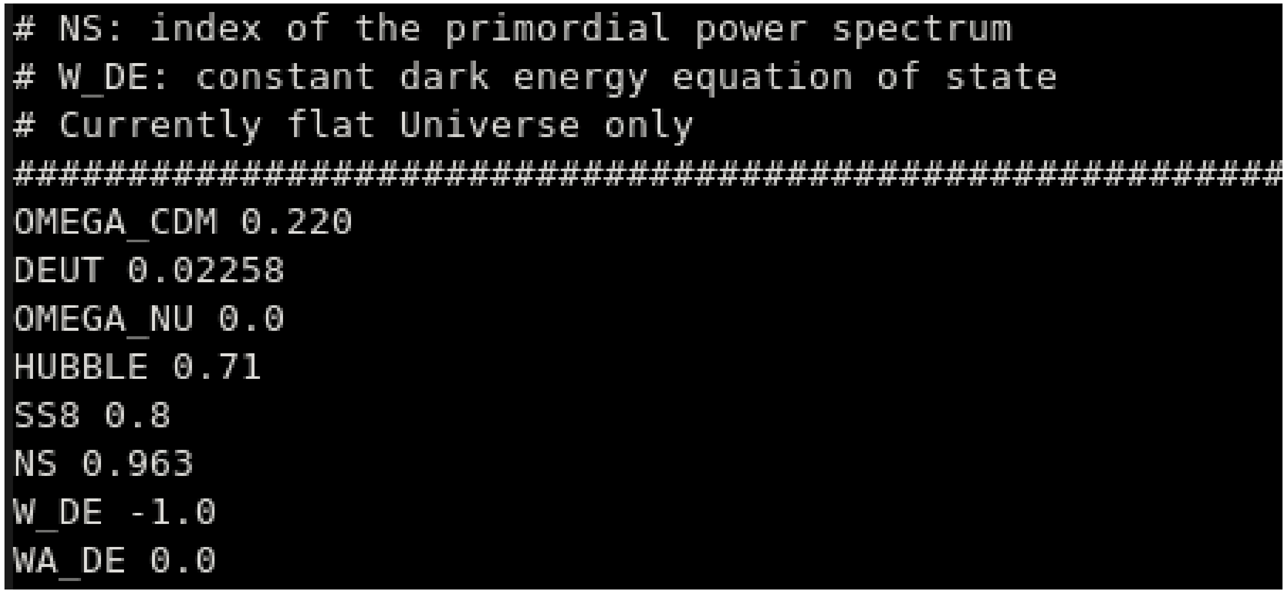
Param

- Same format as indat.params
  - Key word followed by value(s) “foo 1.2” “bar 3 4”
  - “#” indicate comment lines
- Values retrieved by key word and expected type
  - Throws exceptions if the keyword is not found/does not convert
  - Duplicate entries not allowed
  - List all, accessed or unaccessed parameters

Param File Example

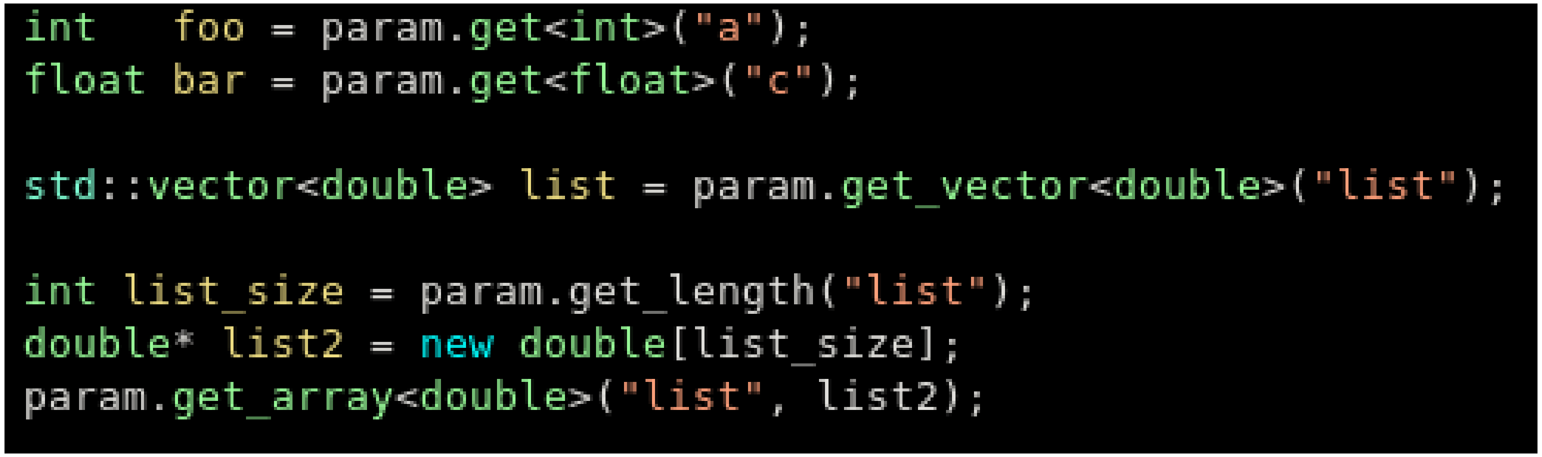


```
#comment 55

a 1
b 2
c 1.41
d Hello
list  1.0 2.5 3.4

sim_file /media1/simulations/Mira/etc

#another comment
```

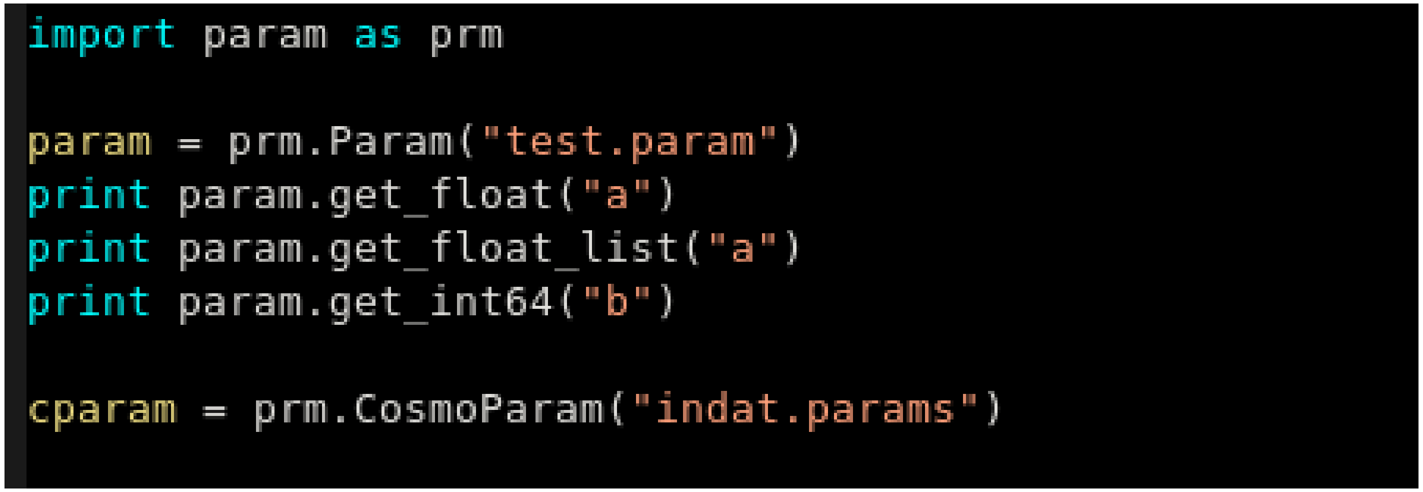


```
# NS: index of the primordial power spectrum
# W_DE: constant dark energy equation of state
# Currently flat Universe only
#######################################################
OMEGA_CDM 0.220
DEUT 0.02258
OMEGA_NU 0.0
HUBBLE 0.71
SS8 0.8
NS 0.963
W_DE -1.0
WA_DE 0.0
```

Param Usage Example



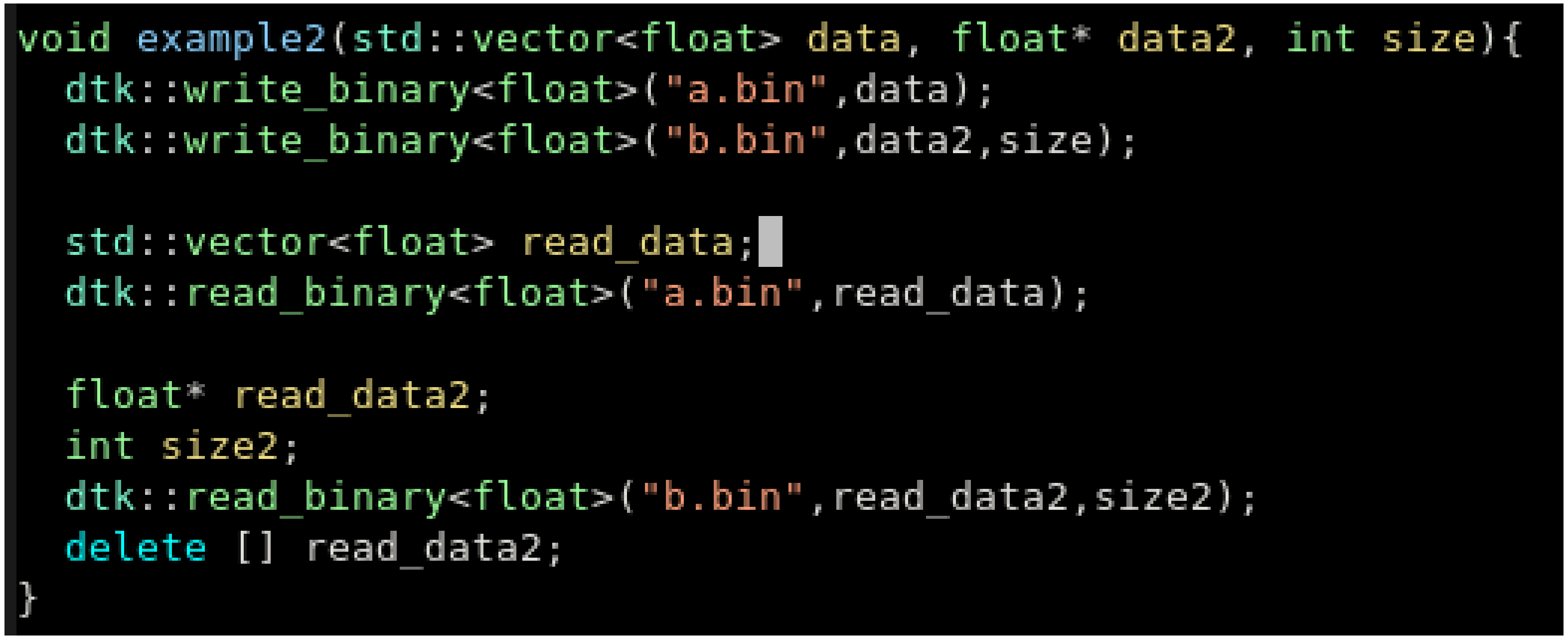
```cpp
int   foo = param.get<int>("a");
float bar = param.get<float>("c");

std::vector<double> list = param.get_vector<double>("list");

int list_size = param.get_length("list");
double* list2 = new double[list_size];
param.get_array<double>("list", list2);
```



```python
import param as prm

param = prm.Param("test.param")
print param.get_float("a")
print param.get_float_list("a")
print param.get_int64("b")

cparam = prm.CosmoParam("indat.params")
```

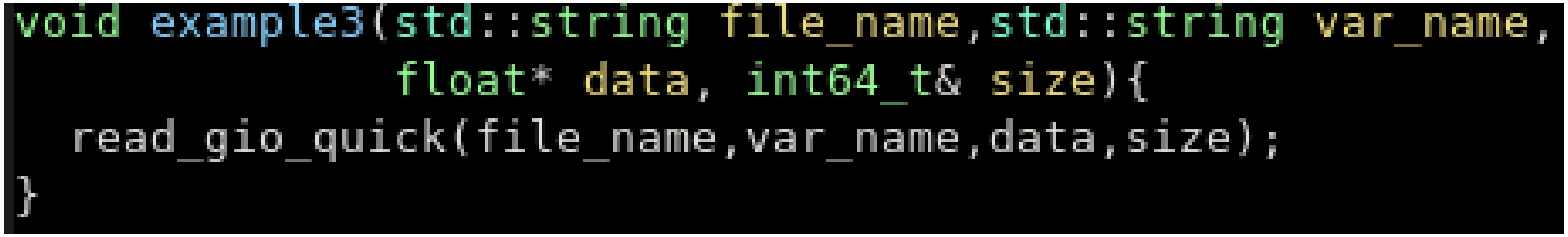
IO Knick-Knacks

- Clean & quick ways to read/write binary



```
void example2(std::vector<float> data, float* data2, int size){
  dtk::write_binary<float>("a.bin",data);
  dtk::write_binary<float>("b.bin",data2,size);

  std::vector<float> read_data;
  dtk::read_binary<float>("a.bin",read_data);

  float* read_data2;
  int size2;
  dtk::read_binary<float>("b.bin",read_data2,size2);
  delete [] read_data2;
}
```

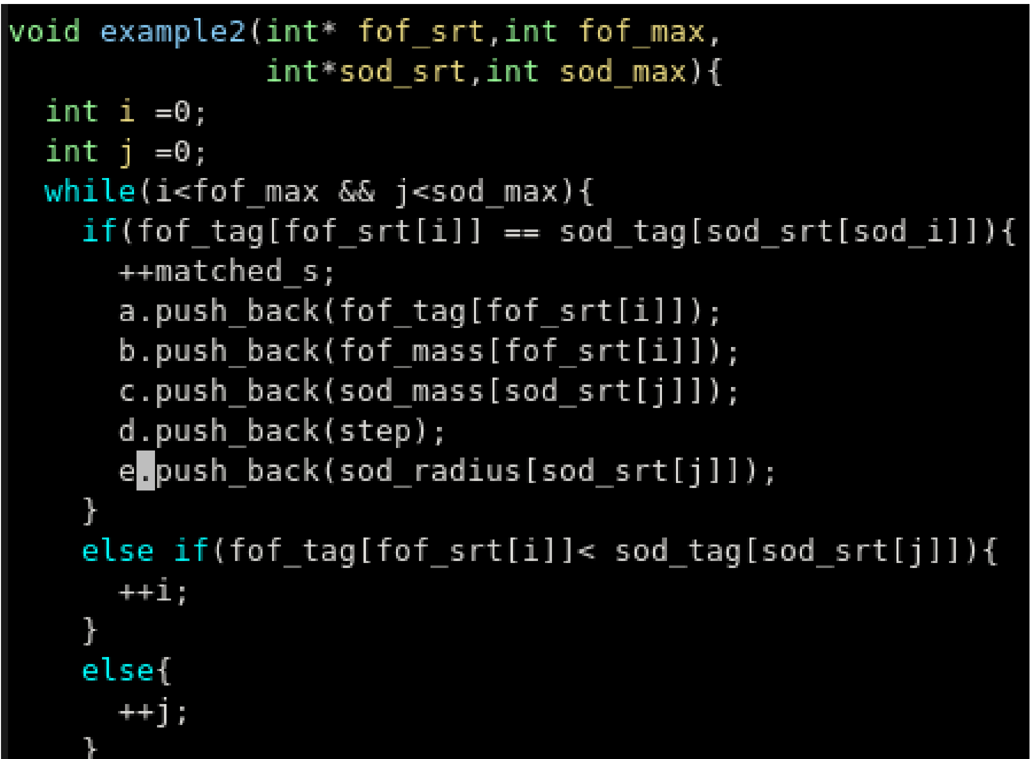
- Quick way to read gio files (needs includes/lib from gio)



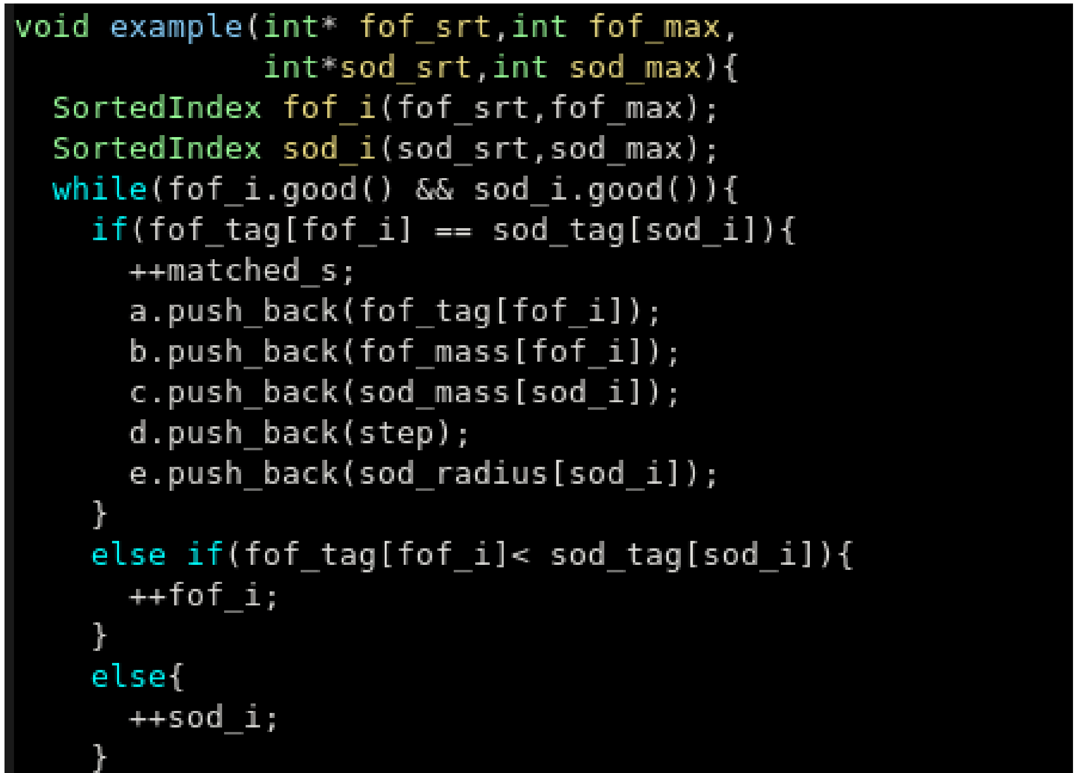
```
void example3(std::string file_name,std::string var_name,
              float* data, int64_t& size){
  read_gio_quick(file_name,var_name,data,size);
}
```

Sorting

- SortedIndex
  - Hides using nested arrays in traversing sorted arrays by index
  - Acts like a normal integer but uses values from an array.
    - Incrementing changes to the next value in the array



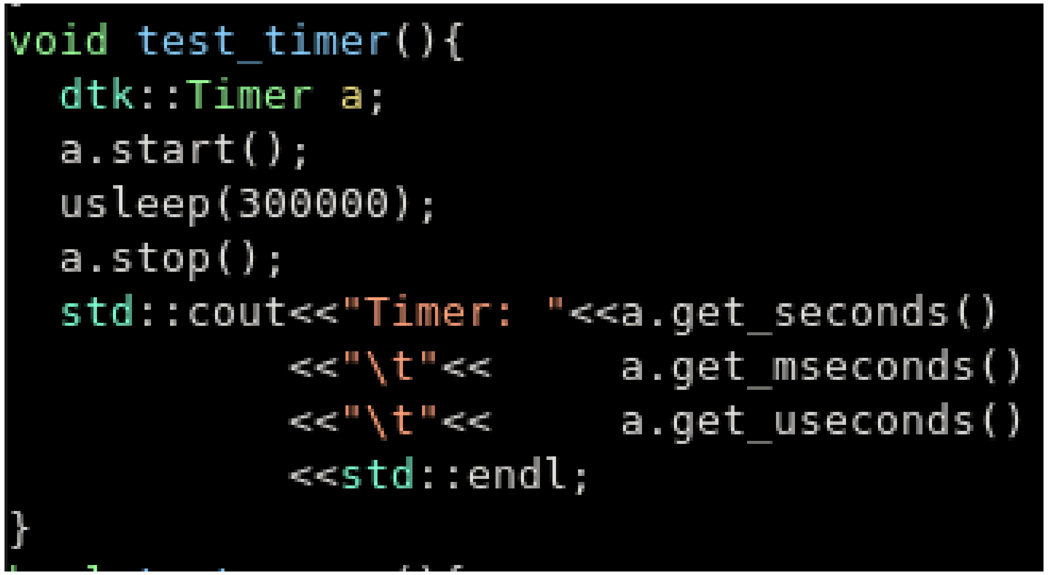
```
void example2(int* fof_srt,int fof_max,
              int*sod_srt,int sod_max){
  int i =0;
  int j =0;
  while(i<fof_max && j<sod_max){
    if(fof_tag[fof_srt[i]] == sod_tag[sod_srt[sod_i]]){
      ++matched_s;
      a.push_back(fof_tag[fof_srt[i]]);
      b.push_back(fof_mass[fof_srt[i]]);
      c.push_back(sod_mass[sod_srt[j]]);
      d.push_back(step);
      e.push_back(sod_radius[sod_srt[j]]);
    }
    else if(fof_tag[fof_srt[i]]< sod_tag[sod_srt[j]]){
      ++i;
    }
    else{
      ++j;
    }
```



```
void example(int* fof_srt,int fof_max,
             int*sod_srt,int sod_max){
  SortedIndex fof_i(fof_srt,fof_max);
  SortedIndex sod_i(sod_srt,sod_max);
  while(fof_i.good() && sod_i.good()){
    if(fof_tag[fof_i] == sod_tag[sod_i]){
      ++matched_s;
      a.push_back(fof_tag[fof_i]);
      b.push_back(fof_mass[fof_i]);
      c.push_back(sod_mass[sod_i]);
      d.push_back(step);
      e.push_back(sod_radius[sod_i]);
    }
    else if(fof_tag[fof_i]< sod_tag[sod_i]){
      ++fof_i;
    }
    else{
      ++sod_i;
    }
```

Timings

- Simple timer
  - Nothing special



```
void test_timer(){
  dtk::Timer a;
  a.start();
  usleep(300000);
  a.stop();
  std::cout<<"Timer: "<<a.get_seconds()
           <<"\t"<<     a.get_mseconds()
           <<"\t"<<     a.get_useconds()
           <<std::endl;
}
```

How to Use in C++

- Put the dtk directory in your compilation directory
- Run Make in dtk folder
- Add the dtk to your include path of your program
  - Compile flag: -Idtk (upper case I)
  - Or include them with as “dtk/util.hpp”, “dtk/timer.hpp”
- Add the dtk library to your lib path and include the library
  - Link flags: -Ldtk -ldtk (lower case L)
- If using gio_util.hpp, you must have gio headers in your include path and link against the gio library.

How to Use Param in Python

- Place the dtk directory into your python path
  - Eg: same folder of your script
- Import the module with
  - a) “import dtk” or b) “import Param from dtk”
  - Use a) “dtk.Param()” or b) “Param()”
  - Same for CosmoParam