Гайд по CberEmissionTrack

Для начала, надо установить трекер(установка с помощью conda появится позже):

In [1]:
```
# во избежании путанницы на случай, если пакет устанавливался раньше, лучше удалить его и
!pip uninstall SberEmissionTrack -y
!pip install git+git://github.com/vladimir-laz/AIRIEmisisonTracker.git
```

```
Found existing installation: SberEmissionTrack 0.2.0
Uninstalling SberEmissionTrack-0.2.0:
  Successfully uninstalled SberEmissionTrack-0.2.0
Collecting git+git://github.com/vladimir-laz/AIRIEmisisonTracker.git
  Cloning git://github.com/vladimir-laz/AIRIEmisisonTracker.git to /tmp/pip-re
q-build-m1q71neq
  Running command git clone -q git://github.com/vladimir-laz/AIRIEmisisonTrack
er.git /tmp/pip-req-build-m1q71neq
  Resolved git://github.com/vladimir-laz/AIRIEmisisonTracker.git to commit 028
e9eb83f5cf5fdd73813d73045ba6254f637c4
Requirement already satisfied: APScheduler in /home/user/conda/lib/python3.7/s
ite-packages (from SberEmissionTrack==0.2.0) (3.7.0)
Requirement already satisfied: pynvml>=5.6.2 in /home/user/conda/lib/python3.
7/site-packages (from SberEmissionTrack==0.2.0) (11.0.0)
Requirement already satisfied: psutil in /home/user/conda/lib/python3.7/site-p
ackages (from SberEmissionTrack==0.2.0) (5.8.0)
Requirement already satisfied: py-cpuinfo in /home/user/conda/lib/python3.7/si
te-packages (from SberEmissionTrack==0.2.0) (8.0.0)
Requirement already satisfied: tzlocal~=2.0 in /home/user/conda/lib/python3.7/
site-packages (from APScheduler->SberEmissionTrack==0.2.0) (2.1)
Requirement already satisfied: pytz in /home/user/conda/lib/python3.7/site-pac
kages (from APScheduler->SberEmissionTrack==0.2.0) (2021.1)
Requirement already satisfied: setuptools>=0.7 in /home/user/conda/lib/python
3.7/site-packages (from APScheduler->SberEmissionTrack==0.2.0) (58.0.4)
Requirement already satisfied: six>=1.4.0 in /home/user/conda/lib/python3.7/si
te-packages (from APScheduler->SberEmissionTrack==0.2.0) (1.16.0)
Building wheels for collected packages: SberEmissionTrack
  Building wheel for SberEmissionTrack (setup.py) ... done
  Created wheel for SberEmissionTrack: filename=SberEmissionTrack-0.2.0-py3-no
ne-any.whl size=5864 sha256=525bcafa72a7a7914773bfbe8b169c87828820a6ee34d2b918
3601678fc61e38
  Stored in directory: /tmp/pip-ephem-wheel-cache-ro86r3c8/wheels/67/e7/41/7ae
3b1d2339881a7b0688acdc1bbcca48d205b78766f2ad749
Successfully built SberEmissionTrack
Installing collected packages: SberEmissionTrack
Successfully installed SberEmissionTrack-0.2.0
```

Делаем необходимые импорты

In [2]:
```
import SberEmissionTrack
```

Можно посмотреть, какие устройства доступны:

In [3]:
```
SberEmissionTrack.available_devices()
```

```
Seeable cpu devices:
        Intel(R) Xeon(R) Platinum 8168 CPU @ 2.70GHz: 96 devices
Seeable gpu devices:
        Tesla V100-SXM3-32GB: 2 devices
```

Соответственно, если нет доступных gpu, это будет обозначенно. Однако если не

будет свободных cpu, то возникнет ошибка, ибо cpu всегда должен быть виден.

С автоматическим определением gpu проблем нет, но вот название cpu выводится не такое, как обычно указывается в официальных базах. В связи с этим полностью автоматические определение модели cpu требует либо составления собственной базы данных, либо тяжелой работы с регулярными выражениями, которая не факт что закончится успехом. В связи с этим, при создании трекера и вызове метода .start() пользователю будет выведена модель cpu и будет предложено ввести его TDP самостоятельно. Попробуйте запустить и проверить(для "Intel(R) Xeon(R) Platinum 8168 CPU @ 2.70GHz." TDP = 205Ватт):

In [4]:
```
tracker = SberEmissionTrack.Tracker(project_name="TEST",
                                    experiment_description="testing AIRI trac
                                    save_file_name="emission.csv",
                                    measure_period=1)
tracker.start()
# здесь мог быть ваш код
tracker.stop()
```

```
Name of your cpu: Intel(R) Xeon(R) Platinum 8168 CPU @ 2.70GHz.
Please, enter it's TDP(watt): 205
```

Конечно, постоянно вводить значение TDP - неудобно, поэтому если оно известно, то можно ввести его заранее:

In [6]:
```
tracker = SberEmissionTrack.Tracker(project_name="TEST",
                                    experiment_description="testing AIRI trac
                                    save_file_name="emission.csv",
                                    measure_period=1,
                                    cpu_tdp=205)
tracker.start()
# здесь мог быть ваш код
tracker.stop()
```

При запуске метода .start() идет подсчет базового(фонового) энергопотребления. Для этого желательно брать не текущие значения, а усредненые, это занимает около 10 секунд.

Также если расчетов несколько, то рекомендуется подождать между ними около 20 секунд. За это время энергопотребление успевает упасть до фонового

Также лучше использовать не очень большой measure_period

Если вдруг что-то непонятно с работой трекера, то всегда можно обратиться к функции help(). Там есть описание класса, описание функций и пример использования

In [7]:
```
help(SberEmissionTrack.Tracker)
```

```
Help on class Tracker in module SberEmissionTrack.emission_track:

class Tracker(builtins.object)
 |  Tracker(project_name, experiment_description, save_file_name='emission.cs
v', measure_period=2, emission_level=511.7942, cpu_tdp=None)
 |
 |  In order to correctly calculate gpu or cpu ppower consumption you should c
reate
 |  Tracker before any gpu and cpu uses as tracker considers background gpu an
d cpu power
```

```
 |
 |  For every new gpu calculation it should be created new tracker
 |
 |  ----------------------------------------------------------------------
 |  Use example:
 |
 |  import SberEmissionTrack.Tracker
 |  tracker = Tracker(project_name=your_project_name,
 |                    experiment_description=your_experiment_description,
 |                    save_file_name="you_file_name",
 |                    measure_period=2,   #measurement will be done every 2 se
conds
 |                    )
 |  tracker.start()
 |  *your gpu calculations*
 |  tracker.stop()
 |
 |  ----------------------------------------------------------------------
 |
 |  Methods defined here:
 |
 |  __init__(self, project_name, experiment_description, save_file_name='emiss
ion.csv', measure_period=2, emission_level=511.7942, cpu_tdp=None)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  consumption(self)
 |
 |  emission_level(self)
 |
 |  measure_period(self)
 |
 |  start(self)
 |
 |  stop(self)
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  __dict__
 |      dictionary for instance variables (if defined)
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
```

In [ ]: