DTI Platform Resource Management Guide

Here, we bring together information regarding how to control your computing resources on the C3 AI Suite.

Jupyter

Access Through Static Console

The Jupyter service is controlled through the '`Jupyter`' type. To start the Juptyer service with the default container profile, execute `Jupyter.inst().start()` within the static console. To stop the Jupyter service execute `Jupyter.inst().stop()`.

To launch Jupyter service with a specific config, use `Jupyter.startWithConfig()`. This accepts an argument of type `JupyterServiceConfig`. The `JupyterServiceConfig` can represent multiple types of services one of which is the '`resourceProfile`'. This can represent pre-designated resource profiles of type `ResourceProfile`. To list the available resource profiles do `c3Grid(ResourceProfile.fetch())`

By default, two `ResourceProfiles` exist, 'Basic', and 'BasicGpu'. For example, to launch the Jupyter service using the 'BasicGpu' profile, you would execute

```
Jupyter.startWithConfig({'resourceProfile': 'BasicGpu'})
```

Creating new Resource Profile

As a default, tags come with "Basic" and "BasicGpu" profiles. In case these do not fit your needs, it is possible to create a new profile from your static console using:

```
var rp = ResourceProfile.upsert({'id': 'YourProfileName', 'name': 'YourProfileName', 'cpuCount': <cpus>,
'memoryMb': <cpu_memory>, 'diskGb': <disk_memory>, 'gpuCount': <gpus>, 'gpuMemoryMb': <gpu_memory>});
```

In addition to that, to effectively use the total memory requested in your resource profile, you will need to change the memory limits that are set at the user level of your Jupyter type. This can be done from the static console with:

```
Jupyter.setMemoryOverride(<cpu_memory>, ConfigOverride.USER)
```

where `<cpu_memory>`

matches the field `'memoryMb'` of your RP instance.

If you want to make this your default profile whenever launching a Jupyter service, you can do so with:

```
Jupyter.setDefaultResourceProfile('YourProfileName')
```

Keep in mind that, currently, the **maximum number of GPUs that can be used on the training cluster is one.** If you request more than one, your static console is going to be unresponsive for some time, and you won't be able to start your Jupyter service. The same thing will happen if you request resources that surpass the limits of the current underlying hardware infrastructure or the limits set by C3 Admin. These are summarized in the table below:

Resource	Max Count	Max Memory (MB)
CPU only	6	54*1024
GPU only	1	12*1024
CPUs + GPU	• 5 CPUs • 1 GPU	• 48*1024 (CPU) • 12*1024 (GPU)

If you are not sure about how to make your choice, but you know you need more than 1 CPU core, we recommend using the `QuarterNode_cpu` Resource Profile:

```
var rp = ResourceProfile.upsert({'id': 'QuarterNode_cpu', 'name': 'QuarterNode_cpu', 'cpuCount': 4, 'memoryMb':
16*1024, 'diskGb': 512, 'gpuCount': 0, 'gpuMemoryMb': 0});

Jupyter.setMemoryOverride(16*1024, ConfigOverride.USER);

Jupyter.startWithConfig({'resourceProfile': 'QuarterNode_cpu'});
```

If you are using GPUs, we recommend using the entire node, since each node has only one GPU. You can do that by creating the `FullNode_gpu` Resource Profile:

```
var rp = ResourceProfile.upsert({'id': 'FullNode_gpu', 'name': 'FullNode_gpu', 'cpuCount': 5, 'memoryMb':
48*1024, 'diskGb': 512, 'gpuCount': 1, 'gpuMemoryMb': 12*1024});

Jupyter.setMemoryOverride(12*1024, ConfigOverride.USER);

Jupyter.startWithConfig({'resourceProfile': 'FullNode_gpu'});
```

If you have additional requirements, please contact DTI DevOps.

Access Through IDS

Through IDS, clear controls regarding Jupyter are exposed under the 'ML Studio' section for Application. When you've selected a project, you need to start the Jupyter service to access notebooks on that project. In the creation process, you're able to select the container profile to use for your Jupyter service. By default, the 'BasicGpu' profile offers a single K80 GPU.

You can create new container profiles. Select the 'App Settings' Menu, and there's a tab called 'Container Profiles'. You can create new profiles which have different resources like more GPUs, and more vCPUs, and more RAM. Create a new container profile, then when creating the Jupyter service select the new profile to gain access to those resources.