
notes

fangjun

May 16, 2022

CONTENTS:

1	Sphinx	3
1.1	Setup	3
1.2	How to include code from a file	4
1.3	Link	4
1.3.1	hello	5
2	git	7
2.1	Commands	7
2.1.1	rev-parse	7
3	docker	9
3.1	Installation	9
3.1.1	macos	9
4	LaTeX	11
4.1	TikZ	11
4.1.1	Basics	11
5	Kaldi	13
5.1	Decoding	13
6	bash	15
6.1	sort	15
6.2	echo	15
7	CUDA	17
7.1	Installation	17
7.1.1	CUDA 10.1.243	17
7.1.2	CUDA 11.0.3	17
7.1.3	CUDA 11.3.1	18
7.1.4	CUDA 11.5.2	18
7.1.5	CUDA 11.6.1	18
8	torch	19
8.1	DDP	19
8.1.1	Initialization	19
8.2	TorchScript	19
8.2.1	Hello	19
8.2.2	Load in C++	23
8.2.3	ArrayRef	25
8.2.4	ScalarType	27

8.2.5	TypeMeta	28
8.2.6	torch::Device	29
8.2.7	TensorOptions	31
8.2.8	Tensor Creation	32
8.2.9	Tensor	33
9	Python	35
9.1	asyncio	35
9.1.1	Hello World	35
9.1.2	References	35
9.2	argv	35
10	java	37
10.1	Install	37
10.1.1	formatter	37
10.1.2	JDK	37
10.2	Hello world	38
10.3	Reference	39
11	javascript	41
11.1	Hello world	41
11.2	TODOs	41
12	HTML	43
12.1	Hello world	43
12.1.1	comments	43
12.1.2	images	43
12.1.3	ordered lists	43
12.1.4	unordered lists	44
12.1.5	links	44
12.2	References	44
13	css	45
13.1	Hello world	45
13.1.1	comment	45
13.1.2	Selector	45
13.2	References	46

Download this website in a single [pdf file](#).

This page describes how this website is setup.

1.1 Setup

1. Install the dependencies in `./docs/requirements.txt`.

```
sphinx==4.3.2
sphinx-autodoc-typehints==1.12.0
sphinx_rtd_theme==1.0.0
sphinxcontrib-bibtex==2.4.1
```

2. Use `sphinx-quickstart` to generate the skeleton. When it prompts:

```
Separate source and build directories(y/n)
```

Answer yes.

3. Edit `docs/source/conf.py` and add the following lines to it:

```
import sphinx_rtd_theme
extensions = [
    'sphinx.ext.autodoc',
    'sphinx.ext.autosummary',
    'sphinx.ext.githubpages',
    'sphinx.ext.mathjax',
    'sphinx.ext.napoleon',
    'sphinx.ext.todo',
    'sphinx.ext.viewcode',
    'sphinxcontrib.bibtex',
]

html_theme = 'sphinx_rtd_theme'

master_doc = 'index'
pygments_style = 'sphinx'
html_theme_path = [sphinx_rtd_theme.get_html_theme_path()]
smartquotes = False
html_show_sourcelink = True

html_context = {
```

(continues on next page)

(continued from previous page)

```
'display_github': True,
'github_user': 'csu-fangjun',
'github_repo': 'notes',
'github_version': 'master',
'conf_py_path': '/docs/source/',
}

html_theme_options = {
    'logo_only': False,
    'display_version': True,
    'prev_next_buttons_location': 'bottom',
    'style_external_links': True,
}
latex_engine = 'xelatex'
```

4. To generate the notes in pdf format, use `make latex`, which generates lots of `tex` files in `./build/latex`. Switch to `build/latex` and run `make`. Assume that you have installed the software to compile `tex` files. It will generate `notes.pdf`.

1.2 How to include code from a file

See <https://www.sphinx-doc.org/en/master/usage/restructuredtext/directives.html#directive-literalinclude>.

1. Show line number: `:linenos:`. By default, line number counts from 0. To add an offset, e.g., 10, to the line number, use `:lineno-start: 10`. Note: It still includes all the contents of the file.
2. To emphasize a line, specified lines, or specified line ranges, use: `:emphasize-lines: 10`, `:emphasize-lines: 10,12,14`, and `:emphasize-lines: 12,15-18`. Note: `emphasize` means to change the background color.
3. Set the language, e.g., `:language: python`.
4. Set the caption, e.g., `:caption: hello world`.
5. To include a function from the python file, use `:pyobject: my_func`.
6. To include specified lines, use `:lines:1,3,5-10,15-`. Note that if using this option, line number counts from 0. Use `:lineno-start: xx` to change the offset for display.

1.3 Link

See <https://sublime-and-sphinx-guide.readthedocs.io/en/latest/references.html> and <https://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html#hyperlinks>

1.3.1 hello

Here is a link to *hello*.

```
.. _Link to hello:
```

```
hello
```

```
-----
```

```
Here is a link to :ref:`Link to hello`.
```


This page describes commonly used git commands.

2.1 Commands

2.1.1 rev-parse

It is quite common to get the root directory of the repository with the command:

```
git rev-parse --show-toplevel
```

For instance, the above command executed in this repository prints something like as follows:

```
/xxx/notes
```

The following shows its usage in a Python script:

```
#!/usr/bin/env python3

import subprocess

d = (
    subprocess.check_output(["git", "rev-parse", "--show-toplevel"])
    .decode("ascii")
    .strip() # remove the trailing \n
)
print(d) # /path/to/notes
```

It can also be used in bash script:

```
root_dir=$(git rev-parse --show-toplevel)
echo "root_dir ${root_dir}"
```

help git-rev-parse outputs helpful information for git rev-parse. In particular, it explains the differences among HEAD~, HEAD~n, HEAD^, and HEAD^n. The following shows the help information about it:

```
<rev>^[<n>], e.g. HEAD^, v1.5.1^0
  A suffix ^ to a revision parameter means the first parent of that commit object. ^
  ↪<n> means the <n>th parent
```

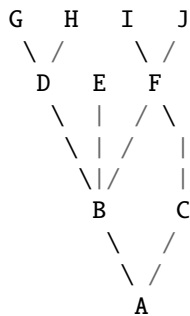
(continues on next page)

(continued from previous page)

(i.e. `<rev>^` is equivalent to `<rev>^1`). As a special rule, `<rev>^0` means the commit itself and is used when `<rev>` is the object name of a tag object that refers to a commit object.

`<rev>~[<n>]`, e.g. `HEAD~`, `master~3`

A suffix `~` to a revision parameter means the first parent of that commit object. A suffix `~<n>` to a revision parameter means the commit object that is the `<n>`th generation ancestor of the named commit object, following only the first parents. I.e. `<rev>~3` is equivalent to `<rev>^^^` which is equivalent to `<rev>^1^1^1`. See below for an illustration of the usage of this form.



$A = A^0$
 $B = A^1 = A^1_1 = A_{~1}$
 $C = A^2$
 $D = A^{11} = A^{1^1_1} = A_{~2}$
 $E = B^2 = A^{12}$
 $F = B^3 = A^{13}$
 $G = A^{111} = A^{1^1_1^1} = A_{~3}$
 $H = D^2 = B^{12} = A^{112} = A_{~2}^2$
 $I = F^1 = B^{13} = A^{113}$
 $J = F^2 = B^{132} = A^{1132}$

3.1 Installation

3.1.1 macos

Refer to <https://docs.docker.com/desktop/mac/install/>.

4.1 TikZ

4.1.1 Basics

This page describes commonly used git commands.

5.1 Decoding

```
CompactLattice compact_lat;  
decoder.GetLattice(true, &compact_lat);  
  
CompactLattice compact_best_path;  
CompactLatticeShortestPath(compact_lat, &compact_best_path);  
  
Lattice best_path;  
ConvertLattice(compact_best_path, best_path);  
  
std::vector<int32_t> tokens;  
std::vector<int32_t> words;  
LatticeWeight weight;  
GetLinearSymbolSequence(best_path, &tokens, &words, &weight);
```

- `decoder/simple-decoder.{h,cc}`

BASH

6.1 sort

Sort files in the folder `t`. The filename has the pattern `xxx.n.txt`, where `n` is some numerical value. Also, exclude `xxx.100.txt`.

```
find ./t -name "xxx*.txt" ! -name "xxx.100.txt" -print0 | sort -z -t. -k2 -n | xargs -r0
```

6.2 echo

Generate a binary file:

```
echo -n -e '\x30\x31\x32' > a.bin  
hexdump a.bin
```


CUDA

7.1 Installation

7.1.1 CUDA 10.1.243

```
./cuda_10.1.243_418.87.00_linux.run --silent --toolkit --installpath=/ceph-data4/fangjun/  
↪software/cuda-10.1.243 --no-opengl-libs --no-drm --no-man-page  
  
# Install cuDNN  
cd /ceph-data4/fangjun/software/cuda-10.1.243  
tar xvf /ceph-sh0/fangjun/cudnn/cudnn-10.1-linux-x64-v8.0.4.30.tgz --strip-components=1
```

```
#!/usr/bin/env bash  
  
export CUDA_HOME=/ceph-data4/fangjun/software/cuda-10.1.243  
export PATH=$CUDA_HOME/bin:$PATH  
export LD_LIBRARY_PATH=$CUDA_HOME/lib64:$LD_LIBRARY_PATH  
  
# See /ceph-fj/fangjun/py38/lib/python3.8/site-packages/torch/share/cmake/Caffe2/Modules_  
↪CUDA_fix/upstream/FindCUDA.cmake  
export CUDA_TOOLKIT_ROOT_DIR=$CUDA_HOME  
export CUDA_TOOLKIT_ROOT=$CUDA_HOME  
export CUDA_BIN_PATH=$CUDA_HOME  
export CUDA_PATH=$CUDA_HOME  
export CUDA_INC_PATH=$CUDA_HOME/targets/x86_64-linux
```

7.1.2 CUDA 11.0.3

```
./cuda_11.0.3_450.51.06_linux.run --silent --toolkit --installpath=/ceph-data4/fangjun/  
↪software/cuda-11.0.3 --no-opengl-libs --no-drm --no-man-page  
  
# Install cuDNN  
cd /ceph-data4/fangjun/software/cuda-11.0.3  
tar xvf /ceph-sh0/fangjun/cudnn/cudnn-11.0-linux-x64-v8.0.4.30.tgz --strip-components=1
```

7.1.3 CUDA 11.3.1

```
./cuda_11.3.1_465.19.01_linux.run --silent --toolkit --installpath=/ceph-data4/fangjun/  
↪ software/cuda-11.3.1 --no-opengl-libs --no-drm --no-man-page  
cd /ceph-data4/fangjun/software/cuda-11.3.1  
tar xvf /ceph-sh0/fangjun/cudnn/cudnn-11.3-linux-x64-v8.2.1.32.tgz --strip-components=1
```

7.1.4 CUDA 11.5.2

```
./cuda_11.5.2_495.29.05_linux.run --silent --toolkit --installpath=/ceph-data4/fangjun/  
↪ software/cuda-11.5.2 --no-opengl-libs --no-drm --no-man-page  
cd /ceph-data4/fangjun/software/cuda-11.5.2  
tar xvf /ceph-sh0/fangjun/cudnn/cudnn-linux-x86_64-8.3.2.44_cuda11.5-archive.tar.xz --  
↪ strip-components=1
```

7.1.5 CUDA 11.6.1

```
./cuda_11.6.1_510.47.03_linux.run --silent --toolkit --installpath=/ceph-data4/fangjun/  
↪ software/cuda-11.6.1 --no-opengl-libs --no-drm --no-man-page  
cd /ceph-data4/fangjun/software/cuda-11.6.1  
tar xvf /ceph-sh0/fangjun/cudnn/cudnn-11.3-linux-x64-v8.2.1.32.tgz --strip-components=1
```

This page describes commonly used git commands.

8.1 DDP

8.1.1 Initialization

8.2 TorchScript

8.2.1 Hello

See https://pytorch.org/tutorials/beginner/Intro_to_TorchScript_tutorial.html.

`torch.jit.script` as a decorator

Listing 1: `./code/1-ex.py`

```
1 @torch.jit.script
2 def adder(x: int):
3     return x + 1
4
5
6 def test_adder():
7     assert isinstance(adder, torch.jit.ScriptFunction)
8     print(adder.graph)
9     print("-" * 10)
10    print(adder.code)
11    adder.save("adder.pt")
12
13    my_adder = torch.jit.load("adder.pt")
14
15    assert isinstance(my_adder, torch.jit._script.RecursiveScriptModule)
16    assert isinstance(my_adder, torch.jit.ScriptModule)
17    assert not isinstance(my_adder, torch.jit.ScriptFunction)
18    print(my_adder(torch.tensor([3])))
19
20
```

(continues on next page)

(continued from previous page)

```

21 """
22 graph(%x.1 : int):
23   %2 : int = prim::Constant[value=1]() # ./1-ex.py:8:15
24   %3 : int = aten::add(%x.1, %2) # ./1-ex.py:8:11
25   return (%3)
26
27 -----
28 def adder(x: int) -> int:
29     return torch.add(x, 1)
30
31 4
32 """

```

torch.jit.script as a function

Listing 2: ./code/2-ex.py

```

1 def adder(x: int):
2     return x + 2
3
4
5 def test_adder():
6     adder_func = torch.jit.script(adder)
7     assert isinstance(adder_func, torch.jit.ScriptFunction)
8     print(adder_func.graph)
9     print(adder_func(3))
10
11
12 """
13 graph(%x.1 : int):
14   %2 : int = prim::Constant[value=2]() # ./2-ex.py:6:15
15   %3 : int = aten::add(%x.1, %2) # ./2-ex.py:6:11
16   return (%3)
17
18 5
19 """

```

torchscript a module

Listing 3: ./code/3-ex.py

```

1 class MyModel(torch.nn.Module):
2     def __init__(self):
3         super().__init__()
4         self.p = torch.nn.Parameter(torch.tensor([2.0]))
5
6     def forward(self, x: torch.Tensor):
7         return self.p * x
8

```

(continues on next page)

(continued from previous page)

```

9
10 def test_my_model():
11     model = MyModel()
12     scripted_model = torch.jit.script(model)
13     print(scripted_model.graph)
14     print("-" * 10)
15     print(scripted_model.code)
16     print(scripted_model(torch.tensor([10])))
17
18
19 """
20 graph(%self : __torch__.MyModel,
21       %x.1 : Tensor):
22     %p : Tensor = prim::GetAttr[name="p"](%self)
23     %4 : Tensor = aten::mul(%p, %x.1) # ./3-ex.py:12:15
24     return (%4)
25
26 -----
27 def forward(self,
28             x: Tensor) -> Tensor:
29     p = self.p
30     return torch.mul(p, x)
31 """

```

Export and ignore methods

1. Use `@torch.jit.export` decorator to export a method.
2. Use `torch.jit.export` function call to export a method.
3. Use `@torch.jit.ignore` decorator to ignore a method.
4. Use `torch.jit.ignore` function call to ignore a method.
5. Use `@torch.jit.unused` or `torch.jit.unused` to ignore a method.

See *Load in C++* to load the saved file.

Listing 4: `./code/4-ex.py`

```

1 class MyModel(torch.nn.Module):
2     def __init__(self):
3         super().__init__()
4         self.p = torch.nn.Parameter(torch.tensor([2.0]))
5
6     def foobar(self, x: torch.Tensor):
7         return x + 3
8
9     def foo(self, x: torch.Tensor):
10        return self.foobar(x)
11
12    def bar(self, x: torch.Tensor):
13        return self.p - x

```

(continues on next page)

```

14
15 @torch.jit.export
16 def baz(self, x: torch.Tensor):
17     return self.p + x + 2
18
19 def forward(self, x: torch.Tensor):
20     return self.p * x
21
22
23 def test_my_model():
24     MyModel.foo = torch.jit.export(MyModel.foo) # manually export
25
26     # Note: forward is exported by default. We ignore it here manually
27     MyModel.forward = torch.jit.ignore(MyModel.forward)
28
29     model = MyModel()
30     scripted_model = torch.jit.script(model)
31     assert hasattr(scripted_model, "foo")
32     assert hasattr(scripted_model, "baz")
33     assert hasattr(scripted_model, "foobar") # because it is called by `foo`
34     assert not hasattr(scripted_model, "bar")
35
36     scripted_model.save("foo.pt")
37
38     m = torch.jit.load("foo.pt")
39     print(m.foo(torch.tensor([1])))
40     print(m.baz(torch.tensor([1])))
41
42
43 """
44 graph(%self : __torch__.MyModel,
45       %x.1 : Tensor):
46     %p : Tensor = prim::GetAttr[name="p"](%self)
47     %4 : Tensor = aten::mul(%p, %x.1) # ./3-ex.py:12:15
48     return (%4)
49
50 -----
51 def forward(self,
52             x: Tensor) -> Tensor:
53     p = self.p
54     return torch.mul(p, x)
55 """

```

8.2.2 Load in C++

See https://pytorch.org/tutorials/advanced/cpp_export.html.

Load the saved `foo.pt` in C++ from *Export and ignore methods*.

Listing 5: `./code/load-in-cpp/Makefile`

```

1  USE_CXX11_ABI := $(shell python3 -c 'import torch; print(int(torch.compiled_with_cxx11_
   ↳abi()))')
2  TORCH_INSTALL_DIR := $(shell python3 -c 'import os; import torch; print(os.path.
   ↳dirname(torch.__file__))')
3
4  $(info USE_CXX11_ABI $(USE_CXX11_ABI))
5  $(info TORCH_INSTALL_DIR $(TORCH_INSTALL_DIR))
6
7  CXXFLAGS := -I$(TORCH_INSTALL_DIR)/include
8  CXXFLAGS += -I$(TORCH_INSTALL_DIR)/include/torch/csrc/api/include
9  CXXFLAGS += -I$(TORCH_INSTALL_DIR)/include/TH
10 CXXFLAGS += -I$(TORCH_INSTALL_DIR)/include/THC
11 CXXFLAGS += -std=c++14
12 CXXFLAGS += -D_GLIBCXX_USE_CXX11_ABI=$(USE_CXX11_ABI)
13
14 CXXFLAGS += -Wno-unknown-pragmas # disable omp warnings
15
16 LDFLAGS := -L$(TORCH_INSTALL_DIR)/lib
17 LDFLAGS += -lc10 -ltorch -ltorch_cpu
18 # LDFLAGS += -lc10 -ltorch
19 LDFLAGS += -Wl,-rpath,$(TORCH_INSTALL_DIR)/lib
20
21 HAS_CUDA := $(shell python3 -c 'import torch; print("yes" if torch.cuda.is_available()_
   ↳else "no")')
22 $(info has cuda $(HAS_CUDA))
23
24 ifeq ($(HAS_CUDA),yes)
25 CUDA_HOME := $(shell which nvcc | xargs dirname | xargs dirname)
26 CXXFLAGS += -I$(CUDA_HOME)/include
27 LDFLAGS += -L$(CUDA_HOME)/lib64
28 LDFLAGS += -lcudart -lc10_cuda -ltorch_cuda
29 LDFLAGS += -Wl,-rpath,$(CUDA_HOME)/lib64
30 endif
31
32 .PHONY: clean
33
34 main: main.o
35     $(CXX) -o $@ $< $(LDFLAGS)
36
37 main.o: main.cc
38     $(CXX) $(CXXFLAGS) -c -o $@ $<
39
40 clean:
41     $(RM) main.o main

```

Note: `torch::jit::script::Module` is deprecated, use `torch::jit::Module` instead.

Listing 6: `./code/load-in-cpp/main.cc`

```

1  #include "torch/script.h"
2
3  int main() {
4      // see torch/csrc/jit/module.h
5      torch::jit::Module m = torch::jit::load("../foo.pt");
6      std::cout << "is training: " << m.is_training() << "\n";
7      m.eval();
8      std::cout << "after m.eval(): is training: " << m.is_training() << "\n";
9      torch::Tensor x = torch::tensor({1, 2, 3}, torch::kFloat);
10     torch::Tensor y = m.run_method("baz", x).toTensor();
11     std::cout << y << "\n";
12
13     return 0;
14 }

```

The output of make is:

```

USE_CXX11_ABI 0
TORCH_INSTALL_DIR /ceph-fj/fangjun/software/py38/lib/python3.8/site-packages/torch
has cuda yes
g++ -I/ceph-fj/fangjun/software/py38/lib/python3.8/site-packages/torch/include \
    -I/ceph-fj/fangjun/software/py38/lib/python3.8/site-packages/torch/include/torch/
↪csrc/api/include \
    -I/ceph-fj/fangjun/software/py38/lib/python3.8/site-packages/torch/include/TH \
    -I/ceph-fj/fangjun/software/py38/lib/python3.8/site-packages/torch/include/THC \
    -std=c++14 \
    -D_GLIBCXX_USE_CXX11_ABI=0 \
    -Wno-unknown-pragmas \
    -I/ceph-sh1/fangjun/software/cuda-10.2.89/include \
    -c -o main.o main.cc
g++ -o main main.o \
    -L/ceph-fj/fangjun/software/py38/lib/python3.8/site-packages/torch/lib \
    -lc10 -ltorch -ltorch_cpu \
    -Wl,-rpath,/ceph-fj/fangjun/software/py38/lib/python3.8/site-packages/torch/lib \
    -L/ceph-sh1/fangjun/software/cuda-10.2.89/lib64 \
    -lcudart -lc10_cuda -ltorch_cuda \
    -Wl,-rpath,/ceph-sh1/fangjun/software/cuda-10.2.89/lib64

```

The output of `./main` is:

```

is training: 1
after m.eval(): is training: 0
5
6
7
[ CPUFloatType{3} ]

```

8.2.3 ArrayRef

See `c10/utils/ArrayRef.h`.

Caution: `IntArrayRef` is an alias to `ArrayRef<int64_t>`.

`ArrayRef<T>` contains only two members: A const data pointer and a size. It is trivially copyable and assignable.

It has similar methods like `std::vector`. It also has two methods to get the front and back: `front()` and `back()`; both return a const reference.

Its method `vec()` converts itself to a `std::vector` by **copying** the underlying data.

Constructors

Data members

Listing 7: `./code/array_ref/main.cc` (Check size)

```
1 struct Foo {
2     const int32_t *p;
3     size_t len;
4 };
5
6 static void TestSize() {
7     // Note: The data pointer in ArrayRef is const!
8     static_assert(sizeof(torch::ArrayRef<int32_t>) == sizeof(Foo), "");
9 }
```

Default constructed

Listing 8: ./code/array_ref/main.cc (Default constructor)

```
1 static void TestDefaultConstructor() {
2     torch::ArrayRef<int32_t> a;
3     TORCH_CHECK(a.data() == nullptr);
4     TORCH_CHECK(a.size() == 0);
5     TORCH_CHECK(a.empty() == true);
6
7     TORCH_CHECK(a.begin() == nullptr);
8     TORCH_CHECK(a.end() == nullptr);
9 }
```

From a single element

Listing 9: ./code/array_ref/main.cc (From a single element)

```
1 static void TestFromSingleElement() {
2     int32_t a = 10;
3     torch::ArrayRef<int32_t> b(a);
4     TORCH_CHECK(b[0] == a);
5     TORCH_CHECK(b.data() == &a);
6     TORCH_CHECK(b.size() == 1);
7 }
```

From an initializer list

Listing 10: ./code/array_ref/main.cc (From an initializer list)

```
1 static void TestFromInitializerList() {
2     torch::ArrayRef<int32_t> a = {1, 2, 3};
3     TORCH_CHECK(a.size() == 3);
4     TORCH_CHECK(a[0] == 1);
5     TORCH_CHECK(a[1] == 2);
6     TORCH_CHECK(a[2] == 3);
7 }
```

Other types of constructors

- From two pointers: begin and end
- From a pointer and a length
- From a *std::vector*
- From a container that has `data()` and `size()` methods
- From a C array
- From a *std::array*

8.2.4 ScalarType

See `c10/core/ScalarType.h` and <https://github.com/pytorch/pytorch/blob/master/torch/csrc/api/include/torch/types.h>.

`ScalarType` is an enum class, i.e., `enum class ScalarType : int8_t { ... }`.

Members

It has the following members:

Listing 11: `./code/scalar-type/members.cc`

```

1  #define AT_FORALL_SCALAR_TYPES_WITH_COMPLEX_EXCEPT_COMPLEX_HALF(_) \
2      _(uint8_t, Byte) \
3      _(int8_t, Char) \
4      _(int16_t, Short) \
5      _(int, Int) \
6      _(int64_t, Long) \
7      _(at::Half, Half) \
8      _(float, Float) \
9      _(double, Double) \
10     _(c10::complex<float>, ComplexFloat) \
11     _(c10::complex<double>, ComplexDouble) \
12     _(bool, Bool) \
13     _(at::BFloat16, BFloat16)

```

Some aliases

Listing 12: `./code/scalar-type/main.cc (alias)`

```

1  static void TestAlias() {
2      static_assert(c10::ScalarType::Int == c10::kInt, "");
3      static_assert(c10::ScalarType::Byte == c10::kByte, "");
4  }

```

Listing 13: `./code/scalar-type/alias.cc`

```

1  // See torch/csrc/api/include/torch/types.h
2  using Dtype = at::ScalarType;
3
4  /// Fixed width dtypes.
5  constexpr auto kUInt8 = at::kByte;
6  constexpr auto kInt8 = at::kChar;
7  constexpr auto kInt16 = at::kShort;
8  constexpr auto kInt32 = at::kInt;
9  constexpr auto kInt64 = at::kLong;
10 constexpr auto kFloat16 = at::kHalf;
11 constexpr auto kFloat32 = at::kFloat;
12 constexpr auto kFloat64 = at::kDouble;
13
14 /// Rust-style short dtypes.
15 constexpr auto kU8 = kUInt8;

```

(continues on next page)

(continued from previous page)

```
16 constexpr auto kI8 = kInt8;
17 constexpr auto kI16 = kInt16;
18 constexpr auto kI32 = kInt32;
19 constexpr auto kI64 = kInt64;
20 constexpr auto kF16 = kFloat16;
21 constexpr auto kF32 = kFloat32;
22 constexpr auto kF64 = kFloat64;
```

ScalarType to CPP type

Listing 14: ./code/scalar-type/main.cc

```
1 static void TestScalarTypeToCppType() {
2     static_assert(
3         std::is_same<
4             int32_t, //
5             c10::impl::ScalarTypeToCPPTType<c10::ScalarType::Int>::type>::value,
6         "");
7 }
```

CPP type to ScalarType

Listing 15: ./code/scalar-type/main.cc

```
1 static void TestCppTypeToScalarType() {
2     static_assert(
3         c10::CppTypeToScalarType<float>::value == c10::ScalarType::Float, "");
4 }
```

Note: It is `c10::impl::ScalarTypeToCPPTType`, but it is `c10::CppTypeToScalarType`.

8.2.5 TypeMeta

See

- <https://github.com/pytorch/pytorch/blob/master/c10/util/typeid.h>
- <https://github.com/pytorch/pytorch/blob/master/c10/core/ScalarTypeToTypeMeta.h>

`struct TypeMeta` contains only a single `int16_t` data member:

Listing 16: ./code/type-meta/main.cc (Check size)

```
1 static void TestSize() {
2     static_assert(sizeof(caffe2::TypeMeta) == sizeof(int16_t), "");
3 }
```


Constructors

Listing 17: ./code/type-meta/main.cc (Make)

```

1 static void TestConstructor() {
2     caffe2::TypeMeta t = caffe2::TypeMeta::Make<int32_t>();
3     TORCH_CHECK(t.Match<int32_t>());
4
5     TORCH_CHECK(t.isScalarType());
6
7     TORCH_CHECK(t.isScalarType(torch::kInt));
8     TORCH_CHECK(t.isScalarType(torch::kFloat) == false);
9
10    TORCH_CHECK(t.name() == "int");
11 }

```

Operations with ScalarType

Listing 18: ./code/type-meta/main.cc (Operations with ScalarType)

```

1 static void TestFromScalarType() {
2     caffe2::TypeMeta t = caffe2::TypeMeta::fromScalarType(torch::kDouble);
3
4     TORCH_CHECK(t.isScalarType(torch::kDouble));
5     TORCH_CHECK(t.name() == "double");
6
7     TORCH_CHECK(t.toScalarType() == torch::kDouble);
8     TORCH_CHECK(t == torch::kDouble);
9     TORCH_CHECK(t != torch::kFloat);
10    TORCH_CHECK(torch::kInt != t);

```

8.2.6 torch::Device

See

- <https://github.com/pytorch/pytorch/blob/master/c10/core/DeviceType.h>
- <https://github.com/pytorch/pytorch/blob/master/c10/core/Device.h>

DeviceType

`torch::DeviceType` is defined as enum class `Device: int8_t {...}`. The most commonly used types are `torch::DeviceType::CPU` and `torch::DeviceType::CUDA`, which are aliased to `torch::kCPU` and `torch::kCUDA`.

Listing 19: ./code/device/main.cc

```

1 void TestDeviceType() {
2     torch::DeviceType d = torch::kCPU;
3     std::ostringstream os;
4     os << d;

```

(continues on next page)

(continued from previous page)

```

5 TORCH_CHECK(os.str() == "cpu");
6
7 TORCH_CHECK(DeviceTypeName(d /*,lower_case=false*/ ) == "CPU");
8 TORCH_CHECK(DeviceTypeName(d, /*lower_case*/ true) == "cpu");

```

Device

A `torch::Device` class has two members: a `torch::DeviceType` and an `int8_t index`.

Listing 20: `./code/device/main.cc` (Constructors)

```

1 void TestDeviceConstructorCPU() {
2     torch::Device d(torch::kCPU);
3     TORCH_CHECK(d.is_cpu() == true);
4     TORCH_CHECK(d.is_cuda() == false);
5     TORCH_CHECK(d.type() == torch::kCPU);
6     TORCH_CHECK(d.has_index() == false);
7     TORCH_CHECK(d.index() == -1);
8     TORCH_CHECK(d.str() == "cpu");
9 }
10
11 void TestDeviceConstructorCUDA() {
12     torch::Device d(torch::kCUDA, 3);
13     TORCH_CHECK(d.is_cpu() == false);
14     TORCH_CHECK(d.is_cuda() == true);
15     TORCH_CHECK(d.type() == torch::kCUDA);
16     TORCH_CHECK(d.has_index() == true);
17     TORCH_CHECK(d.index() == 3);
18     TORCH_CHECK(d.str() == "cuda:3");
19
20     d.set_index(2);
21     TORCH_CHECK(d.index() == 2);
22     TORCH_CHECK(d.str() == "cuda:2");
23
24     d = torch::Device("cpu");
25     TORCH_CHECK(d.is_cpu() == true);
26
27     d = torch::Device("CPU");
28     TORCH_CHECK(d.is_cpu() == true);
29
30     d = torch::Device("cuda:1");
31     TORCH_CHECK(d.is_cuda() == true);
32     TORCH_CHECK(d.index() == 1);
33
34     d = torch::Device("CUDA:1");
35     TORCH_CHECK(d.is_cuda() == true);
36     TORCH_CHECK(d.index() == 1);
37 }

```

8.2.7 TensorOptions

See <https://github.com/pytorch/pytorch/blob/master/c10/core/TensorOptions.h>

Constructors (not recommended)

Listing 21: ./code/tensor-options/main.cc (Not recommended constructors)

```

1 void TestConstructor() {
2     // not recommended
3     torch::TensorOptions opt1(torch::kCPU);
4     torch::TensorOptions opt2(torch::Device(torch::kCPU));
5     torch::TensorOptions opt3(torch::Device({torch::kCUDA, 1}));
6     torch::TensorOptions opt4("cpu");
7     // torch::TensorOptions opt5("CPU") // error;
8     torch::TensorOptions opt6("cuda:1");
9     // torch::TensorOptions opt7("CUDA:1"); // error
10
11     // not recommended, from a scalar type (implicit)
12     torch::TensorOptions opt8(torch::kInt32);
13 }

```

Constructors (Recommended)

Listing 22: ./code/tensor-options/main.cc (Recommended constructors)

```

1 void TestConstructor2() {
2     // recommended
3     torch::TensorOptions opt1 = torch::dtype(torch::kFloat);
4     torch::TensorOptions opt2 = torch::dtype(caffe2::TypeMeta::Make<float>());
5     torch::TensorOptions opt3 = torch::device(torch::kCPU);
6     torch::TensorOptions opt4 = torch::device({torch::kCUDA, 1});
7     // Note: torch::device() returns a TensorOptions
8     // while torch::Device() is the constructor of a class
9
10    torch::TensorOptions opt5 = torch::requires_grad(true);
11    std::cout << opt5 << "\n";
12    // TensorOptions(dtype=float (default), device=cpu (default), layout=Strided
13    // (default), requires_grad=true, pinned_memory=false (default),
14    // memory_format=(nullopt))
15
16    torch::TensorOptions opt6 = torch::dtype<float>();
17    std::cout << torch::toString(opt6) << "\n";
18    // TensorOptions(dtype=float, device=cpu (default), layout=Strided (default),
19    // requires_grad=false (default), pinned_memory=false (default),
20    // memory_format=(nullopt))
21
22    std::cout << "default:" << torch::TensorOptions() << "\n";
23    // default:TensorOptions(dtype=float (default), device=cpu (default),
24    // layout=Strided (default), requires_grad=false (default),

```

(continues on next page)

(continued from previous page)

```

25 // pinned_memory=false (default), memory_format=(nullopt))
26 }

```

Methods

Listing 23: ./code/tensor-options/main.cc (Methods)

```

1 void TestMethods() {
2     torch::TensorOptions opts = torch::dtype<float>();
3     TORCH_CHECK(opts.device() == torch::Device(torch::kCPU));
4     // It has not device_type!
5     TORCH_CHECK(opts.device() == torch::kCPU);
6     TORCH_CHECK(opts.device().type() == torch::kCPU);
7     TORCH_CHECK(opts.requires_grad() == false);
8
9     torch::TensorOptions opts2 =
10         opts.device("cuda:2").dtype(torch::kInt).requires_grad(false);
11
12     TORCH_CHECK(opts2.dtype() == caffe2::TypeMeta::Make<int32_t>());
13     TORCH_CHECK(opts2.dtype() == torch::kInt32);
14     TORCH_CHECK(opts2.requires_grad() == false);
15 }

```

8.2.8 Tensor Creation

See

TensorDataContainer

Note: data is **copied** to the returned tensor!

See

- <https://github.com/pytorch/pytorch/blob/master/torch/csrc/api/include/torch/detail/TensorDataContainer.h>
- https://github.com/pytorch/pytorch/blob/master/tools/autograd/templates/variable_factories.h
- <https://github.com/pytorch/pytorch/blob/master/aten/src/ATen/Utils.cpp>

Support the following data types:

- From a `std::vector<T>`
- From a scalar
- From an initializer list
- From an `ArrayRef<T>`.

8.2.9 Tensor

See

- <https://github.com/pytorch/pytorch/blob/master/aten/src/ATen/core/TensorBase.h>
- <https://github.com/pytorch/pytorch/blob/master/aten/src/ATen/templates/TensorBody.h>
- <https://github.com/pytorch/pytorch/blob/master/c10/core/TensorImpl.h>

Common methods

Listing 24: ./code/tensor/main.cc (Not recommended constructors)

```

1  static void TestCommonMethods() {
2      torch::Tensor t = torch::rand({2, 3, 4});
3
4      TORCH_CHECK(t.dim() == 3);           // 3-d tensor
5      TORCH_CHECK(t.ndimension() == t.dim()); // same
6      TORCH_CHECK(t.numel() == 2 * 3 * 4);
7      TORCH_CHECK(t.is_contiguous() == true);
8      TORCH_CHECK(t.contiguous().is_contiguous() == true);
9
10     t.fill_(10); // fill all entries to 0
11     t.zero_();   // zero out all entries
12
13     t = t.to(torch::kInt);
14     TORCH_CHECK(t.is_floating_point() == false);
15     TORCH_CHECK(t.is_signed() == true);
16
17     TORCH_CHECK(t.size(0) == 2);
18     TORCH_CHECK(t.size(1) == 3);
19     TORCH_CHECK(t.size(2) == 4);
20     TORCH_CHECK(t.sizes() == torch::ArrayRef<int64_t>({2, 3, 4}));
21
22     t = t.contiguous();
23     TORCH_CHECK(t.stride(0) == 3 * 4);
24     TORCH_CHECK(t.stride(1) == 4);
25     TORCH_CHECK(t.stride(2) == 1);
26     TORCH_CHECK(t.strides() == torch::ArrayRef<int64_t>({12, 4, 1}));
27
28     TORCH_CHECK(t.defined() == true);
29     {
30         torch::Tensor a;
31         TORCH_CHECK(a.defined() == false);
32         a = t;
33         TORCH_CHECK(a.defined() == true);
34         a.reset();
35         TORCH_CHECK(a.defined() == false);
36     }
37
38     t = t.to(torch::kShort);
39     TORCH_CHECK(t.itemsize() == sizeof(int16_t));
40     TORCH_CHECK(t.nbytes() == t.numel() * t.itemsize());

```

(continues on next page)

(continued from previous page)

```

41 TORCH_CHECK(t.itemsize() == t.element_size()); // same
42
43 TORCH_CHECK(t.scalar_type() == torch::kShort);
44 TORCH_CHECK(t.dtype() == caffe2::TypeMeta::Make<int16_t>());
45 TORCH_CHECK(t.dtype().toScalarType() == torch::kShort);
46
47 TORCH_CHECK(t.device() == torch::Device("cpu"));
48 TORCH_CHECK(t.device() == torch::Device(torch::kCPU));
49
50 // Note: t.device() return an instance of torch::Device
51 // t.get_device() returns the device index.
52 TORCH_CHECK(t.get_device() == t.device().index());
53
54 TORCH_CHECK(t.is_cpu() == true);
55 TORCH_CHECK(t.is_cuda() == false);
56
57 t = t.to(torch::kInt);
58 int32_t *p = t.data_ptr<int32_t>();
59 p[0] = 100;
60
61 torch::TensorAccessor<int32_t, 3> acc = t.accessor<int32_t, 3>();
62 TORCH_CHECK(acc[0][0][0] == p[0]);
63 p[12] = -2;
64 TORCH_CHECK(acc[1][0][0] == -2);
65
66 acc[1][1][2] = 3;
67 TORCH_CHECK(*(p + 12 + 4 + 2) == 3);
68
69 t = t.to(torch::kFloat);
70 t.set_requires_grad(true);
71 TORCH_CHECK(t.requires_grad() == true);
72
73 t.set_requires_grad(false);
74 TORCH_CHECK(t.requires_grad() == false);
75
76 t = t.cuda();
77 TORCH_CHECK(t.device().type() == torch::kCUDA);
78 t = t.cpu();
79
80 torch::TensorOptions opts = t.options();
81 TORCH_CHECK(opts.device() == t.device());
82 }

```

9.1 asyncio

9.1.1 Hello World

9.1.2 References

- PEP 234 – Iterators
<https://peps.python.org/pep-0234/>
- Why does defining `__getitem__` on a class make it iterable in python?
<https://localcoder.org/why-does-defining-getitem-on-a-class-make-it-iterable-in-python>
- PEP 255 – Simple Generators
<https://peps.python.org/pep-0255/>
- Curious Course on Coroutines and Concurrency
https://www.youtube.com/watch?v=Z_OAlhXziw&ab_channel=DavidBeazley
By David Beazley.
- Generator Tricks for Systems Programmers
<https://www.dabeaz.com/generators2/>
- Generators: The Final Frontier
<https://www.youtube.com/watch?v=5-qadlG7tWo&ab_channel=DavidBeazley>
By David Beazley.

9.2 argv

From the doc <https://docs.python.org/3/library/sys.html>:

The `list` of command line arguments passed to a Python script. `argv[0]` **is** the script name (it **is** operating system dependent whether this **is** a full pathname **or not**). If the command was executed using the `-c` command line option to the interpreter, `argv[0]` **is set** to the string `'-c'`. If no script name was passed to the Python interpreter, `argv[0]` **is** the empty string.

Note that `argv` is at least of size 1, though `argv[0]` may be an empty string.

```
import sys
print(sys.argv)
```


10.1 Install

10.1.1 formatter

Install <https://github.com/google/google-java-format>

```
wget https://github.com/google/google-java-format/releases/download/v1.15.0/google-java-format-1.15.0-all-deps.jar
```

Create a script with filename `google-java-format`:

```
#!/usr/bin/env bash

java -jar /ceph-sh0/fangjun/download/google-java-format-1.15.0-all-deps.jar $@
```

`chmod +x google-java-format` and add the path to `PATH`.

10.1.2 JDK

Go to <https://www.oracle.com/java/technologies/downloads/#java17> and download

```
wget https://download.oracle.com/java/17/latest/jdk-17_linux-x64_bin.tar.gz
mkdir /ceph-fj/fangjun/software/
tar xvf jdk-17_linux-x64_bin.tar.gz -C /ceph-fj/fangjun/software
```

And then set the following environment variables:

```
export JAVA_HOME=/ceph-fj/fangjun/software/jdk-17.0.3
export PATH=$JAVA_HOME/bin:$JAVA_HOME
```

10.2 Hello world

Listing 1: Hello.java

```
// Usage 1:
//  java Hello.java
// Usage 2:
//  javac Hello.java
//  java Hello
//
// Note:
//  - "javac Hello.java" generates a file "Hello.class"
//  - "java Hello" takes as input "Hello.class" and executes it
//
class Hello {
    public static void main(String[] args) {
        System.out.println("hello world");
    }
} // There is no ';' here
```

Listing 2: EqualTest.java

```
class EqualTest {
    public int i;

    public EqualTest(int a) {
        this.i = a;
    }

    public boolean equals(Object anObject) {
        if (this == anObject) {
            return true;
        }
        if (anObject instanceof EqualTest) {
            return this.i == ((EqualTest) anObject).i;
        }
        return false;
    }

    public static void main(String[] args) {
        EqualTest e1 = new EqualTest(10);
        EqualTest e2 = new EqualTest(10);

        System.out.println(e1 == e2); // false, compare the reference
        System.out.println(e1 != e2); // true
        System.out.println(e1.equals(e2)); // true, compare the contained value
    }
}
```

10.3 Reference

- <https://docs.oracle.com/javase/tutorial/>
- <https://docs.oracle.com/en/java/javase/17/docs/api/index.html>
- <https://github.com/openjdk/jdk.git>

Clone it and you can find the source code in `src/java.base/share/classes/java/lang/System.java` for `java.lang.System`.

JAVASCRIPT

11.1 Hello world

```
console.log('hello world')  
console.log(eval('3 + 5'))
```

To write multi-line javascript, use shift + Enter for a new line.

```
(function(){  
    "use strict";  
    /* Start of your code */  
    function greetMe(yourName) {  
        alert('Hello ' + yourName);  
    }  
  
    greetMe('World');  
    /* End of your code */  
})();
```

It is case sensitive. Statements are separated by ;. Comments are the same as in C/C++.

11.2 TODOs

1. This page https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/Installing_basic_software lists some tools to minify code:
 - WebPack: <https://webpack.js.org/>
 - Grunt: <https://gruntjs.com/>
 - Gulp: <https://gulpjs.com/>
2. Color picker tool: https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Colors/Color_picker_tool
3. Google font: <https://fonts.google.com/> and https://developers.google.com/fonts/docs/getting_started

12.1 Hello world

Listing 1: hello_world.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Hello World</title>
  </head>
  <body>
    <p>Hello world</p>
  </body>
</html>
```

12.1.1 comments

```
<!-- this is a comment -->
```

12.1.2 images

```
</img>
</img>
</img>
```

12.1.3 ordered lists

```
<p> The following points </p>

<ol>
  <li> First </li>
  <li> Second </li>
</ol>
```

12.1.4 unordered lists

```
<p> The following points </p>

<ul>
  <li> foo </li>
  <li> bar </li>
</ul>
```

12.1.5 links

```
<a href="https://www.google.com">some text</a>
```

12.2 References

- Structuring the web with HTML
<https://developer.mozilla.org/en-US/docs/Learn/HTML>

13.1 Hello world

13.1.1 comment

```
/* this is a comment */
```

```
p { color: red; }
```

Then, in some html file, use:

```
<link href="abc/foo.css" rel="stylesheet">
```

13.1.2 Selector

- tag name or element name: e.g., p selects <p>; h1 selects <h1>.
- ID:, e.g., #my-id selects or <p id="my-id">
- class: e.g., .my-class selects and <p class="my-class">
- attribute: e.g., img[src] selects but not

See https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/CSS_basics#different_types_of_selectors and https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Selectors for more.

Example with multiple rules:

Listing 1: Example with multiple rules

```
p {  
  color: red;  
  width: 500px;  
  border: 1px solid black;  
}
```

Example with multiple selectors:

13.2 References

- CSS basics

https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/CSS_basics