
notes

fangjun

Jul 09, 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++	22
8.2.3	ArrayRef	24
8.2.4	ScalarType	26

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	35
8.2.10	intrusive_ptr	39
8.2.11	optional	39
8.2.12	PackedSequence	39
8.2.13	ivalue	41
8.3	Logical operations	43
8.4	Note	44
8.5	Quantization	44
8.5.1	References	44
9	Python	45
9.1	asyncio	45
9.1.1	iterator	45
9.1.2	yield	45
9.1.3	Hello World	45
9.1.4	References	46
9.1.5	TODOs	46
9.2	argv	46
9.3	TODO	47
9.4	time	47
9.5	Numbers	47
9.5.1	binary representation	47
9.6	str	48
9.6.1	format	48
9.7	enum	48
9.7.1	Hello	48
9.8	socket	51
9.8.1	AddressFamily	51
9.8.2	SocketKind	52
9.8.3	struct sockaddr_in	52
9.8.4	AddressInfo	53
9.8.5	inet_pton	54
9.8.6	inet_ntop	59
9.8.7	Echo Server and Client	63
9.8.8	TODOs	66
10	java	67
10.1	Install	67
10.1.1	formatter	67
10.1.2	JDK	67
10.2	Hello world	68
10.3	Reference	69
11	javascript	71
11.1	Hello world	71
11.1.1	array	71
11.1.2	class	72
11.2	node	72
11.3	TODOs	73
12	HTML	75

12.1	Hello world	75
12.1.1	comments	75
12.1.2	images	75
12.1.3	ordered lists	75
12.1.4	unordered lists	76
12.1.5	links	76
12.2	References	76
13	css	77
13.1	Hello world	77
13.1.1	comment	77
13.1.2	Selector	77
13.2	References	78
14	pybind11	79
14.1	GIL	79
15	Protocol Buffers	81
15.1	Installation	81
15.1.1	C++	81
15.2	Hello	89
15.2.1	hello.proto	89
15.2.2	makefile	90
15.2.3	hello.pb.h	90
15.2.4	hello.pb.cc	111
16	gRPC	131
16.1	Install	131
17	lwn.net	133
17.1	TODOs	133
18	Linker and Loader	135
18.1	References	135
18.2	Questions	136
19	espnet	137
19.1	aishell	137
19.1.1	AM training	137
20	cmake	139
20.1	Tutorials	139
20.2	Install	139
21	EECS E6870 Speech Recognition	141
21.1	Notes	141

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
```


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
21 """
22 graph(%x.1 : int):
23     %2 : int = prim::Constant[value=1]() # ./1-ex.py:8:15
```

(continues on next page)

(continued from previous page)

```

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
9
10 def test_my_model():
11     model = MyModel()

```

(continues on next page)

(continued from previous page)

```

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
14
15    @torch.jit.export
16    def baz(self, x: torch.Tensor):

```

(continues on next page)

(continued from previous page)

```

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__))')

```

(continues on next page)

(continued from previous page)

```

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()
22 ↪ else "no")')
23 $(info has cuda $(HAS_CUDA))
24
25 ifeq ($(HAS_CUDA),yes)
26 CUDA_HOME := $(shell which nvcc | xargs dirname | xargs dirname)
27 CXXFLAGS += -I$(CUDA_HOME)/include
28 LDFLAGS += -L$(CUDA_HOME)/lib64
29 LDFLAGS += -lcudart -lc10_cuda -ltorch_cuda
30 LDFLAGS += -Wl,-rpath,$(CUDA_HOME)/lib64
31 endif
32
33 .PHONY: clean
34
35 main: main.o
36     $(CXX) -o $@ $< $(LDFLAGS)
37
38 main.o: main.cc
39     $(CXX) $(CXXFLAGS) -c -o $@ $<
40
41 clean:
42     $(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");

```

(continues on next page)

(continued from previous page)

```

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;  
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::ScalarTypeToCppType<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::ScalarTypeToCppType`, 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);

```

(continues on next page)

(continued from previous page)

```
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);
11 }
```

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;
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);

```

(continues on next page)

(continued from previous page)

```

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),

```

(continues on next page)

(continued from previous page)

```

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),
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>`.

From `std::vector`

Listing 24: `./code/tensor-creation/main.cc`

```
1 static void FromStdVecotr() {
2     torch::Tensor t1 = torch::tensor(std::vector<int32_t>{1, 2, 3});
3     TORCH_CHECK(t1.scalar_type() == torch::kLong);
4     t1 = t1.to(torch::kInt);
5     const int32_t *p1 = t1.data_ptr<int32_t>();
6     TORCH_CHECK(p1[0] == 1);
7     TORCH_CHECK(p1[1] == 2);
8     TORCH_CHECK(p1[2] == 3);
9
10    torch::Tensor t2 = torch::tensor(std::vector<float>{1, 2, 3});
11    TORCH_CHECK(t2.scalar_type() == torch::kFloat);
12
13    torch::Tensor t3 =
14        torch::tensor(std::vector<double>{1, 2, 3}, torch::kDouble);
15    TORCH_CHECK(t3.scalar_type() == torch::kDouble);
16
17    torch::Tensor t4 =
18        torch::tensor(std::vector<double>{1, 2, 3},
19                      torch::dtype(torch::kDouble).device("cuda:0"));
20    TORCH_CHECK(t4.is_cuda());
21 }
```

From scalar

Listing 25: ./code/tensor-creation/main.cc

```
1 static void FromScalar() {  
2     torch::Tensor t = torch::tensor(3);  
3     TORCH_CHECK(t.item<int64_t>() == 3);  
4  
5     torch::Tensor t2 = torch::tensor(0.5);  
6     TORCH_CHECK(t2.scalar_type() == torch::kFloat);  
7 }
```

From initializer list

Listing 26: ./code/tensor-creation/main.cc

```
1 static void FromInitializerList() {  
2     torch::Tensor t1 = torch::tensor({1, 2, 3});  
3     torch::Tensor t2 = torch::tensor(std::vector<int32_t>{1, 2, 3});  
4     TORCH_CHECK(torch::allclose(t1, t2));  
5  
6     torch::Tensor t3 = torch::tensor({{1, 2, 3}, {4, 5, 6}});  
7     TORCH_CHECK(t3.dim() == 2);  
8  
9     torch::Tensor t4 = torch::tensor({1, 2, 3});  
10    torch::Tensor t5 = torch::tensor({4, 5, 6});  
11    TORCH_CHECK(torch::allclose(t3[0], t4));  
12    TORCH_CHECK(torch::allclose(t3[1], t5));  
13 }
```

From ArrayRef

Listing 27: ./code/tensor-creation/main.cc

```

1 static void FromArrayRef() {
2     int32_t i[] = {1, 2, 3};
3     torch::ArrayRef<int32_t> a(i);
4     torch::Tensor t = torch::tensor(a);
5     // Data is copied to t
6
7     TORCH_CHECK(t[0].item<int64_t>(), 1);
8     TORCH_CHECK(t[1].item<int64_t>(), 2);
9     TORCH_CHECK(t[2].item<int64_t>(), 3);
10 }

```

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 28: ./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 10
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);

```

(continues on next page)

(continued from previous page)

```

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());
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);

```

(continues on next page)

(continued from previous page)

```
78 t = t.cpu();
79
80 torch::TensorOptions opts = t.options();
81 TORCH_CHECK(opts.device() == t.device());
82 }
```

slice

Listing 29: torch::slice

```
1 static void TestSlice() {
2     auto t = torch::tensor({1, 2, 3, 4, 5}, torch::kInt);
3     torch::TensorAccessor<int32_t, 1> acc = t.accessor<int32_t, 1>();
4
5     // t2 = t[1:3]
6     torch::Tensor t2 = t.slice(/*dim*/ 0, /*start*/ 1,
7                               /*end, exclusive*/ 3); // memory is shared
8     torch::TensorAccessor<int32_t, 1> acc2 = t2.accessor<int32_t, 1>();
9     TORCH_CHECK(acc2[0] == 2);
10    TORCH_CHECK(acc2[1] == 3);
11
12    acc2[0] = 10; // also changes t since the memory is shared
13    TORCH_CHECK(acc[1] == 10);
14 }
```

topk

Listing 30: torch::topk

```

1 // https://pytorch.org/docs/stable/generated/torch.topk.html
2 static void TestTopK() {
3     auto t = torch::tensor({1, 0, 3, -1}, torch::kInt).to(torch::kFloat);
4     torch::Tensor values, indexes;
5     std::tie(values, indexes) =
6         t.topk(/*k*/ 2, /*dim*/ 0, /*largest*/ true, /*sorted*/ true);
7     auto values_acc = values.accessor<float, 1>();
8     auto indexes_acc = indexes.accessor<int64_t, 1>(); // Note: it is int64_t
9
10    TORCH_CHECK(values.numel() == 2); // k in topk is 2
11    TORCH_CHECK(values_acc[0] == 3); // the largest value is 3, at t[2]
12    TORCH_CHECK(values_acc[1] == 1); // the second largest value is 1, at t[0]
13    //
14    TORCH_CHECK(indexes_acc[0] == 2); // the largest value is t[2]
15    TORCH_CHECK(indexes_acc[1] == 0); // the second largest value is t[0]
16 }

```

floor_divide

Listing 31: torch::floor_divide

```

1 static void TestFloorDivide() {
2     auto t = torch::tensor({1, 0, 3, 5, 9}, torch::kInt);
3     auto p = torch::floor_divide(t, 2);
4     auto acc = p.accessor<int32_t, 1>();
5     TORCH_CHECK(acc[0] == 1 / 2);
6     TORCH_CHECK(acc[1] == 0 / 2);
7     TORCH_CHECK(acc[2] == 3 / 2);
8     TORCH_CHECK(acc[3] == 5 / 2);
9     TORCH_CHECK(acc[4] == 9 / 2);
10 }

```

div

Listing 32: torch::div

```

1 // https://pytorch.org/docs/stable/generated/torch.div.html
2 static void TestDiv() {
3     auto t = torch::tensor({1, 0, 3, 5, 9}, torch::kInt);
4     // the rounding mode is supported in torch >= 1.8.0
5     auto p = torch::div(t, 2, /*rounding_mode*/ "trunc");
6     auto acc = p.accessor<int32_t, 1>();
7     TORCH_CHECK(acc[0] == 1 / 2);
8     TORCH_CHECK(acc[1] == 0 / 2);
9     TORCH_CHECK(acc[2] == 3 / 2);
10    TORCH_CHECK(acc[3] == 5 / 2);
11    TORCH_CHECK(acc[4] == 9 / 2);
12 }

```

remainder

Listing 33: torch::remainder

```

1 static void TestRemainder() {
2     auto t = torch::tensor({1, 3, 8}, torch::kInt);
3     auto p = torch::remainder(t, 3);
4     auto acc = p.accessor<int32_t, 1>();
5     TORCH_CHECK(acc[0] == 1);
6     TORCH_CHECK(acc[1] == 0);
7     TORCH_CHECK(acc[2] == 2);
8 }

```

empty

Listing 34: torch::empty

```

1 static void TestEmpty() {
2     auto t = torch::empty({3}, torch::kInt);
3     TORCH_CHECK(t.scalar_type() == torch::kInt);
4     TORCH_CHECK(t.numel() == 3);
5 }

```

8.2.10 intrusive_ptr

8.2.11 optional

8.2.12 PackedSequence

See

- <https://github.com/pytorch/pytorch/blob/master/torch/csrc/api/include/torch/nn/utils/rnn.h>
- <https://github.com/pytorch/pytorch/blob/master/torch/nn/utils/rnn.py>

pack_padded_sequence

Listing 35: ./code/packed-sequence/main.cc

```

1 static void TestPadPackedSequence() {
2     torch::Tensor t = torch::tensor({
3         {{10, 20, 30}, {0, 0, 0}, {0, 0, 0}},
4         {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}},
5         {{-1, 2, 3}, {-4, 5, 6}, {0, 0, 0}},
6     });
7     torch::Tensor lengths = torch::tensor({1, 3, 2});
8     torch::nn::utils::rnn::PackedSequence packed_seq =
9         torch::nn::utils::rnn::pack_padded_sequence(
10         t, lengths, /*batch_first*/ true, /*enforce_sorted*/ false);
11     std::cout << "data: " << packed_seq.data() << "\n";

```

(continues on next page)

(continued from previous page)

```

12     std::cout << "batch_sizes: " << packed_seq.batch_sizes() << "\n";
13     std::cout << "sorted_indices: " << packed_seq.sorted_indices() << "\n";
14     std::cout << "unsorted_indices: " << packed_seq.unsorted_indices() << "\n";
15 }
16 /*
17 data:   1   2   3
18  -1   2   3
19  10  20  30
20   4   5   6
21  -4   5   6
22   7   8   9
23 [ CPULongType{6,3} ]
24 batch_sizes:  3
25   2
26   1
27 [ CPULongType{3} ]
28 sorted_indices: 1
29   2
30   0
31 [ CPULongType{3} ]
32 unsorted_indices: 2
33   0
34   1
35 [ CPULongType{3} ]
36 */

```

The output is

Listing 36: ./code/packed-sequence/main.cc

```

1 }
2 /*
3 data:   1   2   3
4  -1   2   3
5  10  20  30
6   4   5   6
7  -4   5   6
8   7   8   9
9 [ CPULongType{6,3} ]
10 batch_sizes:  3
11   2
12   1
13 [ CPULongType{3} ]
14 sorted_indices: 1
15   2

```

8.2.13 ivalue

Listing 37: ./code/ivalue/main.cc

```

1  #include "torch/script.h"
2
3  static void TestVectorOfTensor() {
4      torch::jit::Module m("m");
5      m.define(R"(
6          def forward(self, x, y):
7              return [x, y]
8      )");
9      auto x = torch::tensor({1, 2, 3});
10     auto y = torch::tensor({4, 5, 6});
11     auto i = m.run_method("forward", x, y);
12
13     assert(i.tagKind() == "GenericList");
14
15     torch::ArrayRef<torch::IValue> tensor_list = i.toListRef();
16     TORCH_CHECK(torch::allclose(x, tensor_list[0].toTensor()));
17     TORCH_CHECK(torch::allclose(y, tensor_list[1].toTensor()));
18
19     torch::List<torch::IValue> k = i.toList();
20
21     torch::List<torch::Tensor> o =
22         c10::impl::toTypedList<torch::Tensor>(std::move(k));
23
24     TORCH_CHECK(torch::allclose(o[0], x));
25     TORCH_CHECK(torch::allclose(o[1], y));
26
27     std::vector<torch::Tensor> p = o.vec();
28     TORCH_CHECK(torch::allclose(p[0], x));
29     TORCH_CHECK(torch::allclose(p[1], y));
30 }
31
32 static void TestVectorOfTensor2() {
33     torch::jit::Module m("m");
34     m.define(R"(
35         def forward(self, x):
36             return [[x], [x,x]]
37     )");
38     auto x = torch::tensor({1, 2, 3});
39     auto i = m.run_method("forward", x);
40     TORCH_CHECK(i.tagKind() == "GenericList");
41
42     torch::List<torch::IValue> list = i.toList();
43     torch::Tensor a = list.get(0).toListRef()[0].toTensor();
44     TORCH_CHECK(torch::allclose(a, x));
45
46     std::vector<torch::Tensor> b =
47         c10::impl::toTypedList<torch::Tensor>(list.get(1).toList()).vec();
48     TORCH_CHECK(torch::allclose(b[0], x));
49     TORCH_CHECK(torch::allclose(b[1], x));

```

(continues on next page)

(continued from previous page)

```

50 }
51
52 static void TestVectorOfTensor3() {
53     torch::jit::Module m("m");
54     m.define(R"(
55         def forward(self, x: List[torch.Tensor]):
56             return x[0] + x[1]
57     )");
58
59     std::vector<torch::Tensor> v;
60     v.push_back(torch::tensor({1, 2}));
61     v.push_back(torch::tensor({3, 4}));
62     c10::List<torch::Tensor> ilist(v);
63
64     c10::impl::GenericList generic_list = c10::impl::toList(ilist);
65
66     c10::List<torch::Tensor> l2 =
67         c10::impl::toTypedList<torch::Tensor>(generic_list);
68
69     TORCH_CHECK(torch::allclose(l2[0], v[0]));
70     TORCH_CHECK(torch::allclose(l2[1], v[1]));
71
72     auto r = m.run_method("forward", generic_list);
73     TORCH_CHECK(torch::allclose(r.toTensor(), v[0] + v[1]));
74
75     // Note: We can pass a vector directly
76     r = m.run_method("forward", v);
77     TORCH_CHECK(torch::allclose(r.toTensor(), v[0] + v[1]));
78
79     r = m.run_method("forward", ilist); // also OK
80     TORCH_CHECK(torch::allclose(r.toTensor(), v[0] + v[1]));
81 }
82
83 static void TestVectorOfTensor4() {
84     torch::jit::Module m("m");
85     m.define(R"(
86         def forward(self, x: Tuple[List[torch.Tensor]]):
87             return x[0][0] + x[0][1]
88     )");
89
90     std::vector<torch::Tensor> v;
91     v.push_back(torch::tensor({1, 2}));
92     v.push_back(torch::tensor({3, 4}));
93     auto t = torch::ivalue::Tuple::create(v);
94
95     auto r = m.run_method("forward", t);
96     TORCH_CHECK(torch::allclose(r.toTensor(), v[0] + v[1]));
97 }
98
99 static void TestVectorOfTensor5() {
100     torch::jit::Module m("m");
101     m.define(R"(

```

(continues on next page)

(continued from previous page)

```

102     def forward(self, x: Tuple[List[List[torch.Tensor]], List[torch.Tensor]]):
103         return x[0][0][0] + x[0][0][1] + x[1][0] + x[1][1]
104     );
105
106     std::vector<torch::Tensor> v;
107     v.push_back(torch::tensor({1, 2}));
108     v.push_back(torch::tensor({3, 4}));
109
110     std::vector<std::vector<torch::Tensor>> vv;
111     vv.push_back(v);
112     vv.push_back(v);
113
114     auto t = torch::ivalue::Tuple::create(vv, v);
115
116     auto r = m.run_method("forward", t);
117     TORCH_CHECK(torch::allclose(r.toTensor(), v[0] + v[1] + v[0] + v[1]));
118 }
119
120 int main() {
121     TestVectorOfTensor();
122     TestVectorOfTensor2();
123     TestVectorOfTensor3();
124     TestVectorOfTensor4();
125     TestVectorOfTensor5();
126     return 0;
127 }

```

8.3 Logical operations

Listing 38: ./code/logical-op.py

```

1  #!/usr/bin/env python3
2
3  import torch
4
5  a = torch.tensor([float("inf")])
6  b = torch.tensor([float("nan")])
7  assert torch.isinf(a).item() is True
8  assert torch.isnan(a).item() is False
9
10 assert torch.isinf(b).item() is False
11 assert torch.isnan(b).item() is True
12
13 assert torch.logical_or(torch.isinf(a), torch.isnan(b)).item() is True
14
15 assert a.isinf().item() is True
16 assert a.isnan().item() is False
17
18 assert b.isinf().item() is False
19 assert b.isnan().item() is True

```

8.4 Note

To clip gradient, use:

```
tot_norm = torch.nn.utils.clip_grad_norm_(model.parameters(), max_norm=5, norm_type=2.0)
if torch.logical_or(tot_norm.isnan(), tot_norm.isinf()):
    # skip this update
    continue
else:
    optimizer.step()
```

8.5 Quantization

8.5.1 References

- <https://pytorch.org/docs/stable/quantization.html>

9.1 asyncio

9.1.1 iterator

See <https://peps.python.org/pep-0234/>

9.1.2 yield

9.1.3 Hello World

Exercise 1

Listing 1: ./code/hello_world/ex1.py

```
1 import asyncio
2
3
4 async def hello():
5     print("hello world")
6
7
8 asyncio.run(hello())
```

Exercise 2

Listing 2: ./code/hello_world/ex2.py

```
1 import asyncio
2 import time
3
4
5 loop = asyncio.get_event_loop()
6
7
8 @asyncio.coroutine
9 def hello():
```

(continues on next page)

(continued from previous page)

```
10 print(f"hello {time.strftime('%X')}")
11 yield from asyncio.sleep(1)
12 print(f"world {time.strftime('%X')}")
13
14
15 if __name__ == "__main__":
16     loop.run_until_complete(hello())
```

9.1.4 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.

9.1.5 TODOs

`asyncio.to_thread()` runs the function in an executor, where the default executor is a threadpool executor, which invokes `loop.run_in_executor()` indirectly.

How to set the executor of a loop? Maybe something related to `set_default_executor`?

If we want to schedule a callback to run in the loop from the C++ code, we can use `loop.call_soon_safe()` method.

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

(continues on next page)

(continued from previous page)

```
'-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)
```

9.3 TODO

Python with zeroMQ (c extension)

9.4 time

```
import time  
print(f'Started at {time.strftime("%X")}')  
# do something  
print(f'Finished at {time.strftime("%X")}')
```

9.5 Numbers

9.5.1 binary representation

Listing 3: ./code/numbers/representations.py

```
1 print(bin(1)) # 0b1  
2 print(bin(3)) # 0b11  
3 print(bin(255)) # 0b11111111  
4 print(bin(256)) # 0b100000000  
5 assert isinstance(bin(1), str)  
6 assert int("11", base=2) == 3  
7 assert int("0b11", base=0) == 3  
8 assert hex(2) == "0x2"  
9 assert hex(10) == "0xa"  
10  
11 assert oct(10) == "0o12"  
12 assert int("12", base=8) == 10  
13 assert int("0o12", base=0) == 10  
14  
15 assert 1_000 == 1000  
16 assert 1_000_000 == 1000000
```

9.6 str

9.6.1 format

See <https://docs.python.org/3/library/string.html#formatspec> and <https://peps.python.org/pep-3101/>

Listing 4: ./code/str/format.py

```

1  a = 1
2  b = 2
3  c = 3
4  assert "{}".format(a) == "1"
5  assert "{}".format(b) == "2"
6  assert "{0} {1} {foo}".format(a, b, foo=c) == "1 2 3"
7
8  # 1 - the first positional argument (counting from 0)
9  # foo - it is a keyword argument
10 # 0 - the zeros positional
11 assert "{1} {foo} {0}".format(a, b, foo=c) == "2 3 1"
12
13 assert "{0} {1} {0} {0}".format(a, b) == "1 2 1 1"
14
15 assert "skip braces {0} {}".format(a) == "skip braces 1 {"
16 print("{}") # {}
17 try:
18     print("{} {}".format(a))
19 except IndexError as e:
20     assert str(e) == "Replacement index 1 out of range for positional args tuple"
21
22 assert "{0:2}".format(a) == " 1"
23 assert "{0:02}".format(a) == "01"
24 assert "{0:03}".format(a) == "001"
25 assert "{0:1}".format(-1) == "-1"
26 assert "{0:2}".format(-1) == "-1"
27 assert "{0:3}".format(-1) == "-1"
28 assert "{0:03}".format(-1) == "-01"
29
30 assert "{0:.2f}".format(0.5) == "0.50"
31 assert "{0:.3f}".format(0.5) == "0.500"

```

9.7 enum

9.7.1 Hello

See <https://docs.python.org/3.11/howto/enum.html>

Enum

Note:

- It is iterable, i.e., supports `__iter__`
- name and value
- alias and `@unique`.
- `__members__`.
- `str` and `repr`.
- `auto`

Listing 5: `./code/hello/ex1.py`

```

1 from enum import Enum
2
3
4 class Color(Enum):
5     RED = 1
6     GREEN = 2
7     BLUE = 3
8     # BLUE = 4 # TypeError: Attempted to reuse key: 'BLUE'
9     ALIAS_FOR_RED = 1 # Use @unique to disallow this
10    MAX_COLOR = 4 # Note the naming convention
11
12
13    assert isinstance(Color.RED, Color)
14
15    assert str(Color(1)) == "Color.RED"
16    assert str(repr(Color(1))) == "<Color.RED: 1>"
17
18    assert Color.RED.name == "RED"
19    assert Color.BLUE.value == 3
20
21    print(list(Color))
22    print(type(list(Color)[0]))
23    for c in Color:
24        print(c, type(c))
25
26    """
27    [<Color.RED: 1>, <Color.GREEN: 2>, <Color.BLUE: 3>, <Color.MAX_COLOR: 4>]
28    <enum 'Color'>
29    Color.RED <enum 'Color'>
30    Color.GREEN <enum 'Color'>
31    Color.BLUE <enum 'Color'>
32    Color.MAX_COLOR <enum 'Color'>
33    """
34
35    assert Color(1) == Color.RED
36    assert Color["RED"] == Color.RED
37    assert Color["ALIAS_FOR_RED"] == Color.RED
38

```

(continues on next page)

(continued from previous page)

```

39 print(Color.__members__)
40 """
41 {'RED': <Color.RED: 1>, 'GREEN': <Color.GREEN: 2>, 'BLUE': <Color.BLUE: 3>, 'ALIAS_FOR_RED':
  ↳<Color.RED: 1>, 'MAX_COLOR': <Color.MAX_COLOR: 4>}
42 """

```

Flag

Listing 6: ./code/hello/ex2.py

```

1  from enum import Flag
2
3
4  class Weekday(Flag):
5      MONDAY = 1 << 0
6      TUESDAY = 1 << 1
7      WEDNESDAY = 1 << 2
8      THURSDAY = 1 << 3
9      FRIDAY = 1 << 4
10     SATURDAY = 1 << 5
11     SUNDAY = 1 << 6
12
13
14     assert Weekday.MONDAY.value == 1
15     assert Weekday.TUESDAY.value == 2
16     assert Weekday.WEDNESDAY.value == 4
17     assert Weekday.THURSDAY.value == 8
18     assert Weekday.FRIDAY.value == 16
19     assert Weekday.SATURDAY.value == 32
20     assert Weekday.SUNDAY.value == 64
21
22     weekend = Weekday.SATURDAY | Weekday.SUNDAY
23     print(weekend) # Weekday.SUNDAY|SATURDAY
24     print(repr(weekend)) # <Weekday.SUNDAY|SATURDAY: 96>
25     assert Weekday.SATURDAY in weekend
26     assert Weekday.SUNDAY in weekend
27     assert Weekday.MONDAY not in weekend

```

auto

Listing 7: ./code/hello/ex3.py

```

1  from enum import Enum, Flag, auto
2
3
4  class Weekday(Flag):
5      MONDAY = auto() # start from 1
6      TUESDAY = auto()
7      WEDNESDAY = auto()

```

(continues on next page)

(continued from previous page)

```

8     THURSDAY = auto()
9     FRIDAY = auto()
10    SATURDAY = 128
11    SUNDAY = auto()
12
13
14    assert Weekday.MONDAY.value == 1
15    assert Weekday.TUESDAY.value == 2
16    assert Weekday.WEDNESDAY.value == 4
17    assert Weekday.THURSDAY.value == 8
18    assert Weekday.FRIDAY.value == 16
19    assert Weekday.SATURDAY.value == 128
20    assert Weekday.SUNDAY.value == 256
21
22
23    class Color(Enum):
24        RED = auto() # start from 1
25        GREEN = auto()
26        BLUE = auto()
27        YELLOW = 10
28        WHITE = auto()
29
30
31    assert Color.RED.value == 1
32    assert Color.GREEN.value == 2
33    assert Color.BLUE.value == 3
34    assert Color.YELLOW.value == 10
35    assert Color.WHITE.value == 11

```

9.8 socket

9.8.1 AddressFamily

It is an IntEnum and all of its members are exported to socket.

Listing 8: ./code/address-family.py

```

1  import socket
2
3  print(list(socket.AddressFamily))
4  """
5  [<AddressFamily.AF_UNSPEC: 0>, <AddressFamily.AF_UNIX: 1>,
6  <AddressFamily.AF_INET: 2>, <AddressFamily.AF_AX25: 3>,
7  <AddressFamily.AF_IPX: 4>, <AddressFamily.AF_APPLETALK: 5>,
8  <AddressFamily.AF_NETROM: 6>, <AddressFamily.AF_BRIDGE: 7>,
9  <AddressFamily.AF_ATMPVC: 8>, <AddressFamily.AF_X25: 9>,
10 <AddressFamily.AF_INET6: 10>, <AddressFamily.AF_ROSE: 11>,
11 <AddressFamily.AF_NETBEUI: 13>, <AddressFamily.AF_SECURITY: 14>,
12 <AddressFamily.AF_KEY: 15>, <AddressFamily.AF_NETLINK: 16>,
13 <AddressFamily.AF_PACKET: 17>, <AddressFamily.AF_ASH: 18>,

```

(continues on next page)

(continued from previous page)

```

14 <AddressFamily.AF_ECONET: 19>, <AddressFamily.AF_ATMSVC: 20>,
15 <AddressFamily.AF_RDS: 21>, <AddressFamily.AF_SNA: 22>,
16 <AddressFamily.AF_IRDA: 23>, <AddressFamily.AF_PPPOX: 24>,
17 <AddressFamily.AF_WANPIPE: 25>, <AddressFamily.AF_LLC: 26>,
18 <AddressFamily.AF_CAN: 29>, <AddressFamily.AF_TIPC: 30>,
19 <AddressFamily.AF_BLUETOOTH: 31>, <AddressFamily.AF_ALG: 38>,
20 <AddressFamily.AF_VSOCK: 40>, <AddressFamily.AF_QIPCRTR: 42>]
21 """
22
23 assert socket.AF_UNIX == socket.AddressFamily.AF_UNIX
24 assert socket.AF_INET == socket.AddressFamily.AF_INET

```

9.8.2 SocketKind

It is an IntEnum and all of its members are exported to socket.

Listing 9: ./code/socket-kind.py

```

1 import socket
2
3 print(list(socket.SocketKind))
4 """
5 [<SocketKind.SOCK_STREAM: 1>, <SocketKind.SOCK_DGRAM: 2>,
6 <SocketKind.SOCK_RAW: 3>, <SocketKind.SOCK_RDM: 4>,
7 <SocketKind.SOCK_SEQPACKET: 5>, <SocketKind.SOCK_NONBLOCK: 2048>,
8 <SocketKind.SOCK_CLOEXEC: 524288>]
9 """
10
11 assert socket.SOCK_STREAM == socket.SocketKind.SOCK_STREAM
12 assert socket.SOCK_DGRAM == socket.SocketKind.SOCK_DGRAM

```

9.8.3 struct sockaddr_in

See also

- https://www.gta.ufrj.br/ensino/eel878/sockets/sockaddr_inman.html
- <https://man7.org/linux/man-pages/man7/ip.7.html>

Listing 10: ./code/sockaddr_in.h

```

1 // https://github.com/lattera/glibc/blob/master/bits/sockaddr.h
2 /* POSIX.1g specifies this type name for the 'sa_family' member. */
3 typedef unsigned short int sa_family_t;
4
5 #define __SOCKADDR_COMMON(sa_prefix) sa_family_t sa_prefix##family
6
7 // https://github.com/lattera/glibc/blob/master/bits/socket.h
8
9 struct sockaddr {
10     __SOCKADDR_COMMON(sa_); /* Common data: address family and length. */

```

(continues on next page)

(continued from previous page)

```

11  char sa_data[14];          /* Address data.  */
12  };
13
14  // https://github.com/lattera/glibc/blob/master/inet/netinet/in.h
15  struct sockaddr_in {
16      __SOCKADDR_COMMON(sin_);
17      in_port_t sin_port;      /* Port number.  */
18      struct in_addr sin_addr; /* Internet address.  */
19
20      /* Pad to size of `struct sockaddr'.  */
21      unsigned char sin_zero[sizeof(struct sockaddr) - __SOCKADDR_COMMON_SIZE -
22                              sizeof(in_port_t) - sizeof(struct in_addr)];
23  };
24
25  typedef uint32_t in_addr_t;
26  struct in_addr {
27      in_addr_t s_addr;
28  };
29
30  /* Address to accept any incoming messages.  */
31  #define INADDR_ANY ((in_addr_t)0x00000000)
32  /* Address to send to all hosts.  */
33  #define INADDR_BROADCAST ((in_addr_t)0xffffffff)
34  /* Address indicating an error return.  */
35  #define INADDR_NONE ((in_addr_t)0xffffffff)
36
37  /* Network number for local host loopback.  */
38  #define IN_LOOPBACKNET 127
39  /* Address to loopback in software to local host.  */
40  #ifndef INADDR_LOOPBACK
41  #define INADDR_LOOPBACK ((in_addr_t)0x7f000001) /* Inet 127.0.0.1.  */
42  #endif

```

9.8.4 AddressInfo

Listing 11: ./code/address-info.py

```

1 import socket
2
3 print(list(socket.AddressInfo))
4 """
5 [<AddressInfo.AI_PASSIVE: 1>, <AddressInfo.AI_CANONNAME: 2>,
6 <AddressInfo.AI_NUMERICHOST: 4>, <AddressInfo.AI_V4MAPPED: 8>,
7 <AddressInfo.AI_ALL: 16>, <AddressInfo.AI_ADDRCONFIG: 32>,
8 <AddressInfo.AI_NUMERICSERV: 1024>]
9 """
10 assert socket.AI_PASSIVE == socket.AddressInfo.AI_PASSIVE

```

9.8.5 inet_pton

https://man7.org/linux/man-pages/man3/inet_pton.3.html

Representation format to network address.

The resulting network address is in network order, i.e., big endian.

Listing 12: ./code/inet_pton.c

```

1 #include <arpa/inet.h>
2 #include <stdio.h>
3
4 int main() {
5     struct in_addr addr;
6     int res = inet_pton(AF_INET, "192.168.1.2", &addr);
7     printf("%08x\n", addr.s_addr);
8     printf("192: %x\n", 192);
9     printf("168: %x\n", 168);
10    printf("1: %x\n", 1);
11    printf("2: %x\n", 2);
12    return 0;
13 }
14 #if 0
15 ./inet_pton
16 0201a8c0
17 192: c0
18 168: a8
19 1: 1
20 2: 2
21 #endif

```

Its implementation can be found at https://github.com/bminor/glibc/blob/master/resolv/inet_pton.c

Listing 13: ./code/inet_pton_impl.c

```

1 // See https://github.com/bminor/glibc/blob/master/resolv/inet_pton.c
2 //
3 /* Copyright (C) 1996-2022 Free Software Foundation, Inc.
4    This file is part of the GNU C Library.

```

(continues on next page)

(continued from previous page)

```

5  The GNU C Library is free software; you can redistribute it and/or
6  modify it under the terms of the GNU Lesser General Public
7  License as published by the Free Software Foundation; either
8  version 2.1 of the License, or (at your option) any later version.

```

```

11 The GNU C Library is distributed in the hope that it will be useful,
12 but WITHOUT ANY WARRANTY; without even the implied warranty of
13 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
14 Lesser General Public License for more details.

```

```

16 You should have received a copy of the GNU Lesser General Public
17 License along with the GNU C Library; if not, see
18 <https://www.gnu.org/licenses/>.  */

```

```

19 /*
20  * Copyright (c) 1996,1999 by Internet Software Consortium.
21  *
22  * Permission to use, copy, modify, and distribute this software for any
23  * purpose with or without fee is hereby granted, provided that the above
24  * copyright notice and this permission notice appear in all copies.
25  *
26  * THE SOFTWARE IS PROVIDED "AS IS" AND INTERNET SOFTWARE CONSORTIUM DISCLAIMS
27  * ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES
28  * OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INTERNET SOFTWARE
29  * CONSORTIUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL
30  * DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR
31  * PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS
32  * ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS
33  * SOFTWARE.
34  */

```

```

36 #include <arpa/inet.h>
37 #include <arpa/nameser.h>
38 #include <ctype.h>
39 #include <errno.h>
40 #include <netinet/in.h>
41 #include <resolv/resolv-internal.h>
42 #include <string.h>
43 #include <sys/socket.h>
44 #include <sys/types.h>

```

```

46
47 static int inet_pton4 (const char *src, const char *src_end, u_char *dst);
48 static int inet_pton6 (const char *src, const char *src_end, u_char *dst);

```

```

49
50 int
51 __inet_pton_length (int af, const char *src, size_t srclen, void *dst)
52 {
53     switch (af)
54     {
55         case AF_INET:
56             return inet_pton4 (src, src + srclen, dst);

```

(continues on next page)

(continued from previous page)

```

57     case AF_INET6:
58         return inet_pton6 (src, src + srclen, dst);
59     default:
60         __set_errno (EAFNOSUPPORT);
61         return -1;
62     }
63 }
64 libc_hidden_def (__inet_pton_length)
65
66 /* Like __inet_pton_length, but use strlen (SRC) as the length of
67 SRC. */
68 int
69 __inet_pton (int af, const char *src, void *dst)
70 {
71     return __inet_pton_length (af, src, strlen (src), dst);
72 }
73 libc_hidden_def (__inet_pton)
74 weak_alias (__inet_pton, inet_pton)
75 libc_hidden_weak (inet_pton)
76
77 /* Like inet_aton but without all the hexadecimal, octal and shorthand
78 (and trailing garbage is not ignored). Return 1 if SRC is a valid
79 dotted quad, else 0. This function does not touch DST unless it's
80 returning 1.
81 Author: Paul Vixie, 1996. */
82 static int
83 inet_pton4 (const char *src, const char *end, unsigned char *dst)
84 {
85     int saw_digit, octets, ch;
86     unsigned char tmp[NS_INADDRSZ], *tp;
87
88     saw_digit = 0;
89     octets = 0;
90     *(tp = tmp) = 0;
91     while (src < end)
92     {
93         ch = *src++;
94         if (ch >= '0' && ch <= '9')
95         {
96             unsigned int new = *tp * 10 + (ch - '0');
97
98             if (saw_digit && *tp == 0)
99                 return 0;
100             if (new > 255)
101                 return 0;
102             *tp = new;
103             if (! saw_digit)
104             {
105                 if (++octets > 4)
106                     return 0;
107                 saw_digit = 1;
108             }

```

(continues on next page)

(continued from previous page)

```

109     }
110     else if (ch == '.' && saw_digit)
111     {
112         if (octets == 4)
113             return 0;
114         *++tp = 0;
115         saw_digit = 0;
116     }
117     else
118         return 0;
119 }
120 if (octets < 4)
121     return 0;
122 memcpy (dst, tmp, NS_INADDRSZ);
123 return 1;
124 }
125
126 /* Return the value of CH as a hexademical digit, or -1 if it is a
127    different type of character. */
128 static int
129 hex_digit_value (char ch)
130 {
131     if ('0' <= ch && ch <= '9')
132         return ch - '0';
133     if ('a' <= ch && ch <= 'f')
134         return ch - 'a' + 10;
135     if ('A' <= ch && ch <= 'F')
136         return ch - 'A' + 10;
137     return -1;
138 }
139
140 /* Convert presentation-level IPv6 address to network order binary
141    form. Return 1 if SRC is a valid [RFC1884 2.2] address, else 0.
142    This function does not touch DST unless it's returning 1.
143    Author: Paul Vixie, 1996. Inspired by Mark Andrews. */
144 static int
145 inet_pton6 (const char *src, const char *src_endp, unsigned char *dst)
146 {
147     unsigned char tmp[NS_IN6ADDRSZ], *tp, *endp, *colonp;
148     const char *curtok;
149     int ch;
150     size_t xdigits_seen; /* Number of hex digits since colon. */
151     unsigned int val;
152
153     tp = memset (tmp, '\0', NS_IN6ADDRSZ);
154     endp = tp + NS_IN6ADDRSZ;
155     colonp = NULL;
156
157     /* Leading :: requires some special handling. */
158     if (src == src_endp)
159         return 0;
160     if (*src == ':')

```

(continues on next page)

(continued from previous page)

```

161 {
162     ++src;
163     if (src == src_endp || *src != ':')
164         return 0;
165 }
166
167 curtok = src;
168 xdigits_seen = 0;
169 val = 0;
170 while (src < src_endp)
171 {
172     ch = *src++;
173     int digit = hex_digit_value (ch);
174     if (digit >= 0)
175     {
176         if (xdigits_seen == 4)
177             return 0;
178         val <= 4;
179         val |= digit;
180         if (val > 0xffff)
181             return 0;
182         ++xdigits_seen;
183         continue;
184     }
185     if (ch == ':')
186     {
187         curtok = src;
188         if (xdigits_seen == 0)
189         {
190             if (colonp)
191                 return 0;
192             colonp = tp;
193             continue;
194         }
195         else if (src == src_endp)
196             return 0;
197         if (tp + NS_INT16SZ > endp)
198             return 0;
199         *tp++ = (unsigned char) (val >> 8) & 0xff;
200         *tp++ = (unsigned char) val & 0xff;
201         xdigits_seen = 0;
202         val = 0;
203         continue;
204     }
205     if (ch == '.' && ((tp + NS_INADDRSZ) <= endp)
206         && inet_pton4 (curtok, src_endp, tp) > 0)
207     {
208         tp += NS_INADDRSZ;
209         xdigits_seen = 0;
210         break; /* '\0' was seen by inet_pton4. */
211     }
212     return 0;

```

(continues on next page)

(continued from previous page)

```

213     }
214     if (xdigits_seen > 0)
215     {
216         if (tp + NS_INT16SZ > endp)
217             return 0;
218         *tp++ = (unsigned char) (val >> 8) & 0xff;
219         *tp++ = (unsigned char) val & 0xff;
220     }
221     if (colonp != NULL)
222     {
223         /* Replace :: with zeros. */
224         if (tp == endp)
225             /* :: would expand to a zero-width field. */
226             return 0;
227         size_t n = tp - colonp;
228         memmove (endp - n, colonp, n);
229         memset (colonp, 0, endp - n - colonp);
230         tp = endp;
231     }
232     if (tp != endp)
233         return 0;
234     memcpy (dst, tmp, NS_IN6ADDRSZ);
235     return 1;
236 }

```

9.8.6 inet_ntop

Network address to representation format.

See https://man7.org/linux/man-pages/man3/inet_ntop.3.html

Listing 14: ./code/inet_ntop.c

```

1  #include <arpa/inet.h>
2  #include <stdio.h>
3
4  int main() {
5      struct in_addr addr;
6      uint8_t *p = (uint8_t *)&addr.s_addr;
7      p[0] = 192;
8      p[1] = 168;
9      p[2] = 1;
10     p[3] = 2;
11     char buf[INET_ADDRSTRLEN];
12     const char *ret = inet_ntop(AF_INET, &addr.s_addr, buf, sizeof(buf));
13     printf("%s\n", buf);
14     printf("%p, %p\n", buf, ret);
15     return 0;
16 }
17 #if 0
18 192.168.1.2

```

(continues on next page)

(continued from previous page)

```

19 0x7ffc808b5e80, 0x7ffc808b5e80
20 #endif

```

Its implementation can be found at https://github.com/bminor/glibc/blob/master/resolv/inet_ntop.c

Listing 15: ./code/inet_ntop_impl.c

```

1  // https://github.com/bminor/glibc/blob/master/resolv/inet_ntop.c
2  /*
3   * Copyright (c) 1996-1999 by Internet Software Consortium.
4   *
5   * Permission to use, copy, modify, and distribute this software for any
6   * purpose with or without fee is hereby granted, provided that the above
7   * copyright notice and this permission notice appear in all copies.
8   *
9   * THE SOFTWARE IS PROVIDED "AS IS" AND INTERNET SOFTWARE CONSORTIUM DISCLAIMS
10  * ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES
11  * OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INTERNET SOFTWARE
12  * CONSORTIUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL
13  * DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR
14  * PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS
15  * ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS
16  * SOFTWARE.
17  */
18
19 #include <sys/param.h>
20 #include <sys/types.h>
21 #include <sys/socket.h>
22
23 #include <netinet/in.h>
24 #include <arpa/inet.h>
25 #include <arpa/nameser.h>
26
27 #include <errno.h>
28 #include <stdio.h>
29 #include <string.h>
30
31 #ifdef SPRINTF_CHAR
32 # define SPRINTF(x) strlen(sprintf/**/x)
33 #else
34 # define SPRINTF(x) ((size_t)sprintf x)
35 #endif
36
37 /*
38  * WARNING: Don't even consider trying to compile this on a system where
39  * sizeof(int) < 4.  sizeof(int) > 4 is fine; all the world's not a VAX.
40  */
41
42 static const char *inet_ntop4 (const u_char *src, char *dst, socklen_t size);
43 static const char *inet_ntop6 (const u_char *src, char *dst, socklen_t size);
44
45 /* char *

```

(continues on next page)

(continued from previous page)

```

46  * inet_ntop(af, src, dst, size)
47  *      convert a network format address to presentation format.
48  * return:
49  *      pointer to presentation format address ('dst'), or NULL (see errno).
50  * author:
51  *      Paul Vixie, 1996.
52  */
53  const char *
54  inet_ntop (int af, const void *src, char *dst, socklen_t size)
55  {
56      switch (af) {
57      case AF_INET:
58          return (inet_ntop4(src, dst, size));
59      case AF_INET6:
60          return (inet_ntop6(src, dst, size));
61      default:
62          __set_errno (EAFNOSUPPORT);
63          return (NULL);
64      }
65      /* NOTREACHED */
66  }
67  libc_hidden_def (inet_ntop)
68
69  /* const char *
70  *      inet_ntop4(src, dst, size)
71  *      format an IPv4 address
72  * return:
73  *      'dst' (as a const)
74  * notes:
75  *      (1) uses no statics
76  *      (2) takes a u_char* not an in_addr as input
77  * author:
78  *      Paul Vixie, 1996.
79  */
80  static const char *
81  inet_ntop4 (const u_char *src, char *dst, socklen_t size)
82  {
83      static const char fmt[] = "%u.%u.%u.%u";
84      char tmp[sizeof "255.255.255.255"];
85
86      if (SPRINTF((tmp, fmt, src[0], src[1], src[2], src[3])) >= size) {
87          __set_errno (ENOSPC);
88          return (NULL);
89      }
90      return strcpy(dst, tmp);
91  }
92
93  /* const char *
94  *      inet_ntop6(src, dst, size)
95  *      convert IPv6 binary address into presentation (printable) format
96  * author:
97  *      Paul Vixie, 1996.

```

(continues on next page)

(continued from previous page)

```

98  */
99  static const char *
100 inet_ntop6 (const u_char *src, char *dst, socklen_t size)
101 {
102     /*
103      * Note that int32_t and int16_t need only be "at least" large enough
104      * to contain a value of the specified size. On some systems, like
105      * Crays, there is no such thing as an integer variable with 16 bits.
106      * Keep this in mind if you think this function should have been coded
107      * to use pointer overlays. All the world's not a VAX.
108      */
109     char tmp[sizeof "ffff:ffff:ffff:ffff:ffff:ffff:255.255.255.255"], *tp;
110     struct { int base, len; } best, cur;
111     u_int words[NS_IN6ADDRSZ / NS_INT16SZ];
112     int i;
113
114     /*
115      * Preprocess:
116      *     Copy the input (bytewise) array into a wordwise array.
117      *     Find the longest run of 0x00's in src[] for :: shorthanding.
118      */
119     memset(words, '\\0', sizeof words);
120     for (i = 0; i < NS_IN6ADDRSZ; i += 2)
121         words[i / 2] = (src[i] << 8) | src[i + 1];
122     best.base = -1;
123     cur.base = -1;
124     best.len = 0;
125     cur.len = 0;
126     for (i = 0; i < (NS_IN6ADDRSZ / NS_INT16SZ); i++) {
127         if (words[i] == 0) {
128             if (cur.base == -1)
129                 cur.base = i, cur.len = 1;
130             else
131                 cur.len++;
132         } else {
133             if (cur.base != -1) {
134                 if (best.base == -1 || cur.len > best.len)
135                     best = cur;
136                 cur.base = -1;
137             }
138         }
139     }
140     if (cur.base != -1) {
141         if (best.base == -1 || cur.len > best.len)
142             best = cur;
143     }
144     if (best.base != -1 && best.len < 2)
145         best.base = -1;
146
147     /*
148      * Format the result.
149      */

```

(continues on next page)

(continued from previous page)

```

150     tp = tmp;
151     for (i = 0; i < (NS_IN6ADDRSZ / NS_INT16SZ); i++) {
152         /* Are we inside the best run of 0x00's? */
153         if (best.base != -1 && i >= best.base &&
154             i < (best.base + best.len)) {
155             if (i == best.base)
156                 *tp++ = ':';
157             continue;
158         }
159         /* Are we following an initial run of 0x00s or any real hex? */
160         if (i != 0)
161             *tp++ = ':';
162         /* Is this address an encapsulated IPv4? */
163         if (i == 6 && best.base == 0 &&
164             (best.len == 6 || (best.len == 5 && words[5] == 0xffff))) {
165             if (!inet_ntop4(src+12, tp, sizeof tmp - (tp - tmp)))
166                 return (NULL);
167             tp += strlen(tp);
168             break;
169         }
170         tp += SPRINTF((tp, "%x", words[i]));
171     }
172     /* Was it a trailing run of 0x00's? */
173     if (best.base != -1 && (best.base + best.len) ==
174         (NS_IN6ADDRSZ / NS_INT16SZ))
175         *tp++ = ':';
176     *tp++ = '\0';
177
178     /*
179     * Check for overflow, copy, and we're done.
180     */
181     if ((socklen_t)(tp - tmp) > size) {
182         __set_errno (ENOSPC);
183         return (NULL);
184     }
185     return strcpy(dst, tmp);
186 }

```

9.8.7 Echo Server and Client

Server

Listing 16: ./code/echo-hello/server.py

```

1  #!/usr/bin/env python3
2  import socket
3  import threading
4
5  # nc localhost 6006
6

```

(continues on next page)

(continued from previous page)

```
7
8 def run_server():
9     sock = socket.socket(family=socket.AF_INET, type=socket.SOCK_STREAM)
10    sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
11    sock.bind("", 6006)
12    # sock.setblocking(False) # It returns socket.BlockingIOError
13    sock.listen(2) # backlog is 2
14    while True:
15        client_sock, addr = sock.accept()
16        assert isinstance(client_sock, socket.socket)
17        assert isinstance(addr, tuple)
18        assert isinstance(addr[0], str)
19        assert isinstance(addr[1], int)
20        print("Connected from", addr) # Connected from ('127.0.0.1', 54266)
21        threading.Thread(target=handle_client, args=(client_sock,)).start()
22
23
24 def handle_client(sock: socket.socket):
25     while True:
26         data = sock.recv(1024)
27         if not data:
28             break
29         sock.sendall(data.decode("utf-8").upper().encode())
30     print("Disconnected from", sock.getpeername())
31     sock.close()
32
33
34 if __name__ == "__main__":
35     run_server()
```

To test the server, use `nc localhost 6006` or use the following client.

Client

Listing 17: ./code/echo-hello/client.py

```

1 def main():
2     sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3     sock.connect(("localhost", 6006))
4     for i in range(5):
5         sock.send("hello".encode())
6         b = sock.recv(1024)
7         print(b.decode())
8         time.sleep(0.5)

```

Server2

With `concurrent.futures.ThreadPoolExecutor`.

Listing 18: ./code/echo-hello/server2.py

```

1  #!/usr/bin/env python3
2  import socket
3  import threading
4  from concurrent.futures import ThreadPoolExecutor
5
6  # nc localhost 6006
7  pool = ThreadPoolExecutor(max_workers=3)
8
9
10 def run_server():
11     sock = socket.socket(family=socket.AF_INET, type=socket.SOCK_STREAM)
12     sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
13     sock.bind("", 6006)
14     # sock.setblocking(False) # It returns socket.BlockingIOError
15     sock.listen(2) # backlog is 2
16     while True:
17         client_sock, addr = sock.accept()
18         assert isinstance(client_sock, socket.socket)
19         assert isinstance(addr, tuple)
20         assert isinstance(addr[0], str)
21         assert isinstance(addr[1], int)
22         print("Connected from", addr) # Connected from ('127.0.0.1', 54266)
23         pool.submit(handle_client, client_sock)
24
25
26 def handle_client(sock: socket.socket):
27     while True:
28         data = sock.recv(1024)
29         if not data:
30             break
31         sock.sendall(data.decode("utf-8").upper().encode())
32     print("Disconnected from", sock.getpeername())
33     sock.close()
34
35

```

(continues on next page)

(continued from previous page)

```
36 if __name__ == "__main__":  
37     run_server()
```

9.8.8 TODOs

- Beej's Guide to Network Programming Using Internet Sockets
<https://www.gta.ufrj.br/ensino/eel878/sockets/index.html>
- LWN.net Weekly Edition Archives
<https://lwn.net/Archives/>

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++.

11.1.1 array

Listing 1: ./code/hello_world/array.js

```
1 let a = [ 1, 2, 3 ];  
2 function sum(arr) {  
3     let s = 0;  
4     for (let x of arr) {  
5         s += x;  
6     }  
7     return s;  
8 }  
9 // Sum of the array [1,2,3] is 6  
10 console.log('Sum of the array [' + a + '] is ' + sum(a));  
11  
12 function sum2(arr) {  
13     let s = 0;  
14     for (let i = 0; i != arr.length; ++i) {
```

(continues on next page)

(continued from previous page)

```
15     s += arr[i];
16   }
17   return s;
18 }
19 console.log(sum2(a)); // 6
```

Note that there are two ways to iterate an array:

- `for(let x of array)`
- `for(let i = 0; i != array.length; ++i) { ... }`

To run the above code, use:

```
node array.js
```

11.1.2 class

Listing 2: `./code/hello_world/class.js`

```
1 class Point {
2   constructor(x, y) {
3     this.x = x;
4     this.y = y;
5   }
6
7   distance() { return Math.sqrt(this.x * this.x + this.y * this.y); }
8 }
9
10 let p = new Point(1, 1);
11 console.log(p.distance()); // 1.4142135623730951
```

It defines a `Point` class with two fields `x`, `y`. `Point` has two methods: a constructor and a method `distance()`.

Note that class names are by convention capitalized.

11.2 node

Go to <https://nodejs.org/en/download/> to download pre-built binaries:

```
wget https://nodejs.org/dist/v16.15.1/node-v16.15.1-linux-x64.tar.xz
tar xvf node-v16.15.1-linux-x64.tar.xz
```

and then add `/path/to/node-v16.15.1-linux-x64/bin/` to `PATH`.

11.3 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

14.1 GIL

PROTOCOL BUFFERS

15.1 Installation

15.1.1 C++

See <https://github.com/protocolbuffers/protobuf/blob/main/src/README.md>.

```
make protocol-buffers
cd protocol-buffers
wget https://github.com/protocolbuffers/protobuf/releases/download/v3.20.1/protobuf-all-
↪3.20.1.tar.gz
tar xvf protobuf-all-3.20.1.tar.gz
cd protobuf-all-3.20.1
./configure --prefix=$HOME/software/protobuf-3.20.1
make -j 20
make -j 10 check
make install 2>&1 | tee my-log.txt
cd $HOME/software/protobuf-3.20.1
tree . > tree-log.txt
```

```
$ export PKG_CONFIG_PATH=$HOME/software/protobuf-3.20.1:$PKG_CONFIG_PATH

$ pkg-config --cflags protobuf
-I/root/fangjun/software/protobuf-3.20.1/include

$ pkg-config --libs protobuf
-L/root/fangjun/software/protobuf-3.20.1/lib -lprotobuf

$ pkg-config --cflags --libs protobuf
-I/root/fangjun/software/protobuf-3.20.1/include -L/root/fangjun/software/protobuf-3.20.
↪1/lib -lprotobuf

$ pkg-config --libs-only-L protobuf
-L/root/fangjun/software/protobuf-3.20.1/lib

$ pkg-config --libs-only-l protobuf
-lprotobuf
```

```
$ export PATH=$HOME/software/protobuf-3.20.1/bin:$PATH
$ protoc --version
libprotoc 3.20.1
```

Listing 1: ./code/my-log.txt (Installation logs)

```
1 Making install in .
2 make[1]: Entering directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.
  ↳ 20.1'
3 make[2]: Entering directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.
  ↳ 20.1'
4 make[2]: Nothing to be done for 'install-exec-am'.
5 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/lib/pkgconfig'
6 /usr/bin/install -c -m 644 protobuf.pc protobuf-lite.pc '/root/fangjun/software/
  ↳ protobuf-3.20.1/lib/pkgconfig'
7 make[2]: Leaving directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.
  ↳ 20.1'
8 make[1]: Leaving directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.
  ↳ 20.1'
9 Making install in src
10 make[1]: Entering directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.
  ↳ 20.1/src'
11 make[2]: Entering directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.
  ↳ 20.1/src'
12 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/lib'
13 /bin/bash ../libtool --mode=install /usr/bin/install -c libprotobuf-lite.la
  ↳ libprotobuf.la libprotoc.la '/root/fangjun/software/protobuf-3.20.1/lib'
14 libtool: install: /usr/bin/install -c .libs/libprotobuf-lite.so.31.0.1 /root/fangjun/
  ↳ software/protobuf-3.20.1/lib/libprotobuf-lite.so.31.0.1
15 libtool: install: (cd /root/fangjun/software/protobuf-3.20.1/lib && { ln -s -f
  ↳ libprotobuf-lite.so.31.0.1 libprotobuf-lite.so.31 || { rm -f libprotobuf-lite.so.31 &&
  ↳ ln -s libprotobuf-lite.so.31.0.1 libprotobuf-lite.so.31; }; })
16 libtool: install: (cd /root/fangjun/software/protobuf-3.20.1/lib && { ln -s -f
  ↳ libprotobuf-lite.so.31.0.1 libprotobuf-lite.so || { rm -f libprotobuf-lite.so && ln -s
  ↳ libprotobuf-lite.so.31.0.1 libprotobuf-lite.so; }; })
17 libtool: install: /usr/bin/install -c .libs/libprotobuf-lite.lai /root/fangjun/software/
  ↳ protobuf-3.20.1/lib/libprotobuf-lite.la
18 libtool: install: /usr/bin/install -c .libs/libprotobuf.so.31.0.1 /root/fangjun/software/
  ↳ protobuf-3.20.1/lib/libprotobuf.so.31.0.1
19 libtool: install: (cd /root/fangjun/software/protobuf-3.20.1/lib && { ln -s -f
  ↳ libprotobuf.so.31.0.1 libprotobuf.so.31 || { rm -f libprotobuf.so.31 && ln -s
  ↳ libprotobuf.so.31.0.1 libprotobuf.so.31; }; })
20 libtool: install: (cd /root/fangjun/software/protobuf-3.20.1/lib && { ln -s -f
  ↳ libprotobuf.so.31.0.1 libprotobuf.so || { rm -f libprotobuf.so && ln -s libprotobuf.so.
  ↳ 31.0.1 libprotobuf.so; }; })
21 libtool: install: /usr/bin/install -c .libs/libprotobuf.lai /root/fangjun/software/
  ↳ protobuf-3.20.1/lib/libprotobuf.la
22 libtool: warning: relinking 'libprotoc.la'
23 libtool: install: (cd /root/fangjun/open-source-2/protocol-buffers/protobuf-3.20.1/src; /
  ↳ bin/bash "/root/fangjun/open-source-2/protocol-buffers/protobuf-3.20.1/libtool" --
  ↳ silent --tag CXX --mode=relink g++ -pthread -DHAVE_PTHREAD=1 -DHAVE_ZLIB=1 -Wall -Wno-
  ↳ sign-compare -O2 -g -std=c++11 -DDEBUG -version-info 31:1:0 -export-dynamic -no-
  ↳ undefined -Wl,--version-script=./libprotoc.map -o libprotoc.la -rpath /root/fangjun/
  ↳ software/protobuf-3.20.1/lib google/protobuf/compiler/code_generator.lo google/
  ↳ protobuf/compiler/command_line_interface.lo google/protobuf/compiler/cpp/cpp_enum.lo
  ↳ google/protobuf/compiler/cpp/cpp_enum_field.lo google/protobuf/compiler/cpp/cpp_
  ↳ extension.lo google/protobuf/compiler/cpp/cpp_field.lo google/protobuf/compiler/cpp/
  ↳ cpp_file.lo google/protobuf/compiler/cpp/cpp_generator.lo google/protobuf/compiler/cpp/
  ↳ cpp_helpers.lo google/protobuf/compiler/cpp/cpp_map_field.lo google/protobuf/compiler/
  ↳ cpp/cpp_message.lo google/protobuf/compiler/cpp/cpp_message_field.lo google/protobuf/
```

(continues on next page)

(continued from previous page)

```

24 libtool: install: /usr/bin/install -c .libs/libprotoc.so.31.0.1T /root/fangjun/software/
↳protobuf-3.20.1/lib/libprotoc.so.31.0.1
25 libtool: install: (cd /root/fangjun/software/protobuf-3.20.1/lib && { ln -s -f libprotoc.
↳so.31.0.1 libprotoc.so.31 || { rm -f libprotoc.so.31 && ln -s libprotoc.so.31.0.1
↳libprotoc.so.31; }; })
26 libtool: install: (cd /root/fangjun/software/protobuf-3.20.1/lib && { ln -s -f libprotoc.
↳so.31.0.1 libprotoc.so || { rm -f libprotoc.so && ln -s libprotoc.so.31.0.1 libprotoc.
↳so; }; })
27 libtool: install: /usr/bin/install -c .libs/libprotoc.lai /root/fangjun/software/
↳protobuf-3.20.1/lib/libprotoc.la
28 libtool: install: /usr/bin/install -c .libs/libprotobuf-lite.a /root/fangjun/software/
↳protobuf-3.20.1/lib/libprotobuf-lite.a
29 libtool: install: chmod 644 /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf-lite.a
30 libtool: install: ranlib /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf-lite.a
31 libtool: install: /usr/bin/install -c .libs/libprotobuf.a /root/fangjun/software/
↳protobuf-3.20.1/lib/libprotobuf.a
32 libtool: install: chmod 644 /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf.a
33 libtool: install: ranlib /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf.a
34 libtool: install: /usr/bin/install -c .libs/libprotoc.a /root/fangjun/software/protobuf-
↳3.20.1/lib/libprotoc.a
35 libtool: install: chmod 644 /root/fangjun/software/protobuf-3.20.1/lib/libprotoc.a
36 libtool: install: ranlib /root/fangjun/software/protobuf-3.20.1/lib/libprotoc.a
37 libtool: finish: PATH="/ceph-fj/fangjun/software/py38/bin:/ceph-fj/fangjun/software/jdk-
↳17.0.3/bin:/ceph-fj/fangjun/software/cmake/bin:/ceph-fj/fangjun/software/texlive2021-
↳20210325/bin/x86_64-linux:/ceph-sh1/fangjun/software/cuda-10.2.89/bin:/ceph-fj/fangjun/
↳software/bin:/ceph-sh1/fangjun/software/bin:/ceph-sh1/fangjun/software/nvim-linux64/
↳bin:/ceph-fj/fangjun/software/py38/bin:/ceph-fj/fangjun/software/cmake/bin:/ceph-fj/
↳fangjun/software/texlive2021-20210325/bin/x86_64-linux:/ceph-sh1/fangjun/software/cuda-
↳10.2.89/bin:/ceph-sh1/fangjun/software/nvim-linux64/bin:/usr/local/sbin:/usr/local/
↳bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/sbin" ldconfig -n /root/fangjun/software/
↳protobuf-3.20.1/lib
38 -----
39 Libraries have been installed in:
40   /root/fangjun/software/protobuf-3.20.1/lib
41
42 If you ever happen to want to link against installed libraries
43 in a given directory, LIBDIR, you must either use libtool, and
44 specify the full pathname of the library, or use the '-LLIBDIR'
45 flag during linking and do at least one of the following:
46   - add LIBDIR to the 'LD_LIBRARY_PATH' environment variable
47     during execution
48   - add LIBDIR to the 'LD_RUN_PATH' environment variable
49     during linking
50   - use the '-Wl,-rpath -Wl,LIBDIR' linker flag
51   - have your system administrator add LIBDIR to '/etc/ld.so.conf'
52
53 See any operating system documentation about shared libraries for
54 more information, such as the ld(1) and ld.so(8) manual pages.
55 -----
56 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/bin'
57 /bin/bash ../libtool --mode=install /usr/bin/install -c protoc '/root/fangjun/
↳software/protobuf-3.20.1/bin'

```

(continues on next page)

(continued from previous page)

```

58 libtool: install: /usr/bin/install -c .libs/protoc /root/fangjun/software/protobuf-3.20.
    ↪ 1/bin/protoc
59 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include'
60 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf'
61 /usr/bin/install -c -m 644 google/protobuf/any.proto google/protobuf/api.proto google/
    ↪ protobuf/descriptor.proto google/protobuf/duration.proto google/protobuf/empty.proto
    ↪ google/protobuf/field_mask.proto google/protobuf/source_context.proto google/protobuf/
    ↪ struct.proto google/protobuf/timestamp.proto google/protobuf/type.proto google/
    ↪ protobuf/wrappers.proto '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf
    ↪ '
62 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler'
63 /usr/bin/install -c -m 644 google/protobuf/compiler/plugin.proto '/root/fangjun/
    ↪ software/protobuf-3.20.1/include/google/protobuf/compiler'
64 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include'
65 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf'
66 /usr/bin/install -c -m 644 google/protobuf/any.h google/protobuf/any.pb.h google/
    ↪ protobuf/api.pb.h google/protobuf/arena.h google/protobuf/arena_impl.h google/protobuf/
    ↪ arenastring.h google/protobuf/arenaz_sampler.h google/protobuf/descriptor.h google/
    ↪ protobuf/descriptor.pb.h google/protobuf/descriptor_database.h google/protobuf/
    ↪ duration.pb.h google/protobuf/dynamic_message.h google/protobuf/empty.pb.h google/
    ↪ protobuf/explicitly_constructed.h google/protobuf/extension_set.h google/protobuf/
    ↪ extension_set_inl.h google/protobuf/field_access_listener.h google/protobuf/field_mask.
    ↪ pb.h google/protobuf/generated_enum_reflection.h google/protobuf/generated_enum_util.h
    ↪ google/protobuf/generated_message_bases.h google/protobuf/generated_message_reflection.
    ↪ h google/protobuf/generated_message_tctable_decl.h google/protobuf/generated_message_
    ↪ tctable_impl.h google/protobuf/generated_message_util.h google/protobuf/has_bits.h
    ↪ google/protobuf/implicit_weak_message.h google/protobuf/inlined_string_field.h google/
    ↪ protobuf/map.h google/protobuf/map_entry.h google/protobuf/map_entry_lite.h google/
    ↪ protobuf/map_field.h google/protobuf/map_field_inl.h google/protobuf/map_field_lite.h
    ↪ google/protobuf/map_type_handler.h google/protobuf/message.h google/protobuf/message_
    ↪ lite.h google/protobuf/metadata.h google/protobuf/metadata_lite.h google/protobuf/
    ↪ parse_context.h '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf'
67 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/
    ↪ js'
68 /usr/bin/install -c -m 644 google/protobuf/compiler/js/js_generator.h '/root/fangjun/
    ↪ software/protobuf-3.20.1/include/google/protobuf/compiler/js'
69 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/
    ↪ cpp'
70 /usr/bin/install -c -m 644 google/protobuf/compiler/cpp/cpp_file.h google/protobuf/
    ↪ compiler/cpp/cpp_generator.h google/protobuf/compiler/cpp/cpp_helpers.h google/
    ↪ protobuf/compiler/cpp/cpp_names.h '/root/fangjun/software/protobuf-3.20.1/include/
    ↪ google/protobuf/compiler/cpp'
71 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/
    ↪ ruby'
72 /usr/bin/install -c -m 644 google/protobuf/compiler/ruby/ruby_generator.h '/root/
    ↪ fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/ruby'
73 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/
    ↪ python'
74 /usr/bin/install -c -m 644 google/protobuf/compiler/python/python_generator.h google/
    ↪ protobuf/compiler/python/python_pyi_generator.h '/root/fangjun/software/protobuf-3.20.
    ↪ 1/include/google/protobuf/compiler/python'
75 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/util'

```

(continues on next page)

(continued from previous page)

```

76 /usr/bin/install -c -m 644 google/protobuf/util/delimited_message_util.h google/
↳ protobuf/util/field_comparator.h google/protobuf/util/field_mask_util.h google/
↳ protobuf/util/json_util.h google/protobuf/util/message_differencer.h google/protobuf/
↳ util/time_util.h google/protobuf/util/type_resolver.h google/protobuf/util/type_
77 ↳ resolver_util.h '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/util'
78 /usr/bin/install -c -m 644 google/protobuf/io/coded_stream.h google/protobuf/io/gzip_
↳ stream.h google/protobuf/io/io_win32.h google/protobuf/io/printer.h google/protobuf/io/
↳ strtod.h google/protobuf/io/tokenizer.h google/protobuf/io/zero_copy_stream.h google/
↳ protobuf/io/zero_copy_stream_impl.h google/protobuf/io/zero_copy_stream_impl_lite.h '/'
79 ↳ root/fangjun/software/protobuf-3.20.1/include/google/protobuf/io'
↳ /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/
↳ csharp'
80 /usr/bin/install -c -m 644 google/protobuf/compiler/csharp/csharp_doc_comment.h google/
↳ protobuf/compiler/csharp/csharp_generator.h google/protobuf/compiler/csharp/csharp_
↳ names.h google/protobuf/compiler/csharp/csharp_options.h '/root/fangjun/software/
↳ protobuf-3.20.1/include/google/protobuf/compiler/csharp'
81 ↳ /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/
↳ php'
82 /usr/bin/install -c -m 644 google/protobuf/compiler/php/php_generator.h '/root/fangjun/
↳ software/protobuf-3.20.1/include/google/protobuf/compiler/php'
83 ↳ /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/stubs'
84 /usr/bin/install -c -m 644 google/protobuf/stubs/bytestream.h google/protobuf/stubs/
↳ callback.h google/protobuf/stubs/casts.h google/protobuf/stubs/common.h google/
↳ protobuf/stubs/hash.h google/protobuf/stubs/logging.h google/protobuf/stubs/macros.h
↳ google/protobuf/stubs/map_util.h google/protobuf/stubs/mutex.h google/protobuf/stubs/
↳ once.h google/protobuf/stubs/platform_macros.h google/protobuf/stubs/port.h google/
↳ protobuf/stubs/status.h google/protobuf/stubs/stl_util.h google/protobuf/stubs/
↳ stringpiece.h google/protobuf/stubs/strutil.h google/protobuf/stubs/template_util.h '/'
85 ↳ root/fangjun/software/protobuf-3.20.1/include/google/protobuf/stubs'
↳ /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/
↳ java'
86 /usr/bin/install -c -m 644 google/protobuf/compiler/java/java_generator.h google/
↳ protobuf/compiler/java/java_kotlin_generator.h google/protobuf/compiler/java/java_
↳ names.h '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/java'
87 ↳ /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf'
88 /usr/bin/install -c -m 644 google/protobuf/port.h google/protobuf/port_def.inc google/
↳ protobuf/port_undef.inc google/protobuf/reflection.h google/protobuf/reflection_ops.h
↳ google/protobuf/repeated_field.h google/protobuf/repeated_ptr_field.h google/protobuf/
↳ service.h google/protobuf/source_context.pb.h google/protobuf/struct.pb.h google/
↳ protobuf/text_format.h google/protobuf/timestamp.pb.h google/protobuf/type.pb.h google/
↳ protobuf/unknown_field_set.h google/protobuf/wire_format.h google/protobuf/wire_format_
↳ lite.h google/protobuf/wrappers.pb.h '/root/fangjun/software/protobuf-3.20.1/include/
↳ google/protobuf'
89 ↳ /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/
↳ objectivec'
90 /usr/bin/install -c -m 644 google/protobuf/compiler/objectivec/objectivec_generator.h
↳ google/protobuf/compiler/objectivec/objectivec_helpers.h '/root/fangjun/software/
↳ protobuf-3.20.1/include/google/protobuf/compiler/objectivec'
91 ↳ /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler'
92 /usr/bin/install -c -m 644 google/protobuf/compiler/code_generator.h google/protobuf/
↳ compiler/command_line_interface.h google/protobuf/compiler/importer.h google/protobuf/
↳ compiler/parser.h google/protobuf/compiler/plugin.h google/protobuf/compiler/plugin.pb_
↳ h '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler'

```

(continued from previous page)

```

93 make[2]: Leaving directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.
    ↳20.1/src'
94 make[1]: Leaving directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.
    ↳20.1/src'

```

Listing 2: ./code/tree-log.txt (Installed files)

```

1  .
2  |-- bin
3  |   |-- protoc
4  |-- include
5  |   |-- google
6  |       |-- protobuf
7  |           |-- any.h
8  |           |-- any.pb.h
9  |           |-- any.proto
10 |           |-- api.pb.h
11 |           |-- api.proto
12 |           |-- arena.h
13 |           |-- arena_impl.h
14 |           |-- arenastring.h
15 |           |-- arenaz_sampler.h
16 |           |-- compiler
17 |               |-- code_generator.h
18 |               |-- command_line_interface.h
19 |               |-- cpp
20 |                   |-- cpp_file.h
21 |                   |-- cpp_generator.h
22 |                   |-- cpp_helpers.h
23 |                   |-- cpp_names.h
24 |               |-- csharp
25 |                   |-- csharp_doc_comment.h
26 |                   |-- csharp_generator.h
27 |                   |-- csharp_names.h
28 |                   |-- csharp_options.h
29 |               |-- importer.h
30 |               |-- java
31 |                   |-- java_generator.h
32 |                   |-- java_kotlin_generator.h
33 |                   |-- java_names.h
34 |               |-- js
35 |                   |-- js_generator.h
36 |               |-- objectivec
37 |                   |-- objectivec_generator.h
38 |                   |-- objectivec_helpers.h
39 |               |-- parser.h
40 |               |-- php
41 |                   |-- php_generator.h
42 |               |-- plugin.h
43 |               |-- plugin.pb.h
44 |               |-- plugin.proto
45 |               |-- python

```

(continues on next page)

(continued from previous page)

```

46 |         | | | -- python_generator.h
47 |         | | | -- python_pyi_generator.h
48 |         | | | -- ruby
49 |         | | | -- ruby_generator.h
50 |         | -- descriptor.h
51 |         | -- descriptor.pb.h
52 |         | -- descriptor.proto
53 |         | -- descriptor_database.h
54 |         | -- duration.pb.h
55 |         | -- duration.proto
56 |         | -- dynamic_message.h
57 |         | -- empty.pb.h
58 |         | -- empty.proto
59 |         | -- explicitly_constructed.h
60 |         | -- extension_set.h
61 |         | -- extension_set_inl.h
62 |         | -- field_access_listener.h
63 |         | -- field_mask.pb.h
64 |         | -- field_mask.proto
65 |         | -- generated_enum_reflection.h
66 |         | -- generated_enum_util.h
67 |         | -- generated_message_bases.h
68 |         | -- generated_message_reflection.h
69 |         | -- generated_message_tctable_decl.h
70 |         | -- generated_message_tctable_impl.h
71 |         | -- generated_message_util.h
72 |         | -- has_bits.h
73 |         | -- implicit_weak_message.h
74 |         | -- inlined_string_field.h
75 |         | -- io
76 |         | | -- coded_stream.h
77 |         | | -- gzip_stream.h
78 |         | | -- io_win32.h
79 |         | | -- printer.h
80 |         | | -- strtod.h
81 |         | | -- tokenizer.h
82 |         | | -- zero_copy_stream.h
83 |         | | -- zero_copy_stream_impl.h
84 |         | | -- zero_copy_stream_impl_lite.h
85 |         | -- map.h
86 |         | -- map_entry.h
87 |         | -- map_entry_lite.h
88 |         | -- map_field.h
89 |         | -- map_field_inl.h
90 |         | -- map_field_lite.h
91 |         | -- map_type_handler.h
92 |         | -- message.h
93 |         | -- message_lite.h
94 |         | -- metadata.h
95 |         | -- metadata_lite.h
96 |         | -- parse_context.h
97 |         | -- port.h

```

(continues on next page)

(continued from previous page)

```

98 |         |-- port_def.inc
99 |         |-- port_undef.inc
100 |        |-- reflection.h
101 |        |-- reflection_ops.h
102 |        |-- repeated_field.h
103 |        |-- repeated_ptr_field.h
104 |        |-- service.h
105 |        |-- source_context.pb.h
106 |        |-- source_context.proto
107 |        |-- struct.pb.h
108 |        |-- struct.proto
109 |        |-- stubs
110 |        |   |-- bytestream.h
111 |        |   |-- callback.h
112 |        |   |-- casts.h
113 |        |   |-- common.h
114 |        |   |-- hash.h
115 |        |   |-- logging.h
116 |        |   |-- macros.h
117 |        |   |-- map_util.h
118 |        |   |-- mutex.h
119 |        |   |-- once.h
120 |        |   |-- platform_macros.h
121 |        |   |-- port.h
122 |        |   |-- status.h
123 |        |   |-- stl_util.h
124 |        |   |-- stringpiece.h
125 |        |   |-- strutil.h
126 |        |   `-- template_util.h
127 |        |-- text_format.h
128 |        |-- timestamp.pb.h
129 |        |-- timestamp.proto
130 |        |-- type.pb.h
131 |        |-- type.proto
132 |        |-- unknown_field_set.h
133 |        |-- util
134 |        |   |-- delimited_message_util.h
135 |        |   |-- field_comparator.h
136 |        |   |-- field_mask_util.h
137 |        |   |-- json_util.h
138 |        |   |-- message_differencer.h
139 |        |   |-- time_util.h
140 |        |   |-- type_resolver.h
141 |        |   `-- type_resolver_util.h
142 |        |-- wire_format.h
143 |        |-- wire_format_lite.h
144 |        |-- wrappers.pb.h
145 |        `-- wrappers.proto
146 |-- lib
147 |   |-- libprotobuf-lite.a
148 |   |-- libprotobuf-lite.la
149 |   |-- libprotobuf-lite.so -> libprotobuf-lite.so.31.0.1

```

(continues on next page)

(continued from previous page)

```

150 | | -- libprotobuf-lite.so.31 -> libprotobuf-lite.so.31.0.1
151 | | -- libprotobuf-lite.so.31.0.1
152 | | -- libprotobuf.a
153 | | -- libprotobuf.la
154 | | -- libprotobuf.so -> libprotobuf.so.31.0.1
155 | | -- libprotobuf.so.31 -> libprotobuf.so.31.0.1
156 | | -- libprotobuf.so.31.0.1
157 | | -- libprotoc.a
158 | | -- libprotoc.la
159 | | -- libprotoc.so -> libprotoc.so.31.0.1
160 | | -- libprotoc.so.31 -> libprotoc.so.31.0.1
161 | | -- libprotoc.so.31.0.1
162 | `-- pkgconfig
163 |     |-- protobuf-lite.pc
164 |     `-- protobuf.pc
165 |-- tree-log.txt
166
167 18 directories, 146 files

```

15.2 Hello

15.2.1 hello.proto

See

- <https://developers.google.com/protocol-buffers/docs/cpptutorial>
- <https://developers.google.com/protocol-buffers/docs/pythontutorial>

Listing 3: ./code/hello.proto

```

1  syntax = "proto2";
2
3  package tutorial;
4
5  // available types:
6  // bool, int32, float, double, string
7  message Person {
8      optional string name = 1;
9      optional int32 id = 2;
10     optional string email = 3;
11
12     enum PhoneType {
13         MOBILE = 0;
14         HOME = 1;
15         WORK = 2;
16     }
17
18     message PhoneNumber {
19         optional string number = 1;
20         optional PhoneType type = 2 [ default = HOME ];

```

(continues on next page)

(continued from previous page)

```

21     }
22
23     repeated PhoneNumber phones = 4;
24 }
25
26 message AddressBook { repeated Person people = 1; }

```

15.2.2 makefile

Listing 4: ./code/Makefile

```

1 .PHONY: all clean
2
3 all: hello.pb.cc hello.pb.h hello_pb2.py
4
5 hello.pb.cc hello.pb.h: hello.proto
6     protoc -I=./ --cpp_out=./ ./hello.proto
7
8 hello_pb2.py: hello.proto
9     protoc -I=./ --python_out=./ ./hello.proto
10
11 clean:
12     $(RM) hello.pb.cc hello.pb.h hello_pb2.py

```

15.2.3 hello.pb.h

Listing 5: ./code/hello.pb.h

```

1 // Generated by the protocol buffer compiler.  DO NOT EDIT!
2 // source: hello.proto
3
4 #ifndef GOOGLE_PROTOBUF_INCLUDED_hello_2eproto
5 #define GOOGLE_PROTOBUF_INCLUDED_hello_2eproto
6
7 #include <limits>
8 #include <string>
9
10 #include <google/protobuf/port_def.inc>
11 #if PROTOBUF_VERSION < 3020000
12 #error This file was generated by a newer version of protoc which is
13 #error incompatible with your Protocol Buffer headers. Please update
14 #error your headers.
15 #endif
16 #if 3020001 < PROTOBUF_MIN_PROTOC_VERSION
17 #error This file was generated by an older version of protoc which is
18 #error incompatible with your Protocol Buffer headers. Please
19 #error regenerate this file with a newer version of protoc.
20 #endif
21

```

(continues on next page)

(continued from previous page)

```

22 #include <google/protobuf/port_undef.inc>
23 #include <google/protobuf/io/coded_stream.h>
24 #include <google/protobuf/arena.h>
25 #include <google/protobuf/arenastring.h>
26 #include <google/protobuf/generated_message_util.h>
27 #include <google/protobuf/metadata_lite.h>
28 #include <google/protobuf/generated_message_reflection.h>
29 #include <google/protobuf/message.h>
30 #include <google/protobuf/repeated_field.h> // IWYU pragma: export
31 #include <google/protobuf/extension_set.h> // IWYU pragma: export
32 #include <google/protobuf/generated_enum_reflection.h>
33 #include <google/protobuf/unknown_field_set.h>
34 // @@protoc_insertion_point(includes)
35 #include <google/protobuf/port_def.inc>
36 #define PROTOBUF_INTERNAL_EXPORT_hello_2eproto
37 PROTOBUF_NAMESPACE_OPEN
38 namespace internal {
39 class AnyMetadata;
40 } // namespace internal
41 PROTOBUF_NAMESPACE_CLOSE
42
43 // Internal implementation detail -- do not use these members.
44 struct TableStruct_hello_2eproto {
45     static const uint32_t offsets[];
46 };
47 extern const ::PROTOBUF_NAMESPACE_ID::internal::DescriptorTable descriptor_table_hello_
↳ 2eproto;
48 namespace tutorial {
49 class AddressBook;
50 struct AddressBookDefaultTypeInternal;
51 extern AddressBookDefaultTypeInternal _AddressBook_default_instance_;
52 class Person;
53 struct PersonDefaultTypeInternal;
54 extern PersonDefaultTypeInternal _Person_default_instance_;
55 class Person_PhoneNumber;
56 struct Person_PhoneNumberDefaultTypeInternal;
57 extern Person_PhoneNumberDefaultTypeInternal _Person_PhoneNumber_default_instance_;
58 } // namespace tutorial
59 PROTOBUF_NAMESPACE_OPEN
60 template<> ::tutorial::AddressBook* Arena::CreateMaybeMessage<::tutorial::AddressBook>
↳ (Arena*);
61 template<> ::tutorial::Person* Arena::CreateMaybeMessage<::tutorial::Person>(Arena*);
62 template<> ::tutorial::Person_PhoneNumber* Arena::CreateMaybeMessage<::tutorial::Person_
↳ PhoneNumber>(Arena*);
63 PROTOBUF_NAMESPACE_CLOSE
64 namespace tutorial {
65
66 enum Person_PhoneType : int {
67     Person_PhoneType_MOBILE = 0,
68     Person_PhoneType_HOME = 1,
69     Person_PhoneType_WORK = 2
70 };

```

(continues on next page)

(continued from previous page)

```

71 bool Person_PhoneType_IsValid(int value);
72 constexpr Person_PhoneType Person_PhoneType_PhoneType_MIN = Person_PhoneType_MOBILE;
73 constexpr Person_PhoneType Person_PhoneType_PhoneType_MAX = Person_PhoneType_WORK;
74 constexpr int Person_PhoneType_PhoneType_ARRAYSIZE = Person_PhoneType_PhoneType_MAX + 1;
75
76 const ::PROTOBUF_NAMESPACE_ID::EnumDescriptor* Person_PhoneType_descriptor();
77 template<typename T>
78 inline const std::string& Person_PhoneType_Name(T enum_t_value) {
79     static_assert(::std::is_same<T, Person_PhoneType>::value ||
80         ::std::is_integral<T>::value,
81         "Incorrect type passed to function Person_PhoneType_Name.");
82     return ::PROTOBUF_NAMESPACE_ID::internal::NameOfEnum(
83         Person_PhoneType_descriptor(), enum_t_value);
84 }
85 inline bool Person_PhoneType_Parse(
86     ::PROTOBUF_NAMESPACE_ID::ConstStringParam name, Person_PhoneType* value) {
87     return ::PROTOBUF_NAMESPACE_ID::internal::ParseNamedEnum<Person_PhoneType>(
88         Person_PhoneType_descriptor(), name, value);
89 }
90 // =====
91
92 class Person_PhoneNumber final :
93     public ::PROTOBUF_NAMESPACE_ID::Message /* @@protoc_insertion_point(class_
94     ↪ definition:tutorial.Person.PhoneNumber) */ {
95 public:
96     inline Person_PhoneNumber() : Person_PhoneNumber(nullptr) {}
97     ~Person_PhoneNumber() override;
98     explicit PROTOBUF_CONSTEXPR Person_PhoneNumber(::PROTOBUF_NAMESPACE_ID::
99     ↪ ID::internal::ConstantInitialized);
100
101     Person_PhoneNumber(const Person_PhoneNumber& from);
102     Person_PhoneNumber(Person_PhoneNumber&& from) noexcept
103         : Person_PhoneNumber() {
104         *this = ::std::move(from);
105     }
106
107     inline Person_PhoneNumber& operator=(const Person_PhoneNumber& from) {
108         CopyFrom(from);
109         return *this;
110     }
111
112     inline Person_PhoneNumber& operator=(Person_PhoneNumber&& from) noexcept {
113         if (this == &from) return *this;
114         if (GetOwningArena() == from.GetOwningArena())
115             && GetOwningArena() != nullptr
116         #endif // !PROTOBUF_FORCE_COPY_IN_MOVE
117         ) {
118             InternalSwap(&from);
119         } else {
120             CopyFrom(from);
121         }
122         return *this;

```

(continues on next page)

(continued from previous page)

```

121 }
122
123 inline const ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet& unknown_fields() const {
124     return _internal_metadata_.unknown_fields<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
↪ (::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance);
125 }
126 inline ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet* mutable_unknown_fields() {
127     return _internal_metadata_.mutable_unknown_fields<::PROTOBUF_NAMESPACE_
↪ ID::UnknownFieldSet>();
128 }
129
130 static const ::PROTOBUF_NAMESPACE_ID::Descriptor* descriptor() {
131     return GetDescriptor();
132 }
133 static const ::PROTOBUF_NAMESPACE_ID::Descriptor* GetDescriptor() {
134     return default_instance().GetMetadata().descriptor;
135 }
136 static const ::PROTOBUF_NAMESPACE_ID::Reflection* GetReflection() {
137     return default_instance().GetMetadata().reflection;
138 }
139 static const Person_PhoneNumber& default_instance() {
140     return *internal_default_instance();
141 }
142 static inline const Person_PhoneNumber* internal_default_instance() {
143     return reinterpret_cast<const Person_PhoneNumber*>(
144         &_Person_PhoneNumber_default_instance_);
145 }
146 static constexpr int kIndexInFileMessages =
147     0;
148
149 friend void swap(Person_PhoneNumber& a, Person_PhoneNumber& b) {
150     a.Swap(&b);
151 }
152 inline void Swap(Person_PhoneNumber* other) {
153     if (other == this) return;
154 #ifdef PROTOBUF_FORCE_COPY_IN_SWAP
155     if (GetOwningArena() != nullptr &&
156         GetOwningArena() == other->GetOwningArena()) {
157     #else // PROTOBUF_FORCE_COPY_IN_SWAP
158     if (GetOwningArena() == other->GetOwningArena()) {
159 #endif // !PROTOBUF_FORCE_COPY_IN_SWAP
160         InternalSwap(other);
161     } else {
162         ::PROTOBUF_NAMESPACE_ID::internal::GenericSwap(this, other);
163     }
164 }
165 void UnsafeArenaSwap(Person_PhoneNumber* other) {
166     if (other == this) return;
167     GOOGLE_DCHECK(GetOwningArena() == other->GetOwningArena());
168     InternalSwap(other);
169 }
170

```

(continues on next page)

(continued from previous page)

```

171 // implements Message -----
172
173 Person_PhoneNumber* New(::PROTOBUF_NAMESPACE_ID::Arena* arena = nullptr) const final {
174     return CreateMaybeMessage<Person_PhoneNumber>(arena);
175 }
176 using ::PROTOBUF_NAMESPACE_ID::Message::CopyFrom;
177 void CopyFrom(const Person_PhoneNumber& from);
178 using ::PROTOBUF_NAMESPACE_ID::Message::MergeFrom;
179 void MergeFrom(const Person_PhoneNumber& from);
180 private:
181 static void MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to, const ::PROTOBUF_NAMESPACE_
↪ ID::Message& from);
182 public:
183 PROTOBUF_ATTRIBUTE_REINITIALIZES void Clear() final;
184 bool IsInitialized() const final;
185
186 size_t ByteSizeLong() const final;
187 const char* _InternalParse(const char* ptr, ::PROTOBUF_NAMESPACE_
↪ ID::internal::ParseContext* ctx) final;
188 uint8_t* _InternalSerialize(
189     uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const,
↪ final;
190 int GetCachedSize() const final { return _cached_size_.Get(); }
191
192 private:
193 void SharedCtor();
194 void SharedDtor();
195 void SetCachedSize(int size) const final;
196 void InternalSwap(Person_PhoneNumber* other);
197
198 private:
199 friend class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata;
200 static ::PROTOBUF_NAMESPACE_ID::StringPiece FullMessageName() {
201     return "tutorial.Person.PhoneNumber";
202 }
203 protected:
204 explicit Person_PhoneNumber(::PROTOBUF_NAMESPACE_ID::Arena* arena,
205                             bool is_message_owned = false);
206 public:
207
208 static const ClassData _class_data_;
209 const ::PROTOBUF_NAMESPACE_ID::Message::ClassData* GetClassData() const final;
210
211 ::PROTOBUF_NAMESPACE_ID::Metadata GetMetadata() const final;
212
213 // nested types -----
214
215 // accessors -----
216
217 enum : int {
218     kNumberFieldName = 1,
219     kTypeFieldName = 2,

```

(continues on next page)

(continued from previous page)

```

220 };
221 // optional string number = 1;
222 bool has_number() const;
223 private:
224 bool _internal_has_number() const;
225 public:
226 void clear_number();
227 const std::string& number() const;
228 template <typename ArgT0 = const std::string&, typename... ArgT>
229 void set_number(ArgT0&& arg0, ArgT... args);
230 std::string* mutable_number();
231 PROTOBUF_NODISCARD std::string* release_number();
232 void set_allocated_number(std::string* number);
233 private:
234 const std::string& _internal_number() const;
235 inline PROTOBUF_ALWAYS_INLINE void _internal_set_number(const std::string& value);
236 std::string* _internal_mutable_number();
237 public:
238
239 // optional .tutorial.Person.PhoneType type = 2 [default = HOME];
240 bool has_type() const;
241 private:
242 bool _internal_has_type() const;
243 public:
244 void clear_type();
245 ::tutorial::Person_PhoneType type() const;
246 void set_type(::tutorial::Person_PhoneType value);
247 private:
248 ::tutorial::Person_PhoneType _internal_type() const;
249 void _internal_set_type(::tutorial::Person_PhoneType value);
250 public:
251
252 // @@protoc_insertion_point(class_scope:tutorial.Person.PhoneNumber)
253 private:
254 class _Internal;
255
256 template <typename T> friend class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper;
257 typedef void InternalArenaConstructable_;
258 typedef void DestructorSkippable_;
259 ::PROTOBUF_NAMESPACE_ID::internal::HasBits<1> _has_bits_;
260 mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize _cached_size_;
261 ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr number_;
262 int type_;
263 friend struct ::TableStruct_hello_2eproto;
264 };
265 // -----
266
267 class Person final :
268     public ::PROTOBUF_NAMESPACE_ID::Message /* @@protoc_insertion_point(class_
↳ definition:tutorial.Person) */ {
269 public:
270     inline Person() : Person(nullptr) {}

```

(continues on next page)

(continued from previous page)

```

271 ~Person() override;
272 explicit PROTOBUF_CONSTEXPR Person(::PROTOBUF_NAMESPACE_
↳ ID::internal::ConstantInitialized);
273
274 Person(const Person& from);
275 Person(Person&& from) noexcept
276 : Person() {
277     *this = ::std::move(from);
278 }
279
280 inline Person& operator=(const Person& from) {
281     CopyFrom(from);
282     return *this;
283 }
284 inline Person& operator=(Person&& from) noexcept {
285     if (this == &from) return *this;
286     if (GetOwningArena() == from.GetOwningArena()
287 #ifdef PROTOBUF_FORCE_COPY_IN_MOVE
288         && GetOwningArena() != nullptr
289 #endif // !PROTOBUF_FORCE_COPY_IN_MOVE
290     ) {
291         InternalSwap(&from);
292     } else {
293         CopyFrom(from);
294     }
295     return *this;
296 }
297
298 inline const ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet& unknown_fields() const {
299     return _internal_metadata_.unknown_fields<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
↳ (::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance);
300 }
301 inline ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet* mutable_unknown_fields() {
302     return _internal_metadata_.mutable_unknown_fields<::PROTOBUF_NAMESPACE_
↳ ID::UnknownFieldSet>();
303 }
304
305 static const ::PROTOBUF_NAMESPACE_ID::Descriptor* descriptor() {
306     return GetDescriptor();
307 }
308 static const ::PROTOBUF_NAMESPACE_ID::Descriptor* GetDescriptor() {
309     return default_instance().GetMetadata().descriptor;
310 }
311 static const ::PROTOBUF_NAMESPACE_ID::Reflection* GetReflection() {
312     return default_instance().GetMetadata().reflection;
313 }
314 static const Person& default_instance() {
315     return *internal_default_instance();
316 }
317 static inline const Person* internal_default_instance() {
318     return reinterpret_cast<const Person*>(
319         &_Person_default_instance_);

```

(continues on next page)

(continued from previous page)

```

320 }
321 static constexpr int kIndexInFileMessages =
322     1;
323
324 friend void swap(Person& a, Person& b) {
325     a.Swap(&b);
326 }
327 inline void Swap(Person* other) {
328     if (other == this) return;
329 #ifdef PROTOBUF_FORCE_COPY_IN_SWAP
330     if (GetOwningArena() != nullptr &&
331         GetOwningArena() == other->GetOwningArena()) {
332     #else // PROTOBUF_FORCE_COPY_IN_SWAP
333     if (GetOwningArena() == other->GetOwningArena()) {
334 #endif // !PROTOBUF_FORCE_COPY_IN_SWAP
335         InternalSwap(other);
336     } else {
337         ::PROTOBUF_NAMESPACE_ID::internal::GenericSwap(this, other);
338     }
339 }
340 void UnsafeArenaSwap(Person* other) {
341     if (other == this) return;
342     GOOGLE_DCHECK(GetOwningArena() == other->GetOwningArena());
343     InternalSwap(other);
344 }
345
346 // implements Message -----
347
348 Person* New(::PROTOBUF_NAMESPACE_ID::Arena* arena = nullptr) const final {
349     return CreateMaybeMessage<Person>(arena);
350 }
351 using ::PROTOBUF_NAMESPACE_ID::Message::CopyFrom;
352 void CopyFrom(const Person& from);
353 using ::PROTOBUF_NAMESPACE_ID::Message::MergeFrom;
354 void MergeFrom(const Person& from);
355 private:
356 static void MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to, const ::PROTOBUF_NAMESPACE_ID::Message& from);
357 public:
358 PROTOBUF_ATTRIBUTE_REINITIALIZES void Clear() final;
359 bool IsInitialized() const final;
360
361 size_t ByteSizeLong() const final;
362 const char* _InternalParse(const char* ptr, ::PROTOBUF_NAMESPACE_ID::internal::ParseContext* ctx) final;
363 uint8_t* _InternalSerialize(
364     uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const final;
365 int GetCachedSize() const final { return _cached_size_.Get(); }
366
367 private:
368 void SharedCtor();

```

(continues on next page)

(continued from previous page)

```

369 void SharedDtor();
370 void SetCachedSize(int size) const final;
371 void InternalSwap(Person* other);
372
373 private:
374 friend class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata;
375 static ::PROTOBUF_NAMESPACE_ID::StringPiece FullMessageName() {
376     return "tutorial.Person";
377 }
378 protected:
379 explicit Person(::PROTOBUF_NAMESPACE_ID::Arena* arena,
380                bool is_message_owned = false);
381 public:
382
383 static const ClassData _class_data_;
384 const ::PROTOBUF_NAMESPACE_ID::Message::ClassData* GetClassData() const final;
385
386 ::PROTOBUF_NAMESPACE_ID::Metadata GetMetadata() const final;
387
388 // nested types -----
389
390 typedef Person_PhoneNumber PhoneNumber;
391
392 typedef Person_PhoneType PhoneType;
393 static constexpr PhoneType MOBILE =
394     Person_PhoneType_MOBILE;
395 static constexpr PhoneType HOME =
396     Person_PhoneType_HOME;
397 static constexpr PhoneType WORK =
398     Person_PhoneType_WORK;
399 static inline bool PhoneType_IsValid(int value) {
400     return Person_PhoneType_IsValid(value);
401 }
402 static constexpr PhoneType PhoneType_MIN =
403     Person_PhoneType_PhoneType_MIN;
404 static constexpr PhoneType PhoneType_MAX =
405     Person_PhoneType_PhoneType_MAX;
406 static constexpr int PhoneType_ARRAYSIZE =
407     Person_PhoneType_PhoneType_ARRAYSIZE;
408 static inline const ::PROTOBUF_NAMESPACE_ID::EnumDescriptor*
409 PhoneType_descriptor() {
410     return Person_PhoneType_descriptor();
411 }
412 template<typename T>
413 static inline const std::string& PhoneType_Name(T enum_t_value) {
414     static_assert(::std::is_same<T, PhoneType>::value ||
415                 ::std::is_integral<T>::value,
416                 "Incorrect type passed to function PhoneType_Name.");
417     return Person_PhoneType_Name(enum_t_value);
418 }
419 static inline bool PhoneType_Parse(::PROTOBUF_NAMESPACE_ID::ConstStringParam name,
420     PhoneType* value) {

```

(continues on next page)

(continued from previous page)

```

421     return Person_PhoneType_Parse(name, value);
422 }
423
424 // accessors -----
425
426 enum : int {
427     kPhonesFieldNumber = 4,
428     kNameFieldNumber = 1,
429     kEmailFieldNumber = 3,
430     kIdFieldNumber = 2,
431 };
432 // repeated .tutorial.Person.PhoneNumber phones = 4;
433 int phones_size() const;
434 private:
435 int _internal_phones_size() const;
436 public:
437 void clear_phones();
438 ::tutorial::Person_PhoneNumber* mutable_phones(int index);
439 ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person_PhoneNumber >*
440     mutable_phones();
441 private:
442 const ::tutorial::Person_PhoneNumber& _internal_phones(int index) const;
443 ::tutorial::Person_PhoneNumber* _internal_add_phones();
444 public:
445 const ::tutorial::Person_PhoneNumber& phones(int index) const;
446 ::tutorial::Person_PhoneNumber* add_phones();
447 const ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person_PhoneNumber >&
448     phones() const;
449
450 // optional string name = 1;
451 bool has_name() const;
452 private:
453 bool _internal_has_name() const;
454 public:
455 void clear_name();
456 const std::string& name() const;
457 template <typename ArgT0 = const std::string&, typename... ArgT>
458 void set_name(ArgT0&& arg0, ArgT... args);
459 std::string* mutable_name();
460 PROTOBUF_NODISCARD std::string* release_name();
461 void set_allocated_name(std::string* name);
462 private:
463 const std::string& _internal_name() const;
464 inline PROTOBUF_ALWAYS_INLINE void _internal_set_name(const std::string& value);
465 std::string* _internal_mutable_name();
466 public:
467
468 // optional string email = 3;
469 bool has_email() const;
470 private:
471 bool _internal_has_email() const;
472 public:

```

(continues on next page)

(continued from previous page)

```

473 void clear_email();
474 const std::string& email() const;
475 template <typename ArgT0 = const std::string&, typename... ArgT>
476 void set_email(ArgT0&& arg0, ArgT... args);
477 std::string* mutable_email();
478 PROTOBUF_NODISCARD std::string* release_email();
479 void set_allocated_email(std::string* email);
480 private:
481 const std::string& _internal_email() const;
482 inline PROTOBUF_ALWAYS_INLINE void _internal_set_email(const std::string& value);
483 std::string* _internal_mutable_email();
484 public:
485
486 // optional int32 id = 2;
487 bool has_id() const;
488 private:
489 bool _internal_has_id() const;
490 public:
491 void clear_id();
492 int32_t id() const;
493 void set_id(int32_t value);
494 private:
495 int32_t _internal_id() const;
496 void _internal_set_id(int32_t value);
497 public:
498
499 // @@protoc_insertion_point(class_scope:tutorial.Person)
500 private:
501 class _Internal;
502
503 template <typename T> friend class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper;
504 typedef void InternalArenaConstructable_;
505 typedef void DestructorSkippable_;
506 ::PROTOBUF_NAMESPACE_ID::internal::HasBits<1> _has_bits_;
507 mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize _cached_size_;
508 ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person_PhoneNumber > phones_;
509 ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr name_;
510 ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr email_;
511 int32_t id_;
512 friend struct ::TableStruct_hello_2eproto;
513 };
514 // -----
515
516 class AddressBook final :
517     public ::PROTOBUF_NAMESPACE_ID::Message /* @@protoc_insertion_point(class_
↳ definition: tutorial.AddressBook) */ {
518 public:
519     inline AddressBook() : AddressBook(nullptr) {}
520     ~AddressBook() override;
521     explicit PROTOBUF_CONSTEXPR AddressBook(::PROTOBUF_NAMESPACE_
↳ ID::internal::ConstantInitialized);
522

```

(continues on next page)

(continued from previous page)

```

523 AddressBook(const AddressBook& from);
524 AddressBook(AddressBook&& from) noexcept
525 : AddressBook() {
526     *this = ::std::move(from);
527 }
528
529 inline AddressBook& operator=(const AddressBook& from) {
530     CopyFrom(from);
531     return *this;
532 }
533 inline AddressBook& operator=(AddressBook&& from) noexcept {
534     if (this == &from) return *this;
535     if (GetOwningArena() == from.GetOwningArena()
536 #ifdef PROTOBUF_FORCE_COPY_IN_MOVE
537         && GetOwningArena() != nullptr
538 #endif // !PROTOBUF_FORCE_COPY_IN_MOVE
539     ) {
540         InternalSwap(&from);
541     } else {
542         CopyFrom(from);
543     }
544     return *this;
545 }
546
547 inline const ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet& unknown_fields() const {
548     return _internal_metadata_.unknown_fields<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
549     ↪ (::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance());
550 }
551 inline ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet* mutable_unknown_fields() {
552     return _internal_metadata_.mutable_unknown_fields<::PROTOBUF_NAMESPACE_
553     ↪ ID::UnknownFieldSet>();
554 }
555
556 static const ::PROTOBUF_NAMESPACE_ID::Descriptor* descriptor() {
557     return GetDescriptor();
558 }
559 static const ::PROTOBUF_NAMESPACE_ID::Descriptor* GetDescriptor() {
560     return default_instance().GetMetadata().descriptor;
561 }
562 static const ::PROTOBUF_NAMESPACE_ID::Reflection* GetReflection() {
563     return default_instance().GetMetadata().reflection;
564 }
565 static const AddressBook& default_instance() {
566     return *internal_default_instance();
567 }
568 static inline const AddressBook* internal_default_instance() {
569     return reinterpret_cast<const AddressBook*>(
570         &_AddressBook_default_instance_);
571 }
572 static constexpr int kIndexInFileMessages =
    2;

```

(continues on next page)

(continued from previous page)

```

573 friend void swap(AddressBook& a, AddressBook& b) {
574     a.Swap(&b);
575 }
576 inline void Swap(AddressBook* other) {
577     if (other == this) return;
578 #ifdef PROTOBUF_FORCE_COPY_IN_SWAP
579     if (GetOwningArena() != nullptr &&
580         GetOwningArena() == other->GetOwningArena()) {
581     #else // PROTOBUF_FORCE_COPY_IN_SWAP
582     if (GetOwningArena() == other->GetOwningArena()) {
583 #endif // !PROTOBUF_FORCE_COPY_IN_SWAP
584         InternalSwap(other);
585     } else {
586         ::PROTOBUF_NAMESPACE_ID::internal::GenericSwap(this, other);
587     }
588 }
589 void UnsafeArenaSwap(AddressBook* other) {
590     if (other == this) return;
591     GOOGLE_DCHECK(GetOwningArena() == other->GetOwningArena());
592     InternalSwap(other);
593 }
594
595 // implements Message -----
596
597 AddressBook* New(::PROTOBUF_NAMESPACE_ID::Arena* arena = nullptr) const final {
598     return CreateMaybeMessage<AddressBook>(arena);
599 }
600 using ::PROTOBUF_NAMESPACE_ID::Message::CopyFrom;
601 void CopyFrom(const AddressBook& from);
602 using ::PROTOBUF_NAMESPACE_ID::Message::MergeFrom;
603 void MergeFrom(const AddressBook& from);
604 private:
605 static void MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to, const ::PROTOBUF_NAMESPACE_
606 ID::Message& from);
607 public:
608 PROTOBUF_ATTRIBUTE_REINITIALIZES void Clear() final;
609 bool IsInitialized() const final;
610
611 size_t ByteSizeLong() const final;
612 const char* _InternalParse(const char* ptr, ::PROTOBUF_NAMESPACE_
613 ID::internal::ParseContext* ctx) final;
614 uint8_t* _InternalSerialize(
615     uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const
616 final;
617 int GetCachedSize() const final { return _cached_size_.Get(); }
618
619 private:
620 void SharedCtor();
621 void SharedDtor();
622 void SetCachedSize(int size) const final;
623 void InternalSwap(AddressBook* other);

```

(continues on next page)

(continued from previous page)

```

622 private:
623 friend class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata;
624 static ::PROTOBUF_NAMESPACE_ID::StringPiece FullMessageName() {
625     return "tutorial.AddressBook";
626 }
627 protected:
628 explicit AddressBook(::PROTOBUF_NAMESPACE_ID::Arena* arena,
629                     bool is_message_owned = false);
630 public:
631
632 static const ClassData _class_data_;
633 const ::PROTOBUF_NAMESPACE_ID::Message::ClassData*GetClassData() const final;
634
635 ::PROTOBUF_NAMESPACE_ID::Metadata GetMetadata() const final;
636
637 // nested types -----
638
639 // accessors -----
640
641 enum : int {
642     kPeopleFieldName = 1,
643 };
644 // repeated .tutorial.Person people = 1;
645 int people_size() const;
646 private:
647 int _internal_people_size() const;
648 public:
649 void clear_people();
650 ::tutorial::Person* mutable_people(int index);
651 ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person >*
652     mutable_people();
653 private:
654 const ::tutorial::Person& _internal_people(int index) const;
655 ::tutorial::Person* _internal_add_people();
656 public:
657 const ::tutorial::Person& people(int index) const;
658 ::tutorial::Person* add_people();
659 const ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person >&
660     people() const;
661
662 // @@protoc_insertion_point(class_scope:tutorial.AddressBook)
663 private:
664 class _Internal;
665
666 template <typename T> friend class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper;
667 typedef void InternalArenaConstructable_;
668 typedef void DestructorSkippable_;
669 ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person > people_;
670 mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize _cached_size_;
671 friend struct ::TableStruct_hello_2eproto;
672 };
673 // =====

```

(continues on next page)

(continued from previous page)

```

674
675
676 // =====
677
678 #ifndef __GNUC__
679     #pragma GCC diagnostic push
680     #pragma GCC diagnostic ignored "-Wstrict-aliasing"
681 #endif // __GNUC__
682 // Person_PhoneNumber
683
684 // optional string number = 1;
685 inline bool Person_PhoneNumber::_internal_has_number() const {
686     bool value = (_has_bits_[0] & 0x00000001u) != 0;
687     return value;
688 }
689 inline bool Person_PhoneNumber::has_number() const {
690     return _internal_has_number();
691 }
692 inline void Person_PhoneNumber::clear_number() {
693     number_.ClearToEmpty();
694     _has_bits_[0] &= ~0x00000001u;
695 }
696 inline const std::string& Person_PhoneNumber::number() const {
697     // @@protoc_insertion_point(field_get:tutorial.Person.PhoneNumber.number)
698     return _internal_number();
699 }
700 template <typename ArgT0, typename... ArgT>
701 inline PROTOBUF_ALWAYS_INLINE
702 void Person_PhoneNumber::set_number(ArgT0&& arg0, ArgT... args) {
703     _has_bits_[0] |= 0x00000001u;
704     number_.Set(static_cast<ArgT0 &&>(arg0), args..., GetArenaForAllocation());
705     // @@protoc_insertion_point(field_set:tutorial.Person.PhoneNumber.number)
706 }
707 inline std::string* Person_PhoneNumber::mutable_number() {
708     std::string* _s = _internal_mutable_number();
709     // @@protoc_insertion_point(field_mutable:tutorial.Person.PhoneNumber.number)
710     return _s;
711 }
712 inline const std::string& Person_PhoneNumber::_internal_number() const {
713     return number_.Get();
714 }
715 inline void Person_PhoneNumber::_internal_set_number(const std::string& value) {
716     _has_bits_[0] |= 0x00000001u;
717     number_.Set(value, GetArenaForAllocation());
718 }
719 inline std::string* Person_PhoneNumber::_internal_mutable_number() {
720     _has_bits_[0] |= 0x00000001u;
721     return number_.Mutable(GetArenaForAllocation());
722 }
723 inline std::string* Person_PhoneNumber::release_number() {
724     // @@protoc_insertion_point(field_release:tutorial.Person.PhoneNumber.number)
725     if (!_internal_has_number()) {

```

(continues on next page)

(continued from previous page)

```

726     return nullptr;
727 }
728 _has_bits_[0] &= ~0x00000001u;
729 auto* p = number_.Release();
730 #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
731 if (number_.IsDefault()) {
732     number_.Set("", GetArenaForAllocation());
733 }
734 #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
735 return p;
736 }
737 inline void Person_PhoneNumber::set_allocated_number(std::string* number) {
738     if (number != nullptr) {
739         _has_bits_[0] |= 0x00000001u;
740     } else {
741         _has_bits_[0] &= ~0x00000001u;
742     }
743     number_.SetAllocated(number, GetArenaForAllocation());
744 #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
745     if (number_.IsDefault()) {
746         number_.Set("", GetArenaForAllocation());
747     }
748 #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
749     // @@protoc_insertion_point(field_set_allocated:tutorial.Person.PhoneNumber.number)
750 }
751
752 // optional .tutorial.Person.PhoneType type = 2 [default = HOME];
753 inline bool Person_PhoneNumber::_internal_has_type() const {
754     bool value = (_has_bits_[0] & 0x00000002u) != 0;
755     return value;
756 }
757 inline bool Person_PhoneNumber::has_type() const {
758     return _internal_has_type();
759 }
760 inline void Person_PhoneNumber::clear_type() {
761     type_ = 1;
762     _has_bits_[0] &= ~0x00000002u;
763 }
764 inline ::tutorial::Person_PhoneType Person_PhoneNumber::_internal_type() const {
765     return static_cast< ::tutorial::Person_PhoneType >(type_);
766 }
767 inline ::tutorial::Person_PhoneType Person_PhoneNumber::type() const {
768     // @@protoc_insertion_point(field_get:tutorial.Person.PhoneNumber.type)
769     return _internal_type();
770 }
771 inline void Person_PhoneNumber::_internal_set_type(::tutorial::Person_PhoneType value) {
772     assert(::tutorial::Person_PhoneType_IsValid(value));
773     _has_bits_[0] |= 0x00000002u;
774     type_ = value;
775 }
776 inline void Person_PhoneNumber::set_type(::tutorial::Person_PhoneType value) {
777     _internal_set_type(value);

```

(continues on next page)

(continued from previous page)

```

778 // @@protoc_insertion_point(field_set:tutorial.Person.PhoneNumber.type)
779 }
780
781 // -----
782
783 // Person
784
785 // optional string name = 1;
786 inline bool Person::_internal_has_name() const {
787     bool value = (_has_bits_[0] & 0x000000001u) != 0;
788     return value;
789 }
790 inline bool Person::has_name() const {
791     return _internal_has_name();
792 }
793 inline void Person::clear_name() {
794     name_.ClearToEmpty();
795     _has_bits_[0] &= ~0x000000001u;
796 }
797 inline const std::string& Person::name() const {
798     // @@protoc_insertion_point(field_get:tutorial.Person.name)
799     return _internal_name();
800 }
801 template <typename ArgT0, typename... ArgT>
802 inline PROTOBUF_ALWAYS_INLINE
803 void Person::set_name(ArgT0&& arg0, ArgT... args) {
804     _has_bits_[0] |= 0x000000001u;
805     name_.Set(static_cast<ArgT0 &&>(arg0), args..., GetArenaForAllocation());
806     // @@protoc_insertion_point(field_set:tutorial.Person.name)
807 }
808 inline std::string* Person::mutable_name() {
809     std::string* _s = _internal_mutable_name();
810     // @@protoc_insertion_point(field_mutable:tutorial.Person.name)
811     return _s;
812 }
813 inline const std::string& Person::_internal_name() const {
814     return name_.Get();
815 }
816 inline void Person::_internal_set_name(const std::string& value) {
817     _has_bits_[0] |= 0x000000001u;
818     name_.Set(value, GetArenaForAllocation());
819 }
820 inline std::string* Person::_internal_mutable_name() {
821     _has_bits_[0] |= 0x000000001u;
822     return name_.Mutable(GetArenaForAllocation());
823 }
824 inline std::string* Person::release_name() {
825     // @@protoc_insertion_point(field_release:tutorial.Person.name)
826     if (!_internal_has_name()) {
827         return nullptr;
828     }
829     _has_bits_[0] &= ~0x000000001u;

```

(continues on next page)

(continued from previous page)

```

830     auto* p = name_.Release();
831 #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
832     if (name_.IsDefault()) {
833         name_.Set("", GetArenaForAllocation());
834     }
835 #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
836     return p;
837 }
838 inline void Person::set_allocated_name(std::string* name) {
839     if (name != nullptr) {
840         _has_bits_[0] |= 0x000000001u;
841     } else {
842         _has_bits_[0] &= ~0x000000001u;
843     }
844     name_.SetAllocated(name, GetArenaForAllocation());
845 #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
846     if (name_.IsDefault()) {
847         name_.Set("", GetArenaForAllocation());
848     }
849 #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
850     // @@protoc_insertion_point(field_set_allocated:tutorial.Person.name)
851 }
852
853 // optional int32 id = 2;
854 inline bool Person::_internal_has_id() const {
855     bool value = (_has_bits_[0] & 0x000000004u) != 0;
856     return value;
857 }
858 inline bool Person::has_id() const {
859     return _internal_has_id();
860 }
861 inline void Person::clear_id() {
862     id_ = 0;
863     _has_bits_[0] &= ~0x000000004u;
864 }
865 inline int32_t Person::_internal_id() const {
866     return id_;
867 }
868 inline int32_t Person::id() const {
869     // @@protoc_insertion_point(field_get:tutorial.Person.id)
870     return _internal_id();
871 }
872 inline void Person::_internal_set_id(int32_t value) {
873     _has_bits_[0] |= 0x000000004u;
874     id_ = value;
875 }
876 inline void Person::set_id(int32_t value) {
877     _internal_set_id(value);
878     // @@protoc_insertion_point(field_set:tutorial.Person.id)
879 }
880
881 // optional string email = 3;

```

(continues on next page)

(continued from previous page)

```

882 inline bool Person::_internal_has_email() const {
883     bool value = (_has_bits_[0] & 0x00000002u) != 0;
884     return value;
885 }
886 inline bool Person::has_email() const {
887     return _internal_has_email();
888 }
889 inline void Person::clear_email() {
890     email_.ClearToEmpty();
891     _has_bits_[0] &= ~0x00000002u;
892 }
893 inline const std::string& Person::email() const {
894     // @@protoc_insertion_point(field_get:tutorial.Person.email)
895     return _internal_email();
896 }
897 template <typename ArgT0, typename... ArgT>
898 inline PROTOBUF_ALWAYS_INLINE
899 void Person::set_email(ArgT0&& arg0, ArgT... args) {
900     _has_bits_[0] |= 0x00000002u;
901     email_.Set(static_cast<ArgT0 &&>(arg0), args..., GetArenaForAllocation());
902     // @@protoc_insertion_point(field_set:tutorial.Person.email)
903 }
904 inline std::string* Person::mutable_email() {
905     std::string* _s = _internal_mutable_email();
906     // @@protoc_insertion_point(field_mutable:tutorial.Person.email)
907     return _s;
908 }
909 inline const std::string& Person::_internal_email() const {
910     return email_.Get();
911 }
912 inline void Person::_internal_set_email(const std::string& value) {
913     _has_bits_[0] |= 0x00000002u;
914     email_.Set(value, GetArenaForAllocation());
915 }
916 inline std::string* Person::_internal_mutable_email() {
917     _has_bits_[0] |= 0x00000002u;
918     return email_.Mutable(GetArenaForAllocation());
919 }
920 inline std::string* Person::release_email() {
921     // @@protoc_insertion_point(field_release:tutorial.Person.email)
922     if (!_internal_has_email()) {
923         return nullptr;
924     }
925     _has_bits_[0] &= ~0x00000002u;
926     auto* p = email_.Release();
927 #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
928     if (email_.IsDefault()) {
929         email_.Set("", GetArenaForAllocation());
930     }
931 #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
932     return p;
933 }

```

(continues on next page)

(continued from previous page)

```

934 inline void Person::set_allocated_email(std::string* email) {
935     if (email != nullptr) {
936         _has_bits_[0] |= 0x000000002u;
937     } else {
938         _has_bits_[0] &= ~0x000000002u;
939     }
940     email_.SetAllocated(email, GetArenaForAllocation());
941 #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
942     if (email_.IsDefault()) {
943         email_.Set("", GetArenaForAllocation());
944     }
945 #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
946     // @@protoc_insertion_point(field_set_allocated:tutorial.Person.email)
947 }
948
949 // repeated .tutorial.Person.PhoneNumber phones = 4;
950 inline int Person::_internal_phones_size() const {
951     return phones_.size();
952 }
953 inline int Person::phones_size() const {
954     return _internal_phones_size();
955 }
956 inline void Person::clear_phones() {
957     phones_.Clear();
958 }
959 inline ::tutorial::Person_PhoneNumber* Person::mutable_phones(int index) {
960     // @@protoc_insertion_point(field_mutable:tutorial.Person.phones)
961     return phones_.Mutable(index);
962 }
963 inline ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person_PhoneNumber >*
964 Person::mutable_phones() {
965     // @@protoc_insertion_point(field_mutable_list:tutorial.Person.phones)
966     return &phones_;
967 }
968 inline const ::tutorial::Person_PhoneNumber& Person::_internal_phones(int index) const {
969     return phones_.Get(index);
970 }
971 inline const ::tutorial::Person_PhoneNumber& Person::phones(int index) const {
972     // @@protoc_insertion_point(field_get:tutorial.Person.phones)
973     return _internal_phones(index);
974 }
975 inline ::tutorial::Person_PhoneNumber* Person::_internal_add_phones() {
976     return phones_.Add();
977 }
978 inline ::tutorial::Person_PhoneNumber* Person::add_phones() {
979     ::tutorial::Person_PhoneNumber* _add = _internal_add_phones();
980     // @@protoc_insertion_point(field_add:tutorial.Person.phones)
981     return _add;
982 }
983 inline const ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person_PhoneNumber >&
984 Person::phones() const {
985     // @@protoc_insertion_point(field_list:tutorial.Person.phones)

```

(continues on next page)

(continued from previous page)

```

986     return phones_;
987 }
988
989 // -----
990
991 // AddressBook
992
993 // repeated .tutorial.Person people = 1;
994 inline int AddressBook::_internal_people_size() const {
995     return people_.size();
996 }
997 inline int AddressBook::people_size() const {
998     return _internal_people_size();
999 }
1000 inline void AddressBook::clear_people() {
1001     people_.Clear();
1002 }
1003 inline ::tutorial::Person* AddressBook::mutable_people(int index) {
1004     // @@protoc_insertion_point(field_mutable:tutorial.AddressBook.people)
1005     return people_.Mutable(index);
1006 }
1007 inline ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person >*
1008 AddressBook::mutable_people() {
1009     // @@protoc_insertion_point(field_mutable_list:tutorial.AddressBook.people)
1010     return &people_;
1011 }
1012 inline const ::tutorial::Person& AddressBook::_internal_people(int index) const {
1013     return people_.Get(index);
1014 }
1015 inline const ::tutorial::Person& AddressBook::people(int index) const {
1016     // @@protoc_insertion_point(field_get:tutorial.AddressBook.people)
1017     return _internal_people(index);
1018 }
1019 inline ::tutorial::Person* AddressBook::_internal_add_people() {
1020     return people_.Add();
1021 }
1022 inline ::tutorial::Person* AddressBook::add_people() {
1023     ::tutorial::Person* _add = _internal_add_people();
1024     // @@protoc_insertion_point(field_add:tutorial.AddressBook.people)
1025     return _add;
1026 }
1027 inline const ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person >&
1028 AddressBook::people() const {
1029     // @@protoc_insertion_point(field_list:tutorial.AddressBook.people)
1030     return people_;
1031 }
1032
1033 #ifdef __GNUC__
1034     #pragma GCC diagnostic pop
1035 #endif // __GNUC__
1036 // -----
1037

```

(continues on next page)

(continued from previous page)

```

1038 // -----
1039
1040 // @@protoc_insertion_point(namespace_scope)
1041
1042 } // namespace tutorial
1043
1044 PROTOBUF_NAMESPACE_OPEN
1045
1046 template <> struct is_proto_enum< ::tutorial::Person_PhoneType> : ::std::true_type {};
1047 template <>
1048 inline const EnumDescriptor* GetEnumDescriptor< ::tutorial::Person_PhoneType>() {
1049     return ::tutorial::Person_PhoneType_descriptor();
1050 }
1051
1052 PROTOBUF_NAMESPACE_CLOSE
1053
1054 // @@protoc_insertion_point(global_scope)
1055
1056 #include <google/protobuf/port_undef.inc>
1057 #endif // GOOGLE_PROTOBUF_INCLUDED_GOOGLE_PROTOBUF_INCLUDED_hello_2eproto

```

15.2.4 hello.pb.cc

Listing 6: ./code/hello.pb.cc

```

1 // Generated by the protocol buffer compiler.  DO NOT EDIT!
2 // source: hello.proto
3
4 #include "hello.pb.h"
5
6 #include <algorithm>
7
8 #include <google/protobuf/io/coded_stream.h>
9 #include <google/protobuf/extension_set.h>
10 #include <google/protobuf/wire_format_lite.h>
11 #include <google/protobuf/descriptor.h>
12 #include <google/protobuf/generated_message_reflection.h>
13 #include <google/protobuf/reflection_ops.h>
14 #include <google/protobuf/wire_format.h>
15 // @@protoc_insertion_point(includes)
16 #include <google/protobuf/port_def.inc>
17
18 PROTOBUF_PRAGMA_INIT_SEG
19
20 namespace _pb = ::PROTOBUF_NAMESPACE_ID;
21 namespace _pbi = _pb::internal;
22
23 namespace tutorial {
24 PROTOBUF_CONSTEXPR Person_PhoneNumber::Person_PhoneNumber(

```

(continues on next page)

(continued from previous page)

```

25     ::_pbi::ConstantInitialized)
26     : number_(&::_pbi::fixed_address_empty_string, ::_pbi::ConstantInitialized{})
27     , type_(1)
28 {}
29 struct Person_PhoneNumberDefaultTypeInternal {
30     PROTOBUF_CONSTEXPR Person_PhoneNumberDefaultTypeInternal()
31         : _instance(::_pbi::ConstantInitialized{}) {}
32     ~Person_PhoneNumberDefaultTypeInternal() {}
33     union {
34         Person_PhoneNumber _instance;
35     };
36 };
37 PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT PROTOBUF_ATTRIBUTE_INIT_PRIORITY1
↪ Person_PhoneNumberDefaultTypeInternal _Person_PhoneNumber_default_instance_;
38 PROTOBUF_CONSTEXPR Person::Person(
39     ::_pbi::ConstantInitialized)
40     : phones_()
41     , name_(&::_pbi::fixed_address_empty_string, ::_pbi::ConstantInitialized{})
42     , email_(&::_pbi::fixed_address_empty_string, ::_pbi::ConstantInitialized{})
43     , id_(0){}
44 struct PersonDefaultTypeInternal {
45     PROTOBUF_CONSTEXPR PersonDefaultTypeInternal()
46         : _instance(::_pbi::ConstantInitialized{}) {}
47     ~PersonDefaultTypeInternal() {}
48     union {
49         Person _instance;
50     };
51 };
52 PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT PROTOBUF_ATTRIBUTE_INIT_PRIORITY1
↪ PersonDefaultTypeInternal _Person_default_instance_;
53 PROTOBUF_CONSTEXPR AddressBook::AddressBook(
54     ::_pbi::ConstantInitialized)
55     : people_(){}
56 struct AddressBookDefaultTypeInternal {
57     PROTOBUF_CONSTEXPR AddressBookDefaultTypeInternal()
58         : _instance(::_pbi::ConstantInitialized{}) {}
59     ~AddressBookDefaultTypeInternal() {}
60     union {
61         AddressBook _instance;
62     };
63 };
64 PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT PROTOBUF_ATTRIBUTE_INIT_PRIORITY1
↪ AddressBookDefaultTypeInternal _AddressBook_default_instance_;
65 } // namespace tutorial
66 static ::_pb::Metadata file_level_metadata_hello_2eproto[3];
67 static const ::_pb::EnumDescriptor* file_level_enum_descriptors_hello_2eproto[1];
68 static constexpr ::_pb::ServiceDescriptor const** file_level_service_descriptors_hello_
↪ 2eproto = nullptr;
69
70 const uint32_t TableStruct_hello_2eproto::offsets[] PROTOBUF_SECTION_VARIABLE(protodesc_
↪ cold) = {
71     PROTOBUF_FIELD_OFFSET(::tutorial::Person_PhoneNumber, _has_bits_),

```

(continues on next page)

(continued from previous page)

```

72 PROTOBUF_FIELD_OFFSET(::tutorial::Person_PhoneNumber, _internal_metadata_),
73 ~0u, // no _extensions_
74 ~0u, // no _oneof_case_
75 ~0u, // no _weak_field_map_
76 ~0u, // no _inlined_string_donated_
77 PROTOBUF_FIELD_OFFSET(::tutorial::Person_PhoneNumber, number_),
78 PROTOBUF_FIELD_OFFSET(::tutorial::Person_PhoneNumber, type_),
79 0,
80 1,
81 PROTOBUF_FIELD_OFFSET(::tutorial::Person, _has_bits_),
82 PROTOBUF_FIELD_OFFSET(::tutorial::Person, _internal_metadata_),
83 ~0u, // no _extensions_
84 ~0u, // no _oneof_case_
85 ~0u, // no _weak_field_map_
86 ~0u, // no _inlined_string_donated_
87 PROTOBUF_FIELD_OFFSET(::tutorial::Person, name_),
88 PROTOBUF_FIELD_OFFSET(::tutorial::Person, id_),
89 PROTOBUF_FIELD_OFFSET(::tutorial::Person, email_),
90 PROTOBUF_FIELD_OFFSET(::tutorial::Person, phones_),
91 0,
92 2,
93 1,
94 ~0u,
95 ~0u, // no _has_bits_
96 PROTOBUF_FIELD_OFFSET(::tutorial::AddressBook, _internal_metadata_),
97 ~0u, // no _extensions_
98 ~0u, // no _oneof_case_
99 ~0u, // no _weak_field_map_
100 ~0u, // no _inlined_string_donated_
101 PROTOBUF_FIELD_OFFSET(::tutorial::AddressBook, people_),
102 };
103 static const ::_pbi::MigrationSchema schemas[] PROTOBUF_SECTION_VARIABLE(protodesc_cold) =
104 ↪ {
105     { 0, 8, -1, sizeof(::tutorial::Person_PhoneNumber)},
106     { 10, 20, -1, sizeof(::tutorial::Person)},
107     { 24, -1, -1, sizeof(::tutorial::AddressBook)},
108 };
109 static const ::_pb::Message* const file_default_instances[] = {
110     &::tutorial::Person_PhoneNumber_default_instance._instance,
111     &::tutorial::Person_default_instance._instance,
112     &::tutorial::AddressBook_default_instance._instance,
113 };
114
115 const char descriptor_table_protodef_hello_2eproto[] PROTOBUF_SECTION_VARIABLE(protodesc_
116 ↪cold) =
117     "\n\013hello.proto\022\010tutorial\"333\001\n\006Person\022\014\n\004na"
118     "me\030\001 \001(\t\022\n\n\002id\030\002 \001(\005\022r\n\005email\030\003 \001(\t\022\022,\n\006p"
119     "hones\030\004 \003(\0132\034.tutorial.Person.PhoneNumbe"
120     "r\032M\n\013PhoneNumber\022\016\n\006number\030\001 \001(\t\022.\n\004type"
    "\030\002 \001(\0162\032.tutorial.Person.PhoneType:\004HOME"

```

(continues on next page)

(continued from previous page)

```

121     "\"+\\n\\tPhoneType\\022\\n\\n\\006MOBILE\\020\\000\\022\\010\\n\\004HOME\\020\\001\\022\\010\\n\\004W"
122     "ORK\\020\\002\\"/\\n\\013AddressBook\\022 \\n\\006people\\030\\001 \\003\\0132\\020.t"
123     "utorial.Person"
124     ;
125     static ::_pbi::once_flag descriptor_table_hello_2eproto_once;
126     const ::_pbi::DescriptorTable descriptor_table_hello_2eproto = {
127         false, false, 294, descriptor_table_protodef_hello_2eproto,
128         "hello.proto",
129         &descriptor_table_hello_2eproto_once, nullptr, 0, 3,
130         schemas, file_default_instances, TableStruct_hello_2eproto::offsets,
131         file_level_metadata_hello_2eproto, file_level_enum_descriptors_hello_2eproto,
132         file_level_service_descriptors_hello_2eproto,
133     };
134     PROTOBUF_ATTRIBUTE_WEAK const ::_pbi::DescriptorTable* descriptor_table_hello_2eproto_
135     ↪getter() {
136         return &descriptor_table_hello_2eproto;
137     }
138     // Force running AddDescriptors() at dynamic initialization time.
139     PROTOBUF_ATTRIBUTE_INIT_PRIORITY2 static ::_pbi::AddDescriptorsRunner dynamic_init_dummy_
140     ↪hello_2eproto(&descriptor_table_hello_2eproto);
141     namespace tutorial {
142     const ::PROTOBUF_NAMESPACE_ID::EnumDescriptor* Person_PhoneType_descriptor() {
143         ::PROTOBUF_NAMESPACE_ID::internal::AssignDescriptors(&descriptor_table_hello_2eproto);
144         return file_level_enum_descriptors_hello_2eproto[0];
145     }
146     bool Person_PhoneType_IsValid(int value) {
147         switch (value) {
148             case 0:
149             case 1:
150             case 2:
151                 return true;
152             default:
153                 return false;
154         }
155     }
156     #if (__cplusplus < 201703) && (!defined(_MSC_VER) || (_MSC_VER >= 1900 && _MSC_VER <
157     ↪1912))
158     constexpr Person_PhoneType Person::MOBILE;
159     constexpr Person_PhoneType Person::HOME;
160     constexpr Person_PhoneType Person::WORK;
161     constexpr Person_PhoneType Person::PhoneType_MIN;
162     constexpr Person_PhoneType Person::PhoneType_MAX;
163     constexpr int Person::PhoneType_ARRAYSIZE;
164     #endif // (__cplusplus < 201703) && (!defined(_MSC_VER) || (_MSC_VER >= 1900 && _MSC_
165     ↪VER < 1912))
166     // =====
167     class Person_PhoneNumber::_Internal {
168     public:

```

(continues on next page)

(continued from previous page)

```

169 using HasBits = decltype(std::declval<Person_PhoneNumber>()._has_bits_);
170 static void set_has_number(HasBits* has_bits) {
171     (*has_bits)[0] |= 1u;
172 }
173 static void set_has_type(HasBits* has_bits) {
174     (*has_bits)[0] |= 2u;
175 }
176 };
177
178 Person_PhoneNumber::Person_PhoneNumber(::PROTOBUF_NAMESPACE_ID::Arena* arena,
179                                         bool is_message_owned)
180     : ::PROTOBUF_NAMESPACE_ID::Message(arena, is_message_owned) {
181     SharedCtor();
182     // @@protoc_insertion_point(arena_constructor:tutorial.Person.PhoneNumber)
183 }
184 Person_PhoneNumber::Person_PhoneNumber(const Person_PhoneNumber& from)
185     : ::PROTOBUF_NAMESPACE_ID::Message(),
186       _has_bits_(from._has_bits_) {
187     _internal_metadata_.MergeFrom<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
188 ↪ metadata_);
189     number_.InitDefault();
190     #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
191     number_.Set("", GetArenaForAllocation());
192     #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
193     if (from._internal_has_number()) {
194         number_.Set(from._internal_number(),
195                     GetArenaForAllocation());
196     }
197     type_ = from.type_;
198     // @@protoc_insertion_point(copy_constructor:tutorial.Person.PhoneNumber)
199 }
200 inline void Person_PhoneNumber::SharedCtor() {
201     number_.InitDefault();
202     #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
203     number_.Set("", GetArenaForAllocation());
204     #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
205     type_ = 1;
206 }
207
208 Person_PhoneNumber::~Person_PhoneNumber() {
209     // @@protoc_insertion_point(destructor:tutorial.Person.PhoneNumber)
210     if (auto *arena = _internal_metadata_.DeleteReturnArena<::PROTOBUF_NAMESPACE_
211 ↪ ID::UnknownFieldSet>()) {
212         (void)arena;
213         return;
214     }
215     SharedDtor();
216 }
217 inline void Person_PhoneNumber::SharedDtor() {
218     GOOGLE_DCHECK(GetArenaForAllocation() == nullptr);

```

(continues on next page)

(continued from previous page)

```

219     number_.Destroy();
220 }
221
222 void Person_PhoneNumber::SetCachedSize(int size) const {
223     _cached_size_.Set(size);
224 }
225
226 void Person_PhoneNumber::Clear() {
227     // @@protoc_insertion_point(message_clear_start:tutorial.Person.PhoneNumber)
228     uint32_t cached_has_bits = 0;
229     // Prevent compiler warnings about cached_has_bits being unused
230     (void) cached_has_bits;
231
232     cached_has_bits = _has_bits_[0];
233     if (cached_has_bits & 0x000000003u) {
234         if (cached_has_bits & 0x000000001u) {
235             number_.ClearNonDefaultToEmpty();
236         }
237         type_ = 1;
238     }
239     _has_bits_.Clear();
240     _internal_metadata_.Clear<:PROTOBUF_NAMESPACE_ID::UnknownFieldSet>();
241 }
242
243 const char* Person_PhoneNumber::_InternalParse(const char* ptr, ::_pbi::ParseContext*
244     ↪ ctx) {
245     #define CHK_(x) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
246     _Internal::HasBits has_bits{};
247     while (!ctx->Done(&ptr)) {
248         uint32_t tag;
249         ptr = ::_pbi::ReadTag(ptr, &tag);
250         switch (tag >> 3) {
251             // optional string number = 1;
252             case 1:
253                 if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 10)) {
254                     auto str = _internal_mutable_number();
255                     ptr = ::_pbi::InlineGreedyStringParser(str, ptr, ctx);
256                     CHK_(ptr);
257                     #ifndef NDEBUG
258                     ::_pbi::VerifyUTF8(str, "tutorial.Person.PhoneNumber.number");
259                     #endif // !NDEBUG
260                 } else
261                     goto handle_unusual;
262                 continue;
263             // optional .tutorial.Person.PhoneType type = 2 [default = HOME];
264             case 2:
265                 if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 16)) {
266                     uint64_t val = ::PROTOBUF_NAMESPACE_ID::internal::ReadVarint64(&ptr);
267                     CHK_(ptr);
268                     if (PROTOBUF_PREDICT_TRUE(::tutorial::Person_PhoneType_IsValid(val))) {
269                         _internal_set_type(static_cast<:tutorial::Person_PhoneType>(val));
270                     } else {

```

(continues on next page)

(continued from previous page)

```

270         ::PROTOBUF_NAMESPACE_ID::internal::WriteVarint(2, val, mutable_unknown_
↪fields());
271     }
272     } else
273         goto handle_unusual;
274     continue;
275     default:
276         goto handle_unusual;
277 } // switch
278 handle_unusual:
279     if ((tag == 0) || ((tag & 7) == 4)) {
280         CHK_(ptr);
281         ctx->SetLastTag(tag);
282         goto message_done;
283     }
284     ptr = UnknownFieldParse(
285         tag,
286         _internal_metadata_.mutable_unknown_fields<::PROTOBUF_NAMESPACE_
↪ID::UnknownFieldSet>(),
287         ptr, ctx);
288     CHK_(ptr != nullptr);
289 } // while
290 message_done:
291     _has_bits_.Or(has_bits);
292     return ptr;
293 failure:
294     ptr = nullptr;
295     goto message_done;
296 #undef CHK_
297 }
298
299 uint8_t* Person_PhoneNumber::_InternalSerialize(
300     uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const {
301     // @@protoc_insertion_point(serialize_to_array_start:tutorial.Person.PhoneNumber)
302     uint32_t cached_has_bits = 0;
303     (void) cached_has_bits;
304
305     cached_has_bits = _has_bits_[0];
306     // optional string number = 1;
307     if (cached_has_bits & 0x00000001u) {
308         ::PROTOBUF_NAMESPACE_ID::internal::WireFormat::VerifyUTF8StringNamedField(
309             this->_internal_number().data(), static_cast<int>(this->_internal_number().
↪length()),
310             ::PROTOBUF_NAMESPACE_ID::internal::WireFormat::SERIALIZE,
311             "tutorial.Person.PhoneNumber.number");
312         target = stream->WriteStringMaybeAliased(
313             1, this->_internal_number(), target);
314     }
315
316     // optional .tutorial.Person.PhoneType type = 2 [default = HOME];
317     if (cached_has_bits & 0x00000002u) {
318         target = stream->EnsureSpace(target);

```

(continues on next page)

(continued from previous page)

```

319     target = ::_pbi::WireFormatLite::WriteEnumToArray(
320         2, this->_internal_type(), target);
321 }
322
323 if (PROTOBUF_PREDICT_FALSE(_internal_metadata_.have_unknown_fields())) {
324     target = ::_pbi::WireFormat::InternalSerializeUnknownFieldsToArray(
325         _internal_metadata_.unknown_fields<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
326         ↳ (::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance), target, stream);
327 }
328 // @@protoc_insertion_point(serialize_to_array_end:tutorial.Person.PhoneNumber)
329 return target;
330 }
331
332 size_t Person_PhoneNumber::ByteSizeLong() const {
333     // @@protoc_insertion_point(message_byte_size_start:tutorial.Person.PhoneNumber)
334     size_t total_size = 0;
335
336     uint32_t cached_has_bits = 0;
337     // Prevent compiler warnings about cached_has_bits being unused
338     (void) cached_has_bits;
339
340     cached_has_bits = _has_bits_[0];
341     if (cached_has_bits & 0x00000003u) {
342         // optional string number = 1;
343         if (cached_has_bits & 0x00000001u) {
344             total_size += 1 +
345                 ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::StringSize(
346                 this->_internal_number());
347         }
348         // optional .tutorial.Person.PhoneType type = 2 [default = HOME];
349         if (cached_has_bits & 0x00000002u) {
350             total_size += 1 +
351                 ::_pbi::WireFormatLite::EnumSize(this->_internal_type());
352         }
353     }
354     return MaybeComputeUnknownFieldsSize(total_size, &cached_size_);
355 }
356
357 const ::PROTOBUF_NAMESPACE_ID::Message::ClassData Person_PhoneNumber::_class_data_ = {
358     ::PROTOBUF_NAMESPACE_ID::Message::CopyWithSizeCheck,
359     Person_PhoneNumber::MergeImpl
360 };
361
362 const ::PROTOBUF_NAMESPACE_ID::Message::ClassData* Person_PhoneNumber::GetClassData() {
363     ↳ const { return &_class_data_; }
364
365 void Person_PhoneNumber::MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to,
366     const ::PROTOBUF_NAMESPACE_ID::Message& from) {
367     static_cast<Person_PhoneNumber*>(to)->MergeFrom(
368         static_cast<const Person_PhoneNumber&>(from));
369 }

```

(continues on next page)

(continued from previous page)

```

369
370
371 void Person_PhoneNumber::MergeFrom(const Person_PhoneNumber& from) {
372 // @@protoc_insertion_point(class_specific_merge_from_start:tutorial.Person.PhoneNumber)
373     GOOGLE_DCHECK_NE(&from, this);
374     uint32_t cached_has_bits = 0;
375     (void) cached_has_bits;
376
377     cached_has_bits = from._has_bits_[0];
378     if (cached_has_bits & 0x000000003u) {
379         if (cached_has_bits & 0x000000001u) {
380             _internal_set_number(from._internal_number());
381         }
382         if (cached_has_bits & 0x000000002u) {
383             type_ = from.type_;
384         }
385         _has_bits_[0] |= cached_has_bits;
386     }
387     _internal_metadata_.MergeFrom<:PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
388 ↪ metadata_);
389 }
390
391 void Person_PhoneNumber::CopyFrom(const Person_PhoneNumber& from) {
392 // @@protoc_insertion_point(class_specific_copy_from_start:tutorial.Person.PhoneNumber)
393     if (&from == this) return;
394     Clear();
395     MergeFrom(from);
396 }
397
398 bool Person_PhoneNumber::IsInitialized() const {
399     return true;
400 }
401
402 void Person_PhoneNumber::InternalSwap(Person_PhoneNumber* other) {
403     using std::swap;
404     auto* lhs_arena = GetArenaForAllocation();
405     auto* rhs_arena = other->GetArenaForAllocation();
406     _internal_metadata_.InternalSwap(&other->_internal_metadata_);
407     swap(_has_bits_[0], other->_has_bits_[0]);
408     ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr::InternalSwap(
409         &number_, lhs_arena,
410         &other->number_, rhs_arena
411     );
412     swap(type_, other->type_);
413 }
414
415 ::PROTOBUF_NAMESPACE_ID::Metadata Person_PhoneNumber::GetMetadata() const {
416     return ::_pbi::AssignDescriptors(
417         &descriptor_table_hello_2eproto_getter, &descriptor_table_hello_2eproto_once,
418         file_level_metadata_hello_2eproto[0]);
419 }

```

(continues on next page)

(continued from previous page)

```

420 // =====
421
422 class Person::_Internal {
423 public:
424     using HasBits = decltype(std::declval<Person>()._has_bits_);
425     static void set_has_name(HasBits* has_bits) {
426         (*has_bits)[0] |= 1u;
427     }
428     static void set_has_id(HasBits* has_bits) {
429         (*has_bits)[0] |= 4u;
430     }
431     static void set_has_email(HasBits* has_bits) {
432         (*has_bits)[0] |= 2u;
433     }
434 };
435
436 Person::Person(::PROTOBUF_NAMESPACE_ID::Arena* arena,
437                bool is_message_owned)
438     : ::PROTOBUF_NAMESPACE_ID::Message(arena, is_message_owned),
439     phones_(arena) {
440     SharedCtor();
441     // @@protoc_insertion_point(arena_constructor:tutorial.Person)
442 }
443 Person::Person(const Person& from)
444     : ::PROTOBUF_NAMESPACE_ID::Message(),
445     _has_bits_(from._has_bits_),
446     phones_(from.phones_) {
447     _internal_metadata_.MergeFrom<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
448 ↪ metadata_);
449     name_.InitDefault();
450     #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
451     name_.Set("", GetArenaForAllocation());
452     #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
453     if (from._internal_has_name()) {
454         name_.Set(from._internal_name(),
455                 GetArenaForAllocation());
456     }
457     email_.InitDefault();
458     #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
459     email_.Set("", GetArenaForAllocation());
460     #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
461     if (from._internal_has_email()) {
462         email_.Set(from._internal_email(),
463                 GetArenaForAllocation());
464     }
465     id_ = from.id_;
466     // @@protoc_insertion_point(copy_constructor:tutorial.Person)
467 }
468
469 inline void Person::SharedCtor() {
470     name_.InitDefault();
471     #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING

```

(continues on next page)

(continued from previous page)

```

471     name_.Set("", GetArenaForAllocation());
472 #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
473 email_.InitDefault();
474 #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
475     email_.Set("", GetArenaForAllocation());
476 #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
477 id_ = 0;
478 }
479
480 Person::~Person() {
481     // @@protoc_insertion_point(destructor:tutorial.Person)
482     if (auto *arena = _internal_metadata_.DeleteReturnArena<:PROTOBUF_NAMESPACE_
↪ID::UnknownFieldSet>()) {
483         (void)arena;
484         return;
485     }
486     SharedDtor();
487 }
488
489 inline void Person::SharedDtor() {
490     GOOGLE_DCHECK(GetArenaForAllocation() == nullptr);
491     name_.Destroy();
492     email_.Destroy();
493 }
494
495 void Person::SetCachedSize(int size) const {
496     _cached_size_.Set(size);
497 }
498
499 void Person::Clear() {
500     // @@protoc_insertion_point(message_clear_start:tutorial.Person)
501     uint32_t cached_has_bits = 0;
502     // Prevent compiler warnings about cached_has_bits being unused
503     (void) cached_has_bits;
504
505     phones_.Clear();
506     cached_has_bits = _has_bits_[0];
507     if (cached_has_bits & 0x00000003u) {
508         if (cached_has_bits & 0x00000001u) {
509             name_.ClearNonDefaultToEmpty();
510         }
511         if (cached_has_bits & 0x00000002u) {
512             email_.ClearNonDefaultToEmpty();
513         }
514     }
515     id_ = 0;
516     _has_bits_.Clear();
517     _internal_metadata_.Clear<:PROTOBUF_NAMESPACE_ID::UnknownFieldSet>();
518 }
519
520 const char* Person::_InternalParse(const char* ptr, ::_pbi::ParseContext* ctx) {
521 #define CHK(x) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure

```

(continues on next page)

(continued from previous page)

```

522 _Internal::HasBits has_bits{};
523 while (!ctx->Done(&ptr)) {
524     uint32_t tag;
525     ptr = ::_pbi::ReadTag(ptr, &tag);
526     switch (tag >> 3) {
527         // optional string name = 1;
528         case 1:
529             if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 10)) {
530                 auto str = _internal_mutable_name();
531                 ptr = ::_pbi::InlineGreedyStringParser(str, ptr, ctx);
532                 CHK_(ptr);
533                 #ifndef NDEBUG
534                 ::_pbi::VerifyUTF8(str, "tutorial.Person.name");
535                 #endif // !NDEBUG
536             } else
537                 goto handle_unusual;
538             continue;
539         // optional int32 id = 2;
540         case 2:
541             if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 16)) {
542                 _Internal::set_has_id(&has_bits);
543                 id_ = ::PROTOBUF_NAMESPACE_ID::internal::ReadVarint32(&ptr);
544                 CHK_(ptr);
545             } else
546                 goto handle_unusual;
547             continue;
548         // optional string email = 3;
549         case 3:
550             if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 26)) {
551                 auto str = _internal_mutable_email();
552                 ptr = ::_pbi::InlineGreedyStringParser(str, ptr, ctx);
553                 CHK_(ptr);
554                 #ifndef NDEBUG
555                 ::_pbi::VerifyUTF8(str, "tutorial.Person.email");
556                 #endif // !NDEBUG
557             } else
558                 goto handle_unusual;
559             continue;
560         // repeated .tutorial.Person.PhoneNumber phones = 4;
561         case 4:
562             if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 34)) {
563                 ptr -= 1;
564                 do {
565                     ptr += 1;
566                     ptr = ctx->ParseMessage(_internal_add_phones(), ptr);
567                     CHK_(ptr);
568                     if (!ctx->DataAvailable(ptr)) break;
569                 } while (::PROTOBUF_NAMESPACE_ID::internal::ExpectTag<34>(ptr));
570             } else
571                 goto handle_unusual;
572             continue;
573         default:

```

(continues on next page)

(continued from previous page)

```

574     goto handle_unusual;
575 } // switch
576 handle_unusual:
577     if ((tag == 0) || ((tag & 7) == 4)) {
578         CHK_(ptr);
579         ctx->SetLastTag(tag);
580         goto message_done;
581     }
582     ptr = UnknownFieldParse(
583         tag,
584         _internal_metadata_.mutable_unknown_fields<::PROTOBUF_NAMESPACE_
↪ ID::UnknownFieldSet>(),
585         ptr, ctx);
586     CHK_(ptr != nullptr);
587 } // while
588 message_done:
589     _has_bits_.Or(has_bits);
590     return ptr;
591 failure:
592     ptr = nullptr;
593     goto message_done;
594 #undef CHK_
595 }
596
597 uint8_t* Person::_InternalSerialize(
598     uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const {
599     // @@protoc_insertion_point(serialize_to_array_start:tutorial.Person)
600     uint32_t cached_has_bits = 0;
601     (void) cached_has_bits;
602
603     cached_has_bits = _has_bits_[0];
604     // optional string name = 1;
605     if (cached_has_bits & 0x000000001u) {
606         ::PROTOBUF_NAMESPACE_ID::internal::WireFormat::VerifyUTF8StringNamedField(
607             this->_internal_name().data(), static_cast<int>(this->_internal_name().length()),
608             ::PROTOBUF_NAMESPACE_ID::internal::WireFormat::SERIALIZE,
609             "tutorial.Person.name");
610         target = stream->WriteStringMaybeAliased(
611             1, this->_internal_name(), target);
612     }
613
614     // optional int32 id = 2;
615     if (cached_has_bits & 0x000000004u) {
616         target = stream->EnsureSpace(target);
617         target = ::_pbi::WireFormatLite::WriteInt32ToArray(2, this->_internal_id(), target);
618     }
619
620     // optional string email = 3;
621     if (cached_has_bits & 0x000000002u) {
622         ::PROTOBUF_NAMESPACE_ID::internal::WireFormat::VerifyUTF8StringNamedField(
623             this->_internal_email().data(), static_cast<int>(this->_internal_email().length()),
624             ::PROTOBUF_NAMESPACE_ID::internal::WireFormat::SERIALIZE,

```

(continues on next page)

(continued from previous page)

```

625     "tutorial.Person.email");
626     target = stream->WriteStringMaybeAliased(
627         3, this->_internal_email(), target);
628 }
629
630 // repeated .tutorial.Person.PhoneNumber phones = 4;
631 for (unsigned i = 0,
632      n = static_cast<unsigned>(this->_internal_phones_size()); i < n; i++) {
633     const auto& repfield = this->_internal_phones(i);
634     target = ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::
635         InternalWriteMessage(4, repfield, repfield.GetCachedSize(), target, stream);
636 }
637
638 if (PROTOBUF_PREDICT_FALSE(_internal_metadata_.have_unknown_fields())) {
639     target = ::_pbi::WireFormat::InternalSerializeUnknownFieldsToArray(
640         _internal_metadata_.unknown_fields<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
641         < (::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance), target, stream);
642 }
643 // @@protoc_insertion_point(serialize_to_array_end:tutorial.Person)
644 return target;
645 }
646
647 size_t Person::ByteSizeLong() const {
648     // @@protoc_insertion_point(message_byte_size_start:tutorial.Person)
649     size_t total_size = 0;
650
651     uint32_t cached_has_bits = 0;
652     // Prevent compiler warnings about cached_has_bits being unused
653     (void) cached_has_bits;
654
655     // repeated .tutorial.Person.PhoneNumber phones = 4;
656     total_size += 1UL * this->_internal_phones_size();
657     for (const auto& msg : this->phones_) {
658         total_size +=
659             ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::MessageSize(msg);
660     }
661
662     cached_has_bits = _has_bits_[0];
663     if (cached_has_bits & 0x000000007u) {
664         // optional string name = 1;
665         if (cached_has_bits & 0x000000001u) {
666             total_size += 1 +
667                 ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::StringSize(
668                     this->_internal_name());
669         }
670
671         // optional string email = 3;
672         if (cached_has_bits & 0x000000002u) {
673             total_size += 1 +
674                 ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::StringSize(
675                     this->_internal_email());
676         }
677     }
678 }

```

(continues on next page)

(continued from previous page)

```

676 // optional int32 id = 2;
677 if (cached_has_bits & 0x00000004u) {
678     total_size += ::_pbi::WireFormatLite::Int32SizePlusOne(this->_internal_id());
679 }
680 }
681 }
682 return MaybeComputeUnknownFieldsSize(total_size, &_cached_size_);
683 }
684 }
685
686 const ::PROTOBUF_NAMESPACE_ID::Message::ClassData Person::_class_data_ = {
687     ::PROTOBUF_NAMESPACE_ID::Message::CopyWithSizeCheck,
688     Person::MergeImpl
689 };
690 const ::PROTOBUF_NAMESPACE_ID::Message::ClassData*Person::GetClassData() const { return &
↪ _class_data_; }
691
692 void Person::MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to,
693     const ::PROTOBUF_NAMESPACE_ID::Message& from) {
694     static_cast<Person*>(to)->MergeFrom(
695         static_cast<const Person&>(from));
696 }
697
698 void Person::MergeFrom(const Person& from) {
699 // @@protoc_insertion_point(class_specific_merge_from_start:tutorial.Person)
700     GOOGLE_DCHECK_NE(&from, this);
701     uint32_t cached_has_bits = 0;
702     (void) cached_has_bits;
703
704     phones_.MergeFrom(from.phones_);
705     cached_has_bits = from._has_bits_[0];
706     if (cached_has_bits & 0x00000007u) {
707         if (cached_has_bits & 0x00000001u) {
708             _internal_set_name(from._internal_name());
709         }
710         if (cached_has_bits & 0x00000002u) {
711             _internal_set_email(from._internal_email());
712         }
713         if (cached_has_bits & 0x00000004u) {
714             id_ = from.id_;
715         }
716         _has_bits_[0] |= cached_has_bits;
717     }
718     _internal_metadata_.MergeFrom<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
↪ metadata_);
719 }
720
721 void Person::CopyFrom(const Person& from) {
722 // @@protoc_insertion_point(class_specific_copy_from_start:tutorial.Person)
723     if (&from == this) return;
724     Clear();
725 }

```

(continues on next page)

(continued from previous page)

```

726     MergeFrom(from);
727 }
728
729 bool Person::IsInitialized() const {
730     return true;
731 }
732
733 void Person::InternalSwap(Person* other) {
734     using std::swap;
735     auto* lhs_arena = GetArenaForAllocation();
736     auto* rhs_arena = other->GetArenaForAllocation();
737     _internal_metadata_.InternalSwap(&other->_internal_metadata_);
738     swap(_has_bits_[0], other->_has_bits_[0]);
739     phones_.InternalSwap(&other->phones_);
740     ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr::InternalSwap(
741         &name_, lhs_arena,
742         &other->name_, rhs_arena
743     );
744     ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr::InternalSwap(
745         &email_, lhs_arena,
746         &other->email_, rhs_arena
747     );
748     swap(id_, other->id_);
749 }
750
751 ::PROTOBUF_NAMESPACE_ID::Metadata Person::GetMetadata() const {
752     return ::_pbi::AssignDescriptors(
753         &descriptor_table_hello_2eproto_getter, &descriptor_table_hello_2eproto_once,
754         file_level_metadata_hello_2eproto[1]);
755 }
756
757 // =====
758
759 class AddressBook::_Internal {
760 public:
761 };
762
763 AddressBook::AddressBook(::PROTOBUF_NAMESPACE_ID::Arena* arena,
764                          bool is_message_owned)
765     : ::PROTOBUF_NAMESPACE_ID::Message(arena, is_message_owned),
766     people_(arena) {
767     SharedCtor();
768     // @@protoc_insertion_point(arena_constructor:tutorial.AddressBook)
769 }
770
771 AddressBook::AddressBook(const AddressBook& from)
772     : ::PROTOBUF_NAMESPACE_ID::Message(),
773     people_(from.people_) {
774     _internal_metadata_.MergeFrom<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
775     metadata_);
776     // @@protoc_insertion_point(copy_constructor:tutorial.AddressBook)
777 }

```

(continues on next page)

(continued from previous page)

```

777 inline void AddressBook::SharedCtor() {
778 }
779
780 AddressBook::~AddressBook() {
781     // @@protoc_insertion_point(destructor:tutorial.AddressBook)
782     if (auto *arena = _internal_metadata_.DeleteReturnArena<:PROTOBUF_NAMESPACE_
↪ ID::UnknownFieldSet>()) {
783         (void)arena;
784         return;
785     }
786     SharedDtor();
787 }
788
789 inline void AddressBook::SharedDtor() {
790     GOOGLE_DCHECK(GetArenaForAllocation() == nullptr);
791 }
792
793 void AddressBook::SetCachedSize(int size) const {
794     _cached_size_.Set(size);
795 }
796
797 void AddressBook::Clear() {
798     // @@protoc_insertion_point(message_clear_start:tutorial.AddressBook)
799     uint32_t cached_has_bits = 0;
800     // Prevent compiler warnings about cached_has_bits being unused
801     (void) cached_has_bits;
802
803     people_.Clear();
804     _internal_metadata_.Clear<:PROTOBUF_NAMESPACE_ID::UnknownFieldSet>();
805 }
806
807 const char* AddressBook::_InternalParse(const char* ptr, ::_pbi::ParseContext* ctx) {
808     #define CHK_(x) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
809     while (!ctx->Done(&ptr)) {
810         uint32_t tag;
811         ptr = ::_pbi::ReadTag(ptr, &tag);
812         switch (tag >> 3) {
813             // repeated .tutorial.Person people = 1;
814             case 1:
815                 if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 10)) {
816                     ptr -= 1;
817                     do {
818                         ptr += 1;
819                         ptr = ctx->ParseMessage(_internal_add_people(), ptr);
820                         CHK_(ptr);
821                         if (!ctx->DataAvailable(ptr)) break;
822                     } while (::PROTOBUF_NAMESPACE_ID::internal::ExpectTag<10>(ptr));
823                 } else
824                     goto handle_unusual;
825                 continue;
826             default:
827                 goto handle_unusual;

```

(continues on next page)

(continued from previous page)

```

828     } // switch
829 handle_unusual:
830     if ((tag == 0) || ((tag & 7) == 4)) {
831         CHK_(ptr);
832         ctx->SetLastTag(tag);
833         goto message_done;
834     }
835     ptr = UnknownFieldParse(
836         tag,
837         _internal_metadata_.mutable_unknown_fields<:PROTOBUF_NAMESPACE_
↪ ID::UnknownFieldSet>(),
838         ptr, ctx);
839     CHK_(ptr != nullptr);
840 } // while
841 message_done:
842     return ptr;
843 failure:
844     ptr = nullptr;
845     goto message_done;
846 #undef CHK_
847 }
848
849 uint8_t* AddressBook::_InternalSerialize(
850     uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const {
851     // @@protoc_insertion_point(serialize_to_array_start:tutorial.AddressBook)
852     uint32_t cached_has_bits = 0;
853     (void) cached_has_bits;
854
855     // repeated .tutorial.Person people = 1;
856     for (unsigned i = 0,
857         n = static_cast<unsigned>(this->_internal_people_size()); i < n; i++) {
858         const auto& repfield = this->_internal_people(i);
859         target = ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::
860             InternalWriteMessage(1, repfield, repfield.GetCachedSize(), target, stream);
861     }
862
863     if (PROTOBUF_PREDICT_FALSE(_internal_metadata_.have_unknown_fields())) {
864         target = ::_pbi::WireFormat::InternalSerializeUnknownFieldsToArray(
865             _internal_metadata_.unknown_fields<:PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
↪ (::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance), target, stream);
866     }
867     // @@protoc_insertion_point(serialize_to_array_end:tutorial.AddressBook)
868     return target;
869 }
870
871 size_t AddressBook::ByteSizeLong() const {
872     // @@protoc_insertion_point(message_byte_size_start:tutorial.AddressBook)
873     size_t total_size = 0;
874
875     uint32_t cached_has_bits = 0;
876     // Prevent compiler warnings about cached_has_bits being unused
877     (void) cached_has_bits;

```

(continues on next page)

(continued from previous page)

```

878
879 // repeated .tutorial.Person people = 1;
880 total_size += 1UL * this->_internal_people_size();
881 for (const auto& msg : this->people_) {
882     total_size +=
883         ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::MessageSize(msg);
884 }
885
886 return MaybeComputeUnknownFieldsSize(total_size, &_cached_size_);
887 }
888
889 const ::PROTOBUF_NAMESPACE_ID::Message::ClassData AddressBook::_class_data_ = {
890     ::PROTOBUF_NAMESPACE_ID::Message::CopyWithSizeCheck,
891     AddressBook::MergeImpl
892 };
893 const ::PROTOBUF_NAMESPACE_ID::Message::ClassData*AddressBook::GetClassData() const {
894     ↪return &_class_data_; }
895
896 void AddressBook::MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to,
897                             const ::PROTOBUF_NAMESPACE_ID::Message& from) {
898     static_cast<AddressBook*>(to)->MergeFrom(
899         static_cast<const AddressBook&>(from));
900 }
901
902 void AddressBook::MergeFrom(const AddressBook& from) {
903     // @@protoc_insertion_point(class_specific_merge_from_start:tutorial.AddressBook)
904     GOOGLE_DCHECK_NE(&from, this);
905     uint32_t cached_has_bits = 0;
906     (void) cached_has_bits;
907
908     people_.MergeFrom(from.people_);
909     _internal_metadata_.MergeFrom<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
910     ↪metadata_);
911 }
912
913 void AddressBook::CopyFrom(const AddressBook& from) {
914     // @@protoc_insertion_point(class_specific_copy_from_start:tutorial.AddressBook)
915     if (&from == this) return;
916     Clear();
917     MergeFrom(from);
918 }
919
920 bool AddressBook::IsInitialized() const {
921     return true;
922 }
923
924 void AddressBook::InternalSwap(AddressBook* other) {
925     using std::swap;
926     _internal_metadata_.InternalSwap(&other->_internal_metadata_);
927     people_.InternalSwap(&other->people_);
928 }

```

(continues on next page)

(continued from previous page)

```

928
929 ::PROTOBUF_NAMESPACE_ID::Metadata AddressBook::GetMetadata() const {
930     return ::_pbi::AssignDescriptors(
931         &descriptor_table_hello_2eproto_getter, &descriptor_table_hello_2eproto_once,
932         file_level_metadata_hello_2eproto[2]);
933 }
934
935 // @@protoc_insertion_point(namespace_scope)
936 } // namespace tutorial
937 PROTOBUF_NAMESPACE_OPEN
938 template<> PROTOBUF_NOINLINE ::tutorial::Person_PhoneNumber*
939 Arena::CreateMaybeMessage< ::tutorial::Person_PhoneNumber >(Arena* arena) {
940     return Arena::CreateMessageInternal< ::tutorial::Person_PhoneNumber >(arena);
941 }
942 template<> PROTOBUF_NOINLINE ::tutorial::Person*
943 Arena::CreateMaybeMessage< ::tutorial::Person >(Arena* arena) {
944     return Arena::CreateMessageInternal< ::tutorial::Person >(arena);
945 }
946 template<> PROTOBUF_NOINLINE ::tutorial::AddressBook*
947 Arena::CreateMaybeMessage< ::tutorial::AddressBook >(Arena* arena) {
948     return Arena::CreateMessageInternal< ::tutorial::AddressBook >(arena);
949 }
950 PROTOBUF_NAMESPACE_CLOSE
951
952 // @@protoc_insertion_point(global_scope)
953 #include <google/protobuf/port_undef.inc>

```


16.1 Install

See <https://grpc.io/docs/languages/cpp/quickstart/>

```
git clone --recurse-submodules -b v1.46.3 --depth 1 --shallow-submodules https://github.
↳com/grpc/grpc
mkdir build
cd build
cmake -Dgrpc_INSTALL=ON -Dgrpc_BUILD_TESTS=OFF -DCMAKE_INSTALL_PREFIX=/ceph-fj/fangjun/
↳software/grpc-1.46.3 .. 2>&1 | tee cmake-configure-1.log
make -j20 2>&1 | tee make-1.log
make install 2>&1 | tee make-2.log
```


17.1 TODOs

- Striking gold in binutils
<https://lwn.net/Articles/274859/>
- A ToC of the 20 part linker essay
<https://lwn.net/Articles/276782/>

There are other resources for linkers and loaders, see

- Executables linking and loading reading
<http://research.tedneward.com/reading/software/linking-loading/index.html>
- Optimizing real-world applications with GCC Link Time Optimization
<https://pdfs.semanticscholar.org/6adf/872e3533f40a607f39cdeaf264585efde9af.pdf>
by Honza Hubicka, whose scholar page is <https://scholar.google.cz/citations?user=vhXJ0JEAAAAJ&hl=en>

LINKER AND LOADER

18.1 References

- A ToC of the 20 part linker essay
<https://lwn.net/Articles/276782/>, which is written by Ian Lance Taylor
 1. Introduction, personal history, first half of what's-a-linker
 2. What's-a-linker: Dynamic linking, linker data types, linker operation
 3. Address spaces, Object file formats
 4. Shared Libraries
 5. More Shared Libraries -- specifically, linker implementation; ELF Symbols
- <https://www.ucw.cz/~hubicka/>, author of the gold linker
- Rod Evans: Surfing With a Linker Alien <http://www.linker-aliens.org/blogs/rie/>
 1. Hello there
- Michael Walker's Weblog <http://www.linker-aliens.org/blogs/msw/>
 1. Hello World
 2. How to build a Shared Library
 3. Library Bindings - let's be a little bit more precise shall we

Note: It shows the usage of LD_DEBUG, pldd, ldd, pgrep elfdump.

- Solaris Linking Blogs (Combined Index) <http://www.linker-aliens.org/blogs/>
- LD_LIBRARY_PATH - just say no
http://www.linker-aliens.org/blogs/rie/entry/tt_ld_library_path_tt/
- <https://github.com/berkus/odin/blob/master/tools/sjofn/sjofn.c>
An ELF linker. Read its source code!

18.2 Questions

1. How to view PLT?
2. How to view the relocation information? How many types of relocation are there?
3. What PIC code and non-PIC code look like?
4. What is lazy binding and how to use LD_BIND_NOW?
5. What is PLT and GOT?

19.1 aishell

19.1.1 AM training

The first one was added on 2019-02-01.

`asr_train.py` is in `espnet/bin/asr_train.py`, which invokes `espnet.asr.pytorch_backend.asr.train`.

The model is from `espnet.nets.pytorch_backend.e2e_asr.E2E`.

The encoder type *vggblstm*, 3 layers, hidden dim, 1024, proj dim 1024, subsampling 1_2_2_1_1.

Command is:

```
asr_train.py \  
  --config conf/train.yaml \  
  --preprocess-conf \  
  --ngpu 1 \  
  --backend pytorch \  
  --outdir exp/xxx \  
  --debugmode 1 \  
  --dict data/lang_char/train_sp_units.txt \  
  --minibatches 0 \  
  --verbose 0 \  
  --resume \  
  --train-json xxx/data.json \  
  --valid-json yyy/data.json
```


20.1 Tutorials

- <https://cmake.org/cmake/help/latest/guide/tutorial/index.html>

20.2 Install

Go to <https://github.com/Kitware/CMake/releases> for download.

```
wget https://github.com/Kitware/CMake/releases/download/v3.10.3/cmake-3.10.3-Linux-x86_
↪ 64.sh
chmod +x ./cmake-3.10.3-Linux-x86_64.sh
./cmake-3.10.3-Linux-x86_64.sh --help
mkdir /path/to/software/cmake-3.10.3
./cmake-3.10.3-Linux-x86_64.sh --prefix=/path/to/software/cmake-3.10.3 --skip-license
export PATH=/path/to/software/cmake-3.10.3/bin:$PATH
```


EECS E6870 SPEECH RECOGNITION

21.1 Notes

- <https://www.ee.columbia.edu/~stanchen/spring16/e6870/outline.html>
 - username: speech
 - password: pythonrules