notes

fangjun

Jul 24, 2022

CONTENTS:

Download this website in a single pdf file.

CHAPTER

ONE

SPHINX

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`.
```

CHAPTER

TWO

GIT

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.

  G   H   I   J
   \ /     \ /
    D   E   F
     \  |  / \
      \ | /   |
       \|/    |
        B     C
         \   /
          \ /
           A

  A =       = A^0
  B = A^    = A^1     = A~1
  C =       = A^2
  D = A^^   = A^1^1   = A~2
  E = B^2   = A^^2
  F = B^3   = A^^3
  G = A^^^  = A^1^1^1 = A~3
  H = D^2   = B^^2    = A^^^2  = A~2^2
  I = F^    = B^3^    = A^^3^
  J = F^2   = B^3^2   = A^^3^2
```

CHAPTER

THREE

DOCKER

3.1 Installation

3.1.1 macos

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

CHAPTER

FOUR

LATEX

4.1 TikZ

4.1.1 Basics

CHAPTER

FIVE

KALDI

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}

CHAPTER

SIX

BASH

6.1 sort

Sort files in the folder `t`. The filename has the patter `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
```

6.3 ffmeg

```
ffprobe xxx.opus
```

6.3.1 Convert format

- opus to wav

```
ffmpeg -i input.opus output.wav

ffmpeg -i input.opus -acodec pcm_s16le -ac 1 -ar 16000 output.wav
```

- Extract part of a file

```
# extract 30 seconds starting at offset 1 minute
ffmpeg -i input.opus -ss 60 -t 30 output.wav
# or use HH:MM:SS format
ffmpeg -i input.opus -ss 0:01:00 -t 0:00:30 output.wav
```

6.3.2 References

See https://gist.github.com/whizkydee/804d7e290f46c73f55a84db8a8936d74

CHAPTER

SEVEN

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
```

CHAPTER

EIGHT

TORCH

8.1 torch.load and torch.save

Listing 1: ./code/load-and-save.py

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

import torch
import tempfile


def main():
    a = torch.arange(3)
    with tempfile.NamedTemporaryFile() as f:
        torch.save(a, f)
        f.seek(0)
        b = torch.load(f)
        assert torch.all(torch.eq(a, b)), (a, b)


if __name__ == "__main__":
    main()
```

8.2 DDP

8.2.1 Initialization

8.3 TorchScript

8.3.1 Hello

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

`torch.jit.script` as a decorator

Listing 2: ./code/1-ex.py

```
@torch.jit.script
def adder(x: int):
    return x + 1


def test_adder():
    assert isinstance(adder, torch.jit.ScriptFunction)
    print(adder.graph)
    print("-" * 10)
    print(adder.code)
    adder.save("adder.pt")

    my_adder = torch.jit.load("adder.pt")

    assert isinstance(my_adder, torch.jit._script.RecursiveScriptModule)
    assert isinstance(my_adder, torch.jit.ScriptModule)
    assert not isinstance(my_adder, torch.jit.ScriptFunction)
    print(my_adder(torch.tensor([3])))


"""
graph(%x.1 : int):
  %2 : int = prim::Constant[value=1]() # ./1-ex.py:8:15
  %3 : int = aten::add(%x.1, %2) # ./1-ex.py:8:11
  return (%3)

----------
def adder(x: int) -> int:
  return torch.add(x, 1)

4
"""
```

`torch.jit.script` as a function

Listing 3: ./code/2-ex.py

```
def adder(x: int):
    return x + 2


def test_adder():
    adder_func = torch.jit.script(adder)
    assert isinstance(adder_func, torch.jit.ScriptFunction)
    print(adder_func.graph)
    print(adder_func(3))


```

(continues on next page)

(continued from previous page)

```
"""
graph(%x.1 : int):
  %2 : int = prim::Constant[value=2]() # ./2-ex.py:6:15
  %3 : int = aten::add(%x.1, %2) # ./2-ex.py:6:11
  return (%3)

5
"""
```

torchscript a module

Listing 4: ./code/3-ex.py

```
class MyModel(torch.nn.Module):
    def __init__(self):
        super().__init__()
        self.p = torch.nn.Parameter(torch.tensor([2.0]))

    def forward(self, x: torch.Tensor):
        return self.p * x


def test_my_model():
    model = MyModel()
    scripted_model = torch.jit.script(model)
    print(scripted_model.graph)
    print("-" * 10)
    print(scripted_model.code)
    print(scripted_model(torch.tensor([10])))


"""
graph(%self : __torch__.MyModel,
      %x.1 : Tensor):
  %p : Tensor = prim::GetAttr[name="p"](%self)
  %4 : Tensor = aten::mul(%p, %x.1) # ./3-ex.py:12:15
  return (%4)

----------
def forward(self,
    x: Tensor) -> Tensor:
  p = self.p
  return torch.mul(p, x)
"""
```

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 5: ./code/4-ex.py

```
class MyModel(torch.nn.Module):
    def __init__(self):
        super().__init__()
        self.p = torch.nn.Parameter(torch.tensor([2.0]))

    def foobar(self, x: torch.Tensor):
        return x + 3

    def foo(self, x: torch.Tensor):
        return self.foobar(x)

    def bar(self, x: torch.Tensor):
        return self.p - x

    @torch.jit.export
    def baz(self, x: torch.Tensor):
        return self.p + x + 2

    def forward(self, x: torch.Tensor):
        return self.p * x


def test_my_model():
    MyModel.foo = torch.jit.export(MyModel.foo)  # manually export

    # Note: forward is exported by default. We ignore it here manually
    MyModel.forward = torch.jit.ignore(MyModel.forward)

    model = MyModel()
    scripted_model = torch.jit.script(model)
    assert hasattr(scripted_model, "foo")
    assert hasattr(scripted_model, "baz")
    assert hasattr(scripted_model, "foobar")  # because it is called by `foo`
    assert not hasattr(scripted_model, "bar")

    scripted_model.save("foo.pt")

    m = torch.jit.load("foo.pt")
    print(m.foo(torch.tensor([1])))
    print(m.baz(torch.tensor([1])))
```

(continues on next page)

(continued from previous page)

```
"""
graph(%self : __torch__.MyModel,
      %x.1 : Tensor):
  %p : Tensor = prim::GetAttr[name="p"](%self)
  %4 : Tensor = aten::mul(%p, %x.1) # ./3-ex.py:12:15
  return (%4)

----------
def forward(self,
    x: Tensor) -> Tensor:
  p = self.p
  return torch.mul(p, x)
"""
```

8.3.2 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 6: ./code/load-in-cpp/Makefile

```
USE_CXX11_ABI := $(shell python3 -c 'import torch; print(int(torch.compiled_with_cxx11_abi()))')
TORCH_INSTALL_DIR := $(shell python3 -c 'import os; import torch; print(os.path.dirname(torch.__file__))')

$(info USE_CXX11_ABI $(USE_CXX11_ABI))
$(info TORCH_INSTALL_DIR $(TORCH_INSTALL_DIR))

CXXFLAGS := -I$(TORCH_INSTALL_DIR)/include
CXXFLAGS += -I$(TORCH_INSTALL_DIR)/include/torch/csrc/api/include
CXXFLAGS += -I$(TORCH_INSTALL_DIR)/include/TH
CXXFLAGS += -I$(TORCH_INSTALL_DIR)/include/THC
CXXFLAGS += -std=c++14
CXXFLAGS += -D_GLIBCXX_USE_CXX11_ABI=$(USE_CXX11_ABI)

CXXFLAGS += -Wno-unknown-pragmas # disable omp warnings

LDFLAGS := -L$(TORCH_INSTALL_DIR)/lib
LDFLAGS += -lc10 -ltorch_cpu
# LDFLAGS += -lc10 -ltorch
LDFLAGS += -Wl,-rpath,$(TORCH_INSTALL_DIR)/lib

HAS_CUDA := $(shell python3 -c 'import torch; print("yes" if torch.cuda.is_available() else "no")')
$(info has cuda $(HAS_CUDA))

ifeq ($(HAS_CUDA),yes)
```

(continues on next page)

(continued from previous page)

```
CUDA_HOME := $(shell which nvcc | xargs dirname | xargs dirname)
CXXFLAGS += -I$(CUDA_HOME)/include
LDFLAGS += -L$(CUDA_HOME)/lib64
LDFLAGS += -lcudart -lc10_cuda -ltorch_cuda
LDFLAGS += -Wl,-rpath,$(CUDA_HOME)/lib64
endif

.PHONY: clean

main: main.o
        $(CXX) -o $@ $< $(LDFLAGS)

main.o: main.cc
        $(CXX) $(CXXFLAGS) -c -o $@ $<

clean:
        $(RM) main.o main
```

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

Listing 7: ./code/load-in-cpp/main.cc

```
#include "torch/script.h"

int main() {
  // see torch/csrc/jit/module.h
  torch::jit::Module m = torch::jit::load("../foo.pt");
  std::cout << "is training: " << m.is_training() << "\n";
  m.eval();
  std::cout << "after m.eval(): is training: " << m.is_training() << "\n";
  torch::Tensor x = torch::tensor({1, 2, 3}, torch::kFloat);
  torch::Tensor y = m.run_method("baz", x).toTensor();
  std::cout << y << "\n";

  return 0;
}
```

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 \
```

(continues on next page)

(continued from previous page)

```
    -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} ]
```

8.3.3 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 8: ./code/array_ref/main.cc (Check size)

```
struct Foo {
  const int32_t *p;
  size_t len;
};

static void TestSize() {
  // Note: The data pointer in ArrayRef is const!
  static_assert(sizeof(torch::ArrayRef<int32_t>) == sizeof(Foo), "");
}
```

Default constructed

Listing 9: ./code/array_ref/main.cc (Default constructor)

```
static void TestDefaultConstructor() {
  torch::ArrayRef<int32_t> a;
  TORCH_CHECK(a.data() == nullptr);
  TORCH_CHECK(a.size() == 0);
  TORCH_CHECK(a.empty() == true);

  TORCH_CHECK(a.begin() == nullptr);
  TORCH_CHECK(a.end() == nullptr);
}
```

From a single element

Listing 10: ./code/array_ref/main.cc (From a single element)

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

From an initializer list

Listing 11: ./code/array_ref/main.cc (From an initializer list)

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

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`

8.3.4 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 12: ./code/scalar-type/members.cc

```
#define AT_FORALL_SCALAR_TYPES_WITH_COMPLEX_EXCEPT_COMPLEX_HALF(_) \
  _(uint8_t, Byte)                                                 \
  _(int8_t, Char)                                                  \
  _(int16_t, Short)                                                \
  _(int, Int)                                                      \
  _(int64_t, Long)                                                 \
  _(at::Half, Half)                                                \
  _(float, Float)                                                  \
  _(double, Double)                                                \
  _(c10::complex<float>, ComplexFloat)                             \
  _(c10::complex<double>, ComplexDouble)                           \
  _(bool, Bool)                                                    \
  _(at::BFloat16, BFloat16)
```

Some aliases

Listing 13: ./code/scalar-type/main.cc (alias)

```
static void TestAlias() {
  static_assert(c10::ScalarType::Int == c10::kInt, "");
  static_assert(c10::ScalarType::Byte == c10::kByte, "");
}
```

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

```
// See torch/csrc/api/include/torch/types.h
using Dtype = at::ScalarType;

/// Fixed width dtypes.
constexpr auto kUInt8 = at::kByte;
constexpr auto kInt8 = at::kChar;
constexpr auto kInt16 = at::kShort;
constexpr auto kInt32 = at::kInt;
constexpr auto kInt64 = at::kLong;
constexpr auto kFloat16 = at::kHalf;
constexpr auto kFloat32 = at::kFloat;
constexpr auto kFloat64 = at::kDouble;

/// Rust-style short dtypes.
constexpr auto kU8 = kUInt8;
```

(continues on next page)

(continued from previous page)

```
constexpr auto kI8 = kInt8;
constexpr auto kI16 = kInt16;
constexpr auto kI32 = kInt32;
constexpr auto kI64 = kInt64;
constexpr auto kF16 = kFloat16;
constexpr auto kF32 = kFloat32;
constexpr auto kF64 = kFloat64;
```

ScalarType to CPP type

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

```
static void TestScalarTypeToCppType() {
  static_assert(
      std::is_same<
          int32_t, //
          c10::impl::ScalarTypeToCPPType<c10 ::ScalarType::Int>::type>::value,
      "");
}
```

CPP type to ScalarType

Listing 16: ./code/scalar-type/main.cc

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

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

8.3.5 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 17: ./code/type-meta/main.cc (Check size)

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

Constructors

Listing 18: ./code/type-meta/main.cc (Make)

```
static void TestConstructor() {
  caffe2::TypeMeta t = caffe2::TypeMeta::Make<int32_t>();
  TORCH_CHECK(t.Match<int32_t>());

  TORCH_CHECK(t.isScalarType());

  TORCH_CHECK(t.isScalarType(torch::kInt));
  TORCH_CHECK(t.isScalarType(torch::kFloat) == false);

  TORCH_CHECK(t.name() == "int");
}
```

Operations with ScalarType

Listing 19: ./code/type-meta/main.cc (Operations with ScalarType)

```
static void TestFromScalarType() {
  caffe2::TypeMeta t = caffe2::TypeMeta::fromScalarType(torch::kDouble);

  TORCH_CHECK(t.isScalarType(torch::kDouble));
  TORCH_CHECK(t.name() == "double");

  TORCH_CHECK(t.toScalarType() == torch::kDouble);
  TORCH_CHECK(t == torch::kDouble);
  TORCH_CHECK(t != torch::kFloat);
  TORCH_CHECK(torch::kInt != t);
```

8.3.6 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 20: ./code/device/main.cc

```
void TestDeviceType() {
  torch::DeviceType d = torch::kCPU;
  std::ostringstream os;
  os << d;
```

(continues on next page)

(continued from previous page)

```
  TORCH_CHECK(os.str() == "cpu");

  TORCH_CHECK(DeviceTypeName(d /*,lower_case=false*/) == "CPU");
  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 21: ./code/device/main.cc (Constructors)

```
void TestDeviceConstructorCPU() {
  torch::Device d(torch::kCPU);
  TORCH_CHECK(d.is_cpu() == true);
  TORCH_CHECK(d.is_cuda() == false);
  TORCH_CHECK(d.type() == torch::kCPU);
  TORCH_CHECK(d.has_index() == false);
  TORCH_CHECK(d.index() == -1);
  TORCH_CHECK(d.str() == "cpu");
}

void TestDeviceConstructorCUDA() {
  torch::Device d(torch::kCUDA, 3);
  TORCH_CHECK(d.is_cpu() == false);
  TORCH_CHECK(d.is_cuda() == true);
  TORCH_CHECK(d.type() == torch::kCUDA);
  TORCH_CHECK(d.has_index() == true);
  TORCH_CHECK(d.index() == 3);
  TORCH_CHECK(d.str() == "cuda:3");

  d.set_index(2);
  TORCH_CHECK(d.index() == 2);
  TORCH_CHECK(d.str() == "cuda:2");

  d = torch::Device("cpu");
  TORCH_CHECK(d.is_cpu() == true);

  d = torch::Device("CPU");
  TORCH_CHECK(d.is_cpu() == true);

  d = torch::Device("cuda:1");
  TORCH_CHECK(d.is_cuda() == true);
  TORCH_CHECK(d.index() == 1);

  d = torch::Device("CUDA:1");
  TORCH_CHECK(d.is_cuda() == true);
  TORCH_CHECK(d.index() == 1);
}
```

8.3.7 TensorOptions

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

Constructors (not recommended)

Listing 22: ./code/tensor-options/main.cc (Not recommended constructors)

```
void TestConstructor() {
  // not recommended
  torch::TensorOptions opt1(torch::kCPU);
  torch::TensorOptions opt2(torch::Device(torch::kCPU));
  torch::TensorOptions opt3(torch::Device({torch::kCUDA, 1}));
  torch::TensorOptions opt4("cpu");
  // torch::TensorOptions opt5("CPU") // error;
  torch::TensorOptions opt6("cuda:1");
  // torch::TensorOptions opt7("CUDA:1"); // error

  // not recommended, from a scalar type (implicit)
  torch::TensorOptions opt8(torch::kInt32);
}
```

Constructors (Recommended)

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

```
void TestConstructor2() {
  // recommended
  torch::TensorOptions opt1 = torch::dtype(torch::kFloat);
  torch::TensorOptions opt2 = torch::dtype(caffe2::TypeMeta::Make<float>());
  torch::TensorOptions opt3 = torch::device(torch::kCPU);
  torch::TensorOptions opt4 = torch::device({torch::kCUDA, 1});
  // Note: torch::device() returns a TensorOptions
  // while torch::Device() is the constructor of a class

  torch::TensorOptions opt5 = torch::requires_grad(true);
  std::cout << opt5 << "\n";
  // TensorOptions(dtype=float (default), device=cpu (default), layout=Strided
  // (default), requires_grad=true, pinned_memory=false (default),
  // memory_format=(nullopt))

  torch::TensorOptions opt6 = torch::dtype<float>();
  std::cout << torch::toString(opt6) << "\n";
  // TensorOptions(dtype=float, device=cpu (default), layout=Strided (default),
  // requires_grad=false (default), pinned_memory=false (default),
  // memory_format=(nullopt))

  std::cout << "default:" << torch::TensorOptions() << "\n";
  // default:TensorOptions(dtype=float (default), device=cpu (default),
  // layout=Strided (default), requires_grad=false (default),
```

(continues on next page)

(continued from previous page)

```
  // pinned_memory=false (default), memory_format=(nullopt))
}
```

Methods

Listing 24: ./code/tensor-options/main.cc (Methods)

```
void TestMethods() {
  torch::TensorOptions opts = torch::dtype<float>();
  TORCH_CHECK(opts.device() == torch::Device(torch::kCPU));
  // It has not device_type()!
  TORCH_CHECK(opts.device() == torch::kCPU);
  TORCH_CHECK(opts.device().type() == torch::kCPU);
  TORCH_CHECK(opts.requires_grad() == false);

  torch::TensorOptions opts2 =
      opts.device("cuda:2").dtype(torch::kInt).requires_grad(false);

  TORCH_CHECK(opts2.dtype() == caffe2::TypeMeta::Make<int32_t>());
  TORCH_CHECK(opts2.dtype() == torch::kInt32);
  TORCH_CHECK(opts2.requires_grad() == false);
}
```

8.3.8 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 25: ./code/tensor-creation/main.cc

```cpp
static void FromStdVecotr() {
  torch::Tensor t1 = torch::tensor(std::vector<int32_t>{1, 2, 3});
  TORCH_CHECK(t1.scalar_type() == torch::kLong);
  t1 = t1.to(torch::kInt);
  const int32_t *p1 = t1.data_ptr<int32_t>();
  TORCH_CHECK(p1[0] == 1);
  TORCH_CHECK(p1[1] == 2);
  TORCH_CHECK(p1[2] == 3);

  torch::Tensor t2 = torch::tensor(std::vector<float>{1, 2, 3});
  TORCH_CHECK(t2.scalar_type() == torch::kFloat);

  torch::Tensor t3 =
      torch::tensor(std::vector<double>{1, 2, 3}, torch::kDouble);
  TORCH_CHECK(t3.scalar_type() == torch::kDouble);

  torch::Tensor t4 =
      torch::tensor(std::vector<double>{1, 2, 3},
                    torch::dtype(torch::kDouble).device("cuda:0"));
  TORCH_CHECK(t4.is_cuda());
}
```

From scalar

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

```cpp
static void FromScalar() {
  torch::Tensor t = torch::tensor(3);
  TORCH_CHECK(t.item<int64_t>() == 3);

  torch::Tensor t2 = torch::tensor(0.5);
  TORCH_CHECK(t2.scalar_type() == torch::kFloat);
}
```

From initializer list

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

```cpp
static void FromInitializerList() {
  torch::Tensor t1 = torch::tensor({1, 2, 3});
  torch::Tensor t2 = torch::tensor(std::vector<int32_t>{1, 2, 3});
  TORCH_CHECK(torch::allclose(t1, t2));

  torch::Tensor t3 = torch::tensor({{1, 2, 3}, {4, 5, 6}});
  TORCH_CHECK(t3.dim() == 2);

```

(continues on next page)

(continued from previous page)

```
  torch::Tensor t4 = torch::tensor({1, 2, 3});
  torch::Tensor t5 = torch::tensor({4, 5, 6});
  TORCH_CHECK(torch::allclose(t3[0], t4));
  TORCH_CHECK(torch::allclose(t3[1], t5));
}
```

From ArrayRef

Listing 28: ./code/tensor-creation/main.cc

```
static void FromArrayRef() {
  int32_t i[] = {1, 2, 3};
  torch::ArrayRef<int32_t> a(i);
  torch::Tensor t = torch::tensor(a);
  // Data is copied to t

  TORCH_CHECK(t[0].item<int64_t>(), 1);
  TORCH_CHECK(t[1].item<int64_t>(), 2);
  TORCH_CHECK(t[2].item<int64_t>(), 3);
}
```

8.3.9 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 29: ./code/tensor/main.cc (Not recommended constructors)

```
static void TestCommonMethods() {
  torch::Tensor t = torch::rand({2, 3, 4});

  TORCH_CHECK(t.dim() == 3);              // 3-d tensor
  TORCH_CHECK(t.ndimension() == t.dim()); // same
  TORCH_CHECK(t.numel() == 2 * 3 * 4);
  TORCH_CHECK(t.is_contiguous() == true);
  TORCH_CHECK(t.contiguous().is_contiguous() == true);

  t.fill_(10); // fill all entries to 0
  t.zero_();   // zero out all entries

  t = t.to(torch::kInt);
  TORCH_CHECK(t.is_floating_point() == false);
  TORCH_CHECK(t.is_signed() == true);
```

(continues on next page)

(continued from previous page)

```
  TORCH_CHECK(t.size(0) == 2);
  TORCH_CHECK(t.size(1) == 3);
  TORCH_CHECK(t.size(2) == 4);
  TORCH_CHECK(t.sizes() == torch::ArrayRef<int64_t>({2, 3, 4}));

  t = t.contiguous();
  TORCH_CHECK(t.stride(0) == 3 * 4);
  TORCH_CHECK(t.stride(1) == 4);
  TORCH_CHECK(t.stride(2) == 1);
  TORCH_CHECK(t.strides() == torch::ArrayRef<int64_t>({12, 4, 1}));

  TORCH_CHECK(t.defined() == true);
  {
    torch::Tensor a;
    TORCH_CHECK(a.defined() == false);
    a = t;
    TORCH_CHECK(a.defined() == true);
    a.reset();
    TORCH_CHECK(a.defined() == false);
  }

  t = t.to(torch::kShort);
  TORCH_CHECK(t.itemsize() == sizeof(int16_t));
  TORCH_CHECK(t.nbytes() == t.numel() * t.itemsize());
  TORCH_CHECK(t.itemsize() == t.element_size()); // same

  TORCH_CHECK(t.scalar_type() == torch::kShort);
  TORCH_CHECK(t.dtype() == caffe2::TypeMeta::Make<int16_t>());
  TORCH_CHECK(t.dtype().toScalarType() == torch::kShort);

  TORCH_CHECK(t.device() == torch::Device("cpu"));
  TORCH_CHECK(t.device() == torch::Device(torch::kCPU));

  // Note: t.device() return an instance of torch::Device
  // t.get_device() returns the device index.
  TORCH_CHECK(t.get_device() == t.device().index());

  TORCH_CHECK(t.is_cpu() == true);
  TORCH_CHECK(t.is_cuda() == false);

  t = t.to(torch::kInt);
  int32_t *p = t.data_ptr<int32_t>();
  p[0] = 100;

  torch::TensorAccessor<int32_t, 3> acc = t.accessor<int32_t, 3>();
  TORCH_CHECK(acc[0][0][0] == p[0]);
  p[12] = -2;
  TORCH_CHECK(acc[1][0][0] == -2);

  acc[1][1][2] = 3;
  TORCH_CHECK(*(p + 12 + 4 + 2) == 3);
```

(continues on next page)

(continued from previous page)

```

  t = t.to(torch::kFloat);
  t.set_requires_grad(true);
  TORCH_CHECK(t.requires_grad() == true);

  t.set_requires_grad(false);
  TORCH_CHECK(t.requires_grad() == false);

  t = t.cuda();
  TORCH_CHECK(t.device().type() == torch::kCUDA);
  t = t.cpu();

  torch::TensorOptions opts = t.options();
  TORCH_CHECK(opts.device() == t.device());
}
```

slice

Listing 30: torch::slice

```
static void TestSlice() {
  auto t = torch::tensor({1, 2, 3, 4, 5}, torch::kInt);
  torch::TensorAccessor<int32_t, 1> acc = t.accessor<int32_t, 1>();

  // t2 = t[1:3]
  torch::Tensor t2 = t.slice(/*dim*/ 0, /*start*/ 1,
                             /*end, exclusive*/ 3); // memory is shared
  torch::TensorAccessor<int32_t, 1> acc2 = t2.accessor<int32_t, 1>();
  TORCH_CHECK(acc2[0] == 2);
  TORCH_CHECK(acc2[1] == 3);

  acc2[0] = 10; // also changes t since the memory is shared
  TORCH_CHECK(acc[1] == 10);
}
```

topk

Listing 31: torch::topk

```
// https://pytorch.org/docs/stable/generated/torch.topk.html
static void TestTopK() {
  auto t = torch::tensor({1, 0, 3, -1}, torch::kInt).to(torch::kFloat);
  torch::Tensor values, indexes;
  std::tie(values, indexes) =
      t.topk(/*k*/ 2, /*dim*/ 0, /*largest*/ true, /*sorted*/ true);
  auto values_acc = values.accessor<float, 1>();
  auto indexes_acc = indexes.accessor<int64_t, 1>(); // Note: it is int64_t

  TORCH_CHECK(values.numel() == 2); // k in topk is 2
```

(continues on next page)

(continued from previous page)

```
  TORCH_CHECK(values_acc[0] == 3);  // the largest value is 3, at t[2]
  TORCH_CHECK(values_acc[1] == 1);  // the second largest value is 1, at t[0]
                                    //
  TORCH_CHECK(indexes_acc[0] == 2); // the largest value is t[2]
  TORCH_CHECK(indexes_acc[1] == 0); // the second largest value is t[0]
}
```

floor_divide

Listing 32: torch::floor_divide

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

div

Listing 33: torch::div

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

remainder

Listing 34: torch::remainder

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

empty

Listing 35: torch::empty

```cpp
static void TestEmpty() {
  auto t = torch::empty({3}, torch::kInt);
  TORCH_CHECK(t.scalar_type() == torch::kInt);
  TORCH_CHECK(t.numel() == 3);
}
```

8.3.10 intrusive_ptr

8.3.11 optional

8.3.12 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 36: ./code/packed-sequence/main.cc

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

The output is

Listing 37: ./code/packed-sequence/main.cc

```
}
/*
data:   1   2   3
 -1   2   3
 10  20  30
  4   5   6
 -4   5   6
  7   8   9
```

(continues on next page)

(continued from previous page)

```
[ CPULongType{6,3} ]
batch_sizes:  3
 2
 1
[ CPULongType{3} ]
sorted_indices:  1
 2
```

8.3.13 ivalue

Listing 38: ./code/ivalue/main.cc

```cpp
#include "torch/script.h"

static void TestVectorOfTensor() {
  torch::jit::Module m("m");
  m.define(R"(
    def forward(self, x, y):
      return [x, y]
  )");
  auto x = torch::tensor({1, 2, 3});
  auto y = torch::tensor({4, 5, 6});
  auto i = m.run_method("forward", x, y);

  assert(i.tagKind() == "GenericList");

  torch::ArrayRef<torch::IValue> tensor_list = i.toListRef();
  TORCH_CHECK(torch::allclose(x, tensor_list[0].toTensor()));
  TORCH_CHECK(torch::allclose(y, tensor_list[1].toTensor()));

  torch::List<torch::IValue> k = i.toList();

  torch::List<torch::Tensor> o =
      c10::impl::toTypedList<torch::Tensor>(std::move(k));

  TORCH_CHECK(torch::allclose(o[0], x));
  TORCH_CHECK(torch::allclose(o[1], y));

  std::vector<torch::Tensor> p = o.vec();
  TORCH_CHECK(torch::allclose(p[0], x));
  TORCH_CHECK(torch::allclose(p[1], y));
}

static void TestVectorOfTensor2() {
  torch::jit::Module m("m");
  m.define(R"(
    def forward(self, x):
      return [[x], [x,x]]
  )");
  auto x = torch::tensor({1, 2, 3});
```

(continues on next page)

(continued from previous page)

```
  auto i = m.run_method("forward", x);
  TORCH_CHECK(i.tagKind() == "GenericList");

  torch::List<torch::IValue> list = i.toList();
  torch::Tensor a = list.get(0).toListRef()[0].toTensor();
  TORCH_CHECK(torch::allclose(a, x));

  std::vector<torch::Tensor> b =
      c10::impl::toTypedList<torch::Tensor>(list.get(1).toList()).vec();
  TORCH_CHECK(torch::allclose(b[0], x));
  TORCH_CHECK(torch::allclose(b[1], x));
}

static void TestVectorOfTensor3() {
  torch::jit::Module m("m");
  m.define(R"(
    def forward(self, x: List[torch.Tensor]):
      return x[0] + x[1]
  )");

  std::vector<torch::Tensor> v;
  v.push_back(torch::tensor({1, 2}));
  v.push_back(torch::tensor({3, 4}));
  c10::List<torch::Tensor> ilist(v);

  c10::impl::GenericList generic_list = c10::impl::toList(ilist);

  c10::List<torch::Tensor> l2 =
      c10::impl::toTypedList<torch::Tensor>(generic_list);

  TORCH_CHECK(torch::allclose(l2[0], v[0]));
  TORCH_CHECK(torch::allclose(l2[1], v[1]));

  auto r = m.run_method("forward", generic_list);
  TORCH_CHECK(torch::allclose(r.toTensor(), v[0] + v[1]));

  // Note: We can pass a vector directly
  r = m.run_method("forward", v);
  TORCH_CHECK(torch::allclose(r.toTensor(), v[0] + v[1]));

  r = m.run_method("forward", ilist); // also OK
  TORCH_CHECK(torch::allclose(r.toTensor(), v[0] + v[1]));
}

static void TestVectorOfTensor4() {
  torch::jit::Module m("m");
  m.define(R"(
    def forward(self, x: Tuple[List[torch.Tensor]]):
      return x[0][0] + x[0][1]
  )");

  std::vector<torch::Tensor> v;
```

(continues on next page)

(continued from previous page)

```
  v.push_back(torch::tensor({1, 2}));
  v.push_back(torch::tensor({3, 4}));
  auto t = torch::ivalue::Tuple::create(v);

  auto r = m.run_method("forward", t);
  TORCH_CHECK(torch::allclose(r.toTensor(), v[0] + v[1]));
}

static void TestVectorOfTensor5() {
  torch::jit::Module m("m");
  m.define(R"(
    def forward(self, x: Tuple[List[List[torch.Tensor]], List[torch.Tensor]]):
      return x[0][0][0] + x[0][0][1] + x[1][0] + x[1][1]
  )");

  std::vector<torch::Tensor> v;
  v.push_back(torch::tensor({1, 2}));
  v.push_back(torch::tensor({3, 4}));

  std::vector<std::vector<torch::Tensor>> vv;
  vv.push_back(v);
  vv.push_back(v);

  auto t = torch::ivalue::Tuple::create(vv, v);

  auto r = m.run_method("forward", t);
  TORCH_CHECK(torch::allclose(r.toTensor(), v[0] + v[1] + v[0] + v[1]));
}

int main() {
  TestVectorOfTensor();
  TestVectorOfTensor2();
  TestVectorOfTensor3();
  TestVectorOfTensor4();
  TestVectorOfTensor5();
  return 0;
}
```

8.4 Logical operations

Listing 39: ./code/logical-op.py

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

import torch

a = torch.tensor([float("inf")])
b = torch.tensor([float("nan")])
assert torch.isinf(a).item() is True
assert torch.isnan(a).item() is False
```

(continues on next page)

(continued from previous page)

```

assert torch.isinf(b).item() is False
assert torch.isnan(b).item() is True

assert torch.logical_or(torch.isinf(a), torch.isnan(b)).item() is True

assert a.isinf().item() is True
assert a.isnan().item() is False

assert b.isinf().item() is False
assert b.isnan().item() is True
```

8.5 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()
```

8.6 Quantization

8.6.1 Internals

https://github.com/pytorch/pytorch/blob/master/aten/src/ATen/core/QuantizerBase.h defines the base class `Quantizer`.

https://github.com/pytorch/pytorch/blob/master/aten/src/ATen/quantized/Quantizer.h defines the subclasses of `Quantizer`, such as

- `PerTensorAffineQuantizer` - qscheme is `kPerTensorAffine`.

QScheme

See https://github.com/pytorch/pytorch/blob/master/c10/core/QScheme.h

Listing 40: ./code/qscheme/main.cc

```
#include "torch/script.h"

static void TestQScheme() {
  TORCH_CHECK(torch::toString(torch::kPerTensorAffine) == "per_tensor_affine");

  TORCH_CHECK(torch::toString(torch::kPerChannelAffine) ==
              "per_channel_affine");
```

(continues on next page)

(continued from previous page)

```
  TORCH_CHECK(torch::toString(torch::kPerTensorSymmetric) ==
              "per_tensor_symmetric");

  TORCH_CHECK(torch::toString(torch::kPerChannelSymmetric) ==
              "per_channel_symmetric");

  TORCH_CHECK(torch::toString(torch::kPerChannelAffineFloatQParams) ==
              "per_channel_affine_float_qparams");
}

int main() {
  TestQScheme();
  return 0;
}
```

PerTensorAffineQuantizer

It has 4 important methods:

- `QScheme qscheme() const`, always returns `kPerTensorAffine`.
- `double scale() const`
- `int64_t zero_point() const`
- `ScalarType scalar_type() const`

It uses quantize_tensor_per_tensor_affine_cpu when FBGEMM is available.

Otherwise, it uses quantize_tensor_per_tensor_affine_cpu.

- For arm, it uses quantize_tensor_arm. It is a template with many specializations.
- For x86, it uses `quantize_val`
  - If FBGEMM is available, it uses quantize_val
  - Otherwise, it uses quantize_val

```
template <typename T>
T quantize_val(double scale, int64_t zero_point, float value) {
  // std::nearbyint results in nearest integer value according to the current
  // rounding mode and the default rounding mode is rounds to even in half-way
  // cases in most popular processor architectures like x86 and ARM. This is
  // typically faster than an alternatives like std::round that rounds half-way
  // cases away from zero, and can be consistent with SIMD implementations for
  // example in x86 using _mm512_cvtps_epi32 or mm512_round_ps with
  // _MM_FROUND_CUR_DIRECTION option that also follow the current rounding mode.
  int64_t qvalue;
  constexpr int64_t qmin = std::numeric_limits<typename T::underlying>::min();
  constexpr int64_t qmax = std::numeric_limits<typename T::underlying>::max();
  float inv_scale = 1.0f / static_cast<float>(scale);
  qvalue = static_cast<int64_t>(zero_point + Round(value * inv_scale));
  qvalue = std::max<int64_t>(qvalue, qmin);
  qvalue = std::min<int64_t>(qvalue, qmax);
```

(continues on next page)

(continued from previous page)

```
  return static_cast<T>(qvalue);
}
```

dequantize_val is defined as:

```
template <typename T>
TORCH_API float dequantize_val(double scale, int64_t zero_point, T value) {
  return static_cast<float>(scale) * (value.val_ - static_cast<int32_t>(zero_point));
}
```

8.6.2 torch.quantize_per_tensor

See https://pytorch.org/docs/stable/generated/torch.quantize_per_tensor.html

```
def test_quantize_per_tensor():
    scale = 0.1
    zero_point = 1
    a = torch.tensor([10.0, 2.0], dtype=torch.float32)
    q = torch.quantize_per_tensor(
        input=a,
        scale=scale,
        zero_point=zero_point,
        dtype=torch.qint8,
    )
    assert isinstance(q, torch.Tensor)
    assert q.is_quantized is True

    assert q.q_scale() == scale
    assert q.q_zero_point() == zero_point
    assert str(q.qscheme()) == "torch.per_tensor_affine"
    assert q.dtype == torch.qint8

    c = q.int_repr()
    assert c[0] == a[0] / scale + zero_point
    assert c[1] == a[1] / scale + zero_point
    assert torch.all(torch.eq(c, torch.tensor([101, 21], dtype=torch.int8)))
    d = q.dequantize()
    assert d.dtype == torch.float32
    assert torch.all(torch.eq(d, a))

    f = torch.dequantize(q)
    assert torch.all(torch.eq(f, a))
    #  print(q)
    """
    tensor([10.,  2.], size=(2,), dtype=torch.qint8,
           quantization_scheme=torch.per_tensor_affine, scale=0.1, zero_point=1)
    """
    assert q[0].item() == 10  # q[0].item() will dequantize() to a float
    assert q[1].item() == 2
    print(type(q[0].item()))
    q[0] = 2.5  # Note: it will quantize 2.5 and store it in q
```

(continues on next page)

(continued from previous page)

```
    print(q.int_repr())
    """
    tensor([26, 21], dtype=torch.int8)
    """
```

Compress ration

```
def test_size():
    r = torch.rand(100, 100, dtype=torch.float32)
    q = torch.quantize_per_tensor(r, scale=0.1, zero_point=0, dtype=torch.qint8)
    torch.save(r, "float32.pt")
    torch.save(q, "int8.pt")
    float_size = os.path.getsize("float32.pt")
    int8_size = os.path.getsize("int8.pt")
    print("float_size:", float_size)
    print("int8_size:", int8_size)
    print(f"ratio: {float_size}/{int8_size}: {float_size/int8_size:.3f}")
    os.remove("float32.pt")
    os.remove("int8.pt")
    """
    float_size: 40747
    int8_size: 10795
    ratio: 40747/10795: 3.775
    """
```

8.6.3 quantize_per_tensor_dynamic

Listing 41: ./code/quantize_per_tensor_dynamic/main.cc

```
// #include "ATen/native/quantized/cpu/QuantUtils.h" // for the latest pytorch

#include "ATen/native/quantized/cpu/quant_utils.h" // for torch 1.10
#include "ATen/ops/quantize_per_tensor_dynamic.h"  // needs torch>=1.11
#include "torch/script.h"

// See
// https://github.com/pytorch/pytorch/blob/master/aten/src/ATen/native/quantized/cpu/
↪QuantUtils.h#L59
static void TestChooseQuantizationParams() {
  quant_utils::TensorQuantizationParams p;
  p = quant_utils::ChooseQuantizationParams(-1 /*min*/, 2 /*max*/,
                                            -128 /*qmin*/, 127 /*qmax*/);
  std::cout << "zero_point: " << p.zero_point << "\n";
  std::cout << "scale: " << p.scale << "\n";
  /**
   * scale = (max - min) / (qmax - qmin) = 3 / 255 = 0.0117647
   * zero_point_min = qmin - min/scale = -128 - (-1)/scale = -43
   * zero_point_max = qmax - max/scale = 127 - 2/scale = -43
   *
```

(continues on next page)

(continued from previous page)

```
   * min_error = abs(qmin) -abs(min/scale) = 128 - 1/scale = 43
   * max_error = abs(qmax) - abs(max/scale) = 127 - 2/scale = -43
   *
   * zero_point =  (min_error < max_error) ? zero_point_min : zero_point_max
   */
}

// See
// https://github.com/pytorch/pytorch/blob/master/aten/src/ATen/native/quantized/QTensor.cpp#L14
// and
static void TestQuantizePerTensorDynamic() {
  torch::Tensor r = torch::tensor({-1, 0, 2}, torch::kFloat32);
  torch::Tensor q = torch::quantize_per_tensor_dynamic(r, torch::kQInt8, false);
  std::cout << "q: " << q << "\n";
#if 0
q: -1
 0
 2
[ QuantizedCPUQInt8Type{3}, qscheme: per_tensor_affine, scale: 0.0117647, zero_point: -43 ]
#endif
  std::cout << "q.int_repr(): " << q.int_repr() << "\n";
#if 0
q.int_repr(): -128
 -43
 127
[ CPUCharType{3} ]
#endif
}

int main() {
  TestChooseQuantizationParams();
  TestQuantizePerTensorDynamic();
  return 0;
}
```

8.6.4 torch.quantize_per_channel

See `<https://pytorch.org/docs/stable/generated/torch.quantize_per_channel.html#torch.quantize_per_channel>`_k

```
def test_quantize_per_channel_2d():
    # (N, C)
    a = torch.tensor(
        [
            [1, 2, 3],
            [4, 5, 6],
        ],
        dtype=torch.float32,
    )
    assert a.shape == (2, 3)
```

(continues on next page)

(continued from previous page)

```
    scales = torch.tensor([0.125, 0.25, 0.5])

    # It will be converted to torch.int64 internally
    zero_points = torch.tensor([10, 20, 30], dtype=torch.int32)
    q = torch.quantize_per_channel(
        input=a,
        scales=scales,
        zero_points=zero_points,
        axis=1,
        dtype=torch.qint8,
    )
    assert q.dtype == torch.qint8

    assert q.q_per_channel_scales().dtype == torch.float64
    assert torch.all(torch.eq(q.q_per_channel_scales(), scales))

    assert q.q_per_channel_zero_points().dtype == torch.int64
    assert torch.all(torch.eq(q.q_per_channel_zero_points(), zero_points))

    assert str(q.qscheme()) == "torch.per_channel_affine"

    assert q.q_per_channel_axis() == 1

    i = q.int_repr()
    expected_i = torch.tensor([[18, 28, 36], [42, 40, 42]], dtype=torch.int8)
    assert i.dtype == torch.int8
    assert torch.all(torch.eq(i, expected_i))

    assert i[0][0].item() == a[0][0].item() / scales[0] + zero_points[0]
    assert i[0][1].item() == a[0][1].item() / scales[1] + zero_points[1]
    assert i[0][2].item() == a[0][2].item() / scales[2] + zero_points[2]

    assert i[1][0].item() == a[1][0].item() / scales[0] + zero_points[0]
    assert i[1][1].item() == a[1][1].item() / scales[1] + zero_points[1]
    assert i[1][2].item() == a[1][2].item() / scales[2] + zero_points[2]

    d = q.dequantize()
    assert torch.all(torch.eq(d, a))

    f = torch.dequantize(q)
    assert torch.all(torch.eq(f, a))

    #  print(q)
    """
    tensor([[1., 2., 3.],
            [4., 5., 6.]], size=(2, 3), dtype=torch.qint8,
           quantization_scheme=torch.per_channel_affine,
           scale=tensor([0.1250, 0.2500, 0.5000], dtype=torch.float64),
           zero_point=tensor([10, 20, 30]), axis=1)
    """
```

8.6.5 Observer

Listing 42: ./code/observer/ex0.py

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

import torch
from torch.ao.quantization.observer import _with_args, MinMaxObserver


class Foo:
    def __init__(self, a=1, b=2):
        self.a = a
        self.b = b


def test_with_args():
    Foo.with_args = classmethod(_with_args)
    foo_builder = Foo.with_args(a=3).with_args(b=4).with_args(a=10)
    f = foo_builder()
    assert f.a == 10  # the last a=10 replaces the first a=3
    assert f.b == 4

    f2 = foo_builder()
    assert id(f) != id(f2)


def test_min_max_observer():
    ob = MinMaxObserver(dtype=torch.qint8)
    print(ob)  # MinMaxObserver(min_val=inf, max_val=-inf)

    ob(torch.tensor([1, 2, 3]))
    print(ob)  # MinMaxObserver(min_val=1.0, max_val=3.0)

    ob(torch.tensor([-1, 30]))
    print(ob)  # MinMaxObserver(min_val=-1.0, max_val=30.0)
    scale, zero_point = ob.calculate_qparams()
    print("scale", scale)  # scale tensor([0.1216])
    print("zero_point", zero_point)  # zero_point tensor([-120], dtype=torch.int32)


def main():
    test_with_args()
    test_min_max_observer()


if __name__ == "__main__":
    main()
```

8.6.6 Hello

Listing 43: ./code/ex1.py

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

import torch
import torch.nn as nn


class Model(torch.nn.Module):
    def __init__(self):
        super().__init__()
        self.fc = nn.Linear(1, 1)

    def forward(self, x):
        x = self.fc(x)
        return x


def main():
    m = Model()
    model_int8 = torch.quantization.quantize_dynamic(
        model=m,
        qconfig_spec={torch.nn.Linear},
        dtype=torch.qint8,
    )
    print(model_int8)
    print(model_int8.fc)
    assert model_int8.fc.weight().is_quantized
    assert model_int8.fc.weight().dtype == torch.qint8

    assert model_int8.fc.bias().is_quantized is False
    assert model_int8.fc.bias().dtype == torch.float32

    x = torch.tensor([[1.0]], dtype=torch.float32)
    y = m(x)
    print(x, y)  # tensor([[1.]]) tensor([[-1.2900]], grad_fn=<AddmmBackward0>)

    qy = model_int8(x)
    print(qy)  # tensor([[-1.2931]])


if __name__ == "__main__":
    torch.manual_seed(20220723)
    main()
```

8.6.7 References

The main implementation is in

https://github.com/pytorch/pytorch/tree/master/aten/src/ATen/native/quantized

- Introducing Quantized Tensor

  https://github.com/pytorch/pytorch/wiki/Introducing-Quantized-Tensor

- Model Quantization for PyTorch (Proposal) #18318

  https://github.com/pytorch/pytorch/issues/18318

- torch_quantization_design_proposal

  https://github.com/pytorch/pytorch/wiki/torch_quantization_design_proposal

Links

- https://github.com/pytorch/pytorch/blob/master/test/quantization/core/test_quantized_tensor.py
- https://github.com/pytorch/pytorch/blob/master/test/quantization/core/experimental/test_quantized_tensor.py
- https://github.com/pytorch/pytorch/blob/master/aten/src/ATen/test/quantized_test.cpp
- https://github.com/pytorch/pytorch/blob/master/aten/src/ATen/native/quantized/README.md
- https://pytorch.org/blog/introduction-to-quantization-on-pytorch/
- https://pytorch.org/docs/stable/quantization.html
- Deep Dive on PyTorch Quantization - Chris Gottbrath

  YouTube: https://www.youtube.com/watch?v=c3MT2qV5f9w

8.7 android

8.7.1 References

- https://zhuanlan.zhihu.com/p/54665674
- Pytorch model to Caffe & ncnn

  https://github.com/starimeL/PytorchConverter

8.8 onnx

8.8.1 Install

```
pip install onnx onnxruntime
pip install netron # for visualization
# Or go to https://netron.app/
```

API references

- https://github.com/onnx/onnx/blob/main/docs/PythonAPIOverview.md
- https://onnxruntime.ai/docs/api/python/api_summary.html#inferencesession

8.8.2 Hello

Listing 44: ./code/hello/ex0.py

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

import torch
import torch.nn as nn


class Foo(nn.Module):
    def __init__(self, i):
        super().__init__()
        self.relu = nn.ReLU()
        self.i = 1

    def forward(self, x):
        if x.sum().item() > 0:
            return self.relu(x + 1)
        else:
            return self.relu(x + 2)


def main():
    f = Foo(1)
    f.eval()  # f.train(False)
    f = torch.jit.script(f)

    x = torch.rand(2, 3, 4)
    # [N, T, C]
    torch.onnx.export(
        f,
        x,
        "f.onnx",
        verbose=False,
        input_names=["x"],
        output_names=["y"],
        dynamic_axes={"x": {0: "batch_size", 1: "T"}, "y": [0, 1]},
        #  dynamic_axes={"x": [0, 1], "y": [0, 1]},
    )


if __name__ == "__main__":
    main()
```

Listing 45: ./code/hello/ex0-1.py

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

import onnx


def main():
    model = onnx.load("f.onnx")
    #  print(model)
    # Check that the model is well formed
    onnx.checker.check_model(model)
    # Print a human readable representation of the graph
    print(onnx.helper.printable_graph(model.graph))
    onnx.save(model, "f2.onnx")


if __name__ == "__main__":
    main()
```

Listing 46: ./code/hello/ex0-2.py

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

import onnxruntime as ort
import numpy as np


def main():
    # https://github.com/microsoft/onnxruntime/issues/10113
    options = ort.SessionOptions()
    options.inter_op_num_threads = 1
    options.intra_op_num_threads = 1

    ort_session = ort.InferenceSession("f.onnx", sess_options=options)

    x = np.arange(24).reshape(2, 3, 4).astype(np.float32)
    ortvalue = ort.OrtValue.ortvalue_from_numpy(x)
    assert ortvalue.device_name() == "cpu"
    assert list(ortvalue.shape()) == list(x.shape)
    assert ortvalue.data_type() == "tensor(float)"
    assert ortvalue.is_tensor() is True

    results = ort_session.run(["y"], {"x": ortvalue})
    print(results)

    ort_inputs = {ort_session.get_inputs()[0].name: x}
    results = ort_session.run(["y"], ort_inputs)
    print(results)

    results = ort_session.run(["y"], {"x": x})
    print(results)
```

(continues on next page)

(continued from previous page)

```

    # https://onnxruntime.ai/docs/api/python/api_summary.html#onnxruntime.NodeArg
    inputs = ort_session.get_inputs()
    assert isinstance(inputs, list)
    assert len(inputs) == 1
    assert isinstance(inputs[0], ort.NodeArg)
    print(inputs[0].name, inputs[0].type, inputs[0].shape)
    assert inputs[0].name == "x"
    assert inputs[0].type == "tensor(float)"
    assert inputs[0].shape == ["batch_size", "T", 4]

    outputs = ort_session.get_outputs()
    assert isinstance(outputs, list)
    assert isinstance(outputs[0], ort.NodeArg)
    assert len(outputs) == 1
    assert outputs[0].name == "y"
    assert outputs[0].type == "tensor(float)"
    assert outputs[0].shape == ["y_dynamic_axes_1", "y_dynamic_axes_2", 4]


if __name__ == "__main__":
    main()
```

8.8.3 References

- (OPTIONAL) EXPORTING A MODEL FROM PYTORCH TO ONNX AND RUNNING IT USING ONNX RUNTIME

  https://pytorch.org/tutorials/advanced/super_resolution_with_onnxruntime.html

CHAPTER

NINE

PYTHON

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

```
import asyncio


async def hello():
    print("hello world")


asyncio.run(hello())
```

Exercise 2

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

```
import asyncio
import time


loop = asyncio.get_event_loop()


@asyncio.coroutine
def hello():
```

(continues on next page)

(continued from previous page)

```
    print(f"hello {time.strftime('%X')}")
    yield from asyncio.sleep(1)
    print(f"world {time.strftime('%X')}")


if __name__ == "__main__":
    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_OAlIhXziw&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

```
print(bin(1))  # 0b1
print(bin(3))  # 0b11
print(bin(255))  # 0b11111111
print(bin(256))  # 0b100000000
assert isinstance(bin(1), str)
assert int("11", base=2) == 3
assert int("0b11", base=0) == 3
assert hex(2) == "0x2"
assert hex(10) == "0xa"

assert oct(10) == "0o12"
assert int("12", base=8) == 10
assert int("0o12", base=0) == 10

assert 1_000 == 1000
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

```
a = 1
b = 2
c = 3
assert "{}".format(a) == "1"
assert "{}".format(b) == "2"
assert "{0} {1} {foo}".format(a, b, foo=c) == "1 2 3"

# 1 - the first positional argument (counting from 0)
# foo - it is a keyword argument
# 0 - the zeros positional
assert "{1} {foo} {0}".format(a, b, foo=c) == "2 3 1"

assert "{0} {1} {0} {0}".format(a, b) == "1 2 1 1"

assert "skip braces {0} {{}}".format(a) == "skip braces 1 {}"
print("{}")  # {}
try:
    print("{} {}".format(a))
except IndexError as e:
    assert str(e) == "Replacement index 1 out of range for positional args tuple"

assert "{0:2}".format(a) == " 1"
assert "{0:02}".format(a) == "01"
assert "{0:03}".format(a) == "001"
assert "{0:1}".format(-1) == "-1"
assert "{0:2}".format(-1) == "-1"
assert "{0:3}".format(-1) == " -1"
assert "{0:03}".format(-1) == "-01"

assert "{0:.2f}".format(0.5) == "0.50"
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

```
from enum import Enum


class Color(Enum):
    RED = 1
    GREEN = 2
    BLUE = 3
    #  BLUE = 4 # TypeError: Attempted to reuse key: 'BLUE'
    ALIAS_FOR_RED = 1  # Use @unique to disallow this
    MAX_COLOR = 4  # Note the naming convention


assert isinstance(Color.RED, Color)

assert str(Color(1)) == "Color.RED"
assert str(repr(Color(1))) == "<Color.RED: 1>"

assert Color.RED.name == "RED"
assert Color.BLUE.value == 3

print(list(Color))
print(type(list(Color)[0]))
for c in Color:
    print(c, type(c))

"""
[<Color.RED: 1>, <Color.GREEN: 2>, <Color.BLUE: 3>, <Color.MAX_COLOR: 4>]
<enum 'Color'>
Color.RED <enum 'Color'>
Color.GREEN <enum 'Color'>
Color.BLUE <enum 'Color'>
Color.MAX_COLOR <enum 'Color'>
"""

assert Color(1) == Color.RED
assert Color["RED"] == Color.RED
assert Color["ALIAS_FOR_RED"] == Color.RED

```

(continues on next page)

(continued from previous page)

```
print(Color.__members__)
"""
{'RED': <Color.RED: 1>, 'GREEN': <Color.GREEN: 2>, 'BLUE': <Color.BLUE: 3>, 'ALIAS_FOR_RED':
↪<Color.RED: 1>, 'MAX_COLOR': <Color.MAX_COLOR: 4>}
"""
```

Flag

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

```
from enum import Flag


class Weekday(Flag):
    MONDAY = 1 << 0
    TUESDAY = 1 << 1
    WEDNESDAY = 1 << 2
    THURSDAY = 1 << 3
    FRIDAY = 1 << 4
    SATURDAY = 1 << 5
    SUNDAY = 1 << 6


assert Weekday.MONDAY.value == 1
assert Weekday.TUESDAY.value == 2
assert Weekday.WEDNESDAY.value == 4
assert Weekday.THURSDAY.value == 8
assert Weekday.FRIDAY.value == 16
assert Weekday.SATURDAY.value == 32
assert Weekday.SUNDAY.value == 64

weekend = Weekday.SATURDAY | Weekday.SUNDAY
print(weekend)  # Weekday.SUNDAY|SATURDAY
print(repr(weekend))  # <Weekday.SUNDAY|SATURDAY: 96>
assert Weekday.SATURDAY in weekend
assert Weekday.SUNDAY in weekend
assert Weekday.MONDAY not in weekend
```

auto

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

```
from enum import Enum, Flag, auto


class Weekday(Flag):
    MONDAY = auto()  # start from 1
    TUESDAY = auto()
    WEDNESDAY = auto()
```

(continues on next page)

(continued from previous page)

```
    THURSDAY = auto()
    FRIDAY = auto()
    SATURDAY = 128
    SUNDAY = auto()


assert Weekday.MONDAY.value == 1
assert Weekday.TUESDAY.value == 2
assert Weekday.WEDNESDAY.value == 4
assert Weekday.THURSDAY.value == 8
assert Weekday.FRIDAY.value == 16
assert Weekday.SATURDAY.value == 128
assert Weekday.SUNDAY.value == 256


class Color(Enum):
    RED = auto()  # start from 1
    GREEN = auto()
    BLUE = auto()
    YELLOW = 10
    WHITE = auto()


assert Color.RED.value == 1
assert Color.GREEN.value == 2
assert Color.BLUE.value == 3
assert Color.YELLOW.value == 10
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

```
import socket

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

(continues on next page)

(continued from previous page)

```
<AddressFamily.AF_ECONET: 19>, <AddressFamily.AF_ATMSVC: 20>,
<AddressFamily.AF_RDS: 21>, <AddressFamily.AF_SNA: 22>,
<AddressFamily.AF_IRDA: 23>, <AddressFamily.AF_PPPOX: 24>,
<AddressFamily.AF_WANPIPE: 25>, <AddressFamily.AF_LLC: 26>,
<AddressFamily.AF_CAN: 29>, <AddressFamily.AF_TIPC: 30>,
<AddressFamily.AF_BLUETOOTH: 31>, <AddressFamily.AF_ALG: 38>,
<AddressFamily.AF_VSOCK: 40>, <AddressFamily.AF_QIPCRTR: 42>]
"""

assert socket.AF_UNIX == socket.AddressFamily.AF_UNIX
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

```
import socket

print(list(socket.SocketKind))
"""
[<SocketKind.SOCK_STREAM: 1>, <SocketKind.SOCK_DGRAM: 2>,
<SocketKind.SOCK_RAW: 3>, <SocketKind.SOCK_RDM: 4>,
<SocketKind.SOCK_SEQPACKET: 5>, <SocketKind.SOCK_NONBLOCK: 2048>,
<SocketKind.SOCK_CLOEXEC: 524288>]
"""

assert socket.SOCK_STREAM == socket.SocketKind.SOCK_STREAM
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

```
// https://github.com/lattera/glibc/blob/master/bits/sockaddr.h
/* POSIX.1g specifies this type name for the `sa_family' member.  */
typedef unsigned short int sa_family_t;

#define __SOCKADDR_COMMON(sa_prefix) sa_family_t sa_prefix##family

// https://github.com/lattera/glibc/blob/master/bits/socket.h

struct sockaddr {
  __SOCKADDR_COMMON(sa_); /* Common data: address family and length.  */
```

(continues on next page)

(continued from previous page)

```
  char sa_data[14];       /* Address data.  */
};

// https://github.com/lattera/glibc/blob/master/inet/netinet/in.h
struct sockaddr_in {
  __SOCKADDR_COMMON(sin_);
  in_port_t sin_port;      /* Port number.  */
  struct in_addr sin_addr; /* Internet address.  */

  /* Pad to size of `struct sockaddr'.  */
  unsigned char sin_zero[sizeof(struct sockaddr) - __SOCKADDR_COMMON_SIZE -
                         sizeof(in_port_t) - sizeof(struct in_addr)];
};

typedef uint32_t in_addr_t;
struct in_addr {
  in_addr_t s_addr;
};

/* Address to accept any incoming messages.  */
#define INADDR_ANY ((in_addr_t)0x00000000)
/* Address to send to all hosts.  */
#define INADDR_BROADCAST ((in_addr_t)0xffffffff)
/* Address indicating an error return.  */
#define INADDR_NONE ((in_addr_t)0xffffffff)

/* Network number for local host loopback.  */
#define IN_LOOPBACKNET 127
/* Address to loopback in software to local host.  */
#ifndef INADDR_LOOPBACK
#define INADDR_LOOPBACK ((in_addr_t)0x7f000001) /* Inet 127.0.0.1.  */
#endif
```

9.8.4 AddressInfo

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

```python
import socket

print(list(socket.AddressInfo))
"""
[<AddressInfo.AI_PASSIVE: 1>, <AddressInfo.AI_CANONNAME: 2>,
<AddressInfo.AI_NUMERICHOST: 4>, <AddressInfo.AI_V4MAPPED: 8>,
<AddressInfo.AI_ALL: 16>, <AddressInfo.AI_ADDRCONFIG: 32>,
<AddressInfo.AI_NUMERICSERV: 1024>]
"""
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

```c
#include <arpa/inet.h>
#include <stdio.h>

int main() {
  struct in_addr addr;
  int res = inet_pton(AF_INET, "192.168.1.2", &addr);
  printf("%08x\n", addr.s_addr);
  printf("192: %x\n", 192);
  printf("168: %x\n", 168);
  printf("1: %x\n", 1);
  printf("2: %x\n", 2);
  return 0;
}
#if 0
./inet_pton
0201a8c0
192: c0
168: a8
1: 1
2: 2
#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

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

(continues on next page)

(continued from previous page)

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

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

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

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

#include <arpa/inet.h>
#include <arpa/nameser.h>
#include <ctype.h>
#include <errno.h>
#include <netinet/in.h>
#include <resolv/resolv-internal.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>

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

int
__inet_pton_length (int af, const char *src, size_t srclen, void *dst)
{
  switch (af)
    {
    case AF_INET:
      return inet_pton4 (src, src + srclen, dst);
```

(continues on next page)

(continued from previous page)

```
    case AF_INET6:
      return inet_pton6 (src, src + srclen, dst);
    default:
      __set_errno (EAFNOSUPPORT);
      return -1;
    }
}
libc_hidden_def (__inet_pton_length)

/* Like __inet_pton_length, but use strlen (SRC) as the length of
   SRC.  */
int
__inet_pton (int af, const char *src, void *dst)
{
  return __inet_pton_length (af, src, strlen (src), dst);
}
libc_hidden_def (__inet_pton)
weak_alias (__inet_pton, inet_pton)
libc_hidden_weak (inet_pton)

/* Like inet_aton but without all the hexadecimal, octal and shorthand
   (and trailing garbage is not ignored).  Return 1 if SRC is a valid
   dotted quad, else 0.  This function does not touch DST unless it's
   returning 1.
   Author: Paul Vixie, 1996.  */
static int
inet_pton4 (const char *src, const char *end, unsigned char *dst)
{
  int saw_digit, octets, ch;
  unsigned char tmp[NS_INADDRSZ], *tp;

  saw_digit = 0;
  octets = 0;
  *(tp = tmp) = 0;
  while (src < end)
    {
      ch = *src++;
      if (ch >= '0' && ch <= '9')
        {
          unsigned int new = *tp * 10 + (ch - '0');

          if (saw_digit && *tp == 0)
            return 0;
          if (new > 255)
            return 0;
          *tp = new;
          if (! saw_digit)
            {
              if (++octets > 4)
                return 0;
              saw_digit = 1;
            }
```

(continues on next page)

(continued from previous page)

```
        }
      else if (ch == '.' && saw_digit)
        {
          if (octets == 4)
            return 0;
          *++tp = 0;
          saw_digit = 0;
        }
      else
        return 0;
    }
  if (octets < 4)
    return 0;
  memcpy (dst, tmp, NS_INADDRSZ);
  return 1;
}

/* Return the value of CH as a hexademical digit, or -1 if it is a
   different type of character.  */
static int
hex_digit_value (char ch)
{
  if ('0' <= ch && ch <= '9')
    return ch - '0';
  if ('a' <= ch && ch <= 'f')
    return ch - 'a' + 10;
  if ('A' <= ch && ch <= 'F')
    return ch - 'A' + 10;
  return -1;
}

/* Convert presentation-level IPv6 address to network order binary
   form.  Return 1 if SRC is a valid [RFC1884 2.2] address, else 0.
   This function does not touch DST unless it's returning 1.
   Author: Paul Vixie, 1996.  Inspired by Mark Andrews.  */
static int
inet_pton6 (const char *src, const char *src_endp, unsigned char *dst)
{
  unsigned char tmp[NS_IN6ADDRSZ], *tp, *endp, *colonp;
  const char *curtok;
  int ch;
  size_t xdigits_seen;        /* Number of hex digits since colon.  */
  unsigned int val;

  tp = memset (tmp, '\0', NS_IN6ADDRSZ);
  endp = tp + NS_IN6ADDRSZ;
  colonp = NULL;

  /* Leading :: requires some special handling.  */
  if (src == src_endp)
    return 0;
  if (*src == ':')
```

(continues on next page)

(continued from previous page)

```
    {
      ++src;
      if (src == src_endp || *src != ':')
        return 0;
    }

  curtok = src;
  xdigits_seen = 0;
  val = 0;
  while (src < src_endp)
    {
      ch = *src++;
      int digit = hex_digit_value (ch);
      if (digit >= 0)
        {
          if (xdigits_seen == 4)
            return 0;
          val <<= 4;
          val |= digit;
          if (val > 0xffff)
            return 0;
          ++xdigits_seen;
          continue;
        }
      if (ch == ':')
        {
          curtok = src;
          if (xdigits_seen == 0)
            {
              if (colonp)
                return 0;
              colonp = tp;
              continue;
            }
          else if (src == src_endp)
            return 0;
          if (tp + NS_INT16SZ > endp)
            return 0;
          *tp++ = (unsigned char) (val >> 8) & 0xff;
          *tp++ = (unsigned char) val & 0xff;
          xdigits_seen = 0;
          val = 0;
          continue;
        }
      if (ch == '.' && ((tp + NS_INADDRSZ) <= endp)
          && inet_pton4 (curtok, src_endp, tp) > 0)
        {
          tp += NS_INADDRSZ;
          xdigits_seen = 0;
          break;  /* '\0' was seen by inet_pton4.  */
        }
      return 0;
```

(continues on next page)

(continued from previous page)

```
    }
  if (xdigits_seen > 0)
    {
      if (tp + NS_INT16SZ > endp)
        return 0;
      *tp++ = (unsigned char) (val >> 8) & 0xff;
      *tp++ = (unsigned char) val & 0xff;
    }
  if (colonp != NULL)
    {
      /* Replace :: with zeros.  */
      if (tp == endp)
        /* :: would expand to a zero-width field.  */
        return 0;
      size_t n = tp - colonp;
      memmove (endp - n, colonp, n);
      memset (colonp, 0, endp - n - colonp);
      tp = endp;
    }
  if (tp != endp)
    return 0;
  memcpy (dst, tmp, NS_IN6ADDRSZ);
  return 1;
}
```

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

```
#include <arpa/inet.h>
#include <stdio.h>

int main() {
  struct in_addr addr;
  uint8_t *p = (uint8_t *)&addr.s_addr;
  p[0] = 192;
  p[1] = 168;
  p[2] = 1;
  p[3] = 2;
  char buf[INET_ADDRSTRLEN];
  const char *ret = inet_ntop(AF_INET, &addr.s_addr, buf, sizeof(buf));
  printf("%s\n", buf);
  printf("%p, %p\n", buf, ret);
  return 0;
}
#if 0
192.168.1.2
```

(continues on next page)

(continued from previous page)

```
0x7ffc808b5e80, 0x7ffc808b5e80
#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

```
// https://github.com/bminor/glibc/blob/master/resolv/inet_ntop.c
/*
 * Copyright (c) 1996-1999 by Internet Software Consortium.
 *
 * Permission to use, copy, modify, and distribute this software for any
 * purpose with or without fee is hereby granted, provided that the above
 * copyright notice and this permission notice appear in all copies.
 *
 * THE SOFTWARE IS PROVIDED "AS IS" AND INTERNET SOFTWARE CONSORTIUM DISCLAIMS
 * ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INTERNET SOFTWARE
 * CONSORTIUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL
 * DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR
 * PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS
 * ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS
 * SOFTWARE.
 */

#include <sys/param.h>
#include <sys/types.h>
#include <sys/socket.h>

#include <netinet/in.h>
#include <arpa/inet.h>
#include <arpa/nameser.h>

#include <errno.h>
#include <stdio.h>
#include <string.h>

#ifdef SPRINTF_CHAR
# define SPRINTF(x) strlen(sprintf/**/x)
#else
# define SPRINTF(x) ((size_t)sprintf x)
#endif

/*
 * WARNING: Don't even consider trying to compile this on a system where
 * sizeof(int) < 4.  sizeof(int) > 4 is fine; all the world's not a VAX.
 */

static const char *inet_ntop4 (const u_char *src, char *dst, socklen_t size);
static const char *inet_ntop6 (const u_char *src, char *dst, socklen_t size);

/* char *
```

(continues on next page)

(continued from previous page)

```
 * inet_ntop(af, src, dst, size)
 *        convert a network format address to presentation format.
 * return:
 *        pointer to presentation format address (`dst'), or NULL (see errno).
 * author:
 *        Paul Vixie, 1996.
 */
const char *
inet_ntop (int af, const void *src, char *dst, socklen_t size)
{
        switch (af) {
        case AF_INET:
                return (inet_ntop4(src, dst, size));
        case AF_INET6:
                return (inet_ntop6(src, dst, size));
        default:
                __set_errno (EAFNOSUPPORT);
                return (NULL);
        }
        /* NOTREACHED */
}
libc_hidden_def (inet_ntop)

/* const char *
 * inet_ntop4(src, dst, size)
 *        format an IPv4 address
 * return:
 *        `dst' (as a const)
 * notes:
 *        (1) uses no statics
 *        (2) takes a u_char* not an in_addr as input
 * author:
 *        Paul Vixie, 1996.
 */
static const char *
inet_ntop4 (const u_char *src, char *dst, socklen_t size)
{
        static const char fmt[] = "%u.%u.%u.%u";
        char tmp[sizeof "255.255.255.255"];

        if (SPRINTF((tmp, fmt, src[0], src[1], src[2], src[3])) >= size) {
                __set_errno (ENOSPC);
                return (NULL);
        }
        return strcpy(dst, tmp);
}

/* const char *
 * inet_ntop6(src, dst, size)
 *        convert IPv6 binary address into presentation (printable) format
 * author:
 *        Paul Vixie, 1996.
```

(continues on next page)

(continued from previous page)

```
 */
static const char *
inet_ntop6 (const u_char *src, char *dst, socklen_t size)
{
        /*
         * Note that int32_t and int16_t need only be "at least" large enough
         * to contain a value of the specified size.  On some systems, like
         * Crays, there is no such thing as an integer variable with 16 bits.
         * Keep this in mind if you think this function should have been coded
         * to use pointer overlays.  All the world's not a VAX.
         */
        char tmp[sizeof "ffff:ffff:ffff:ffff:ffff:ffff:255.255.255.255"], *tp;
        struct { int base, len; } best, cur;
        u_int words[NS_IN6ADDRSZ / NS_INT16SZ];
        int i;

        /*
         * Preprocess:
         *        Copy the input (bytewise) array into a wordwise array.
         *        Find the longest run of 0x00's in src[] for :: shorthanding.
         */
        memset(words, '\0', sizeof words);
        for (i = 0; i < NS_IN6ADDRSZ; i += 2)
                words[i / 2] = (src[i] << 8) | src[i + 1];
        best.base = -1;
        cur.base = -1;
        best.len = 0;
        cur.len = 0;
        for (i = 0; i < (NS_IN6ADDRSZ / NS_INT16SZ); i++) {
                if (words[i] == 0) {
                        if (cur.base == -1)
                                cur.base = i, cur.len = 1;
                        else
                                cur.len++;
                } else {
                        if (cur.base != -1) {
                                if (best.base == -1 || cur.len > best.len)
                                        best = cur;
                                cur.base = -1;
                        }
                }
        }
        if (cur.base != -1) {
                if (best.base == -1 || cur.len > best.len)
                        best = cur;
        }
        if (best.base != -1 && best.len < 2)
                best.base = -1;

        /*
         * Format the result.
         */
```

(continues on next page)

(continued from previous page)

```
        tp = tmp;
        for (i = 0; i < (NS_IN6ADDRSZ / NS_INT16SZ); i++) {
                /* Are we inside the best run of 0x00's? */
                if (best.base != -1 && i >= best.base &&
                    i < (best.base + best.len)) {
                        if (i == best.base)
                                *tp++ = ':';
                        continue;
                }
                /* Are we following an initial run of 0x00s or any real hex? */
                if (i != 0)
                        *tp++ = ':';
                /* Is this address an encapsulated IPv4? */
                if (i == 6 && best.base == 0 &&
                    (best.len == 6 || (best.len == 5 && words[5] == 0xffff))) {
                        if (!inet_ntop4(src+12, tp, sizeof tmp - (tp - tmp)))
                                return (NULL);
                        tp += strlen(tp);
                        break;
                }
                tp += SPRINTF((tp, "%x", words[i]));
        }
        /* Was it a trailing run of 0x00's? */
        if (best.base != -1 && (best.base + best.len) ==
            (NS_IN6ADDRSZ / NS_INT16SZ))
                *tp++ = ':';
        *tp++ = '\0';

        /*
         * Check for overflow, copy, and we're done.
         */
        if ((socklen_t)(tp - tmp) > size) {
                __set_errno (ENOSPC);
                return (NULL);
        }
        return strcpy(dst, tmp);
}
```

9.8.7 Echo Server and Client

Server

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

```
#!/usr/bin/env python3
import socket
import threading

# nc localhost 6006

```

(continues on next page)

(continued from previous page)

```python

def run_server():
    sock = socket.socket(family=socket.AF_INET, type=socket.SOCK_STREAM)
    sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    sock.bind(("", 6006))
    #  sock.setblocking(False) # It returns socket.BlockingIOError
    sock.listen(2)  # backlog is 2
    while True:
        client_sock, addr = sock.accept()
        assert isinstance(client_sock, socket.socket)
        assert isinstance(addr, tuple)
        assert isinstance(addr[0], str)
        assert isinstance(addr[1], int)
        print("Connected from", addr)  # Connected from ('127.0.0.1', 54266)
        threading.Thread(target=handle_client, args=(client_sock,)).start()


def handle_client(sock: socket.socket):
    while True:
        data = sock.recv(1024)
        if not data:
            break
        sock.sendall(data.decode("utf-8").upper().encode())
    print("Disconnected from", sock.getpeername())
    sock.close()


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

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

Client

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

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

Server2

With `concurrent.futures.ThreadPoolExecutor`.

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

```
#!/usr/bin/env python3
import socket
import threading
from concurrent.futures import ThreadPoolExecutor

# nc localhost 6006
pool = ThreadPoolExecutor(max_workers=3)


def run_server():
    sock = socket.socket(family=socket.AF_INET, type=socket.SOCK_STREAM)
    sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    sock.bind(("", 6006))
    #  sock.setblocking(False) # It returns socket.BlockingIOError
    sock.listen(2)  # backlog is 2
    while True:
        client_sock, addr = sock.accept()
        assert isinstance(client_sock, socket.socket)
        assert isinstance(addr, tuple)
        assert isinstance(addr[0], str)
        assert isinstance(addr[1], int)
        print("Connected from", addr)  # Connected from ('127.0.0.1', 54266)
        pool.submit(handle_client, client_sock)


def handle_client(sock: socket.socket):
    while True:
        data = sock.recv(1024)
        if not data:
            break
        sock.sendall(data.decode("utf-8").upper().encode())
    print("Disconnected from", sock.getpeername())
    sock.close()


```

(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/

CHAPTER

TEN

JAVA

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

```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

```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`.

CHAPTER

ELEVEN

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

```
let a = [ 1, 2, 3 ];
function sum(arr) {
  let s = 0;
  for (let x of arr) {
    s += x;
  }
  return s;
}
// Sum of the array [1,2,3] is 6
console.log('Sum of the array [' + a + '] is ' + sum(a));

function sum2(arr) {
  let s = 0;
  for (let i = 0; i != arr.length; ++i) {
```

(continues on next page)

(continued from previous page)

```
    s += arr[i];
  }
  return s;
}
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

```
class Point {
  constructor(x, y) {
    this.x = x;
    this.y = y;
  }

  distance() { return Math.sqrt(this.x * this.x + this.y * this.y); }
}

let p = new Point(1, 1);
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

CHAPTER

TWELVE

HTML

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 src="a.png" alt="yyy"></img>
<img src="foo/bar/b.png" alt="yyy"></img>
<img src="../../c.png" alt="yyy"></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

CHAPTER

THIRTEEN

CSS

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 <a id="my-id"> or <p id="my-id">
- class: e.g., `.my-class` selects <a class="my-class"> and <p class="my-class">
- attribute: e.g., `img[src]` selects <img src="a.png"> but not <img>

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

CHAPTER

FOURTEEN

PYBIND11

14.1 GIL

CHAPTER

FIFTEEN

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)

```
Making install in .
make[1]: Entering directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.20.1'
make[2]: Entering directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.20.1'
make[2]: Nothing to be done for 'install-exec-am'.
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/lib/pkgconfig'
 /usr/bin/install -c -m 644 protobuf.pc protobuf-lite.pc '/root/fangjun/software/protobuf-3.20.1/lib/pkgconfig'
make[2]: Leaving directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.20.1'
make[1]: Leaving directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.20.1'
Making install in src
make[1]: Entering directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.20.1/src'
make[2]: Entering directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.20.1/src'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/lib'
 /bin/bash ../libtool   --mode=install /usr/bin/install -c   libprotobuf-lite.la libprotobuf.la libprotoc.la '/root/fangjun/software/protobuf-3.20.1/lib'
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
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; }; })
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; }; })
libtool: install: /usr/bin/install -c .libs/libprotobuf-lite.lai /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf-lite.la
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
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; }; })
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; }; })
libtool: install: /usr/bin/install -c .libs/libprotobuf.lai /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf.la
libtool: warning: relinking 'libprotoc.la'
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 -DNDEBUG -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)

```
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
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; }; })
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; }; })
libtool: install: /usr/bin/install -c .libs/libprotoc.lai /root/fangjun/software/protobuf-3.20.1/lib/libprotoc.la
libtool: install: /usr/bin/install -c .libs/libprotobuf-lite.a /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf-lite.a
libtool: install: chmod 644 /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf-lite.a
libtool: install: ranlib /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf-lite.a
libtool: install: /usr/bin/install -c .libs/libprotobuf.a /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf.a
libtool: install: chmod 644 /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf.a
libtool: install: ranlib /root/fangjun/software/protobuf-3.20.1/lib/libprotobuf.a
libtool: install: /usr/bin/install -c .libs/libprotoc.a /root/fangjun/software/protobuf-3.20.1/lib/libprotoc.a
libtool: install: chmod 644 /root/fangjun/software/protobuf-3.20.1/lib/libprotoc.a
libtool: install: ranlib /root/fangjun/software/protobuf-3.20.1/lib/libprotoc.a
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
----------------------------------------------------------------------
Libraries have been installed in:
   /root/fangjun/software/protobuf-3.20.1/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the '-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the 'LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the 'LD_RUN_PATH' environment variable
     during linking
   - use the '-Wl,-rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to '/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/bin'
  /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)

```
libtool: install: /usr/bin/install -c .libs/protoc /root/fangjun/software/protobuf-3.20.1/bin/protoc
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler'
 /usr/bin/install -c -m 644  google/protobuf/compiler/plugin.proto '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/js'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/cpp'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/ruby'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/python'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/util'
```

(continues on next page)

(continued from previous page)

```
 /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_resolver_util.h '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/util'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/io'
 /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 '/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'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/php'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/stubs'
 /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 '/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'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler/objectivec'
 /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'
 /bin/mkdir -p '/root/fangjun/software/protobuf-3.20.1/include/google/protobuf/compiler'
 /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'
```

(continues on next page)

(continued from previous page)

```
make[2]: Leaving directory '/ceph-fj/fangjun/open-source-2/protocol-buffers/protobuf-3.
↪20.1/src'
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)

```
.
|-- bin
|   `-- protoc
|-- include
|   `-- google
|       `-- protobuf
|           |-- any.h
|           |-- any.pb.h
|           |-- any.proto
|           |-- api.pb.h
|           |-- api.proto
|           |-- arena.h
|           |-- arena_impl.h
|           |-- arenastring.h
|           |-- arenaz_sampler.h
|           |-- compiler
|           |   |-- code_generator.h
|           |   |-- command_line_interface.h
|           |   |-- cpp
|           |   |   |-- cpp_file.h
|           |   |   |-- cpp_generator.h
|           |   |   |-- cpp_helpers.h
|           |   |   `-- cpp_names.h
|           |   |-- csharp
|           |   |   |-- csharp_doc_comment.h
|           |   |   |-- csharp_generator.h
|           |   |   |-- csharp_names.h
|           |   |   `-- csharp_options.h
|           |   |-- importer.h
|           |   |-- java
|           |   |   |-- java_generator.h
|           |   |   |-- java_kotlin_generator.h
|           |   |   `-- java_names.h
|           |   |-- js
|           |   |   `-- js_generator.h
|           |   |-- objectivec
|           |   |   |-- objectivec_generator.h
|           |   |   `-- objectivec_helpers.h
|           |   |-- parser.h
|           |   |-- php
|           |   |   `-- php_generator.h
|           |   |-- plugin.h
|           |   |-- plugin.pb.h
|           |   |-- plugin.proto
|           |   |-- python
```

(continues on next page)

(continued from previous page)

```
|           |   |   |-- python_generator.h
|           |   |   `-- python_pyi_generator.h
|           |   `-- ruby
|           |       `-- ruby_generator.h
|           |-- descriptor.h
|           |-- descriptor.pb.h
|           |-- descriptor.proto
|           |-- descriptor_database.h
|           |-- duration.pb.h
|           |-- duration.proto
|           |-- dynamic_message.h
|           |-- empty.pb.h
|           |-- empty.proto
|           |-- explicitly_constructed.h
|           |-- extension_set.h
|           |-- extension_set_inl.h
|           |-- field_access_listener.h
|           |-- field_mask.pb.h
|           |-- field_mask.proto
|           |-- generated_enum_reflection.h
|           |-- generated_enum_util.h
|           |-- generated_message_bases.h
|           |-- generated_message_reflection.h
|           |-- generated_message_tctable_decl.h
|           |-- generated_message_tctable_impl.h
|           |-- generated_message_util.h
|           |-- has_bits.h
|           |-- implicit_weak_message.h
|           |-- inlined_string_field.h
|           |-- io
|           |   |-- coded_stream.h
|           |   |-- gzip_stream.h
|           |   |-- io_win32.h
|           |   |-- printer.h
|           |   |-- strtod.h
|           |   |-- tokenizer.h
|           |   |-- zero_copy_stream.h
|           |   |-- zero_copy_stream_impl.h
|           |   `-- zero_copy_stream_impl_lite.h
|           |-- map.h
|           |-- map_entry.h
|           |-- map_entry_lite.h
|           |-- map_field.h
|           |-- map_field_inl.h
|           |-- map_field_lite.h
|           |-- map_type_handler.h
|           |-- message.h
|           |-- message_lite.h
|           |-- metadata.h
|           |-- metadata_lite.h
|           |-- parse_context.h
|           |-- port.h
```

(continues on next page)

(continued from previous page)

```
|           |-- port_def.inc
|           |-- port_undef.inc
|           |-- reflection.h
|           |-- reflection_ops.h
|           |-- repeated_field.h
|           |-- repeated_ptr_field.h
|           |-- service.h
|           |-- source_context.pb.h
|           |-- source_context.proto
|           |-- struct.pb.h
|           |-- struct.proto
|           |-- stubs
|           |   |-- bytestream.h
|           |   |-- callback.h
|           |   |-- casts.h
|           |   |-- common.h
|           |   |-- hash.h
|           |   |-- logging.h
|           |   |-- macros.h
|           |   |-- map_util.h
|           |   |-- mutex.h
|           |   |-- once.h
|           |   |-- platform_macros.h
|           |   |-- port.h
|           |   |-- status.h
|           |   |-- stl_util.h
|           |   |-- stringpiece.h
|           |   |-- strutil.h
|           |   `-- template_util.h
|           |-- text_format.h
|           |-- timestamp.pb.h
|           |-- timestamp.proto
|           |-- type.pb.h
|           |-- type.proto
|           |-- unknown_field_set.h
|           |-- util
|           |   |-- delimited_message_util.h
|           |   |-- field_comparator.h
|           |   |-- field_mask_util.h
|           |   |-- json_util.h
|           |   |-- message_differencer.h
|           |   |-- time_util.h
|           |   |-- type_resolver.h
|           |   `-- type_resolver_util.h
|           |-- wire_format.h
|           |-- wire_format_lite.h
|           |-- wrappers.pb.h
|           `-- wrappers.proto
|-- lib
|   |-- libprotobuf-lite.a
|   |-- libprotobuf-lite.la
|   |-- libprotobuf-lite.so -> libprotobuf-lite.so.31.0.1
```

(continues on next page)

(continued from previous page)

```
|   |-- libprotobuf-lite.so.31 -> libprotobuf-lite.so.31.0.1
|   |-- libprotobuf-lite.so.31.0.1
|   |-- libprotobuf.a
|   |-- libprotobuf.la
|   |-- libprotobuf.so -> libprotobuf.so.31.0.1
|   |-- libprotobuf.so.31 -> libprotobuf.so.31.0.1
|   |-- libprotobuf.so.31.0.1
|   |-- libprotoc.a
|   |-- libprotoc.la
|   |-- libprotoc.so -> libprotoc.so.31.0.1
|   |-- libprotoc.so.31 -> libprotoc.so.31.0.1
|   |-- libprotoc.so.31.0.1
|   `-- pkgconfig
|       |-- protobuf-lite.pc
|       `-- protobuf.pc
`-- tree-log.txt

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

```
syntax = "proto2";

package tutorial;

// available types:
// bool, int32, float, double, string
message Person {
  optional string name = 1;
  optional int32 id = 2;
  optional string email = 3;

  enum PhoneType {
    MOBILE = 0;
    HOME = 1;
    WORK = 2;
  }

  message PhoneNumber {
    optional string number = 1;
    optional PhoneType type = 2 [ default = HOME ];
```

(continues on next page)

(continued from previous page)

```
  }

  repeated PhoneNumber phones = 4;
}

message AddressBook { repeated Person people = 1; }
```

15.2.2 makefile

Listing 4: ./code/Makefile

```
.PHONY: all clean

all: hello.pb.cc hello.pb.h hello_pb2.py

hello.pb.cc hello.pb.h: hello.proto
	protoc -I=./ --cpp_out=./ ./hello.proto

hello_pb2.py: hello.proto
	protoc -I=./ --python_out=./ ./hello.proto

clean:
	$(RM) hello.pb.cc hello.pb.h hello_pb2.py
```

15.2.3 hello.pb.h

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

```
// Generated by the protocol buffer compiler.  DO NOT EDIT!
// source: hello.proto

#ifndef GOOGLE_PROTOBUF_INCLUDED_hello_2eproto
#define GOOGLE_PROTOBUF_INCLUDED_hello_2eproto

#include <limits>
#include <string>

#include <google/protobuf/port_def.inc>
#if PROTOBUF_VERSION < 3020000
#error This file was generated by a newer version of protoc which is
#error incompatible with your Protocol Buffer headers. Please update
#error your headers.
#endif
#if 3020001 < PROTOBUF_MIN_PROTOC_VERSION
#error This file was generated by an older version of protoc which is
#error incompatible with your Protocol Buffer headers. Please
#error regenerate this file with a newer version of protoc.
#endif

```

(continues on next page)

(continued from previous page)

```
#include <google/protobuf/port_undef.inc>
#include <google/protobuf/io/coded_stream.h>
#include <google/protobuf/arena.h>
#include <google/protobuf/arenastring.h>
#include <google/protobuf/generated_message_util.h>
#include <google/protobuf/metadata_lite.h>
#include <google/protobuf/generated_message_reflection.h>
#include <google/protobuf/message.h>
#include <google/protobuf/repeated_field.h>  // IWYU pragma: export
#include <google/protobuf/extension_set.h>  // IWYU pragma: export
#include <google/protobuf/generated_enum_reflection.h>
#include <google/protobuf/unknown_field_set.h>
// @@protoc_insertion_point(includes)
#include <google/protobuf/port_def.inc>
#define PROTOBUF_INTERNAL_EXPORT_hello_2eproto
PROTOBUF_NAMESPACE_OPEN
namespace internal {
class AnyMetadata;
}  // namespace internal
PROTOBUF_NAMESPACE_CLOSE

// Internal implementation detail -- do not use these members.
struct TableStruct_hello_2eproto {
  static const uint32_t offsets[];
};
extern const ::PROTOBUF_NAMESPACE_ID::internal::DescriptorTable descriptor_table_hello_
↪2eproto;
namespace tutorial {
class AddressBook;
struct AddressBookDefaultTypeInternal;
extern AddressBookDefaultTypeInternal _AddressBook_default_instance_;
class Person;
struct PersonDefaultTypeInternal;
extern PersonDefaultTypeInternal _Person_default_instance_;
class Person_PhoneNumber;
struct Person_PhoneNumberDefaultTypeInternal;
extern Person_PhoneNumberDefaultTypeInternal _Person_PhoneNumber_default_instance_;
}  // namespace tutorial
PROTOBUF_NAMESPACE_OPEN
template<> ::tutorial::AddressBook* Arena::CreateMaybeMessage<::tutorial::AddressBook>
↪(Arena*);
template<> ::tutorial::Person* Arena::CreateMaybeMessage<::tutorial::Person>(Arena*);
template<> ::tutorial::Person_PhoneNumber* Arena::CreateMaybeMessage<::tutorial::Person_
↪PhoneNumber>(Arena*);
PROTOBUF_NAMESPACE_CLOSE
namespace tutorial {

enum Person_PhoneType : int {
  Person_PhoneType_MOBILE = 0,
  Person_PhoneType_HOME = 1,
  Person_PhoneType_WORK = 2
};
```

(continues on next page)

(continued from previous page)

```
bool Person_PhoneType_IsValid(int value);
constexpr Person_PhoneType Person_PhoneType_PhoneType_MIN = Person_PhoneType_MOBILE;
constexpr Person_PhoneType Person_PhoneType_PhoneType_MAX = Person_PhoneType_WORK;
constexpr int Person_PhoneType_PhoneType_ARRAYSIZE = Person_PhoneType_PhoneType_MAX + 1;

const ::PROTOBUF_NAMESPACE_ID::EnumDescriptor* Person_PhoneType_descriptor();
template<typename T>
inline const std::string& Person_PhoneType_Name(T enum_t_value) {
  static_assert(::std::is_same<T, Person_PhoneType>::value ||
    ::std::is_integral<T>::value,
    "Incorrect type passed to function Person_PhoneType_Name.");
  return ::PROTOBUF_NAMESPACE_ID::internal::NameOfEnum(
    Person_PhoneType_descriptor(), enum_t_value);
}
inline bool Person_PhoneType_Parse(
    ::PROTOBUF_NAMESPACE_ID::ConstStringParam name, Person_PhoneType* value) {
  return ::PROTOBUF_NAMESPACE_ID::internal::ParseNamedEnum<Person_PhoneType>(
    Person_PhoneType_descriptor(), name, value);
}
// ===================================================================

class Person_PhoneNumber final :
    public ::PROTOBUF_NAMESPACE_ID::Message /* @@protoc_insertion_point(class_
↪definition:tutorial.Person.PhoneNumber) */ {
 public:
  inline Person_PhoneNumber() : Person_PhoneNumber(nullptr) {}
  ~Person_PhoneNumber() override;
  explicit PROTOBUF_CONSTEXPR Person_PhoneNumber(::PROTOBUF_NAMESPACE_
↪ID::internal::ConstantInitialized);

  Person_PhoneNumber(const Person_PhoneNumber& from);
  Person_PhoneNumber(Person_PhoneNumber&& from) noexcept
    : Person_PhoneNumber() {
    *this = ::std::move(from);
  }

  inline Person_PhoneNumber& operator=(const Person_PhoneNumber& from) {
    CopyFrom(from);
    return *this;
  }
  inline Person_PhoneNumber& operator=(Person_PhoneNumber&& from) noexcept {
    if (this == &from) return *this;
    if (GetOwningArena() == from.GetOwningArena()
  #ifdef PROTOBUF_FORCE_COPY_IN_MOVE
        && GetOwningArena() != nullptr
  #endif  // !PROTOBUF_FORCE_COPY_IN_MOVE
    ) {
      InternalSwap(&from);
    } else {
      CopyFrom(from);
    }
    return *this;
```

(continues on next page)

(continued from previous page)

```
  }

  inline const ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet& unknown_fields() const {
    return _internal_metadata_.unknown_fields<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
↪(::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance);
  }
  inline ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet* mutable_unknown_fields() {
    return _internal_metadata_.mutable_unknown_fields<::PROTOBUF_NAMESPACE_
↪ID::UnknownFieldSet>();
  }

  static const ::PROTOBUF_NAMESPACE_ID::Descriptor* descriptor() {
    return GetDescriptor();
  }
  static const ::PROTOBUF_NAMESPACE_ID::Descriptor* GetDescriptor() {
    return default_instance().GetMetadata().descriptor;
  }
  static const ::PROTOBUF_NAMESPACE_ID::Reflection* GetReflection() {
    return default_instance().GetMetadata().reflection;
  }
  static const Person_PhoneNumber& default_instance() {
    return *internal_default_instance();
  }
  static inline const Person_PhoneNumber* internal_default_instance() {
    return reinterpret_cast<const Person_PhoneNumber*>(
               &_Person_PhoneNumber_default_instance_);
  }
  static constexpr int kIndexInFileMessages =
    0;

  friend void swap(Person_PhoneNumber& a, Person_PhoneNumber& b) {
    a.Swap(&b);
  }
  inline void Swap(Person_PhoneNumber* other) {
    if (other == this) return;
  #ifdef PROTOBUF_FORCE_COPY_IN_SWAP
    if (GetOwningArena() != nullptr &&
        GetOwningArena() == other->GetOwningArena()) {
   #else  // PROTOBUF_FORCE_COPY_IN_SWAP
    if (GetOwningArena() == other->GetOwningArena()) {
  #endif  // !PROTOBUF_FORCE_COPY_IN_SWAP
      InternalSwap(other);
    } else {
      ::PROTOBUF_NAMESPACE_ID::internal::GenericSwap(this, other);
    }
  }
  void UnsafeArenaSwap(Person_PhoneNumber* other) {
    if (other == this) return;
    GOOGLE_DCHECK(GetOwningArena() == other->GetOwningArena());
    InternalSwap(other);
  }

```

(continues on next page)

(continued from previous page)

```
  // implements Message ----------------------------------------------

  Person_PhoneNumber* New(::PROTOBUF_NAMESPACE_ID::Arena* arena = nullptr) const final {
    return CreateMaybeMessage<Person_PhoneNumber>(arena);
  }
  using ::PROTOBUF_NAMESPACE_ID::Message::CopyFrom;
  void CopyFrom(const Person_PhoneNumber& from);
  using ::PROTOBUF_NAMESPACE_ID::Message::MergeFrom;
  void MergeFrom(const Person_PhoneNumber& from);
  private:
  static void MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to, const ::PROTOBUF_NAMESPACE_
↪ID::Message& from);
  public:
  PROTOBUF_ATTRIBUTE_REINITIALIZES void Clear() final;
  bool IsInitialized() const final;

  size_t ByteSizeLong() const final;
  const char* _InternalParse(const char* ptr, ::PROTOBUF_NAMESPACE_
↪ID::internal::ParseContext* ctx) final;
  uint8_t* _InternalSerialize(
      uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const␣
↪final;
  int GetCachedSize() const final { return _cached_size_.Get(); }

  private:
  void SharedCtor();
  void SharedDtor();
  void SetCachedSize(int size) const final;
  void InternalSwap(Person_PhoneNumber* other);

  private:
  friend class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata;
  static ::PROTOBUF_NAMESPACE_ID::StringPiece FullMessageName() {
    return "tutorial.Person.PhoneNumber";
  }
  protected:
  explicit Person_PhoneNumber(::PROTOBUF_NAMESPACE_ID::Arena* arena,
                              bool is_message_owned = false);
  public:

  static const ClassData _class_data_;
  const ::PROTOBUF_NAMESPACE_ID::Message::ClassData*GetClassData() const final;

  ::PROTOBUF_NAMESPACE_ID::Metadata GetMetadata() const final;

  // nested types ----------------------------------------------------

  // accessors -------------------------------------------------------

  enum : int {
    kNumberFieldNumber = 1,
    kTypeFieldNumber = 2,
```

(continues on next page)

(continued from previous page)

```
  };
  // optional string number = 1;
  bool has_number() const;
  private:
  bool _internal_has_number() const;
  public:
  void clear_number();
  const std::string& number() const;
  template <typename ArgT0 = const std::string&, typename... ArgT>
  void set_number(ArgT0&& arg0, ArgT... args);
  std::string* mutable_number();
  PROTOBUF_NODISCARD std::string* release_number();
  void set_allocated_number(std::string* number);
  private:
  const std::string& _internal_number() const;
  inline PROTOBUF_ALWAYS_INLINE void _internal_set_number(const std::string& value);
  std::string* _internal_mutable_number();
  public:

  // optional .tutorial.Person.PhoneType type = 2 [default = HOME];
  bool has_type() const;
  private:
  bool _internal_has_type() const;
  public:
  void clear_type();
  ::tutorial::Person_PhoneType type() const;
  void set_type(::tutorial::Person_PhoneType value);
  private:
  ::tutorial::Person_PhoneType _internal_type() const;
  void _internal_set_type(::tutorial::Person_PhoneType value);
  public:

  // @@protoc_insertion_point(class_scope:tutorial.Person.PhoneNumber)
 private:
  class _Internal;

  template <typename T> friend class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper;
  typedef void InternalArenaConstructable_;
  typedef void DestructorSkippable_;
  ::PROTOBUF_NAMESPACE_ID::internal::HasBits<1> _has_bits_;
  mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize _cached_size_;
  ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr number_;
  int type_;
  friend struct ::TableStruct_hello_2eproto;
};
// -------------------------------------------------------------------

class Person final :
    public ::PROTOBUF_NAMESPACE_ID::Message /* @@protoc_insertion_point(class_
↪definition:tutorial.Person) */ {
 public:
  inline Person() : Person(nullptr) {}
```

(continues on next page)

(continued from previous page)

```
  ~Person() override;
  explicit PROTOBUF_CONSTEXPR Person(::PROTOBUF_NAMESPACE_
↪ID::internal::ConstantInitialized);

  Person(const Person& from);
  Person(Person&& from) noexcept
    : Person() {
    *this = ::std::move(from);
  }

  inline Person& operator=(const Person& from) {
    CopyFrom(from);
    return *this;
  }
  inline Person& operator=(Person&& from) noexcept {
    if (this == &from) return *this;
    if (GetOwningArena() == from.GetOwningArena()
  #ifdef PROTOBUF_FORCE_COPY_IN_MOVE
        && GetOwningArena() != nullptr
  #endif  // !PROTOBUF_FORCE_COPY_IN_MOVE
    ) {
      InternalSwap(&from);
    } else {
      CopyFrom(from);
    }
    return *this;
  }

  inline const ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet& unknown_fields() const {
    return _internal_metadata_.unknown_fields<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
↪(::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance);
  }
  inline ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet* mutable_unknown_fields() {
    return _internal_metadata_.mutable_unknown_fields<::PROTOBUF_NAMESPACE_
↪ID::UnknownFieldSet>();
  }

  static const ::PROTOBUF_NAMESPACE_ID::Descriptor* descriptor() {
    return GetDescriptor();
  }
  static const ::PROTOBUF_NAMESPACE_ID::Descriptor* GetDescriptor() {
    return default_instance().GetMetadata().descriptor;
  }
  static const ::PROTOBUF_NAMESPACE_ID::Reflection* GetReflection() {
    return default_instance().GetMetadata().reflection;
  }
  static const Person& default_instance() {
    return *internal_default_instance();
  }
  static inline const Person* internal_default_instance() {
    return reinterpret_cast<const Person*>(
               &_Person_default_instance_);
```

(continues on next page)

(continued from previous page)

```
  }
  static constexpr int kIndexInFileMessages =
    1;

  friend void swap(Person& a, Person& b) {
    a.Swap(&b);
  }
  inline void Swap(Person* other) {
    if (other == this) return;
  #ifdef PROTOBUF_FORCE_COPY_IN_SWAP
    if (GetOwningArena() != nullptr &&
        GetOwningArena() == other->GetOwningArena()) {
   #else  // PROTOBUF_FORCE_COPY_IN_SWAP
    if (GetOwningArena() == other->GetOwningArena()) {
  #endif  // !PROTOBUF_FORCE_COPY_IN_SWAP
      InternalSwap(other);
    } else {
      ::PROTOBUF_NAMESPACE_ID::internal::GenericSwap(this, other);
    }
  }
  void UnsafeArenaSwap(Person* other) {
    if (other == this) return;
    GOOGLE_DCHECK(GetOwningArena() == other->GetOwningArena());
    InternalSwap(other);
  }

  // implements Message ----------------------------------------------

  Person* New(::PROTOBUF_NAMESPACE_ID::Arena* arena = nullptr) const final {
    return CreateMaybeMessage<Person>(arena);
  }
  using ::PROTOBUF_NAMESPACE_ID::Message::CopyFrom;
  void CopyFrom(const Person& from);
  using ::PROTOBUF_NAMESPACE_ID::Message::MergeFrom;
  void MergeFrom(const Person& from);
  private:
  static void MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to, const ::PROTOBUF_NAMESPACE_
↪ID::Message& from);
  public:
  PROTOBUF_ATTRIBUTE_REINITIALIZES void Clear() final;
  bool IsInitialized() const final;

  size_t ByteSizeLong() const final;
  const char* _InternalParse(const char* ptr, ::PROTOBUF_NAMESPACE_
↪ID::internal::ParseContext* ctx) final;
  uint8_t* _InternalSerialize(
      uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const␣
↪final;
  int GetCachedSize() const final { return _cached_size_.Get(); }

  private:
  void SharedCtor();
```

(continues on next page)

(continued from previous page)

```
  void SharedDtor();
  void SetCachedSize(int size) const final;
  void InternalSwap(Person* other);

  private:
  friend class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata;
  static ::PROTOBUF_NAMESPACE_ID::StringPiece FullMessageName() {
    return "tutorial.Person";
  }
  protected:
  explicit Person(::PROTOBUF_NAMESPACE_ID::Arena* arena,
                  bool is_message_owned = false);
  public:

  static const ClassData _class_data_;
  const ::PROTOBUF_NAMESPACE_ID::Message::ClassData*GetClassData() const final;

  ::PROTOBUF_NAMESPACE_ID::Metadata GetMetadata() const final;

  // nested types ----------------------------------------------------

  typedef Person_PhoneNumber PhoneNumber;

  typedef Person_PhoneType PhoneType;
  static constexpr PhoneType MOBILE =
    Person_PhoneType_MOBILE;
  static constexpr PhoneType HOME =
    Person_PhoneType_HOME;
  static constexpr PhoneType WORK =
    Person_PhoneType_WORK;
  static inline bool PhoneType_IsValid(int value) {
    return Person_PhoneType_IsValid(value);
  }
  static constexpr PhoneType PhoneType_MIN =
    Person_PhoneType_PhoneType_MIN;
  static constexpr PhoneType PhoneType_MAX =
    Person_PhoneType_PhoneType_MAX;
  static constexpr int PhoneType_ARRAYSIZE =
    Person_PhoneType_PhoneType_ARRAYSIZE;
  static inline const ::PROTOBUF_NAMESPACE_ID::EnumDescriptor*
  PhoneType_descriptor() {
    return Person_PhoneType_descriptor();
  }
  template<typename T>
  static inline const std::string& PhoneType_Name(T enum_t_value) {
    static_assert(::std::is_same<T, PhoneType>::value ||
      ::std::is_integral<T>::value,
      "Incorrect type passed to function PhoneType_Name.");
    return Person_PhoneType_Name(enum_t_value);
  }
  static inline bool PhoneType_Parse(::PROTOBUF_NAMESPACE_ID::ConstStringParam name,
      PhoneType* value) {
```

(continues on next page)

(continued from previous page)

```
  return Person_PhoneType_Parse(name, value);
}

// accessors -------------------------------------------------------

enum : int {
  kPhonesFieldNumber = 4,
  kNameFieldNumber = 1,
  kEmailFieldNumber = 3,
  kIdFieldNumber = 2,
};
// repeated .tutorial.Person.PhoneNumber phones = 4;
int phones_size() const;
private:
int _internal_phones_size() const;
public:
void clear_phones();
::tutorial::Person_PhoneNumber* mutable_phones(int index);
::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person_PhoneNumber >*
    mutable_phones();
private:
const ::tutorial::Person_PhoneNumber& _internal_phones(int index) const;
::tutorial::Person_PhoneNumber* _internal_add_phones();
public:
const ::tutorial::Person_PhoneNumber& phones(int index) const;
::tutorial::Person_PhoneNumber* add_phones();
const ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person_PhoneNumber >&
    phones() const;

// optional string name = 1;
bool has_name() const;
private:
bool _internal_has_name() const;
public:
void clear_name();
const std::string& name() const;
template <typename ArgT0 = const std::string&, typename... ArgT>
void set_name(ArgT0&& arg0, ArgT... args);
std::string* mutable_name();
PROTOBUF_NODISCARD std::string* release_name();
void set_allocated_name(std::string* name);
private:
const std::string& _internal_name() const;
inline PROTOBUF_ALWAYS_INLINE void _internal_set_name(const std::string& value);
std::string* _internal_mutable_name();
public:

// optional string email = 3;
bool has_email() const;
private:
bool _internal_has_email() const;
public:
```

(continues on next page)

(continued from previous page)

```
  void clear_email();
  const std::string& email() const;
  template <typename ArgT0 = const std::string&, typename... ArgT>
  void set_email(ArgT0&& arg0, ArgT... args);
  std::string* mutable_email();
  PROTOBUF_NODISCARD std::string* release_email();
  void set_allocated_email(std::string* email);
  private:
  const std::string& _internal_email() const;
  inline PROTOBUF_ALWAYS_INLINE void _internal_set_email(const std::string& value);
  std::string* _internal_mutable_email();
  public:

  // optional int32 id = 2;
  bool has_id() const;
  private:
  bool _internal_has_id() const;
  public:
  void clear_id();
  int32_t id() const;
  void set_id(int32_t value);
  private:
  int32_t _internal_id() const;
  void _internal_set_id(int32_t value);
  public:

  // @@protoc_insertion_point(class_scope:tutorial.Person)
 private:
  class _Internal;

  template <typename T> friend class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper;
  typedef void InternalArenaConstructable_;
  typedef void DestructorSkippable_;
  ::PROTOBUF_NAMESPACE_ID::internal::HasBits<1> _has_bits_;
  mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize _cached_size_;
  ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person_PhoneNumber > phones_;
  ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr name_;
  ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr email_;
  int32_t id_;
  friend struct ::TableStruct_hello_2eproto;
};
// ---------------------------------------------------------------

class AddressBook final :
    public ::PROTOBUF_NAMESPACE_ID::Message /* @@protoc_insertion_point(class_
↪definition:tutorial.AddressBook) */ {
 public:
  inline AddressBook() : AddressBook(nullptr) {}
  ~AddressBook() override;
  explicit PROTOBUF_CONSTEXPR AddressBook(::PROTOBUF_NAMESPACE_
↪ID::internal::ConstantInitialized);

```

(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>
      ↪(::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance);
549   }
550   inline ::PROTOBUF_NAMESPACE_ID::UnknownFieldSet* mutable_unknown_fields() {
551     return _internal_metadata_.mutable_unknown_fields<::PROTOBUF_NAMESPACE_
      ↪ID::UnknownFieldSet>();
552   }
553
554   static const ::PROTOBUF_NAMESPACE_ID::Descriptor* descriptor() {
555     return GetDescriptor();
556   }
557   static const ::PROTOBUF_NAMESPACE_ID::Descriptor* GetDescriptor() {
558     return default_instance().GetMetadata().descriptor;
559   }
560   static const ::PROTOBUF_NAMESPACE_ID::Reflection* GetReflection() {
561     return default_instance().GetMetadata().reflection;
562   }
563   static const AddressBook& default_instance() {
564     return *internal_default_instance();
565   }
566   static inline const AddressBook* internal_default_instance() {
567     return reinterpret_cast<const AddressBook*>(
568                &_AddressBook_default_instance_);
569   }
570   static constexpr int kIndexInFileMessages =
571     2;
572
```

(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_
      ↪ID::Message& from);
606   public:
607   PROTOBUF_ATTRIBUTE_REINITIALIZES void Clear() final;
608   bool IsInitialized() const final;
609
610   size_t ByteSizeLong() const final;
611   const char* _InternalParse(const char* ptr, ::PROTOBUF_NAMESPACE_
      ↪ID::internal::ParseContext* ctx) final;
612   uint8_t* _InternalSerialize(
613       uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const␣
      ↪final;
614   int GetCachedSize() const final { return _cached_size_.Get(); }
615
616   private:
617   void SharedCtor();
618   void SharedDtor();
619   void SetCachedSize(int size) const final;
620   void InternalSwap(AddressBook* other);
621
```

(continues on next page)

(continued from previous page)

```
  private:
  friend class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata;
  static ::PROTOBUF_NAMESPACE_ID::StringPiece FullMessageName() {
    return "tutorial.AddressBook";
  }
  protected:
  explicit AddressBook(::PROTOBUF_NAMESPACE_ID::Arena* arena,
                       bool is_message_owned = false);
  public:

  static const ClassData _class_data_;
  const ::PROTOBUF_NAMESPACE_ID::Message::ClassData*GetClassData() const final;

  ::PROTOBUF_NAMESPACE_ID::Metadata GetMetadata() const final;

  // nested types ----------------------------------------------------

  // accessors -------------------------------------------------------

  enum : int {
    kPeopleFieldNumber = 1,
  };
  // repeated .tutorial.Person people = 1;
  int people_size() const;
  private:
  int _internal_people_size() const;
  public:
  void clear_people();
  ::tutorial::Person* mutable_people(int index);
  ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person >*
      mutable_people();
  private:
  const ::tutorial::Person& _internal_people(int index) const;
  ::tutorial::Person* _internal_add_people();
  public:
  const ::tutorial::Person& people(int index) const;
  ::tutorial::Person* add_people();
  const ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person >&
      people() const;

  // @@protoc_insertion_point(class_scope:tutorial.AddressBook)
 private:
  class _Internal;

  template <typename T> friend class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper;
  typedef void InternalArenaConstructable_;
  typedef void DestructorSkippable_;
  ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person > people_;
  mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize _cached_size_;
  friend struct ::TableStruct_hello_2eproto;
};
// ===================================================================
```

(continues on next page)

(continued from previous page)

```
674
675
676 // ===================================================================
677
678 #ifdef __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)

```
    return nullptr;
  }
  _has_bits_[0] &= ~0x00000001u;
  auto* p = number_.Release();
#ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
  if (number_.IsDefault()) {
    number_.Set("", GetArenaForAllocation());
  }
#endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
  return p;
}
inline void Person_PhoneNumber::set_allocated_number(std::string* number) {
  if (number != nullptr) {
    _has_bits_[0] |= 0x00000001u;
  } else {
    _has_bits_[0] &= ~0x00000001u;
  }
  number_.SetAllocated(number, GetArenaForAllocation());
#ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
  if (number_.IsDefault()) {
    number_.Set("", GetArenaForAllocation());
  }
#endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
  // @@protoc_insertion_point(field_set_allocated:tutorial.Person.PhoneNumber.number)
}

// optional .tutorial.Person.PhoneType type = 2 [default = HOME];
inline bool Person_PhoneNumber::_internal_has_type() const {
  bool value = (_has_bits_[0] & 0x00000002u) != 0;
  return value;
}
inline bool Person_PhoneNumber::has_type() const {
  return _internal_has_type();
}
inline void Person_PhoneNumber::clear_type() {
  type_ = 1;
  _has_bits_[0] &= ~0x00000002u;
}
inline ::tutorial::Person_PhoneType Person_PhoneNumber::_internal_type() const {
  return static_cast< ::tutorial::Person_PhoneType >(type_);
}
inline ::tutorial::Person_PhoneType Person_PhoneNumber::type() const {
  // @@protoc_insertion_point(field_get:tutorial.Person.PhoneNumber.type)
  return _internal_type();
}
inline void Person_PhoneNumber::_internal_set_type(::tutorial::Person_PhoneType value) {
  assert(::tutorial::Person_PhoneType_IsValid(value));
  _has_bits_[0] |= 0x00000002u;
  type_ = value;
}
inline void Person_PhoneNumber::set_type(::tutorial::Person_PhoneType value) {
  _internal_set_type(value);
```

(continues on next page)

(continued from previous page)

```
  // @@protoc_insertion_point(field_set:tutorial.Person.PhoneNumber.type)
}

// -------------------------------------------------------------------

// Person

// optional string name = 1;
inline bool Person::_internal_has_name() const {
  bool value = (_has_bits_[0] & 0x00000001u) != 0;
  return value;
}
inline bool Person::has_name() const {
  return _internal_has_name();
}
inline void Person::clear_name() {
  name_.ClearToEmpty();
  _has_bits_[0] &= ~0x00000001u;
}
inline const std::string& Person::name() const {
  // @@protoc_insertion_point(field_get:tutorial.Person.name)
  return _internal_name();
}
template <typename ArgT0, typename... ArgT>
inline PROTOBUF_ALWAYS_INLINE
void Person::set_name(ArgT0&& arg0, ArgT... args) {
 _has_bits_[0] |= 0x00000001u;
 name_.Set(static_cast<ArgT0 &&>(arg0), args..., GetArenaForAllocation());
  // @@protoc_insertion_point(field_set:tutorial.Person.name)
}
inline std::string* Person::mutable_name() {
  std::string* _s = _internal_mutable_name();
  // @@protoc_insertion_point(field_mutable:tutorial.Person.name)
  return _s;
}
inline const std::string& Person::_internal_name() const {
  return name_.Get();
}
inline void Person::_internal_set_name(const std::string& value) {
  _has_bits_[0] |= 0x00000001u;
  name_.Set(value, GetArenaForAllocation());
}
inline std::string* Person::_internal_mutable_name() {
  _has_bits_[0] |= 0x00000001u;
  return name_.Mutable(GetArenaForAllocation());
}
inline std::string* Person::release_name() {
  // @@protoc_insertion_point(field_release:tutorial.Person.name)
  if (!_internal_has_name()) {
    return nullptr;
  }
  _has_bits_[0] &= ~0x00000001u;
```

(continues on next page)

(continued from previous page)

```
  auto* p = name_.Release();
#ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
  if (name_.IsDefault()) {
    name_.Set("", GetArenaForAllocation());
  }
#endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
  return p;
}
inline void Person::set_allocated_name(std::string* name) {
  if (name != nullptr) {
    _has_bits_[0] |= 0x00000001u;
  } else {
    _has_bits_[0] &= ~0x00000001u;
  }
  name_.SetAllocated(name, GetArenaForAllocation());
#ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
  if (name_.IsDefault()) {
    name_.Set("", GetArenaForAllocation());
  }
#endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
  // @@protoc_insertion_point(field_set_allocated:tutorial.Person.name)
}

// optional int32 id = 2;
inline bool Person::_internal_has_id() const {
  bool value = (_has_bits_[0] & 0x00000004u) != 0;
  return value;
}
inline bool Person::has_id() const {
  return _internal_has_id();
}
inline void Person::clear_id() {
  id_ = 0;
  _has_bits_[0] &= ~0x00000004u;
}
inline int32_t Person::_internal_id() const {
  return id_;
}
inline int32_t Person::id() const {
  // @@protoc_insertion_point(field_get:tutorial.Person.id)
  return _internal_id();
}
inline void Person::_internal_set_id(int32_t value) {
  _has_bits_[0] |= 0x00000004u;
  id_ = value;
}
inline void Person::set_id(int32_t value) {
  _internal_set_id(value);
  // @@protoc_insertion_point(field_set:tutorial.Person.id)
}

// optional string email = 3;
```

(continues on next page)

(continued from previous page)

```
inline bool Person::_internal_has_email() const {
  bool value = (_has_bits_[0] & 0x00000002u) != 0;
  return value;
}
inline bool Person::has_email() const {
  return _internal_has_email();
}
inline void Person::clear_email() {
  email_.ClearToEmpty();
  _has_bits_[0] &= ~0x00000002u;
}
inline const std::string& Person::email() const {
  // @@protoc_insertion_point(field_get:tutorial.Person.email)
  return _internal_email();
}
template <typename ArgT0, typename... ArgT>
inline PROTOBUF_ALWAYS_INLINE
void Person::set_email(ArgT0&& arg0, ArgT... args) {
 _has_bits_[0] |= 0x00000002u;
 email_.Set(static_cast<ArgT0 &&>(arg0), args..., GetArenaForAllocation());
  // @@protoc_insertion_point(field_set:tutorial.Person.email)
}
inline std::string* Person::mutable_email() {
  std::string* _s = _internal_mutable_email();
  // @@protoc_insertion_point(field_mutable:tutorial.Person.email)
  return _s;
}
inline const std::string& Person::_internal_email() const {
  return email_.Get();
}
inline void Person::_internal_set_email(const std::string& value) {
  _has_bits_[0] |= 0x00000002u;
  email_.Set(value, GetArenaForAllocation());
}
inline std::string* Person::_internal_mutable_email() {
  _has_bits_[0] |= 0x00000002u;
  return email_.Mutable(GetArenaForAllocation());
}
inline std::string* Person::release_email() {
  // @@protoc_insertion_point(field_release:tutorial.Person.email)
  if (!_internal_has_email()) {
    return nullptr;
  }
  _has_bits_[0] &= ~0x00000002u;
  auto* p = email_.Release();
#ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
  if (email_.IsDefault()) {
    email_.Set("", GetArenaForAllocation());
  }
#endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
  return p;
}
```

(continues on next page)

(continued from previous page)

```
inline void Person::set_allocated_email(std::string* email) {
  if (email != nullptr) {
    _has_bits_[0] |= 0x00000002u;
  } else {
    _has_bits_[0] &= ~0x00000002u;
  }
  email_.SetAllocated(email, GetArenaForAllocation());
#ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
  if (email_.IsDefault()) {
    email_.Set("", GetArenaForAllocation());
  }
#endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
  // @@protoc_insertion_point(field_set_allocated:tutorial.Person.email)
}

// repeated .tutorial.Person.PhoneNumber phones = 4;
inline int Person::_internal_phones_size() const {
  return phones_.size();
}
inline int Person::phones_size() const {
  return _internal_phones_size();
}
inline void Person::clear_phones() {
  phones_.Clear();
}
inline ::tutorial::Person_PhoneNumber* Person::mutable_phones(int index) {
  // @@protoc_insertion_point(field_mutable:tutorial.Person.phones)
  return phones_.Mutable(index);
}
inline ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person_PhoneNumber >*
Person::mutable_phones() {
  // @@protoc_insertion_point(field_mutable_list:tutorial.Person.phones)
  return &phones_;
}
inline const ::tutorial::Person_PhoneNumber& Person::_internal_phones(int index) const {
  return phones_.Get(index);
}
inline const ::tutorial::Person_PhoneNumber& Person::phones(int index) const {
  // @@protoc_insertion_point(field_get:tutorial.Person.phones)
  return _internal_phones(index);
}
inline ::tutorial::Person_PhoneNumber* Person::_internal_add_phones() {
  return phones_.Add();
}
inline ::tutorial::Person_PhoneNumber* Person::add_phones() {
  ::tutorial::Person_PhoneNumber* _add = _internal_add_phones();
  // @@protoc_insertion_point(field_add:tutorial.Person.phones)
  return _add;
}
inline const ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person_PhoneNumber >&
Person::phones() const {
  // @@protoc_insertion_point(field_list:tutorial.Person.phones)
```

(continues on next page)

(continued from previous page)

```
  return phones_;
}

// -------------------------------------------------------------------

// AddressBook

// repeated .tutorial.Person people = 1;
inline int AddressBook::_internal_people_size() const {
  return people_.size();
}
inline int AddressBook::people_size() const {
  return _internal_people_size();
}
inline void AddressBook::clear_people() {
  people_.Clear();
}
inline ::tutorial::Person* AddressBook::mutable_people(int index) {
  // @@protoc_insertion_point(field_mutable:tutorial.AddressBook.people)
  return people_.Mutable(index);
}
inline ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person >*
AddressBook::mutable_people() {
  // @@protoc_insertion_point(field_mutable_list:tutorial.AddressBook.people)
  return &people_;
}
inline const ::tutorial::Person& AddressBook::_internal_people(int index) const {
  return people_.Get(index);
}
inline const ::tutorial::Person& AddressBook::people(int index) const {
  // @@protoc_insertion_point(field_get:tutorial.AddressBook.people)
  return _internal_people(index);
}
inline ::tutorial::Person* AddressBook::_internal_add_people() {
  return people_.Add();
}
inline ::tutorial::Person* AddressBook::add_people() {
  ::tutorial::Person* _add = _internal_add_people();
  // @@protoc_insertion_point(field_add:tutorial.AddressBook.people)
  return _add;
}
inline const ::PROTOBUF_NAMESPACE_ID::RepeatedPtrField< ::tutorial::Person >&
AddressBook::people() const {
  // @@protoc_insertion_point(field_list:tutorial.AddressBook.people)
  return people_;
}

#ifdef __GNUC__
  #pragma GCC diagnostic pop
#endif  // __GNUC__
// -------------------------------------------------------------------

```

(continues on next page)

(continued from previous page)

```
// -------------------------------------------------------------------


// @@protoc_insertion_point(namespace_scope)

}  // namespace tutorial

PROTOBUF_NAMESPACE_OPEN

template <> struct is_proto_enum< ::tutorial::Person_PhoneType> : ::std::true_type {};
template <>
inline const EnumDescriptor* GetEnumDescriptor< ::tutorial::Person_PhoneType>() {
  return ::tutorial::Person_PhoneType_descriptor();
}

PROTOBUF_NAMESPACE_CLOSE

// @@protoc_insertion_point(global_scope)

#include <google/protobuf/port_undef.inc>
#endif  // GOOGLE_PROTOBUF_INCLUDED_GOOGLE_PROTOBUF_INCLUDED_hello_2eproto
```

15.2.4 hello.pb.cc

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

```
// Generated by the protocol buffer compiler.  DO NOT EDIT!
// source: hello.proto

#include "hello.pb.h"

#include <algorithm>

#include <google/protobuf/io/coded_stream.h>
#include <google/protobuf/extension_set.h>
#include <google/protobuf/wire_format_lite.h>
#include <google/protobuf/descriptor.h>
#include <google/protobuf/generated_message_reflection.h>
#include <google/protobuf/reflection_ops.h>
#include <google/protobuf/wire_format.h>
// @@protoc_insertion_point(includes)
#include <google/protobuf/port_def.inc>

PROTOBUF_PRAGMA_INIT_SEG

namespace _pb = ::PROTOBUF_NAMESPACE_ID;
namespace _pbi = _pb::internal;

namespace tutorial {
PROTOBUF_CONSTEXPR Person_PhoneNumber::Person_PhoneNumber(
```

(continues on next page)

(continued from previous page)

```
    ::_pbi::ConstantInitialized)
  : number_(&::_pbi::fixed_address_empty_string, ::_pbi::ConstantInitialized{})
  , type_(1)
{}
struct Person_PhoneNumberDefaultTypeInternal {
  PROTOBUF_CONSTEXPR Person_PhoneNumberDefaultTypeInternal()
      : _instance(::_pbi::ConstantInitialized{}) {}
  ~Person_PhoneNumberDefaultTypeInternal() {}
  union {
    Person_PhoneNumber _instance;
  };
};
PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT PROTOBUF_ATTRIBUTE_INIT_PRIORITY1␣
↪Person_PhoneNumberDefaultTypeInternal _Person_PhoneNumber_default_instance_;
PROTOBUF_CONSTEXPR Person::Person(
    ::_pbi::ConstantInitialized)
  : phones_()
  , name_(&::_pbi::fixed_address_empty_string, ::_pbi::ConstantInitialized{})
  , email_(&::_pbi::fixed_address_empty_string, ::_pbi::ConstantInitialized{})
  , id_(0){}
struct PersonDefaultTypeInternal {
  PROTOBUF_CONSTEXPR PersonDefaultTypeInternal()
      : _instance(::_pbi::ConstantInitialized{}) {}
  ~PersonDefaultTypeInternal() {}
  union {
    Person _instance;
  };
};
PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT PROTOBUF_ATTRIBUTE_INIT_PRIORITY1␣
↪PersonDefaultTypeInternal _Person_default_instance_;
PROTOBUF_CONSTEXPR AddressBook::AddressBook(
    ::_pbi::ConstantInitialized)
  : people_(){}
struct AddressBookDefaultTypeInternal {
  PROTOBUF_CONSTEXPR AddressBookDefaultTypeInternal()
      : _instance(::_pbi::ConstantInitialized{}) {}
  ~AddressBookDefaultTypeInternal() {}
  union {
    AddressBook _instance;
  };
};
PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT PROTOBUF_ATTRIBUTE_INIT_PRIORITY1␣
↪AddressBookDefaultTypeInternal _AddressBook_default_instance_;
}  // namespace tutorial
static ::_pb::Metadata file_level_metadata_hello_2eproto[3];
static const ::_pb::EnumDescriptor* file_level_enum_descriptors_hello_2eproto[1];
static constexpr ::_pb::ServiceDescriptor const** file_level_service_descriptors_hello_
↪2eproto = nullptr;

const uint32_t TableStruct_hello_2eproto::offsets[] PROTOBUF_SECTION_VARIABLE(protodesc_
↪cold) = {
  PROTOBUF_FIELD_OFFSET(::tutorial::Person_PhoneNumber, _has_bits_),
```

(continues on next page)

(continued from previous page)

```
  PROTOBUF_FIELD_OFFSET(::tutorial::Person_PhoneNumber, _internal_metadata_),
  ~0u,  // no _extensions_
  ~0u,  // no _oneof_case_
  ~0u,  // no _weak_field_map_
  ~0u,  // no _inlined_string_donated_
  PROTOBUF_FIELD_OFFSET(::tutorial::Person_PhoneNumber, number_),
  PROTOBUF_FIELD_OFFSET(::tutorial::Person_PhoneNumber, type_),
  0,
  1,
  PROTOBUF_FIELD_OFFSET(::tutorial::Person, _has_bits_),
  PROTOBUF_FIELD_OFFSET(::tutorial::Person, _internal_metadata_),
  ~0u,  // no _extensions_
  ~0u,  // no _oneof_case_
  ~0u,  // no _weak_field_map_
  ~0u,  // no _inlined_string_donated_
  PROTOBUF_FIELD_OFFSET(::tutorial::Person, name_),
  PROTOBUF_FIELD_OFFSET(::tutorial::Person, id_),
  PROTOBUF_FIELD_OFFSET(::tutorial::Person, email_),
  PROTOBUF_FIELD_OFFSET(::tutorial::Person, phones_),
  0,
  2,
  1,
  ~0u,
  ~0u,  // no _has_bits_
  PROTOBUF_FIELD_OFFSET(::tutorial::AddressBook, _internal_metadata_),
  ~0u,  // no _extensions_
  ~0u,  // no _oneof_case_
  ~0u,  // no _weak_field_map_
  ~0u,  // no _inlined_string_donated_
  PROTOBUF_FIELD_OFFSET(::tutorial::AddressBook, people_),
};
static const ::_pbi::MigrationSchema schemas[] PROTOBUF_SECTION_VARIABLE(protodesc_cold)␣
↪= {
  { 0, 8, -1, sizeof(::tutorial::Person_PhoneNumber)},
  { 10, 20, -1, sizeof(::tutorial::Person)},
  { 24, -1, -1, sizeof(::tutorial::AddressBook)},
};

static const ::_pb::Message* const file_default_instances[] = {
  &::tutorial::_Person_PhoneNumber_default_instance_._instance,
  &::tutorial::_Person_default_instance_._instance,
  &::tutorial::_AddressBook_default_instance_._instance,
};

const char descriptor_table_protodef_hello_2eproto[] PROTOBUF_SECTION_VARIABLE(protodesc_
↪cold) =
  "\n\013hello.proto\022\010tutorial\"\333\001\n\006Person\022\014\n\004na"
  "me\030\001 \001(\t\022\n\n\002id\030\002 \001(\005\022\r\n\005email\030\003 \001(\t\
↪022,\n\006p"
  "hones\030\004 \003(\0132\034.tutorial.Person.PhoneNumbe"
  "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)

```
  "\"+\n\tPhoneType\022\n\n\006MOBILE\020\000\022\010\n\004HOME\020\001\022\010\n\004W"
  "ORK\020\002\"/\n\013AddressBook\022 \n\006people\030\001 \003(\0132\020.t"
  "utorial.Person"
  ;
static ::_pbi::once_flag descriptor_table_hello_2eproto_once;
const ::_pbi::DescriptorTable descriptor_table_hello_2eproto = {
    false, false, 294, descriptor_table_protodef_hello_2eproto,
    "hello.proto",
    &descriptor_table_hello_2eproto_once, nullptr, 0, 3,
    schemas, file_default_instances, TableStruct_hello_2eproto::offsets,
    file_level_metadata_hello_2eproto, file_level_enum_descriptors_hello_2eproto,
    file_level_service_descriptors_hello_2eproto,
};
PROTOBUF_ATTRIBUTE_WEAK const ::_pbi::DescriptorTable* descriptor_table_hello_2eproto_
↪getter() {
  return &descriptor_table_hello_2eproto;
}

// Force running AddDescriptors() at dynamic initialization time.
PROTOBUF_ATTRIBUTE_INIT_PRIORITY2 static ::_pbi::AddDescriptorsRunner dynamic_init_dummy_
↪hello_2eproto(&descriptor_table_hello_2eproto);
namespace tutorial {
const ::PROTOBUF_NAMESPACE_ID::EnumDescriptor* Person_PhoneType_descriptor() {
  ::PROTOBUF_NAMESPACE_ID::internal::AssignDescriptors(&descriptor_table_hello_2eproto);
  return file_level_enum_descriptors_hello_2eproto[0];
}
bool Person_PhoneType_IsValid(int value) {
  switch (value) {
    case 0:
    case 1:
    case 2:
      return true;
    default:
      return false;
  }
}

#if (__cplusplus < 201703) && (!defined(_MSC_VER) || (_MSC_VER >= 1900 && _MSC_VER <␣
↪1912))
constexpr Person_PhoneType Person::MOBILE;
constexpr Person_PhoneType Person::HOME;
constexpr Person_PhoneType Person::WORK;
constexpr Person_PhoneType Person::PhoneType_MIN;
constexpr Person_PhoneType Person::PhoneType_MAX;
constexpr int Person::PhoneType_ARRAYSIZE;
#endif  // (__cplusplus < 201703) && (!defined(_MSC_VER) || (_MSC_VER >= 1900 && _MSC_
↪VER < 1912))

// ===================================================================

class Person_PhoneNumber::_Internal {
 public:
```

(continues on next page)

(continued from previous page)

```
  using HasBits = decltype(std::declval<Person_PhoneNumber>()._has_bits_);
  static void set_has_number(HasBits* has_bits) {
    (*has_bits)[0] |= 1u;
  }
  static void set_has_type(HasBits* has_bits) {
    (*has_bits)[0] |= 2u;
  }
};

Person_PhoneNumber::Person_PhoneNumber(::PROTOBUF_NAMESPACE_ID::Arena* arena,
                         bool is_message_owned)
  : ::PROTOBUF_NAMESPACE_ID::Message(arena, is_message_owned) {
  SharedCtor();
  // @@protoc_insertion_point(arena_constructor:tutorial.Person.PhoneNumber)
}
Person_PhoneNumber::Person_PhoneNumber(const Person_PhoneNumber& from)
  : ::PROTOBUF_NAMESPACE_ID::Message(),
      _has_bits_(from._has_bits_) {
  _internal_metadata_.MergeFrom<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
↪metadata_);
  number_.InitDefault();
  #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
    number_.Set("", GetArenaForAllocation());
  #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
  if (from._internal_has_number()) {
    number_.Set(from._internal_number(),
      GetArenaForAllocation());
  }
  type_ = from.type_;
  // @@protoc_insertion_point(copy_constructor:tutorial.Person.PhoneNumber)
}

inline void Person_PhoneNumber::SharedCtor() {
number_.InitDefault();
#ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
  number_.Set("", GetArenaForAllocation());
#endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
type_ = 1;
}

Person_PhoneNumber::~Person_PhoneNumber() {
  // @@protoc_insertion_point(destructor:tutorial.Person.PhoneNumber)
  if (auto *arena = _internal_metadata_.DeleteReturnArena<::PROTOBUF_NAMESPACE_
↪ID::UnknownFieldSet>()) {
  (void)arena;
    return;
  }
  SharedDtor();
}

inline void Person_PhoneNumber::SharedDtor() {
  GOOGLE_DCHECK(GetArenaForAllocation() == nullptr);
```

(continues on next page)

(continued from previous page)

```
  number_.Destroy();
}

void Person_PhoneNumber::SetCachedSize(int size) const {
  _cached_size_.Set(size);
}

void Person_PhoneNumber::Clear() {
// @@protoc_insertion_point(message_clear_start:tutorial.Person.PhoneNumber)
  uint32_t cached_has_bits = 0;
  // Prevent compiler warnings about cached_has_bits being unused
  (void) cached_has_bits;

  cached_has_bits = _has_bits_[0];
  if (cached_has_bits & 0x00000003u) {
    if (cached_has_bits & 0x00000001u) {
      number_.ClearNonDefaultToEmpty();
    }
    type_ = 1;
  }
  _has_bits_.Clear();
  _internal_metadata_.Clear<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>();
}

const char* Person_PhoneNumber::_InternalParse(const char* ptr, ::_pbi::ParseContext*
↪ctx) {
#define CHK_(x) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
  _Internal::HasBits has_bits{};
  while (!ctx->Done(&ptr)) {
    uint32_t tag;
    ptr = ::_pbi::ReadTag(ptr, &tag);
    switch (tag >> 3) {
      // optional string number = 1;
      case 1:
        if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 10)) {
          auto str = _internal_mutable_number();
          ptr = ::_pbi::InlineGreedyStringParser(str, ptr, ctx);
          CHK_(ptr);
          #ifndef NDEBUG
          ::_pbi::VerifyUTF8(str, "tutorial.Person.PhoneNumber.number");
          #endif  // !NDEBUG
        } else
          goto handle_unusual;
        continue;
      // optional .tutorial.Person.PhoneType type = 2 [default = HOME];
      case 2:
        if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 16)) {
          uint64_t val = ::PROTOBUF_NAMESPACE_ID::internal::ReadVarint64(&ptr);
          CHK_(ptr);
          if (PROTOBUF_PREDICT_TRUE(::tutorial::Person_PhoneType_IsValid(val))) {
            _internal_set_type(static_cast<::tutorial::Person_PhoneType>(val));
          } else {
```

(continues on next page)

(continued from previous page)

```
            ::PROTOBUF_NAMESPACE_ID::internal::WriteVarint(2, val, mutable_unknown_
↪fields());
          }
        } else
          goto handle_unusual;
        continue;
      default:
        goto handle_unusual;
    }  // switch
  handle_unusual:
    if ((tag == 0) || ((tag & 7) == 4)) {
      CHK_(ptr);
      ctx->SetLastTag(tag);
      goto message_done;
    }
    ptr = UnknownFieldParse(
        tag,
        _internal_metadata_.mutable_unknown_fields<::PROTOBUF_NAMESPACE_
↪ID::UnknownFieldSet>(),
        ptr, ctx);
    CHK_(ptr != nullptr);
  }  // while
message_done:
  _has_bits_.Or(has_bits);
  return ptr;
failure:
  ptr = nullptr;
  goto message_done;
#undef CHK_
}

uint8_t* Person_PhoneNumber::_InternalSerialize(
    uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const {
  // @@protoc_insertion_point(serialize_to_array_start:tutorial.Person.PhoneNumber)
  uint32_t cached_has_bits = 0;
  (void) cached_has_bits;

  cached_has_bits = _has_bits_[0];
  // optional string number = 1;
  if (cached_has_bits & 0x00000001u) {
    ::PROTOBUF_NAMESPACE_ID::internal::WireFormat::VerifyUTF8StringNamedField(
      this->_internal_number().data(), static_cast<int>(this->_internal_number().
↪length()),
      ::PROTOBUF_NAMESPACE_ID::internal::WireFormat::SERIALIZE,
      "tutorial.Person.PhoneNumber.number");
    target = stream->WriteStringMaybeAliased(
        1, this->_internal_number(), target);
  }

  // optional .tutorial.Person.PhoneType type = 2 [default = HOME];
  if (cached_has_bits & 0x00000002u) {
    target = stream->EnsureSpace(target);
```

(continues on next page)

(continued from previous page)

```
    target = ::_pbi::WireFormatLite::WriteEnumToArray(
      2, this->_internal_type(), target);
  }

  if (PROTOBUF_PREDICT_FALSE(_internal_metadata_.have_unknown_fields())) {
    target = ::_pbi::WireFormat::InternalSerializeUnknownFieldsToArray(
        _internal_metadata_.unknown_fields<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
↪(::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance), target, stream);
  }
  // @@protoc_insertion_point(serialize_to_array_end:tutorial.Person.PhoneNumber)
  return target;
}

size_t Person_PhoneNumber::ByteSizeLong() const {
// @@protoc_insertion_point(message_byte_size_start:tutorial.Person.PhoneNumber)
  size_t total_size = 0;

  uint32_t cached_has_bits = 0;
  // Prevent compiler warnings about cached_has_bits being unused
  (void) cached_has_bits;

  cached_has_bits = _has_bits_[0];
  if (cached_has_bits & 0x00000003u) {
    // optional string number = 1;
    if (cached_has_bits & 0x00000001u) {
      total_size += 1 +
        ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::StringSize(
          this->_internal_number());
    }

    // optional .tutorial.Person.PhoneType type = 2 [default = HOME];
    if (cached_has_bits & 0x00000002u) {
      total_size += 1 +
        ::_pbi::WireFormatLite::EnumSize(this->_internal_type());
    }

  }
  return MaybeComputeUnknownFieldsSize(total_size, &_cached_size_);
}

const ::PROTOBUF_NAMESPACE_ID::Message::ClassData Person_PhoneNumber::_class_data_ = {
    ::PROTOBUF_NAMESPACE_ID::Message::CopyWithSizeCheck,
    Person_PhoneNumber::MergeImpl
};
const ::PROTOBUF_NAMESPACE_ID::Message::ClassData*Person_PhoneNumber::GetClassData()␣
↪const { return &_class_data_; }

void Person_PhoneNumber::MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to,
                      const ::PROTOBUF_NAMESPACE_ID::Message& from) {
  static_cast<Person_PhoneNumber *>(to)->MergeFrom(
      static_cast<const Person_PhoneNumber &>(from));
}
```

(continues on next page)

(continued from previous page)

```cpp


void Person_PhoneNumber::MergeFrom(const Person_PhoneNumber& from) {
// @@protoc_insertion_point(class_specific_merge_from_start:tutorial.Person.PhoneNumber)
  GOOGLE_DCHECK_NE(&from, this);
  uint32_t cached_has_bits = 0;
  (void) cached_has_bits;

  cached_has_bits = from._has_bits_[0];
  if (cached_has_bits & 0x00000003u) {
    if (cached_has_bits & 0x00000001u) {
      _internal_set_number(from._internal_number());
    }
    if (cached_has_bits & 0x00000002u) {
      type_ = from.type_;
    }
    _has_bits_[0] |= cached_has_bits;
  }
  _internal_metadata_.MergeFrom<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
↪metadata_);
}

void Person_PhoneNumber::CopyFrom(const Person_PhoneNumber& from) {
// @@protoc_insertion_point(class_specific_copy_from_start:tutorial.Person.PhoneNumber)
  if (&from == this) return;
  Clear();
  MergeFrom(from);
}

bool Person_PhoneNumber::IsInitialized() const {
  return true;
}

void Person_PhoneNumber::InternalSwap(Person_PhoneNumber* other) {
  using std::swap;
  auto* lhs_arena = GetArenaForAllocation();
  auto* rhs_arena = other->GetArenaForAllocation();
  _internal_metadata_.InternalSwap(&other->_internal_metadata_);
  swap(_has_bits_[0], other->_has_bits_[0]);
  ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr::InternalSwap(
      &number_, lhs_arena,
      &other->number_, rhs_arena
  );
  swap(type_, other->type_);
}

::PROTOBUF_NAMESPACE_ID::Metadata Person_PhoneNumber::GetMetadata() const {
  return ::_pbi::AssignDescriptors(
      &descriptor_table_hello_2eproto_getter, &descriptor_table_hello_2eproto_once,
      file_level_metadata_hello_2eproto[0]);
}


```

(continues on next page)

(continued from previous page)

```
// ===================================================================

class Person::_Internal {
 public:
  using HasBits = decltype(std::declval<Person>()._has_bits_);
  static void set_has_name(HasBits* has_bits) {
    (*has_bits)[0] |= 1u;
  }
  static void set_has_id(HasBits* has_bits) {
    (*has_bits)[0] |= 4u;
  }
  static void set_has_email(HasBits* has_bits) {
    (*has_bits)[0] |= 2u;
  }
};

Person::Person(::PROTOBUF_NAMESPACE_ID::Arena* arena,
                         bool is_message_owned)
  : ::PROTOBUF_NAMESPACE_ID::Message(arena, is_message_owned),
  phones_(arena) {
  SharedCtor();
  // @@protoc_insertion_point(arena_constructor:tutorial.Person)
}
Person::Person(const Person& from)
  : ::PROTOBUF_NAMESPACE_ID::Message(),
      _has_bits_(from._has_bits_),
      phones_(from.phones_) {
  _internal_metadata_.MergeFrom<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
↪metadata_);
  name_.InitDefault();
  #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
    name_.Set("", GetArenaForAllocation());
  #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
  if (from._internal_has_name()) {
    name_.Set(from._internal_name(),
      GetArenaForAllocation());
  }
  email_.InitDefault();
  #ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
    email_.Set("", GetArenaForAllocation());
  #endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
  if (from._internal_has_email()) {
    email_.Set(from._internal_email(),
      GetArenaForAllocation());
  }
  id_ = from.id_;
  // @@protoc_insertion_point(copy_constructor:tutorial.Person)
}

inline void Person::SharedCtor() {
name_.InitDefault();
#ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
```

(continues on next page)

(continued from previous page)

```
  name_.Set("", GetArenaForAllocation());
#endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
email_.InitDefault();
#ifdef PROTOBUF_FORCE_COPY_DEFAULT_STRING
  email_.Set("", GetArenaForAllocation());
#endif // PROTOBUF_FORCE_COPY_DEFAULT_STRING
id_ = 0;
}

Person::~Person() {
  // @@protoc_insertion_point(destructor:tutorial.Person)
  if (auto *arena = _internal_metadata_.DeleteReturnArena<::PROTOBUF_NAMESPACE_
↪ID::UnknownFieldSet>()) {
  (void)arena;
    return;
  }
  SharedDtor();
}

inline void Person::SharedDtor() {
  GOOGLE_DCHECK(GetArenaForAllocation() == nullptr);
  name_.Destroy();
  email_.Destroy();
}

void Person::SetCachedSize(int size) const {
  _cached_size_.Set(size);
}

void Person::Clear() {
// @@protoc_insertion_point(message_clear_start:tutorial.Person)
  uint32_t cached_has_bits = 0;
  // Prevent compiler warnings about cached_has_bits being unused
  (void) cached_has_bits;

  phones_.Clear();
  cached_has_bits = _has_bits_[0];
  if (cached_has_bits & 0x00000003u) {
    if (cached_has_bits & 0x00000001u) {
      name_.ClearNonDefaultToEmpty();
    }
    if (cached_has_bits & 0x00000002u) {
      email_.ClearNonDefaultToEmpty();
    }
  }
  id_ = 0;
  _has_bits_.Clear();
  _internal_metadata_.Clear<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>();
}

const char* Person::_InternalParse(const char* ptr, ::_pbi::ParseContext* ctx) {
#define CHK_(x) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
```

(continues on next page)

(continued from previous page)

```
  _Internal::HasBits has_bits{};
  while (!ctx->Done(&ptr)) {
    uint32_t tag;
    ptr = ::_pbi::ReadTag(ptr, &tag);
    switch (tag >> 3) {
      // optional string name = 1;
      case 1:
        if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 10)) {
          auto str = _internal_mutable_name();
          ptr = ::_pbi::InlineGreedyStringParser(str, ptr, ctx);
          CHK_(ptr);
          #ifndef NDEBUG
          ::_pbi::VerifyUTF8(str, "tutorial.Person.name");
          #endif  // !NDEBUG
        } else
          goto handle_unusual;
        continue;
      // optional int32 id = 2;
      case 2:
        if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 16)) {
          _Internal::set_has_id(&has_bits);
          id_ = ::PROTOBUF_NAMESPACE_ID::internal::ReadVarint32(&ptr);
          CHK_(ptr);
        } else
          goto handle_unusual;
        continue;
      // optional string email = 3;
      case 3:
        if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 26)) {
          auto str = _internal_mutable_email();
          ptr = ::_pbi::InlineGreedyStringParser(str, ptr, ctx);
          CHK_(ptr);
          #ifndef NDEBUG
          ::_pbi::VerifyUTF8(str, "tutorial.Person.email");
          #endif  // !NDEBUG
        } else
          goto handle_unusual;
        continue;
      // repeated .tutorial.Person.PhoneNumber phones = 4;
      case 4:
        if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 34)) {
          ptr -= 1;
          do {
            ptr += 1;
            ptr = ctx->ParseMessage(_internal_add_phones(), ptr);
            CHK_(ptr);
            if (!ctx->DataAvailable(ptr)) break;
          } while (::PROTOBUF_NAMESPACE_ID::internal::ExpectTag<34>(ptr));
        } else
          goto handle_unusual;
        continue;
      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 & 0x00000001u) {
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 & 0x00000004u) {
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 & 0x00000002u) {
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)

```
      "tutorial.Person.email");
    target = stream->WriteStringMaybeAliased(
        3, this->_internal_email(), target);
  }

  // repeated .tutorial.Person.PhoneNumber phones = 4;
  for (unsigned i = 0,
      n = static_cast<unsigned>(this->_internal_phones_size()); i < n; i++) {
    const auto& repfield = this->_internal_phones(i);
    target = ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::
        InternalWriteMessage(4, repfield, repfield.GetCachedSize(), target, stream);
  }

  if (PROTOBUF_PREDICT_FALSE(_internal_metadata_.have_unknown_fields())) {
    target = ::_pbi::WireFormat::InternalSerializeUnknownFieldsToArray(
        _internal_metadata_.unknown_fields<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
↪(::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance), target, stream);
  }
  // @@protoc_insertion_point(serialize_to_array_end:tutorial.Person)
  return target;
}

size_t Person::ByteSizeLong() const {
// @@protoc_insertion_point(message_byte_size_start:tutorial.Person)
  size_t total_size = 0;

  uint32_t cached_has_bits = 0;
  // Prevent compiler warnings about cached_has_bits being unused
  (void) cached_has_bits;

  // repeated .tutorial.Person.PhoneNumber phones = 4;
  total_size += 1UL * this->_internal_phones_size();
  for (const auto& msg : this->phones_) {
    total_size +=
      ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::MessageSize(msg);
  }

  cached_has_bits = _has_bits_[0];
  if (cached_has_bits & 0x00000007u) {
    // optional string name = 1;
    if (cached_has_bits & 0x00000001u) {
      total_size += 1 +
        ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::StringSize(
          this->_internal_name());
    }

    // optional string email = 3;
    if (cached_has_bits & 0x00000002u) {
      total_size += 1 +
        ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::StringSize(
          this->_internal_email());
    }
```

(continues on next page)

(continued from previous page)

```
    // optional int32 id = 2;
    if (cached_has_bits & 0x00000004u) {
      total_size += ::_pbi::WireFormatLite::Int32SizePlusOne(this->_internal_id());
    }

  }
  return MaybeComputeUnknownFieldsSize(total_size, &_cached_size_);
}

const ::PROTOBUF_NAMESPACE_ID::Message::ClassData Person::_class_data_ = {
    ::PROTOBUF_NAMESPACE_ID::Message::CopyWithSizeCheck,
    Person::MergeImpl
};
const ::PROTOBUF_NAMESPACE_ID::Message::ClassData*Person::GetClassData() const { return &
↪_class_data_; }

void Person::MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to,
                       const ::PROTOBUF_NAMESPACE_ID::Message& from) {
  static_cast<Person *>(to)->MergeFrom(
      static_cast<const Person &>(from));
}


void Person::MergeFrom(const Person& from) {
// @@protoc_insertion_point(class_specific_merge_from_start:tutorial.Person)
  GOOGLE_DCHECK_NE(&from, this);
  uint32_t cached_has_bits = 0;
  (void) cached_has_bits;

  phones_.MergeFrom(from.phones_);
  cached_has_bits = from._has_bits_[0];
  if (cached_has_bits & 0x00000007u) {
    if (cached_has_bits & 0x00000001u) {
      _internal_set_name(from._internal_name());
    }
    if (cached_has_bits & 0x00000002u) {
      _internal_set_email(from._internal_email());
    }
    if (cached_has_bits & 0x00000004u) {
      id_ = from.id_;
    }
    _has_bits_[0] |= cached_has_bits;
  }
  _internal_metadata_.MergeFrom<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
↪metadata_);
}

void Person::CopyFrom(const Person& from) {
// @@protoc_insertion_point(class_specific_copy_from_start:tutorial.Person)
  if (&from == this) return;
  Clear();
```

(continues on next page)

(continued from previous page)

```
  MergeFrom(from);
}

bool Person::IsInitialized() const {
  return true;
}

void Person::InternalSwap(Person* other) {
  using std::swap;
  auto* lhs_arena = GetArenaForAllocation();
  auto* rhs_arena = other->GetArenaForAllocation();
  _internal_metadata_.InternalSwap(&other->_internal_metadata_);
  swap(_has_bits_[0], other->_has_bits_[0]);
  phones_.InternalSwap(&other->phones_);
  ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr::InternalSwap(
      &name_, lhs_arena,
      &other->name_, rhs_arena
  );
  ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr::InternalSwap(
      &email_, lhs_arena,
      &other->email_, rhs_arena
  );
  swap(id_, other->id_);
}

::PROTOBUF_NAMESPACE_ID::Metadata Person::GetMetadata() const {
  return ::_pbi::AssignDescriptors(
      &descriptor_table_hello_2eproto_getter, &descriptor_table_hello_2eproto_once,
      file_level_metadata_hello_2eproto[1]);
}

// ===================================================================

class AddressBook::_Internal {
 public:
};

AddressBook::AddressBook(::PROTOBUF_NAMESPACE_ID::Arena* arena,
                         bool is_message_owned)
  : ::PROTOBUF_NAMESPACE_ID::Message(arena, is_message_owned),
  people_(arena) {
  SharedCtor();
  // @@protoc_insertion_point(arena_constructor:tutorial.AddressBook)
}
AddressBook::AddressBook(const AddressBook& from)
  : ::PROTOBUF_NAMESPACE_ID::Message(),
      people_(from.people_) {
  _internal_metadata_.MergeFrom<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>(from._internal_
↪metadata_);
  // @@protoc_insertion_point(copy_constructor:tutorial.AddressBook)
}

```

(continues on next page)

(continued from previous page)

```
inline void AddressBook::SharedCtor() {
}

AddressBook::~AddressBook() {
  // @@protoc_insertion_point(destructor:tutorial.AddressBook)
  if (auto *arena = _internal_metadata_.DeleteReturnArena<::PROTOBUF_NAMESPACE_
↪ID::UnknownFieldSet>()) {
  (void)arena;
    return;
  }
  SharedDtor();
}

inline void AddressBook::SharedDtor() {
  GOOGLE_DCHECK(GetArenaForAllocation() == nullptr);
}

void AddressBook::SetCachedSize(int size) const {
  _cached_size_.Set(size);
}

void AddressBook::Clear() {
// @@protoc_insertion_point(message_clear_start:tutorial.AddressBook)
  uint32_t cached_has_bits = 0;
  // Prevent compiler warnings about cached_has_bits being unused
  (void) cached_has_bits;

  people_.Clear();
  _internal_metadata_.Clear<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>();
}

const char* AddressBook::_InternalParse(const char* ptr, ::_pbi::ParseContext* ctx) {
#define CHK_(x) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
  while (!ctx->Done(&ptr)) {
    uint32_t tag;
    ptr = ::_pbi::ReadTag(ptr, &tag);
    switch (tag >> 3) {
      // repeated .tutorial.Person people = 1;
      case 1:
        if (PROTOBUF_PREDICT_TRUE(static_cast<uint8_t>(tag) == 10)) {
          ptr -= 1;
          do {
            ptr += 1;
            ptr = ctx->ParseMessage(_internal_add_people(), ptr);
            CHK_(ptr);
            if (!ctx->DataAvailable(ptr)) break;
          } while (::PROTOBUF_NAMESPACE_ID::internal::ExpectTag<10>(ptr));
        } else
          goto handle_unusual;
        continue;
      default:
        goto handle_unusual;
```

(continues on next page)

(continued from previous page)

```
    }  // switch
  handle_unusual:
    if ((tag == 0) || ((tag & 7) == 4)) {
      CHK_(ptr);
      ctx->SetLastTag(tag);
      goto message_done;
    }
    ptr = UnknownFieldParse(
        tag,
        _internal_metadata_.mutable_unknown_fields<::PROTOBUF_NAMESPACE_
↪ID::UnknownFieldSet>(),
        ptr, ctx);
    CHK_(ptr != nullptr);
  }  // while
message_done:
  return ptr;
failure:
  ptr = nullptr;
  goto message_done;
#undef CHK_
}

uint8_t* AddressBook::_InternalSerialize(
    uint8_t* target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream* stream) const {
  // @@protoc_insertion_point(serialize_to_array_start:tutorial.AddressBook)
  uint32_t cached_has_bits = 0;
  (void) cached_has_bits;

  // repeated .tutorial.Person people = 1;
  for (unsigned i = 0,
      n = static_cast<unsigned>(this->_internal_people_size()); i < n; i++) {
    const auto& repfield = this->_internal_people(i);
    target = ::PROTOBUF_NAMESPACE_ID::internal::WireFormatLite::
        InternalWriteMessage(1, repfield, repfield.GetCachedSize(), target, stream);
  }

  if (PROTOBUF_PREDICT_FALSE(_internal_metadata_.have_unknown_fields())) {
    target = ::_pbi::WireFormat::InternalSerializeUnknownFieldsToArray(
        _internal_metadata_.unknown_fields<::PROTOBUF_NAMESPACE_ID::UnknownFieldSet>
↪(::PROTOBUF_NAMESPACE_ID::UnknownFieldSet::default_instance), target, stream);
  }
  // @@protoc_insertion_point(serialize_to_array_end:tutorial.AddressBook)
  return target;
}

size_t AddressBook::ByteSizeLong() const {
// @@protoc_insertion_point(message_byte_size_start:tutorial.AddressBook)
  size_t total_size = 0;

  uint32_t cached_has_bits = 0;
  // Prevent compiler warnings about cached_has_bits being unused
  (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 {␣
    ↪return &_class_data_; }
894
895 void AddressBook::MergeImpl(::PROTOBUF_NAMESPACE_ID::Message* to,
896                       const ::PROTOBUF_NAMESPACE_ID::Message& from) {
897   static_cast<AddressBook *>(to)->MergeFrom(
898       static_cast<const AddressBook &>(from));
899 }
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_
    ↪metadata_);
910 }
911
912 void AddressBook::CopyFrom(const AddressBook& from) {
913 // @@protoc_insertion_point(class_specific_copy_from_start:tutorial.AddressBook)
914   if (&from == this) return;
915   Clear();
916   MergeFrom(from);
917 }
918
919 bool AddressBook::IsInitialized() const {
920   return true;
921 }
922
923 void AddressBook::InternalSwap(AddressBook* other) {
924   using std::swap;
925   _internal_metadata_.InternalSwap(&other->_internal_metadata_);
926   people_.InternalSwap(&other->people_);
927 }
```

(continues on next page)

(continued from previous page)

```

::PROTOBUF_NAMESPACE_ID::Metadata AddressBook::GetMetadata() const {
  return ::_pbi::AssignDescriptors(
      &descriptor_table_hello_2eproto_getter, &descriptor_table_hello_2eproto_once,
      file_level_metadata_hello_2eproto[2]);
}

// @@protoc_insertion_point(namespace_scope)
}  // namespace tutorial
PROTOBUF_NAMESPACE_OPEN
template<> PROTOBUF_NOINLINE ::tutorial::Person_PhoneNumber*
Arena::CreateMaybeMessage< ::tutorial::Person_PhoneNumber >(Arena* arena) {
  return Arena::CreateMessageInternal< ::tutorial::Person_PhoneNumber >(arena);
}
template<> PROTOBUF_NOINLINE ::tutorial::Person*
Arena::CreateMaybeMessage< ::tutorial::Person >(Arena* arena) {
  return Arena::CreateMessageInternal< ::tutorial::Person >(arena);
}
template<> PROTOBUF_NOINLINE ::tutorial::AddressBook*
Arena::CreateMaybeMessage< ::tutorial::AddressBook >(Arena* arena) {
  return Arena::CreateMessageInternal< ::tutorial::AddressBook >(arena);
}
PROTOBUF_NAMESPACE_CLOSE

// @@protoc_insertion_point(global_scope)
#include <google/protobuf/port_undef.inc>
```

CHAPTER

SIXTEEN

GRPC

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
```

CHAPTER

SEVENTEEN

LWN.NET

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

CHAPTER

EIGHTEEN

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?

CHAPTER

NINETEEN

ESPNET

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
```

CHAPTER

TWENTY

CMAKE

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
```

CHAPTER

TWENTYONE

HUGGINGFACE

21.1 spaces

21.1.1 Install client API

```
pip install huggingface_hub
```

```
(py38) kuangfangjun:t$ python3
Python 3.8.0 (default, Oct 28 2019, 16:14:01)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from huggingface_hub import hf_hub_download
>>> hf_hub_download(repo_id="google/pegasus-xsum", filename="config.json")
Downloading: 100%|_____________________________________________________________________________
↪_____________________| 1.39k/1.39k [00:00<00:00, 1.12MB/s]
'/root/fangjun/.cache/huggingface/hub/models--google--pegasus-xsum/snapshots/
↪a0aa5531c00f59a32a167b75130805098b046f9c/config.json'
>>>
```

21.1.2 gradio

- https://huggingface.co/spaces/alphacep/asr
- https://huggingface.co/spaces/jonatasgrosman/asr
- https://github.com/gradio-app/gradio/issues/1359
- https://huggingface.co/spaces/Gradio-Blocks/neon-tts-plugin-coqui/blob/main/app.py, css styles

CHAPTER

TWENTYTWO

EECS E6870 SPEECH RECOGNITION

22.1 Notes

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