notes

fangjun

May 16, 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
```

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

This page describes commonly used git commands.

8.1 DDP

8.1.1 Initialization

8.2 TorchScript

8.2.1 Hello

`torch.jit.script` as a decorator

Listing 1: ./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
```

(continues on next page)

(continued from previous page)

```
  %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 2: ./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))


"""
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 3: ./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()
```

(continues on next page)

(continued from previous page)

```
    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 4: ./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):
```

(continues on next page)

(continued from previous page)

```
        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])))


"""
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)
"""
```

8.2.2 Load in C++

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

Listing 5: ./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__))')

```

(continues on next page)

(continued from previous page)

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

Listing 6: ./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();
```

(continues on next page)

(continued from previous page)

```
  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 \
    -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} ]
```

CHAPTER

NINE

PYTHON

9.1 asyncio

9.1.1 Hello World

9.1.2 References

- PEP 234 – Iterators

  https://peps.python.org/pep-0234/

- Why does defining __getitem__ on a class make it iterable in python?

  https://localcoder.org/why-does-defining-getitem-on-a-class-make-it-iterable-in-python

- PEP 255 – Simple Generators

  https://peps.python.org/pep-0255/

- Curious Course on Coroutines and Concurrency

  https://www.youtube.com/watch?v=Z_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.

9.2 argv

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

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

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

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

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++.

11.2 TODOs

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

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