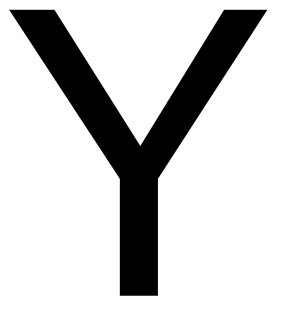
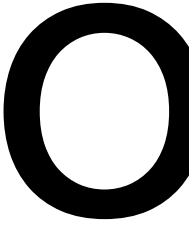
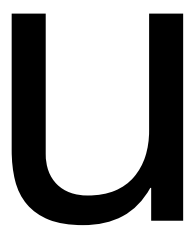
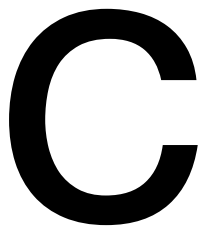
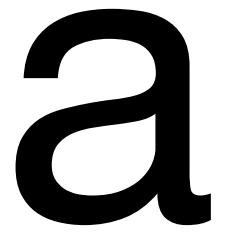
Alternatively Using Docker

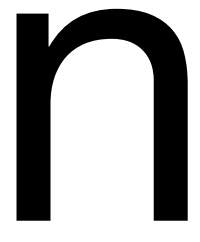


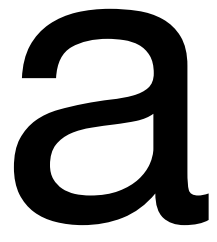


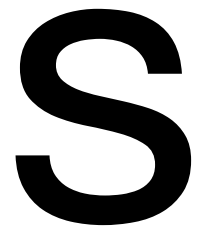


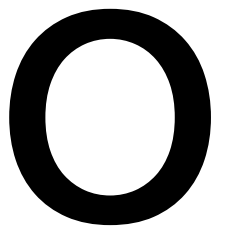

C

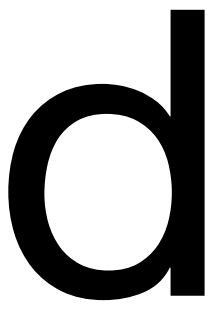


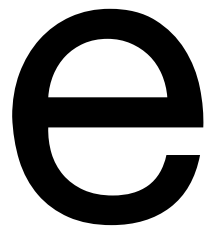


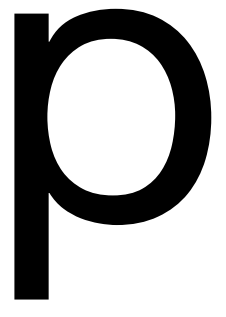


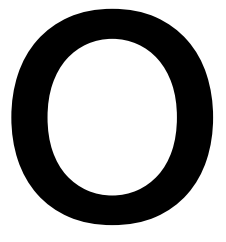


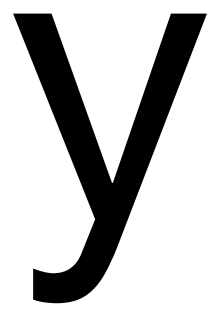


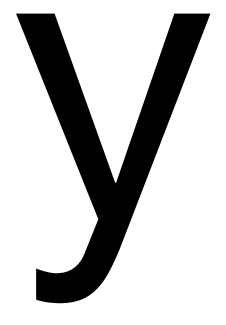


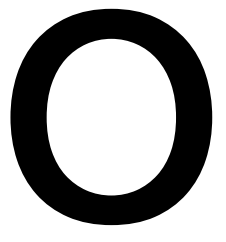


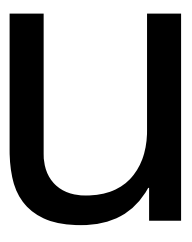


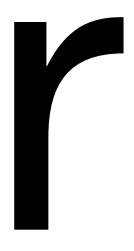


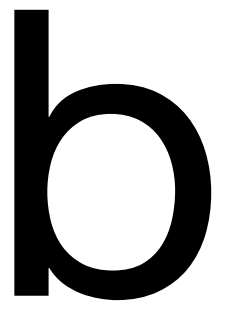


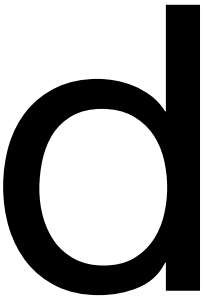


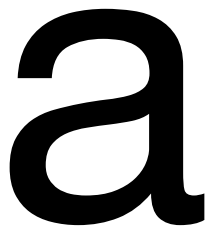


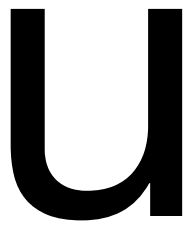


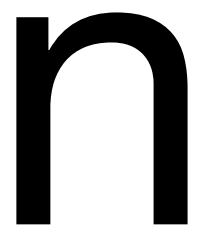


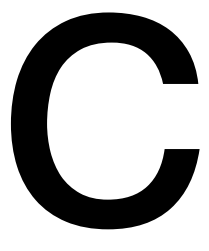

b

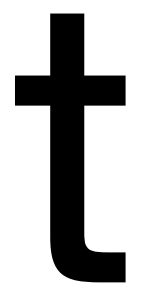


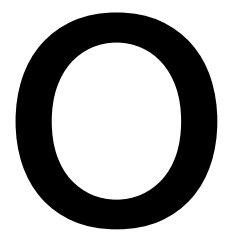


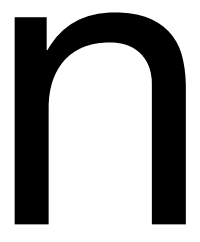


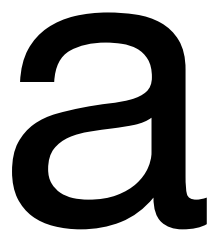


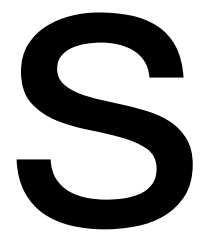

C

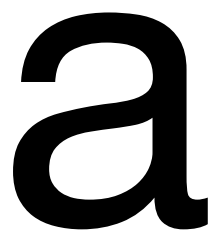


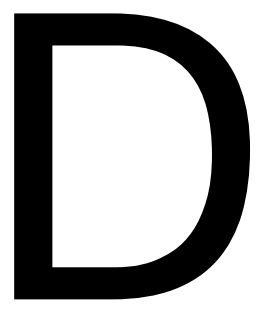


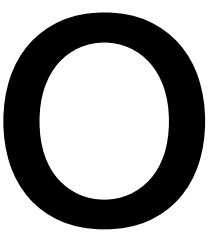


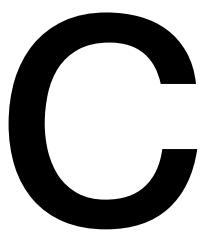


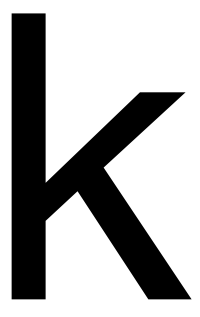


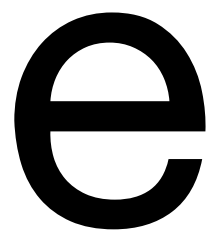


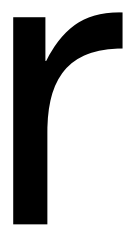


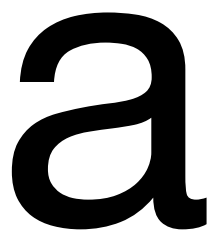


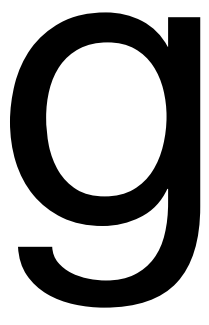

C

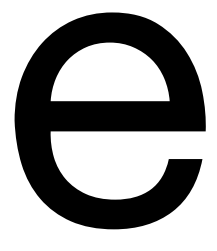


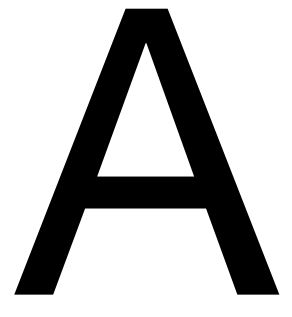


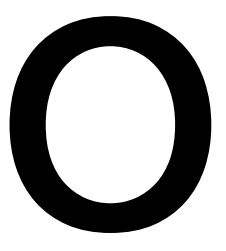


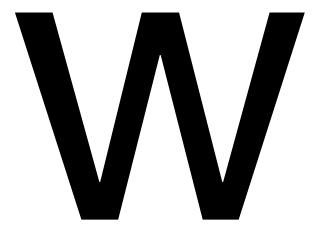


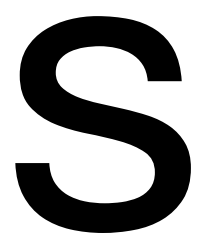


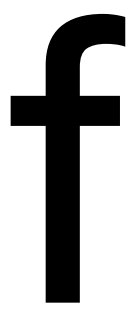


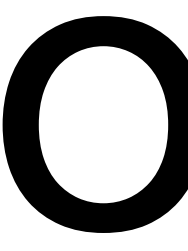


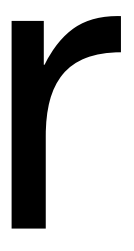


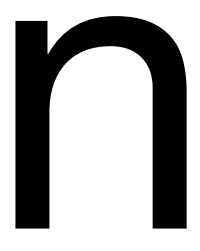


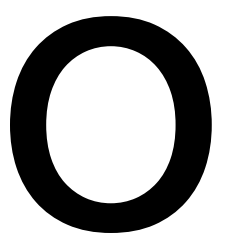


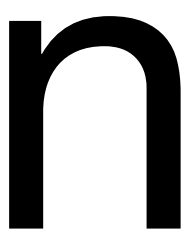


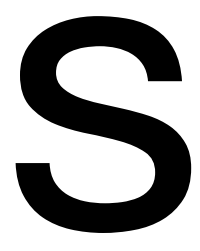


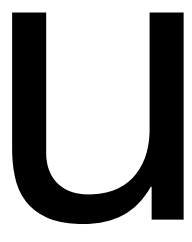


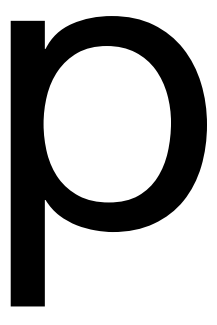


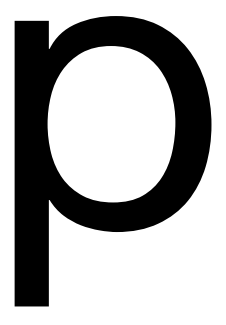


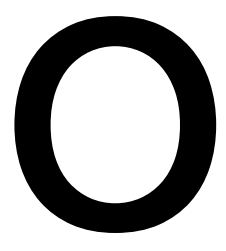


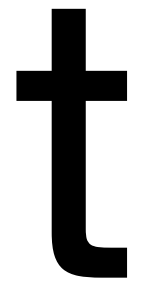

S

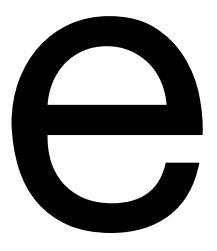


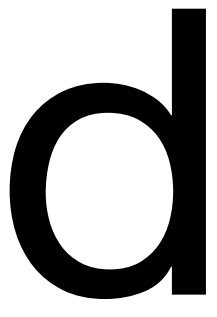


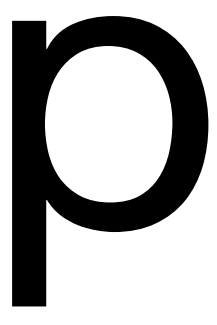


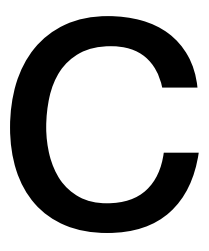


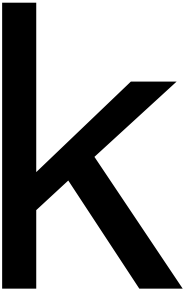


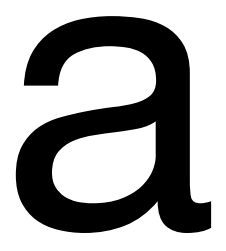


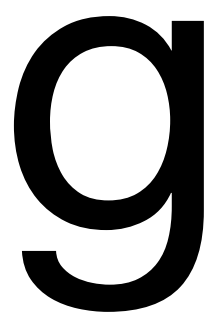


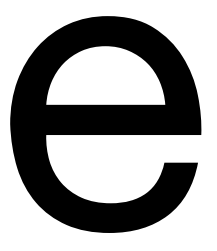


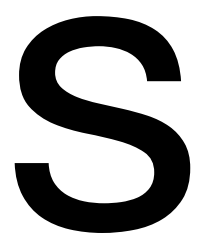

C

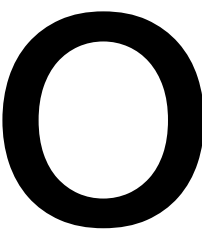


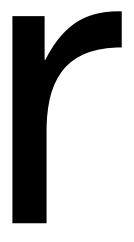


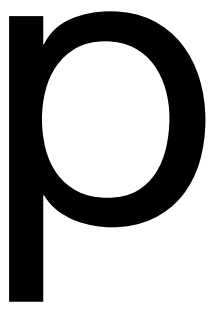


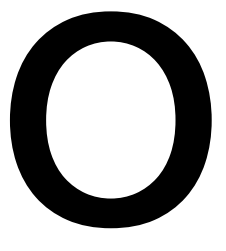


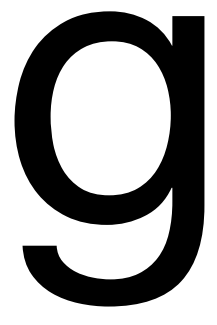


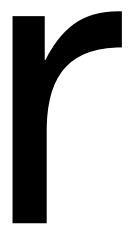


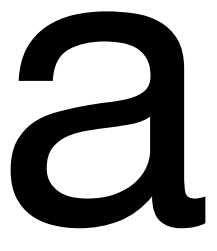


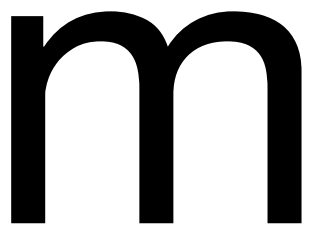


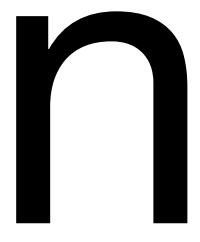


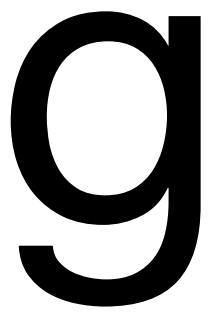


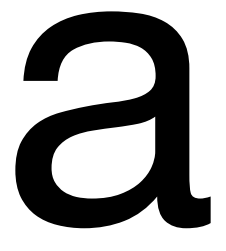


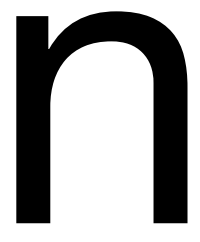


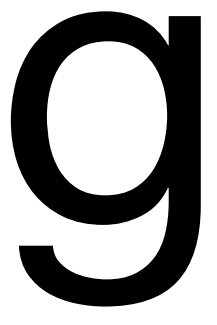


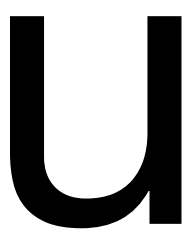


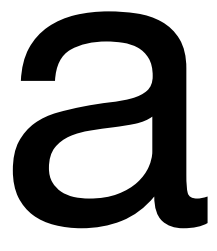


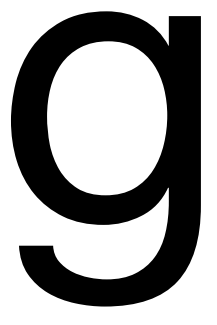


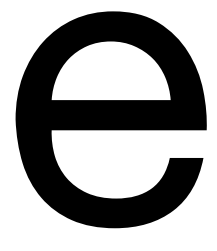


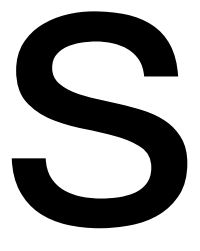


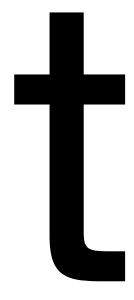


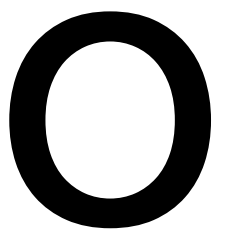


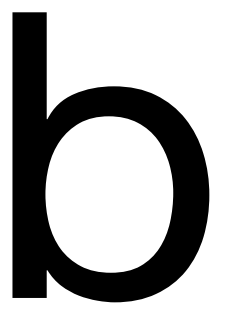


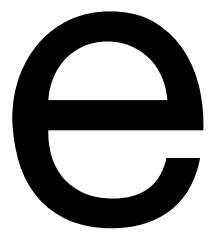


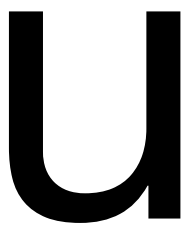


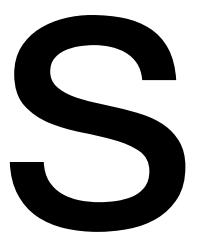


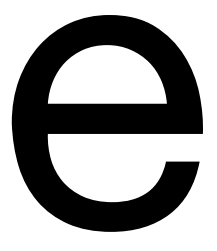


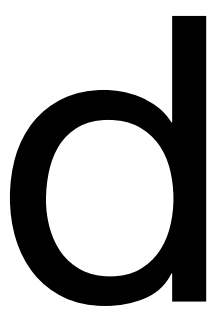


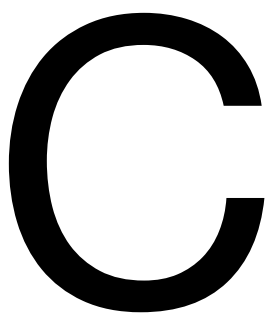


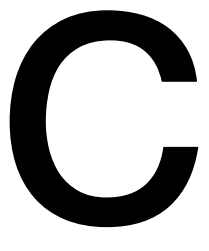


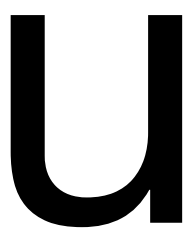


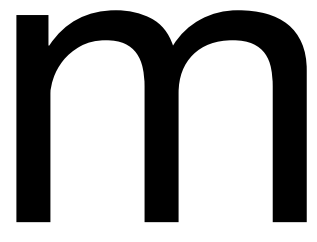


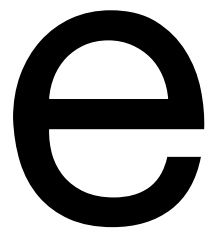


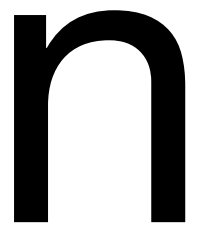


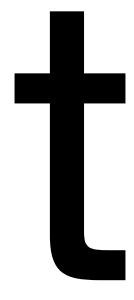

C

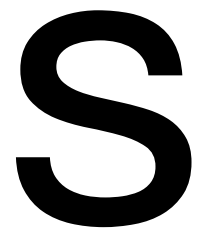


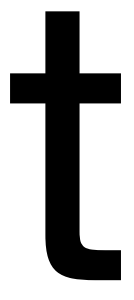


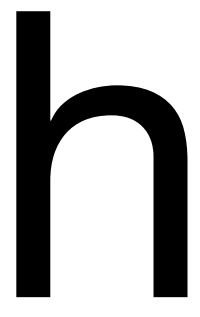


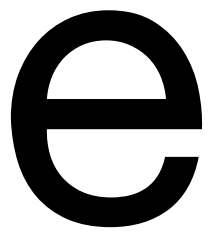


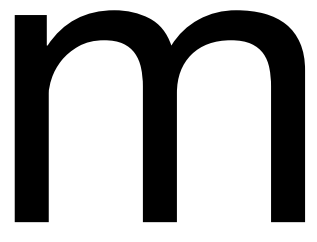


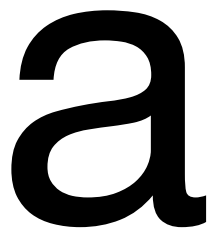


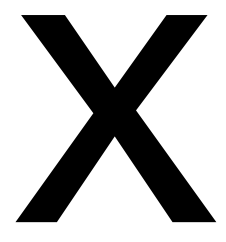


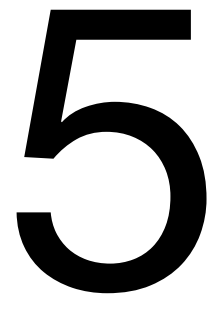


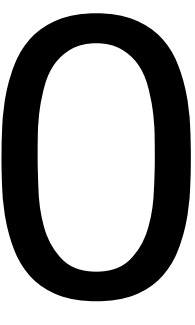


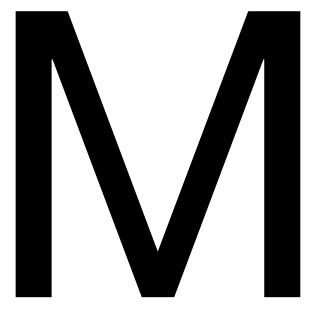


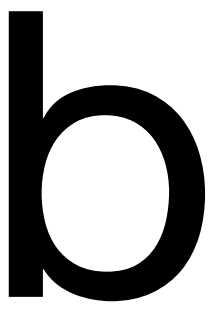


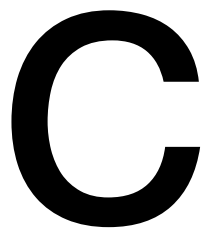


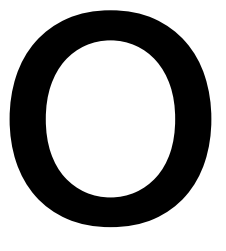


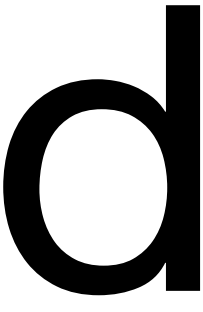


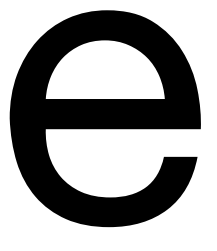


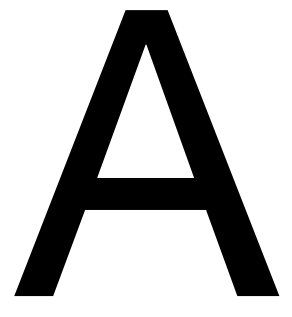


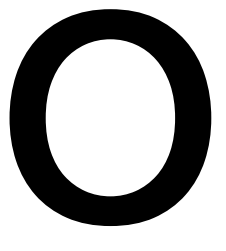

C

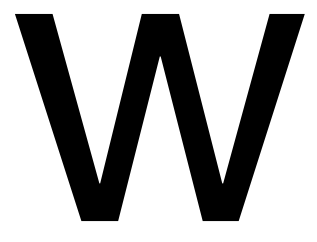


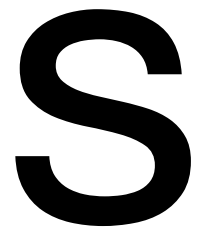

d

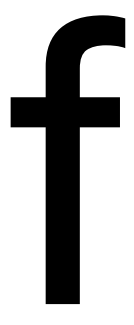


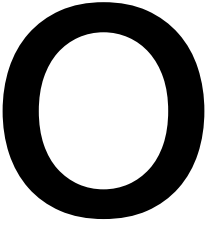


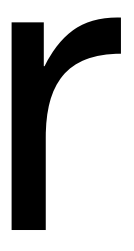


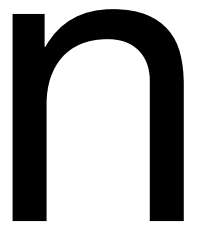


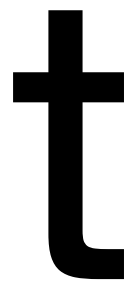


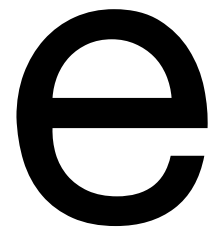


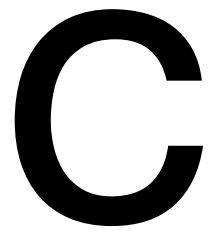


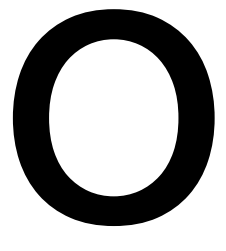


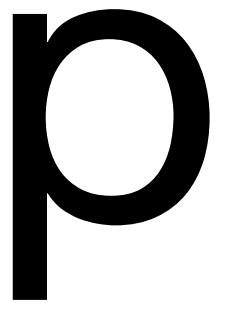


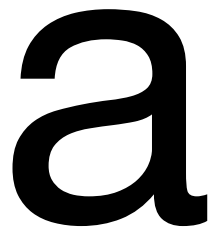


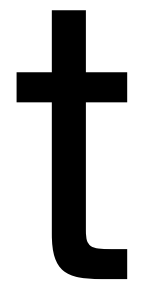


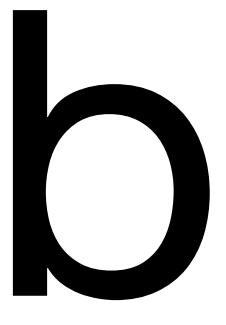

C

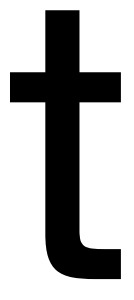


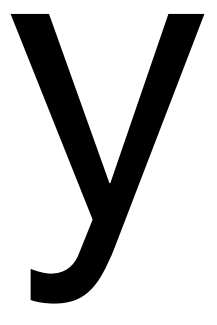


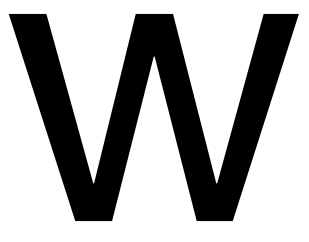


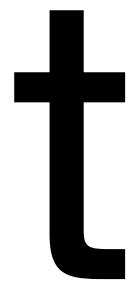


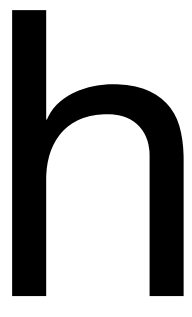


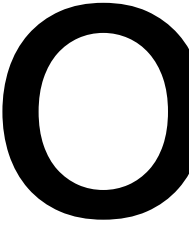


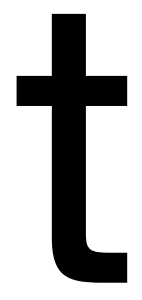


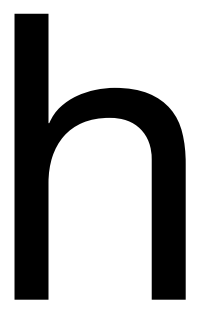


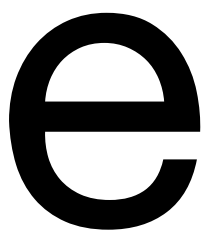


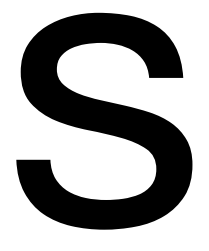


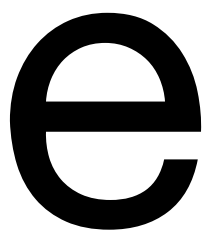


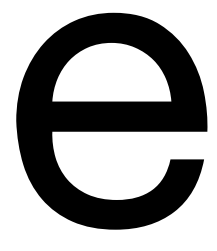


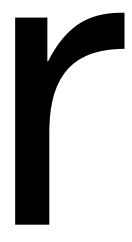


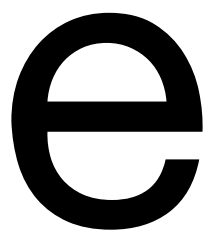


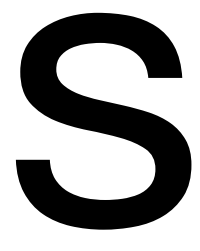


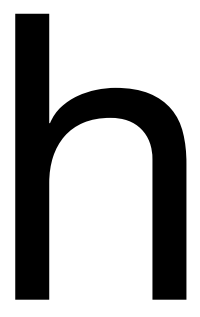


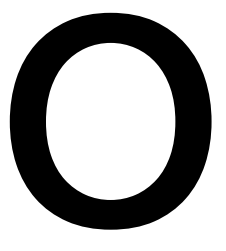


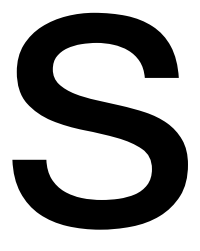


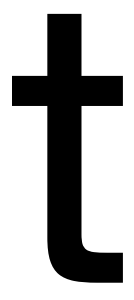


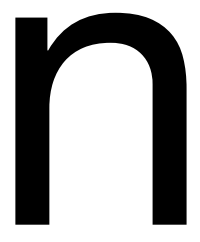


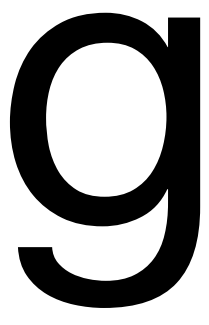


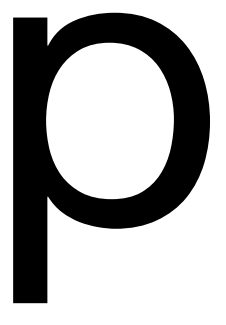


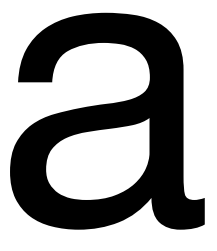


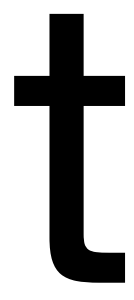


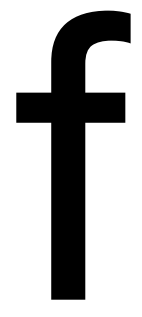


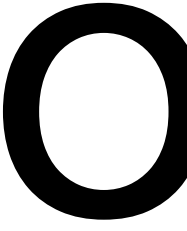


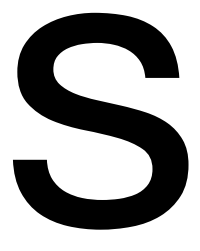


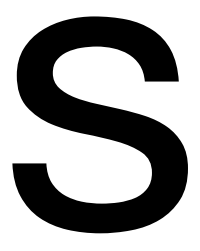


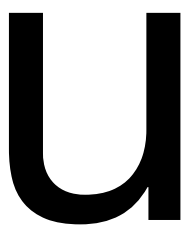


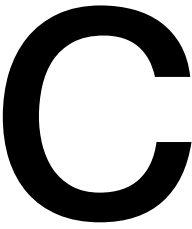


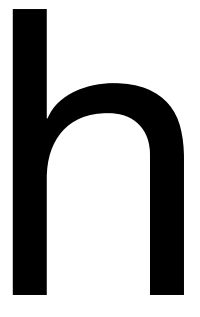


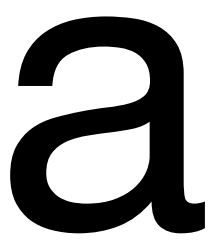


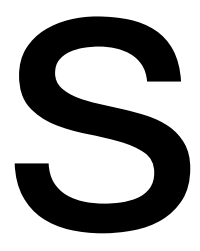


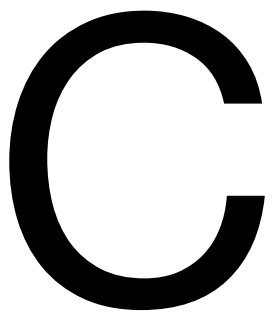


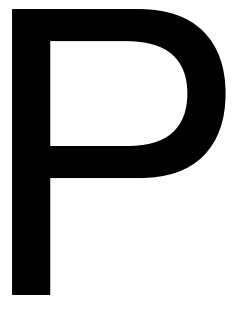

C

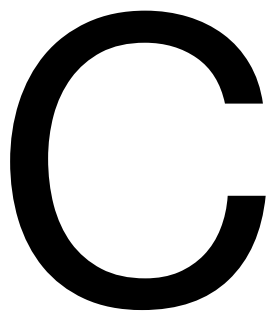


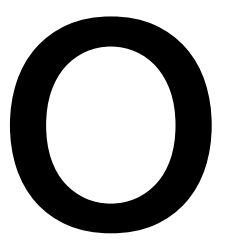


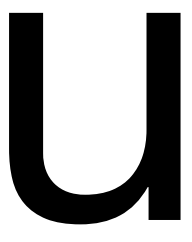


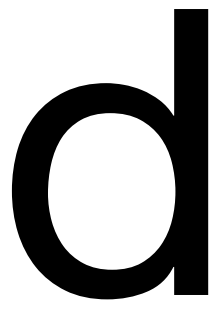


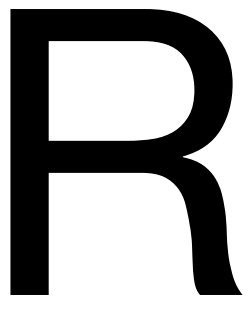


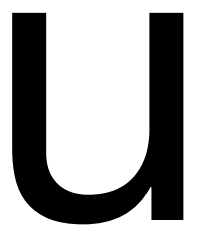


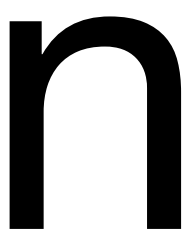


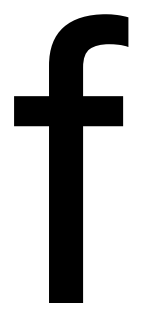


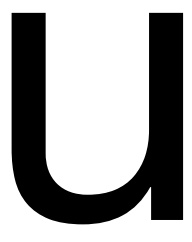


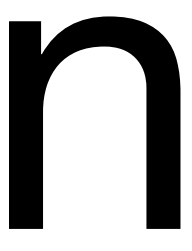


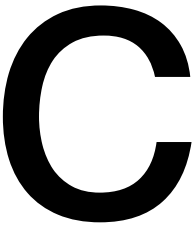


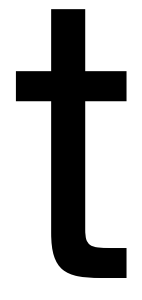


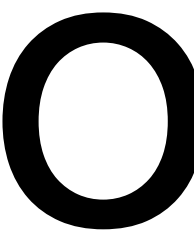


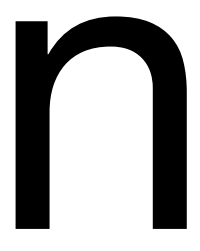


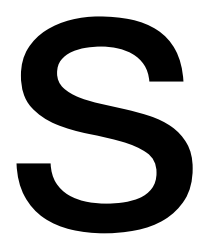


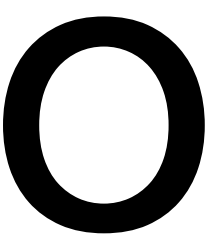

C

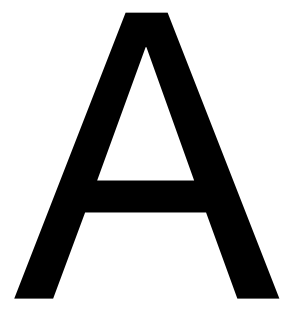


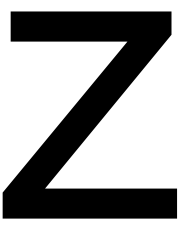


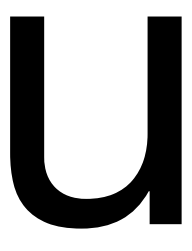


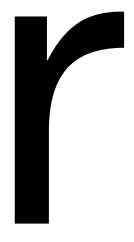


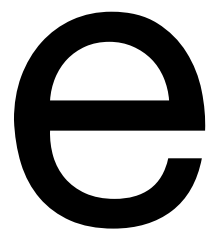


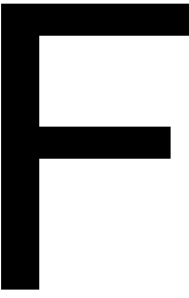


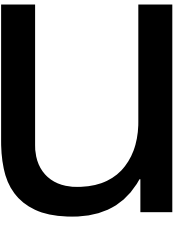


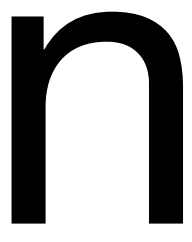


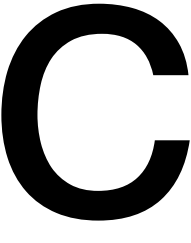


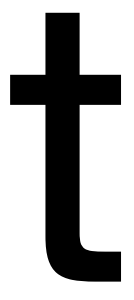


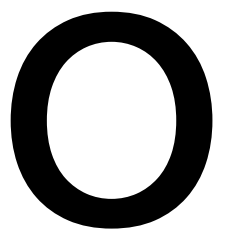


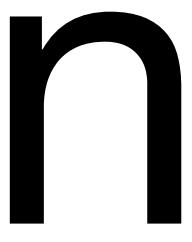


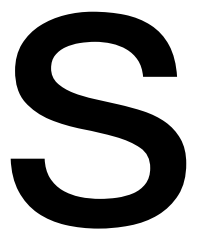



C



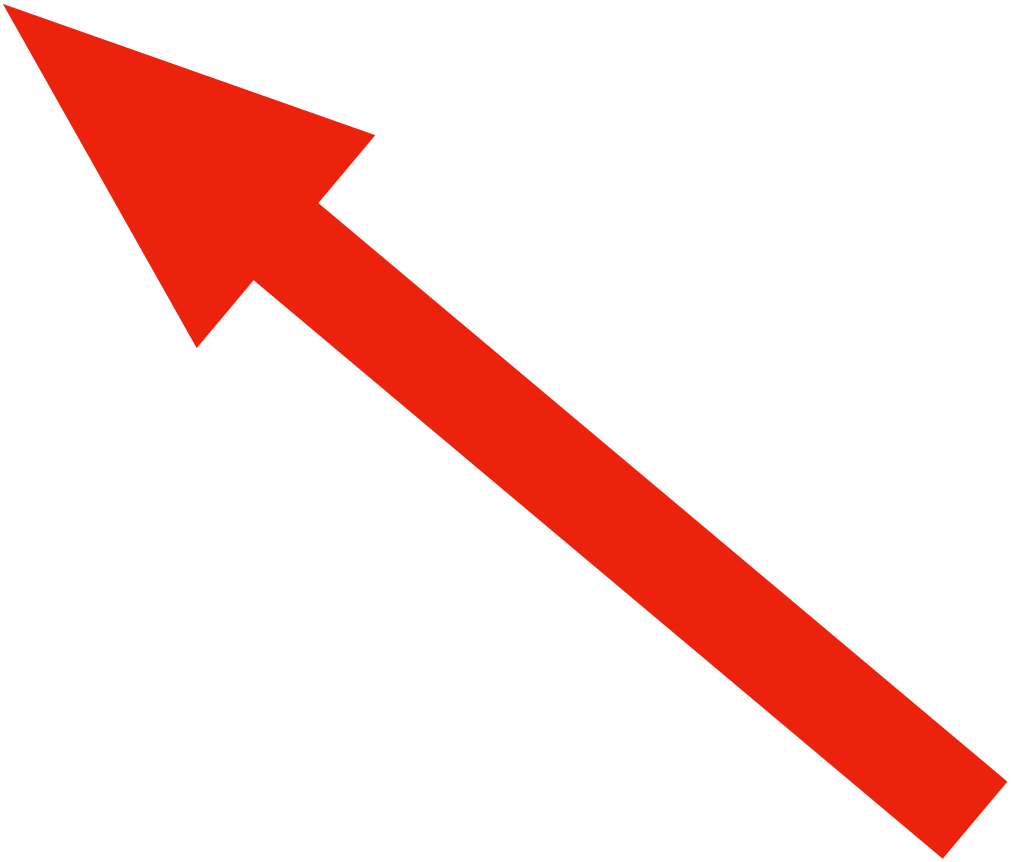






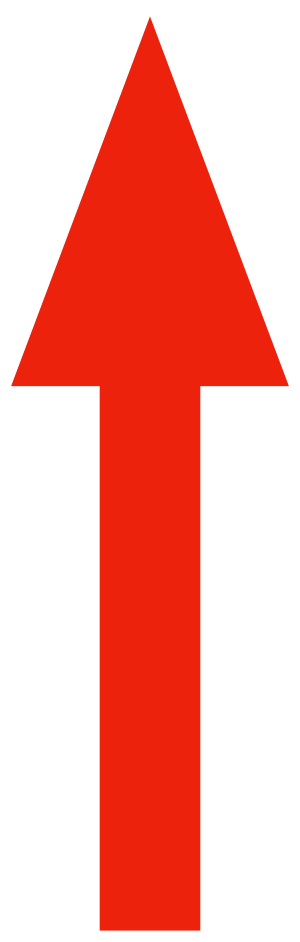
Dockerfile

```dockerfile
FROM public.ecr.aws/lambda/python:3.10-x86_64

# Copy function code

COPY * ${LAMBDA_TASK_ROOT}

# Copy requirements.txt
COPY requirements.txt  .

# Install the specified packages
RUN pip install --upgrade pip
RUN pip3 install --default-timeout=10000  -r requirements.txt  --target "${LAMBDA_TASK_ROOT}"

# Set the CMD to your handler (could also be done as a parameter override outside of the Dockerfile)
CMD [ "app.lambda_handler" ]
```



Example using Flask and PyTorch

Invokes the handler



Alternatively Using Docker

- You can also deploy your Lambda function as a Docker image.
- Allows for non-supported packages or programming languages to be used.
- Circumvents the max 250 Mb code limit.
- Allows for inter-compatibility with other serverless hosting platforms such as GCP Cloud Run functions or Azure Functions.

Dockerfile

```
Dockerfile > ...
1   FROM public.ecr.aws/lambda/python:3.10-x86_64
2
3   # Copy function code
4
5   COPY * ${LAMBDA_TASK_ROOT}
6
7   # Copy requirements.txt
8   COPY requirements.txt  .
9
10  # Install the specified packages
11  RUN pip install --upgrade pip
12  RUN pip3 install --default-timeout=10000  -r requirements.txt  --target "${LAMBDA_TASK_ROOT}"
13
14  # Set the CMD to your handler (could also be done as a parameter override outside of the Dockerfile)
15  CMD [ "app.lambda_handler" ]
```

Pulls the runtime from AWS

Invokes the handler

Example using Flask and PyTorch

Setting a Trigger for the Lambda