coqui TTS

The coquí, pronounced "ko-kee", is a species of tree frog native to Puerto Rico.

TTS is Text-To-Speech. Also known as Speech Synthesis.

Hamilton Python Users Group
11 July 2022
Ian Stewart

Machines that speak.

- Alarm clock, Answer phone. Playing local audio files.
- Supermarket self service checkout. Stock sentences, no logic.
- Espeak TTS (off-line)
- Google TTS (online)

Espeak NG TTS Python Example

```python
# copy from: https://pypi.org/project/espeakng/
import espeaking
import time

esp = espeaking.Speaker()

esp.say("""This is Espeak next generation accessed by the python module espeaking.
This is the default English voice.""", wait4prev=True)

while esp.is_talking():
    "The wait4prev doesn't seem to work so add delay using is_talking probe."
    time.sleep(1)

esp.pitch = 120 # default is 50
esp.wpm = 225
esp.say("""This is Espeak next generation accessed by the python module
espeaking. This is the default English voice with a pitch of 120 and words per
minute setting of 225.""")

while esp.is_talking():
    time.sleep(1)

# set back to default valules...
esp.pitch = 50
esp.wpm = 175
esp.say("""My pitch is back to 50 and my words per minute is back to 175.""")
```

```
$ sudo apt install espeak-ng
From: espeakng-1.0.2.tar.gz
Extract: __init__.py from
Rename: as espeaking.py
```

Google TTS Python Example

```python
# Import the required module for text to speech conversion
from gtts import gTTS

# This module is imported so that we can play the audio
import os

# The text that you want to convert to audio
mytext = ("""Welcome to Google text to speech. I know you are in New Zealand
so I am the one that talks to you in English. If you were in the UK a British
gentleman would be talking to you.""")

# Language in which you want to convert
language = 'en'

# Passing the text and language to the engine, here we have marked slow=False.
# Which tells the module that the converted audio should have a high speed
myobj = gTTS(text=mytext, lang=language, slow=False)

# Saving the converted text in a mp3 file named audio.mp3
myobj.save("audio.mp3")

# Playing the converted file using mpv
os.system("mpv audio.mp3")
```

```
pip3 install gTTS-token --upgrade
pip3 install gTTS --upgrade
Successfully installed gTTS-token-1.1.4
Successfully installed gTTS-2.2.4
```

Google TTS – gspeak.py sample of code...

```
26    url = 'https://translate.google.com/translate_tts'
27    user_agent = 'Mozilla'
28    values = {'tl' : language,
29              'client' : 'tw-ob',
30              'ie' : 'UTF-8',
31              'q' : message }
32
33    data = urllib.parse.urlencode(values)
34    headers = { 'User-Agent' : user_agent }
35
36    req = urllib.request.Request(url + "?" + data, None, headers)
37
38    # mpv seems to work better than mplayer. Doesn't have connect messages
39    player = subprocess.Popen \
40      (
41        args = ("mpv", "-cache", "1024", "-really-quiet", "/dev/stdin"),
42        stdin = subprocess.PIPE
43       )
44    # Send the request to google, and send mp3 data to mp3 player.
45    try:
46        with urllib.request.urlopen(req) as response:
47            mp3_data = response.read()
48            player.stdin.write(mp3_data)
```

```
gspeak("Gspeak is a python3 program...")
gspeak("Bonjour comment-allez vous?", "fr")
gspeak("Guten Morgen.", "de")
```

Google TTS - Python sample from “saytime”

```
33 # Get the local times' hours and minutes
34 hour = time.localtime()[3]
35 minute = time.localtime()[4]
36
37 # Message lists
38 five_minute_list = ["the hour of", "five past", "ten past", "quarter past",
39             "twenty past", "twenty-five past", "half past", "twenty-five to",
40             "twenty to", "quarter to", "ten to", "five to", "the hour of"]
41
42 how_near_list = ["soon to be", "almost", "exactly", "just after",
43             "a little after"]
44
45 hour_list = ["twelve", "one", "two", "three", "four", "five", "six", "seven",
46             "eight", "nine", "ten", "eleven", "twelve", "one", "two", "three",
47             "four", "five", "six", "seven", "eight", "nine", "ten", "eleven",
48             "twelve"]
49
50 time_of_day = ["at night", "in the morning", "in the afternoon",
51             "in the evening"]
52
53 def round_to_5_minute(x, base=5):
54     # Round the minutes to 5 minute intervals. E.g. 3 to 7 will rounded to 5.
55     return int(base * round(float(x)/base))
```

Google TTS – Gstreamer 1/2.

```
15 gi.require_version('Gst', '1.0')
16 gi.require_version('GLib', '2.0')
17 from gi.repository import Gst, GLib
```

```
22 URI =  'https://translate.google.com/translate_tts?'
23 URI += 'ie=UTF-8&client=tw-ob&tl={}&q={}'
```

```
26 URI_COMPOSED = URI.format("en-US", "This should be with an American accent.")
27
28 def main(uri=URI_COMPOSED):
37     # Init  - call initialize function.
38     player, loop = initialize()
39     # Set the uri to be sent to google
40     player.set_property('uri', uri)
41     # Send text to google, and start streaming the mp3 audio with playbin
42     player.set_state(Gst.State.PLAYING)
43     # Loop while waiting for audio to finish.
44     loop.run()
45     # On exiting loop() set playbin state to Null.
46     player.set_state(Gst.State.NULL)
```

```
75     loop = GLib.MainLoop()
```

Google TTS – Gstreamer 2/2.

```
49 def initialize():
58     Gst.init(None)
63     player = Gst.ElementFactory.make("playbin") #, 'player')
69     # Stop video. Only sending audio to playbin
70     fakesink = Gst.ElementFactory.make("fakesink", "fakesink")
71     player.set_property("video-sink", fakesink)
75     loop = GLib.MainLoop()
77     # Instantiate and initialize the bus call-back
78     bus = player.get_bus()
79     bus.add_signal_watch()
80     bus.connect ("message", bus_call, loop)
82     return player, loop
```

Do a demo of TTS:

- Espeak (off-line) – Using python espeaking module
  - `python dev/espeak-tts/py-espeak.py`
- Google TTS (online) – Using python gTTS module
  - `python dev/google-tts/google-tts.py`
- Google TTS (online) – Lawrence and Ian connection to Google
  - `python dev/pylib/gspeak.py`
- Google TTS (online) – Apply logic
  - `python dev/pylib/saytime.py`
  - `python dev/pylib/saytime.py full`
- Google TTS (online) – Gstreamer – playbin
  - `python dev/gstreamer/google_tts_gstreamer.py`

What’s happening with Speech Synthesis:

- Terminology:
  - Acoustic Model
  - Prosody
  - Vocoder
  - Deep Neuronal Networks (DNN)
  - Mel Spectrogram - https://en.wikipedia.org/wiki/Mel_scale
  - Speech Dataset

- TTS Steps:
  - Generating a frequency representation of the sentence (the mel spectrogram)
  - Generating the waveform from this representation.

Steps:

Sentence

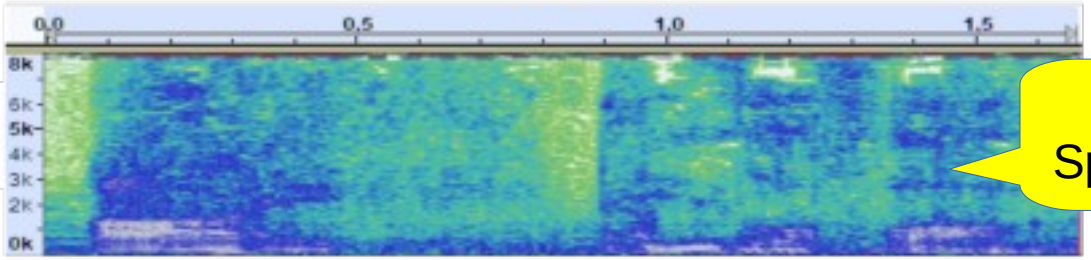
I am your father

Phonemes

aɪ æm jʊər ˈfɑðər

Text Analysis / Acoustic Analysis Model

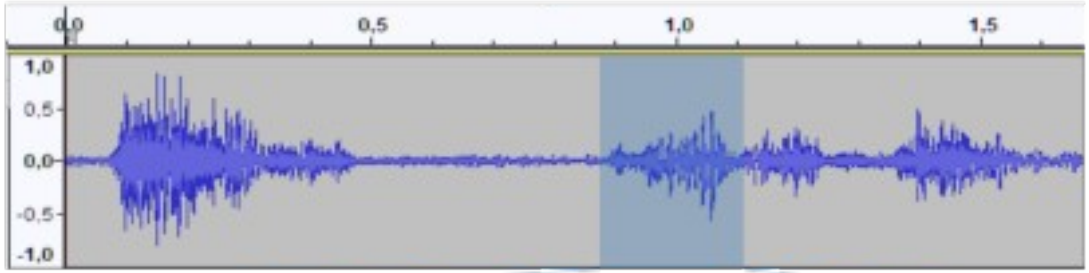
Spectrogram



Mel Spectrogram

Vocoder

Waveform



Audio Signal

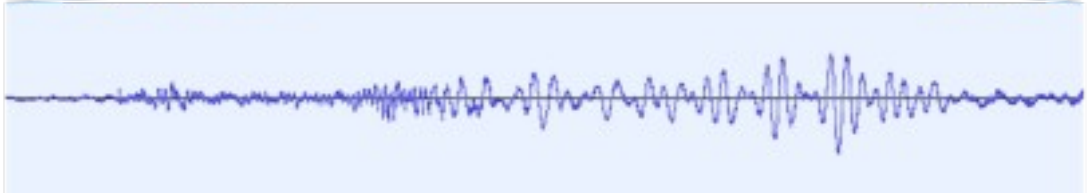


Diagram by: Patrick Meyer Jun 18, 2021

Speech Synthesis Architectures 2016-2021

Colour Code:

Yellow: Acoustic Model

Brown: Vocoder

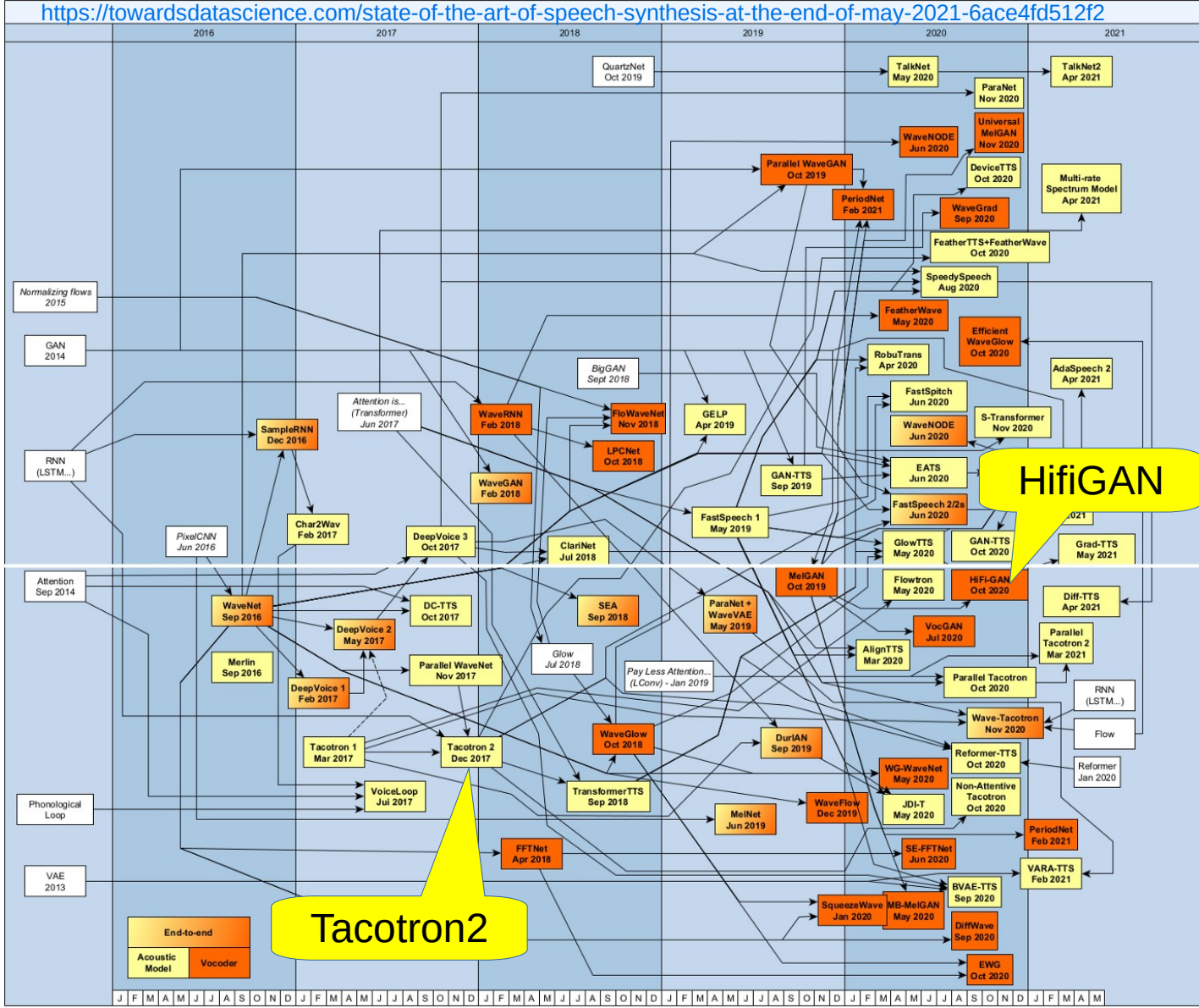
Mix: End-to-end

Coqui TTS Defaults to:

Model: Tacotron 2 Model

Vocoder: Hifi Generative Adversarial Network (GAN)

Diagram by: Patrick Meyer Jun 18, 2021



Spectrogram

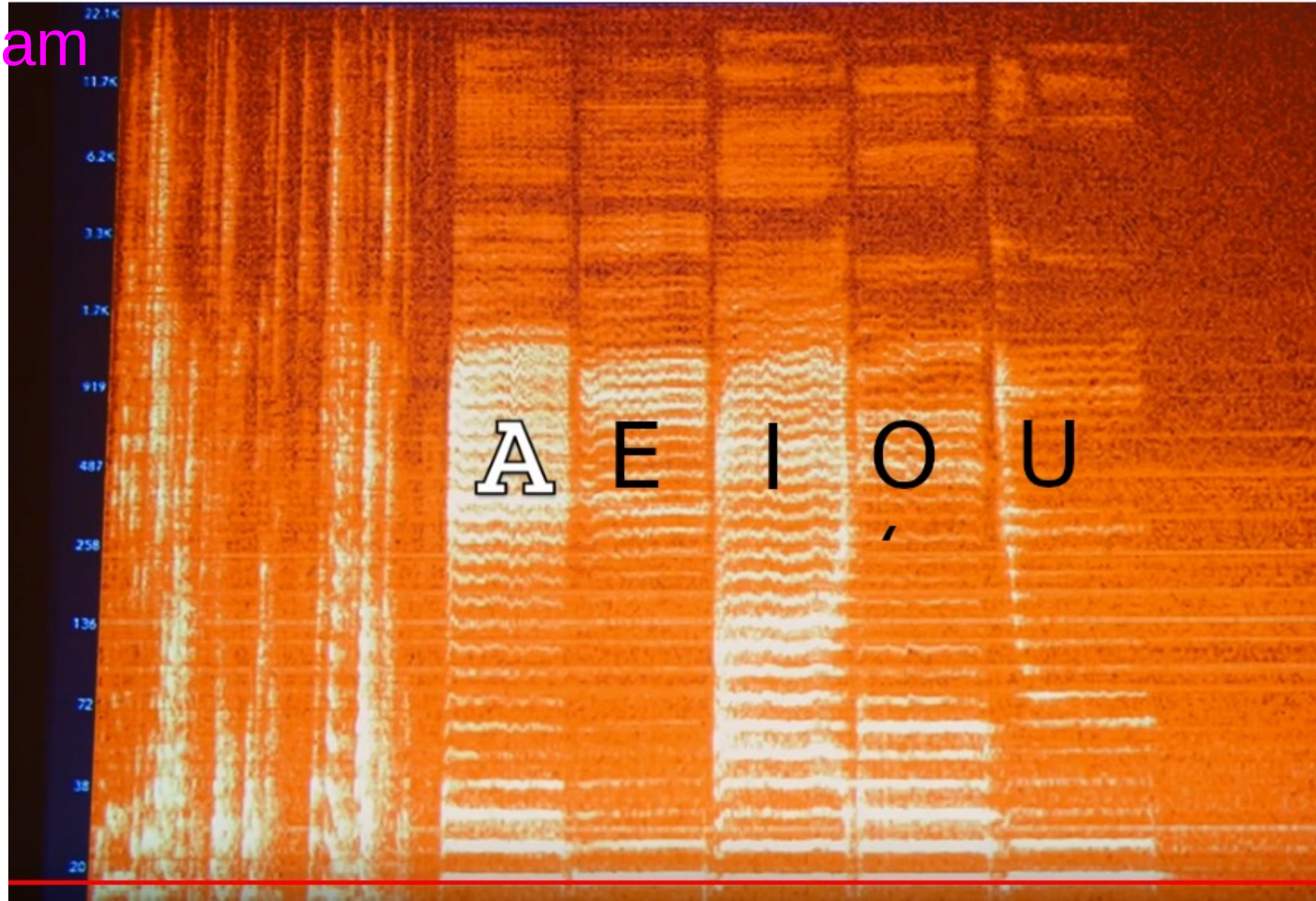
Axis:

X: Time

Y: Frequency

Z: Power/Volume

Mel: Melody Scale



Coqui History

- Called Coquí because the frog is well-known for being small but having a loud, clear voice.
- In 2016 developers at Mozilla commenced open-source projects:
- Developers:
  - DeepSpeech engine for STT
    `https://github.com/mozilla/DeepSpeech/`
  - TTS engines.
  - Capture thousands of hours of speech training data
- Start the Coqui organization. Fork DeepSpeech
- Main website: https://coqui.ai
- Github repository: https://github.com/coqui-ai
- PyPI: https://pypi.org/project/TTS/

Coqui TTS Installation on Ubuntu 22.04 Mate

- `$ sudo apt install python3.10-venv`
  To avoid the following error:
  TTS/tts/utils/monotonic_align/core.c:19:10: fatal error: Python.h: No such file or directory
- `$ sudo apt install libpython3.10-dev`
- `$ python -m venv tts-venv`
- `$ source tts-venv/bin/activate`
- `(tts-venv):~/tts-venv$ pip install TTS`
- OR
- `(tts-venv):~/tts-venv$ pip install pip setuptools wheel tts --upgrade`
- `(tts-venv):~/tts-venv$ pip install python-mpv`
- Installs the following 83 python libraries (3.3GB+)...

Coqui TTS – Python libraries installed 1/2

anyascii-0.3.1	Flask-2.1.2	gruut-ipa-0.13.0
appdirs-1.4.4	fonttools-4.33.3	idna-3.3
audioread-2.1.9	fsspec-2022.5.0	inflect-5.6.0
Babel-2.10.3	gruut_lang_cs-2.0.0	itsdangerous-2.1.2
certifi-2022.6.15	gruut_lang_de-2.0.0	jieba-0.42.1
cffi-1.15.0	gruut_lang_en-2.0.0	Jinja2-3.1.2
charset-normalizer-2.0.12	gruut_lang_es-2.0.0	joblib-1.1.0
click-8.1.3	gruut_lang_fr-2.0.2	jsonlines-1.2.0
coqpit-0.0.16	gruut_lang_it-2.0.0	kiwisolver-1.4.3
cycler-0.11.0	gruut_lang_nl-2.0.2	librosa-0.8.0
cython-0.29.28	gruut_lang_pt-2.0.0	llvmlite-0.38.1
dateparser-1.1.1	gruut_lang_ru-2.0.0	MarkupSafe-2.1.1
decorator-5.1.1	gruut_lang_sv-2.0.0	matplotlib-3.5.2
docopt-0.6.2	gruut-2.2.3	mecab-python3-1.0.5

Coqui TTS – Python libraries installed 2/2

networkx-2.8.4	Python-crfsuite-0.9.8	threadpoolctl-3.1.0
num2words-0.5.10	python-dateutil-2.8.2	torch-1.12.0
numba-0.55.1	pytz-2022.1	torchaudio-0.12.0
numpy-1.21.6	pytz-deprecation-shim-0.1.0.post0	tqdm-4.64.0
packaging-21.3	pyworld-0.2.10	trainer-0.0.12
pandas-1.4.3	pyyaml-6.0	TTS-0.7.1
pillow-9.1.1	regex-2022.3.2	typing-extensions-4.2.0
pooch-1.6.0	requests-2.28.0	tzdata-2022.1
protobuf-3.19.4	resampy-0.2.2	tzlocal-4.2
pycparser-2.21	scikit-learn-1.1.1	umap-learn-0.5.1
pynndescent-0.5.7	scipy-1.8.1	unidic-lite-1.0.8
pyparsing-3.0.9	six-1.16.0	urllib3-1.26.9
pypinyin-0.46.0	soundfile-0.10.3.post1	Werkzeug-2.1.2
pysbd-0.3.4	tensorboardX-2.5.1	

776MB

The LJ Speech Dataset – LJ = Linda Johnson

- Available from: https://keithito.com/LJ-Speech-Dataset/ (2.6GB)
- Total Clips: 13,100
- Total Words: 225,715
- Total Characters: 1,308,678
- Total Duration: 23:55:17
- Mean Clip Duration: 6.57 sec
- Min Clip Duration: 1.11 sec
- Max Clip Duration: 10.10 sec
- Mean Words per Clip: 17.23
- Distinct Words: 13,821

- Linda Johnson from LibriVox, 13100 passages from 7 non-fiction books (2017). Published between 1884 and 1964.
- https://librivox.org/ Free public domain audiobooks

The LJ Speech Dataset – LJ = Linda Johnson

- Keith Ito provided the metadata file associated with the .wav files.

- Metadata is provided in transcripts.csv.
  - One record per line, delimited by the pipe character (0x7c).
- File Format. Fields are:
  - **ID**: this is the name of the corresponding .wav file
  - **Transcription**: words spoken by the reader (UTF-8)
  - **Normalized Transcription**: transcription with numbers, ordinals, and monetary units expanded into full words (UTF-8).

- Each audio file is a single-channel 16-bit PCM WAV with a sample rate of 22050 Hz.
- Available from: https://keithito.com/LJ-Speech-Dataset/ (2.6GB)

The LJ Speech Dataset – Notes.

- Dataset is Public Domain.
- Version 1.1 (fixes 30 mistakes in V1.0)
- Clip boundaries generally align with sentence or clause boundaries, but not always.
- The text was matched to the audio manually.
- High quality. E.g. Don’t get hum in the recording.
- 19 of the transcriptions contain non-ASCII characters. E.g. LJ016-0257 contains "raison d'être".
- Original LibriVox recordings were 128 kbps MP3 files.

- Nancy corpus dataset. Nancy Krebs speaking. 16K wav files.
- Blizzard 2012 dataset

The LJ Speech Dataset – Notes.

- The first set was trained for 441K steps on the LJ Speech Dataset
- Speech started to become intelligible around 20K steps.

After installation – in tts virtual environment

- Bash command: tts
- Bash command: tts-server
- Python: TTS library

```
>>> from TTS import *
>>>
>>> dir()
['__annotations__', '__builtins__', '__doc__', '__loader__', '__name__', '__package__',
 '__spec__', 'bin', 'config', 'encoder', 'f', 'model', 'os', 'server', 'tts', 'utils',
'version', 'vocoder']
>>>
```

Bash tts

$ tts –text “Some text.” will use the default model and vocoder and output a wav file to the current directory.
&& mpv tts_output.wav will play the wav file.

```
(tts-venv) ian@dell:~$ tts --text "This is a test of the text to speech bash com
mand. $(date +'%A %H:%m %d %B %Y')" && mpv tts_output.wav
```

Defaults. Based on ljspeech dataset...
Model: tacotron2
Vocoder: hifigan_v2

```
> tts_models/en/ljspeech/tacotron2-DDC is already downloaded.
> vocoder_models/en/ljspeech/hifigan_v2 is already downloaded.
> Using model: Tacotron2
```

First time, need to wait a few minutes for the model and vocoder to download.

Bash tts – Default settings. 1/3

```
> Setting up Audio Processor...
 | > sample_rate:22050
 | > resample:False
 | > num_mels:80
 | > log_func:np.log
 | > min_level_db:-100
 | > frame_shift_ms:None
 | > frame_length_ms:None
 | > ref_level_db:20
 | > fft_size:1024
 | > power:1.5
 | > preemphasis:0.0
 | > griffin_lim_iters:60
 | > signal_norm:False
 | > symmetric_norm:True
 | > mel_fmin:0
 | > mel_fmax:8000.0
 | > pitch_fmin:0.0
 | > pitch_fmax:640.0
 | > spec_gain:1.0
 | > stft_pad_mode:reflect
 | > max_norm:4.0
 | > clip_norm:True
 | > do_trim_silence:True
 | > trim_db:60
 | > do_sound_norm:False
 | > do_amp_to_db_linear:True
 | > do_amp_to_db_mel:True
 | > do_rms_norm:False
 | > db_level:None
 | > stats_path:None
 | > base:2.718281828459045
 | > hop_length:256
 | > win_length:1024
```

Bash tts – Default settings. 2/3

```
> Model's reduction rate `r` is set to: 1
> Vocoder Model: hifigan
> Setting up Audio Processor...
| > sample_rate:22050
| > resample:False
| > num_mels:80
| > log_func:np.log
| > min_level_db:-100
| > frame_shift_ms:None
| > frame_length_ms:None
| > ref_level_db:20
| > fft_size:1024
| > power:1.5
| > preemphasis:0.0
| > griffin_lim_iters:60
| > signal_norm:False
| > symmetric_norm:True
| > mel_fmin:0
| > mel_fmax:8000.0
| > pitch_fmin:0.0
| > pitch_fmax:640.0
| > spec_gain:1.0
| > stft_pad_mode:reflect
| > max_norm:4.0
| > clip_norm:True
| > do_trim_silence:False
| > trim_db:60
| > do_sound_norm:False
| > do_amp_to_db_linear:True
| > do_amp_to_db_mel:True
| > do_rms_norm:False
| > db_level:None
| > stats_path:None
| > base:2.718281828459045
| > hop_length:256
| > win_length:1024
```

Bash tts – Default settings. 3/3

```
 > Generator Model: hifigan_generator
 > Discriminator Model: hifigan_discriminator
Removing weight norm...
 > Text: This is a test of the text to speech bash command. Sunday 20:07 10 July
 2022
 > Text splitted to sentences.
['This is a test of the text to speech bash command.', 'Sunday 20:07 10 July 202
2']
[W NNPACK.cpp:51] Could not initialize NNPACK! Reason: Unsupported hardware.
 > Processing time: 3.792480230331421
 > Real-time factor: 0.4733301772709192
 > Saving output to tts_output.wav
 (+) Audio --aid=1 (pcm_s16le 1ch 22050Hz)
AO: [pulse] 22050Hz mono 1ch s16
A: 00:00:07 / 00:00:08 (96%)

Exiting... (End of file)
```

Old PC. Doesn't have AVX2 instructions. Need Haswell microarchitecture (4th generation) Jun 2013 onwards.

Bash tts-server 1/4

```
(tts-venv) ian@dell:~$ tts-server
 > tts_models/en/ljspeech/tacotron2-DDC is already downloaded.
 > vocoder_models/en/ljspeech/hifigan_v2 is already downloaded.
```

```
 > Using model: Tacotron2
 > Setting up Audio Processor...
 | > sample_rate:22050
 | > resample:False
 | > num_mels:80
 | > log_func:np.log
 | > min_level_db:-100
```

... snip ...

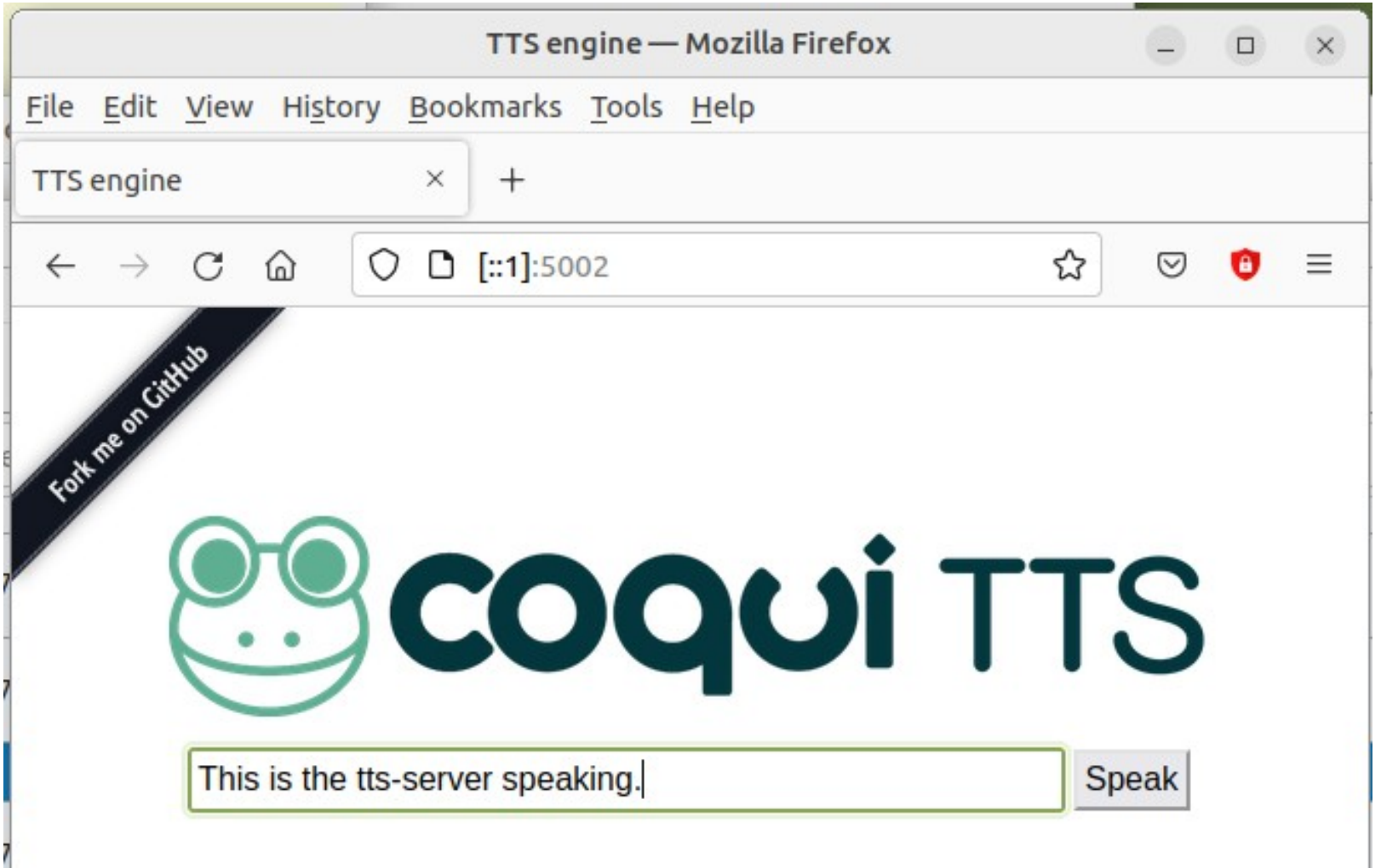
```
 | > base:2.718281828459045
 | > hop_length:256
 | > win_length:1024
 > Generator Model: hifigan_generator
 > Discriminator Model: hifigan_discriminator
```

Bash tts-server 2/4

```
Removing weight norm...
 * Serving Flask app 'TTS.server.server' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: off
INFO:werkzeug: * Running on all addresses (::)
   WARNING: This is a development server. Do not use it in a production deployme
nt.
 * Running on http://[::1]:5002
 * Running on http://[::1]:5002 (Press CTRL+C to quit)
```

Go to blank tab on Browser and paste http://[::1]:5002

Bash tts-server 3/4

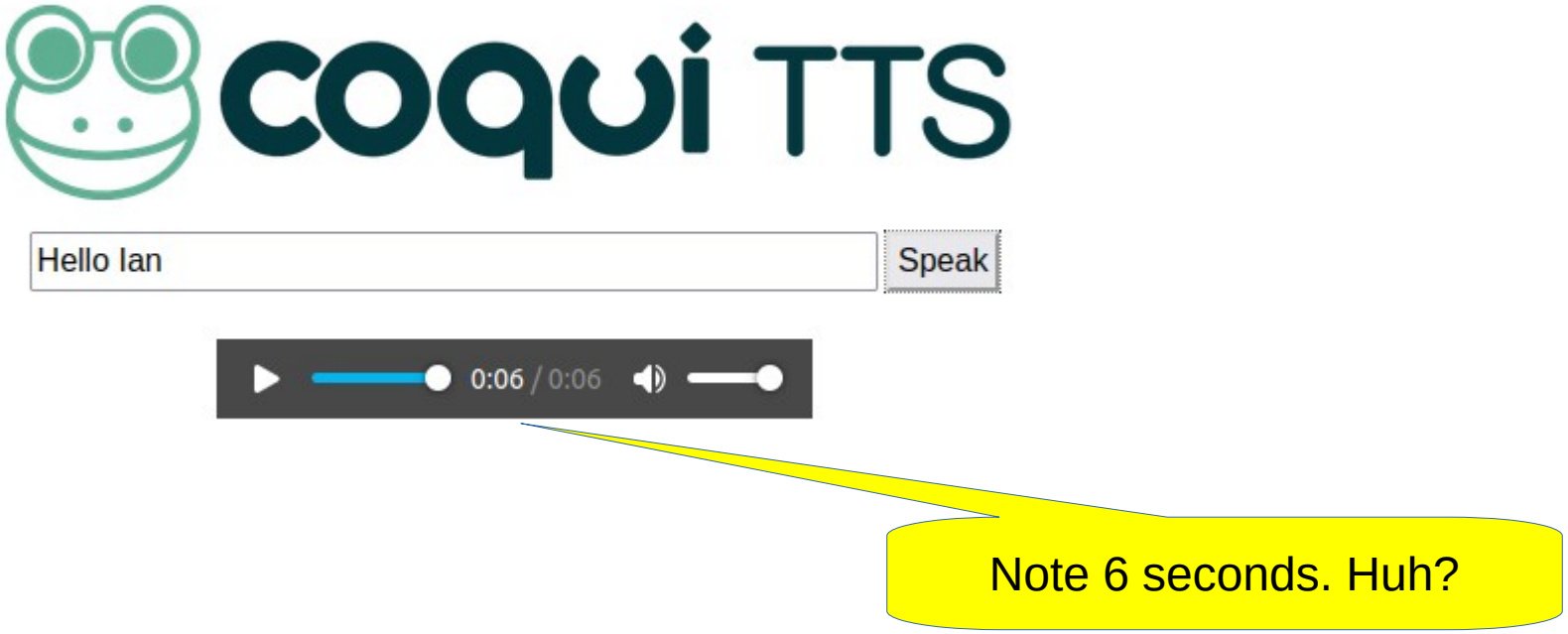


Bash tts-server 4/4. On clicking “Speak”

```
 > Model input: This is the server speaking.
 > Speaker Idx:
 > Text splitted to sentences.
['This is the server speaking.']
 > Processing time: 0.9368383884429932
 > Real-time factor: 0.42233575533954854
INFO:werkzeug:::1 - - [10/Jul/2022 21:31:09] "GET /api/tts?text=This%20is%20the%
20server%20speaking.&speaker_id=&style_wav= HTTP/1.1" 200 -
```

Changed “the tts-server speaking” to the server speaking”

Demo: Bash tts-server.



Python?

```
>>> from TTS import *
>>>
>>> dir()
['__annotations__', '__builtins__', '__doc__', '__loader__', '__name__', '__package__',
 '__spec__', 'bin', 'config', 'encoder', 'f', 'model', 'os', 'server', 'tts', 'utils',
'version', 'vocoder']
>>>
```

```
>>> from TTS.bin import synthesize
>>> synthesize.Synthesizer.tts("Hello")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/home/ian/tts-venv/lib/python3.10/site-packages/TTS/utils/synthesizer.py", line
203, in tts
    raise ValueError(
ValueError: You need to define either `text` (for sythesis) or a `reference_wav` (for
voice conversion) to use the Coqui TTS API.

>>> from TTS.tts.configs.tacotron_config import TacotronConfig
>>> config = TacotronConfig()
```

Models:

```
(tts-venv) ian@dell:~$ tts --list_models
 Name format: type/language/dataset/model
 1: tts_models/multilingual/multi-dataset/your_tts
 2: tts_models/en/ek1/tacotron2
 3: tts_models/en/ljspeech/tacotron2-DDC [already downloaded]
 4: tts_models/en/ljspeech/tacotron2-DDC_ph
 5: tts_models/en/ljspeech/glow-tts [already downloaded]
 6: tts_models/en/ljspeech/speedy-speech
 7: tts_models/en/ljspeech/tacotron2-DCA
 8: tts_models/en/ljspeech/vits
 9: tts_models/en/ljspeech/fast_pitch
 10: tts_models/en/vctk/vits
 11: tts_models/en/vctk/fast_pitch
 12: tts_models/en/sam/tacotron-DDC
 13: tts_models/en/blizzard2013/capacitron-t2-c50
 14: tts_models/en/blizzard2013/capacitron-t2-c150

 15: tts_models/es/mai/tacotron2-DDC
 16: tts_models/fr/mai/tacotron2-DDC
 17: tts_models/uk/mai/glow-tts
```

18 to 33 Other languages: zh, nl, de, ja, tw, ...

Models (continued):

```
(tts-venv) ian@dell:~$ tts –list_models
 Name format: type/language/dataset/model

 1: vocoder_models/universal/libri-tts/wavegrad
 2: vocoder_models/universal/libri-tts/fullband-melgan
 3: vocoder_models/en/ek1/wavegrad
 4: vocoder_models/en/ljspeech/multiband-melgan [already downloaded]
 5: vocoder_models/en/ljspeech/hifigan_v2 [already downloaded]
 6: vocoder_models/en/ljspeech/univnet
 7: vocoder_models/en/blizzard2013/hifigan_v2
 8: vocoder_models/en/vctk/hifigan_v2
 9: vocoder_models/en/sam/hifigan_v2

 10: vocoder_models/nl/mai/parallel-wavegan
 11: vocoder_models/de/thorsten/wavegrad
 12: vocoder_models/de/thorsten/fullband-melgan
 13: vocoder_models/ja/kokoro/hifigan_v1
 14: vocoder_models/uk/mai/multiband-melgan
 15: vocoder_models/tr/common-voice/hifigan

 ...
```

Bash tts:

```
(tts-venv) ian@dell:~$ tts \
--text "This is the default model and vocoder." \
--model_name "tts_models/en/ljspeech/tacotron2-DDC" \
--vocoder_name "vocoder_models/en/ljspeech/hifigan_v2" \
--out_path default.wav \
&& mpv default.wav
 > tts_models/en/ljspeech/tacotron2-DDC is already downloaded.
 > vocoder_models/en/ljspeech/hifigan_v2 is already downloaded.
 > Using model: Tacotron2
 > Setting up Audio Processor...
```


```
(tts-venv) ian@dell:~$ tts \
--text "This is the default model and vocoder." \
--model_name "tts_models/en/ljspeech/tacotron2-DDC" \
--vocoder_name "vocoder_models/en/ljspeech/hifigan_v2" \
--out_path default.wav \
&& mpv default.wav
```


...

Bash tts scripts. Compare the audio results:

```
tts \
--text "This is the default model and vocoder." \
--model_name "tts_models/en/ljspeech/tacotron2-DDC" \
--vocoder_name "vocoder_models/en/ljspeech/hifigan_v2" \
--out_path default.wav \
&& mpv default.wav

tts \
--text "This is the glow model and melgan vocoder." \
--model_name "tts_models/en/ljspeech/glow-tts" \
--vocoder_name "vocoder_models/en/ljspeech/multiband-melgan" \
--out_path alternative.wav \
&& mpv alternative.wav

mpv default.wav && mpv alternative.wav && mpv default.wav &&
alternative.wav
```

Model: Tacotron2 vs. Glow-tts and Vocoder: HifiGAN vs. MelGAN

Bash tts scripts. Compare the “Hello” results:

```
tts \
--text "Hello" \
--model_name "tts_models/en/ljspeech/tacotron2-DDC" \
--vocoder_name "vocoder_models/en/ljspeech/hifigan_v2" \
--out_path default.wav \
&& mpv default.wav \
&& tts \
--text "Hello" \
--model_name "tts_models/en/ljspeech/glow-tts" \
--vocoder_name "vocoder_models/en/ljspeech/multiband-melgan" \
--out_path alternative.wav \
&& mpv alternative.wav

mpv default.wav && mpv alternative.wav && mpv default.wav &&
alternative.wav
```

Model: Tacotron2 vs. Glow-tts and Vocoder: HifiGAN vs. MelGAN

Bash tts scripts. Try the uk model and vocoder:

```
tts \
--text "This is the glow model and melgan vocoder." \
--model_name "tts_models/uk/mai/glow-tts" \
--vocoder_name "vocoder_models/uk/mai/multiband-melgan" \
--out_path glow_melgan.wav \
&& mpv glow_melgan.wav


 > Downloading model to /home/ian/.local/share/tts/tts_models--uk--
mai--glow-tts
 > Model's license - MIT
 > Check https://choosealicense.com/licenses/mit/ for more info.
 > Downloading model to /home/ian/.local/share/tts/vocoder_models--
uk--mai--multiband-melgan
```

2 x Downloads took 5 mins

Model: uk/mai/glow-tts and Vocoder: uk/mai/multiband-melgan

Bash tts scripts. Result uk model and vocoder:

```
 > Text splitted to sentences.
['This is the glow model and melgan vocoder.']
this is the glow model and melgan vocoder.
 [!] Character 't' not found in the vocabulary. Discarding it.
this is the glow model and melgan vocoder.
 [!] Character 'h' not found in the vocabulary. Discarding it.
this is the glow model and melgan vocoder.
 [!] Character 's' not found in the vocabulary. Discarding it.
this is the glow model and melgan vocoder.
 [!] Character 'g' not found in the vocabulary. Discarding it.
this is the glow model and melgan vocoder.
 [!] Character 'l' not found in the vocabulary. Discarding it.
this is the glow model and melgan vocoder.
 [!] Character 'w' not found in the vocabulary. Discarding it.
this is the glow model and melgan vocoder.
 [!] Character 'm' not found in the vocabulary. Discarding it.
this is the glow model and melgan vocoder.
 [!] Character 'd' not found in the vocabulary. Discarding it.
this is the glow model and melgan vocoder.
 [!] Character 'v' not found in the vocabulary. Discarding it.
```

Doesn’t know: t, h, s, g, l, w, m, d, v. OK with: i, e, o, n, c, r

Bash tts scripts. Try ljspeech/glow-tts and sam/hifigan_v2:

```
tts \
--text "This is the glow model and hifilgan vocoder." \
--model_name "tts_models/en/ljspeech/glow-tts" \
--vocoder_name "vocoder_models/en/sam/hifigan_v2" \
--out_path glow_hifigan.wav \
&& mpv glow_hifigan.wav
```

Dalek

Demos & Questions...