Module used:

1.**pyttsx3:-**

* Introduction

It shows how to use Python for text-to-speech using a cross-platform library, **pyttsx3**. This lets you synthesize text in to audio you can hear. This package works in Windows, Mac, and Linux. It uses native speech drivers when available and works completely offline.

There are some other cool features that are not covered here, like the event system. You can hook in to the engine on certain events. You can use this to count how many words are said and cut it off if it has received input that is too long. You can inspect each word and cut it off if there are inappropriate words. The event hooks are not covered here but are worth a mention. Check the [official examples](https://pyttsx3.readthedocs.io/en/latest/engine.html#examples) to see how this is done.

Always refer to the [official documentation](https://pyttsx3.readthedocs.io/en/latest/) for the most accurate, complete, and up-to-date information. This is only meant to serve as a primer.

The **pyttsx3** module supports native Windows and Mac speech APIs but also supports espeak, making it the best available text-to-speech package in my opinion. If you are interested specifically and only in speak, you might be interested in my [Python text-to-speech with espeak tutorial](https://www.devdungeon.com/content/text-speech-python-espeak).

* Install the package

Use **pip** to install the package. If you are in Windows, you will need an additional package, **pypiwin32** which it will need to access the native Windows speech API.

pip install pyttsx3  
pip install pypiwin32  # Windows only

* Convert text to speech

# pip install pyttsx3 pypiwin32  
**import** pyttsx3  
  
# One time initialization  
engine = pyttsx3.init()  
  
# Set properties \_before\_ you add things to say  
engine.setProperty('rate', 150)    # Speed percent (can go over 100)  
engine.setProperty('volume', 0.9)  # Volume 0-1  
  
# Queue up things to say.  
# There will be a short break between each one  
# when spoken, like a pause between sentences.  
engine.say("You've got mail!")  
engine.say("You can queue up multiple items")  
  
# Flush the say() queue and play the audio  
engine.runAndWait()  
  
# Program will not continue execution until  
# all speech is done talking

* Change voice and language

The voices available will depend on what your system has installed. You can get a list of available voices on your machine by pulling the **voices** property from the engine. Note that the voices you have available on your computer might be different from someone else's machine. There is a default voice set so you are not required to pick a voice. This is only if you want to change it from the default.

In Windows, you can learn more about installing other languages with this Microsoft support article, [How to download Text-to-Speech languages for Windows 10](https://support.office.com/en-us/article/how-to-download-text-to-speech-languages-for-windows-10-d5a6b612-b3ae-423f-afa5-4f6caf1ec5d3). It also covers how to install espeak open source languages.

You can get a list of available voices like this:

# Print all available voices  
**import** pyttsx3  
engine = pyttsx3.init()  
  
voices = engine.getProperty('voices')  
**for** voice **in** voices:  
    **print**("Voice:")  
    **print**(" - ID: %s" % voice.id)  
    **print**(" - Name: %s" % voice.name)  
    **print**(" - Languages: %s" % voice.languages)  
    **print**(" - Gender: %s" % voice.gender)  
    **print**(" - Age: %s" % voice.age)

Example output from my Windows 10 machine with three voices available.

**Voice:**

**- ID: HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS\_MS\_EN-US\_DAVID\_11.0**

**- Name: Microsoft David Desktop - English (United States)**

**- Languages: []**

**- Gender: None**

**- Age: None**

**Voice:**

**- ID: HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS\_MS\_EN-US\_ZIRA\_11.0**

**- Name: Microsoft Zira Desktop - English (United States)**

**- Languages: []**

**- Gender: None**

**- Age: None**

**Voice:**

**- ID: HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS\_MS\_RU-RU\_IRINA\_11.0**

**- Name: Microsoft Irina Desktop - Russian**

**- Languages: []**

**- Gender: None**

**- Age: None**

Set the voice you want to use with the **setProperty()** method on the engine. For example, using voice IDs found earlier, this is how you would set the voice. This example shows how to set one voice to say soemthing, and then use a different voice from a different language to say something else.

**import** pyttsx3  
engine = pyttsx3.init()  
  
# Voice IDs pulled from engine.getProperty('voices')  
# These will be system specific  
en\_voice\_id = "HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS\_MS\_EN-US\_ZIRA\_11.0"  
ru\_voice\_id = "HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS\_MS\_RU-RU\_IRINA\_11.0"  
  
# Use female English voice  
engine.setProperty('voice', en\_voice\_id)  
engine.say('Hello with my new voice')  
  
# Use female Russian voice  
engine.setProperty('voice', ru\_voice\_id)  
engine.say('Привет. где хакер')  
  
engine.runAndWait()

* Conclusion

After reading this you should feel comfortable using Python for basic text-to-speech applications on all major platforms. What uses can you think of?