Para poder usar este módulo primero hay que instalarlo usando el comando: pip install SpeechRecognition. Luego de instalarlo comprobamos la version dentro de python con el siguiente comando: sr.\_\_version\_\_

Lo primero que tenemos que hacer para poder usar correctamente este módulo es crear una instancia de la clase Recognizer que es tan fácil como hacer:

r= sr.Recognizer()

Cada instancia de la clase Recognizer, tiene 7 métodos:

* recognize\_bing(): [Microsoft Bing Speech](https://azure.microsoft.com/en-us/services/cognitive-services/speech/)
* recognize\_google(): [Google Web Speech API](https://w3c.github.io/speech-api/speechapi.html)
* recognize\_google\_cloud(): [Google Cloud Speech](https://cloud.google.com/speech/) - requires installation of the google-cloud-speech package
* recognize\_houndify(): [Houndify](https://www.houndify.com/) by SoundHound
* recognize\_ibm(): [IBM Speech to Text](https://www.ibm.com/watson/services/speech-to-text/)
* recognize\_sphinx(): [CMU Sphinx](https://cmusphinx.github.io/) - requires installing PocketSphinx
* recognize\_wit(): [Wit.ai](https://wit.ai/)

De estos siete métodos solamente el metodo recognize\_sphinx() es capaz de trabajar sin conexión a internet, de no estar conectado a internet, podemos recibir un RequestError.

Supported File Types:

* WAV: must be in PCM/LPCM format
* AIFF
* AIFF-C
* FLAC: must be native FLAC format; OGG-FLAC is not supported

Explicación de cómo se usa el código en su forma más básica, dentro del script **S\_R.py**

Podemos acortar la cantidad de cadenas de texto que se reciben, acortando la duración:

Para acortar la duración lo que hacemos es que cuando estamos declarando la clase **AudioData**, podemos hacerlo usando el metodo **record**, pasándoselo al recognizer **r**, de la clase **sr.Recognizer** como sigue:

audio\_short = **r. record (source, offset = 2.6, duration = 4)**

Siendo 4 la cantidad de tiempo en segundos.

**offset**: el parámetro que indica en que momento del audio se comienza a grabar.

Si el audio tiene ruido le pasamos un metodo antes, este metodo es como el sort() de las listas, solo hay que pasárselo, no guardarlo en otra variable ya que este afecta la variable original.

**r.adjust\_for\_ambient\_noise(source)**

Por su puesto, puede haber muchas posibilidades cuando estamos reconociendo un audio, para ver todas las posibilidades podemos usar:

r.recognize\_google(audio, show\_all=True)

Esto nos devolverá un diccionario con todas las posibilidades.