```
    def md5(self,fname):
        hash_md5 = hashlib.md5()
        with open(fname, "rb") as f:
            for chunk in iter(lambda: f.read(4096), b""):
                hash_md5.update(chunk)
        return hash_md5.hexdigest()

    def selectSounds(self, minDuration=config.MINSOUNDDURATION, maxDuration=config.MAXSOUNDDURATION, geo=[-10,52,4000]):
        """
        Diese Methode wÃ¤hlt Sounds aufgrund folgender Parameter aus der Freesound Library
        Parameter: minimale Dauer, maximale Dauer, Geotags: Breitengrad, LÃ¤ngengrad, Entfernung (Radius) als INT

        Text Search Request:
       >>> sounds = c.text_search(
       >>>     query="dubstep", filter="tag:loop", fields="id,name,url"
       >>> )
       >>> for snd in sounds: print snd.name

        Geotag Filter:
            #filter={!geofilt sfield=geotag pt=<LATITUDE>,<LONGITUDE> d=<MAX_DISTANCE_IN_KM>}
        """
        if not (self.state):
            return False
        if config.DEBUG:
            print ("fetching sound data from freesounds")
            print ("geotags:")
            print (geo)
        soundGeoTagging = geo
        start = time.time()
        queryFilter = "{{!geofilt sfield=geotag pt={0},{1} d={2}}}".format(geo[0],geo[1],geo[2])
        #queryFilter = "type:wav {!geofilt sfield=geotag pt=13,52 d=2000}"
        queryFields ="id,name,duration,md5,type,previews"
        sounds = self.fsClient.text_search(filter=queryFilter,fields=queryFields, page_size=config.PAGESIZE)
        stop  = time.time()
        if config.DEBUG:
            print ("dauer fÃ¼r freesounds abruf: ")
            print (stop-start)
        return self.filterByDuration(sounds, minDuration, maxDuration)

    def downloadSounds(self, soundsObject):
        if not (self.state):
            return False
        #self.fsClient.set_token(config.OAUTHTOKEN, "oauth")
        """
        Download Sound Files
        Erwartet ein Sound Objekt mit einer Liste von Sounds
        """
        i = 0
        for sound in soundsObject:
            if (i >= 0) & (i < config.COUNT):
                self.nameFileByIndex(sound, i)
                #filename = "sound_"+str(i)+".wav"
                                #sound.retrieve_preview(self.path_name, name=filename)
                i += 1
            else:
                return
        return
    def nameFileByIndex(self, soundObject, i):
        if not (self.state):
            return False
        filename = "sound_"+str(i)
        if config.DEBUG:
            print("\t\tDownloading:", soundObject.name)
            print("as: "+filename)
        soundObject.retrieve_preview(self.path_name, name=filename)
        """
        Loglevel ffmpeg
            -loglevel [repeat+]loglevel | -v [repeat+]loglevel
```