ПРИКЛАД ВИКОРИСТАННЯ JSON

**def from\_url**(self, url):  
 *"""Sets the url for the video.* ***:param*** *str url:  
 The url to the YouTube video.  
 """* self.\_video\_url = url  
  
 # Reset the filename and videos list in case the same instance is  
 # reused.  
 self.\_filename = **None** self.\_videos = []  
  
 # Get the video details.  
 video\_data = self.get\_video\_data()  
  
 # Set the title from the title.  
 self.title = video\_data.get("args", {}).get("title")  
  
 # Rewrite and add the url to the javascript file, we'll need to fetch  
 # this if YouTube doesn't provide us with the signature.  
 js\_partial\_url = video\_data.get("assets", {}).get("js")  
 **if** js\_partial\_url.startswith('//'):  
 js\_url = 'http:' + js\_partial\_url  
 **elif** js\_partial\_url.startswith('/'):  
 js\_url = 'https://youtube.com' + js\_partial\_url  
  
 # Just make these easily accessible as variables.  
 stream\_map = video\_data.get("args", {}).get("stream\_map")  
 video\_urls = stream\_map.get("url")  
  
 # For each video url, identify the quality profile and add it to list  
 # of available videos.  
 **for** i, url **in** enumerate(video\_urls):  
 log.debug("attempting to get quality profile from url: %s", url)  
 **try**:  
 itag, quality\_profile = self.\_get\_quality\_profile\_from\_url(url)  
 **if not** quality\_profile:  
 log.warn("unable to identify profile for itag=%s", itag)  
 **continue  
 except** (TypeError, KeyError) **as** e:  
 log.exception("passing on exception %s", e)  
 **continue** # Check if we have the signature, otherwise we'll need to get the  
 # cipher from the js.  
 **if** "signature=" **not in** url:  
 log.debug("signature not in url, attempting to resolve the "  
 "cipher.")  
 signature = self.\_get\_cipher(stream\_map["s"][i], js\_url)  
 url = "{0}&signature={1}".format(url, signature)  
 self.\_add\_video(url, self.filename, \*\*quality\_profile)  
 # Clear the cached js. Make sure to keep this at the end of  
 # `from\_url()` so we can mock inject the js in unit tests.  
 self.\_js\_cache = **None**