Можливості пакету JSON

JSON (JavaScipt Object Notation) – це текстовий формат обміну даними між комп’ютерами. Він заснований на синтаксисі мови програмування JavaScript. Модуль json дозволяє кодувати та розкодовувати дані в зручному форматі. Основи використання:

json.dump(obj, fp, skipkeys=False, ensure\_ascii=True, check\_circular=True, allow\_nan=True, cls=None, indent=None, separators=None, default=None, sort\_keys=False, \*\*kw)- серіалізує obj як відформатований JSON потік.

Якщо skipkeys=Truе, то ключі словника, які не є базового типу (str, float, bool, None) будуть пропущені, замість того, щоб збуджувати помилку TypeError.

Якщо ensure\_ascii=True, то всі у виводі будуть лише ASCII символи.

Якщо check\_circular=False, то перевірка циклічних посилань буде пропущена, а такі посилання викликатимуть OverflowError.

Якщо allow\_nan=False, при спробі серіалізувати значення з комою, яка виходить за допустимі межі, тоді буде збуджуватись помилка ValueError у строгій відповідності зі специфікацією JSON.

Якщо indent - невід’ємне число або рядок, то масиви і об’єкти в JSON будуть виводитися з таким рівнем відступу. Якщо рівень відступу 0, від’ємний або «», то будуть просто використовуватись нові рядки. Значення по замовчуванню (None) вибирає найбільш компактне відображення. Якщо indent – це рядок ( такий як ‘\t’), то цей рядок використовуватись у якості відступу.

Якщо sort\_keys=True, то словник відсортується за ключем.

json.dumps(obj, fp, skipkeys=False, ensure\_ascii=True, check\_circular=True, allow\_nan=True, cls=None, indent=None, separators=None, default=None, sort\_keys=False, \*\*kw) – серіалізує obj в рядок JSON-формату. Аргументи мають те саме значення, що і для dump().

Ключі в парах ключ/значення JSON-у завжди типу str. Коли словник конвертується в JSON, всі ключі словника перетворюються у тип str. Як результат, якщо словник конвертований у JSON і потім назад його сконвертувати, то отриманий словник може відрізнятись від того, який був з самого початку. Тобто loads(dumps(x)) != x, якщо х має ключі не типу str.

json.load( fp, cls=None, object\_hook=None, parse\_float=None, parse\_int=None, parse\_constant=None, object\_pairs\_hook=None, \*\*kw) – дисеріалізує JSON з fp.

object\_hook – це необов’язкова функція, яка застосовується до результату декодування об’єкта (dict). Використовуватися буде значення, яке повертає ця функція, а не отриманий словник.

object\_pairs\_hook - це необов’язкова функція, яка застосовується до результату декодування об’єкта з певною послідовністю пар ключ/значення. Використовуватись буде результат, який повертає функція, замість вихідного словника. Якщо є заданий також object\_hook, то пріоритет віддається object\_pairs\_hook.

parse\_float – якщо визначений, буде викликаний для кожного значення JSON з плаваючою крапкою. По замовчуванню еквівалентно float(num\_str).

parse\_int – якщо визначений, буде викликаний для наступних рядків: “-Infinity”, “Infinity”, “NaN”. Можна використовувати для збудження винятків при виявленні помилкових чисел JSON.

Якщо не вдається дисеріалізувати JSON, то буде збуджена виняток ValueError.

json.loads( s, encoding=None, object\_hook=None, parse\_float=None, parse\_int=None, parse\_constant=None, object\_pairs\_hook=None, \*\*kw) – дисеріалізує s (екземпляр str, який містить документ JSON) в об’єкт Python. Решта аргументів мають те саме значення, що і в json.load(), крім encoding, який ігнорується.

Якщо дані, які були дисеріалізовані, недійсного значення, то буде збуджений виняток JSONDecodeError.

**Кодувальники та декодувальники**

Class json.JSONDecoder( object\_hook=None, parse\_float=None, parse\_int=None, parse\_constant=None, strict=True, object\_pairs\_hook=None) – простий кодувальник JSON. Виконує наступні перетворення при декодуванні: (JSON -> Python) object -> dict, array -> list, string -> unicode, number(int) -> int, long, number(real) -> float, true -> True, false -> False, null -> None.

object\_hook, якщо визначений, буде викликаний для результату декодування кожного JSON об’єкту і повернене ним значення буде використане замість декодованого значення.

object\_pairs\_hook, якщо визначений, буде викликаний для декодованого об’єкту JSON з упорядкованим списком пар. Повернене значення цього методу буде використовуватись замість декодованого словника.

parse\_float, якщо визначений, буде викликаний для кожного значення JSON з плаваючою крапкою для його декодування.

parse\_int, якщо визначений, буде викликаний для рядка з числовим значенням JSON, щоб його декодувати.

parse\_constant

Якщо strict=False ( по замовчуванню True), тоді всередині рядка можуть використоуватись керуючі символи. В даному контексті керуючі символи – це символи з кодами в інтервалі 0-31, включаючи ‘\t’ (tab), ‘\n’, ‘\r’, ‘\0’.

decode(s) – повертає представлення Python для s (екземпляр str, що містить JSON документ). JSONDecodeError буде викликаний, якщо JSON документ недійсний.

Raw\_decode(s) – декодує JSON документ з s і повертає кортеж з 2-ох елементів: представлення Python та індекс рядка s, на якому закінчується документ. Це можна використовувати для декодування JSON документу з рядка, який може мати додаткові дані в кінці.

Class json.JSONEncoder( skipkeys=False, ensure\_ascii=True, check\_circular=True, allow\_nan=True, sort\_keys=False, indent=None, separators=None, default=None)

(Python -> JSON) dict -> object; list, tuple -> array; str, unicode -> string; int, long, float -> number; True -> true; False -> false; None -> null.

Якщо skipkeys=False, збуджується виняток TypeError при спробі закодувати ключ, який не є str, float, int, None. В іншому випадку ці елементи ігноруються.

Якщо ensure\_ascii=True, то всі не ASCII символи у виводі екрануються послідовностями \uXXXX і результатом є рядок, який містить лише ASCII символи.

Якщо check\_circular=True, тоді списки, словники і вручну закодовані об’єкти перевіряються на циклічні посилання при кодуванні для запобігання нескінченної рекурсії (що може призвести до OverflowError).

Якщо sort\_keys=True, то словник, що виводиться, буде відсортований по ключах.

Якщо indent невід’ємне число, тоді масиви та об’єкти JSON будуть виведені з цим рівнем відступу.

Якщо задано, то default – це функція, яка викликається для об’єктів, які були серіалізовані. Вона повинна повертати JSON кодовану версію для об’єкту або збуджувати TypeError.

Якщо encoding не None, то всі рядки, які вводяться будуть перетворені в юнікод, використовуючи це кодування.

encode(o) – повертає рядкове представлення JSON для структури даних Python o.

Iterencode(o) – кодує переданий об’єкт о і видає кожне рядкове представлення, як тільки воно стає доступним.

Специфікація RFC говорить, що імена в об’єкті JSON повинні бути унікальними, але не визначає, як повинні оброблятися імена, що повторюються. По замовчуванню, цей модуль не викликає винятку, а просто ігнорує всі пари ключ/значення, крім останнього.