claujson

vztpv@naver.com

ClauJson

- experimental parallel json parser (using simd and multi-thread)
- scj3 (using modified simdjson and clau-parser code)
- use mimalloc, ... for performance.

init

first, call init function.

Data

class
json data : 3, 4.5, “hello”, [1, 2, 3], and { “key”: “value” } , ...

Data`s destructor do not delete object or array.
(if Data are object or array)

if you want to delete object or array, call claujson::clean function.

use std::move! no copy constructor, no assignment.

if need copy, then use clone() method.

Array

class

std::vector<Data> child;

Array`s Destructor delete child! (if child is array or object)

Object

class

std::vector<Data> key_vec; // must string!

std::vector<Data> val_vec;

Object`s Destructor delete child! (if val is array or object)

clean

function

for Data.

check dup of key

- in object.

parse, parse_str

- thr_num <- 0 : use all thread of cpu.
- not thread-safe
- parse(_from_file)

save, save_parallel

- save - slow
- save_parallel - not thread-safe; when use this function, must not access data.

log

- claujson::log.no_print();

check

- A. no check dup of key when parsing.
- B. Data j(”test string”); // utf-8 check, unicode check. etc..
- C. object <- because of A, like array, object also access using index. (order by input)