
Subject: Re: Core 2019

Posted by [Novo](#) on Sun, 09 Jun 2019 15:15:19 GMT

[View Forum Message](#) <> [Reply to Message](#)

I hacked your TIMING macro and made a similar RMEMUSE one:

```
namespace Upp {

struct MemInspector {
protected:
    static bool active;

    const char *name;
    int      call_count;
    int      min_mem;
    int      max_mem;
    int      max_nesting;
    int      all_count;
    StaticMutex mutex;

public:
    MemInspector(const char *name = NULL); // Not String !!!
    ~MemInspector();

    void Add(int mem, int nesting);

    String Dump();

class Routine {
public:
    Routine(MemInspector& stat, int& nesting)
    : nesting(nesting), stat(stat) {
        ++nesting;
    }

    ~Routine() {
        --nesting;
        int mem = MemoryUsedKb();
        stat.Add(mem, nesting);
    }

protected:
    int& nesting;
    MemInspector& stat;
};

    static void Activate(bool b) { active = b; }
};
```

```

bool MemInspector::active = true;

MemInspector::MemInspector(const char *_name) {
    name = _name ? _name : "";
    all_count = call_count = max_nesting = min_mem = max_mem = 0;
}

MemInspector::~MemInspector() {
    Mutex::Lock ____(mutex);
    StdLog() << Dump() << "\r\n";
}

void MemInspector::Add(int mem, int nesting)
{
    // mem = MemoryUsedKb() - mem;
    Mutex::Lock ____(mutex);
    if(!active) return;
    all_count++;
    if(nesting > max_nesting)
        max_nesting = nesting;
    if(nesting == 0) {
        if(call_count++ == 0)
            min_mem = max_mem = mem;
        else {
            if(mem < min_mem)
                min_mem = mem;
            if(mem > max_mem)
                max_mem = mem;
        }
    }
}

String MemInspector::Dump() {
    Mutex::Lock ____(mutex);
    String s = Sprintf("MEMUSE %-15s: ", name);
    if(call_count == 0)
        return s + "No active hit";
    return s
        << "min: " << min_mem
        << ", max: " << max_mem
        << Sprintf(", nesting: %d - %d", max_nesting, all_count);
}

}

#define RMEMUSE(x) \

```

```
static UPP::MemInspector COMBINE(sMemStat, __LINE__)(x); \
static thread_local int COMBINE(sMemStatNesting, __LINE__); \
UPP::MemInspector::Routine COMBINE(sMemStatR, __LINE__)(COMBINE(sMemStat,
__LINE__), COMBINE(sMemStatNesting, __LINE__))
```

What I'm getting in case of HPAGE = 7 * 8192 is

TIMING Chunk : 4108.80 s - 22.66 ms (4108.80 s / 181363), min: 1.00 ms, max: 1.24 s ,
nesting: 0 - 181363

MEMUSE Chunk : min: 30844, max: 341052, nesting: 0 - 181363

TIMING Read Data : 228.28 s - 228.28 s (228.28 s / 1), min: 228.28 s , max: 228.28 s ,
nesting: 0 - 1

top is saying max used memory (RES) is ~771 Mb.
