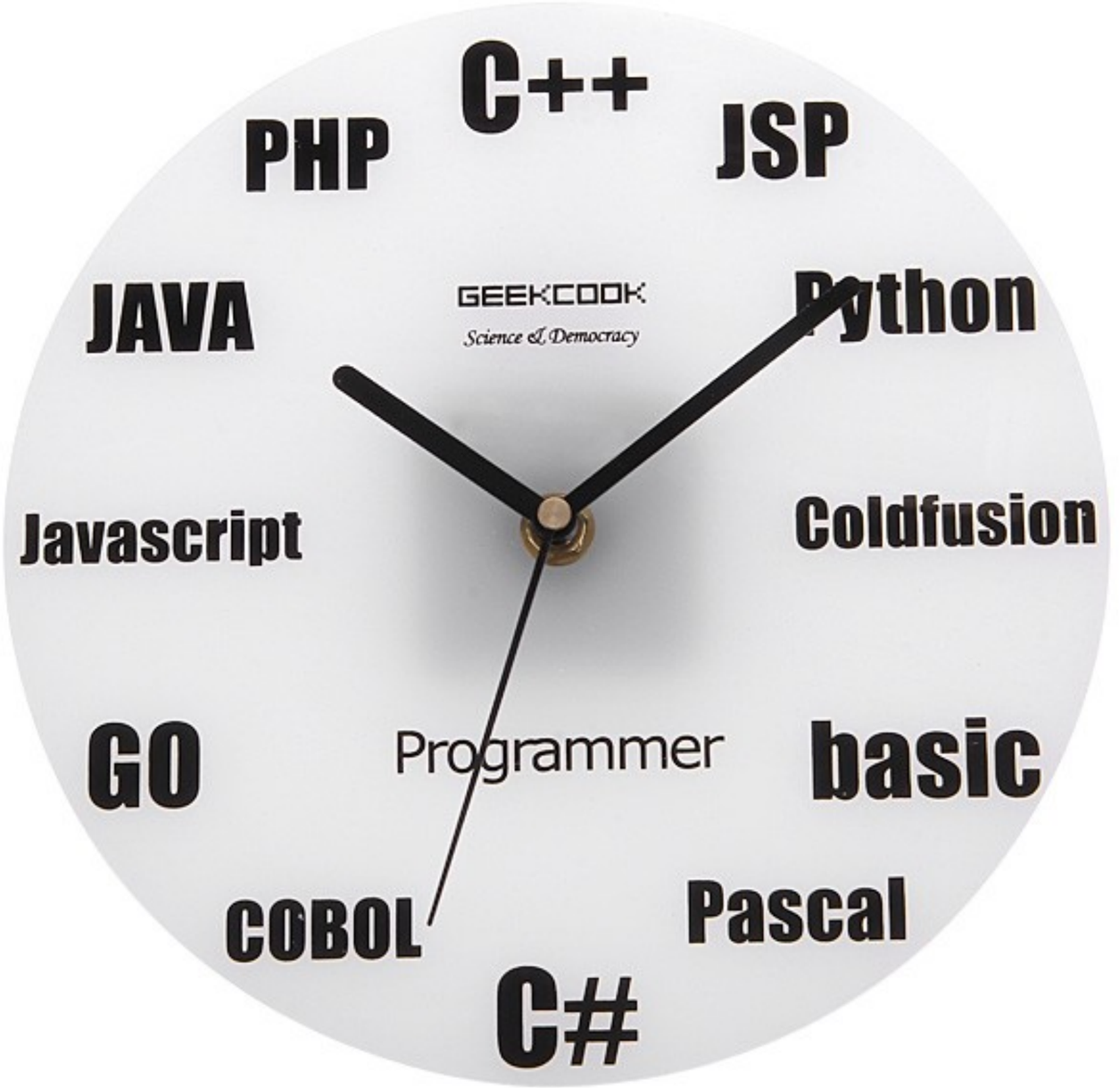
Objektinis Programavimas

Programos spartos matavimas



Turinys

Motyvacija (1)

- Viena iš labiausiai akivaizdžių ir reikalingų bibliotekų, kurią turi turėti programavimo kalbą yra skirta darbui su laiku.
- Tokią biblioteką suprojektuoti yra gana sudėtinga.
- Praeityje naudoti interfeisai (`C`, `POSIX`) keitėsi matavimo tikslumu: nuo sekundžių iki milisekundžių, tuomet iki mikrosekundžių ir galiausiai iki nanosekundžių.

Motyvacija (2)

- Bėda ta, kad kiekvienam atnaujinimui buvo sukurta vis nauja sąsaja (interfeisas).
- Dėl šios priežasties C++11 standarte buvo pasiūlyta tikslumui neutrali biblioteka.
- Ši biblioteka paprastai vadinama **chrono biblioteka**, nes jos funkcijos yra apibrėžtos `<chrono>`.

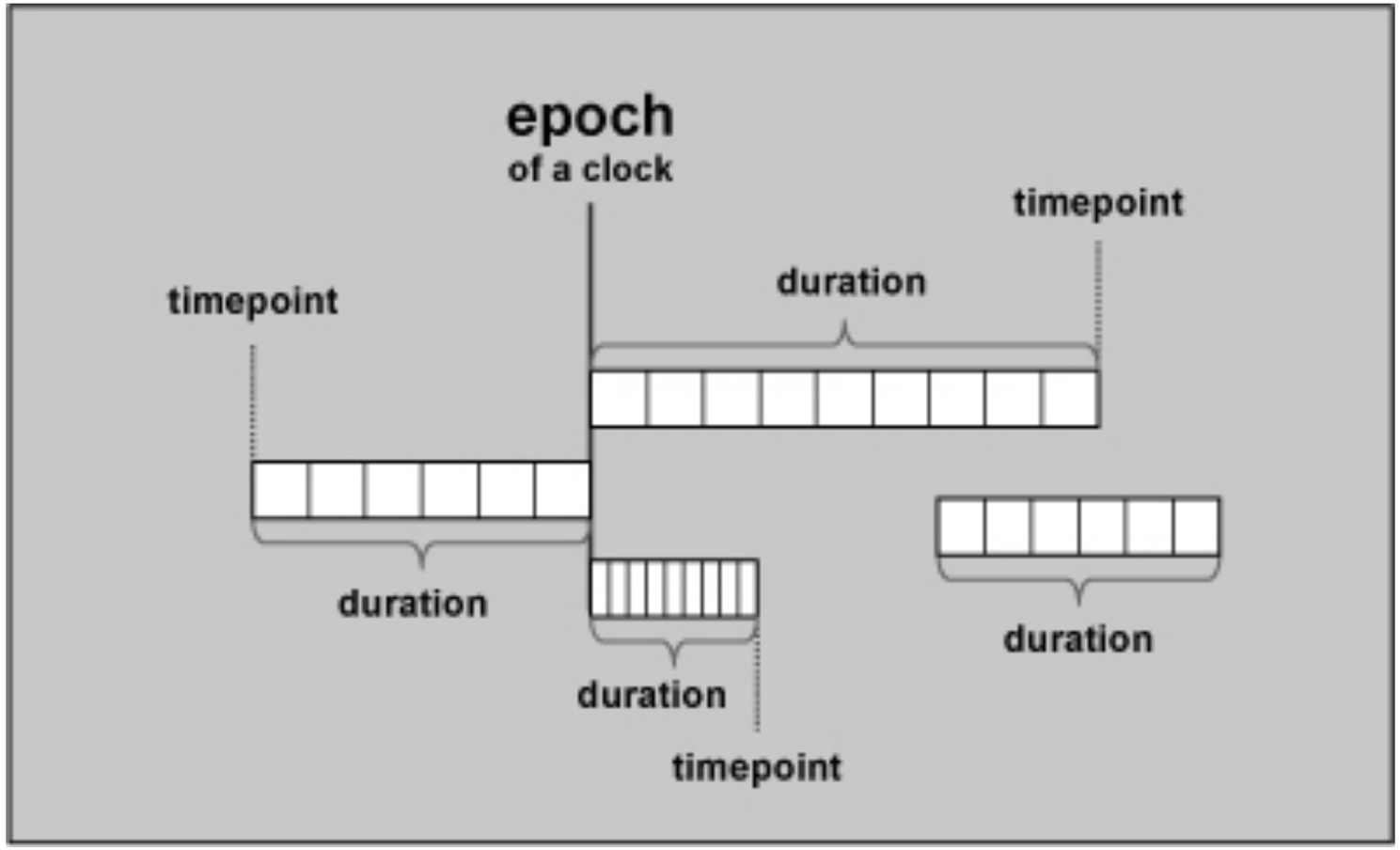
`std::chrono` biblioteka (1)

- chrono biblioteka sukurta atsižvelgiant į tai, kad laikmačiai (*timers*) ir laikrodžiai (*clocks*) gali skirtingose sistemose skirtis, o laikui bėgant tikslėti.
- Tikslumui neutrali koncepcija sukurta atskyrus trukmę (*duration*) ir laiko tašką ("*timepoint*") nuo konkrečių laikrodžių (*clocks*).

`std::chrono` biblioteka (2)

- **Trukmė** (*duration*) yra apibrėžiama, kaip **ticks**'ų skaičius per tam tikrą laiko vienetą. Pvz. "3 minučių trukmė" suprantama, kaip 3 (vienos) minutės trukmės **ticks**'ai.
- **Laiko taškas** (*timepoint*) apibrėžiamas per trukmę ir laiko pradžią (vadinamą **epocha**). Tipiškas pavyzdys yra laiko taškas žymintis: "2000 metų vidurnaktį", kuris suprantamas kaip "1 262 300 400 sekundžių nuo 1970-01-01 dienos" (ši diena yra UNIX ir POSIX sistemų laikrodžių epocha).

`std::chrono` biblioteka (3)



`std::chrono` naudojimo pavyzdys

```
#include <iostream>
#include <chrono>
#include <vector>

int main() {
    auto start = std::chrono::high_resolution_clock::now();  // Paleisti
    std::vector<int> v;
    for (size_t i = 0; i < 10000000; ++i)
        v.push_back(i);
    auto end = std::chrono::high_resolution_clock::now();    // Stabdyti
    std::chrono::duration<double> diff = end-start;          // Skirtumas (s)
    std::cout << "10 000 000 elementų užpildymas užtruko: "
              << diff.count() << " s\n";
}
```

`std::chrono` trūkumai

- C++11 standarte atsiradusi laiko matavimui skirta `std::chrono` biblioteka yra labai funkcionali ir išsprendė ankstesniuose C++ standartuose buvusias problemas.
- Deja, bet `std::chrono` naudojimas nėra labai patogus ir reikalauja nemažai papildomo kodo rašyti.

Laiko matavimo klasė (1)

```
#include <chrono>

class Timer {
private:
    std::chrono::time_point<std::chrono::high_resolution_clock> start;
public:
    Timer() : start{std::chrono::high_resolution_clock::now()} {}
    void reset() {
        start = std::chrono::high_resolution_clock::now();
    }
    double elapsed() const {
        return std::chrono::duration<double>
        (std::chrono::high_resolution_clock::now() - start).count();
    }
};
```

Laiko klasės naudojimo pavyzdys

```
#include <iostream>
#include <chrono>
#include <vector>
#include "Timer.h"

int main() {
    Timer t;  // Paleisti
    std::vector<int> v;
    for (size_t i = 0; i < 10000000; ++i)
        v.push_back(i);
    std::cout << "10 000 000 elementų užpildymas užtruko: "
              << t.elapsed() << " s\n";
}
```

Laiko matavimo klasė (2)

```
#include <chrono>

class Timer {
private:
  // panaudojame using
  using hrClock = std::chrono::high_resolution_clock;
  using durationDouble = std::chrono::duration<double>;
  std::chrono::time_point<hrClock> start;
public:
    Timer() : start{ hrClock::now() } {}
    void reset() {
        start = hrClock::now();
    }
    double elapsed() const {
        return durationDouble (hrClock::now() - start).count();
    }
};
```

Komentarai dėl laiko matavimo

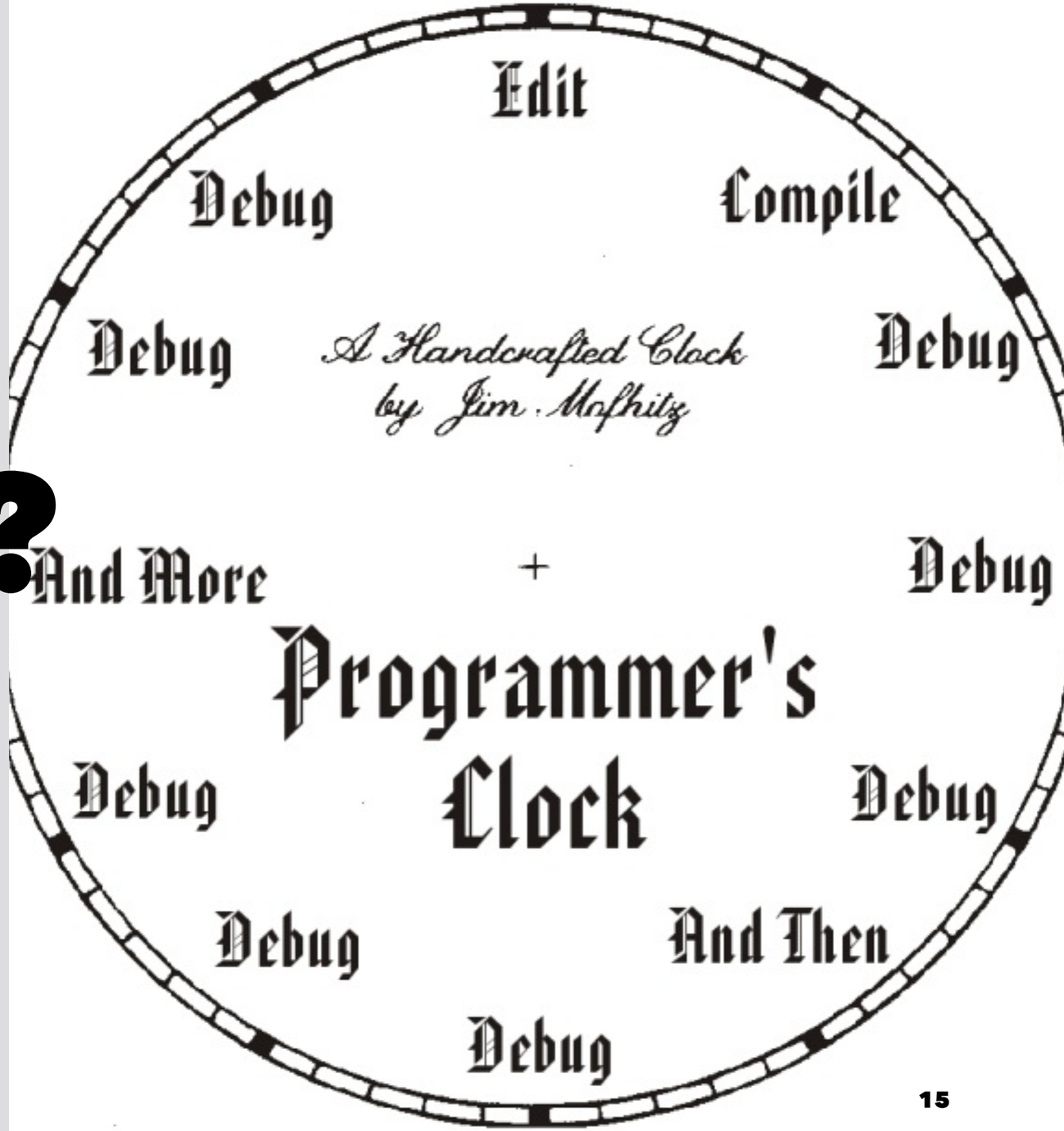
1. Įsitikinkite, kad sukompiliavote naudodame `release`, o ne debug, nes debug programa veikia gerokai lėčiau.
2. Eksperimentą pakartokite keletą kartų, pvz. 5-10 ir imkite vidurkinius rezultatus.
3. Eksperimentų metu geriausia, kad kompiuteris nedarytų jokių kitų veiksmų, galinčių įtakoti laiko matavimus.

Literatūra

1. **5.7. Clocks and Timers** iš *C++ Standard Library, The: A Tutorial and Reference, 2nd Edition*
2. http://en.cppreference.com/w/cpp/header/chrono


eat
code
code
code
code
Programmer's Clock
code
code
code
code
code
code
code

?


Edit
Debug
Compile
Debug
A Handcrafted Clock
by Jim Mofhitz
Debug
And More
+
Debug
Programmer's
Clock
Debug
Debug
Debug
And Then
Debug