

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Programowanie Współbieżne w C++ 4 fun && \$

Bartek 'BaSz' Szurgot

[bartosz.1.szurgot@nokia.com](mailto:bartosz.1.szurgot@nokia.com)

21 kwietnia 2016

# Ktoś Waść?!

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

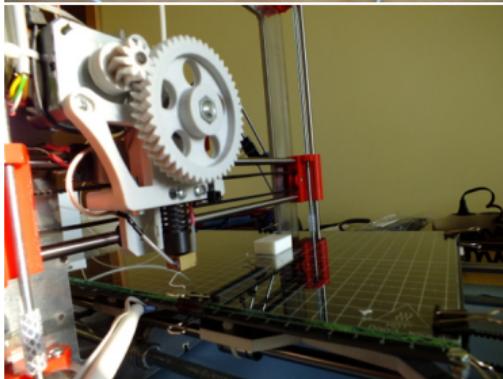
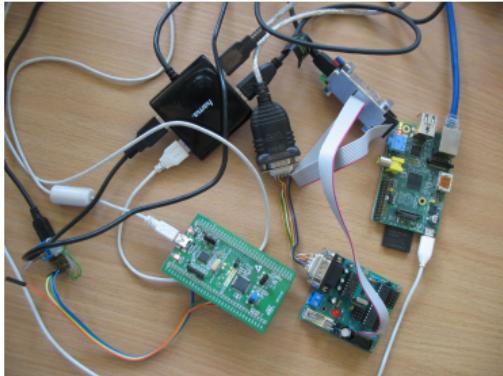
Co przyszłość  
niesie?

Zamiast  
zakończenia

# Co to za hałasy?

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Podstawa to dobre wrażenie...

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

***Lorem ipsum dolor sit amet, consectetur adipiscing elit,  
  sed do eiusmod tempor incididunt ut labore et dolore  
  magna aliqua. Ut enim ad minim veniam, quis nostrud  
  exercitation ullamco laboris nisi ut aliquip ex ea commodo  
  consequat. Duis aute irure dolor in reprehenderit in  
  voluptate velit esse cillum dolore eu fugiat nulla pariatur.  
  Excepteur sint occaecat cupidatat non proident, sunt in  
  culpa qui officia deserunt mollit anim id est laborum.***

A teraz...

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



**And Now For Something  
Completely Different**

# Pora na...

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Prawo Moore'a

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Prawo Moore'a

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

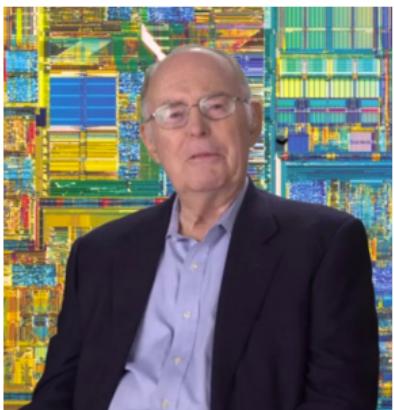
Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

*Liczba **tranzystorów** w procesorze podwaja się co około dwa lata.*

– Gordon Moore, 1965



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

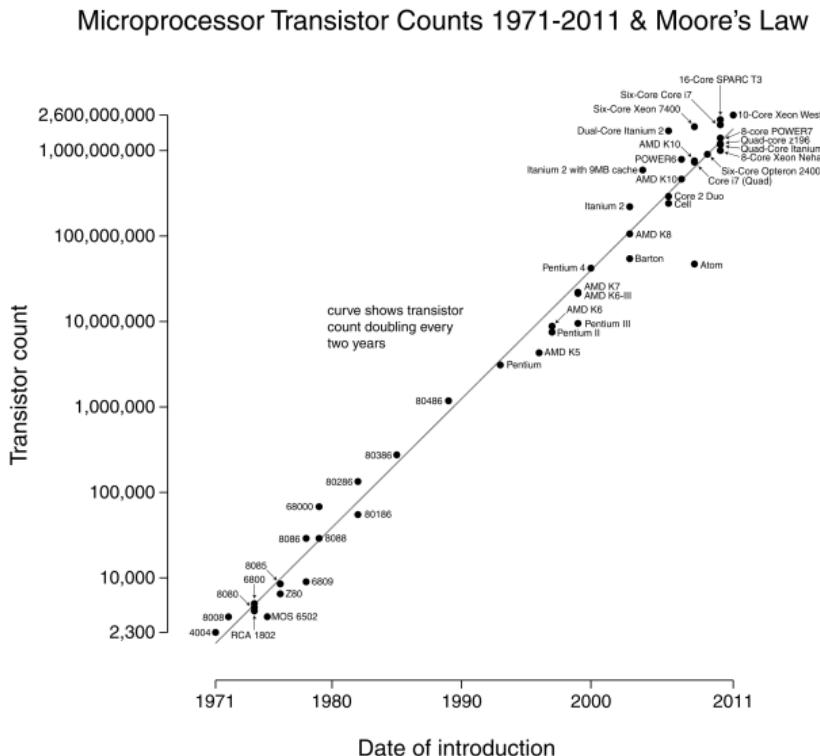
Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Prawo Moore'a graficznie



# Program sekwencyjny

```
1 auto out = 0;  
2 for(auto i=0; i<8*1000*1000; ++i)  
3     out += veryComplexStuff(i);  
4 cout << "answer:_" << out << endl;
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Program sekwencyjny

```
1 auto out = 0;  
2 for(auto i=0; i<8*1000*1000; ++i)  
3     out += veryComplexStuff(i );  
4 cout << "answer:_" << out << endl;
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Program sekwencyjny

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

```
1 auto out = 0;  
2 for(auto i=0; i<8*1000*1000; ++i)  
3     out += veryComplexStuff(i);  
4 cout << "answer:_" << out << endl;
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Program sekwencyjny

```
1 auto out = 0;  
2 for(auto i=0; i<8*1000*1000; ++i)  
3     out += veryComplexStuff(i );  
4 cout << "answer:_" << out << endl;
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Program sekwencyjny

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

```
1 auto out = 0;  
2 for(auto i=0; i<8*1000*1000; ++i)  
    out += veryComplexStuff(i);  
4 cout << "answer:_" << out << endl;
```

► Jak szybko się wykona?

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Program sekwencyjny

```
1 auto out = 0;  
2 for(auto i=0; i<8*1000*1000; ++i)  
    out += veryComplexStuff(i);  
4 cout << "answer:_" << out << endl;
```

- ▶ Jak szybko się wykona?
- ▶ Zależy od sprzętu!

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Podgląd wykonania

## ► Rok 2005:

- $t_{func} = 100s$
- Obciążenie systemu:



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Podgląd wykonania

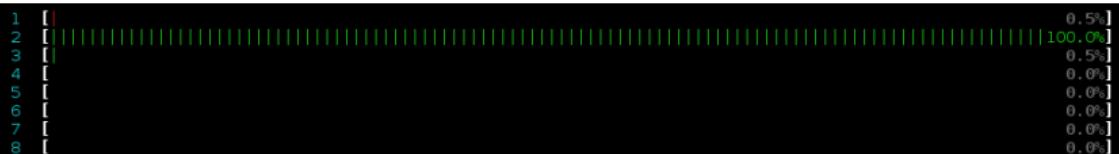
## ► Rok 2005:

- $t_{func} = 100s$
- Obciążenie systemu:



## ► Rok 2015:

- $t_{func} = 70s$
- Obciążenie systemu:



Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Podgląd wykonania

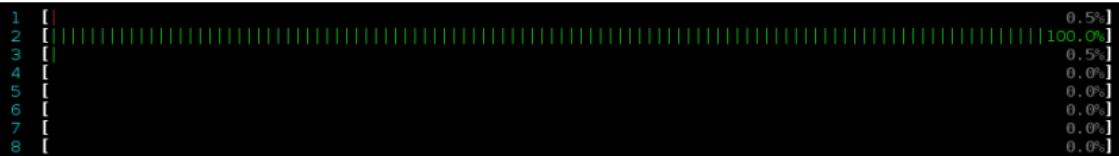
## ► Rok 2005:

- $t_{func} = 100s$
- Obciążenie systemu:



## ► Rok 2015:

- $t_{func} = 70s$
- Obciążenie systemu:



## ► Wniosek?

- Przybyło rdzeni
- $\frac{7}{8}$  rdzeni (87.5%) nie robi **nic!**
- Ogromne rezerwy mocy do wykorzystania!

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

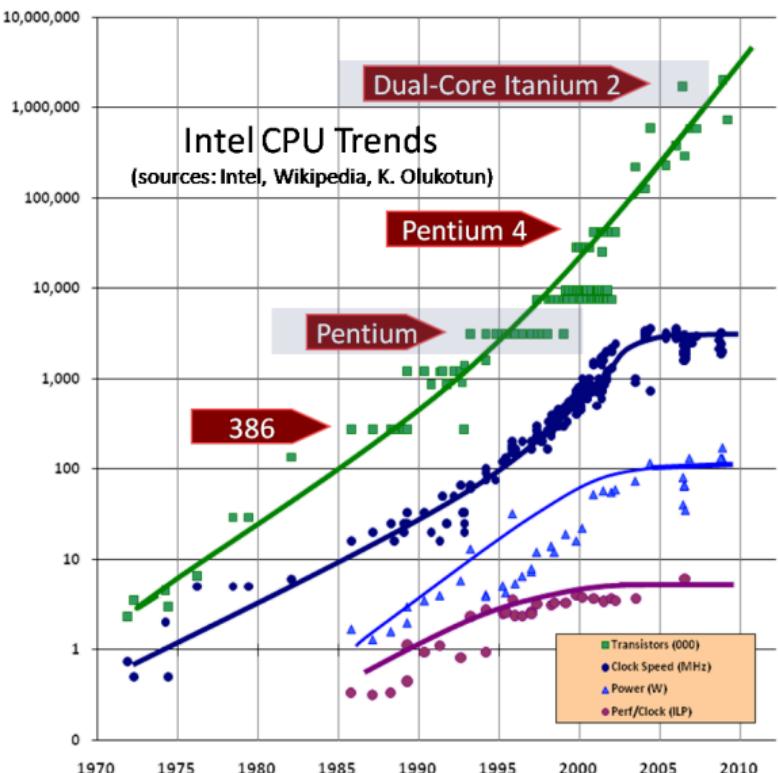
Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

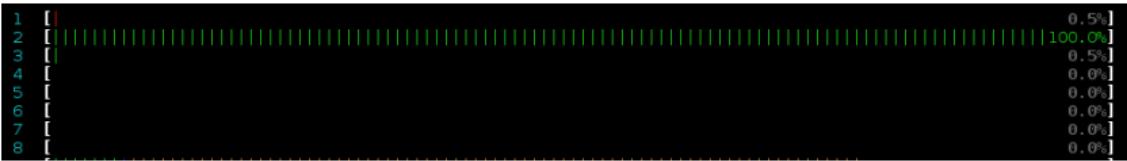
Zamiast  
zakończenia

# Prawo Moore'a w praktyce



# Problem

- ▶ Obecne obciążenie procesora:



Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

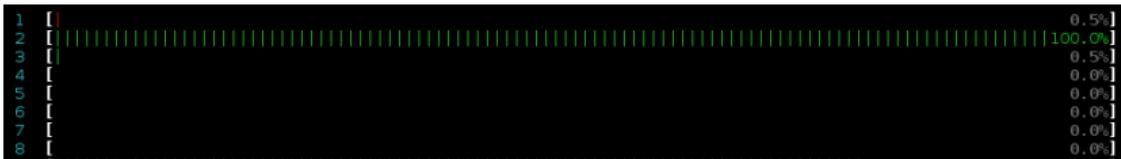
Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Problem

- ▶ Obecne obciążenie procesora:



- ▶ Oczekiwane obciążenie procesora:



- ▶ Jak ów cel osiągnąć?

# Problem

Programowanie  
Współbieżne  
w C++ 4fun&&\\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

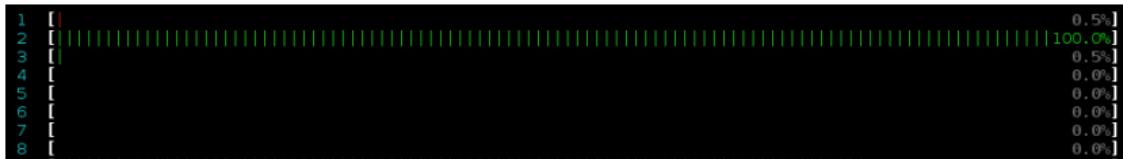
Przerwa

Współdzielenie  
inaczej

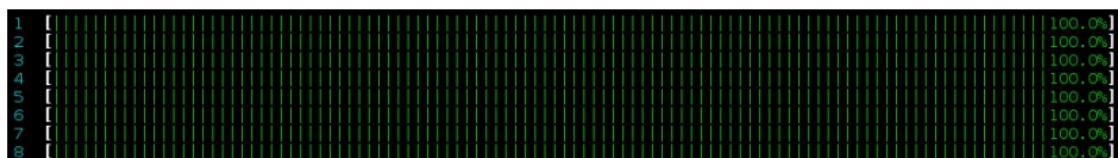
Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



- ▶ Obecne obciążenie procesora:



- ▶ Oczekiwane obciążenie procesora:

- ▶ Jak ów cel osiągnąć?
- ▶ Używać wszystkich rdzeni
- ▶ Zmienić program

# Szkic programu równoległego 1/2

## ► Punkty wyjściowy:

```
1 auto out = 0;  
2 for(auto i=0; i<8*1000*1000; ++i)  
3     out += veryComplexStuff(i);  
4 cout << "answer:_" << out << endl;
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Szkic programu równoległego 1/2

## ► Punkty wyjściowy:

```
1 auto out = 0;  
2 for(auto i=0; i<8*1000*1000; ++i)  
3     out += veryComplexStuff(i);  
4 cout << "answer:_" << out << endl;
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Szkic programu równoległego 1/2

## ► Punkty wyjściowy:

```
1 auto out = 0;
2 for(auto i=0; i<8*1000*1000; ++i)
3     out += veryComplexStuff(i);
4 cout << "answer:_" << out << endl;
```

## ► Wyciągnięcie algorytmu:

```
1 auto computePart(int begin, int end)
2 {
3     auto out = 0;
4     for(auto i=begin; i<end; ++i)
5         out += veryComplexStuff(i);
6     return out;
7 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Szkic programu równoległego 1/2

## ► Punkty wyjściowy:

```
1 auto out = 0;  
2 for(auto i=0; i<8*1000*1000; ++i)  
3     out += veryComplexStuff(i);  
4 cout << "answer:_" << out << endl;
```

## ► Wyciągnięcie algorytmu:

```
1 auto computePart(int begin, int end)  
2 {  
3     auto out = 0;  
4     for(auto i=begin; i<end; ++i)  
5         out += veryComplexStuff(i);  
6     return out;  
7 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Szkic programu równoległego 1/2

## ► Punkty wyjściowy:

```
1 auto out = 0;
2 for(auto i=0; i<8*1000*1000; ++i)
3     out += veryComplexStuff(i);
4 cout << "answer:_" << out << endl;
```

## ► Wyciągnięcie algorytmu:

```
1 auto computePart(int begin, int end)
2 {
3     auto out = 0;
4     for(auto i=begin; i<end; ++i)
5         out += veryComplexStuff(i);
6     return out;
7 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Szkic programu równoległego 1/2

## ► Punkty wyjściowy:

```
1 auto out = 0;
2 for(auto i=0; i<8*1000*1000; ++i)
3     out += veryComplexStuff(i );
4 cout << "answer:_" << out << endl;
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

## ► Wyciągnięcie algorytmu:

```
1 auto computePart(int begin, int end)
2 {
3     auto out = 0;
4     for(auto i=begin; i<end; ++i)
5         out += veryComplexStuff(i );
6     return out;
7 }
```

# Szkic programu równoległego 2/2

```
1 int main()
2 {
3     auto m = 1000*1000; // short for 'million'
4     int part[8];        // partial results
5     part[0] = computePart(0*m, 1*m); // TODO: thread #1
6     part[1] = computePart(1*m, 2*m); // TODO: thread #2
7     // etc...
8     part[7] = computePart(7*m, 8*m); // TODO: thread #8
9     auto out = accumulate( begin(part), end(part), 0u );
10    cout << "answer:_" << out << endl;
11 }
```

# Szkic programu równoległego 2/2

```
1 int main()
2 {
3     auto m = 1000*1000; // short for 'million'
4     int part[8];         // partial results
5     part[0] = computePart(0*m, 1*m); // TODO: thread #1
6     part[1] = computePart(1*m, 2*m); // TODO: thread #2
7     // etc...
8     part[7] = computePart(7*m, 8*m); // TODO: thread #8
9     auto out = accumulate( begin(part), end(part), 0u );
10    cout << "answer: " << out << endl;
11 }
```

# Szkic programu równoległego 2/2

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 int main()
2 {
3     auto m = 1000*1000; // short for 'million'
4     int part[8];        // partial results
5     part[0] = computePart(0*m, 1*m); // TODO: thread #1
6     part[1] = computePart(1*m, 2*m); // TODO: thread #2
7     // etc...
8     part[7] = computePart(7*m, 8*m); // TODO: thread #8
9     auto out = accumulate( begin(part), end(part), 0u );
10    cout << "answer: " << out << endl;
11 }
```

# Szkic programu równoległego 2/2

```
1 int main()
2 {
3     auto m = 1000*1000; // short for 'million'
4     int part[8];         // partial results
5     part[0] = computePart(0*m, 1*m); // TODO: thread #1
6     part[1] = computePart(1*m, 2*m); // TODO: thread #2
7     // etc...
8     part[7] = computePart(7*m, 8*m); // TODO: thread #8
9     auto out = accumulate( begin(part), end(part), 0u );
10    cout << "answer: " << out << endl;
11 }
```

# Szkic programu równoległego 2/2

```
1 int main()
2 {
3     auto m = 1000*1000; // short for 'million'
4     int part[8];         // partial results
5     part[0] = computePart(0*m, 1*m); // TODO: thread #1
6     part[1] = computePart(1*m, 2*m); // TODO: thread #2
7     // etc...
8     part[7] = computePart(7*m, 8*m); // TODO: thread #8
9     auto out = accumulate( begin(part), end(part), 0u );
10    cout << "answer: " << out << endl;
11 }
```

# Szkic programu równoległego 2/2

```
1 int main()
2 {
3     auto m = 1000*1000; // short for 'million'
4     int part[8];        // partial results
5     part[0] = computePart(0*m, 1*m); // TODO: thread #1
6     part[1] = computePart(1*m, 2*m); // TODO: thread #2
7     // etc...
8     part[7] = computePart(7*m, 8*m); // TODO: thread #8
9     auto out = accumulate( begin(part), end(part), 0u );
10    cout << "answer: " << out << endl;
11 }
```

# Szkic programu równoległego 2/2

```
1 int main()
2 {
3     auto m = 1000*1000; // short for 'million'
4     int part[8];         // partial results
5     part[0] = computePart(0*m, 1*m); // TODO: thread #1
6     part[1] = computePart(1*m, 2*m); // TODO: thread #2
7     // etc...
8     part[7] = computePart(7*m, 8*m); // TODO: thread #8
9     auto out = accumulate( begin(part), end(part), 0u );
10    cout << "answer: " << out << endl;
11 }
```

# Szkic programu równoległego 2/2

```
1 int main()
2 {
3     auto m = 1000*1000; // short for 'million'
4     int part[8];         // partial results
5     part[0] = computePart(0*m, 1*m); // TODO: thread #1
6     part[1] = computePart(1*m, 2*m); // TODO: thread #2
7     // etc...
8     part[7] = computePart(7*m, 8*m); // TODO: thread #8
9     auto out = accumulate( begin(part), end(part), 0u );
10    cout << "answer:_" << out << endl;
11 }
```

# Szkic programu równoległego 2/2

```
1 int main()
2 {
3     auto m = 1000*1000; // short for 'million'
4     int part[8];        // partial results
5     part[0] = computePart(0*m, 1*m); // TODO: thread #1
6     part[1] = computePart(1*m, 2*m); // TODO: thread #2
7     // etc...
8     part[7] = computePart(7*m, 8*m); // TODO: thread #8
9     auto out = accumulate( begin(part), end(part), 0u );
10    cout << "answer:_" << out << endl;
11 }
```

# Dlaczego wątki?

## 1. Maksymalne wykorzystanie krzemu

- ▶ Szybsze obliczenia
- ▶ Oszczędność prądu
  - ▶ Szybciej uzyskiwany wynik
  - ▶ Przejście w tryb uśpienia
- ▶ Szybka komunikacja
  - ▶ Dane lokalne
  - ▶ Współdzielenie

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Dlaczego wątki?

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

## 1. Maksymalne wykorzystanie krzemu

- ▶ Szybsze obliczenia
- ▶ Oszczędność prądu
  - ▶ Szybciej uzyskiwany wynik
  - ▶ Przejście w tryb uśpienia
- ▶ Szybka komunikacja
  - ▶ Dane lokalne
  - ▶ Współdzielenie

## 2. Minimalizacja opóźnień

- ▶ Powolne I/O w osobnym wątku
- ▶ Responsywność (GUI)

No i oczywiście...

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Dlaczego C++?

# C++98 – jak 1998...

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



# C++14 – jak 2014!

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



# Pora na...

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wątki vs. procesy

- ▶ Wątek (ang. *thread*)
- ▶ Proces (ang. *process*)

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wątki vs. procesy

- ▶ Wątek (ang. *thread*)
- ▶ Proces (ang. *process*)
- ▶ Pamięć:
  - ▶ Procesy:
    - ▶ Osobne stosy
    - ▶ Osobne sterty

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

BaSz

# Wątki vs. procesy

- ▶ Wątek (ang. *thread*)
- ▶ Proces (ang. *process*)
- ▶ Pamięć:
  - ▶ Procesy:

- ▶ Osobne stosy
- ▶ Osobne sterty

- ▶ Wątki:
  - ▶ Osobne stosy
  - ▶ Wspólna sterta

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

BaSz

# Wątki vs. procesy

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

- ▶ Wątek (ang. *thread*)
- ▶ Proces (ang. *process*)
- ▶ Pamięć:

- ▶ Procesy:
  - ▶ Osobne stosy
  - ▶ Osobne sterty
- ▶ Komunikacja:
  - ▶ Procesy:
    - ▶ Wiadomości
    - ▶ Sygnały

- ▶ Wątki:
  - ▶ Osobne stosy
  - ▶ Wspólna sterta

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Wątki vs. procesy

Programowanie  
Współbieżne  
w C++ 4fun&& \$

BaSz

- ▶ Wątek (ang. *thread*)
- ▶ Proces (ang. *process*)
- ▶ Pamięć:
  - ▶ Procesy:
    - ▶ Osobne stosy
    - ▶ Osobne sterty
  - ▶ Komunikacja:
    - ▶ Procesy:
      - ▶ Wiadomości
      - ▶ Sygnały
    - ▶ Wątki:
      - ▶ Wskaźniki
      - ▶ Struktury danych (sterta)

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wątki vs. procesy

- ▶ Wątek (ang. *thread*)
- ▶ Proces (ang. *process*)
- ▶ Pamięć:
  - ▶ Procesy:
    - ▶ Osobne stosy
    - ▶ Osobne sterty
  - ▶ Komunikacja:
    - ▶ Procesy:
      - ▶ Wiadomości
      - ▶ Sygnały
    - ▶ Wątek – aka. „lekki proces”
    - ▶ Proces może mieć wiele wątków
- ▶ Wątki:
  - ▶ Osobne stosy
  - ▶ Wspólna sterta
- ▶ Wątki:
  - ▶ Wskaźniki
  - ▶ Struktury danych (sterta)

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Współbieżność

## Współbieżność

Rodzaj przetwarzania, w którym kilka obliczeń jest realizowanych, w nachodzących okresach czasowych.

[https://en.wikipedia.org/wiki/Concurrent\\_programming](https://en.wikipedia.org/wiki/Concurrent_programming)

BaSz

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

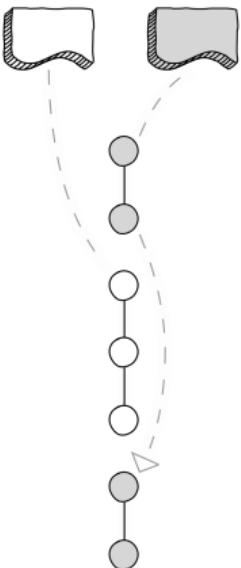
Zamiast  
zakończenia

# Współbieżność

## Współbieżność

Rodzaj przetwarzania, w  
którym kilka obliczeń jest  
realizowanych, w  
nachodzących okresach  
czasowych.

[https://en.wikipedia.org/wiki/Concurrent\\_programming](https://en.wikipedia.org/wiki/Concurrent_programming)



<http://bytearcher.com/articles/parallel-vs-concurrent/concurrent-1.svg>

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Równoległość

## Równoległość

Rodzaj przetwarzania, w którym kilka obliczeń jest realizowanych w tym samym czasie.

[https://en.wikipedia.org/wiki/Parallel\\_computing](https://en.wikipedia.org/wiki/Parallel_computing)

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

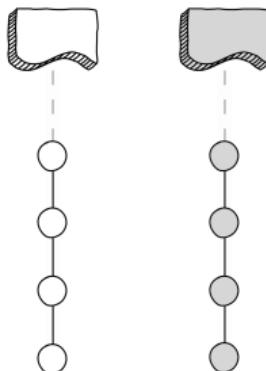
Zamiast  
zakończenia

# Równoległość

## Równoległość

Rodzaj przetwarzania, w którym kilka obliczeń jest realizowanych w tym samym czasie.

[https://en.wikipedia.org/wiki/Parallel\\_computing](https://en.wikipedia.org/wiki/Parallel_computing)



<http://bytearcher.com/articles/parallel-vs-concurrent/parallel-1.svg>

# Równoległość... a współbieżność

Programowanie  
Współbieżne  
w C++ 4fun&& \$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



# Równoległość... a współbieżność

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

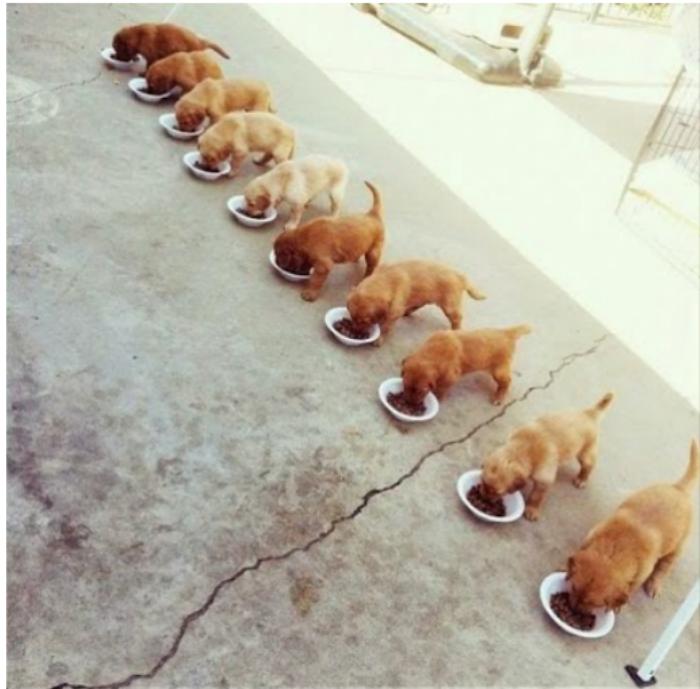
Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



# Pora na...

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wątek jest klasą

- ▶ `#include <thread>`
- ▶ `std::thread`

# Wątek jest klasą

- ▶ `#include <thread>`
- ▶ `std::thread`
- ▶ Najważniejsze operacje
  - ▶ Konstruktor – uruchamianie
  - ▶ `get_id()` – pobieranie identyfikatora
  - ▶ `join()` – „przyłączenie” skońzonego wątku
- ▶ ... `std::this_thread::get_id() :-)`

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# „Hello thread”

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void action()
2 {
3     cout << "hello_";
4     cout << this_thread::get_id();
5     cout << "_thread" << endl;
6 }
7
8 int main()
9 {
10    thread th(action);
11    // [ ...can do other stuff here... ]
12    th.join();
13 }
```

# „Hello thread”

```
1 void action()
2 {
3     cout << "hello_";
4     cout << this_thread::get_id();
5     cout << "_thread" << endl;
6 }
7
8 int main()
9 {
10    thread th(action);
11    // [ ...can do other stuff here... ]
12    th.join();
13 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# „Hello thread”

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void action()
2 {
3     cout << "hello_";
4     cout << this_thread::get_id();
5     cout << "_thread" << endl;
6 }
7
8 int main()
9 {
10    thread th(action);
11    // [ ...can do other stuff here... ]
12    th.join();
13 }
```

# „Hello thread”

```
1 void action()
2 {
3     cout << "hello_";
4     cout << this_thread::get_id();
5     cout << "_thread" << endl;
6 }
7
8 int main()
9 {
10    thread th(action);
11    // [ ...can do other stuff here... ]
12    th.join();
13 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# „Hello thread”

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void action()
2 {
3     cout << "hello_";
4     cout << this_thread::get_id();
5     cout << "_thread" << endl;
6 }
7
8 int main()
9 {
10    thread th(action);
11    // [ ...can do other stuff here... ]
12    th.join();
13 }
```

# „Hello thread”

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void action()
2 {
3     cout << "hello_";
4     cout << this_thread::get_id();
5     cout << "_thread" << endl;
6 }
7
8 int main()
9 {
10    thread th(action);
11    // [ ...can do other stuff here... ]
12    th.join();
13 }
```

# „Hello thread”

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void action()
2 {
3     cout << "hello_";
4     cout << this_thread::get_id();
5     cout << "_thread" << endl;
6 }
7
8 int main()
9 {
10    thread th(action);
11    // [ ...can do other stuff here... ]
12    th.join();
13 }
```

# „Hello thread”

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void action()
2 {
3     cout << "hello_";
4     cout << this_thread::get_id();
5     cout << "_thread" << endl;
6 }
7
8 int main()
9 {
10    thread th(action);
11    // [ ...can do other stuff here... ]
12    th.join();
13 }
```

```
$ ./a.out
hello 139829549139712 thread
$ █
```

# Przekazywanie parametrów

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void function(int arg) { cout << arg; }
2 auto lambda = [](int arg) { cout << arg; };
3 struct Functor
4 {
5     void operator()(int arg) { cout << arg; }
6 };
7
8 int main()
9 {
10    const auto arg = 42;
11    thread th1(function, arg); // function(arg)
12    thread th2(lambda, arg);   // lambda(arg)
13    Functor obj;
14    thread th3(obj, arg);      // obj(arg)
15    // ... and join them all here :-)
16 }
```

# Przekazywanie parametrów

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void function(int arg) { cout << arg; }
2 auto lambda = [](int arg) { cout << arg; };
3 struct Functor
4 {
5     void operator()(int arg) { cout << arg; }
6 };
7
8 int main()
9 {
10    const auto arg = 42;
11    thread th1(function, arg); // function(arg)
12    thread th2(lambda, arg);   // lambda(arg)
13    Functor obj;
14    thread th3(obj, arg);      // obj(arg)
15    // ... and join them all here :—)
16 }
```

# Przekazywanie parametrów

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void function(int arg) { cout << arg; }
2 auto lambda = [](int arg) { cout << arg; };
3 struct Functor
4 {
5     void operator()(int arg) { cout << arg; }
6 };
7
8 int main()
9 {
10    const auto arg = 42;
11    thread th1(function, arg); // function(arg)
12    thread th2(lambda, arg);   // lambda(arg)
13    Functor obj;
14    thread th3(obj, arg);     // obj(arg)
15    // ... and join them all here :—)
16 }
```

# Przekazywanie parametrów

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void function(int arg) { cout << arg; }
2 auto lambda = [](int arg) { cout << arg; };
3 struct Functor
4 {
5     void operator()(int arg) { cout << arg; }
6 };
7
8 int main()
9 {
10    const auto arg = 42;
11    thread th1(function, arg); // function(arg)
12    thread th2(lambda, arg);   // lambda(arg)
13    Functor obj;
14    thread th3(obj, arg);     // obj(arg)
15    // ... and join them all here :—)
16 }
```

# Przekazywanie parametrów

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void function(int arg) { cout << arg; }
2 auto lambda = [](int arg) { cout << arg; };
3 struct Functor
4 {
5     void operator()(int arg) { cout << arg; }
6 };
7
8 int main()
9 {
10    const auto arg = 42;
11    thread th1(function, arg); // function(arg)
12    thread th2(lambda, arg);   // lambda(arg)
13    Functor obj;
14    thread th3(obj, arg);     // obj(arg)
15    // ... and join them all here :—)
16 }
```

# Przekazywanie parametrów

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void function(int arg) { cout << arg; }
2 auto lambda = [](int arg) { cout << arg; };
3 struct Functor
4 {
5     void operator()(int arg) { cout << arg; }
6 };
7
8 int main()
9 {
10    const auto arg = 42;
11    thread th1(function, arg); // function(arg)
12    thread th2(lambda, arg);   // lambda(arg)
13    Functor obj;
14    thread th3(obj, arg);      // obj(arg)
15    // ... and join them all here :—)
16 }
```

# Przekazywanie parametrów

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void function(int arg) { cout << arg; }
2 auto lambda = [](int arg) { cout << arg; };
3 struct Functor
4 {
5     void operator()(int arg) { cout << arg; }
6 };
7
8 int main()
9 {
10    const auto arg = 42;
11    thread th1(function, arg); // function(arg)
12    thread th2(lambda, arg);   // lambda(arg)
13    Functor obj;
14    thread th3(obj, arg);      // obj(arg)
15    // ... and join them all here :—)
16 }
```

# Przekazywanie parametrów

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 void function(int arg) { cout << arg; }
2 auto lambda = [](int arg) { cout << arg; };
3 struct Functor
4 {
5     void operator()(int arg) { cout << arg; }
6 };
7
8 int main()
9 {
10    const auto arg = 42;
11    thread th1(function, arg); // function(arg)
12    thread th2(lambda, arg);   // lambda(arg)
13    Functor obj;
14    thread th3(obj, arg);      // obj(arg)
15    // ... and join them all here :-)
16 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# A gdyby tak...

```
1 auto print = [](auto n)
2 {
3     cout << "hello_" << n << "_/_";
4     cout << this_thread::get_id();
5     cout << endl;
6 };
7 thread th1(print, 1);
8 thread th2(print, 2);
9 th1.join();
10 th2.join();
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# A gdyby tak...

```
1 auto print = [](auto n)
2 {
3     cout << "hello_" << n << "_/_";
4     cout << this_thread::get_id();
5     cout << endl;
6 };
7 thread th1(print, 1);
8 thread th2(print, 2);
9 th1.join();
10 th2.join();
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# A gdyby tak...

```
1 auto print = [](auto n)
2 {
3     cout << "hello_" << n << "_/_";
4     cout << this_thread::get_id();
5     cout << endl;
6 };
7 thread th1(print, 1);
8 thread th2(print, 2);
9 th1.join();
10 th2.join();
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# A gdyby tak...

```
1 auto print = [](auto n)
2 {
3     cout << "hello_" << n << "_/_";
4     cout << this_thread::get_id();
5     cout << endl;
6 };
7 thread th1(print, 1);
8 thread th2(print, 2);
9 th1.join();
10 th2.join();
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# A gdyby tak...

```
1 auto print = [](auto n)
2 {
3     cout << "hello_" << n << "_/_";
4     cout << this_thread::get_id();
5     cout << endl;
6 };
7 thread th1(print, 1);
8 thread th2(print, 2);
9 th1.join();
10 th2.join();
```

## Pytanie

Co pojawi się na ekranie?

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wyniki konkursu

- ▶ Przykładowe wyniki:
  - ▶ *hello 1 / 140039617115904*
  - ▶ *hello 2 / 140039608723200*

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wyniki konkursu

## ▶ Przykładowe wyniki:

- ▶ *hello 1 / 140039617115904*
- ▶ *hello 2 / 140039608723200*
- ▶ *hello 2 / 140039615687506*
- ▶ *hello 1 / 130029638763246*

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wyniki konkursu

## ▶ Przykładowe wyniki:

- ▶ *hello 1 / 140039617115904*
- ▶ *hello 2 / 140039608723200*
- ▶ *hello 2 / 140039615687506*
- ▶ *hello 1 / 130029638763246*
- ▶ *hello hello 2 / 1 / 139726453303040139726461695744*

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wyniki konkursu

## ▶ Przykładowe wyniki:

- ▶ *hello 1 / 140039617115904*
- ▶ *hello 2 / 140039608723200*
- ▶ *hello 2 / 140039615687506*
- ▶ *hello 1 / 130029638763246*
- ▶ *hello hello 2 / 1 / 139726453303040139726461695744*
- ▶ *hello hello 21 // 139798176425728139798184818432*

# Wyniki konkursu

## ▶ Przykładowe wyniki:

- ▶ *hello 1 / 140039617115904*
- ▶ *hello 2 / 140039608723200*
- ▶ *hello 2 / 140039615687506*
- ▶ *hello 1 / 130029638763246*
- ▶ *hello hello 2 / 1 / 139726453303040139726461695744*
- ▶ *hello hello 21 // 139798176425728139798184818432*
- ▶ *hello hello 12 // 140534621857536140534630250240*

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wyniki konkursu

- ▶ Przykładowe wyniki:
  - ▶ *hello 1 / 140039617115904*
  - ▶ *hello 2 / 140039608723200*
  - ▶ *hello 2 / 140039615687506*
  - ▶ *hello 1 / 130029638763246*
  - ▶ *hello hello 2 / 1 / 139726453303040139726461695744*
  - ▶ *hello hello 21 // 139798176425728139798184818432*
  - ▶ *hello hello 12 // 140534621857536140534630250240*
  - ▶ ...
- ▶ itp...
- ▶ itd...
- ▶ No cóż...;-)

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Nadspodziewanie dziwne



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# „Przeplot”...

```
1 cout << "hello\u20ac";  
2 // nop...  
3 cout << 1;  
4 cout << "\u20ac/\u20ac";  
5 // nop...  
6 cout << this_thread::get_id();  
7 cout << endl;
```

hello hello

```
1 cout << "hello\u20ac";  
2 cout << 2;  
3 cout << "\u20ac/\u20ac";  
4 // nop...  
5 cout << this_thread::get_id();  
6 cout << endl;  
7 // nop...
```

# „Przeplot”...

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 cout << "hello_\\";  
2 // nop...  
3 cout << 1;  
4 cout << "\_/\_\";  
5 // nop...  
6 cout << this_thread::get_id();  
7 cout << endl;
```

```
1 cout << "hello_\\";  
2 cout << 2;  
3 cout << "\_/\_\";  
4 // nop...  
5 cout << this_thread::get_id();  
6 cout << endl;  
7 // nop...
```

hello hello 2

# „Przeplot”...

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 cout << "hello_\\";  
2 // nop...  
3 cout << 1;  
4 cout << "\_/\_\";  
5 // nop...  
6 cout << this_thread::get_id();  
7 cout << endl;
```

*hello hello 21 /*

```
1 cout << "hello_\\";  
2 cout << 2;  
3 cout << "\_/\_\";  
4 // nop...  
5 cout << this_thread::get_id();  
6 cout << endl;  
7 // nop...
```

# „Przeplot”...

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 cout << "hello_\\";  
2 // nop...  
3 cout << 1;  
4 cout << "\_/\_\";  
5 // nop...  
6 cout << this_thread::get_id();  
7 cout << endl;
```

*hello hello 21 //*

```
1 cout << "hello_\\";  
2 cout << 2;  
3 cout << "\_/\_\";  
4 // nop...  
5 cout << this_thread::get_id();  
6 cout << endl;  
7 // nop...
```

# „Przeplot”...

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 cout << "hello_\\";  
2 // nop...  
3 cout << 1;  
4 cout << "\_/\_\\";  
5 // nop...  
6 cout << this_thread::get_id();  
7 cout << endl;
```

hello hello 21 // 1272383474580

```
1 cout << "hello_\\";  
2 cout << 2;  
3 cout << "\_/\_\\";  
4 // nop...  
5 cout << this_thread::get_id();  
6 cout << endl;  
7 // nop...
```

# „Przeplot”...

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 cout << "hello_\\";  
2 // nop...  
3 cout << 1;  
4 cout << "\_/\_\\";  
5 // nop...  
6 cout << this_thread::get_id();  
7 cout << endl;
```

```
1 cout << "hello_\\";  
2 cout << 2;  
3 cout << "\_/\_\\";  
4 // nop...  
5 cout << this_thread::get_id();  
6 cout << endl;  
7 // nop...
```

hello hello 21 // 12723834745802382386453484

# „Przeplot”...

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 cout << "hello_\\";  
2 // nop...  
3 cout << 1;  
4 cout << "\_/\_\";  
5 // nop...  
6 cout << this_thread::get_id();  
7 cout << endl;
```

```
1 cout << "hello_\\";  
2 cout << 2;  
3 cout << "\_/\_\";  
4 // nop...  
5 cout << this_thread::get_id();  
6 cout << endl;  
7 // nop...
```

hello hello 21 // 12723834745802382386453484

# Pora na...

Dlaczego?

Garść pojęć

Startowanie wątków

**Podstawowe współdzielenie danych**

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Blokada

- ▶ Q: Jak unikać "przeplotu"?

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Blokada

- ▶ Q: Jak unikać "przeplotu"?
- ▶ A: Tylko jeden wątek może wypisywać!

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Blokada

- ▶ Q: Jak unikać "przeplotu"?
- ▶ A: Tylko jeden wątek może wypisywać!
- ▶ Mutex (ang. MUTual EXclusion) jest klasą :)
- ▶ `#include <mutex>`
- ▶ `std::mutex`

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Blokada

- ▶ Q: Jak unikać "przeplotu"?
- ▶ A: Tylko jeden wątek może wypisywać!
- ▶ Mutex (ang. MUTual EXclusion) jest klasą :)
- ▶ `#include <mutex>`
- ▶ `std::mutex`
- ▶ Najważniejsze metody
  - ▶ `lock()` – zakładanie blokady
  - ▶ `unlock()` – ściąganie blokady

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Blokada

- ▶ Q: Jak unikać "przeplotu"?
- ▶ A: Tylko jeden wątek może wypisywać!
- ▶ Mutex (ang. MUTual EXclusion) jest klasą :)
- ▶ `#include <mutex>`
- ▶ `std::mutex`
- ▶ Najważniejsze metody
  - ▶ `lock()` – zakładanie blokady
  - ▶ `unlock()` – ściąganie blokady
- ▶ Przykład (sekcja krytyczna):

```
1 std::mutex m;  
2 // ...  
3 {  
4     m.lock();  
5     // only one thread at a time here!  
6     m.unlock();  
7 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Blokada

- ▶ Q: Jak unikać "przeplotu"?
- ▶ A: Tylko jeden wątek może wypisywać!
- ▶ Mutex (ang. MUTual EXclusion) jest klasą :)
- ▶ `#include <mutex>`
- ▶ `std::mutex`
- ▶ Najważniejsze metody
  - ▶ `lock()` – zakładanie blokady
  - ▶ `unlock()` – ściąganie blokady
- ▶ Przykład (sekcja krytyczna):

```
1 std ::mutex m;  
2 // ...  
3 {  
4     m.lock();  
5     // only one thread at a time here!  
6     m.unlock();  
7 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Sprytniejsze blokowanie

- ▶ Jest do tego klasa :)
- ▶ `std::unique_lock`
- ▶ Najważniejsze metody
  - ▶ konstruktor – zakładanie blokady
  - ▶ destruktor – ściąganie blokady

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Sprytniejsze blokowanie

- ▶ Jest do tego klasa :)
- ▶ `std::unique_lock`
- ▶ Najważniejsze metody
  - ▶ konstruktor – zakładanie blokady
  - ▶ destruktor – ściąganie blokady
- ▶ Przykład (sekcja krytyczna):

```
1  mutex m;  
2  // ...  
3  {  
4      unique_lock<mutex> lock(m);  
5      // only one thread at a time here!  
6  } // !!! :-D
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Sprytniejsze blokowanie

- ▶ Jest do tego klasa :)
- ▶ `std::unique_lock`
- ▶ Najważniejsze metody
  - ▶ konstruktor – zakładanie blokady
  - ▶ destruktor – ściąganie blokady
- ▶ Przykład (sekcja krytyczna):

```
1 mutex m;  
2 // ...  
3 {  
4     unique_lock<mutex> lock(m);  
5     // only one thread at a time here!  
6 } // !!! :-D
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Sprytniejsze blokowanie

- ▶ Jest do tego klasa :)
- ▶ `std::unique_lock`
- ▶ Najważniejsze metody
  - ▶ konstruktor – zakładanie blokady
  - ▶ destruktor – ściąganie blokady
- ▶ Przykład (sekcja krytyczna):

```
1 mutex m;  
2 // ...  
3 {  
4     unique_lock<mutex> lock(m);  
5     // only one thread at a time here!  
6 } // !!! :-D
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Sprytniejsze blokowanie

- ▶ Jest do tego klasa :)
- ▶ `std::unique_lock`
- ▶ Najważniejsze metody
  - ▶ konstruktor – zakładanie blokady
  - ▶ destruktor – ściąganie blokady
- ▶ Przykład (sekcja krytyczna):

```
1 mutex m;  
2 // ...  
3 {  
4     unique_lock<mutex> lock(m);  
5     // only one thread at a time here!  
6 } // !!! :-D
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Sprytniejsze blokowanie

- ▶ Jest do tego klasa :)
- ▶ `std::unique_lock`
- ▶ Najważniejsze metody
  - ▶ konstruktor – zakładanie blokady
  - ▶ destruktor – ściąganie blokady
- ▶ Przykład (sekcja krytyczna):

```
1 mutex m;  
2 // ...  
3 {  
4     unique_lock<mutex> lock(m);  
5     // only one thread at a time here!  
6 } // !!! :-D
```

- ▶ `using Lock = unique_lock<mutex>`

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Powrót do przeszłości

- ▶ Jak naprawić „przeplot”?

```
1 auto print = [](auto n)
2 {
3     cout << "hello_" << n << "_/_";
4     cout << this_thread::get_id();
5     cout << endl;
6 };
7 thread th1(print, 1);
8 thread th2(print, 2);
9 th1.join();
10 th2.join();
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Powrót do przeszłości

- ▶ Jak naprawić „przeplot”?

```
1  mutex m;
2  auto print = [&](auto n)
3  {
4      unique_lock<mutex> lock(m); // da-dam!
5      cout << "hello_" << n << "_/_";
6      cout << this_thread::get_id();
7      cout << endl;
8  };
9  thread th1(print, 1);
10 thread th2(print, 2);
11 th1.join();
12 th2.join();
```

- ▶ Voila!

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Powrót do przeszłości

- ▶ Jak naprawić „przeplot”?

```
1 mutex m;
2 auto print = [&](auto n)
3 {
4     unique_lock<mutex> lock(m); // da-dam!
5     cout << "hello_" << n << "_/_";
6     cout << this_thread::get_id();
7     cout << endl;
8 };
9 thread th1(print, 1);
10 thread th2(print, 2);
11 th1.join();
12 th2.join();
```

- ▶ Voila!

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Powrót do przeszłości

- ▶ Jak naprawić „przeplot”?

```
1  mutex m;
2  auto print = [&](auto n)
3  {
4      unique_lock<mutex> lock(m); // da-dam!
5      cout << "hello_" << n << "_/_";
6      cout << this_thread::get_id();
7      cout << endl;
8  };
9  thread th1(print, 1);
10 thread th2(print, 2);
11 th1.join();
12 th2.join();
```

- ▶ Voila!

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Powrót do przeszłości

- ▶ Jak naprawić „przeplot”?

```
1 mutex m;
2 auto print = [&](auto n)
3 {
4     unique_lock<mutex> lock(m); // da-dam!
5     cout << "hello_" << n << "_/_";
6     cout << this_thread::get_id();
7     cout << endl;
8 };
9 thread th1(print, 1);
10 thread th2(print, 2);
11 th1.join();
12 th2.join();
```

- ▶ Voila!

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Efekt?

- ▶ Możliwe wyniki:

- ▶ *hello 1 / 140039617115904*
  - hello 2 / 140039608723200*

# Efekt?

## ► Możliwe wyniki:

- ▶ *hello 1 / 140039617115904*
- ▶ *hello 2 / 140039608723200*
- ▶ *hello 2 / 140039615687506*
- ▶ *hello 1 / 130029638763246*

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Efekt?

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

► Możliwe wyniki:

- ▶ *hello 1 / 140039617115904*
- ▶ *hello 2 / 140039608723200*
- ▶ *hello 2 / 140039615687506*
- ▶ *hello 1 / 130029638763246*

► Ale **NIE**:

- ▶ *hello hello 2 / 1 / 139726453303040139726461695744*
- ▶ *hello hello 21 // 139798176425728139798184818432*
- ▶ *hello hello 12 // 140534621857536140534630250240*
- ▶ ...
- ▶ itp...

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1 {  
2     unique_lock<mutex> lock(m);  
3     cout << "hello_" << 1 << "_" / "_";  
4     cout << this_thread::get_id();  
5     cout << endl;  
6 } // unlock  
7 // nop...  
8 // nop...  
9 // nop...  
10 // nop...  
11 // nop...  
  
1 {  
2     unique_lock<mutex> lock(m);  
3     // nop...  
4     // nop...  
5     // nop...  
6     // nop...  
7     // can lock now (finally!)  
8     cout << "hello_" << 2 << "_" / "_";  
9     cout << this_thread::get_id();  
10    cout << endl;  
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1 {  
2     unique_lock<mutex> lock(m);  
3     cout << "hello_" << 1 << "_" / "_";  
4     cout << this_thread::get_id();  
5     cout << endl;  
6 } // unlock  
7 // nop...  
8 // nop...  
9 // nop...  
10 // nop...  
11 // nop...  
  
1 {  
2     unique_lock<mutex> lock(m);  
3     // nop...  
4     // nop...  
5     // nop...  
6     // nop...  
7     // can lock now (finally!)  
8     cout << "hello_" << 2 << "_" / "_";  
9     cout << this_thread::get_id();  
10    cout << endl;  
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1 {  
2     unique_lock<mutex> lock(m);  
3     cout << "hello_" << 1 << "_/_";  
4     cout << this_thread::get_id();  
5     cout << endl;  
6 } // unlock  
7 // nop...  
8 // nop...  
9 // nop...  
10 // nop...  
11 // nop...  
  
1 {  
2     unique_lock<mutex> lock(m);  
3     // nop...  
4     // nop...  
5     // nop...  
6     // nop...  
7     // can lock now (finally!)  
8     cout << "hello_" << 2 << "_/_";  
9     cout << this_thread::get_id();  
10    cout << endl;  
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1 {  
2     unique_lock<mutex> lock(m);  
3     cout << "hello_" << 1 << "_" / "_";  
4     cout << this_thread::get_id();  
5     cout << endl;  
6 } // unlock  
7 // nop...  
8 // nop...  
9 // nop...  
10 // nop...  
11 // nop...  
  
1 {  
2     unique_lock<mutex> lock(m);  
3     // nop...  
4     // nop...  
5     // nop...  
6     // can lock now (finally!)  
7     cout << "hello_" << 2 << "_" / "_";  
8     cout << this_thread::get_id();  
9     cout << endl;  
10 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1 {  
2     unique_lock<mutex> lock(m);  
3     cout << "hello_" << 1 << "_" / "_";  
4     cout << this_thread::get_id();  
5     cout << endl;  
6 } // unlock  
7 // nop...  
8 // nop...  
9 // nop...  
10 // nop...  
11 // nop...  
  
1 {  
2     unique_lock<mutex> lock(m);  
3     // nop...  
4     // nop...  
5     // nop...  
6     // can lock now (finally!)  
7     cout << "hello_" << 2 << "_" / "_";  
8     cout << this_thread::get_id();  
9     cout << endl;  
10 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1 {  
2     unique_lock<mutex> lock(m);  
3     cout << "hello_" << 1 << "_" / "_";  
4     cout << this_thread::get_id();  
5     cout << endl;  
6 } // unlock  
7 // nop...  
8 // nop...  
9 // nop...  
10 // nop...  
11 // nop...  
  
1 {  
2     unique_lock<mutex> lock(m);  
3     // nop...  
4     // nop...  
5     // nop...  
6     // nop...  
7     // can lock now (finally!)  
8     cout << "hello_" << 2 << "_" / "_";  
9     cout << this_thread::get_id();  
10    cout << endl;  
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1 {  
2     unique_lock<mutex> lock(m);  
3     cout << "hello_" << 1 << "_" / "_";  
4     cout << this_thread::get_id();  
5     cout << endl;  
6 } // unlock  
7 // nop...  
8 // nop...  
9 // nop...  
10 // nop...  
11 // nop...  
  
1 {  
2     unique_lock<mutex> lock(m);  
3     // nop...  
4     // nop...  
5     // nop...  
6     // nop...  
7     // can lock now (finally!)  
8     cout << "hello_" << 2 << "_" / "_";  
9     cout << this_thread::get_id();  
10    cout << endl;  
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1  {  
2      unique_lock<mutex> lock(m);  
3      cout << "hello_" << 1 << "_" / "_";  
4      cout << this_thread::get_id();  
5      cout << endl;  
6  } // unlock  
7 // nop...  
8 // nop...  
9 // nop...  
10 // nop...  
11 // nop...
```

```
1  {  
2      unique_lock<mutex> lock(m);  
3      // nop...  
4      // nop...  
5      // nop...  
6      // nop...  
7      // can lock now (finally!)  
8      cout << "hello_" << 2 << "_" / "_";  
9      cout << this_thread::get_id();  
10     cout << endl;  
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1 {  
2     unique_lock<mutex> lock(m);  
3     cout << "hello_" << 1 << "_" / "_";  
4     cout << this_thread::get_id();  
5     cout << endl;  
6 } // unlock  
7 // nop...  
8 // nop...  
9 // nop...  
10 // nop...  
11 // nop...  
  
1 {  
2     unique_lock<mutex> lock(m);  
3     // nop...  
4     // nop...  
5     // nop...  
6     // nop...  
7     // can lock now (finally!)  
8     cout << "hello_" << 2 << "_" / "_";  
9     cout << this_thread::get_id();  
10    cout << endl;  
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1  {  
2      unique_lock<mutex> lock(m);  
3      cout << "hello_" << 1 << "_" / "_";  
4      cout << this_thread::get_id();  
5      cout << endl;  
6  } // unlock  
7 // nop...  
8 // nop...  
9 // nop...  
10 // nop...  
11 // nop...
```

```
1  {  
2      unique_lock<mutex> lock(m);  
3      // nop...  
4      // nop...  
5      // nop...  
6      // nop...  
7      // can lock now (finally!)  
8      cout << "hello_" << 2 << "_" / "_";  
9      cout << this_thread::get_id();  
10     cout << endl;  
11  } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria i praktyka...

```
1  {  
2      unique_lock<mutex> lock(m);  
3      cout << "hello_" << 1 << "_" / "_";  
4      cout << this_thread::get_id();  
5      cout << endl;  
6  } // unlock  
7  // nop...  
8  // nop...  
9  // nop...  
10 // nop...  
11 // nop...
```

```
1  {  
2      unique_lock<mutex> lock(m);  
3      // nop...  
4      // nop...  
5      // nop...  
6      // nop...  
7      // can lock now (finally!)  
8      cout << "hello_" << 2 << "_" / "_";  
9      cout << this_thread::get_id();  
10     cout << endl;  
11 } // unlock
```

# Brain Oops...

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



# Jak to naprawić?

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

I CAN FIX IT



TRUST ME I'M AN ENGINEER

# Formatowanie poza blokadą!

BaSz

```
1 mutex m;
2 auto print = [&](auto n)
3 {
4     stringstream ss;
5     ss << "hello_" << n << "_/_";
6     ss << this_thread::get_id();
7     ss << endl;
8     unique_lock<mutex> lock(m);
9     cout << ss.rdbuf();
10    };
11 thread th1(print, 1);
12 thread th2(print, 2);
13 th1.join();
14 th2.join();
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Formatowanie poza blokadą!

BaSz

```
1 mutex m;
2 auto print = [&](auto n)
3 {
4     stringstream ss;
5     ss << "hello_" << n << "_/_";
6     ss << this_thread::get_id();
7     ss << endl;
8     unique_lock<mutex> lock(m);
9     cout << ss.rdbuf();
10    };
11 thread th1(print, 1);
12 thread th2(print, 2);
13 th1.join();
14 th2.join();
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Formatowanie poza blokadą!

```
1 mutex m;
2 auto print = [&](auto n)
3 {
4     stringstream ss;
5     ss << "hello_" << n << "_/_";
6     ss << this_thread::get_id();
7     ss << endl;
8     unique_lock<mutex> lock(m);
9     cout << ss.rdbuf();
10    };
11 thread th1(print, 1);
12 thread th2(print, 2);
13 th1.join();
14 th2.join();
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Formatowanie poza blokadą!

```
1 mutex m;
2 auto print = [&](auto n)
3 {
4     stringstream ss;
5     ss << "hello_" << n << "_/_";
6     ss << this_thread::get_id();
7     ss << endl;
8     unique_lock<mutex> lock(m);
9     cout << ss.rdbuf();
10    };
11 thread th1(print, 1);
12 thread th2(print, 2);
13 th1.join();
14 th2.join();
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Formatowanie poza blokadą!

BaSz

```
1 mutex m;
2 auto print = [&](auto n)
3 {
4     stringstream ss;
5     ss << "hello_" << n << "_/_";
6     ss << this_thread::get_id();
7     ss << endl;
8     unique_lock<mutex> lock(m);
9     cout << ss.rdbuf();
10    };
11 thread th1(print, 1);
12 thread th2(print, 2);
13 th1.join();
14 th2.join();
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8  } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_/_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8  } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_/_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8  } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8  } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8  } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8  } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_"/_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8  } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_"/_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8 } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8  } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8  } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Jak teraz?

```
1  stringstream ss;
2  ss << "hello_" << 1 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      cout << ss.rdbuf();
8  } // unlock
9 // nop...
10 // nop...
11 // nop...
```

```
1  stringstream ss;
2  ss << "hello_" << 2 << "_" / "_";
3  ss << this_thread::get_id();
4  ss << endl;
5  {
6      unique_lock<mutex> lock(m);
7      // nop...
8      // nop...
9      // can lock now
10     cout << ss.rdbuf();
11 } // unlock
```

HA!

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



**VICTORY  
IS MINE**

# Wnioski

1. Blokować – gdzie trzeba
2. Unikać blokad – gdzie się da
3. Zmienne globalne == problemy...
  - ▶ Nietrywialne zależności
  - ▶ Utrudnia analizę kodu
  - ▶ Dzielenie wymaga blokad

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wnioski

1. Blokować – gdzie trzeba
2. Unikać blokad – gdzie się da
3. Zmienne globalne == problemy...
  - ▶ Nietrywialne zależności
  - ▶ Utrudnia analizę kodu
  - ▶ Dzielenie wymaga blokad
4. Kiedy blokować?
5. ...

# Model pamięci C++

Programowanie  
Współbieżne  
w C++ 4fun&&\$

- ▶ Około 4 stron! :)

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Model pamięci C++

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

- ▶ Około 4 stron! :)
- ▶ Blokować trzeba gdy:
  - ▶ Jednocześnie:
    1. Wiele wątków używa danej zmiennej
    2. Co najmniej jeden z nich zapisuje

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Model pamięci C++

Programowanie  
Współbieżne  
w C++ 4fun&& \$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

- ▶ Około 4 stron! :)
- ▶ Blokować trzeba gdy:
  - ▶ Jednocześnie:
    1. Wiele wątków używa danej zmiennej
    2. Co najmniej jeden z nich zapisuje
  - ▶ ... albo niepodzielny ciąg operacji
- ▶ That's it! :D

# Przykłady

- ▶ Monitoring temperatury (wielodostęp):
  - ▶ Czujnik aktualizuje odczyt
  - ▶ UI wyświetla

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Przykłady

Programowanie  
Współbieżne  
w C++ 4fun&& \$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

- ▶ Monitoring temperatury (wielodostęp):
  - ▶ Czujnik aktualizuje odczyt
  - ▶ UI wyświetla
- ▶ Przelew bankowy (operacja niepodzielna):
  - ▶ Pobranie pieniędzy z jednego konta
  - ▶ Przekazanie na drugie konto
- ▶ itd...

# Pora na...

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Ach to I/O...



BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

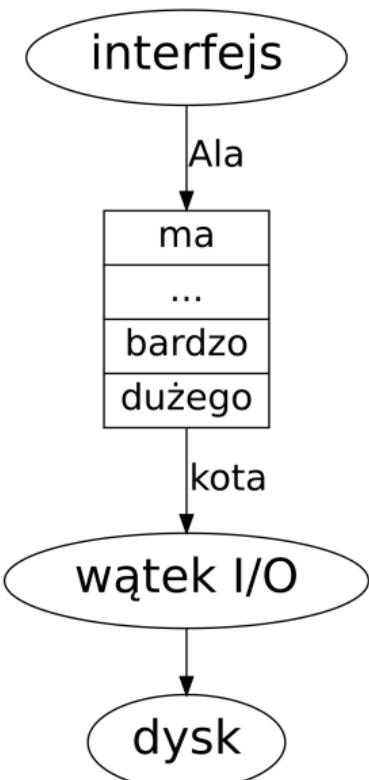
Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wątek I/O

## Wątek I/O

- ▶ Dysk:
  - ▶ I/O często wolne
  - ▶ Potrafi zablokować (zapis/odczyt)



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

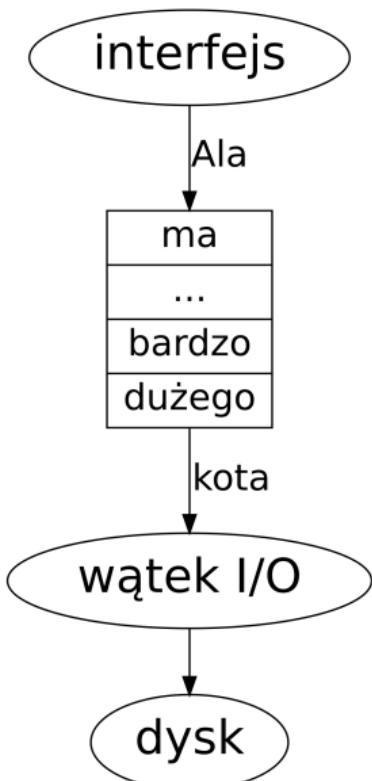
Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

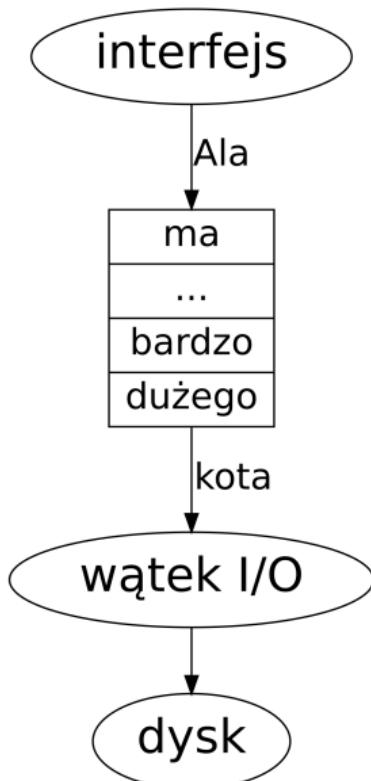
# Wątek I/O

- ▶ Dysk:
  - ▶ I/O często wolne
  - ▶ Potrafi zablokować (zapis/odczyt)
- ▶ Interfejs:
  - ▶ Zapis podanych słów
  - ▶ Koniecznie responsywny



# Wątek I/O

- ▶ Dysk:
  - ▶ I/O często wolne
  - ▶ Potrafi zablokować (zapis/odczyt)
- ▶ Interfejs:
  - ▶ Zapis podanych słów
  - ▶ Koniecznie responsywny
- ▶ Kolejka:
  - ▶ Bufor danych do zapisania
  - ▶ Dodawanie/zdejmowanie jest szybkie



Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Prosta kolejka FIFO

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 class SimpleQueue final
2 {
3     public:
4         void push(string str);
5         string pop();
6         bool empty() const;
7
8     private:
9         using Lock = unique_lock<mutex>;
10        mutable mutex m_;
11        deque<string> data_;
12    };
```

# Prosta kolejka FIFO

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 class SimpleQueue final
2 {
3     public:
4         void push(string str);
5         string pop();
6         bool empty() const;
7
8     private:
9         using Lock = unique_lock<mutex>;
10        mutable mutex m_;
11        deque<string> data_;
12 }
```

# Prosta kolejka FIFO

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 class SimpleQueue final
2 {
3     public:
4         void push(string str);
5         string pop();
6         bool empty() const;
7
8     private:
9         using Lock = unique_lock<mutex>;
10        mutable mutex m_;
11        deque<string> data_;
12 }
```

# Prosta kolejka FIFO – empty

```
1 #include "SimpleQueue.hpp"  
2  
3 bool SimpleQueue::empty() const  
4 {  
5     Lock lock(m_);  
6     return data_.empty();  
7 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Prosta kolejka FIFO – pop

```
1 #include "SimpleQueue.hpp"  
2  
3 string SimpleQueue::pop()  
4 {  
5     Lock lock(m_);  
6     auto tmp = std::move( data_.back() );  
7     data_.pop_back();  
8     return tmp;  
9 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Prosta kolejka FIFO – push

```
1 #include "SimpleQueue.hpp"  
2  
3 void SimpleQueue::push(string str)  
4 {  
5     Lock lock(m_);  
6     data_.push_front( std::move(str) );  
7 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wysyłanie danych z UI

```
1 #include "SimpleQueue.hpp"  
2  
3 void interface(SimpleQueue& q)  
4 {  
5     string txt;  
6     while( getline(cin, txt) )  
7         q.push(txt);  
8 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wysyłanie danych z UI

```
1 #include "SimpleQueue.hpp"  
2  
3 void interface(SimpleQueue& q)  
4 {  
5     string txt;  
6     while( getline(cin, txt) )  
7         q.push(txt);  
8 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wysyłanie danych z UI

```
1 #include "SimpleQueue.hpp"  
2  
3 void interface(SimpleQueue& q)  
4 {  
5     string txt;  
6     while( getline(cin, txt) )  
7         q.push(txt);  
8 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wysyłanie danych z UI

```
1 #include "SimpleQueue.hpp"  
2  
3 void interface(SimpleQueue& q)  
4 {  
5     string txt;  
6     while( getline(cin, txt) )  
7         q.push(txt);  
8 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wysyłanie danych z UI

```
1 #include "SimpleQueue.hpp"  
2  
3 void interface(SimpleQueue& q)  
4 {  
5     string txt;  
6     while( getline(cin, txt) )  
7         q.push(txt);  
8 }
```

# Wysyłanie danych z UI

```
1 #include "SimpleQueue.hpp"  
2  
3 void interface(SimpleQueue& q)  
4 {  
5     string txt;  
6     while( getline(cin, txt) )  
7         q.push(txt);  
8 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wysyłanie danych z UI

```
1 #include "SimpleQueue.hpp"  
2  
3 void interface(SimpleQueue& q)  
4 {  
5     string txt;  
6     while( getline(cin, txt) )  
7         q.push(txt);  
8 }
```

# Zapisywanie danych na dysk

```
1 #include "SimpleQueue.hpp"  
2  
3 void io(SimpleQueue& q)  
4 {  
5     ofstream file("/tmp/sth.txt");  
6     while(file)  
7     {  
8         while( q.empty() ) { /* nop */ }  
9         file << q.pop() << endl;  
10    }  
11 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zapisywanie danych na dysk

```
1 #include "SimpleQueue.hpp"  
2  
3 void io(SimpleQueue& q)  
4 {  
5     ofstream file("/tmp/sth.txt");  
6     while(file)  
7     {  
8         while( q.empty() ) { /* nop */ }  
9         file << q.pop() << endl;  
10    }  
11 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zapisywanie danych na dysk

```
1 #include "SimpleQueue.hpp"  
2  
3 void io(SimpleQueue& q)  
4 {  
5     ofstream file("/tmp/sth.txt");  
6     while(file)  
7     {  
8         while( q.empty() ) { /* nop */ }  
9         file << q.pop() << endl;  
10    }  
11 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zapisywanie danych na dysk

```
1 #include "SimpleQueue.hpp"  
2  
3 void io(SimpleQueue& q)  
4 {  
5     ofstream file("/tmp/sth.txt");  
6     while(file)  
7     {  
8         while( q.empty() ) { /* nop */ }  
9         file << q.pop() << endl;  
10    }  
11 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zapisywanie danych na dysk

```
1 #include "SimpleQueue.hpp"  
2  
3 void io(SimpleQueue& q)  
4 {  
5     ofstream file("/tmp/sth.txt");  
6     while(file)  
7     {  
8         while( q.empty() ) { /* nop */ }  
9         file << q.pop() << endl;  
10    }  
11 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zapisywanie danych na dysk

```
1 #include "SimpleQueue.hpp"
2
3 void io(SimpleQueue& q)
4 {
5     ofstream file("/tmp/sth.txt");
6     while(file)
7     {
8         while( q.empty() ) { /* nop */ }
9         file << q.pop() << endl;
10    }
11 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zapisywanie danych na dysk

```
1 #include "SimpleQueue.hpp"  
2  
3 void io(SimpleQueue& q)  
4 {  
5     ofstream file("/tmp/sth.txt");  
6     while(file)  
7     {  
8         while( q.empty() ) { /* nop */ }  
9         file << q.pop() << endl;  
10    }  
11 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zapisywanie danych na dysk

```
1 #include "SimpleQueue.hpp"  
2  
3 void io(SimpleQueue& q)  
4 {  
5     ofstream file("/tmp/sth.txt");  
6     while(file)  
7     {  
8         while( q.empty() ) { /* nop */ }  
9         file << q.pop() << endl;  
10    }  
11 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zapisywanie danych na dysk

```
1 #include "SimpleQueue.hpp"  
2  
3 void io(SimpleQueue& q)  
4 {  
5     ofstream file("/tmp/sth.txt");  
6     while(file)  
7     {  
8         while( q.empty() ) { /* nop */ }  
9         file << q.pop() << endl;  
10    }  
11 }
```

## Pytanie

Co z tym kodem jest nie tak?

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zapisywanie danych na dysk

```
1 #include "SimpleQueue.hpp"  
2  
3 void io(SimpleQueue& q)  
4 {  
5     ofstream file("/tmp/sth.txt");  
6     while(file)  
7     {  
8         while( q.empty() ) { /* nop */ }  
9         file << q.pop() << endl;  
10    }  
11 }
```

## Pytanie

Co z tym kodem jest nie tak?

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Grzanie CPU

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



# Uwaga na efekt cieplarniany!

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienna warunku

- ▶ ang. *Condition Variable*
- ▶ Jest do tego klasa :)
- ▶ `#include <condition_variable>`
- ▶ `std::condition_variable`

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienna warunku

- ▶ ang. *Condition Variable*
- ▶ Jest do tego klasa :)
- ▶ `#include <condition_variable>`
- ▶ `std::condition_variable`
- ▶ Najważniejsze metody:
  - ▶ `wait` – oczekuje na zmianę
  - ▶ `notify_all` – wybudza czekających na zmianę

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienna warunku

- ▶ ang. *Condition Variable*
- ▶ Jest do tego klasa :)
- ▶ `#include <condition_variable>`
- ▶ `std::condition_variable`
- ▶ Najważniejsze metody:
  - ▶ `wait` – oczekuje na zmianę
  - ▶ `notify_all` – wybudza czekających na zmianę
- ▶ Jak używać?
- ▶ Z blokadą!
- ▶ ...

# Kolejka 2.0

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 class WaitQueue final
2 {
3     public:
4         void push(string str);
5         string pop();
6
7     private:
8         using Lock = unique_lock<mutex>;
9         condition_variable nonEmpty_;
10        mutable mutex m_;
11        deque<string> data_;
12    };
```

# Kolejka 2.0

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 class WaitQueue final
2 {
3     public:
4         void push(string str);
5         string pop();
6
7     private:
8         using Lock = unique_lock<mutex>;
9         condition_variable nonEmpty_;
10        mutable mutex m_;
11        deque<string> data_;
12 }
```

# Kolejka 2.0

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 class WaitQueue final
2 {
3     public:
4         void push(string str);
5         string pop();
6
7     private:
8         using Lock = unique_lock<mutex>;
9         condition_variable nonEmpty_;
10        mutable mutex m_;
11        deque<string> data_;
12 }
```

# Kolejka 2.0

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 class WaitQueue final
2 {
3     public:
4         void push(string str);
5         string pop();
6
7     private:
8         using Lock = unique_lock<mutex>;
9         condition_variable nonEmpty_;
10        mutable mutex m_;
11        deque<string> data_;
12    };
```

# Kolejka 2.0 – pop

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 #include "WaitQueue.hpp"
2
3 string WaitQueue::pop()
4 {
5     Lock lock(m_);
6     auto hasData = [&]{ return !data_.empty(); };
7     nonEmpty_.wait(lock, hasData);
8     auto tmp = std::move( data_.back() );
9     data_.pop_back();
10    return tmp;
11 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Kolejka 2.0 – pop

```
1 #include "WaitQueue.hpp"
2
3 string WaitQueue::pop()
4 {
5     Lock lock(m_);
6     auto hasData = [&]{ return !data_.empty(); };
7     nonEmpty_.wait(lock, hasData);
8     auto tmp = std::move(data_.back());
9     data_.pop_back();
10    return tmp;
11 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Kolejka 2.0 – pop

```
1 #include "WaitQueue.hpp"  
2  
3 string WaitQueue::pop()  
4 {  
5     Lock lock(m_);  
6     auto hasData = [&]{ return not data_.empty(); };  
7     nonEmpty_.wait(lock, hasData);  
8     auto tmp = std::move( data_.back() );  
9     data_.pop_back();  
10    return tmp;  
11 }
```

# Kolejka 2.0 – pop

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 #include "WaitQueue.hpp"
2
3 string WaitQueue::pop()
4 {
5     Lock lock(m_);
6     auto hasData = [&]{ return !data_.empty(); };
7     nonEmpty_.wait(lock, hasData);
8     auto tmp = std::move( data_.back() );
9     data_.pop_back();
10    return tmp;
11 }
```

# Kolejka 2.0 – push

```
1 #include "WaitQueue.hpp"  
2  
3 void WaitQueue::push(string str)  
4 {  
5     Lock lock(m_);  
6     data_.push_front( std::move(str) );  
7     nonEmpty_.notify_all();  
8 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Kolejka 2.0 – push

```
1 #include "WaitQueue.hpp"
2
3 void WaitQueue::push(string str)
4 {
5     Lock lock(m_);
6     data_.push_front( std::move(str) );
7     nonEmpty_.notify_all();
8 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Kolejka 2.0 – push

```
1 #include "WaitQueue.hpp"
2
3 void WaitQueue::push(string str)
4 {
5     Lock lock(m_);
6     data_.push_front( std::move(str) );
7     nonEmpty_.notify_all();
8 }
```

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Wysyłanie danych z UI – nowa kolejka

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 #include "WaitQueue.hpp"  
2  
3 void interface(WaitQueue& q)  
4 {  
5     string txt;  
6     while( getline(cin, txt) )  
7         q.push(txt);  
8 }
```

# Zapisywanie danych na dysk – nowa kolejka

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 #include "WaitQueue.hpp"
2
3 void io(WaitQueue& q)
4 {
5     ofstream file("/tmp/sth.txt");
6     while(file)
7         file << q.pop() << endl;
8 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zapisywanie danych na dysk – nowa kolejka

```
1 #include "WaitQueue.hpp"
2
3 void io(WaitQueue& q)
4 {
5     ofstream file("/tmp/sth.txt");
6     while(file)
7         file << q.pop() << endl;
8 }
```

# Zapisywanie danych na dysk – nowa kolejka

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

```
1 #include "WaitQueue.hpp"
2
3 void io(WaitQueue& q)
4 {
5     ofstream file("/tmp/sth.txt");
6     while(file)
7         file << q.pop() << endl;
8 }
```

# Gdy oszczędzasz CPU...

Programowanie  
Współbieżne  
w C++ 4fun&& \$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



# Pora na...

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# power-up time! ;-)

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

charging... ;-)

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# 15 minut przerwy

# Pora na...

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zatrzymanie

## Problem

Jak zakończyć wątek w pętli?

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zatrzymanie

## Problem

Jak zakończyć wątek w pętli?

```
1 void someThread()
2 {
3     while( not shouldStop() )
4     {
5         /* do sth... */
6     }
7 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zatrzymanie

## Problem

Jak zakończyć wątek w pętli?

```
1 void someThread()
2 {
3     while( not shouldStop() )
4     {
5         /* do sth ... */
6     }
7 }
```

# Nasz problem

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zwykła zmienna?

```
1 bool stop = false; // set externally
2
3 void someThread()
{
    while(not stop)
    {
        /* do sth... */
    }
}
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zwykła zmienna?

```
1 bool stop = false; // set externally
2
3 void someThread()
4 {
5     while(not stop)
6     {
7         /* do sth... */
8     }
9 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zwykła zmienna?

```
1 bool stop = false; // set externally
2
3 void someThread()
4 {
5     while(not stop)
6     {
7         /* do sth... */
8     }
9 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zwykła zmienna?

```
1 bool stop = false; // set externally
2
3 void someThread()
{
    while(not stop)
    {
        /* do sth... */
    }
}
```

- ▶ Problemy:
  - ▶ Zapis wykonywany w kilku krokach
  - ▶ Optymalizacje kompilatora
  - ▶ Optymalizacje w procesorze (*cache*)
- ▶ Oops...

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



# EPIC FAIL.

You're doing it wrong!

# Zmienna volatile?

```
1 volatile bool stop = false; // set externally
2
3 void someThread()
4 {
5     while(not stop)
6     {
7         /* do sth... */
8     }
9 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zmienna volatile?

```
1 volatile bool stop = false; // set externally
2
3 void someThread()
4 {
5     while(not stop)
6     {
7         /* do sth... */
8     }
9 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienna volatile?

```
1 volatile bool stop = false; // set externally
2
3 void someThread()
4 {
5     while(not stop)
6     {
7         /* do sth... */
8     }
9 }
```

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Zmienna volatile?

```
1 volatile bool stop = false; // set externally
2
3 void someThread()
{
    while(not stop)
    {
        /* do sth... */
    }
}
```

- ▶ Problemy:
  - ▶ Zapis wykonywany w kilku krokach
  - ▶ *Optymalizacje kompilatora*
  - ▶ Optymalizacje w procesorze (cache)
- ▶ Oops...

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Iluzja poprawności...

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



# FAIL

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienna z blokadą?

```
1 mutex mut;
2 bool flag = false; // set externally
3
4 void someThread()
5 {
6     auto stop = [&]{
7         unique_lock<mutex> l(mut);
8         return flag;
9     };
10    while( not stop() )
11    {
12        /* do sth... */
13    }
14 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienna z blokadą?

```
1 mutex mut;
2 bool flag = false; // set externally
3
4 void someThread()
5 {
6     auto stop = [&]{
7         unique_lock<mutex> l(mut);
8         return flag;
9     };
10    while( not stop() )
11    {
12        /* do sth... */
13    }
14 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienna z blokadą?

```
1 mutex mut;
2 bool flag = false; // set externally
3
4 void someThread()
5 {
6     auto stop = [&]{
7             unique_lock<mutex> l(mut);
8             return flag;
9         };
10    while( not stop() )
11    {
12        /* do sth... */
13    }
14 }
```

# Zmienna z blokadą?

```
1 mutex mut;
2     bool flag = false; // set externally
3
4 void someThread()
5 {
6     auto stop = [&]{
7             unique_lock<mutex> l(mut);
8             return flag;
9         };
10    while( not stop() )
11    {
12        /* do sth... */
13    }
14 }
```

- ▶ Poprawne... ale wolne i...

# O'Rly... ??

Programowanie  
Współbieżne  
w C++ 4fun&\\$&

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



# SPIDER-PIG

# Zmienne atomowe

- ▶ ang. *atomic variable*
- ▶ „Lekka synchronizacja”

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienne atomowe

- ▶ ang. *atomic variable*
- ▶ „Lekka synchronizacja”
- ▶ Jest klasa :)
- ▶ `#include <atomic>`
- ▶ `std::atomic`

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienne atomowe

- ▶ ang. *atomic variable*
- ▶ „Lekka synchronizacja”
- ▶ Jest klasa :)
- ▶ `#include <atomic>`
- ▶ `std::atomic`
- ▶ Pozwala na:
  - ▶ Zapis
  - ▶ Odczyt
  - ▶ Arytmetykę
  - ▶ ...
- ▶ Typowo: liczby, wskaźniki

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienna atomowa!

```
1 atomic<bool> stop{false}; // set externally
2
3 void someThread()
4 {
5     while(not stop)
6     {
7         /* do sth... */
8     }
9 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienna atomowa!

```
1 atomic<bool> stop{false}; // set externally
2
3 void someThread()
4 {
5     while(not stop)
6     {
7         /* do sth... */
8     }
9 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Zmienna atomowa!

```
1 atomic<bool> stop{false}; // set externally
2
3 void someThread()
4 {
5     while(not stop)
6     {
7         /* do sth... */
8     }
9 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

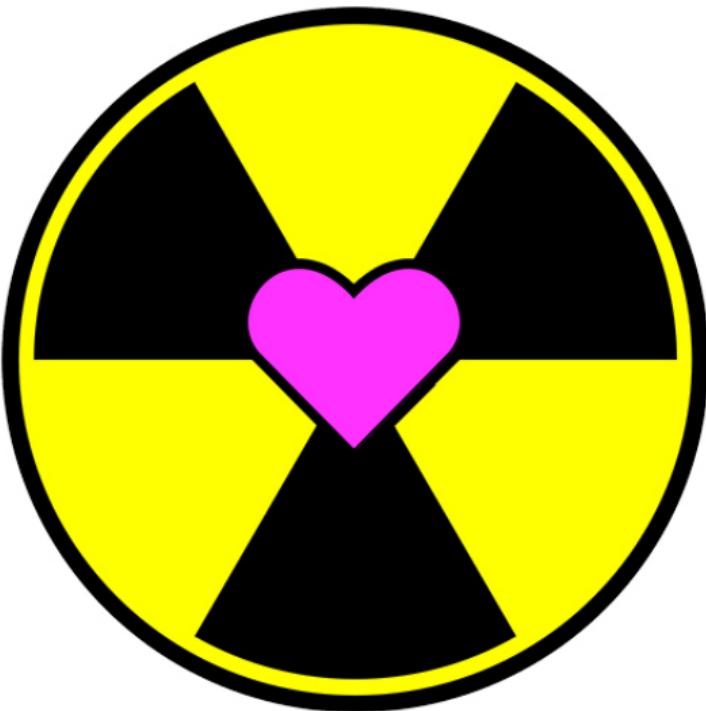
# Zmienna atomowa!

```
1 atomic<bool> stop{false}; // set externally
2
3 void someThread()
4 {
5     while(not stop)
6     {
7         /* do sth... */
8     }
9 }
```

# We love atomic! :-D

Programowanie  
Współbieżne  
w C++ 4fun&& \$

BaSz



[http://www.gbfans.com/shop/images/nuclear\\_love\\_button\\_522655.jpg](http://www.gbfans.com/shop/images/nuclear_love_button_522655.jpg)

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

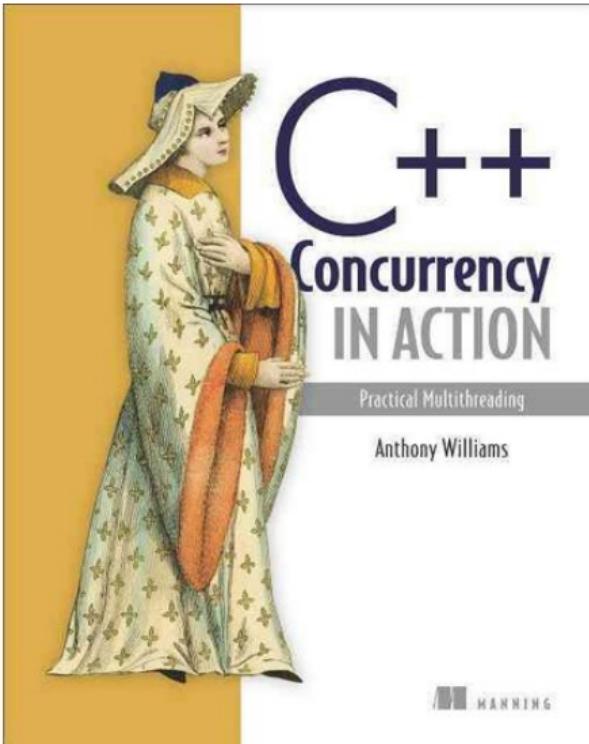
Co przyszłość  
niesie?

Zamiast  
zakończenia

# Lock-free programming – czarny szlak 1/2

Programowanie  
Współbieżne  
w C++ 4fun&&§

BaSz



[http://i.walmartimages.com/i/p/97/81/93/39/88/9781933988771\\_500X500.jpg](http://i.walmartimages.com/i/p/97/81/93/39/88/9781933988771_500X500.jpg)

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Lock-free programming – czarny szlak 2/2

Programowanie  
Współbieżne  
w C++ 4fun&\\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Going Fully “Lock-Free”:  
Atomic Mail Slots

LOCK-FREE PROGRAMMING (OR, JUGGLING RAZOR BLADES), PART I

Herb Sutter

# Pora na...

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

**Wzorce projektowe**

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

**Wzorce  
projektowe**

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Teoria

## Wzorce projektowe

W informatyce wzorzec projektowy jest ogólnym rozwiązaniem typowego problemu, pozwalającym na jego ponowne użycie. Nie jest końcowym rozwiązaniem, lecz szablonem jak rozwiązać problem, mogący wystąpić w wielu różnych sytuacjach. Wzorce projektowe są najlepszymi praktykami, jakich można użyć, do rozwiązania danego problemu.

[https://en.wikipedia.org/wiki/Software\\_design\\_pattern](https://en.wikipedia.org/wiki/Software_design_pattern)

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

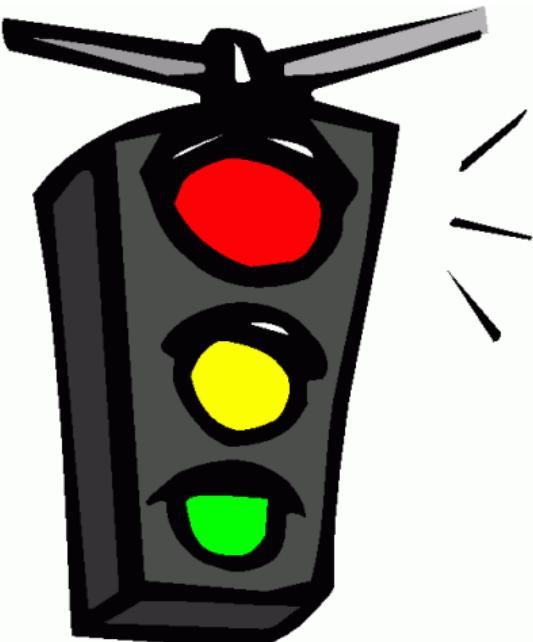
Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



<http://4.bp.blogspot.com/-H5wL0sG74JI/Tbb4g2LMvLI/AAAAAAA9I/ipdpoPLI9t0/s1600/traffic.signal.gif>

# Flaga/zdarzenie – zasada działania

- ▶ Ang. *event flag*
- ▶ Oczekiwanie na zdarzenie
- ▶ Operacje:
  - ▶ `set()` – ustaw
  - ▶ `clear()` – wyczyść
  - ▶ `wait()` – czekaj na zmianę

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

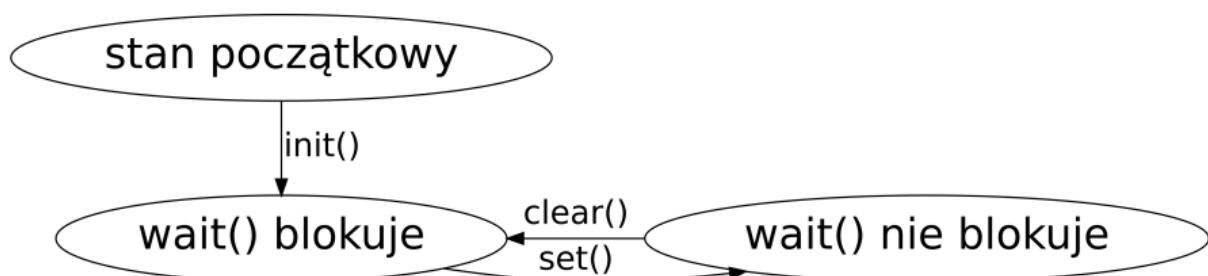
Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Flaga/zdarzenie – zasada działania

- ▶ Ang. *event flag*
- ▶ Oczekiwanie na zdarzenie
- ▶ Operacje:
  - ▶ `set()` – ustaw
  - ▶ `clear()` – wyczyść
  - ▶ `wait()` – czekaj na zmianę



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

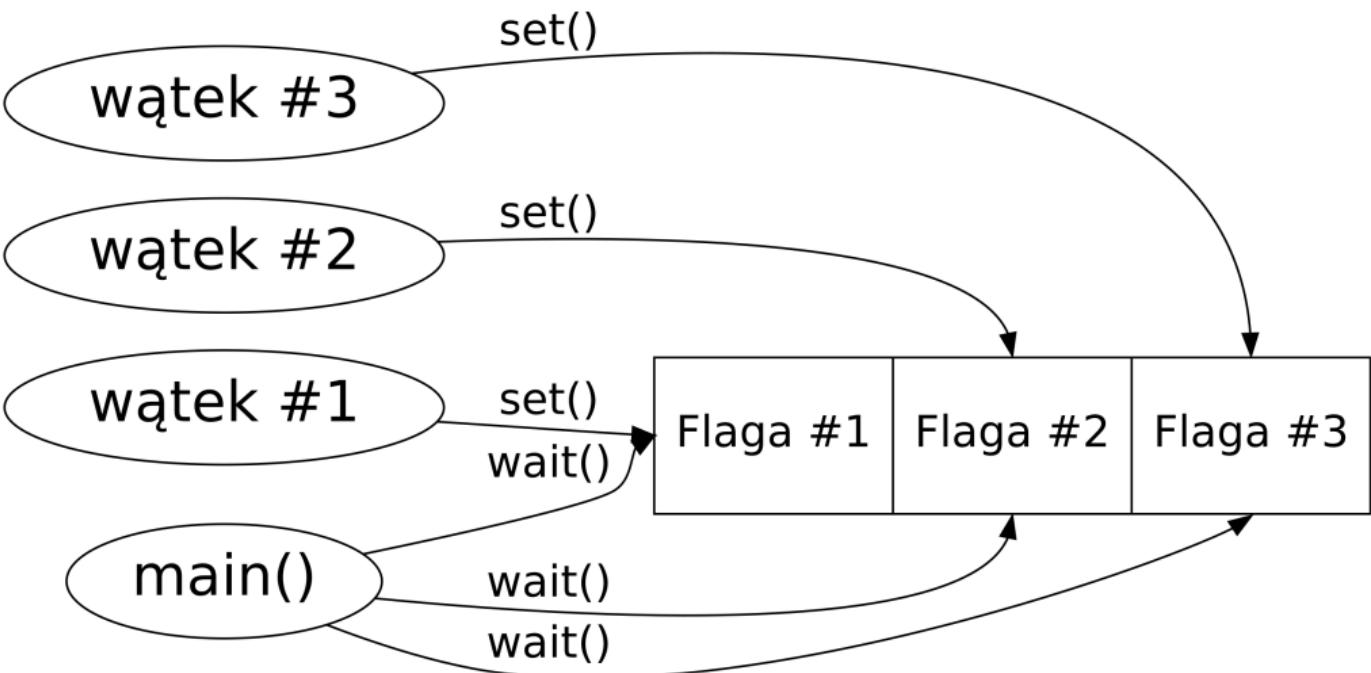
Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Flaga/zdarzenie – oczekiwanie na start





Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

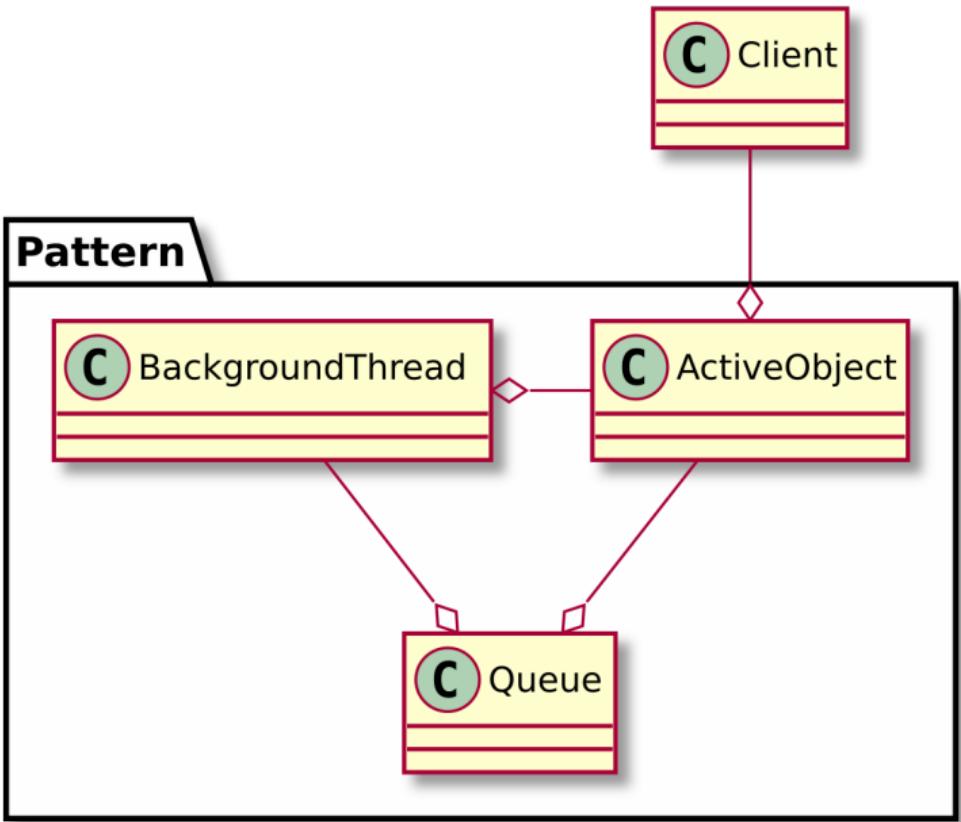
Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

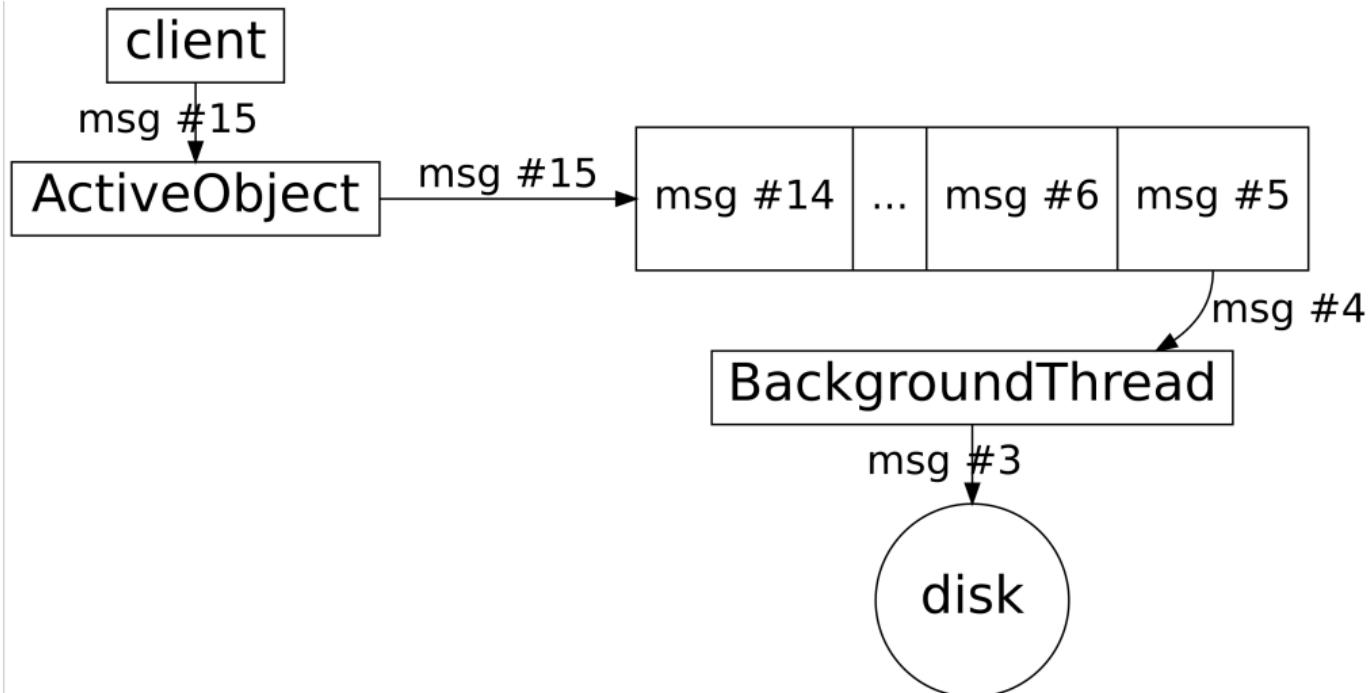
Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Aktywny obiekt – wysyłanie wiadomości



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Future/promise



# Future/promise – budowa

Programowanie  
Współbieżne  
w C++ 4fun&& \$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

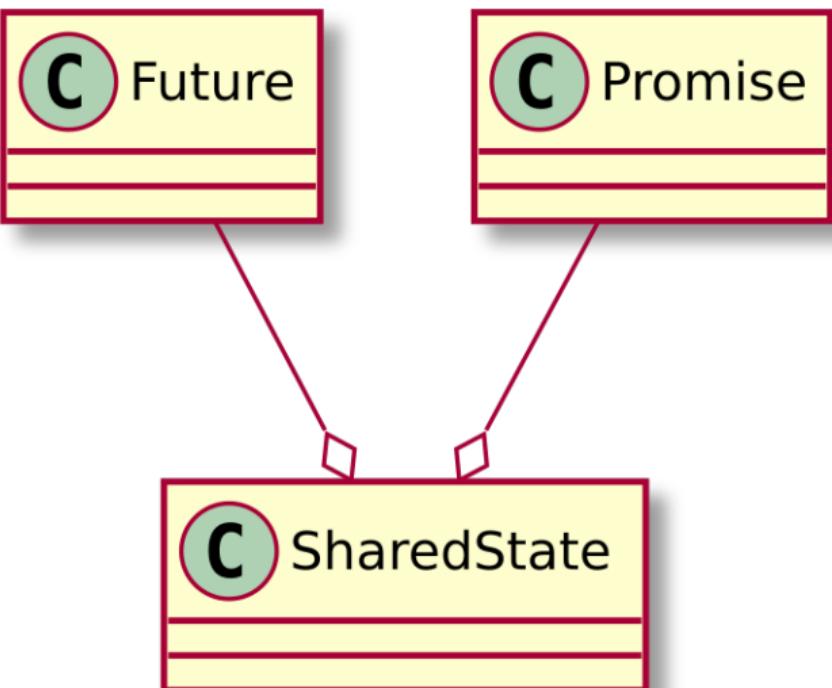
Przerwa

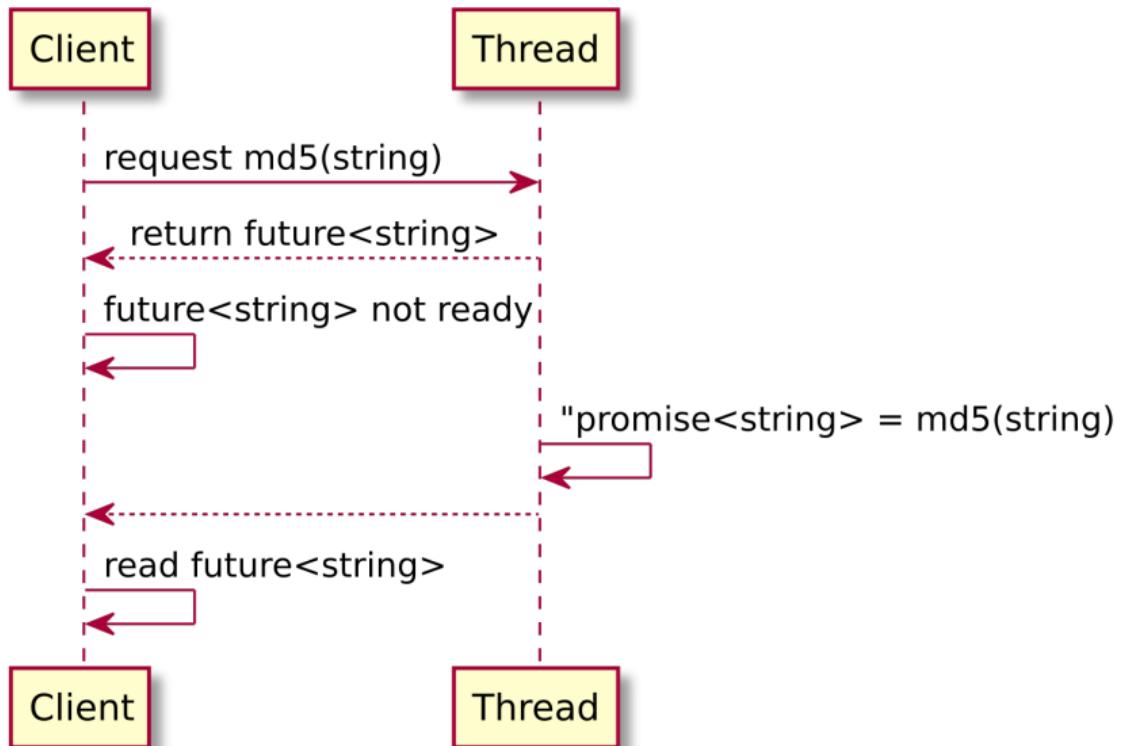
Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia





Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Future/promise – połączenie z...

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

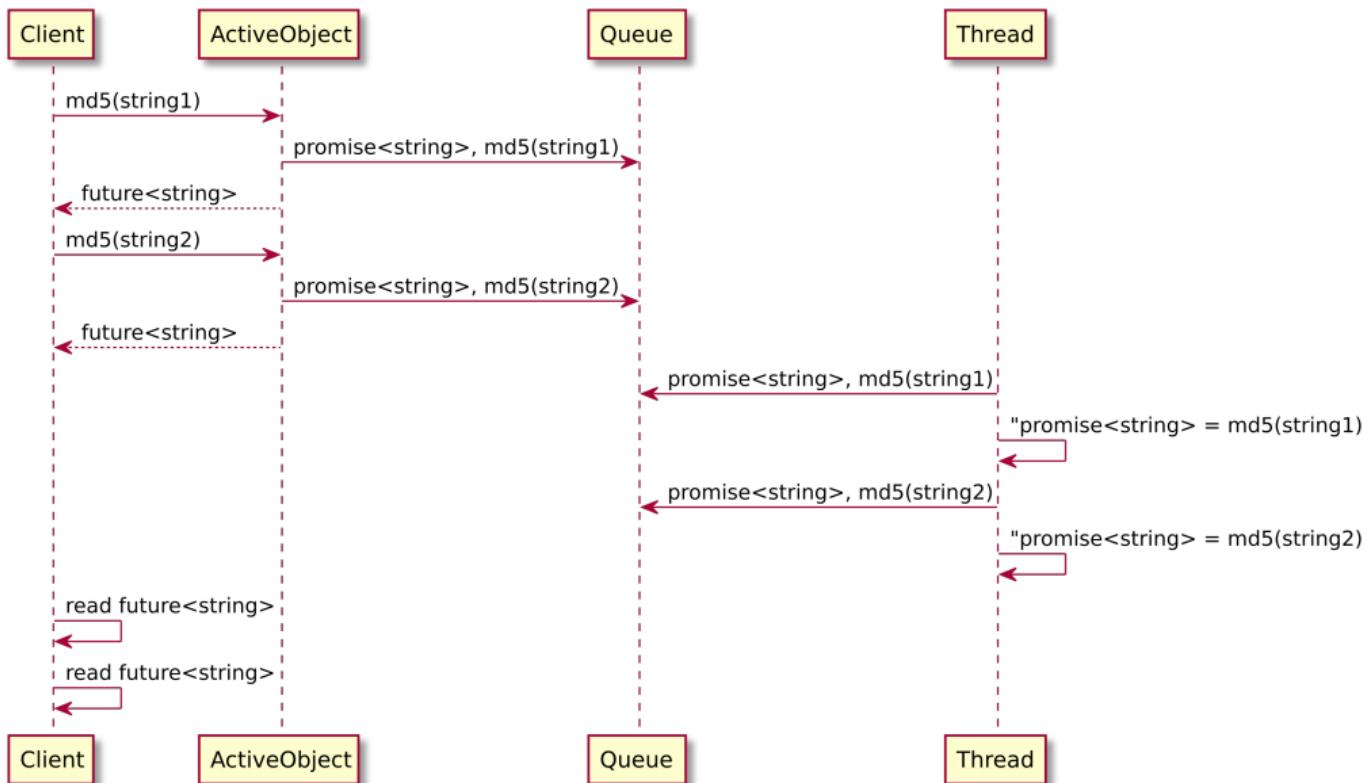
Przerwa

Współdzielenie  
inaczej

**Wzorce  
projektowe**

Co przyszłość  
niesie?

Zamiast  
zakończenia



Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Pula wątków

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

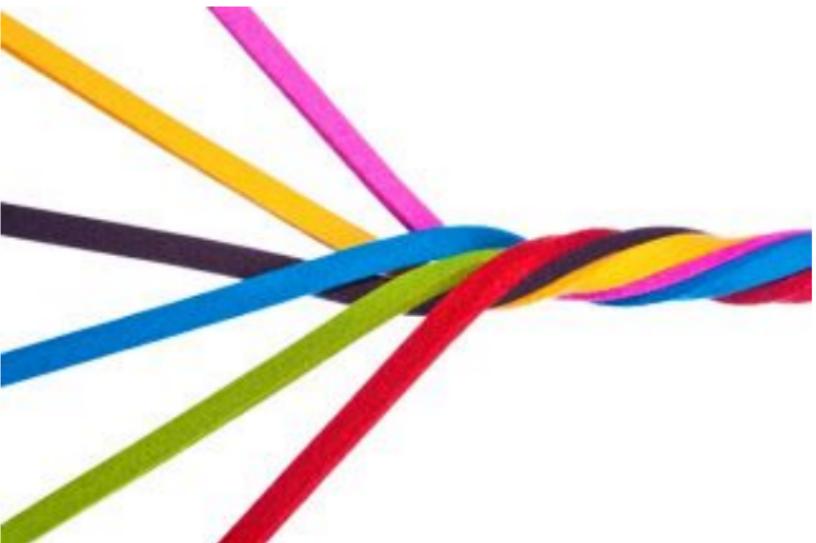
Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



<http://media-cache-ak0.pinimg.com/736x/d0/e1/f3/d0e1f3b0d874e6ec68c78830a203ed90.jpg>

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

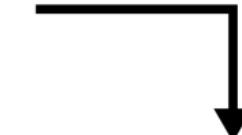
Wzorce  
projektowe

Co przyszłość  
niesie?

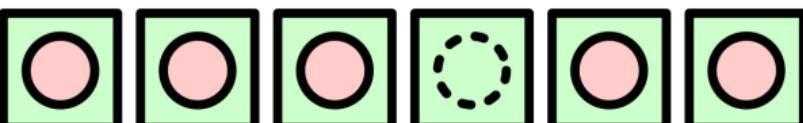
Zamiast  
zakończenia

# Pula wątków – zasada działania

## Task Queue



## Thread Pool



## Completed Tasks



# Pula wątków – obróbka obrazów

Programowanie  
Współbieżne  
w C++ 4fun&\\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

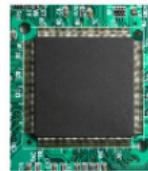
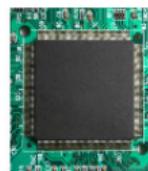
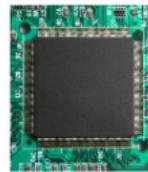
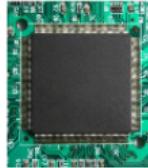
Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



[http://wallpaper.com/images/00/43/76/65/panoramic-view-of-mountains\\_00437665.jpg](http://wallpaper.com/images/00/43/76/65/panoramic-view-of-mountains_00437665.jpg)

<http://images.wisegeek.com/dual-core-processor-mounted-to-a-motherboard.jpg>

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

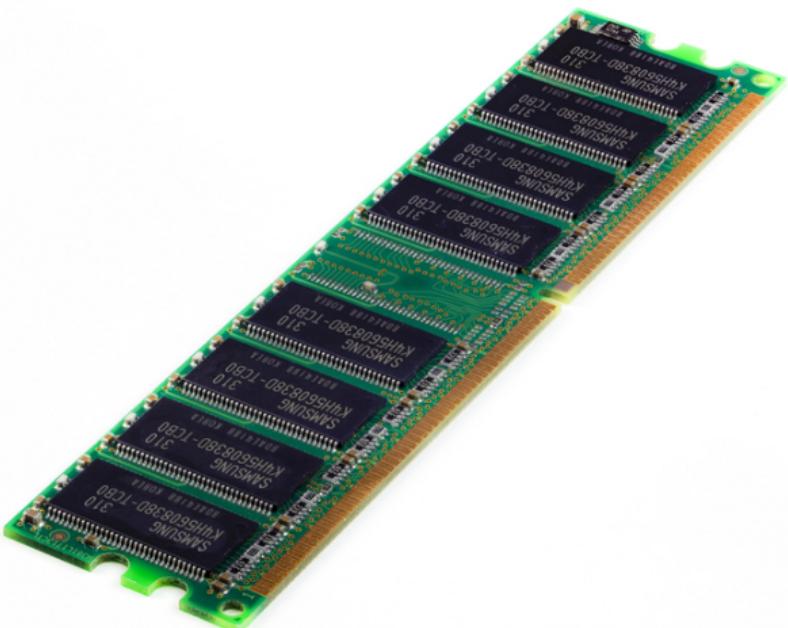
Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



<http://www.vps.net/blog/wp-content/uploads/2014/07/Random-Access-Memory.jpg>

# Thread local storage – zasada działania

- ▶ *TLS*
- ▶ Globalna...
- ▶ ... per wątek
- ▶ Jak to zrobić?

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

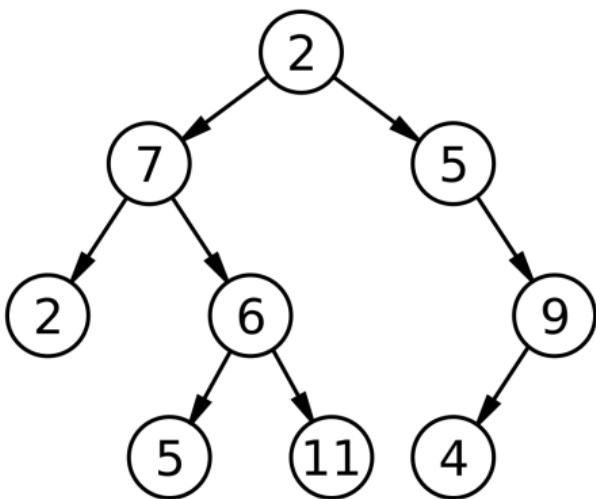
**Wzorce  
projektowe**

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Thread local storage – zasada działania

- ▶ *TLS*
- ▶ Globalna...
- ▶ ... per wątek
- ▶ Jak to zrobić?



Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Thread local storage – get\_id()

- ▶ Mając:

```
1 void threadLogger()
2 {
3     cout << "hello_";
4     cout << this_thread::get_id();
5     cout << "_thread!" << endl;
6 }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Thread local storage – get\_id()

- ▶ Mając:

```
1 void threadLogger()
2 {
3     cout << "hello_";
4     cout << this_thread::get_id();
5     cout << "_thread!" << endl;
6 }
```

- ▶ Dostajemy:

*hello 140039617115904 thread  
hello 140039608723200 thread*

# Thread local storage – identyfikator wątku

- ▶ Mając:

```
1 atomic<uint64_t> freeId{0};  
2 const thread_local auto id = ++freeId;  
3  
4 void threadLogger()  
5 {  
6     cout << "hello_" << id << "_thread!" << endl;  
7 }
```

# Thread local storage – identyfikator wątku

- ▶ Mając:

```
1 atomic<uint64_t> freeId{0};  
2 const thread_local auto id = ++freeId;  
3  
4 void threadLogger()  
5 {  
6     cout << "hello_" << id << "_thread!" << endl;  
7 }
```

# Thread local storage – identyfikator wątku

- ▶ Mając:

```
1 atomic<uint64_t> freeId{0};  
2 const thread_local auto id = ++freeId;  
3  
4 void threadLogger()  
5 {  
6     cout << "hello_" << id << "_thread!" << endl;  
7 }
```

# Thread local storage – identyfikator wątku

- ▶ Mając:

```
1 atomic<uint64_t>          freeId{0};  
2 const thread_local auto id = ++freeId;  
3  
4 void threadLogger()  
5 {  
6     cout << "hello_" << id << "_thread!" << endl;  
7 }
```

# Thread local storage – identyfikator wątku

- ▶ Mając:

```
1 atomic<uint64_t> freeId{0};  
2 const thread_local auto id = ++freeId;  
3  
4 void threadLogger()  
5 {  
6     cout << "hello_" << id << "_thread!" << endl;  
7 }
```

# Thread local storage – identyfikator wątku

- ▶ Mając:

```
1 atomic<uint64_t> freeId{0};  
2 const thread_local auto id = ++freeId;  
3  
4 void threadLogger()  
5 {  
6     cout << "hello_" << id << "_thread!" << endl;  
7 }
```

- ▶ Dostajemy:

*hello 1 thread  
hello 2 thread*

# Inne przykłady

- ▶ *Barrier*
- ▶ *Double-checked locking*
- ▶ *Guarded suspension*
- ▶ *Monitor Object*
- ▶ *Reactor pattern*
- ▶ *Read write lock pattern*
- ▶ *Scheduler pattern*
- ▶ ...



<https://upload.wikimedia.org/wikipedia/en/8/80/Wikipedia-logo-v2.svg>

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

# Pora na...

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# future::then()

```
1 boost::future<string> readFileContent();  
2  
3 boost::future<void> printFile()  
4 {  
5     return readFileContent()  
6         .then(  
7             [](boost::future<string> s)  
8             { cout << s.get(); }  
9         );  
10    }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# future::then()

```
1 boost::future<string> readFileContent();  
2  
3 boost::future<void> printFile()  
4 {  
5     return readFileContent()  
6         .then(  
7             [](boost::future<string> s)  
8                 { cout << s.get(); }  
9             );  
10    }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# future::then()

```
1 boost::future<string> readFileContent();  
2  
3 boost::future<void> printFile()  
4 {  
5     return readFileContent()  
6         .then(  
7             [](boost::future<string> s)  
8                 { cout << s.get(); }  
9             );  
10    }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# future::then()

```
1 boost::future<string> readFileContent();  
2  
3 boost::future<void> printFile()  
4 {  
5     return readFileContent()  
6         .then(  
7             [](boost::future<string> s)  
8                 { cout << s.get(); }  
9             );  
10    }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# future::then()

```
1 boost::future<string> readFileContent();  
2  
3 boost::future<void> printFile()  
4 {  
5     return readFileContent()  
6         .then(  
7             [](boost::future<string> s)  
8                 { cout << s.get(); }  
9             );  
10    }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# future::then()

```
1 boost::future<string> readFileContent();  
2  
3 boost::future<void> printFile()  
4 {  
5     return readFileContent()  
6         .then(  
7             [](boost::future<string> s)  
8             { cout << s.get(); }  
9         );  
10    }
```

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# future::then()

```
1 boost::future<string> readFileContent();  
2  
3 boost::future<void> printFile()  
4 {  
5     return readFileContent()  
6         .then(  
7             [](boost::future<string> s)  
8             { cout << s.get(); }  
9         );  
10    }
```

# Korutyny

```
1 future<string> tcp_reader(unsigned bytes)
2 {
3     std::stringstream ss;
4     auto conn = await TcpConnection("1.2.3.4", 31337);
5     while(bytes)
6     {
7         auto got = await conn.read(ss, bytes);
8         bytes -= got;
9     }
10    return ss.str();
11 }
12
13 int main()
14 {
15     cout << tcp_reader(1024).get();
16 }
```

# Korutyny

```
1 future<string> tcp_reader(unsigned bytes)
2 {
3     std::stringstream ss;
4     auto conn = await TcpConnection("1.2.3.4", 31337);
5     while(bytes)
6     {
7         auto got = await conn.read(ss, bytes);
8         bytes -= got;
9     }
10    return ss.str();
11 }
12
13 int main()
14 {
15     cout << tcp_reader(1024).get();
16 }
```

# Korutyny

```
1 future<string> tcp_reader(unsigned bytes)
2 {
3     std::stringstream ss;
4     auto conn = await TcpConnection("1.2.3.4", 31337);
5     while(bytes)
6     {
7         auto got = await conn.read(ss, bytes);
8         bytes -= got;
9     }
10    return ss.str();
11 }
12
13 int main()
14 {
15     cout << tcp_reader(1024).get();
16 }
```

# Korutyny

```
1 future<string> tcp_reader(unsigned bytes)
2 {
3     std::stringstream ss;
4     auto conn = await TcpConnection("1.2.3.4", 31337);
5     while(bytes)
6     {
7         auto got = await conn.read(ss, bytes);
8         bytes -= got;
9     }
10    return ss.str();
11 }
12
13 int main()
14 {
15     cout << tcp_reader(1024).get();
16 }
```

# Korutyny

```
1 future<string> tcp_reader(unsigned bytes)
2 {
3     std::stringstream ss;
4     auto conn = await TcpConnection("1.2.3.4", 31337);
5     while(bytes)
6     {
7         auto got = await conn.read(ss, bytes);
8         bytes -= got;
9     }
10    return ss.str();
11 }
12
13 int main()
14 {
15     cout << tcp_reader(1024).get();
16 }
```

# JavaScript opanuje świat?

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# JavaScript opanuje świat?

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

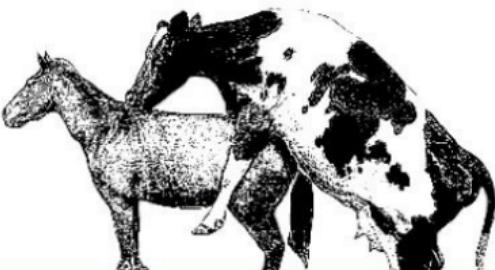
Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

*Good luck with that*

## *Writing* Device Drivers with JavaScript



O'REILLY®

David Flanagan

# Pora na...

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe współdzielenie danych

W oczekiwaniu na zmianę

Przerwa

Współdzielenie inaczej

Wzorce projektowe

Co przyszłość niesie?

Zamiast zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Ciąg dalszy nastąpi...

## ► Zapraszamy na laboratoria!

- ▶ Współbieżność w praktyce
- ▶ Implementacja wybranych wzorców
- ▶ Małe grupy!

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Ciąg dalszy nastąpi...

- ▶ Zapraszamy na laboratoria!
  - ▶ Współbieżność w praktyce
  - ▶ Implementacja wybranych wzorców
  - ▶ Małe grupy!
- ▶ Co przygotować?
  - ▶ Własny laptop
  - ▶ Linux (lub maszyna wirtualna)
  - ▶ Zainstalowane wymagane pakiety
  - ▶ [https://github.com/nokia-wroclaw/wroclaw\\_does\\_it/  
concurrent\\_cpp](https://github.com/nokia-wroclaw/wroclaw_does_it/concurrent_cpp)



# Do końca daleko...

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz

- ▶ Podstawowa znajomość:



- ▶ Konieczność!

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Do końca daleko...

- ▶ Podstawowa znajomość:



- ▶ Konieczność!

- ▶ Zaawansowana znajomość:



- ▶ Elitarna wiedza

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Do końca daleko...

Programowanie  
Współbieżne  
w C++ 4fun&& \$

BaSz

- ▶ Podstawowa znajomość:



- ▶ Zaawansowana znajomość:



- ▶ Konieczność!

- ▶ Elitarna wiedza

- ▶ Wiele rdzeni – codzienność
- ▶ Wysoka wydajność – konieczność!

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Dalsza lektura

- ▶ „Język C++ i przetwarzanie współbieżne w akcji”,  
*Anthony Williams*
- ▶ „The free lunch is over”, *Herb Sutter*
- ▶ „Threads and shared variables in C++11”, *Hans Boehm*
- ▶ „Atomic<> weapons”, *Herb Sutter*
- ▶ „Eliminate false sharing”, *Herb Sutter*
- ▶ „Threading: dos and don'ts”, *Bartek 'BaSz' Szurgot*
- ▶ „Lock-free programming”, *Herb Sutter*
- ▶ „Effective modern C++”, *Scott Meyers*
- ▶ <http://cppreference.com>
- ▶ [https://en.wikipedia.org/wiki/Concurrency\\_pattern](https://en.wikipedia.org/wiki/Concurrency_pattern)



BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Oświadczenie o nieprzekarmianiu psów

Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia



<http://i.imgur.com/1DVtstR.jpg>



<http://canitbesaturdaynow.com/images/fpics/1557/1245499798151.jpg>

# cya@lab!



Programowanie  
Współbieżne  
w C++ 4fun&&\$\$

BaSz

Dlaczego?

Garść pojęć

Startowanie  
wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia

# Materiały na laboratoria

Programowanie  
Współbieżne  
w C++ 4fun&&\$

BaSz



[https://github.com/nokia-wroclaw/  
wroclaw\\_does\\_it](https://github.com/nokia-wroclaw/wroclaw_does_it)

Dlaczego?

Garść pojęć

Startowanie wątków

Podstawowe  
współdzielenie  
danych

W oczekiwaniu na  
zmianę

Przerwa

Współdzielenie  
inaczej

Wzorce  
projektowe

Co przyszłość  
niesie?

Zamiast  
zakończenia