

Sztuka Wytwarzania Oprogramowania

Wykład 15 - Zasoby internetowe, standardy języków programowania

Robert Nowak

24Z

Kolokwium 2

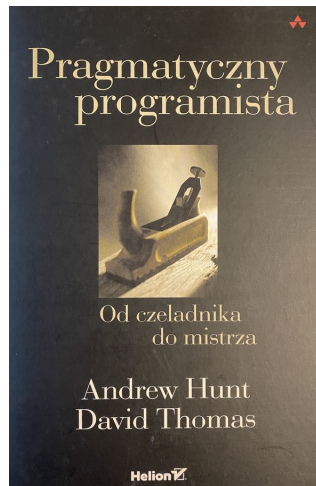
Programowanie to zajęcie praktyczne, więc warto:

- ▶ czytać kod innych,
- ▶ programować.

Warto znać:

- ▶ Edytor Emacs, www.gnu.org
- ▶ Edytor vi (vim),
- ▶ The GNU Project (Free Software Foundation) www.gnu.org,
- ▶ The Object Management Group www.omg.org.

- ▶ Hunt, Thomas. Pragmatyczny programista. Od czeladnika do mistrza.
- ▶ Martin. Czysty kod. Podręcznik dobrego programisty.
- ▶ Martin. Mistrz czystego kodu. Kodeks postępowania profesjonalnych programistów.
- ▶ Gamma et al. Wzorce projektowe.
- ▶ Fowler. Refaktoryzacja. Ulepszanie struktury istniejącego kodu.
- ▶ Brooks. Legendarny osobomiesiąc.



- ▶ IEEE (<http://ieee.org>) - największe światowe stowarzyszenie inżynierów elektryków, elektroników i informatyków, Instytut Inżynierów Elektryków i Elektroników, IEEE (*Institute of Electrical and Electronics Engineers*)
- ▶ ACM (<http://acm.org>) - Association for Computing Machinery - stowarzyszenie informatyków

Wydają ok. 100 czasopism naukowych z informatyki.

Kompilowany do kodu maszynowego, statyczny, abstrakcje wysokiego poziomu np. funkcje wirtualne, wyjątki, tablice asocjacyjne, współbieżność.

- + możliwość tworzenia bardzo wydajnych aplikacji
- + standard języka jest tworzony przez organizację standaryzacyjną ANSI/ISO (nie przez firmę)
 - nieco przestarzały standard (C++11, C++14, C++17, C++20, C++23)
- + wspiera wiele paradygmatów: programowanie obiektowe, generyczne, strukturalne
 - złożony, wiele poprawnych konstrukcji do rozwiązywania tego samego problemu
- + duża popularność, dostępność wielu narzędzi dla wielu platform (w tym wbudowanych)
- + bardzo duża stabilność
 - uboga biblioteka standardowa

C++ jest zgodny wstecz.

- ▶ ISO/IEC 14882, opublikowany w 1998 (C++98)
- ▶ ISO/IEC 14882:2003, zmodyfikowany w 2003 (C++03)
- ▶ ISO/IEC 14882:2011, C++11, c++0x (III 2011), draft:N3290
<http://www.open-std.org/jtc1/sc22/wg21>
- ▶ ISO/IEC 14882:2014, C++14, c++1y (VIII 2014), draft:N3797
- ▶ ISO/IEC 14882:2017, C++17, c++1z (XII 2017), draft:N4661
- ▶ ISO/IEC 14882:2020, C++20, c++2a (XII 2020), draft:N4878
- ▶ ISO/IEC 14882:2024 C++23, c++2b (XII 2023) draft:N4950
`git clone https://github.com/cplusplus/draft.git`
- ▶ C++26, draft:N4981 (2024.04.16)

<http://boost.org>

zbiór bibliotek eksperymentalnych związanych ze standardem C++.

Język Python

Interpretowany, dynamiczny, zwięzły, duży zbiór pak. standardowych.

- + możliwość szybkiego tworzenia różnych aplikacji
- + standard języka jest nadzorowany przez organizację non-profit (Python Software Foundation)
 - 3 wersje standardu (niezgodne), Python 1 (nie używana), Python 2 (wycofana od 2020), Python 3
- + wspiera wiele paradygmatów: funkcyjne, obiektowe, strukturalne
- + prosty, zazwyczaj jedna poprawnych konstrukcja do rozwiązywania danego problemu
- + duża popularność, dostępność wielu narzędzi dla wielu platform (w tym wbudowanych)
- + bogata biblioteka pakietów standardowych
- + bogata biblioteka pakietów dodatkowych, dostępnych w standardowy sposób

Standard Python

Python Enhancement Proposals (PEP) <http://python.org>
rozszerzanie i zmiany w języku i bibliotekach

Przykłady:

- ▶ PEP 0 - definicja PEP, lista PEP (jest ich ponad 8000)
- ▶ PEP 8 - styl kodowania
- ▶ PEP 20 - 'The Zen of Python' (przesłanie)

```
Beautiful is better than ugly.  
Explicit is better than implicit.  
Simple is better than complex.  
Complex is better than complicated.  
Flat is better than nested.  
Sparse is better than dense.  
Readability counts.  
...
```

Konferencja PyCon: <http://pycon.org>

Dokumentacja w kodzie zmniejsza ryzyko braku spójności

Komentarze - poprawiają czytelność kodu.

Komentarz mówi DLACZEGO, kod mówi JAK.

- ▶ każdy byt powinien mieć pojedynczą odpowiedzialność,
- ▶ dokumentacja projektowa powinna być generowana z kodu.

Hierarchia komentarzy:

1. odpowiedzialność bibliotek, pakietów, przestrzeni nazw, katalogów
2. odpowiedzialność modułów (plików), klas,
3. odpowiedzialność metod publicznych,
4. niebanalne algorytmy

Dziękuję

robert.nowak@pw.edu.pl