

XOPERO SOFTWARE

**XOPERO Praktyki 2023-09**

Materiały dla praktykantów

# Wstęp

## Wymagane oprogramowanie

1. Visual Studio  
   <https://visualstudio.microsoft.com/>  
   najprawdopodobniej: <https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Community&rel=16>
2. Git for Windows (można też używać wersji wbudowanej w Visual Studio) <https://git-scm.com/downloads>  
   najprawdopodobniej:  
   <https://git-scm.com/download/win>

## Organizacja praktyk

Czas w którym odbywa się praktyki rejestruje się przez wpis na liście obecności.

Przez pierwsze 3 dni omawiane są zagadnienia teoretyczne (poniżej) - proszę o przygotowanie krótkiej notatki i korzystanie z niej w razie potrzeby.

Po rozpoczęciu części praktycznej należy założyć nowe repozytorium w wybranym serwisie (GitHub, GitLab, BitBucket) i przesłać prowadzącym ([p.pogorzelec@xopero.com](mailto:p.pogorzelec@xopero.com), [p.kowalski@xopero.com](mailto:p.kowalski@xopero.com)) link do repozytorium (w przypadku wybrania konfiguracji z ograniczoną widocznością - należy udostępnić repozytorium prowadzącemu). Praktyczna część praktyk powinna znaleźć się w stosownej części (.gitignore - co to, po co to, użyć) w w/w repozytorium online.

# Pojęcia teoretyczne

Poniższe pojęcia teoretyczne umożliwiają wprowadzenie się w tematykę i poznanie podstawowych informacji. Ich omówienie odbywa się w kolejnych dniach do omówienia wszystkich części - wskazane jest korzystanie z notatek. Dla poszczególnych pojęć należy przygotować krótkie podsumowanie i przedstawić wady/zalety - warto również wspomnieć o alternatywach w przypadku starych technologii.

### Część pierwsza

1. VSS
2. Deduplikacja
3. Kompresja
   1. Rodzaje kompresji
   2. Stopnie kompresji
4. Szyfrowanie (typy)
   1. Szyfrowanie symetryczne
      1. Algorytm AES
      2. Tryby - ECB, CBC
   2. Klucz domyślny i klucz użytkownika w kontekście naszej aplikacji
5. Retencja plików
6. Kopie zapasowe - pełne, przyrostowe i różnicowe
7. Wykonywanie kopii zapasowych na prawach użytkownika
8. Zmienne środowiskowe
9. Harmonogram (systemowy i obecny w naszej aplikacji)
10. Pliki ukryte
11. Filtry plików
12. Usługa systemowa (poziomy/rodzaje uprawnień)
13. Active Directory (podstawowe informacje)
14. Wirtualizacja (rodzaje wirtualizatorów, maszyny wirtualne)
15. Redundancja
16. Chmura

### Część druga

1. NAS
2. QNAP, Synology, Asustor, Netgear
3. Mono
4. Systemy wersjonowania (Git i inne)
5. Docker, Kubernetes
6. Strony wspierające wersjonowanie w chmurze (GitHub i inne)
7. Klastry
8. Centra Danych
9. RAID (rodzaje)
10. Systemy plików (FAT, NTFS, ext - platformy, podstawowe różnice)
11. Rejestr systemowy Windows
12. Foldery systemowe (Windows, Linux)
13. Amazon Web Service (S3, EC2), Azure
14. OpenStack

### Część trzecia

1. Linki symboliczne
2. Specjalne pliki systemowe (pagefile.sys, hiberfil.sys, swapfile.sys)
3. Rejestr zdarzeń Windows i logowanie zdarzeń w kontekście programowania
4. Grupy robocze Windows
5. IPv4, IPv6
6. MSI, EXE
7. Skrypty .BAT, skrypty .sh
8. Ograniczenia systemów w kontekście ograniczeń dotyczących ścieżek do plików
9. Typy plików a rozszerzenia
10. DLL a EXE
11. Procesy a wątki
12. Wielowątkowość
13. Grafika wektorowa a rastrowa
14. Bitlocker
15. HDD a SSD
16. UEFI a BIOS

# Wymagania wstępne do pracy z książką

1. Sprawne środowisko Visual Studio (<https://visualstudio.microsoft.com/downloads/>) albo Rider (<https://www.jetbrains.com/rider/>) albo Visual Code (<https://code.visualstudio.com/download>) i dotnet (<https://docs.microsoft.com/en-us/dotnet/core/install/linux>) (w celach testowych należy skompilować (i uruchomić, jeżeli ma to sens) pusty projekt: aplikacji konsolowej, Windows Forms, WPF, class library).
2. Zainstalowany git bash bądź dowolny inny program do zarządzania repo git (w zależności od wyborów podczas konfiguracji wbudowane w VisualStudio / dostępne jako wtyczka do Visual Code)
3. Skonfigurowane repozytorium git (GitLab/GitHub/BitBucket/…) - po skonfigurowaniu proszę wysłać do mnie adres repo z przykładowym projektem typu Hello world

# Zadania z książki

Po omówieniu teorii należy opracować jak najwięcej zagadnień z książki - postępy należy codziennie zamieszczać na repozytorium Git.

# Skrót informacji o organizacji pracy z Git

Preferowane jest posiadanie jednego repozytorium na wszystkie zadania realizowane podczas praktyk (jako jedne rozwiązanie w Visual Studio).

Branch master powinien zawierać stabilny kod - dzienne zmiany powinny być umieszczane na osobnych branchach które powinny być mergowane do master minimum raz w tygodniu używając interfejsu webowego (tak jak miałoby to miejsce w realnym scenariuszu pracy - gdzie nikt nie ma prawa pushowania zmian do master).

# Materiały dotyczące Git

1. Podstawowe informacje i komendy (bez inicjowania klucza SSH):

<https://rogerdudler.github.io/git-guide/>

2. Generowanie kluczy SSH:

<https://git-scm.com/book/pl/v1/Git-na-serwerze-Generacja-pary-kluczy-SSH>

3. GitLab na temat konfiguracji SSH:

<https://docs.gitlab.com/ee/ssh/>

4. .gitignore

<https://git-scm.com/docs/gitignore>

5. Git LFS

<https://www.atlassian.com/git/tutorials/git-lfs>

<https://git-lfs.github.com/>

# Dodatkowe materiały

1. RFC

<https://www.ietf.org/standards/rfcs/>

2. RFC IPv4

<https://www.rfc-editor.org/info/rfc791>

<https://www.rfc-editor.org/rfc/rfc791.txt>