rule FINANCIAL

The Sector Specialists

Serializacja i protokoły wymiany danych

projektowanie rozproszonych aplikacji w .NET

Prepared by: Janek Polak

Version: 1.0

Confidential

Serializacja i protokoły wymiany danych
Plan zajęć

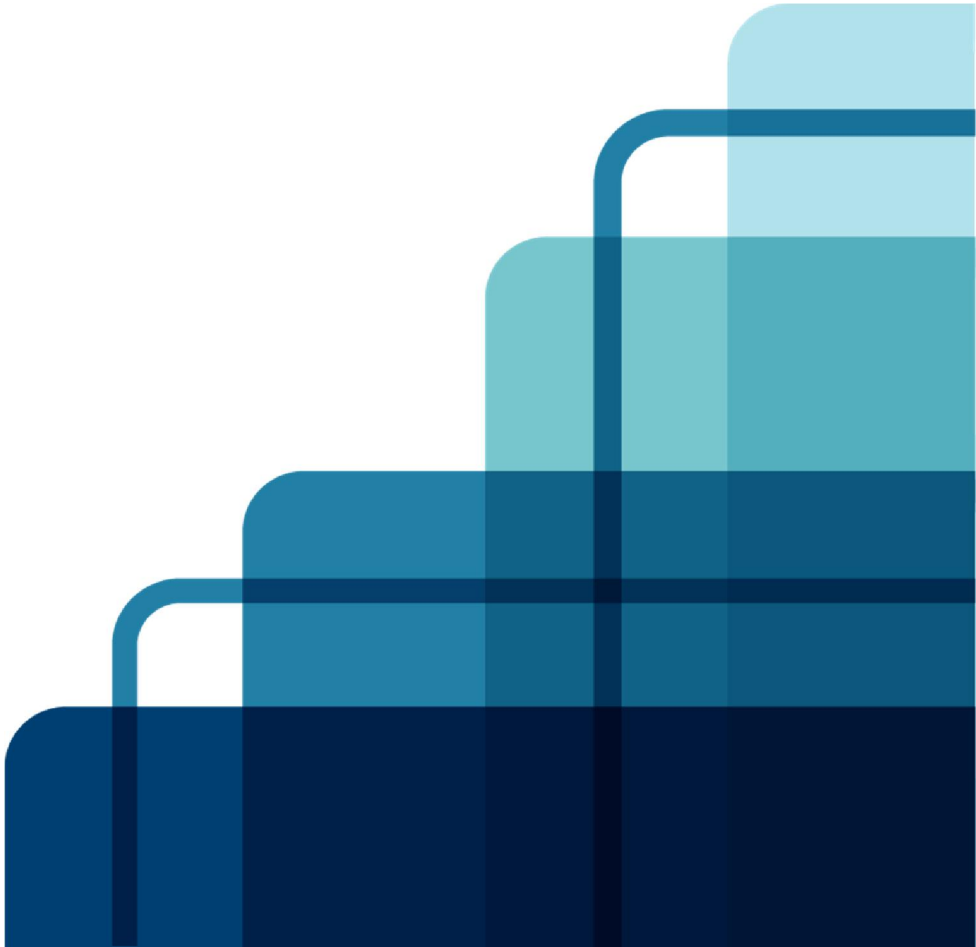
- Wprowadzenie
- Serializacja danych
- Przegląd technologii
- Wybrane problemy
- Q&A
- Zadanie

rule FINANCIAL

The Sector Specialists

Wprowadzenie

Confidential



Wprowadzenie

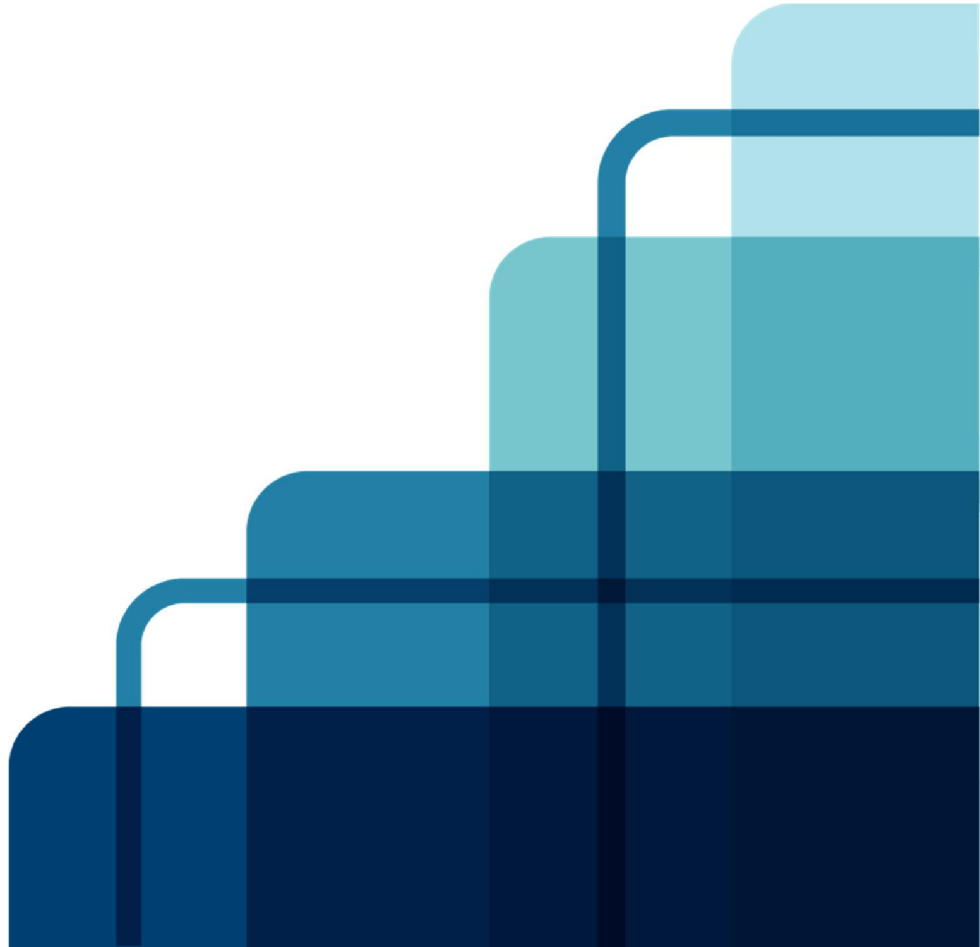
- [Za Wikipedią] **Serializacja** – w programowaniu komputerów proces przekształcania obiektów, tj. instancji określonych klas, do postaci szeregowej, czyli w strumień bajtów, z zachowaniem aktualnego stanu obiektu. Serializowany obiekt może zostać utrwalony w pliku dyskowym, przesłany do innego procesu lub innego komputera poprzez sieć. Procesem odwrotnym do serializacji jest deserializacja. Proces ten polega na odczytaniu wcześniej zapisanego strumienia danych i odtworzeniu na tej podstawie obiektu klasy wraz z jego stanem bezpośrednio sprzed serializacji.
- Serializacja to problem nietrywialny:
  - Różne architektury (x86 vs x64, big vs lil endian),
  - Złożone obiekty, cykle referencji,
  - Ograniczenia środowiska, protokołu,
  - Ograniczenia formatu danych:
    - Jak przesłać liczbę zmiennoprzecinkową jako tekst?
    - Jak reprezentować datę?
    - Jak przesłać tekst z polskimi znakami jako dane binarne?
- W dziś opisanych przykładach zakładam, że wynikiem serializacji są dane binarne (byte[]) lub tekst (string)

rule FINANCIAL

The Sector Specialists

Przegląd technologii

Confidential



Binary Formatter

- Mechanizm wbudowany w .NET, konwertuje dane do binarnego formatu zawierającego pełne dane o typie, dzięki czemu do deserializacji nie jest wymagana znajomość typu binarnego strumienia.
- Jako jedyny z dziś opisanych mechanizmów zapisuje wartości *fields* klasy. Pozostałe używają *properties*.
- W podstawowej wersji jedyne co trzeba zrobić to opisać klasę atrybutem `[Serializable]`
- Bardzo wygodna i szybka metoda jeżeli trzeba zapisać dane, które nie będą używane poza .NETem
- W wynikowym strumieniu zapisywane są dane o typie razem z *Fully Qualified Name*, więc do deserializacji jest wymagane *assembly*, które zostało użyte przy serializacji. Wymusza to używanie współdzielonej biblioteki między serwerem i klientem zawierającej definicje dto. W dużych projekach może to prowadzić do problemów jeżeli wersje biblioteki się rozjadą.
- Użycie:

```
using (var stream = new MemoryStream())
{
    var serializer = new BinaryFormatter();
    serializer.Serialize(stream, dto);
    return stream.ToArray();
}
```

```
using (var stream = new MemoryStream(data))
{
    var serializer = new BinaryFormatter();
    return serializer.Deserialize(stream) as PayloadDto;
}
```

Xml Serializer

- Drugi mechanizm wbudowany w .NET. W oparciu o atrybuty klasy produkuje dokument xml.
- Jeżeli ktoś nie wie, czym jest XML to za wikipedią: XML (ang. Extensible Markup Language, w wolnym tłumaczeniu Rozszerzalny Język Znaczników) – uniwersalny język znaczników przeznaczony do reprezentowania różnych danych w strukturalizowany sposób. XML jest niezależny od platformy, co umożliwia łatwą wymianę dokumentów pomiędzy heterogenicznymi (różnymi) systemami i znacząco przyczyniło się do popularności tego języka w dobie Internetu. XML jest standardem rekomendowanym oraz specyfikowanym przez organizację W3C.
- Do działania nie wymaga opisania klasy żadnymi atrybutami, jednak opisując klasę można dowolnie zmodyfikować wynikowy dokument.
- De facto standard przy zapisie danych, plików konfiguracyjnych w formacie niezależnym od platformy i czytelnym dla człowieka.
- Kontrakty zdefiniowane z użyciem XML Schema pozwalają na komunikację między komponentami projektu bez bibliotek współdzielonych bez obawy o problemy z integracją.
- Ciekawostka: nie serializuje typu Dictionary

Json Serializer

- Json jest bardzo prostym formatem zapisu danych pozwalającym na przesyłanie prostych struktur danych jako tekst.
- Jedna z lepszych implementacji jest udostępniona przez bibliotekę ServiceStack.
- Najpopularniejszy format wymiany danych w serwisach www, websockets, ajax etc.
- Nie zawiera informacji o typie, a deserializator jest bardzo liberalny dlatego trzeba bardzo uważać używając tego formatu.

```
1  {
2      "Number":516,
3      "Text":"str66",
4      "List":[81,21,18]
5  }
```

Protocol Buffers

- Protokół wymyślony przez google, umożliwia komunikację protokołem binarnym pomiędzy różnymi platformami.
- Jedną z implementacji jest protobuf-net.
- Bardzo wydajny protokół. Nie zawiera informacji o typie, więc w aplikacjach biznesowych zazwyczaj stosuje się serializację dwupoziomową. Przesyłany jest zawsze jeden i ten sam obiekt zawierający dwa pola – binarny payload i jego typ. Kolejna warstwa odpakowuje te dane do konkretnego obiektu.

Data Contracts

- Serializator XML dla WCF. Podobnie jak protobuf-net implementacja ta wymaga opisania klasy atrybutami. Do Xml Serializera odróżnia ją większa szybkość działania kosztem mniejszej elastyczności (np. nie można używać atrybutów w XMLu)
- Potrafi serializować grafy obiektów jako jedyna z opisanych dziś technologii.
- Szczegółowo opowie o nim Tomek na zajęciach z WCF.

Wybrane problemy z serializacją

- Dziedziczenie: czy utrzymywać hierarchię kontraktów?
- Współdzielone biblioteki: stary klient może się nie dogadać z nowym serwisem mimo że protokół się nie zmienił, a jedynie biblioteki serwisu mają podbitą wersję.
- Kompatybilność wstecz: dodanie nowego pola w kontrakcie wysypuje wszystkich starych klientów.
- Serializacja może zależeć od ustawień lokalnych: a = (123.456).ToString() w Polsce i Convert.ToDouble(a) w USA rzuci wyjątek. Dlaczego?
- XML wprowadza gigantyczny narzut gdy przechowuje się tablice liczb.
- Co się przy serializacji DAGa (czymś innym niż Data Contactami)?

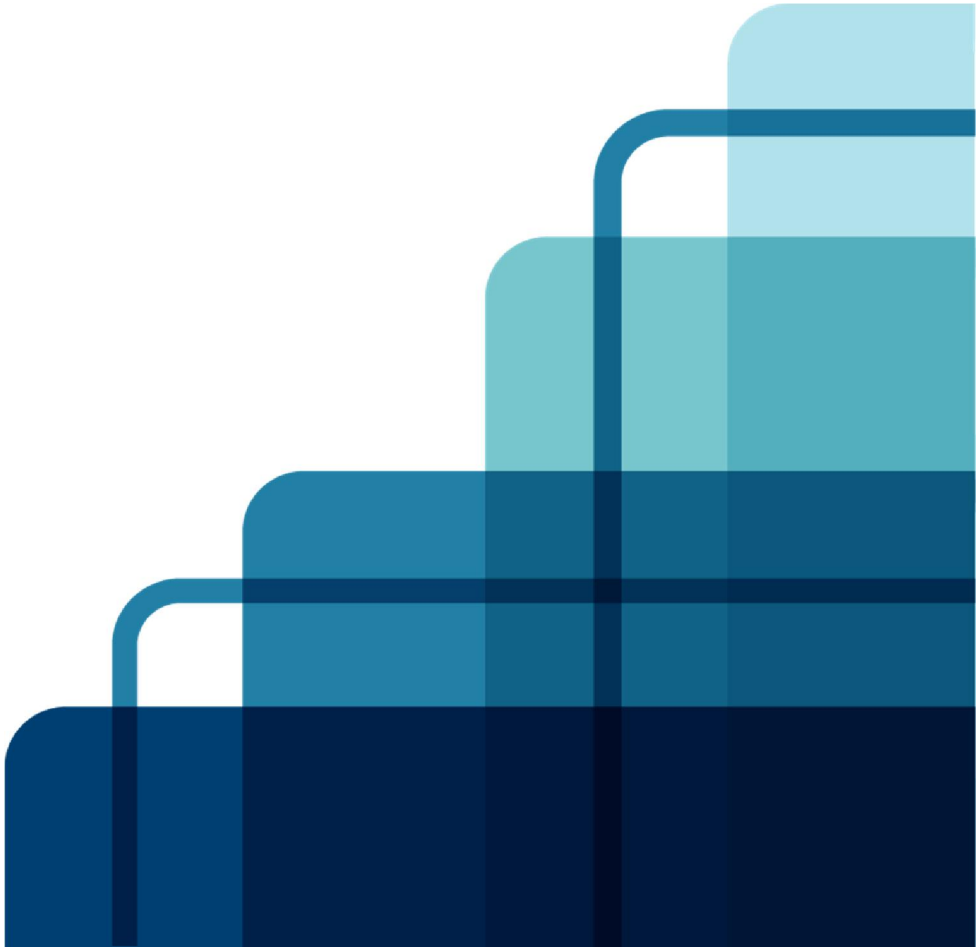
Q&A

rule FINANCIAL

The Sector Specialists

Zadanie

Confidential



Zadanie

- Pobrać projekt
- Z konsoli nugeta dodać pakiety: ServiceStack, protobuf-net
- Zadanie 1: dodać klasy do serializacji za pomocą: JSON, protocol buffers, xml. Wystarczy, że jedna metoda działa na raz.
- Zadanie 2: zmienić program tak, żeby do kolejki była wrzucana wiadomość serializowana losowo wybranym protokołem.
- W razie pytań o szczegóły każdej z metod serializacji polecam http://lmgtfy.com/