ИНТЕГРАЦИЯ ВИРТУАЛЬНЫХ МАШИН .NET И JAVA

Joker Conf

Григорий Кошелев

Санкт-Петербург, 19-20 октября 2018

О чём поговорим

- Использование Java в .NET-проекте
- .NET и Java
- C++, нативный код и прочие «кишки»

План (крупно)

- Задачи
- Радикальные способы решения
- Интеграция: Акт I (микросервисы)
- Интеграция: Акт II (Java Inside)
- Интеграция: Акт III (Deep Integration)

Дисклеймер

- СКБ Контур: 85% стека – .NET
- Докладчик: 85% стека – Java

Задачи

Задача №1

Много XML

Очень много XML

И всё это надо конвертировать в PDF

Решение: Apache FOP (XSL-FO + XML -> PDF)

Apache FOP

- Начало разработки – неизвестно
- Передана в ASF в 1999 году
- Apache FOP 1.0 – 01.07.2011
- Apache FOP 1.1 – 16.10.2012
- Apache FOP 2.0 – 02.06.2015
- Apache FOP 2.1 – 15.01.2016
- Apache FOP 2.2 – 10.04.2017

Задача №2

Много ЭП

Очень много ЭП

И всё это надо разбирать / проверять

Решение: Bouncy Castle

Bouncy Castle

	Версия	Первый релиз		Актуальный
Java	1.60	xx.05.2000	09.01.2018	02.07.2018
C#	1.8.3	13.10.2003	28.12.2015	11.08.2018

Задача №3

- Интеграция с Альфа-банком
- Java SDK

Радикальные способы решения

Радикальные способы

- **Выучить Java и писать всё на ней**

Радикальные способы

- Выучить Java и писать всё на ней
- **Переписать всё на C#**

Радикальные способы

- Выучить Java и писать всё на ней
- Переписать всё на C#
- **Сконвертировать код какой-нибудь тулзой**, а потом яростно допиливать

Конвертация кода Java -> C#

Where can I find a Java to C# converter? (StackOverflow, 2009)

27

Even if there is such a tool, I'd highly recommend you to do the conversion by hand. Automatic converters will often faithfully reproduce the code, but ignore idioms - because they'd be really, really hard to get right.

Furthermore, the differences between generics in .NET and Java could lead to some very different decisions in the two codebases.

Really, you'll be better off doing it by hand.

share

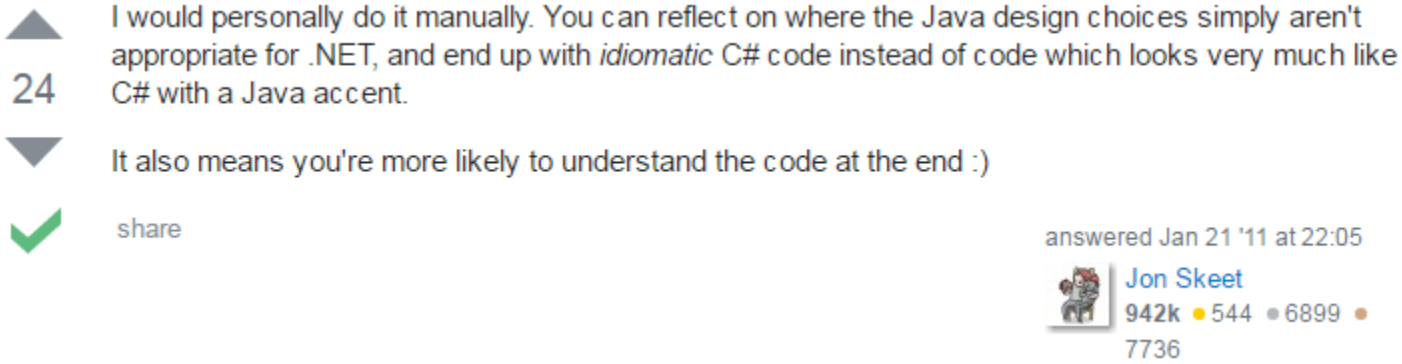
answered Jan 14 '09 at 13:54

Jon Skeet

942k ● 544 ● 6899 ● 7736

Конвертация кода Java -> C#

Convert Java to C# with a tool, or manually? (StackOverflow, 2011)



24

I would personally do it manually. You can reflect on where the Java design choices simply aren't appropriate for .NET, and end up with *idiomatic* C# code instead of code which looks very much like C# with a Java accent.

It also means you're more likely to understand the code at the end :)

share

answered Jan 21 '11 at 22:05

Jon Skeet

942k • 544 • 6899 • 7736

Радикальные способы

- Выучить Java и писать всё на ней
- Переписать всё на C#
- Сконвертировать код какой-нибудь тулзой, а потом яростно допиливать
- **Использовать кросс-компиляцию байткода**

Кросс-компиляция байткода

- IKVM.NET (https://www.ikvm.net/)
- Реализация JVM под .NET
- Реализация библиотеки классов Java в .NET
- Транслятор байткода (jar -> dll)
- IKVM 8.1 – 26.08.2015
- https://www.nuget.org/packages/IKVM/
- The End of IKVM.NET (21.04.2017)

Кросс-компиляция байткода

- Удачный опыт: конвертация Java-клиента для ZooKeeper и обёртки Curator; используется в продакшене
- Неудачный опыт: конвертация Apache FOP; стало работать в разы медленнее, выкинули

Радикальные способы

- Выучить Java и писать всё на ней
- Переписать всё на C#
- Сконвертировать код какой-нибудь тулзой, а потом яростно допиливать
- Использовать кросс-компиляцию байткода

Это всё разовые операции!

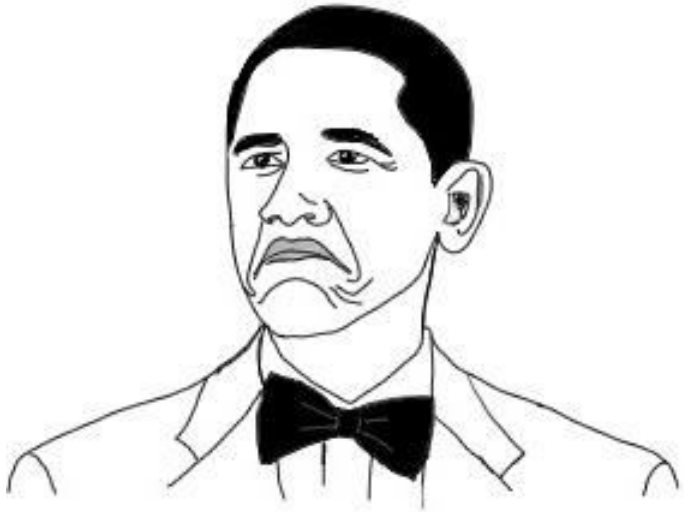
Радикальные способы

… не работают

Интеграция: Акт I

Java <-> .NET

- Микросервисы же!
- Транспорт? Формат?



Модельный пример

Будем считать хэши

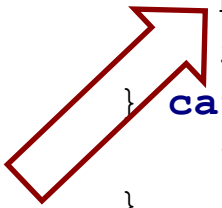
Пример:

SHA-512, { (byte) 'a' }

```
"1F40FC92DA241694750979EE6CF582F2D5D7D28E18
335DE05ABC54D0560E0F5302860C652BF08D560252A
A5E74210546F369FBBBCE8C12CFC7957B2652FE9A75
"
```

Модельный пример

```
public static String compute(String algorithm, byte[] bytes) {
    try {
        MessageDigest md = MessageDigest.getInstance(algorithm);
        md.update(bytes);
        return bytesToHex(md.digest());
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(e);
    }
}
```



Тестирование

- Intel Core i7-4710MQ, 2.5 ГГц
- 12 ГБ ОЗУ
- Windows 7 Professional x64
- Java 8 (1.8.0_162)
- .NET 4.6.1

Тестирование

-Xms4G

-Xmx4G

-XX:+UseG1GC

Инструменты для бенчмарков

- JMH 1.20
- BenchmarkDotNet 0.10.13

Результаты бенчмарка

- Алгоритм SHA-512
- Переменная длина массива байтов

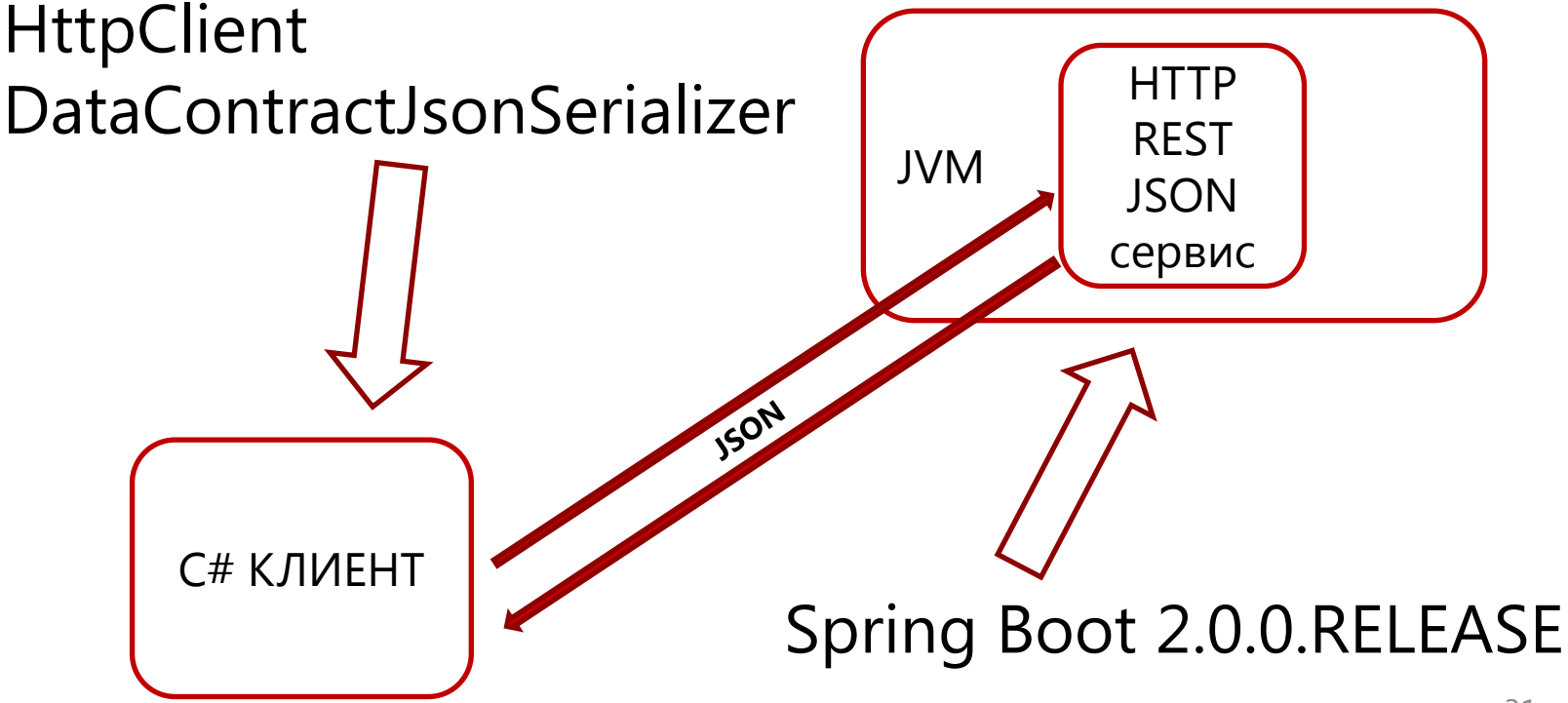
Результаты бенчмарка

- Алгоритм SHA-512
- Переменная длина массива байтов

	5 000	50 000	500 000
plain java	25 ± 1 мкс	240 ± 1 мкс	2 400 ± 20 мкс

Java HTTP REST JSON сервис

HttpClient
DataContractJsonSerializer



Результаты бенчмарка

- Алгоритм SHA-512
- Переменная длина массива байтов

	5 000	50 000	500 000
plain java	25 ± 1 мкс	240 ± 1 мкс	2 400 ± 20 мкс
JSON	?	?	?

Результаты бенчмарка

- Алгоритм SHA-512
- Переменная длина массива байтов

	5 000	50 000	500 000
plain java	25 ± 1 мкс	240 ± 1 мкс	2 400 ± 20 мкс
JSON	1 620 ± 30 мкс	10 270 ± 40 мкс	93 800 ± 500 мкс

gRPC

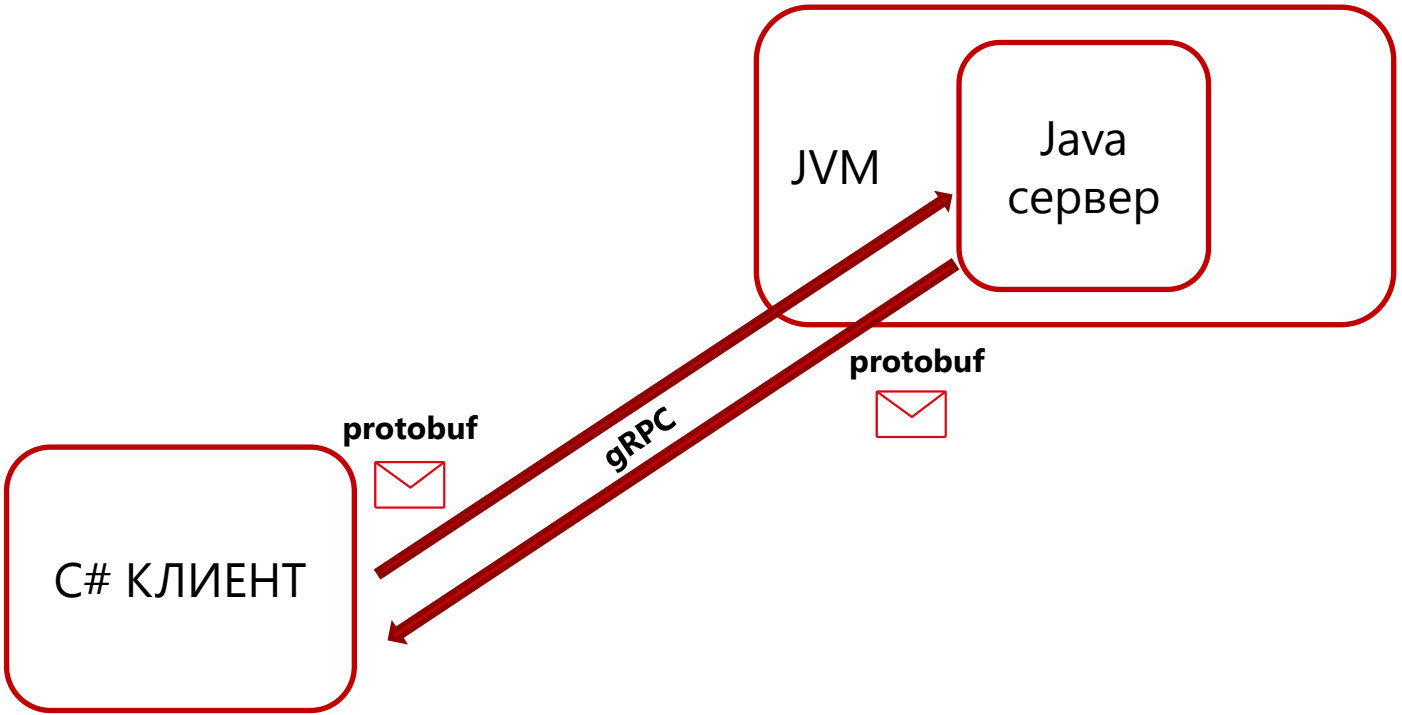
- Библиотека для удалённого вызова процедур
- Поддерживаются: C/C++, Node.js, Python, Ruby, Objective-C, PHP, C# и Java

- Первый публичный релиз – 26.02.2015
- gRPC 1.0.0 – 19.08.2016
- gRPC 1.10.0 – 01.03.2018
- gRPC 1.15.0 – 12.09.2018

gRPC изнутри

- Protobuf 3 – описание типов данных и сериализация
- HTTP/2 в качестве транспорта
- Кодогенерация плагином для Protobuf

Java gRPC сервис



Результаты бенчмарка

- Алгоритм SHA-512
- Переменная длина массива байтов

	5 000	50 000	500 000
plain java	25 ± 1 мкс	240 ± 1 мкс	2 400 ± 20 мкс
gRPC	?	?	?
JSON	1 620 ± 30 мкс	10 270 ± 40 мкс	93 800 ± 500 мкс

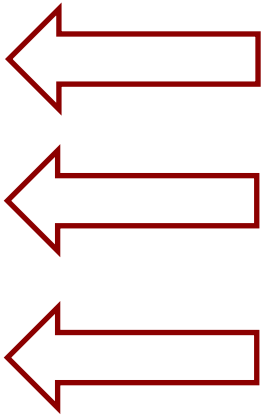
Результаты бенчмарка

- Алгоритм SHA-512
- Переменная длина массива байтов

	5 000	50 000	500 000
plain java	25 ± 1 мкс	240 ± 1 мкс	2 400 ± 20 мкс
gRPC	207 ± 3 мкс	612 ± 2 мкс	5 300 ± 500 мкс
JSON	1 620 ± 30 мкс	10 270 ± 40 мкс	93 800 ± 500 мкс

Лог бенчмарка

№ итерации	мс / оп
1	4,2
2	4,2
…	…
10	4,2
11	8,9
12	4,3
13	6,5
14	6,5
15	4,5



Memory Traffic (gc.log)

- Паузы от 6 до 8 мс (Young Gen)
- Allocation rate ~ 200 МБ/с

Performance Counters (C#)

- Heap поделён на три поколения: 0, 1 и 2

Performance Counters (C#)

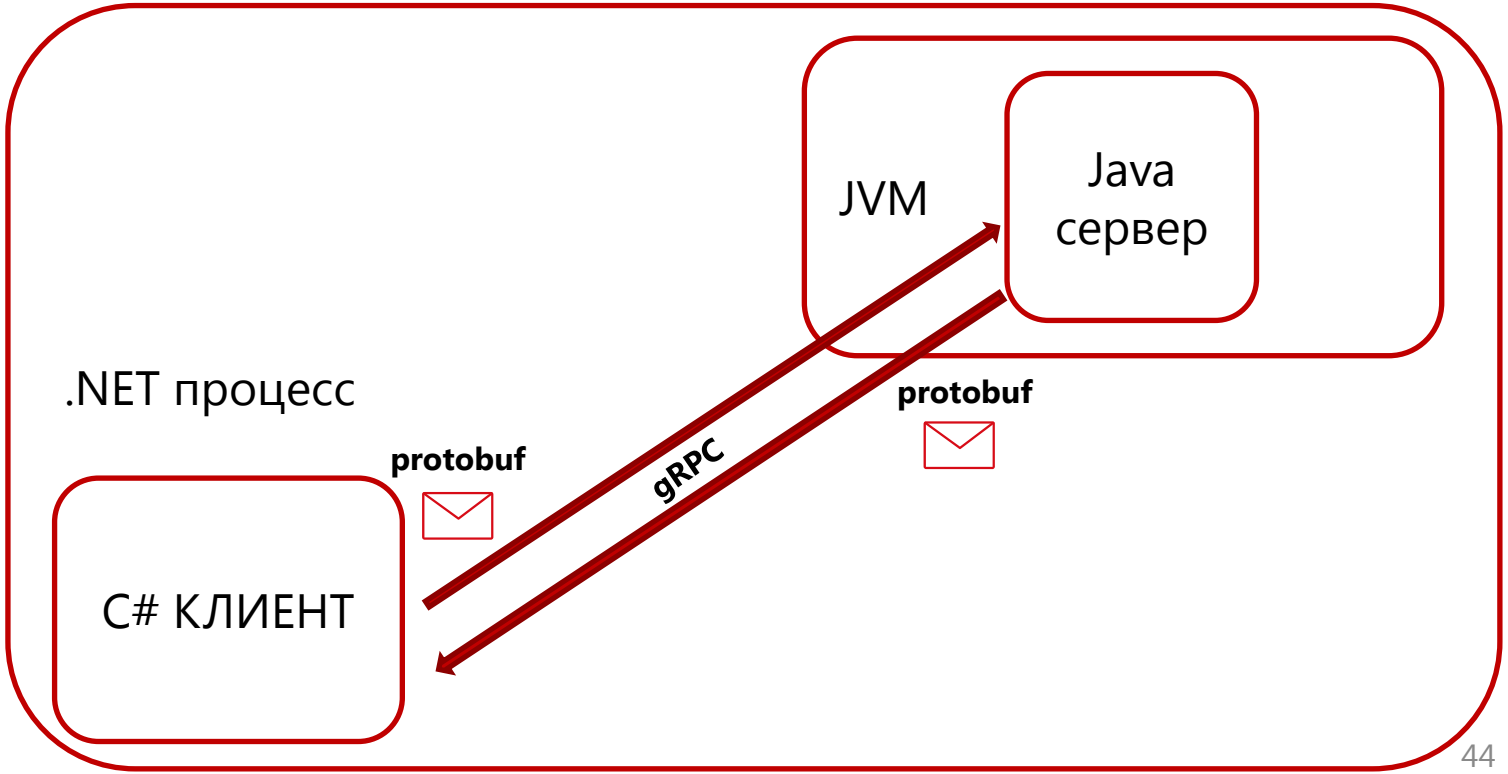
- Heap поделён на три поколения: 0, 1 и 2
- Счётчики GC по поколениям:

	5 000	50 000	500 000
Gen 0	300	2 583	133
Gen 1	2	5	88
Gen 2	0	0	88

- Gen 0 и Gen 1 – Background GC

Интеграция: Акт II

JVM внутри .NET-процесса



Мотивация

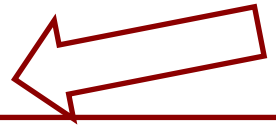
- Инфраструктурные ограничения
- Перспектива снижения издержек на передачу данных

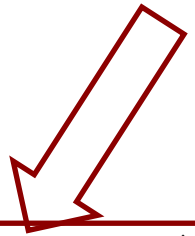
Java Native Interface

- Появился в 1.1
- Позволяет вызывать нативный код из Java
- Но это не всё! Есть Invocation API
- Позволяет управлять JVM из нативного кода

Java Native Interface

```
JavaVM *jvm;
JNIEnv *env;
JavaVMInitArgs args;
JavaVMOption* options = new JavaVMOption[1];
options[0].optionString = params;
args.nOptions = 1;
args.options = options;
args.version = JNI_VERSION_1_6;
args.ignoreUnrecognized = 0;
int result = JNI_CreateJavaVM(&jvm, (void**)&env, &args);
```





Осторожно!

C++

Java Native Interface

```
jclass programClass = env->FindClass("ru/kontur/Program");
jmethodID doSmthMethod = env->GetStaticMethodID(programClass, "doSmth",
"(ILjava/lang/String;)I");
jint intParam = …;
jstring stringParam = …;
jint result = env->CallStaticIntMethod(programClass, doSmthMethod, intParam,
stringParam);
```

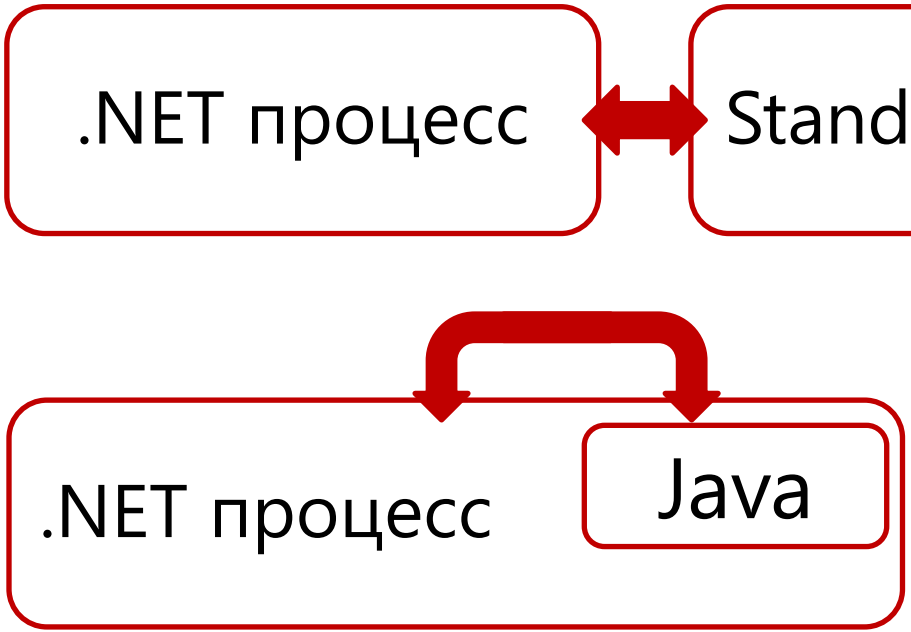
Осторожно!

C++

```
package ru.kontur;

public class Program {
    public static int doSmth(int x, String str) {
        return 0;
    }
}
```

C#-обёртка вокруг JNI

gRPC снаружи vs gRPC внутри



gRPC снаружи vs gRPC внутри

Результат нагрузочного теста

gRPC снаружи vs gRPC внутри

Результат нагрузочного теста

	Среднее, мс	Стд. откл, мс	Относительное время
Снаружи	73 065	2 367	100%
Внутри	72 002	1 266	99%

gRPC снаружи vs gRPC внутри

План по реализации

- Импорт Java-бинаря в .NET-процесс
- Создаём JVM внутри .NET-процесса
- Поднимаем gRPC-сервис

План по реализации

- **Импорт Java-бинаря в .NET-процесс**
- Создаём JVM внутри .NET-процесса
- Поднимаем gRPC-сервис

Импорт JVM.dll

```
C:\Program Files\Java\jre1.8.0_xxx\bin\server\jvm.dll
```

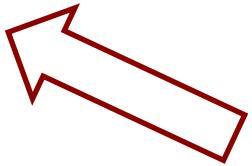
PInvoke – вызов неуправляемого кода (DLL) из управляемой среды CLR

План по реализации

- Импорт Java-бинаря в .NET-процесс
- **Создаём JVM внутри .NET-процесса**
- Поднимаем gRPC-сервис

Процедура создания JVM

```
public unsafe void CreateJavaVm(IntPtr* vm, IntPtr* env, IntPtr args)
{
    /* ... */
}
```



План по реализации

- Импорт Java-бинаря в .NET-процесс
- Создаём JVM внутри .NET-процесса
- **Поднимаем gRPC-сервис**

C#-обёртка вокруг JNI

```
JavaVmWrapper vm;
JniEnvWrapper env;
unsafe
{
    IntPtr envP; IntPtr vmP;
    CreateJavaVm(&vmP, &envP, vmArgsP);
    vm = new JavaVmWrapper(vmP);
    env = new JniEnvWrapper(envP);
}
var classRef = env.FindClass("ru/kontur/Program");
var constructor = env.GetMethodId(classRef, "<init>", "()V");
var runMethod = env.GetMethodId(classRef, "run", "()V");
var obj = env.NewObject(classRef, constructor, port);
env.CallObjectMethod(obj, runMethod);
```

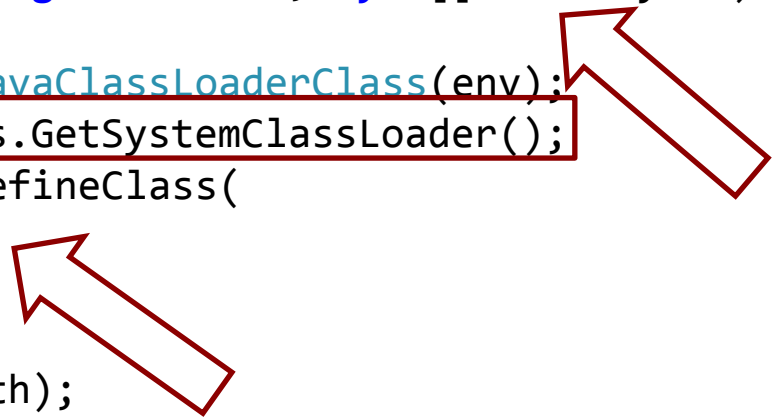
Загрузка классов

- classpath `java -classpath C:\fop\jars ru.kontur.fop.FopService`
- ClassLoader

```
URL[] jars =
    new URL[]{new URL("file:///c:/fop/jars/fop.jar")};
URLClassLoader cl = new URLClassLoader(jars);
Class<?> serviceClass =
    cl.loadClass("ru.kontur.fop.FopService");
```

Загрузка классов

```
public void InjectClass(string className, byte[] classBytes)
{
    var loaderClass = new JavaClassLoaderClass(env);
    var loader = loaderClass.GetSystemClassLoader();
    var loadedClass = env.DefineClass(
            className,
            loader,
            classBytes,
            classBytes.Length);
}
```



Промежуточные итоги

- Запустили виртуальную машину Java внутри .NET процесса
- Реализовали обмен данными между программами на Java и на C#
- Получили готовый работающий прототип решения задачи конвертации документов

Мотивация? Но я же…

- ~~Инфраструктурные ограничения~~
- ~~Перспектива снижения издержек на передачу данных~~

Интеграция: Акт III

Фундаментальные типы

Фундаментальные типы

Java	.NET
double	double
float	float
long	long
int	int
short	short
char	char
byte	sbyte, byte
boolean	bool

Фундаментальные типы

Java	.NET	JNI
double	double	jdouble
float	float	jfloat
long	long	jlong
int	int	jint
short	short	jshort
char	char	jchar
byte	sbyte, byte	jbyte
boolean	bool	jboolean

JNI-типы и Native-типы

```
// jni_x86.h
#if defined(_WIN32)
  typedef int jint;
  typedef __int64 jlong;
#else
  typedef int jint;
  #if defined(_LP64)
    typedef long jlong;
  #else
    typedef long long jlong;
  #endif
#endif
typedef signed char jbyte;
```

```
// jni.h
typedef unsigned char  jboolean;
typedef unsigned short jchar;
typedef short          jshort;
typedef float          jfloat;
typedef doable         jdouble;
```

Фундаментальные типы

- Не нужно (пока) беспокоиться за Endianness
- Типы совпадают
  (за исключением bool/boolean)
- Маршаллинг из unsigned int8 для bool

Указатели в нативную память

- Ручное управление объектами, размещёнными в нативной памяти
- Используем using-блоки (аналог try-with-resources)

Объекты из Java heap

- Объект существует на стороне Java
- В нативный код передаётся по «локальной» ссылке
- Как Java GC узнает, что пора собирать?

```
var wrappedMethodId =
        Env.GetMethodId(classPtr, "<init>", "(I)V");
/* ... */
Env.DeleteLocalRef(wrappedMethodId.Ptr);
```

А что с многопоточностью?

```
var env = Jvm.AttachCurrentThread();
/* process */
Jvm.DetachCurrentThread();
```

```
var globalRef = Env.NewGlobalRef(localRef);
/* process */
Env.DeleteGlobalRef(globalRef.Ptr);
```

```
synchronized (obj) {
    /* synchronized block */
}
```

```
Env.MonitorEnter(obj);
    /* synchronized block */
Env.MonitorExit(obj);
```

Строки

- Строка – массив UTF-16 char в .NET и Java (до Java 9)
- Native-строка – массив UTF-8 char

C#, Java и «нативные строки»

```
jclass (JNICALL *FindClass) (JNIEnv *env, const char *name);
```

```
delegate IntPtr FindClass(
    IntPtr env,
    [MarshalAs(UnmanagedType.LPArray)] byte[] utf);
```

```
Encoding.UTF8.GetBytes(str);
```

Латинская буква «s» (0x73) ✔
Кириллическая буква «К» (0xD0 0x9A) ✔
Символ года на тамильском «ஶ்ரீ» (0xE0 0xAF 0xB5) ✔
Китайский иероглиф «缔» (0xF0 0xA6 0x88 0x98)

C#, Java и «нативные строки»

Символ	UTF-8	Unicode
s	0x73	U+0073
К	0xD0 0x9A	U+041A
	0xE0 0xAF 0xB5	U+0BF5
绺	0xF0 0xA6 0x88 0x98	U+26218

C#, Java и «нативные строки»

Символ	UTF-8	Unicode	Размер
s	0x73	U+0073	1
К	0xD0 0x9A	U+041A	2
	0xE0 0xAF 0xB5	U+0BF5	3
绺	0xF0 0xA6 0x88 0x98	U+26218	6

Modified UTF-8

Неуловимый баг

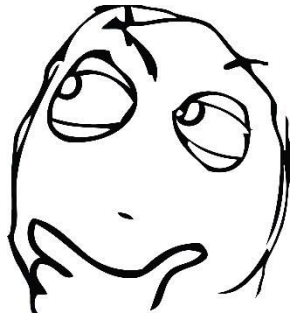
```
jclass (JNICALL *FindClass) (JNIEnv *env, const char *name);
```

```
byte[] utf = Encoding.UTF8.GetBytes(str);
FindClass(env, utf);
```

Нет терминального символа «null» (0x00)!

Почему работает?

Значит, «нулевой» байт стоит следом!



Ловим неуловимого

- Так zeroing же!

```
var obj = new object();
```

Ловим неуловимого

- Так zeroing же!
- Так выравнивание же!

```
byte[] bytes = /* ... */
```

Ловим неуловимого

- Так zeroing же!
- Так выравнивание же!

```
byte[] bytes = /* ... */
```

Segment	Size, bytes
Sync block	4 (x32), 8 (x64)
Type handle	4 (x32), 8 (x64)
Length	4
Array values	Length
Alignment	0 - 7

Ловим неуловимого

- Так zeroing же!
- Так выравнивание же!
- Так Sync block == 0 же!

```
lock (obj) { /* ... */ }

obj.GetHashCode();
```

Ловим неуловимого

- Так zeroing же!
- Так выравнивание же!
- Так Sync block == 0 же!
- Фикс! Фикс! Фикс!

```
int count = Encoding.UTF8.GetByteCount(str);
byte[] utf = new byte[count + 1];
Encoding.UTF8.GetBytes(str, 0, str.Length, utf, 0);
FindClass(env, utf);
```

UUID vs GUID

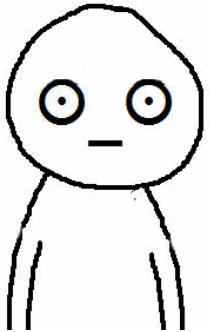
```
  0                                              15
"6e5bde46-c043-11e8-a355-529269fb1459"
```

```
Guid guid1 = Guid.Parse("6e5bde46-c043-11e8-a355-529269fb1459");
Guid guid2 = new Guid(new byte[] {
               0x6e, 0x5b, 0xde, 0x46,
               0xc0, 0x43,
               0x11, 0xe8,
               0xa3, 0x55,
               0x52, 0x92, 0x69, 0xfb, 0x14, 0x59 });

Guid guid3 = Guid.Parse("46de5b6e-43c0-e811-a355-529269fb1459");
```

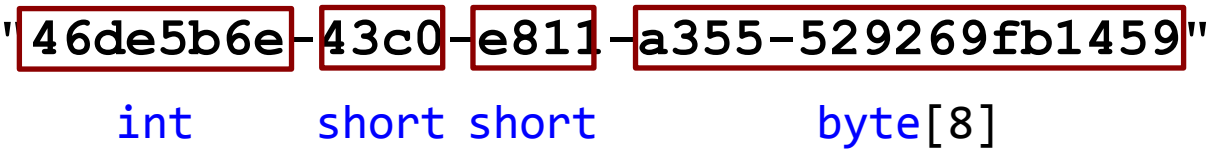
UUID vs GUID

```
  0                                      15
"6e5bde46-c043-11e8-a355-529269fb1459"
```

```
Guid guid1 = Guid.Parse("6e5bde46-c043-11e8-a355-529269fb1459");
Guid guid2 = new Guid(new byte[] {
                0x6e, 0x5b, 0xde, 0x46,
                0xc0, 0x43,
                0x11, 0xe8,
                0xa3, 0x55,
                0x52, 0x92, 0x69, 0xfb, 0x14, 0x59 });
Guid guid3 = Guid.Parse("46de5b6e-43c0-e811-a355-529269fb1459");
```



UUID vs GUID



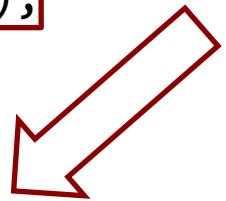
Little Endian наносит ответный удар

Exception

```
Env.ExceptionCheck()

Env.ExceptionDescribe()

Env.ExceptionOccured()

Env.ExceptionClear()
```

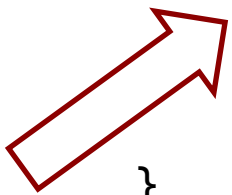
Exception

```
T SafeJniCall<T>(Func<T> func)
{
    Env.ExceptionClear();
    var result = func();
    return result;
}

void SafeJniCallVoid(Action func)
{
    Env.ExceptionClear();
    func();
}
```



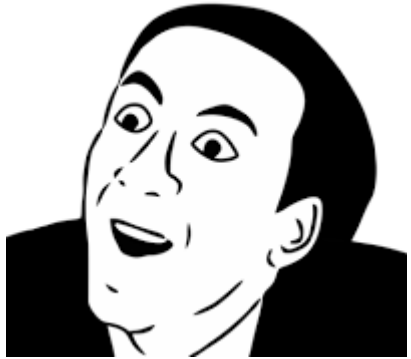
Exception

```
bool TryCatchException(out LocalRef ex)
{
    if (Env.ExceptionCheck())
    {
        ex = new LocalRef(Env.ExceptionOccurred());
        Env.ExceptionClear();
        return true;
    }
    ex = null;
    return false;
}
```



Enum

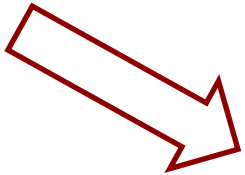
```
package ru.kontur.fop;
public enum TransformationType {
    CSV, PDF, SVG;
}
```

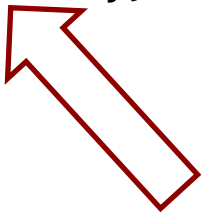


```
var enumClass = env.FindClass("ru/kontur/fop/TransformationType");
var fieldIdPDF = env.GetStaticFieldID(enumClass, "PDF",
                 "Lru/kontur/fop/TransformationType;");
var PDF = env.GetStaticObjectField(enumClass, fieldIdPDF);
```

byte[]

```
using (var byteArray = Env.NewByteArray(bytes))
{
    Env.CallObjectMethod(
        obj,
        method,
        new JValue() { PointerValue = byteArray.Ptr };
}
```





```
public byte[] convert(byte[] data) {
    // ...
}
```

Результаты бенчмарка

- Алгоритм SHA-512
- Переменная длина массива байтов

	5 000	50 000	500 000
plain java	25 ± 1 мкс	240 ± 1 мкс	2 400 ± 20 мкс
j4net proxy	?	?	?
gRPC	207 ± 3 мкс	612 ± 2 мкс	5 300 ± 500 мкс
JSON	1 620 ± 30 мкс	10 270 ± 40 мкс	93 800 ± 500 мкс

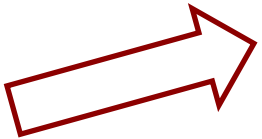
Результаты бенчмарка

- Алгоритм SHA-512
- Переменная длина массива байтов

	5 000	50 000	500 000
plain java	25 ± 1 мкс	240 ± 1 мкс	2 400 ± 20 мкс
j4net proxy	32 ± 1 мкс	266 ± 1 мкс	2 600 ± 23 мкс
gRPC	207 ± 3 мкс	612 ± 2 мкс	5 300 ± 500 мкс
JSON	1 620 ± 30 мкс	10 270 ± 40 мкс	93 800 ± 500 мкс

DirectByteBuffer

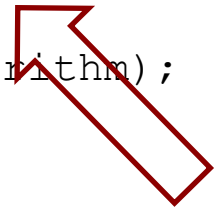
```
fixed (byte* b = bytes)
{
    var buffer =
        Env.NewDirectByteBuffer((IntPtr)b, bytes.Length));
    Env.CallIntMethod(
        obj,
        method,
        new JValue() { PointerValue = buffer.Ptr });
}
```



```
public int convert(ByteBuffer buffer) {
    byte[] data = new byte[buffer.capacity()];
    buffer.get(data);
    // convert
}
```

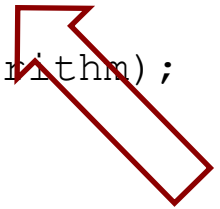
DirectByteBuffer

```
public static String compute(String algorithm, byte[] bytes) {
    try {
        MessageDigest md = MessageDigest.getInstance(algorithm);
        md.update(bytes);
        return bytesToHex(md.digest());
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(e);
    }
}
```



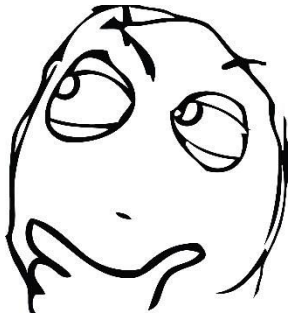
DirectByteBuffer

```
public static String compute(String algorithm, ByteBuffer buff) {
    try {
        MessageDigest md = MessageDigest.getInstance(algorithm);
        md.update(buff);
        return bytesToHex(md.digest());
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(e);
    }
}
```



DirectByteBuffer (бенчмарк)

	byte []	DirectByteBuffer
5 000	32 ± 1 мкс	31 ± 1 мкс
50 000	266 ± 1 мкс	250 ± 4 мкс
500 000	2 600 ± 23 мкс	2 420 ± 10 мкс



DirectByteBuffer (лоад-тест)

	byte []	DirectByteBuffer
5 000	2	2
50 000	3	2
500 000	22	2

* Количество вызовов GC в Java

GC pause (всего): 56 мс vs 9 мс

Allocation rate: 200 МБ/с vs 2,5 МБ/с

DetachCurrentThread

```
var env = Jvm.AttachCurrentThread();
/* processing */
Jvm.DetachCurrentThread();
```

DetachCurrentThread

```
[Benchmark]
public void AttachDetach()
{
    var env = Jvm.AttachCurrentThread();
    Jvm.DetachCurrentThread();
}
```

63 ± 1 мкс

Где может пригодиться?

- Десктоп – всё в одном процессе (удобно)
- Большой memory traffic (быстро)
- Не нужна дополнительная инфраструктура для работы (просто и надёжно)

j4net

https://github.com/j4net/

Осторожно!
Прототип!

Спасибо за внимание!

ВОПРОСЫ?

СКБ Контур

https://kontur.ru/
https://t.me/java_ural_Meetup

Григорий Кошелев

@GregoryKoshelev

kgn@skbkontur.ru

Ссылки

- Примеры на Github: https://github.com/gnkoshelev (ASAP)
- j4net: https://github.com/j4net (прототип, J4Net.Core использовался в примерах)
- gRPC http://www.grpc.io/
- JNI http://docs.oracle.com/javase/8/docs/technotes/guides/jni/spec/jniTOC.html
- Invocation API http://docs.oracle.com/javase/8/docs/technotes/guides/jni/spec/invocation.html