Производительность: нюансы против очевидностей

Пару слов о себе

- работаю в «Леви9»
- пишу на Java
- качаю производительность

https://github.com/stsypanov

https://habr.com/ru/users/tsypanov/posts

Что, чем и на чём измеряли

Код	https://github.com/stsypanov/ovn
JMH	1.28
Железо	Intel Core i7-7700
Java	14, 11 и 8
Режим	среднее время выполнения + расход памяти (меньше = лучше)

О чём доклад?

Мне часто попадаются любопытные примеры ***очевидно*** полезных и улучшающих производительность изменений, которые на поверку оказываются не только бесполезными, а иногда и вредными.

Доклад о нюансах, скрытых за ***очевидно*** производительным кодом.

Что такое нюанс?

Нюанс (франц. nuance) - это оттенок, тонкое различие, едва заметный переход (в цвете, мысли, звуке и т. п.) - БЭС

https://slovar.cc/enc/bolshoy/2105945.html

Что такое нюанс?

Нюанс (франц. nuance) - это оттенок, тонкое различие, едва заметный переход (в цвете, мысли, звуке и т. п.) - БЭС

https://slovar.cc/enc/bolshoy/2105945.html

Очень пошлое (вместе с тем очень верное) объяснение по ссылке

https://www.anekdot.ru/id/101180/

Почему возникают нюансы?

Почему возникают нюансы?

Краткий ответ: потому что Java многослойна.

Почему возникают нюансы?

Краткий ответ: потому что Java многослойна.

В общих чертах исполнение программы выглядит так:

- компилятор превращает исходный код в байт-код
- байт-код исполняется виртуальной машиной
- оптимизирующий компилятор (многослойный) меняет код на лету по мере его исполнения
- процессор исполняет итоговые инструкции, полученные от компилятора

Какие бывают нюансы?

- нюансы реализации/алгоритма

- нюансы исполнения

- нюансы измерения

Нюансы измерения

Очевидность

Неправильно написанный бенчмарк даёт неверные результаты.

Ложное и опасное заблуждение

Неправильно написанный бенчмарк даёт неверные результаты.

Из этого наблюдения делается ложный и опасный вывод:

если бенчмарк написан правильно, то его вывод можно использовать как доказательство очевидной полезности тех или иных изменений в коде.

Прогуляемся по массиву?

```
byte[] bunn;


for (int i = 0; i < bunn.length; i++) {
  use(bunn[i]);
}
```

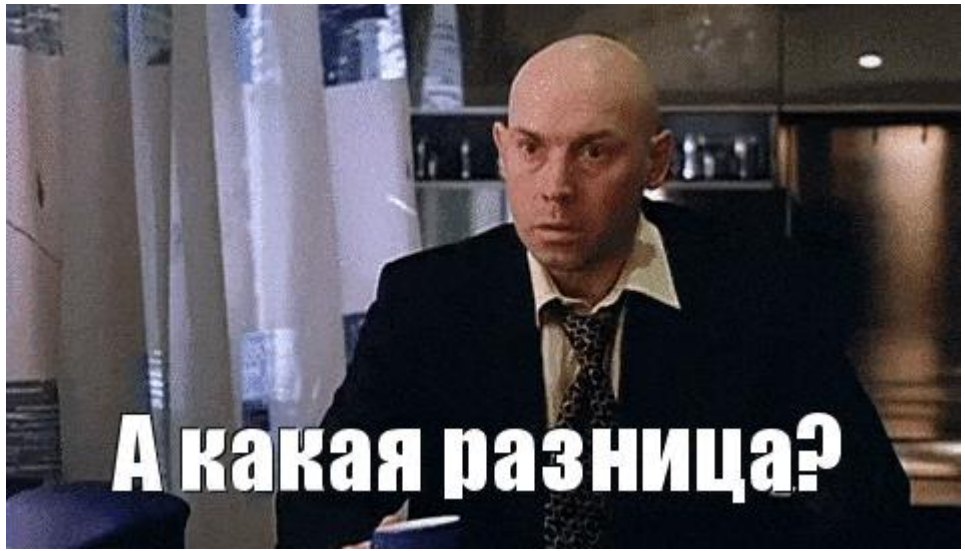
Прогуляемся по массиву?

```
byte[] bunn;
```

```
for (int i = 0; i < bunn.length; i++) {
  use(bunn[i]);
}
```

```
for (byte bunny : bunn) {
  use(bunny);
}
```

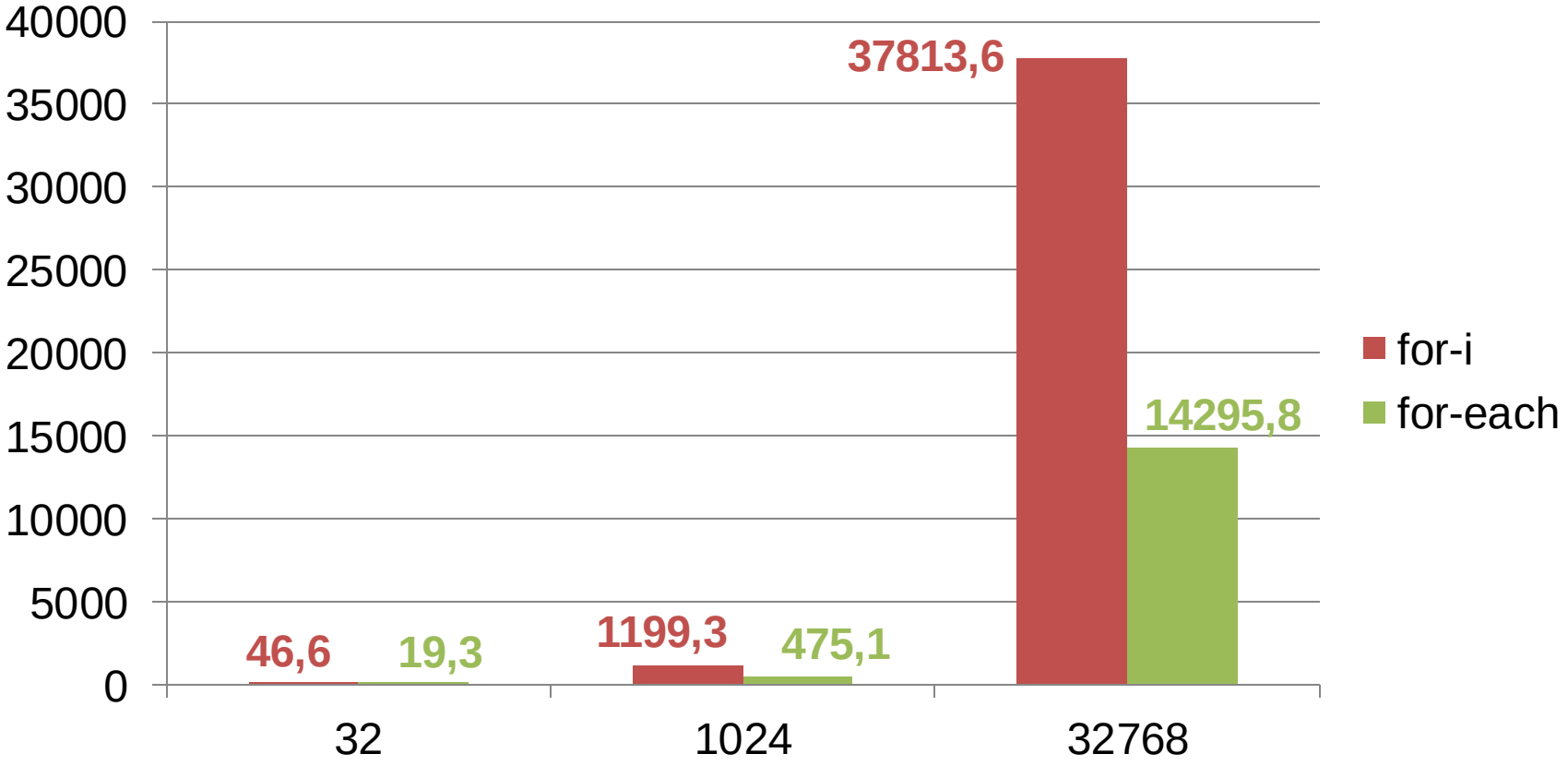
Действительно



Прогуляемся по массиву?

```
byte[] bunn;

@Benchmark
public void goodOldLoop(Blackhole bh) {
  for (int i = 0; i < bunn.length; i++)
    bh.consume(bunn[i]);
}

@Benchmark
public void sweetLoop(Blackhole bh) {
  for (byte bunny : bunn)
    bh.consume(bunny);
}
```

LoopyBenchmark, время в нс (Java 8)



Как в лучших домах Потсдама

```
int[] xs;

@Benchmark
public void measureRight_1(Blackhole bh) {
  for (int x : xs) bh.consume(work(x));
}
```

https://hg.openjdk.java.net/code-tools/jmh/file/523c524305e2/jmh-samples/src/main/java/org/openjdk/jmh/samples/JMHSample_34_SafeLooping.java#l144

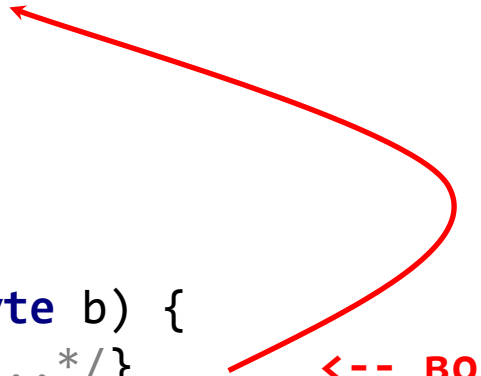
Внутри org.openjdk.jmh.infra.Blackhole (< 1.3)

```
public volatile byte b1, b2;                <--
public volatile Blackhole nullB = null;

public final void consume(byte b) {
 if (b == b1 & b == b2) {
  // SHOULD NEVER HAPPEN
  nullB.b1 = b; // implicit NPE
 }
}
```

http://psy-lob-saw.blogspot.com/2014/08/the-volatile-read-suprise.html

Чем плохо?

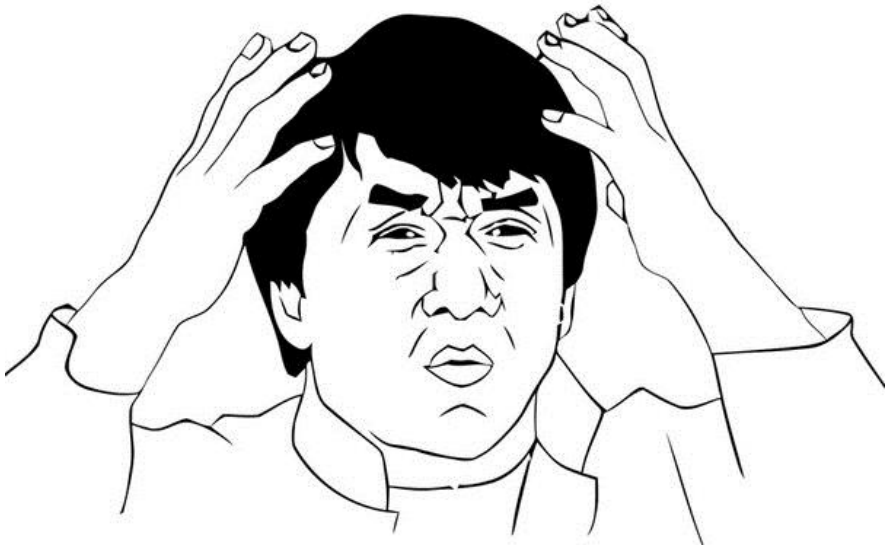
```
byte[] bunn;

@Benchmark
public void goodOldLoop(Blackhole bh) {
  for (int i = 0; i < bunn.length; i++)
    bh.consume(bunn[i]);
}

public volatile byte b1, b2;

public final void consume(byte b) {
 if (b == b1 & b == b2) {/*...*/}
```



<-- волатильный доступ

А на чём же выехал for-each?

Это тот же самый for-i!!!



А на чём же выехал for-each?

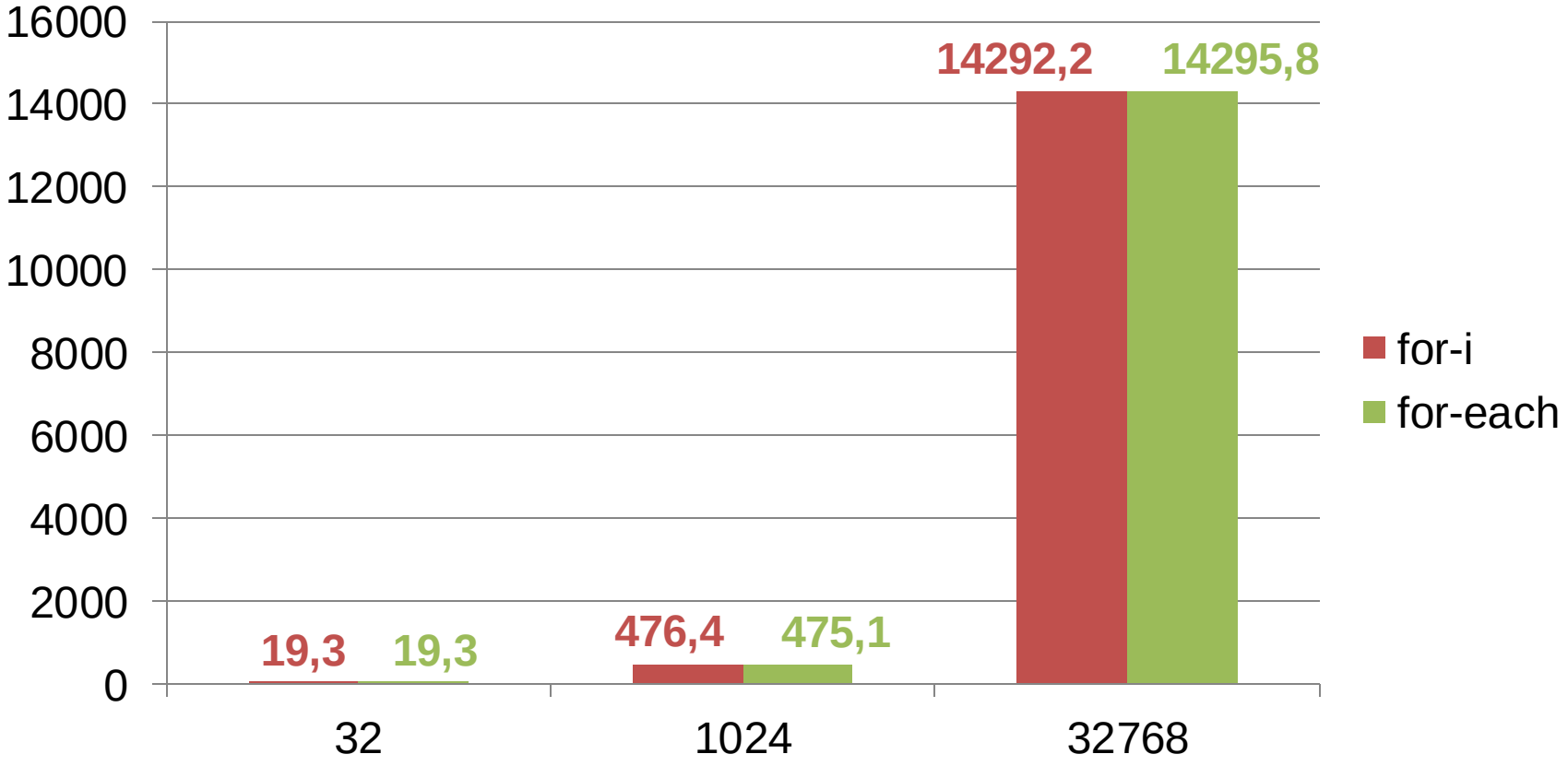
Это **почти** тот же самый for-i

```
@Benchmark
public void sweetLoop(Blackhole fox) {
  byte[] var2 = this.bunn;                    <-- неявное чтение поля
  int var3 = var2.length;

  for(int var4 = 0; var4 < var3; ++var4) {
    byte bunny = var2[var4];
    fox.consume(bunny);
  }
}
```

Синхронизация на стеке творит чудеса

```
byte[] bunn;

@Benchmark
void sweetLoop(Blackhole bh) {
  for (byte bunny : bunn)
    bh.consume(bunny);
}

@Benchmark
void goodOldLoopReturns(Blackhole fox) {
  byte[] sunn = bunn;
  for (int y = 0; y < sunn.length; y++)
    fox.consume(sunn[y]);
}
```

LoopyBenchmark, время в нс (Java 8)



Внутри Blackhole (JMH >= 1.3)

- было

```
public volatile byte b1, b2;

public final void consume(byte b) {
  if (b == b1 & b == b2) { nullB.b1 = b; }
}
```

Внутри Blackhole (1.3 < JMH <= 1.27)

- было

```
public volatile byte b1, b2;

public final void consume(byte b) {
  if (b == b1 & b == b2) { nullB.b1 = b; }
}
```

- стало

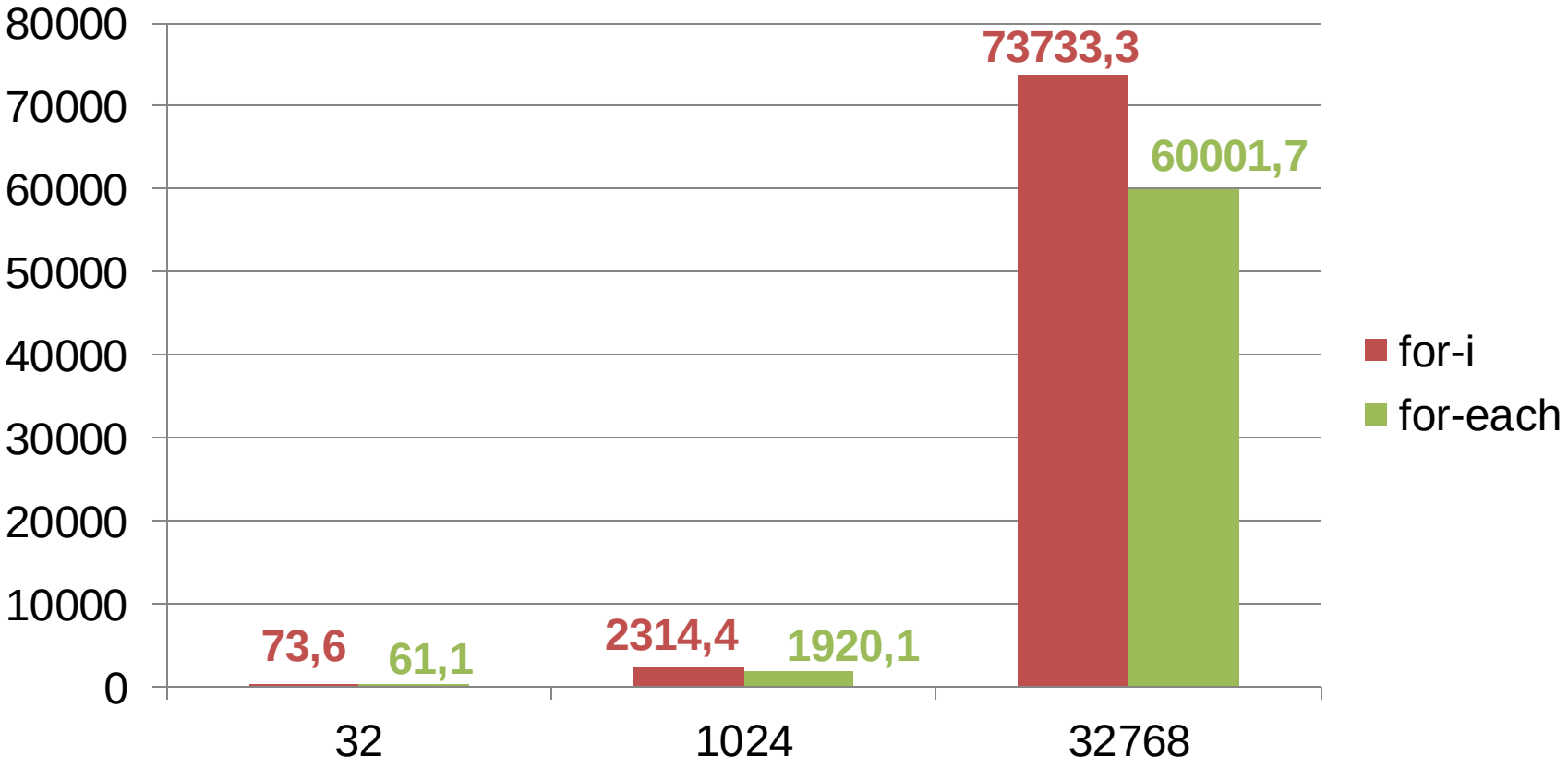
```
public volatile byte b1;
public byte b2;

public final void consume(byte b) {
  byte b1 = this.b1;
  byte b2 = this.b2;
  if ((b ^ b1) == (b ^ b2)) nullB.b1 = b;
}
```

Внутри Blackhole (1.3 < JMH <= 1.27)

https://bugs.openjdk.java.net/browse/CODETOOLS-7901096

http://cr.openjdk.java.net/~shade/7901096/webrev.00/jmh-core/src/main/java/org/openjdk/jmh/infra/Blackhole.java.sdiff.html

LoopyBenchmark, время в нс (Java 8)



Внутри Blackhole (JMH >= 1.28)

```
public final void consume(byte b) {
  if (COMPILER_BLACKHOLE) {
    consumeCompiler(b);
  } else {
    consumeFull(b);
  }
}

private static void consumeCompiler(byte v) {}
```

https://github.com/openjdk/jmh/pull/16
https://bugs.openjdk.java.net/browse/CODETOOLS-7902813

Но это же выдуманный пример?

```
byte[] bunn;

@Benchmark
public void goodOldLoop(Blackhole bh) {
  for (int i = 0; i < bunn.length; i++)
    bh.consume(bunn[i]);
}
```

Не такой уж и выдуманный ;)

```
E[] elements;

@Benchmark @Override
public void goodOldLoop forEach(Blackhole Consumer<E> consumer) {
  for (int i = 0; i < elements.length; i++) {
    consumer.accept(elements[i]);
  }
}
```

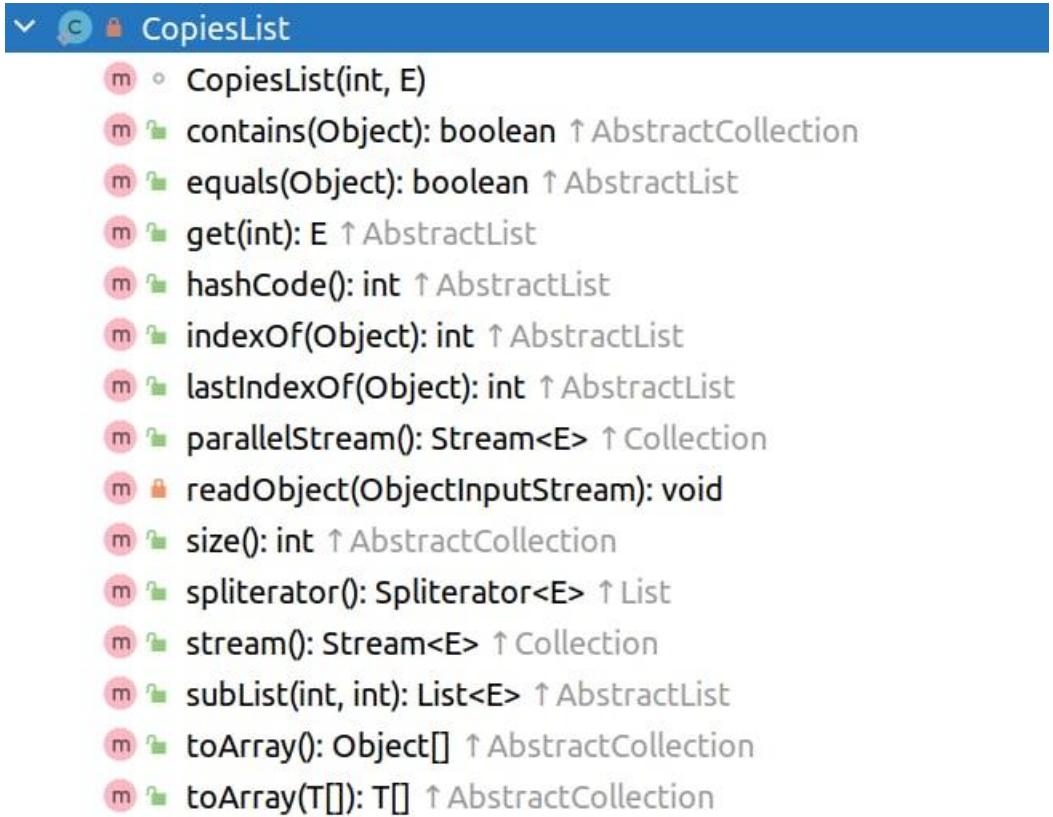
В 99/100 разницы нет, но есть 1/100, поэтому

```
E[] elements;

@Override
public void forEach(Consumer<? super E> consumer) {
  for (E e : elements) {
    consumer.accept(e);
  }
}

// см. ArrayList::forEach
       Arrays.asList::forEach
       CopyOnWriteArrayList::forEach
       ArrayDeque::forEach
```

java.util.Collections::nCopies

```
class CopiesList<E> extends AbstractList<E> {
  final int n;
  final E element;
}
```

java.util.Collections::nCopies



java.util.Collections::nCopies

```
class CopiesList<E> extends AbstractList<E> {
  final int n;
  final E element;

  @Override
  public void forEach(Consumer<? super E> action) {


    for (int i = 0; i < this.n; i++)
      action.accept(this.element);
  }
}
```

http://mail.openjdk.java.net/pipermail/core-libs-dev/2018-December/057336.html

java.util.Collections::nCopies

```
class CopiesList<E> extends AbstractList<E> {
  final int n;
  final E element;

  @Override
  public void forEach(Consumer<? super E> action) {
    int n = this.n;
    E e = this.element;
    for (int i = 0; i < n; i++)
      action.accept(e);
  }
}
```

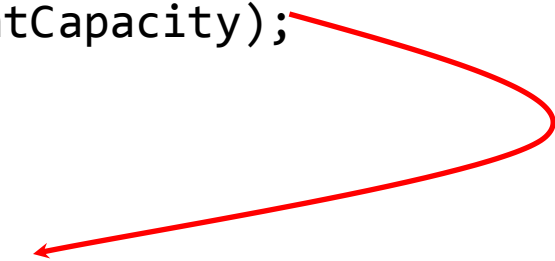
http://mail.openjdk.java.net/pipermail/core-libs-dev/2018-December/057336.html

Всё ещё неубедительно?

```
// org.springframework.util.ConcurrentReferenceHashMap

public ConcurrentReferenceHashMap(/* */) {
 this.segments = (Segment[]) Array.newInstance(Segment.class, size);
 for (int i = 0; i < this.segments.length; i++)
   this.segments[i] = new Segment(roundedUpSegmentCapacity);
}
```

Волатильность снова внутри цикла

```
// org.springframework.util.ConcurrentReferenceHashMap

public ConcurrentReferenceHashMap(/* */) {
 this.segments = (Segment[]) Array.newInstance(Segment.class, size);
 for (int i = 0; i < this.segments.length; i++)
   this.segments[i] = new Segment(roundedUpSegmentCapacity);
}

class Segment extends ReentrantLock {
  private volatile Reference<K, V>[] references;
  public Segment(int initialCapacity) {
    this.references = createReferenceArray(this.initialSize);
  }
}
```



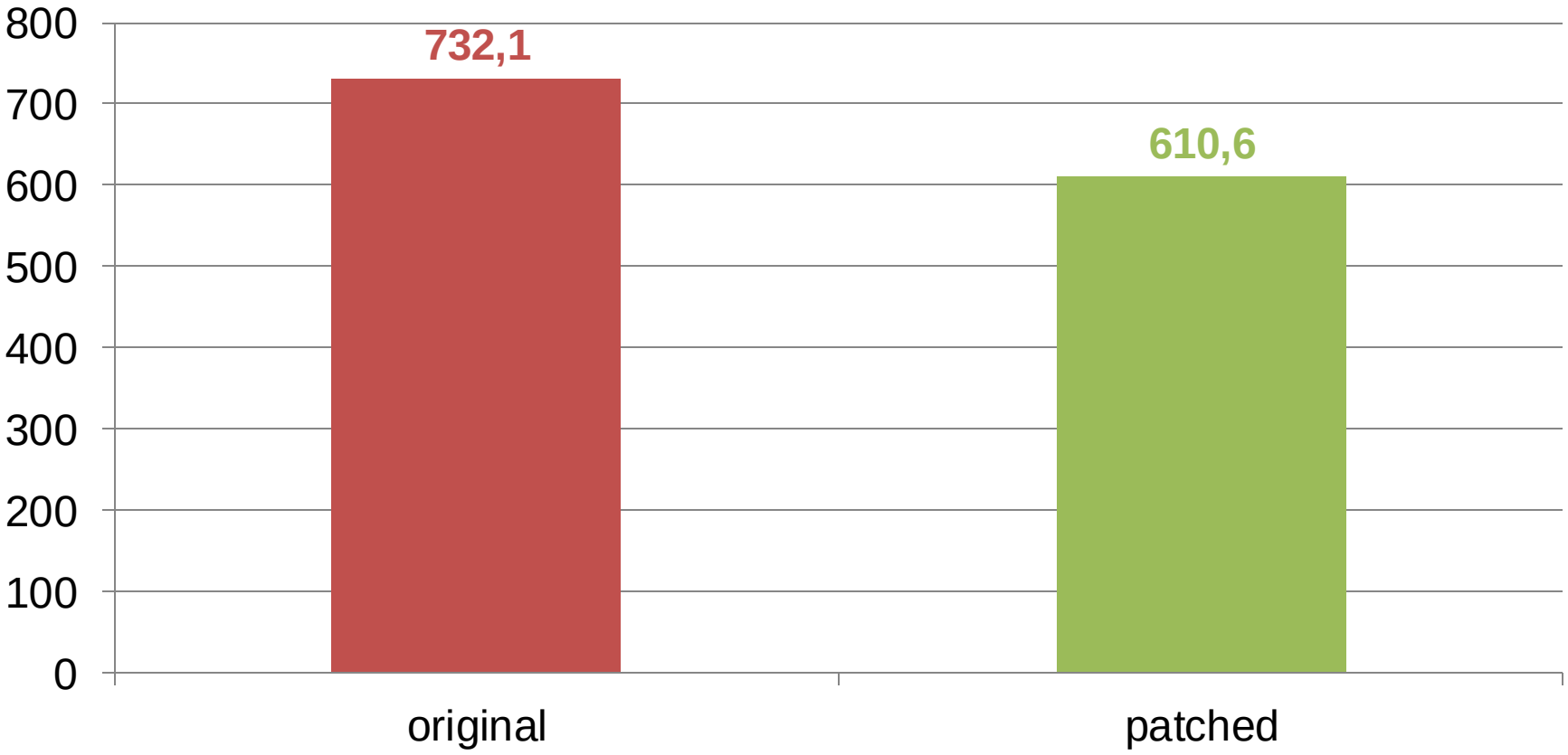
Видишь суслика?

```
@OutputTimeUnit(TimeUnit.NANOSECONDS)
@BenchmarkMode(value = Mode.AverageTime)
public class ConcurrentReferenceHashMapBenchmark {

  @Benchmark
  public Object instantiate() {
    return new ConcurrentReferenceHashMap<>();
  }

}
```

Видишь суслика? – А он есть.

```
// org.springframework.util.ConcurrentReferenceHashMap

class Segment extends ReentrantLock {
  private ~~volatile~~ Reference<K, V>[] references;
  public Segment(int initialCapacity) {
    this.references = createReferenceArray(this.initialSize);
  }
}
```

CRHMBenchmark, время в нс (Java 8)



Было - стало

```
this.segments = (Segment[]) Array.newInstance(Segment.class, size);
for (int i = 0; i < this.segments.length; i++)
  this.segments[i] = new Segment(roundedUpSegmentCapacity);
```

```
var segments = (Segment[]) Array.newInstance(Segment.class, size);
for (int i = 0; i < segments.length; i++)
  segments[i] = new Segment(roundedUpSegmentCapacity);
this.segments = segments;
```

Для внеурочного чтения

Основная статья

https://habr.com/post/432824

Изменения

https://github.com/spring-projects/spring-framework/pull/2051

Код поиграться

https://github.com/stsypanov/concurrent-ref-hash-map-example

Таким образом

- бенчмарк показал существенную разницу между for-i и for-each

Таким образом

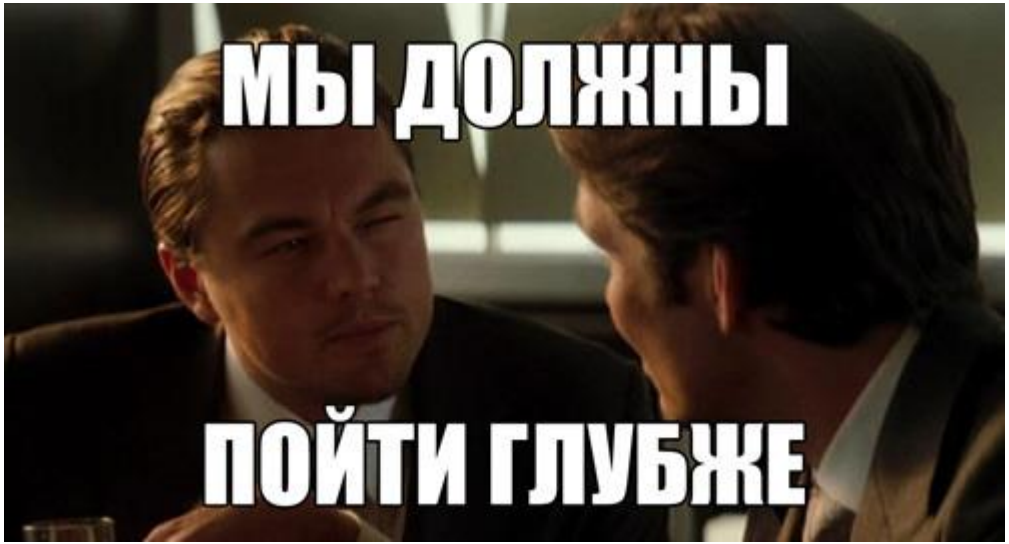
- бенчмарк показал существенную разницу между for-i и for-each
- нюанс: причина в инфраструктуре JMH

Таким образом

- бенчмарк показал существенную разницу между for-i и for-each
- нюанс: причина в инфраструктуре JMH
- способ перебора в 999/1000 не влияет на производительность

Таким образом

- бенчмарк показал существенную разницу между for-i и for-each
- нюанс: причина в инфраструктуре JMH
- способ перебора в 999/1000 не влияет на производительность
- нюанс: есть редкие случаи, когда всё же влияет



Вывод для рядового разработчика прямиком из JMH

REMEMBER: **The numbers below are just data.** To gain reusable insights, you need to follow up on why the numbers are the way they are. Use profilers (see -prof, -lprof), design factorial experiments, perform baseline and negative tests that provide experimental control, make sure the benchmarking environment is safe on JVM/OS/HW level, ask for reviews from the domain experts. **Do not assume the numbers tell you what you want them to tell.**

Помните, что цифры – это просто данные.
Не рассчитывайте, что они скажут вам то, что вы ожидаете услышать.
Цифры – ничто, понимание – всё.

Нюансы исполнения

Несложный пример

```
org.springframework.util.StringUtils.uriDecode(String, Charset)
```

Несложный пример

```
org.springframework.util.StringUtils.uriDecode(String, Charset)

https%3A%2F%2Fru.wikipedia.org%2Fwiki%2F%D0%9E%D1%80%D0%B3%D0%B0%D0
%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D0%9E%D0%B1%D1%8A%D0%B5%D0
%B4%D0%B8%D0%BD%D1%91%D0%BD%D0%BD%D1%8B%D1%85_%D0%9D%D0%B0%D1%86%D0
%B8%D0%B9
```

Несложный пример

```
org.springframework.util.StringUtils.uriDecode(String, Charset)

https%3A%2F%2Fru.wikipedia.org%2Fwiki%2F%D0%9E%D1%80%D0%B3%D0%B0%D0
%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D0%9E%D0%B1%D1%8A%D0%B5%D0
%B4%D0%B8%D0%BD%D1%91%D0%BD%D0%BD%D1%8B%D1%85_%D0%9D%D0%B0%D1%86%D0
%B8%D0%B9


После StringUtils.uriDecode()

https://ru.wikipedia.org/wiki/Организация_Объединённых_Наций
```

Под капотом

```
public static String uriDecode(String source, Charset charset) {
 // ...
 ByteArrayOutputStream baos = new ByteArrayOutputStream(length);
 // ...
 return changed ? new String(baos.toByteArray(), charset) : source;
}
```

Под капотом

```
public static String uriDecode(String source, Charset charset) {

 // ...

 ByteArrayOutputStream baos = new ByteArrayOutputStream(length);

 // ...

 return changed ? new String(baos.toByteArray(), charset) : source;
}
```

Под капотом

```
public synchronized byte[] toByteArray() {
 return Arrays.copyOf(buf, count);
}
```

Под капотом

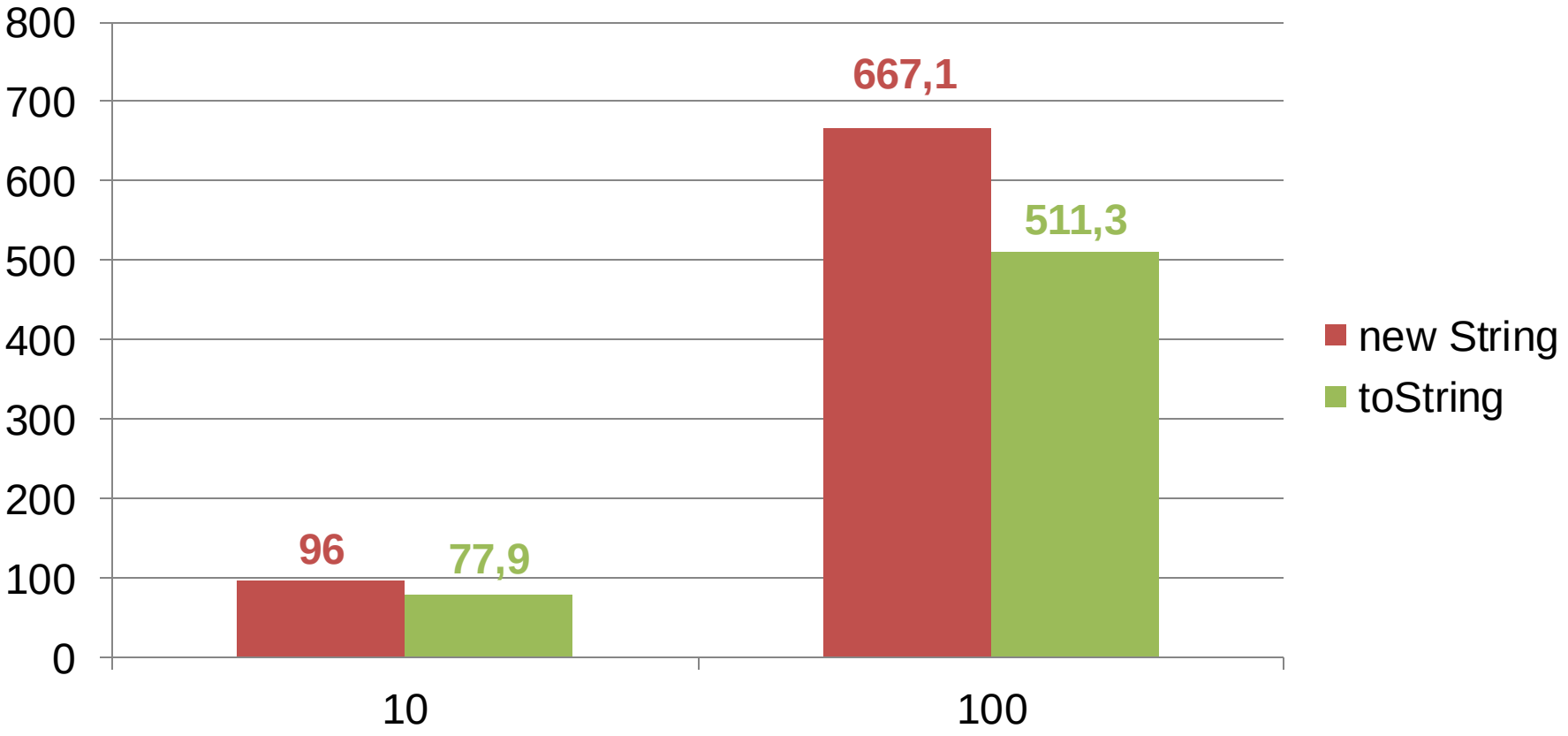
```
public synchronized byte[] toByteArray() {
 return Arrays.copyOf(buf, count);
}
```

```
public synchronized String toString(Charset charset) {
    return new String(buf, 0, count, charset);
}
```

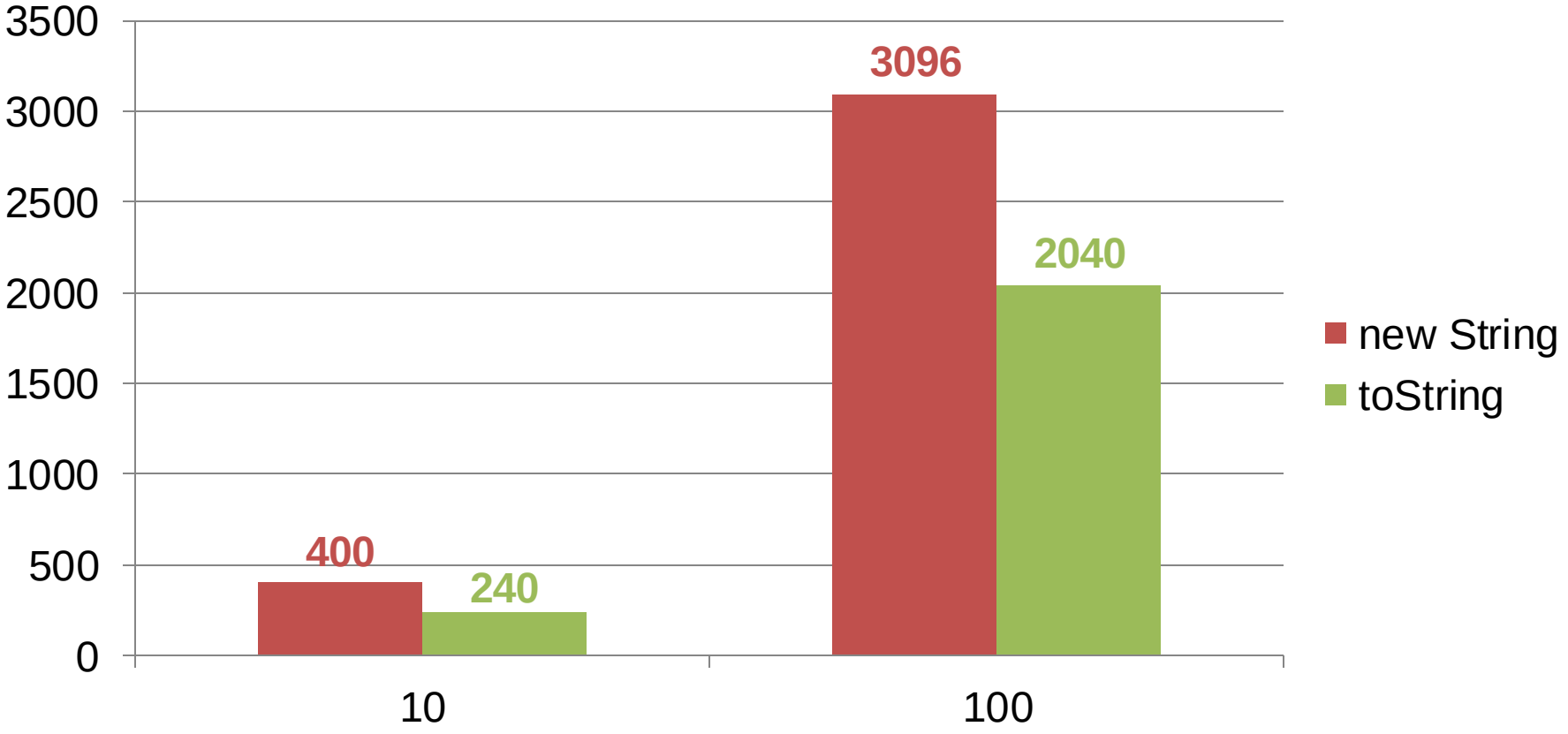
Что быстрее?

```
ByteArrayOutputStream baos;
Charset charset = Charset.defaultCharset();

@Benchmark
public String newString() {
  return new String(baos.toByteArray(), charset);
}

@Benchmark
public String toString() {
  return baos.toString(charset);
}
```

Проверим (Java 8, время в нс)



Проверим (Java 8, память в байтах)

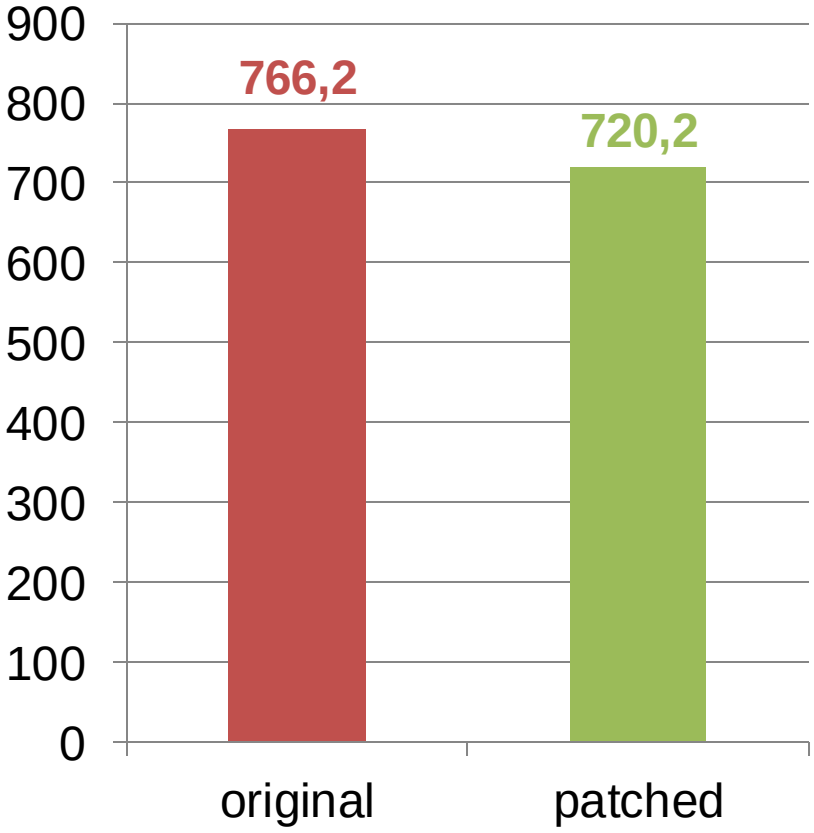


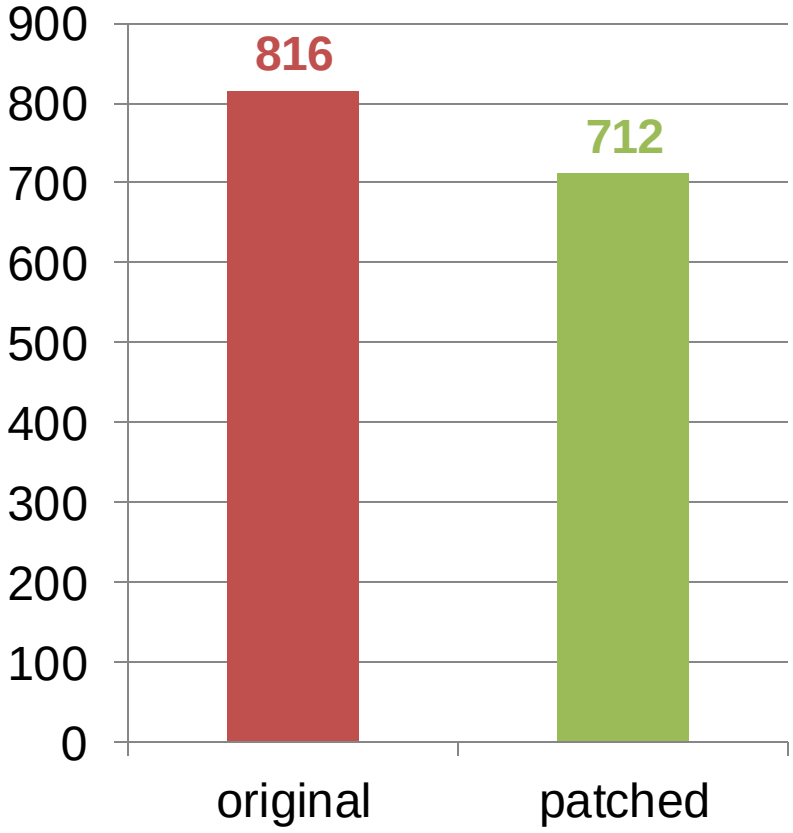
Проверим StringUtils.uriDecode() (Java 8)

```
String encoded =
"https%3A%2F%2Fru.wikipedia.org%2Fwiki%2F%D0%9E%D1%80%D0%B3%D0%B0%D
0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D0%9E%D0%B1%D1%8A%D0%B5%D
0%B4%D0%B8%D0%BD%D1%91%D0%BD%D0%BD%D1%8B%D1%85_%D0%9D%D0%B0%D1%86%D
0%B8%D0%B9";

@Benchmark
public String uriDecode() {
  return StringUtils.uriDecode(encoded);
}
```

https://ru.wikipedia.org/wiki/Организация_Объединённых_Наций

Слева время в нс, справа память в байтах

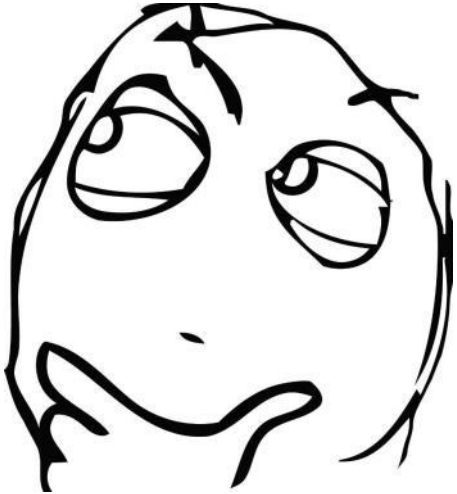




Пусть небольшой, но толк есть

https://github.com/spring-projects/spring-framework/pull/24805

Там где декодер, там и энкодер



java.net.URLEncoder

```
public static String encode(String s, Charset charset) {

  // ...

  CharArrayWriter charArrayWriter = new CharArrayWriter();

  // ...

  String str = new String(charArrayWriter.toCharArray());

  // ...
}
```

java.io.CharArrayWriter

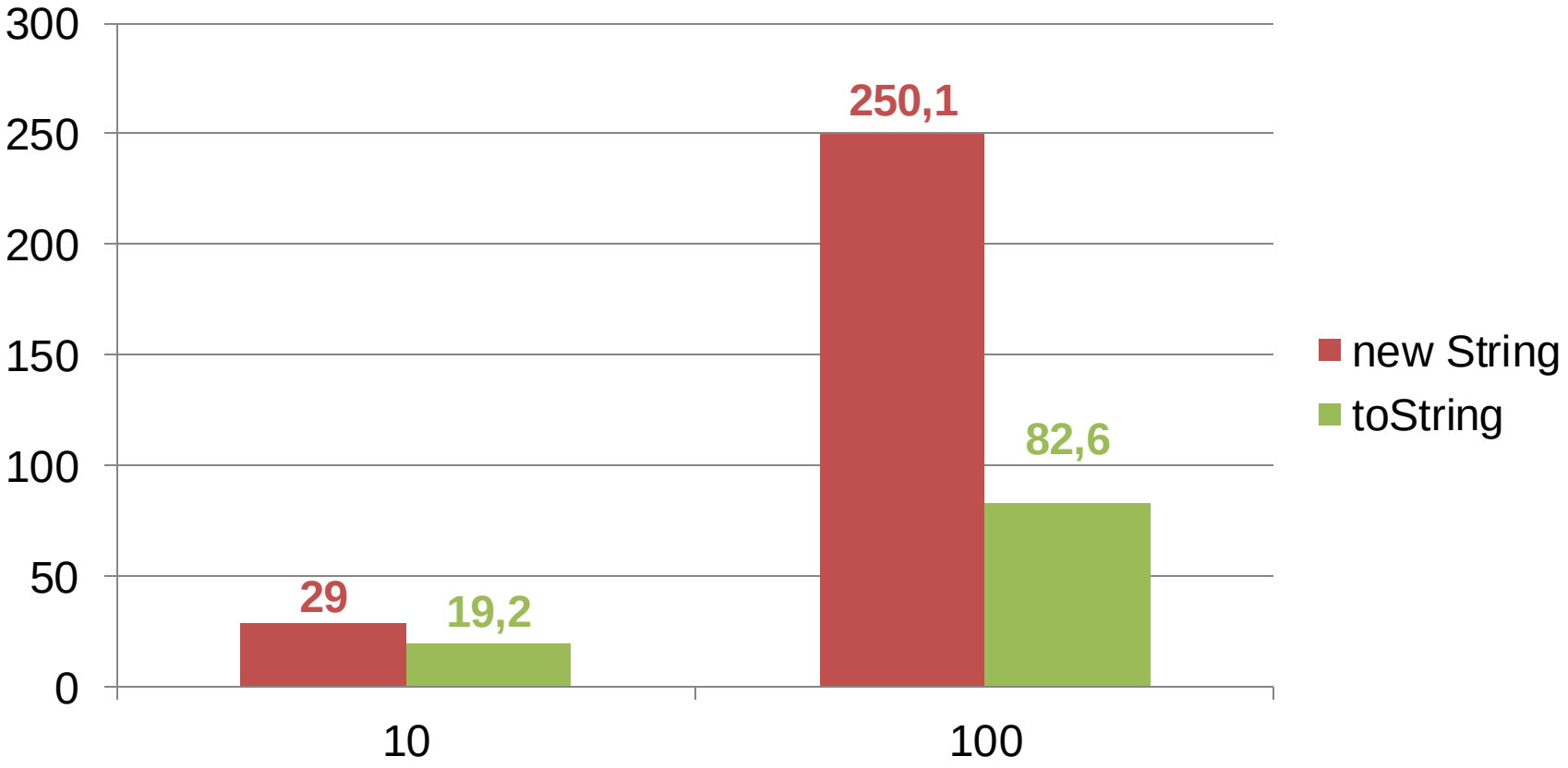
```
public char[] toCharArray() {
  synchronized (lock) {
    return Arrays.copyOf(buf, count);
  }
}
```

```
public String toString() {
  synchronized (lock) {
    return new String(buf, 0, count);
  }
}
```

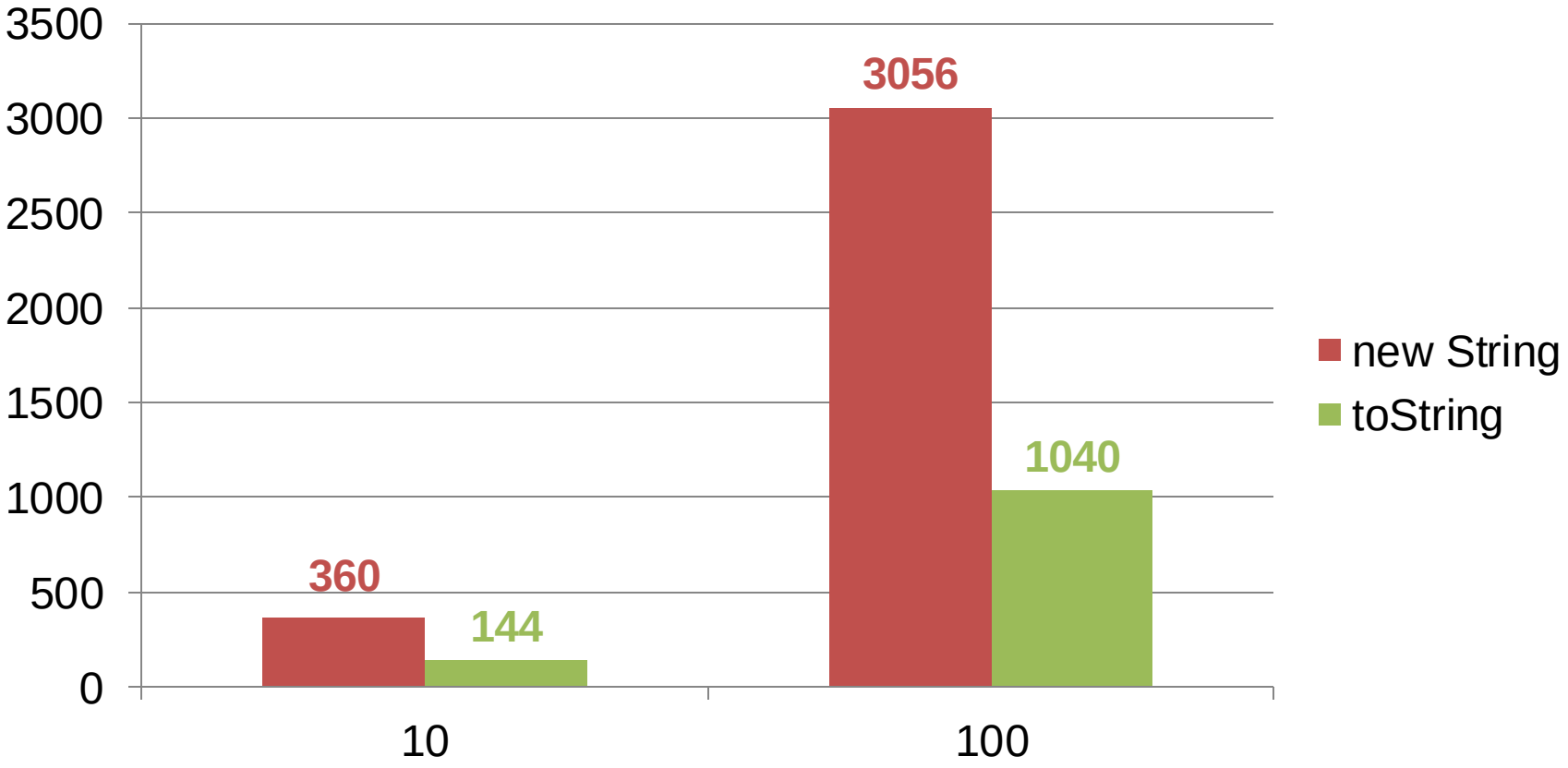
Сравним (Java 11)

```
CharArrayWriter writer;

@Benchmark
public String toString(Data data) {
  return data.writer.toString();
}

@Benchmark
public String toCharArray(Data data) {
  return new String(data.writer.toCharArray());
}
```

CharArrayWriterBenchmark, время в нс



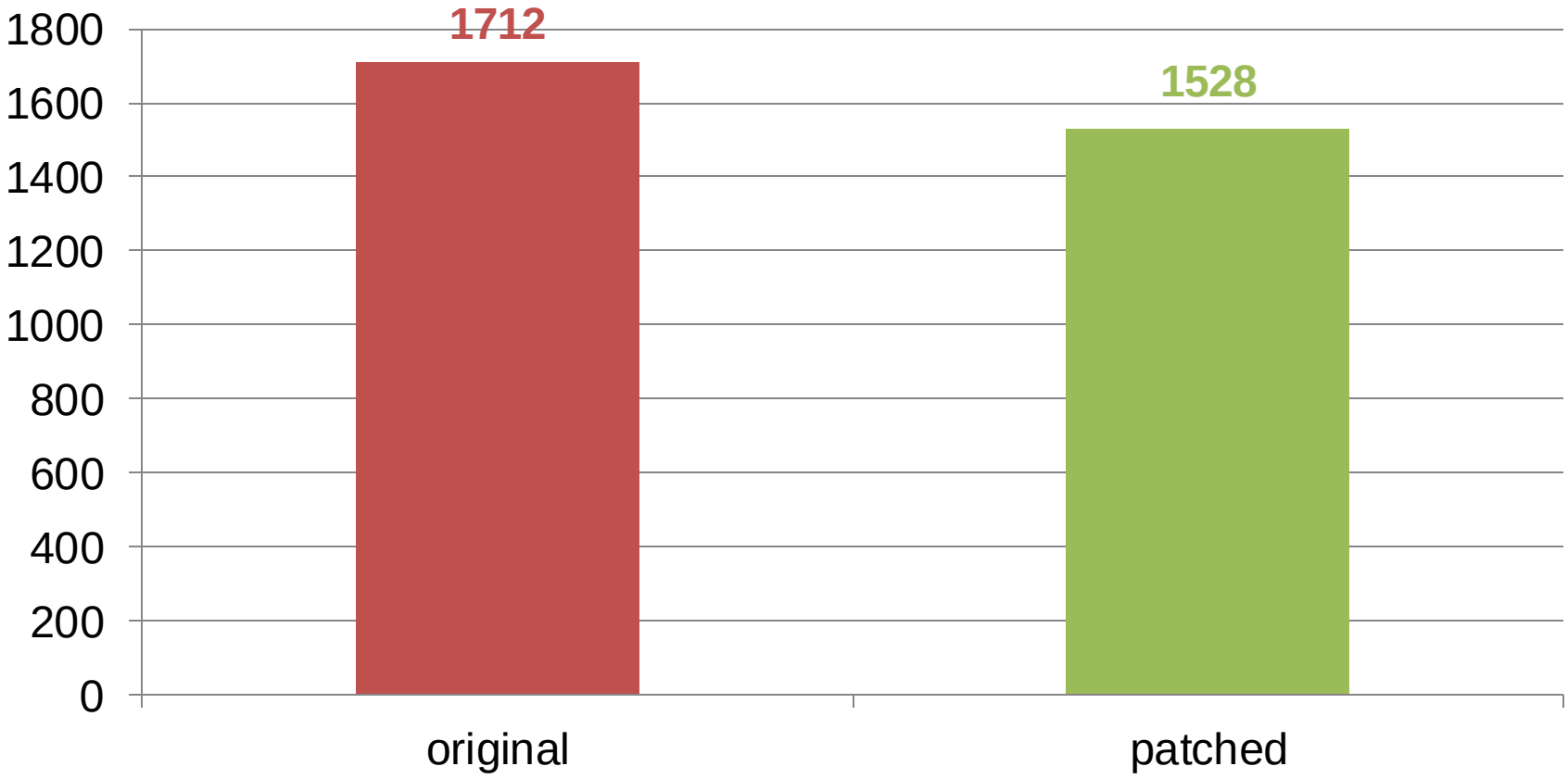
CharArrayWriterBenchmark, память в байтах



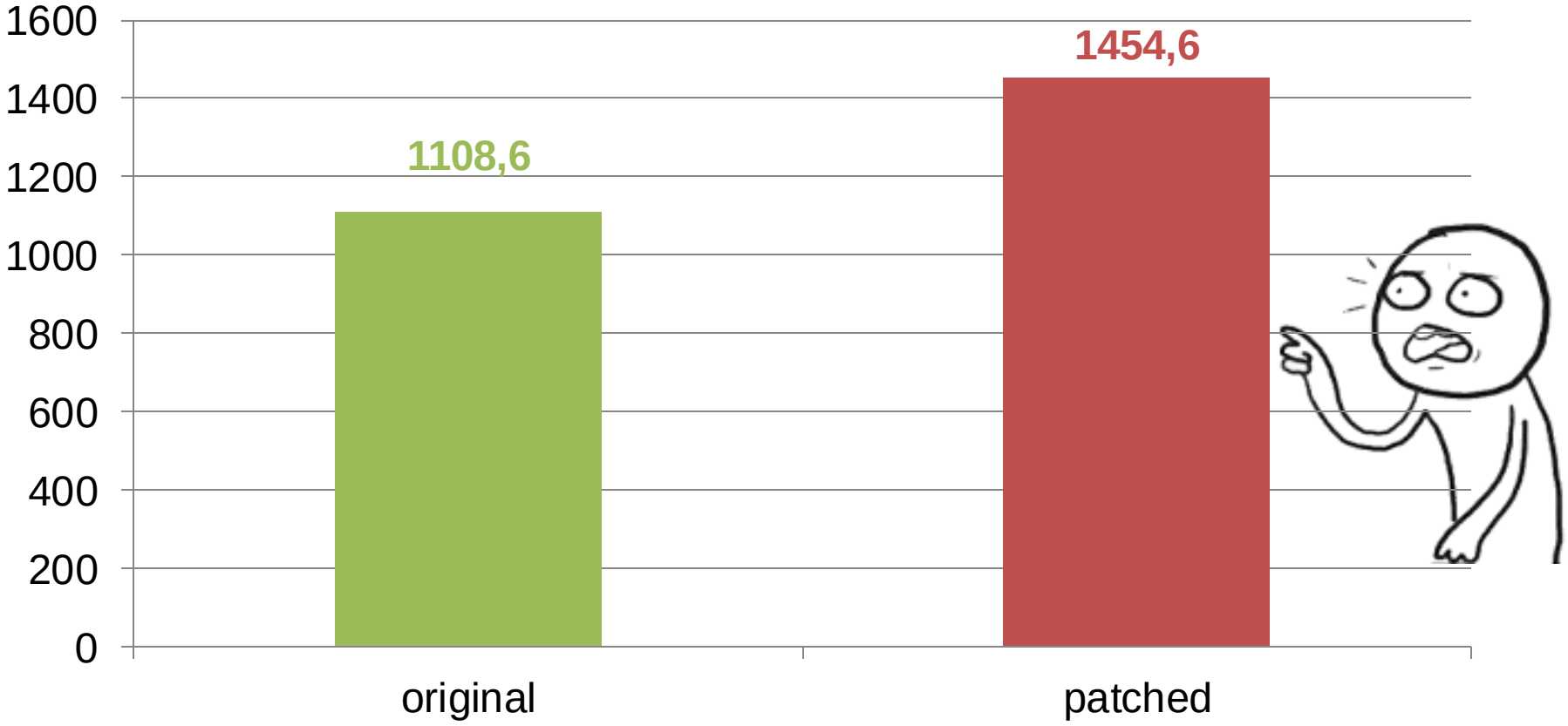
UrlEncoderBenchmark

```
Charset charset = Charset.defaultCharset();

String utf8Url =
"https://ru.wikipedia.org/wiki/Организация_Объединённых_Наций";

@Benchmark
public String encodeUtf8() {
  return URLEncoder.encode(utf8Url, charset);
}
```

Java 15, память в байтах



Но внезапно (Java 15, время в нс)



Что под крышечкой?

```
1)
public String(char value[]) {
  this(value, 0, value.length, null);
}

2)
public String(char[] value, int offset, int count) {
  this(value, offset, count, rangeCheck(value, offset, count));
}

        String(char[] value, int off, int len, Void sig) {...}
```

-XX:+PrintInlining до

```
 @ 186  j.i.CharArrayWriter::flush (1 bytes)   inline (hot)
!m @ 195  j.i.CharArrayWriter::toCharArray (26 bytes)   inline (hot)
     @ 15  j.u.Arrays::copyOf (19 bytes)   inline (hot)
       @ 11  j.l.Math::min (11 bytes)   (intrinsic)
       @ 14  j.l.System::arraycopy (0 bytes)   (intrinsic)
   @ 198  j.l.String::<init> (10 bytes)   inline (hot)
     @ 6  j.l.String::<init> (74 bytes)   inline (hot)
       @ 1  j.l.Object::<init> (1 bytes)   inline (hot)
       @ 36  j.l.StringUTF16::compress (20 bytes)   inline (hot)
         @ 9  j.l.StringUTF16::compress (50 bytes)   (intrinsic)
       @ 67  j.l.StringUTF16::toBytes (34 bytes)   (intrinsic)
```

-XX:+PrintInlining после

```
 @ 186 j.i.CharArrayWriter::flush (1 bytes)  inline (hot)
!m @ 191 j.i.CharArrayWriter::toString (31 bytes) already compiled into a big
                                                   method
   @ 199 j.l.String::getBytes (25 bytes)   inline (hot)
    @ 14 j.l.String::coder (15 bytes)   inline (hot)
!   @ 21 j.l.StringCoding::encode (324 bytes)   inline (hot)
     @ 10 j.l.StringCoding::encodeUTF8 (132 bytes)   inline (hot)
      @ 7 j.l.StringCoding::encodeUTF8_UTF16 (369 bytes) hot method too big
      @ 15 j.l.StringCoding::hasNegatives (25 bytes)   (intrinsic)
      @ 24 j.u.Arrays::copyOf (19 bytes)   inline (hot)
       @ 11 j.l.Math::min (11 bytes)   (intrinsic)
       @ 14 j.l.System::arraycopy (0 bytes)   (intrinsic)
```

-XX:InlineSmallCode

Inline a previously compiled method only if its generated native code size is less than this.

https://www.oracle.com/java/technologies/javase/vmoptions-jsp.html

Встраиванию не подлежат скомпилированные на последнем уровне методы, занимающие более 1000 байт при выключенной многоуровневой компиляции и **2000** байт при включенной

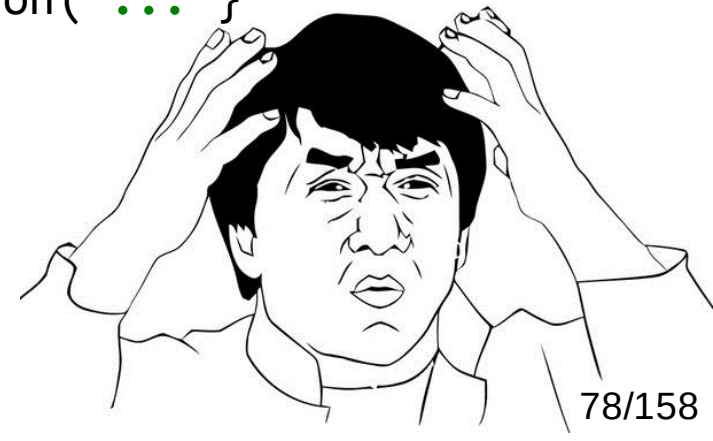
https://habr.com/ru/post/536514/

Нам гадит String.rangeCheck()

```
1)
public String(char value[]) {
  this(value, 0, value.length, null);  ---------------------------
}                                                                  |
                                                                   |
2)                                                                 |
public String(char[] value, int offset, int count) {              |
  this(value, offset, count, rangeCheck(value, offset, count)); --|
}                                                                  |
                                                                   |
                                                                   |
            String(char[] value, int off, int len, Void sig) {...}
```

Нам гадит String.rangeCheck()

```
private static Void rangeCheck(char[] value, int off, int count) {
  checkBoundsOffCount(off, count, value.length);
  return null;
}

static void checkBoundsOffCount(int off, int count, int length) {
  if (off < 0 || count < 0 || off > length - count) {
    throw new StringIndexOutOfBoundsException("..."}
}
```



Однако, в Java 16 теперь так

```
public static String encode(String s, Charset charset) {
  // ...

  CharArrayWriter charArrayWriter = new CharArrayWriter();

  // ...
  String str = charArrayWriter.toString();
}
```

https://github.com/openjdk/jdk/pull/1598

https://bugs.openjdk.java.net/browse/JDK-8259699

Как мне это удалось?

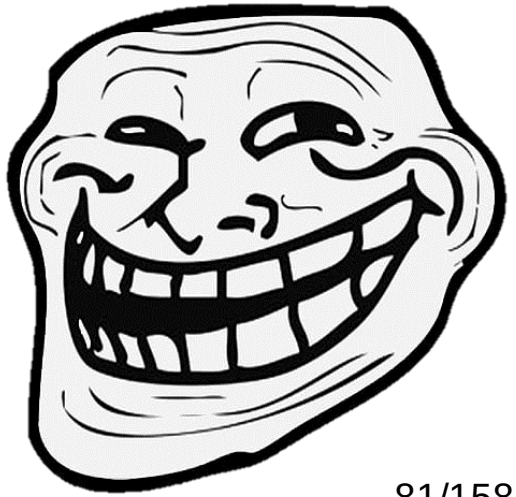
Как мне это удалось?

16 октября 2020 создан

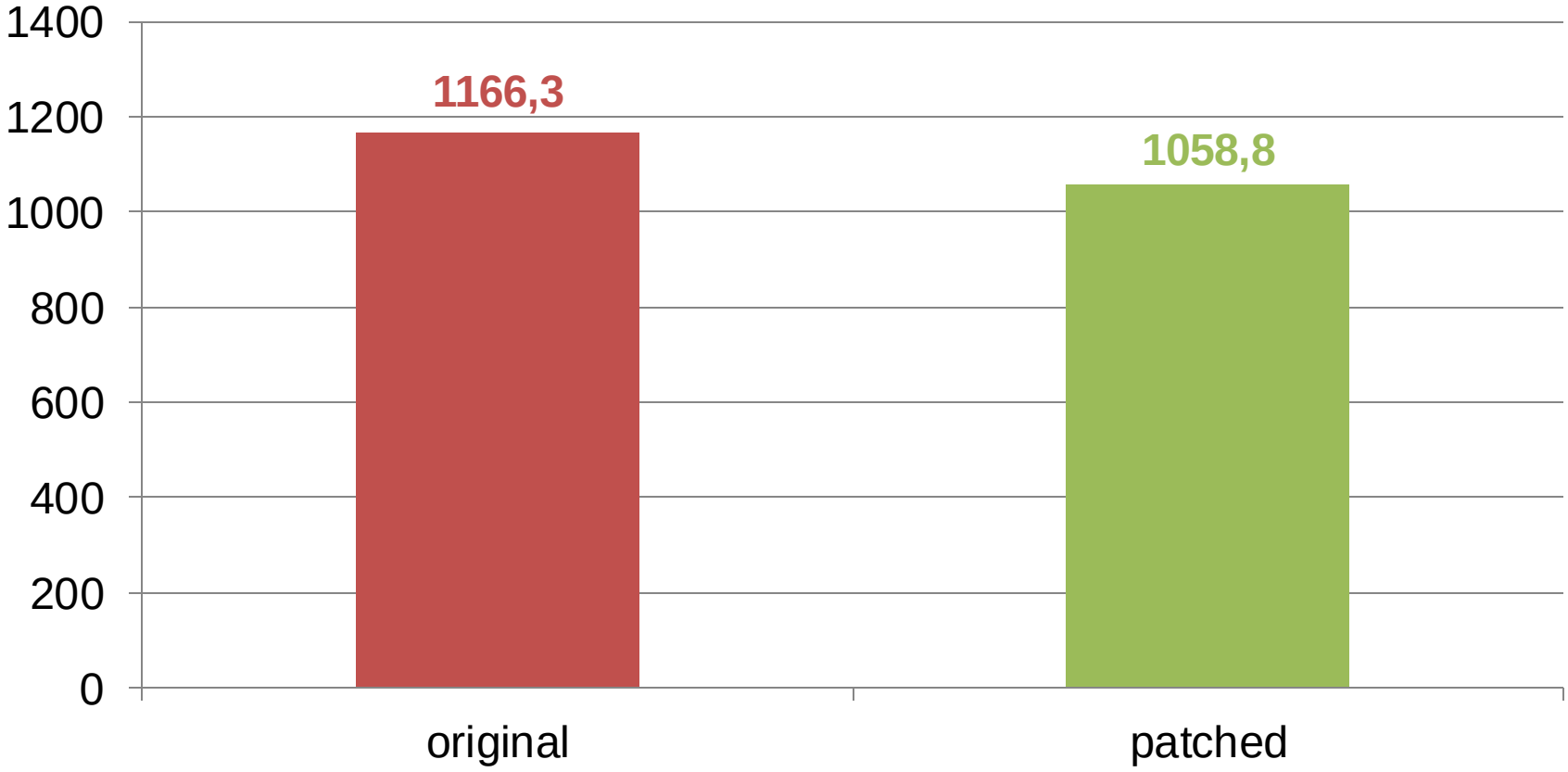
https://github.com/openjdk/jdk/pull/705

Increase InlineSmallCode default from 2000 to 2500 for x64

https://bugs.openjdk.java.net/browse/JDK-8254913



В сухом остатке (время в нс)



Выводы для рядового разработчика

- улучшения под копирку не всегда работают

Выводы для рядового разработчика

- улучшения под копирку не всегда работают
- замеряйте конечные сценарии

Нюансы реализации

BufferedReaderBenchmark

```
File file;

@Setup
public void setUp() throws Exception {
  var p = "tsypanov/strings/buffered/BufferedReaderBenchmark.class";
  URL url = getClass()
    .getClassLoader()
    .getResource(path);
  file = new File(url.getFile());
}
```

BufferedReaderBenchmark

```
@Benchmark
public void readFromFile(Blackhole bh) throws Exception {
  int value;
  try (var is = new FileInputStream(file))
    while ((value = is.read()) != -1) bh.consume(value);
}
```

BufferedReaderBenchmark

read	2132,4 ± 18,2	us/op
read:·gc.alloc.rate.norm	153,0 ± 0,9	B/op

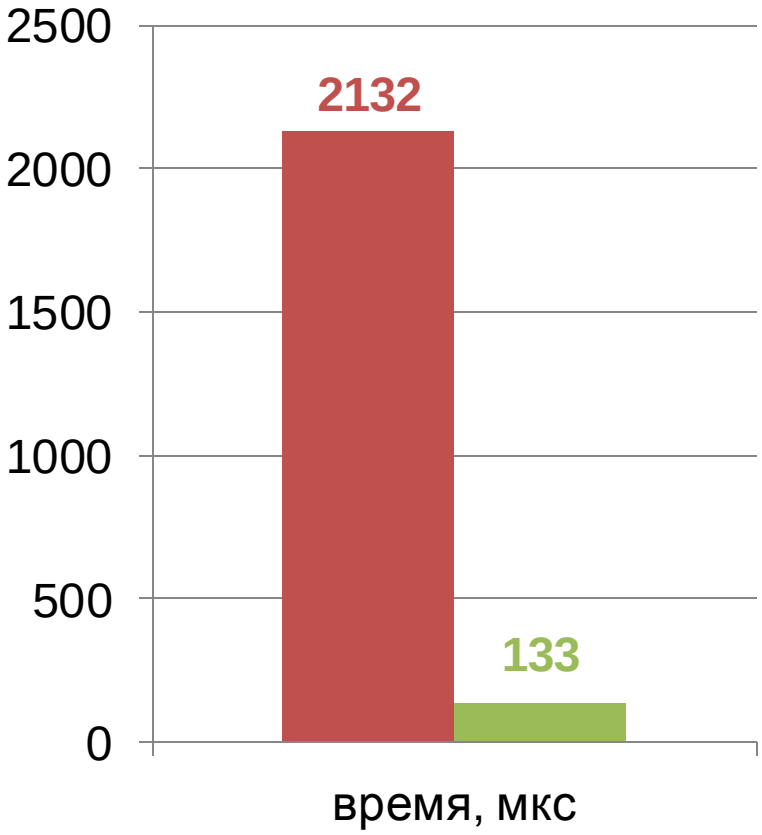
BufferedReaderBenchmark

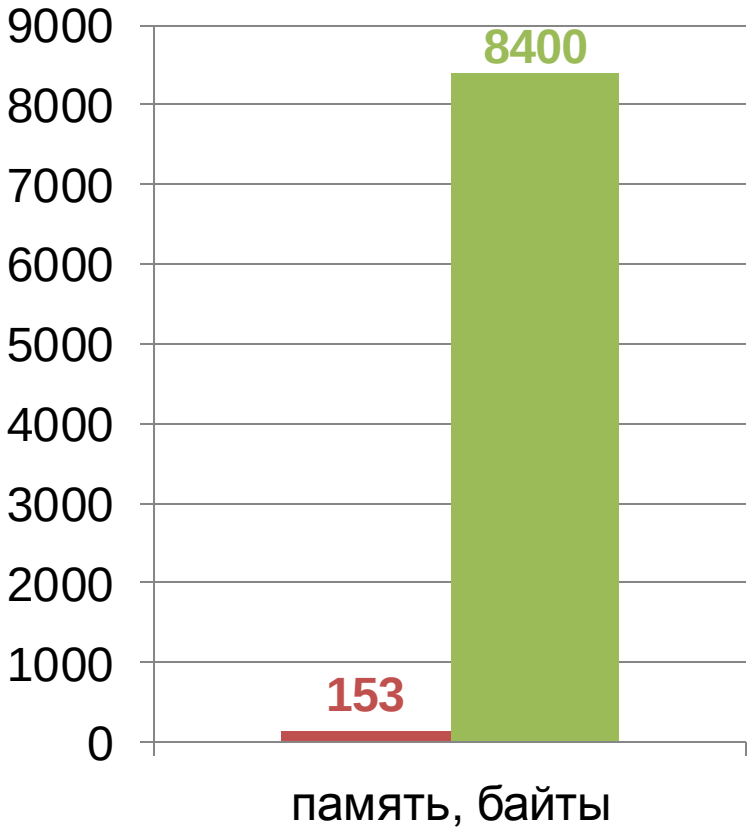
```
@Benchmark
public void readFromFile(Blackhole bh) throws Exception {
  int value;
  try (var is = new FileInputStream(file))
    while ((value = is.read()) != -1) bh.consume(value);
}
```

BufferedReaderBenchmark

```
@Benchmark
public void readFromFile(Blackhole bh) throws Exception {
  int value;
  try (var is = new FileInputStream(file))
    while ((value = is.read()) != -1) bh.consume(value);
}

@Benchmark
public void bufferedReadFromFile(Blackhole bh) throws Exception {
  int value;
  try (var is = new BufferedInputStream(new FileInputStream(file)))
    while ((value = is.read()) != -1) bh.consume(value);
}
```

BufferedReaderBenchmark

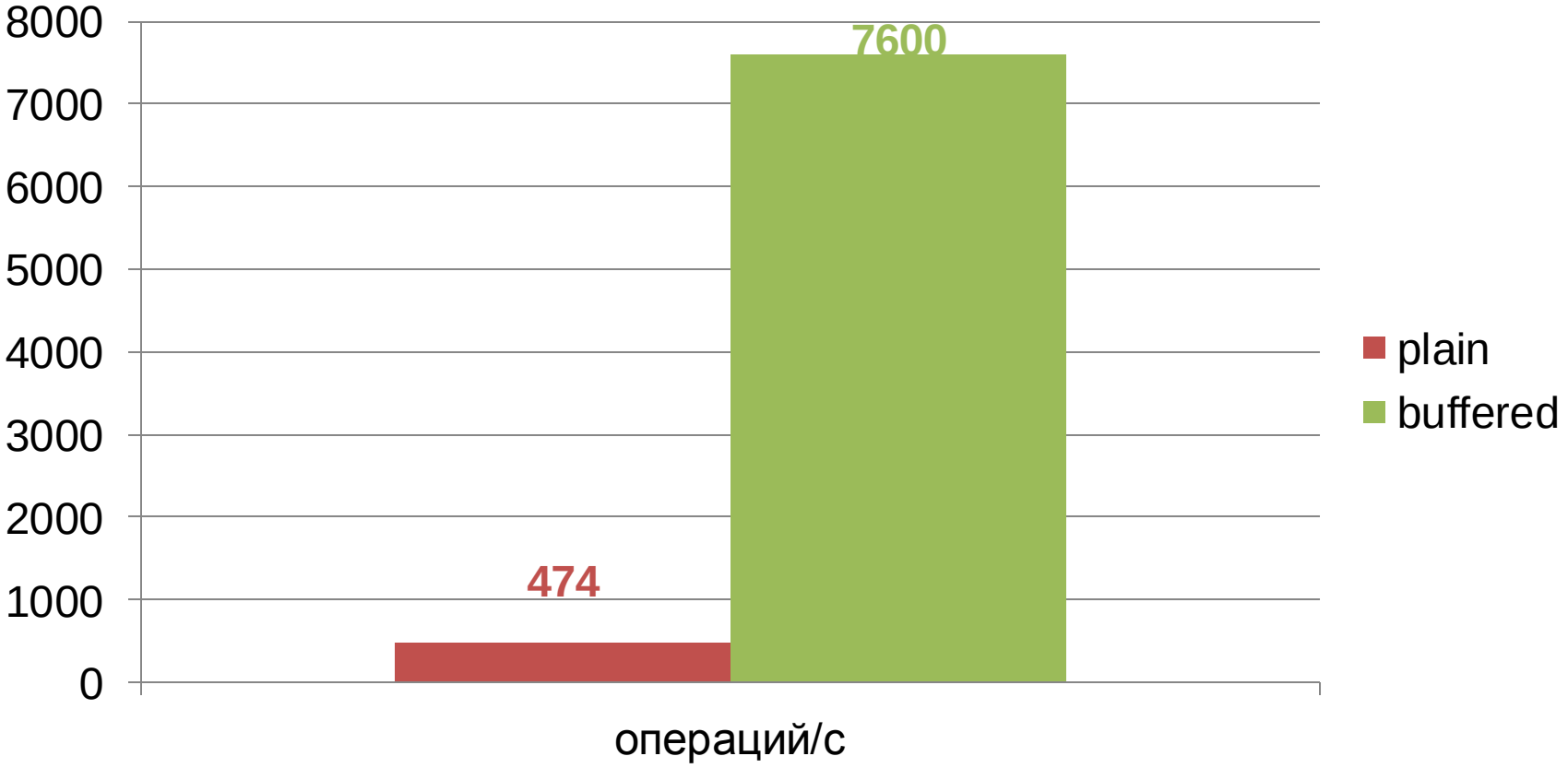




plain

buffered

BufferedReaderBenchmark



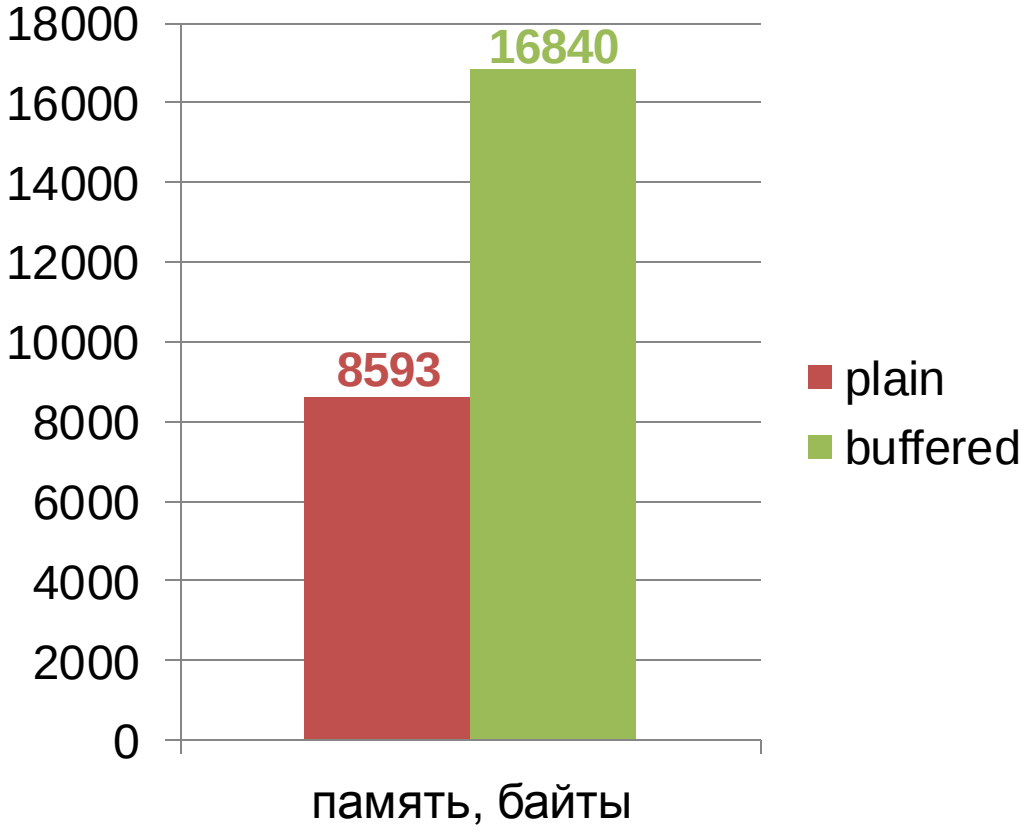
BufferedReaderBenchmark

```
URL url;

@Setup
public void setUp() throws Exception {
  var p = "tsypanov/strings/buffered/BufferedReaderBenchmark.class";
  url = getClass()
         .getClassLoader()
         .getResource(path)
         .toURI()
         .toURL();
}
```

BufferedReaderBenchmark

```
@Benchmark
public void readFromURL(Blackhole bh) throws Exception {
  int value;
  try (var is = url.openStream())
    while ((value = is.read()) != -1) bh.consume(value);
}

@Benchmark
public void readBufferedFromURL(Blackhole bh) throws Exception {
  int value;
  try (var is = new BufferedInputStream(url.openStream())) {
    while ((value = is.read()) != -1) bh.consume(value);
  }
}
```

BufferedReaderBenchmark

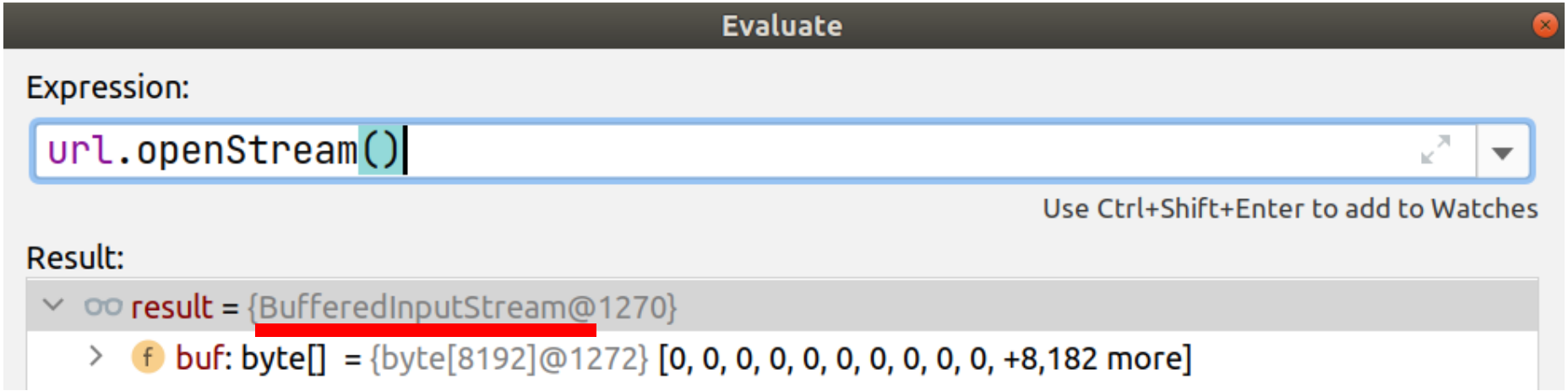


Как же так?

```
try (var is = new BufferedInputStream(url.openStream())) {
  // ...
}
```

Как же так?

```
try (var is = new BufferedInputStream(url.openStream())) {
  // ...
}
```



sun.net.www.protocol.file.FileURLConnection

```
public void connect() throws IOException {
  // ...
  if (isDirectory) {
  // ...
  } else {
    is = new BufferedInputStream(new FileInputStream(filename));
  }
  // ...
}
```

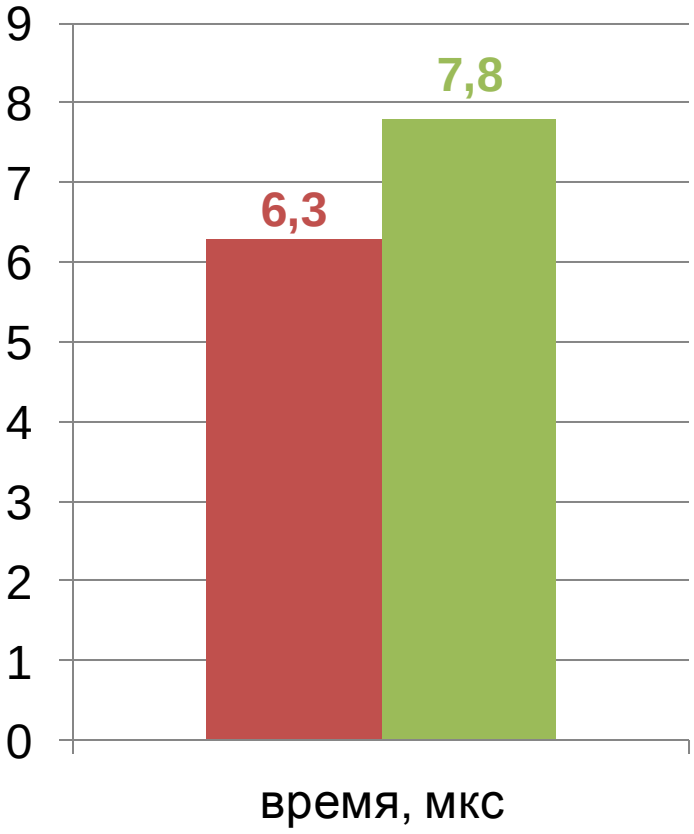
ПМСМ это прокол разработчиков

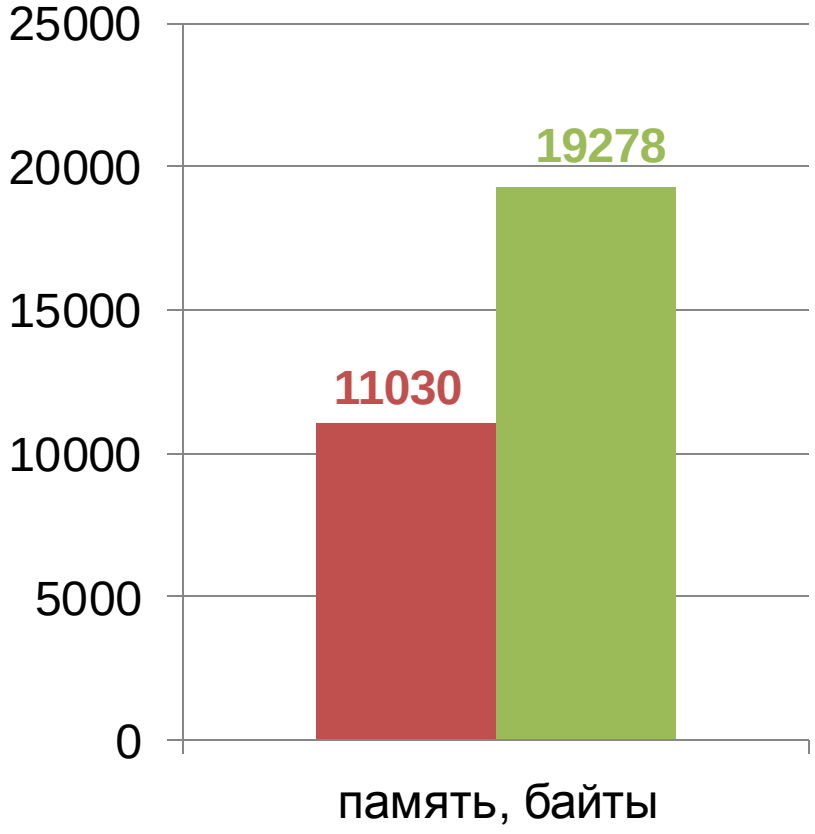
```
public void connect() throws IOException {
  // ...
  if (isDirectory) {
  // ...
  } else {
    is = new BufferedInputStream(new FileInputStream(filename));
  }
  // ...
}
```

FileInputStreamBenchmark

```
String file = "/home/s.tsypanov/.bashrc"; // 4362 байта

@Benchmark
public byte[] readAllBytesFromFile() throws Exception {
  try (var is = new FileInputStream(file))
    return is.readAllBytes();
}

@Benchmark
public byte[] bufferedReadAllBytesFromFile() throws Exception {
  try (var is = new BufferedInputStream(new FileInputStream(file)))
    return is.readAllBytes();
}
```

FileInputStreamBenchmark





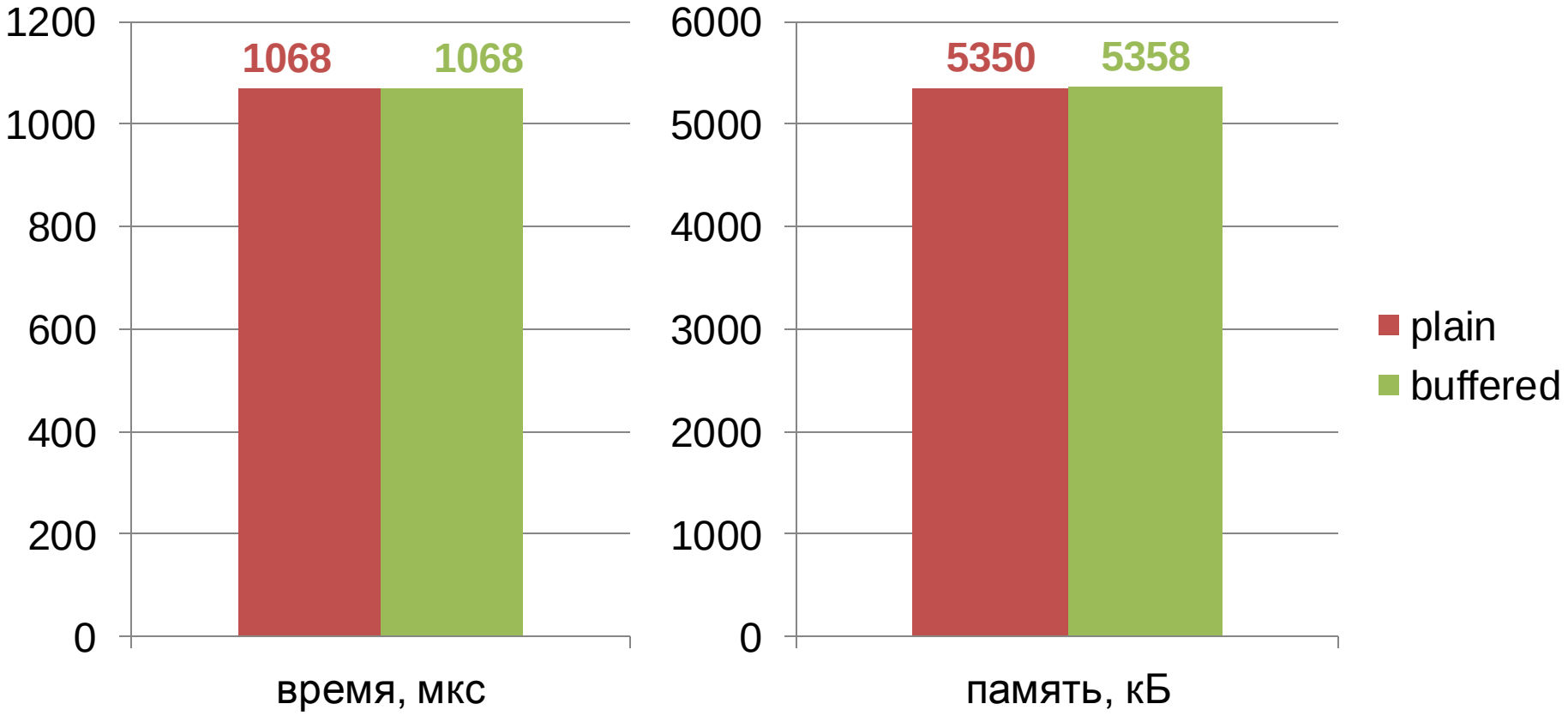
plain

buffered

FileInputStreamBenchmark: большой файл

```
String file = "/home/s.tsypanov/Downloads/IMAG2549.jpg"; // 2,7 Мб

@Benchmark
public byte[] readAllBytesFromURL() throws IOException {
  try (var is = new FileInputStream(file))
    return is.readAllBytes();
}

@Benchmark
public byte[] bufferedReadAllBytesFromURL() throws IOException {
  try (var is = new BufferedInputStream(new FileInputStream(file)))
    return is.readAllBytes();
}
```

FileInputStreamBenchmark: большой файл



InputStream.readAllBytes() -> readNBytes(**int**)

```
List<byte[]> bufs = null;
// ...

byte[] buf = new byte[Math.min(remaining, DEFAULT_BUFFER_SIZE)];
int nread = 0;

// ...
while ((n = read(buf, nread,
        Math.min(buf.length - nread, remaining))) > 0) {
    nread += n;
    remaining -= n;
}
// ...
bufs.add(buf);
```

Внутри j.io.BufferedInputStream то же самое

```
private void fill() throws IOException {

  byte[] buffer = getBufIfOpen();

  // ...

  int n = getInIfOpen().read(buffer, pos, buffer.length - pos);
  if (n > 0)
    count = n + pos;
}
```

Тяжёлое историческое наследие сложно выпилить

```
public void connect() throws IOException {
  // ...
  if (isDirectory) {
  // ...
  } else {
    is = new BufferedInputStream(new FileInputStream(filename));
  }
  // ...
}
```

Тяжёлое историческое наследие иногда выпиливается

```
// jdk.internal.jmod.JmodFile.checkMagic(Path file)

try (InputStream in = Files.newInputStream(file);
  BufferedInputStream bis = new BufferedInputStream(in)) {
  byte[] magic = new byte[4];
  bis.read(magic);
```

Тяжёлое историческое наследие иногда выпиливается

```
// jdk.internal.jmod.JmodFile.checkMagic(Path file)

try (InputStream in = Files.newInputStream(file);
  BufferedInputStream bis = new BufferedInputStream(in)) {
  byte[] magic = new byte[4];
  bis.read(magic);


// sun.net.www.http.HttpClient.available()

var buf = new BufferedInputStream(serverSocket.getInputStream());
int r = buf.read();
```

Тяжёлое историческое наследие иногда выпиливается

```
// sun.awt.image.ByteArrayImageSource

protected ImageDecoder getDecoder() {
  var is = new BufferedInputStream(new ByteArrayInputStream(data,
                                                            offset,
                                                            length));
  return getDecoder(is);
}
```

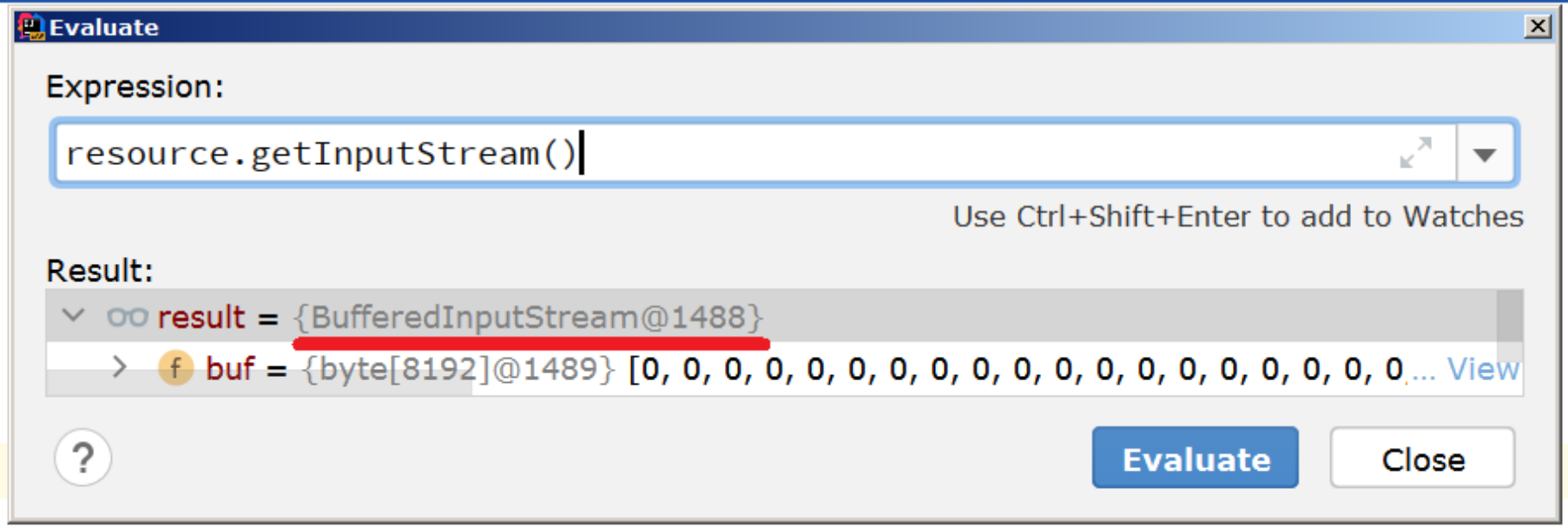
https://github.com/openjdk/jdk/pull/2992/files
https://bugs.openjdk.java.net/browse/JDK-8263599
https://bugs.openjdk.java.net/browse/JDK-8263560

o.s.core.type.classreading.SimpleMetadataReader

```
static ClassReader getClassReader(Resource rsc) throws Exception {

 try (var is = new BufferedInputStream(rsc.getInputStream())) {
  try {
   return new ClassReader(is);
  }
  catch (IllegalArgumentException ex) {
   throw new NestedIOException("..." + rsc, ex);
  }
 }

}
```

o.s.core.type.classreading.SimpleMetadataReader

```
private static ClassReader getClassReader(Resource resource) throws IOException
    try (InputStream is = new BufferedInputStream(resource.getInputStream())) {
```



o.s.core.type.classreading.SimpleMetadataReader

```
static ClassReader getClassReader(Resource rsc) throws Exception {

 try (var is = new BufferedInputStream(rsc.getInputStream())) {
  try {
   return new ClassReader(is);
  }
  catch (IllegalArgumentException ex) {
   throw new NestedIOException("..." + rsc, ex);
  }
 }

}
```

org.springframework.asm.ClassReader

```
byte[] readStream(InputStream inputStream, boolean close) {

  try (ByteArrayOutputStream os = new ByteArrayOutputStream()) {
    byte[] data = new byte[INPUT_STREAM_DATA_CHUNK_SIZE];
    int read;
    while ((read = inputStream.read(data, 0, data.length)) != -1) {
      os.write(data, 0, read);
    }
    os.flush();
    return os.toByteArray();
  }
}
```

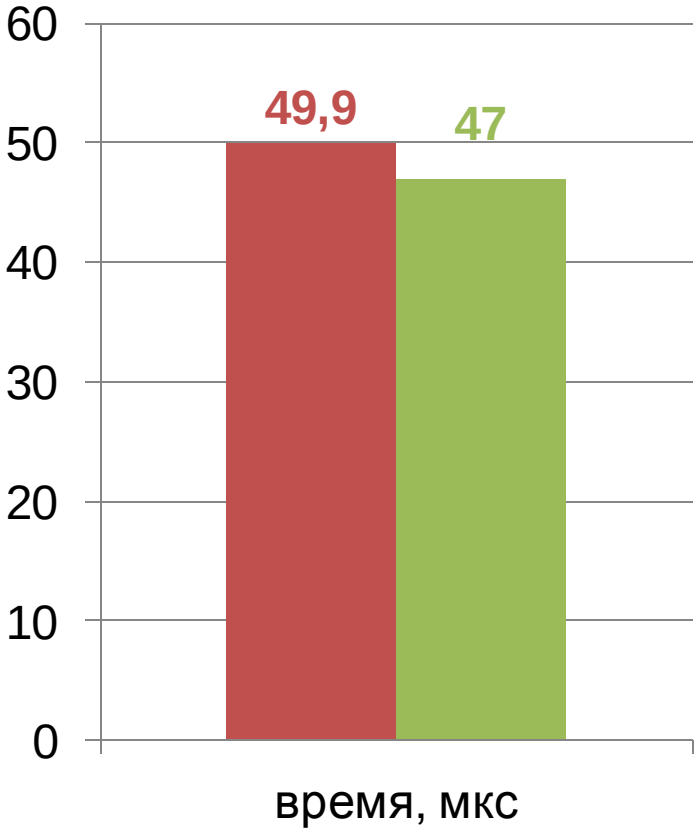
MetadataReaderBenchmark

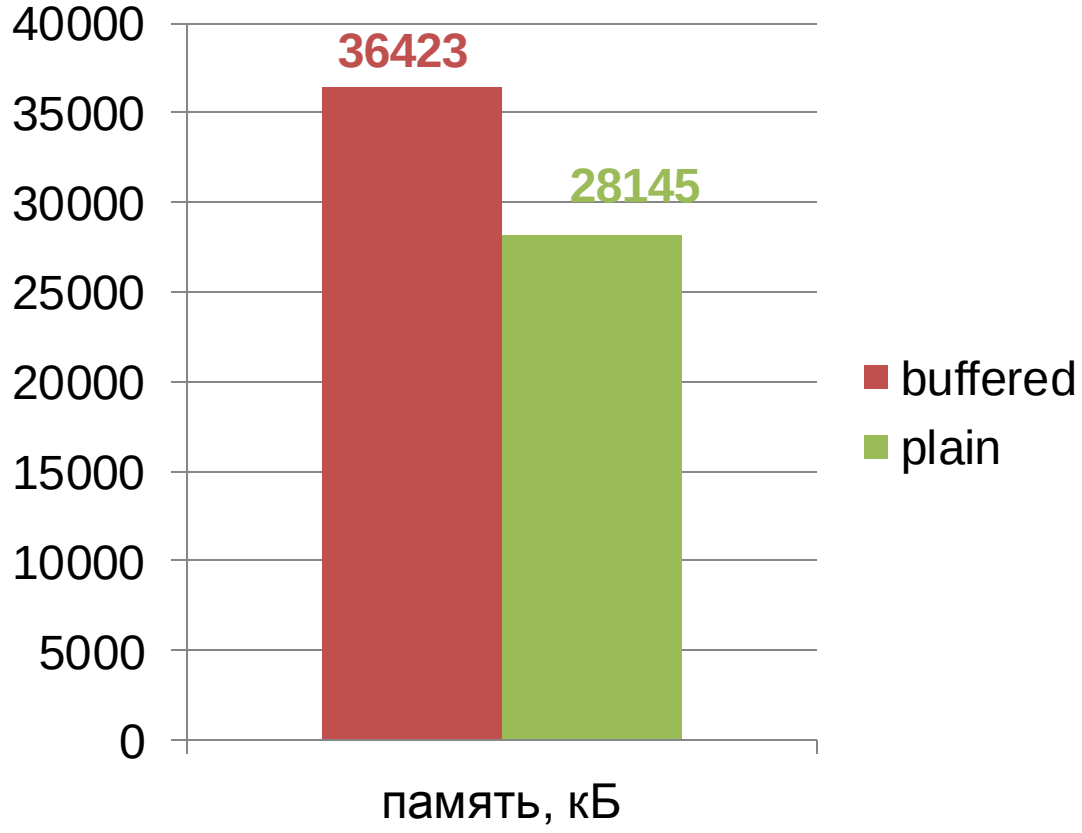
```
MetadataReaderFactory factory = new SimpleMetadataReaderFactory();

@Benchmark
public Object read() throws IOException {
  String name = AnnotatedComponent.class.getName();
  return factory.getMetadataReader(name).getAnnotationMetadata();
}
```

Подопытный кролик для MetadataReaderBenchmark

```
@Component("myName")
@Scope(BeanDefinition.SCOPE_PROTOTYPE)
private static class AnnotatedComponent implements Serializable {
  private final Dependency dep;

  @Autowired
  public AnnotatedComponent(@Qualifier("myColor") Dependency dep) {
    this.dep = dep;
  }

  private static class Dependency {}
}
```

MetadataReaderBenchmark: что по чём





В крупную клетку

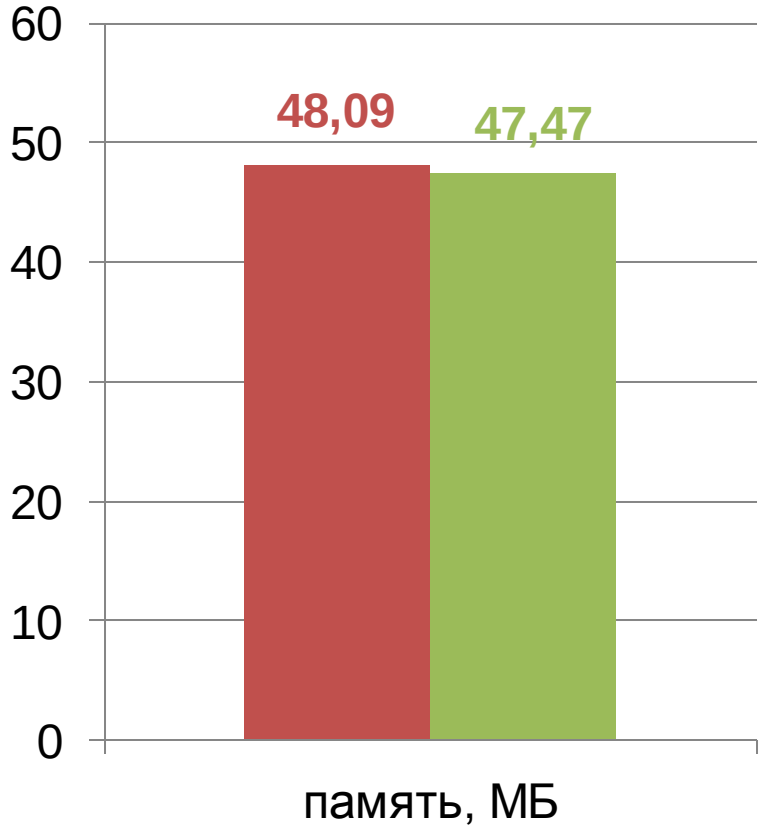
```
@SpringBootApplication
public class BenchmarkApplication {

  public static void main(String[] args) {
    SpringApplication.run(BenchmarkApplication.class, args);
  }

}
```

https://github.com/stsypanov/spring-boot-benchmark

В крупную клетку

```
@BenchmarkMode(Mode.SingleShot)
public class SpringBootApplicationBenchmark {
  private ApplicationContext c;

  @Benchmark
  public Object startUp() {
    return c = SpringApplication.run(BenchmarkApplication.class);
  }

  @TearDown(Level.Invocation)
  public void close() { c.close(); }
}
```

SpringBootApplicationBenchmark

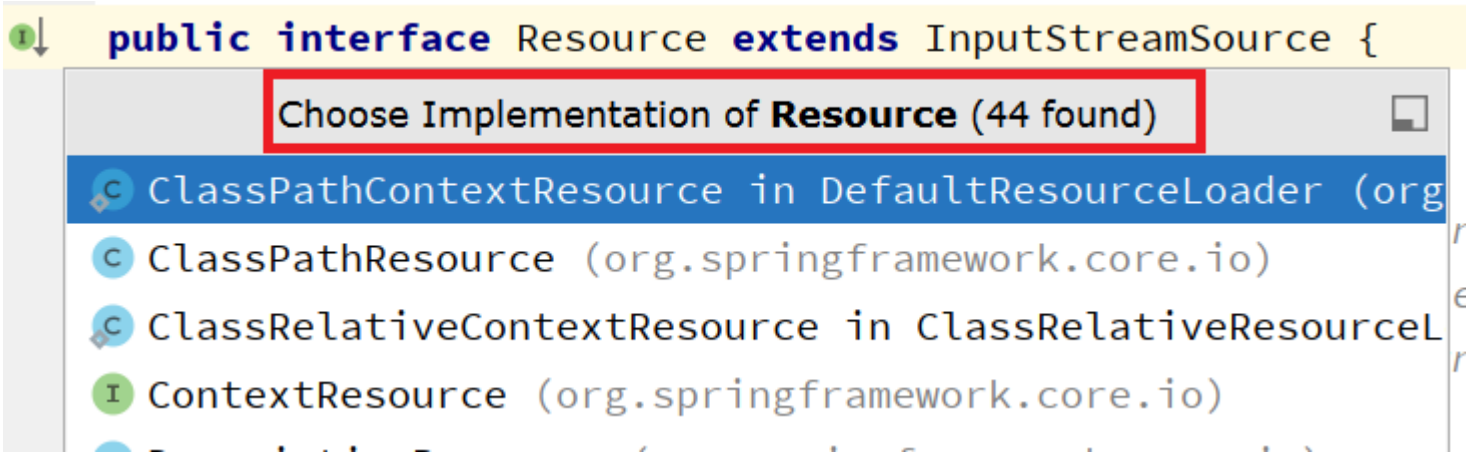


buffered

plain

Не погорячились ли мы?

```
public interface Resource extends InputStreamSource { }
```



А чё там у котлиновцев?

А чё там у котлиновцев?

```
return File(name).inputStream().buffered();
```

А чё там у котлиновцев?

```
return File(name).inputStream().buffered();
```

```
@kotlin.internal.InlineOnly
fun InputStream.buffered(size: Int = DEFAULT_BUFFER_SIZE) =
  if (this is BufferedInputStream)
    this
  else
    BufferedInputStream(this, size)
```

Напрашивается простое решение а-ля Котлин

```
static InputStream buffered(InputStream is) {
  return is instanceof BufferedInputStream
         ? is
         : new BufferedInputStream(is);
}
```

Но есть нюанс :)

```
public class ByteArrayInputStream extends InputStream {









}
```

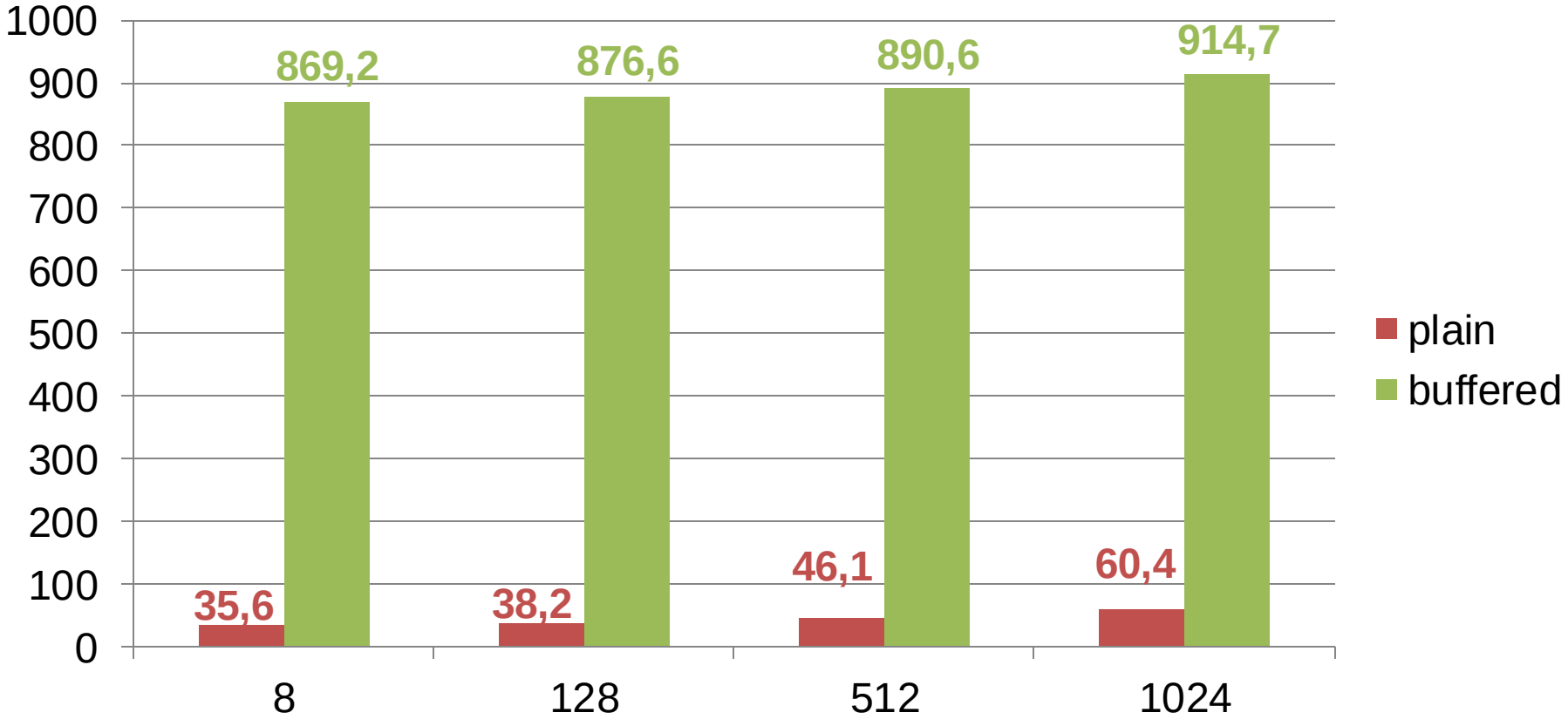
Но есть нюанс :)

```
public class ByteArrayInputStream extends InputStream {
  protected byte buf[];

  public synchronized int read() {
    return pos < count ? buf[pos++] & 0xff : -1;
  }

  public synchronized int read(byte b[], int off, int len) {
    //...
    System.arraycopy(buf, pos, b, off, len);
  }

}
```

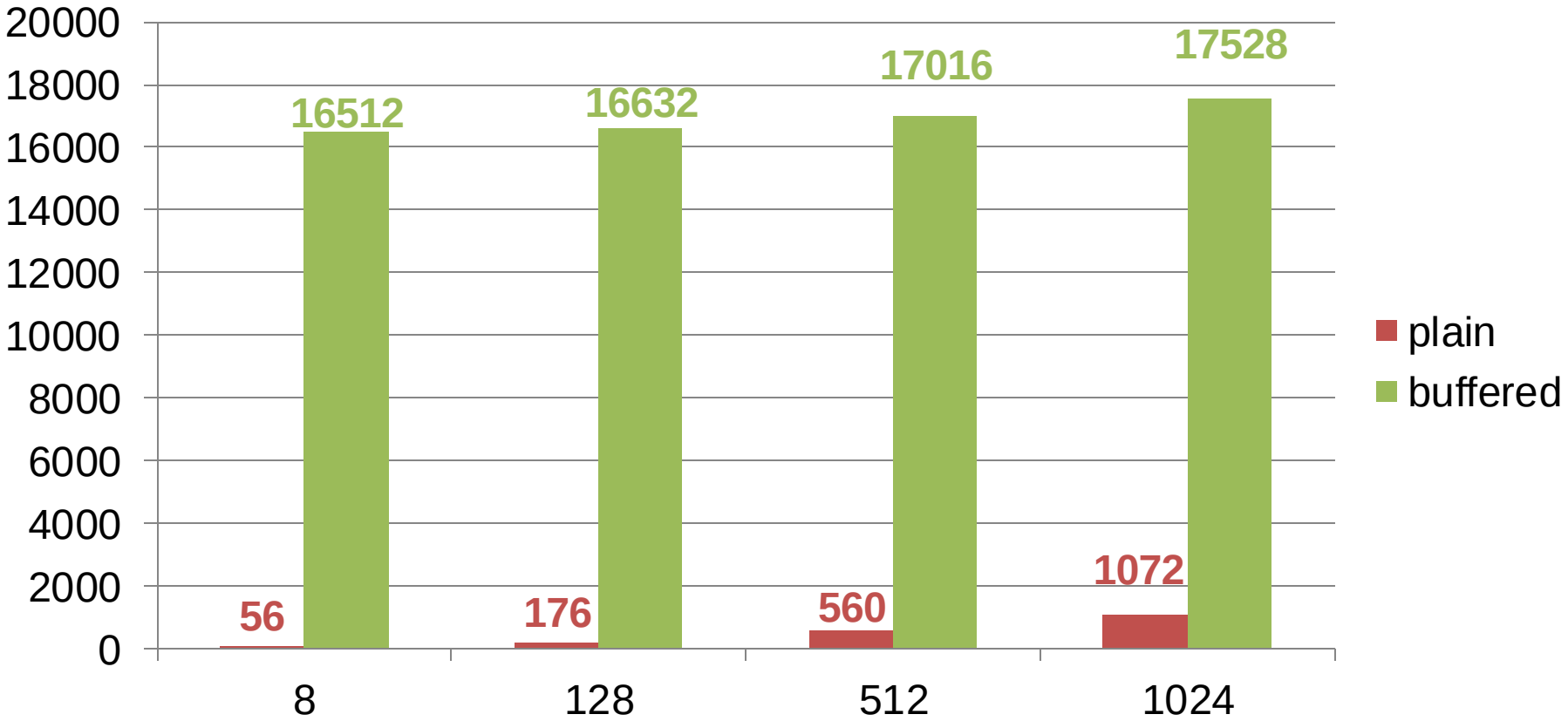
ByteArrayInputStreamBenchmark

```
@Benchmark
public Object read(Data data) {
  return data.bais.readAllBytes();
}

@Benchmark
public Object readBuffered(Data data) throws IOException {
  return new BufferedInputStream(data.bais).readAllBytes();
}
```

«Скрипач не нужен, родной» (время в нс)



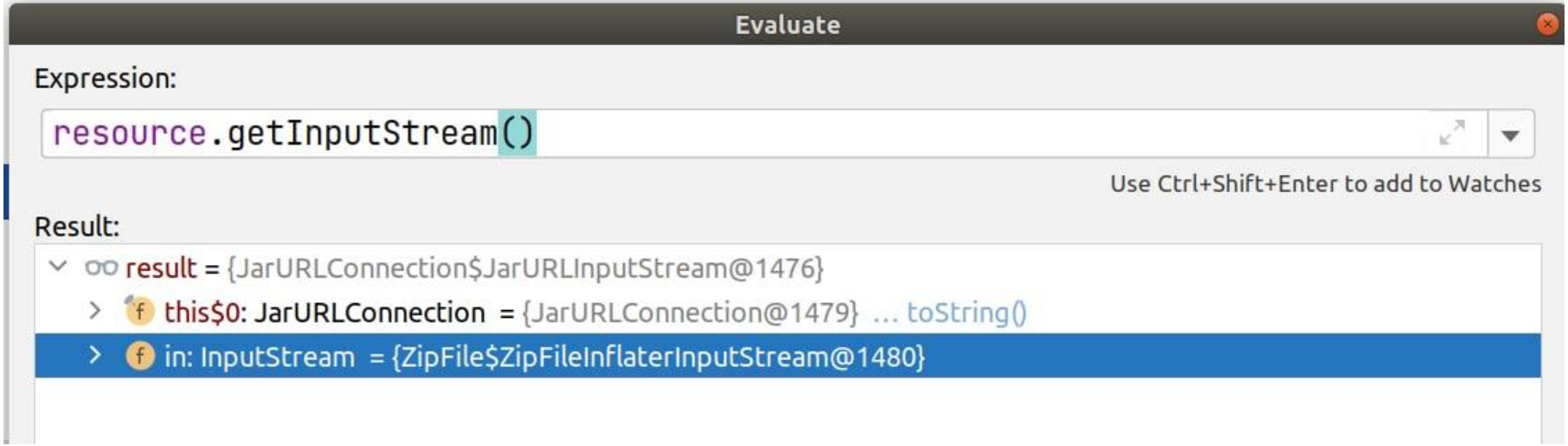
«Скрипач не нужен, родной» (память в байтах)



Напрашивается простое решение а-ля Котлин

```
public static InputStream buffered(InputStream is) {
  return requiresBuffer(is) ? is : new BufferedInputStream(is);
}

private static boolean requiresBuffer(InputStream is) {
  return is instanceof BufferedInputStream
      || is instanceof ByteArrayInputStream;
}
```

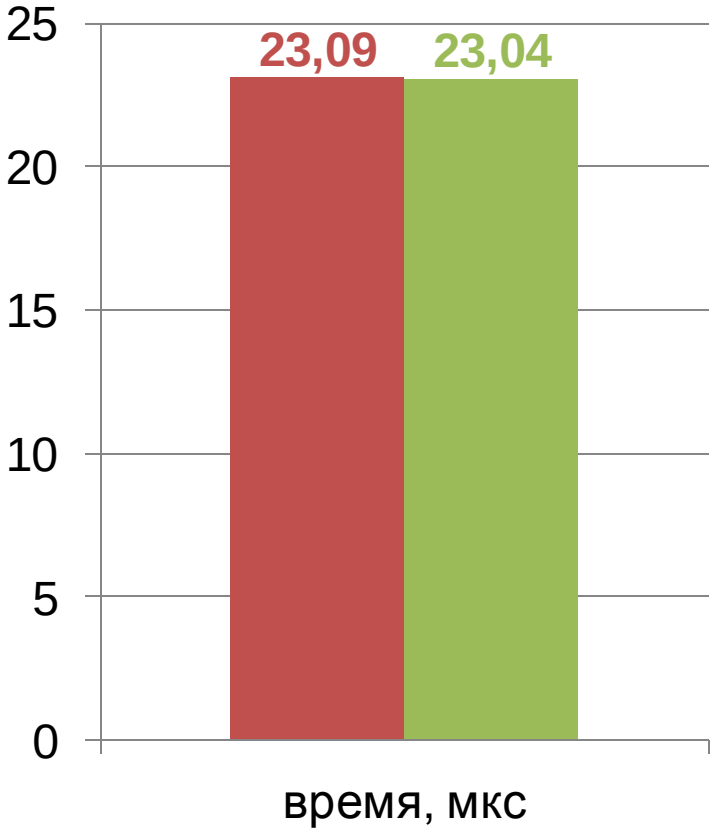
А если в названии класса нет Byte*?

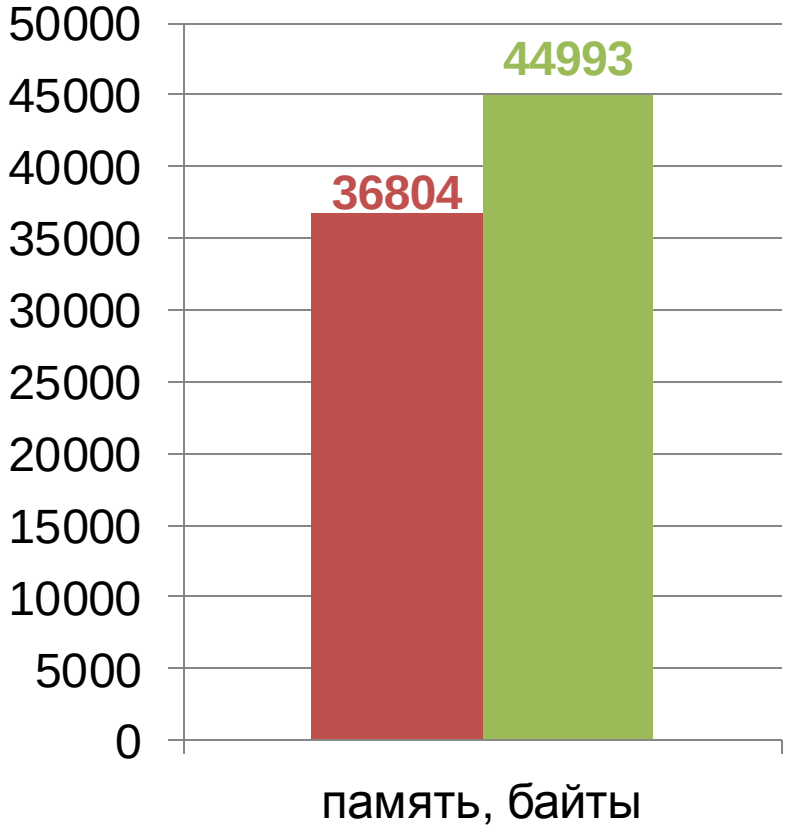


А если в названии класса нет Byte*?

```
@Setup
public void setUp() { rsc = loader.getResource(s); }

@Benchmark
public Object read() throws IOException {
  try (var is = rsc.getInputStream())
    return new ClassReader(is);
}

@Benchmark
public Object readBuffered() throws IOException {
  try (var is = new BufferedInputStream(rsc.getInputStream()))
    return new ClassReader(is);
}
```

«Скрипач не нужен, родной»





plain
buffered

Напрашивается простое решение а-ля Котлин

```
public static InputStream buffered(InputStream is) {
  return requiresBuffer(is) ? is : new BufferedInputStream(is);
}

private static boolean requiresBuffer(InputStream is) {
  return is instanceof BufferedInputStream
      || is instanceof ByteArrayInputStream
      || is instanceof ChannelInputStream
      || is instanceof ZipFile.ZipFileInflaterInputStream;
}
```

Появляется неудобство

```
private static boolean requiresBuffer(InputStream is) {
  return is instanceof BufferedInputStream
      || is instanceof ByteArrayInputStream
      || is instanceof ChannelInputStream
      || is instanceof ZipFile.ZipFileInflaterInputStream;
}
```

```
private class ZipFileInflaterInputStream extends InflaterInputStream{
}
```

Появляется неудобство

```
private static boolean requiresBuffer(InputStream is) {
  return is instanceof BufferedInputStream
      || is instanceof ByteArrayInputStream
      || is instanceof ChannelInputStream
      || is instanceof ZipFile.ZipFileInflaterInputStream;
}

error: package sun.nio.ch is not visible
import sun.nio.ch.ChannelInputStream;
               ^
  (package sun.nio.ch is declared in module java.base, which does not
export it to the unnamed module)
```

Что делать*?

```
public interface InputStream extends Closeable {

  default boolean isBuffered() { return false; }

}

public static InputStream buffered(InputStream is) {
  return is.isBuffered() ? is : new BufferedInputStream(is);
}
```

* в идеальном мире

Что делать здесь и сейчас?

- по возможности подстраиваться под своё приложение

Что делать здесь и сейчас?

- по возможности подстраиваться под своё приложение
- использовать практические знания и убирать лишнюю буферизацию

Будь как org.apache.commons.io.IOUtils

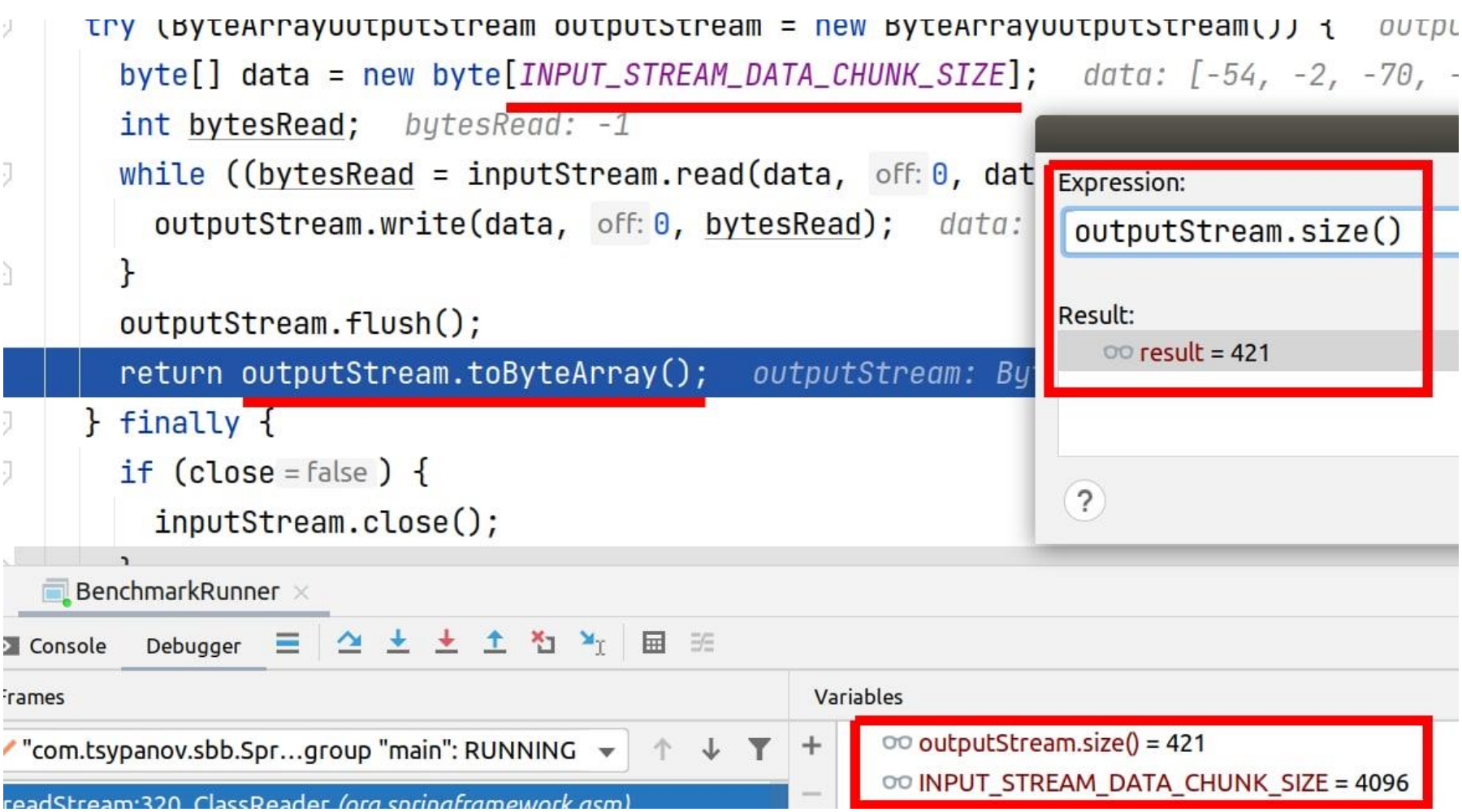
```
/**
 *
 * All the methods in this class
 * that read a stream are buffered internally.
 *
 */
public class IOUtils {

  /**
   * This method buffers the input internally, so there is no need
   * to use a BufferedInputStream.
   */
  List<String> readLines(InputStream input, Charset encoding) {}
}
```

Вернёмся к org.springframework.asm.ClassReader

```
byte[] readStream(InputStream inputStream, boolean close) {

 try (ByteArrayOutputStream os = new ByteArrayOutputStream()) {
  byte[] data = new byte[INPUT_STREAM_DATA_CHUNK_SIZE];
  int read;
  while ((read = inputStream.read(data, 0, data.length)) != -1) {
   os.write(data, 0, read);
  }
  os.flush();
  return os.toByteArray();
 }
}
```

Вернёмся к org.springframework.asm.ClassReader

```
byte[] readStream(InputStream inputStream, boolean close) {

 try (ByteArrayOutputStream os = new ByteArrayOutputStream()) {
  byte[] data = new byte[INPUT_STREAM_DATA_CHUNK_SIZE];
  int read;
  while ((read = inputStream.read(data, 0, data.length)) != -1) {
   os.write(data, 0, read);
  }
  os.flush();
  return os.toByteArray();
 }
}
```


try (ByteArrayOutputStream outputStream = new ByteArrayOutputStream()) {
byte[] data = new byte[INPUT_STREAM_DATA_CHUNK_SIZE];
data: [-54, -2, -70,
int bytesRead;
bytesRead: -1
while ((bytesRead = inputStream.read(data, off: 0, dat
outputStream.write(data, off: 0, bytesRead);
data:
}
outputStream.flush();
return outputStream.toByteArray();
outputStream: By
} finally {
if (close = false) {
inputStream.close();
Expression:
outputStream.size()
Result:
result = 421
?
BenchmarkRunner
Console
Debugger
Frames
"com.tsypanov.sbb.Spr...group "main": RUNNING
Variables
outputStream.size() = 421
INPUT_STREAM_DATA_CHUNK_SIZE = 4096

Зайдём в лоб

```
int expectedLength = inputStream.available();

try (ByteArrayOutputStream os = new ByteArrayOutputStream()) {
 byte[] data = new byte[expectedLength];
 int bytesRead;
 while ((bytesRead = inputStream.read(data, 0, available)) != -1) {
  os.write(data, 0, bytesRead);
 }
 os.flush();
 return expectedLength == os.size() ? data : os.toByteArray();
}
```

Есть нюанс

```
/**
 * ... A single read or skip of this many bytes will not block, but
 * may read or skip fewer bytes.
 * ...
 **/
public int available() throws IOException { }
```

Зайдём в лоб: возможно > 1 чтение!

```
int expectedLength = inputStream.available();

try (ByteArrayOutputStream os = new ByteArrayOutputStream()) {
 byte[] data = new byte[expectedLength];
 int bytesRead;
 while ((bytesRead = inputStream.read(data, 0, available)) != -1) {
  os.write(data, 0, bytesRead);
 }
 os.flush();
 return expectedLength == os.size() ? data : os.toByteArray();
}
```

Зайдём с тыла

```
int expectedLength = inputStream.available();

try (ByteArrayOutputStream os = new ByteArrayOutputStream()) {
 byte[] data = new byte[expectedLength];
 int bytesRead;
 int readCount = 0;
 while ((bytesRead = inputStream.read(data, 0, bufferSize)) != -1) {
  os.write(data, 0, bytesRead);
  readCount++;
 }
 os.flush();
 return readCount == 1 ? data : os.toByteArray();
}
```

Пришла беда, откуда не ждали

```
int expectedLength = inputStream.available();

try (ByteArrayOutputStream os = new ByteArrayOutputStream()) {
 byte[] data = new byte[expectedLength];
 int bytesRead;
 int readCount = 0;
 while ((bytesRead = inputStream.read(data, 0, bufferSize)) != -1) {
  os.write(data, 0, bytesRead);
  readCount++;
 }
 os.flush();
 return readCount == 1 ? data : os.toByteArray();
}
```

Пришла беда, откуда не ждали

```
var file = new File("/home/s.tsypanov/.bashrc");
var fileInputStream = new FileInputStream(file);
var available = fileInputStream.available();                // 4362
var arrayLength = fileInputStream.readAllBytes().length; // 4362
```

Пришла беда, откуда не ждали

```
var file = new File("/home/s.tsypanov/.bashrc");
var fileInputStream = new FileInputStream(file);
var available = fileInputStream.available();                // 4362
var arrayLength = fileInputStream.readAllBytes().length; // 4362
```

```
var file = new File("/proc/self/stat");
var fileInputStream = new FileInputStream(file);
var available = fileInputStream.available();                // 0
var arrayLength = fileInputStream.readAllBytes().length; // 324
```

Зайдём по третьему кругу

```
int expectedLength = inputStream.available();

// some implementations can return 0 while holding available data

int bufferSize = expectedLength == 0
        ? INPUT_STREAM_DATA_CHUNK_SIZE
        : expectedLength;

// ...

byte[] data = new byte[bufferSize];
```

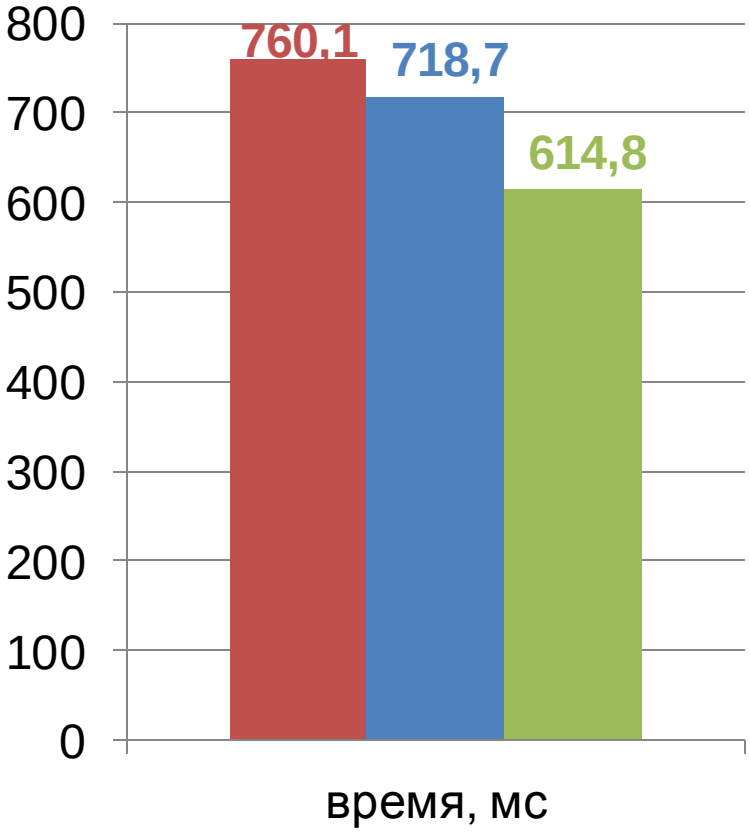
Есть нюанс

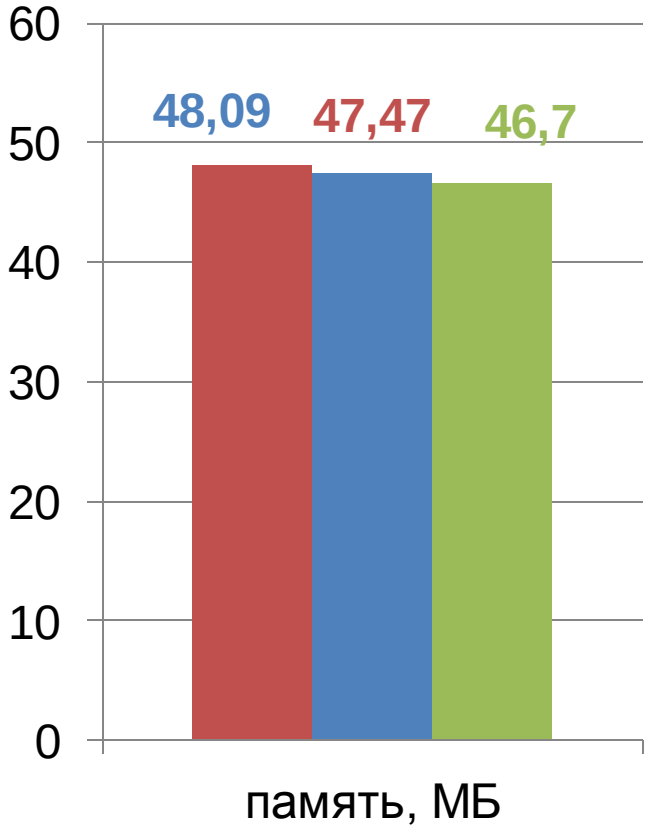
```
/**
 * Returns an estimate of the number of bytes that can be read (or
 * skipped over) from this input stream without blocking, which may
 * be 0, or 0 when end of stream is detected.
 * ...
 *
 **/
public int available() throws IOException { }
```

Зайдём по четвёртому кругу

```
int expectedLength = inputStream.available();



int bufferSize = expectedLength < 256
        ? INPUT_STREAM_DATA_CHUNK_SIZE
        : expectedLength;

// ...

byte[] data = new byte[bufferSize];
```

В крупную клетку





buffered
plain
improved CR

665

https://gitlab.ow2.org/asm/asm/-/merge_requests/314

Вывод для рядового разработчика

Производительность IO – это страна, где «много-много диких обезьян».

Выводы итоговые

- правильно истолковать вывод бенчмарка столь же важно, как и правильно написать этот самый бенчмарк
- правильно истолкованные результаты правильно написанного бенчмарка не являются руководством к немедленному действию, ибо...
- измерять нужно именно конечный сценарий, улучшение частного может сильно ухудшить целое
- вникайте в контекст, производительность – это немного про археологию
- имея дело с интерфейсом не стесняйтесь заглянуть внутрь реализаций и не доверяйте слепо красотам/ужасам документации

Благодарю за внимание

sergei.tsypanov@yandex.ru