



@JohnWings

#Jokerconf

# Жизненный цикл JIT кода

Иван Крылов  
Октябрь, 2016













# А туда ли вы пришли?

**2**

**Бликие Контакты JMM-степени  
Алексей Шипилёв**

**3**

**Low Latency & Mechanical Sympathy  
Jean-Philippe Bempel**

**4**

**Advanced search for your legacy application  
David Pilato**

**5**

**Polyglot on the JVM with Graal  
Vojin Jovanovic**

**6**

**Анти-введение в Big Data  
Владимир Красильщик**

# А туда ли вы пришли?

- Этот доклад про внутренности JVM и про производительность

**2**

**Ближкие Контакты JMM-степени  
Алексей Шипилёв**

**3**

**Low Latency & Mechanical Sympathy  
Jean-Philippe Bempel**

**4**

**Advanced search for your legacy application  
David Pilato**

**5**

**Polyglot on the JVM with Graal  
Vojin Jovanovic**

**6**

**Анти-введение в Big Data  
Владимир Красильщик**

# А туда ли вы пришли?

- Этот доклад про внутренности JVM и про производительность
- Общий интерес

**2**

**Ближкие Контакты JMM-степени  
Алексей Шипилёв**

**3**

**Low Latency & Mechanical Sympathy  
Jean-Philippe Bempel**

**4**

**Advanced search for your legacy application  
David Pilato**

**5**

**Polyglot on the JVM with Graal  
Vojin Jovanovic**

**6**

**Анти-введение в Big Data  
Владимир Красильщик**

# А туда ли вы пришли?

- Этот доклад про внутренности JVM и про производительность
- Общий интерес
- Отвечаете за производительность

2

**Ближкие Контакты JMM-степени**  
**Алексей Шипилёв**

3

**Low Latency & Mechanical Sympathy**  
**Jean-Philippe Bempel**

4

**Advanced search for your legacy application**  
**David Pilato**

5

**Polyglot on the JVM with Graal**  
**Vojin Jovanovic**

6

**Анти-введение в Big Data**  
**Владимир Красильщик**

# А туда ли вы пришли?

- Этот доклад про внутренности JVM и про производительность
- Общий интерес
- Отвечаете за производительность
  - Есть на чем тестировать

2

**Ближкие Контакты JMM-степени**  
**Алексей Шипилёв**

3

**Low Latency & Mechanical Sympathy**  
**Jean-Philippe Bempel**

4

**Advanced search for your legacy application**  
**David Pilato**

5

**Polyglot on the JVM with Graal**  
**Vojin Jovanovic**

6

**Анти-введение в Big Data**  
**Владимир Красильщик**



# А туда ли вы пришли?

- Этот доклад про внутренности JVM и про производительность
- Общий интерес
- Отвечаете за производительность
  - Есть на чем тестировать
  - Возможность собирать логи

2

**Ближкие Контакты JMM-степени**  
**Алексей Шипилёв**

3

**Low Latency & Mechanical Sympathy**  
**Jean-Philippe Bempel**

4

**Advanced search for your legacy application**  
**David Pilato**

5

**Polyglot on the JVM with Graal**  
**Vojin Jovanovic**

6

**Анти-введение в Big Data**  
**Владимир Красильщик**



# О чем будем говорить





# О чем будем говорить

- Трансформация представления кода





# О чем будем говорить

- Трансформация представления кода
- Профили кода





# О чем будем говорить

- Трансформация представления кода
- Профили кода
- Деоптимизация





# О чем будем говорить

- Трансформация представления кода
- Профили кода
- Деоптимизация
- 4 API для тюнинга компиляции





# Конвейер

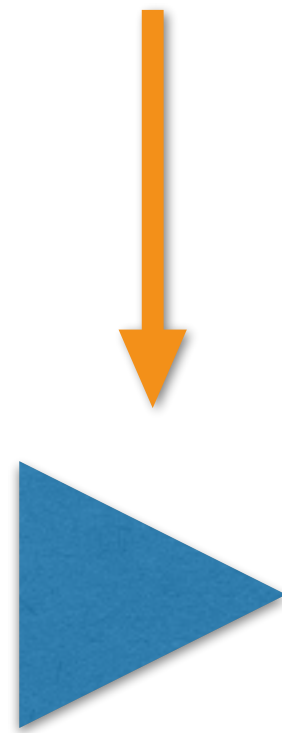


# Конвейер

**Javac**

- 1) Статическая верификация
- 2) Неоптимизирующая компиляция

**Исходный  
код  
\*.java**



**Байткод**

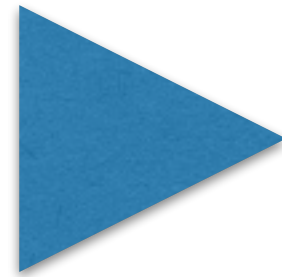


# Конвейер

**Javac**

- 1) Статическая верификация
- 2) Неоптимизирующая компиляция

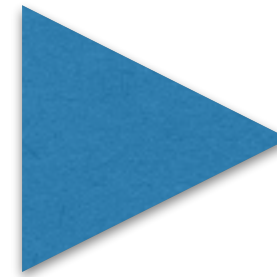
**Исходный  
код  
\*.java**



**Байткод**

**Runtime**

- 1) Верификация
- 2) Линковка



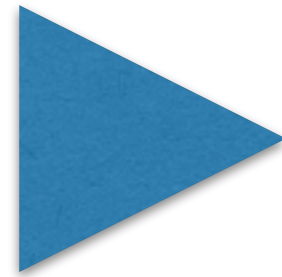


# Конвейер

Javaс

- 1) Статическая верификация
- 2) Неоптимизирующая компиляция

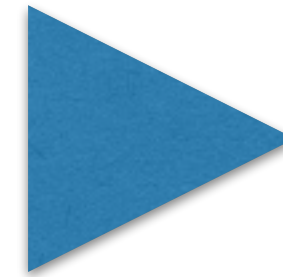
Исходный  
код  
\*.java



Байткод

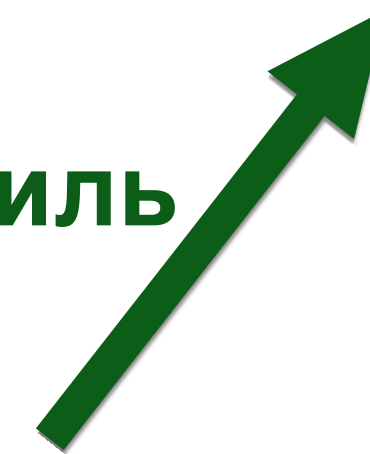
Runtime

- 1) Верификация
- 2) Линковка



Интерпретатор

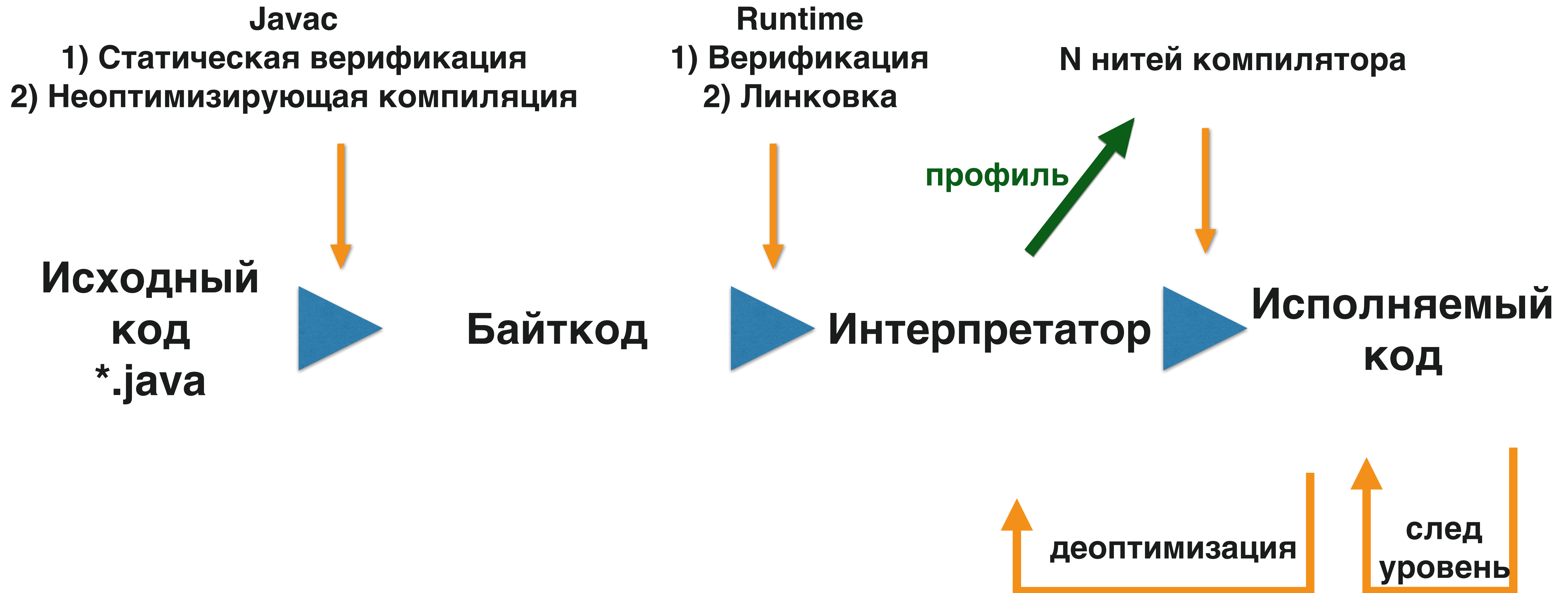
профиль



N нитей компилятора



# Конвейер





# Но может быть и иначе





# Но может быть и иначе





# Но может быть и иначе

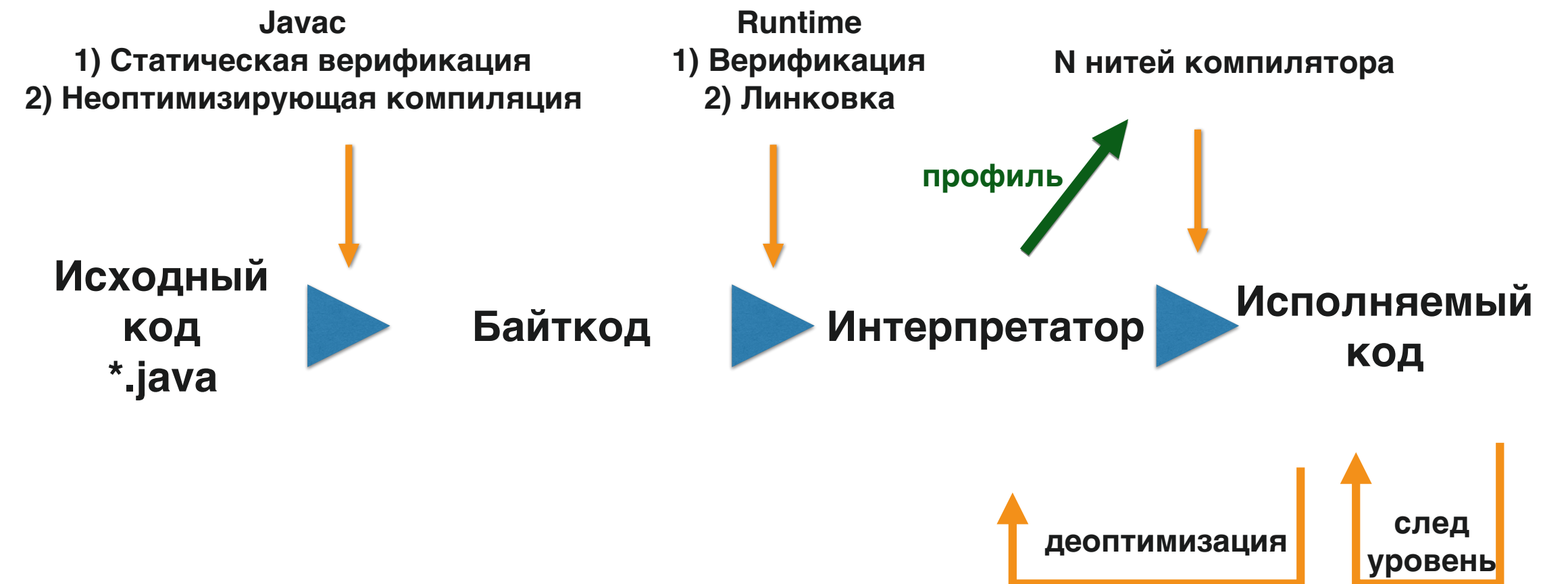
- Байткод в VM может “прилететь” вовсе не из javac
- Компилятор любого JVM языка; jasm, ByteBuddy, ...





# Но может быть и иначе

- Байткод в VM может “прилететь” вовсе не из javac
  - Компилятор любого JVM языка; jasm, ByteBuddy, ...
- VM бывают без интерпретатора
  - JRockit VM





# Но может быть и иначе

- Байткод в VM может “прилететь” вовсе не из javac
  - Компилятор любого JVM языка; jasm, ByteBuddy, ...
- VM бывают без интерпретатора
  - JRockit VM
- VM бывают лишь с одним интерпретатором
  - Zero-port или -Xint





# Но может быть и иначе

- Байткод в VM может “прилететь” вовсе не из javac
  - Компилятор любого JVM языка; jasm, ByteBuddy, ...

- VM бывают без интерпретатора

- JRockit VM

- VM бывают лишь с одним интерпретатором

- Zero-port или -Xint

- AOT

- Полностью AOT можно скомпилировать далеко не всякую программу

- Гибридные JVM : Excelsior JET, AOT на базе Graal (в разработке)





# AOT в OpenJDK грядет

OpenJDK

OpenJDK FAQ

Installing

Contributing

Sponsoring

Developers' Guide

Mailing lists

IRC - Wiki

Bylaws - Census

Legal

JEP Process

Source code

Mercurial

Bundles (6)

Groups

(overview)

2D Graphics

Adoption

AWT

Build

Compiler

Conformance

Core Libraries

Governing Board

HotSpot

Internationalization

JMX

Members

Networking

NetBeans Projects

Porters

Quality

Security

Serviceability

Sound

Swing

Web

Projects

(overview)

Annotations Pipeline

2-D

Audio Engine

Build Infrastructure

Caciocavallo

Closures

Code Tools

Coin

Common VM

Interface

Compiler Grammar

Device I/O

Font Scaler

Framebuffer Toolkit

Graal

Graphics Rasterizer

Harfbuzz Integration

IcedTea

JDK 6

JDK 7

JDK 7 Updates

JDK 8 - Java SE 8

JDK 8 Updates

JDK 9

JavaDoc Next

Jigsaw

Kona

Kulla

Lambda

Locale Enhancement

Memory Model

Update

Module

Modules

Multi-Language VM

Nashorn

New I/O

OpenJFX

Panama

Penrose

Port: AArch32

Port: AArch64

Port: BSD

Port: Haiku

Port: Mac OS X

Port: MIPS

Port: PowerPC/AAIX

JEP 295: Ahead-of-Time Co...

openjdk.java.net/jeps/295

JEP 295: Ahead-of-Time Compilation

Owner

Created

Updated

Type

Status

Component

Scope

Discussion

Effort

Duration

Priority

Reviewed by

Endorsed by

Release

Issue

Vladimir Kozlov

2016/09/15 01:20

2016/10/13 02:53

Feature

Proposed to Target

hotspot / compiler

Implementation

hotspot dash compiler dash dev at openjdk dot java dot net

M

M

1

John Rose, Mikael Vidstedt

John Rose

9

8166089

Summary

Compile Java classes to native code prior to launching the virtual machine.

Goals

Improve the start-up time of both small and large Java applications, with at most a limited impact on peak performance.

Change the end user's work flow as little as possible.

Non-Goals

It is not necessary to provide an explicit, exposed library-like mechanism for saving and loading compiled code.

Motivation

JIT compilers are fast, but Java programs can become so large that it takes a long time for the JIT to warm up completely. Infrequently-used Java methods might never be compiled at all, potentially incurring a performance penalty due to repeated interpreted invocations.

Description

For the initial release, the only supported module is java.base. This is done to limit the problem space, since the Java code in java.base is well-known in advance and can be thoroughly tested internally. AOT compilation of any other JDK module, or of user code, is experimental.

To use the AOTed java.base module, the user will have to compile the module and copy the resulting AOT library into the JDK installation directory, or specify it on java command line. The usage of AOT-compiled code is otherwise completely transparent to end users.

AOT compilation is done by new a tool, jaotc:

jaotc --output libHelloWorld.so HelloWorld.class

jaotc --output libjava.base.so --module java.base

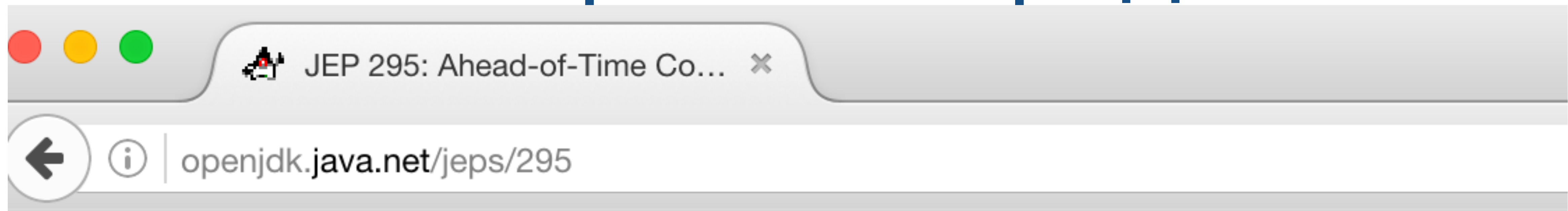
It uses Graal as the code-generating backend.

During JVM startup the AOT initialization code looks for well-known shared libraries in a well-known location, or as specified on the command line with the AOTLibrary flag. If shared libraries are found, these are picked up and used. If no shared libraries are found then AOT will be turned off for this JVM instance.

java -XX:AOTLibrary=./libHelloWorld.so,./libjava.base.so HelloWorld



# АОТ в OpenJDK грядет



## JEP 295: Ahead-of-Time Compilation

[OpenJDK FAQ](#)  
[Installing](#)  
[Contributing](#)  
[Sponsoring](#)  
[Developers' Guide](#)

[Mailing lists](#)  
[IRC · Wiki](#)  
[Bylaws · Census](#)  
[Legal](#)

### JEP Process

### Source code

[Mercurial](#)  
[Bundles \(6\)](#)

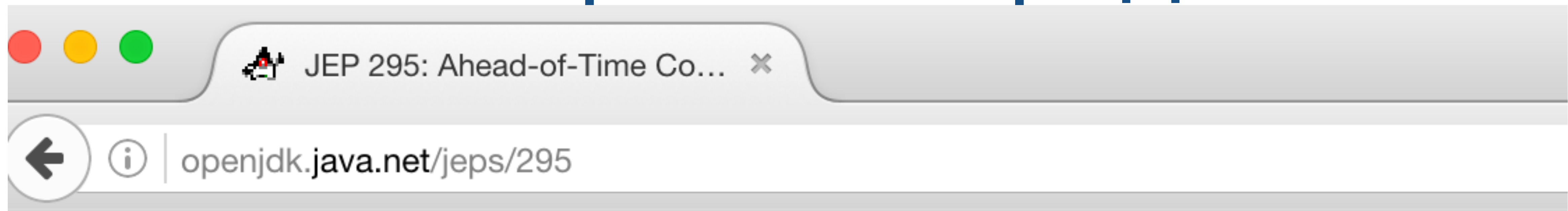
### Groups

[\(overview\)](#)  
[2D Graphics](#)

<i>Owner</i>	Vladimir Kozlov
<i>Created</i>	2016/09/15 01:20
<i>Updated</i>	2016/10/13 02:53
<i>Type</i>	Feature
<i>Status</i>	Proposed to Target
<i>Component</i>	hotspot / compiler
<i>Scope</i>	Implementation
<i>Discussion</i>	hotspot dash compiler dash dev at openjdk dot java dot net
<i>Effort</i>	M
<i>Duration</i>	M
<i>Priority</i>	1
<i>Reviewed by</i>	John Rose, Mikael Vidstedt
<i>Endorsed by</i>	John Rose



# АОТ в OpenJDK грядет



## JEP 295: Ahead-of-Time Compilation

[OpenJDK FAQ](#)  
[Installing](#)  
[Contributing](#)  
[Sponsoring](#)  
[Developers' Guide](#)

[Mailing lists](#)  
[IRC · Wiki](#)  
[Bylaws · Census](#)  
[Legal](#)

### JEP Process

### Source code

[Mercurial](#)  
[Bundles \(6\)](#)

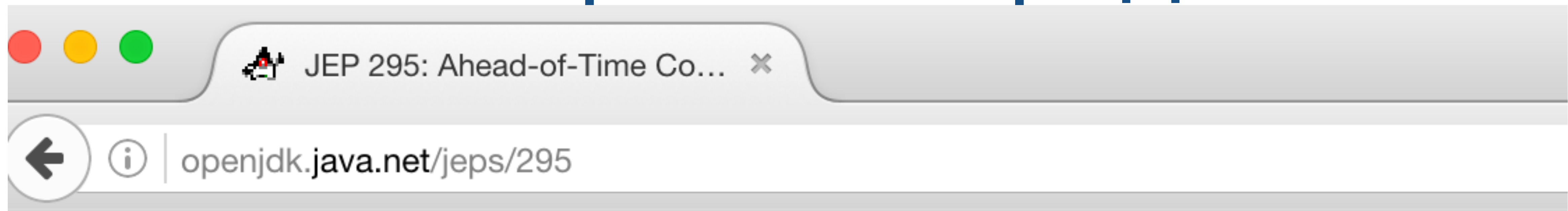
### Groups

[\(overview\)](#)  
[2D Graphics](#)

<i>Owner</i>	Vladimir Kozlov
<i>Created</i>	2016/09/15 01:20
<i>Updated</i>	2016/10/13 02:53
<i>Type</i>	Feature
<i>Status</i>	Proposed to Target
<i>Component</i>	hotspot / compiler
<i>Scope</i>	Implementation
<i>Discussion</i>	hotspot dash compiler dash dev at openjdk dot java dot net
<i>Effort</i>	M
<i>Duration</i>	M
<i>Priority</i>	1
<i>Reviewed by</i>	John Rose, Mikael Vidstedt
<i>Endorsed by</i>	John Rose



# АОТ в OpenJDK грядет



## JEP 295: Ahead-of-Time Compilation

[OpenJDK FAQ](#)  
[Installing](#)  
[Contributing](#)  
[Sponsoring](#)  
[Developers' Guide](#)

[Mailing lists](#)  
[IRC · Wiki](#)  
[Bylaws · Census](#)  
[Legal](#)

### JEP Process

### Source code

[Mercurial](#)  
[Bundles \(6\)](#)

### Groups

[\(overview\)](#)  
[2D Graphics](#)

*Owner* Vladimir Kozlov

*Created* 2016/09/15 01:20

*Updated* 2016/10/13 02:53

*Type* Feature

*Status* Proposed to Target

*Component* hotspot / compiler

*Scope* Implementation

*Discussion* hotspot dash compiler dash dev at openjdk dot java dot net

*Effort* M

*Duration* M

*Priority* 1

*Reviewed by* John Rose, Mikael Vidstedt

*Endorsed by* John Rose



# Шаблонной интерпретатор





# Шаблонной интерпретатор

- По сути - легковесный компилятор
- На базе asm-шаблонов
- Инстанциация адресами





# Шаблонной интерпретатор

- По сути - легковесный компилятор
- На базе asm-шаблонов
- Инстанциация адресами
- Ведет подсчет вызовов





# Шаблонной интерпретатор

- По сути - легковесный компилятор
- На базе asm-шаблонов
- Инстанциация адресами
- Ведет подсчет вызовов
- Поддерживает смешанный стек





# Шаблонной интерпретатор

- По сути - легковесный компилятор
- На базе asm-шаблонов
- Инстанциация адресами
- Ведет подсчет вызовов
- Поддерживает смешанный стек
- Код выглядит как одна большая функция





# Пример - ladd байткод

```
0x0000000010a59a920: mov      (%rsp), %rax
0x0000000010a59a924: add      $0x10, %rsp
0x0000000010a59a928: mov      (%rsp), %rdx
0x0000000010a59a92c: add      $0x10, %rsp
0x0000000010a59a930: add      %rdx, %rax
0x0000000010a590933: movzbl   0x1(%r13), %ebx
0x0000000010a59a938: inc      %r13
0x0000000010a59a93b: movabs   $0x10a0f6600, %r10
0x0000000010a59a945: jmpq     *(%r10, %rbx, 8)
```



# Пример - ladd байткод

результат

```
0x0000000010a59a920: mov      (%rsp), %rax
0x0000000010a59a924: add      $0x10, %rsp
0x0000000010a59a928: mov      (%rsp), %rdx
0x0000000010a59a92c: add      $0x10, %rsp
0x0000000010a59a930: add      %rdx, %rax
0x0000000010a590933: movzbl   0x1(%r13), %ebx
0x0000000010a59a938: inc      %r13
0x0000000010a59a93b: movabs   $0x10a0f6600, %r10
0x0000000010a59a945: jmpq     *(%r10, %rbx, 8)
```



# Пример - ladd байткод

результат

Чтение аргументов

```
0x0000000010a59a920: mov    (%rsp), %rax
0x0000000010a59a924: add    $0x10, %rsp
0x0000000010a59a928: mov    (%rsp), %rdx
0x0000000010a59a92c: add    $0x10, %rsp
0x0000000010a59a930: add    %rdx, %rax
0x0000000010a590933: movzbl 0x1(%r13), %ebx
0x0000000010a59a938: inc    %r13
0x0000000010a59a93b: movabs $0x10a0f6600, %r10
0x0000000010a59a945: jmpq   *(%r10, %rbx, 8)
```



# Пример - ladd байткод

результат

Чтение аргументов

```
0x0000000010a59a920: mov    (%rsp), %rax
0x0000000010a59a924: add    $0x10, %rsp
0x0000000010a59a928: mov    (%rsp), %rdx
0x0000000010a59a92c: add    $0x10, %rsp
0x0000000010a59a930: add    %rdx, %rax
```

Чтение следующего  
байткода и переход

```
0x0000000010a590933: movzbl 0x1(%r13), %ebx
0x0000000010a59a938: inc    %r13
0x0000000010a59a93b: movabs $0x10a0f6600, %r10
0x0000000010a59a945: jmpq   *(%r10, %rbx, 8)
```



# Пример - ladd байткод

результат

№1

Если в %RAX  
нет операндов

Чтение аргументов

Чтение следующего  
байткода и переход

```
0x0000000010a59a920: mov    (%rsp), %rax
0x0000000010a59a924: add    $0x10, %rsp
0x0000000010a59a928: mov    (%rsp), %rdx
0x0000000010a59a92c: add    $0x10, %rsp
0x0000000010a59a930: add    %rdx, %rax
0x0000000010a590933: movzbl 0x1(%r13), %ebx
0x0000000010a59a938: inc    %r13
0x0000000010a59a93b: movabs $0x10a0f6600, %r10
0x0000000010a59a945: jmpq   *(%r10, %rbx, 8)
```



# Пример - ladd байткод

результат

Если в %RAX нет операндов			
№1	→	0x0000000010a59a920:	mov (%rsp), %rax
		0x0000000010a59a924:	add \$0x10, %rsp
В других случаях		0x0000000010a59a928:	mov (%rsp), %rdx
№2	→	0x0000000010a59a92c:	add \$0x10, %rsp
Чтение аргументов		0x0000000010a59a930:	<b>add %rdx, %rax</b>
		0x0000000010a590933:	movzbl 0x1(%r13), %ebx
		0x0000000010a59a938:	inc %r13
		0x0000000010a59a93b:	movabs \$0x10a0f6600, %r10
Чтение следующего байткода и переход		0x0000000010a59a945:	jmpq *(%r10, %rbx, 8)



# Пример - ladd байткод

результат

№1 Если в %RAX  
нет операндов

0x0000000010a59a920: mov (%rsp), %rax

№2 В других случаях

0x0000000010a59a924: add \$0x10, %rsp

Чтение аргументов

0x0000000010a59a928: mov (%rsp), %rdx

0x0000000010a59a92c: add \$0x10, %rsp

0x0000000010a59a930: **add %rdx, %rax**

Чтение следующего  
байткода и переход

0x0000000010a590933: movzbl 0x1(%r13), %ebx

0x0000000010a59a938: inc %r13

0x0000000010a59a93b: movabs \$0x10a0f6600, %r10

0x0000000010a59a945: jmpq \*(%r10, %rbx, 8)

указатель  
на байткоды



# Пример - ladd байткод

результат

№1 Если в %RAX  
нет операндов

0x0000000010a59a920: mov (%rsp), %rax

0x0000000010a59a924: add \$0x10, %rsp

№2 В других случаях

0x0000000010a59a928: mov (%rsp), %rdx

Чтение аргументов

0x0000000010a59a92c: add \$0x10, %rsp

0x0000000010a59a930: **add %rdx, %rax**

Чтение следующего  
байткода и переход

0x0000000010a590933: movzbl 0x1(%r13), %ebx

0x0000000010a59a938: inc %r13

0x0000000010a59a93b: movabs \$0x10a0f6600, %r10

0x0000000010a59a945: jmpq \*(%r10, %rbx, 8)

указатель  
на байткоды

фактическое значение



# Пример - ladd байткод

начало метода – таблица переходов  
0x0000000010a0f6600:

результат

№1    Если в %RAX  
      нет операндов

№2    В других случаях

Чтение аргументов

Чтение следующего  
байткода и переход

```
0x0000000010a59a920: mov     (%rsp), %rax
0x0000000010a59a924: add     $0x10, %rsp
0x0000000010a59a928: mov     (%rsp), %rdx
0x0000000010a59a92c: add     $0x10, %rsp
0x0000000010a59a930: add     %rdx, %rax
0x0000000010a590933: movzbl 0x1(%r13), %ebx
0x0000000010a59a938: inc     %r13
0x0000000010a59a93b: movabs  $0x10a0f6600, %r10
0x0000000010a59a945: jmpq    *(%r10, %rbx, 8)
```

указатель  
на байткоды

фактическое значение



# Что такое профиль ?





# Что такое профиль ?

- Это счетчики





# Что такое профиль ?

- Это счетчики
- Неточные





# Что такое профиль ?

- Это счетчики
- Неточные
- Важно соотношение, а не абсолютные величины





# Что такое профиль ?

- Это счетчики
- Неточные
- Важно соотношение, а не абсолютные величины
- С защитой от переполнения





# Что такое профиль ?

- Это счетчики
- Неточные
- Важно соотношение, а не абсолютные величины
- С защитой от переполнения
- Иногда есть дополнительное поле для свойств





# Что такое профиль ?

- Это счетчики
- Неточные
- Важно соотношение, а не абсолютные величины
- С защитой от переполнения
- Иногда есть дополнительное поле для свойств
- Какие классы попались при профилировании





# Посмотрим на `methodCounter.hpp`



# Посмотрим на methodCounter.hpp

**interpreter\_invocation\_count &  
invocation\_counter**

**interpreter\_throwout\_count**

**backedge\_counter**

number\_of\_breakpoints (4 dbg)

nmethod\_age

interpreter\_invocation\_limit &  
interpreter\_backward\_branch\_limit

interpreter\_profile\_limit

invoke\_mask

backedge\_mask

rate & prev\_time

highest\_comp\_level &  
highest\_osr\_comp\_level



# А кроме профиля что нужно?



# А кроме профиля что нужно?

```
class A {  
    static List<String> list = new LinkedList<>();  
    static {  
        DateFormat dateFormat = new  
            SimpleDateFormat("yyyy/MM/dd HH:mm:ss");  
        list.add(dateFormat.format(new Date()));  
    }  
    static List<String> getList() {  
        return list;  
    }  
}
```



# А кроме профиля что нужно?

```
class A {  
    static List<String> list = new LinkedList<>();  
    static {  
        DateFormat dateFormat = new  
            SimpleDateFormat("yyyy/MM/dd HH:mm:ss");  
        list.add(dateFormat.format(new Date()));  
    }  
    static List<String> getList() {  
        return list;  
    }  
}
```

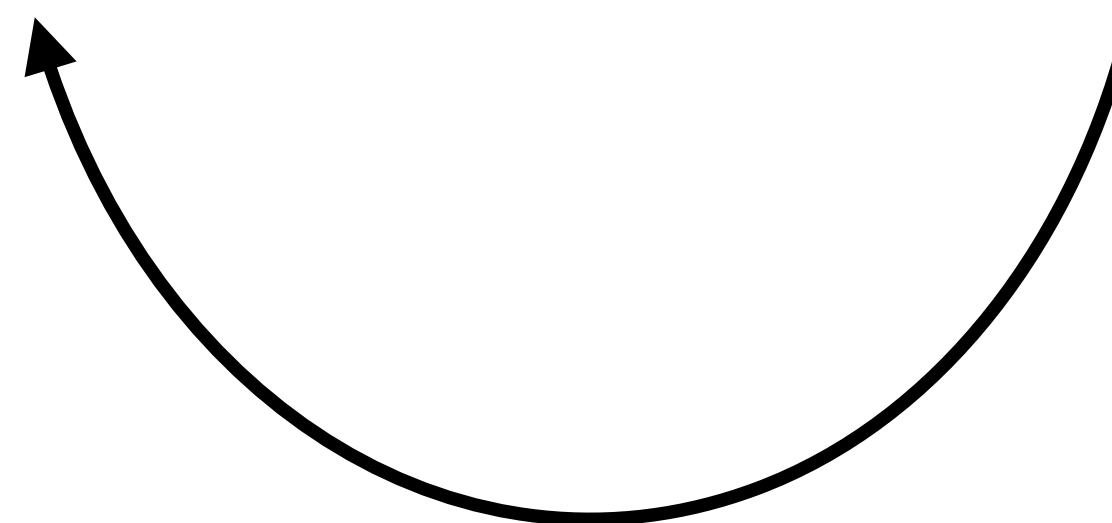
```
class B  
{  
    String process(String s, int i)  
    {  
        if (i < A.list.size())  
            return s.concat(A.list.get(i));  
        else  
            return s;  
    }  
}
```



# А кроме профиля что нужно?

```
class A {  
    static List<String> list = new LinkedList<>();  
    static {  
        DateFormat dateFormat = new  
            SimpleDateFormat("yyyy/MM/dd HH:mm:ss");  
        list.add(dateFormat.format(new Date()));  
    }  
    static List<String> getList() {  
        return list;  
    }  
}
```

```
class B  
{  
    String process(String s, int i)  
    {  
        if (i < A.list.size())  
            return s.concat(A.list.get(i));  
        else  
            return s;  
    }  
}
```

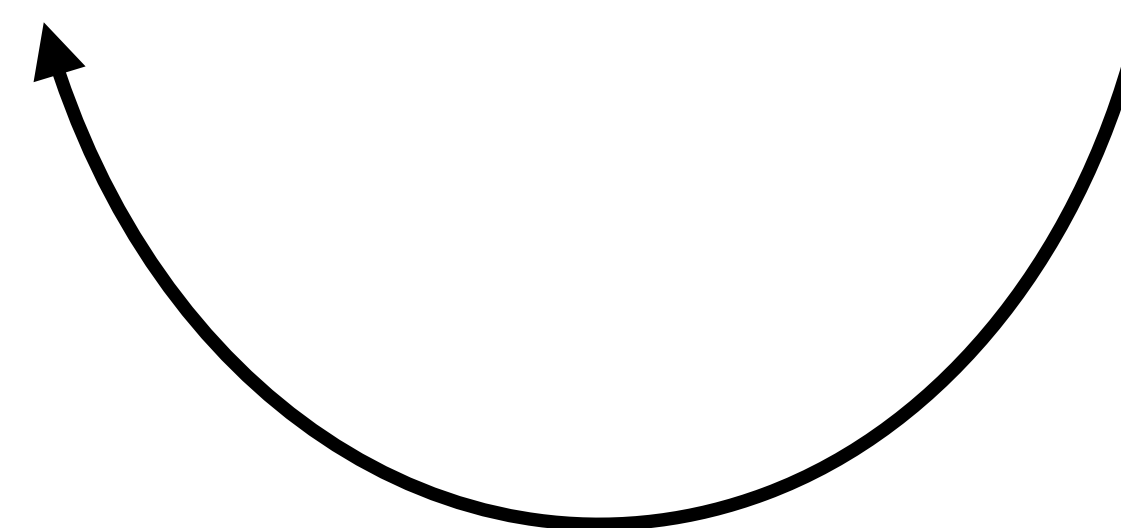




# А кроме профиля что нужно?

```
class A {  
    static List<String> list = new LinkedList<>();  
    static {  
        DateFormat dateFormat = new  
            SimpleDateFormat("yyyy/MM/dd HH:mm:ss");  
        list.add(dateFormat.format(new Date()));  
    }  
    static List<String> getList() {  
        return list;  
    }  
}
```

```
class B  
{  
    String process(String s, int i)  
    {  
        if (i < A.list.size())  
            return s.concat(A.list.get(i));  
        else  
            return s;  
    }  
}
```



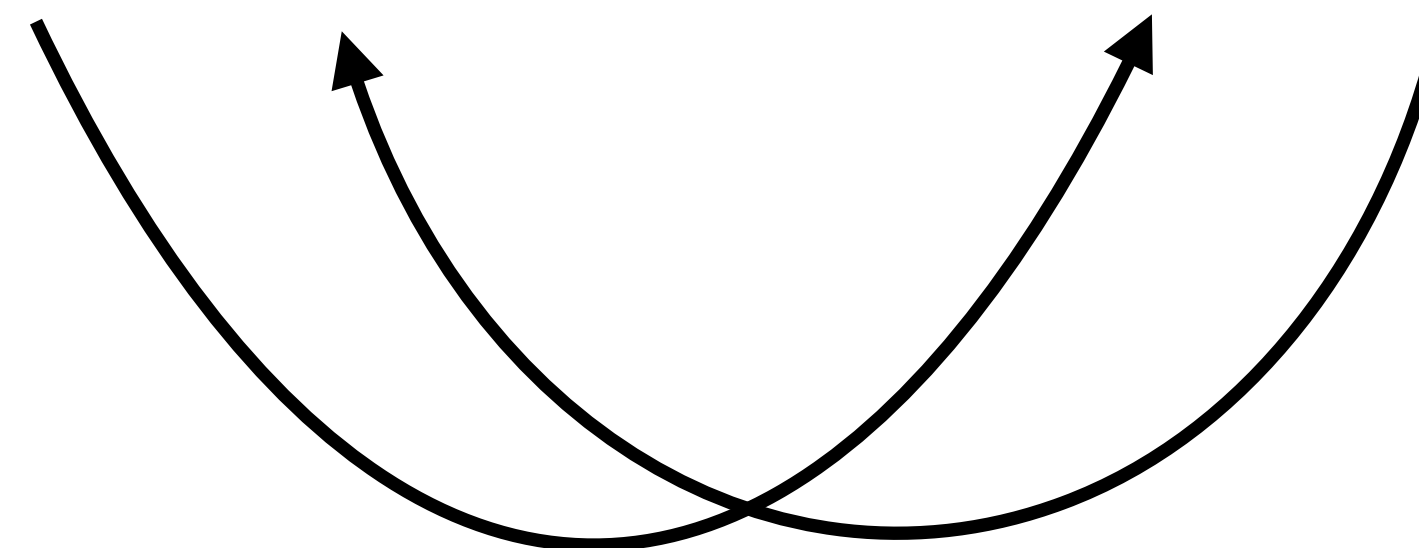
В.process() можно  
**скомпилировать**  
после инициализации  
класса А



# А кроме профиля что нужно?

```
class A {  
    static List<String> list = new LinkedList<>();  
    static {  
        DateFormat dateFormat = new  
            SimpleDateFormat("yyyy/MM/dd HH:mm:ss");  
        list.add(dateFormat.format(new Date()));  
    }  
    static List<String> getList() {  
        return list;  
    }  
}
```

```
class B  
{  
    String process(String s, int i)  
    {  
        if (i < A.list.size())  
            return s.concat(A.list.get(i));  
        else  
            return s;  
    }  
}
```



В.process() можно  
скомпилировать  
после инициализации  
класса А



# А кроме профиля что нужно?

```
class A {  
    static List<String> list = new LinkedList<>();  
    static {  
        DateFormat dateFormat = new  
            SimpleDateFormat("yyyy/MM/dd HH:mm:ss");  
        list.add(dateFormat.format(new Date()));  
    }  
    static List<String> getList() {  
        return list;  
    }  
}
```

```
class B  
{  
    String process(String s, int i)  
    {  
        if (i < A.list.size())  
            return s.concat(A.list.get(i));  
        else  
            return s;  
    }  
}
```

**Инициализация A** происходит  
после первого вызова

`B.process()`

(Если только `B.process()` использует A)

`B.process()` можно  
**скомпилировать**  
после инициализации  
класса A

# Инлайнинг - ключик к оптимизациям



# Инлайнинг - ключик к оптимизациям

```
class A {  
    boolean Cond;  
    A (bool v) { Cond = v; }  
  
    int outerMethod(B b,C c)  
    {  
        int a=2;  
        if (Cond) {  
            a+=b.inner1();  
        }  
        else  
            a+=c.inner2();  
        return a;  
    }  
}
```

# Инлайнинг - ключик к оптимизациям

```
class A {  
    boolean Cond;  
    A (bool v) { Cond = v; }  
  
    int outerMethod(B b,C c)  
    {  
        int a=2;  
        if (Cond) {  
            a+=b.inner1();  
        }  
        else  
            a+=c.inner2();  
        return a;  
    }  
}
```

```
class B {  
    int inner1() {  
        ...;  
    }  
}
```



# Инлайнинг - ключик к оптимизациям

```
class A {  
    boolean Cond;  
    A (bool v) { Cond = v; }  
  
    int outerMethod(B b,C c)  
    {  
        int a=2;  
        if (Cond) {  
            a+=b.inner1();  
        }  
        else  
            a+=c.inner2();  
        return a;  
    }  
}
```

```
class B {  
    int inner1() {  
        ...;  
    }  
}
```

```
class C {  
    int inner2() {  
        ... .  
    }  
}
```

# Инлайнинг - ключик к оптимизациям

```
class A {  
    boolean Cond;  
    A (bool v) { Cond = v; }  
  
    int outerMethod(B b,C c)  
    {  
        int a=2;  
        if (Cond) {  
            a+=b.inner1();  
        }  
        else  
            a+=c.inner2();  
        return a;  
    }  
}
```

```
class B {  
    int inner1() {  
        ...;  
    }  
}
```

```
class C {  
    int inner2() {  
        ...  
    }  
}
```



# Инлайнинг - ключик к оптимизациям

```
class A {  
    boolean Cond;  
    A (bool v) { Cond = v; }  
  
    int outerMethod(B b,C c)  
    {  
        int a=2;  
        if (Cond) {  
            a+=b.inner1();  
        }  
        else  
            a+=c.inner2();  
        return a;  
    }  
}
```

**N**

```
class B {  
    int inner1() {  
        ...;  
    }  
}
```

**M1**

```
class C {  
    int inner2() {  
        ... .  
    }  
}
```

**M2**

# Недавний баг про инлайнинг



# Недавний баг про инлайнинг

- Обнаружили регрессию в 24% на бенчмарке

# Недавний баг про инлайнинг

- Обнаружили регрессию в 24% на бенчмарке
- Анализ данных инлайнинга



# Недавний баг про инлайнинг

- Обнаружили регрессию в 24% на бенчмарке
- Анализ данных инлайнинга
  - 4249 - заинлайнено до регрессии

# Недавний баг про инлайнинг

- Обнаружили регрессию в 24% на бенчмарке
- Анализ данных инлайнинга
  - 4249 - заинлайнено до регрессии
  - 3184 - в “проблемном” билде



# Инлайнинг, которого мы так ждали



# Инлайнинг, которого мы так ждали

- *Вызываемый метод большой*





# Инлайнинг, которого мы так ждали

- *Вызываемый метод большой*
- *Уровень вложенности большой*



# Инлайнинг, которого мы так ждали

- *Вызываемый метод большой*
  - *Уровень вложенности большой*
- } Можно настроить





# Инлайнинг, которого мы так ждали

- *Вызываемый метод большой*
  - *Уровень вложенности большой*
  - *Вызываемый метод обрабатывает исключения ?  
(InlineMethodsWithExceptionHandlers)*
- } Можно настроить



# Инлайнинг, которого мы так ждали

- *Вызываемый метод большой*
  - *Уровень вложенности большой*
- } Можно настроить
- *Вызываемый метод обрабатывает исключения ?*  
(`InlineMethodsWithExceptionHandlers`)
  - *Вызываемый метод `synchronized`*  
(`InlineSynchronizedMethods`)





# Инлайнинг, которого мы так ждали

- *Вызываемый метод большой*
  - *Уровень вложенности большой*
- } Можно настроить
- *Вызываемый метод обрабатывает исключения ?  
(InlineMethodsWithExceptionHandlers)*
  - *Вызываемый метод synchronized  
(InlineSynchronizedMethods)*
  - *Класс с вызываемым методом не инициализирован*



# Инлайнинг, которого мы так ждали

- *Вызываемый метод большой*
  - *Уровень вложенности большой*
- } Можно настроить
- *Вызываемый метод обрабатывает исключения ?  
(InlineMethodsWithExceptionHandlers)*
  - *Вызываемый метод synchronized  
(InlineSynchronizedMethods)*
  - *Класс с вызываемым методом не инициализирован*
  - *Несбалансирование мониторов*





# Инлайнинг, которого мы так ждали

- *Вызываемый метод большой*
  - *Уровень вложенности большой*
- } Можно настроить
- *Вызываемый метод обрабатывает исключения ?  
(InlineMethodsWithExceptionHandlers)*
  - *Вызываемый метод synchronized  
(InlineSynchronizedMethods)*
  - *Класс с вызываемым методом не инициализирован*
  - *Несбалансированные мониторы*
  - *Содержит байткод jsr (!)*



# Как инлайнинг может навредить?



# Как инлайнинг может навредить?

```
class A {  
    boolean Cond;  
    A (bool v) { Cond = v; }  
  
    int outerMethod(B b,C c)  
    {  
        int a=2;  
        if (Cond) {  
            a+=b.inner1();  
        }  
        else  
            a+=c.inner2();  
        return a;  
    }  
}
```

# Как инлайнинг может навредить?

```
class A {  
    boolean Cond;  
    A (bool v) { Cond = v; }  
  
    int outerMethod(B b,C c)  
    {  
        int a=2;  
        if (Cond) {  
            a+=b.inner1();  
        }  
        else  
            a+=c.inner2();  
        return a;  
    }  
}
```

```
class B {  
    int inner1() {  
        ...;  
    }  
}
```



# Как инлайнинг может навредить?

```
class A {  
    boolean Cond;  
    A (bool v) { Cond = v; }  
  
    int outerMethod(B b, C c)  
    {  
        int a=2;  
        if (Cond) {  
            a+=b.inner1();  
        }  
        else  
            a+=c.inner2();  
        return a;  
    }  
}
```

```
class B {  
    int inner1() {  
        ...;  
    }  
}
```

```
class C {  
    int inner2() {  
        ... .  
    }  
}
```

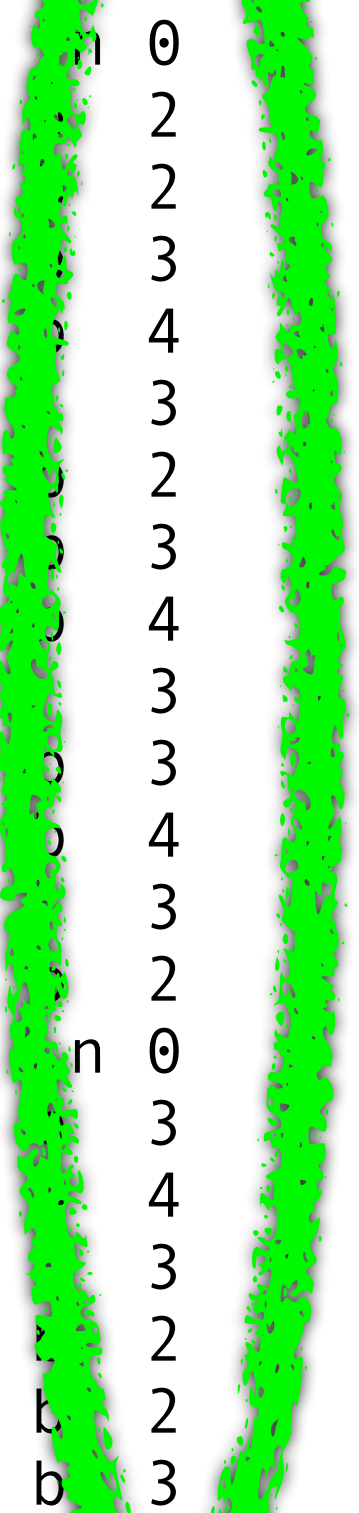
# Уровень компиляции?



# Уровень КОМПИЛЯЦИИ?

495	7	n	0	jdk.internal.reflect.Reflection::getCallerClass (native) (static)
495	8	b	2	java.util.Properties::getProperty (49 bytes)
497	9	b	2	java.util.concurrent.ConcurrentHashMap::get (162 bytes)
500	10	b	3	java.lang.String::hashCode (48 bytes)
503	11	b	4	java.lang.String::hashCode (48 bytes)
507	10		3	java.lang.String::hashCode (48 bytes) made not entrant
507	12	b	2	java.lang.StringLatin1::hashCode (42 bytes)
509	13	b	3	java.lang.Boolean::<clinit> (31 bytes)
510	14	b	4	java.lang.Boolean::<clinit> (31 bytes)
510	13		3	java.lang.Boolean::<clinit> (31 bytes) made not entrant
511	15	b	3	java.lang.Boolean::<init> (10 bytes)
511	16	b	4	java.lang.Boolean::<init> (10 bytes)
513	15		3	java.lang.Boolean::<init> (10 bytes) made not entrant
513	17	b	2	java.lang.Object::<init> (1 bytes)
513	18	n	0	java.lang.Class::getPrimitiveClass (native) (static)
514	19	b	3	java.lang.Boolean::parseBoolean (19 bytes)
514	20	b	4	java.lang.Boolean::parseBoolean (19 bytes)
516	19		3	java.lang.Boolean::parseBoolean (19 bytes) made not entrant
516	21	b	2	java.lang.String::isLatin1 (19 bytes)
517	22	b	2	java.lang.Integer::parseInt (259 bytes)
533	23	b	3	java.lang.Character::<clinit> (25 bytes)

# Уровень КОМПИЛЯЦИИ?



495	7	n 0	jdk.internal.reflect.Reflection::getCallerClass (native) (static)
495	8	n 2	java.util.Properties::getProperty (49 bytes)
497	9	n 2	java.util.concurrent.ConcurrentHashMap::get (162 bytes)
500	10	n 3	java.lang.String::hashCode (48 bytes)
503	11	n 4	java.lang.String::hashCode (48 bytes)
507	10	n 3	java.lang.String::hashCode (48 bytes) made not entrant
507	12	n 2	java.lang.StringLatin1::hashCode (42 bytes)
509	13	n 3	java.lang.Boolean::<clinit> (31 bytes)
510	14	n 4	java.lang.Boolean::<clinit> (31 bytes)
510	13	n 3	java.lang.Boolean::<clinit> (31 bytes) made not entrant
511	15	n 3	java.lang.Boolean::<init> (10 bytes)
511	16	n 4	java.lang.Boolean::<init> (10 bytes)
513	15	n 3	java.lang.Boolean::<init> (10 bytes) made not entrant
513	17	n 2	java.lang.Object::<init> (1 bytes)
513	18	n 0	java.lang.Class::getPrimitiveClass (native) (static)
514	19	n 3	java.lang.Boolean::parseBoolean (19 bytes)
514	20	n 4	java.lang.Boolean::parseBoolean (19 bytes)
516	19	n 3	java.lang.Boolean::parseBoolean (19 bytes) made not entrant
516	21	n 2	java.lang.String::isLatin1 (19 bytes)
517	22	n 2	java.lang.Integer::parseInt (259 bytes)
533	23	n 3	java.lang.Character::<clinit> (25 bytes)



# Уровни компиляции

# Уровни КОМПИЛЯЦИИ

**4 - C2 со всеми оптимизациями**

**3 - C1 с максимальным  
профилированием**

**2 - C1 с минимальным  
профилированием**

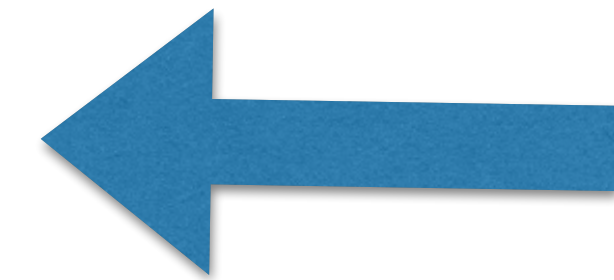
**1 - C1 без профилирования**

**0 - Интерпретатор**



# Уровни КОМПИЛЯЦИИ

**4 - C2 со всеми оптимизациями**



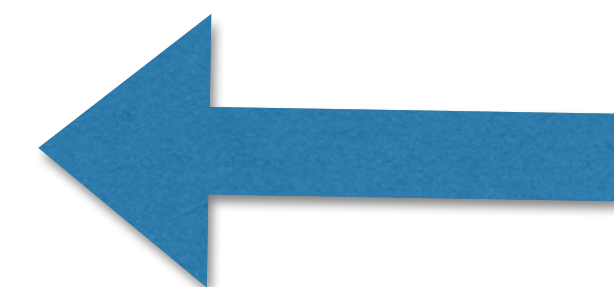
Быстро

**3 - C1 с максимальным  
профилированием**

**2 - C1 с минимальным  
профилированием**

**1 - C1 без профилирования**

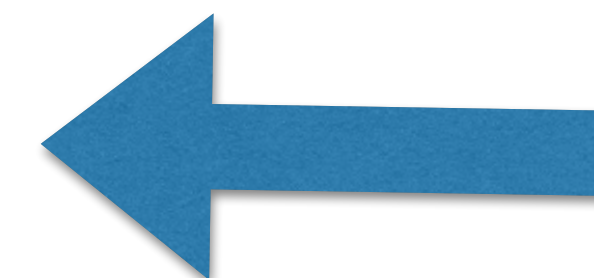
**0 - Интерпретатор**



Медленно

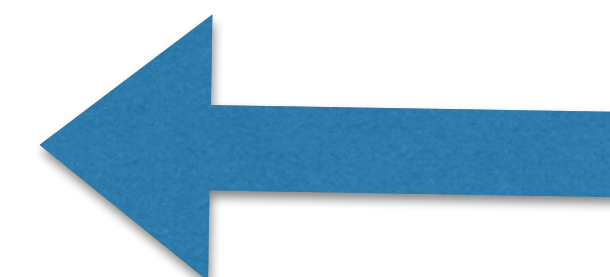
# Уровни КОМПИЛЯЦИИ

**4 - C2 со всеми оптимизациями**



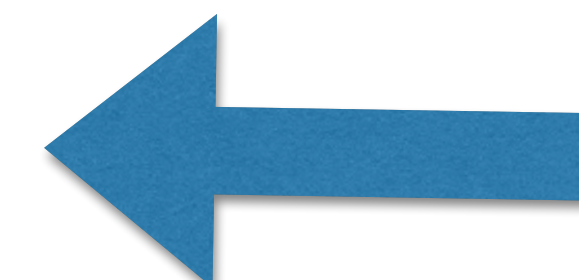
Быстро

**3 - C1 с максимальным  
профилированием**



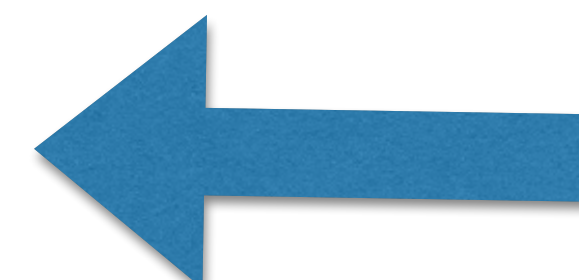
Тоже  
медленно

**2 - C1 с минимальным  
профилированием**



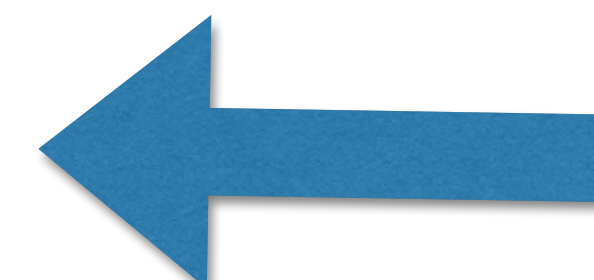
Средне

**1 - C1 без профилирования**



Побыстрее

**0 - Интерпретатор**



Медленно



# Уровни компиляции



# Уровни КОМПИЛЯЦИИ





# Уровни КОМПИЛЯЦИИ



# Уровни КОМПИЛЯЦИИ





# Уровни КОМПИЛЯЦИИ



# Уровни компиляции





# Уровни компиляции



# Уровни КОМПИЛЯЦИИ





# Уровни компиляции



# Уровни компиляции





# Деоптимизации бывают так как...



# Деоптимизации бывают так как...

- ... компилятор делает “оптимистичные” предположения





# Деоптимизации бывают так как...

- ... компилятор делает “оптимистичные” предположения
- а АОТы так не могут



# Деоптимизации бывают так как...

- ... компилятор делает “оптимистичные” предположения
- а АОТы так не могут
- ... невозможен анализ “всей программы”





Не все оптимизации спекулятивные

# Не все оптимизации спекулятивные

## Детерминированные

- Constant propagation
- Loop invariant
- ...

## Спекулятивные с механизмом отката

- Bias locking
- NUMA-aware allocation
- TSX transactions
- Uncommon trap
- ...

## Спекулятивные с немедленной остановкой

- CHA invalidation
- Final fields modified
- ...



# Невидимый код

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```

# Невидимый код

- Проверки из JVM

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```



# Невидимый код

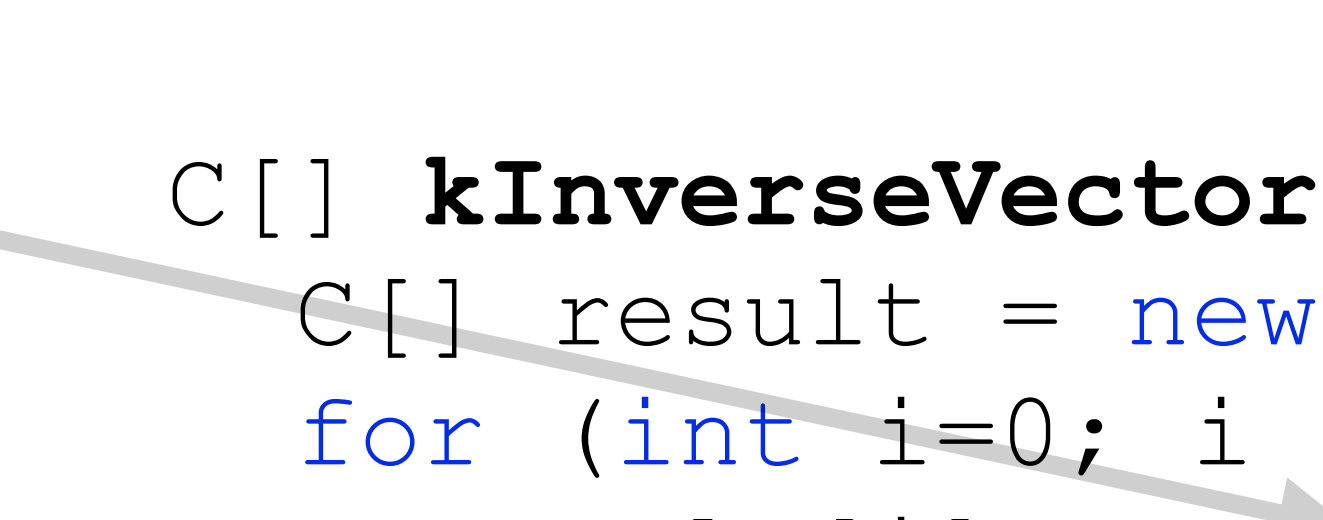
- Проверки из JVM
- Обнуление

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```

# Невидимый код

- Проверки из JVM
- Обнуление

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```

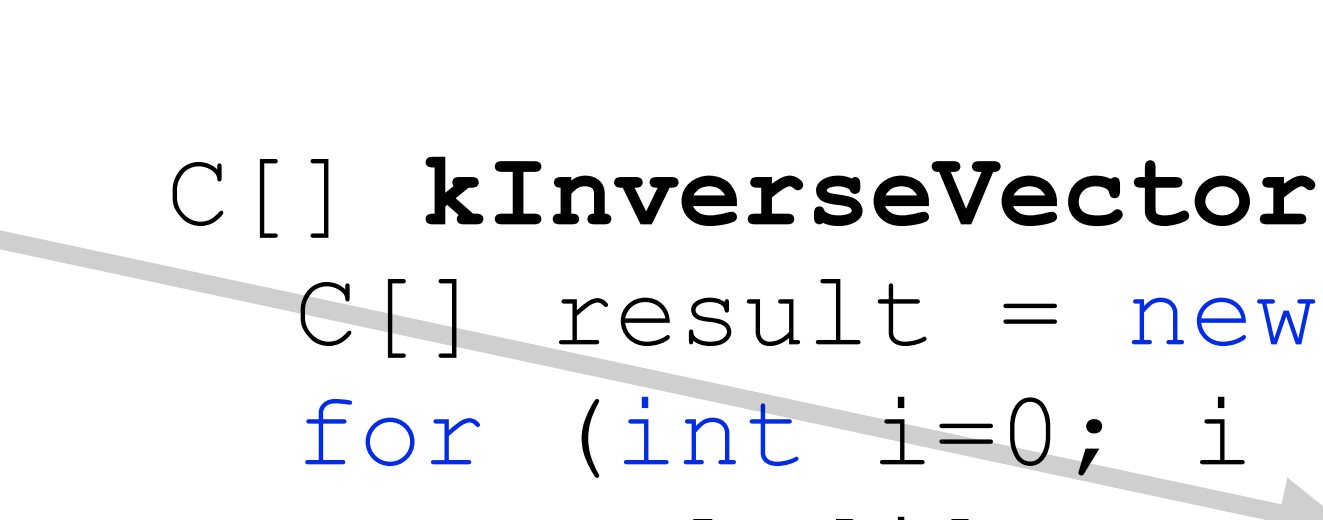




# Невидимый код

- Проверки из JVMMS
- Обнуление
- Выход за пределы

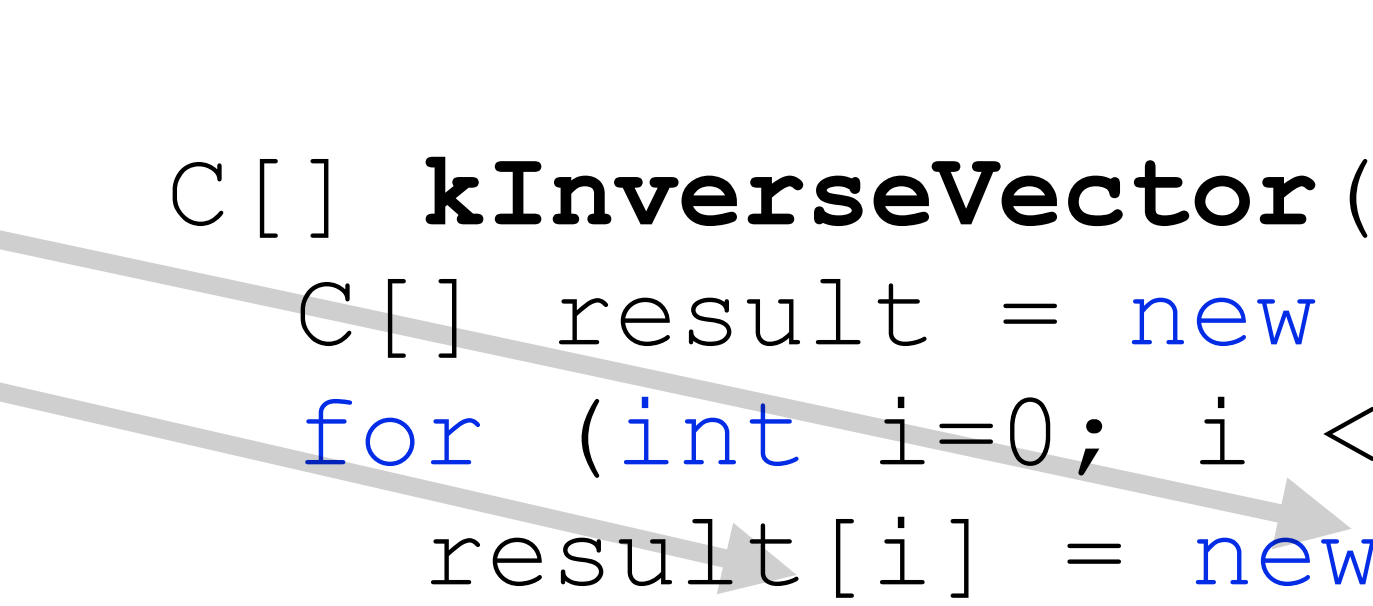
```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```



# Невидимый код

- Проверки из JVMMS
- Обнуление
- Выход за пределы

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```

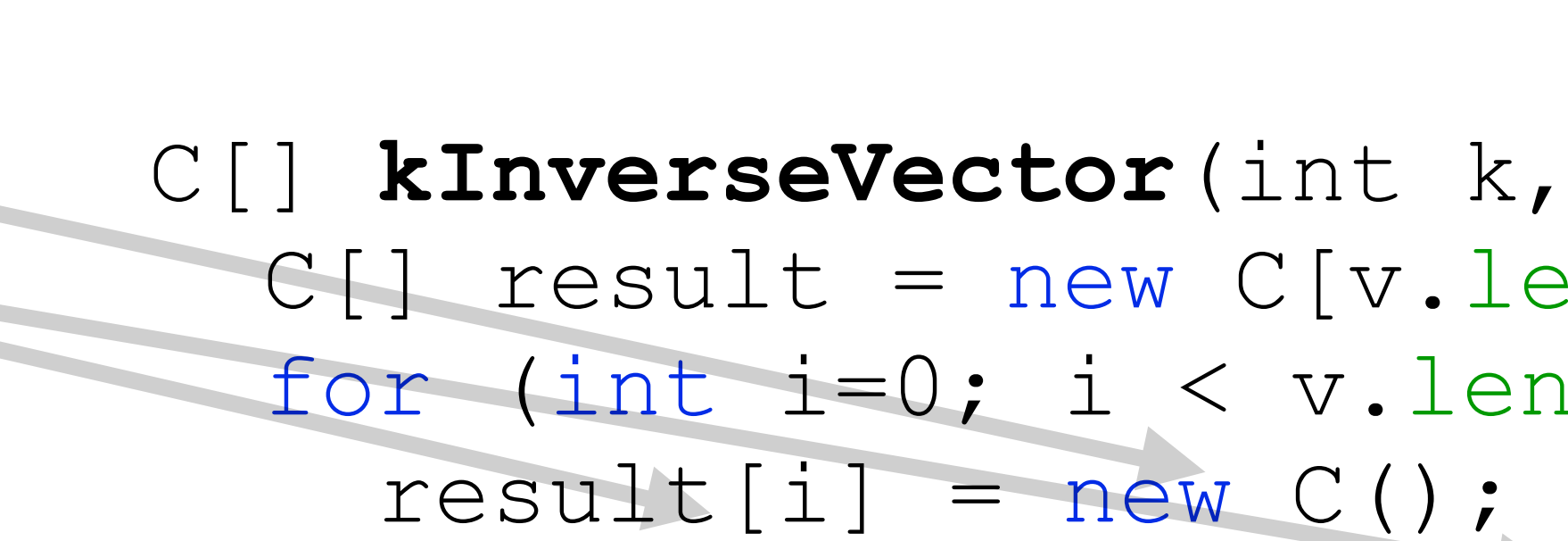




# Невидимый код

- Проверки из JVMMS
- Обнуление
- Выход за пределы

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```



# Невидимый код

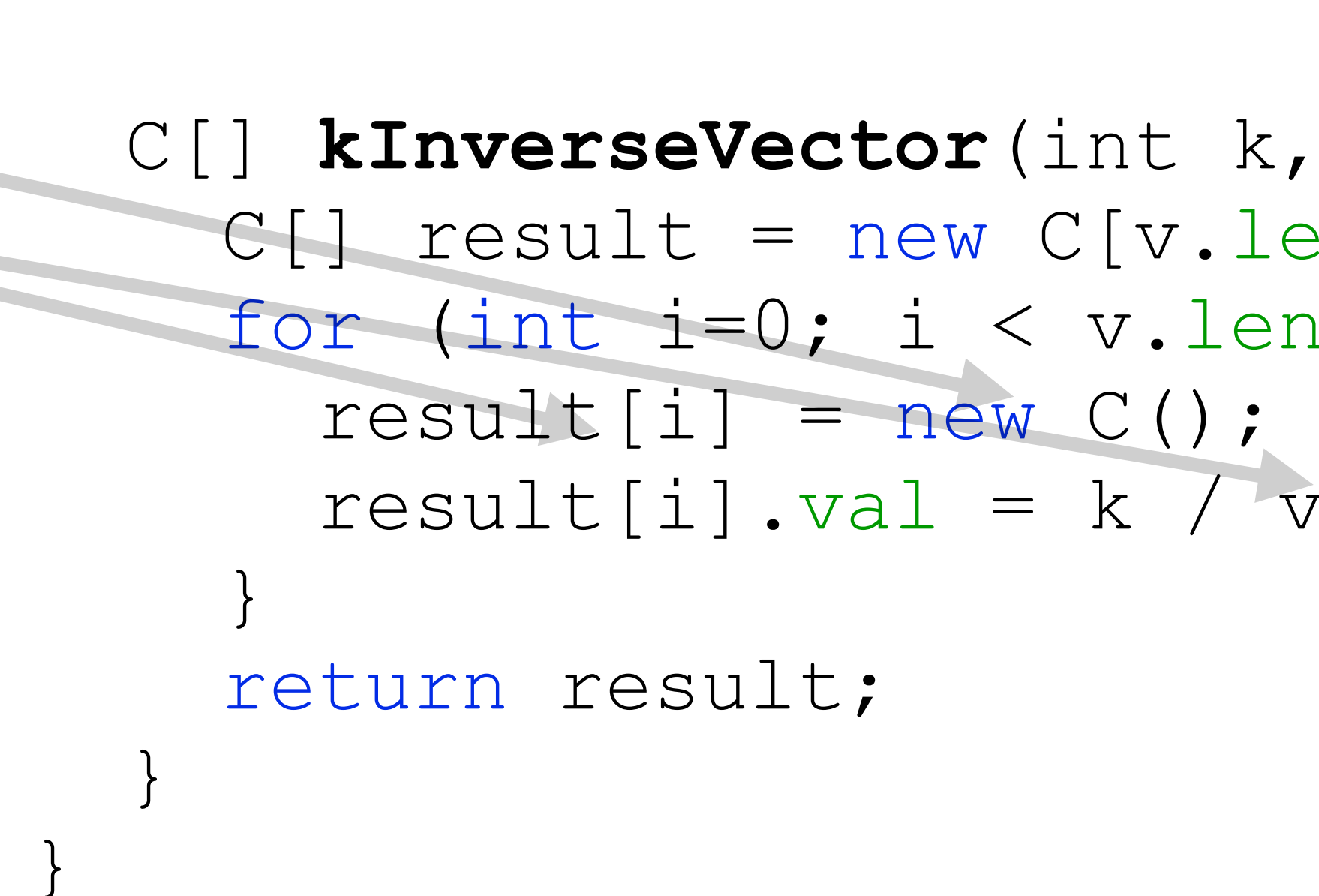
- Проверки из JVMMS

- Обнуление

- Выход за пределы

- Деление на 0

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```





# Невидимый код

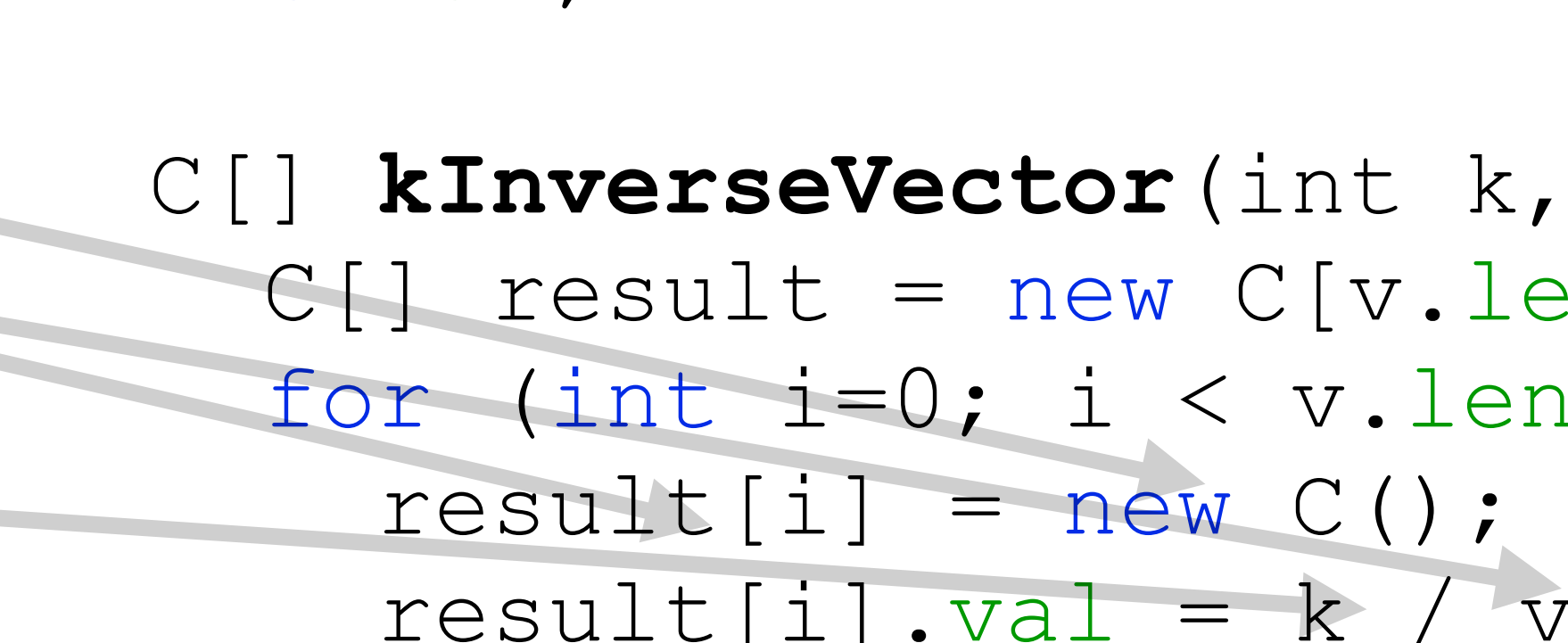
- Проверки из JVM

- Обнуление

- Выход за пределы

- Деление на 0

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```



# Невидимый код

- Проверки из JVM

- Обнуление

- Выход за пределы

- Деление на 0

- 0-ссылки

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```



# Невидимый код

- Проверки из JVM

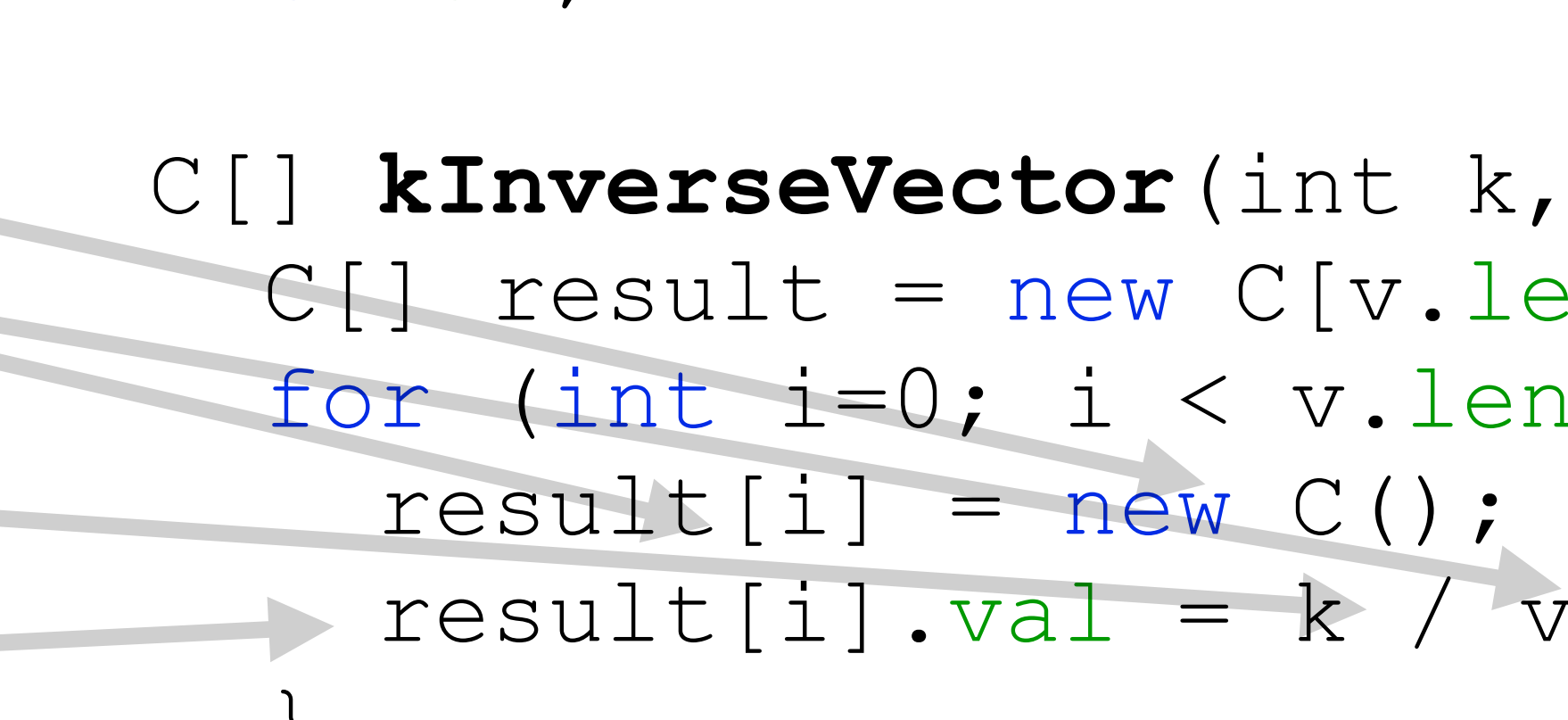
- Обнуление

- Выход за пределы

- Деление на 0

- 0-ссылки

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```



# Невидимый код

- Проверки из JVM

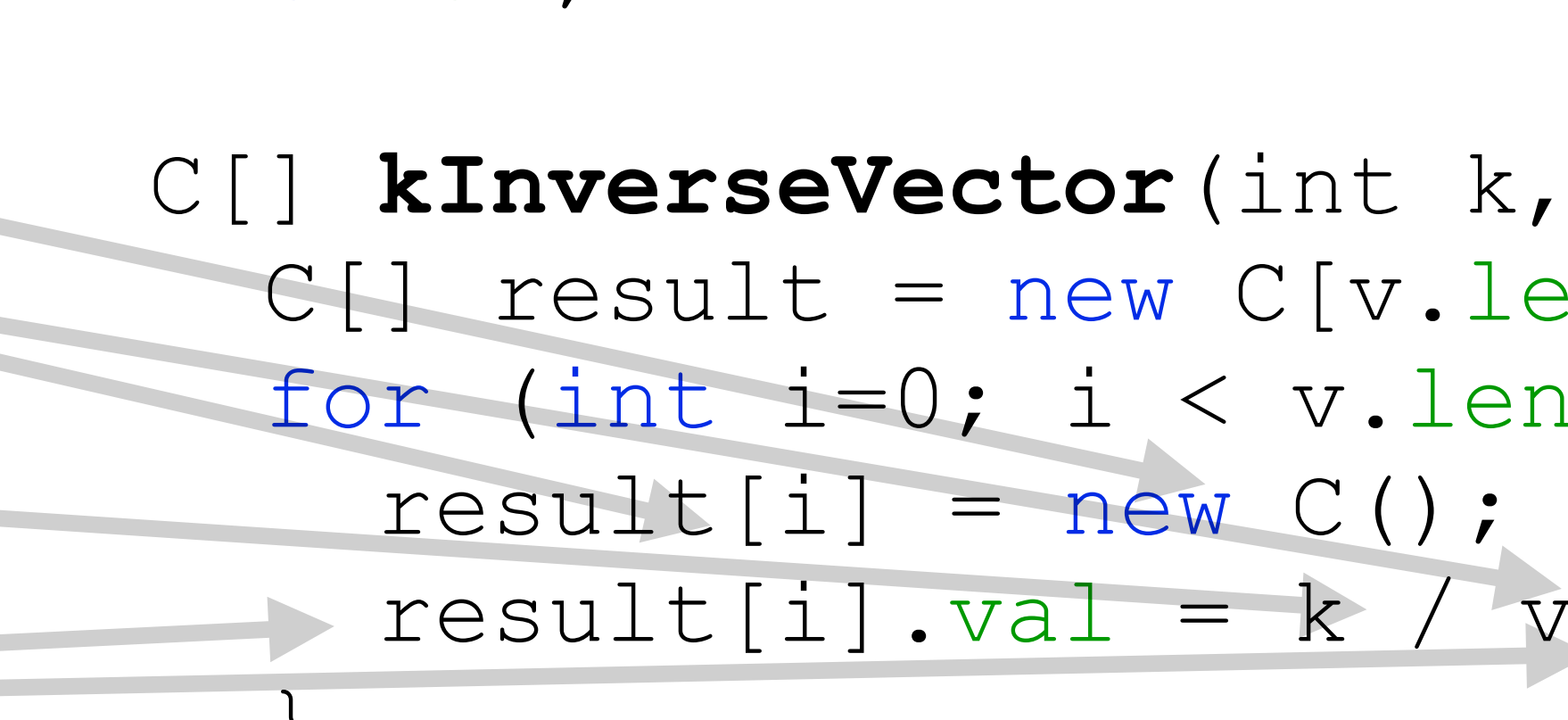
- Обнуление

- Выход за пределы

- Деление на 0

- 0-ссылки

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```





# Невидимый код

- Проверки из JVM

- Обнуление

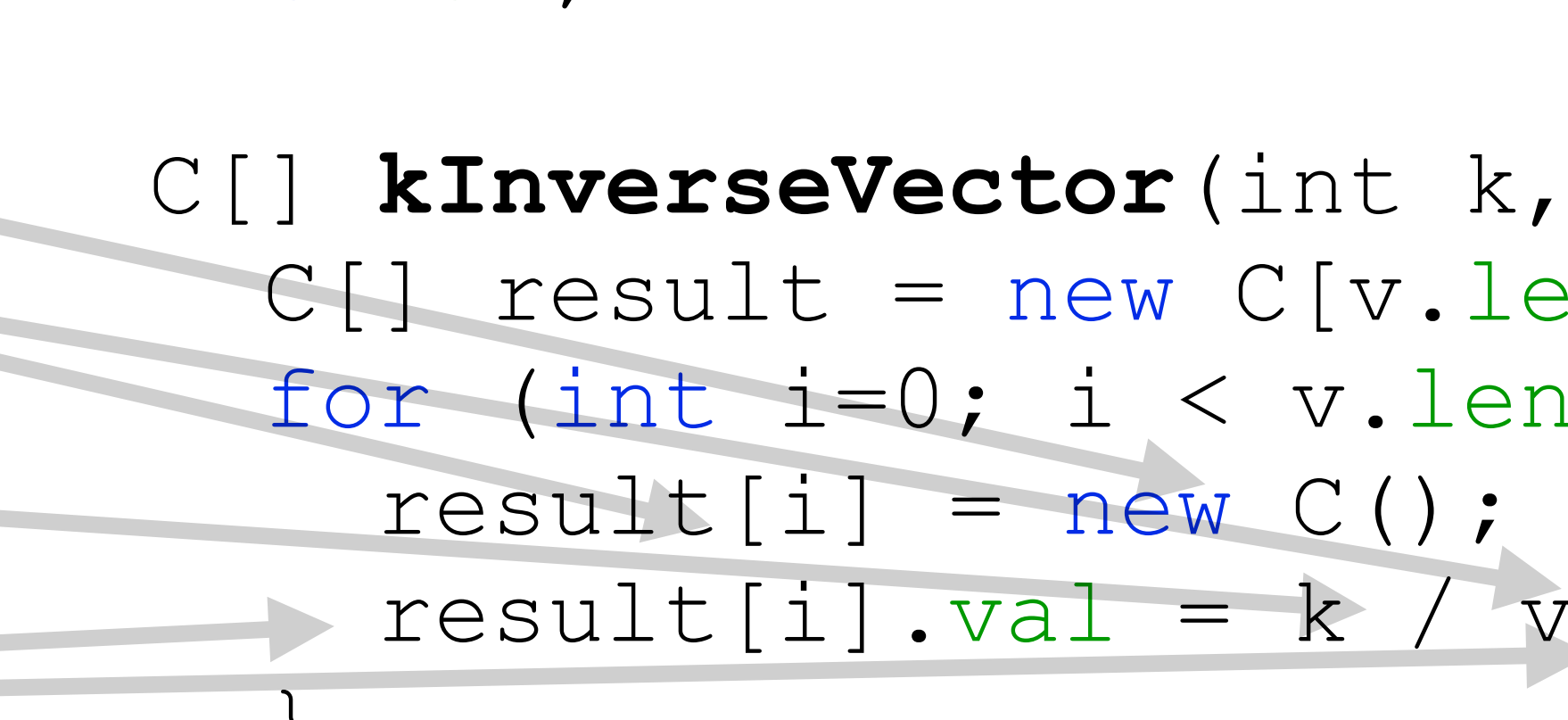
- Выход за пределы

- Деление на 0

- 0-ссылки

- Проверка типов

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```



# Невидимый код

- Проверки из JVM

- Обнуление

- Выход за пределы

- Деление на 0

- 0-ссылки

- Проверка типов

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```



# Невидимый код

- Проверки из JVM

- Обнуление

- Выход за пределы

- Деление на 0

- 0-ссылки

- Проверка типов

- ....

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i] = new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```

# Причины деоптимизации с детализацией источника с точностью до байткода



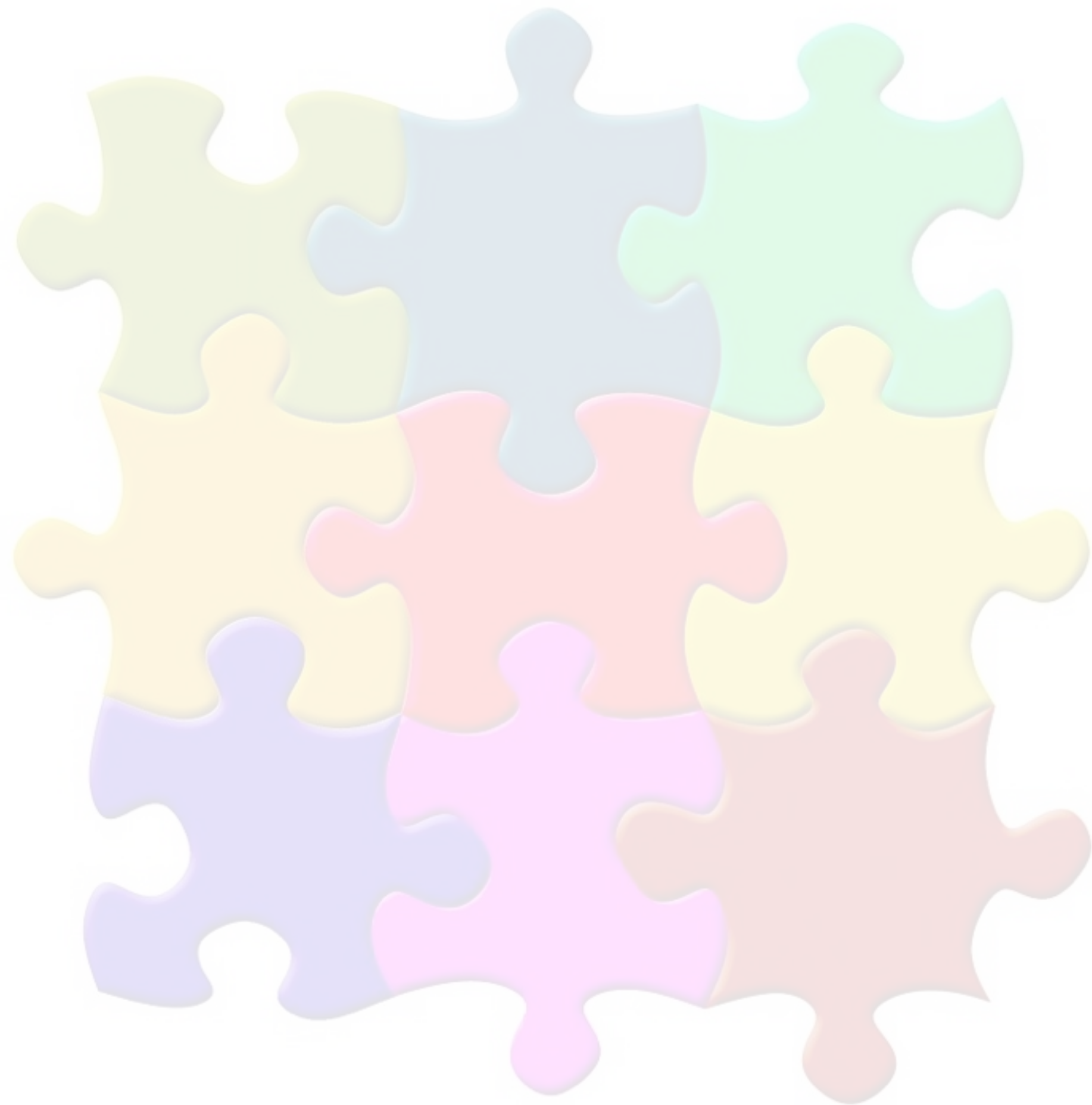
# Причины деоптимизации с детализацией источника с точностью до байткода

- Null Check (null object or div 0)
- Null Assert
- Range Check (OOB)
- Class Check (Unexpected class)
- Array Check (Unexpected array class)
- Intrinsic operand
- Bimorphic inlining failed





# Причины деоптимизации с детализацией источника с точностью до метода



# Причины деоптимизации с детализацией источника с точностью до метода

- Unloaded class
- Uninitialized class
- Unreached code
- Unhandled exception
- Unexpected Constraint
- Div0 check
- Age (tier threshold reached)

- Predicate failed
- Loop limit check
- Speculate class check
- Speculate Null Check
- Rm state change
- Unstable if
- Reason unstable fused if

# Uncommon trap. Что делаем?

- Случайность?
  - Паттерн сменился?
  - А что со старым кодом делать?
  - Надо ли компилировать снова?
- 

- Что будет происходить при следующих вызовах?

```
class C {  
    int val;  
  
    C[] kInverseVector(int k, C[] v) {  
        C[] result = new C[v.length];  
        for (int i=0; i < v.length; i++) {  
            result[i]=new C();  
            result[i].val = k / v[i].val ;  
        }  
        return result;  
    }  
}
```



# API для настройки компилятора





# API для настройки компилятора



[https://commons.wikimedia.org/wiki/File:Airbus\\_A380\\_cockpit.jpg](https://commons.wikimedia.org/wiki/File:Airbus_A380_cockpit.jpg)



# I. Старый добрый CompileCommand





# I. Старый добрый CompileCommand

- Можно указать статически
  - Через `-XX:HotspotCompile`
  - `.hotspot_compiler` or  
`-XX:CompileCommandFile=/path/to/thefile`

# I. Старый добрый CompileCommand

- Можно указать статически
  - Через `-XX:HotspotCompile`
  - `.hotspot_compiler` or  
`-XX:CompileCommandFile=/path/to/thefile`
- Можно указать динамически
  - `jcmd Compiler.directives_add`

# I. Старый добрый CompileCommand

- Можно указать статически
  - Через `-XX:HotspotCompile`
  - `.hotspot_compiler` or `-XX:CompileCommandFile=/path/to/thefile`
- Можно указать динамически
  - `jcmd Compiler.directives_add`
- Синтакс:  
команда пакет/Класс метод

```
exclude    java/lang/Thread setPriority  
dontinline java/lang/String charAt
```



# I. Старый добрый CompileCommand

- Можно указать статически
  - Через -XX:HotspotCompile
  - .hotspot\_compiler or  
-XX:CompileCommandFile=/path/to/thefile
- Можно указать динамически
  - jcmd Compiler.directives\_add
- Синтакс:  
команда пакет/Класс метод
- Команды:
  - dontinline **или** exclude **или** excludec2 **или** excludec1

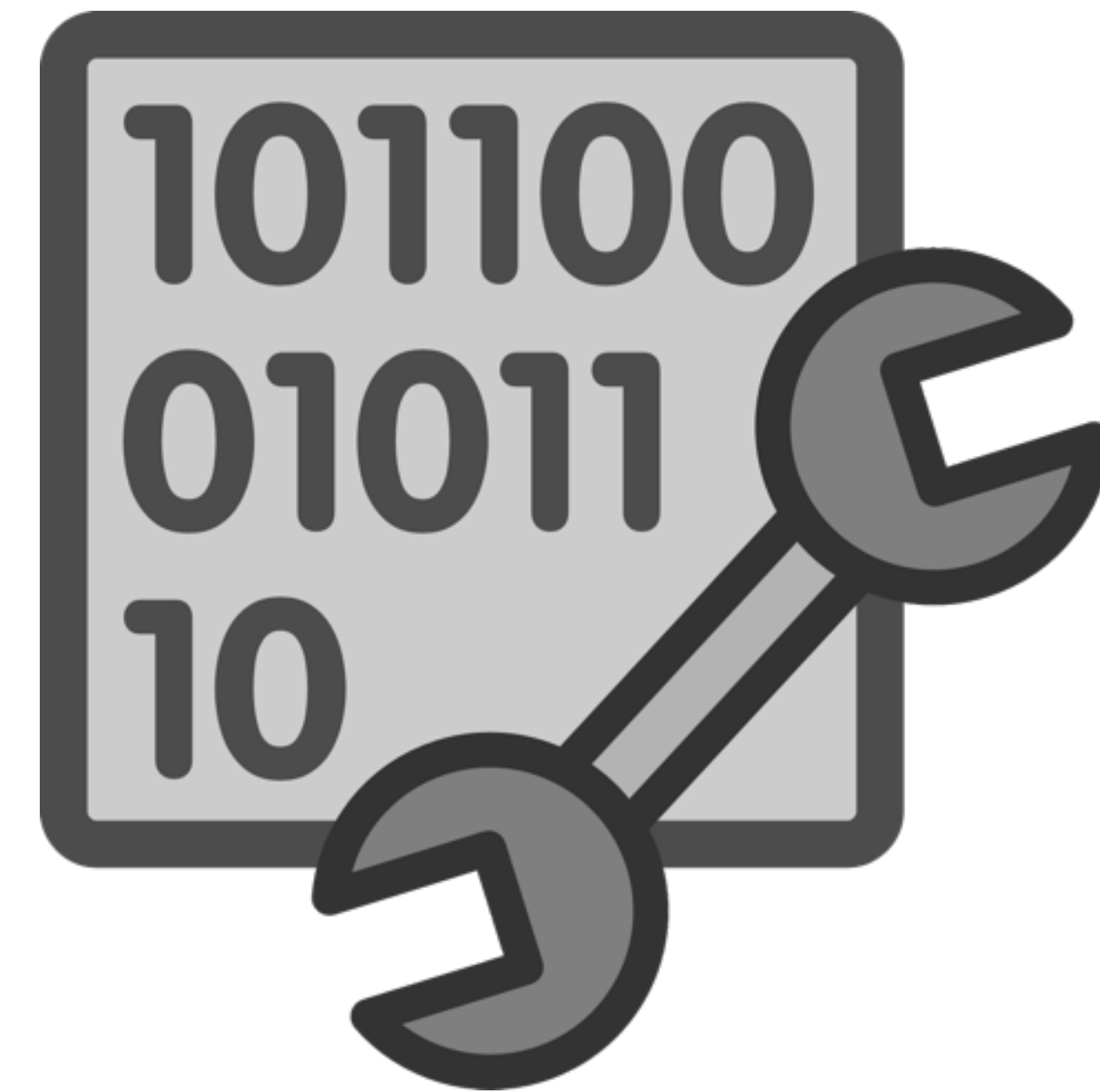
```
exclude    java/lang/Thread setPriority  
dontinline java/lang/String charAt
```

# II. Compiler Control в Java 9

```
{  
  // паттерны  
  match: ["steve.*", "alex.*"]  
  
  c2: {  
    Enable: false      // Игнорируем только для c2.  
  }  
  
  // Для обоих компиляторов  
  // “+” значит принудительно, “-” значит запретить  
  inline : [ "+java/util.*", "-com/sun.*"],  
  PrintInlining: true  
}
```



# Директивы для обоих компиляторов





# Директивы для обоих компиляторов

Enable

bool Exclude

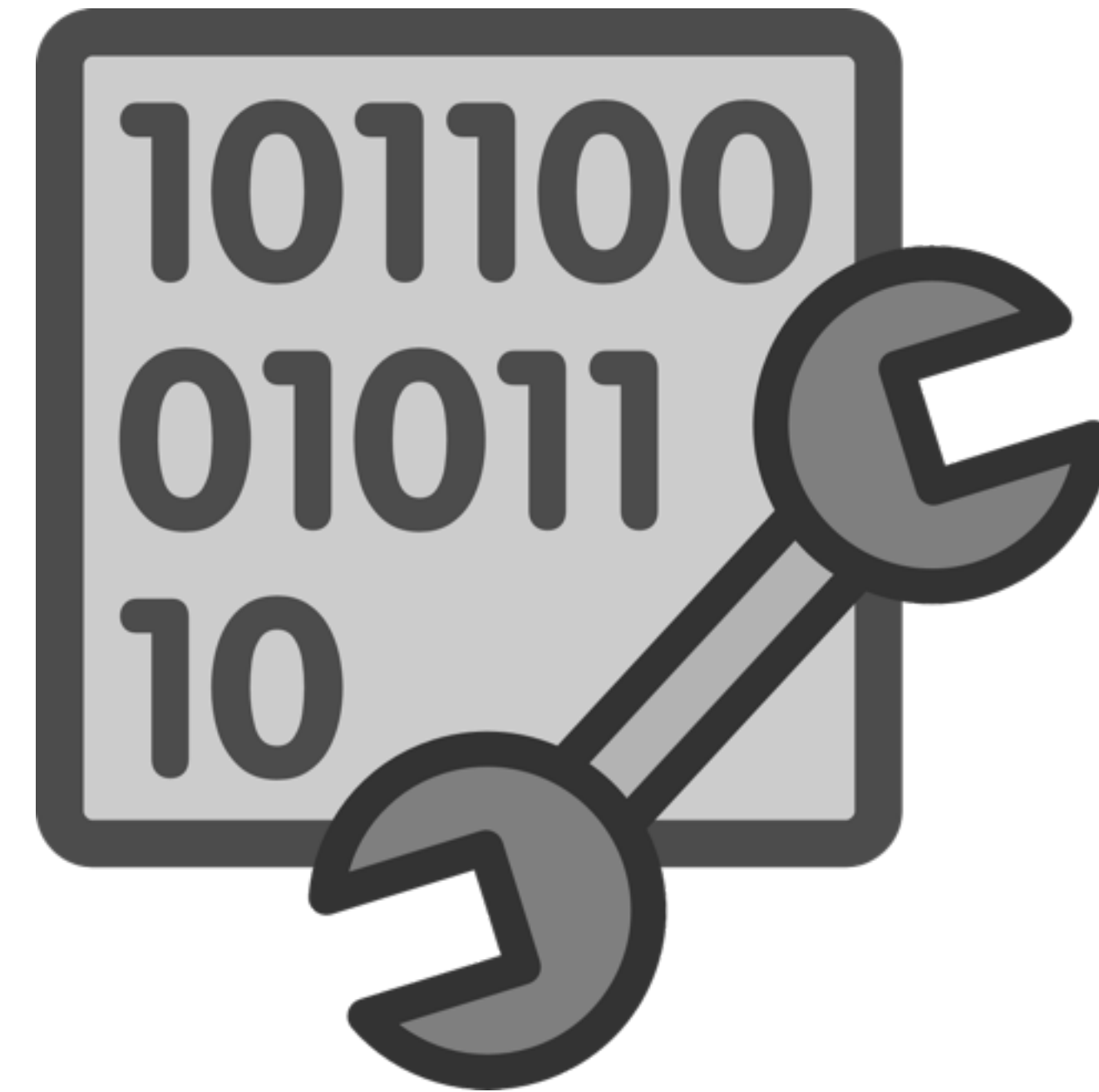
bool Inline

bool BreakAtExecute

bool BreakAtCompile

bool Log

bool PrintAssembly



# Директивы для обоих компиляторов

Enable

bool Exclude

bool Inline

bool BreakAtExecute

bool BreakAtCompile

bool Log

bool PrintAssembly

bool PrintInlining

bool PrintNMethods

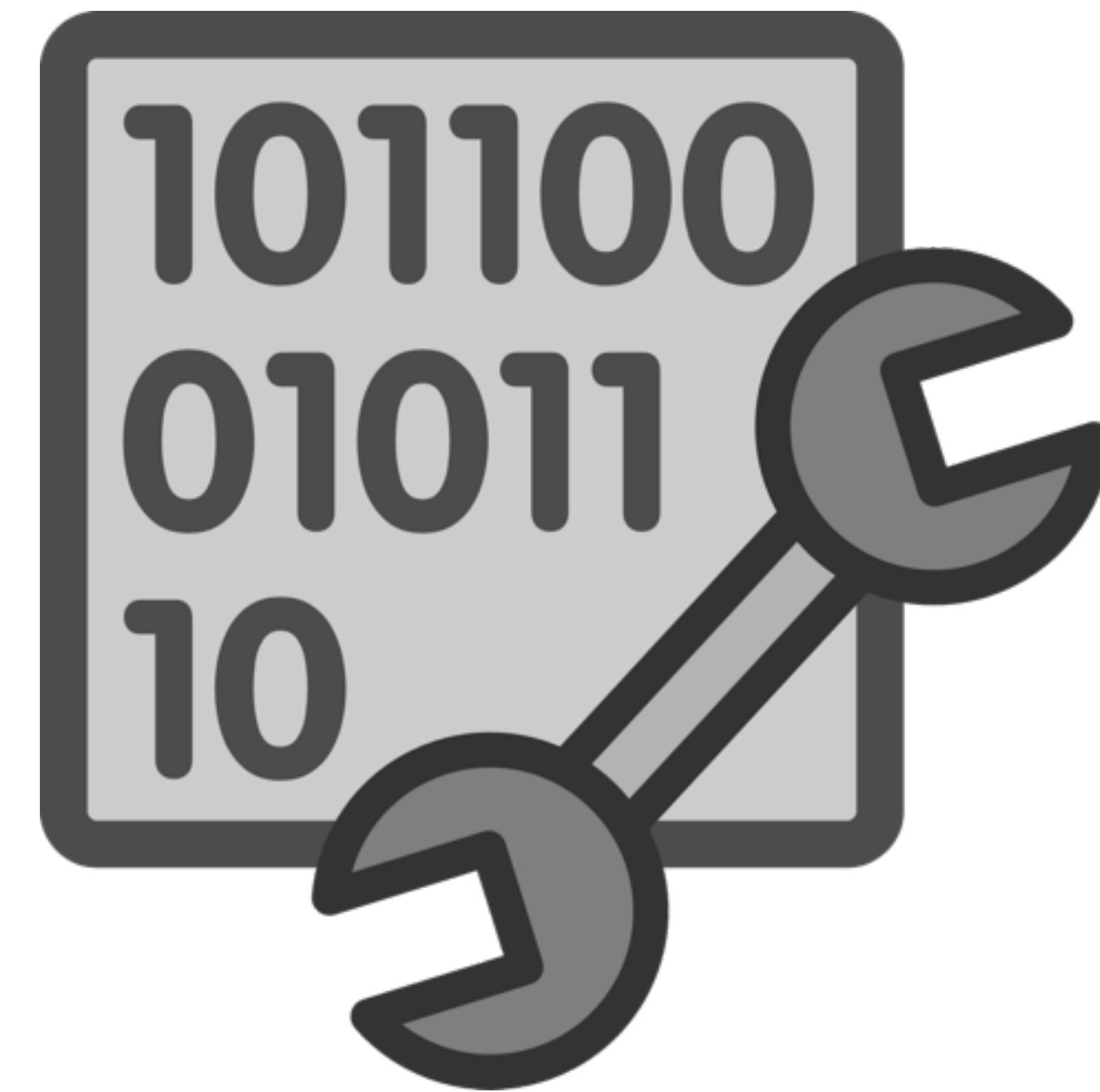
bool ReplayInline

bool DumpReplay

bool DumpInline

bool

CompilerDirectivesIgnoreCompileCommands



# Директивы для C2





# Директивы для C2

BlockLayoutByFrequency

bool raceOptoPipelining

bool Vectorize

bool VectorizeDebug

intx MaxNodeLimit

intx DisableIntrinsics

# Директивы для C2

BlockLayoutByFrequency

bool raceOptoPipelining

bool Vectorize

bool VectorizeDebug

intx MaxNodeLimit

intx DisableIntrinsics

bool PrintOptoAssembly

bool PrintIntrinsics

bool TraceOptoOutput

bool TraceSpilling

bool CloneMapDebug

bool IGVPrintLevel

# III. java.lang.Compiler



# III. java.lang.Compiler

```
{
    Compiler.enable(); //
    Compiler.command("{com.mycompany.*}(compile)");
    System.out.println("Now let's wait till compilation is done");
    Compiler.command("waitOnCompilationQueue");
    System.out.println("Compilation is complete");
    Compiler.disable(); // turn the compiler off
}
```

# III. java.lang.Compiler

- OpenJDK / Oracle - нет поддержки

```
{  
    Compiler.enable(); //  
    Compiler.command("{com.mycompany.*}(compile)");  
    System.out.println("Now let's wait till compilation is done");  
    Compiler.command("waitOnCompilationQueue");  
    System.out.println("Compilation is complete");  
    Compiler.disable(); // turn the compiler off  
}
```

# III. java.lang.Compiler

- OpenJDK / Oracle - нет поддержки
- IBM J9 поддерживают  
[http://www.ibm.com/support/knowledgecenter/en/SSSTCZ\\_2.0.0/com.ibm.rt.doc.20/realtime/rt\\_jit.html](http://www.ibm.com/support/knowledgecenter/en/SSSTCZ_2.0.0/com.ibm.rt.doc.20/realtime/rt_jit.html)

```
{  
    Compiler.enable(); //  
    Compiler.command("{com.mycompany.*}(compile)");  
    System.out.println("Now let's wait till compilation is done");  
    Compiler.command("waitOnCompilationQueue");  
    System.out.println("Compilation is complete");  
    Compiler.disable(); // turn the compiler off  
}
```



# III. java.lang.Compiler

- OpenJDK / Oracle - нет поддержки
- IBM J9 поддерживают  
[http://www.ibm.com/support/knowledgecenter/en/SSSTCZ\\_2.0.0/com.ibm.rt.doc.20/realtime/rt\\_jit.html](http://www.ibm.com/support/knowledgecenter/en/SSSTCZ_2.0.0/com.ibm.rt.doc.20/realtime/rt_jit.html)
- У Zing-а немного другой синтаксис  
[http://docs.azul.com/zing/Zing\\_UserGuide/#Zing\\_UserGuide/Zing\\_AT\\_ReadyNow\\_EnsureCriticalMethodsareCompiled\\_CompilerAPI.htm](http://docs.azul.com/zing/Zing_UserGuide/#Zing_UserGuide/Zing_AT_ReadyNow_EnsureCriticalMethodsareCompiled_CompilerAPI.htm)

```
{  
    Compiler.enable(); //  
    Compiler.command("{com.mycompany.*}(compile)");  
    System.out.println("Now let's wait till compilation is done");  
    Compiler.command("waitOnCompilationQueue");  
    System.out.println("Compilation is complete");  
    Compiler.disable(); // turn the compiler off  
}
```

# III. java.lang.Compiler

- OpenJDK / Oracle - нет поддержки
- IBM J9 поддерживают  
[http://www.ibm.com/support/knowledgecenter/en/SSSTCZ\\_2.0.0/com.ibm.rt.doc.20/realtime/rt\\_jit.html](http://www.ibm.com/support/knowledgecenter/en/SSSTCZ_2.0.0/com.ibm.rt.doc.20/realtime/rt_jit.html)
- У Zing-а немного другой синтаксис  
[http://docs.azul.com/zing/Zing\\_UserGuide/#Zing\\_UserGuide/Zing\\_AT\\_ReadyNow\\_EnsureCriticalMethodsareCompiled\\_CompilerAPI.htm](http://docs.azul.com/zing/Zing_UserGuide/#Zing_UserGuide/Zing_AT_ReadyNow_EnsureCriticalMethodsareCompiled_CompilerAPI.htm)
- Oracle ~~хотел~~ вынести этот API из 9ки  
<https://bugs.openjdk.java.net/browse/JDK-4285505>

```
{  
    Compiler.enable(); //  
    Compiler.command("{com.mycompany.*}(compile)");  
    System.out.println("Now let's wait till compilation is done");  
    Compiler.command("waitOnCompilationQueue");  
    System.out.println("Compilation is complete");  
    Compiler.disable(); // turn the compiler off  
}
```

# IV. Java-Level JVM Compiler Interface



# IV. Java-Level JVM Compiler Interface ?



## *Заметки на полях*



*Заметки на полях*

*JEP-243 Java-Level JVM Compiler Interface*

*Заметки на полях*

*JEP-243 Java-Level JVM Compiler Interface*

- *Работает только с Graal*

## *Заметки на полях*

### *JEP-243 Java-Level JVM Compiler Interface*

- *Работает только с Graal*
- *Нет поддержки C2 (и не планируется)*



# IV. ReadyNow

# IV. ReadyNow

- Напоминает PGO-оптимизацию gcc

# IV. ReadyNow

- Напоминает PGO-оптимизацию gcc
- Для регулярно используемых приложение



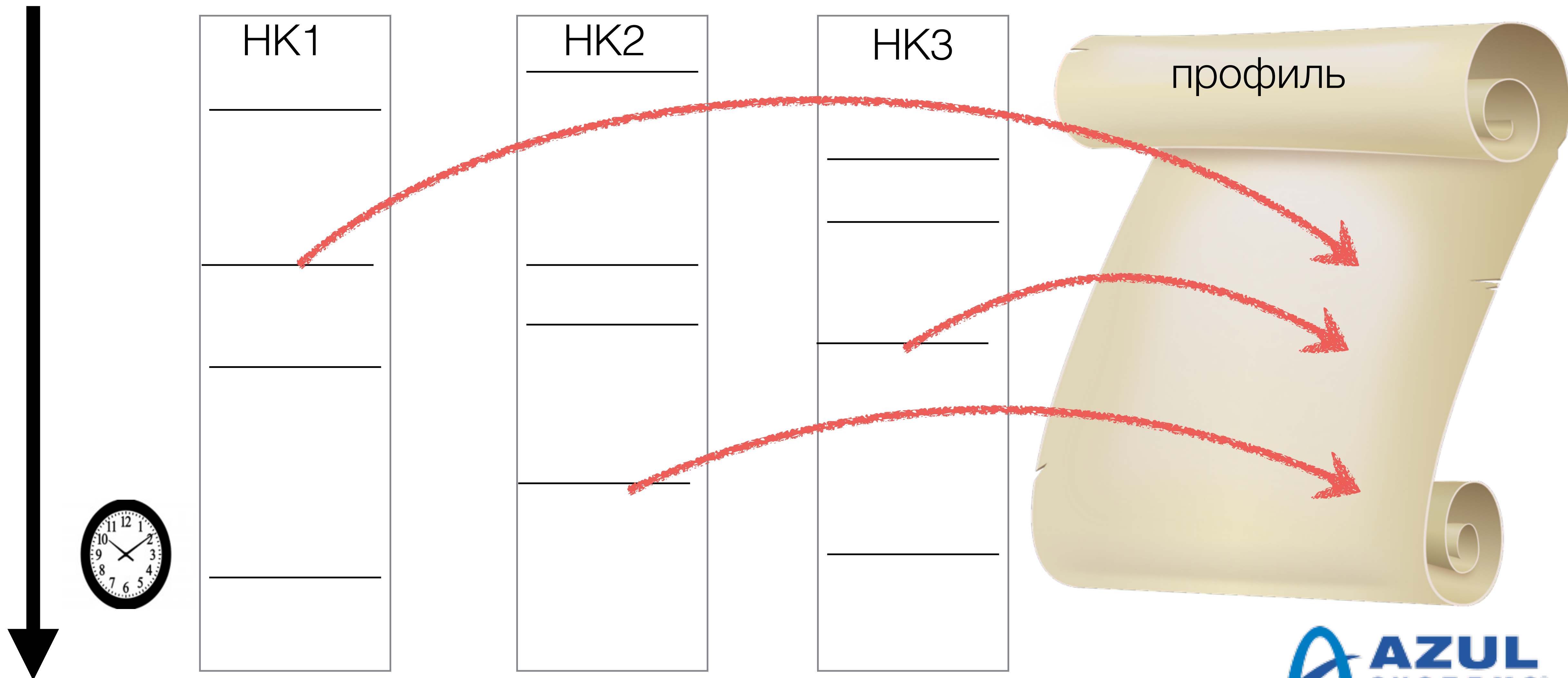
# IV. ReadyNow

- Напоминает PGO-оптимизацию gcc
- Для регулярно используемых приложение
- При первом запуске приложения - профиль собирается

# IV. ReadyNow

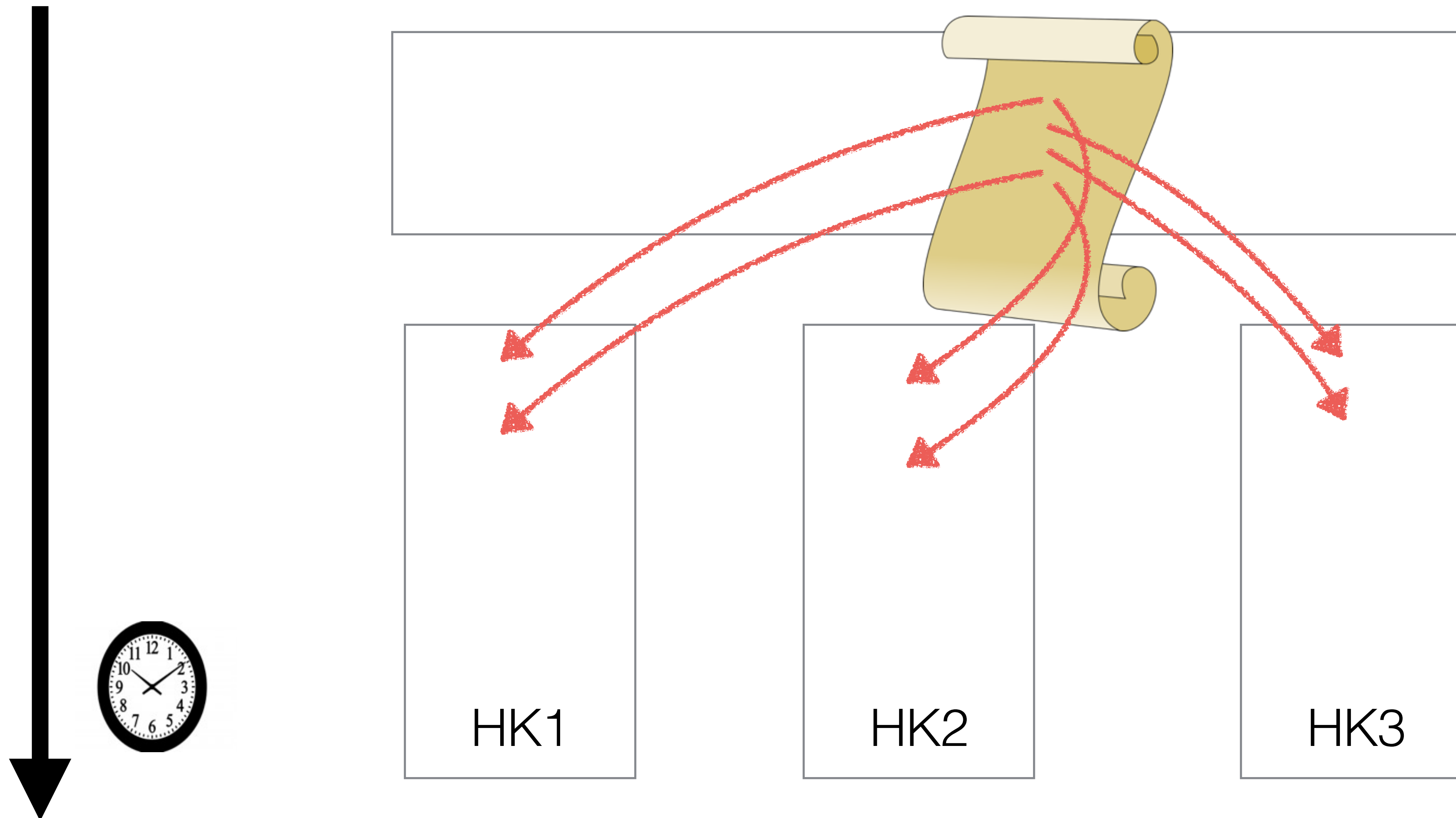
- Напоминает PGO-оптимизацию gcc
- Для регулярно используемых приложение
- При первом запуске приложения - профиль собирается
- При последующих - используется и обновляется

# Запись профиля RN





# Использование профиля RN



# IV. ReadyNow



# IV. ReadyNow

- Между AOT и JIT
- Большинство компиляций происходят раньше обычного и не в ущерб качеству





# IV. ReadyNow

- Между AOT и JIT
- Большинство компиляций происходят раньше обычного и не в ущерб качеству
- Подгружается профиль, а не скомпилированный код



# IV. ReadyNow

- Между AOT и JIT
- Большинство компиляций происходят раньше обычного и не в ущерб качеству
- Подгружается профиль, а не скомпилированный код
- “Ошибок прошлых мы уже не повторим”
- Деоптимизации многократно реже обычного



# IV. ReadyNow

- Между AOT и JIT
- Большинство компиляций происходят раньше обычного и не в ущерб качеству
- Подгружается профиль, а не скомпилированный код
- “Ошибок прошлых мы уже не повторим”
- Деоптимизации многократно реже обычного
- Ощутимый эффект в “хороших” случаях





# IV. ReadyNow

- Между AOT и JIT
- Большинство компиляций происходят раньше обычного и не в ущерб качеству
- Подгружается профиль, а не скомпилированный код
- “Ошибок прошлых мы уже не повторим”
- Деоптимизации многократно реже обычного
- Ощутимый эффект в “хороших” случаях
- Запасной вариант - обычная JIT компиляция



# Три требования к ReadyNow



# Три требования к ReadyNow

## I. Адекватность профиля





# Три требования к ReadyNow

- I. Адекватность профиля
- II. Применимость профиля

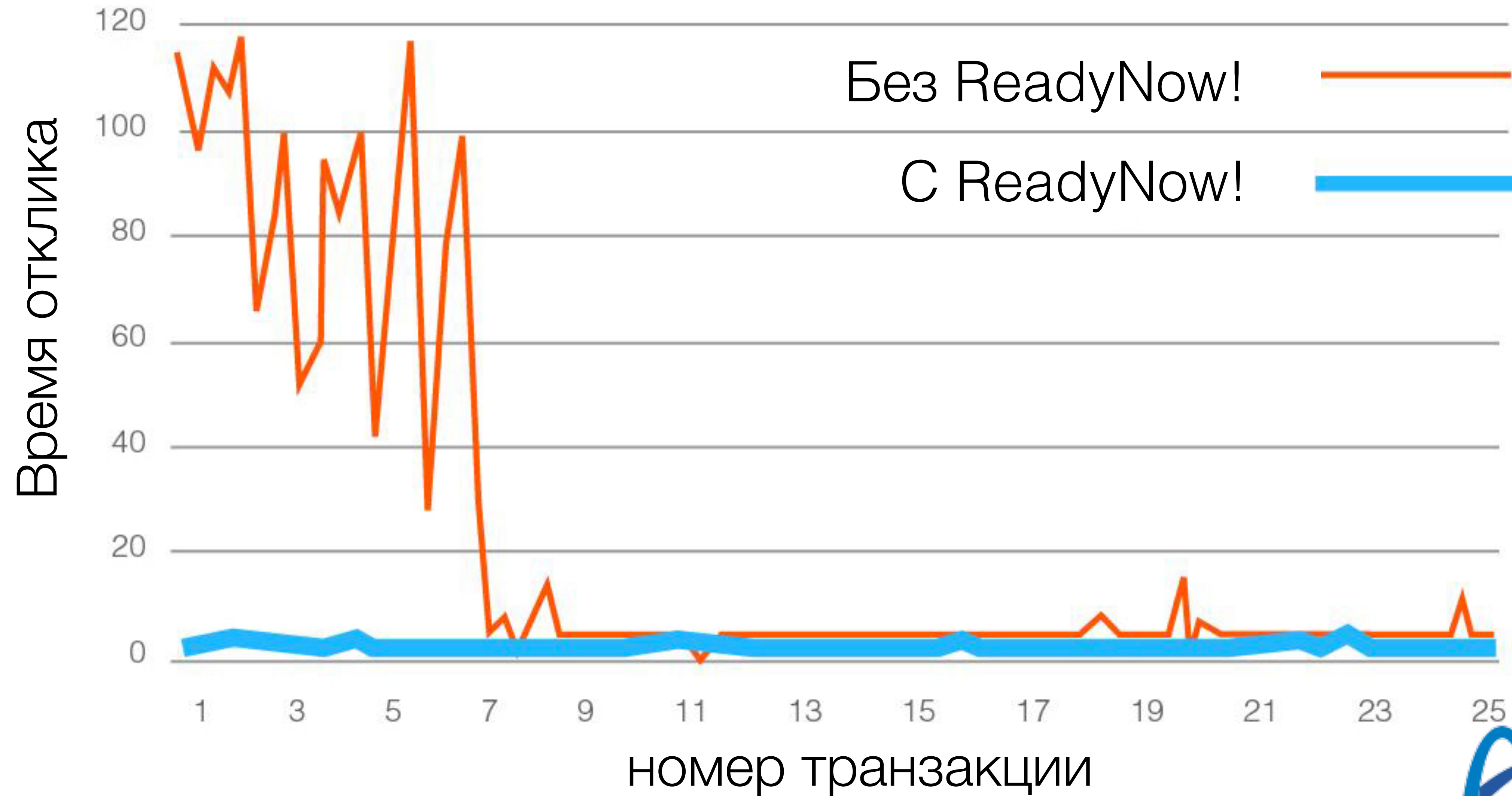


# Три требования к ReadyNow

- I. Адекватность профиля
- II. Применимость профиля
- III. Разрешимость зависимостей



# ReadyNow







**Zing:** Виртуальная Java-машина для крупного бизнеса

- Главная цель - улучшение показателей функционирования виртуальной Java-машины для предприятий
- Неизменная производительность - не просто быстро, а ВСЕГДА быстро
- Сборка мусора больше не влияет на производительность приложений
- Широчайшая сфера применения:
  - от интерактивных приложений до задач, критичных ко времени отклика
  - от микросервисов до приложений, требующих больших объемов памяти
- Устраняет необходимость в большинстве известных "костылей" в вашем коде









**Zulu Embedded**: Когда нужны решения для встроенных систем

- 100% открытый код - технология, основанная на OpenJDK
- Сертифицирована на совместимость и соответствие Java SE
- Показатели функционирования, идентичные OpenJDK и Oracle Java SE
- Высококласная поддержка
- Релизация для Linux x86, ARM32, ARM64, PPC32, MIPS, а также Windows и Mac OS



## Zulu Embedded: Когда нужны решения для встроенных систем

- 100% открытый код - технология, основанная на OpenJDK
- Сертифицирована на совместимость и соответствие Java SE
- Показатели функционирования, идентичные OpenJDK и Oracle Java SE
- Высококласная поддержка
- Релизация для Linux x86, ARM32, ARM64, PPC32, MIPS, а также Windows и Mac OS



# Время для вопросов

Спасибо за помощь - Douglas Hawkins



@dougqh

Иван Крылов



@JohnWings



Ivan Krylov