Дебаг продакшена

без перекладывания
без остановок
без риска

Дебаг? Какой дебаг?

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {









}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    // nums == [3, 2, 4]
    // target == 6









}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    // nums == [3, 2, 4]
    // target == 6

    //  partA  +  partB  = target
    // nums[1] + nums[2] = 6




}
```

Leetcode: Two Sum

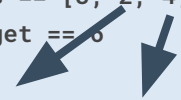
Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    // nums == [3, 2, 4]
    // target == 6

    //  partA  +  partB  = target
    // nums[1] + nums[2] = 6




}
```



Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    // nums == [3, 2, 4]
    // target == 6

    //  partA  +  partB  = target
    // nums[1] + nums[2] = 6




    return new int[] {1, 2};
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    // nums == [3, 2, 4]
    // target == 6

    //  partA  +  partB  = target
    // nums[1] + nums[2] = 6
                 ↘     ↘
    return new int[] {1, 2};
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        for (int iB = iA + 1; iB < nums.length; iB++) {
            int partB = nums[iB];
            if (partA + partB == target) {
                return new int[] {iA, iB};
            }
        }
    }
    return null;
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        for (int iB = iA + 1; iB < nums.length; iB++) {
            int partB = nums[iB];
            if (partA + partB == target) {
                return new int[] {iA, iB};
            }
        }
    }
    return null;
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        for (int iB = iA + 1; iB < nums.length; iB++) {
            int partB = nums[iB];
            if (partA + partB == target) {
                return new int[] {iA, iB};
            }
        }
    }
    return null;
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        for (int iB = iA + 1; iB < nums.length; iB++) {
            int partB = nums[iB];
            if (partA + partB == target) {
                return new int[] {iA, iB};
            }
        }
    }
    return null;
}
```

Leetcode: Two Sum

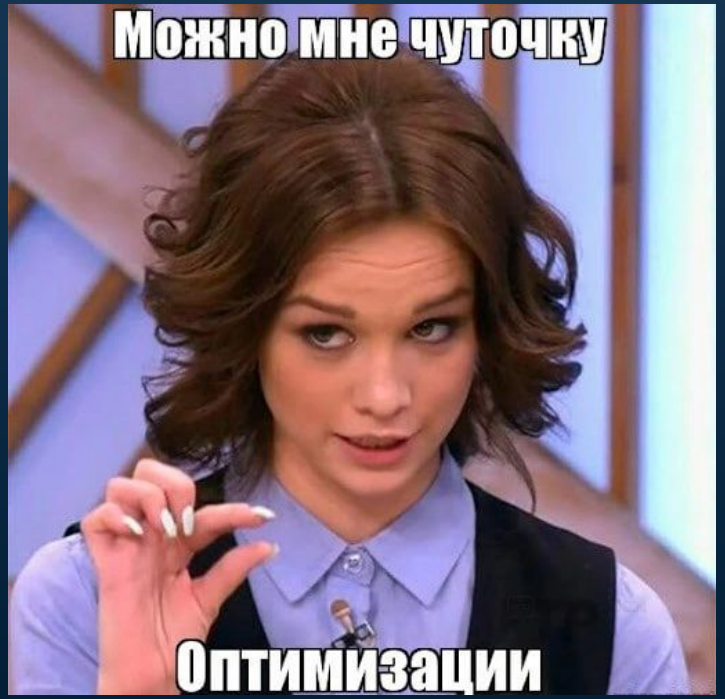
Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        for (int iB = iA + 1; iB < nums.length; iB++) {
            int partB = nums[iB];
            if (partA + partB == target) {
                return new int[] {iA, iB};
            }
        }
    }
    return null;
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        for (int iB = iA + 1; iB < nums.length; iB++)
            int partB = nums[iB];
            if (partA + partB == target) {
                return new int[] {iA, iB};
            }
        }
    }
    return null;
}
```



Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
int[] twoSum(int[] nums, int target) {
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];

        int partB = target - partA;
        int iB = //index of partB;

        return new int[] {iA, iB};

    }
    return null;
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];

        int partB = target - partA;
        Integer iB = numToIndex.get(partB);
        if (iB != null) {
          return new int[] {iA, iB};
        }
    }
    return null;
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        int partB = target - partA;
        Integer iB = numToIndex.get(partB);
        if (iB != null) {
          return new int[] {iA, iB};
        }
    }
    return null;
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        int partB = target - partA;
        Integer iB = numToIndex.get(partB);
        if (iB != null) {
          return new int[] {iA, iB};
        }
    }
    return null;
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        int partB = target - partA;
        Integer iB = numToIndex.get(partB);
        if (iB != null) {
          return new int[] {iA, iB};
        }
    }
    return null;
}
```

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        int partB = target - partA;
        Integer iB = numToIndex.get(partB);
        if (iB != null) {
          return new int[] {iA, iB};
        }
    }
    return null;
}
```

nums = [3,2,4]
target = 6

result = [1, 2]

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        int partB = target - partA;
        Integer iB = numToIndex.get(partB);
        if (iB != null) {
          return new int[] {iA, iB};
        }
    }
    return null;
}
```

nums = [3,2,4]
target = 6

result = [0, 0]

Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
```

nums = [3,2,4]
target = 6

result = [0, 0]

```
for (int iA = 0; iA < nums.length; iA++) {   iA: 0
    int partA = nums[iA];   nums: [3, 2, 4]   iA: 0   partA: 3
    int partB = target - partA;   target: 6   partA: 3   partB: 3
    Integer iB = numToIndex.get(partB);   partB: 3   iB: 0   numToIndex:  size = 3
    if (iB != null = true) {   iB: 0
        return new int[] {iA, iB};
```

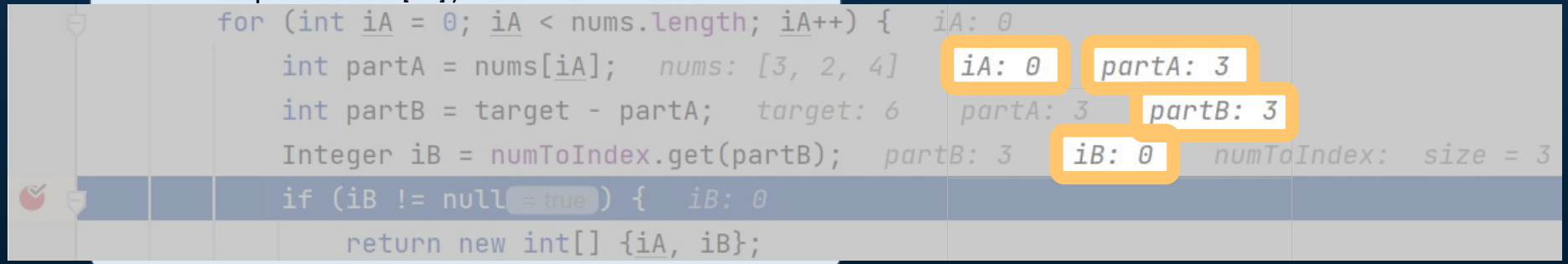
Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
```

nums = [3,2,4]
target = 6

result = [0, 0]



Leetcode: Two Sum

Вернуть индексы двух элементов в массиве num, дающих в сумме target:

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        int partB = target - partA;
        Integer iB = numToIndex.get(partB);
        if (iB != null && iA != iB) {
          return new int[] {iA, iB};
        }
    }
    return null;
}
```

nums = [3,2,4]
target = 6

result = [1,2]

Подключиться дебагером к продакшену?

Подключиться дебагером к продакшену?

- Торчит открытый порт

Подключиться дебагером к продакшену?

- Торчит открытый порт
- Действия “отлаживающего” не аудируемы

Подключиться дебагером к продакшену?

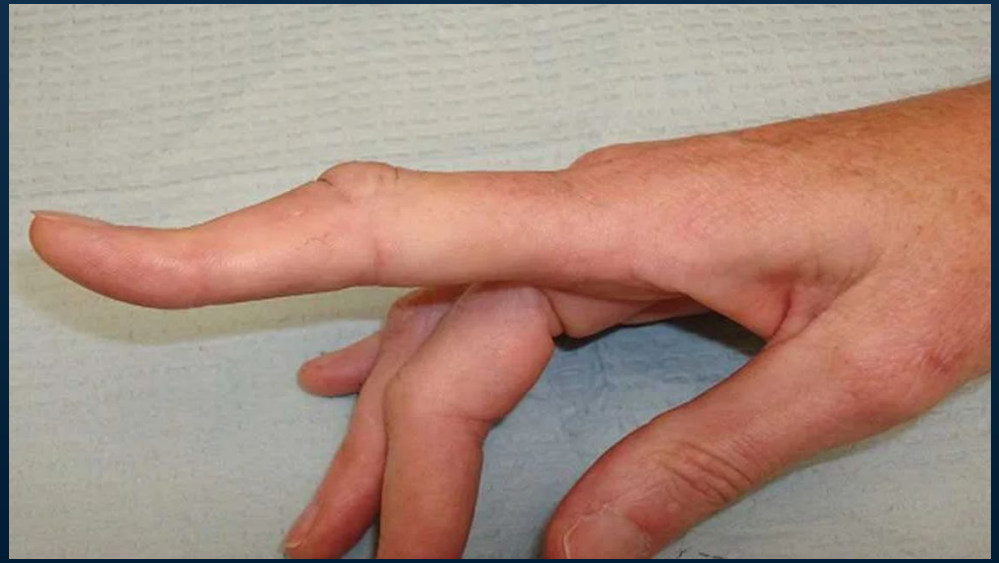
- Торчит открытый порт
- Действия “отлаживающего” не аудируемы

Подключиться дебагером к продакшену?

- Торчит открытый порт
- Действия “отлаживающего” не аудируемы
- Невозможно запретить останавливать потоки

Подключиться дебагером к продакшену?

- Торчит открытый порт
- Действия “отлаживающего” не аудируемы
- Невозможно запретить останавливать потоки
- Можно уложить продакшн движением пальца



Подключиться дебагером к продакшену?

- Торчит открытый порт
- Действия “отлаживающего” не аудируемы
- Невозможно запретить останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (злонамеренно или случайно) модифицировать состояние

Добавим логи?

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        int partB = target - partA;
        Integer iB = numToIndex.get(partB);
        if (iB != null) {
          return new int[] {iA, iB};
        }
    }
    return null;
}
```

Добавим логи?

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        int partB = target - partA;
        Integer iB = numToIndex.get(partB);
        if (iB != null) {
          return new int[] {iA, iB};
        }
    }
    return null;
}
```

```
Map<Integer, Integer> numToIndex = new HashMap<>();
int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        numToIndex.put(nums[i], i);
    }
    for (int iA = 0; iA < nums.length; iA++) {
        int partA = nums[iA];
        int partB = target - partA;
        Integer iB = numToIndex.get(partB);
        log(iA, partA, iB, partB, target);
        if (iB != null) {
          return new int[] {iA, iB};
        }
    }
    return null;
}
```

А как добавить логи?

- Переложить продакшн

А как добавить логи?

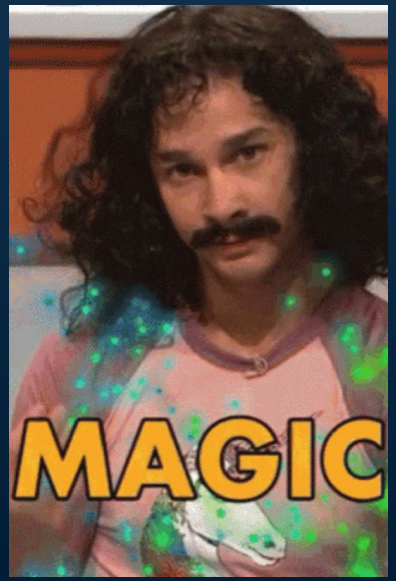
- Переложить продакшн

А как добавить логи?

- Переложить продакшн
- HCR - ???

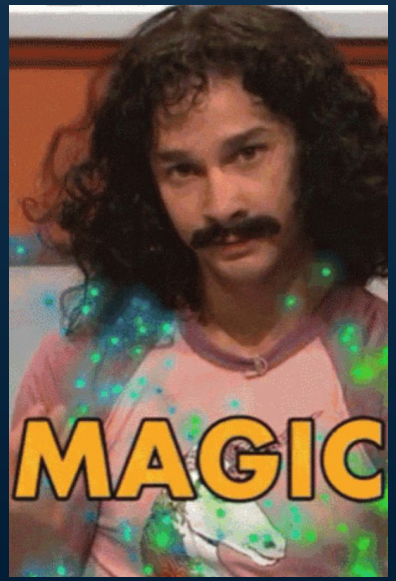
А как добавить логи?

- Переложить продакшн
- HCR - ???



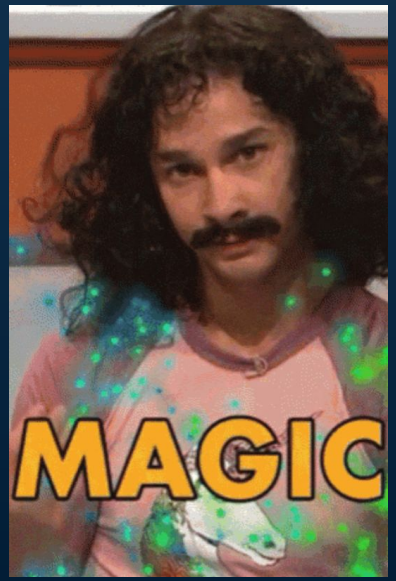
А как добавить логи?

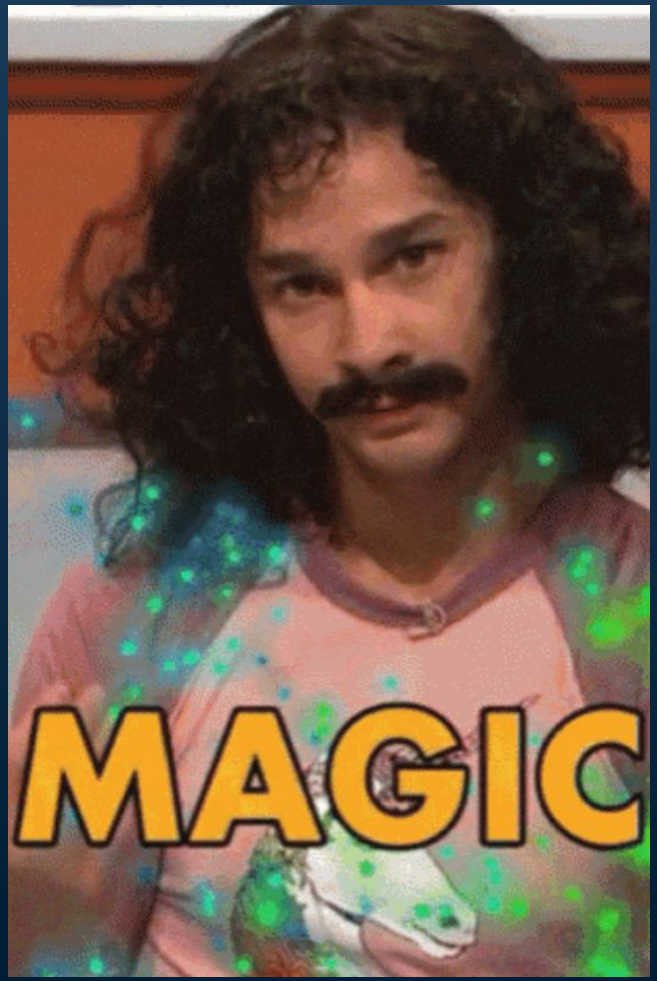
- Переложить продакшн
- HCR - ???
- JVMTI - ???



А как добавить логи?

- Переложить продакшн
- HCR - ???
- JVMTI - ???





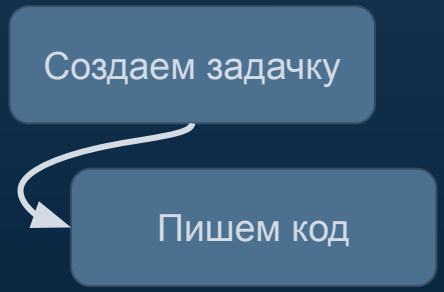
Переложить продакшн? (

- Долго

Создаем задачку

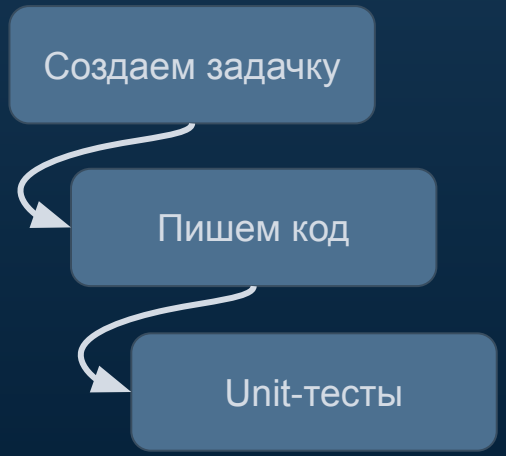
Переложить продакшн? (

- Долго



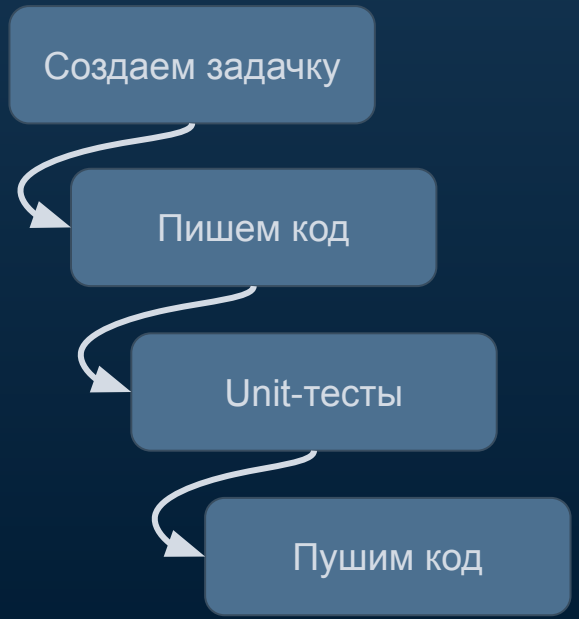
Переложить продакшн? (

- Долго



Переложить продакшн? (

- Долго



Переложить продакшн? (

- Долго

Переложить продакшн? (

- Долго

Переложить продакшн? (

- Долго

Переложить продакшн? (

- Долго

Переложить продакшн? (

- Долго

Переложить продакшн? (

- Долго

Переложить продакшн? (

- Долго

Переложить продакшн? (

- Долго

Переложить продакшн? (

- Долго

Переложить продакшн? (

- Долго

Переложить продакшн? (

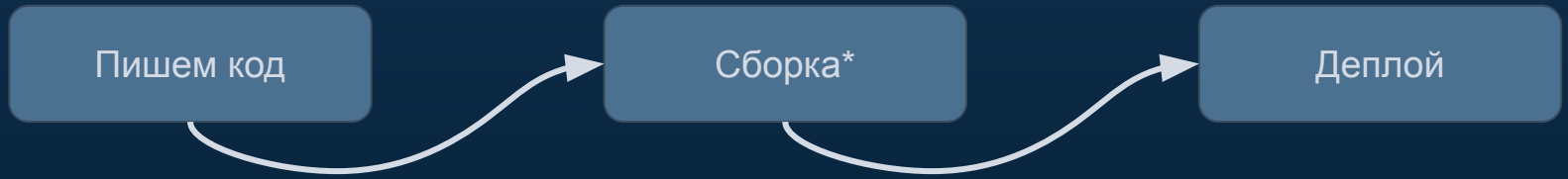
- Долго

Переложить продакшн? (

- Долго
- Опасно

Переложить продакшн? (

- Долго
- Опасно



Переложить продакшн? (

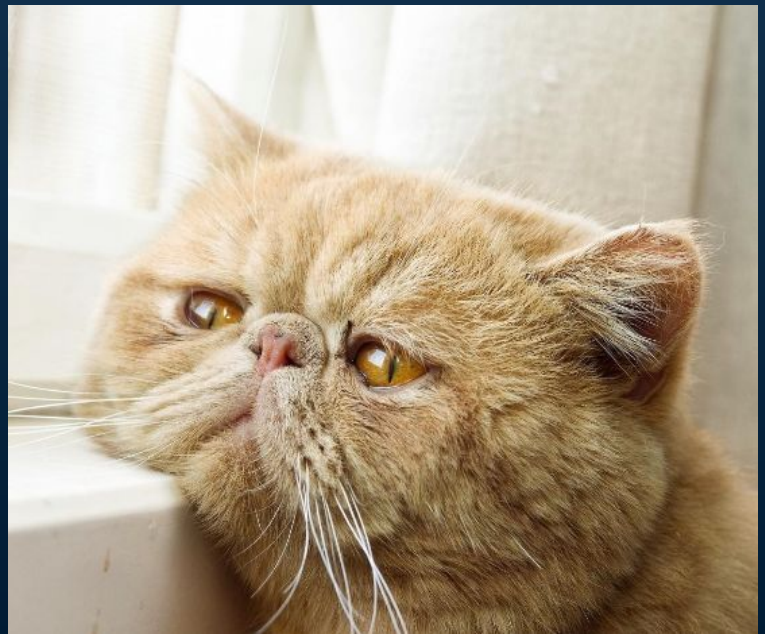
- Долго
- Опасно

Переложить продакшн? (

- Долго
- Опасно
- Не интересно)

Переложить продакшн? (

- Долго
- Опасно
- Не интересно)



Идём в отрыв: HCR

Идём в отрыв:
Hot Code Replacement

HCR: Меняем код на лету

```
class Test { String hello() { return "hello"; } }

public static void main(String[] args) throws UnmodifiableClassException, ClassNotFoundException {
   Test test = new Test();
   System.out.println(test.hello());   // hello

   byte[] compiled = compile(Test.class.getName(),
       "class Test { String hello() { return \"bye\"; } }"
   );
   Agent.INSTRUMENTATION.redefineClasses(new ClassDefinition(Test.class, compiled));

   System.out.println(test.hello());   // bye
}
```

HCR: Меняем код на лету

```
class Test { String hello() { return "hello"; } }

public static void main(String[] args) throws UnmodifiableClassException, ClassNotFoundException {
    Test test = new Test();
    System.out.println(test.hello());   // hello

    byte[] compiled = compile(Test.class.getName(),
        "class Test { String hello() { return \"bye\"; } }"
    );
    Agent.INSTRUMENTATION.redefineClasses(new ClassDefinition(Test.class, compiled));

    System.out.println(test.hello());   // bye
}
```

HCR: Меняем код на лету

```
class Test { String hello() { return "hello"; } }

public static void main(String[] args) throws UnmodifiableClassException, ClassNotFoundException {
   Test test = new Test();
   System.out.println(test.hello());   // hello

   byte[] compiled = compile(Test.class.getName(),
       "class Test { String hello() { return \"bye\"; } }"
   );
   Agent.INSTRUMENTATION.redefineClasses(new ClassDefinition(Test.class, compiled));

   System.out.println(test.hello());   // bye
}
```

HCR: Меняем код на лету

```
class Test { String hello() { return "hello"; } }

public static void main(String[] args) throws UnmodifiableClassException, ClassNotFoundException {
   Test test = new Test();
   System.out.println(test.hello());   // hello

   byte[] compiled = compile(Test.class.getName(),
       "class Test { String hello() { return \"bye\"; } }"
   );
   Agent.INSTRUMENTATION.redefineClasses(new ClassDefinition(Test.class, compiled));

   System.out.println(test.hello());   // bye
}
```

HCR: Меняем код на лету

```
class Test { String hello() { return "hello"; } }

public static void main(String[] args) throws UnmodifiableClassException, ClassNotFoundException {
   Test test = new Test();
   System.out.println(test.hello());   // hello

   byte[] compiled = compile(Test.class.getName(),
       "class Test { String hello() { return \"bye\"; } }"
   );
   Agent.INSTRUMENTATION.redefineClasses(new ClassDefinition(Test.class, compiled));

   System.out.println(test.hello());   // bye
}
```

HCR: Меняем код на лету

```
class Test { String hello() { return "hello"; } }

public static void main(String[] args) throws UnmodifiableClassException, ClassNotFoundException {
   Test test = new Test();
   System.out.println(test.hello());   // hello

   byte[] compiled = compile(Test.class.getName(),
       "class Test { String hello() { return \"bye\"; } }"
   );
   Agent.INSTRUMENTATION.redefineClasses(new ClassDefinition(Test.class, compiled));

   System.out.println(test.hello());   // bye
}
```

HCR: Меняем код на лету

```
public class Agent {

   public static Instrumentation INSTRUMENTATION;

   public static void premain(String args, Instrumentation inst) {
       Agent.INSTRUMENTATION = instrumentation;
   }

}
```

HCR: Меняем код на лету

```
public class Agent {

   public static Instrumentation INSTRUMENTATION;

   public static void premain(String args, Instrumentation inst) {
       Agent.INSTRUMENTATION = instrumentation;
   }

}
```

HCR: Меняем код на лету

```
public class Agent {

   public static Instrumentation INSTRUMENTATION;

   public static void premain(String args, Instrumentation inst) {
       Agent.INSTRUMENTATION = instrumentation;
   }

}
```

```
MANIFEST.MF

Premain-Class: my.package.Agent
```

HCR: Меняем код на лету

```
public class Agent {

   public static Instrumentation INSTRUMENTATION;

   public static void premain(String args, Instrumentation inst) {
       Agent.INSTRUMENTATION = instrumentation;
   }

}
```

```
MANIFEST.MF

Premain-Class: my.package.Agent
```

```
java -javaagent:<path_to_agent_jar> ...
```

HCR: Меняем код на лету

- Торчит открытый порт

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” не аудируемы

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно запретить останавливать потоки

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца

```
class User {

  String name = ...;
  Group group = ...;

  @Override
  public String toString() {
    return "name: " + name + ", group: " + group;
  }
}
```

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца

```
class User {

  String name = ...;
  Group group = ...;

  @Override
  public String toString() {
    return "name: " + name + ", group: " + group;
  }
}
```

```
class Group {

  String name = ...;
  List<User> users = ...;

  @Override
  public String toString() {
    return "name: " + name + ", users: " + users;
  }
}
```

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца

```
class User {

  String name = ...;
  Group group = ...;

  @Override
  public String toString() {
    return "name: " + name + ", group: " + group;
  }
}
```

```
class Group {

  String name = ...;
  List<User> users = ...;

  @Override
  public String toString() {
    return "name: " + name + ", users: " + users;
  }
}
```

```
log(user);
```

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца

```
class User {

  String name = ...;
  Group group = ...;

  @Override
  public String toString() {
    return "name: " + name + ", group: " + group;
  }
}
```

```
class Group {

  String name = ...;
  List<User> users = ...;

  @Override
  public String toString() {
    return "name: " + name + ", users: " + users;
  }
}
```

```
log(user);
```

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца

```
class User {

  String name = ...;
  Group group = ...;

  @Override
  public String toString() {
    return "name: " + name + ", group: " + group;
  }
}
```

```
class Group {

  String name = ...;
  List<User> users = ...;

  @Override
  public String toString() {
```

```
if (user.group.name.equals("Not Found")) {
  log(user);
}
```

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (злонамеренно или случайно) модифицировать состояние

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (~~злонамеренно или~~ случайно) модифицировать состояние

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (~~злонамеренно или~~ случайно) модифицировать состояние

```
void saveUsers(List<User> users) {
  ... // some work with users
  users = filterUsers(users);
  ...
}
```

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (~~злонамеренно или~~ случайно) модифицировать состояние

```
void saveUsers(List<User> users) {
  ... // some work with users
  users = filterUsers(users);
  ...
}
```

```
if (...) log(filterUsers(users));
saveUsers(users);
```

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (~~злонамеренно или~~ случайно) модифицировать состояние

```
void saveUsers(List<User> users) {
  ... // some work with users
  users = filterUsers(users);
  ...
}
```

```
List<User> filterUsers(List<User> users) {
  users.removeIf(...);
  return users;
}
```

```
if (...) log(filterUsers(users));
saveUsers(users);
```

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (~~злонамеренно или~~ случайно) модифицировать состояние

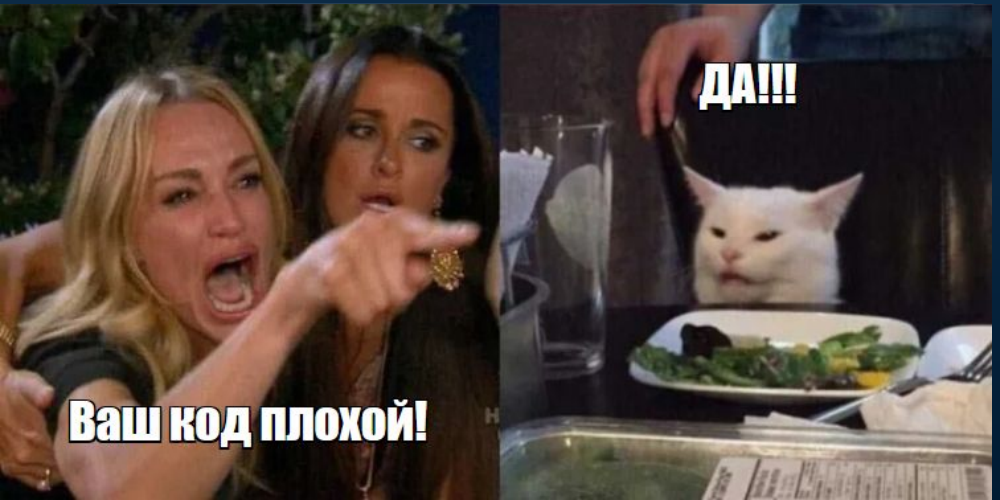

```
ser> users) {
th users
(users);
```

```
List<User> filterUsers(List<User> users) {
  users.removeIf(...);
  return users;
}
```

```
ers(users));
```

HCR: Меняем код на лету

- Торчит открытый порт не торчит
- Действия “отлаживающего” не аудируемы
- Невозможно запретить останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (злонамеренно или случайно) модифицировать состояние




```
List<User> filterUsers(List<User> users) {
  users.removeIf(...);
  return users;
}
```

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (~~злонамеренно или~~ случайно) модифицировать состояние

HCR: Ограничения

Нельзя:

HCR: Ограничения

Нельзя:

- Добавить/удалить метод

HCR: Ограничения

Нельзя:

- Добавить/удалить метод
- Изменить сигнатуру метода

HCR: Ограничения

Нельзя:

- Добавить/удалить метод
- Изменить сигнатуру метода
- Добавить/удалить/изменить поле

HCR: Ограничения

Нельзя:

- Добавить/удалить метод
- Изменить сигнатуру метода
- Добавить/удалить/изменить поле
- Изменить иерархию наследования

HCR: Ограничения

Нельзя:

- Добавить/удалить метод
- Изменить сигнатуру метода
- Добавить/удалить/изменить поле
- Изменить иерархию наследования

Короче, можно:

- Изменить код метода

HCR: Ограничения

Нельзя:

- Добавить/удалить метод
- Изменить сигнатуру метода
- Добавить/удалить/изменить поле
- Изменить иерархию наследования

Короче, можно:

- Изменить код метода
- Добавить новый класс

HCR: Ограничения

Нельзя:

- Добавить/удалить метод
- Изменить сигнатуру метода
- Добавить/удалить/изменить поле
- Изменить иерархию наследования

Короче, можно:

- Изменить код метода
- Добавить новый класс

JEP 159: Enhanced Class Redefinition:

Можно:

- Добавлять методы
- Добавлять поля
- Добавлять интерфейс

HCR: Ограничения

Нельзя:

- Добавить/удалить метод
- Изменить сигнатуру метода
- Добавить/удалить/изменить поле
- Изменить иерархию наследования

Короче, можно:

- Изменить код метода
- Добавить новый класс

~~**JEP 159: Enhanced Class Redefinition:**~~

Status: Withdrawn/Closed

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (~~злонамеренно или~~ случайно) модифицировать состояние

без перекладывания

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (~~злонамеренно или~~ случайно) модифицировать состояние

без перекладывания

без остановок

HCR: Меняем код на лету

- ~~Торчит~~ открытый порт не торчит
- Действия “отлаживающего” ~~не~~ аудируемы
- Невозможно ~~запретить~~ останавливать потоки
- Можно уложить продакшн движением пальца
- Можно (~~злонамеренно или~~ случайно) модифицировать состояние

без перекладывания

без остановок

без риска

Идём в отрыв по-настоящему: JVMTI

Идём в отрыв по-настоящему: JVM Tool Interface

JVMTI

JVM Tool Interface - “a way to inspect the state and to control the execution of applications”

JVMTI

JVM Tool Interface - “a way to inspect the state and to control the execution of applications”

JVMTI - способ трогать лапками кишки JVM наживую, меняя конечности местами [без наркоза]

JVMTI Hello World

```
#include <jvmti.h>
#include <stdio.h>

JNIEXPORT jint JNICALL
Agent_OnLoad(JavaVM* vm, char* options, void* reserved)
{
    jvmtiEnv* jvmti;
    vm->GetEnv((void**) &jvmti, JVMTI_VERSION_1_0);

    char* vm_name = NULL;
    jvmti->GetSystemProperty("java.vm.name", &vm_name);
    printf("JVM name = %s\n", vm_name);
    fflush(stdout);
    return 0;
}
```

JVMTI Hello World

```
#include <jvmti.h>
#include <stdio.h>

JNIEXPORT jint JNICALL
Agent_OnLoad(JavaVM* vm, char* options, void* reserved)
{
    jvmtiEnv* jvmti;
    vm->GetEnv((void**) &jvmti, JVMTI_VERSION_1_0);

    char* vm_name = NULL;
    jvmti->GetSystemProperty("java.vm.name", &vm_name);
    printf("JVM name = %s\n", vm_name);
    fflush(stdout);
    return 0;
}
```

JVMTI Hello World

```
#include <jvmti.h>
#include <stdio.h>

JNIEXPORT jint JNICALL
Agent_OnLoad(JavaVM* vm, char* options, void* reserved)
{
    jvmtiEnv* jvmti;
    vm->GetEnv((void**) &jvmti, JVMTI_VERSION_1_0);

    char* vm_name = NULL;
    jvmti->GetSystemProperty("java.vm.name", &vm_name);
    printf("JVM name = %s\n", vm_name);
    fflush(stdout);

    return 0;
}
```

JVMTI Hello World

```c
#include <jvmti.h>
#include <stdio.h>

JNIEXPORT jint JNICALL
Agent_OnLoad(JavaVM* vm, char* options, void* reserved)
{
    jvmtiEnv* jvmti;
    vm->GetEnv((void**) &jvmti, JVMTI_VERSION_1_0);

    char* vm_name = NULL;
    jvmti->GetSystemProperty("java.vm.name", &vm_name);
    printf("JVM name = %s\n", vm_name);
    fflush(stdout);
    return 0;
}
```

```java
public static void main(String[] args) {
    System.out.println("Main method");
}
```

```
java -agentpath:<path_to_agent_lib> ...
```

```
> JVM name = OpenJDK 64-Bit Server VM
> Main method
```

JVMTI

JVM Tool Interface - “a way to inspect the state and to control the execution of applications”

```
GetLoadedClasses
GetClassMethods
GetClassFields
...

IterateOverReachableObjects
IterateOverHeap
IterateOverInstancesOfClass
...

GetCurrentThread
GetAllThreads
...

GetLocalObject
GetLocalInt
...
```

JVMTI

JVM Tool Interface - “a way to inspect the state and to control the execution of applications”

```
GetLoadedClasses
GetClassMethods
GetClassFields
...

IterateOverReachableObjects
IterateOverHeap
IterateOverInstancesOfClass
...

GetCurrentThread
GetAllThreads
...

GetLocalObject
GetLocalInt
...
```

```
SetBreakpoint
SetFieldAccessWatch
SetFieldModificationWatch
...

SuspendThread
SuspendThreadList
ResumeThread
ResumeThreadList
...
```

JVMTI

JVM Tool Interface - “a way to inspect the state and to control the execution of applications”

```
GetLoadedClasses
GetClassMethods
GetClassFields
...

IterateOverReachableObjects
IterateOverHeap
IterateOverInstancesOfClass
...

GetCurrentThread
GetAllThreads
...

GetLocalObject
GetLocalInt
...
```

```
SetBreakpoint
SetFieldAccessWatch
SetFieldModificationWatch
...

SuspendThread
SuspendThreadList
ResumeThread
ResumeThreadList
...
```

Оказывается,
JDWP тоже JVMTI Agent!

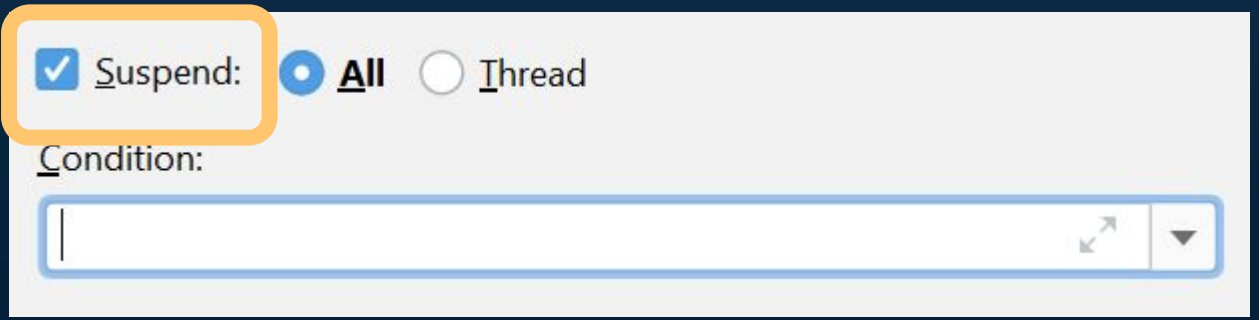
JVMTI Debugger

Intellij IDEA Debugger

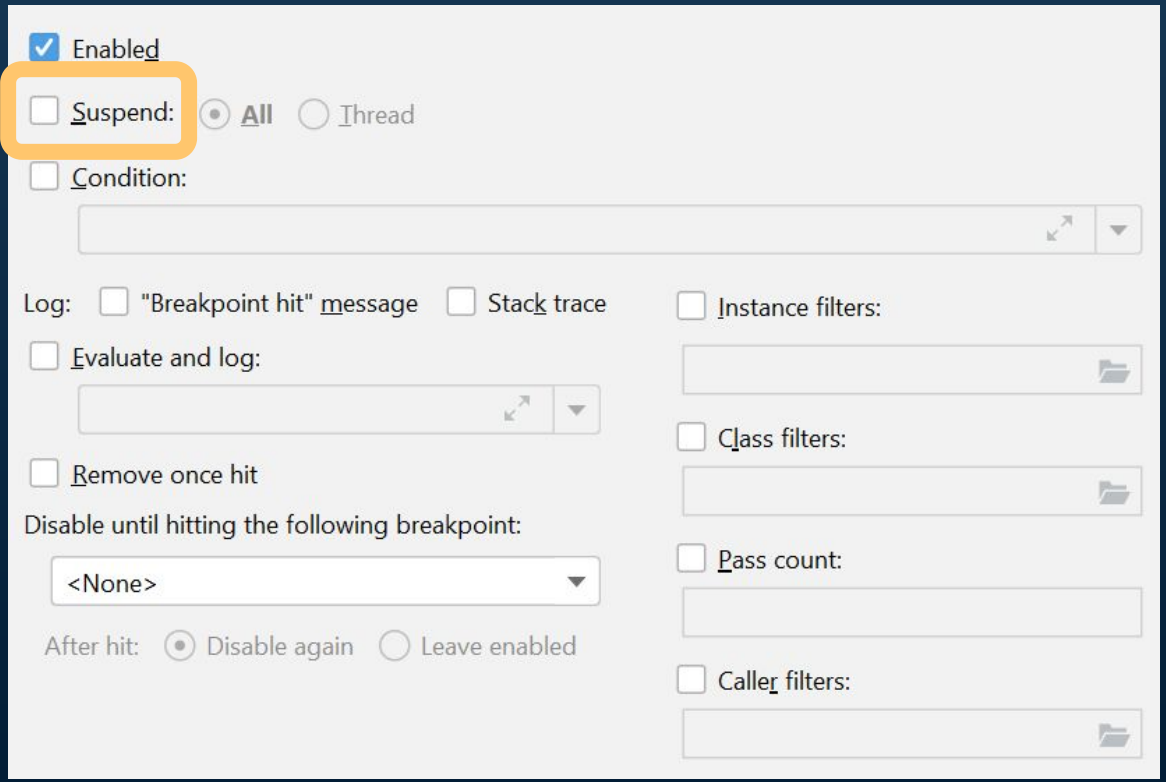
```
for (int iA = 0; iA < nums.length; iA++) {   iA: 0
    int partA = nums[iA];   nums: [3, 2, 4]    iA: 0    partA: 3
    int partB = target - partA;   target: 6    partA: 3    partB: 3
    Integer iB = numToIndex.get(partB);   partB: 3    iB: 0    numToIndex:  size = 3
    if (iB != null = true) {   iB: 0
        return new int[] {iA, iB};
```

Intellij IDEA Debugger

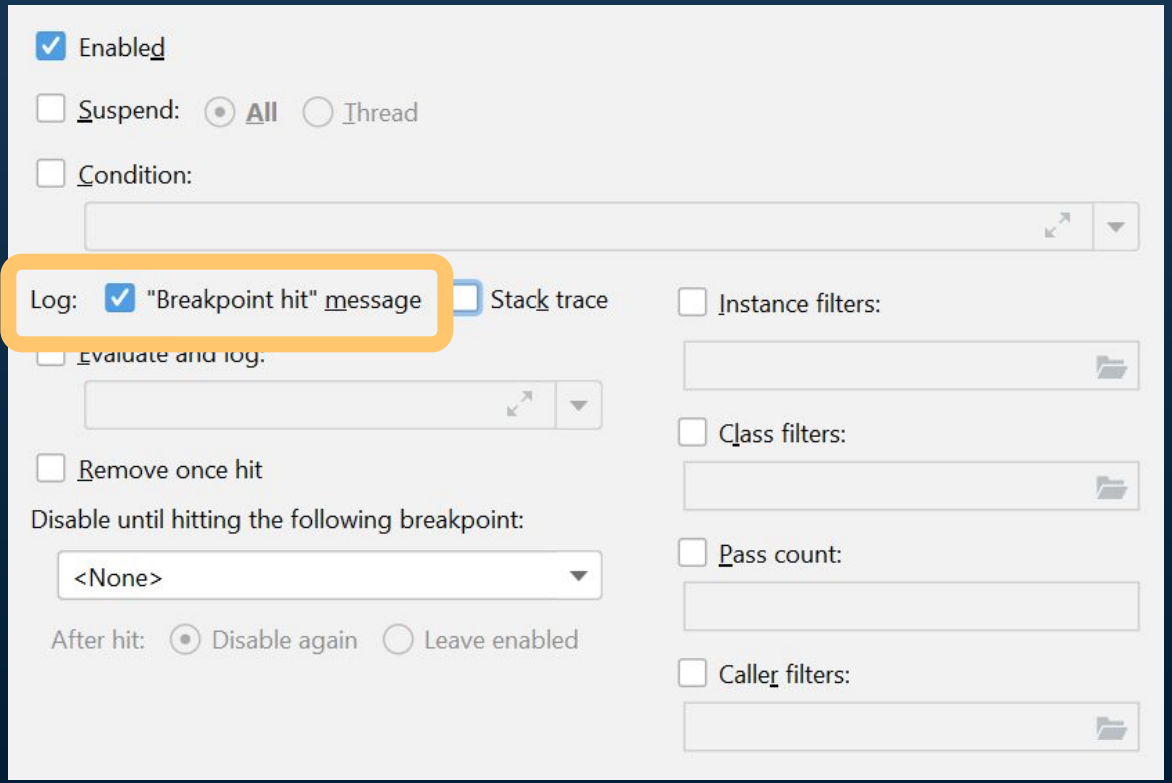
```
for (int iA = 0; iA < nums.length; iA++) {   iA: 0
    int partA = nums[iA];   nums: [3, 2, 4]    iA: 0    partA: 3
    int partB = target - partA;   target: 6    partA: 3    partB: 3
    Integer iB = numToIndex.get(partB);   partB: 3    iB: 0    numToIndex:  size = 3
    if (iB != null = true) {   iB: 0
        return new int[] {iA, iB};
```



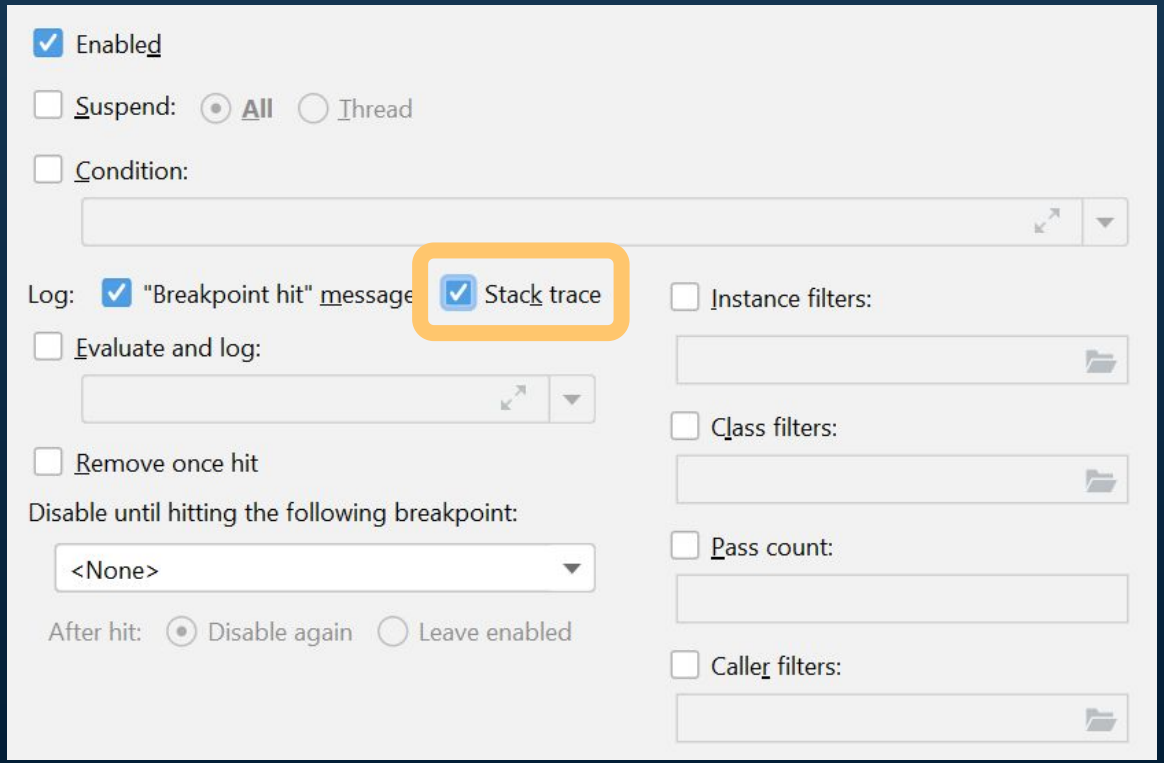
Intellij IDEA Debugger



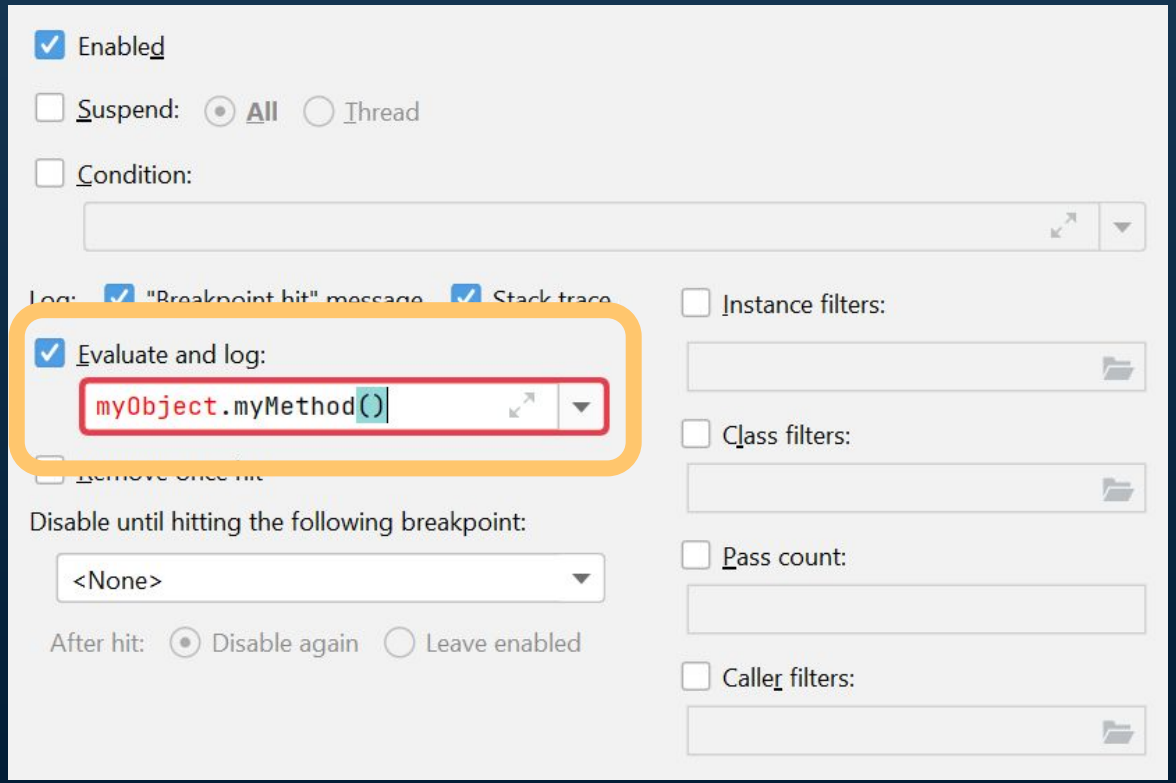
Intellij IDEA Debugger



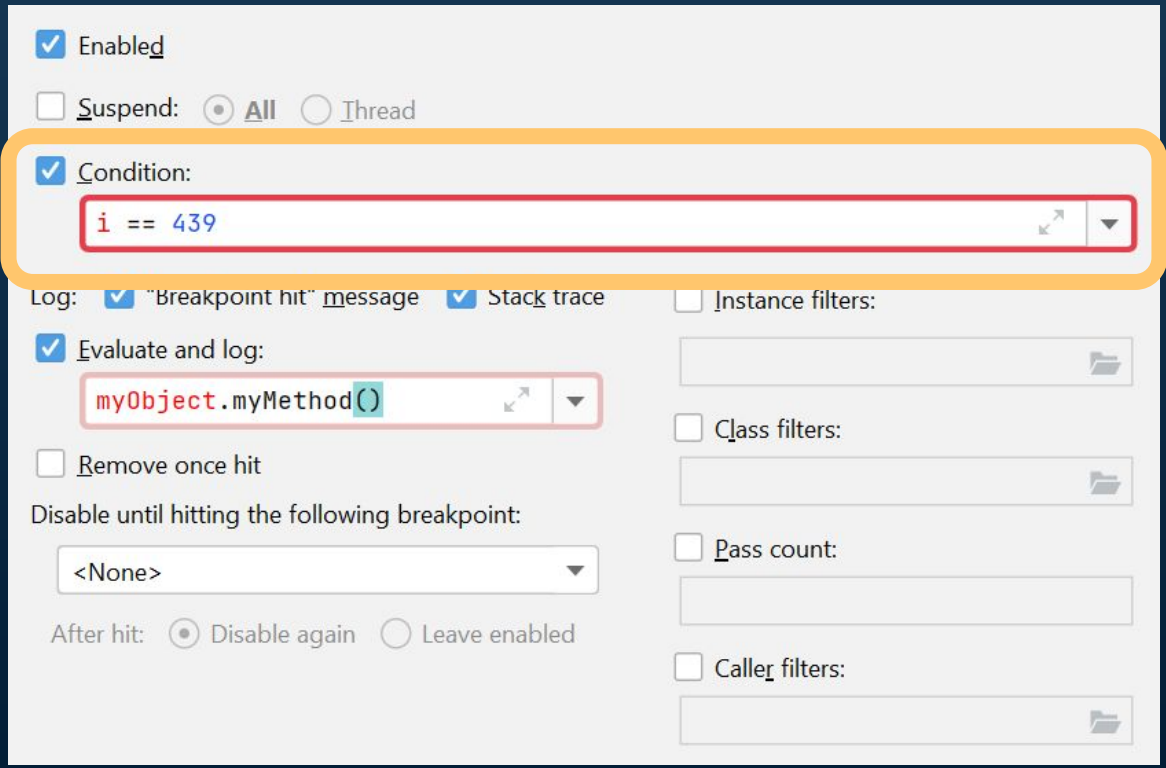
Intellij IDEA Debugger



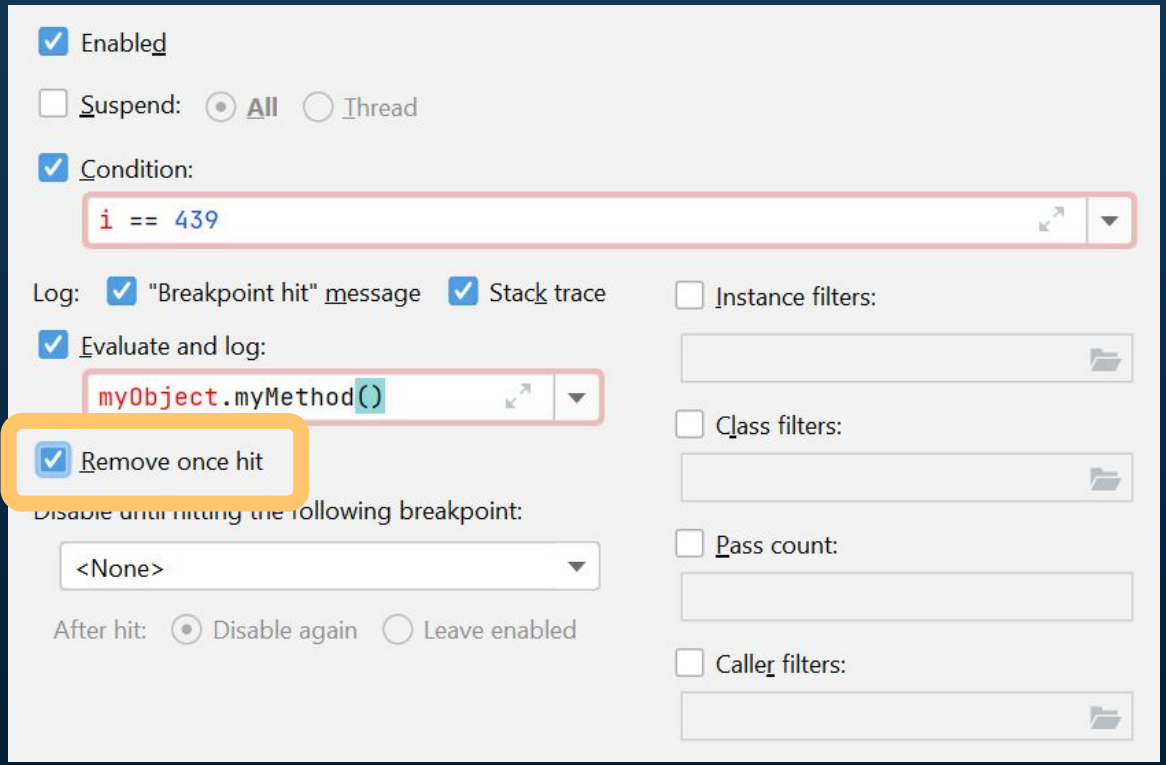
Intellij IDEA Debugger



Intellij IDEA Debugger



Intellij IDEA Debugger



JVMTI Debugger

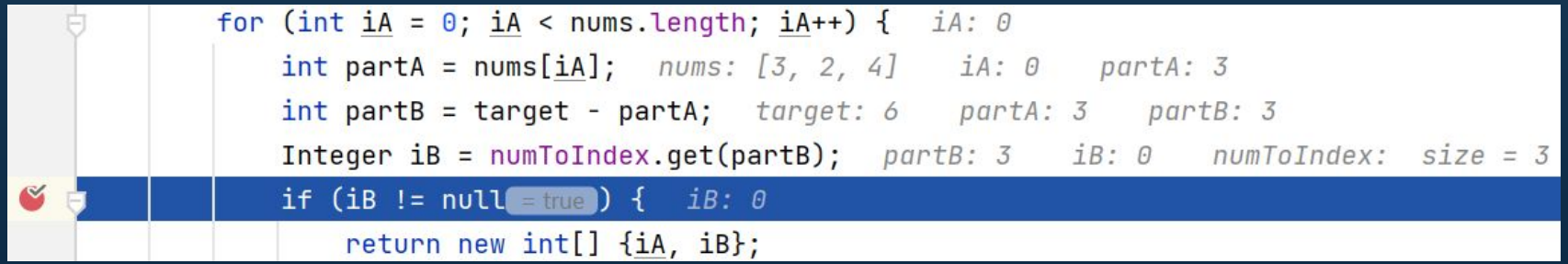
```
for (int iA = 0; iA < nums.length; iA++) {   iA: 0
    int partA = nums[iA];   nums: [3, 2, 4]    iA: 0    partA: 3
    int partB = target - partA;   target: 6    partA: 3    partB: 3
    Integer iB = numToIndex.get(partB);   partB: 3    iB: 0    numToIndex:  size = 3
    if (iB != null  = true ) {   iB: 0
        return new int[] {iA, iB};
```

```
SetBreakpoint
SetFieldAccessWatch
SetFieldModificationWatch

SuspendThread
SuspendThreadList
ResumeThread
ResumeThreadList

GetLocalObject
GetLocalInt
```

JVMTI Debugger




```
SetBreakpoint
SetFieldAccessWatch
SetFieldModificationWatch

SuspendThread
SuspendThreadList
ResumeThread
ResumeThreadList

GetLocalObject
GetLocalInt
```

JVMTI Debugger

```
Condition:
i == 459
```

1. Парсим выражение
2. Читаем локальную переменную i
3. Сравниваем с константой

JVMTI Debugger

```
Condition:
i == 459
```

1. Парсим выражение
2. Читаем локальную переменную i
3. Сравниваем с константой

```
Evaluate and log:
myObject.toString()
```

1. Парсим выражение
2. Читаем локальную переменную myObject (упс)
3. Читаем поле this.myObject через JNI
4. Вызываем метод через JNI

JVMTI Debugger

```
Condition:
i == 459
```

1. Парсим выражение
2. Читаем локальную переменную i
3. Сравниваем с константой

```
Evaluate and log:
myObject.toString()
```

1. Парсим выражение
2. Читаем локальную переменную myObject (упс)
3. Читаем поле this.myObject через JNI
4. Вызываем метод через JNI

JVMTI Debugger

```
Evaluate and log:
myObject.toString()
```

1. Парсим выражение
2. Читаем локальную переменную myObject (упс)
3. Читаем поле this.myObject через JNI
4. ...

JVMTI Debugger

```
Evaluate and log:
myObject.toString()
```

1. Парсим выражение
2. Читаем локальную переменную myObject (упс)
3. Читаем поле this.myObject через JNI
4. Интерпретируем выполнение метода

JVMTI Загрузка байткода

```
jvmti->GetBytecodes(
    jvmti,
    methodId,
    byteCodeSizeOut,
    byteCodeOut
);
```

JVMTI Интерпретатор

```
while((nextOp = readNextOp()) != -1) {
   interpret(nextOp, stack);
}
```

JVMTI Интерпретатор

```
while((nextOp = readNextOp()) != -1) {
   interpret(nextOp, stack);
}
```

JVMTI Интерпретатор

```
while((nextOp = readNextOp()) != -1) {
    interpret(nextOp, stack);
}
```

JVMTI Интерпретатор

```
while((nextOp = readNextOp()) != -1) {
    interpret(nextOp, stack);
}
```

JVMTI Интерпретатор

```
while((nextOp = readNextOp()) != -1) {
   interpret(nextOp, stack);
}
```

без перекладывания

без остановок

без риска

JVMTI Интерпретатор

```
while((nextOp = readNextOp()) != -1) {
    if (timeout()) {
        throw new TimeoutException();
    }
    interpret(nextOp, stack);
}
```

без перекладывания

без остановок

без риска

JVMTI Интерпретатор

```
while((nextOp = readNextOp()) != -1) {
    if (timeout()) {
        throw new TimeoutException();
    }
    if (!allowed(nextOp)) {
        throw new IllegalAccessException();
    }
    interpret(nextOp, stack);
}
```

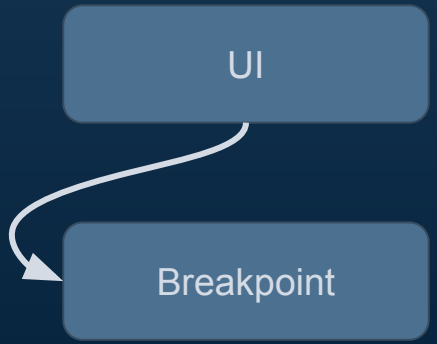
без перекладывания

без остановок

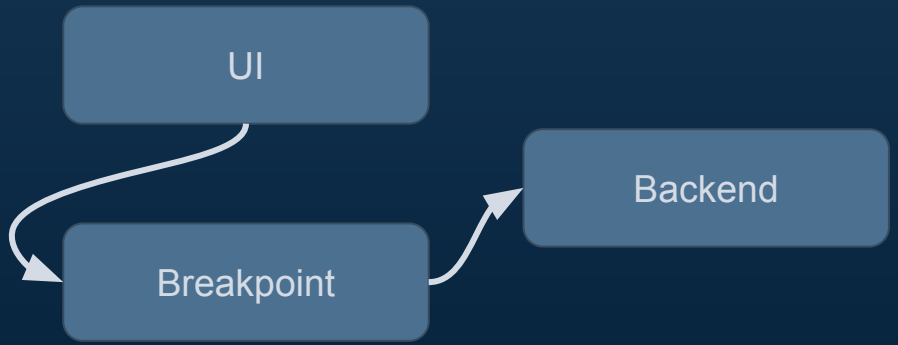
без риска

Архитектура отладчика

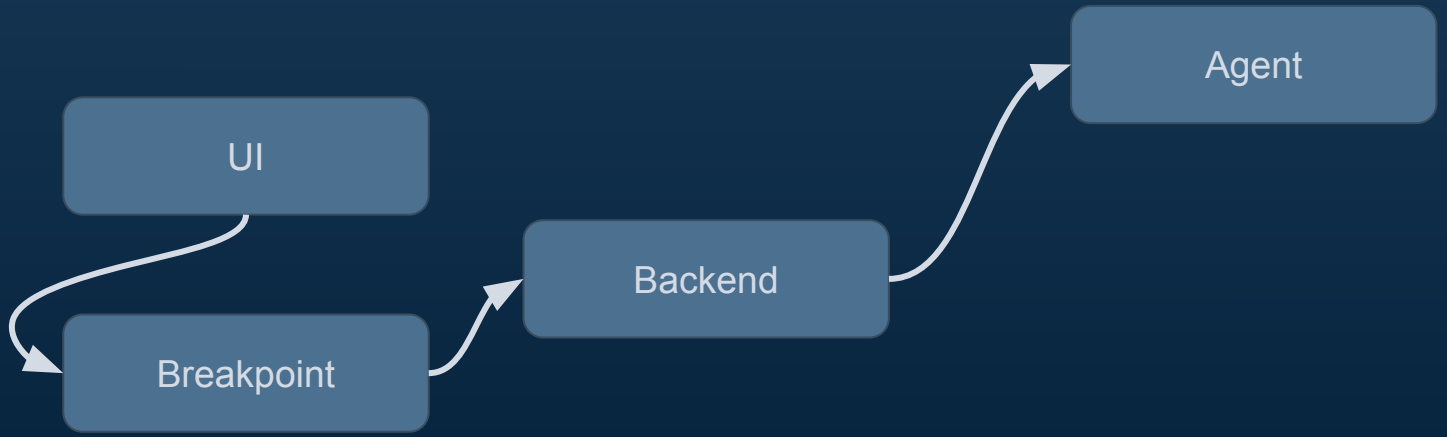
Архитектура отладчика



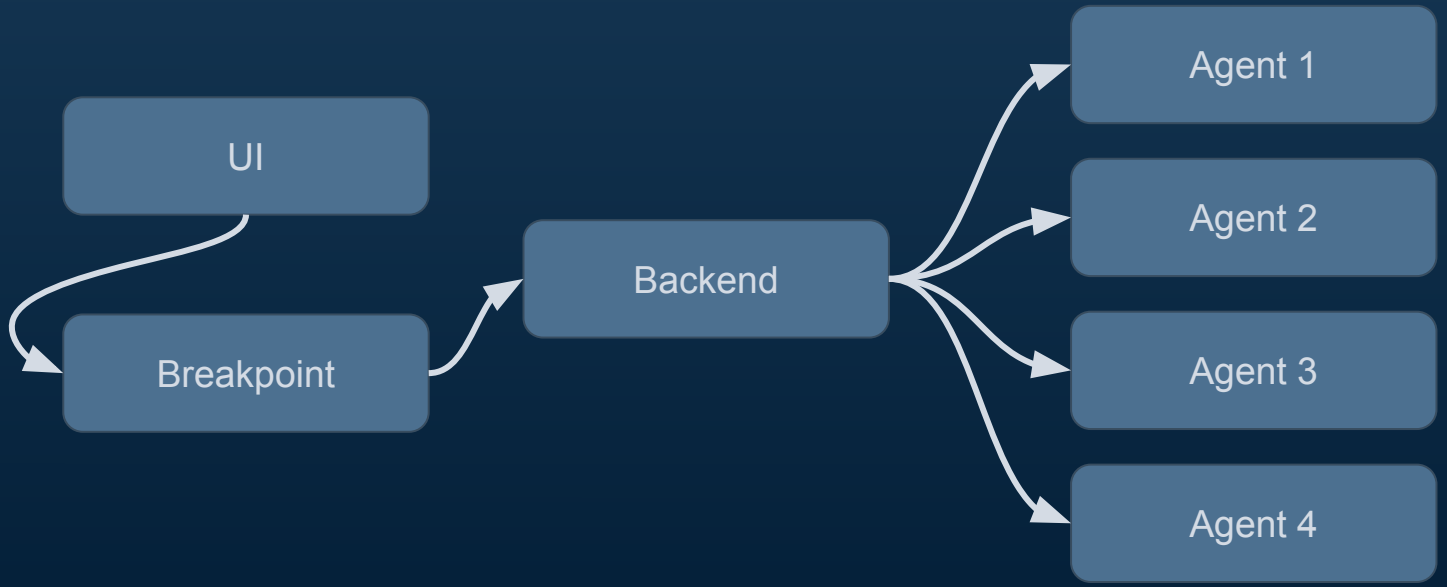
Архитектура отладчика



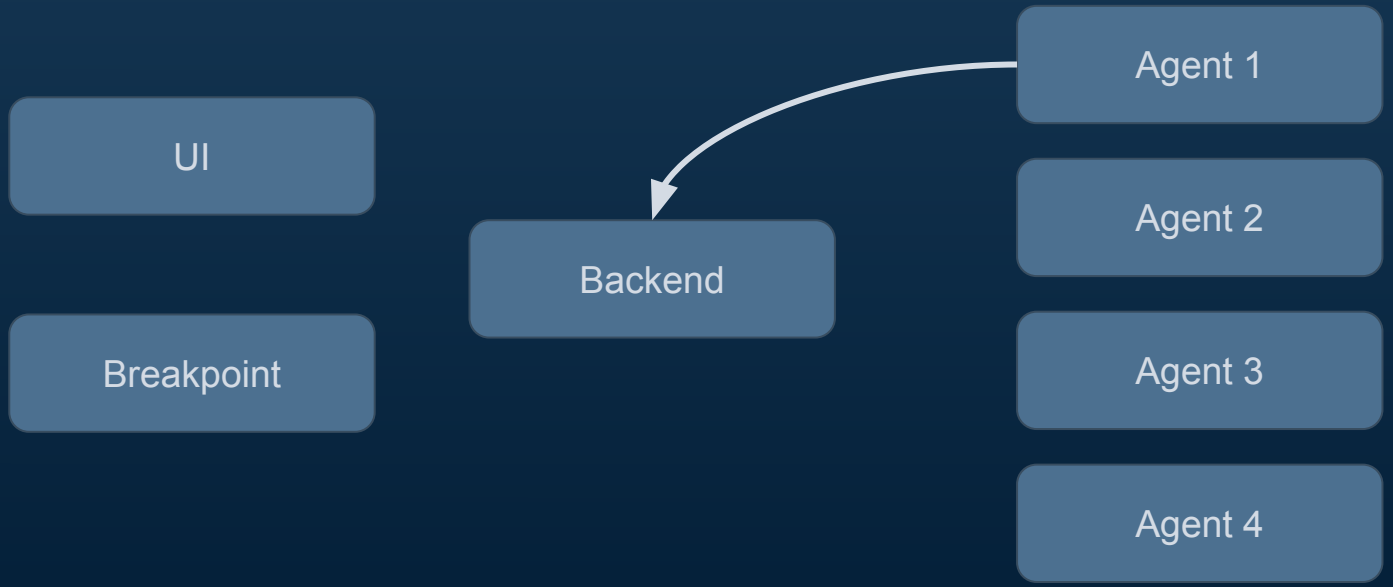
Архитектура отладчика



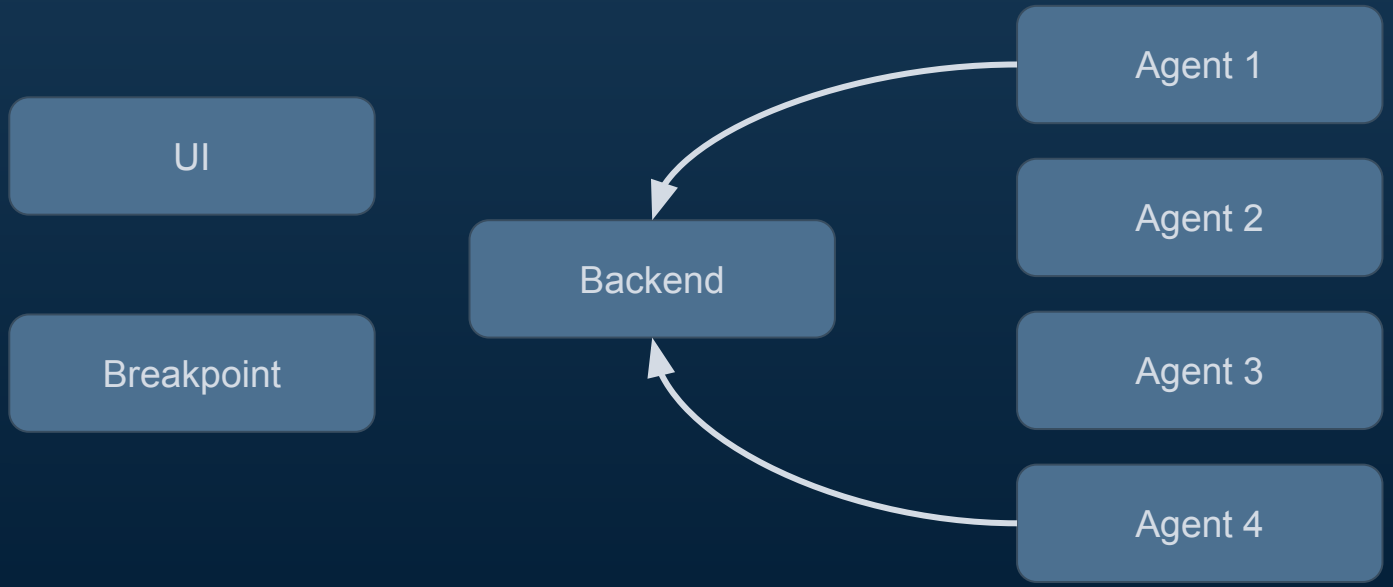
Архитектура отладчика



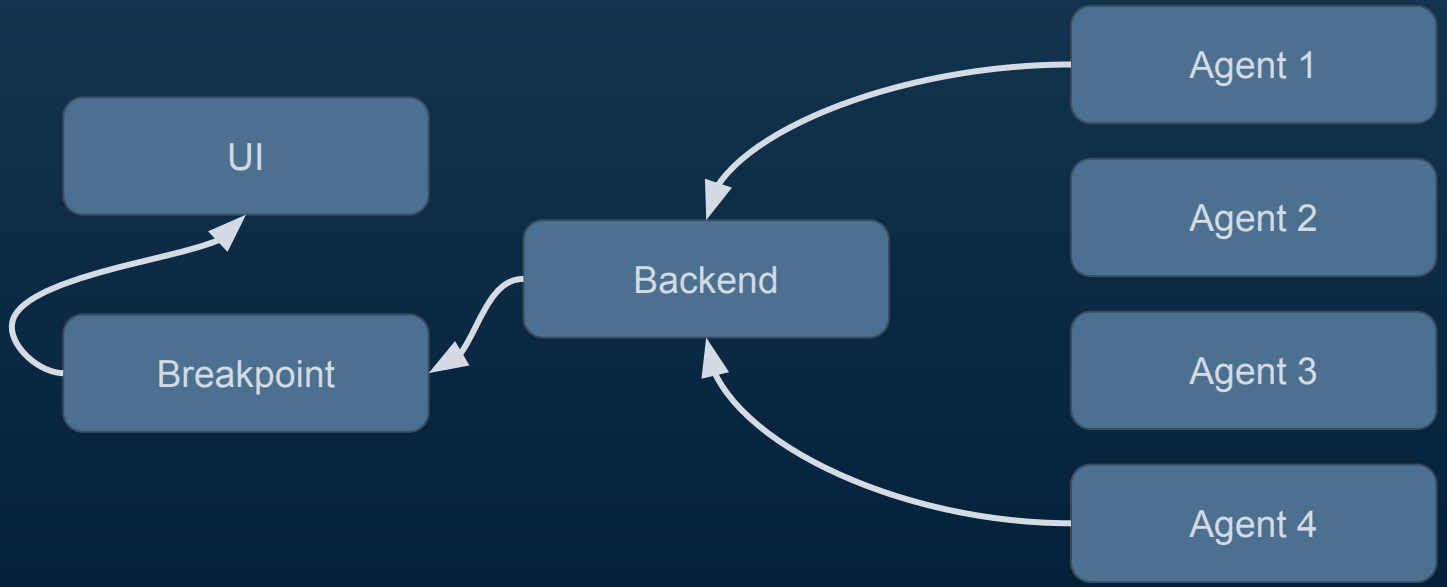
Архитектура отладчика



Архитектура отладчика



Архитектура отладчика



Аудит и безопасность

- Block list
  - Запрещенные классы/методы
- Allow list
  - Разрешенные классы/методы
- Разделение прав
  - Отладка
  - Изменения block/allow list’ов
  - Доступ к отдельным агентам
- Каждый breakpoint остаётся в логах аудита

Квоты

- Latency
- Memory
- Hit count

Так как же отлаживать продакшн?

- Remote Debugger
  - Не аудируемо
  - Страшно
- Переложить продакшн
  - Долго
  - Страшно
  - Скучно
- HCR
  - Быстро
  - Просто
  - Всё ещё страшно

Альтернатива: JVM TI

- Долго
  - Но один раз
- Сложно
  - Но один раз
- Страшно
  - Но один раз

Итого: больно только в первый раз, а потом...

Вопросы?

Артём Дроздов
Lightrun