美团Robust热更新产品化之路

美团点评 / 吴坤

吴坤

美团平台Android基础设施组负责人

2015年加入美团，先后负责客户端安全组件、美团APP动态化等项目。为公司各APP提供统一的基础设施服务。

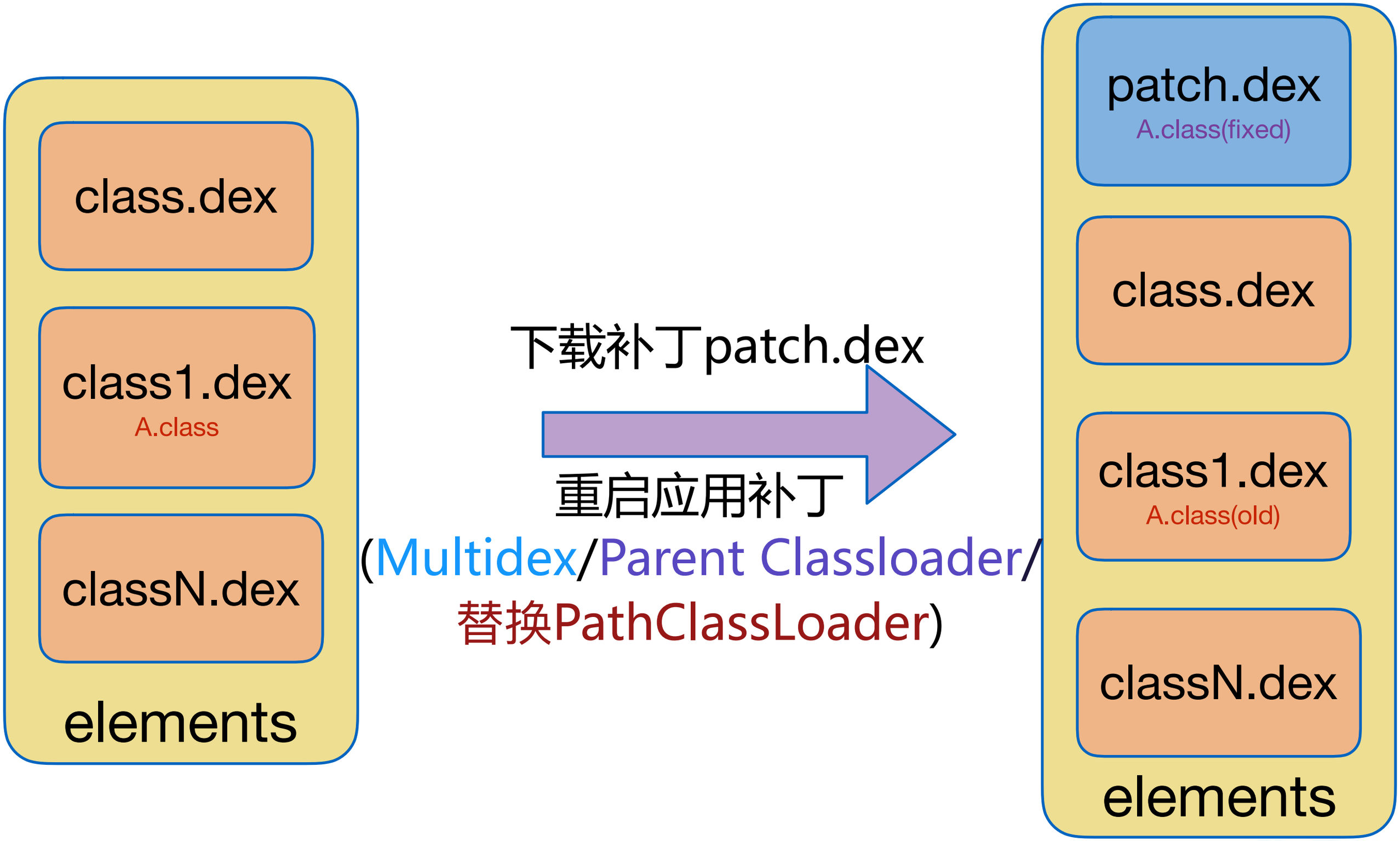
内容

- Robust原理
- Robust产品化
- Robust线上效果

Robust原理－业内状况

- Java
- Native

Robust原理－Java层框架



Robust原理－Java层框架

Android上的多Dex加载实现

class加载的特性

热修复效果通过class替换达到

原理

效果

Robust原理－Java层框架

补丁要在进程重新启动才生效

通过hack操作系统内部过程实现

Robust原理－Native层框架

```
public class A {
    void method() {...}

    void method1() {...}

    void method2() {...}
}
```

jni层将被patch的函数修改为native的

patching

```
public class A {
    native void method();

    native void method1();

    native void method2();
}
```

```
public class APatch {
    void method() {...}

    void method1() {...}

    void method2() {...}
}
```

jni层将被patch的函数指向patched中对应的函数

Robust原理－Native层框架

native层修改指定class的函数为native

native层做方法调用时的转发

原理

热修复效果通过方法替换达到

效果

Robust原理－Native层框架

native层hack Android java虚拟机内部实现

核心逻辑汇编实现

业内方案的共同点

热更新效果通过hack系统内部实现

理想中的热更新系统

不Hack
系统

+

使用
Android
公开API

→

高实时性
+
高兼容性

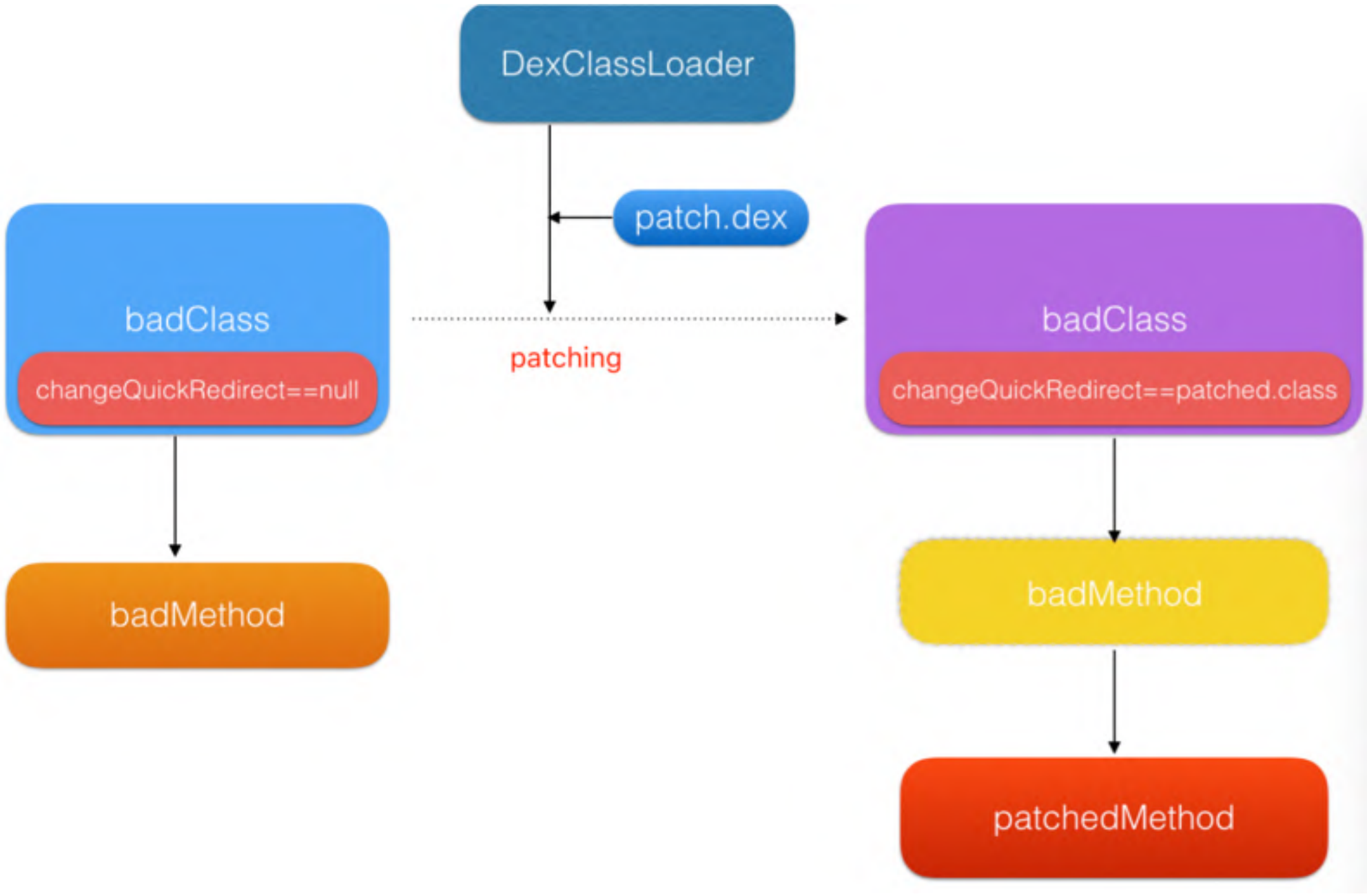
Robust原理

DexClassLoader + 插桩

Developers

A class loader that loads classes from `.jar` and `.apk` files containing a `classes.dex` entry. This can be used to execute code not installed as part of an application.

Robust原理



Robust原理

编译期

- class添加changeQuickRedirect字段
- 给方法添加changeQuickRedirect字段使用逻辑

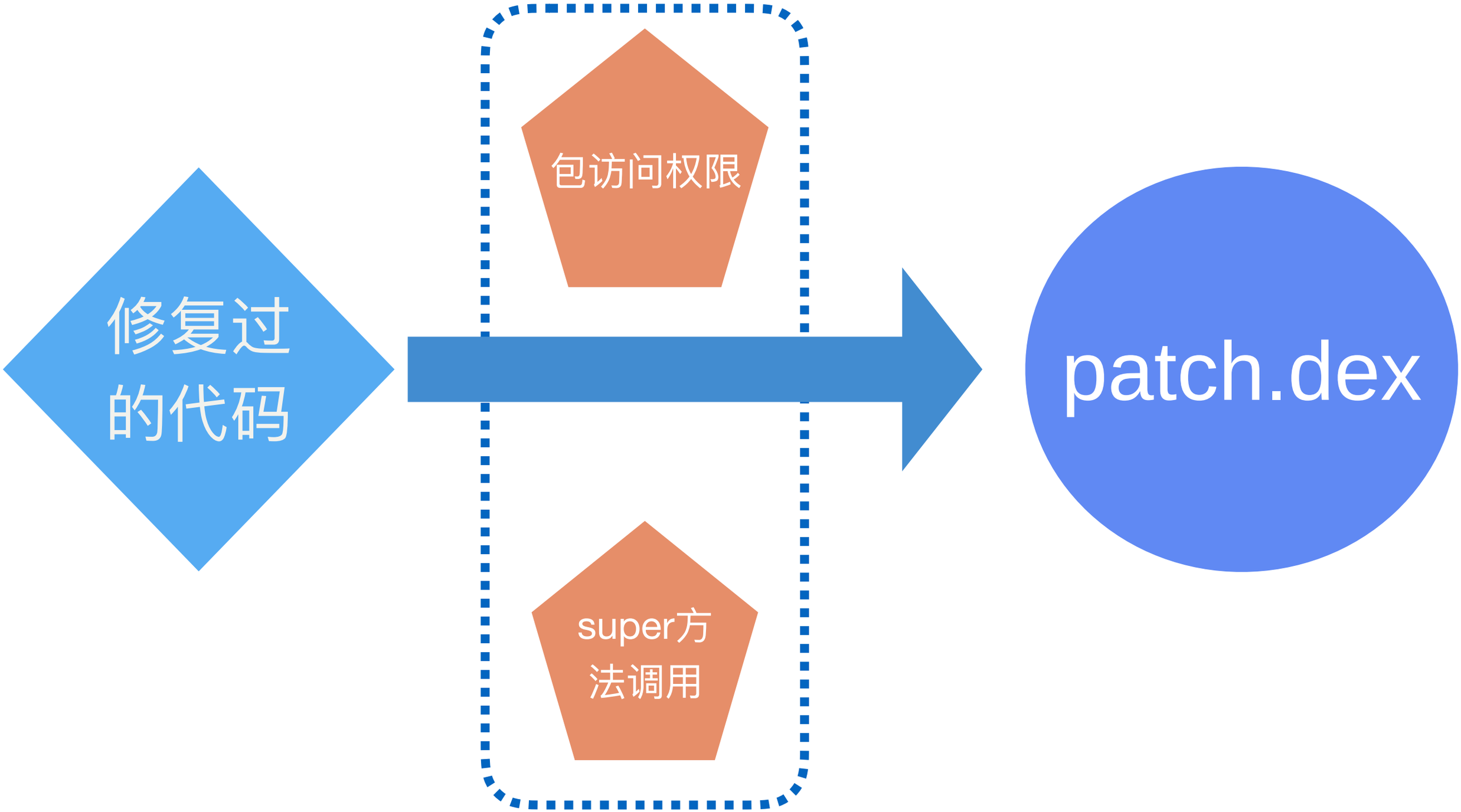
运行期

- DexClassLoader加载补丁
- 初始化指定class的changeQuickRedirect字段

效果

- 热更新效果通过方法替换达到

Robust原理－难点



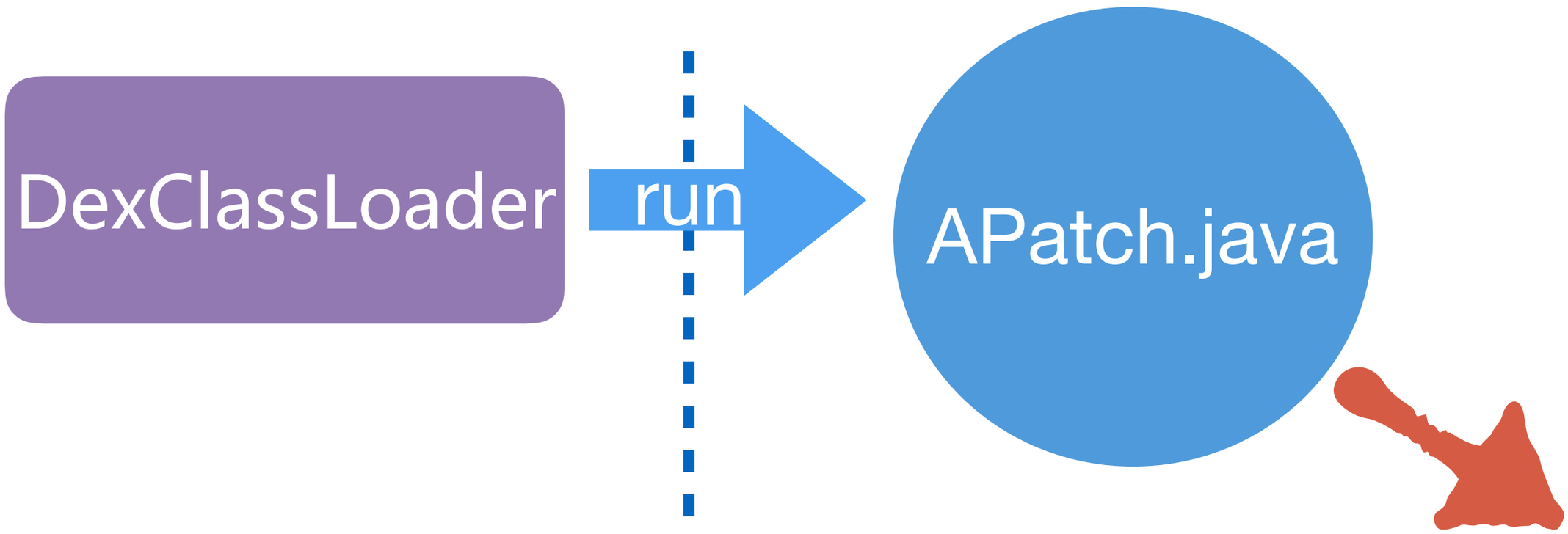
Robust原理－难点－包访问权限问题

```java
package com.meituan.sample.testpackage;

import android.util.Log;

class B {
    public static void testB(int i) {
        Log.d("testB", "in testB " + i);
    }
}
```

```java
package com.meituan.sample.testpackage;

public class APatch {
    public static void testA() {
        B.testB(6);
    }
}
```

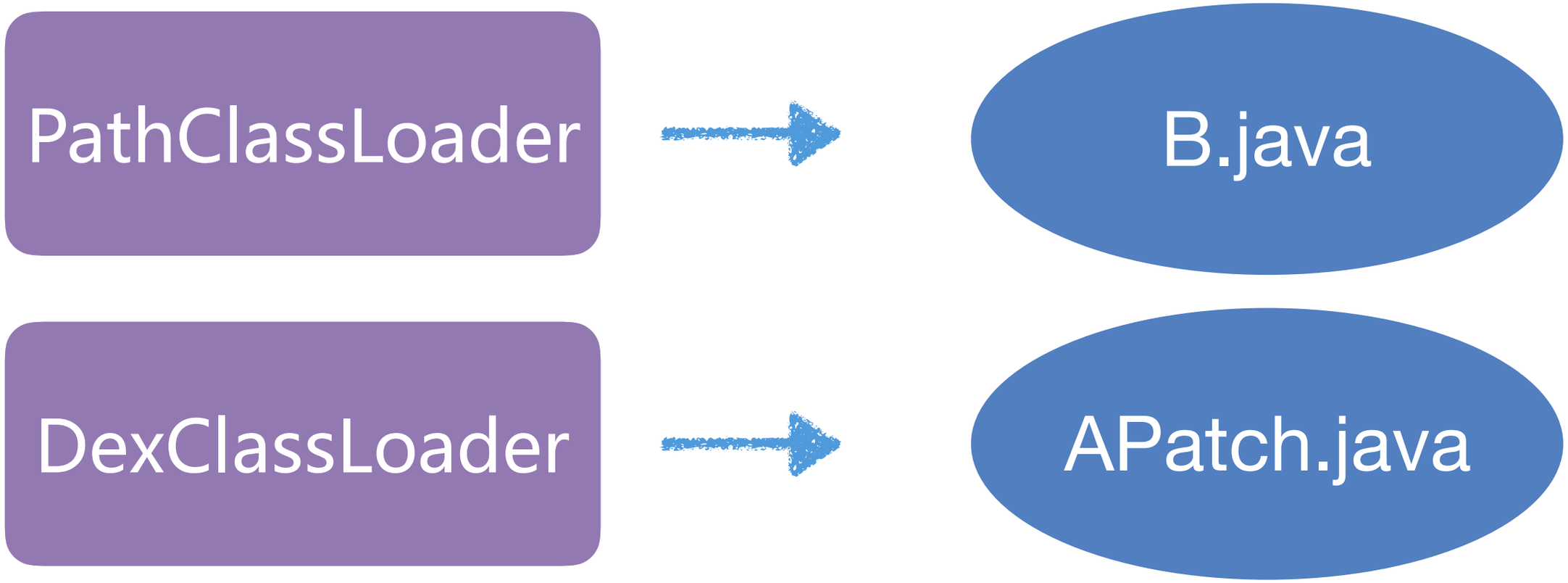
```java
package com.meituan.sample.testpackage;

public class A {
    public static void testA() {
        B.testB(1);
    }
}
```

Robust原理－难点－包访问权限问题




```
10-11 11:51:35.333 1149-1149/com.meituan.robust.sample W/System.err: Caused by: java.lang.IllegalAccessError: tried to access class com.meituan.
sample.testpackage.B from class com.meituan.sample.testpackage.APatch
10-11 11:51:35.333 1149-1149/com.meituan.robust.sample W/System.err:     at com.meituan.sample.testpackage.APatch.testA(APatch.java:5)
10-11 11:51:35.333 1149-1149/com.meituan.robust.sample W/System.err:     ... 15 more
10-11 11:51:35.333 1149-1149/com.meituan.robust.sample D/testxxx: findClass APatch: class com.meituan.sample.testpackage.APatch
10-11 11:51:35.333 1149-1149/com.meituan.robust.sample D/testPackage: m:testA
10-11 11:51:35.333 1149-1149/com.meituan.robust.sample W/dalvikvm: DexOpt: resolve class illegal access: Lcom/meituan/sample/testpackage/APatch; ->
 Lcom/meituan/sample/testpackage/B;
10-11 11:51:35.333 1149-1149/com.meituan.robust.sample I/dalvikvm: Could not find method com.meituan.sample.testpackage.B.testB, referenced from
method com.meituan.sample.testpackage.APatch.testA
10-11 11:51:35.333 1149-1149/com.meituan.robust.sample W/dalvikvm: VFY: unable to resolve static method 4: Lcom/meituan/sample/testpackage/B;.
testB (I)V
10-11 11:51:35.333 1149-1149/com.meituan.robust.sample D/dalvikvm: VFY: replacing opcode 0x71 at 0x001a
```

Robust原理－难点－包访问权限问题



包访问权限的引用必须发生在同一个classLoader中

Robust原理－难点－包访问权限问题

包访问权限

classLoader统一　？

反射　？

编译时改变为public

Robust原理－难点－super方法调用问题

```java
public class Activity1 extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        ...
    }
}
```

```java
public class Activity1Patch {
    void onCreate(Activity1 activity1, Bundle savedInstanceState) {
        activity1.onCreate(savedInstanceState);
        ...
    }
}
```

其实想调用的是activity1对象的超类的onCreate函数

新的onCreate的实现

activity1.super.onCtreate(…) X

?

Robust原理－难点－super方法调用问题

一个类如何调用另一个类的对象的super方法?

java语法层面难以解决

Robust原理－难点－super方法调用问题

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
  protected void onCreate(android.os.Bundle);
    Code:
       0: getstatic     #590                // Field com/meituan/sample/MainActivity.changeQuickRedirect:Lcom/meituan/robust/C
hangeQuickRedirect;
       3: aconst_null
       4: if_acmpeq     47
       7: aconst_null
       8: astore_2
       9: aload_0
      10: astore_2
      11: iconst_1
      12: anewarray     #233                // class java/lang/Object
      15: dup
      16: iconst_0
      17: aload_1
      18: aastore
      19: aload_2
      20: getstatic     #590                // Field com/meituan/sample/MainActivity.changeQuickRedirect:Lcom/meituan/robust/C
hangeQuickRedirect;
      23: iconst_0
      24: invokestatic  #596                // Method com/meituan/robust/PatchProxy.isSupport:([Ljava/lang/Object;Ljava/lang/O
bject;Lcom/meituan/robust/ChangeQuickRedirect;Z)Z
      27: ifeq          47
      30: iconst_1
      31: anewarray     #233                // class java/lang/Object
      34: dup
      35: iconst_0
      36: aload_1
      37: aastore
      38: aload_2
      39: getstatic     #590                // Field com/meituan/sample/MainActivity.changeQuickRedirect:Lcom/meituan/robust/C
hangeQuickRedirect;
      42: iconst_0
      43: invokestatic  #600                // Method com/meituan/robust/PatchProxy.accessDispatchVoid:([Ljava/lang/Object;Lja
va/lang/Object;Lcom/meituan/robust/ChangeQuickRedirect;Z)V
      46: return
      47: aload_0
      48: aload_1
      49: invokespecial #15                 // Method android/support/v7/app/AppCompatActivity.onCreate:(Landroid/os/Bundle;)V
```

class中super调用的指令：invokespecial

普通的调用指令：invokevirtual

Robust原理－难点－super方法调用问题

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
.method protected onCreate(Landroid/os/Bundle;)V
    .registers 8
    .param p1, "savedInstanceState"    # Landroid/os/Bundle;

    .prologue
    const/4 v5, 0x1

    const/4 v4, 0x0

    sget-object v2, Lcom/meituan/sample/MainActivity;->changeQuickRedirect:Lcom/meituan/robust/ChangeQuickRedirect;

    if-eqz v2, :cond_1c

    new-array v2, v5, [Ljava/lang/Object;

    aput-object p1, v2, v4

    sget-object v3, Lcom/meituan/sample/MainActivity;->changeQuickRedirect:Lcom/meituan/robust/ChangeQuickRedirect;

    invoke-static {v2, p0, v3, v4}, Lcom/meituan/robust/PatchProxy;->isSupport([
Ljava/lang/Object;Ljava/lang/Object;Lcom/meituan/robust/ChangeQuickRedirect;Z)Z

    move-result v2

    if-eqz v2, :cond_1c

    new-array v2, v5, [Ljava/lang/Object;

    aput-object p1, v2, v4

    sget-object v3, Lcom/meituan/sample/MainActivity;->changeQuickRedirect:Lcom/meituan/robust/ChangeQuickRedirect;

    invoke-static {v2, p0, v3, v4}, Lcom/meituan/robust/PatchProxy;->accessDispatchVoid([
Ljava/lang/Object;Ljava/lang/Object;Lcom/meituan/robust/ChangeQuickRedirect;Z)V

    .line 172
    :goto_1b
    return-void

    .line 47
    :cond_1c
    invoke-super {p0, p1}, Landroid/support/v7/app/AppCompatActivity;->onCreate(Landroid/os/Bundle;)V
```

Android中java虚拟机执行的super指令(dex)
invoke-super
普通的调用指令：
invoke-virtual

Robust原理－难点－super方法调用问题

invokespecial

Operation

Invoke instance method; special handling for superclass, private, and instance initialization method invocations

super

private

init

invokevirtual

Operation

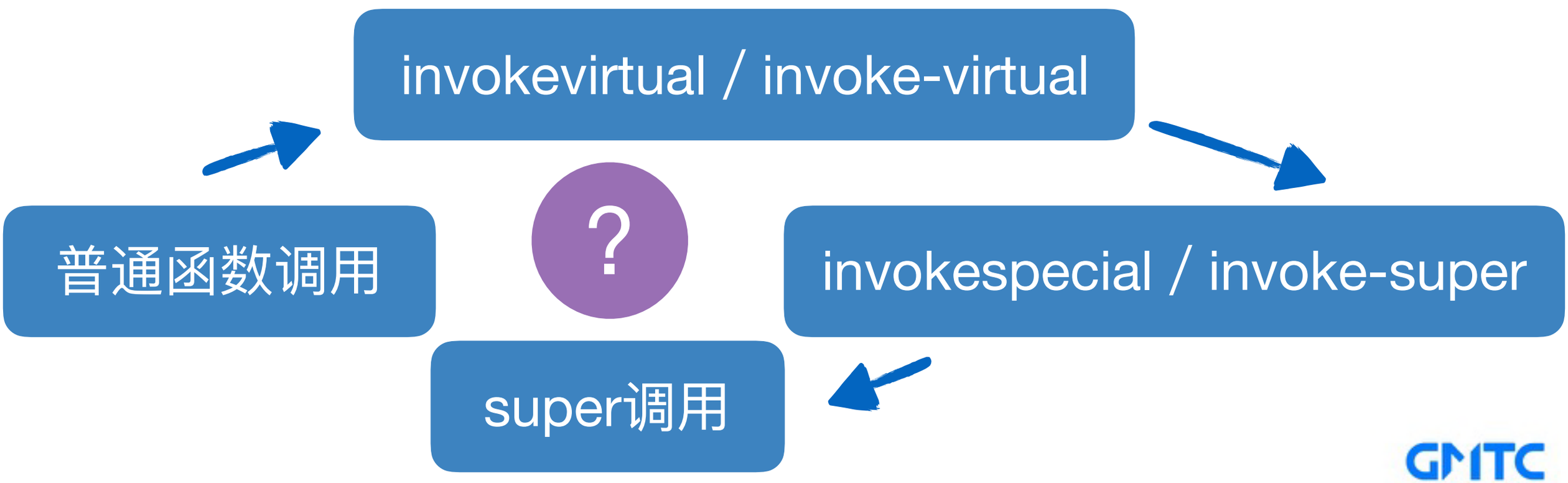
Invoke instance method; dispatch based on class

多态

Robust原理－难点－super方法调用问题

super调用 → invokespecial / invoke-super

普通函数调用 → invokevirtual / invoke-virtual



Robust原理－难点－super方法调用问题

```java
public class SuperClass {
    String uuid;
    public void setUuid(String id) {
        uuid = id;
    }
    public void thisIsSuper() {
        Log.d("SuperClass", "thisIsSuper "+uuid);
    }
}
```

```java
public class TestSuperClass extends SuperClass{
    String subUuid;
    public void setSubUuid(String id) {
        subUuid = id;
    }

    @Override
    public void thisIsSuper() {
        Log.d("TestSuperClass", "thisIsSuper no call");
    }
}
```

```java
public class TestSuperPatch {
    public static void testSuperCall() {
        TestSuperClass testSuperClass = new TestSuperClass();
        String t = UUID.randomUUID().toString();
        Log.d("TestSuperPatch", "UUID " + t);
        testSuperClass.setUuid(t);
        testSuperClass.thisIsSuper();
    }
}
```

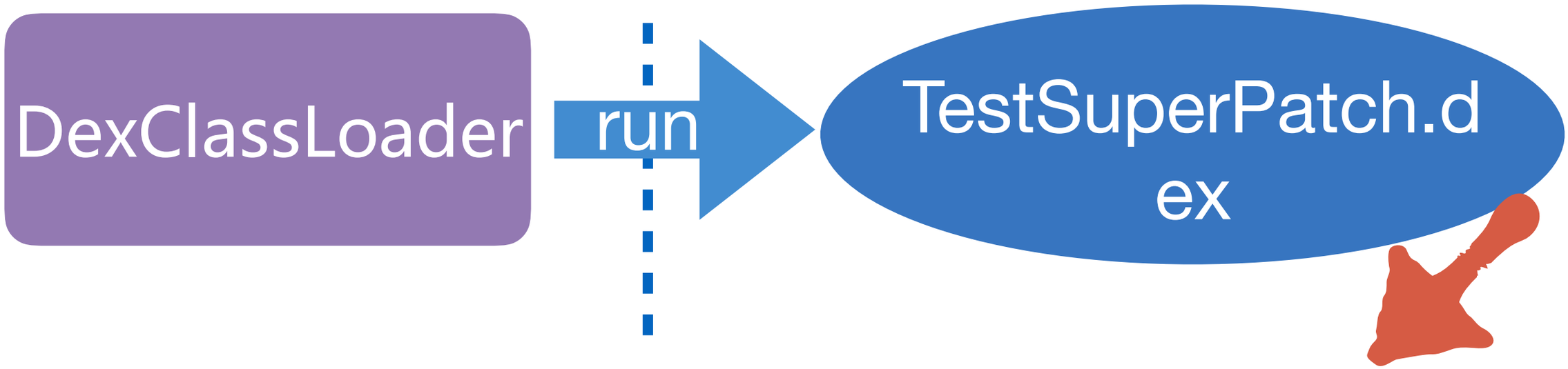
invokevirtual /
invoke-virtual

修改为

invokespecial /
invoke-super

testSuperClass.super.thisIsSuper();

Robust原理－难点－super方法调用问题




```
Caused by: java.lang.NoSuchMethodError: No super method thisIsSuper()V in class
Lcom/meituan/sample/TestSuperClass; or its super classes (declaration of 'com.meituan.
sample.TestSuperClass' appears in /data/app/com.meituan.robust.sample-3/base.apk)
```

invokespecial / invoke-super → Android java 虚拟机 → 从执行当前指令的class的超类中找指定方法

Robust原理－难点－super方法调用问题

testSuperClass.
thisIsSuper();

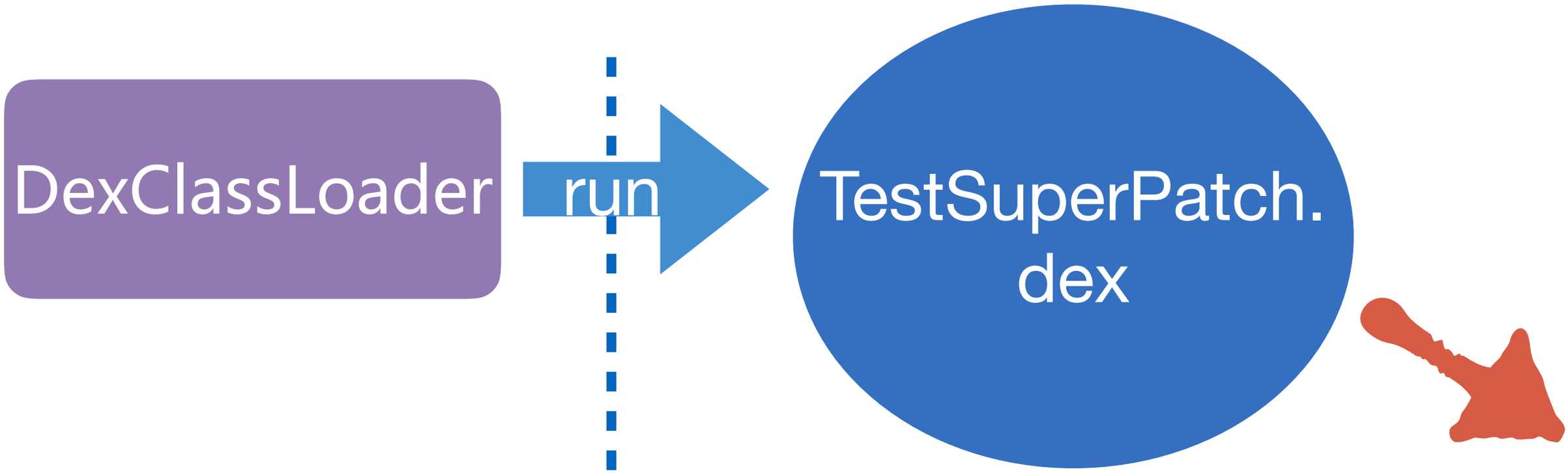
invokespecial /
invoke-super

TestSuperPatch

TestSuperPatch
extends
SuperClass

TestSuperPatch.dex

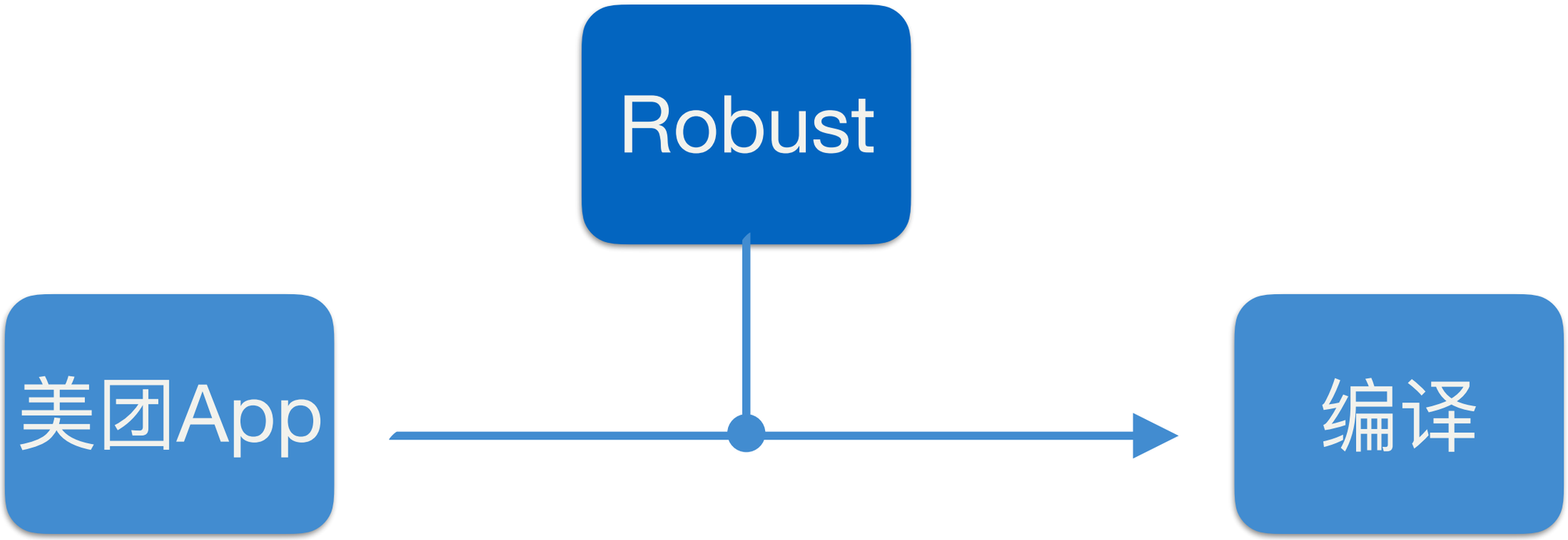
Robust原理－难点－super方法调用问题

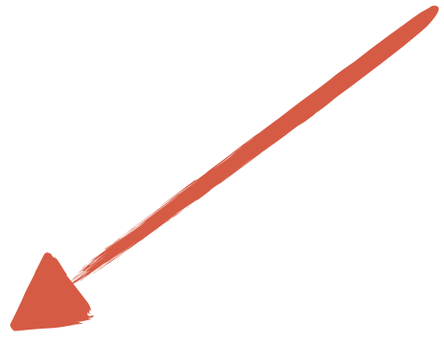



```
08-11 09:12:03.012 1787-1787/? D/TestSuperPatch: UUID c5216480-5c3a-4990-896d-58c3696170c5
08-11 09:12:03.012 1787-1787/? D/SuperClass: thisIsSuper c5216480-5c3a-4990-896d-58c3696170c5
```

- Robust原理
- Robust产品化
- Robust线上效果

Robust产品化－插桩问题





```
Conversion to Dalvik format failed:Unable to execute dex: method ID not in [0, 0xffff]: 65536
```

Robust产品化－插桩问题

使用Robust前后的dex对比

Robust产品化－插桩问题

f.class

OrderCenterListAdapter.java

“多”出来的函数

```
public boolean isEditMode() {
        return isEditMode;
    }
private int incrementDelCount() {
        return delCount.incrementAndGet();
    }
private boolean isNeedDisplayRemainingTime(OrderData orderData) {
        return null != orderData.remindtime && getRemainingTimeMillis(orderData.remindtime) > 0;
    }
private boolean isNeedDisplayUnclickableButton(OrderData orderData) {
        return null != orderData.remindtime && getRemainingTimeMillis(orderData.remindtime) <= 0;
    }
private boolean isNeedDisplayExpiring(boolean expiring) {
        return expiring && isNeedDisplayExpiring;
    }
private View getViewByTemplate(int template, View convertView, ViewGroup parent) {
        View view = null;
        switch (template) {
            case TEMPLATE_DEFALUT:
            default:
                view = mInflater.inflate(R.layout.order_center_list_item, null);
        }
        return view;
    }
```

Robust产品化－插桩问题

```
if (methodInliningUnique) {
    // Inline methods that are only invoked once.
    programClassPool.classesAccept(
        new AllMethodVisitor(
        new AllAttributeVisitor(
        new MethodInliner(configuration.microEdition,
                          configuration.allowAccessModification,
                          true,
                          methodInliningUniqueCounter))));
}
if (methodInliningShort) {
    // Inline short methods.
    programClassPool.classesAccept(
        new AllMethodVisitor(
        new AllAttributeVisitor(
        new MethodInliner(configuration.microEdition,
                          configuration.allowAccessModification,
                          false,
                          methodInliningShortCounter))));
}
```

只被调用一次

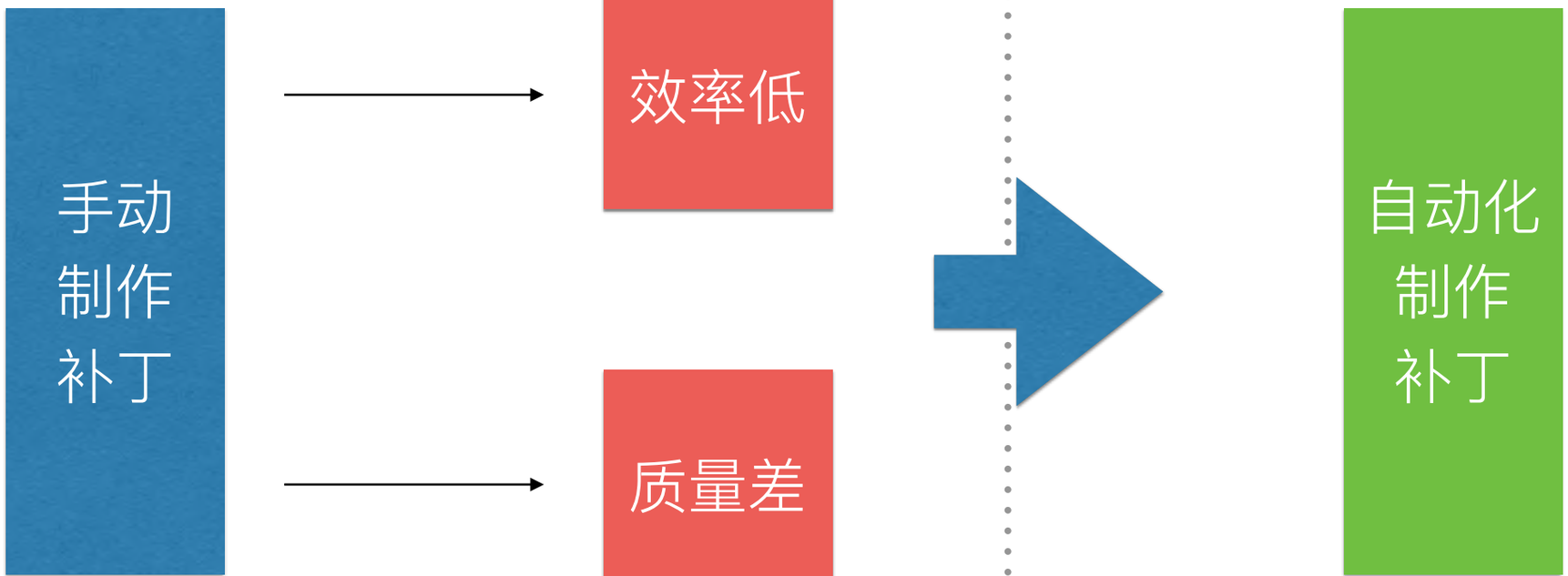
简单

Robust产品化－插桩问题

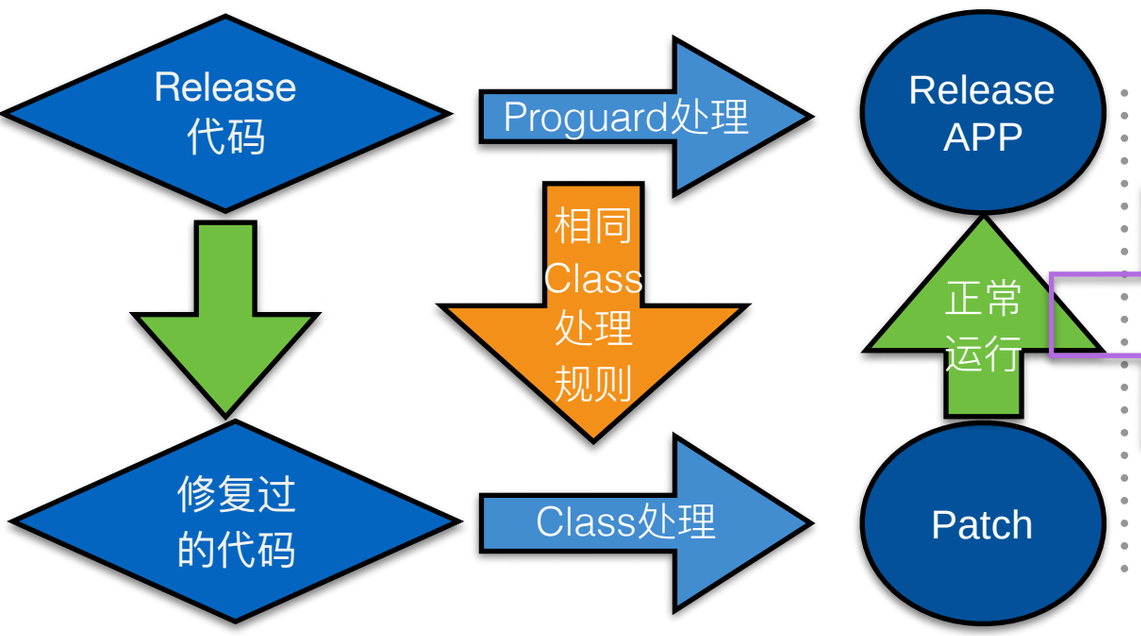
优化Robust插件 → 过滤可能内联的函数 → 成功生成美团apk

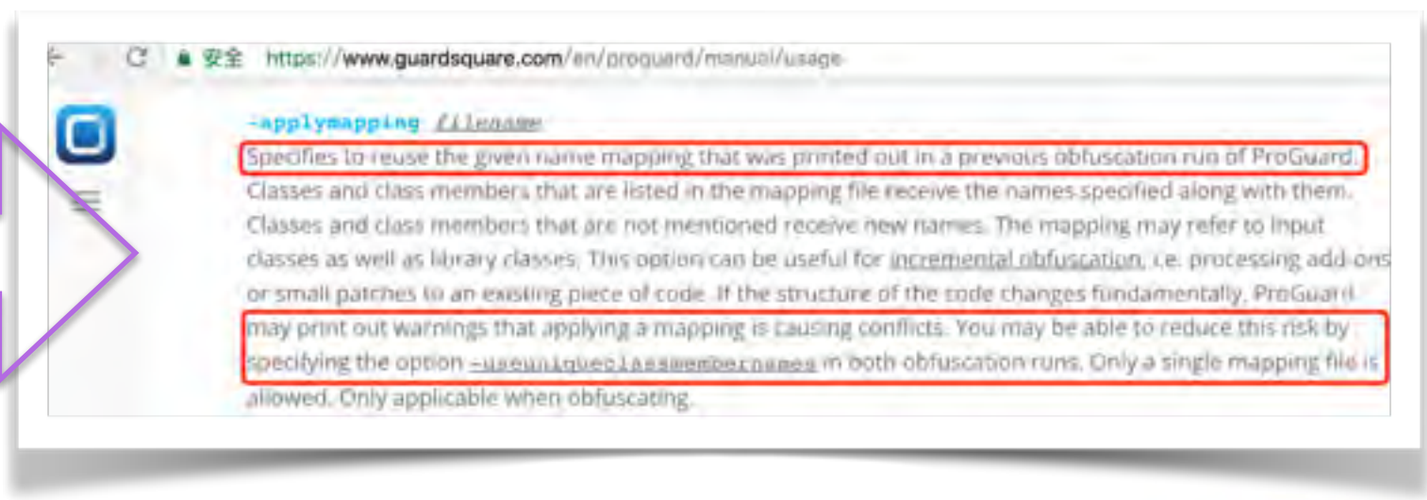
ProGuard内联问题

Robust产品化－补丁自动化

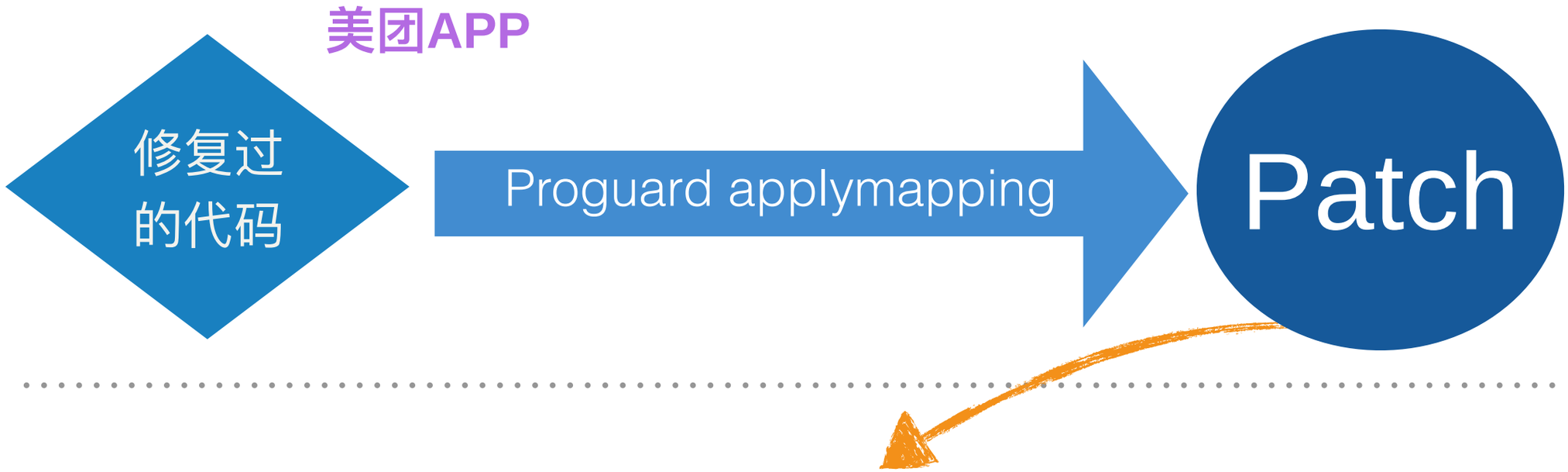


如何保证Patch代码跟Release产品代码两次编译的处理逻辑一致?





Robust产品化－补丁自动化



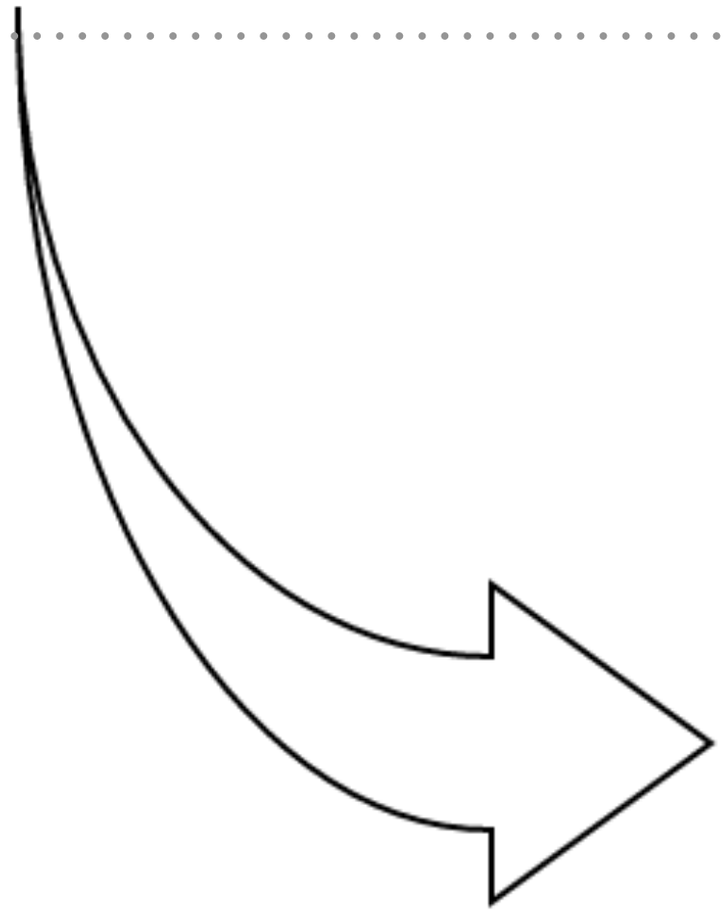
官方applymapping不work

```
Obfuscating...
Applying mapping [/Users/wukun/workspace/aimeituan/aimeituan/mapping.txt]
Printing mapping to [/Users/wukun/workspace/aimeituan/aimeituan/build/outputs/mapping/meituan/release/mapping.txt]...
Warning: cloudwns.d.e: method 'cloudwns.d.g q()' is not being kept as 'a', but remapped to 'b'
Warning: cloudwns.d.e: method 'cloudwns.d.g q()' is not being kept as 'b', but remapped to 'hr'
Warning: cloudwns.d.e: method 'cloudwns.d.g q()' is not being kept as 'hr', but remapped to 'hs'
Warning: cloudwns.d.e: method 'cloudwns.d.g q()' is not being kept as 'hs', but remapped to 'e'
Warning: cloudwns.d.e: method 'cloudwns.d.g q()' is not being kept as 'e', but remapped to 'ht'
Warning: cloudwns.d.e: method 'cloudwns.d.g q()' is not being kept as 'ht', but remapped to 'hw'
Warning: cloudwns.d.e: method 'cloudwns.d.g q()' is not being kept as 'hw', but remapped to 's'
Warning: cloudwns.g.b: method 'com.tencent.wns.data.B2Ticket b(long)' is not being kept as 'a', but remapped to 'l'
Warning: cloudwns.q.a: method 'long g()' is not being kept as 'g', but remapped to 'a'
Warning: cloudwns.q.a: method 'long g()' is not being kept as 'a', but remapped to 'b'
Warning: cloudwns.q.a: method 'long g()' is not being kept as 'b', but remapped to 'a'
Warning: cloudwns.q.a: method 'long g()' is not being kept as 'a', but remapped to 'g'
Warning: cloudwns.q.a: method 'long g()' is not being kept as 'g', but remapped to 'a'
Warning: com.dianping.agentsdk.agent.HoloAgent: method 'android.content.Context getContext()' is not being kept as 'setNavigationIcon',
Warning: com.dianping.base.push.pushservice.Push: method 'void startPushService(android.content.Context,java.lang.String)' is not being
Warning: com.dianping.feed.adapter.AbstractFeedListAdapter: method 'void setAccountService(com.dianping.feed.common.AccountService)' is
apped to 'exit'
Warning: com.dianping.feed.adapter.AbstractFeedListAdapter: method 'void setFeedService(com.dianping.feed.common.BaseFeedService)' is no
 to 'exit'
```

Robust产品化－补丁自动化

```
public class TestShrink {
    int op;
    void test(int p) {
        int kl = 0;
        if (op > 6) {
            Log.d("TestShrink", "TestShrink:op > 6");
        }
        if (p == 9) {
            for (int i = 0; i < op; i++) {
                kl++;
            }
            Log.d("TestShrink", "TestShrink:p == 9"+kl);
        }
        Log.d("TestShrink", "TestShrink:"+p);
    }
}
```

```
public class TestProguard {
    public static void invoke() {
        TestShrink testShrink = new TestShrink();
        testShrink.op = new Random(100).nextInt();
        testShrink.test(9);
    }
}
```



```
public final class a {
    public int a;

    public a() {
    }

    public final void a() {
        int var1 = 0;
        if(this.a > 6) {
            Log.d("TestShrink", "TestShrink:op > 6");
        }

        for(int var2 = 0; var2 < this.a; ++var2) {
            ++var1;
        }

        Log.d("TestShrink", "TestShrink:p == 9" + var1);
        Log.d("TestShrink", "TestShrink:9");
    }
}
```

Robust产品化－补丁自动化

线上代码

```
public class TestProguard {
    public static void invoke() {
        TestShrink testShrink = new TestShrink();
        testShrink.op = new Random(100).nextInt();
        testShrink.test(9);
    }
}
```

```
public final class a {
    public int a;

    public a() {
    }

    public final void a() {
        int var1 = 0;
        if(this.a > 6) {
            Log.d("TestShrink", "TestShrink:op > 6");
        }

        for(int var2 = 0; var2 < this.a; ++var2) {
            ++var1;
        }

        Log.d("TestShrink", "TestShrink:p == 9" + var1);
        Log.d("TestShrink", "TestShrink:9");
    }
}
```

修复后代码

```
public class TestProguard {
    public static void invoke() {
        TestShrink testShrink = new TestShrink();
        testShrink.op = new Random(100).nextInt();
        testShrink.test(9);
        testShrink.test(10);
    }
}
```

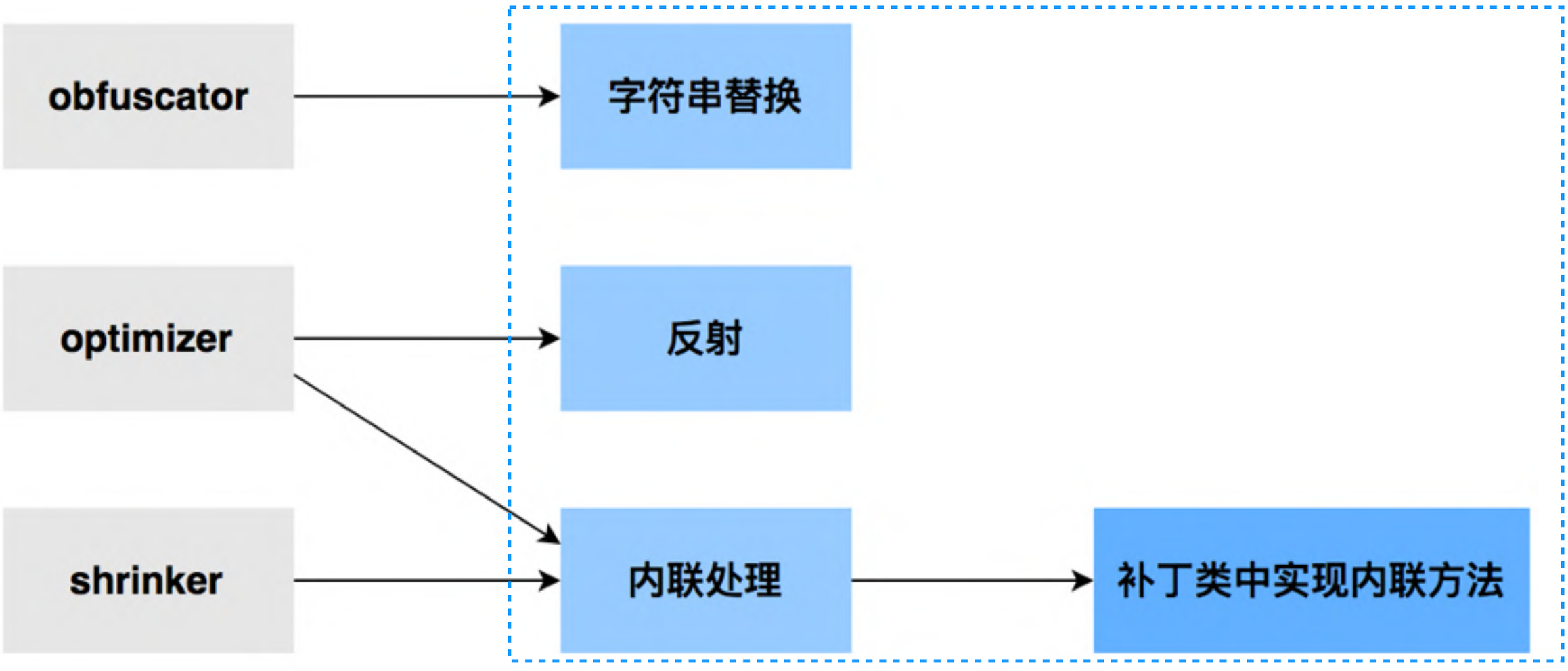
```
public final class a {
    public int a;

    public a() {
    }

    public final void a(int var1) {
        int var2 = 0;
        if(this.a > 6) {
            Log.d("TestShrink", "TestShrink:op > 6");
        }

        if(var1 == 9) {
            for(int var3 = 0; var3 < this.a; ++var3) {
                ++var2;
            }

            Log.d("TestShrink", "TestShrink:p == 9" + var2);
        }

        Log.d("TestShrink", "TestShrink:" + var1);
    }
}
```

Robust产品化－补丁自动化

影响内联规则

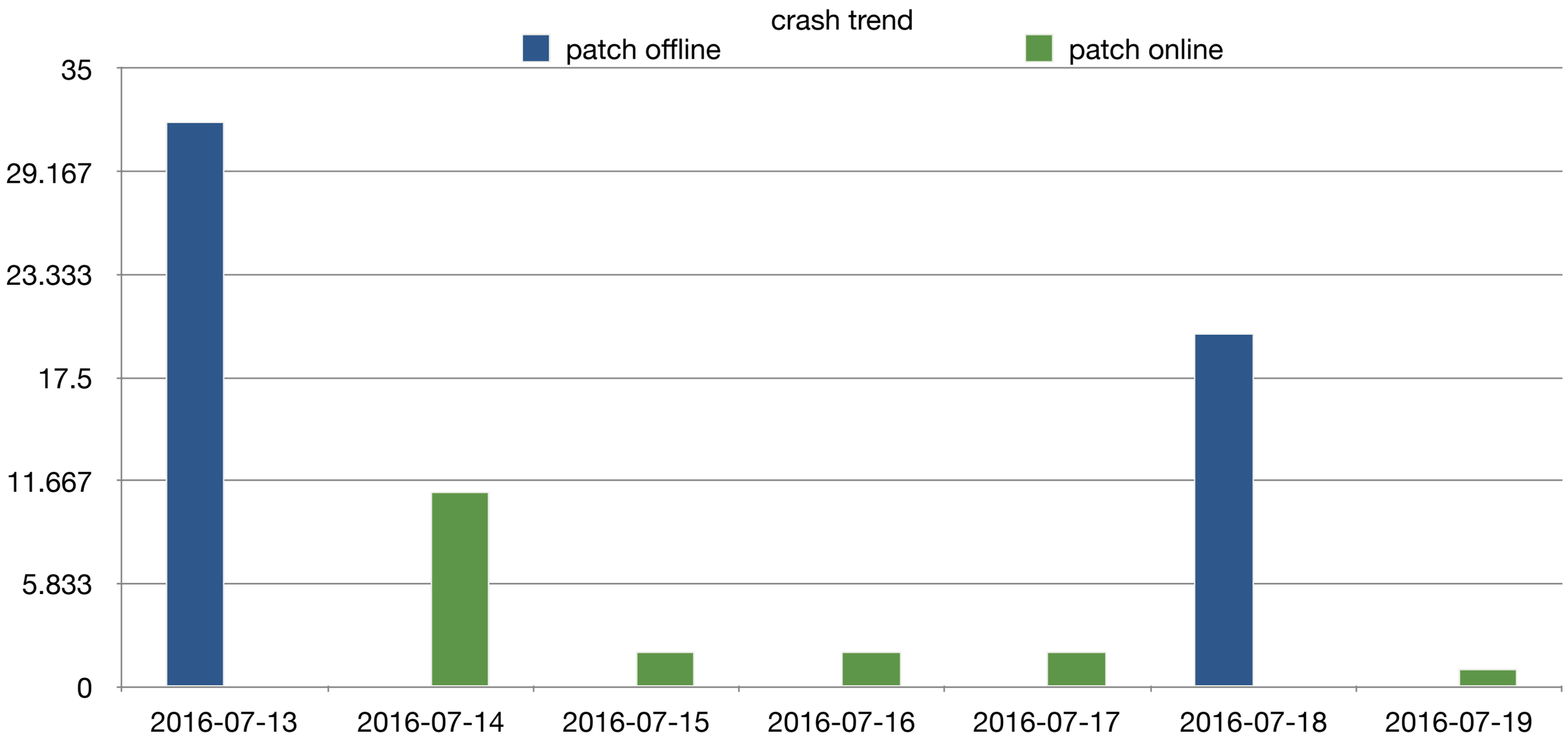
```
if (methodInliningUnique) {
    // Inline methods that are only invoked once.
    programClassPool.classesAccept(
        new AllMethodVisitor(
        new AllAttributeVisitor(
        new MethodInliner(configuration.microEdition,
                          configuration.allowAccessModification,
                          true,
                          methodInliningUniqueCounter))));
}
if (methodInliningShort) {
    // Inline short methods.
    programClassPool.classesAccept(
        new AllMethodVisitor(
        new AllAttributeVisitor(
        new MethodInliner(configuration.microEdition,
                          configuration.allowAccessModification,
                          false,
                          methodInliningShortCounter))));
}
```

Robust产品化－补丁自动化



- Robust原理
- Robust产品化
- Robust线上效果

Robust线上效果


crash trend
patch offline
patch online
35
29.167
23.333
17.5
11.667
5.833
0
2016-07-13
2016-07-14
2016-07-15
2016-07-16
2016-07-17
2016-07-18
2016-07-19

Robust线上效果


美团点评
技术团队

THANKS!