

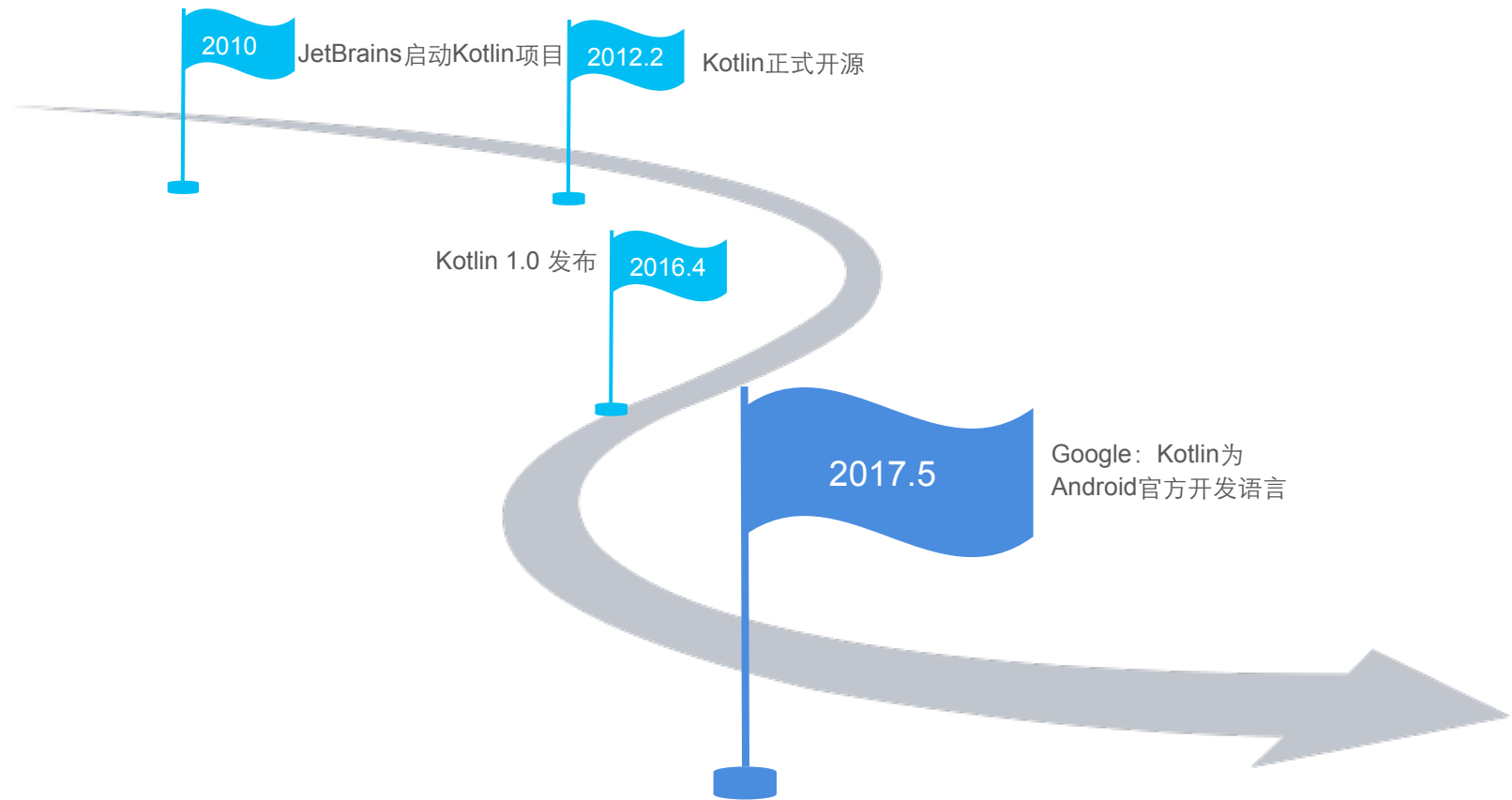
Kotlin

实战分享

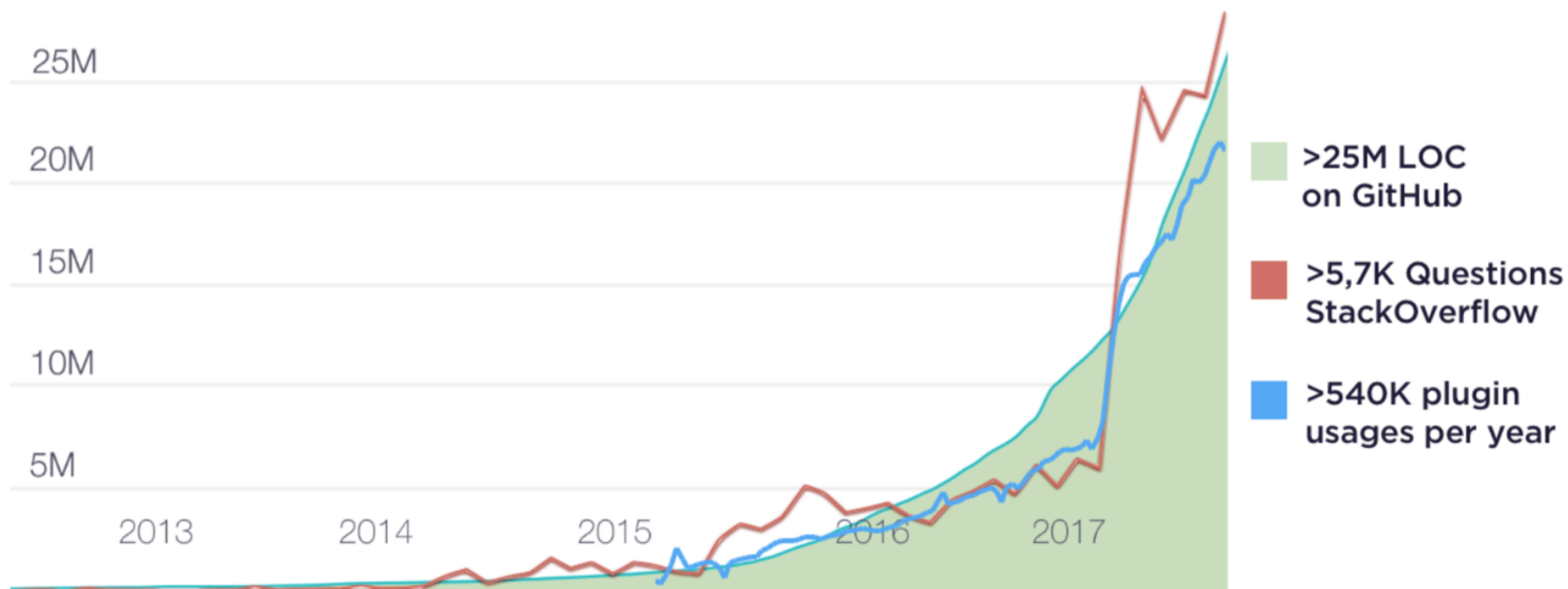
Tencent-应用宝

张元超

Kotlin 简史




Kotlin发展趋势



Kotlin发展趋势

TIOBE 指数 2017.12

Dec 2017	Dec 2016	Change	Programming Language	Ratings
1	1		Java	13.268%
2	2		C	10.158%
3	3		C++	4.717%
....				
28	80	 52	Kotlin	0.994%
29			COBOL	0.961%
30			D	0.950%

Kotlin和C成为年度编程语言候选语言

“

A Better Java

”

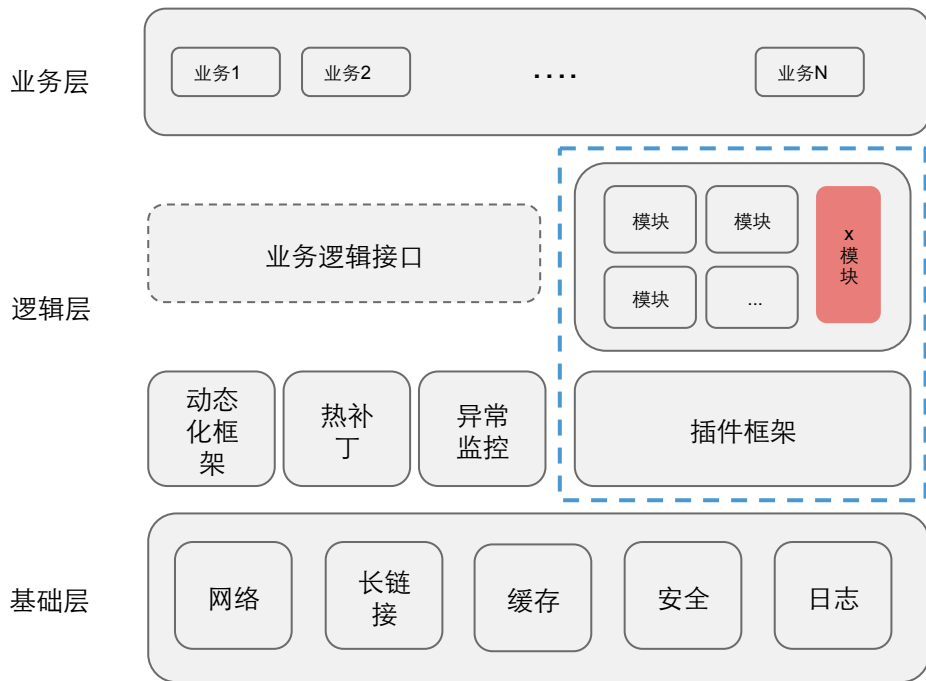
Kotlin特点

- Java兼容
- 安全高效
- 简洁可读
- 工具友好

- 一、项目实战
- 二、优化效果
- 三、总结

项目背景

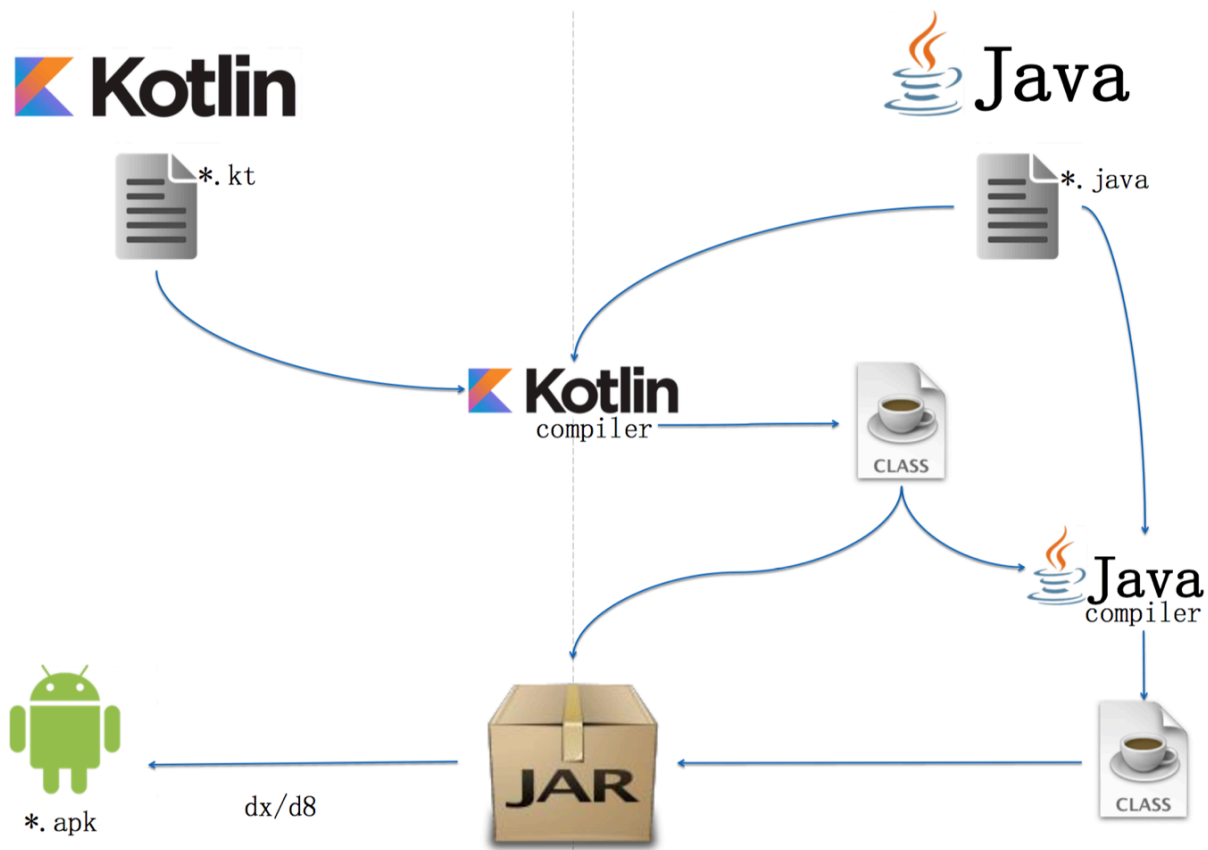
应用宝框架



应用宝现状

- 1、应用宝功能丰富，代码行数超过100万行
- 2、应用宝发展迅速今年代码增量10万行以上
- 3、版本节奏快：2周一个版本

兼容Java



兼容Java : Getter、Setter

```
class MainActivity : BaseActivity() {  
    val TAG = "MainActivity"  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
        printName()  
    }  
  
    var status : String? = null  
    ...  
}
```

var
val

```
public class LoginActivity extends Activity {  
    ...  
    Intent intent = new Intent(LoginActivity.this,  
        MainActivity.class);  
    startActivity(intent);  
    ...  
  
    mainActivity.setStatus(state);  
  
    ...  
}
```

getter + setter
getter



兼容Java：伴生对象

```
class MainActivity : BaseActivity() {  
    val TAG = "MainActivity"  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        printName()  
    }  
    companion object {  
        var status : String? = null  
        val TAG = "Companion.MainActivity"  
        fun printName() {  
            ...  
        }  
    }  
}
```



```
public class LoginActivity extends Activity {  
    ...  
    Intent intent = new Intent(LoginActivity.this,  
        MainActivity.class);  
    startActivity(intent);  
  
    MainActivity.Companion.printName();  
  
    ...  
}
```



Kotlin特性：空安全

```
class MainActivity : BaseActivity() {  
    ...  
    override fun onCreate(savedInstanceState: Bundle?) {  
        ...  
  
        Log.d(TAG, "length:" + status.length)  
  
        Log.d(TAG, "length:" + status?.length)  
    }  
  
    companion object {  
        var status : String? = null  
        val TAG : String = "TAG"  
        ...  
    }  
}
```

nullable
not nullable



```
public class MainActivity extends Activity {  
    String status = null  
    protected void onCreate(Bundle bundle) {  
        ...  
  
        Log.d(TAG, "length:" + status.length());  
  
        if (status != null){  
            Log.d(TAG, "length:" + status.length());  
        }  
    }  
    ...  
}
```



Kotlin特性：智能类型推导

```
class MainActivity : BaseActivity() {  
    ...  
    var count = 0  
    var status : String? = ""  
    ...  
    fun handleMessage(msg: Message) {  
        msg.obj.let {  
            when(it) {  
                is String -> { status = it }  
                is Int -> { count += it }  
            }  
        }  
    }  
    ...  
}
```



```
public class MainActivity extends Activity {  
    ...  
    String status;  
    int count;  
    ...  
    void handleMessage(Message msg) {  
        if(msg.obj instanceof String) {  
            status = (String)msg.obj;  
        }  
        if(msg.obj instanceof Integer) {  
            count += (Integer)msg.obj;  
        }  
    }  
    ...  
}
```



Kotlin特性：默认参数

```
class DataInfo() {  
  
    fun refreshData(appId: Int,    isInstalled: Boolean = false) {  
        ...  
    }  
}
```

```
dataInfo.refreshData(appId)  
dataInfo.refreshData(appId, false)
```



```
public class DataInfo {  
    ...  
    void refreshData(int appId, boolean isInstalled){  
        ...  
    }  
    void refreshData(int appId){  
    }  
}
```



Kotlin特性 : **async** 协程

```
class ScanActivity : Activity() {  
    ...  
    private fun onPageCreate() {  
        async {  
            val result = sendRequest(url)  
        }  
    }  
    suspend fun sendRequest(url : String) {  
        val dataInfo = async {  
            val res = httpGet(url)  
            return@async res  
        }  
        refreshData(dataInfo.await())  
    }  
    ...  
}
```



```
public class ScanActivity extends Activity {  
    ...  
    private void onPageCreate() {  
        new Thread(new Runnable() {  
            @Override  
            public void run() {  
                int result = sendRequest();  
                handler.sendMessage(result)  
                ...  
            }  
        }).start();  
    }  
    ...  
}
```



Kotlin特性 : Anko布局

```
class ScanActivity : Activity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        linearLayout {  
            titleView { setTitleTxt("安装包清理") }  
            listView { adapter = resultAdapter }  
            footerView {  
                onClick { showConfirmDialog() }  
            }  
        }  
    }  
}
```

```
inline fun ViewManager.titleView(theme: Int = 0, init: TitleView.() -> Unit) = ankoView({ TitleView(it) }, theme, init)  
inline fun ViewManager.footerView(theme: Int = 0, init: FooterView.() -> Unit) = ankoView({ FooterView(it) }, theme, init)
```



设计模式：object 单例

```
object ProtocolManager {  
    ...  
  
    fun sendRequest(task: () -> Unit) {  
        ...  
    }  
}
```

Call in Kotlin:

```
ProtocolManager.sendRequest { ... }
```

Call in Java:

```
ProtocolManager.INSTANCE.sendRequest( ... )
```



```
public class ProtocolManager {  
    private volatile static ProtocolManager instance;  
  
    private ProtocolManager() { ... }  
  
    public static ProtocolManager getInstance() {  
        if (instance == null) {  
            synchronized (ProtocolManager.class) {  
                if (instance == null) {  
                    instance = new ProtocolManager();  
                }  
            }  
        }  
        return instance;  
    }  
    ...  
}
```



设计模式：by lazy 懒加载

```
class ProtocolManager private constructor() {  
    ...  
  
    companion object {  
        val instance: ProtocolManager by lazy {  
            ProtocolManager()  
        }  
    }  
}
```

Call in Kotlin:

```
ProtocolManager.instance.sendRequest { ... }
```

Call in Java:

```
ProtocolManager.Companion.getInstance().sendRequest(...)
```



```
public class ProtocolManager {  
    private volatile static ProtocolManager instance;  
  
    private ProtocolManager() { ... }  
  
    public static ProtocolManager getInstance() {  
        if (instance == null) {  
            synchronized (ProtocolManager.class) {  
                if (instance == null) {  
                    instance = new ProtocolManager();  
                }  
            }  
        }  
        return instance;  
    }  
    ...  
}
```



设计模式：by 代理模式

```
class ScanActivity : Activity() {  
    private var lastScanTime: Long by  
        LongPreference(this, LAST_TIME, 0)  
  
    ...  
    private fun initView() {  
        if (lastScanTime > DURATION) {  
            ...  
        }  
        ...  
    }  
  
    ...  
    private fun onScanFinish() {  
        lastScanTime = System.currentTimeMillis()  
        ...  
    }  
}
```

```
class LongPreference(private val context: Context,  
    private val key: String,  
    private val default: Long = 0) :  
    ReadWriteProperty<Any?, Long> {  
  
    override fun getValue(...): Long  
        = prefs.getLong(key, default)  
  
    override fun setValue(...)  
        = prefs.edit().putLong(key, value).apply()  
}
```



设计模式：by 观察者模式

```
class CommentAdapter : RecyclerView.Adapter<CommonViewHolder>() {  
  
    var commentList: List<Comment> by Delegates.observable(emptyList()) { prop, old, new ->  
        autoNotify(old, new) { o, n -> o.id == n.id }  
    }  
    ...  
}  
  
fun <T> RecyclerView.Adapter<*>.autoNotify(oldList: List<T>, newList: List<T>, compare: (T, T) -> Boolean) {  
    DiffUtil.calculateDiff(...).dispatchUpdatesTo(this)  
}
```



Kotlin常见问题：空参数

```
class FriendResultListAdapter(var context: Context) : BaseAdapter() {  
  
    ...  
  
    override fun getView(position : Int, convertView : View , viewGroup : ViewGroup ) : View {  
        ...  
    }  
    ...  
}
```



Kotlin常见问题：空参数

```
class FriendResultListAdapter(var context: Context) : BaseAdapter() {
```

```
...
```


```
override fun getView(position : Int, convertView : View , viewGroup : ViewGroup ) : View {
```

```
...
```

```
}
```

```
...
```

```
}
```



```
n]
java.lang.IllegalArgumentException: Parameter specified as non-null is null:
.plugin.lab.adapter.FriendResultListAdapter.getView(Unknown Source:2)
tView.obtainView(AbsListView.java:2365)
ew.makeAndAddView(ListView.java:2052)
6136 ... (list item) ... 4100)
```



Kotlin

Kotlin常见问题：空参数

```
class FriendResultListAdapter(var context: Context) : BaseAdapter() {  
  
    ...  
  
    override fun getView(position : Int, convertView : View?, viewGroup : ViewGroup?) : View {  
        ...  
    }  
    ...  
}
```



Kotlin常见问题：反射调用Kotlin

```
class MainActivity : BaseActivity() {  
    ...  
    companion object {  
        var status : String? = null  
        val TAG = "Companion.MainActivity"  
        fun printName(): String {  
            ...  
        }  
    }  
}
```



```
public class RflectionUtils {  
    ...  
  
    String className = "MainActivity.Companion";  
    String methodName = "getStatus";  
  
    ...  
}
```



Kotlin常见问题：反射调用Kotlin

```
class MainActivity : BaseActivity() {
```

```
...
```

```
    companion object {
```

```
        var status : String? = null
```

```
        val TAG = "Companion.MainActivity"
```

```
        fun printName(): String {
```

```
            ...
```

```
        }
```

```
public static final class Companion {
```

```
    @Nullable
```

```
    public final String getStatus() { return MainActivity.status; }
```

```
    public final void setStatus(@Nullable String var1) { MainActivity.status = var1; }
```

```
    public final void printName() {
```

```
    }
```

```
    private Companion() {
```

```
    }
```

```
public class RflectionUtils {
```

```
...
```

```
String className = "MainActivity.Companion";
```

```
String methodName = "getStatus";
```

```
...
```

```
}
```



Kotlin常见问题：命名冲突

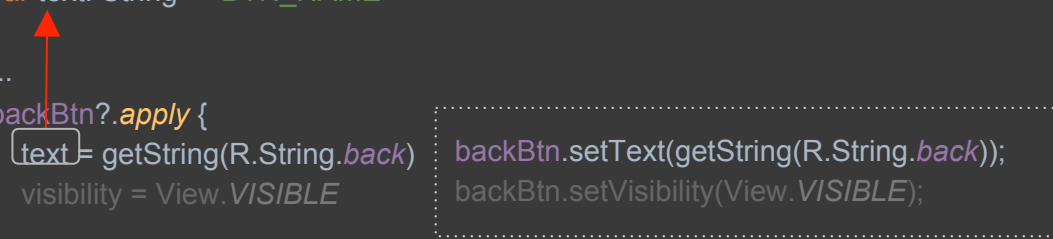
```
class ScanActivity : Activity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        ...  
        var text: String = "BTN_NAME"  
  
        ...  
        backBtn?.apply {  
            text = getString(R.String.back)  
            visibility = View.VISIBLE  
        }  
        ...  
    }  
}
```

```
backBtn.setText(getString(R.String.back));  
backBtn.setVisibility(View.VISIBLE);
```



Kotlin常见问题：命名冲突

```
class ScanActivity : Activity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        ...  
        var text: String = "BTN_NAME"  
        ...  
        backBtn?.apply {  
            text = getString(R.String.back)  
            visibility = View.VISIBLE  
        }  
        ...  
    }  
}
```

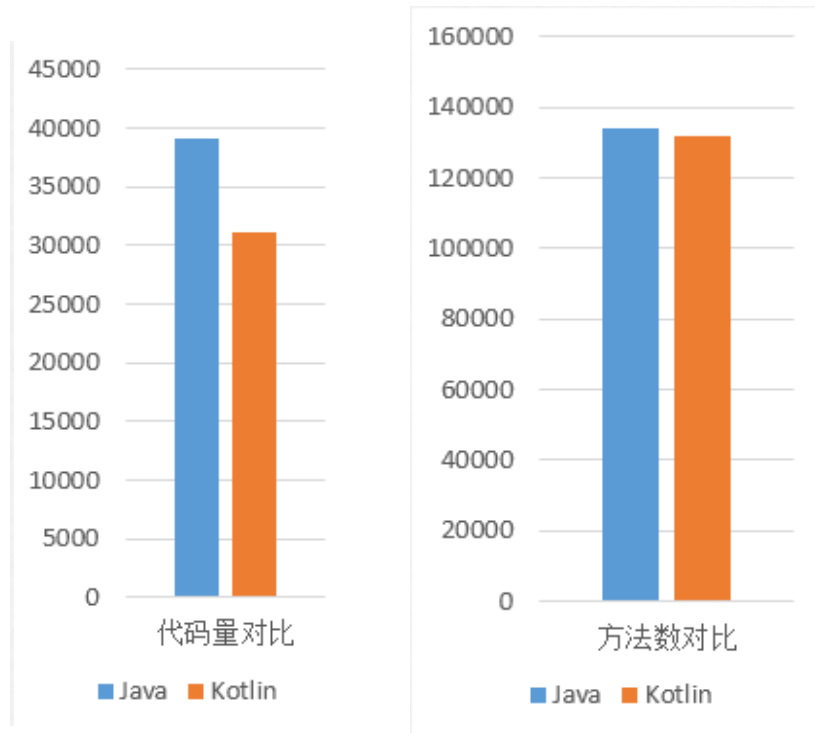


The diagram illustrates a naming conflict resolution in Kotlin. A red arrow points from the `text` property access inside the `backBtn?.apply` block to the `var text` declaration above it. A dashed box highlights the `backBtn` variable and its associated code block.

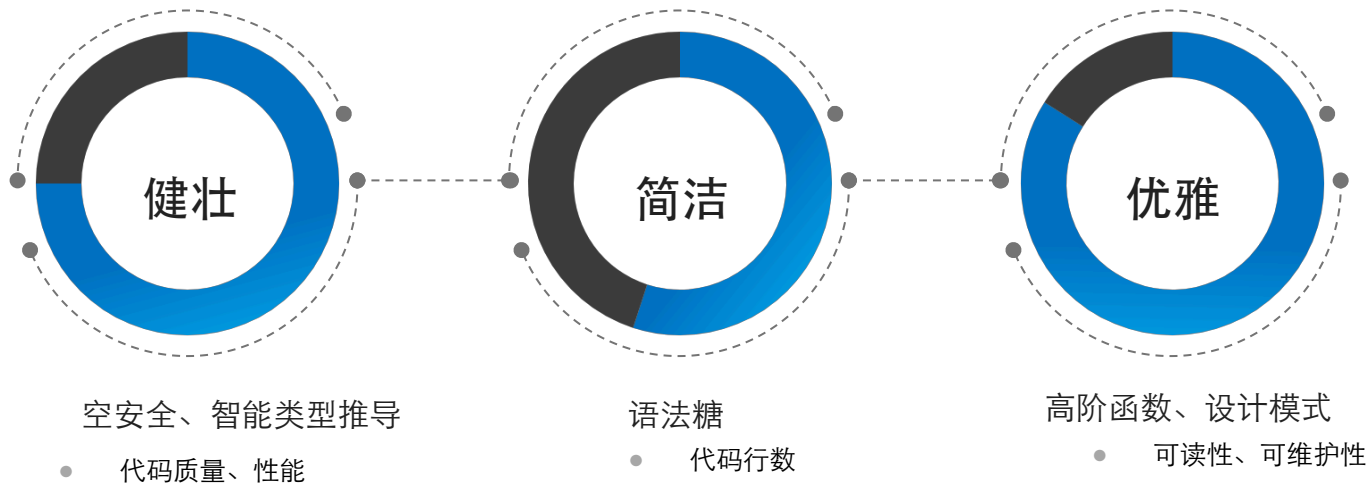
- 一、项目实战
- 二、优化效果
- 三、总结

项目数据

- 代码量减少25%
- 总方法数基本持平
- 包大小基本持平
- 模块编译时间基本持平
- 页面布局速度提升10%



项目效果



后续规划

- 后续新代码使用Kotlin
- 公共组件开源
- Kotlin社区建设

- 一、项目实战
- 二、优化效果
- 三、总结

Kotlin周边生态

- 工具支持

IntelliJ IDEA, Android Studio, Ant, Gradle, Maven...

- 社区

- 3W+开源项目

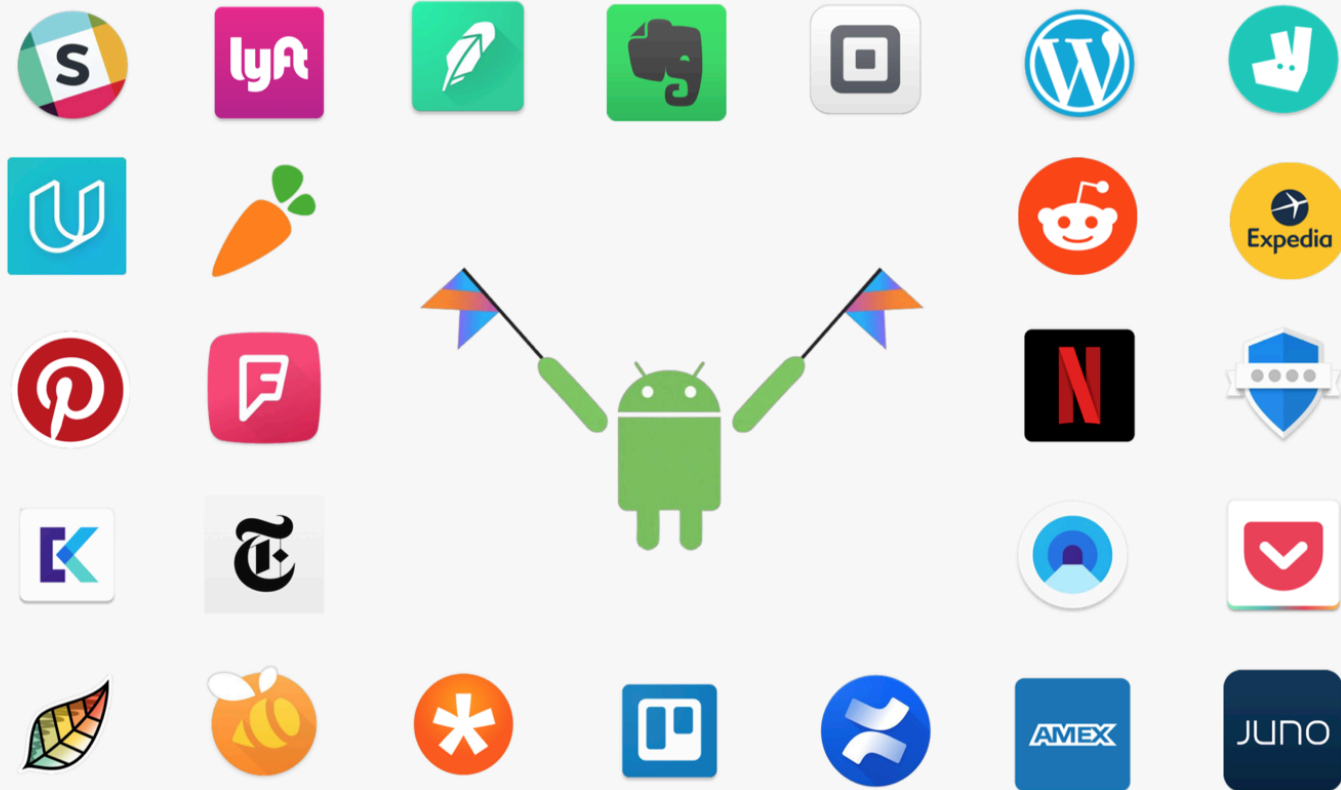
- 自动化测试支持

Spek, Mockito

- 书籍

<https://kotlinlang.org/docs/books.html>

业界项目



腾讯项目

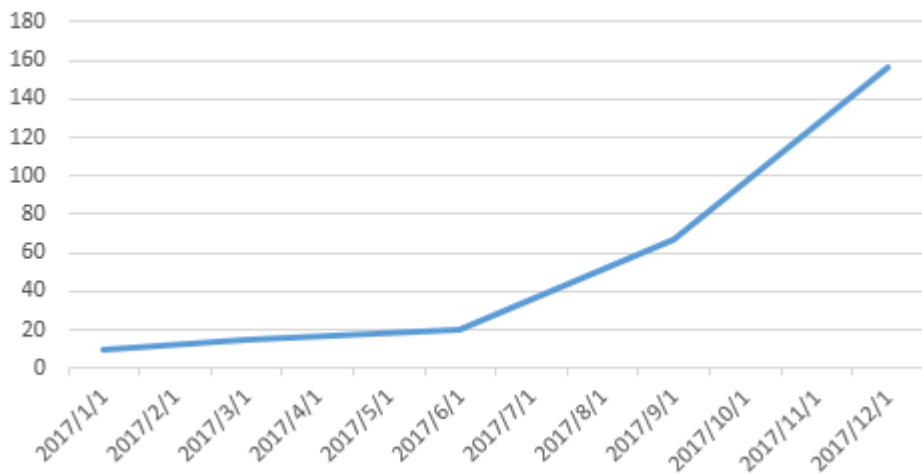
- 实际上线产品



- 多产品线正在研究改造

- 内部定期技术讨论

腾讯内部Kotlin相关文章数目



Kotlin中文站 欢迎加入！

中文站

<https://kotliner.cn/>

中文博客

<https://blog.kotliner.cn/>

微信公众号

[Kotlin](#)

GitHub

<https://github.com/enzowyf/Okhttp4k>

<https://github.com/enbandari/>

[QCloudImageUploaderForMarkDown](#)



QQ群：162452394



微信公众号：Kotlin

应用宝技术历程

用户体验

- 性能优化（多进程、H5离线缓存）

新特性快速触达用户

- 模块化（插件化框架、模块化dex开发）
- 动态化（热补丁、光子、动态UI框架）

研发效率、维护成本

- 代码测试自动化（质量扫描、安全扫描）
- 研发流程优化（灰度发布系统、pipeline）
- 线上数据实时监控系統（染色日志、异常监控）

Q&A