透过语法树来实现安卓预编译

李健民

个推 高级研发工程师

大数据服务实体产业

独角兽

领跑推送市场

个推
GeTui

最具公益力企业

D轮融资

独立的智能大数据服务商

个推
GeTui

数据驱动未来

开发者服务

个推·消息推送
个像·用户画像
个数·应用统计

用户增长服务

个启·用户促活
云发·用户拉新

精准营销服务

个灯·DMP数据服务
个信·精准投放平台

数据服务

个旅·旅游大数据
个真·金融大数据
公共服务

个推 GeTui

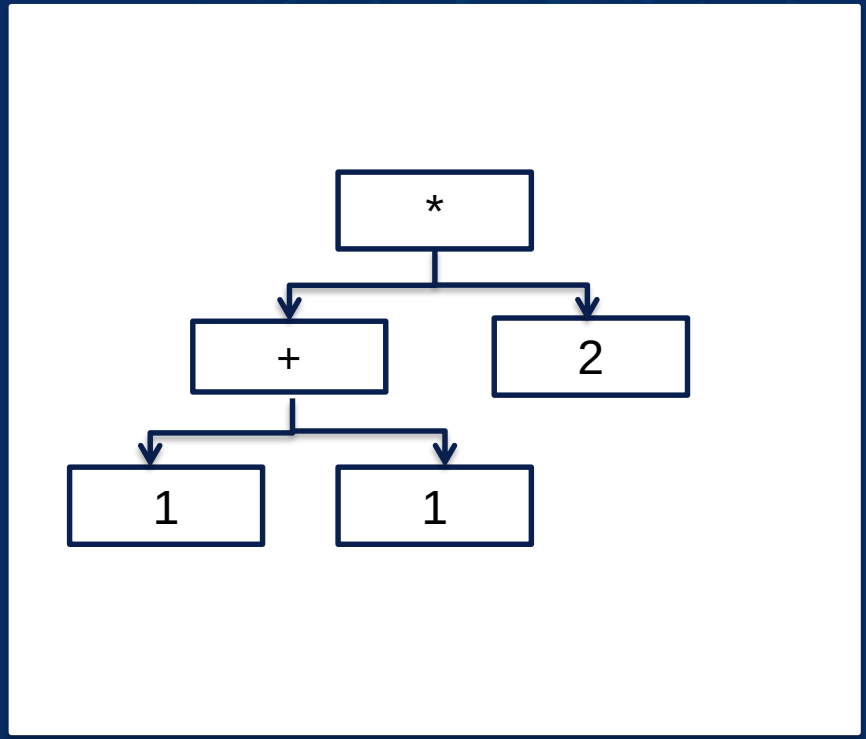
目录

语法树

预编译

延伸

个推 GeTui



个推
GeTui

编译器工作流程

词法分析

语法分析

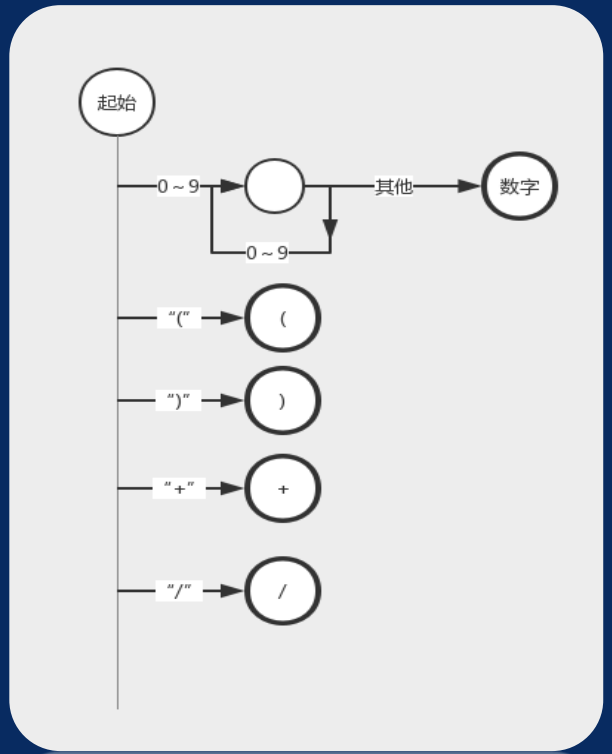
中间代码

目标代码

个推 GeTui

词法分析

(10+5)/3+1




```
[
    { type: 'paren',  value: '('    },
    { type: 'number',   value: '10'   },
    { type: 'operator', value: '+'    },
    { type: 'number',  value: '5'    },
    { type: 'paren',   value: ')'    },
    { type: 'operator', value: '/'    },
    { type: 'number', value: '3'    },
    { type: 'operator', value: '+'    },
    { type: 'number', value: '1'    }
]
```

个推 GeTui

语法分析

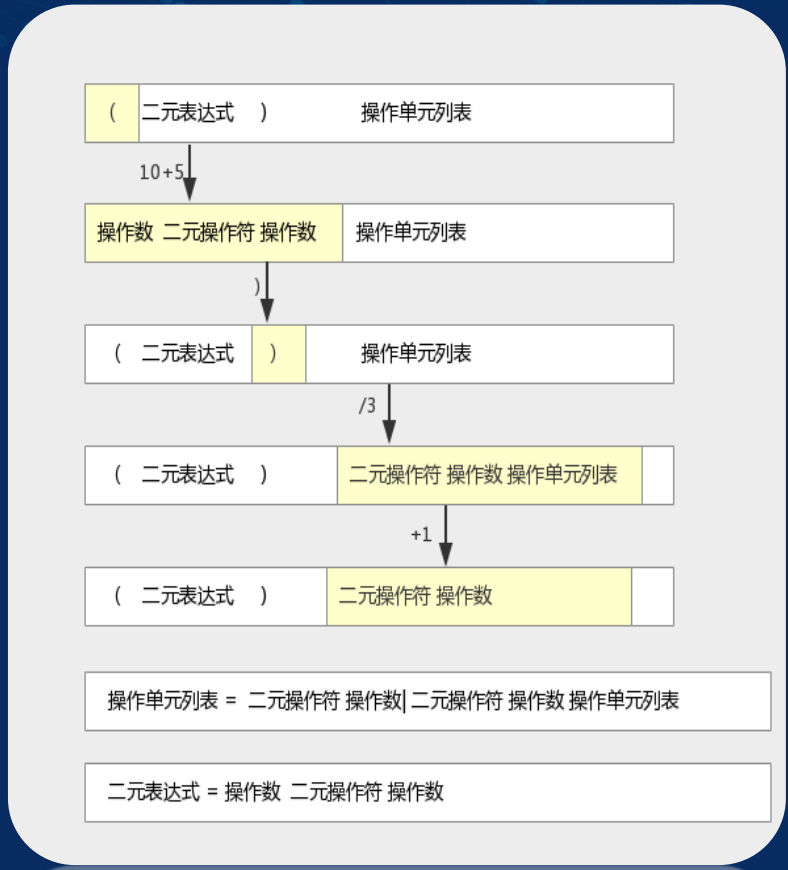
目的：读取token流进行语法匹配

定义(10+5)/3+1模板如下：

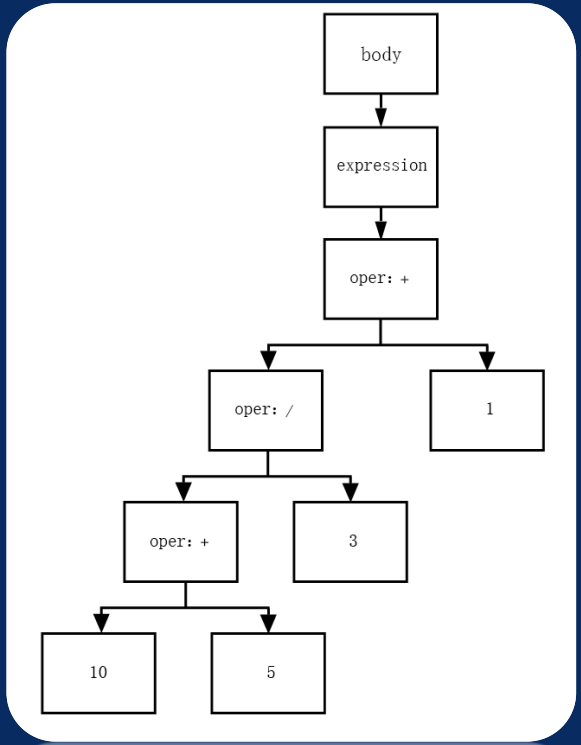
(二元表达式) 操作单元列表

操作单元列表 = 二元操作符 操作数| 二元操作符 操作数 操作单元列表

二元表达式 = 操作数 二元操作符 操作数



语法树：(10+5)/3+1



预编译=编译?

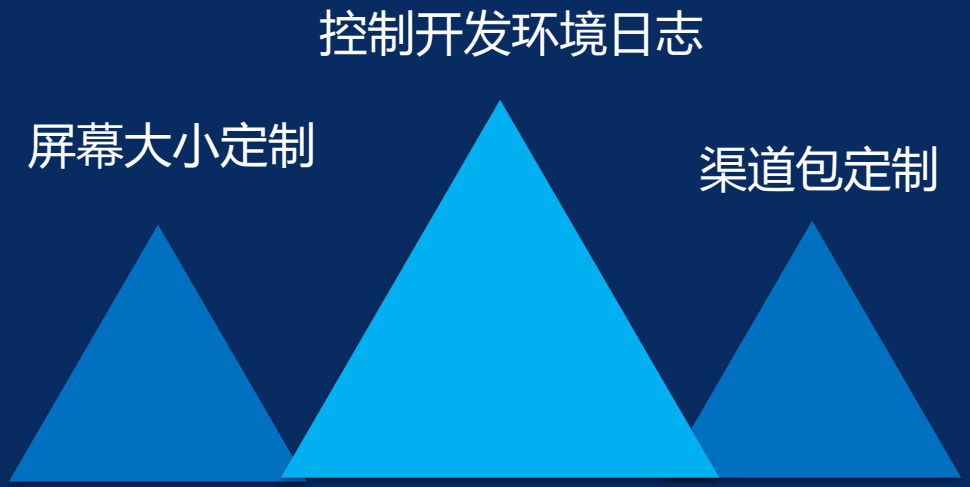
个推
GeTui

预编译

- #include
- #ifdef
- #ifndef
- #if
- #else
- #endif
- #define

个推 GeTui

应用场景



个推 GeTui

实现工具

Antenna

个推 GeTui

Antenna

if

#if [expression]
#elif[expression]
#else
#endif

ifdef/ifndef

#ifdef [identifier]
#elifdef[identifier]
#endif

expand

#expand public int ver =
%ver%;
public int VER = 5;

debug/mdebug

#mdebug
#enddebug
#debug
[expression]

个推 GeTui

```
#if !('nokia-ui'@JavaPackage)
//# setFullScreenMode(true);
//#endif
```

```
#if !('nokia-ui'@JavaPackage)
setFullScreenMode(true);
//#endif
```

```
//#ifdef ver
//#expand int ver = %ver%;
int ver;
//#endif
```

```
//#ifdef ver
//#expand int ver = %ver%;
int ver = 6;
//#endif
```

```
//#debug
Log.d(“TAG”,”print some”);
```

```
//#debug
//Log.d(“TAG”,”print some”);
```

个推 GeTui

Antenna缺陷

环境变量

文件配置繁琐

不支持gradle集成

个推 GeTui

对比

个推预编译插件 Antenna

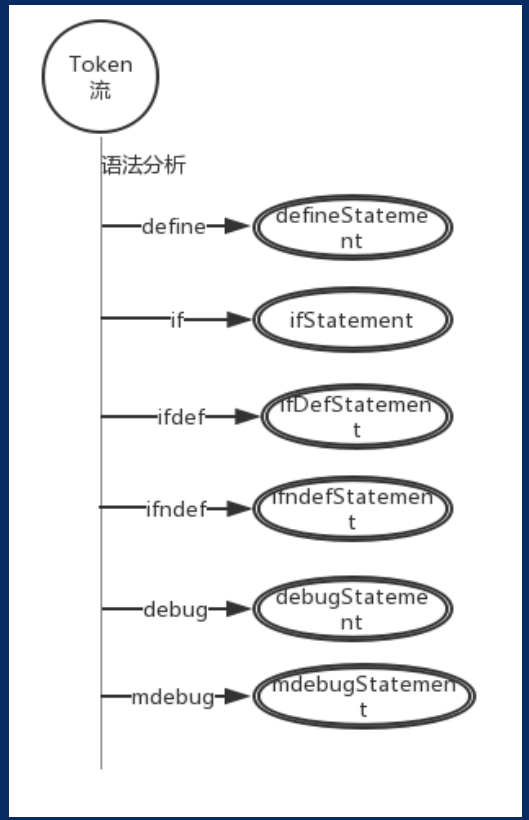
	个推预编译插件	Antenna
string	Y	Y
boolean	Y	Y
int	Y	Y
float	Y	N
double	Y	N
换行/空格	Y	N
startwith/endwith	Y	N
gradle集成	Y	N

个推 GeTui

个推预编译插件实现过程

词法分析—>token流

语法分析—>执行相应指令



个推 GeTui

延伸

个推 GeTui

怎么实现

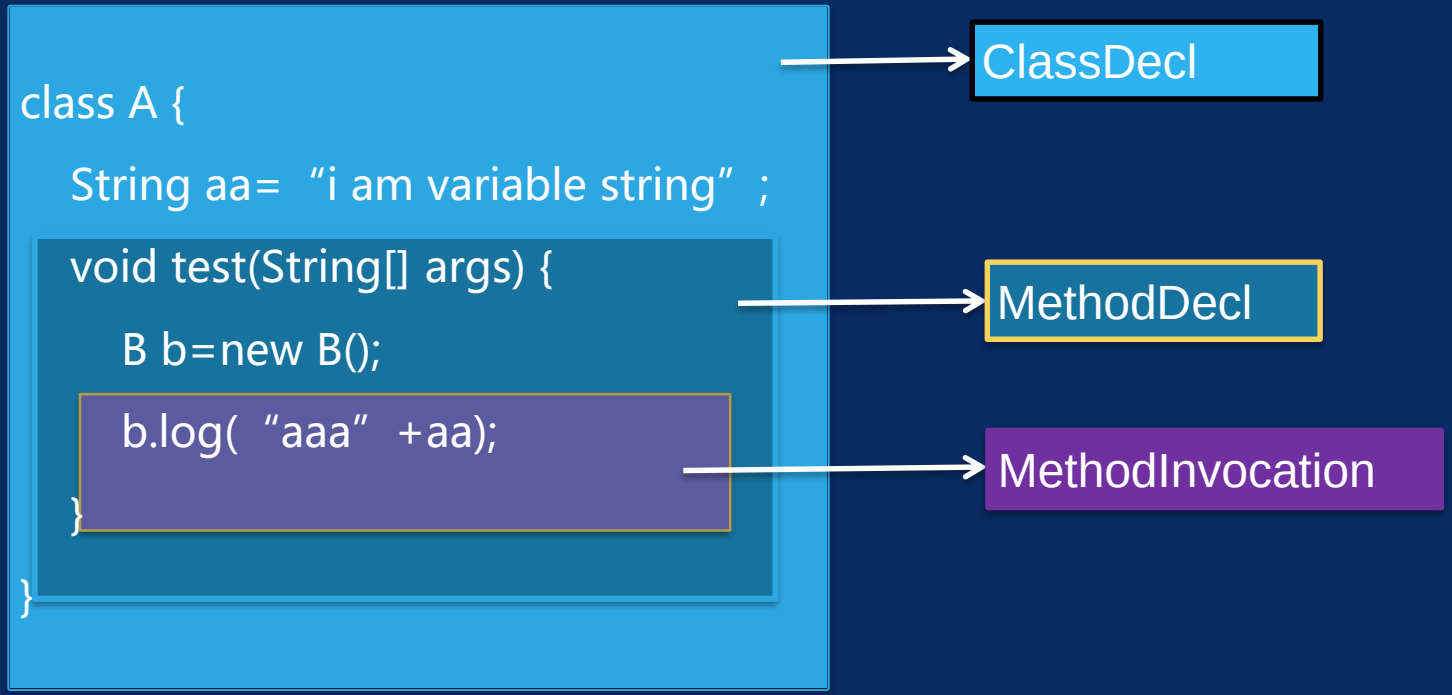
```
void method(){
    LogUtil.d(TAG,”hello world!”);
}

To

void method(){
    LogUtil.d(TAG,”<-(1)->”);
}
```

个推 GeTui

个推日志混淆插件

将源码解析为AST语法树，然后遍历每一个编译单元与单元的类声明，根据日志方法的签名找到所有的方法调用，然后遍历每个方法调用，将方法调用的对应的参数表达式放入递归方法，对字符串字面值进行改写。

个推 GeTui


class A {
String aa= “i am variable string”;
void test(String[] args) {
B b=new B();
b.log(“aaa” +aa);
}
}
ClassDecl
MethodDecl
MethodInvocation

个推 GeTui

最后一个问题

个推
GeTui

能做什么

代码检测

语法高亮

自动埋点

个推 GeTui

谢谢！

个推 GeTui