

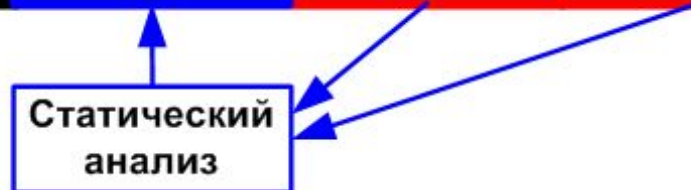
Об универсальном анализе кода

или

«Зачем нам ещё один анализатор,
как его можно сделать и куда применять»

Зачем SAST?

| | Время обнаружения дефекта | | | | |
|-------------------------------|---------------------------|----------------------------|-------------------------------|--------------|------------------|
| Время внесения дефекта | Выработка требований | Проектирование архитектуры | Конструирование (кодирование) | Тестирование | После выпуска ПО |
| Выработка требований | 1 | 3 | 5-10 | 10 | 10-100 |
| Проектирование архитектуры | - | 1 | 10 | 15 | 25-100 |
| Конструирование (кодирование) | - | - | 1 | 10 | 10-25 |



Ладно, зачем ещё один?

КАК МНОЖАТСЯ СТАНДАРТЫ:

(СМ.: ЗАРЯДНЫЕ УСТРОЙСТВА, КОДИРОВКИ, МГНОВЕННЫЕ СООБЩЕНИЯ И Т.Д.)



SQL-ИНЪЕКЦИИ В 2К17?!

| OWASP Top 10 - 2013 | → | OWASP Top 10 - 2017 |
|--|---|--|
| A1 – Injection | → | A1:2017-Injection |
| A2 – Broken Authentication and Session Management | → | A2:2017-Broken Authentication |
| A3 – Cross-Site Scripting (XSS) | ↘ | A3:2017-Sensitive Data Exposure |
| A4 – Insecure Direct Object References [Merged+A7] | U | A4:2017-XML External Entities (XXE) [NEW] |
| A5 – Security Misconfiguration | ↘ | A5:2017-Broken Access Control [Merged] |
| A6 – Sensitive Data Exposure | ↗ | A6:2017-Security Misconfiguration |
| A7 – Missing Function Level Access Contr [Merged+A4] | U | A7:2017-Cross-Site Scripting (XSS) |
| A8 – Cross-Site Request Forgery (CSRF) | ☒ | A8:2017-Insecure Deserialization [NEW, Community] |
| A9 – Using Components with Known Vulnerabilities | → | A9:2017-Using Components with Known Vulnerabilities |
| A10 – Unvalidated Redirects and Forwards | ☒ | A10:2017-Insufficient Logging&Monitoring [NEW,Comm.] |

А языки правда отличаются?

Java

```
String query = "SELECT * FROM user WHERE name = " + name;
```

Python

```
query = "SELECT * FROM user WHERE name = " + name
```

C++

```
std::string query = "SELECT * FROM user WHERE name = " + name;
```

PHP

```
$query = "SELECT * FROM user WHERE name = " . $name;
```

Спойлер: иногда не очень

Классика

DRY

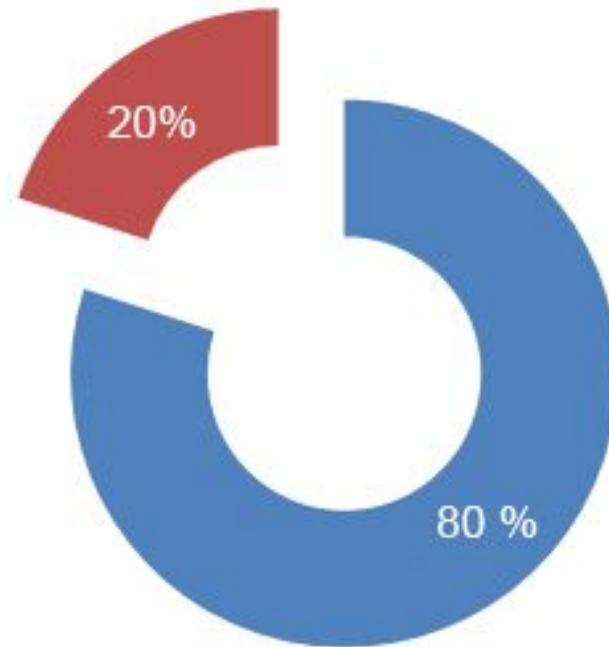
Don't Repeat Yourself

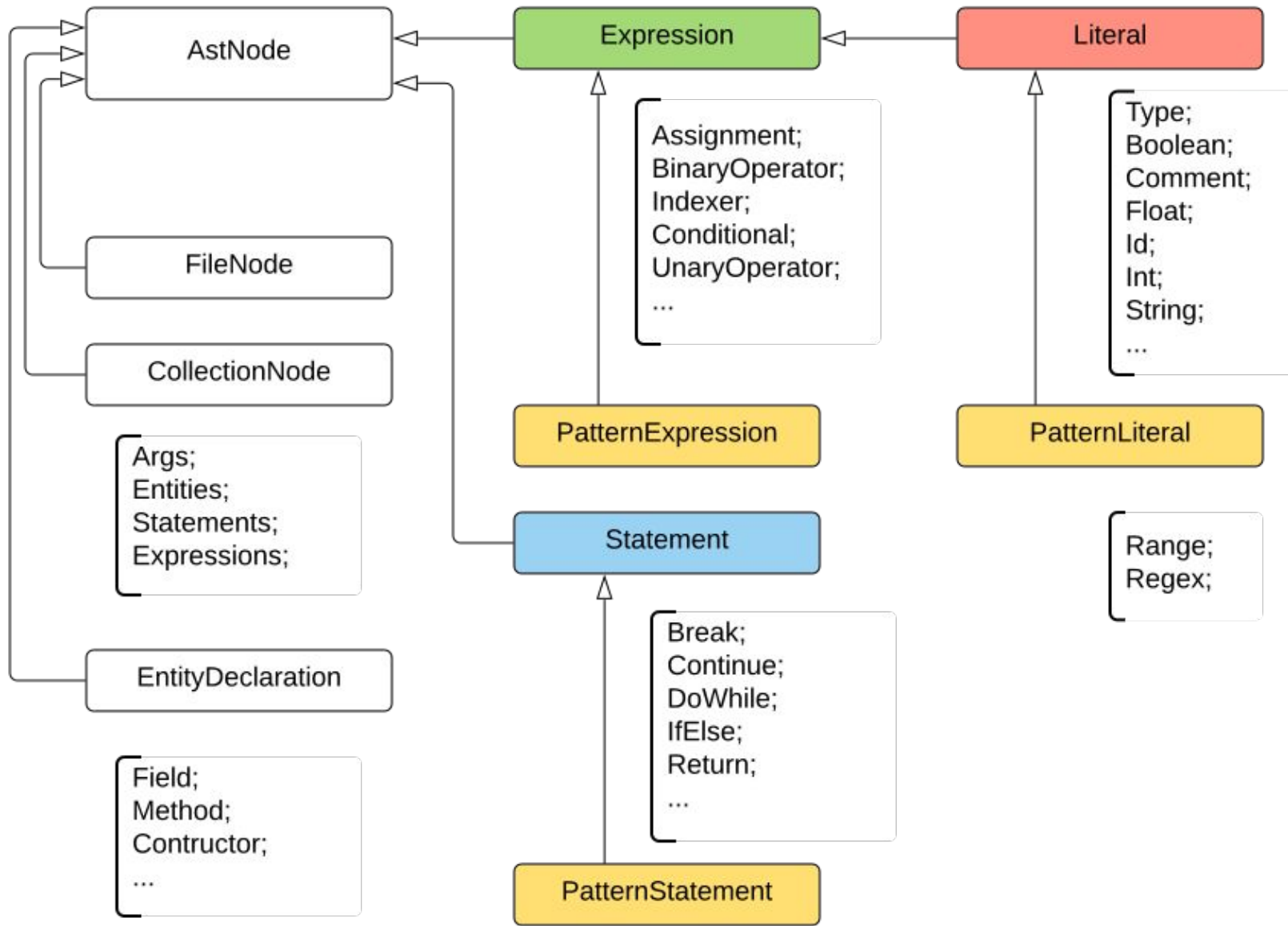
KISS

Keep It Simple Stupid

YAGNI

You Ain't Gonna Need It





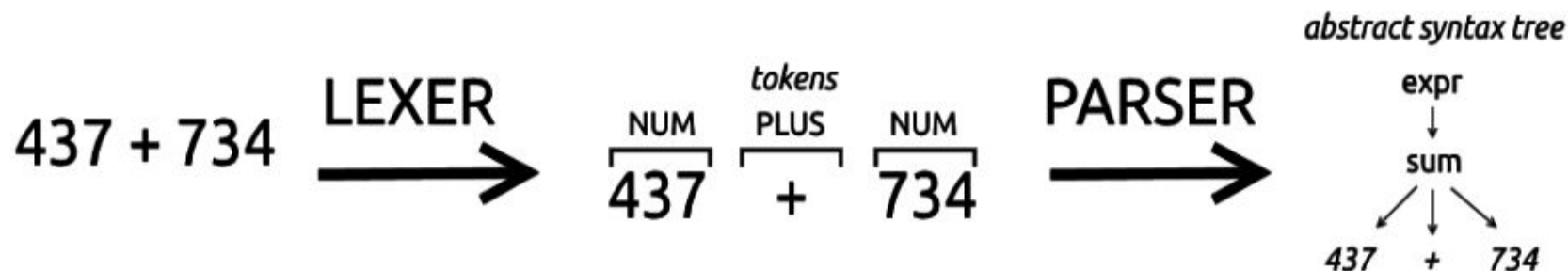
Сужаем область

Цель – расширение класса задач модуля PT.PM путем интеграции подсистемы анализа языка Python.

Почему Python:

- популярный (топы: Tiobe, PYPL, RedMonk)
- простой (КС-грамматика по Хомскому)
- могу, умею, практикую 😊

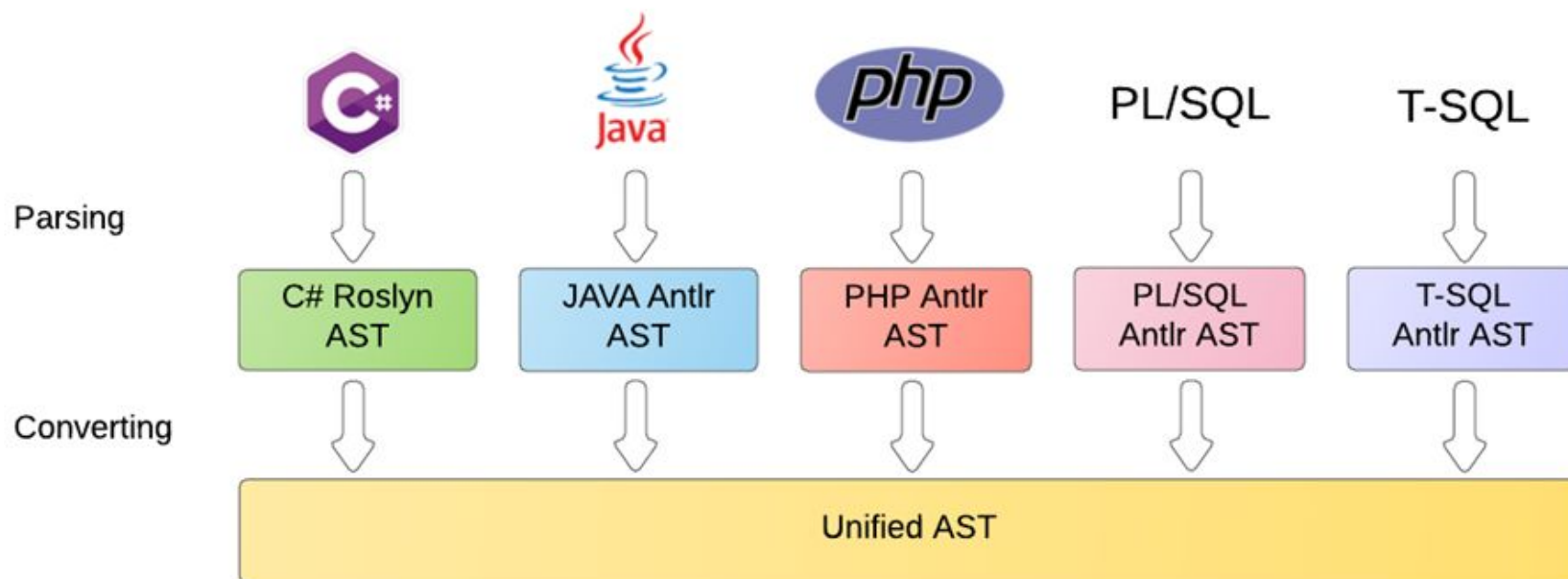
Что и как делаем? - 1



Как: ANTLR4 +
репозиторий `antlr4-grammars` +
документация Python +
официальная грамматика Python

Looks good!

Что и как делаем? - 2



Как: PT.PM API +
C#-парсер +
руки

Still looks good!

Но... - 1

```
async_funcdef: 'async' funcdef
funcdef: 'def' NAME parameters ['->' test] ':' suite

parameters: '(' [typedargslist] ')'
typedargslist: (tfpdef ['=' test] (',' tfpdef ['=' test]))* [',' [
    '*' [tfpdef] (',' tfpdef ['=' test]))* [',' ['**' tfpdef [',']]
    | '**' tfpdef [',']]
    | '*' [tfpdef] (',' tfpdef ['=' test]))* [',' ['**' tfpdef [',']]
    | '**' tfpdef [','])
tfpdef: NAME [':' test]
varargslist: (vfpdef ['=' test] (',' vfpdef ['=' test]))* [',' [
    '*' [vfpdef] (',' vfpdef ['=' test]))* [',' ['**' vfpdef [',']]
    | '**' vfpdef [',']]
    | '*' [vfpdef] (',' vfpdef ['=' test]))* [',' ['**' vfpdef [',']]
    | '**' vfpdef [','])
)
vfpdef: NAME
```

ИСТОЧНИК: <https://docs.python.org/3/reference/grammar.html>

Ho... - 2

decorated

: decorators (classdef | funcdef | async_funcdef)

<**LOTS OF** stuff here>

compound_stmt

: if_stmt | while_stmt | for_stmt | try_stmt
| with_stmt | funcdef | classdef
| decorated // added later
| async_stmt // added even later

Ho... - 3

```
assignment_stmt ::= (target_list "=")+ (starred_expression | yield_expression)
target_list     ::= target ("," target)* [","]
target         ::= identifier
                | "(" [target_list] ")"
                | "[" [target_list] "]"
                | attributeref
                | subscription
                | slicing
                | "*" target
```

VS.

```
expr_stmt: testlist_star_expr (annassign | augassign (yield_expr|testlist) |
    ('=' (yield_expr|testlist_star_expr))*
```

Но... - n

- INDENTS & DEDENTS
- Неэффективные алгоритмы
- Мало документации по ANTLR4 в свободном доступе
- Устаревшая документация на PT.PM
- Меняющийся API PT.PM
- И так далее ...

Так что в итоге?

- Полный рефакторинг грамматики
- Порт парсера в C#
- Порт AST в UST
- Полное покрытие грамматики тестами
- Несколько тестов на матчинг шаблонов

Что дальше?

- Увеличение базы шаблонов
- Рефакторинг кода
- Документация
- Покрытие шаблонов тестами
- **Фикс документации Python**

Спасибо!

Никита Субботин

Github: **inkoit**

Telegram: **nsubbotin**

Mail: **sub.nik.and@gmail.com**