Setup for Cool and Antlr

Greta Yorsh

Marking PA1-PA3

- Team sign up

- PA1: START EARLY!

- PA1 marking guidelines

  - 80% testsuite

  - 20% code review

  - testsuite hurdle: 90% by submission deadline

- Feedback and final submission deadline

- Independent components

Today

- Slides and Q-review of tutorials
- GitHub repositories for PA1-PA3
- Antlr4 workflow and example
- PA1 Setup
- Testsuites and testscripts
- Inspect outputs not just PASS/FAIL counts!!

Repositories

QMUL github server
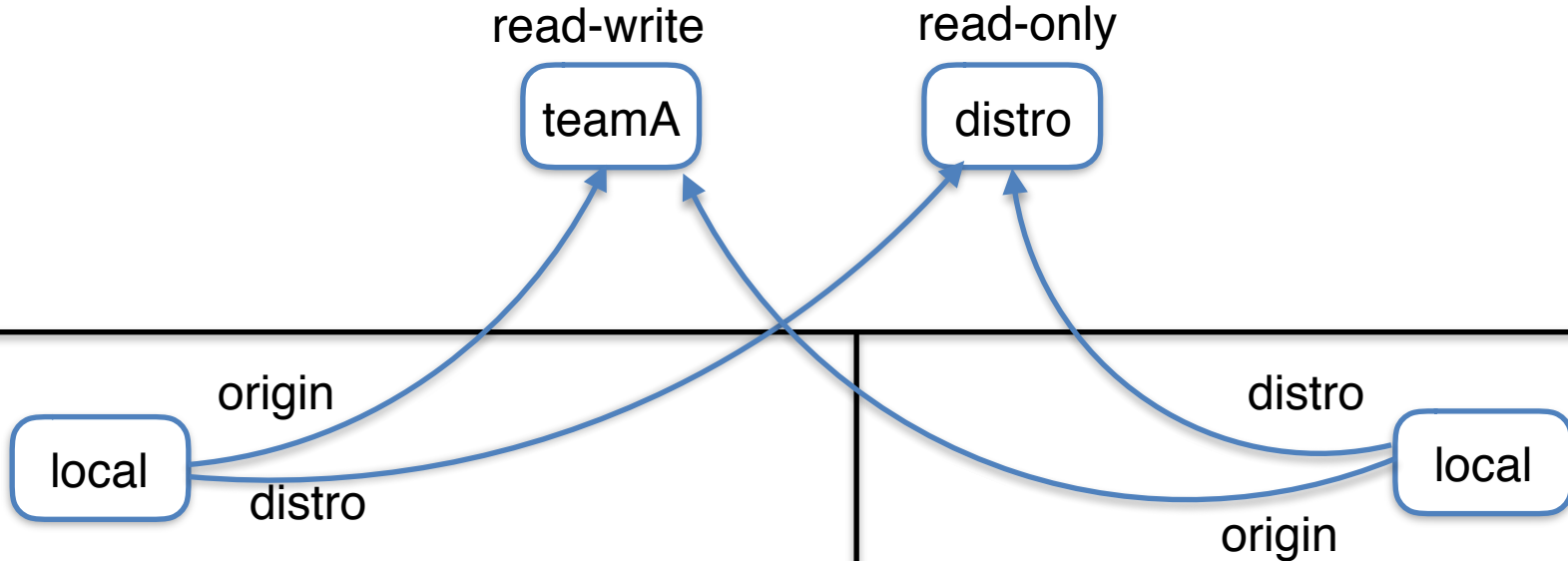
read-write

read-only

teamA

distro

origin

distro

local

distro

origin

local

student1 local: mac
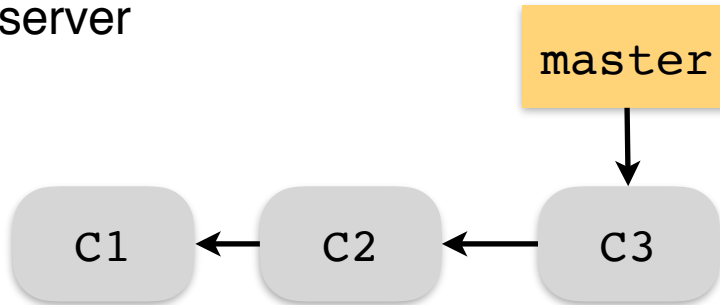
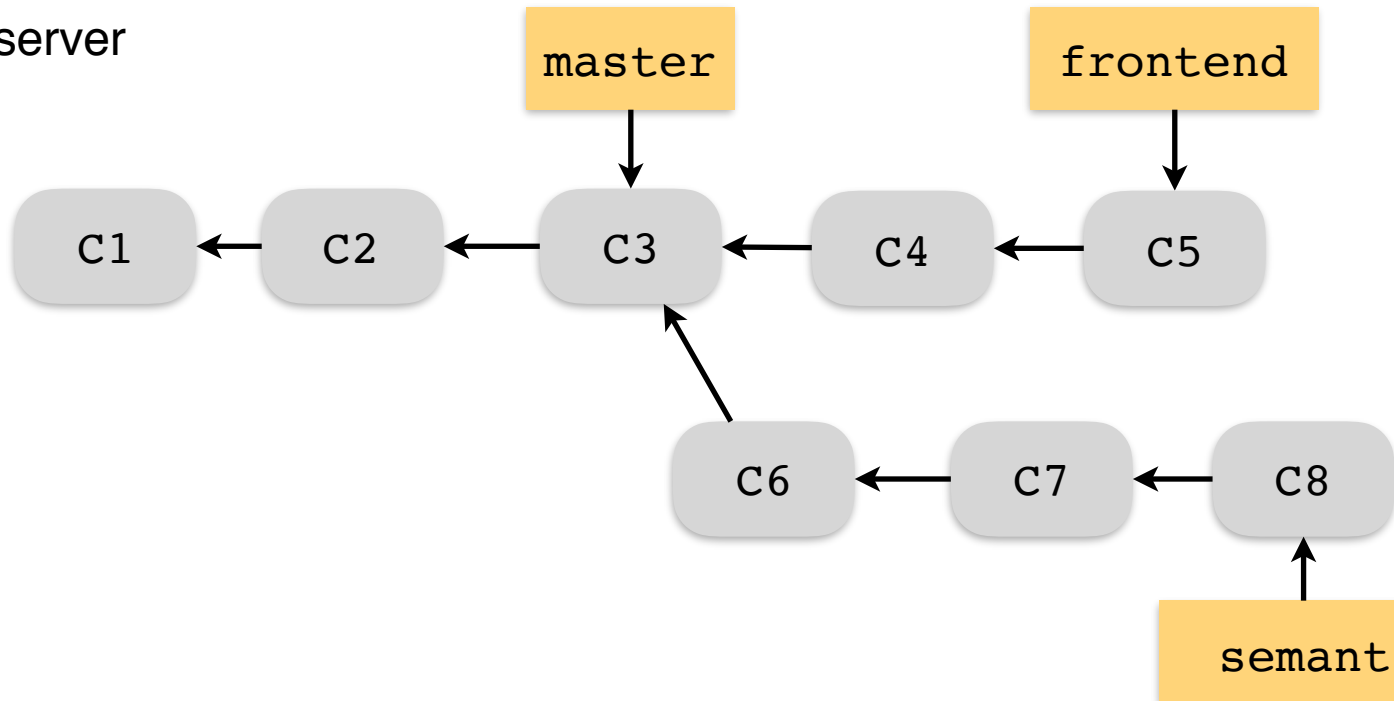student2 local: ITL machine

Branches

QMUL github server

read-only

distro

master

C1 ← C2 ← C3

QMUL github server

read-write

teamA

master   frontend

C1 ← C2 ← C3 ← C4 ← C5
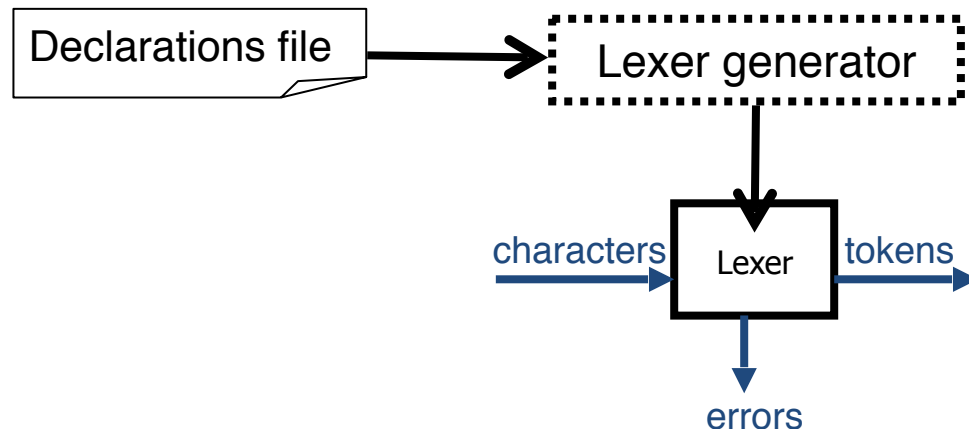
C6 ← C7 ← C8

semant

Integrated Development Environment (IDE)

- IntelliJ - free for students
- NetBeans
- Eclipse

- Antlr4 plugin for each IDE

Lexer Generation

- Terminology: lexical analysis = lexical analyzer = lexer = scanner
- Describe tokens as regular expressions
- Decide which attributes are saved for each token
- Construction is done automatically by common tools
- **Lexer generator** automatically creates **lexer** from **declarations file**
  - lexer generator builds DFA table
  - lexer simulates (runs) the DFA on a given input
- We will use **Antlr** lexer generator

Declarations file → Lexer generator

characters → Lexer → tokens

errors

Declarations file in Antlr4

```
lexer grammar ExampleLexer;

INT             : DIGIT+ ;
fragment
DIGIT           : [0-9] ;


ID              :  [a-z] IDENTIFIER* ;
fragment
LETTER          :  [a-zA-Z];

COMMENT     : '/*' .*? '*/'   -> skip;

WHITESPACE : (' ' | '\n' | '\r' | '\t' | '\u000B')+ -> skip;

ERROR           : . ;
```
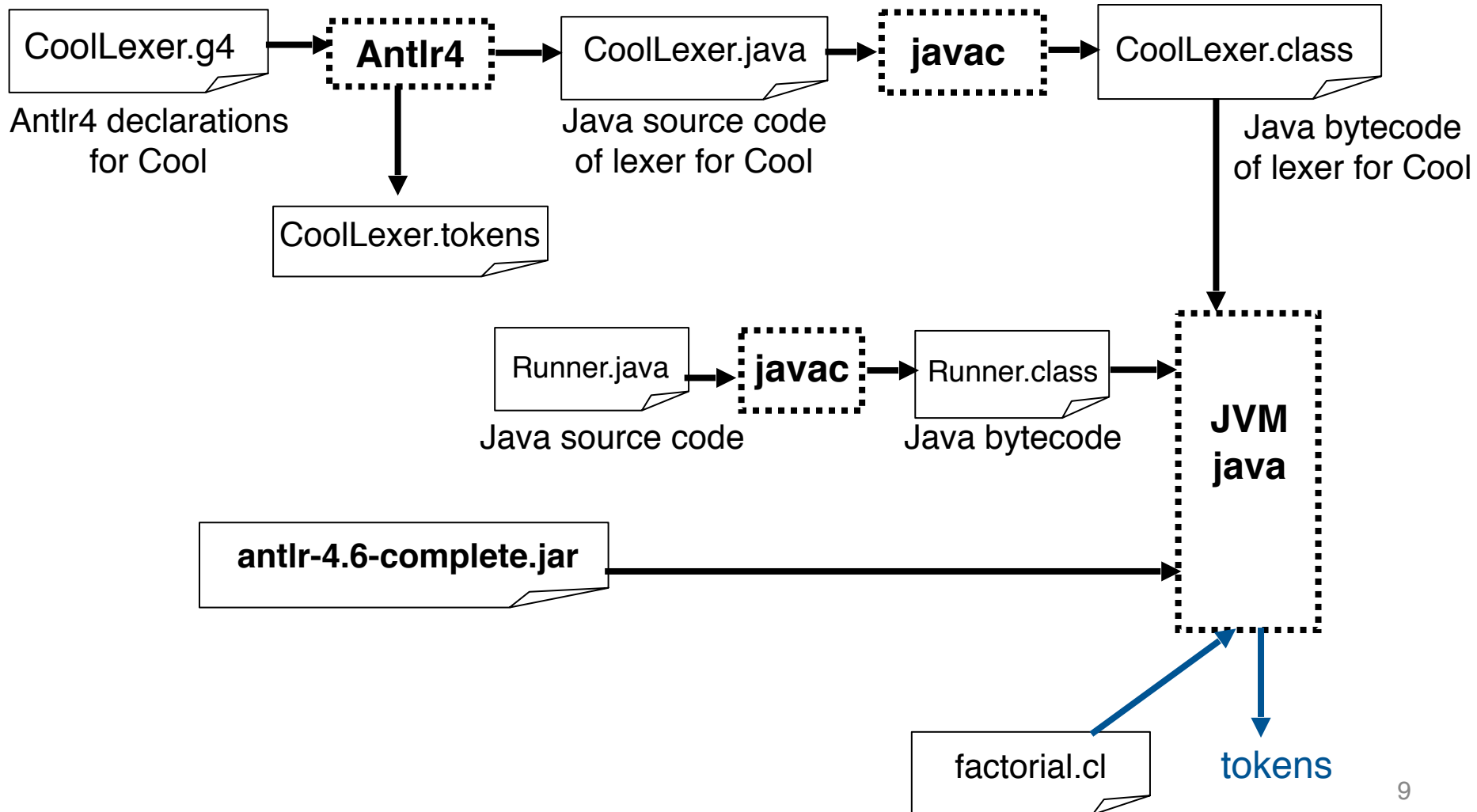
Antlr: build and run a lexer

CoolLexer.g4 → **Antlr4** → CoolLexer.java → **javac** → CoolLexer.class

Antlr4 declarations for Cool

Java source code of lexer for Cool

Java bytecode of lexer for Cool

CoolLexer.tokens

Runner.java → **javac** → Runner.class

Java source code

Java bytecode

JVM java

antlr-4.6-complete.jar

factorial.cl

tokens
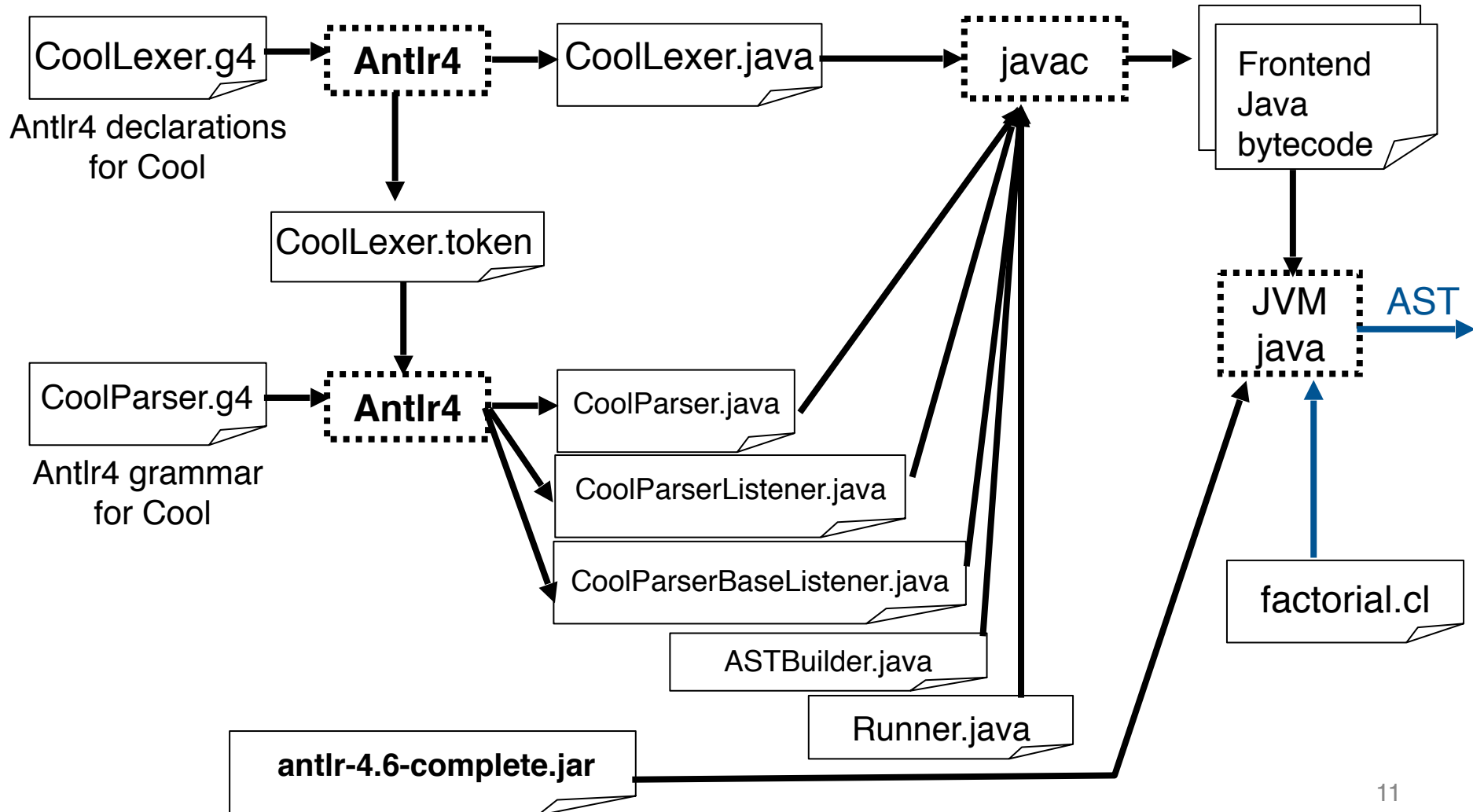
Runner

```java
import org.antlr.v4.runtime.*;

import org.antlr.v4.runtime.tree.*;

public class Runner {

  public static void main(String[] args) throws Exception {

    ANTLRInputStream input = new ANTLRInputStream(System.in);

    CoolLexer lexer = new CoolLexer(input);

    CommonTokenStream tokens = new CommonTokenStream(lexer);

    for (Token t : tokens.getTokens()) {
        System.out.println(t.toString());
    }

  }

}
```

Antlr: build and run a parser

CoolLexer.g4

Antlr4 declarations for Cool

CoolParser.g4

Antlr4 grammar for Cool

Antlr4

Antlr4

CoolLexer.java

CoolLexer.token

CoolParser.java

CoolParserListener.java

CoolParserBaseListener.java

ASTBuilder.java

Runner.java

antlr-4.6-complete.jar

javac

Frontend Java bytecode

JVM java

AST

factorial.cl

Runner

```java
import org.antlr.v4.runtime.*;
import org.antlr.v4.runtime.tree.*;
public class Runner {
  public static void main(String[] args) throws Exception {
    ANTLRInputStream input = new ANTLRInputStream(System.in);
    CoolLexer lexer = new CoolLexer(input);
    CommonTokenStream tokens = new CommonTokenStream(lexer);
    CoolParser parser = new CoolParser(tokens);
    ParseTree tree = parser.init();
    System.out.println(tree.toStringTree(parser));
  }
}
```

Grammar file in Antlr4

```
grammar ExampleParser;

program : method* EOF;

method : TYPE ID LPAREN formals RPAREN
body;

TYPE : [A-Z]LETTER*;
```