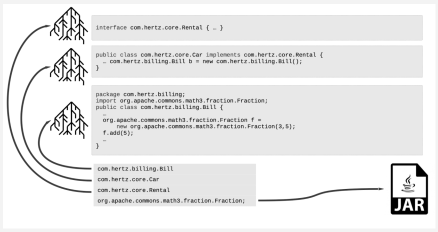
Advanced Software Engineering Notes Cheat Sheet

by User23456 via cheatography.com/151394/cs/32718/

Source code to fully qualified name resolution



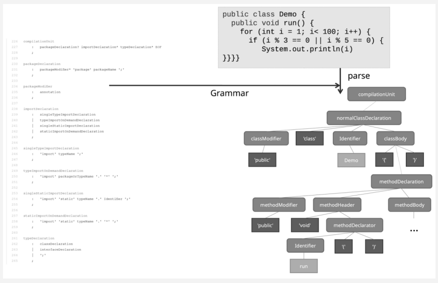
```
interface com.hertz.core.Rental { … }
```

```
public class com.hertz.core.Car implements com.hertz.core.Rental {
  … com.hertz.billing.Bill b = new com.hertz.billing.Bill();
}
```

```
package com.hertz.billing;
import org.apache.commons.math3.fraction.Fraction;
public class com.hertz.billing.Bill {
  …
  org.apache.commons.math3.fraction.Fraction f =
      new org.apache.commons.math3.fraction.Fraction(3,5);
  f.add(5);
  …
}
```

```
com.hertz.billing.Bill
com.hertz.core.Car
com.hertz.core.Rental
org.apache.commons.math3.fraction.Fraction;
```

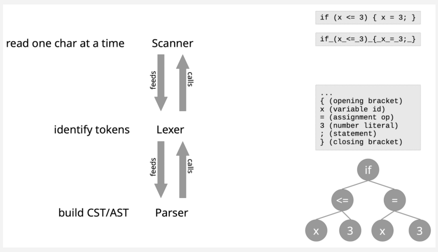
Concrete Syntax Tree (CST)



```
public class Demo {
  public void run() {
    for (int i = 1; i< 100; i++) {
      if (i % 3 == 0 || i % 5 == 0) {
        System.out.println(i)
}}}}
```

- 1:1 mapping of source code to a tree representation.
- Consists of simple tokens (i.e. strings, numbers).

From source code to lexing and parsing

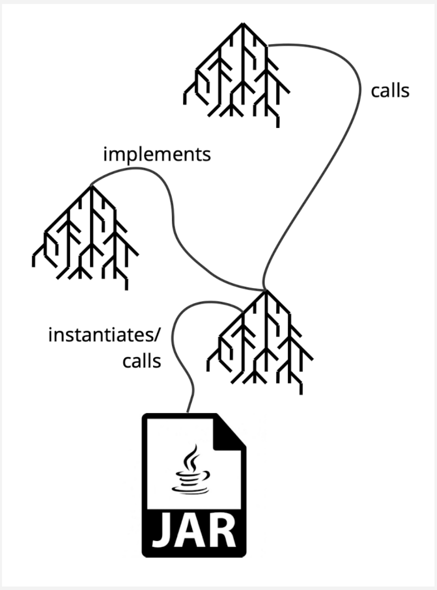


```
if (x <= 3) { x = 3; }
```

```
if_(x_<=_3)_{_x_=_3;_}
```

```
...
{ (opening bracket)
x (variable id)
= (assignment op)
3 (number literal)
; (statement)
} (closing bracket)
```

From source code to building an object model



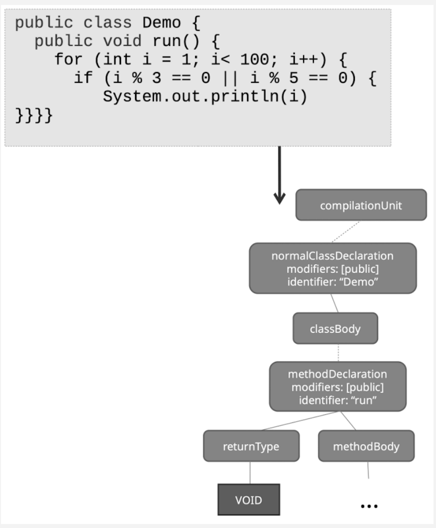
Source code to writing output from the model

Bytecode / assembly / machine code.
- Generate machine or VM executable code.
Source code of a different language.
- Code synthesis: generating source code from the object model.
- Decompilation: generating the original source code.
Code metrics:
- Analyse the object model to compute metrics, i.e. number of classes, methods, complexity etc. > next week.
Metadata, e.g. lookup tables, API documentation.
- IDE needs info on classes / methods for code completion.

Source code to writing output from the model (cont)

- HTML API docs are hyperlinked.

Abstract Syntax Tree (AST)



```
public class Demo {
  public void run() {
    for (int i = 1; i< 100; i++) {
      if (i % 3 == 0 || i % 5 == 0) {
        System.out.println(i)
}}}}
```

- Only necessary parts.
- Possible consisting of complex node objects.
- Possibly restructured for simplicity.

Source code to fully qualified name resolution

Internal representations:
E.g. JVM method descriptors:
- Method signature in Java source:
`Object[] m(int i, double d, Thread t) { ... }`
- Method descriptor in JVM class file:
`(IDLjava/lang/Thread;)[Ljava/lang/Object`