CSNIPPEX: Automated Synthesis of Compilable Code Snippets from Q&A Sites

Valerio Terragni Yepang Liu Shing-Chi Cheung

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
{vterragni, andrewust, scc}@cse.ust.hk

ISSTA 2016

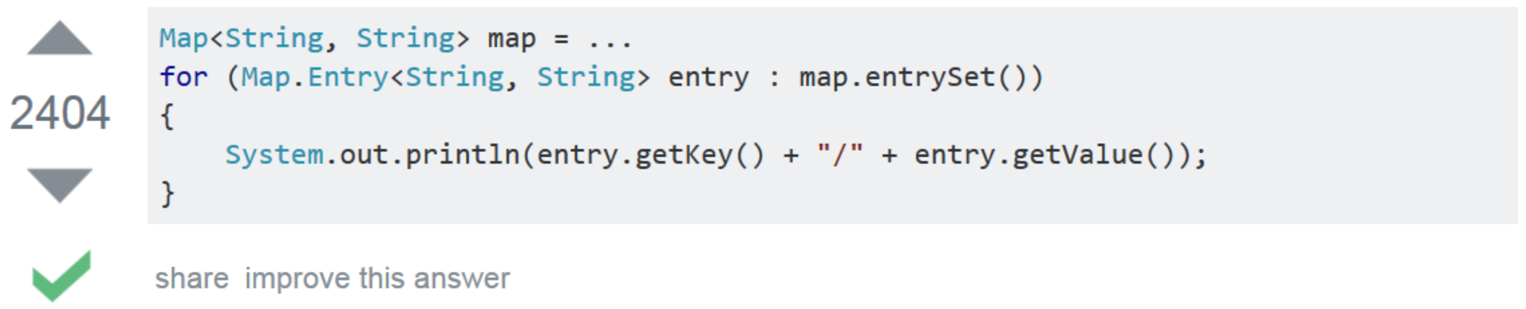
18 July 2016

Social Network Revolution
Q&A sites for developers

stackoverflow

12 Million Questions [1]
19 Million Answers

Millions of high quality code snippets!


2404

```
Map<String, String> map = ...
for (Map.Entry<String, String> entry : map.entrySet())
{
    System.out.println(entry.getKey() + "/" + entry.getValue());
}
```

share improve this answer

- Solutions of programming tasks
- Bug fixes
- API usage examples

[1] http://stackexchange.com/sites?view=list#traffic June 2016

Usefulness of Q&A's Code Snippets

code reuse and analysis

- Developers often search code snippets in Q&A sites [Mao@RN2010, Stoole@TOSEM2014]
- stackoverflow receives **500 Million** views per month[1]

DEVEL

The Best

All The Ti

LOGAN BOO

FACEBOOK

Stack Overf

(SODD) — I

The newest "-driven dev

by John Vester · Jan. 1

Like (13) Comment

Computer Programming To Be Officially Renamed "Googling Stackoverflow"

103k SHARES Facebook 101k Twitter 10 Google+ 1.1k LinkedIn 824 Reddit

Washington DC – The IEEE have produced a report today where they strongly recommend that from now on, the discipline of Computer Programming should be officially renamed to "Googling Stackoverflow"

[1] http://www.quantcast.com/stackoverflow.com January 2016.

Usefulness of Q&A's Code Snippets

code reuse and analysis

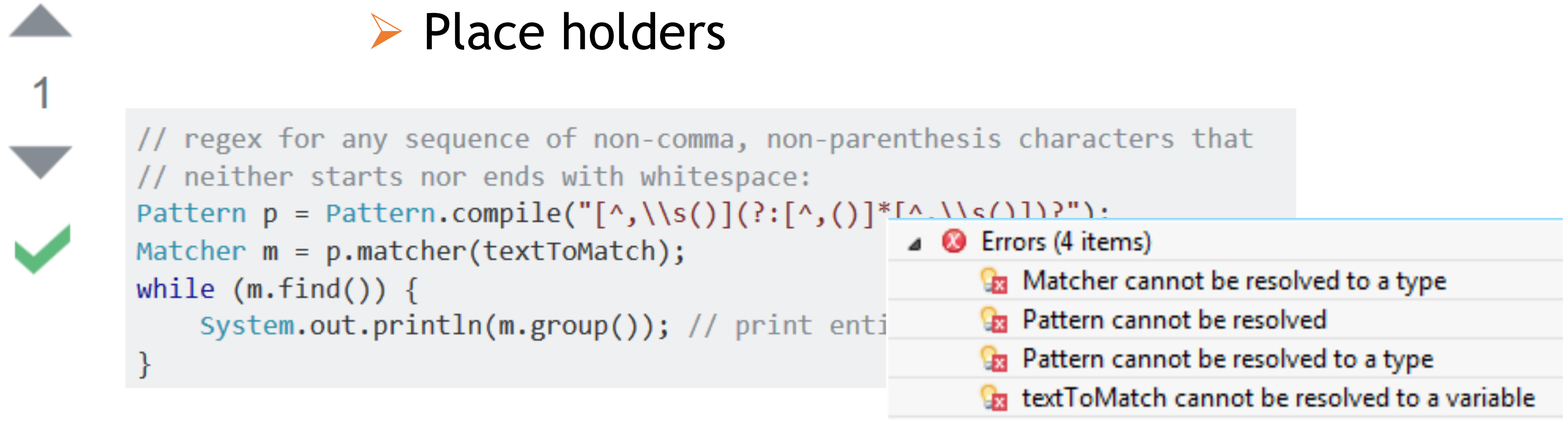
Dynamic/Static Analysis

- **Collect API usage profiles**
  - Regression testing
  - Mining temporal specifications
- **Crowd debugging** [Chen@FSE2015]
- **Crowd bug fixing** [Gao@ASE2015]

Many Code Snippets Do Not Compile

Written concisely, without implementation details [Naeshi@ICSM2012]

- Absence of import declarations or fully qualified names
- Dangling statements/methods
- Typos
- Place holders



1

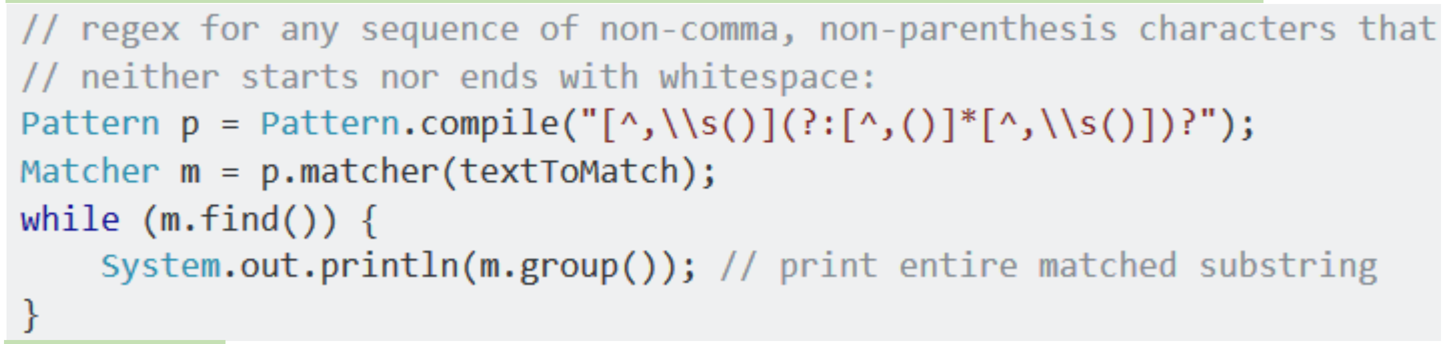
```
// regex for any sequence of non-comma, non-parenthesis characters that
// neither starts nor ends with whitespace:
Pattern p = Pattern.compile("[^,\\s()](?:[^,()]*[^,\\s()])?");
Matcher m = p.matcher(textToMatch);
while (m.find()) {
    System.out.println(m.group()); // print enti
}
```

```
Errors (4 items)
    Matcher cannot be resolved to a type
    Pattern cannot be resolved
    Pattern cannot be resolved to a type
    textToMatch cannot be resolved to a variable
```

Many code snippets are non-executable and semantically incomplete for precise static analysis

Many Code Snippets Do Not Compile

```
import java.util.regex.Matcher;                    ++
import java.util.regex.Pattern;                    ++

public class Answer9745185 {                       ++
private static CharSequence textToMatch;           ++

public static void main(String[] args){            ++
```



```
// regex for any sequence of non-comma, non-parenthesis characters that
// neither starts nor ends with whitespace:
Pattern p = Pattern.compile("[^,\\s()](?:[^,()]*[^,\\s()])?");
Matcher m = p.matcher(textToMatch);
while (m.find()) {
    System.out.println(m.group()); // print entire matched substring
}
```

```
  }   ++
}     ++
```

Manual synthesis

- Tedious
- Requires familiarity with libraries
- Not scalable

Can we do it **automatically?**

Problem Understanding
baseline synthesis

stackoverflow

Accepted answers or with score >= 2

491,906

Posts with Java code snippets

Baseline Synthesis

- Download external JARs from fully qualified names
- Create synthetic classes and methods for dangling statements/methods

Apache maven

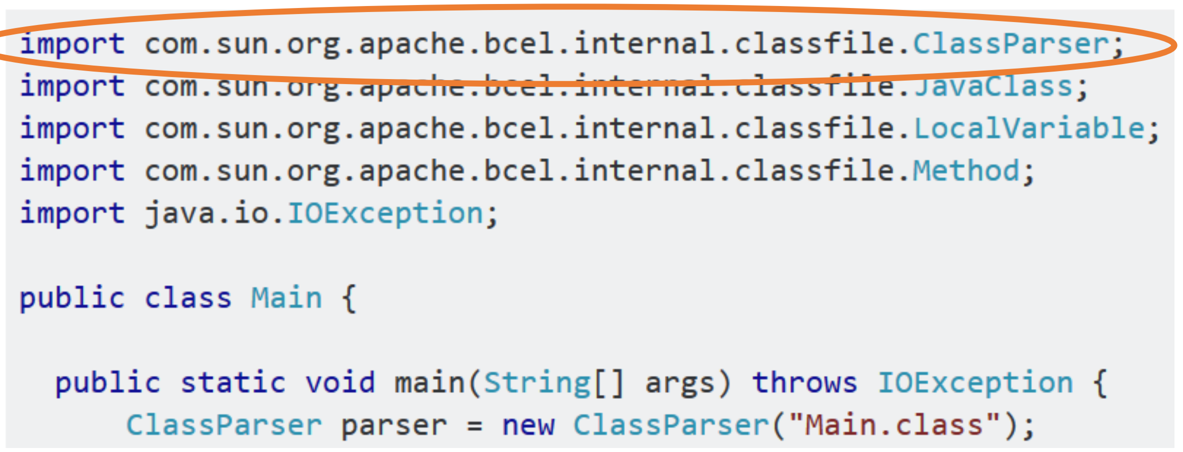
JAR

Sun ORACLE

Jaxp Ri

Jaxp Ri

https://mvnrepository.com/artifact/com.sun.org.apache/jaxp-ri/1.4

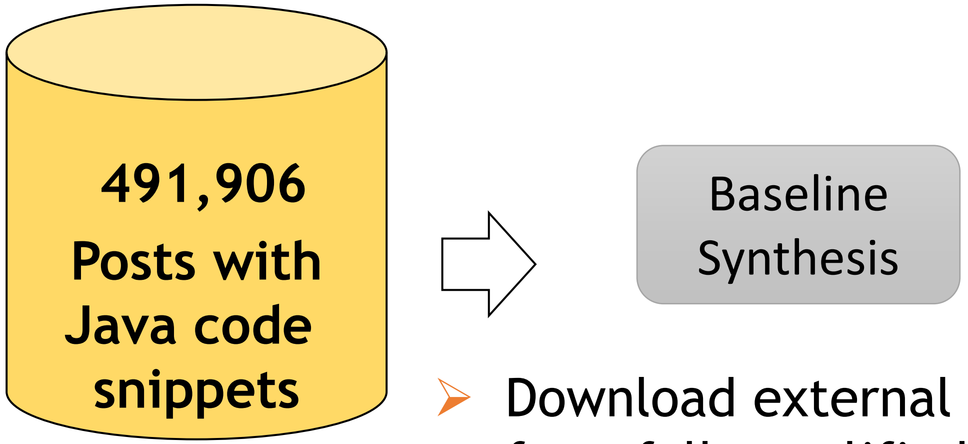


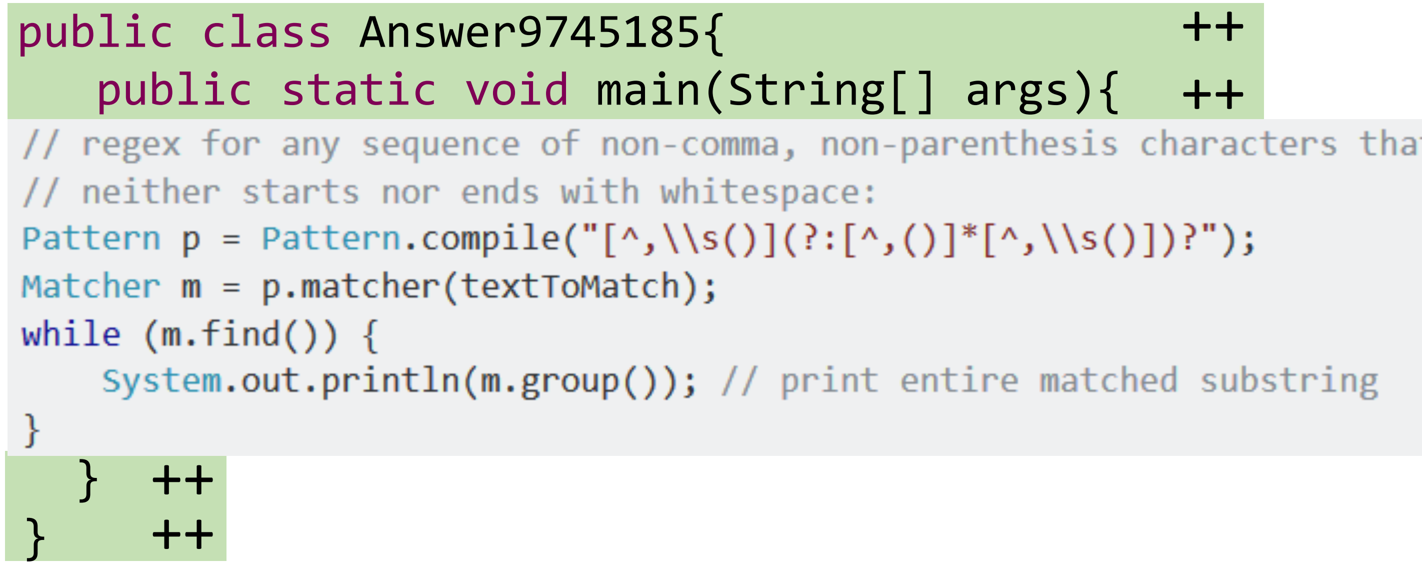
```
import com.sun.org.apache.bcel.internal.classfile.ClassParser;
import com.sun.org.apache.bcel.internal.classfile.JavaClass;
import com.sun.org.apache.bcel.internal.classfile.LocalVariable;
import com.sun.org.apache.bcel.internal.classfile.Method;
import java.io.IOException;

public class Main {

  public static void main(String[] args) throws IOException {
      ClassParser parser = new ClassParser("Main.class");
```

Problem Understanding
baseline synthesis

stackoverflow

Accepted answers or with score >= 2





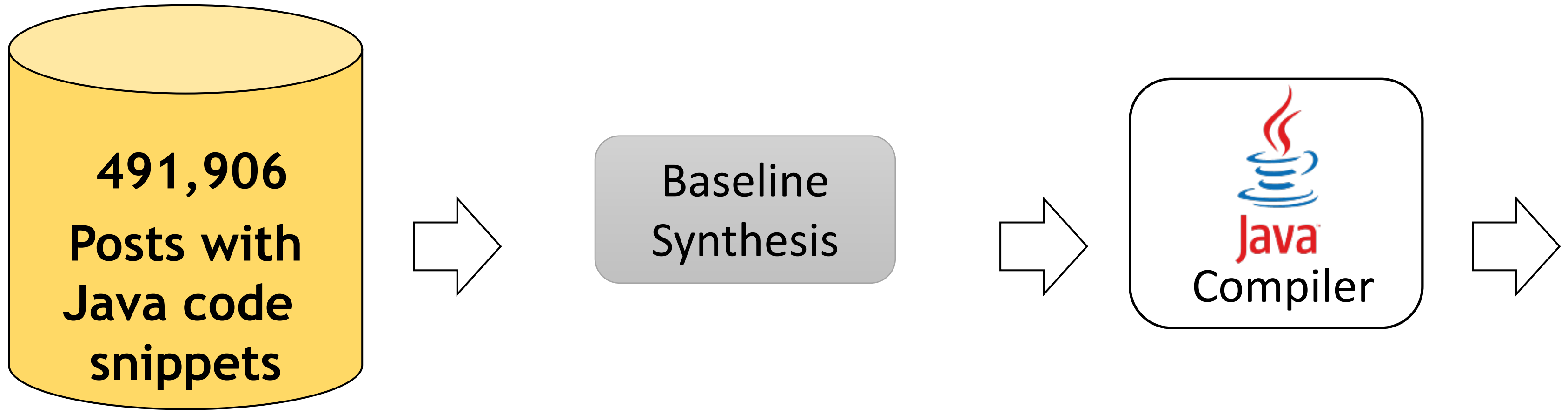
```
public class Answer9745185{                          ++
   public static void main(String[] args){   ++
// regex for any sequence of non-comma, non-parenthesis characters tha
// neither starts nor ends with whitespace:
Pattern p = Pattern.compile("[^,\\s()](?:[^,()]*[^,\\s()])?");
Matcher m = p.matcher(textToMatch);
while (m.find()) {
    System.out.println(m.group()); // print entire matched substring
}
  }  ++
}    ++
```

- Download external JARs from fully qualified names
- Create synthetic classes and methods for dangling statements/methods

Problem Understanding
baseline synthesis

stackoverflow

Accepted answers
or with score >= 2



Only 8.41% (41,349)
successfully compile

Missing Declarations
most common error type

3,905,444 compilation errors

Top@	Error code	freq.	%
Top 1	`compiler.err.cant.resolve`	1,485,626	38.04%
Top 2	`compiler.err.expected`	1,188,663	30.44%
Top 3	`compiler.err.not.stmt`	256,926	6.58%

class	950,324	(24.33%)
variable	484,035	(12.39%)
method	50,677	(1.30%)
others	590	(0.02%)

Main reasons

1. Wrong inference of compilation-units
2. Missing external dependencies
3. Undeclared variables

How To Infer Compilation Units?

35.71% (175,653) stackoverflow posts contain multiple code snippets

Strategy 1 (baseline synthesis) each code snippet in a separate Java class/file

Example 1:

```
public class C1{                ++
 static void a(){
 […]
 }
}                               ++
```
txtxtxtxtxtxtxt xtxtx xtxtx
```
public class C2{                ++
 static void b(){
 […]
 }
}                               ++
```
txtx.
```
public class C3{                ++
 public static void
 main(String[] args){
 a();
 b();
 }
}                               ++
```

compilation errors

```
The method a() is undefined for the type C3
The method b() is undefined for the type C3
```

`compiler.err.cant.resolve`

Example 2:

```
public class C1{                ++
 public void a(){
 […]
 }
}                               ++
```
txtxtxt. tx.
```
public class C2{                ++
 public void a(){
 […]
 }
}                               ++
```

How To Infer Compilation Units?

35.71% (175,653) stackoverflow posts contain multiple code snippets

Strategy 2: always merge all code snippets in a post in the same Java class

Example 1:

```
public class C1{                ++
 static void a(){
 […]
 }
```

txtxtxtxtxtxtxt xtxtx xtxtx
txtxtxtxtxtxtxt xtxtx xtxtx

```
 static void b(){
 […]
 }
```

txtxtxt xtxtxtxt xtxtx xtxt
txtx.

```
 public static void
 main(String[] args){
 a();
 b();
 }
}                               ++
```

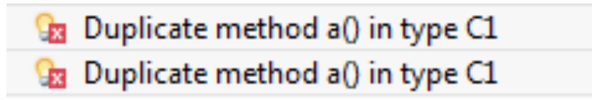
Example 2:

```
public class C1{                ++
 public void a(){
 […]
 }
```

txtxtxt, tx.

```
 public void a(){
 […]
 }
}                               ++
```

compilation errors



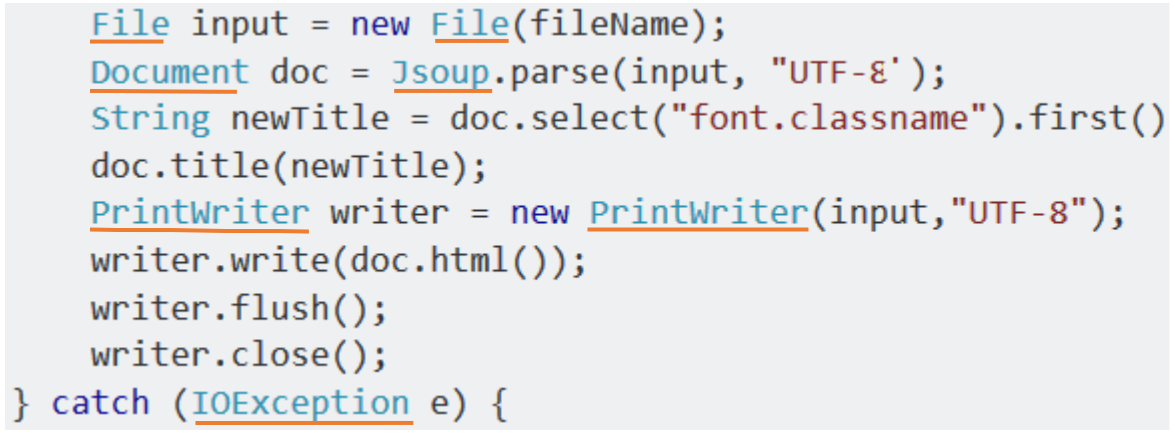
Duplicate method a() in type C1
Duplicate method a() in type C1

```
compiler.err
.already.defined
```

How To Resolve Missing Dependencies?

Only 6.88% (33,833) posts contain import declarations

A simple name can match many fully qualified names in different libraries [Subramanian@ICSE2014]



```
    File input = new File(fileName);
    Document doc = Jsoup.parse(input, "UTF-8');
    String newTitle = doc.select("font.classname").first()
    doc.title(newTitle);
    PrintWriter writer = new PrintWriter(input,"UTF-8");
    writer.write(doc.html());
    writer.flush();
    writer.close();
} catch (IOException e) {
```

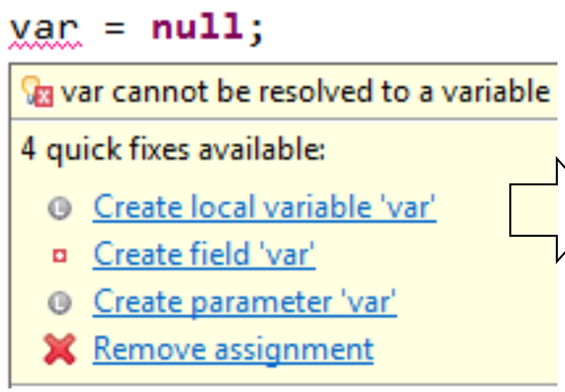
File	IOException	PrintWriter	Document	Jsoup
org.specs.runner	com.sun.star.io	java.io	org.bson	org.jsoup
scala.reflect.io	java.io		org.jdom	
java.io	net.kuujo.vertigo.io		org.jsoup.nodes	
…..	…….		……	

10 * 14 * 1 * 97 * 1 = **13,580**

on average (in our experiments) # possibile configurations for each Java file is **2.51 x 10^{34}**

How To Declare Undeclared Variables?

Quick-fix

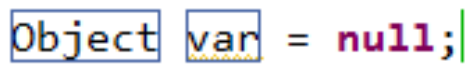


```
var = null;
```

var cannot be resolved to a variable

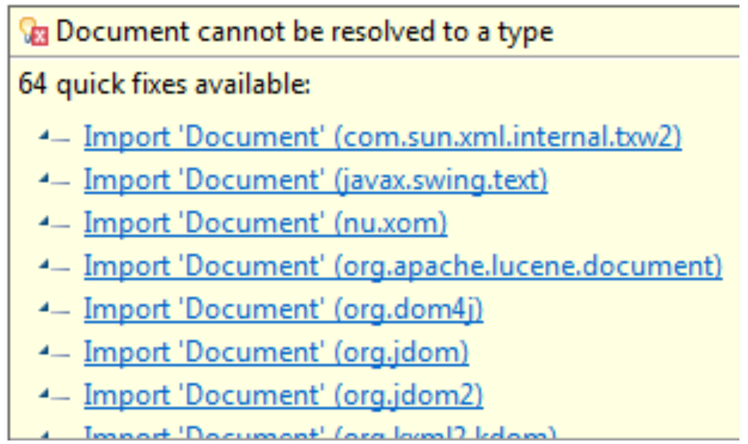
4 quick fixes available:

- Create local variable 'var'
- Create field 'var'
- Create parameter 'var'
- Remove assignment



```
Object var = null;
```

➢ Does not suggest which import declaration to generate



Document cannot be resolved to a type

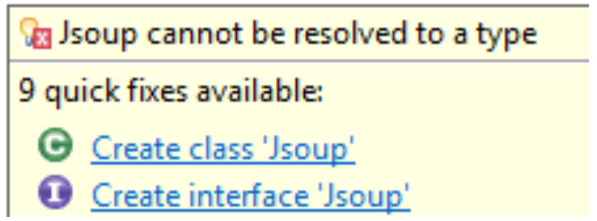
64 quick fixes available:

- Import 'Document' (com.sun.xml.internal.txw2)
- Import 'Document' (javax.swing.text)
- Import 'Document' (nu.xom)
- Import 'Document' (org.apache.lucene.document)
- Import 'Document' (org.dom4j)
- Import 'Document' (org.jdom)
- Import 'Document' (org.jdom2)
- Import 'Document' (org.kxml2.kdom)

How to partition multiple code snippets in Java files?

How to recover external dependencies by simple names?

➢ Without the right JAR in the buildpath it can only suggest to mock declarative completeness

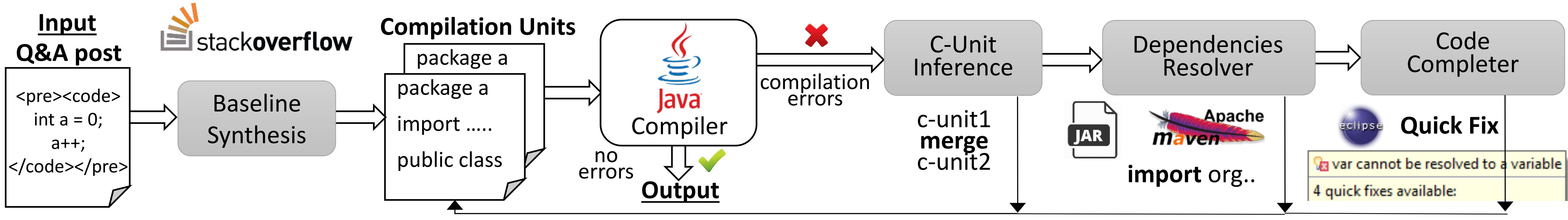


Jsoup cannot be resolved to a type

9 quick fixes available:

- Create class 'Jsoup'
- Create interface 'Jsoup'

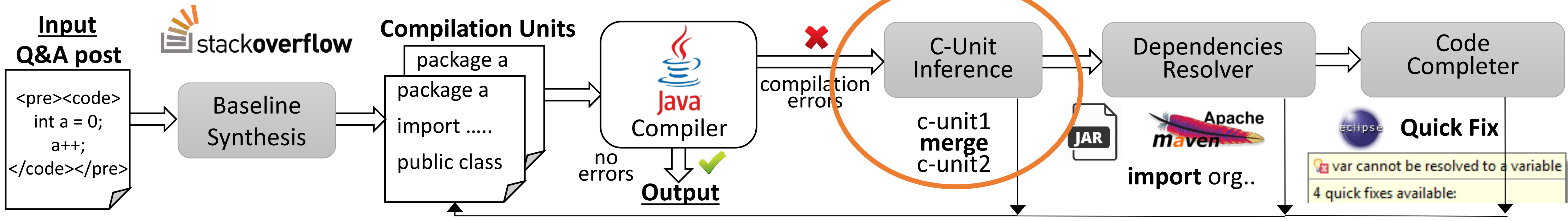
CSNIPPEX
Code SNIPPet Extractor



- **Feedback-directed approach**
  guided by compilation errors

- C-units inference and dependency resolution prepare the working environment for Eclipse Quick Fix

C-Unit Inference



C-Unit Inference

Follow the order of occurrence!

Example 1:

```
public class C1{          ++
 static void a(){
 […]
 }
```

txtxtxtxtxtxtxt xtxtx xtxtx
txtxtxtxtxtxtxt xtxtx xtxtx

```
 static void b(){
 […]
 }
}                         ++
```

txtx.

```
public static void
main(String[] args){
a();
b();
}
}                         ++
```

✓

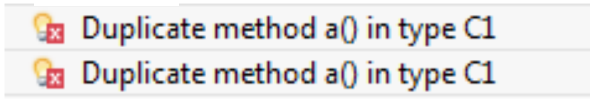
```
compiler.err
.already.defined ?
```

YES unmerge [✓ NO keep]

Example 2:

```
public class C1{          ++
 public void a(){
 […]
 }
public class C2{          ++
}                         ++
 public void a(){
 […]
 }
}                         ++
```

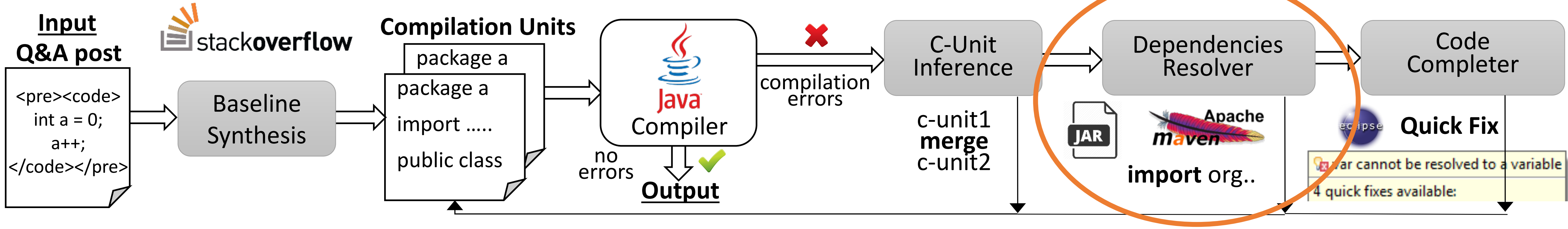
✓


Duplicate method a() in type C1
Duplicate method a() in type C1

```
compiler.err
.already.defined ?
```

[✓ YES unmerge] NO keep

Dependencies Resolver



Clustering Hypothesis

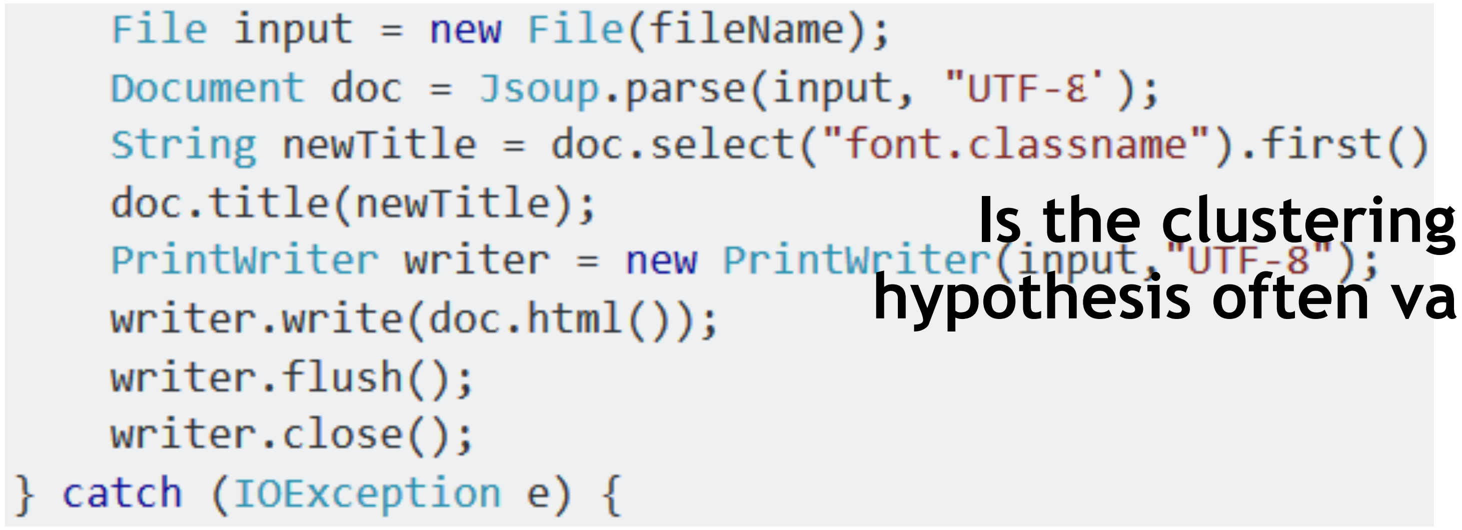
Import declarations in the same compilation unit likely form clusters naturally, each of which refers to a package or sub-package

correct import declaration

```
import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
```

Why?

Types from the same package more likely interact with one another than with those from other packages.



```
    File input = new File(fileName);
    Document doc = Jsoup.parse(input, "UTF-8' );
    String newTitle = doc.select("font.classname").first()
    doc.title(newTitle);
    PrintWriter writer = new PrintWriter(input, "UTF-8");
    writer.write(doc.html());
    writer.flush();
    writer.close();
} catch (IOException e) {
```

Is the clustering hypothesis often valid?

Validating the Clustering Hypothesis

Import declarations in the same compilation unit likely form clusters naturally, each of which refers to a package or sub-package

I

```
import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
```

- **Distance between two packages**
  $d(p_A,p_B)$ = the length of the longest uncommon suffix

$$\tau = 2$$

d(java.util, java.util)= 0
d(org.jsoup, java.util) = 2
d(org.jsoup, org.jsoup.nodes)= 1
d(java.util, org.jsoup.nodes)= 3

- $\mathcal{P}_I^{\tau}$

partition of I such that each pair of packages in the same subset (cluster) has a distance less than a threshold **τ**

Heterogeneity Degree

# clusters

$$HD_I^2 = \frac{2}{5} \cdot 100 = 40$$

# import declarations

Validating the Clustering Hypothesis

~31 Million **complete** (compilable) Java files
~198 Million import declarations

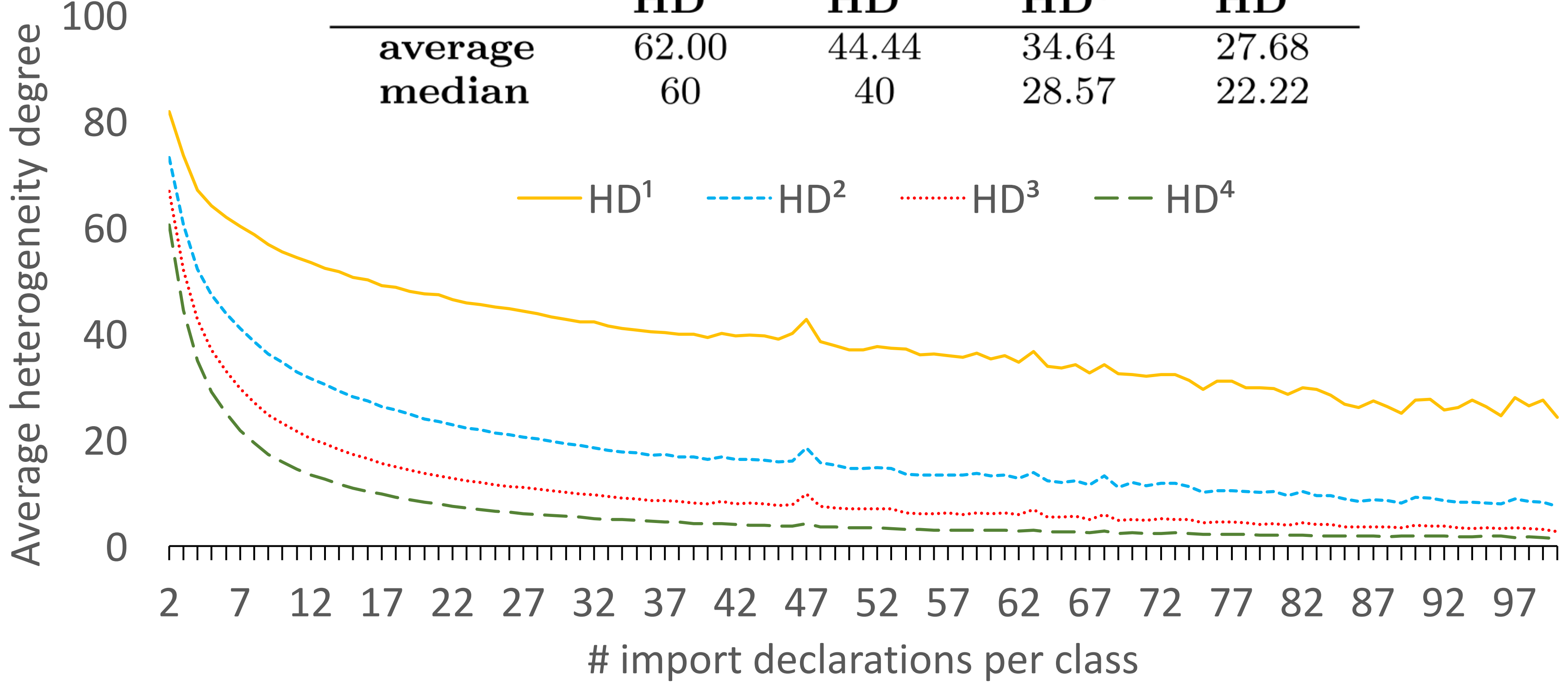
boa
Mining Ultra–Large–Scale Software Repositories

http://boa.cs.iastate.edu/stats/index.php **Dyer@ICSE2013**

GitHub

sourceforge

	HD1	HD2	HD3	HD4
average	62.00	44.44	34.64	27.68
median	60	40	28.57	22.22



Dependencies Resolver

- A solution with low HD is more likely to be the correct one

- Too expensive to enumerate all possible solutions and compute HD
  - We propose a **greedy** algorithm

STEP1: compute the global frequency for each package

File	IOException	PrintWriter	Document	Jsoup
1 org.specs.runner	1 com.sun.star.io	3 java.io	1 org.bson	1 org.jsoup
1 scala.reflect.io	3 java.io		1 org.jdom	
3 java.io	1 net.kuujo.vertigo.io		1 org.jsoup.nodes	
.....	....		......	

Dependencies Resolver

- A solution with low HD is more likely to be the correct one

- Enumerate all possible solutions and compute HD is too expensive
  - We propose a **greedy** algorithm

File	IOException	PrintWriter	Document	Jsoup
3 java.io	3 java.io	3 java.io	1 org.bson	1 org.jsoup
1 org.specs.runner	1 com.sun. star.io		1 org.jdom	
1 scala.reflect.io	1 net.kuujo.vertigo.io		1 org.jsoup.nodes	
.....	....		......	

STEP1: compute the global frequency for each package

STEP2: For each simple name order packages by their frequency

Dependencies Resolver

- A solution with low HD is more likely to be the correct one
- Enumerate all possible solutions and compute HD is too expensive
  - We propose a **greedy** algorithm

STEP1: compute the global frequency for each package

STEP2: For each simple name order packages by their frequency

TOP solution has the biggest cluster

File	IOException	PrintWriter	Document	Jsoup
3 java.io	3 java.io	3 java.io	1 org.bson	1 org.jsoup
1 org.specs.runner	1 com.sun. star.io		1 org.jdom	
1 scala.reflect.io	1 net.kuujo.vertigo.io		1 org.jsoup.nodes	
.....	....		......	

```
import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;
```

Dependencies Resolver

- A solution with low HD is more likely to be the correct one
- Enumerate all possible solutions and compute HD is too expensive
  - We propose a **greedy** algorithm

File	IOException	PrintWriter	Document	Jsoup
3 java.io	3 java.io	3 java.io	1 ~~org.bson~~	1 org.jsoup
1 org.specs.runner	1 com.sun. star.io		1 org.jdom	
1 scala.reflect.io	1 net.kuujo.vertigo.io		1 org.jsoup.nodes	
.....	....		......	

STEP1: compute the global frequency for each package

STEP2: For each simple name order packages by their frequency

STEP3: Refining the top solution

by compilation errors

Dependencies Resolver

- A solution with low HD is more likely to be the correct one

- Enumerate all possible solutions and compute HD is too expensive
  - We propose a **greedy** algorithm

File	IOException	PrintWriter	Document	Jsoup
3 java.io	3 java.io	3 java.io	2 org.jsoup.nodes	1 org.jsoup
1 org.specs.runner	1 com.sun. star.io		1 org.jdom	
1 scala.reflect.io	1 net.kuujo.vertigo.io		1 org.bson	
.....	....		......	

```
import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
```

STEP1: compute the global frequency for each package

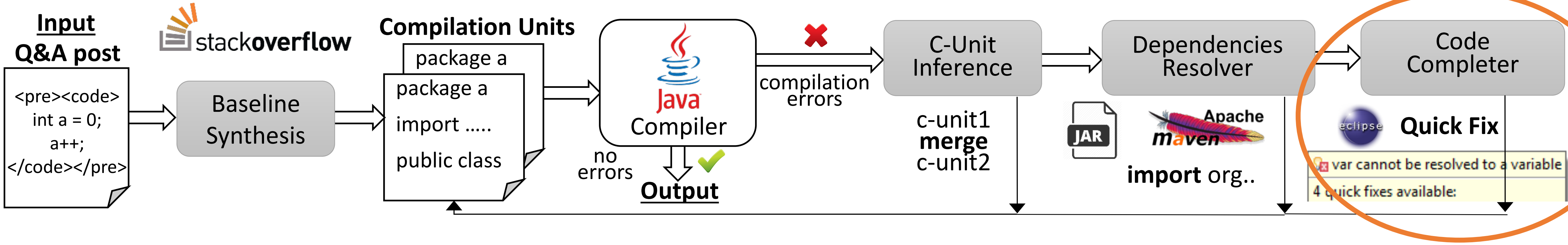
STEP2: For each simple name order packages by their frequency

STEP3: Refining the top solution

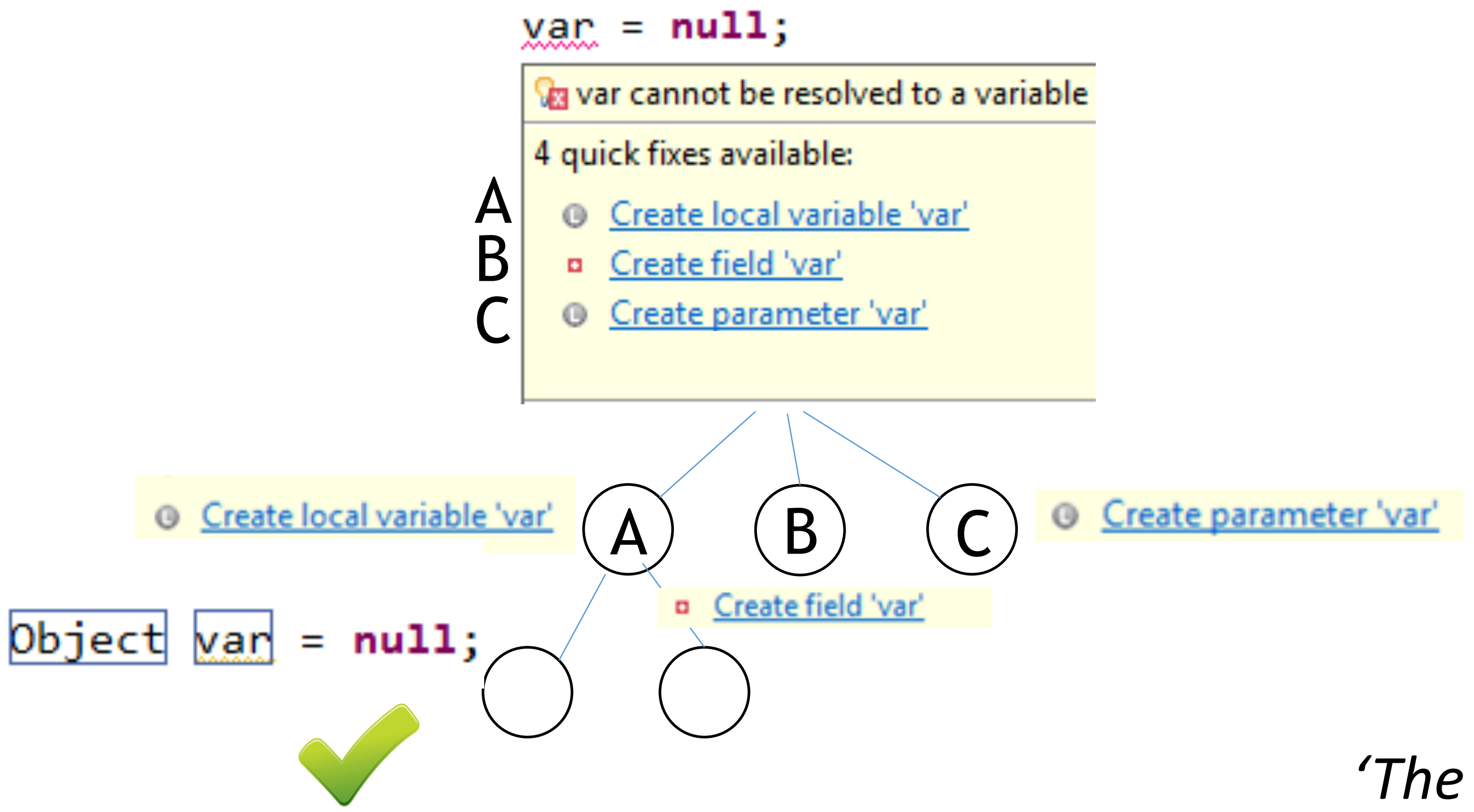
by compilation errors

by higher density threshold

Code Completer



Code Completer



```
var = null;
```

var cannot be resolved to a variable

4 quick fixes available:

- A Create local variable 'var'
- B Create field 'var'
- C Create parameter 'var'

```
Object var = null;
```

eclipse Quick-fix

Systematic exploration of suggested quick-fixes

Occam's razor[1]

'The simplest answer is most often correct!'

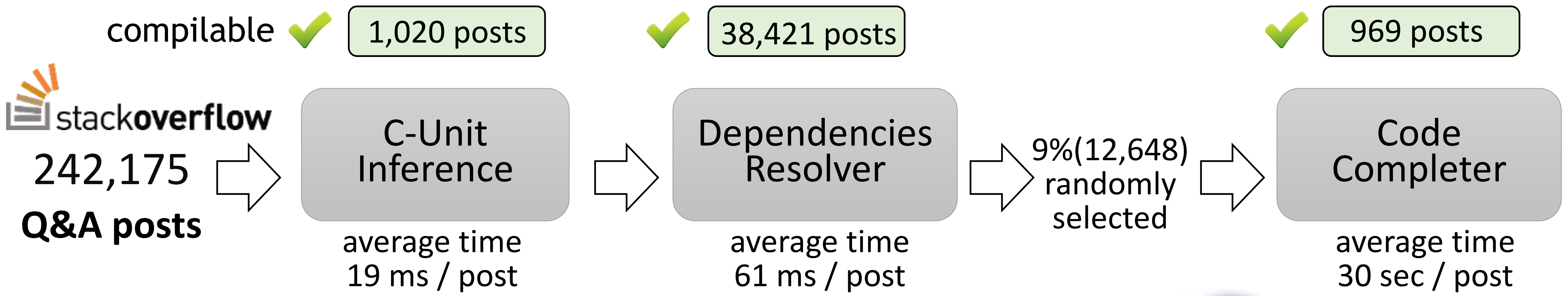
⇩

Breadth First Search (BFS)

[1] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's razor. Information processing letters, 1987.

Evaluation
RQ1 synthesis effectiveness

- Download 3,000 popular jars from Apache maven
- 242,175 posts with **at least one** `compiler.err.cant.resolve` error



Top@	Error code	freq.	%
Top 1	`compiler.err.cant.resolve`	1,485,626	38.04%
Top 2	`compiler.err.expected`	1,188,663	30.44%
Top 3	`compiler.err.not.stmt`	256,926	6.58%

Many errors are outside the scope of the paper

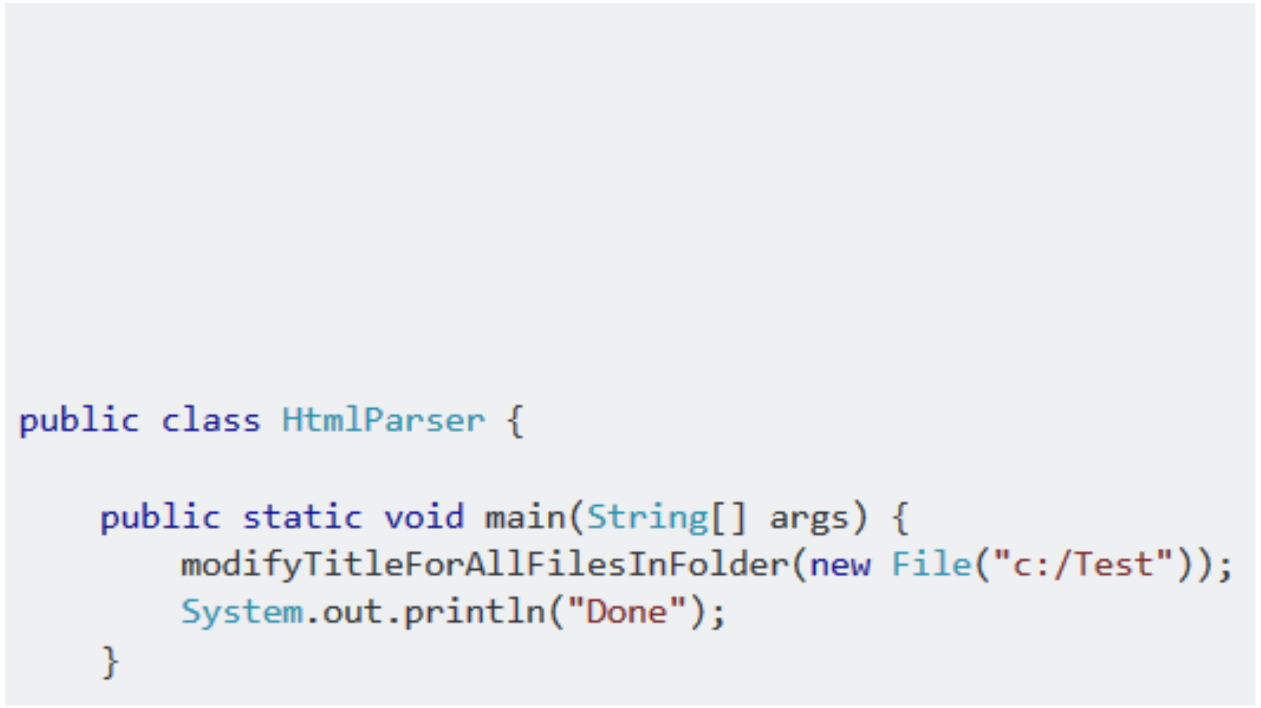
eclipse **Only variable generation**

Evaluation

RQ2 precision of the dependencies resolving

Golden set: **13,444 compilable** code snippets with import declarations

stackoverflow



```
public class HtmlParser {

    public static void main(String[] args) {
        modifyTitleForAllFilesInFolder(new File("c:/Test"));
        System.out.println("Done");
    }
```

We **removed** the user-specified import declarations to evaluate to what extent CSNIPPEX is able to recover them

Evaluation

RQ2 precision of the dependencies resolving

Golden set: **13,444 compilable** code snippets **with** import declarations

solution Top@	% compile	% equivalent	average time each post (ms)	median time each post (ms)
Top1	76.87%	76.30%	66	32
Top10	89.66%	87.35%	103	47
Top100	91.04%	88.27%	4,454	1,889

Clustering by [illegible]
is effective

Refining hypothesis is
effective

Compilation is a good
proxy for correctness

Efficient

Evaluation

RQ2 comparison with Baker [Subramanian@ICSE2014]

Golden set: **13,444 compilable** code snippets **with** import declarations

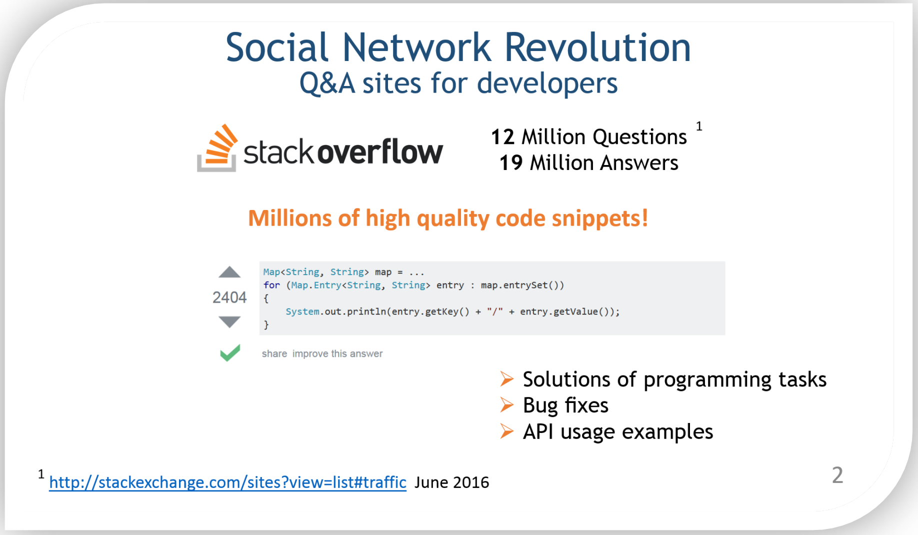
↓

BAKER state-of-the-art
in API link recovering

↓

It found unique matches
of external class types for
36.71% code snippets

Conclusion



Social Network Revolution
Q&A sites for developers

stackoverflow — 12 Million Questions [1]
19 Million Answers

Millions of high quality code snippets!

2404

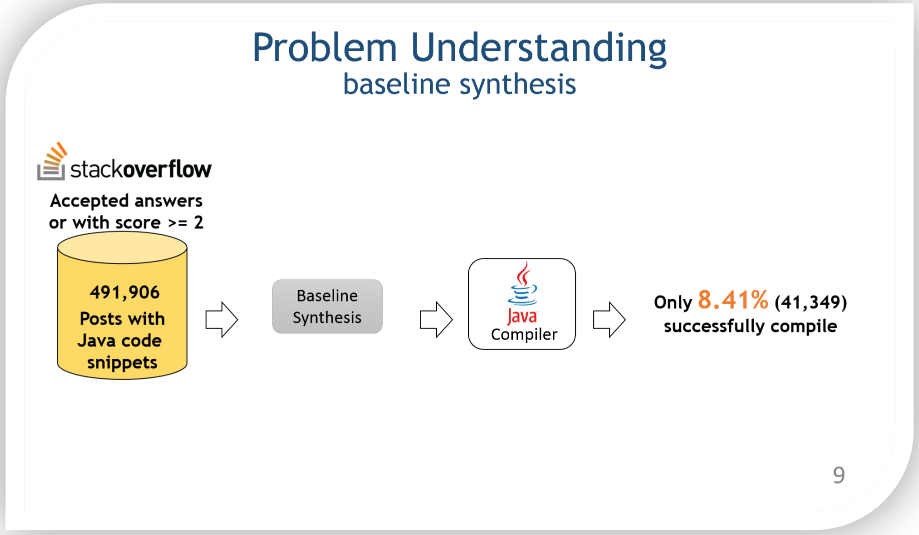
```
Map<String, String> map = ...
for (Map.Entry<String, String> entry : map.entrySet())
{
    System.out.println(entry.getKey() + "/" + entry.getValue());
}
```

share improve this answer

- Solutions of programming tasks
- Bug fixes
- API usage examples

[1] http://stackexchange.com/sites?view=list#traffic June 2016

2

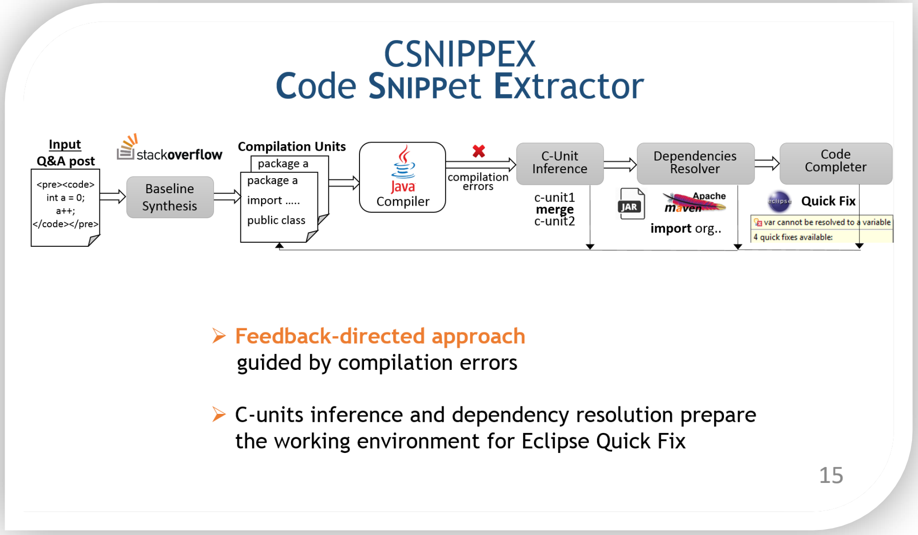


Problem Understanding
baseline synthesis

stackoverflow
Accepted answers or with score >= 2

491,906 Posts with Java code snippets → Baseline Synthesis → Java Compiler → Only 8.41% (41,349) successfully compile

9



CSNIPPEX
Code SNIPPet EXtractor

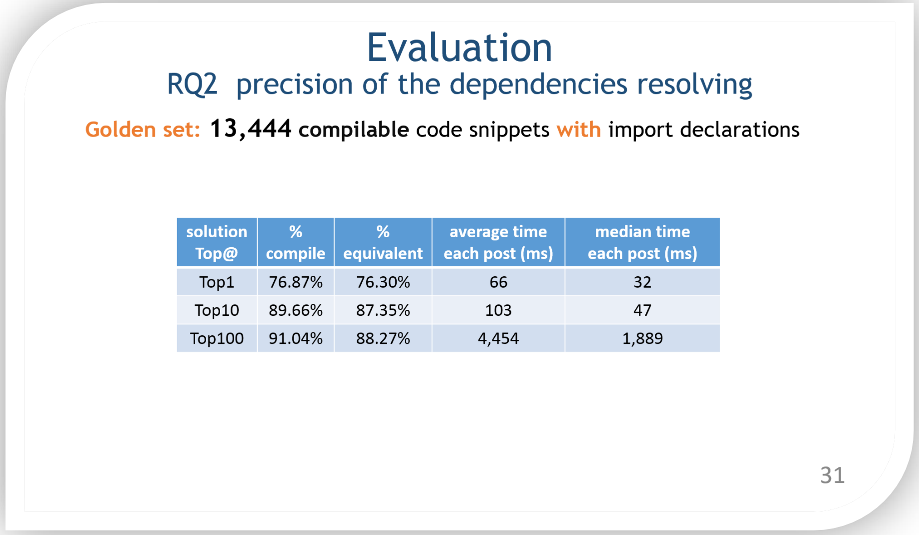
Input Q&A post

```
<pre><code>
int a = 0;
a++;
</code></pre>
```

stackoverflow → Baseline Synthesis → Compilation Units (package a / package a, import ....., public class) → Java Compiler → compilation errors → C-Unit Inference (c-unit1 merge c-unit2) → Dependencies Resolver (JAR, Apache maven, import org..) → Code Completer (eclipse Quick Fix: var cannot be resolved to a variable; 4 quick fixes available:)

- Feedback-directed approach
  guided by compilation errors
- C-units inference and dependency resolution prepare the working environment for Eclipse Quick Fix

15



Evaluation
RQ2 precision of the dependencies resolving

Golden set: 13,444 compilable code snippets with import declarations

solution Top@	% compile	% equivalent	average time each post (ms)	median time each post (ms)
Top1	76.87%	76.30%	66	32
Top10	89.66%	87.35%	103	47
Top100	91.04%	88.27%	4,454	1,889

31

Future Work

- Focus on other types of error (place holders, broken code snippets etc..)

Top@	Error code	freq.	%
Top 1	`compiler.err.cant.resolve`	1,485,626	38.04%
Top 2	`compiler.err.expected`	1,188,663	30.44%
Top 3	`compiler.err.not.stmt`	256,926	6.58%

- Compilability is only a necessary but not a sufficient condition to obtain executable code
  - Automated Synthesis of **Executable** Code Snippets from Q&A Sites
    - Feedback-directed approach guided by **runtime exceptions**

- Use the stackoverflow code snippets for **regression testing** API libraries

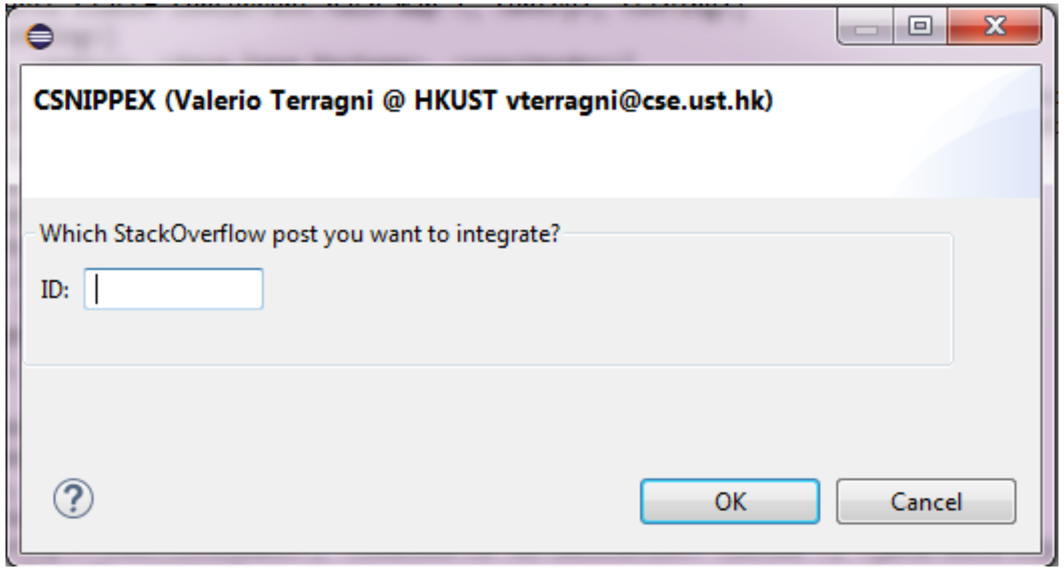
CXNIPPEX tool

&

dataset of 93,092 compilable code snippets

are available at

http://sccpu2.cse.ust.hk/csnippex/


CSNIPPEX (Valerio Terragni @ HKUST vterragni@cse.ust.hk)

Which StackOverflow post you want to integrate?

ID:

BACKUP SLIDES

Dependencies Resolver

- A solution with low HD is more likely to be the correct one

- Enumerate all possible solutions and compute HD is too expensive
  - We propose a **greedy** algorithm

Temporary ignore a package if it is involved in a compilation error

Why temporary?

Example

```
the constructor
java.io.PrintWriter(scala.io.File, java.lang.String)
is undefined.
```

Either
`java.io.PrintWriter`
or
`scala.io.File`
could be wrong

STEP1: compute the global frequency for each package

STEP2: For each simple name order packages by their frequency

STEP3: Refining the top solution

by compilation errors