
ISSTA 2015 Demo

Feedback-controlled Random Test Generation

Kohsuke Yatch^{1*}, Kazunori Sakamoto², Fuyuki Ishikawa²,
Shinichi Honiden¹²

1:University of Tokyo, 2:National Institute of Informatics

* He is currently affiliated with Google Inc., Japan. All work is done in Univ. Tokyo and nothing to do with Google.

Introduction

Today's demonstrations:

- Randoop (with multi-pool generator)
 - To generate random tests
- OpenCodeCoverageFramework
 - To instrument Java sources
 - To observe coverage on the fly

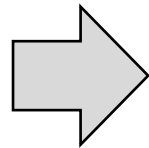
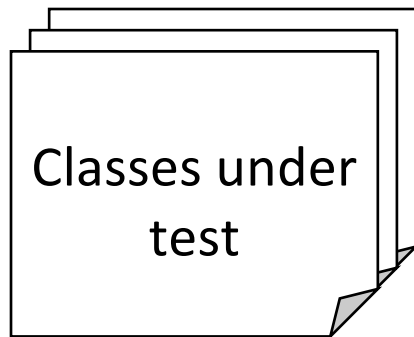
All artifacts are available on

<http://www.klazz.net/pub/issta2015/>

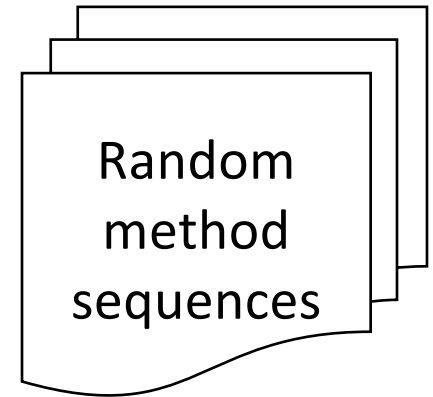
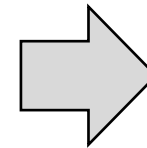
Randoop

Random tester for object-oriented programs

[ICSE 07, <http://mernst.github.io/randoop/>]



Randoop



Example:

```
class List;  
class Int;
```

Example:

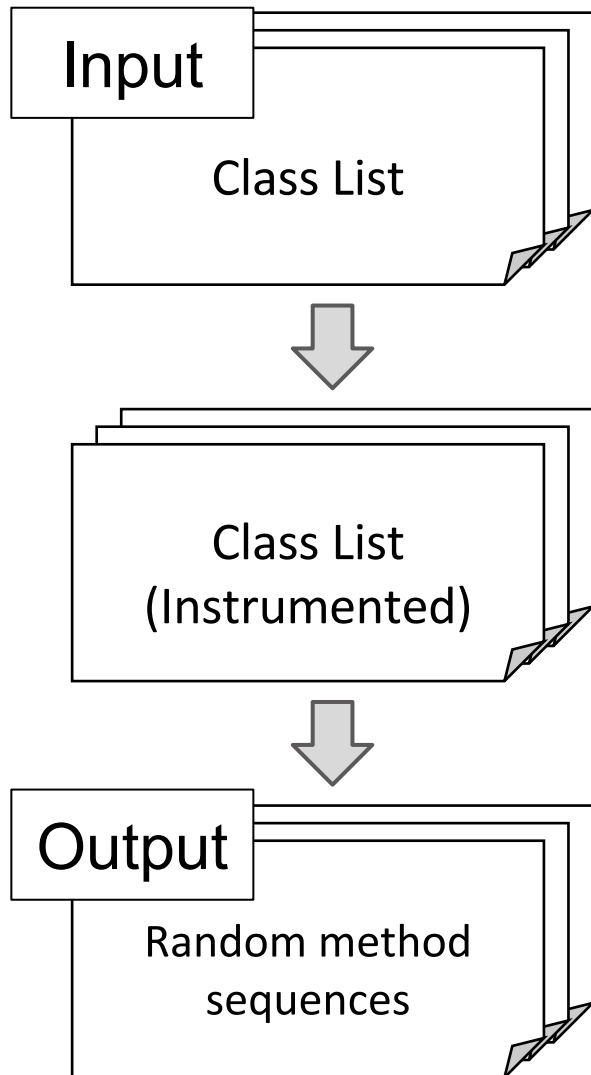
```
List l1 = new List();  
Int i1 = new Int(0);  
l1.add(i1);  
List l2 = new List();  
l2.addAll(l1);
```

Randoop Multi-pool Generator

We wrote 'multi-pool' generator for Randoop.

- Improves coverage of generated tests
 - No change on Input / Output formats
- Tracks coverage on the fly by
OpenCodeCoverageFramework
- Detailed technical talk on Friday!

Workflow



1. OpenCodeCoverage Framework

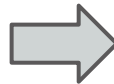
2. Randoop with multi-pool generator

OpenCodeCoverageFramework

[ICST 2013, <https://github.com/exKAZUu/OpenCodeCoverageFramework>]

- AST(CST) manipulation library for 8+ languages
- Can insert coverage handler function calls
- Not-so-optimized, but very flexible

```
int abs(int a) {  
    if (a < 0) {  
        a = -a;  
    }  
    return a;  
}
```



```
int abs(int a) {  
    H.stmt(0);  
    if (H.branch(0, a < 0)) {  
        H.stmt(1);  
        a = -a;  
    }  
    H.stmt(2);  
    return a;  
}
```

Demo 1: OpenCodeCoverageFramework

On typical project with ant build file:

1. Instrument source files

> \$ SimpleOccf.exe src/

2. Add soccf-runtime.jar to classpath

> \$ vi build.xml

3. Compile

> \$ ant

Randoop (multi-pool)

Creates random method sequences
from given class list

- We added 'multi-pool' generator
- We wrapped Randoop by Docker to isolate test environment for safe execution

Demo 2: Randoop (multi-pool) on Docker

1. *Run randoop (multi-pool) on Docker*

```
> $ python ./run.py \
    /path/to/randoop-control.jar \
    /path/to/target.jar
```

2. *Pull generated regression tests*

```
> $ sudo docker cp \
    `cat data/0/cid.txt` :/work .
```

Summary

Today's demonstrations:

- Randoop (with multi-pool generator)
 - To generate random tests
- OpenCodeCoverageFramework
 - To instrument Java sources
 - To observe coverage on the fly

All artifacts are available on

<http://www.klazz.net/pub/issta2015/>