자바 언어에 정확한 타입을 추가한 ThisJava 소개

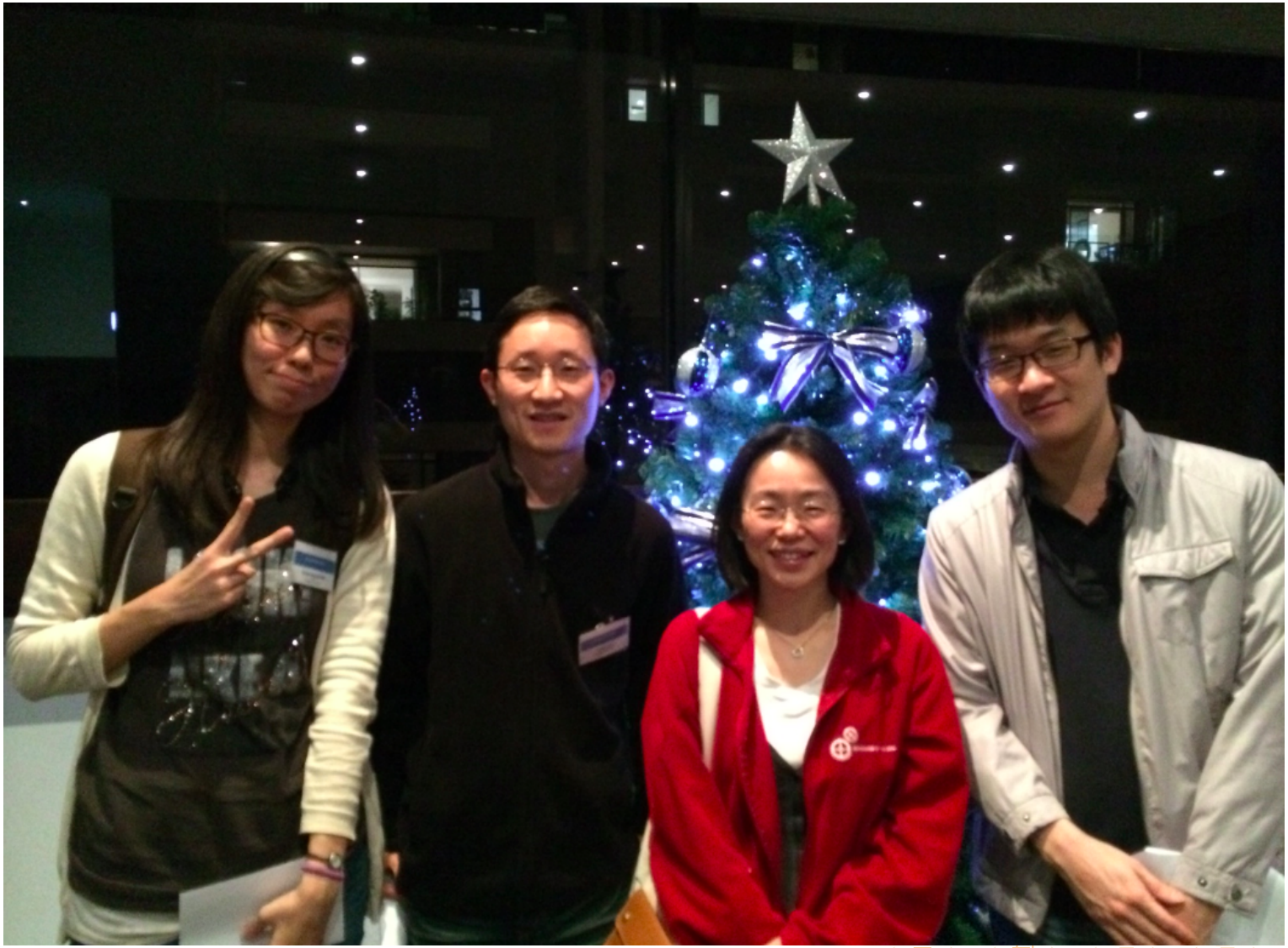
나현익, *류석영*

프로그래밍 언어 연구실

KAIST

2014년 1월 14일

APLAS 2013



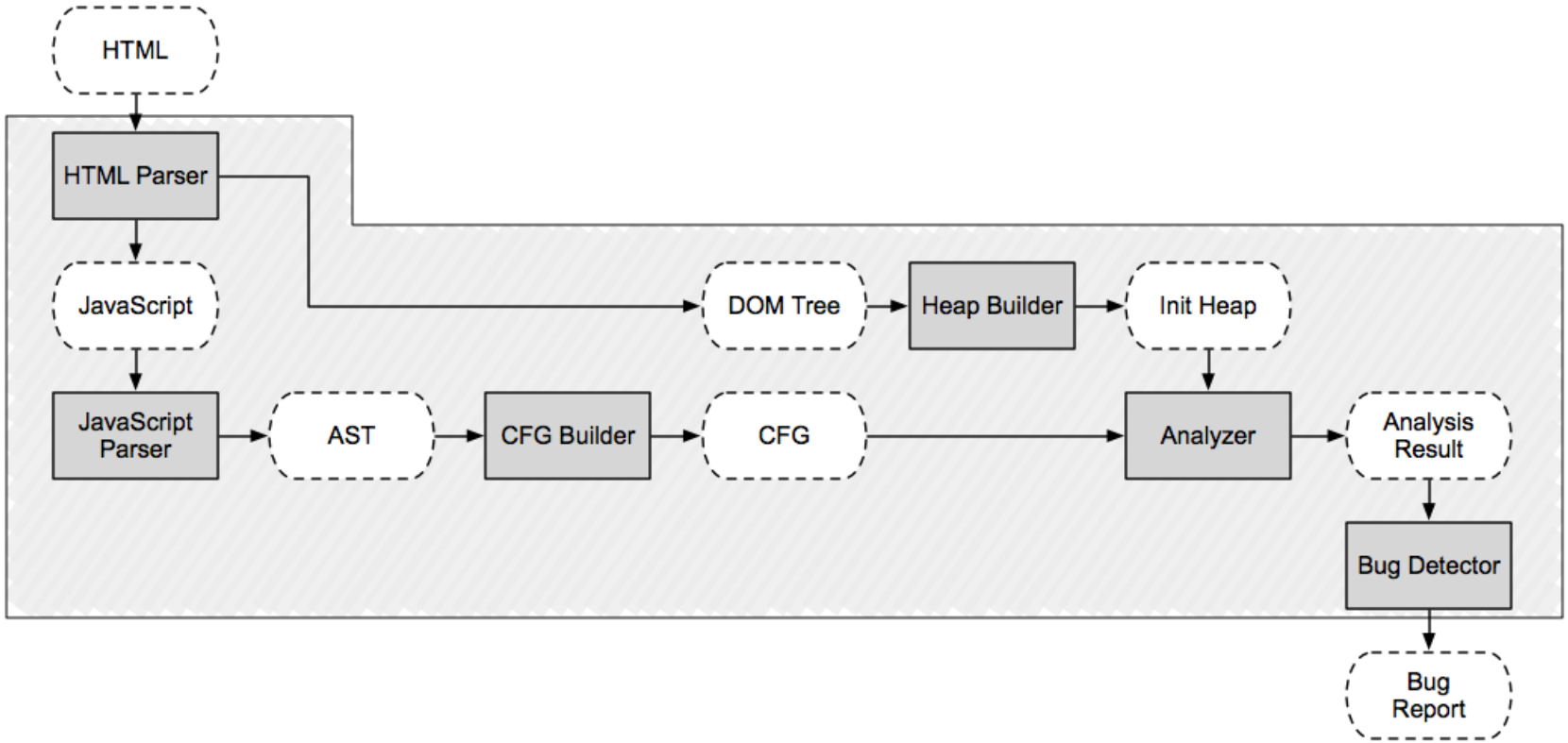
실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인

실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인

  SAFE: Scalable Analysis Framework for ECMAScript



실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인

$$\hat{v} \in \widehat{\mathsf{Value}} = \widehat{\mathsf{PValue}} \times \wp(\widehat{\mathsf{Loc}})$$

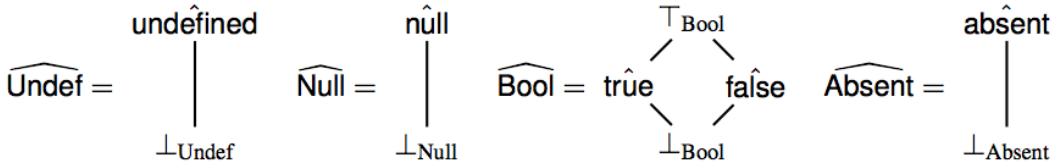
$$\hat{pv} \in \widehat{\mathsf{PValue}} = \widehat{\mathsf{Undef}} \times \widehat{\mathsf{Null}} \times \widehat{\mathsf{Bool}} \times \widehat{\mathsf{Number}} \times \widehat{\mathsf{String}}$$

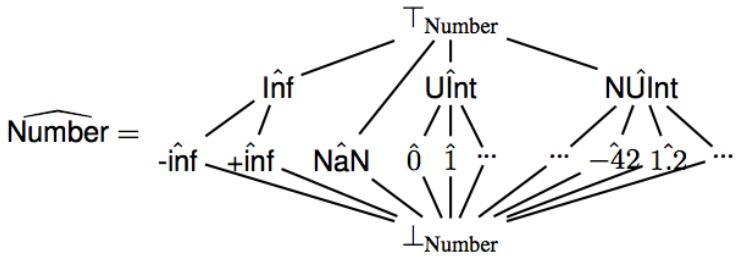
실제로 부딪힌 문제

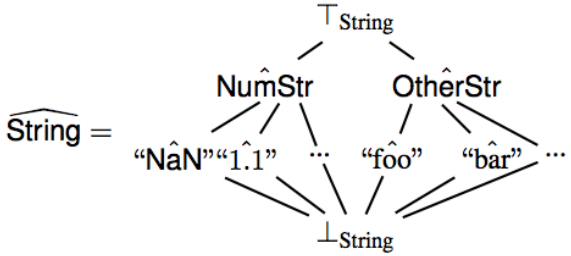
- 자바스크립트 프로그램 분석을 위한 요약 도메인

$$\hat{v} \in \widehat{\mathsf{Value}} = \widehat{\mathsf{PValue}} \times \wp(\widehat{\mathsf{Loc}})$$

$$\hat{pv} \in \widehat{\mathsf{PValue}} = \widehat{\mathsf{Undef}} \times \widehat{\mathsf{Null}} \times \widehat{\mathsf{Bool}} \times \widehat{\mathsf{Number}} \times \widehat{\mathsf{String}}$$







실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인

```
abstract class AbsBase {
  def isTop(): Boolean
  def isBottom(): Boolean
  def isConcrete(): Boolean
  def toAbsString(): AbsString
}
```

실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인

```
sealed abstract class AbsNull extends AbsBase {
  /* partial order */
  def <= (that: AbsNull) = { ... }

  /* join */
  def + (that: AbsNull) = { ... }

  /* meet */
  def <> (that: AbsNull) = { ... }

  ...
}
```

실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인

```
sealed abstract class AbsString extends AbsBase {
  /* partial order */
  def <= (that: AbsString) = { ... }

  /* join */
  def + (that: AbsString) = { ... }

  /* meet */
  def <> (that: AbsString) = { ... }

  ...
}
```

실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인

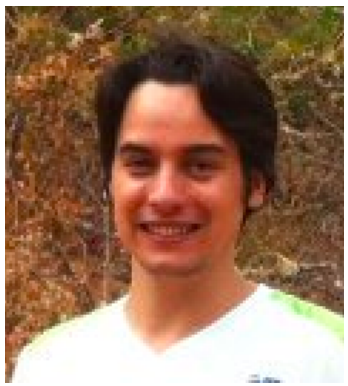
```
abstract class AbsBase {
  /* partial order */
  def <= (that: ThisType): Boolean

  /* join */
  def + (that: ThisType): ThisType

  /* meet */
  def <> (that: ThisType): ThisType
  ...
}
```

실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인
  - AbsBase에서 요구하는 함수의 인자와 결과 타입
  - 실행 시간에 정확한 클래스 타입 매칭

실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인
  - AbsBase에서 요구하는 함수의 인자와 결과 타입
  - 실행 시간에 정확한 클래스 타입 매칭
- Bruno Oliveira's solution



```
abstract class AbsBase {
  type ThisType <: AbsBase
  def + (that: ThisType): ThisType
  ...
}
case class AbsString extends AbsBase {
  type ThisType = AbsString
  override def + (that : ThisType): ThisType =
    ... new AbsString() ...
}
```

실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인
  - AbsBase에서 요구하는 함수의 인자와 결과 타입
  - 실행 시간에 정확한 클래스 타입 매칭
  - 한 요약 도메인의 여러 구현

```
abstract class AbsDomain { ... }

abstract class AbsBase[A] extends AbsDomain { ... }

class AbsString extends AbsBase[String] { ... }

class AbsStringSet extends AbsString { ... }

class AbsStringAutomata extends AbsString { ... }
```

실제로 부딪힌 문제

- 자바스크립트 프로그램 분석을 위한 요약 도메인
  - AbsBase에서 요구하는 함수의 인자와 결과 타입
  - 실행 시간에 정확한 클래스 타입 매칭
  - 한 요약 도메인의 여러 구현

```
abstract class AbsDomain { ... }

abstract class AbsBase[A] extends AbsDomain { ... }

class AbsString extends AbsBase[String] { ... }

class AbsStringSet extends AbsString { ... }

class AbsStringAutomata extends AbsString { ... }
```

실제로 부딪힌 문제

12/03/2013 12:17 pm	Sooncheol Won	[AbsDomain] Changed string domain to set domain.
12/02/2013 05:48 pm	Sooncheol Won	[AbsDomain] 1) Renamed AbsStringSimple to AbsStringSe Added an AbsStringAutomata prototype class.
12/02/2013 04:45 pm	Sooncheol Won	[AbsDomain] Separated common domain elements(StrTo from AbsStringSimple.
12/02/2013 01:48 pm	Sooncheol Won	[AbsDomain] Removed useless parenthesises.
12/02/2013 12:01 pm	Sukyoung Ryu	[AbsDomain] Gave up with ThisType to support multiple implementations for AbsString. Refactored AbsString to Al and AbsStringSimple.
12/02/2013 12:04 am	Sora Bae	Merge branch 'master' of ssh://plrg.kaist.ac.kr/var/git/sa
12/02/2013 12:03 am	Sora Bae	[Concolic] Create input objects for integer properties.
11/29/2013 10:13 pm	Sukyoung Ryu	[Refactoring] Refactored PropValue(Value(_))
11/29/2013 03:41 pm	Sooncheol Won	[Domain] Fixed the meet operator bug.
11/29/2013 12:51 pm	Sukyoung Ryu	[AbsNumber] Removed the getSingleValue method.
11/29/2013 09:34 am	Sukyoung Ryu	[AbsString] Moved one method from the AbsString object AbsString class.
11/28/2013 11:02 am	Sukyoung Ryu	[AbsDomain] Removed uses of abstract value constructor
11/21/2013 09:59 am	Sukyoung Ryu	[AbsDomain] Revised the AbsUndef domain and added the AbsCa classes for pattern matching of abstract values.
11/20/2013 10:21 am	Sukyoung Ryu	[AbsNull] Revised the AbsNull domain.
11/20/2013 07:52 am	Sukyoung Ryu	[AbstractDomain] Made AbsBase covariantly parameterized by th of its cocrete values.
11/14/2013 07:31 pm	Sooncheol Won	[Domain] Changed the string domain.

이론적 배경

This-typed 메소드:
메소드의 인자 타입이나 결과 타입에 메소드 주인의 타입을 사용하는 경우

```
abstract class AbsBase {
  /* partial order */
  def <= (that: ThisType): Boolean

  /* join */
  def + (that: ThisType): ThisType

  /* meet */
  def <> (that: ThisType): ThisType
  ...
}
```

이론적 배경

This-typed 메소드:
메소드의 인자 타입이나 결과 타입에 메소드 주인의 타입을 사용하는 경우

- 전통적인 `This` 타입
  - 클래스를 "선언한" 타입
  - 부정확한 컴파일 시간 타입
- 우리가 제안하는 `This` 타입
  - 객체의 "실행 시간" 타입
  - 정확한 실행 시간 타입
  - 컴파일 시에 존재하지는 않지만 부를 수는 있는 타입

이론적 배경

This-typed 메소드:
메소드의 인자 타입이나 결과 타입에 메소드 주인의 타입을 사용하는 경우

- 전통적인 `This` 타입
  - 클래스를 "선언한" 타입
  - 부정확한 컴파일 시간 타입
- 우리가 제안하는 `This` 타입
  - 객체의 "실행 시간" 타입
  - 정확한 실행 시간 타입
  - 컴파일 시에 존재하지는 않지만 부를 수는 있는 타입

이론적 해결 방안

- 새로운 타입 성질
  - 클래스 C의 *정확한 클래스 타입* #C를 추가
  - This 타입 변수
  - 이름 붙인 타입 변수 </X/>를 사용하여 서로 같은 정확한 타입을 더 많이 명시 가능
  - 정확한 타입 유추를 사용하여 프로그래머의 타입 명시 부담 감소
- 새로운 언어 성질
  - 가상 생성자를 사용하여 This 타입의 값을 만들어내는 메소드 생성
  - classesmatch를 추가하여 실행 시간에 정확한 타입 검사 가능

실제적 해결 방안

ThisJava, JastAddJ를 기반한 공개 소프트웨어 구현:

`http://plrg.kaist.ac.kr/research/software`

- 기존 자바 코드와 문제없이 혼용 가능
  자바 바이트코드로 컴파일
- 실용적
  기존 자바 언어의 특이한 성질과 부드럽게 작용

자바 바이트코드로 컴파일

- 정확한 타입
  - 자바의 *type erasure*와 비슷한 부정확한 타입 만들기 적용
  - 클래스 `C`의 정의 안에 나오는 `This` 타입은 `C`로 변환
  - 클래스 `D`와 이름 붙인 타입 변수 `X`에 대해서 `#D`와 `D</X/>`는 `D`로 변환
  - ...

자바 바이트코드로 컴파일

- 정확한 타입
- 가상 생성자

```
class C {
  int fi; Point fp;
  This(int i, Point p) { fi = i; fp = p; }
  This copy() { return new This(fi, fp); }
}
```

자바 바이트코드로 컴파일

- 정확한 타입
- 가상 생성자

```
class C {
  int fi; Point fp;
  C(int i, Point p) { fi = i; fp = p; }
  C vcStub0(int i, Point p) {
    Object[] pack = new Object[] {Integer.valueOf(i), p};
    return vcStub1(pack);
  }
  C vcStub0(Object[] pack) {
    int i = ((Integer)pack[0]).intValue();
    Point p = (Point)pack[1];
    return new C(i, p);
  }
  C copy() { return vcStub0(fi, fp); }
}
```

자바 바이트코드로 컴파일

- 정확한 타입
- 가상 생성자
- 정확한 타입을 사용한 타입 테스트와 변환

```
... (o instanceof #Point) ...
... (#Point) o ...
```

```
Object o; ...
if (o instanceof This) {
  This t = (This) o; ...
}
```

```
Object o; ...
classesmatch (o, this) { ... }
```

자바 바이트코드로 컴파일

- 정확한 타입
- 가상 생성자
- 정확한 타입을 사용한 타입 테스트와 변환

```
... (o instanceof #Point) ...
... (#Point) o ...
```

```
Object o; ...
if (o instanceof This) {
  This t = (This) o; ...
}
```

```
Object o; ...
classesmatch (o, this) { ... }
```

자바 바이트코드로 컴파일

- 정확한 타입
- 가상 생성자
- 정확한 타입을 사용한 타입 테스트와 변환

```
... (o instanceof #Point) ...
... (#Point) o ...
```

```
Object o; ...
if (o instanceof This) {
  This t = (This) o; ...
}
```

```
Object o; ...
classesmatch (o, this) { ... }
```

기존 자바 언어 성질과의 관계

- 같은 이름의 여러 메소드
  - 부정확한 타입으로 만들고 이름 변환을 거친 후 호출된 메소드 검사

기존 자바 언어 성질과의 관계

- 같은 이름의 여러 메소드
- Covariant 배열 타입
  - `#C[]`, `This[][]`, 와 `C</X/>[]` 같이 invariant 배열 타입 사용

기존 자바 언어 성질과의 관계

- 같은 이름의 여러 메소드
- Covariant 배열 타입
- 타입 변수

```
class C<X extends #Point> { ...
  class I<Y extends This> { ... }
  void m(Point</E/> p) {
    class L<Z extends Point</E/>> { ... } ...
  }
}
```

기존 자바 언어 성질과의 관계

- 같은 이름의 여러 메소드
- Covariant 배열 타입
- 타입 변수

```
class C { /* #Point instead of X */ ...
  class I { /* C.This instead of Y */ ... }
  void m(Point</E/> p) {
    class L { /* Point</E/> instead of Z */ ... } ...
  }
}
```

기존 자바 언어 성질과의 관계

- 같은 이름의 여러 메소드
- Covariant 배열 타입
- 타입 변수
- 타입 검사
  - OpenJDK 1.6 소스에 있는 7637 자바 파일 중 수정이 필요한 한 가지 경우

```
public static <T> List<T> asList(T... a) { ... }
List<C> l = Arrays.asList(new C(...));
List<C> l = Arrays.asList(new C(...)); // List<#C>
List<C> l = Arrays.<C>asList(new C(...));
```

기존 자바 언어 성질과의 관계

- 같은 이름의 여러 메소드
- Covariant 배열 타입
- 타입 변수
- 타입 검사
  - OpenJDK 1.6 소스에 있는 7637 자바 파일 중 수정이 필요한 한 가지 경우

```
public static <T> List<T> asList(T... a) { ... }
List<C> l = Arrays.asList(new C(...));
List<C> l = Arrays.asList(new C(...)); // List<#C>
List<C> l = Arrays.<C>asList(new C(...));
```

기존 자바 언어 성질과의 관계

- 같은 이름의 여러 메소드
- Covariant 배열 타입
- 타입 변수
- 타입 검사
  - OpenJDK 1.6 소스에 있는 7637 자바 파일 중 수정이 필요한 한 가지 경우

```
public static <T> List<T> asList(T... a) { ... }
List<C> l = Arrays.asList(new C(...));
List<C> l = Arrays.asList(new C(...)); // List<#C>
List<C> l = Arrays.<C>asList(new C(...));
```

정리

- This-typed 메소드는 실제 상황에서 발생하는 중요한 문제
- 약간의 타입과 언어 성질을 추가해서 자바와 같은 기존 언어에 This-typed 메소드 추가 가능
- 구현 내용 공개:

  http://plrg.kaist.ac.kr/research/software

- 하지만, 자바 언어의 제약 때문에 SAFE는 ThisJava을 사용하지 않을 예정
  - ThisScala?

뒷 이야기

Jacques Garrigue의 OCaml로 흉내내기

Scala 플러그인으로 ThisScala 구현?