

Writing Secure Programs

David Wagner and Adrian Mettler, UC Berkeley. <http://www.joe-e.org>

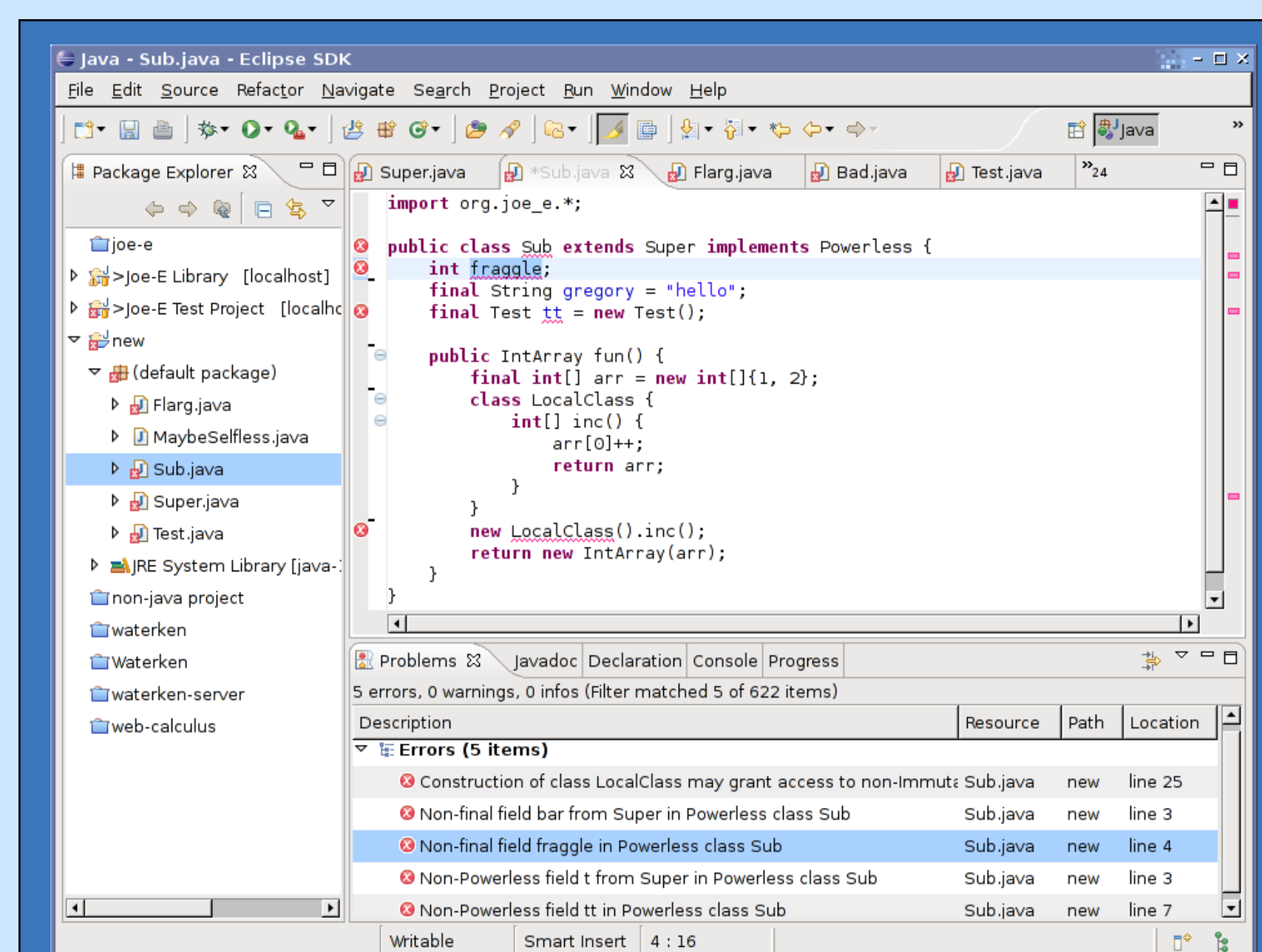


OBJECT CAPABILITIES: THE JOE-E PROJECT

Nearly every program on your computer is allowed to delete all your files, send your financial data to thieves, and spam all of your friends. The object-capability programming model allows each program (and part of a program) to be granted only the rights and information it needs to do its job.

Object capabilities allow these rights to be represented explicitly in the program as objects that are only passed where they are needed.

The Joe-E project makes the object-capability approach practical by building on the popular Java language. This leverages the tools and expertise of Java programmers to provide an easier transition to capability programming.



The Joe-E verifier implemented as a plugin for the Eclipse IDE

Approach and Impact

New approach

- Object-capabilities
- Joe-E is a subset of Java
- Leverage Java type system

Research Impact

- Open source implementation
- Used by HP Labs for projects including Waterken server for untrusted web applications

Type safety and encapsulation lay the foundation

- Memory safety makes references unforgeable
- One object can grant limited access to another one by storing it in a private field and defining methods that use it

... but are not enough

- Static methods in Java can make sensitive or side-effecting operations accessible to all Java code
- Reflection can be used to bypass encapsulation

Solution: Subset Java

- Joe-E is a subset of Java, so all Joe-E code is still Java
 - Joe-E programs work with Java tools and run on standard JVMs
- Joe-E restrictions ensure that no capabilities are made globally available everywhere in the program
 - static fields are required to be final and immutable
 - native methods are forbidden
- Java libraries are “tamed” to hide features incompatible with capability discipline and to ensure determinism
 - Prevents global access to file system, network, clock, etc.

Immutability

- Immutable types allow classes to be verified to be unmodifiable, even when they have subclasses
- Use of immutable types restricts possible side-effects and information flow in the program, simplifying reasoning
- Immutability and other semantic properties represented in the Java type system, using overlay types to avoid rewriting the Java standard libraries

Purity

- Can verify functional purity of Joe-E methods simply by inspection of method signature
- Joe-E restrictions and taming of libraries ensure that any method with only immutable arguments is pure
- Property useful for guaranteeing reproducible behavior

Joe-E Library

- Provides immutable array types
- Capability view of file system and safe reflection API