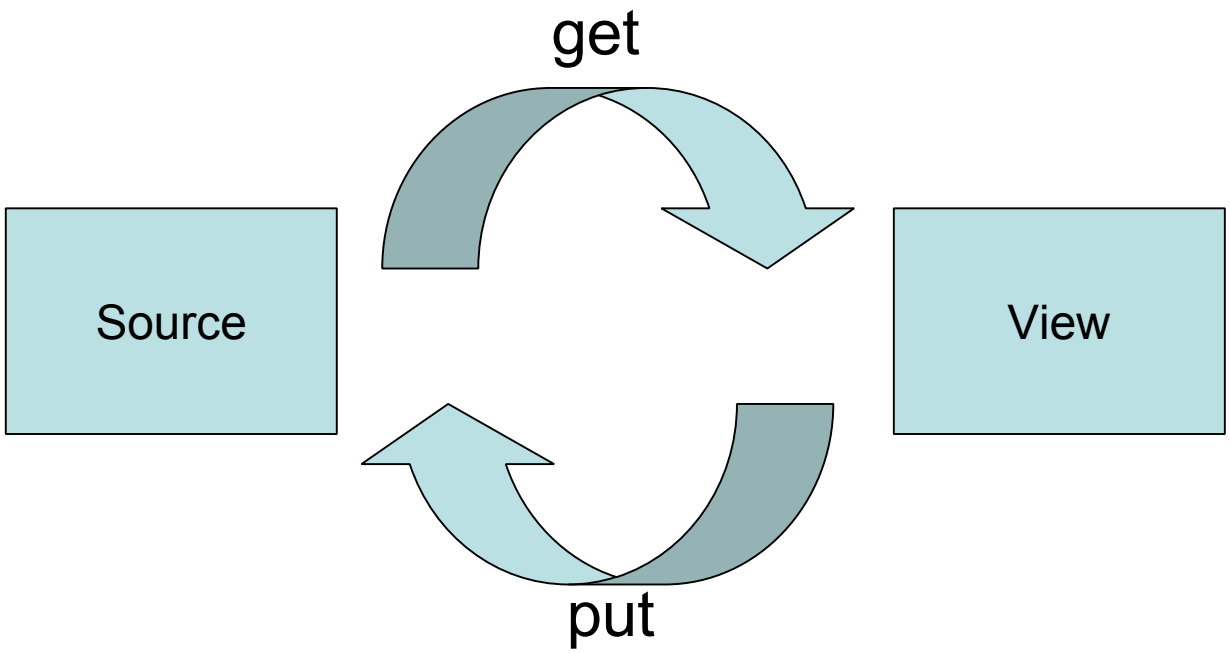
Fix Generation

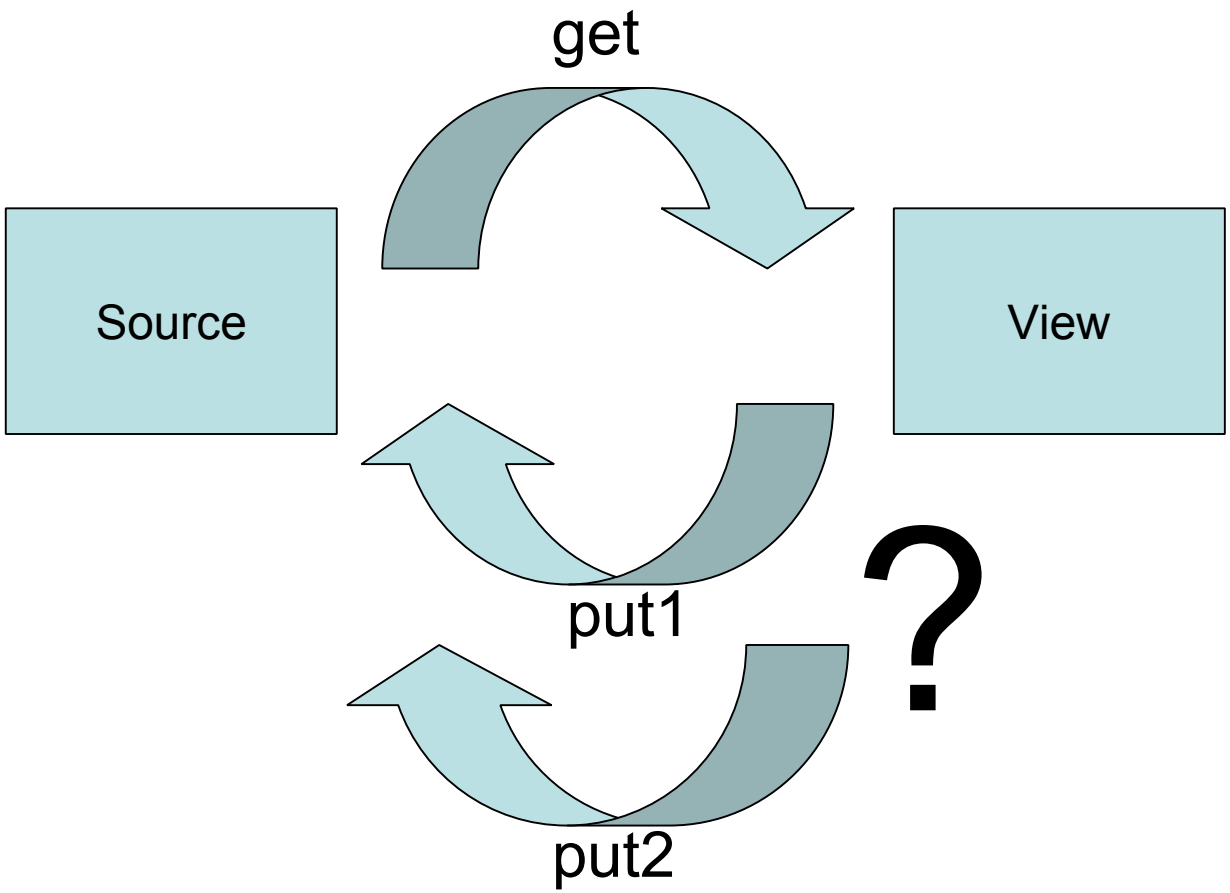
Yingfei Xiong, 2010

BX

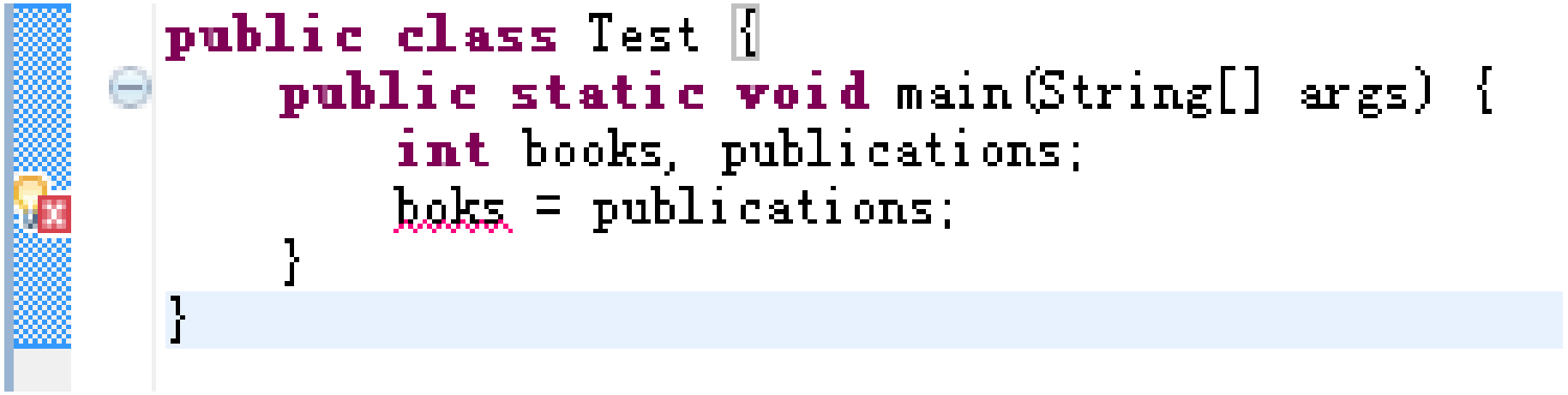


Maintain consistency between source and view automatically.

What if we need users to decide?

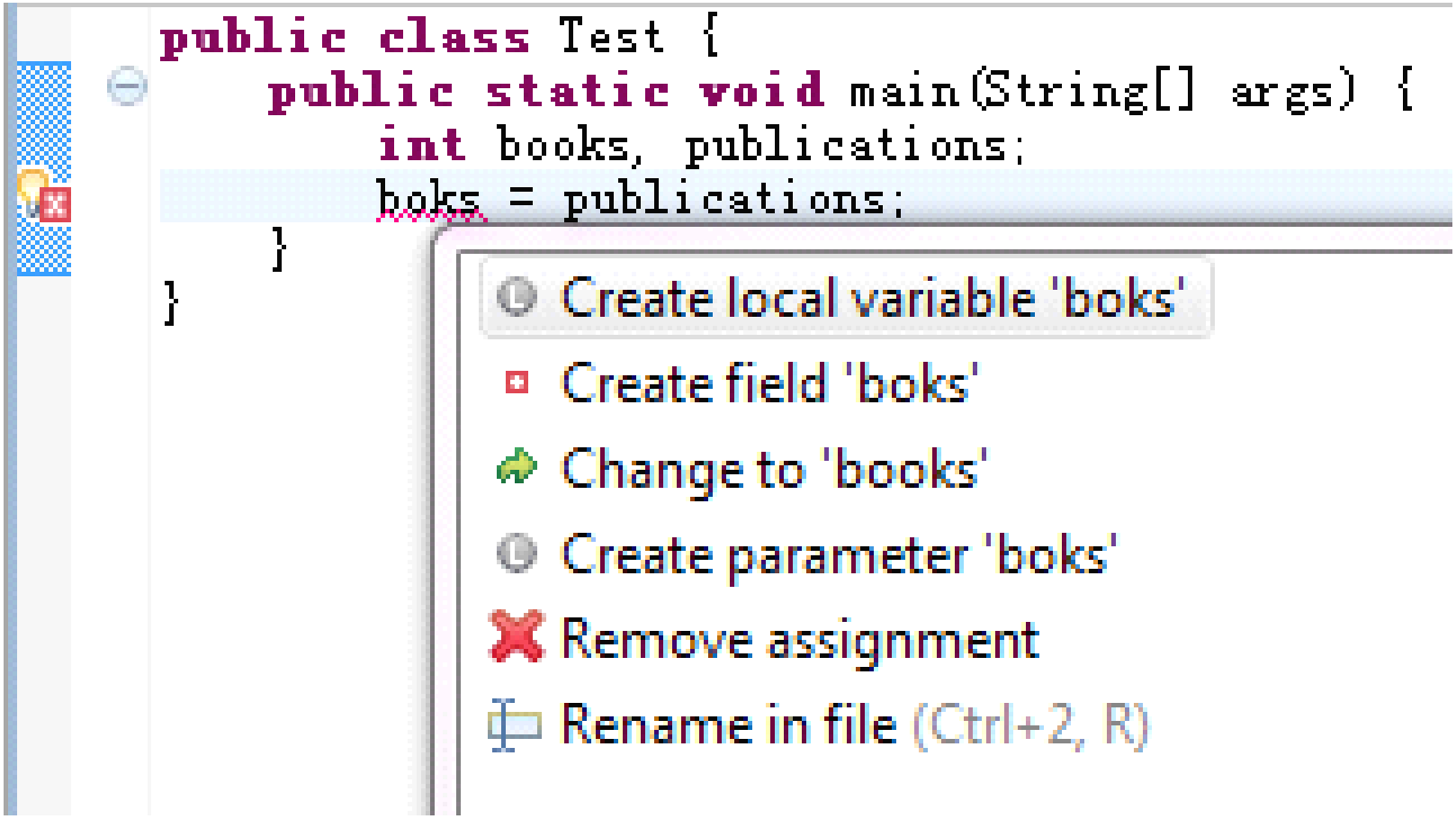


An Example



```
public class Test {
    public static void main(String[] args) {
        int books, publications;
        boks = publications;
    }
}
```

Fixes


public class Test {
public static void main(String[] args) {
int books, publications;
boks = publications;
}
}
Create local variable 'boks'
Create field 'boks'
Change to 'books'
Create parameter 'boks'
Remove assignment
Rename in file (Ctrl+2, R)

Framework for Fix Generation

- D, the set of data
- E, the set of errors
- $check : D \rightarrow 2^E$, the check function
- $generate : E \rightarrow 2^{D \rightarrow D}$, the fix generating function

Law for Fix Generation

- Let

$$e \in check(d)$$

- We have

$$\begin{aligned} &\forall f \in generate(e), \\ &\quad check(f(d)) \subseteq check(d) \ \wedge \\ &\quad e \notin check(f(d)) \end{aligned}$$

Fixes are widely used

- Eclipse: Fixes for compilation errors
- [Egyed:ASE08] Fixes for UML models
- [Dallmeier:ASE09] Fixes for object behavior anomalies
- [Chen:LISA10] Fixes for firewall policy faults
- ...

Fixes are widely used

- Eclipse: Fixes for compilation errors
- [Ehmuth:ASE08] Fixes for UML models
- [Dallmeier:ASE09] Fixes for object behavior anomalies
- [Chen:LISA10] Fixes for firewall policy faults
- ...

But little general support is provided

Open Issues

- How to describe check and generate?
  - using general programming languages
  - using domain-specific languages (like Boomerang / Beanbag / TGGs)
- How to ensure the law?
  - check the law
  - construct only well-defined *check* and *generate* (like Focal)
- Is it possible to construct *generate* automatically?
  - from consistency rules (like bidirectionalization)
  - from user operations

Bidirectional Transformation is Fix Generation with One Error

- D=S×V
- E={e}
- check (s, v) | get(s)==v = {}
  | otherwise = {e}
- generate e (s,v) = {fget, fput}
  - where
    - fget = (s, get(s))
    - fput = (put(s,v), v)

Conclusion