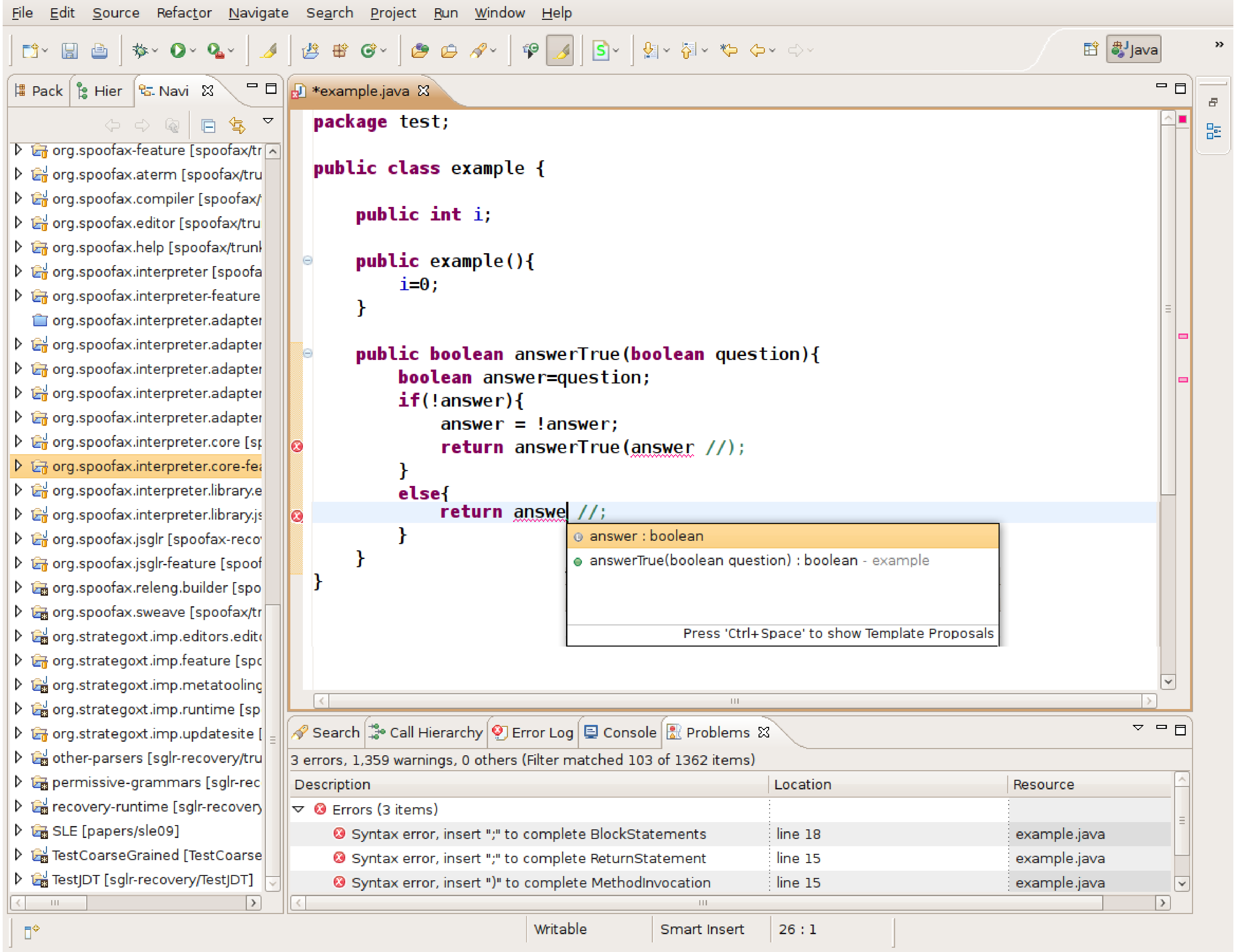
Natural and Flexible Error Recovery for Generated Parsers

Maartje de Jonge
Emma Nilsson-Nyman
Lennart Kats
Eelco Visser


File
Edit
Source
Refactor
Navigate
Search
Project
Run
Window
Help
Java
Pack
Hier
Navi
*example.java
org.spoofax-feature [spoofax/tr
org.spoofax.aterm [spoofax/tru
org.spoofax.compiler [spoofax/
org.spoofax.editor [spoofax/tru
org.spoofax.help [spoofax/trunk
org.spoofax.interpreter [spoofa
org.spoofax.interpreter-feature
org.spoofax.interpreter.adapter
org.spoofax.interpreter.adapter
org.spoofax.interpreter.adapter
org.spoofax.interpreter.adapter
org.spoofax.interpreter.adapter
org.spoofax.interpreter.core [sp
org.spoofax.interpreter.core-fe
org.spoofax.interpreter.library.e
org.spoofax.interpreter.library.js
org.spoofax.jsglr [spoofax-reco
org.spoofax.jsglr-feature [spoof
org.spoofax.releng.builder [spo
org.spoofax.sweave [spoofax/tr
org.strategoxt.imp.editors.edit
org.strategoxt.imp.feature [spo
org.strategoxt.imp.metatooling
org.strategoxt.imp.runtime [sp
org.strategoxt.imp.updatesite [
other-parsers [sglr-recovery/tru
permissive-grammars [sglr-rec
recovery-runtime [sglr-recovery
SLE [papers/sle09]
TestCoarseGrained [TestCoarse
TestJDT [sglr-recovery/TestJDT]
Search
Call Hierarchy
Error Log
Console
Problems
Writable
Smart Insert
26 : 1

```
package test;

public class example {

    public int i;

    public example(){
        i=0;
    }

    public boolean answerTrue(boolean question){
        boolean answer=question;
        if(!answer){
            answer = !answer;
            return answerTrue(answer //);
        }
        else{
            return answe //;
        }
    }
}
```

answer : boolean

answerTrue(boolean question) : boolean - example

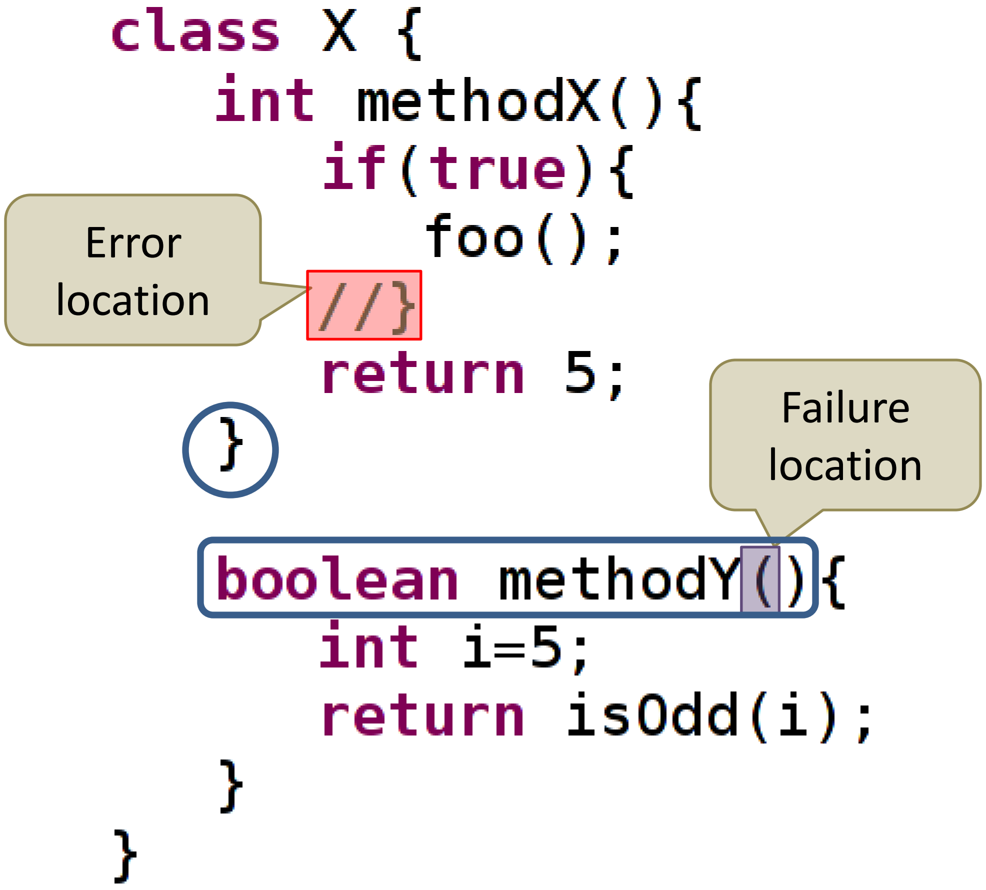
Press 'Ctrl+Space' to show Template Proposals

3 errors, 1,359 warnings, 0 others (Filter matched 103 of 1362 items)

Description	Location	Resource
Errors (3 items)		
Syntax error, insert ";" to complete BlockStatements	line 18	example.java
Syntax error, insert ";" to complete ReturnStatement	line 15	example.java
Syntax error, insert ")" to complete MethodInvocation	line 15	example.java

Error Recovery

- Error Location
- Failure Location



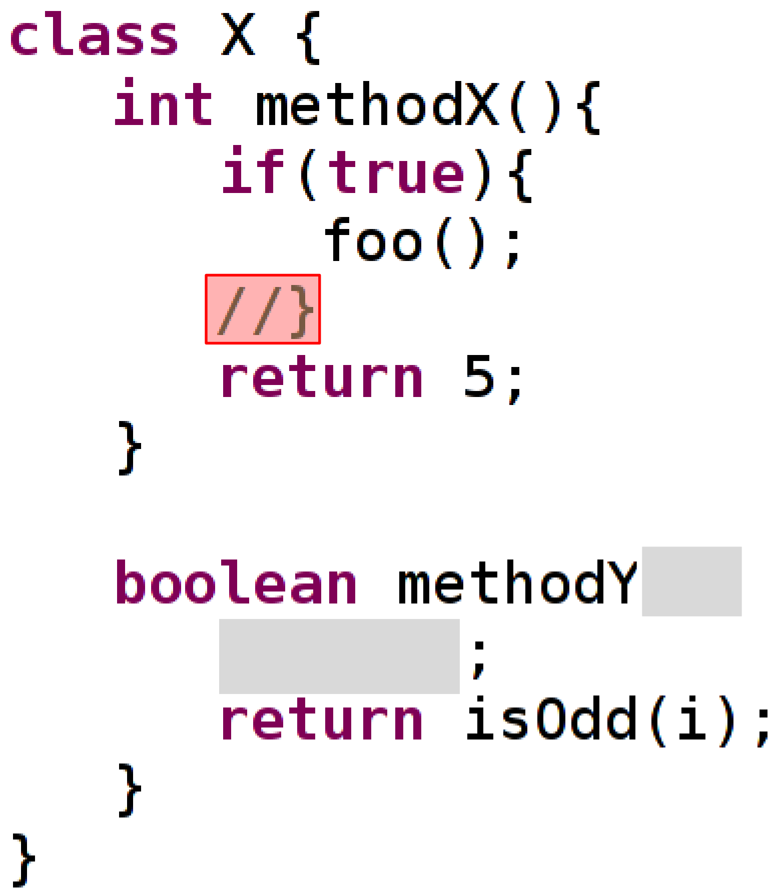
```
class X {
   int methodX(){
      if(true){
         foo();
      //}
      return 5;
   }

   boolean methodY(){
      int i=5;
      return isOdd(i);
   }
}
```

Error location

Failure location

Error Recovery

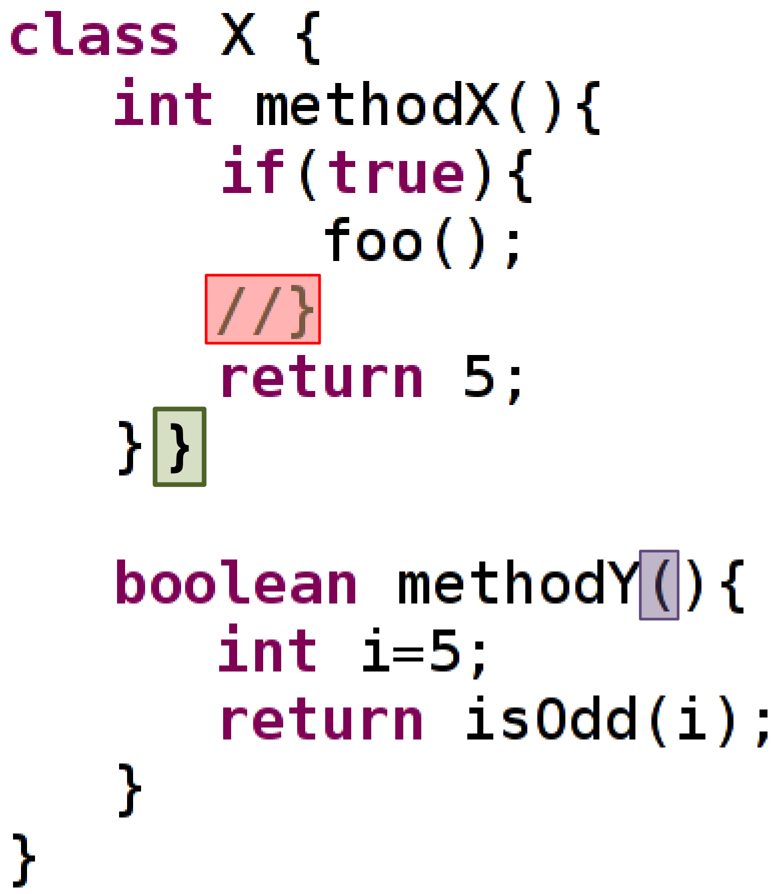
- Traditional approaches
  - Panic mode



```
class X {
    int methodX(){
        if(true){
            foo();
        //}
        return 5;
    }

    boolean methodY
                 ;
        return isOdd(i);
    }
}
```

Error Recovery

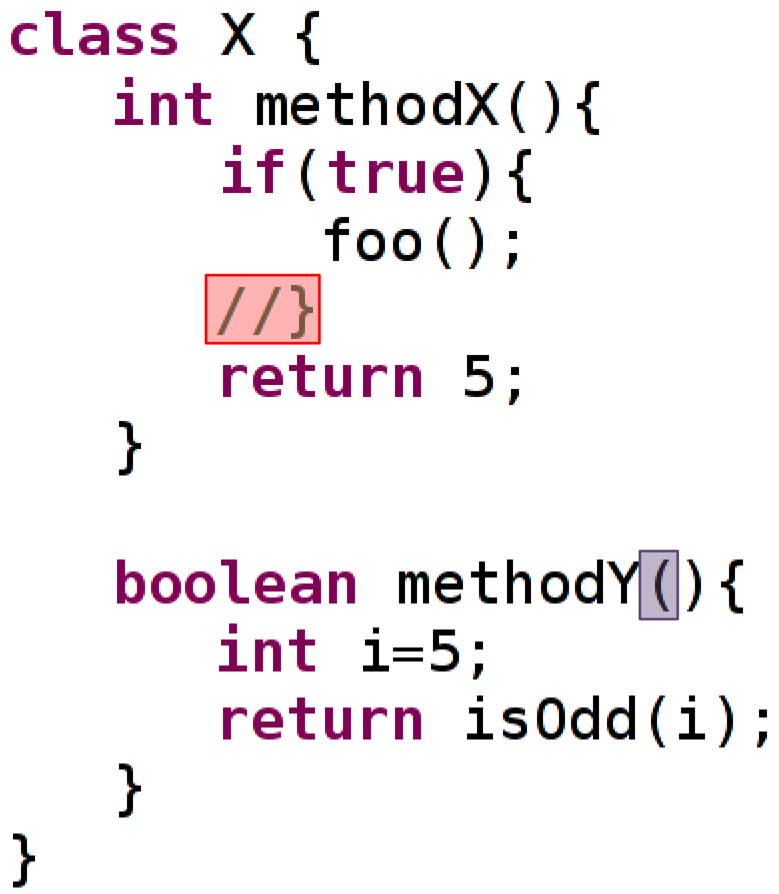
- Traditional approaches
  - Panic mode
  - Delete / insert tokens



```
class X {
   int methodX(){
      if(true){
         foo();
      //}
      return 5;
   }}

   boolean methodY(){
      int i=5;
      return isOdd(i);
   }
}
```

Error Recovery

- Traditional approaches
  - Panic mode
  - Delete / insert tokens
  - Recover productions

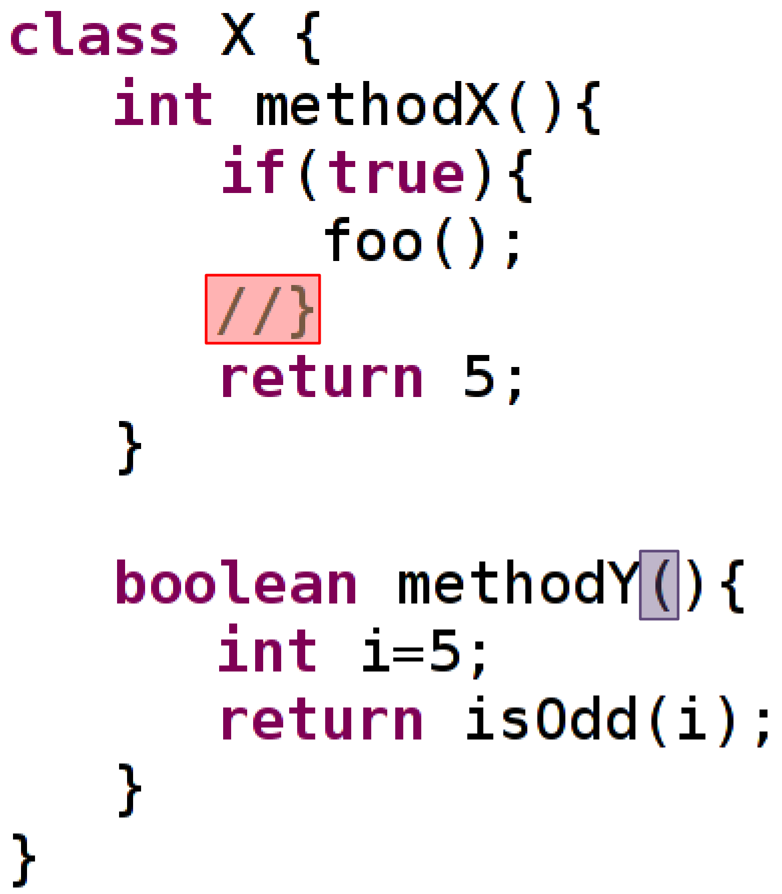


```
class X {
   int methodX(){
      if(true){
         foo();
      //}
      return 5;
   }

   boolean methodY(){
      int i=5;
      return isOdd(i);
   }
}
```

Error Recovery

- Traditional approaches
  - Panic mode
  - Delete / insert tokens
  - Recover productions

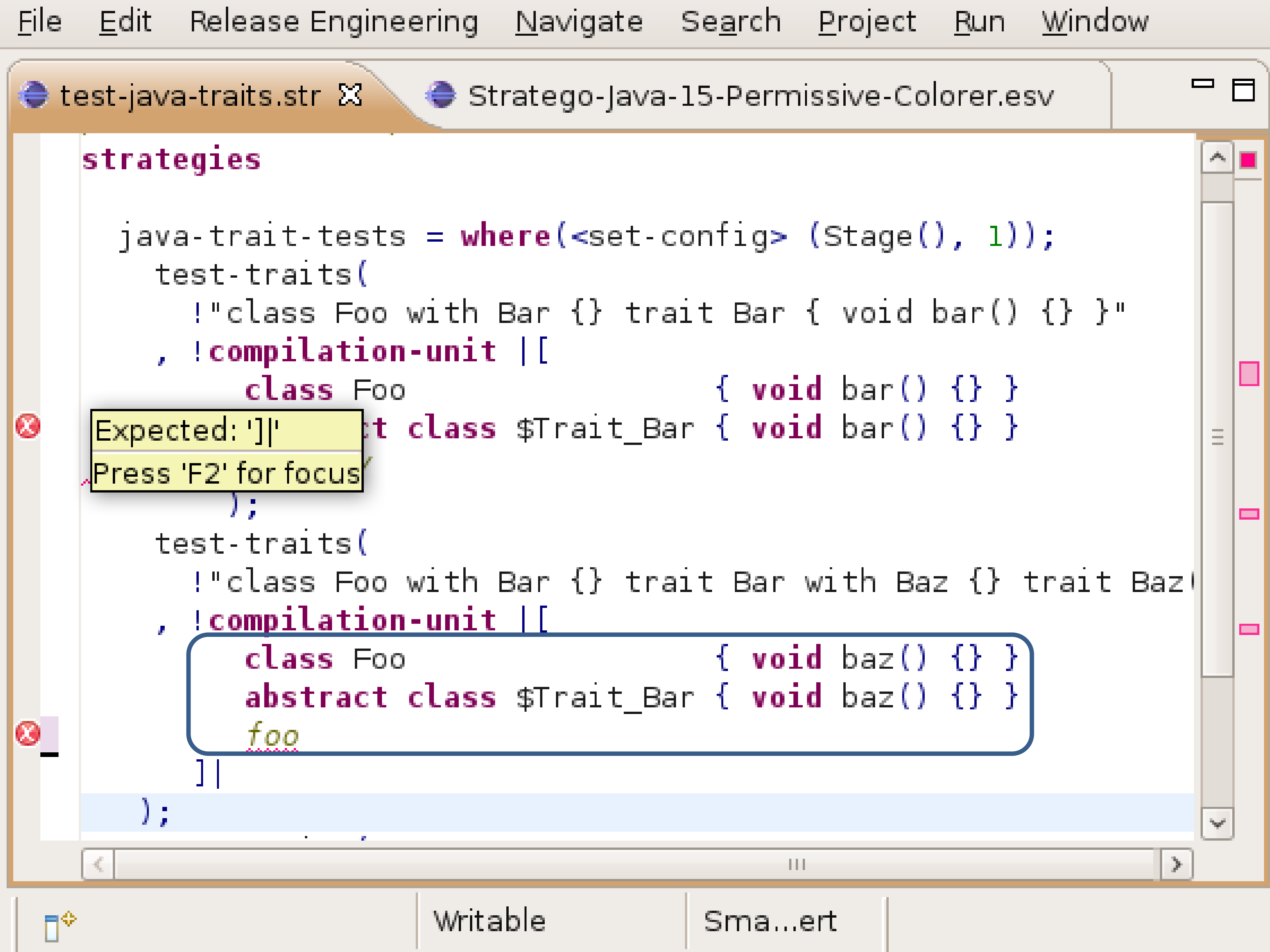
- Issues
  - Poor quality
  - Language dependency



```
class X {
    int methodX(){
        if(true){
            foo();
        //}
        return 5;
    }

    boolean methodY(){
        int i=5;
        return isOdd(i);
    }
}
```

Error Recovery

- Requirements
  - High quality
  - Language independent
  - SGLR



```
strategies

  java-trait-tests = where(<set-config> (Stage(), 1));
    test-traits(
      !"class Foo with Bar {} trait Bar { void bar() {} }"
    , !compilation-unit |[
          class Foo                 { void bar() {} }
            t class $Trait_Bar { void bar() {} }


        );
    test-traits(
      !"class Foo with Bar {} trait Bar with Baz {} trait Baz
    , !compilation-unit |[
          class Foo                 { void baz() {} }
          abstract class $Trait_Bar { void baz() {} }
          foo
      ]|
  );
```

Expected: ']|'

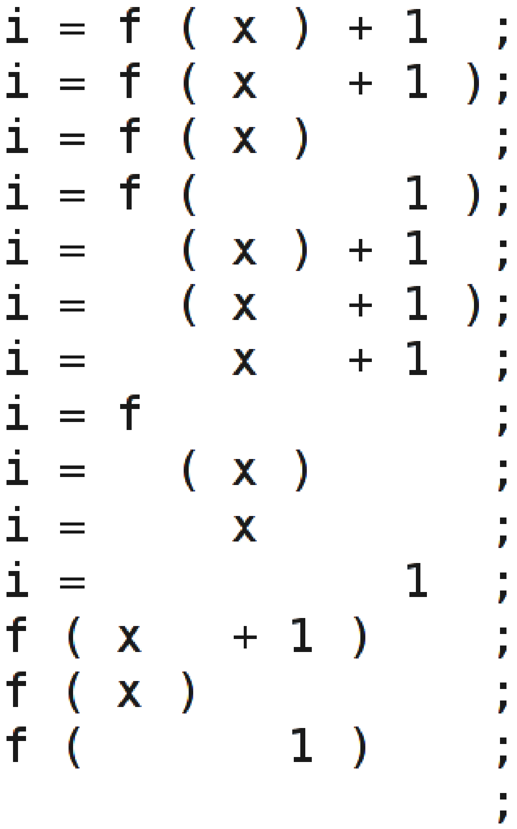
Press 'F2' for focus

Fine-Grained Repair

- Error recovery for SGLR (OOPSLA 2009)
  - Extend grammar with recover productions
    - Insert special characters
    - Delete special characters and words
  - Derive recover rules from grammar
  - Adapt parse algorithm to parse recover options

Fine-Grained Repair

- Recover productions introduce ambiguities
- Ambiguities create a search space of alternate parses
- Problem: find the best parse alternative



```
i = f ( x ) + 1  ;
i = f ( x   + 1 );
i = f ( x )      ;
i = f (       1 );
i =   ( x ) + 1  ;
i =   ( x   + 1 );
i =     x   + 1  ;
i = f            ;
i =   ( x )      ;
i =     x        ;
i =           1  ;
f ( x   + 1 )    ;
f ( x )          ;
f (       1 )    ;
                 ;
```

Figure: Alternate interpretations of “ i = f (x + 1 ;”

Fine-Grained Repair

- Parallel Parsing
  - Bad performance if applied on large regions
- Backtracking
  - Good performance in regular cases
  - Bad performance in worst-case scenarios

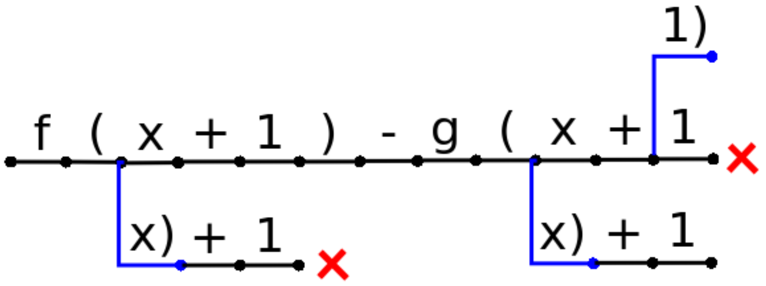
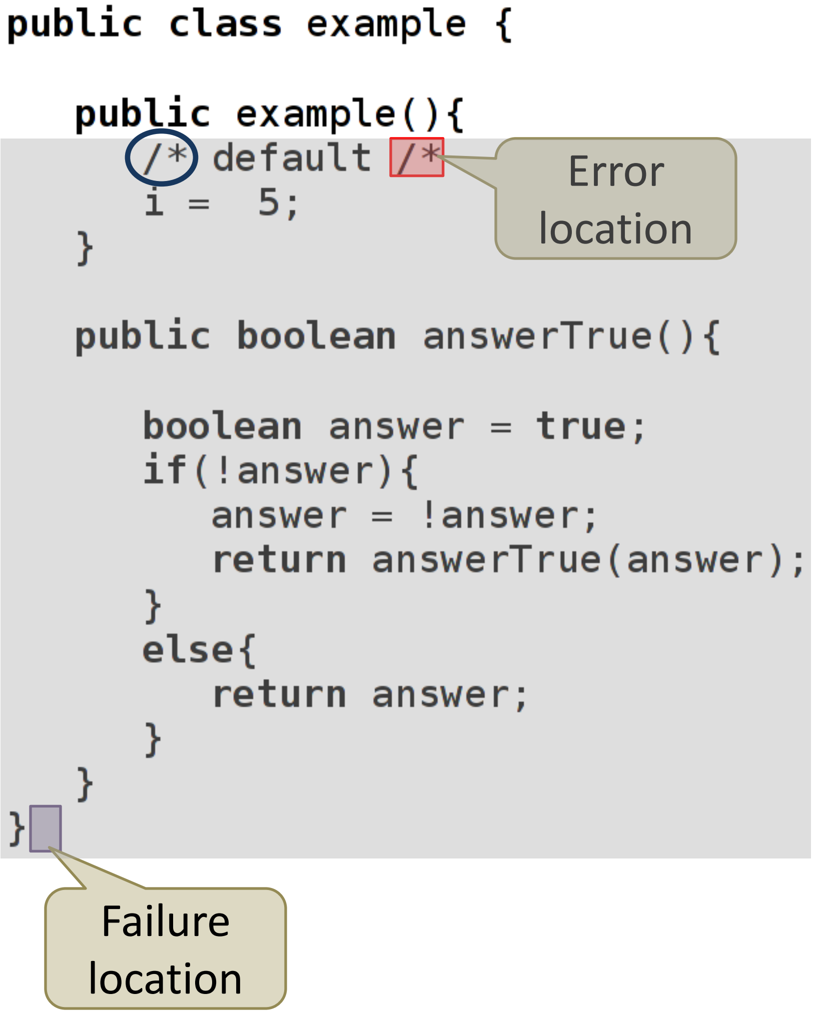


Figure: Search space for recover rule: insert ‘)’

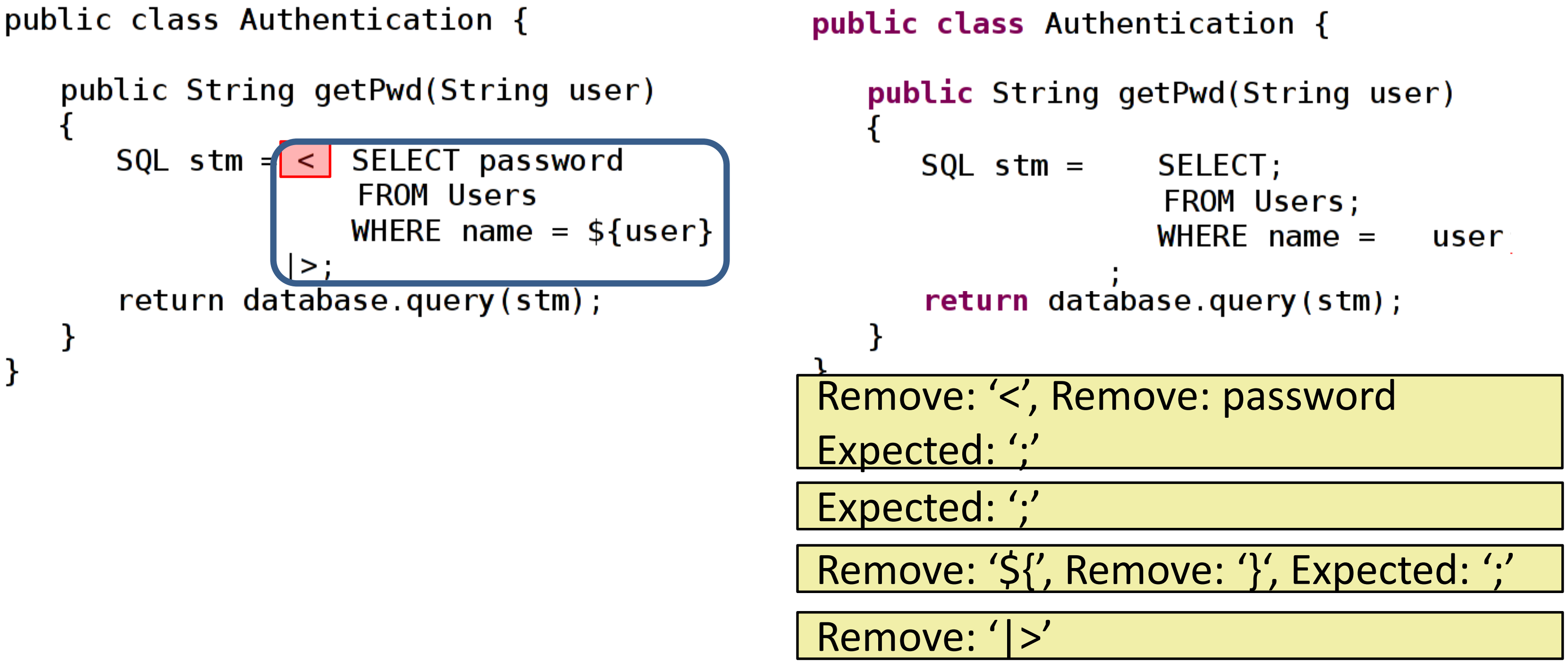


```
public class example {

    public example(){
        /* default /*
        i =  5;
    }

    public boolean answerTrue(){

        boolean answer = true;
        if(!answer){
            answer = !answer;
            return answerTrue(answer);
        }
        else{
            return answer;
        }
    }
}
```

Error location

Failure location

Figure: Backtracking over a large region



```
public class Authentication {

    public String getPwd(String user)
    {
        SQL stm = < SELECT password
                     FROM Users
                    WHERE name = ${user}
                 |>;
        return database.query(stm);
    }
}
```

```
public class Authentication {

    public String getPwd(String user)
    {
        SQL stm =     SELECT;
                       FROM Users;
                      WHERE name =    user
                    ;
        return database.query(stm);
    }
}
```

Remove: ‘<’, Remove: password
Expected: ‘;’

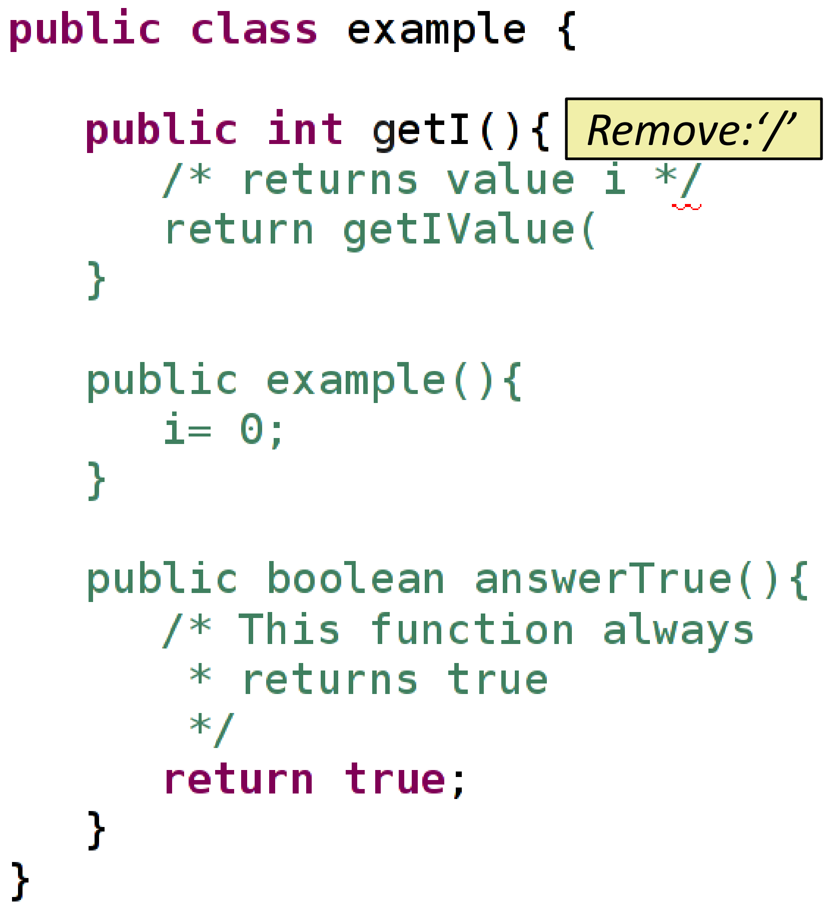
Expected: ‘;’

Remove: ‘${’, Remove: ‘}’, Expected: ‘;’

Remove: ‘|>’

Figure: Parsing SQL as Java

```
public class example {

    public int getI(){
        /* returns value i */
        return getIValue(□
    }

    public example(){
        i= 0;
    }

    public boolean answerTrue(){
        /* This function always
         * returns true
         */
        return true;
    }
}
```



```
public class example {

    public int getI(){
        /* returns value i */
        return getIValue(
    }

    public example(){
        i= 0;
    }

    public boolean answerTrue(){
        /* This function always
         * returns true
         */
        return true;
    }
}
```

Remove:'/'

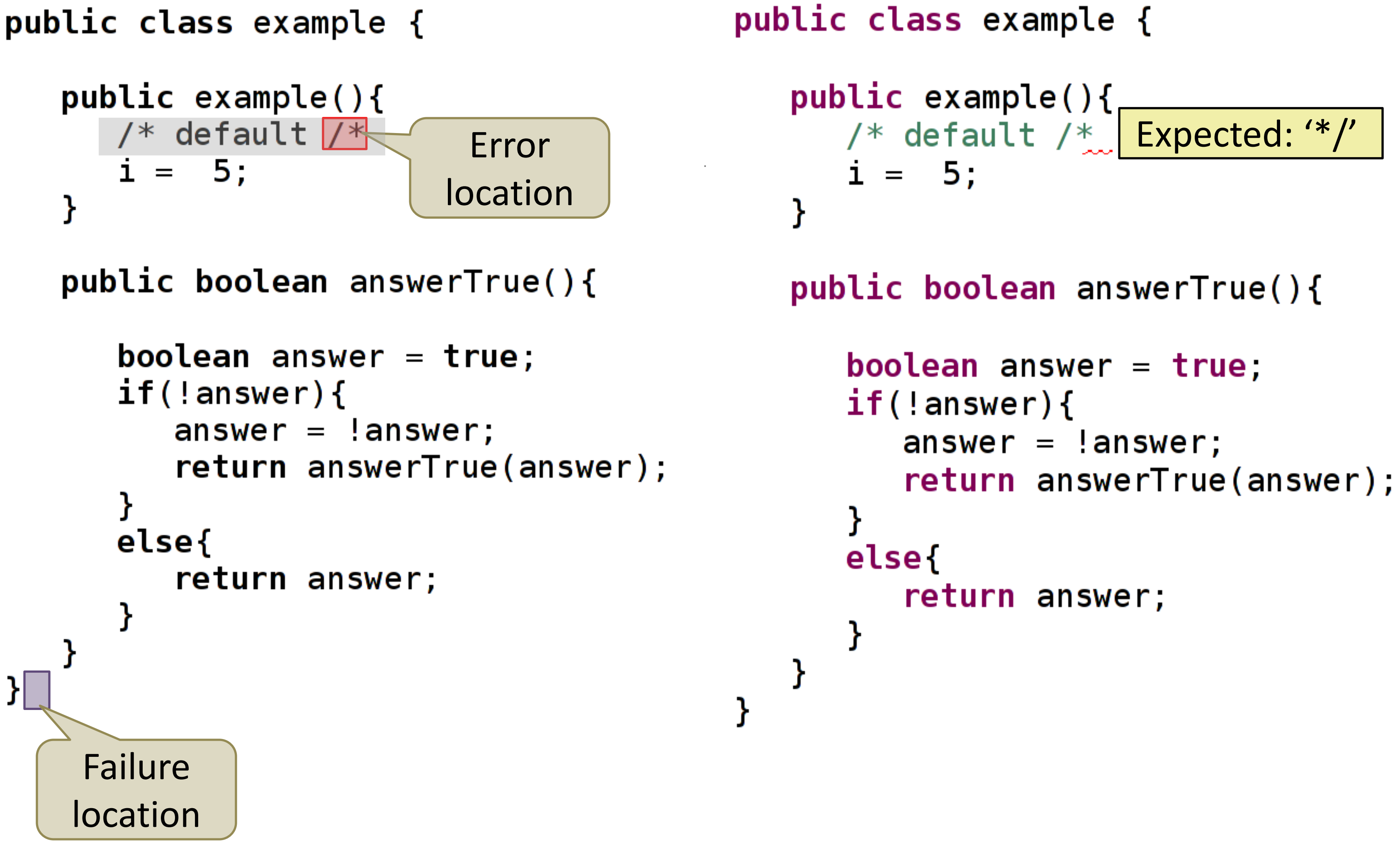
Figure: Clever but unnatural recovery

Problems with Fine-Grained Recovery

- Performance problems
  - Large area of text is inspected
  - Many recover actions are required
- Quality problems
  - ‘Clever’ solutions

Solution in SLE Paper

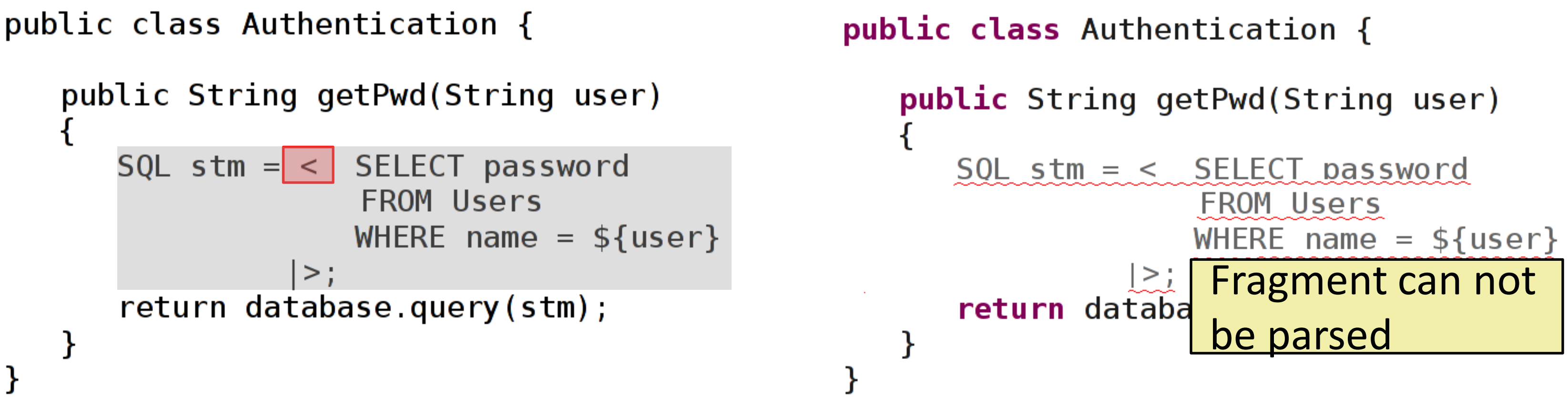
- Technique for selecting erroneous region
  - Restricts area of text that is inspected
  - Fallback recovery: skip erroneous region



```
public class example {

   public example(){
      /* default /*
      i =  5;
   }

   public boolean answerTrue(){

      boolean answer = true;
      if(!answer){
         answer = !answer;
         return answerTrue(answer);
      }
      else{
         return answer;
      }
   }
}
```

```
public class example {

   public example(){
      /* default /*
      i =  5;
   }

   public boolean answerTrue(){

      boolean answer = true;
      if(!answer){
         answer = !answer;
         return answerTrue(answer);
      }
      else{
         return answer;
      }
   }
}
```

Figure: Backtracking on a small region improves performance

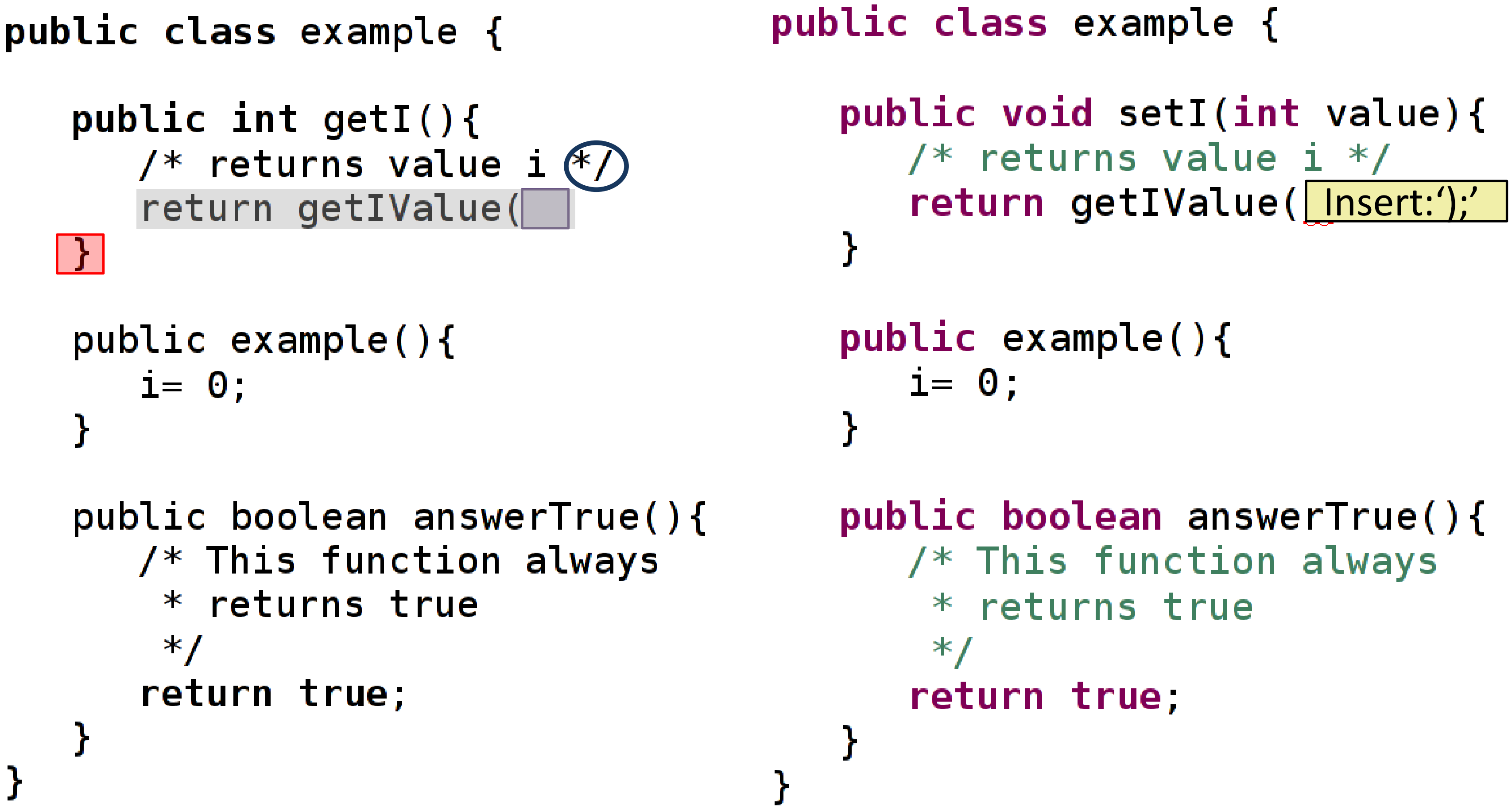


```
public class Authentication {

    public String getPwd(String user)
    {
        SQL stm = <  SELECT password
                      FROM Users
                     WHERE name = ${user}
                 |>;
        return database.query(stm);
    }
}
```

```
public class Authentication {

    public String getPwd(String user)
    {
        SQL stm = <  SELECT password
                      FROM Users
                     WHERE name = ${user}
                 |>;
        return databa
    }
}
```

Fragment can not be parsed

Figure: Fallback recovery solves problematic errors



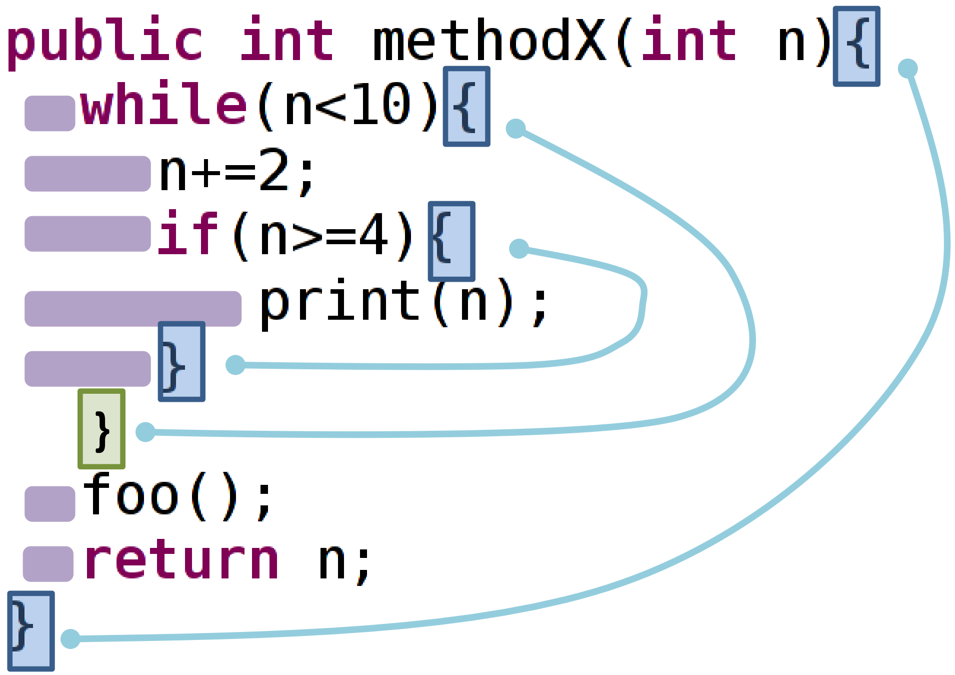
```
public class example {

   public int getI(){
      /* returns value i */
      return getIValue(
   }

   public example(){
      i= 0;
   }

   public boolean answerTrue(){
      /* This function always
       * returns true
       */
      return true;
   }
}
```

```
public class example {

   public void setI(int value){
      /* returns value i */
      return getIValue(
   }

   public example(){
      i= 0;
   }

   public boolean answerTrue(){
      /* This function always
       * returns true
       */
      return true;
   }
}
```

Figure: Restricting backtracking to erroneous region avoids unnatural recoveries

How to select the erroneous region?

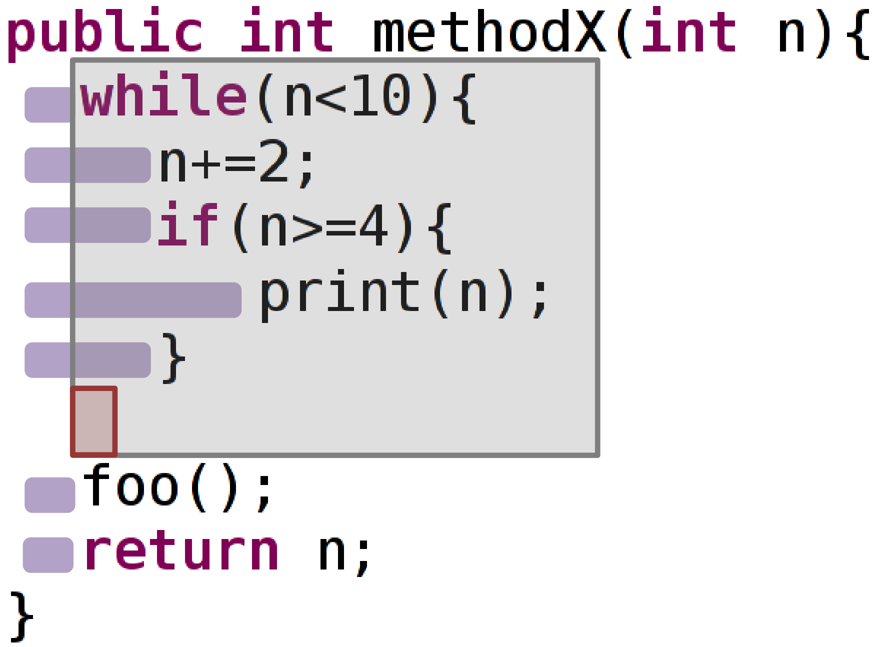
Bridge Parsing



```
public int methodX(int n){
  while(n<10){
     n+=2;
     if(n>=4){
        print(n);
     }
  }
  foo();
  return n;
}
```

Figure: Scope recovery by indentation

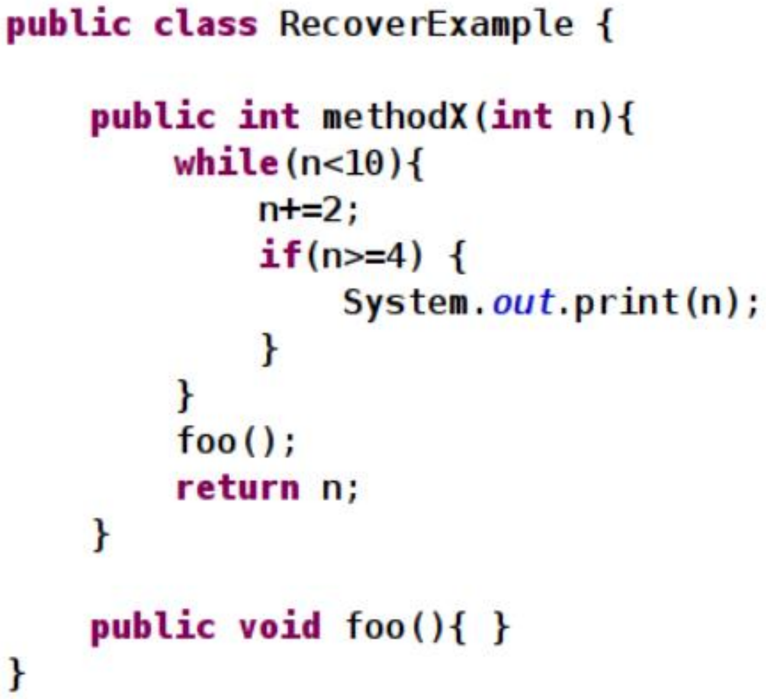
Idea



```
public int methodX(int n){
    while(n<10){
        n+=2;
        if(n>=4){
            print(n);
        }

    foo();
    return n;
}
```

Figure: Region selection by indentation

Idea



```java
public class RecoverExample {

    public int methodX(int n){
        while(n<10){
            n+=2;
            if(n>=4) {
                System.out.print(n);
            }
        }
        foo();
        return n;
    }

    public void foo(){ }
}
```

Figure: Regions are independent blocks

Idea

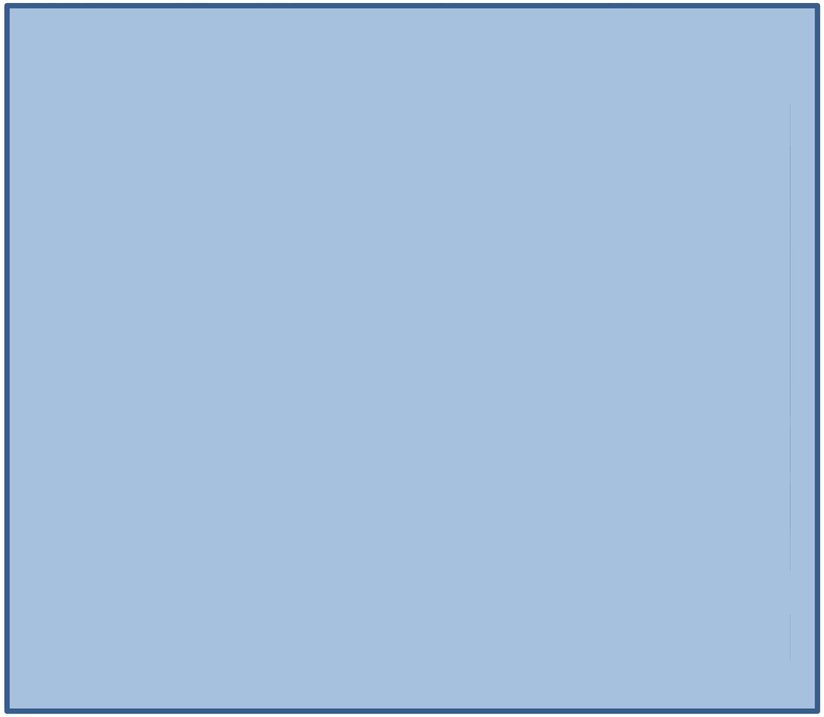


Figure: Regions are independent blocks

Idea

- Issues
  - Assumption on use of indentation
  - Assumption on structure of language

Region Selection

- Select a candidate region
- Check if the candidate contains the error
- Repeat till the erroneous region is found

Region Selection

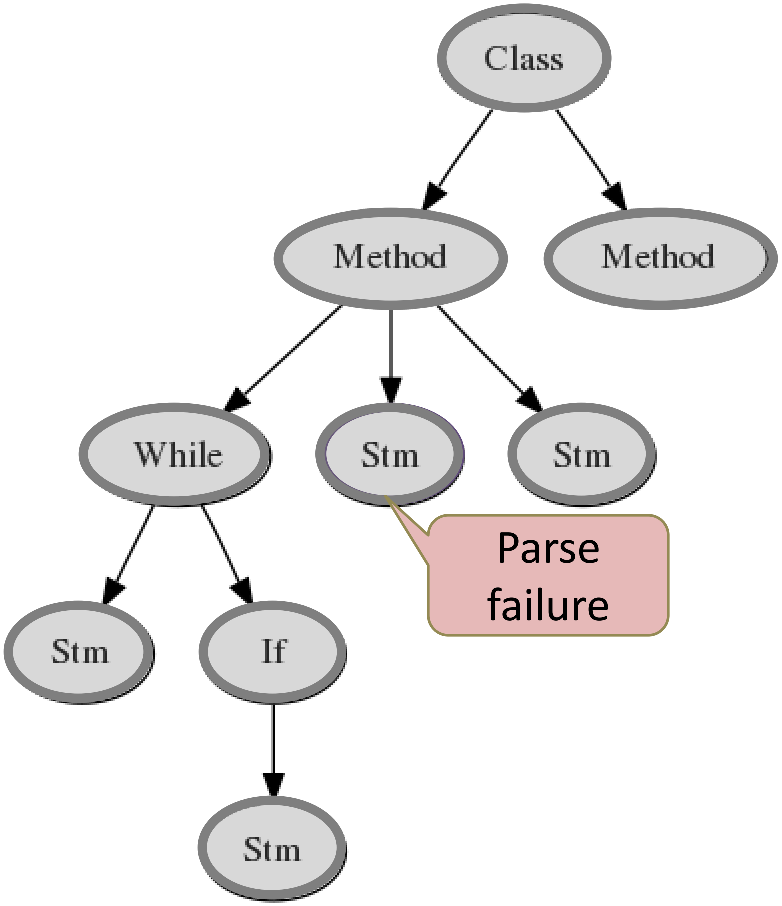
- Parser fails because of unexpected token
- Select current region
- Reset parser to prior position
- Skip the selected region and resume parsing
- Parsing continues, so the erroneous region is detected

```
class X {
    int i;
```

```
    void methodY(){
        bar();
    }
}
```

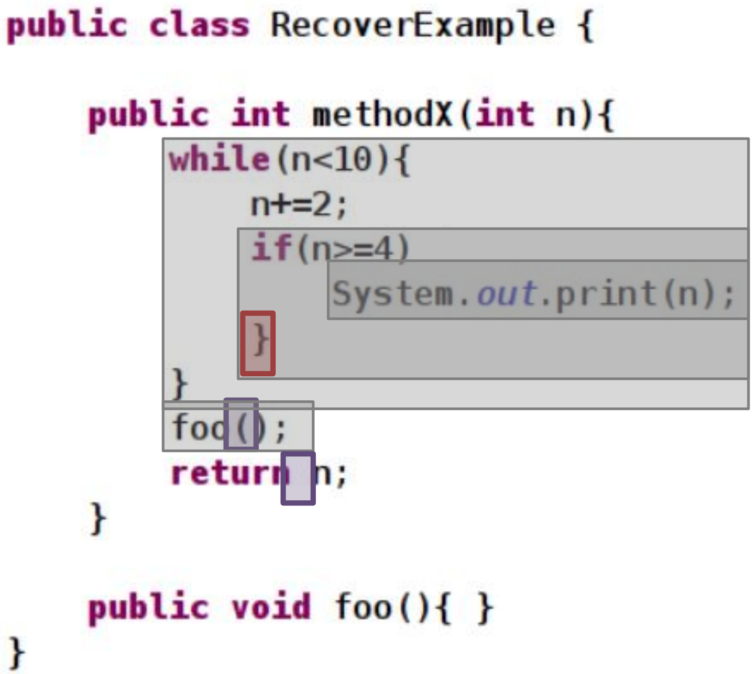
Region Selection

- Current
- Previous
  - Child regions
- Siblings
- Parent
- Grand parent
- …

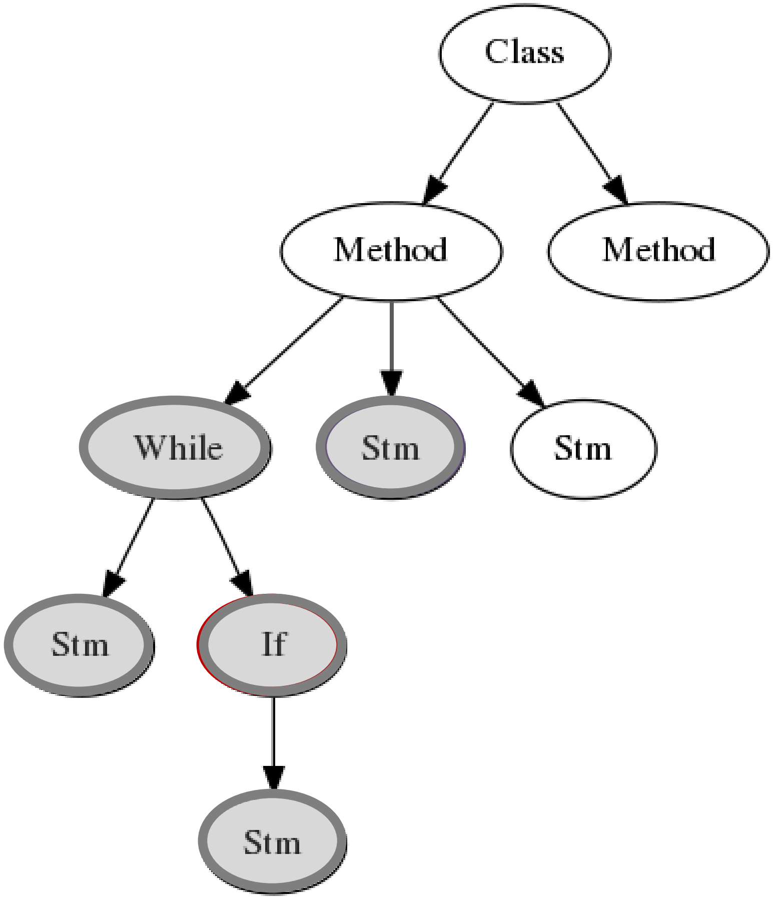


Parse failure

Region Selection



```
public class RecoverExample {

    public int methodX(int n){
        while(n<10){
            n+=2;
            if(n>=4)
                System.out.print(n);
            }
        }
        foo();
        return n;
    }

    public void foo(){ }
}
```



Final Solution

- Select erroneous region
- Try Bridge Parsing
- Try Fine Grained Repair
- Skip region

Evaluation

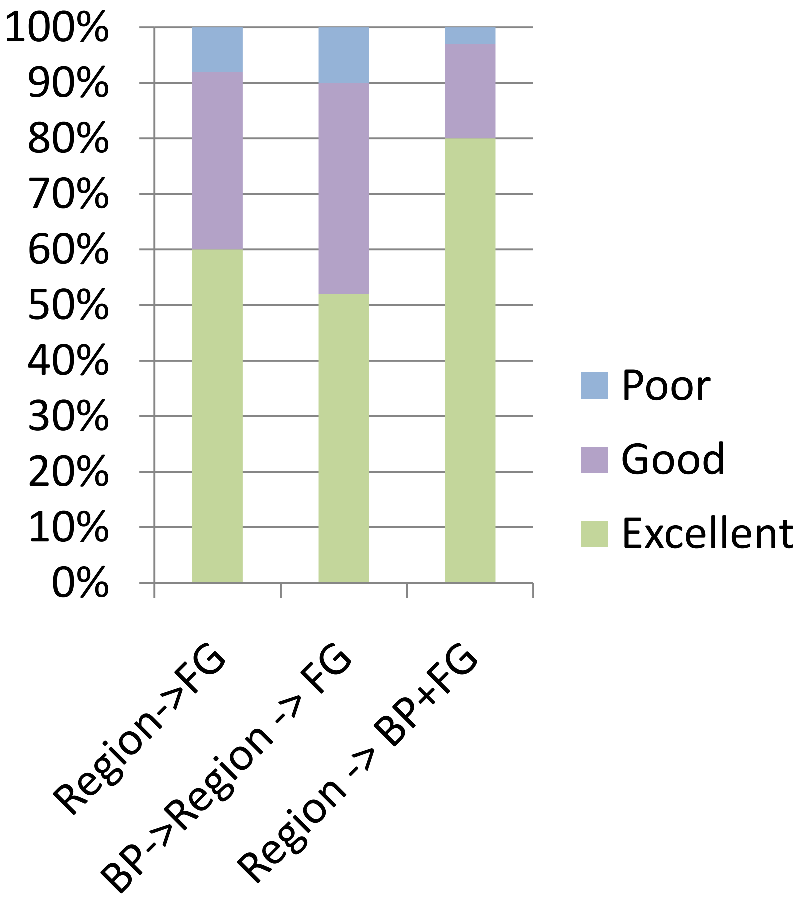
- Testset
  - Missing tokens (65 tests)
  - Wrongly inserted tokens (8 tests)
  - Others (3 tests)

Evaluation

- Criteria
  - Excellent: Same as recovery by a human being
  - Good: Reasonable recovery without spurious errors
  - Poor: Poor recovery creating spurious errors

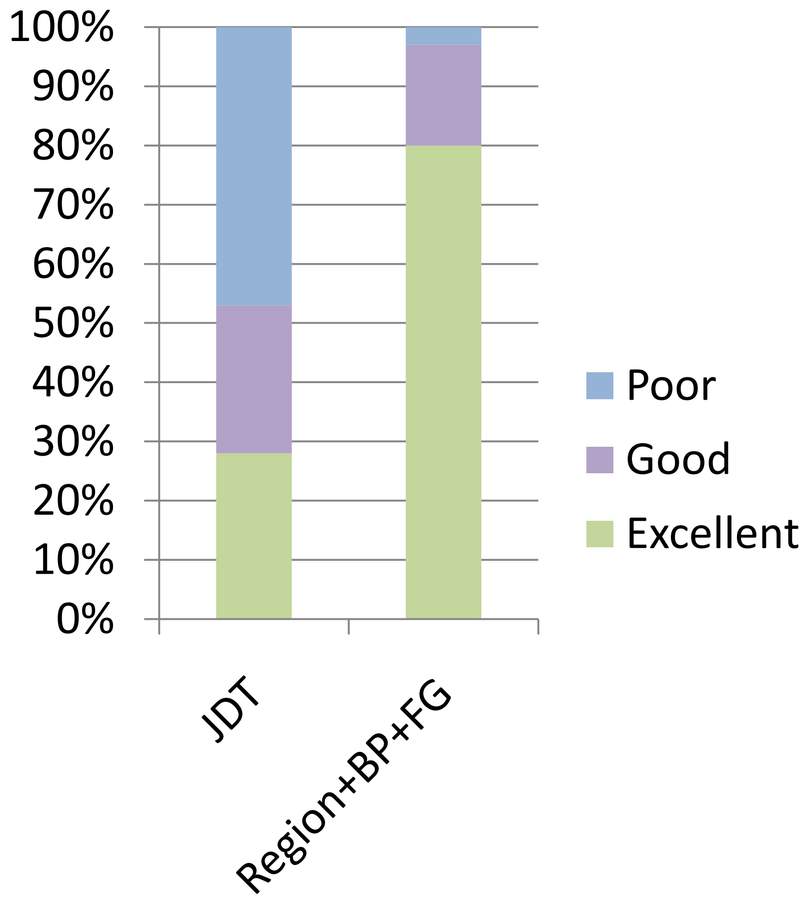
Evaluation

- Contribution of techniques
  - Region -> Fine Grained
  - Bridge Parsing -> Region -> Fine Grained
  - Region -> Bridge Parsing + Fine Grained



Evaluation

- Comparison with JDT
  - JDT
  - Region -> Bridge Parsing + Fine-Grained



Evaluation

- Language User
  - Quality
  - Performance
- Language Developer
  - Language independent
  - Flexible
  - Transparent

Summary

- Region Selection
  - Selects erroneous region by using indentation
  - Used as a preprocessor for a correcting technique, or as fallback recovery
  - Can be implemented for all parsing algorithms
- Bridge Parsing
  - Scope recovery based on indentation
  - Works for all parsing algorithms
- Fine-Grained Repair
  - Inserting and deleting special tokens
  - Extends grammar with recover productions
  - Requires (S)GLR parsing

More Information

Permissive Grammars Project:

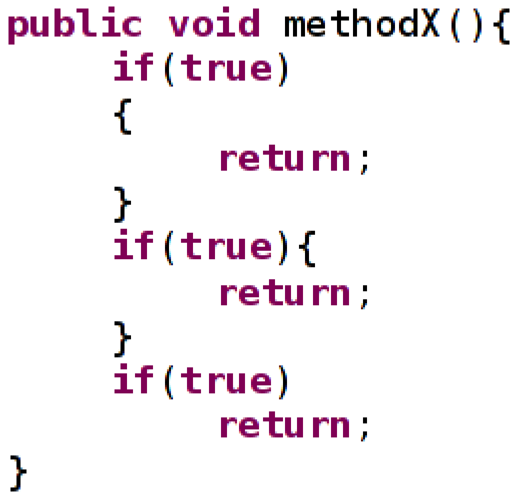
strategoxt.org/Stratego/PermissiveGrammars

Email & Homepage:

m.dejonge@tudelft.nl

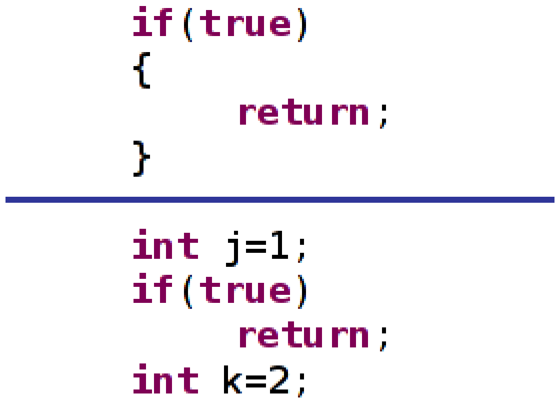
swerl.tudelft.nl/bin/view/Main/MaartjeDeJonge

Braces



```
public void methodX(){
    if(true)
    {
        return;
    }
    if(true){
        return;
    }
    if(true)
        return;
}
```

Figure: Different notations for braces

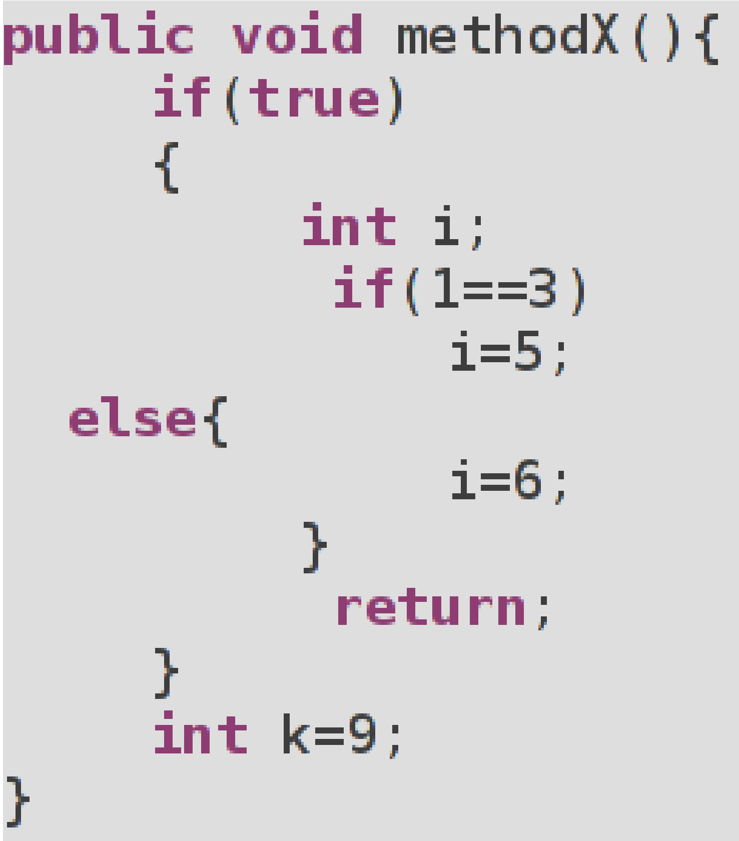


```
if(true)
{
    return;
}
```

```
int j=1;
if(true)
    return;
int k=2;
```

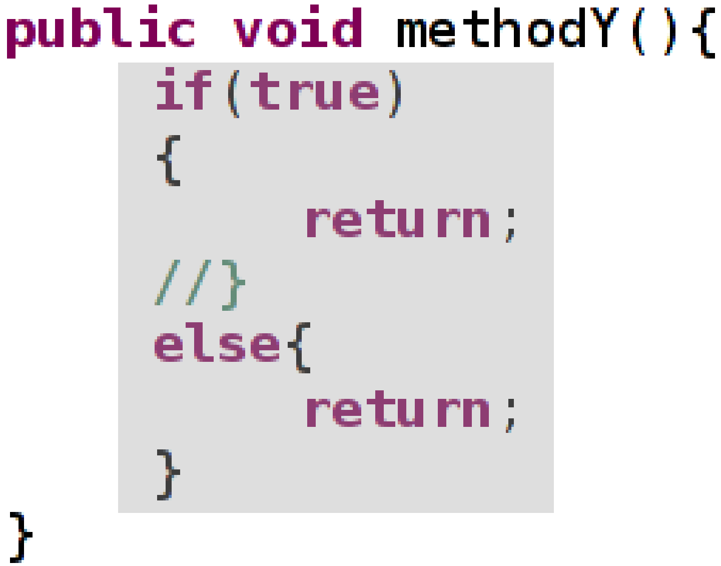
Figure: Same indentation pattern, different regions

Robustness

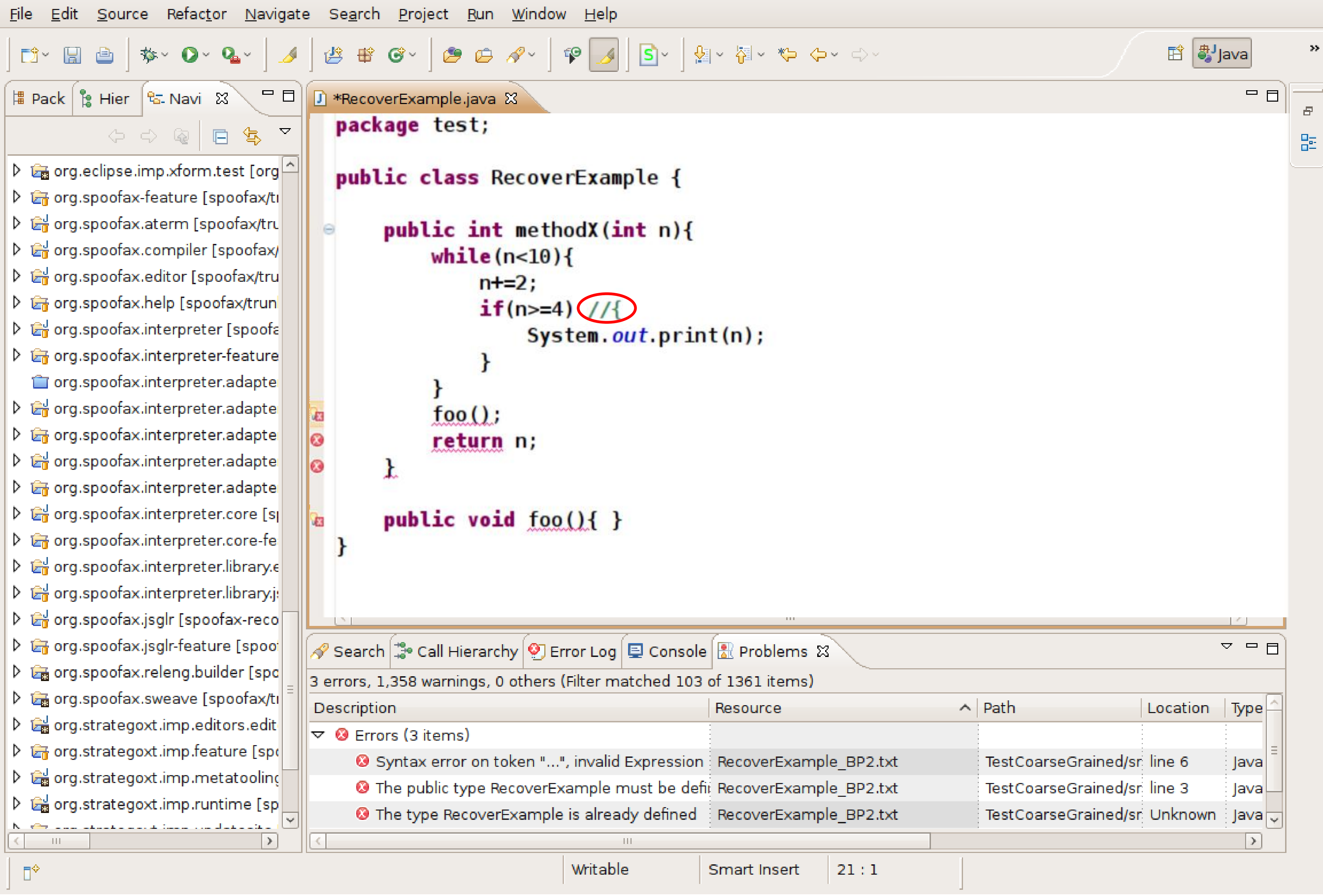


```
public void methodX(){
     if(true)
     {
         int i;
          if(1==3)
              i=5;
  else{
              i=6;
         }
          return;
     }
     int k=9;
}
```

Dependent blocks



```
public void methodY(){
    if(true)
    {
        return;
    //}
    else{
        return;
    }
}
```

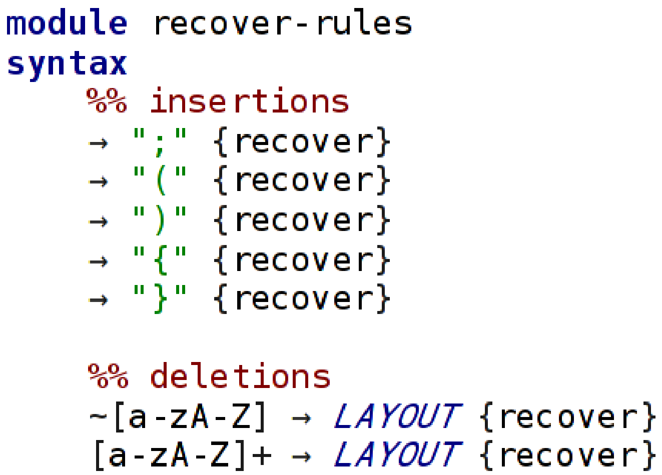

File
Edit
Source
Refactor
Navigate
Search
Project
Run
Window
Help
Java
Pack
Hier
Navi
*RecoverExample.java
org.eclipse.imp.xform.test [org
org.spoofax-feature [spoofax/t
org.spoofax.aterm [spoofax/tru
org.spoofax.compiler [spoofax/
org.spoofax.editor [spoofax/tru
org.spoofax.help [spoofax/trun
org.spoofax.interpreter [spoofa
org.spoofax.interpreter-feature
org.spoofax.interpreter.adapte
org.spoofax.interpreter.adapte
org.spoofax.interpreter.adapte
org.spoofax.interpreter.adapte
org.spoofax.interpreter.adapte
org.spoofax.interpreter.core [s
org.spoofax.interpreter.core-fe
org.spoofax.interpreter.library.e
org.spoofax.interpreter.library.j
org.spoofax.jsglr [spoofax-reco
org.spoofax.jsglr-feature [spoo
org.spoofax.releng.builder [spo
org.spoofax.sweave [spoofax/t
org.strategoxt.imp.editors.edit
org.strategoxt.imp.feature [sp
org.strategoxt.imp.metatoolin
org.strategoxt.imp.runtime [sp
Search
Call Hierarchy
Error Log
Console
Problems
Writable
Smart Insert
21 : 1

```
package test;

public class RecoverExample {

    public int methodX(int n){
        while(n<10){
            n+=2;
            if(n>=4) //{
                System.out.print(n);
            }
        }
        foo();
        return n;
    }

    public void foo(){ }
}
```

3 errors, 1,358 warnings, 0 others (Filter matched 103 of 1361 items)

Description	Resource	Path	Location	Type
Errors (3 items)				
Syntax error on token "...", invalid Expression	RecoverExample_BP2.txt	TestCoarseGrained/sr	line 6	Java
The public type RecoverExample must be defi	RecoverExample_BP2.txt	TestCoarseGrained/sr	line 3	Java
The type RecoverExample is already defined	RecoverExample_BP2.txt	TestCoarseGrained/sr	Unknown	Java

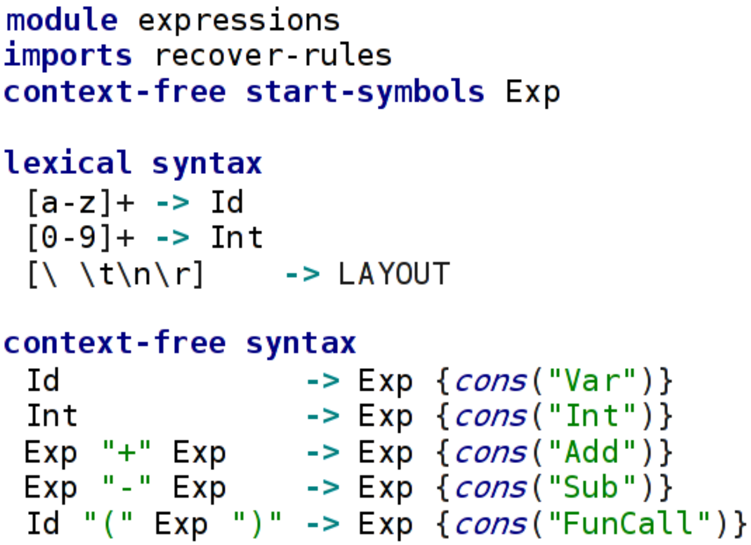
Recovery Rules



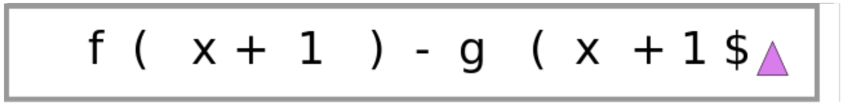
```
module recover-rules
syntax
    %% insertions
    → ";" {recover}
    → "(" {recover}
    → ")" {recover}
    → "{" {recover}
    → "}" {recover}

    %% deletions
    ~[a-zA-Z] → LAYOUT {recover}
    [a-zA-Z]+ → LAYOUT {recover}
```

- Java recovery module
  - Insertions
  - Deletions

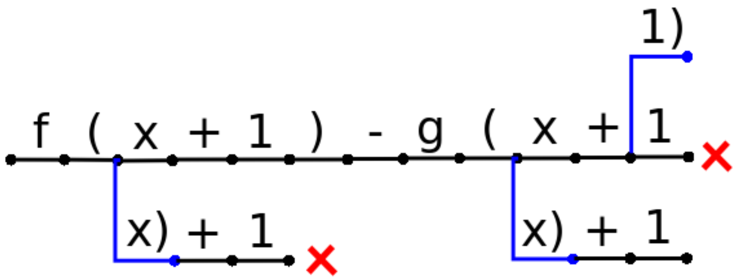
Generalized Parsing

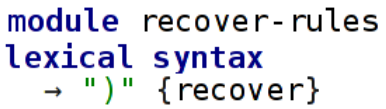


```
module expressions
imports recover-rules
context-free start-symbols Exp

lexical syntax
 [a-z]+ -> Id
 [0-9]+ -> Int
 [\ \t\n\r]    -> LAYOUT

context-free syntax
 Id              -> Exp {cons("Var")}
 Int             -> Exp {cons("Int")}
 Exp "+" Exp     -> Exp {cons("Add")}
 Exp "-" Exp     -> Exp {cons("Sub")}
 Id "(" Exp ")"  -> Exp {cons("FunCall")}
```



```
f ( x+ 1 ) - g ( x +1 $
```





```
module recover-rules
lexical syntax
  → ")" {recover}
```