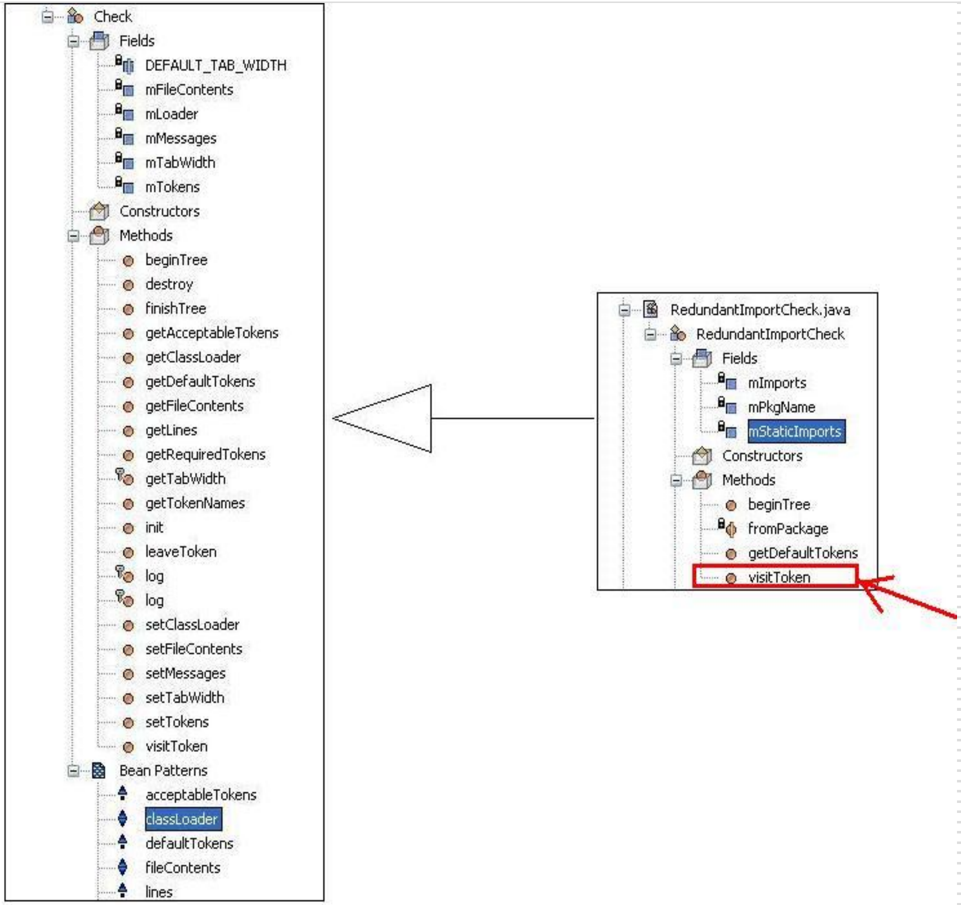
Advance Topics In Software Engineering

Episode 2: Design presentation

```
if(everything_keeps_going_fine)
{
   contribution_to_checkstyle(DeadLine dL);
       if(enough_time_left)
          System.out.println(“more changes in mind”);
}
```

Ioannis Sermetziadis
Department Of Management Science and Technology
Athens University Of Economics and Business

1st Change: **RedundantImportCheck.java**

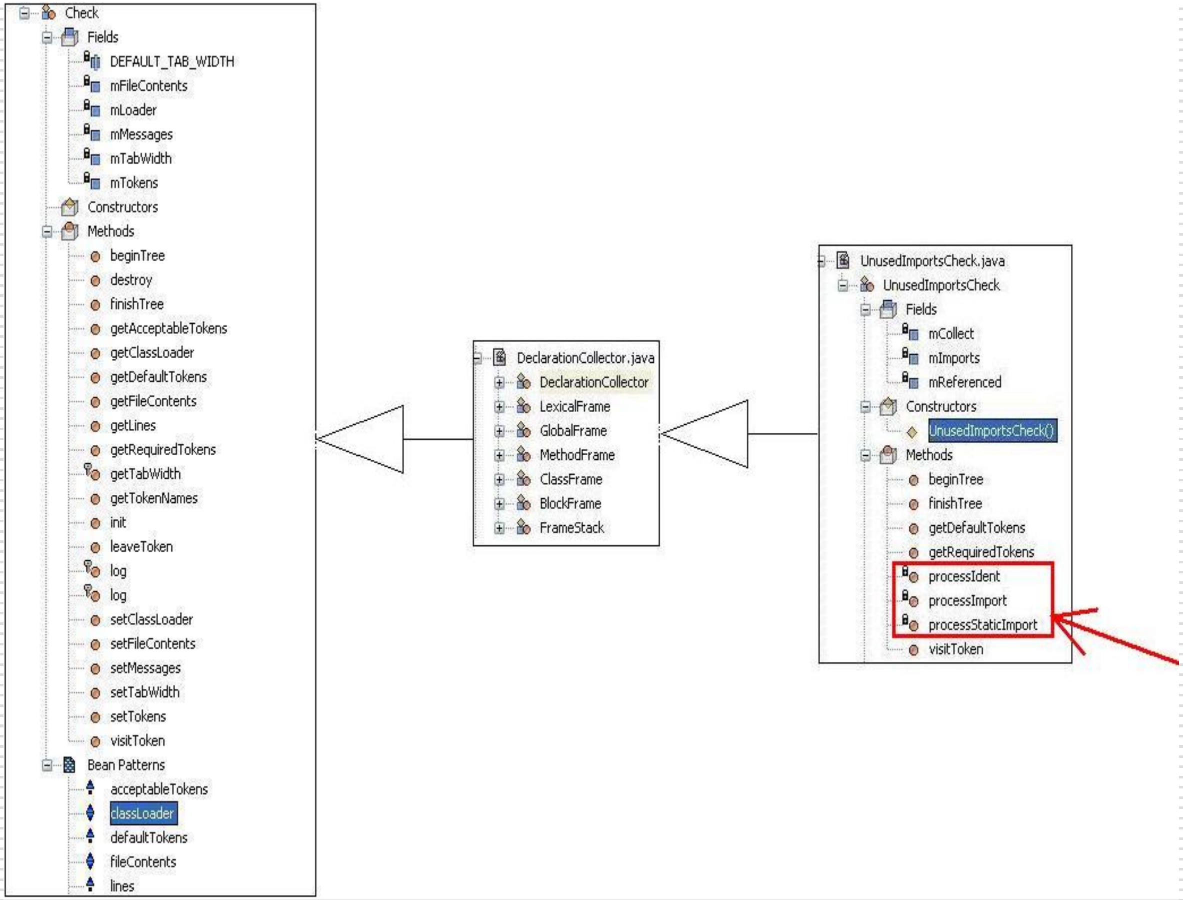


- ➢Have to change the method **visitToken(DetailAST aAST)**

- ➢**DetailAST** is a class that extends the functionality of **CommonAST** class that is imported from **antlr** library and records the line and the column numbers of the class parameters.

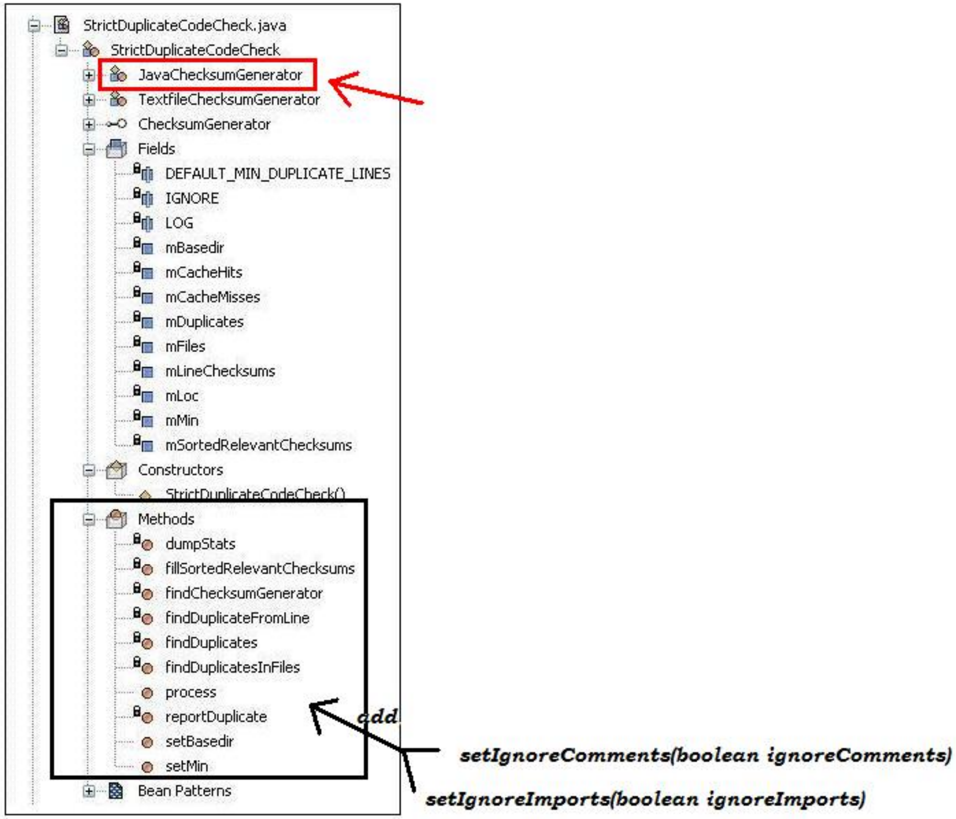
- ➢**RedundantImportCheck** *extends* **Check** abstract class which is the base class for all checks.

2nd Change: **UnusedImportsCheck.java**



- ➢Have to change the methods **processIdent (DetailAST aAST)**, **processImport(DetailAST aAST)** and **processStaticImport(DetailAST aAST),** that **visitToken(DetailAST aAST)** uses in order to check if the imported classes from a ".*" import are finally used.
- ➢ **UnusedImportsCheck** *extends* **DeclarationCollector** abstract class which is used for checks which need to collect information about declared members, parameters or variables of a class. **DeclarationCollector** *extends* **Check** abstract class.

3rd Change: **StrictDuplicateCode.class**



- StrictDuplicateCodeCheck.java
  - StrictDuplicateCodeCheck
    - JavaChecksumGenerator
    - TextfileChecksumGenerator
    - ChecksumGenerator
    - Fields
      - DEFAULT_MIN_DUPLICATE_LINES
      - IGNORE
      - LOG
      - mBasedir
      - mCacheHits
      - mCacheMisses
      - mDuplicates
      - mFiles
      - mLineChecksums
      - mLoc
      - mMin
      - mSortedRelevantChecksums
    - Constructors
      - StrictDuplicateCodeCheck()
    - Methods
      - dumpStats
      - fillSortedRelevantChecksums
      - findChecksumGenerator
      - findDuplicateFromLine
      - findDuplicates
      - findDuplicatesInFiles
      - process
      - reportDuplicate
      - setBasedir
      - setMin
    - Bean Patterns

add

setIgnoreComments(boolean ignoreComments)

setIgnoreImports(boolean ignoreImports)

- Gives the functionality to the user to decide if he wants to include the import statements or the comments in the duplication check. I have to add two **setter methods**. Afterwards, I should add some code to the method **calcChecksum(String aLine)** that reads the file text in order to include or not the imports and the comments in the check.
- **XML properties example:**

```
<module name="Checker">
  <module name="StrictDuplicateCode">
    <property name="min" value="5"/>
    <property name="ignoreImports"
value="false"/>
    <property name="ignoreComments"
value="false"/>
  </module>
</module>
```

- **AutomaticBean.class** from checkstyle's API detects the setter methods when property values are given in the configuration file.

4th Change: Write my own check **ClassBlankness.class**

- Add this class in the package **com.puppycrawl.tools.checkstyle.checks**

- A method called **blankCalculator(String aLine)** will implement the check by calculating the percentage of the blank lines in the file. The lines considered as blank are those which include at most a '{' or a '}'.

- There will be a setter method **setMinBlank(double minBlank)** to give the percentage of the blank lines of a class which should be the minimum in a file in order to not be reported. This will be the check's property.

%Further thoughts%

➢Enrich the “poor” gui of checkstyle, which can just display the structure of selected java file.

(new class in the package **com.puppycrawl.tools.checkstyle.gui)**

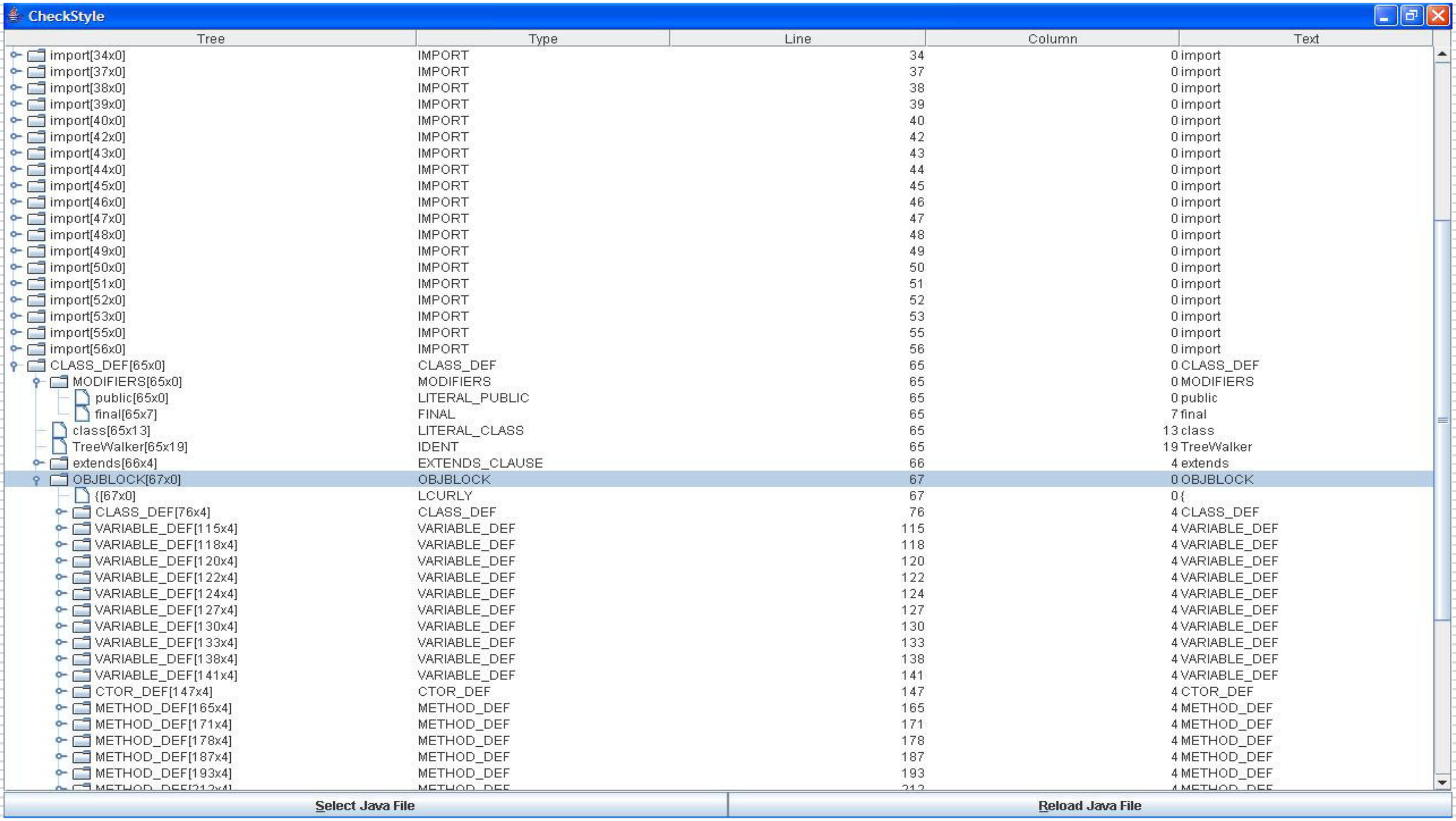
- Add a read only editor field that highlights the selected AST node.
- Clicking in the editor field could open the corresponding tree path.

➢ Export the results of the checks in a .java file (Files’_name+”checked”.class) including the results of the check in the specific line that they are referred to.

A new class is needed (JavaLogger.java) in addition to DefaultLogger.java and XMLLogger.java that already exist and give .txt and .xml outputs.

➢ Functionality to check C# code instead of Java is something that interests many of the project’s users.

How checkstyle’s GUI looks like



Tree	Type	Line	Column	Text
import[34x0]	IMPORT	34	0	import
import[37x0]	IMPORT	37	0	import
import[38x0]	IMPORT	38	0	import
import[39x0]	IMPORT	39	0	import
import[40x0]	IMPORT	40	0	import
import[42x0]	IMPORT	42	0	import
import[43x0]	IMPORT	43	0	import
import[44x0]	IMPORT	44	0	import
import[45x0]	IMPORT	45	0	import
import[46x0]	IMPORT	46	0	import
import[47x0]	IMPORT	47	0	import
import[48x0]	IMPORT	48	0	import
import[49x0]	IMPORT	49	0	import
import[50x0]	IMPORT	50	0	import
import[51x0]	IMPORT	51	0	import
import[52x0]	IMPORT	52	0	import
import[53x0]	IMPORT	53	0	import
import[55x0]	IMPORT	55	0	import
import[56x0]	IMPORT	56	0	import
CLASS_DEF[65x0]	CLASS_DEF	65	0	CLASS_DEF
MODIFIERS[65x0]	MODIFIERS	65	0	MODIFIERS
public[65x0]	LITERAL_PUBLIC	65	0	public
final[65x7]	FINAL	65	7	final
class[65x13]	LITERAL_CLASS	65	13	class
TreeWalker[65x19]	IDENT	65	19	TreeWalker
extends[66x4]	EXTENDS_CLAUSE	66	4	extends
OBJBLOCK[67x0]	OBJBLOCK	67	0	OBJBLOCK
{[67x0]	LCURLY	67	0	{
CLASS_DEF[76x4]	CLASS_DEF	76	4	CLASS_DEF
VARIABLE_DEF[115x4]	VARIABLE_DEF	115	4	VARIABLE_DEF
VARIABLE_DEF[118x4]	VARIABLE_DEF	118	4	VARIABLE_DEF
VARIABLE_DEF[120x4]	VARIABLE_DEF	120	4	VARIABLE_DEF
VARIABLE_DEF[122x4]	VARIABLE_DEF	122	4	VARIABLE_DEF
VARIABLE_DEF[124x4]	VARIABLE_DEF	124	4	VARIABLE_DEF
VARIABLE_DEF[127x4]	VARIABLE_DEF	127	4	VARIABLE_DEF
VARIABLE_DEF[130x4]	VARIABLE_DEF	130	4	VARIABLE_DEF
VARIABLE_DEF[133x4]	VARIABLE_DEF	133	4	VARIABLE_DEF
VARIABLE_DEF[138x4]	VARIABLE_DEF	138	4	VARIABLE_DEF
VARIABLE_DEF[141x4]	VARIABLE_DEF	141	4	VARIABLE_DEF
CTOR_DEF[147x4]	CTOR_DEF	147	4	CTOR_DEF
METHOD_DEF[165x4]	METHOD_DEF	165	4	METHOD_DEF
METHOD_DEF[171x4]	METHOD_DEF	171	4	METHOD_DEF
METHOD_DEF[178x4]	METHOD_DEF	178	4	METHOD_DEF
METHOD_DEF[187x4]	METHOD_DEF	187	4	METHOD_DEF
METHOD_DEF[193x4]	METHOD_DEF	193	4	METHOD_DEF
METHOD_DEF[212x4]	METHOD_DEF	212	4	METHOD_DEF

What is done till now…

- Changes to **RedundantImportCheck.java** are already finished.

- Changes to **UnusedImportsCheck.java** are on the way to be finished.

- The half part of the intended changes to **StrictDuplicateCode.class** is finished.

- I have understood the parts of the checkstyle’s API that need to be used in order to write my own check.

- Already started thinking of some more changes.

Thank you for your attention-)

ANY QUESTIONS?