Robert Rozmus

30.12.2014

ISI – dokumentacja

Temat - Metoda i implementacja detekcji plagiatów w kodach programów

Metoda

Wykrywanie plagiatów odbywa się poprzez zamianę zawartości plików na znaczniki oraz zaznaczanie podobnych wierszy w macierzy podobieństwa.

Implementacja (algorytm)

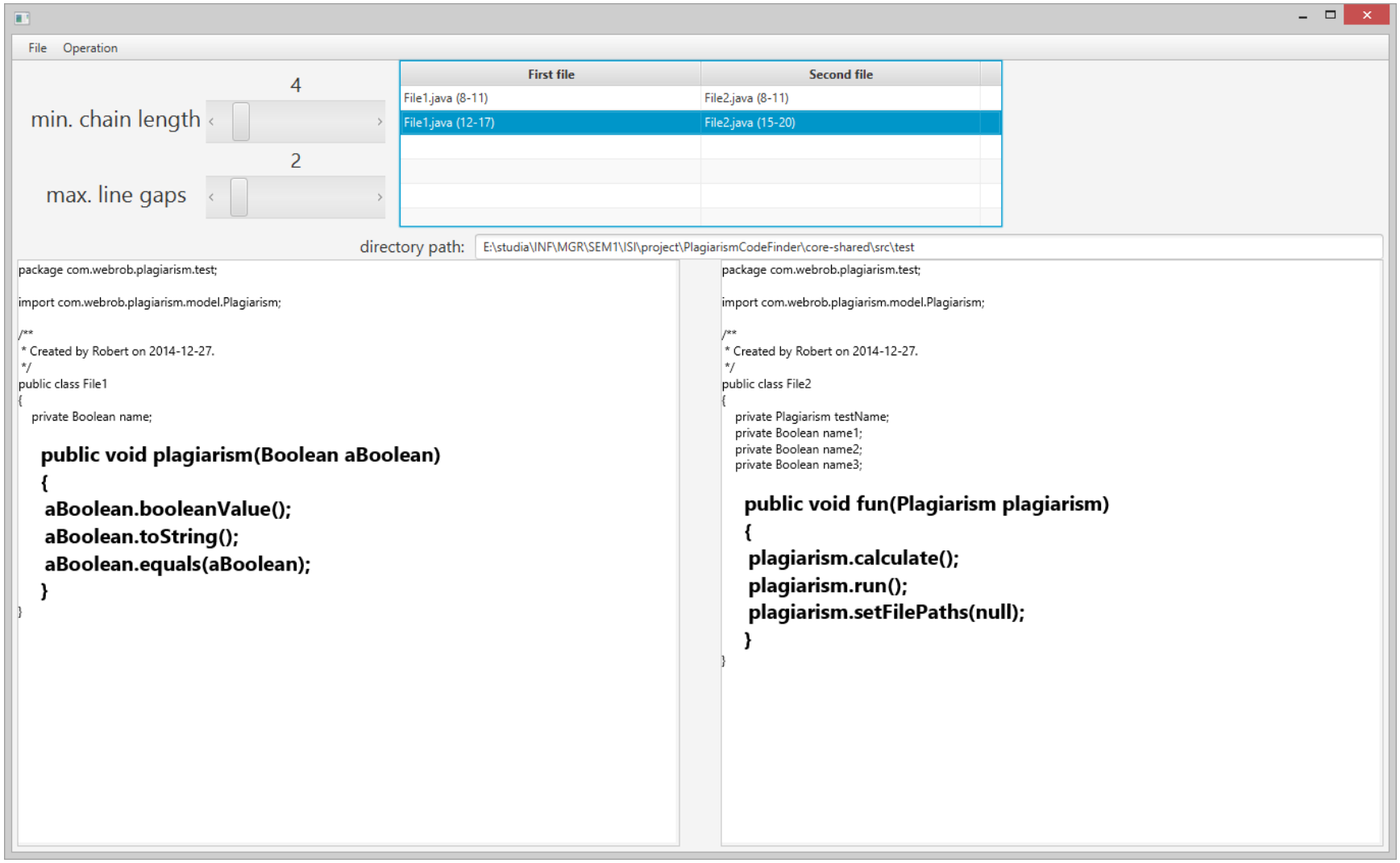
1. We wskazanym folderze i jego folderach wewnętrznych poszukiwane są pliki z rozszerzeniem .java lub .cpp.
2. Zawartość każdego pliku przechodzi analizę leksykalną, która część zawartości zamienia na znaczniki, a część usuwa:
    a. Usuwane są białe znaki, nawiasy, średniki, komentarze oraz wyrazy: *package*, *import*;
    b. Liczby zamieniane są na znacznik *NUMBER*;
    c. Nazwy zmiennych i klas zamienianie są na znacznik *ID*;
    d. Nazwy funkcji zamieniane są na znacznik FUNCTION_NAME;
    e. Ciągi znaków występujące pomiędzy cudzysłowami zamieniane są znacznik *STRING*;
    f. Pozostałe znaki nie ulegają zmianom.
3. Dla każdej pary plików:
    a. Tworzona jest tablica haszująca zawierająca numery wierszy obu plików.
    b. Każda taka sama para linii z plików po analizie leksykalnej jest oznaczana w macierzy.
    c. Oznaczone w macierzy linie są łączone w łańcuchy o minimalnej długości zadanej parametrem *min. chain length*.
    d. Łańcuch jest rozszerzany o linie nieoznaczone w macierzy zgodnie z parametrem *max. line gaps*.

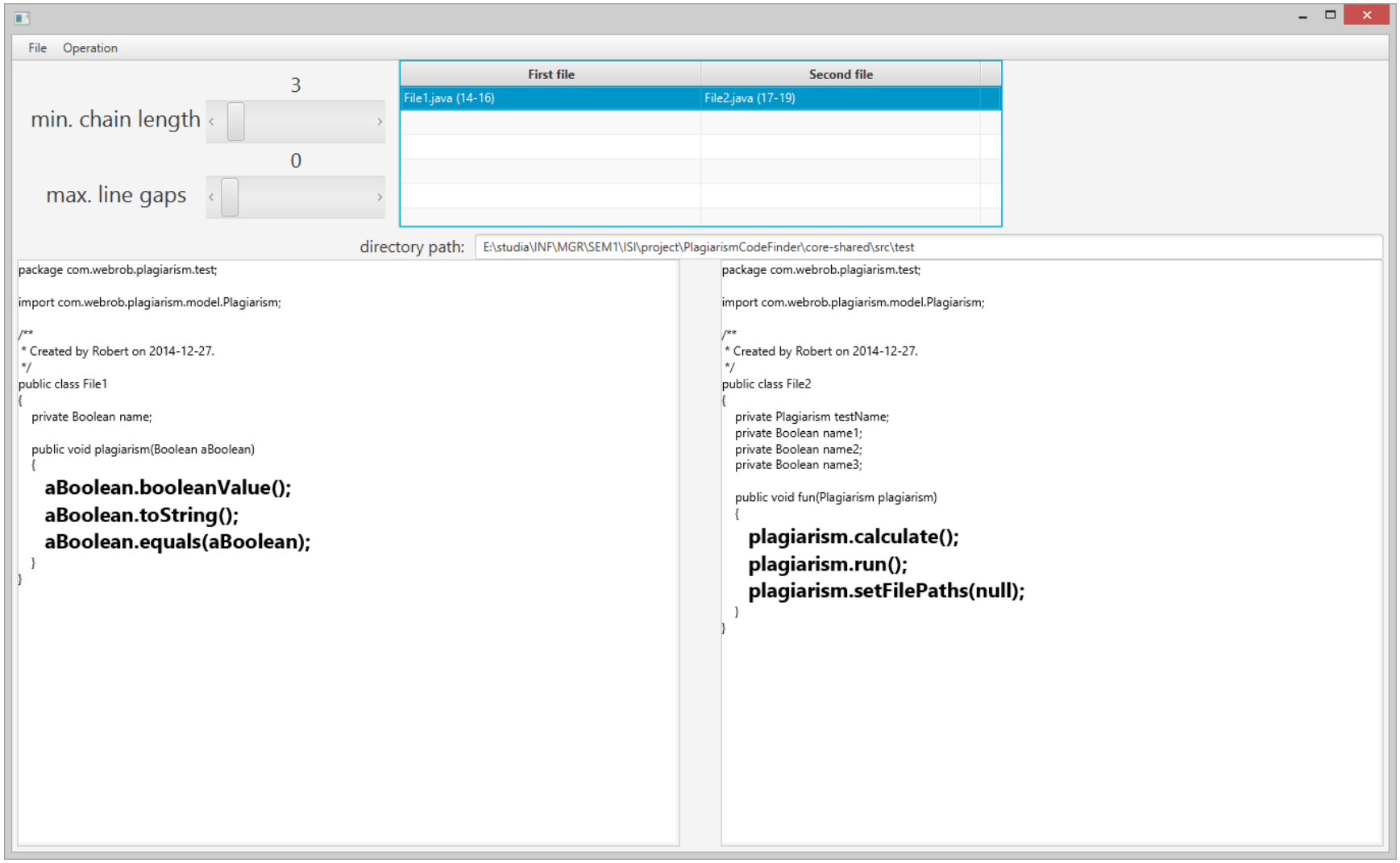
Test

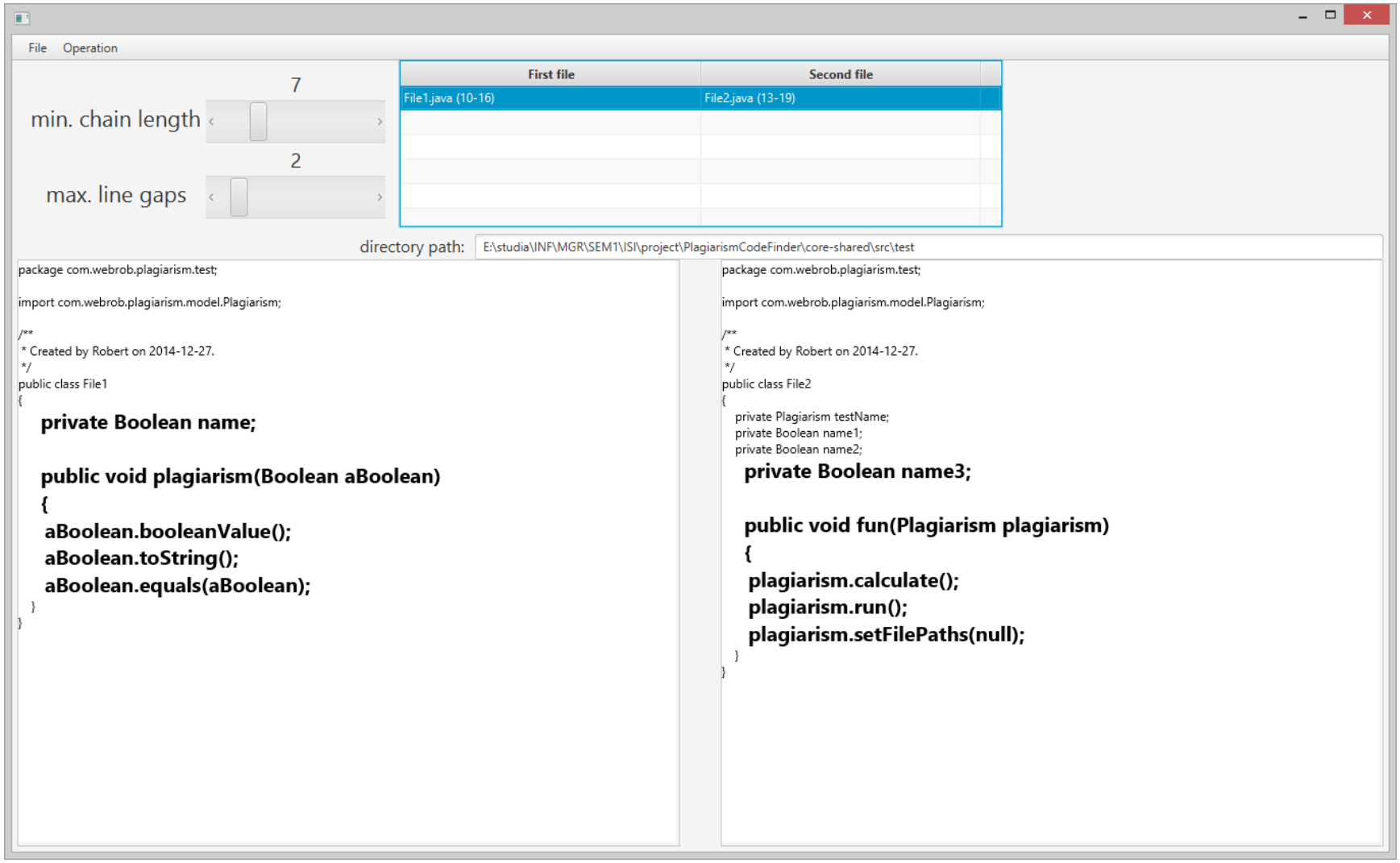
File1	File2
oryginalny	

package com.webrob.plagiarism.test;
 	
import com.webrob.plagiarism.model.Plagiarism;
 	
/**
 * Created by Robert on 2014-12-27.
 */
public class File1
{
  private Boolean name;
 	
  public void plagiarism(Boolean aBoolean)
  {
  aBoolean.booleanValue();
  aBoolean.toString();
  aBoolean.equals(aBoolean);
  }
}
	package com.webrob.plagiarism.test;
	
	import com.webrob.plagiarism.model.Plagiarism;
	
	/**
	 * Created by Robert on 2014-12-27.
	 */
	public class File2
	{
	  private Plagiarism testName;
	  private Boolean name1;
	  private Boolean name2;
	  private Boolean name3;
	
	  public void fun(Plagiarism plagiarism)
	  {
	  plagiarism.calculate();
	  plagiarism.run();
	  plagiarism.setFilePaths(null);
	  }
	}

po analizie leksykalnej	
publicclassID privateIDID publicvoidFUNCTION_NAMEIDID ID.FUNCTION_NAME ID.FUNCTION_NAME ID.FUNCTION_NAMEID	publicclassID privateIDID privateIDID privateIDID privateIDID publicvoidFUNCTION_NAMEIDID ID.FUNCTION_NAME ID.FUNCTION_NAME ID.FUNCTION_NAMEID


File Operation
4
min. chain length
2
max. line gaps
First file
Second file
File1.java (8-11)
File2.java (8-11)
File1.java (12-17)
File2.java (15-20)
directory path:
E:\studia\INF\MGR\SEM1\ISI\project\PlagiarismCodeFinder\core-shared\src\test
package com.webrob.plagiarism.test;
import com.webrob.plagiarism.model.Plagiarism;
/**
* Created by Robert on 2014-12-27.
*/
public class File1
{
private Boolean name;
public void plagiarism(Boolean aBoolean)
{
aBoolean.booleanValue();
aBoolean.toString();
aBoolean.equals(aBoolean);
}
}
package com.webrob.plagiarism.test;
import com.webrob.plagiarism.model.Plagiarism;
/**
* Created by Robert on 2014-12-27.
*/
public class File2
{
private Plagiarism testName;
private Boolean name1;
private Boolean name2;
private Boolean name3;
public void fun(Plagiarism plagiarism)
{
plagiarism.calculate();
plagiarism.run();
plagiarism.setFilePaths(null);
}
}


File
Operation
3
min. chain length
0
max. line gaps
First file
Second file
File1.java (14-16)
File2.java (17-19)
directory path:
E:\studia\INF\MGR\SEM1\ISI\project\PlagiarismCodeFinder\core-shared\src\test
package com.webrob.plagiarism.test;
import com.webrob.plagiarism.model.Plagiarism;
/**
* Created by Robert on 2014-12-27.
*/
public class File1
{
private Boolean name;
public void plagiarism(Boolean aBoolean)
{
aBoolean.booleanValue();
aBoolean.toString();
aBoolean.equals(aBoolean);
}
}
package com.webrob.plagiarism.test;
import com.webrob.plagiarism.model.Plagiarism;
/**
* Created by Robert on 2014-12-27.
*/
public class File2
{
private Plagiarism testName;
private Boolean name1;
private Boolean name2;
private Boolean name3;
public void fun(Plagiarism plagiarism)
{
plagiarism.calculate();
plagiarism.run();
plagiarism.setFilePaths(null);
}
}


File
Operation
7
min. chain length
2
max. line gaps
First file
Second file
File1.java (10-16)
File2.java (13-19)
directory path:
E:\studia\INF\MGR\SEM1\ISI\project\PlagiarismCodeFinder\core-shared\src\test
package com.webrob.plagiarism.test;
import com.webrob.plagiarism.model.Plagiarism;
/**
* Created by Robert on 2014-12-27.
*/
public class File1
{
private Boolean name;
public void plagiarism(Boolean aBoolean)
{
aBoolean.booleanValue();
aBoolean.toString();
aBoolean.equals(aBoolean);
}
}
package com.webrob.plagiarism.test;
import com.webrob.plagiarism.model.Plagiarism;
/**
* Created by Robert on 2014-12-27.
*/
public class File2
{
private Plagiarism testName;
private Boolean name1;
private Boolean name2;
private Boolean name3;
public void fun(Plagiarism plagiarism)
{
plagiarism.calculate();
plagiarism.run();
plagiarism.setFilePaths(null);
}
}