

Eclipse plugin for FindBugs és Microsoft Visual Studio Code Analysis eszközök összehasonlítása



BUDAI PÉTER

(budai@iit.bme.hu)

BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM

2011. 12. 01.

Áttekintés



- Cél a lehetséges programozói hibák detektálása
 - A fejlesztési folyamat minél korábbi pontján
 - ✦ Ezzel időt és költséget spórolhatunk
- Statikus analízis
 - A kód lefuttatása nélkül próbáljuk felismerni a potenciális hibaforrásokat
 - Az analízis kiterjedhet
 - ✦ Forráskódra
 - ✦ Lefordított tárgykódra
 - Ez utóbbi a jellemző (egyszerűbb)
 - Köztes kódot alkalmazó nyelveknél (Java/.NET) még könnyebb dolgunk van, a tárgykód "beszédese"

FindBugs



- **University of Maryland**
 - William Pugh & David Hovemeyer, 2003
 - Nyílt forráskód (LGPL)
- **Java nyelvhez (Java SE 1.5+) készült**
 - Hibakeresés és helytelen programozói gyakorlatok kiszűrése
 - Hasonló eszközök még:
 - ✦ PMD, JLint, (Bandera), (ESC/Java)
- **Többféle UI**
 - Swing alapú önálló alkalmazás
 - Eclipse plugin
 - Ant task, parancssor, ...

FindBugs



- **Hogyan működik?**
 - .class fájlok statikus analízise (BCEL vagy ASM library)
 - Különböző hibamintákat (bug pattern) keres
 - ✦ Előfordulhatnak hamis pozitív esetek (<50%)
 - ✦ Könnyen bővíthető újabb mintával
 - Az összegyűjtött hibalistát valamilyen módon visszkapjuk
 - ✦ XML állomány
 - ✦ Felhasználói felület: Hibalista és forráskód nézet
 - Ez utóbbi feltétele, hogy hozzáférhető legyen a forráskód
 - Szöveges leírások ez egyes hibákhoz

FindBugs Eclipse plugin



FindBugs - blade-forgersrc/test/java/blade/forgersrc/engine/charset/CharSetTest.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Bug Explorer

- blade-forgersrc 0.1-SNAPSHOT (5) [15 svn+ssh://bu...
- Could be refactored into a named static inner c...
- The class blade.forgersrc.engine.charset.Refer...
- Field not initialized in constructor (2)
 - CharSetTest.refSets not initialized in constr...**
 - CharSetTest.testSets not initialized in const...
- Method uses toArray() with zero-length array a...
- Method blade.forgersrc.engine.charset.Fragm...
- Non-transient non-serializable instance field in...
- Class blade.forgersrc.parser.SyntaxException d...

```
CharsetsTest.java | SyntaxException.java | CompleteCharSet.java
```

```
 * Collection of various character sets under test.
 */
private List<DefaultCharSet> testSets;

/**
 * Collection of various reference character sets.
 */
private List<ReferenceCharSet> refSets;

/**
 * Set up initial character set collections.
 */
```

Properties Problems

Bug: CharSetTest.refSets not initialized in constructor

Bug: CharSetTest.refSets not initialized in constructor
Pattern id: UWF_FIELD_NOT_INITIALIZED_IN_CONSTRUCTOR, **type:** UwF,
category: STYLE

This field is never initialized within any constructor, and is therefore could be null after the object is constructed. This could be either an error or a questionable design, since it means a null pointer exception will be generated if that field is dereferenced before being initialized.

CharSetTest.refSets not initialized in constructor

Microsoft Code Analysis



- **Microsoft**
 - Visual Studio 2010 része
 - ✦ Ultimate és Premium változatokban
 - ✦ FxCop utódja
 - Nem ingyenes, de MSDNAA-ról elérhető
- **C/C++ és bármilyen .NET nyelvhez**
 - Hibák felfedezésére és a forráskód minőségének javítására
 - Hasonló eszközök:
 - ✦ Gendarme, StyleCop, dotTEST
- **Visual Studio-ba integrált kezelőfelület**

Microsoft Code Analysis



- **Hogyan működik**
 - CIL köztes nyelvű szerelvények statikus analízise
 - Szintén hibamintákkal dolgozik
 - ✦ Bővíthető ugyan, de nem támogatott
 - ✦ Az egyes hibajelzések attribútumokkal kikapcsolhatók a kódból
 - A felismert hibák a fejlesztőkörnyezetben jelennek meg
 - Szöveges leírások az egyes hibákhoz
 - ✦ Bővebb ismertető
 - ✦ Tippek javítási lehetőségekre

Microsoft Code Analysis



The screenshot shows the Microsoft Visual Studio interface with the following components:

- Code Editor:** Displays the `OnClickTerritory` method in `MainForm.cs`. The code is as follows:

```
protected override void OnClickTerritory(object sender, TerritoryEventArgs e)
{
    Territory territory = e.Territory;
    if (e.Cell.IsCore)
    {
        territory.Clear();
        panel.Table.Territories.Remove(territory);
    }
    else
    {
        territory.Remove(e.Cell);
    }
}
```
- Solution Explorer:** Shows the project structure for 'Bdice', including files like `Properties`, `References`, `Resources`, `Data.cs`, `Gen2Form.cs`, `GenerateForm.cs`, `IntUpDown.cs`, and `MainForm.cs`.
- Error List:** Displays a table of analysis results. The table has columns for 'Description', 'File', 'Line', 'Column', and 'Project'. The following table summarizes the data shown in the screenshot:

Line	Description	File	Line	Column	Project
20	CA1062 : Microsoft.Design : In externally visible method 'CellEditMode.OnClickTerritory (object, TerritoryEventArgs)', validate parameter 'e' before using it.	MainForm.cs	396		Bdice
21	CA1062 : Microsoft.Design : In externally visible method 'CoreEditMode.OnClickCell(object, CellEventArgs)', validate parameter 'e' before using it.	MainForm.cs	333		Bdice
22	CA1062 : Microsoft.Design : In externally visible method 'CoreEditMode.OnClickTerritory (object, TerritoryEventArgs)', validate parameter 'e' before using it.	MainForm.cs	342		Bdice
23	CA1062 : Microsoft.Design : In externally visible method 'DeleteEditMode.OnClickTerritory (object, TerritoryEventArgs)', validate parameter 'e' before using it.	MainForm.cs	360		Bdice
24	CA1012 : Microsoft.Design : Change the accessibility of all public constructors in 'EditMode' to protected.	MainForm.cs	277		Bdice

The Error List also shows a summary at the top: 0 Errors, 227 Warnings, and 0 Messages. The status bar at the bottom indicates 'Ready'.

Összehasonlítás



- Mindkét eszköz hasonló elven működik
 - Köztes nyelvű kódon dolgoznak
 - ✦ Java bytecode \Leftrightarrow .NET IL
 - Hibamintákat keresnek
 - ✦ A különbség az ismert minták számában van:
 - 548 (FindBugs) \Leftrightarrow 214 (CodeAnalysis)
 - ✦ Ez egyes minták vizsgálata ki/be kapcsolhatók
 - Vannak csak költségesen ellenőrizhető minták
 - ✦ Bővíthetőek újabb mintákkal
 - Csak a szükséges erőfeszítés mértéke eltérő
- Integráció a fejlesztőkörnyezetbe

Összehasonlítás



- Hibaminták csoportosíthatók
 - Súlyosság szerint
 - ✦ FindBugs: Low, Medium, High prioritás
 - ✦ CodeAnalysis: Ignore, Warning, Error
 - Hiba típusa szerint
 - ✦ FindBugs:
 - Bad practice, Correctness, Multithreaded correctness, Malicious code vulnerability, *Internationalization*, *Performance*, *Security*, Dodgy code
 - ✦ CodeAnalysis:
 - Design, *Globalization*, Interoperability, Maintainability, Mobility, Naming, *Performance*, Portability, Reliability, *Security*, Usage

Hibakategóriák



- **FindBugs**

- Internationalization
- Bad practice
- Performance
- Malicious code vulnerab.
- Correctness
- Multithreaded correctn.
- Security
- Dodgy code

- **CodeAnalysis**

- Globalization
- Design
- Naming
- Usage
- Maintainability
- Reliability
- Security
- Mobility
- Interoperability
- Portability
- Performance

Hibakategóriák



- FindBugs

- Internationalization (2)
 - ✦ String-byte konverzió
- Bad practice (86)
 - ✦ equals() és társai
 - ✦ finalize()
 - ✦ API konvenciók
 - ✦ Elnevezési konvenciók
 - ✦ I/O, szerializáció
- Performance (61)
 - ✦ Boxing
 - ✦ String művetek

- CodeAnalysis

- Globalization (11)
 - ✦ Alkalmazás-lokalizáció
- Design (62)
 - ✦ API konvenciók
 - ✦ Névtér, interfész és metódus tervezési kritériumok
- Naming (25)
 - ✦ Pozitív és negatív elnevezési konvenciók
- Usage (44)
 - ✦ Dispose, finalizer
 - ✦ Szerializáció
 - ✦ Operátor felüldefiniálás

Hibakategóriák



• FindBugs

- Malicious code vulnerab. (14)
 - ✦ Fokozott adatretjtés
- Correctness (213)
 - ✦ Rendkívül változatos
 - ✦ Bitműveletek
 - ✦ equals()
 - ✦ null
 - ✦ Dead store, visszatérési érték
- Multithreaded correctn. (52)
 - ✦ Szinkronizáció helytelen használata
 - ✦ Nem szálbiztos hívások

• CodeAnalysis

- Maintainability (6)
 - ✦ Kód komplexitás
- Reliability (6)
 - ✦ Natív erőforrások
- Security (44)
 - ✦ CLR-specifikus biztonsági elemek
- Mobility (2)
 - ✦ Energiagazdálkodás

Hibakategóriák



- FindBugs

- Security (9)

- ✦ SQL és HTTP sebezhetőségek

- Dodgy code (92)

- ✦ Felesleges, ismétlődő kódrészletek
- ✦ Elérhetetlen kód, változó, metódus

- CodeAnalysis

- Interoperability (15)

- ✦ COM és Win32

- Portability (3)

- ✦ OS és .NET kompatibilitás

- Performance (17)

- ✦ Használaton kívüli elemek

Konklúzió



- Jól használható eszközök
- A hibaminták között aránylag kicsi az átfedés
 - Nem várt eredmény
 - Sok a platformfüggő elem
 - Mindkettőben előfordul:
 - ✦ Nem használt kód felderítése
 - ✦ Az Object örökölt metódusainak működése
 - ✦ Elnevezési konvenciók

Konklúzió



- Kicsit más megközelítések
 - FindBugs
 - ✦ Több, aprólékosabb szabály
 - ✦ A kód helyessége, hibalehetőségek felderítése hangsúlyos
 - CodeAnalysis
 - ✦ Kevesebb, átfogóbb, magasabb szintű szabályok
 - ✦ Több a nyelvi és platform konvenciókat ellenőrző minta
 - Megfigyelhető az akadémiai és az ipari világ közti különbség
- Statikus analízis önmagában nem elegendő

Kérdések

