A vizsgálat folyamata Codacy-val történi statikusanalízis, majd manuális átnézés.

Összesen talált problémák: 55 (ebből 45 java, 10 markdown)

1.

The utility class name 'Beans' doesn't match '[A-Z][a-zA-Z0-9]+(Utils?|Helper|Constants)'

(main/src/com/complexible/common/beans/Beans.java)

```
public final class Beans {

        private Beans() {
```

A problémát abban látja, hogy nincs jelezve az osztály nevében, hogy final.

Annak megítélése, hogy ez így hibás-e kétes, mert nem ismerem a fejlesztő előírását a nevekre kapcsolódóan, hagyományosan azonban ajánlott az osztály ezen tulajdonságának jelzése a névben.

Továbbá következik az az észrevétel is ebből, hogy nincs dokumentálva az elvárt design, ez hosszútávon gondot okozhat.

(becsült javítási idő 5 perc)

2. The utility class name 'Classes' doesn't match '[A-Z][a-zA-Z0-9]+(Utils?|Helper|Constants)'

(main/src/com/complexible/common/reflect/Classes.java)

Hasonló az előzőhöz, ugyanaz a megállapítás tehető

3. The static method name '_implements' doesn't match '[a-z][a-zA-Z0-9]*'

(main/src/com/complexible/common/reflect/Classes.java)

```
    * @param theInterface  the interface
    * @return             true if it implements it, false otherwise
    */
   public static boolean _implements(final Class<?> theClass, final Class<?> theInterface) {
           return any(interfaces(theClass), Predicates.<Class<?>>equalTo(theInterface));
   }
```

Itt a static feltüntetésének az osztálynévben való feltüntetésének iánya mellett, még a _ karakterrel kezdés is problémás. Ez 9-es verziónál régebbi javaban nem működött, azonban azóta se ajánlják a használatát.

Szintén csak code design probléma (5 perc a becsült javítási idő)

4. The utility class name 'Fields' doesn't match '[A-Z][a-zA-Z0-9]+(Utils?|Helper|Constants)'

(main/src/com/complexible/common/reflect/Fields.java)

Elsőhöz hasonló, ugyanúgy final

5. The utility class name 'Methods' doesn't match '[A-Z][a-zA-Z0-9]+(Utils?|Helper|Constants)'

(main/src/com/complexible/common/reflect/Methods.java)

Ugyanúgy elsőhöz hasonlóan final nincs jelezve.

6. Fields should be declared at the top of the class, before any method declarations, constructors, initializers or inner classes.

(main/src/com/complexible/pinto/MappingOptions.java)

A problémája az, hogy van metódus valamely mező deklarálása előtt, ez modern programozási nyelveknél szintaktikailag nem hiba, csak áttekinthetőségileg. Azonban én úgy gondolom, és úgy szoktam csinálni, hogy ha valamely változót csak egy adott függvény használ, akkor közvetlen fölötte deklarálom, így valamennyire látszik, hogy hozzá tartozik. (ennek 5 perc becsülve)

7. Unnecessary use of fully qualified name 'java.net.URI.create' due to existing import 'java.net.URI'

(test/src/com/complexible/pinto/RDFMapperTests.java)

Változó deklarálásnál specifikusan megadja, hogy melyik osztálynak a URI-ját szeretné használni, ami nem szükséges.

Törölhető a specifikusabb, kb 5 perc.

8. Document empty method body

(test/src/com/complexible/pinto/RDFMapperTests.java)

```
@Test
@Ignore
public void testWriteWithLangTag() throws Exception {
}
```

jelezve van, hogy csak a teszt miatt, szóval figyelmen kívül hagyható

9. Document empty method body

Előzőhöz hasonló

10. Unnecessary use of fully qualified name 'java.net.URI' due to existing import 'java.net.URI'

(test/src/com/complexible/pinto/RDFMapperTests.java)

Ugyanolyan, mint a 7-es

```
    private int mInt;
    private java.net.URI mURI;
    private float mFloat;
```

11. The utility class name 'MappingOptions' doesn't match '[A-Z][a-zA-Z0-9]+(Utils?|Helper|Constants)'

(main/src/com/complexible/pinto/MappingOptions.java)

A final tulajdonságát ajánlott lenne jelezni a nevében.

kb 5 perc a javítása.

12. Fields should be declared at the top of the class, before any method declarations, constructors, initializers or inner classes.

(main/src/com/complexible/pinto/MappingOptions.java)

```
public static final Options DEFAULTS = Options.unmodifiable(Options.empty());
```

Előtte van a konstruktor, ami viszont üres, így nem zavaró, hogy azután jön a mező, mert nem használjuk előtte. (a probléma forrása az, ami a 6-diknál)

13. Fields should be declared at the top of the class, before any method declarations, constructors, initializers or inner classes.

(main/src/com/complexible/pinto/MappingOptions.java)

```
public static final Option<Boolean> REQUIRE_IDS = Option.create("require.ids", false);
```

előzőhöz hasonló

14. Fields should be declared at the top of the class, before any method declarations, constructors, initializers or inner classes.

(main/src/com/complexible/pinto/MappingOptions.java)

```
public static final Option<Boolean> SERIALIZE_COLLECTIONS_AS_LISTS = Option.create("serialize.collections.as.lists", false);
```

12-eshez hasonló

15. Fields should be declared at the top of the class, before any method declarations, constructors, initializers or inner classes.

(main/src/com/complexible/pinto/MappingOptions.java)

```
public static final Option<Boolean> IGNORE_CARDINALITY_VIOLATIONS =
Option.create("ignore.cardinality.violations", false);
```

12-eshez hasonló

16. The utility class name 'Dates2' doesn't match '[A-Z][a-zA-Z0-9]+(Utils?|Helper|Constants)'

(main/src/com/complexible/pinto/RDFMapper.java)

```
private static final class Dates2
```

elsőhöz hasonló

17. Document empty method body

(main/src/com/complexible/pinto/RDFMapper.java)

```
@Test
@Ignore
public void testReadCustomCollectionMapping() throws Exception {
}
```

8-ashoz hasonló

18. Document empty method body

(main/src/com/complexible/pinto/RDFMapper.java)

```
@Test(expected = RDFMappingException.class)
@Ignore
public void testCharBeanTypeWithLongString() throws Exception {
}
```

8-ashoz hasonló

19. The utility class name 'Files3' doesn't match '[A-Z][a-zA-Z0-9]+(Utils?|Helper|Constants)'

(main/src/com/complexible/pinto/RDFMapper.java)

```
public static final class Files3 {
```

Elsőhöz hasonló

20. Avoid unnecessary constructors - the compiler will generate these for you

(main/src/com/complexible/pinto/RDFMapper.java)

```
public CannotConstructMe2() {
}
```

A problémát abban látja, hogy a konstruktort explicit fölösleges kiírni, ha az üres, csak növeli a kód nagyságát, a fordító enélkül is létrehozza. 5 perc változtatni ezen.

21. Unnecessary use of fully qualified name 'java.net.URI.create' due to existing import 'java.net.URI'

(test/src/com/complexible/pinto/RDFMapperTests.java)

7-eshez hasonló

```
106  aExpected.setURI(java.net.URI.create("urn:any"));
```

7-eshez hasonló

22. Document empty method body

(test/src/com/complexible/pinto/RDFMapperTests.java)

```
@Test
@Ignore
public void testReadEnumSet() throws Exception {
}
```

8-ashoz hasonló

23. Document empty method body

(test/src/com/complexible/pinto/RDFMapperTests.java)

```
@Test
@Ignore
public void testWriteEnumSet() throws Exception {
}
```

8-ashoz hasonló

24. Document empty method body

(test/src/com/complexible/pinto/RDFMapperTests.java)

```
@Test
@Ignore
public void testReadWithLangTag() throws Exception {
}
```

8-ashoz hasonló

25 Avoid really long classes.

(main/src/com/complexible/pinto/RDFMapper.java)

God class gyanús, és a komplexitása is rendkívül magas a többihez képest. Kb 1200 sor kód, lényegében az alkalmazás lényege ebben az osztályban történik.

Size		Structure		Complexity	
Lines of code:	1252	Number of Methods:	0	Complexity:	35
Source lines of code:	853	sLoC / Method:	0	Complexity / Method:	0
Commented lines of code:	224			Churn:	1

A fent említett osztályban több hiba is van, ezért azokat nem említeném meg felsorolás szerűen, hanem most egyben írnék róluk, a részletekbe belemenve.

Avoid really long classes.

```
95 public final class RDFMapper {
```

Avoid unnecessary if..then..else statements when returning booleans

```
206 if (thePropertyDescriptor.getName().equals("class")
```

The method 'readValue(Model, Class, Resource)' has an NPath complexity of 756

```
226 public <T> T readValue(final Model theGraph, final Class<T> theClass, final Resource theObj) {
```

Use one line for each declaration, it enhances code readability.

```
295 Object aKeyObj = null, aValueObj = null;
```

Avoid throwing raw exception types.

```
626 throw new RuntimeException("Unsupported or unknown literal datatype: " + aLit);
```

Mint fent is említettem ez egy ,,God Class” codesmell. Ez több szempontból is problémát okozhat, ezért érdemes lenne több kisebb osztályra bontani a felelősségek mentén Vannak hosszú függvények amik konverterként szolgálnak, azokat ki lehetne szervezni külön osztályba. Mint sokszor ilyen nagy osztályoknál, ez rengeteg if else ágat tartalmaz.

A második hiba ennél a szakasznál amit jelez, hogy egy if elágazásnak egyetlen művelete ha igaz a feltétel, hogy visszaadja, hogy igaz. Ez elhagyható lenne, és magát a feltétel kifejezést adná vissza. (Van mikor ez nem gond, mert a fordító magának már elhagyja, és emberi olvasás szempontjából hatékonyabb)

Harmadiknak a aciklikus végrehajtási utak magas számát jelzi (számszerint 756). Ez azonban téves jelzés.

Negyediknek egy sorban 4 változó deklarálás, nem probléma, sok rövidnél jobb, ha valami rendszer szerint egy sorba rendezettek.

Végül manuálisan nem ajánlja a raw típusú exception használatát, mert azzal általában a környezet él. Egyszerűen kicserélhető Exception-re például

A ,,God Class" jelzésen kívül mindegyik hiba javítható pár perc alatt, lényegében csak olvashatósághoz, értelmezés egyszerűségéhez kapcsolódó problémák, amik könnyen lehet hogy csak amiatt fordulnak elő, mert a fejelsztőcsapatnál ezek a standard irányelvek. Viszont az előbb említett kijavításához szükség van a projekt nagyobb átlátásához, a csapat olyan tagja(i) végezze, aki átlátja a kód egészét amennyire lehet.

26.

Javából még hiba, ami nincs megemlítve azok JUnit tesztekhez köthetők.

(/test/src/com/complexible/pinto/RDFMapperTests.java)

JUnit tests should include assert() or fail()

```
310 public void testURIMapping() throws Exception {
```

JUnit tests should include assert() or fail()

```
470 public void testProxyCreation() throws Exception {
```

JUnit tests should include assert() or fail()

```
647 public void testReadEnumSet() throws Exception {
```

JUnit tests should include assert() or fail()

```
652 public void testWriteEnumSet() throws Exception {
```

JUnit tests should include assert() or fail()

```
662 public void testWriteWithLangTag() throws Exception {
```

Üres tesztek, amik nem adnak vissza semmit. A Codacy nem vette figyelembe az ignore jelzést, ezért várta el tőlük. Hibás jelzés.

Vannak további jelzések a java kódhoz, de semmi olyan aminek a típusa nem lett volna említve.

Markdown language issue-k

A readme.md-ben találhatóak ezek

[fenced-code-flag] Missing code language flag

```
81 ```
```

[list-item-indent] Incorrect list-item indent: add 2 spaces

```
139 * `REQUIRE_IDS` - By default, Pinto will auto-generate URIs for objects when `@
```

[list-item-indent] Incorrect list-item indent: add 2 spaces

```
140 * `SERIALIZE_COLLECTIONS_AS_LISTS` - When true, collections are serialized as RD
```

[list-item-indent] Incorrect list-item indent: add 2 spaces

```
141 * `IGNORE_INVALID_ANNOTATIONS` - Whether or not to ignore an annotation which is
    F. (default: `true`)
```

[list-item-indent] Incorrect list-item indent: add 2 spaces

```
146 * `#map(URI, Class)` - Specify the provided type corresponds to instances of the
```

[list-item-indent] Incorrect list-item indent: add 2 spaces

```
147 * `#namespace(...)` - Methods to specify namespace mappings which are used to ex
```

[list-item-indent] Incorrect list-item indent: add 2 spaces

```
148 * `#valueFactory(ValueFactory)` - Provide the `ValueFactory` to be used when cre
```

[list-item-indent] Incorrect list-item indent: add 2 spaces

```
149 * `#collectionFactory(CollectionFactory)` - The factory to be used for creating
```

[list-item-indent] Incorrect list-item indent: add 2 spaces

```
150 * `#mapFactory(MapFactory)` - The factory to be used for creating instances of `
```

[no-heading-punctuation] Don't add a trailing `?` to headings

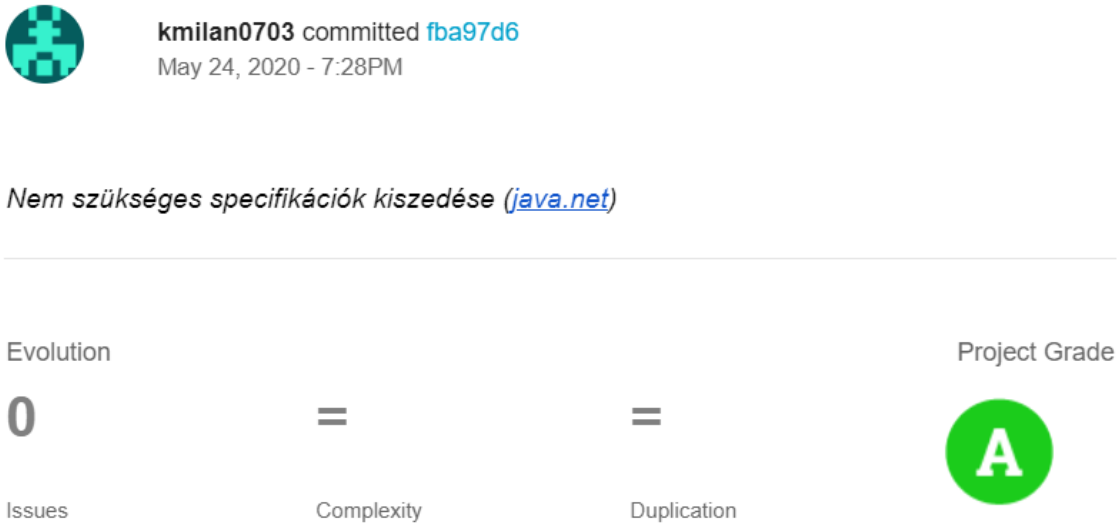
```
161 ## Why Pinto?
```

Ahogy a Codacy is jelezte, itt is csak kódolástechnikai problémák vannak, az összes javítása pár percet vesz igénybe. 8 intendálás probléma, hogy a lista elemeket bentebb kéne kezdeni.

Javítások:

8 intendálási probléma a readme.md-ben.

4 szükségtelen specifikáció kiszedés.



Mások commitja során keletkeztek még issues-okat javítottam.