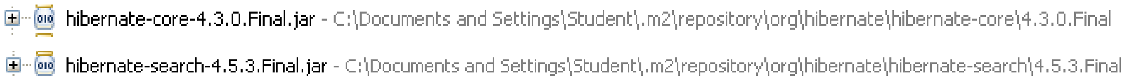
Report of A+

First of all, we looked for Hibernate Full – Text Search documentation and I found it in https://docs.jboss.org/hibernate/search/4.5/reference/en-US/html_single/#preface.

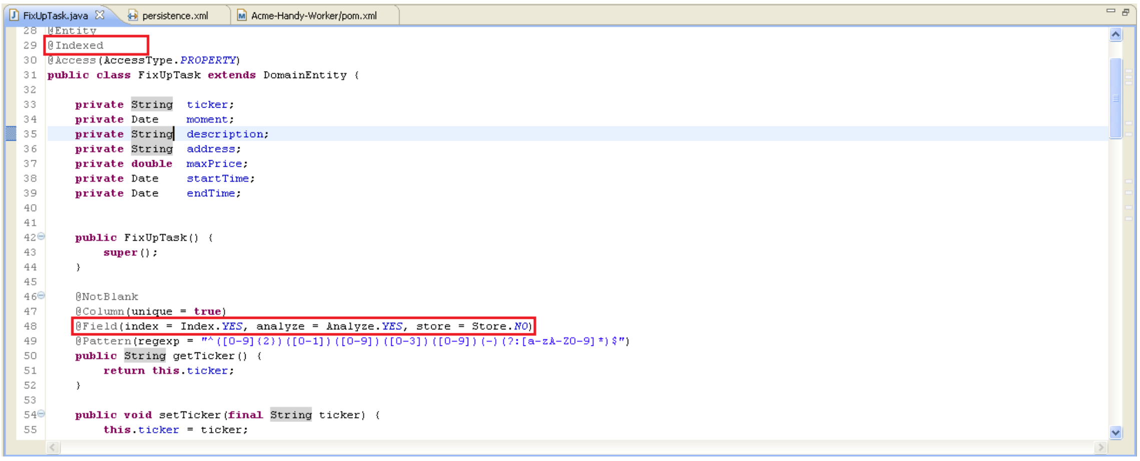
(We used 4.5.3 Final Version of Hibernate Search and 4.3.0 Version of Hibernate Core)



hibernate-core-4.3.0.Final.jar - C:\Documents and Settings\Student\.m2\repository\org\hibernate\hibernate-core\4.3.0.Final

hibernate-search-4.5.3.Final.jar - C:\Documents and Settings\Student\.m2\repository\org\hibernate\hibernate-search\4.5.3.Final

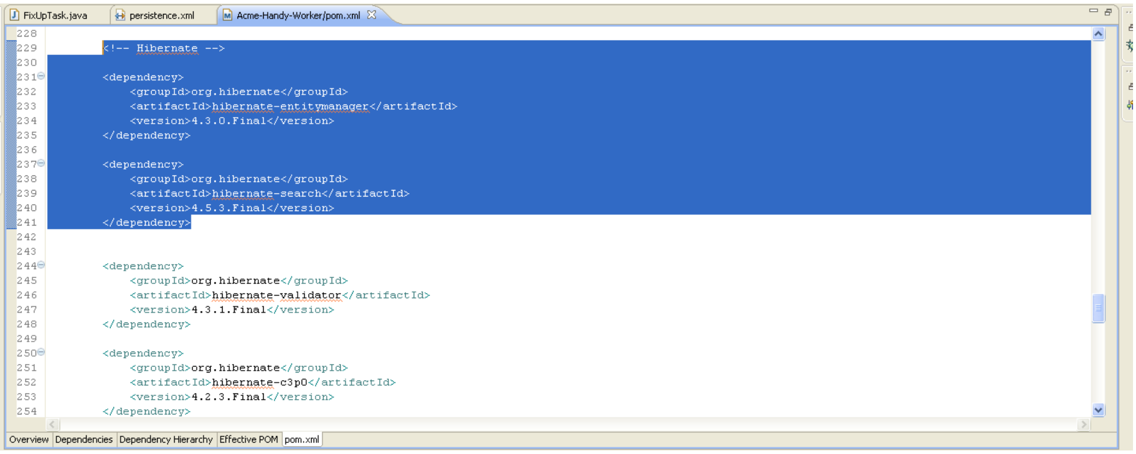
Then, we added the following annotations, properties and dependencies to the class FixUpTask, pom.xml and persistence.xml.



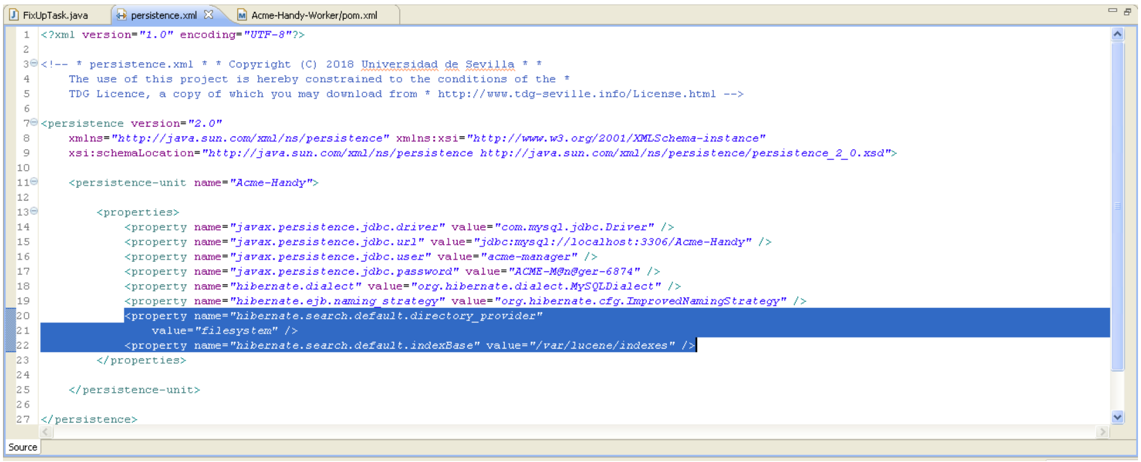
```
@Entity
@Indexed
@Access(AccessType.PROPERTY)
public class FixUpTask extends DomainEntity {

    private String  ticker;
    private Date    moment;
    private String  description;
    private String  address;
    private double  maxPrice;
    private Date    startTime;
    private Date    endTime;


    public FixUpTask() {
        super();
    }

    @NotBlank
    @Column(unique = true)
    @Field(index = Index.YES, analyze = Analyze.YES, store = Store.NO)
    @Pattern(regexp = "^([0-9]{2})([0-1])([0-9])([0-3])([0-9])(-)(?:[a-zA-Z0-9]*)$")
    public String getTicker() {
        return this.ticker;
    }

    public void setTicker(final String ticker) {
        this.ticker = ticker;
```

@Indexed marks FixUpTask as indexable.

@Field marks the field we want to make searchable (ticker, description and address in our case) . The parameter index=Index.YES will ensure that the text will be indexed, while analyze=Analyze.YES ensures that the text will be analyzed using the default Lucene analyzer. The third parameter we specify within @Field, store=Store.NO, ensures that the actual data will not be stored in the index.

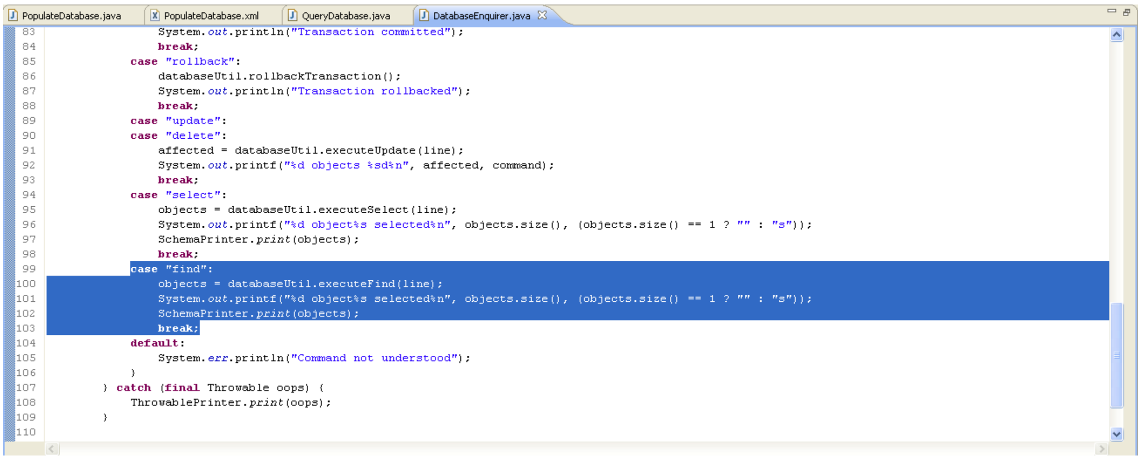


```xml

        <!-- Hibernate -->

        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-entitymanager</artifactId>
            <version>4.3.0.Final</version>
        </dependency>

        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-search</artifactId>
            <version>4.5.3.Final</version>
        </dependency>


        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-validator</artifactId>
            <version>4.3.1.Final</version>
        </dependency>

        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-c3p0</artifactId>
            <version>4.2.3.Final</version>
        </dependency>
```



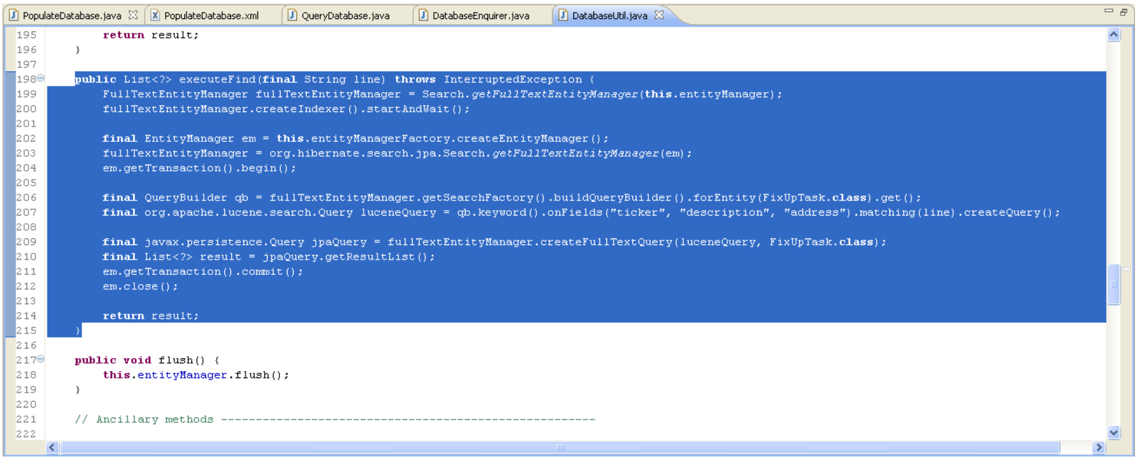
```xml
<?xml version="1.0" encoding="UTF-8"?>

<!-- * persistence.xml * * Copyright (C) 2018 Universidad de Sevilla * *
    The use of this project is hereby constrained to the conditions of the *
    TDG Licence, a copy of which you may download from * http://www.tdg-seville.info/License.html -->

<persistence version="2.0"
    xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">

    <persistence-unit name="Acme-Handy">

        <properties>
            <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver" />
            <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/Acme-Handy" />
            <property name="javax.persistence.jdbc.user" value="acme-manager" />
            <property name="javax.persistence.jdbc.password" value="ACME-M@n@ger-6874" />
            <property name="hibernate.dialect" value="org.hibernate.dialect.MySQLDialect" />
            <property name="hibernate.ejb.naming_strategy" value="org.hibernate.cfg.ImprovedNamingStrategy" />
            <property name="hibernate.search.default.directory_provider"
                value="filesystem" />
            <property name="hibernate.search.default.indexBase" value="/var/lucene/indexes" />
        </properties>

    </persistence-unit>

</persistence>
```

We analyzed **SchemaPrinter, ConsoleReader, QueryDatabase, DatabaseEnquirer** and **DatabaseUtil** classes and we got to understand which was the functioning.



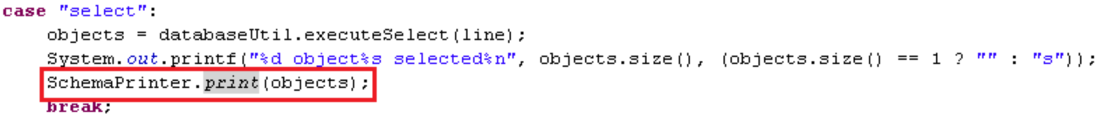
```java
                    System.out.println("Transaction commited");
                    break;
                case "rollback":
                    databaseUtil.rollbackTransaction();
                    System.out.println("Transaction rollbacked");
                    break;
                case "update":
                case "delete":
                    affected = databaseUtil.executeUpdate(line);
                    System.out.printf("%d objects %sd%n", affected, command);
                    break;
                case "select":
                    objects = databaseUtil.executeSelect(line);
                    System.out.printf("%d object%s selected%n", objects.size(), (objects.size() == 1 ? "" : "s"));
                    SchemaPrinter.print(objects);
                    break;
                case "find":
                    objects = databaseUtil.executeFind(line);
                    System.out.printf("%d object%s selected%n", objects.size(), (objects.size() == 1 ? "" : "s"));
                    SchemaPrinter.print(objects);
                    break;
                default:
                    System.err.println("Command not understood");
                }
            } catch (final Throwable oops) {
                ThrowablePrinter.print(oops);
            }

```

As we can see at the photo, we added a new case named **“find”** that is similar to the **“select”** case and we created a new method in DatabaseUtil class called executeFind.



```
        return result;
    }

    public List<?> executeFind(final String line) throws InterruptedException {
        FullTextEntityManager fullTextEntityManager = Search.getFullTextEntityManager(this.entityManager);
        fullTextEntityManager.createIndexer().startAndWait();

        final EntityManager em = this.entityManagerFactory.createEntityManager();
        fullTextEntityManager = org.hibernate.search.jpa.Search.getFullTextEntityManager(em);
        em.getTransaction().begin();

        final QueryBuilder qb = fullTextEntityManager.getSearchFactory().buildQueryBuilder().forEntity(FixUpTask.class).get();
        final org.apache.lucene.search.Query luceneQuery = qb.keyword().onFields("ticker", "description", "address").matching(line).createQuery();

        final javax.persistence.Query jpaQuery = fullTextEntityManager.createFullTextQuery(luceneQuery, FixUpTask.class);
        final List<?> result = jpaQuery.getResultList();
        em.getTransaction().commit();
        em.close();

        return result;
    }

    public void flush() {
        this.entityManager.flush();
    }

    // Ancillary methods -----------------------------------------------------
```

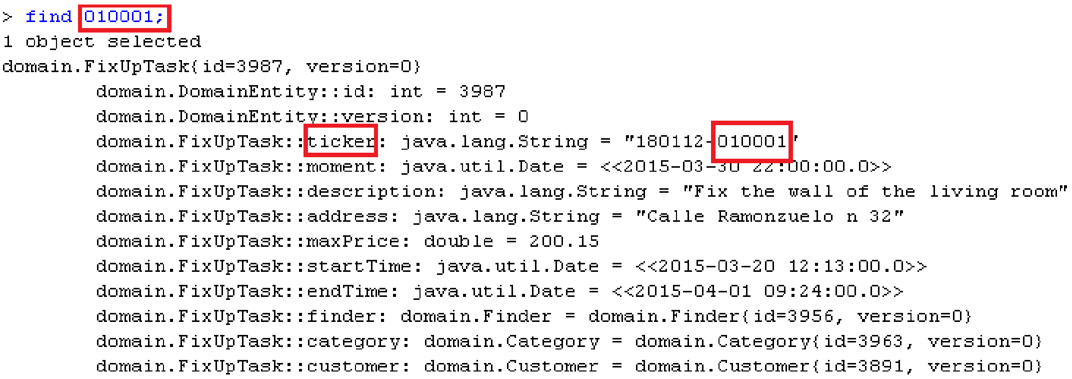
We catch the entire body of method in the URL above and we modified it a bit to adapt it to our case. The method receives a **String** as parameter, that will be the search´s parameter that we insert in the console. It results a List of objects that will be printed out by **SchemaPrinter.**



```
case "select":
    objects = databaseUtil.executeSelect(line);
    System.out.printf("%d object%s selected%n", objects.size(), (objects.size() == 1 ? "" : "s"));
    SchemaPrinter.print(objects);
    break;
```

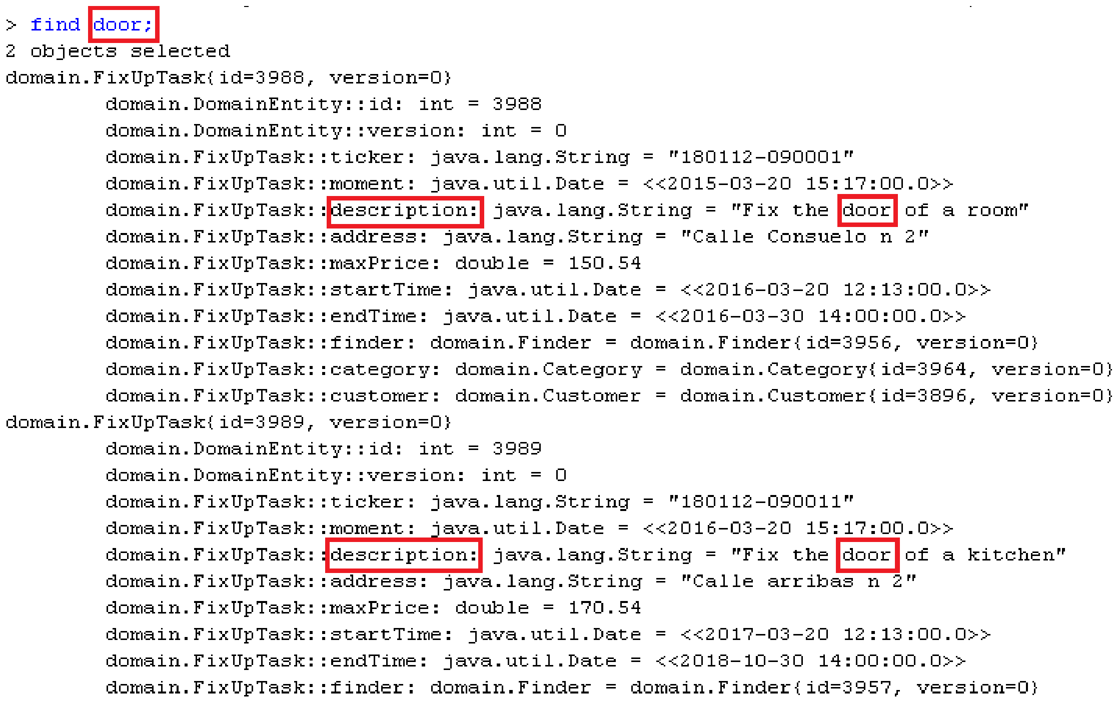
Now, it´s time to test it!

1. Ticker



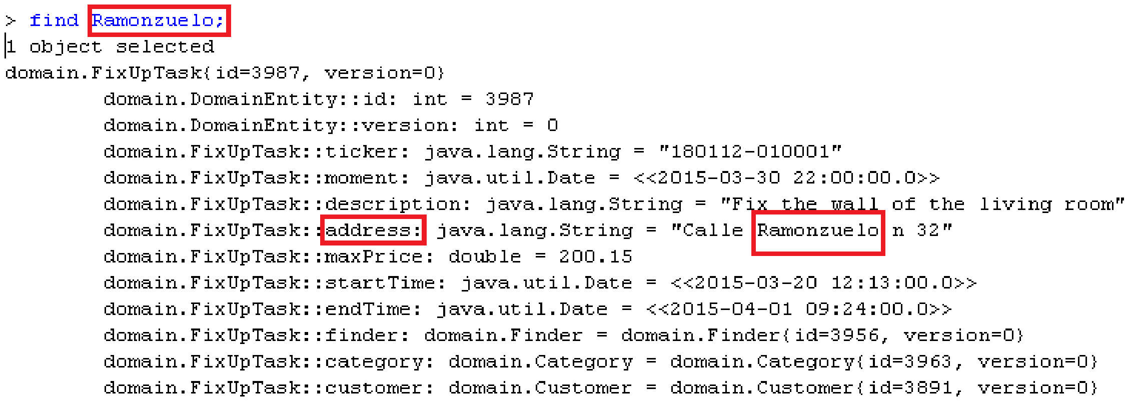
```
> find 010001;
1 object selected
domain.FixUpTask{id=3987, version=0}
        domain.DomainEntity::id: int = 3987
        domain.DomainEntity::version: int = 0
        domain.FixUpTask::ticker: java.lang.String = "180112-010001"
        domain.FixUpTask::moment: java.util.Date = <<2015-03-30 22:00:00.0>>
        domain.FixUpTask::description: java.lang.String = "Fix the wall of the living room"
        domain.FixUpTask::address: java.lang.String = "Calle Ramonzuelo n 32"
        domain.FixUpTask::maxPrice: double = 200.15
        domain.FixUpTask::startTime: java.util.Date = <<2015-03-20 12:13:00.0>>
        domain.FixUpTask::endTime: java.util.Date = <<2015-04-01 09:24:00.0>>
        domain.FixUpTask::finder: domain.Finder = domain.Finder{id=3956, version=0}
        domain.FixUpTask::category: domain.Category = domain.Category{id=3963, version=0}
        domain.FixUpTask::customer: domain.Customer = domain.Customer{id=3891, version=0}
```

2. Description



```
> find door;
2 objects selected
domain.FixUpTask{id=3988, version=0}
        domain.DomainEntity::id: int = 3988
        domain.DomainEntity::version: int = 0
        domain.FixUpTask::ticker: java.lang.String = "180112-090001"
        domain.FixUpTask::moment: java.util.Date = <<2015-03-20 15:17:00.0>>
        domain.FixUpTask::description: java.lang.String = "Fix the door of a room"
        domain.FixUpTask::address: java.lang.String = "Calle Consuelo n 2"
        domain.FixUpTask::maxPrice: double = 150.54
        domain.FixUpTask::startTime: java.util.Date = <<2016-03-20 12:13:00.0>>
        domain.FixUpTask::endTime: java.util.Date = <<2016-03-30 14:00:00.0>>
        domain.FixUpTask::finder: domain.Finder = domain.Finder{id=3956, version=0}
        domain.FixUpTask::category: domain.Category = domain.Category{id=3964, version=0}
        domain.FixUpTask::customer: domain.Customer = domain.Customer{id=3896, version=0}
domain.FixUpTask{id=3989, version=0}
        domain.DomainEntity::id: int = 3989
        domain.DomainEntity::version: int = 0
        domain.FixUpTask::ticker: java.lang.String = "180112-090011"
        domain.FixUpTask::moment: java.util.Date = <<2016-03-20 15:17:00.0>>
        domain.FixUpTask::description: java.lang.String = "Fix the door of a kitchen"
        domain.FixUpTask::address: java.lang.String = "Calle arribas n 2"
        domain.FixUpTask::maxPrice: double = 170.54
        domain.FixUpTask::startTime: java.util.Date = <<2017-03-20 12:13:00.0>>
        domain.FixUpTask::endTime: java.util.Date = <<2018-10-30 14:00:00.0>>
        domain.FixUpTask::finder: domain.Finder = domain.Finder{id=3957, version=0}
```

3. Address



```
> find Ramonzuelo;
1 object selected
domain.FixUpTask{id=3987, version=0}
        domain.DomainEntity::id: int = 3987
        domain.DomainEntity::version: int = 0
        domain.FixUpTask::ticker: java.lang.String = "180112-010001"
        domain.FixUpTask::moment: java.util.Date = <<2015-03-30 22:00:00.0>>
        domain.FixUpTask::description: java.lang.String = "Fix the wall of the living room"
        domain.FixUpTask::address: java.lang.String = "Calle Ramonzuelo n 32"
        domain.FixUpTask::maxPrice: double = 200.15
        domain.FixUpTask::startTime: java.util.Date = <<2015-03-20 12:13:00.0>>
        domain.FixUpTask::endTime: java.util.Date = <<2015-04-01 09:24:00.0>>
        domain.FixUpTask::finder: domain.Finder = domain.Finder{id=3956, version=0}
        domain.FixUpTask::category: domain.Category = domain.Category{id=3963, version=0}
        domain.FixUpTask::customer: domain.Customer = domain.Customer{id=3891, version=0}
```