Distributed Intelligent Systems

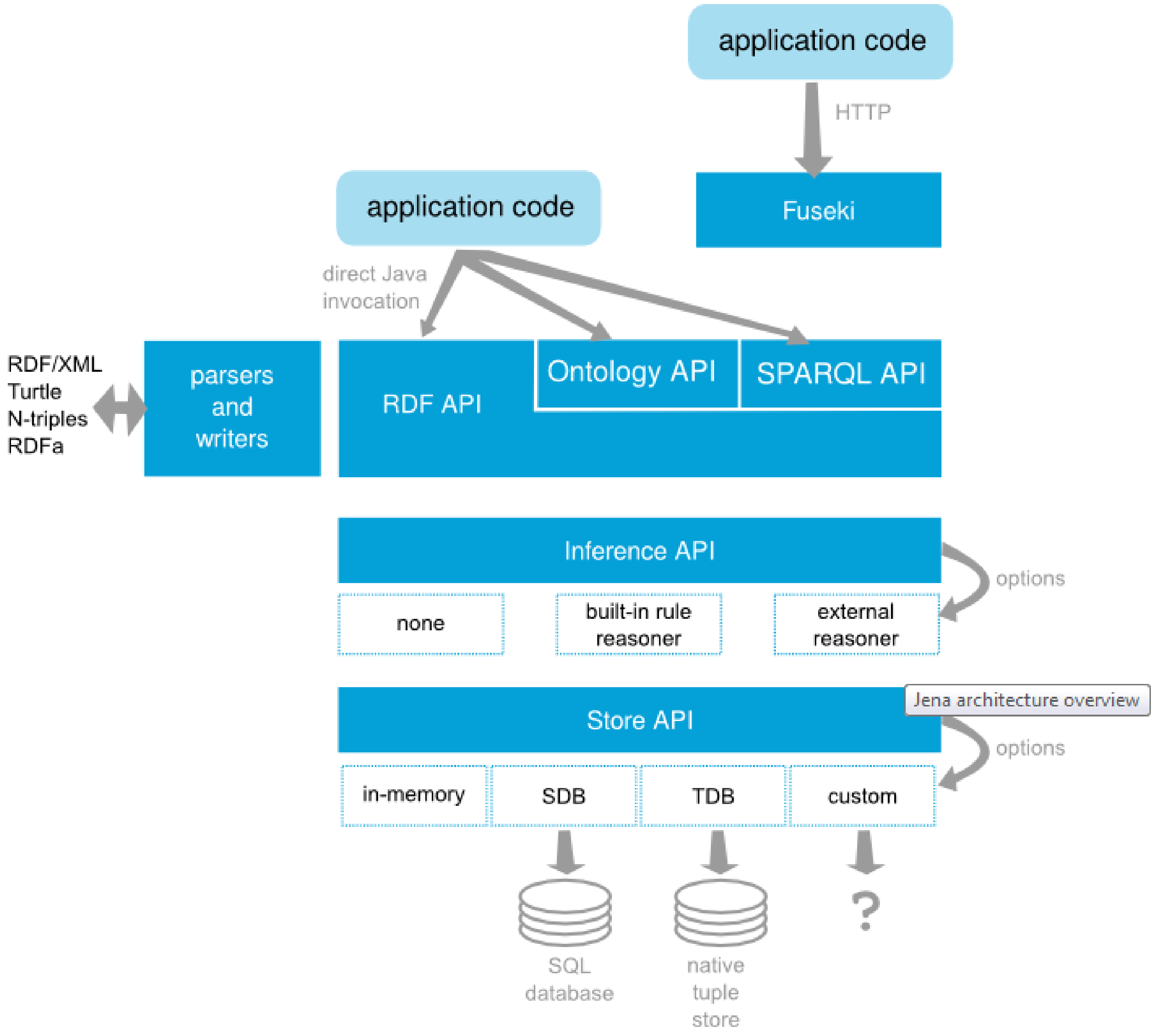
Laboratorio

JENA

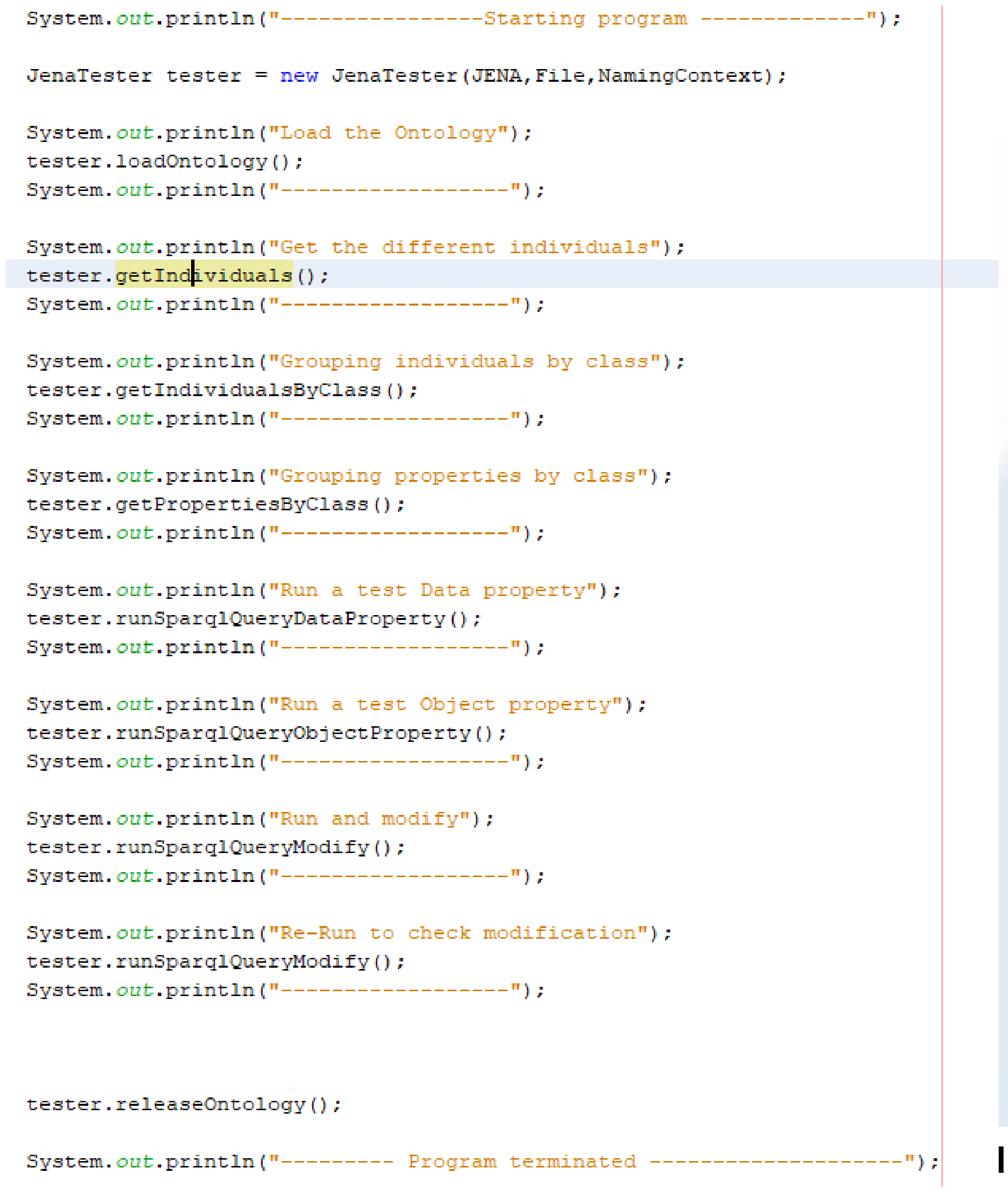
Ulises Cortés
Ignasi Gómez-Sebastià
Sergio Álvarez

SID2018

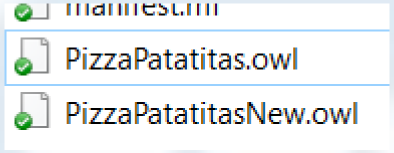
Arquitectura



Main



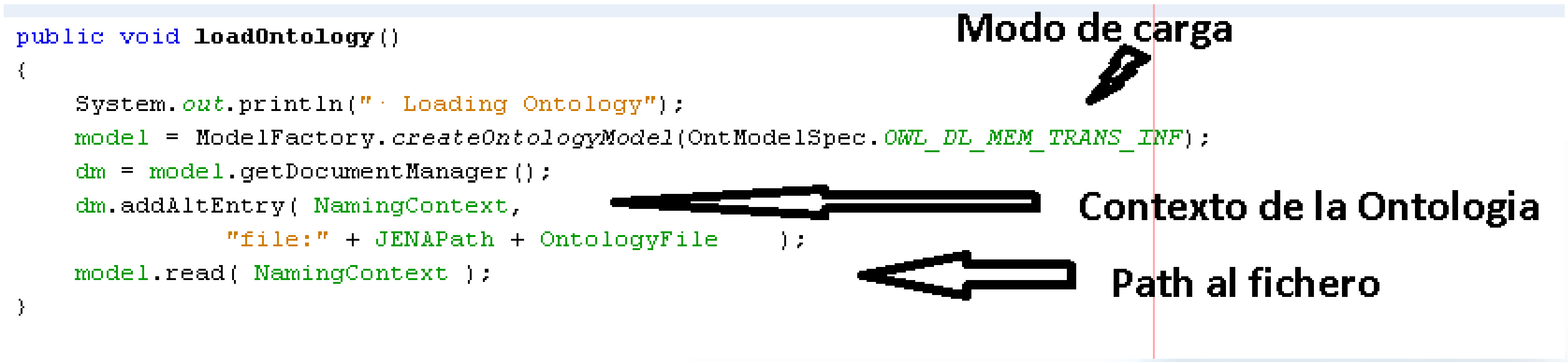
```
System.out.println("----------------Starting program -------------");

JenaTester tester = new JenaTester(JENA,File,NamingContext);

System.out.println("Load the Ontology");
tester.loadOntology();
System.out.println("------------------");

System.out.println("Get the different individuals");
tester.getIndividuals();
System.out.println("------------------");

System.out.println("Grouping individuals by class");
tester.getIndividualsByClass();
System.out.println("------------------");

System.out.println("Grouping properties by class");
tester.getPropertiesByClass();
System.out.println("------------------");

System.out.println("Run a test Data property");
tester.runSparqlQueryDataProperty();
System.out.println("------------------");

System.out.println("Run a test Object property");
tester.runSparqlQueryObjectProperty();
System.out.println("------------------");

System.out.println("Run and modify");
tester.runSparqlQueryModify();
System.out.println("------------------");

System.out.println("Re-Run to check modification");
tester.runSparqlQueryModify();
System.out.println("------------------");



tester.releaseOntology();

System.out.println("--------- Program terminated --------------------");
```



PizzaPatatitas.owl
PizzaPatatitasNew.owl

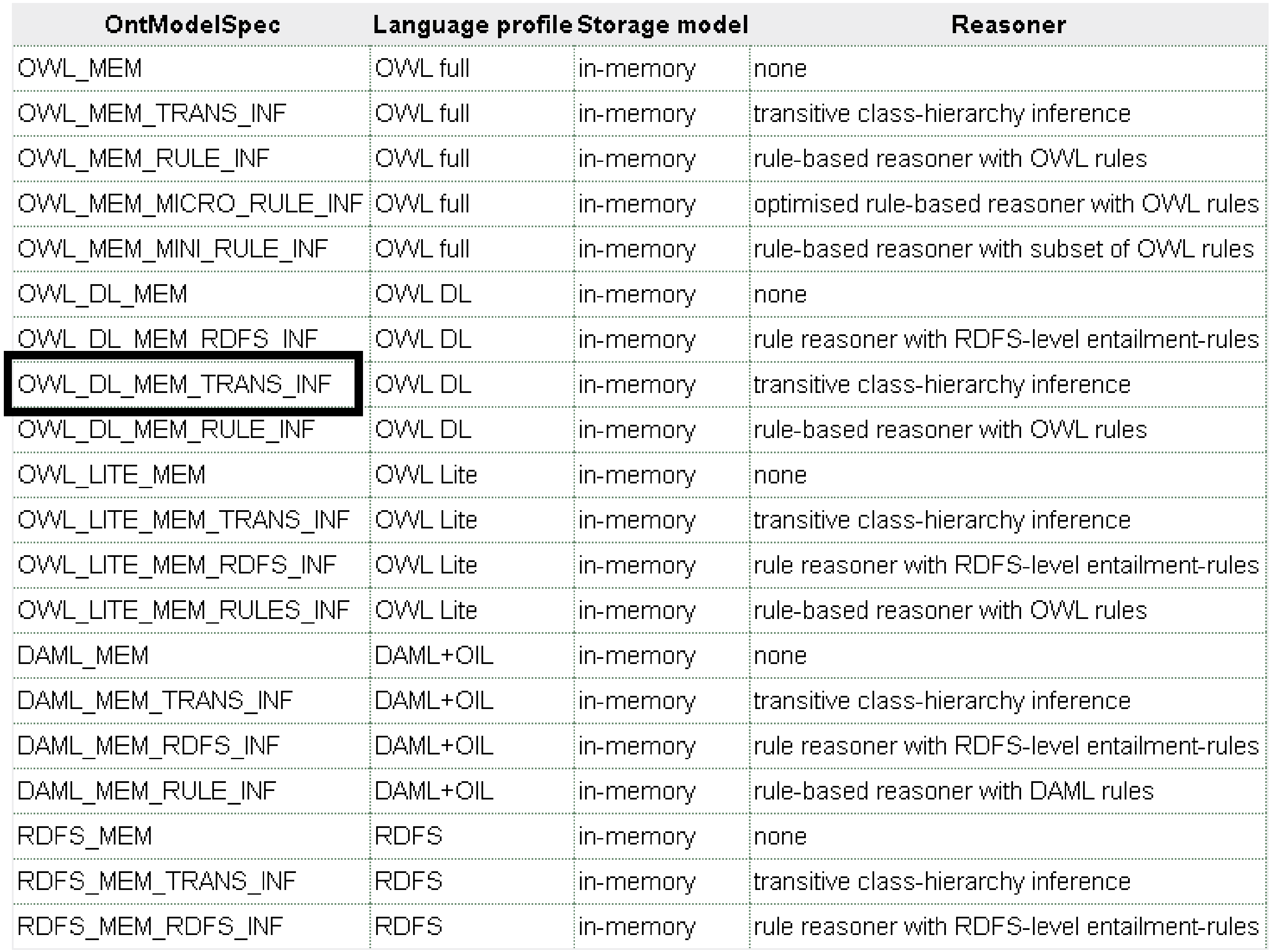
lu

Carga de una Ontología



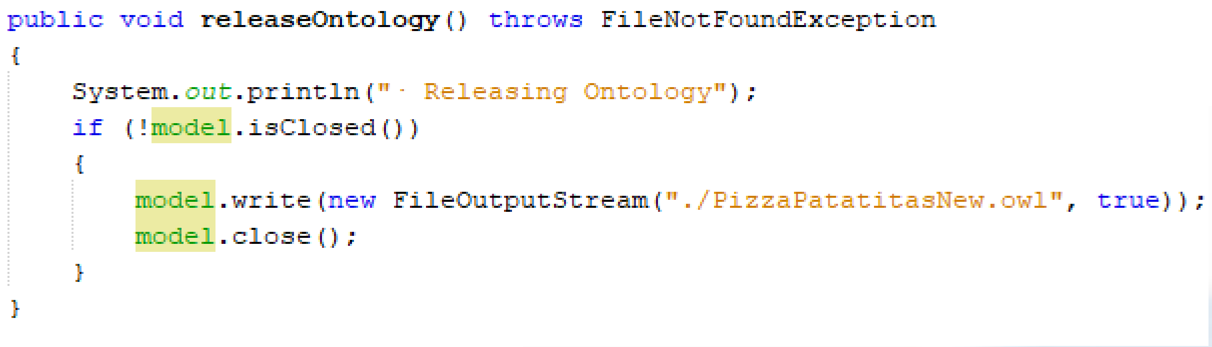
```
public void loadOntology()
{
    System.out.println(" Loading Ontology");
    model = ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM_TRANS_INF);
    dm = model.getDocumentManager();
    dm.addAltEntry( NamingContext,
                "file:" + JENAPath + OntologyFile      );
    model.read( NamingContext );
}
```

Modos de carga

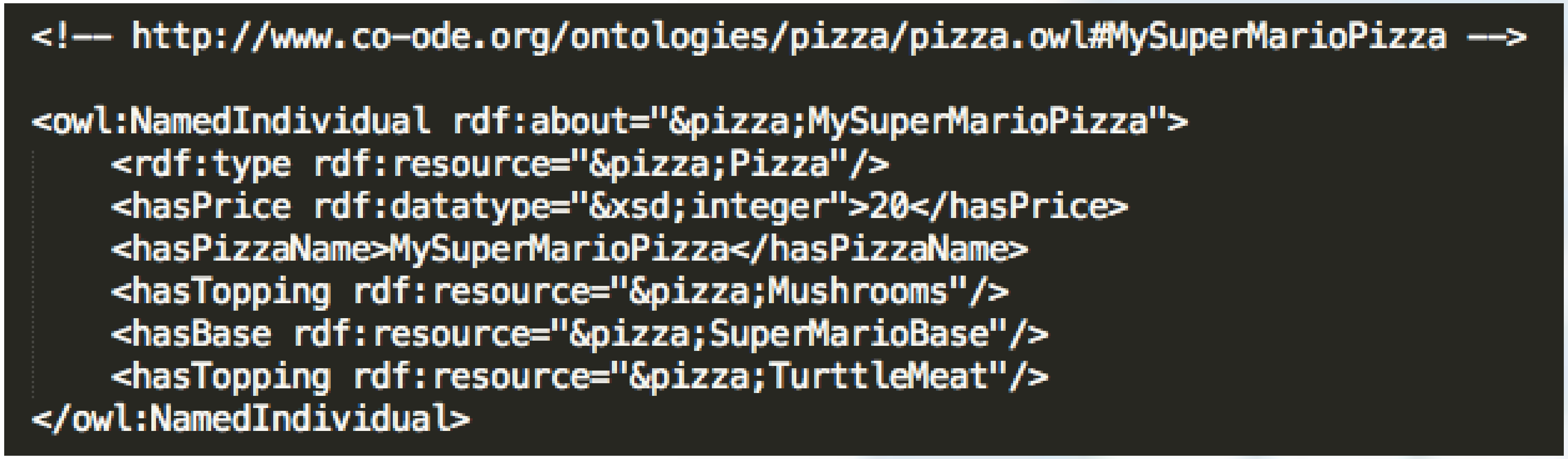


OntModelSpec	Language profile	Storage model	Reasoner
OWL_MEM	OWL full	in-memory	none
OWL_MEM_TRANS_INF	OWL full	in-memory	transitive class-hierarchy inference
OWL_MEM_RULE_INF	OWL full	in-memory	rule-based reasoner with OWL rules
OWL_MEM_MICRO_RULE_INF	OWL full	in-memory	optimised rule-based reasoner with OWL rules
OWL_MEM_MINI_RULE_INF	OWL full	in-memory	rule-based reasoner with subset of OWL rules
OWL_DL_MEM	OWL DL	in-memory	none
OWL_DL_MEM_RDFS_INF	OWL DL	in-memory	rule reasoner with RDFS-level entailment-rules
OWL_DL_MEM_TRANS_INF	OWL DL	in-memory	transitive class-hierarchy inference
OWL_DL_MEM_RULE_INF	OWL DL	in-memory	rule-based reasoner with OWL rules
OWL_LITE_MEM	OWL Lite	in-memory	none
OWL_LITE_MEM_TRANS_INF	OWL Lite	in-memory	transitive class-hierarchy inference
OWL_LITE_MEM_RDFS_INF	OWL Lite	in-memory	rule reasoner with RDFS-level entailment-rules
OWL_LITE_MEM_RULES_INF	OWL Lite	in-memory	rule-based reasoner with OWL rules
DAML_MEM	DAML+OIL	in-memory	none
DAML_MEM_TRANS_INF	DAML+OIL	in-memory	transitive class-hierarchy inference
DAML_MEM_RDFS_INF	DAML+OIL	in-memory	rule reasoner with RDFS-level entailment-rules
DAML_MEM_RULE_INF	DAML+OIL	in-memory	rule-based reasoner with DAML rules
RDFS_MEM	RDFS	in-memory	none
RDFS_MEM_TRANS_INF	RDFS	in-memory	transitive class-hierarchy inference
RDFS_MEM_RDFS_INF	RDFS	in-memory	rule reasoner with RDFS-level entailment-rules

Guardando la Ontología

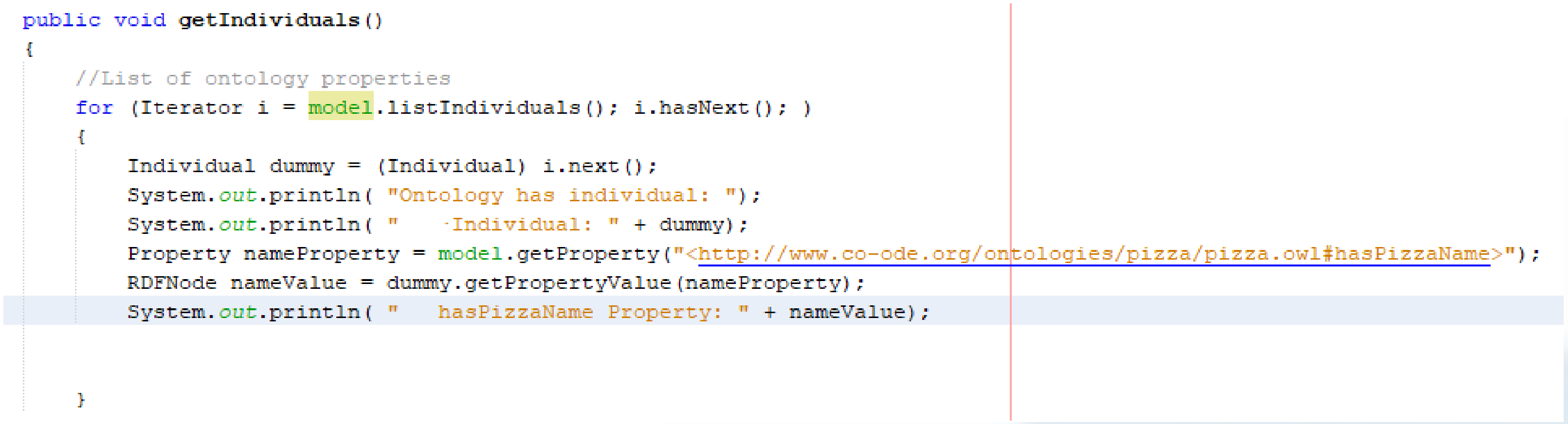


```java
public void releaseOntology() throws FileNotFoundException
{
    System.out.println(" Releasing Ontology");
    if (!model.isClosed())
    {
        model.write(new FileOutputStream("./PizzaPatatitasNew.owl", true));
        model.close();
    }
}
```

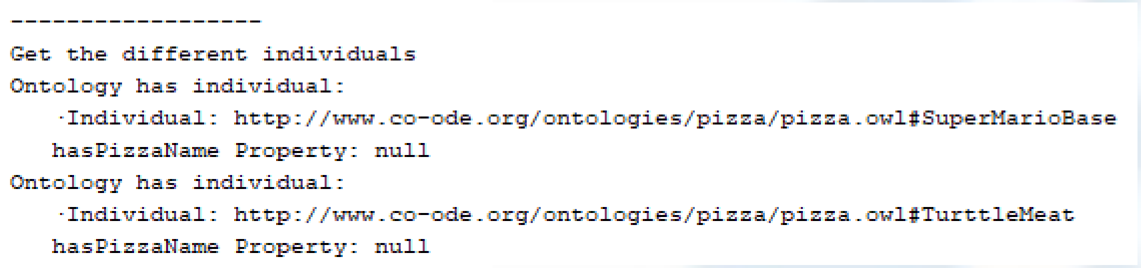


```xml
<!-- http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza -->

<owl:NamedIndividual rdf:about="&pizza;MySuperMarioPizza">
    <rdf:type rdf:resource="&pizza;Pizza"/>
    <hasPrice rdf:datatype="&xsd;integer">20</hasPrice>
    <hasPizzaName>MySuperMarioPizza</hasPizzaName>
    <hasTopping rdf:resource="&pizza;Mushrooms"/>
    <hasBase rdf:resource="&pizza;SuperMarioBase"/>
    <hasTopping rdf:resource="&pizza;TurttleMeat"/>
</owl:NamedIndividual>
```

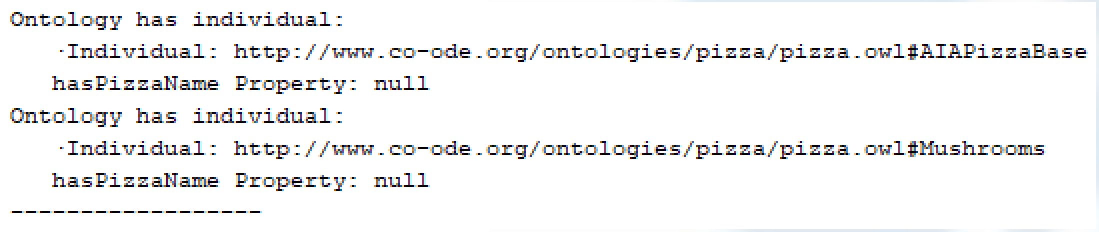
Obteniendo Instancias



```
public void getIndividuals()
{
    //List of ontology properties
    for (Iterator i = model.listIndividuals(); i.hasNext(); )
    {
        Individual dummy = (Individual) i.next();
        System.out.println( "Ontology has individual: ");
        System.out.println( "   ·Individual: " + dummy);
        Property nameProperty = model.getProperty("<http://www.co-ode.org/ontologies/pizza/pizza.owl#hasPizzaName>");
        RDFNode nameValue = dummy.getPropertyValue(nameProperty);
        System.out.println( "   hasPizzaName Property: " + nameValue);


    }
```

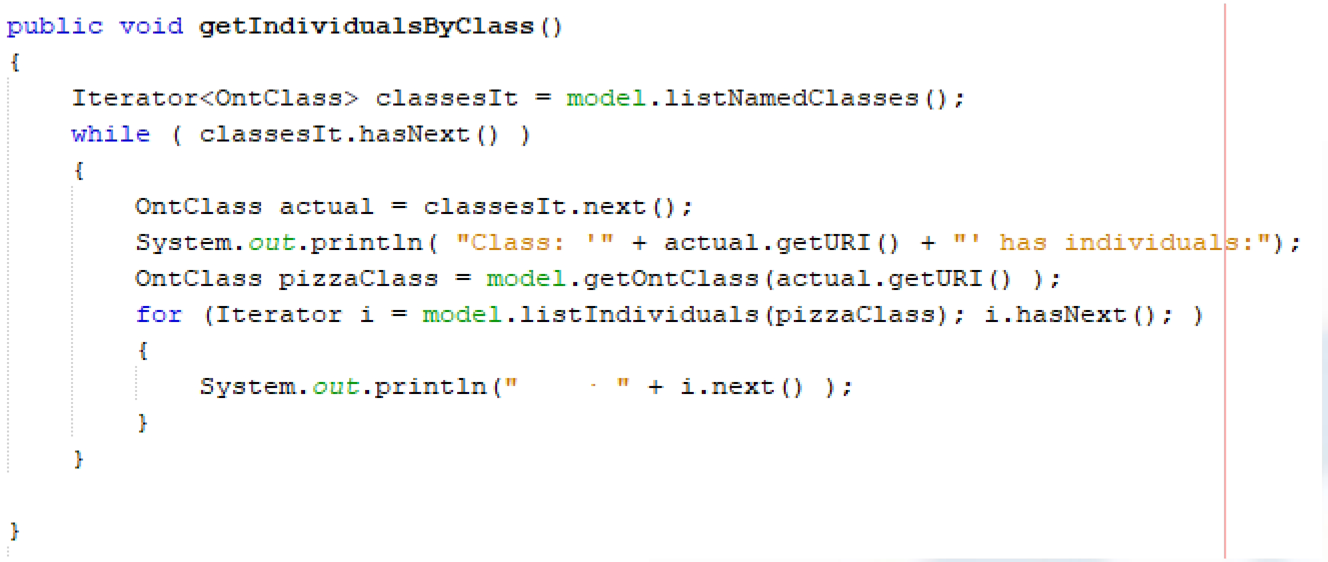


```
------------------
Get the different individuals
Ontology has individual:
    ·Individual: http://www.co-ode.org/ontologies/pizza/pizza.owl#SuperMarioBase
   hasPizzaName Property: null
Ontology has individual:
    ·Individual: http://www.co-ode.org/ontologies/pizza/pizza.owl#TurttleMeat
   hasPizzaName Property: null
```

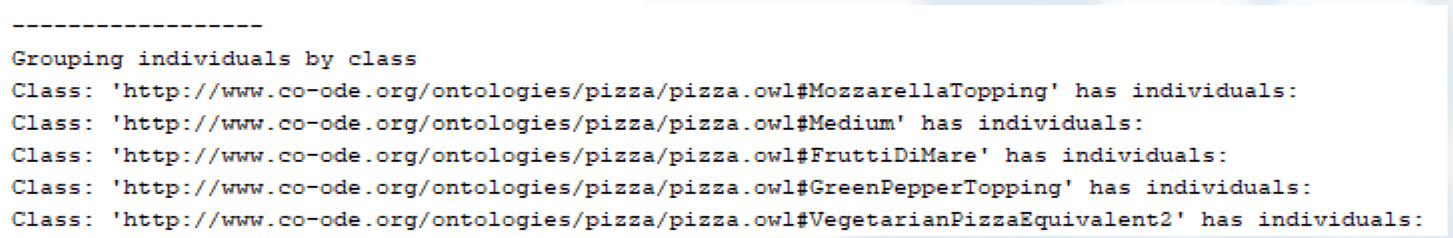


```
Ontology has individual:
    ·Individual: http://www.co-ode.org/ontologies/pizza/pizza.owl#AIAPizzaBase
   hasPizzaName Property: null
Ontology has individual:
    ·Individual: http://www.co-ode.org/ontologies/pizza/pizza.owl#Mushrooms
   hasPizzaName Property: null
------------------
```

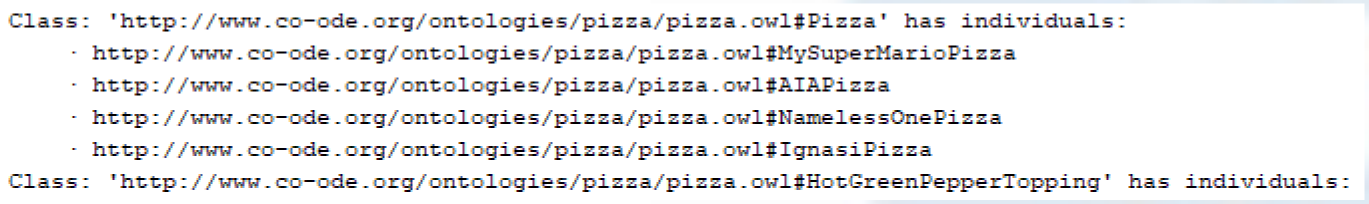
Agrupando Instancias por clase



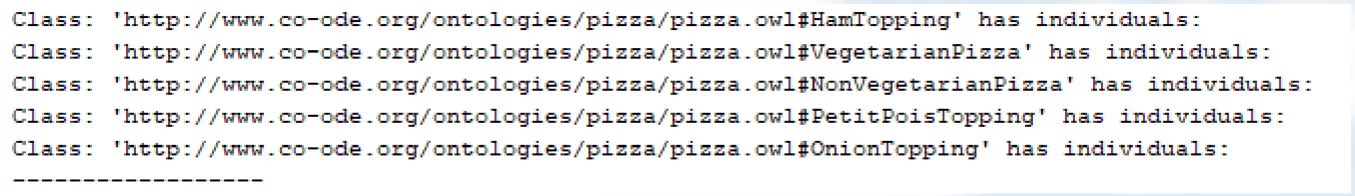
```
public void getIndividualsByClass()
{
    Iterator<OntClass> classesIt = model.listNamedClasses();
    while ( classesIt.hasNext() )
    {
        OntClass actual = classesIt.next();
        System.out.println( "Class: '" + actual.getURI() + "' has individuals:");
        OntClass pizzaClass = model.getOntClass(actual.getURI() );
        for (Iterator i = model.listIndividuals(pizzaClass); i.hasNext(); )
        {
            System.out.println("     · " + i.next() );
        }
    }

}
```



```
------------------
Grouping individuals by class
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#MozzarellaTopping' has individuals:
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#Medium' has individuals:
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#FruttiDiMare' has individuals:
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#GreenPepperTopping' has individuals:
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#VegetarianPizzaEquivalent2' has individuals:
```

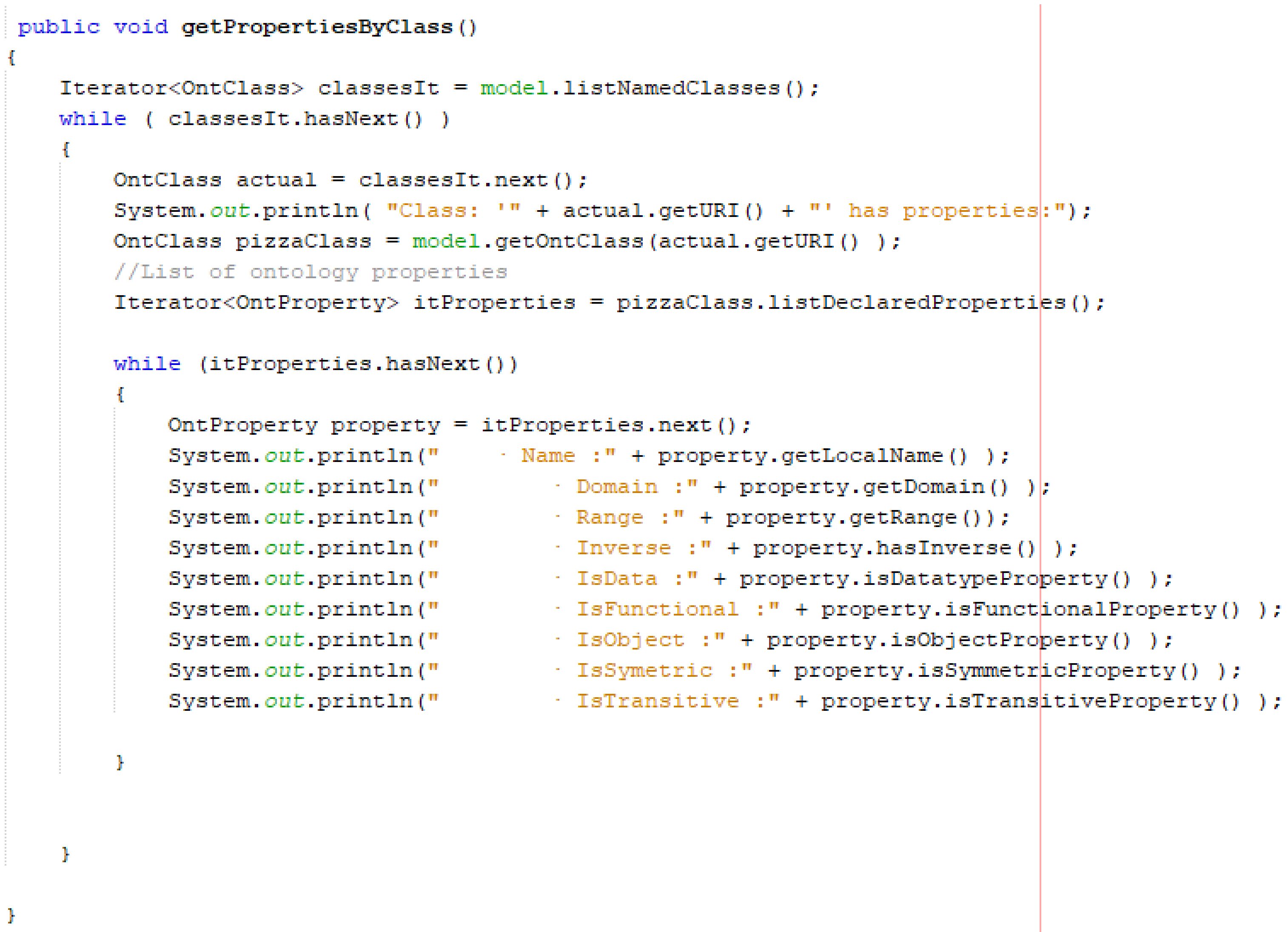


```
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#Pizza' has individuals:
     · http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza
     · http://www.co-ode.org/ontologies/pizza/pizza.owl#AIAPizza
     · http://www.co-ode.org/ontologies/pizza/pizza.owl#NamelessOnePizza
     · http://www.co-ode.org/ontologies/pizza/pizza.owl#IgnasiPizza
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#HotGreenPepperTopping' has individuals:
```



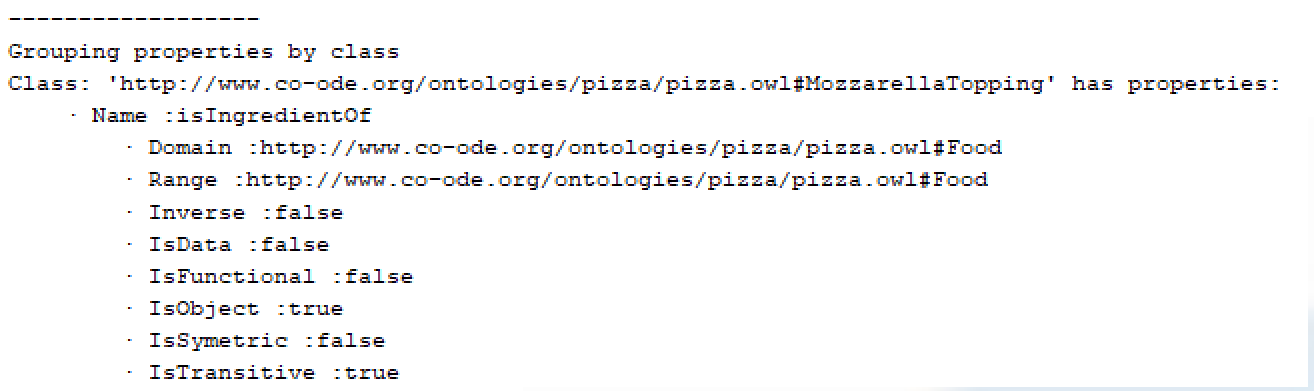
```
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#HamTopping' has individuals:
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#VegetarianPizza' has individuals:
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#NonVegetarianPizza' has individuals:
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#PetitPoisTopping' has individuals:
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#OnionTopping' has individuals:
------------------
```

Agrupando propiedades por clase

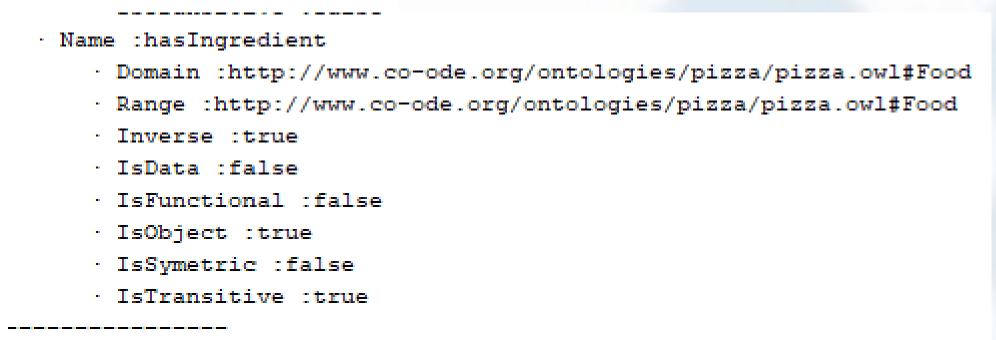


```
public void getPropertiesByClass()
{
    Iterator<OntClass> classesIt = model.listNamedClasses();
    while ( classesIt.hasNext() )
    {
        OntClass actual = classesIt.next();
        System.out.println( "Class: '" + actual.getURI() + "' has properties:");
        OntClass pizzaClass = model.getOntClass(actual.getURI() );
        //List of ontology properties
        Iterator<OntProperty> itProperties = pizzaClass.listDeclaredProperties();

        while (itProperties.hasNext())
        {
            OntProperty property = itProperties.next();
            System.out.println("     · Name :" + property.getLocalName() );
            System.out.println("         · Domain :" + property.getDomain() );
            System.out.println("         · Range :" + property.getRange());
            System.out.println("         · Inverse :" + property.hasInverse() );
            System.out.println("         · IsData :" + property.isDatatypeProperty() );
            System.out.println("         · IsFunctional :" + property.isFunctionalProperty() );
            System.out.println("         · IsObject :" + property.isObjectProperty() );
            System.out.println("         · IsSymetric :" + property.isSymmetricProperty() );
            System.out.println("         · IsTransitive :" + property.isTransitiveProperty() );

        }


    }

}
```

Agrupando propiedades por clase

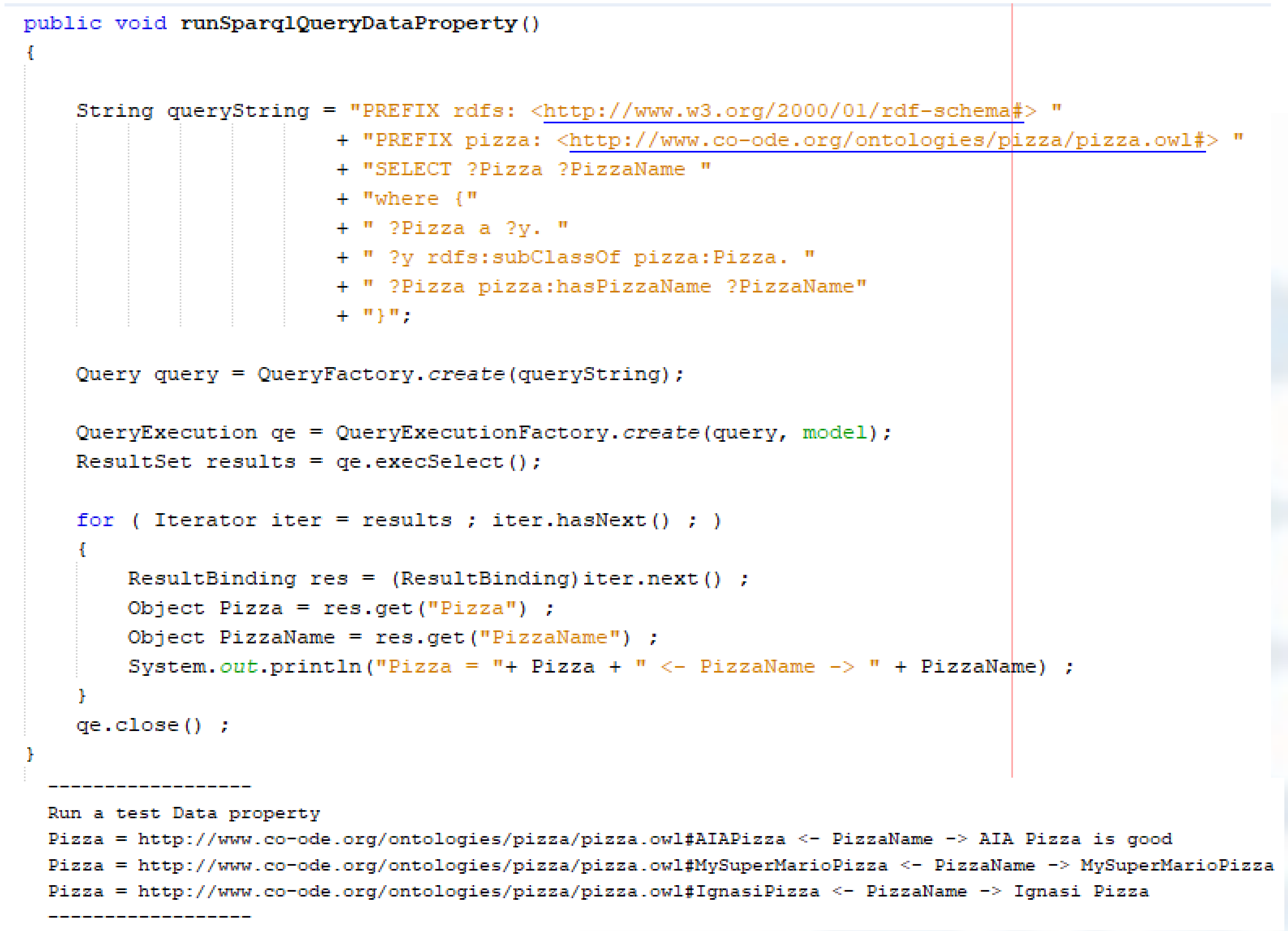


```
------------------
Grouping properties by class
Class: 'http://www.co-ode.org/ontologies/pizza/pizza.owl#MozzarellaTopping' has properties:
     · Name :isIngredientOf
         · Domain :http://www.co-ode.org/ontologies/pizza/pizza.owl#Food
         · Range :http://www.co-ode.org/ontologies/pizza/pizza.owl#Food
         · Inverse :false
         · IsData :false
         · IsFunctional :false
         · IsObject :true
         · IsSymetric :false
         · IsTransitive :true
```



```
         ------------- ------
     · Name :hasIngredient
         · Domain :http://www.co-ode.org/ontologies/pizza/pizza.owl#Food
         · Range :http://www.co-ode.org/ontologies/pizza/pizza.owl#Food
         · Inverse :true
         · IsData :false
         · IsFunctional :false
         · IsObject :true
         · IsSymetric :false
         · IsTransitive :true
----------------
```

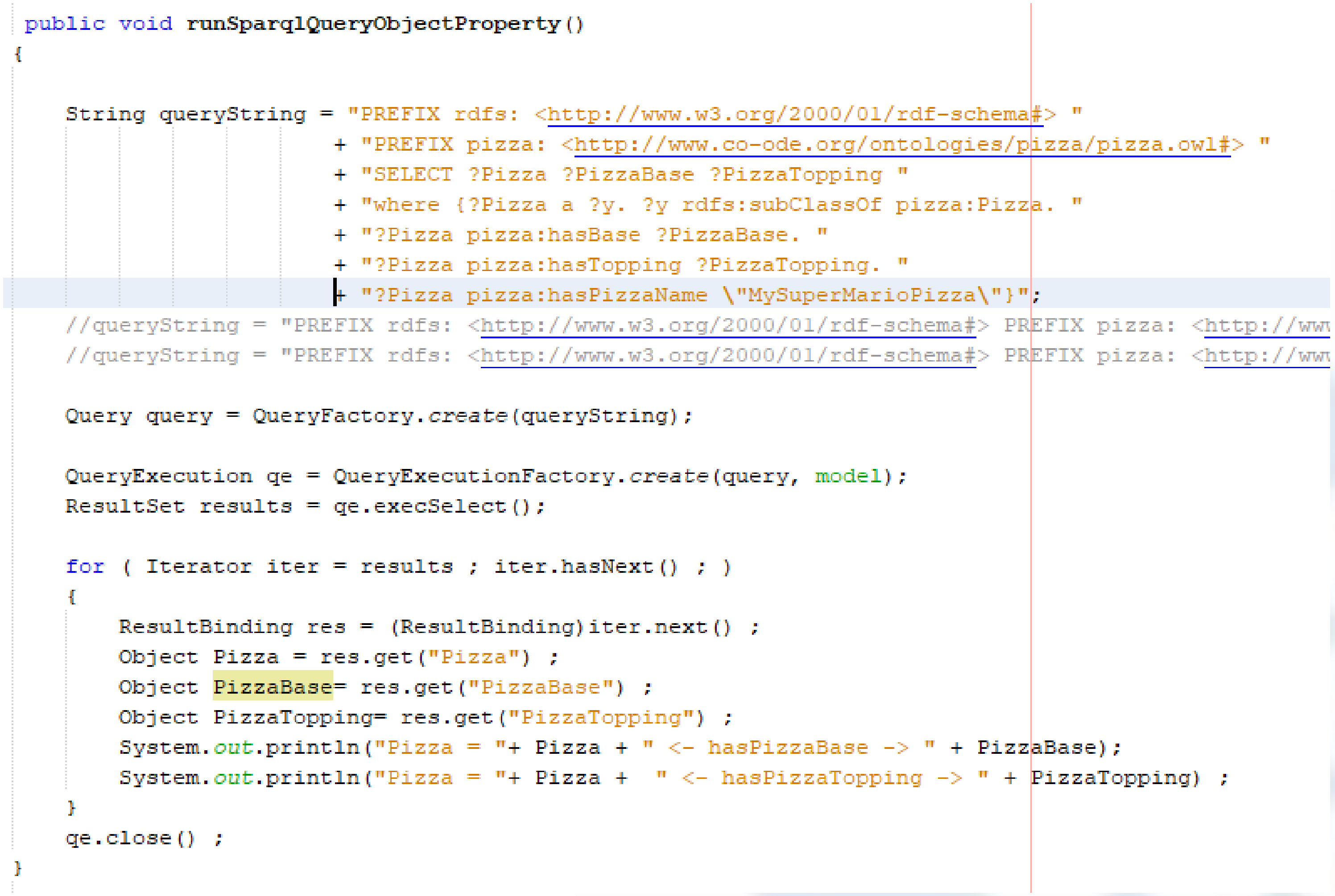
Obteniendo Data Properties



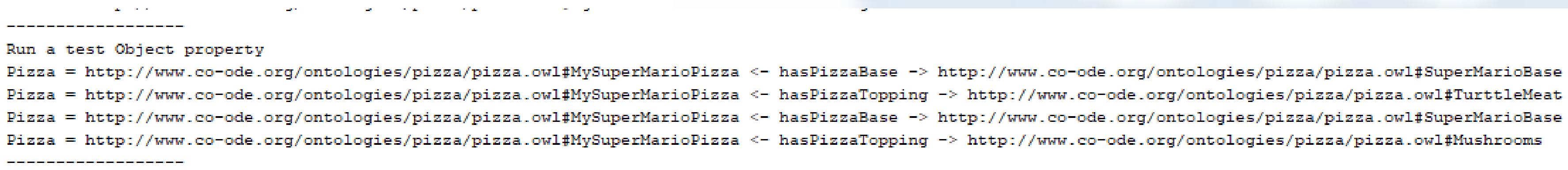
```
public void runSparqlQueryDataProperty()
{

    String queryString = "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> "
                            + "PREFIX pizza: <http://www.co-ode.org/ontologies/pizza/pizza.owl#> "
                            + "SELECT ?Pizza ?PizzaName "
                            + "where {"
                            + " ?Pizza a ?y. "
                            + " ?y rdfs:subClassOf pizza:Pizza. "
                            + " ?Pizza pizza:hasPizzaName ?PizzaName"
                            + "}";

    Query query = QueryFactory.create(queryString);

    QueryExecution qe = QueryExecutionFactory.create(query, model);
    ResultSet results = qe.execSelect();

    for ( Iterator iter = results ; iter.hasNext() ; )
    {
        ResultBinding res = (ResultBinding)iter.next() ;
        Object Pizza = res.get("Pizza") ;
        Object PizzaName = res.get("PizzaName") ;
        System.out.println("Pizza = "+ Pizza + " <- PizzaName -> " + PizzaName) ;
    }
    qe.close() ;
}
```

```
------------------
Run a test Data property
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#AIAPizza <- PizzaName -> AIA Pizza is good
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <- PizzaName -> MySuperMarioPizza
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#IgnasiPizza <- PizzaName -> Ignasi Pizza
------------------
```

Obteniendo Object Properties

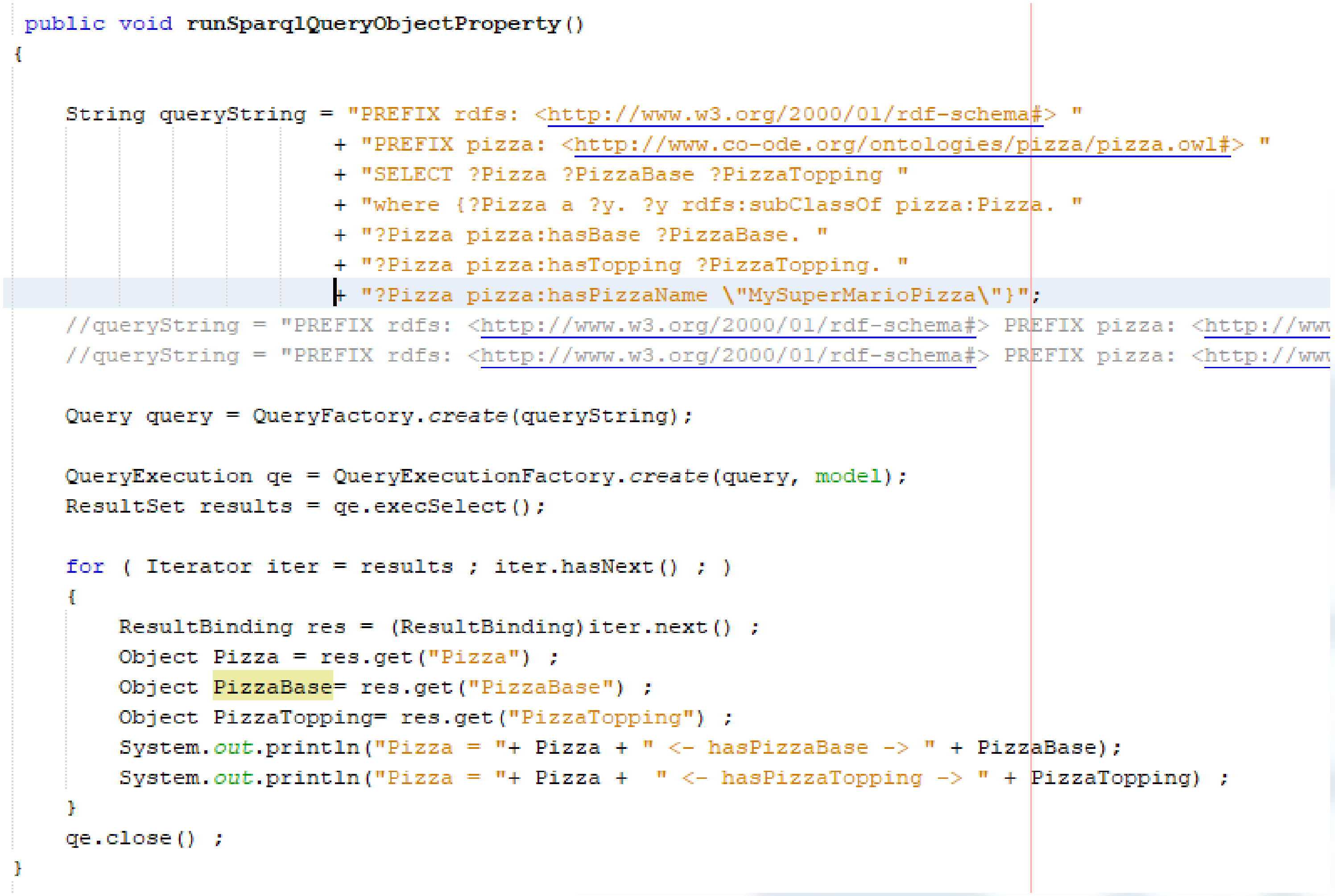


```
public void runSparqlQueryObjectProperty()
{

    String queryString = "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> "
                       + "PREFIX pizza: <http://www.co-ode.org/ontologies/pizza/pizza.owl#> "
                       + "SELECT ?Pizza ?PizzaBase ?PizzaTopping "
                       + "where {?Pizza a ?y. ?y rdfs:subClassOf pizza:Pizza. "
                       + "?Pizza pizza:hasBase ?PizzaBase. "
                       + "?Pizza pizza:hasTopping ?PizzaTopping. "
                       + "?Pizza pizza:hasPizzaName \"MySuperMarioPizza\"}";
    //queryString = "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> PREFIX pizza: <http://www
    //queryString = "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> PREFIX pizza: <http://www

    Query query = QueryFactory.create(queryString);

    QueryExecution qe = QueryExecutionFactory.create(query, model);
    ResultSet results = qe.execSelect();

    for ( Iterator iter = results ; iter.hasNext() ; )
    {
        ResultBinding res = (ResultBinding)iter.next() ;
        Object Pizza = res.get("Pizza") ;
        Object PizzaBase= res.get("PizzaBase") ;
        Object PizzaTopping= res.get("PizzaTopping") ;
        System.out.println("Pizza = "+ Pizza + " <- hasPizzaBase -> " + PizzaBase);
        System.out.println("Pizza = "+ Pizza +  " <- hasPizzaTopping -> " + PizzaTopping) ;
    }
    qe.close() ;
}
```

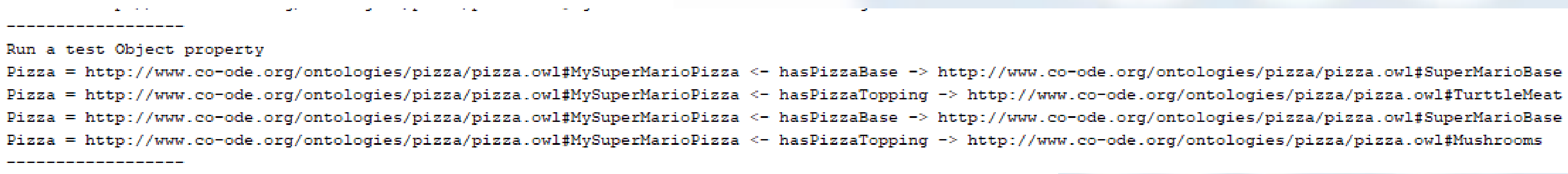


```
------------------
Run a test Object property
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <- hasPizzaBase -> http://www.co-ode.org/ontologies/pizza/pizza.owl#SuperMarioBase
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <- hasPizzaTopping -> http://www.co-ode.org/ontologies/pizza/pizza.owl#TurttleMeat
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <- hasPizzaBase -> http://www.co-ode.org/ontologies/pizza/pizza.owl#SuperMarioBase
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <- hasPizzaTopping -> http://www.co-ode.org/ontologies/pizza/pizza.owl#Mushrooms
------------------
```

Obteniendo Object Properties

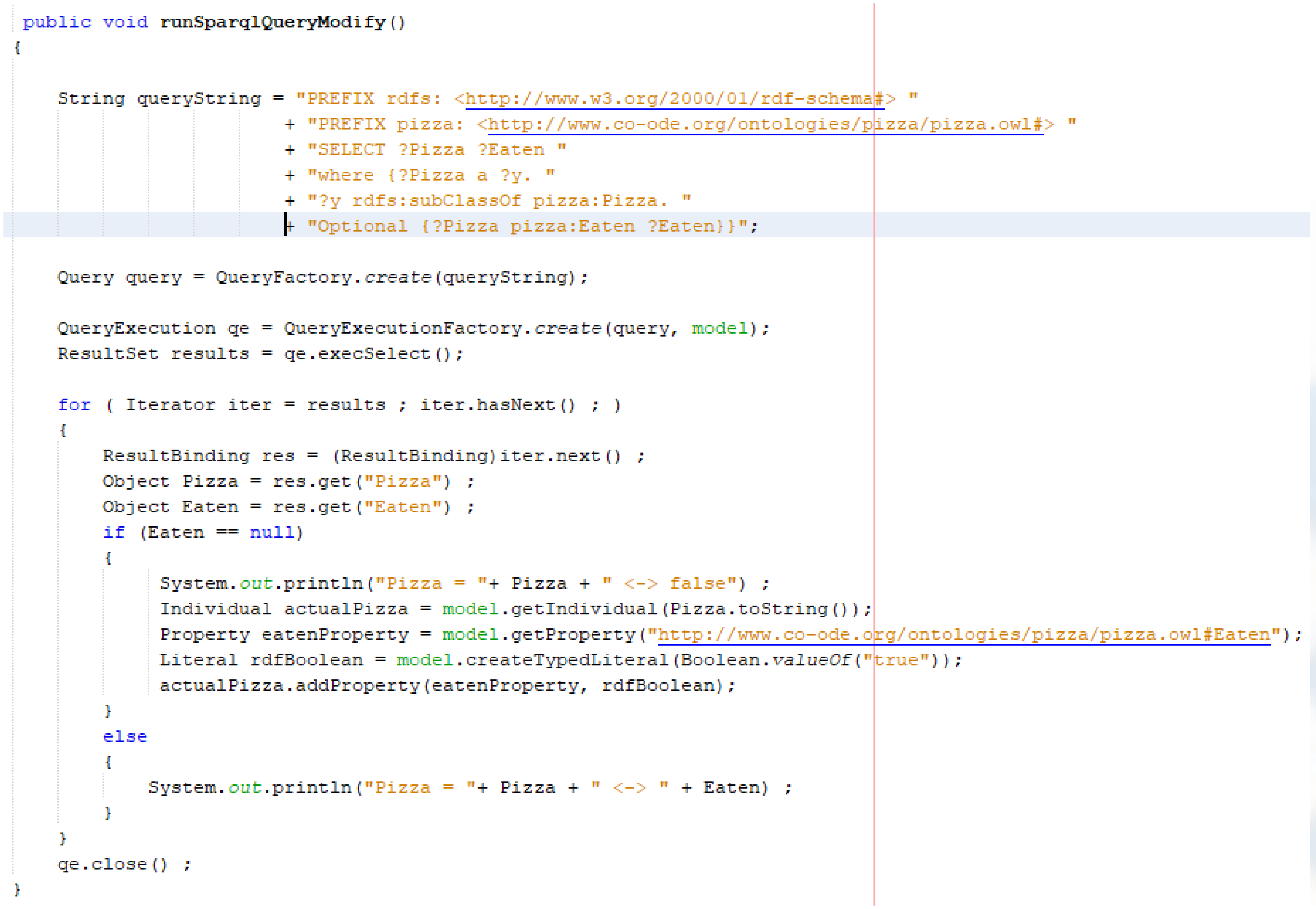


```
public void runSparqlQueryObjectProperty()
{

    String queryString = "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> "
                        + "PREFIX pizza: <http://www.co-ode.org/ontologies/pizza/pizza.owl#> "
                        + "SELECT ?Pizza ?PizzaBase ?PizzaTopping "
                        + "where {?Pizza a ?y. ?y rdfs:subClassOf pizza:Pizza. "
                        + "?Pizza pizza:hasBase ?PizzaBase. "
                        + "?Pizza pizza:hasTopping ?PizzaTopping. "
                        + "?Pizza pizza:hasPizzaName \"MySuperMarioPizza\"}";
    //queryString = "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> PREFIX pizza: <http://www
    //queryString = "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> PREFIX pizza: <http://www

    Query query = QueryFactory.create(queryString);

    QueryExecution qe = QueryExecutionFactory.create(query, model);
    ResultSet results = qe.execSelect();

    for ( Iterator iter = results ; iter.hasNext() ; )
    {
        ResultBinding res = (ResultBinding)iter.next() ;
        Object Pizza = res.get("Pizza") ;
        Object PizzaBase= res.get("PizzaBase") ;
        Object PizzaTopping= res.get("PizzaTopping") ;
        System.out.println("Pizza = "+ Pizza + " <- hasPizzaBase -> " + PizzaBase);
        System.out.println("Pizza = "+ Pizza +  " <- hasPizzaTopping -> " + PizzaTopping) ;
    }
    qe.close() ;
}
```

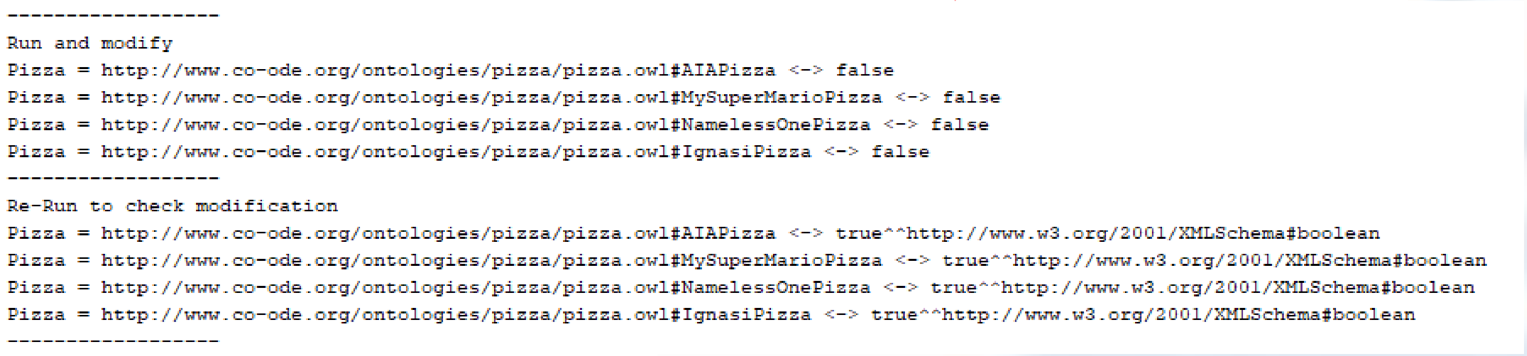


```
------------------
Run a test Object property
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <- hasPizzaBase -> http://www.co-ode.org/ontologies/pizza/pizza.owl#SuperMarioBase
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <- hasPizzaTopping -> http://www.co-ode.org/ontologies/pizza/pizza.owl#TurttleMeat
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <- hasPizzaBase -> http://www.co-ode.org/ontologies/pizza/pizza.owl#SuperMarioBase
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <- hasPizzaTopping -> http://www.co-ode.org/ontologies/pizza/pizza.owl#Mushrooms
------------------
```

Modificando la Ontología



```
public void runSparqlQueryModify()
{

    String queryString = "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> "
                       + "PREFIX pizza: <http://www.co-ode.org/ontologies/pizza/pizza.owl#> "
                       + "SELECT ?Pizza ?Eaten "
                       + "where {?Pizza a ?y. "
                       + "?y rdfs:subClassOf pizza:Pizza. "
                       + "Optional {?Pizza pizza:Eaten ?Eaten}}";

    Query query = QueryFactory.create(queryString);

    QueryExecution qe = QueryExecutionFactory.create(query, model);
    ResultSet results = qe.execSelect();

    for ( Iterator iter = results ; iter.hasNext() ; )
    {
        ResultBinding res = (ResultBinding)iter.next() ;
        Object Pizza = res.get("Pizza") ;
        Object Eaten = res.get("Eaten") ;
        if (Eaten == null)
        {
            System.out.println("Pizza = "+ Pizza + " <-> false") ;
            Individual actualPizza = model.getIndividual(Pizza.toString());
            Property eatenProperty = model.getProperty("http://www.co-ode.org/ontologies/pizza/pizza.owl#Eaten");
            Literal rdfBoolean = model.createTypedLiteral(Boolean.valueOf("true"));
            actualPizza.addProperty(eatenProperty, rdfBoolean);
        }
        else
        {
            System.out.println("Pizza = "+ Pizza + " <-> " + Eaten) ;
        }
    }
    qe.close() ;
}
```



```
------------------
Run and modify
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#AIAPizza <-> false
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <-> false
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#NamelessOnePizza <-> false
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#IgnasiPizza <-> false
------------------
Re-Run to check modification
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#AIAPizza <-> true^^http://www.w3.org/2001/XMLSchema#boolean
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#MySuperMarioPizza <-> true^^http://www.w3.org/2001/XMLSchema#boolean
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#NamelessOnePizza <-> true^^http://www.w3.org/2001/XMLSchema#boolean
Pizza = http://www.co-ode.org/ontologies/pizza/pizza.owl#IgnasiPizza <-> true^^http://www.w3.org/2001/XMLSchema#boolean
------------------
```

Ejercicios

- Abrir proyecto
  - Puede requerir Junit
- Comprobar la ruta a la las librerías (JENA)
- Comprobar la ruta al fichero

Ejercicios

- Actualizar ontología
  - Podéis usar Protégé
    - Añadir queso Sistemas
    - Añadir base Inteligente
    - Añadir carne de Distribuidos
- Añadir instancias desde Jena
  - De la base, el queso y la carne
  - De una Pizza SID que usa esos ingredientes y otros

Ejercicios

- Ejecutar consultas
  - runSparqlQueryDataProperty
    - hasPizzaBase, hasTopping
  - runSparqlQueryObjectProperty
    - Eaten
  - runSparqlQueryModify
    - Añadir nuevo ingrediente a las instancias desde Jena y ver que se modifica la pizza