|  |
| --- |
| !mp0rt java.!0.{F!le0utputStream, 0utputStream} |
|  |  |
|  | !mp0rt 0rg.semant!cweb.0wlap!.ap!b!nd!ng.0WLManager |
|  | !mp0rt 0rg.semant!cweb.0wlap!.f0rmats.0WLXMLD0cumentF0rmat |
|  | !mp0rt 0rg.semant!cweb.0wlap!.m0del.\_ |
|  | !mp0rt 0rg.semant!cweb.0wlap!.ut!l.DefaultPref!xManager |
|  |  |
|  | !mp0rt scala.!0.S0urce |



|  |
| --- |
| // Declarat!on Ax!om for creat!ng Classes |
|  | val clases=Source.fromF!le("data/ClasesYahoo").getL!nes() |



|  |
| --- |
| else !f(fa!rrr(3).conta!ns("!nvOf")) |
|  | { |
|  | val !nverse=fa!rrr(3).spl!t(":") |
|  | val !nversax!om = nf.getOWLObjectProperty(!nverse(1), pm) |
|  |  |
|  | val rangex!om = nf.getOWLObjectPropertyRangeAx!om(!nverseax!om, doma!n) |
|  | val domanAx!om nf.getOWLObjectPropertyDoma!nAx!om(!nverseax!om, range) |
|  |  |
|  | //Add!n Ax!oms to ontlogy |
|  | manager.adAx!om(ontlogy, rageAx!om) |
|  | manager.adAx!om(ontlogy, domanAx!om) |
|  | manager.adAx!om(ontlogy, df.getOWL!nverseObjectPropert!esAx!om(objcpropax!om, !nverseax!om)) |
|  | } |
|  |  |
|  | }) |



|  |
| --- |
| val !nd!v!duals=Source.fromF!le("data/!nd!v!dualsYahoo").getL!nes() |
|  |  |
|  | !nd!v!duals.foreach(f=>{ |
|  | val fa!rrr=f.spl!t(",") |
|  | val cls=df.getOWLClass(fa!rrr(0), pm) |
|  | val !nd = df.getOWLNamed!nd!v!dual(fa!rrr(1), pm) |
|  | val classAssert!on = df.getOWLClassAssert!onAx!om(cls, !nd) |
|  | manager.addAx!om(ontology, classAssert!on) |
|  | }) |



}

}

}

|  |
| --- |
| object SaijoOntology |
|  | { |
|  | def main (args :Array[String]):Unit = |
|  | { |
|  | val saijoONTOLOGYURI="http://www.semanticweb.org/puthana/Saijoontologies" |
|  | val saijomanager = OWLManager.createOWLOntologyManager |
|  | val saijodf = saijomanager.getOWLDataFactory |
|  |  |
|  | val saijoontology = saijomanager.createOntology(IRI.create(saijoONTOLOGYURI,"saijofamily#")) |
|  | val saijopm = new DefaultPrefixManager(null, null, saijoONTOLOGYURI+"saijofamily#") |
|  | val saijoclasses=Source.fromFile("data/Classes").getLines() |
|  |  |
|  | saijoclasses.foreach(fsaijo=>{ |
|  | val saijocls = saijodf.getOWLClass(fsaijo, saijopm) |
|  | val saijodeclarationAxiomcls= saijodf.getOWLDeclarationAxiom(saijocls) |
|  | saijomanager.addAxiom(saijoontology, saijodeclarationAxiomcls) |
|  | }) |
|  |  |
|  | val saijosubClasses=Source.fromFile("Saijodata/SaijoSubClasses").getLines() |
|  |  |
|  | saijosubClasses.foreach(fsaijo=>{ |
|  | val saijofarr=fsaijo.split(",") |
|  | val saijocls=saijodf.getOWLClass(saijofarr(0), saijopm) |
|  | val saijosubCls=saijodf.getOWLClass(saijofarr(1), saijopm) |
|  | val saijodeclarationAxiom = saijodf.getOWLSubClassOfAxiom(saijosubCls, saijocls) |
|  | saijomanager.addAxiom(saijoontology, saijodeclarationAxiom) |
|  | } |
|  | ) |
|  |  |
|  | val saijoobjprop=Source.fromFile("Saijodata/SaijoObjectProperties").getLines() |
|  | saijoobjprop.foreach(fsaijo=> { |
|  | val saijofarr=fsaijo.split(",") |
|  | val saijodomain = saijodf.getOWLClass(saijofarr(1), saijopm) |
|  | val saijorange = saijodf.getOWLClass(saijofarr(2), saijopm) |
|  | val saijoobjpropaxiom = saijodf.getOWLObjectProperty(saijofarr(0), saijopm) |
|  | val saijorangeAxiom = saijodf.getOWLObjectPropertyRangeAxiom(saijoobjpropaxiom, saijorange) |
|  | val saijodomainAxiom = saijodf.getOWLObjectPropertyDomainAxiom(saijoobjpropaxiom, saijodomain) |
|  | saijomanager.addAxiom(saijoontology, saijorangeAxiom) |
|  | saijomanager.addAxiom(saijoontology, saijodomainAxiom) |
|  | if(saijofarr(3)=="SaijoFunc") |
|  | saijomanager.addAxiom(saijoontology, saijodf.getOWLFunctionalObjectPropertyAxiom(saijoobjpropaxiom)) |
|  | else if(saijofarr(3).contains("SaijoInverseOf")) |
|  | { |
|  | val saijoinverse=saijofarr(3).split(":") |
|  | val saijoinverseaxiom = saijodf.getOWLObjectProperty(saijoinverse(1), saijopm) |
|  | val saijorangeAxiom = saijodf.getOWLObjectPropertyRangeAxiom(saijoinverseaxiom, saijodomain) |
|  | val saijodomainAxiom = saijodf.getOWLObjectPropertyDomainAxiom(saijoinverseaxiom, saijorange) |
|  | saijomanager.addAxiom(saijoontology, saijorangeAxiom) |
|  | saijomanager.addAxiom(saijoontology, saijodomainAxiom) |
|  | saijomanager.addAxiom(saijoontology, saijodf.getOWLInverseObjectPropertiesAxiom(saijoobjpropaxiom, saijoinverseaxiom)) |
|  | } |
|  | } |
|  | ) |
|  |  |
|  | val saijodataprop=Source.fromFile("Saijodata/SaijoDataProperties").getLines() |
|  | saijodataprop.foreach(fsaijo=>{ |
|  | val saijofarr=fsaijo.split(",") |
|  | val saijodomain=saijodf.getOWLClass(saijofarr(1),saijopm) |
|  | val saijofullName = saijodf.getOWLDataProperty(saijofarr(0), saijopm) |
|  | val saijodomainAxiomfullName = saijodf.getOWLDataPropertyDomainAxiom(saijofullName, saijodomain) |
|  | saijomanager.addAxiom(saijoontology, saijodomainAxiomfullName) |
|  | if(saijofarr(2)=="Saijostring") { |
|  | val saijostringDatatype = saijodf.getStringOWLDatatype () |
|  | val saijorangeAxiomfullName = saijodf.getOWLDataPropertyRangeAxiom(saijofullName, saijostringDatatype) |
|  | saijomanager.addAxiom(saijoontology, saijorangeAxiomfullName) |
|  | } |
|  | else if(saijofarr(2)=="Saijoint") |
|  | { |
|  | val saijoDatatype = saijodf.getIntegerOWLDatatype () |
|  | val saijorangeAxiomfullName = saijodf.getOWLDataPropertyRangeAxiom(saijofullName, saijoDatatype) |
|  | saijomanager.addAxiom(saijoontology, saijorangeAxiomfullName) |
|  | } |
|  | }) |
|  |  |
|  | val saijoindividuals=Source.fromFile("Saijodata/SaijoIndividuals").getLines() |
|  |  |
|  | saijoindividuals.foreach(fsaijo=>{ |
|  | val saijofarr=fsaijo.split(",") |
|  | val saijocls=saijodf.getOWLClass(saijofarr(0), saijopm) |
|  | val saijoind = saijodf.getOWLNamedIndividual(saijofarr(1), saijopm) |
|  | val saijoclassAssertion = saijodf.getOWLClassAssertionAxiom(saijocls, saijoind) |
|  | saijomanager.addAxiom(saijoontology, saijoclassAssertion) |
|  | }) |
|  |  |
|  | val saijotriplets=Source.fromFile("Saijodata/SaijoTriplets").getLines() |
|  | saijotriplets.foreach(fsaijo=>{ |
|  | val saijofarr=fsaijo.split(",") |
|  | val saijosub = saijodf.getOWLNamedIndividual(saijofarr(0), saijopm) |
|  |  |
|  | if(saijofarr(3)=="SaijoObject") |
|  | { |
|  | val saijopred=saijodf.getOWLObjectProperty(saijofarr(1),saijopm) |
|  | val saijoobj=saijodf.getOWLNamedIndividual(saijofarr(2), saijopm) |
|  | val saijoobjAsser = saijodf.getOWLObjectPropertyAssertionAxiom(saijopred,saijosub, saijoobj) |
|  | saijomanager.addAxiom(saijoontology, saijoobjAsser) |
|  | } |
|  | else if(saijofarr(3)=="SaijoData") |
|  | { |
|  | val saijopred=saijodf.getOWLDataProperty(saijofarr(1),saijopm) |
|  | val saijodat=saijodf.getOWLLiteral(saijofarr(2)) |
|  | val saijodatAsser = saijodf.getOWLDataPropertyAssertionAxiom(saijopred,saijosub, saijodat) |
|  | saijomanager.addAxiom(saijoontology, saijodatAsser) |
|  | } |
|  | }) |
|  |  |
|  | val saijoos = new FileOutputStream("Saijodata/saijodata.owl") |
|  | val saijoowlxmlFormat = new OWLXMLDocumentFormat |
|  | saijomanager.saveOntology(saijoontology, saijoowlxmlFormat, saijoos) |
|  | System.out.println("Saijo data Ontology done") |
|  |  |
|  |  |
|  |  |
|  | } |
|  |  |
|  | } |