//Controllor是啥样

//View怎么写

//缺少Background和InputFile、OutputFile、Flow类型

//DataDispatch是什么类？

//GWTdao是什么类型，是干啥的？不懂

//部分类操作没有写清楚

//数据调度器是什么类，在类图中如何体现

//类中Attribute的具体类型是啥Integer?String?自定义？

//class BackgroundImporter

对应至少一个领域背景文件

context BackgroundImporter //Background

inv:self.InputFile ->size()>=1

操作的功能

context BackgroundImporter::importBackground()

pre DataDispatcher -> size() = 1

post DataDispatcher.Background -> size() >= (DataDispatcher.Background -> size())@pre

//class GWTImporter

对应至少一个GWT输入文件

Context:GWTImporter

inv:self.inputFile->size()>=1

Context:GWTImporter::importGWT() //未给出

post:

//class RUCMGenerator

//gwtIdList是啥

对应一个NLPExecutor、dataDispatcher、LabelGenerator、OutputFile，多个RUCM

Context RUCMGenerator

inv:

self.NLPExecutor->size()=1

self.LabelGenerator->size()=1

self.dataDispatcher->size()=1

self.OutputFile->size()=1

self.RUCM->size()>=1

//生成一组RUCM

Context RUCMGenerator::generateRUCMs(gwtIdList:未知类型)TaggedGWT

post:return self.RUCMs

//生成一个RUCM

Context RUCMGenerator::generateRUCM(taggedList:未知类型)

post:return self.rucm

//生成briefDescription

Context RUCMGenerator::briefDescription(taggedList:未知类型，rucm:rucmType)

post:return self.rucm.briefDescription=self.rucm.briefDescription@pre+sentence

//未给出

Context RUCMGenerator::basicflow(taggedList:未知类型，rucm:rucmType)

post:

//未给出

Context RUCMGenerator::alternativeFlow(taggedList：未知类型，rucm:rucmType)

post:

//未给出

Context RUCMGenerator::generateOutput()

post:

//Class GWTdao是什么类，干啥的？

//初始化\_connect=connect\_db

Context GWTdao::\_connection()

init:connect\_db()

//各函数的作用：

context GWTDao::getGWTListById(id int):GWT

post: return=gwtList

context GWTDao::getid\_scenario\_list():(GWT.id:int,GWT.Scenario:String)List

post: return self.\_\_connection.execute('select id,scenario from GWT\_tb').fetchall()

context GWTDao::insert\_gwt(data:GWT)

post:self.\_\_connection.commit()

context GWTDao::insert\_Tagged\_gwt(data:GWT):Boolean

post: self.\_\_connection.execute()

//class LableGenerator()

//一个LableGenerator处理多个GWT

Context LableGenerator

inv:self.GWT->size()>=1

/./初始化，调用NLPExecutor

Context LableGenerator

init self.nlp=NLPExexutor

//各函数的作用：

Context generateLable(gwtlist:未知类型)

pre:

post:

Context \_simpleLable()

pre:

post:

Context \_normalize()

pre:

post:

Context \_addGWTLable()

pre:

post:

//class NLPExecutor()

//一次处理对应一个领域背景

Context NLPExecutor

inv:self.Background->size()=1

Context NLPExecutor::nlp

init: self.nlp=StanfordCoreNLP(path,lang='zh')

Context NLPExecutor::wordTokenize(sentence:String)

pre:

post:

Context NLPExecutor::posTag(sentence:String)

pre:

post:

Context NLPExecutor::dictUpdate(wordDict:未知类型)

pre:

post:

Context NLPExecutor::anaphoraResolution(text:未知类型)

pre:

post:

Context NLPExecutor::sentComposite(text:未知类型)

pre:

post:

Context NLPExecutor::feathreExtract(text:未知类型)

pre：

post：

//class GWT

//字段不能为空

Context GWT

inv:

self.gwt\_id<>NULL

self.Scenario<>NULL

self.Features<>NULL

self.Given<>NULL

self.Whens<>NULL

self.Thens<>NULL

对应一个InputFile和一个GWTImporter、多个sentence

Context GWT

inv:self.GWTImporter->size()=1

self.InputFile->size()=1

self.sentence->size()>=1

self.LabelGenerator->size()=1

//class TagedGWT

//字段不能为空

Context TagedGWT

inv:

self.useCaseName<>NULL

self.BranchScenarios<>NULL

self.flowType<>NULL

//对应一个GWT、RUCMGenerator,多个Tagedsentence、Scenario、LabelGenerator

Context TagedGWT

inv:self.GWT->size()=1

self.Tagedsentence->size()>=1

self.Scenario->size()>=1

self.LabelGenerator->size()=1

self.RUCMGenerator->size()=1

//class RUCM

Context RUCM

inv:

self.briefDescription<>NULL

self.precondition<>NULL

self.primaryActor<>NULL

self.basic<>NULL

//初始化usecasename

Context RUCM

init:

self.useCaseName=name

Context RUCM

inv:self.OutputFile->size()=1

self.RUCMGenerator->size()=1

self.flow->size()>=1 //缺少flow类

//class Sentence

Context Sentence

inv:

self.sentence\_id<>NULL

self.type<>NULL

self.content<>NULL

self.sequence<>NULL //sequence与id的作用分别是啥

Context Sentence

inv:self.GWT->size()=1

//class Tagedsentence

Context Tagedsentence

inv:

self.secondType<>NULL

Context Tagedsentence

inv:

self.TagedGWT->size()=1

self.Association->size()>=0

//class Association

Context Association

inv:

self.gwtId<>NULL

self.sentenceid<>NULL

self.conntect\_type<>NULL

Context Association

inv:self.Tagedsentence->size()=1

//class Scenario

Context Scenario

inv:

self.gwtId

self.conditionIds

Context Scenario

inv:

self.Tagedsentence->size()=1

//class BasicFlow

//初始化

Context BasicFlow::action

init:[]

Context BasicFlow::postCondition

init:''

//加入action

Context BasicFlow::addAction(action:String)

post:self.actions=self.actions@pre+action

//class SpecificFlow

Context SpecificFlow::rfs

init:0

Context SpecificFlow::actions

init:[]

Context SpecificFlow::postCondition

init:''

Context SpecificFlow::addRFS(num:?) //num类型

post:self.rfs=self.rfs@pre+num

//class BoundedFlow

Context BoundedFlow::rfs

init:0

Context BoundedFlow::actions

init:[]

Context BoundedFlow::postCondition

init:''

Context BoundedFlow::addRFS(num:?) //num类型

post:self.rfs=self.rfs@pre+num

Context BoundedFlow::addAction(action:string)

post:self.actions=self.actions@pre+action

//class GolbalFlow

Context GolbalFlow::condition

init:[]

Context GolbalFlow::actions

init:[]

Context GolbalFlow::postCondition

init:[]

Context GolbalFlow::addAction(action:String)

post:self.actions=self.actions@pre+action