

UML第二次作业接口使用说明

本次我们继续像是之前一样，提供封装好的jar包给大家。

这次的话，我们已经将全部的主干业务逻辑进行了封装，只需要同学们实现一个核心交互类即可。

除此之外，本次的官方包还可以作为命令行工具使用，以便快速从 `mdj` 文件中导出并生成输入数据。

功能实现

UmlGeneralInteraction接口

学生需要实现一个自己的 `UmlGeneralInteraction` 类，这个类必须继承接口

```
com.oocourse.uml2.interact.format.UmlGeneralInteraction。
```

```
import com.oocourse.uml2.interact.format.UmlGeneralInteraction;

public class MyUmlGeneralInteraction implements UmlGeneralInteraction {
    // TODO : IMPLEMENT
}
```

接口源码设定（`UmlGeneralInteraction`）：

```
package com.oocourse.uml2.interact.format;

/**
 * UML总交互接口
 */
public interface UmlGeneralInteraction
    extends UmlClassModelInteraction, UmlCollaborationInteraction,
    UmlStateChartInteraction {
}
```

接口源码设定（`UmlStateChartInteraction`）：

```
package com.oocourse.uml2.interact.format;

import com.oocourse.uml2.interact.exceptions.user.StateDuplicatedException;
import com.oocourse.uml2.interact.exceptions.user.StateMachineDuplicatedException;
import com.oocourse.uml2.interact.exceptions.user.StateMachineNotFoundException;
import com.oocourse.uml2.interact.exceptions.user.StateNotFoundException;
import com.oocourse.uml2.interact.exceptions.user.TransitionNotFoundException;

import java.util.List;

/**
 * UML状态图交互
 */
public interface UmlStateChartInteraction {
    /**
```

```

    * 获取状态机的状态数
    * 指令: STATE_COUNT
    *
    * @param stateMachineName 状态机名称
    * @return 状态数
    * @throws StateMachineNotFoundException 状态机未找到
    * @throws StateMachineDuplicatedException 状态机存在重名
    */
    int getStateCount(String stateMachineName)
        throws StateMachineNotFoundException,
        StateMachineDuplicatedException;

    /**
    * 获取后继状态数
    * 指令: SUBSEQUENT_STATE_COUNT
    *
    * @param stateMachineName 状态机名称
    * @param stateName 状态名称
    * @return 后继状态数
    * @throws StateMachineNotFoundException 状态机未找到
    * @throws StateMachineDuplicatedException 状态机存在重名
    * @throws StateNotFoundException 状态未找到
    * @throws StateDuplicatedException 状态存在重名
    */
    int getSubsequentStateCount(String stateMachineName, String stateName)
        throws StateMachineNotFoundException,
        StateMachineDuplicatedException,
        StateNotFoundException, StateDuplicatedException;

    /**
    * 获取引起状态迁移的触发事件
    * 指令: TRANSITION_TRIGGER
    *
    * @param stateMachineName 状态机名称
    * @param sourceStateName 状态迁移源状态名称
    * @param targetStateName 状态迁移目标状态名称
    * @return 引起状态迁移的触发事件列表
    * @throws StateMachineNotFoundException 状态机未找到
    * @throws StateMachineDuplicatedException 状态机存在重名
    * @throws StateNotFoundException 状态未找到
    * @throws StateDuplicatedException 状态存在重名
    * @throws TransitionNotFoundException 不存在从源状态到目标状态的状态迁移
    */
    List<String> getTransitionTrigger(
        String stateMachineName, String sourceStateName, String targetStateName
    )
        throws StateMachineNotFoundException,
        StateMachineDuplicatedException,
        StateNotFoundException, StateDuplicatedException,
        TransitionNotFoundException;
}

```

接口源码设定 (UmlCollaborationInteraction) :

```
package com.oocourse.uml2.interact.format;
```

```

import
com.oocourse.uml2.interact.exceptions.user.InteractionDuplicatedException;
import com.oocourse.uml2.interact.exceptions.user.InteractionNotFoundException;
import com.oocourse.uml2.interact.exceptions.user.LifelineDuplicatedException;
import com.oocourse.uml2.interact.exceptions.user.LifelineNotFoundException;
import com.oocourse.uml2.models.common.MessageSort;

/**
 * UML顺序图交互
 */
public interface UmlCollaborationInteraction {
    /**
     * 获取参与对象数
     * 指令: PTCP_OBJ_COUNT
     *
     * @param interactionName 交互名称
     * @return 参与对象数
     * @throws InteractionNotFoundException 交互未找到
     * @throws InteractionDuplicatedException 交互重名
     */
    int getParticipantCount(String interactionName)
        throws InteractionNotFoundException,
        InteractionDuplicatedException;

    /**
     * 获取对象的进入消息数
     * 指令: INCOMING_MSG_COUNT
     *
     * @param interactionName 交互名称
     * @param lifelineName 消息名称
     * @return 进入消息数
     * @throws InteractionNotFoundException 交互未找到
     * @throws InteractionDuplicatedException 交互重名
     * @throws LifelineNotFoundException 生命线未找到
     * @throws LifelineDuplicatedException 生命线重名
     */
    int getIncomingMessageCount(String interactionName, String lifelineName)
        throws InteractionNotFoundException,
        InteractionDuplicatedException,
        LifelineNotFoundException, LifelineDuplicatedException;

    /**
     * 获取对象发出的某个类别的消息数
     * 指令: SENT_MESSAGE_COUNT
     *
     * @param interactionName 交互名称
     * @param lifelineName 消息名称
     * @param sort 消息类别
     * @return 发出的某个类别的消息数
     * @throws InteractionNotFoundException 交互未找到
     * @throws InteractionDuplicatedException 交互重名
     * @throws LifelineNotFoundException 生命线未找到
     * @throws LifelineDuplicatedException 生命线重名
     */
    int getSentMessageCount(String interactionName, String lifelineName,
        MessageSort sort)
        throws InteractionNotFoundException,
        InteractionDuplicatedException,

```

```
        LifelineNotFoundException, LifelineDuplicatedException;
    }
}
```

接口源码设定（UmlClassModelInteraction）：

```
package com.oocourse.uml2.interact.format;

import com.oocourse.uml2.interact.common.AttributeClassInformation;
import com.oocourse.uml2.interact.common.OperationParamInformation;
import com.oocourse.uml2.interact.exceptions.user.AttributeDuplicatedException;
import com.oocourse.uml2.interact.exceptions.user.AttributeNotFoundException;
import com.oocourse.uml2.interact.exceptions.user.AttributeWrongTypeException;
import com.oocourse.uml2.interact.exceptions.user.ClassDuplicatedException;
import com.oocourse.uml2.interact.exceptions.user.MethodDuplicatedException;
import com.oocourse.uml2.interact.exceptions.user.MethodWrongTypeException;
import com.oocourse.uml2.models.common.Visibility;

import java.util.List;
import java.util.Map;

/**
 * UML交互接口
 */
public interface UmlClassModelInteraction {
    /**
     * 获取类数量
     * 指令: CLASS_COUNT
     *
     * @return 类数量
     */
    int getClassCount();

    /**
     * 获取类操作数量
     * 指令: CLASS_OPERATION_COUNT
     *
     * @param className 类名
     * @return 类的操作数量
     * @throws ClassNotFoundException 类未找到异常
     * @throws ClassDuplicatedException 类重复异常
     */
    int getClassOperationCount(String className)
        throws ClassNotFoundException, ClassDuplicatedException;

    /**
     * 获取类属性数量
     * 指令: CLASS_ATTR_COUNT
     *
     * @param className 类名
     * @return 类属性操作数量
     * @throws ClassNotFoundException 类未找到异常
     * @throws ClassDuplicatedException 类重复异常
     */
    int getClassAttributeCount(String className)
        throws ClassNotFoundException, ClassDuplicatedException;
}
```

```

/**
 * 统计类操作可见性
 * 指令: CLASS_OPERATION_VISIBILITY
 *
 * @param className      类名
 * @param operationName 操作名
 * @return 类操作可见性统计结果
 * @throws ClassNotFoundException 类未找到异常
 * @throws ClassDuplicatedException 类重复异常
 */
Map<Visibility, Integer> getClassOperationVisibility(String className,
String operationName)
    throws ClassNotFoundException, ClassDuplicatedException;

/**
 * 获取类属性可见性
 * 指令: CLASS_ATTR_VISIBILITY
 *
 * @param className      类名
 * @param attributeName 属性名
 * @return 属性可见性
 * @throws ClassNotFoundException 类未找到异常
 * @throws ClassDuplicatedException 类重复异常
 * @throws AttributeNotFoundException 属性未找到异常
 * @throws AttributeDuplicatedException 属性重复异常
 */
Visibility getClassAttributeVisibility(String className, String
attributeName)
    throws ClassNotFoundException, ClassDuplicatedException,
AttributeNotFoundException, AttributeDuplicatedException;

/**
 * 获取类属性类型
 * 指令: CLASS_ATTR_TYPE
 *
 * @param className      类名
 * @param attributeName 属性名
 * @return 属性类型
 * @throws ClassNotFoundException 类未找到异常
 * @throws ClassDuplicatedException 类重复异常
 * @throws AttributeNotFoundException 属性未找到异常
 * @throws AttributeDuplicatedException 属性重复异常
 * @throws AttributeWrongTypeException 属性类型错误异常
 */
String getClassAttributeType(String className, String attributeName)
    throws ClassNotFoundException, ClassDuplicatedException,
AttributeNotFoundException, AttributeDuplicatedException,
AttributeWrongTypeException;

/**
 * 统计类操作参数类型
 * 指令: CLASS_OPERATION_PARAM_TYPE
 *
 * @param className      类名
 * @param operationName 操作名
 * @return 类操作参数类型列表
 * @throws ClassNotFoundException 类未找到异常
 * @throws ClassDuplicatedException 类重复异常

```

```

    * @throws MethodWrongTypeException 方法参数类型错误异常
    * @throws MethodDuplicatedException 方法重复异常
    */
    List<OperationParamInformation> getClassOperationParamType(
        String className, String operationName
    ) throws ClassNotFoundException, ClassDuplicatedException,
        MethodWrongTypeException, MethodDuplicatedException;

    /**
     * 获取与类相关联的类列表
     * 指令: CLASS ASSO CLASS LIST
     *
     * @param className 类名
     * @return 与类关联的类列表
     * @throws ClassNotFoundException 类未找到异常
     * @throws ClassDuplicatedException 类重复异常
     */
    List<String> getClassAssociatedClassList(String className)
        throws ClassNotFoundException, ClassDuplicatedException;

    /**
     * 获取顶级父类
     * 指令: CLASS TOP BASE
     *
     * @param className 类名
     * @return 顶级父类名
     * @throws ClassNotFoundException 类未找到异常
     * @throws ClassDuplicatedException 类重复异常
     */
    String getTopParentClass(String className)
        throws ClassNotFoundException, ClassDuplicatedException;

    /**
     * 获取实现的接口列表
     * 指令: CLASS IMPLEMENT INTERFACE LIST
     *
     * @param className 类名
     * @return 实现的接口列表
     * @throws ClassNotFoundException 类未找到异常
     * @throws ClassDuplicatedException 类重复异常
     */
    List<String> getImplementInterfaceList(String className)
        throws ClassNotFoundException, ClassDuplicatedException;

    /**
     * 获取类中未隐藏的属性
     * 即违背了面向对象设计中的隐藏信息原则的属性
     * 指令: CLASS INFO HIDDEN
     *
     * @param className 类名
     * @return 未隐藏的属性信息列表
     * @throws ClassNotFoundException 类未找到异常
     * @throws ClassDuplicatedException 类重复异常
     */
    List<AttributeClassInformation> getInformationNotHidden(String className)
        throws ClassNotFoundException, ClassDuplicatedException;
}

```

除此之外，`UmlGeneralInteraction` 类必须实现一个构造方法

```
public class MyUmlGeneralInteraction implements UmlUmlGeneralInteraction {
    public MyUmlClassModelInteraction(UmlElement[] elements);
}
```

或者

```
public class MyUmlGeneralInteraction implements UmlUmlGeneralInteraction {
    public MyUmlClassModelInteraction(UmlElement... elements);
}
```

构造函数的逻辑为将 `elements` 数组内的各个UML类图元素传入 `UmlGeneralInteraction` 类，以备后续解析。

交互模式

交互的模式为：

- 调用上述构造函数，生成一个实例，并将UML模型元素传入。
- 之后将调用此实例的各个接口方法，以实现基于之前传入的UML模型元素各类查询操作。
- 官方接口通过调用方法的返回值，自动生成对应的输出文本。

开始运行

运行的模式和之前基本类似：

```
import com.oocourse.uml2.interact.AppRunner;

public class Main {
    public static void main(String[] args) throws Exception {
        AppRunner appRunner =
        AppRunner.newInstance(MyUmlGeneralInteraction.class);
        appRunner.run(args);
    }
}
```

将自己实现的类进行载入，即可运行。

数据生成

命令行工具

和之前不同的是，此次的官方jar包还可以作为命令行工具使用，简单的几个用法如下。

参考样例文件：[传送门 \(open-close.mdj\)](#)。

查看可导出的数据模型列表

用户可以通过这一功能查看支持导出的数据模型列表。

```
java -jar uml-homework.jar list -s "open-close.mdj"
```

输出结果

+-----+-----+		
	Type	Name
+-----+-----+		
	UMLModel	Model
	UMLModel	Model1
+-----+-----+		

在本次，此命令支持类别筛选，可以通过 `-t` 指令进行类别指定（不指定表示全部显示）。

目前支持：

- `UMLModel`
- `UMLStateMachine`
- `UMLCollaboration`

查看数据模型

用户可以通过这一功能查看数据模型内含的全部可识别元素。

例如，下述指令查看类型为 `UMLModel` 名称为 `Model` 的元素表：

```
java -jar uml-homework.jar list -s "open-close.mdj" -t UMLModel -n Model
```

输出结果（受限于页面宽度，部分地方可能存在换行，建议在命令行中使用以获得更佳体验）

+-----+-----+-----+-----+			
-----+			
	Type		Name
			ID
Parent Id			
+-----+-----+-----+-----+			
-----+			
	UMLClass		Door
AAAAAAFF+qBWK6M3Z8Y=		AAAAAAFqpiMge7NXBnk=	
	UMLOperation		Door
AAAAAAFFqpiMge7NXBnk=		AAAAAAFqpiQWH7O0bzI=	
	UMLOperation		Open
AAAAAAFFqpiMge7NXBnk=		AAAAAAFqpiRcY7O7pzM=	
	UMLParameter		
AAAAAAFFqpiRcY7O7pzM=		AAAAAAFqpi3MbPYrBA=	
	UMLParameter		k
AAAAAAFFqpiRcY7O7pzM=		AAAAAAFqpz3cylqvuQ=	
	UMLOperation		Close
AAAAAAFFqpiMge7NXBnk=		AAAAAAFqpyDeZlAA9wA=	
	UMLParameter		
AAAAAAFFqpyDeZlAA9wA=		AAAAAAFqpyECbVAHLpo=	
	UMLOperation		Register
AAAAAAFFqpiMge7NXBnk=		AAAAAAFqpz7UOVfbTr8=	
	UMLParameter		e
AAAAAAFFqpz7UOVfbTr8=		AAAAAAFqpz83wlgSehs=	
	UMLParameter		
AAAAAAFFqpz7UOVfbTr8=		AAAAAAFqpz83wlgTXoQ=	
	UMLOperation		UnRegister
AAAAAAFFqpiMge7NXBnk=		AAAAAAFqpz98blheYb8=	
	UMLParameter		e
AAAAAAFFqpz98blheYb8=		AAAAAAFqpz/Q6linSCc=	

UMLParameter	k	AAAAAAFqpz/Q61iokxk=
AAAAAFqpz98b1heYb8=		
UMLParameter		AAAAAAFqpz/Q61ipA8c=
AAAAAFqpz98b1heYb8=		
UMLOperation	isOpen	AAAAAAFqwQTh/MG8LKk=
AAAAAFqpiMge7NXBnk=		
UMLParameter		AAAAAAFqwRJT8PKJ0k=
AAAAAFqwQTh/MG8LKk=		
UMLOperation	getRoomNo	AAAAAAFqwRE8ucKwxBA=
AAAAAFqpiMge7NXBnk=		
UMLParameter		AAAAAAFqwRHwR8NkxtQ=
AAAAAFqwRE8ucKwxBA=		
UMLAssociation		AAAAAAFqpyLHQ1A/uHQ=
AAAAAFqpiMge7NXBnk=		
UMLAssociationEnd	locker	AAAAAAFqpyLHQ1BBCwQ=
AAAAAFqpyLHQ1A/uHQ=		
UMLAssociationEnd	lockedDoor	AAAAAAFqpyLHQ1BA8jU=
AAAAAFqpyLHQ1A/uHQ=		
UMLAssociation		AAAAAAFqwUWWHPTahS8=
AAAAAFqpiMge7NXBnk=		
UMLAssociationEnd	client	AAAAAAFqwUWWHPTc/rg=
AAAAAFqwUWWHPTahS8=		
UMLAssociationEnd	rooms	AAAAAAFqwUWWHPTbrlg=
AAAAAFqwUWWHPTahS8=		
UMLStateMachine	simpe_sm	AAAAAAFqyONLFLlV140=
AAAAAFqpiMge7NXBnk=		
UMLRegion		AAAAAAFqyONLFLlWdXI=
AAAAAFqyONLFLlV140=		
UMLPseudostate		AAAAAAFqyOVx3rmCP2Y=
AAAAAFqyONLFLlWdXI=		
UMLState	opened	AAAAAAFqyOW7gLmTuE4=
AAAAAFqyONLFLlWdXI=		
UMLState	closed	AAAAAAFqyOXm0Lm5/v8=
AAAAAFqyONLFLlWdXI=		
UMLTransition	open	AAAAAAFqyOY/GLngY5I=
AAAAAFqyONLFLlWdXI=		
UMLOpaqueBehavior	locker.unlock(key)	AAAAAAFqyPbIMrvFRtg=
AAAAAFqyOY/GLngY5I=		
UMLEvent	Open(key)	AAAAAAFqyO3ytLoyjlA=
AAAAAFqyOY/GLngY5I=		
UMLTransition	close	AAAAAAFqyObAnrny29A=
AAAAAFqyONLFLlWdXI=		
UMLOpaqueBehavior	locker.lock()	AAAAAAFqyR4HIb4itVs=
AAAAAFqyObAnrny29A=		
UMLEvent	Close()	AAAAAAFqyP2QWL3jOls=
AAAAAFqyObAnrny29A=		
UMLTransition	open	AAAAAAFqyOksebodfMo=
AAAAAFqyONLFLlWdXI=		
UMLOpaqueBehavior	locker.unlock(key)	AAAAAAFqyQALdb33GjU=
AAAAAFqyOksebodfMo=		
UMLEvent	Open(key)	AAAAAAFqyP/c/b3zRtQ=
AAAAAFqyOksebodfMo=		
UMLStateMachine	complex_sm	AAAAAAFqyQWs9L3/cek=
AAAAAFqpiMge7NXBnk=		
UMLRegion		AAAAAAFqyQWs9b4A8Bk=
AAAAAFqyQWs9L3/cek=		
UMLPseudostate		AAAAAAFqyeEMPTDVjII=
AAAAAFqyQWs9b4A8Bk=		

UMLState	trying2Open	AAAAAAFqyeFWgDDmGrM=
AAAAAAFqyQWs9b4A8Bk=		
UMLOpaqueBehavior	locker.unlock(key)	AAAAAAFqyexozJYj3E=
AAAAAAFqyeFWgDDmGrM=		
UMLOpaqueBehavior	trying = trying + 1	AAAAAAFqyezvVTJi0oM=
AAAAAAFqyeFWgDDmGrM=		
UMLState	opened	AAAAAAFqyeGaeDEN0do=
AAAAAAFqyQWs9b4A8Bk=		
UMLOpaqueBehavior	bOpen = true	AAAAAAFqygqiCTLMoxQ=
AAAAAAFqyeGaeDEN0do=		
UMLState	blocked	AAAAAAFqyeHHXDE0fXE=
AAAAAAFqyQWs9b4A8Bk=		
UMLRegion	Region1	AAAAAAFq3lVFLbl/ABk=
AAAAAAFqyeHHXDE0fXE=		
UMLState	closed	AAAAAAFqyeH7hjFbnBs=
AAAAAAFqyQWs9b4A8Bk=		
UMLOpaqueBehavior	bOpen=false	AAAAAAFqygr7KDLTsxk=
AAAAAAFqyeH7hjFbnBs=		
UMLFinalState		AAAAAAFqyeKjvDGGayc=
AAAAAAFqyQWs9b4A8Bk=		
UMLTransition	open	AAAAAAFqyeLuBjGMJ9M=
AAAAAAFqyQWs9b4A8Bk=		
UMLOpaqueBehavior	trying = 0	AAAAAAFqyetqrDJRthg=
AAAAAAFqyeLuBjGMJ9M=		
UMLEvent	Open(key)	AAAAAAFqyealLTirDKQ=
AAAAAAFqyeLuBjGMJ9M=		
UMLTransition	open	AAAAAAFqyeMDizGdIG4=
AAAAAAFqyQWs9b4A8Bk=		
UMLEvent	Open(key)	AAAAAAFqye1oTDJqUtU=
AAAAAAFqyeMDizGdIG4=		
UMLTransition		AAAAAAFqyeMf8zGuRsE=
AAAAAAFqyQWs9b4A8Bk=		
UMLTransition	failed	AAAAAAFqyeNVjDHD+zw=
AAAAAAFqyQWs9b4A8Bk=		
UMLTransition		AAAAAAFqyeN2RjHUzPY=
AAAAAAFqyQWs9b4A8Bk=		
UMLTransition	close	AAAAAAFqyeOXijHmln4=
AAAAAAFqyQWs9b4A8Bk=		
UMLOpaqueBehavior	locker.lock()	AAAAAAFqygIsMDKDqXI=
AAAAAAFqyeOXijHmln4=		
UMLEvent	Close()	AAAAAAFqygJgLzKHY4=
AAAAAAFqyeOXijHmln4=		
UMLTransition	open	AAAAAAFqyeOtiDH3utM=
AAAAAAFqyQWs9b4A8Bk=		
UMLOpaqueBehavior	trying = 0	AAAAAAFqygVxwzKyAK8=
AAAAAAFqyeOtiDH3utM=		
UMLEvent	Open(key)	AAAAAAFqygU6wDKunp0=
AAAAAAFqyeOtiDH3utM=		
UMLTransition	reset	AAAAAAFqyeQalzIMhrI=
AAAAAAFqyQWs9b4A8Bk=		
UMLOpaqueBehavior	trying = 0	AAAAAAFqygEf8DJ4k9k=
AAAAAAFqyeQalzIMhrI=		
UMLEvent	Reset()	AAAAAAFqygGAUDJ8opk=
AAAAAAFqyeQalzIMhrI=		
UMLStateMachine	StateMachine1	AAAAAAFq3t1EnL6iQpQ=
AAAAAAFqpiMge7NXBnk=		
UMLRegion		AAAAAAFq3t1EnL6jADA=
AAAAAAFq3t1EnL6iQpQ=		

UMLAssociation		AAAAAAFq4pz3MMFoTW8=	
AAAAAFqpiMge7NXBnk=			
UMLAssociationEnd		AAAAAAFq4pz3McFqCSQ=	
AAAAAFq4pz3MMFoTW8=			
UMLAssociationEnd	sdfdsfgsfdg	AAAAAAFq4pz3MMFpo88=	
AAAAAFq4pz3MMFoTW8=			
UMLClass	Class1	AAAAAAFq6iC1sOB0huU=	
AAAAAFqpiMge7NXBnk=			
UMLAttribute	bOpen	AAAAAAFqpiN8GLOssfo=	
AAAAAFqpiMge7NXBnk=			
UMLAttribute	roomNO	AAAAAAFqpyGbn1AMoqE=	
AAAAAFqpiMge7NXBnk=			
UMLAttribute	guests	AAAAAAFqp0ZAqWCp/yc=	
AAAAAFqpiMge7NXBnk=			
UMLAttribute	assignedKeys	AAAAAAFqp0bpg2FufMY=	
AAAAAFqpiMge7NXBnk=			
UMLAttribute	availableKeys	AAAAAAFqp0frlGIqTHo=	
AAAAAFqpiMge7NXBnk=			
UMLClass	Lock	AAAAAAFqpyKBqVAUSAo=	
AAAAAFF+qBWK6M3Z8Y=			
UMLOperation	lock	AAAAAAFqpyVxfVFaqSg=	
AAAAAFqpyKBqVAUSAo=			
UMLParameter	k	AAAAAAFqpyW721F53Fg=	
AAAAAFqpyVxfVFaqSg=			
UMLParameter		AAAAAAFqpyW721F6New=	
AAAAAFqpyVxfVFaqSg=			
UMLOperation	unlock	AAAAAAFqpyXW4FGSWdU=	
AAAAAFqpyKBqVAUSAo=			
UMLParameter		AAAAAAFqpyYDplGyRh8=	
AAAAAFqpyXW4FGSWdU=			
UMLOperation	match	AAAAAAFqp3wEn26eYK0=	
AAAAAFqpyKBqVAUSAo=			
UMLParameter	k	AAAAAAFqp3xbj27tCmE=	
AAAAAFqp3wEn26eYK0=			
UMLParameter		AAAAAAFqp3xbj27uWUQ=	
AAAAAFqp3wEn26eYK0=			
UMLOperation	getLockId	AAAAAAFqyPHMP7qoa18=	
AAAAAFqpyKBqVAUSAo=			
UMLParameter		AAAAAAFqyPJLDbSckeg=	
AAAAAFqyPHMP7qoa18=			
UMLInterfaceRealization		AAAAAAFqyz3DUrUBj9E=	
AAAAAFqpyKBqVAUSAo=			
UMLAttribute	totalKeys	AAAAAAFqpyQOxlEmyts=	
AAAAAFqpyKBqVAUSAo=			
UMLAttribute	keys	AAAAAAFqpyoRiFMTmMs=	
AAAAAFqpyKBqVAUSAo=			
UMLAttribute	lockID	AAAAAAFqpywyFpNwW8=	
AAAAAFqpyKBqVAUSAo=			
UMLClass	Key	AAAAAAFqpyZaw1HqYaU=	
AAAAAFF+qBWK6M3Z8Y=			
UMLOperation	equals	AAAAAAFqp0vL7mYHuPo=	
AAAAAFqpyZaw1HqYaU=			
UMLParameter	o	AAAAAAFqp0xjgmZWAXk=	
AAAAAFqp0vL7mYHuPo=			
UMLParameter		AAAAAAFqp0xjgmZXPzs=	
AAAAAFqp0vL7mYHuPo=			
UMLOperation	getMatchedLockId	AAAAAAFqp37jkXF7CJ4=	
AAAAAFqpyZaw1HqYaU=			

UMLParameter		AAAAAAFqp38tFHHKHMI=
AAAAAFqp37jkXF7CJ4=		
UMLAttribute	keyID	AAAAAFqpyZ7clI8H7g=
AAAAAFqpyZaw1HqYaU=		
UMLAttribute	matchedLockID	AAAAAFqpy7tKFUvHfM=
AAAAAFqpyZaw1HqYaU=		
UMLClass	NoMoreKeyException	AAAAAFqp0EJi1lLqGo=
AAAAAFF+qBWK6M3Z8Y=		
UMLGeneralization		AAAAAFqp1LTBmtxfv4=
AAAAAFqp0EJi1lLqGo=		
UMLAssociation		AAAAAFq5htejtC5T6Q=
AAAAAFqp0EJi1lLqGo=		
UMLAssociationEnd		AAAAAFq5htejtC7/sM=
AAAAAFq5htejtC5T6Q=		
UMLAssociationEnd		AAAAAFq5htejtC6gxI=
AAAAAFq5htejtC5T6Q=		
UMLAssociation		AAAAAFq5htsJ9FBdyU=
AAAAAFqp0EJi1lLqGo=		
UMLAssociationEnd		AAAAAFq5htsJ9FDz58=
AAAAAFq5htsJ9FBdyU=		
UMLAssociationEnd		AAAAAFq5htsJ9FCuk4=
AAAAAFq5htsJ9FBdyU=		
UMLClass	Exception	AAAAAFqp1KmH2r29Ds=
AAAAAFF+qBWK6M3Z8Y=		
UMLClass	Client	AAAAAFqwTWWKvND/ug=
AAAAAFF+qBWK6M3Z8Y=		
UMLOperation	enterRoom	AAAAAFqwTZbePPJQUA=
AAAAAFqwTWWKvND/ug=		
UMLParameter	rn	AAAAAFqwTaykvPsLIM=
AAAAAFqwTZbePPJQUA=		
UMLParameter		AAAAAFqwTbHdfP1AjM=
AAAAAFqwTZbePPJQUA=		
UMLOperation	leaveRoom	AAAAAFqwUSAY/Q9Sfs=
AAAAAFqwTWWKvND/ug=		
UMLParameter	d	AAAAAFqwUS7n/RcqDM=
AAAAAFqwUSAY/Q9Sfs=		
UMLParameter		AAAAAFqwUS7oPRdXXs=
AAAAAFqwUSAY/Q9Sfs=		
UMLOperation	locateRoom	AAAAAFqwUTaWPR1AfU=
AAAAAFqwTWWKvND/ug=		
UMLParameter	rn	AAAAAFqwUUKLfSRkmw=
AAAAAFqwUTaWPR1AfU=		
UMLParameter		AAAAAFqwUUfk/SalJI=
AAAAAFqwUTaWPR1AfU=		
UMLAssociation		AAAAAFqwUbWV/aG5TQ=
AAAAAFqwTWWKvND/ug=		
UMLAssociationEnd	keys	AAAAAFqwUbWV/aI8Po=
AAAAAFqwUbWV/aG5TQ=		
UMLAssociationEnd		AAAAAFqwUbWV/aHDjw=
AAAAAFqwUbWV/aG5TQ=		
UMLAttribute	clientID	AAAAAFqwTXvtfOC318=
AAAAAFqwTWWKvND/ug=		
UMLClass	ElcKey	AAAAAFqyyULIat6fvE=
AAAAAFF+qBWK6M3Z8Y=		
UMLOperation	equals	AAAAAFqy0Q7JMCG23I=
AAAAAFqyyULIat6fvE=		
UMLParameter	o	AAAAAFqy0SKNsDVObs=
AAAAAFqy0Q7JMCG23I=		

UMLParameter		AAAAAAFqy0SKN8DW850=	
AAAAAFqy0Q7JMCg23I=			
UMLAttribute	sigCode	AAAAAAFqy0FPcb5DotA=	
AAAAAFqyyULIat6fvE=			
UMLInterface	Locker	AAAAAAFqyyuTsa1CnU8=	
AAAAAFAFF+qBWK6M3Z8Y=			
UMLOperation	lock	AAAAAAFqyz66dreg3Oc=	
AAAAAFqyyuTsa1CnU8=			
UMLParameter	k	AAAAAAFqyz9BVbhUGOc=	
AAAAAFqyz66dreg3Oc=			
UMLParameter		AAAAAAFqyz9BVrhV7D8=	
AAAAAFqyz66dreg3Oc=			
UMLOperation	unlock	AAAAAAFqyz9aIbipNj8=	
AAAAAFqyyuTsa1CnU8=			
UMLParameter		AAAAAAFqyz+Ga7j4px8=	
AAAAAFqyz9aIbipNj8=			
UMLInterface	Interface1	AAAAAAFq5hWfnsrejMQ=	
AAAAAFAFF+qBWK6M3Z8Y=			
UMLClass	Door	AAAAAAFq6i/M3ODSOBc=	
AAAAAFAFF+qBWK6M3Z8Y=			
+-----+-----+-----+-----+			
-----+			

导出指定的数据模型

用户可以通过这一功能对数据模型进行导出。

与上次不同的是，本次由于存在多种模型，所以必须通过 `-t` 指定模型类型（目前支持的三种类型同上）。

导出的数据格式可以直接作为数据模型的输入内容，在其后接上 `END_OF_MODEL` 和各类指令，即可构建为一个输入数据。

```
java -jar uml-homework.jar dump -s "open-close.mdj" -n Model1 -t UMLModel
```

输出结果

```
{ "_parent": "AAAAAFq3tvYM76UevI=", "visibility": "public", "name": "Key", "_type": "UMLClass", "_id": "AAAAAFq7weIMsb5xqQ=" }
{ "_parent": "AAAAAFq7weIMsb5xqQ=", "visibility": "public", "name": "equals", "_type": "UMLOperation", "_id": "AAAAAFq7weIMsb8qxc=" }
{ "_parent": "AAAAAFq7weIMsb8qxc=", "name": "o", "_type": "UMLParameter", "_id": "AAAAAFq7weIMsb9G0k=", "type": "Object", "direction": "in" }
{ "_parent": "AAAAAFq7weIMsb8qxc=", "name": null, "_type": "UMLParameter", "_id": "AAAAAFq7weIMsb+Au4=", "type": "boolean", "direction": "return" }
{ "_parent": "AAAAAFq7weIMsb5xqQ=", "visibility": "public", "name": "getMatchedLockId", "_type": "UMLOperation", "_id": "AAAAAFq7weIMsb\6gM=" }
{ "_parent": "AAAAAFq7weIMsb\6gM=", "name": null, "_type": "UMLParameter", "_id": "AAAAAFq7weIMScAoOk=", "type": "int", "direction": "return" }
{ "_parent": "AAAAAFq7weIMsb5xqQ=", "visibility": "public", "name": "Operation1", "_type": "UMLOperation", "_id": "AAAAAFq7w1zLCePJrI=" }
{ "_parent": "AAAAAFq7w1zLCePJrI=", "name": "Parameter1", "_type": "UMLParameter", "_id": "AAAAAFq7w2dZCeV4K8=", "type": "int" }
{ "$ref": "AAAAAFq7weIMsb5xqQ=", "direction": "return" }
{ "_parent": "AAAAAFq7weIMsb5xqQ=", "name": null, "_type": "UMLGeneralization", "_id": "AAAAAFq7wfNvyd+GgY=", "source": "AAAAAFq7weIMsb5xqQ=", "target": "AAAAAFq7weIMsb5xqQ=" }
```

```
{ "_parent": "AAAAAAAFq7weIMsb5xqQ=", "visibility": "private", "name": "keyID", "_type": "UMLAttribute", "_id": "AAAAAAAFq7weIMsb6+v8=", "type": "int" }
{ "_parent": "AAAAAAAFq7weIMsb5xqQ=", "visibility": "private", "name": "matchedLockID", "_type": "UMLAttribute", "_id": "AAAAAAAFq7weIMsb7oPM=", "type": "int" }
{ "_parent": "AAAAAAAFq3tvYM76UevI=", "visibility": "public", "name": "ElcKey", "_type": "UMLClass", "_id": "AAAAAAAFq7weqoCcQE7I=" }
{ "_parent": "AAAAAAAFq7weqoCcQE7I=", "visibility": "public", "name": "equals", "_type": "UMLOperation", "_id": "AAAAAAAFq7weqoCcTngY=" }
{ "_parent": "AAAAAAAFq7weqoCcTngY=", "name": "o", "_type": "UMLParameter", "_id": "AAAAAAAFq7weqoCcUI6g=", "type": "Object", "direction": "in" }
{ "_parent": "AAAAAAAFq7weqoCcTngY=", "name": null, "_type": "UMLParameter", "_id": "AAAAAAAFq7weqoCcVxI0=", "type": "boolean", "direction": "return" }
{ "_parent": "AAAAAAAFq7weqoCcQE7I=", "name": "sdfsdf", "_type": "UMLGeneralization", "_id": "AAAAAAAFq7weqoCcRDg8=", "source": "AAAAAAAFq7weqoCcQE7I=", "target": "AAAAAAAFqpyZaw1HqYaU=" }
{ "_parent": "AAAAAAAFq7weqoCcQE7I=", "visibility": "private", "name": "sigCode", "_type": "UMLAttribute", "_id": "AAAAAAAFq7weqoCcSulQ=", "type": "long" }
```

其他

其他的一些操作在此不做过多描述，欢迎各位通过 `-h`（或 `--help`）参数查看帮助并探索。

注意事项

- 请确保构造函数正确实现，且类和构造函数均定义为 `public`，否则将无法进行实例化。
- 请保证传入的类继承了 `UmlGeneralInteraction` 接口，否则将无法载入。
- 此外，对于 `ClassNotFoundException`（全称 `com.oocourse.uml2.interact.exceptions.user.ClassNotFoundException`）等几个异常类，在Java的标准库里面有与之同名的类（全称 `java.lang.ClassNotFoundException`）。请各位在使用的时候注意甄别，以免误用。

其他

- 如果还有不清楚的地方，建议去阅读相关部分的源代码
 - 源码大部分地方均配有javadoc注释
 - 最关键的部分依然为一众 `Uml` 开头，且继承自 `UmlElement` 的类，以及各个继承自 `UserProcessException` 的异常类
- ~~一如既往地~~，本次作业依然在输出层面上分为加密版和非加密版
 - 非加密版完全公开。
 - 加密版只在评测机上使用且闭源，会对输出进行一定程度的加密处理。
 - ~~所以，请不要试图伪造输出，还请使用我们的接口。~~
 - 不仅如此，加密版本次编译时加入了源码混淆选项，所有非public的字段、方法、类以及方法实现都会被混淆。
 - ~~所以，请不要试图通过反射来破解接口。~~发现此类情况，也可以直接举报。