# Finding Jack The Ripper

Created by

Panida Yimsanga 6130304521
Chanat Tanasarnsopaporn 6231309521
2110215 Programming Methodology

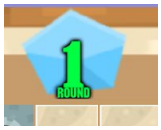
Semester 2 Year 2019
Chulalongkorn University

# Game: Finding Jack The Ripper

**Introduction**:

**Window game**


Round




**Character in choice**

## Item remove


## Turn

**Component**

Character

**Gin**


walk or ability

Walk : 1-3 step

Ability : Choose the character to walk

**Shadow man**


walk or ability

Walk : 1-3 step

Ability : Switch the position with another character

**Conan**


walk or ability
Walk : 1-3 step
Ability : Randomly cross the innocence out of the choice.

**Kogoro**




walk or ability
Walk : 1-3 step
Ability : move the lamp

**Heiji**


walk or ability
Walk : 1-3 step
Ability : move the hole cover

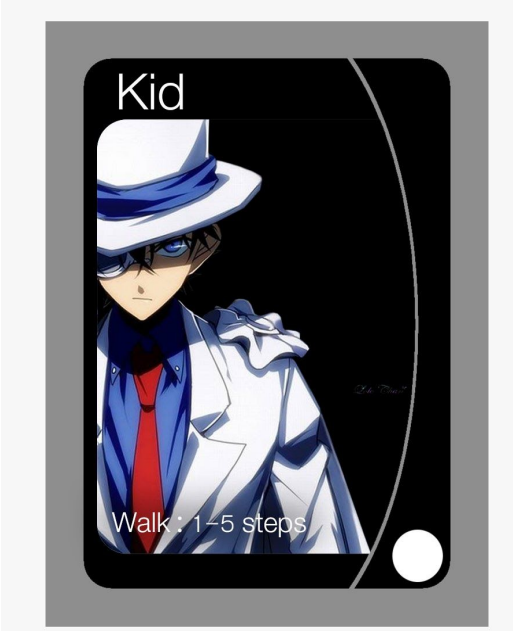
**Ran**


walk or ability
Walk : 1-3 step
Ability : move the exit barricade.

**Haibara**


walk and ability
Walk : 1-3 step
Ability : set the direction of the light

**Kid**




Only walk
Walk : 1-5 step

Item

**Lamp**


The lamp brightens the character around the Lamp.
The lamp can place on the lamp base only.

**Hole cover**


Cover the hole on map. Character can’t go in the hole.
The hole cover can place on the hole only.

**Exit barricade**


Barricade block the jack escape.
The exit barricade can place on the exit block only.

Map

**Wall**

Can’t pass this block

**Exit**



Mr.jack can escape from this block.

**Lamp base**

Base for the lamp

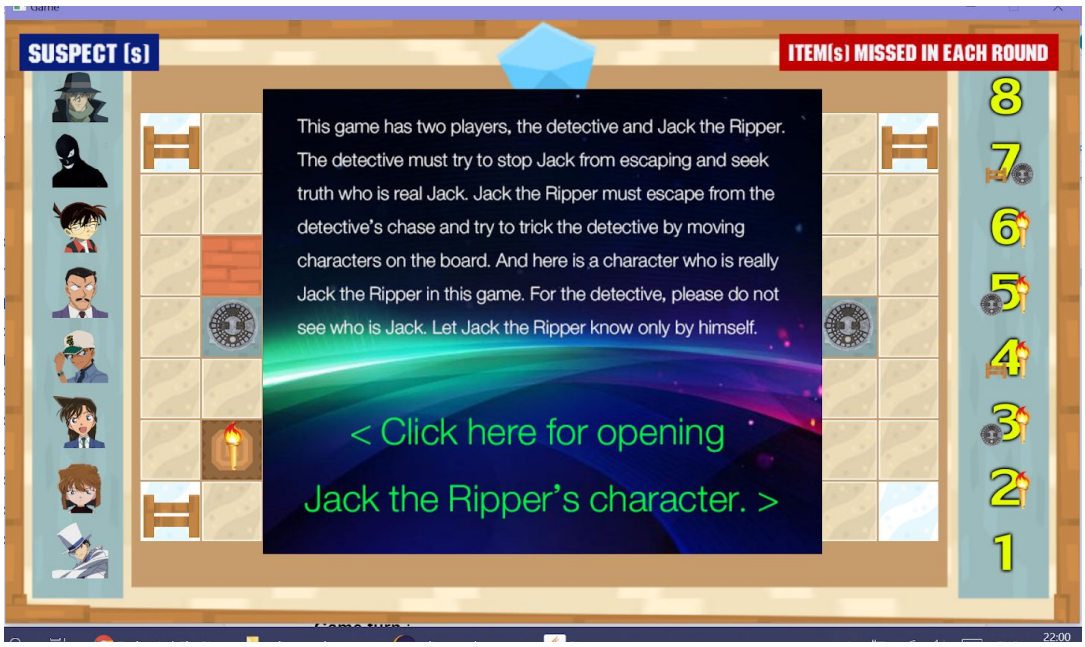
**Hole**

Character can warp to another hole

**Rule** :

The players choose who is to play Jack and who plays the detective.

One of the players plays Mr Jack. He will be called Jack in these rules. He is the only one who knows which identity he has borrowed. His goal is to escape the investigator before dawn (at the end of the eighth turn) or to leave the district by taking advantage of the darkness.

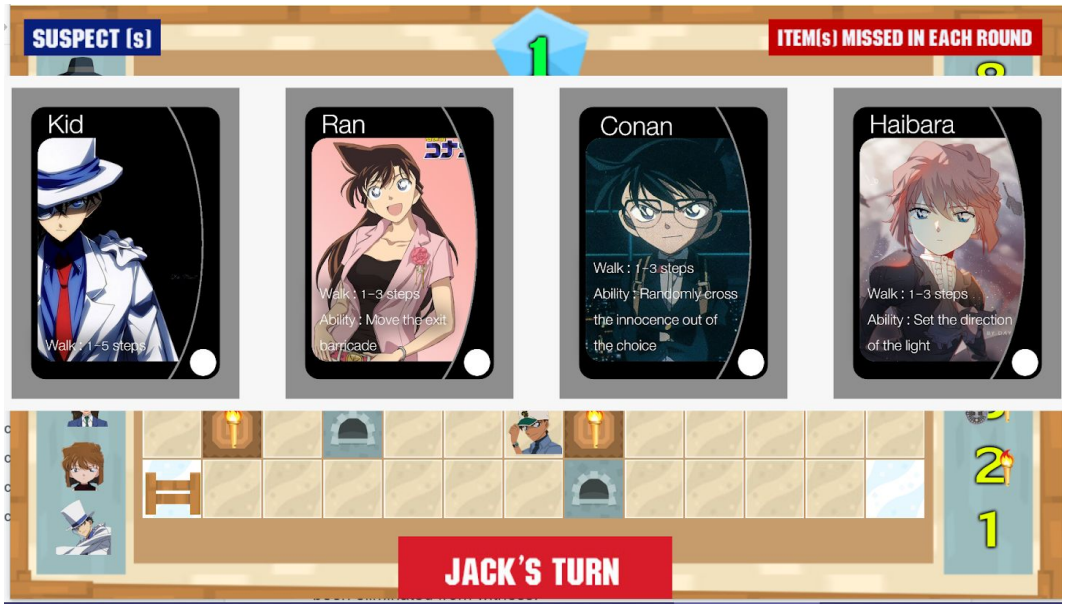



The other player plays the police detective. He will be called «the detective» in these rules. His goal is to uncover which investigator Jack is impersonating and to catch him before dawn.

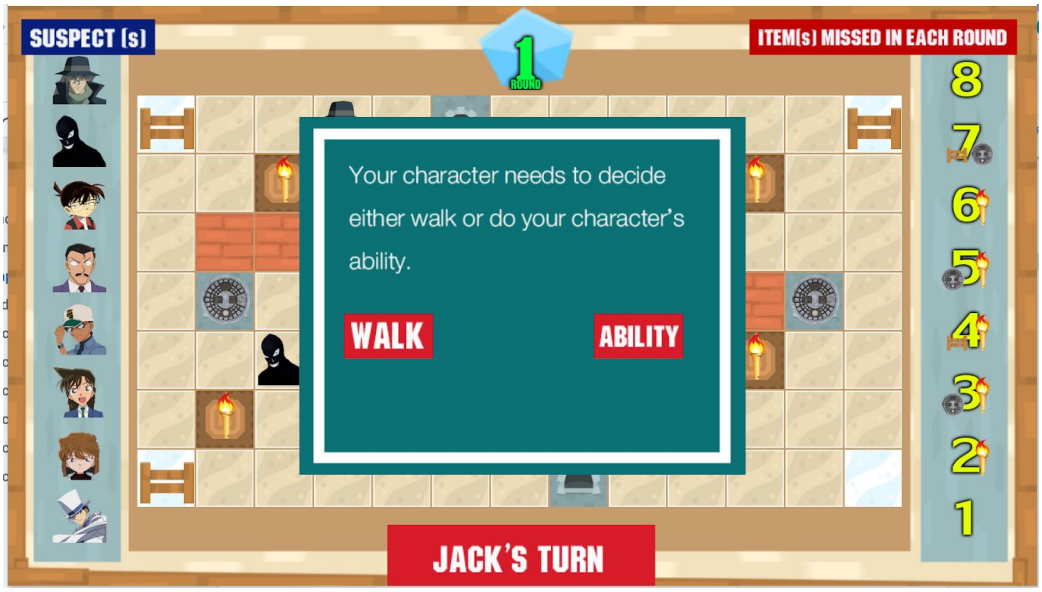
At each turn, four characters are used (two by the detective and two by Jack). Then there is a call for witnesses. Jack must now announce whether the character which he is impersonating is visible or not.

**Game turn** :

Choosing and using the character




Draw the four character from all character.Turn can see from bottom of program. Player in turn choose a character among the four and play it .

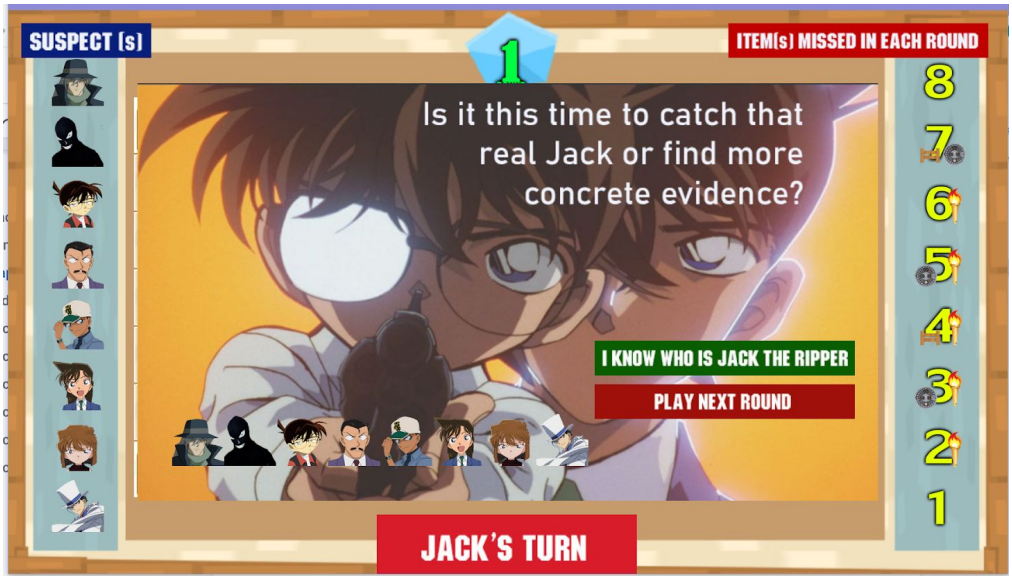



Character able to choose ability or walk except haibara and kid. Haibara can walk ,then set direction of light. Kid can walk maximum 5 step.

Jack is in light or dark


When the four character have been played. Program will automatic calculate. What status of jack,and show all players that jack is in light or dark. Character who is difference status from Jack will. This character have been eliminated from witness.

Is jack??




After character have been eliminate, detective have alternative to play next round or investigate from witness.

If detective choose investigate now,and select the correct jack **detective win .** But if it wrong, **Mr.jack win**

Remove item

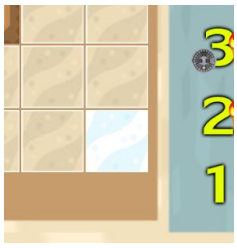
In each round,Item have been removed automatic by program random. Remove with current turn number in right of program.

**End of game :**

There are THREE possible ways to end a game:

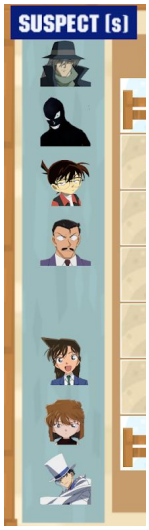
1. Jack leaves the district

Jack succeeds in moving his character out of the district, by taking the token through an exit that is not blocked by the exit barricade. Jack wins the game.




2. The detective catches Jack

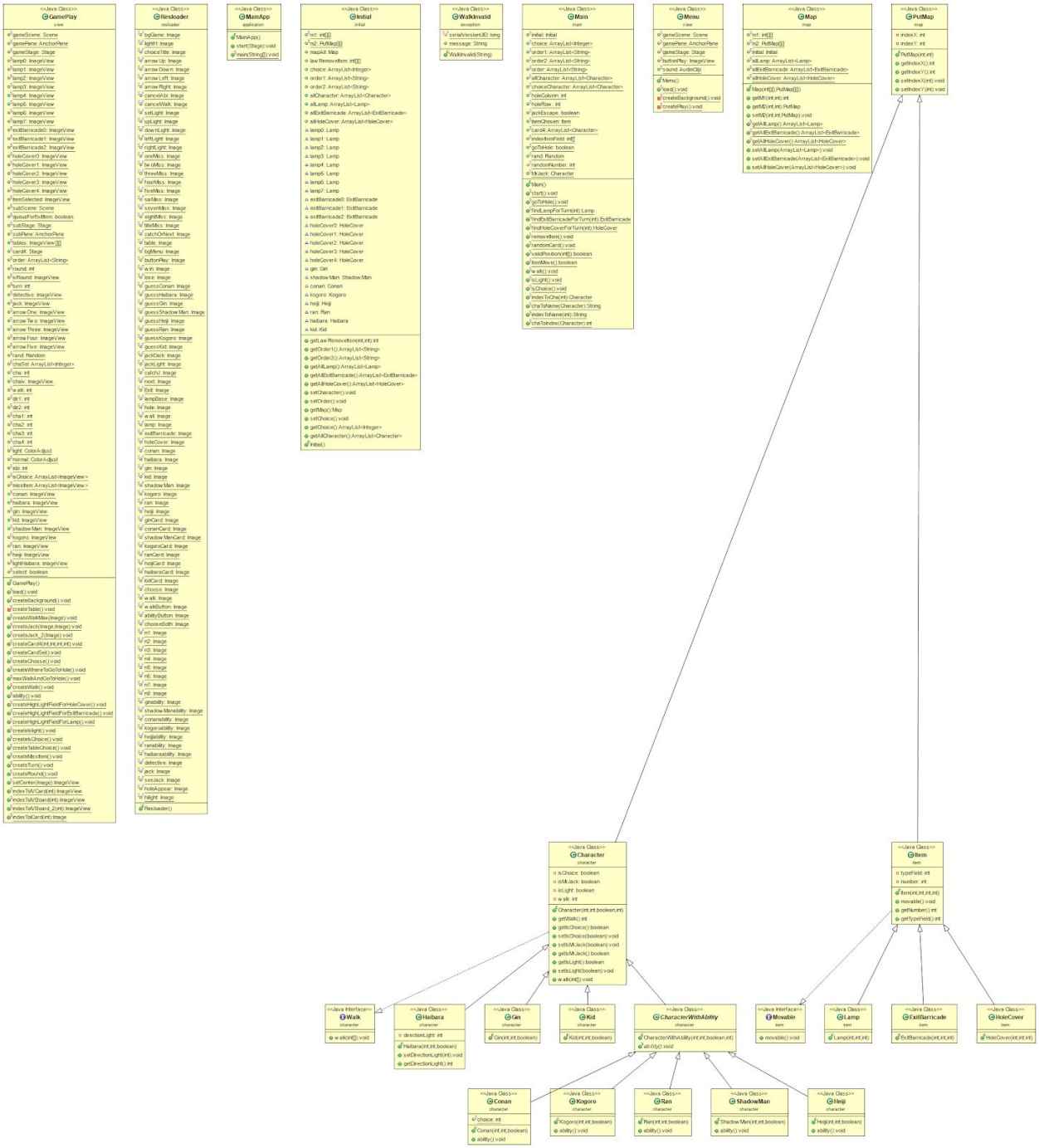
- If the accusation is right: the detective wins the game.
- If the accusation is wrong: Jack wins the game by taking advantage of the confusion caused by this miscarriage of justice to escape!


3. Jack is not caught

If, at the end of the eighth turn, Jack has not been caught, then Jack wins the game!

# Class diagram



1. **Package Application**

1.1 Class MainApp

- extends Application
- This is a class for starting and opening UI of this game.

1.1.1 Methods

| | |
|---|---|
| + void start(Stage primaryStage) | Initialize and create Menu class. |
| + *void main(String[] args)* | Use for running Menu page. |

2. **Package Character**

2.1 Class Character

- extends PutMap
- implements Walk
- It is a class generally for all characters in Finding Jack The Ripper.

2.1.1 Fields

| | |
|---|---|
| - boolean isChoice | Status for being suspect choices in the game. |
| - boolean isMrJack | Determine whether this character is Jack The Ripper. |
| - boolean isLight | Determine whether this character is near the light or not. |
| - int walk | Collect maximum walk steps of each character. |

2.1.2 Constructor

| | |
|---|---|
| + Character(int indexX, int indexY, boolean isLight, int walk) | - Set position of the character (indexX represents row, indexY represents column)<br>- Set maximum walk steps<br>- Set value of being by the light<br>- Initialize isChoice and isMrJack |

2.1.3 Methods

| | |
|---|---|
| + void walk(int[] k) | Get value of new position from array int[] k for walking on the game's map. |
| + int getWalk() | Getter method for walk steps |
| + boolean getIsChoice() | Getter method for isChoice |
| + boolean getIsLight() | Getter method for isLight |
| + boolean getIsMrJack() | Getter method for isMrJack |
| + void setIsLight(boolean isLight) | Setter method for isLight |
| + void setIsChoice(boolean isLight) | Setter method for isChoice |
| + void setIsMrJack(boolean isJack) | Setter method for isMrJack |

2.2 Class CharacterWithAbility

- extends Character
- This class is an abstract class. The objective is to add a simple ability method for some characters. Even though many characters have ability skills, some characters have complicated structure of coding ability methods. So characters

with complicated ability methods or no ability skills would not extend from this class, CharacterWithAbility.

2.2.1 Constructor

| | |
|---|---|
| + CharacterWithAbility(int indexX, int indexY, boolean isLight, int walk) | Initialize all values like Character class. |

2.2.2 Methods

| | |
|---|---|
| + abstract void ability() | Method for simple ability skills |

2.3 Class Conan

- extends CharacterWithAbility

2.3.1 Fields

| | |
|---|---|
| + *int choice* | Collect the representative number of the character that would be crossed out of the suspect choices. |

2.3.2 Constructor

| | |
|---|---|
| + Conan(int indexX, int indexY, boolean isLight) | Initialize all general values of the character. |

2.3.3 Methods

| | |
|---|---|
| + void ability() | - His ability skill is to randomly cross the innocence out of the suspect choice. Then collect |

| | |
|---|---|
| | random number in + *int choice.* |

2.4 Class Gin

- extends Character
- His ability skill is to choose another character to walk. According to using repeat walk processes, this class does not have an ability method as others. His skill method would be partially written in Class GamePlay and Class Main instead.

2.4.1 Constructor

| | |
|---|---|
| + Gin(int indexX, int indexY, boolean isLight) | Initialize all general values of the character. |

2.5 Class Haibara

- extends Character
- Her ability skill is to set the direction of her light.

2.5.1 Fields

| | |
|---|---|
| - int directionLight | Collection direction value of her light |

2.5.2 Constructor

| | |
|---|---|
| + Haibara(int indexX, int indexY, boolean isLight) | - Initialize all general values of the character.<br>- Initialize directionLight |

2.5.3 Methods

| | |
|---|---|
| + void setDirectionLight(int h) | Setter method for directionLight |
| + int getDirectionLight() | Getter method for directionLight |

2.6 Class Heiji

- extends CharacterWithAbility

2.6.1 Constructor

| | |
|---|---|
| + Heiji(int indexX, int indexY, boolean isLight) | Initialize all general values of the character. |

2.6.2 Methods

| | |
|---|---|
| + void ability() | His ability skill is to move a hole cover. |

2.7 Class Kid

- extends Character
- He has no ability skills.

2.7.1 Constructor

| | |
|---|---|
| + Kid(int indexX, int indexY, boolean isLight) | Initialize all general values of the character. |

2.8 Class Kogoro

- extends CharacterWithAbility

2.8.1 Constructor

| | |
|---|---|
| + Kogoro(int indexX, int indexY, boolean isLight) | Initialize all general values of the character. |

2.8.2 Methods

| | |
|---|---|
| + void ability() | His ability skill is to move a lamp. |

2.9 Class Ran

- extends CharacterWithAbility

2.9.1 Constructor

| | |
|---|---|
| + Ran(int indexX, int indexY, boolean isLight) | Initialize all general values of the character. |

2.9.2 Methods

| | |
|---|---|
| + void ability() | Her ability skill is to move an exit barricade. |

2.10 Class ShadowMan

- extends CharacterWithAbility

2.10.1 Constructor

| | |
|---|---|
| + ShadowMan(int indexX, int indexY, boolean isLight) | Initialize all general values of the character. |

2.10.2 Methods

| | |
|---|---|
| + void ability() | His ability skill is to switch the position with another character. |

2.11 Interface Walk

2.11.1 Methods

| | |
|---|---|
| + abstract void walk(int[] u) | Interface Walk for the character's |

| | |
|---|---|
| | walk method |

## 3. Package Exception

3.1 Class WalkInvalid

- extends Exception

3.1.1 Fields

| | |
|---|---|
| - *long **serialVersionUID*** | = 1567811503293933 97L |
| + String message | Message for Exception |

3.1.2 Constructor

| | |
|---|---|
| + WalkInvalid(String message) | Initialize message |

## 4. Package Initial

4.1 Class Initial

4.1.1 Fields

| | |
|---|---|
| + *int[][] m1* | Initialize m1 |
| + *PutMap[][] m2* | Initialize m2 |
| + Map mapAll | Initialize map |
| + int[][] lawRemoveItem | Initialize law item in any round |
| + ArrayList<Integer> choice | Initialize character in choice |
| + ArrayList<String> order1 | Initialize order first form |
| + ArrayList<String> order2 | Initialize order second form |
| + ArrayList<Character> allCharacter | Initialize arraylist all of character |

| | |
|---|---|
| + ArrayList<Lamp> allLamp | Initialize arraylist all of lamp |
| + ArrayList<ExitBarricade> allExitBarricade | Initialize arraylist all of exit barricade |
| + ArrayList<HoleCover> allHoleCover | Initialize arraylist all of hole cover |
| Lamp lamp0 | Initialize first lamp |
| Lamp lamp1 | Initialize second lamp |
| Lamp lamp2 | Initialize third lamp |
| Lamp lamp3 | Initialize fourth lamp |
| Lamp lamp4 | Initialize fifth lamp |
| Lamp lamp5 | Initialize sixth lamp |
| Lamp lamp6 | Initialize seventh lamp |
| Lamp lamp7 | Initialize eight lamp |
| ExitBarricade exitBarricade0 | Initialize first exit barricade |
| ExitBarricade exitBarricade1 | Initialize second exit barricade |
| ExitBarricade exitBarricade2 | Initialize third exit barricade |
| HoleCover holeCover0 | Initialize first hole cover |
| HoleCover holeCover1 | Initialize second hole cover |
| HoleCover holeCover2 | Initialize fourth hole cover |
| HoleCover holeCover3 | Initialize fifth hole cover |
| HoleCover holeCover4 | Initialize sixth hole cover |
| Gin gin | Initialize gin character |
| ShadowMan shadowMan | Initialize shadowMan character |
| Conan conan | Initialize conan character |

| | |
|---|---|
| Kogoro kogoro | Initialize kogoro character |
| Heiji heiji | Initialize heiji character |
| Ran ran | Initialize ran character |
| Haibara haibara | Initialize haibara character |
| Kid kid | Initialize kid character |

4.1.2 Constructor

| | |
|---|---|
| + Initial() | - Add all items to item arrays<br>- Set and initialize game map in Map class<br>- Set items and characters on game map<br>- Set order of player's turn<br>- Set the suspect choice |

4.1.3 Methods

| | |
|---|---|
| Getter and setter methods for initializing the game | |

## 5. Package Item

5.1 Class Item

- extends PutMap
- implements Movable

5.1.1 Fields

| | |
|---|---|
| - int typeField | Determine the specific type field which this item could place on |

| | |
|---|---|
| - Int number | Collect a number for item reference (ex: holeCover1, lamp0, ...) |

5.1.2 Constructor

| | |
|---|---|
| + Item(int indexX, int indexY, int typeField, int number) | Initialize row and column position for an item and set all general values and information. |

5.1.3 Methods

| | |
|---|---|
| + void movable() | Move item position to another position |
| + getter methods of all fields | |

5.2 Class ExitBarricade

- extends Item

5.2.1 Constructor

| | |
|---|---|
| + ExitBarricade(int indexX, int indexY, int number) | Initialize all general values of an exit barricade |

5.3 Class HoleCover

- extends Item

5.3.1 Constructor

| | |
|---|---|
| + HoleCover(int indexX, int indexY, int number) | Initialize all general values of a hole cover |

5.4 Class Lamp

- extends Item

5.4.1 Constructor

| | |
|---|---|
| + Lamp(int indexX, int indexY, int number) | Initialize all general values of a lamp |

5.5 Interface Movable

5.5.1 Methods

| | |
|---|---|
| + abstract void movable() | Interface Movable for item move process |

## 6. Package Main

6.1 Class Main

- It is a class for the game's logic.

6.1.1 Fields

| | |
|---|---|
| + *Initial initial* | Create Initial class for initializing the whole game's values and objects |
| + *ArrayList<Integer> choice* | Get the suspect choice's array list from Initial class |
| + *ArrayList<String> order1* | Get the first order of game turns from Initial class |
| + *ArrayList<String> order2* | Get the second order of game turns from Initial class |
| + *ArrayList<String> order* | Initialize the whole game's order of turns |
| + *ArrayList<Character> allCharacter* | Get an array list of all characters |
| + *ArrayList<Character> choiceCharacter* | Initialize the suspect choice |

| | |
|---|---|
| + *int holeColumn* | Collect the hole's column for moving the character to another hole |
| + *int holeRow* | Collect the hole's row for moving the character to another hole |
| + *boolean jackEscape* | Check the game's status whether Jack The Ripper have escaped to the exit |
| + *Item itemChosen* | Determine the chosen item for itemMove method of some characters' ability |
| + *ArrayList<Character> card4* | Initialize the cards for choosing turns by players in every round |
| + *int[] indexItemField* | Collect the field where a player wants to place an item |
| + *boolean goToHole* | Check the status whether the character is moving from recent hole to another hole successfully |
| + *Random rand* | Initialize Random class for random function |
| - *int randomNumber* | Collect the number from randoming |
| + *Character MrJack* | Collect a character who is the real Jack The Ripper in this game |

6.1.2 Methods

| | |
|---|---|
| + *int chaToIndex(Character cha)* | Convert from a particular character into a specific index number |
| + *String chaToName(Character character)* | Convert from a particular character into its name |

| | |
|---|---|
| + *ExitBarricade findExitBarricadeForTurn(int x)* | Return a particular exit barricade from its specific number. |
| + *HoleCover findHoleCoverForTurn(int x)* | Return a particular hole cover from its specific number. |
| + *Lamp findLampForTurn(int x)* | Return a particular lamp from its specific number. |
| + *void goToHole()* | Process when the character has walked to the hole, requiring moving to another hole |
| + *Character indexToCha(int index)* | Convert from a specific index number into a particular character in the game |
| + *String indexToName(int index)* | Convert from a specific index number into the character's name |
| + *void isChoice()* | Check the possible suspect choice after changing characters and items' position |
| + *void isLight()* | Check whether Jack The Ripper is near the light or in the dark place |
| + *boolean itemMove()* | Move the chosen item and return true or false whether it is possible to move |
| + *void randomCard()* | Random 4 cards in each round for choosing game actions from players' turn |
| + *void removeItem()* | Remove random items on the game's field in each round |
| + *void start()* | Perform the starting point of the game |

| | |
|---|---|
| + *boolean validPosition(int[] pos)* | Check whether the position of row and column is possibly on the game's field |
| + *void walk() throws WalkInvalid* | Perform the character's walking processes in the whole game |

## 7. Package Map

### 7.1 Class Map

This class contain all data on map in the game

#### 7.1.1 Fields

| | |
|---|---|
| + *int[][] m1* | map |
| + *PutMap[][] m2* | the object on the map |
| + *Initial initial* | Initialize the data in the game |
| + *ArrayList<Lamp> allLamp* | All data of Lamp |
| + *ArrayList<ExitBarricade> allExitBarricade* | All data of ExitBarricade |
| + *ArrayList<HoleCover> allHoleCover* | All data of HoleCover |

#### 7.1.2 Constructor

| | |
|---|---|
| + Map(int[][] m1,int PutMap[][] m2) | Set map for game and characters |

#### 7.1.3 Methods

| | |
|---|---|
| + *int getM1(int i, int j)* | Getter method for m1 |
| + *PutMap getM2(int i, int j)* | Getter method for m2 |
| + *void setM2(int i, int j, PutMap item)* | Setter method for m2 |
| + *ArrayList<Lamp> getAllLamp()* | Getter method for allLamp |
| + *ArrayList<ExitBarricade> getAllExitBarricade()* | Getter method for allExitBarricade |
| + *ArrayList<HoleCover> getAllHoleCover()* | Getter method for allHoleCover |

7.2 Class PutMap

This class contain all data on map in the game

7.2.1 Fields

| | |
|---|---|
| - int indexX | Position row |
| - int indexY | Position column |

7.2.2 Constructor

| | |
|---|---|
| + PutMap(int indexX, int indexY) | Initialize position |

7.2.3 Methods

| | |
|---|---|
| + int getIndexX() | Getter method for indexX |
| + int getIndexY() | Getter method for indexY |

| | |
|---|---|
| + void setIndexX(int indexX) | Setter method for indexX |
| + void setIndexY(int indexY) | Setter method for indexY |

**8. Package Resloader**

8.1 Class Resloader

- This class contains all resources which are necessary for the game's UI such as backgrounds, pictures of all buttons, audio files, etc.

**9. Package View**

9.1 Class Menu

- This is a page of the game's menu where a play button is located.

9.1.1 Fields

| | |
|---|---|
| + *Scene gameScene* | The game scene of Menu page |
| + *AnchorPane gamePane* | The whole Menu pane styled in an anchor pane |
| + *Stage gameStage* | Initialize the game stage |
| + *ImageView buttonPlay* | Initialize the picture for a play button |
| + *AudioClip sound* | Initialize music for menu |

9.1.2 Methods

| | |
|---|---|
| + *void load()* | Initialize and create the whole page of Menu |
| - *void createBackground()* | Create the Menu's background |
| - *void createPlay()* | Create a play button and set the |

| | |
|---|---|
| | condition when a button is clicked |

9.2 Class GamePlay

- This page is the whole game's page

9.2.1 Fields

| | |
|---|---|
| + *Scene gameScene* | Scene for GamePlay |
| + *AnchorPane gamePane* | The whole game's pane styled with AnchorPane |
| + *Stage gameStage* | The game's stage |
| + *ImageView lamp0* | pictures for lamps in the game |
| + *ImageView lamp1* | |
| + *ImageView lamp2* | |
| + *ImageView lamp3* | |
| + *ImageView lamp4* | |
| + *ImageView lamp5* | |
| + *ImageView lamp6* | |
| + *ImageView lamp7* | |
| + *ImageView exitBarricade0* | pictures for exit barricades in the game |
| + *ImageView exitBarricade1* | |
| + *ImageView exitBarricade2* | |
| + *ImageView holeCover0* | pictures for hole covers in the game |
| + *ImageView holeCover1* | |
| + *ImageView holeCover2* | |
| + *ImageView holeCover3* | |

| | |
|---|---|
| + *ImageView holeCover4* | |
| + *ImageView itemSelected* | Collect information of the selected item in the game |
| + *Scene subScene* | Subscene for another window aside from the main game's window |
| + *boolean queueForExitItem* | Check the status whether moving item's ability have been finished |
| + *Stage subStage* | Stage for sub windows |
| + *AnchorPane subPane* | Style sub window's pane with AnchorPane |
| + *ImageView[][] tables* | Background picture for each grid of the game's board (or we call it a map) |
| + *Stage card4* | Stage for card selection pop up window |
| + *ArrayList<String> order* | Collect player's turn whether Detective or Jack The Ripper |
| + *int round* | Collect the number of each round |
| + *ImageView ivRound* | Each round's number image |
| + *int turn* | Collect the number of turn |
| + *ImageView detective* | Image that tells Detective's turn |
| + *ImageView jack* | Image that tells Jack The Ripper's turn |
| + *ImageView arrowOne* | Buttons for walking action:<br>- arrowOne to arrowFour are walking directions<br>- arrowFive is for stopping this action |
| + *ImageView arrowTwo* | |
| + *ImageView arrowThree* | |

| | |
|---|---|
| + *ImageView arrowFour* | |
| + *ImageView arrowFive* | |
| + *Random rand* | Function for randoming numbers |
| + *ArrayList<Integer> chaSel* | Collect the characters which are selected already in each round |
| + *int cha* | Collect the representative number of the recently selected character |
| + *ImageView chaiv* | The selected character's picture in the game |
| + *int walk* | Collect the recent walk steps |
| + *int dir1* | The representative walk direction(left right up down) which is converted to the direction for map's array index |
| + *int dir2* | |
| + *int cha1* | Collect the represent number of each character that is possibly in character card selection |
| + *int cha2* | |
| + *int cha3* | |
| + *int cha4* | |
| + *ColorAdjust light* | Adjust the image more light |
| + *ColorAdjust light* | Adjust the image into normal |
| + *int abi* | Collect choosing the character's ability |
| + *ArrayList<ImageView> isChoice* | Image for all suspects in the choice |
| + *ArrayList<ImageView> missItem* | Image for the deleted items in each round |

| | |
|---|---|
| + *ImageView conan* | Image for all characters on the game's board (or map) |
| + *ImageView kid* | |
| + *ImageView haibara* | |
| + *ImageView ran* | |
| + *ImageView kogoro* | |
| + *ImageView gin* | |
| + *ImageView shadowMan* | |
| + *ImageView heiji* | |
| + *ImageView LightHaibara* | Image for light of haibara |
| + *boolean select* | status of select item |

9.2.2 Methods

| | |
|---|---|
| + *void load()* | Load all components of the whole GamePlay: map, characters, items, suspect choice, backgrounds, pictures, song |
| + *void createBackground()* | Set the background of GamePlay |
| - *void createTable()* | Create pictures of all map components such as fields, items, and characters |
| + *void createWalkMax(Image i)* | Create pop up picture that tells the player it is now to walk |
| + *void createJack(Image i1, Image i2)* | Create pop up picture for showing real Jack The Ripper's character and warning Detective not to see Jack The Ripper's character while opening |

| | |
|---|---|
| + *void createJack_2(Image i)* | Open real Jack The Ripper's character card |
| + *void createCard4(int i1,int 2,int i3,int i4)* | Initialize 4 random character cards for selection in each round |
| + *void createCardSel()* | Character card selection method |
| + *void createChosse()* | Create pop up picture of choosing walking action or ability action for the selected character |
| + *void createWhereToGoToHole()* | Function when the selected character recently walks to the hole and the character must choose another hole to appear |
| + *void maxWalkAndGoToHole()* | Function after finishing hole warping with the condition of the final walk step<br><br>Note: The objective is to create sub methods of finishing another hole appearance for special conditions and not to get bugged with another normal condition of game processes. |
| + *void createWalk()* | Function when the character's walking action is selected |
| + *void ability()* | Function when the character's ability action is selected |
| + *void createHighLightFieldForHoleCover()* | Function when the ability action of moving a hole cover is selected and highlight possible fields where a hole cover can move |
| + *void createHighLightFieldForExitB* | Function when the ability action of moving an exit barricade is selected |

| | |
|---|---|
| *arricade()* | and highlight possible fields where an exit barricade can move |
| + *void createHighLightFieldForLamp ()* | Function when the ability action of moving a lamp is selected and highlight possible fields where a lamp can move |
| + *void createIslight()* | Function after finishing all 4 turns of selection and show the status of Jack The Ripper whether he is near the light all in the dark place |
| + *void createIsChoice()* | Show the pop up window whether catch Jack The Ripper in the suspect choice or play next round after warning Jack The Ripper's status of being near light or in the dark place |
| + *void createTableChoice()* | Initialize all characters' picture in the suspect choice |
| + *void createMissItem()* | Show all item deletion pictures in all rounds |
| + *void createTurn()* | Show whose turn is whether Detective or Jack The Ripper |
| + *void createRound()* | Show the round's number picture |
| + *ImageView setCenter(Image i)* | Set the center position of any image you want |
| + *ImageView indexToIVCard(int i)* | Convert the representative number of the character into the character card's image view |
| + *ImageView indexToIVBoard(int i)* | Convert the representative number of the character into the character's picture on the game's board (or |

| | |
|---|---|
| | map) |
| + *ImageView indexToIVBoard_2(int i)* | Create all suspects' picture in createIsChoice() 's pop up window |
| + *Image indexToICard(int i)* | Return Image from the representative number of the character |