Reverse Engineering


```
Movie
_tilte (string)
_priceCode (int)
Children (int)
New release (int)
Regular (int)

public Movie(String title, int priceCode)
public int getPriceCode()
public void setPriceCode(int priceCode)
public string getTitle()
```

```
Customer
_name(string)

public void addRental(Rental rental)
public String getName()
public String statement()
```

```
Rental
private int _daysRented;
private Movie _movie;

public Rental(Movie movie, int daysRented)
 public int getDaysRented()
public Movie getMovie()
```
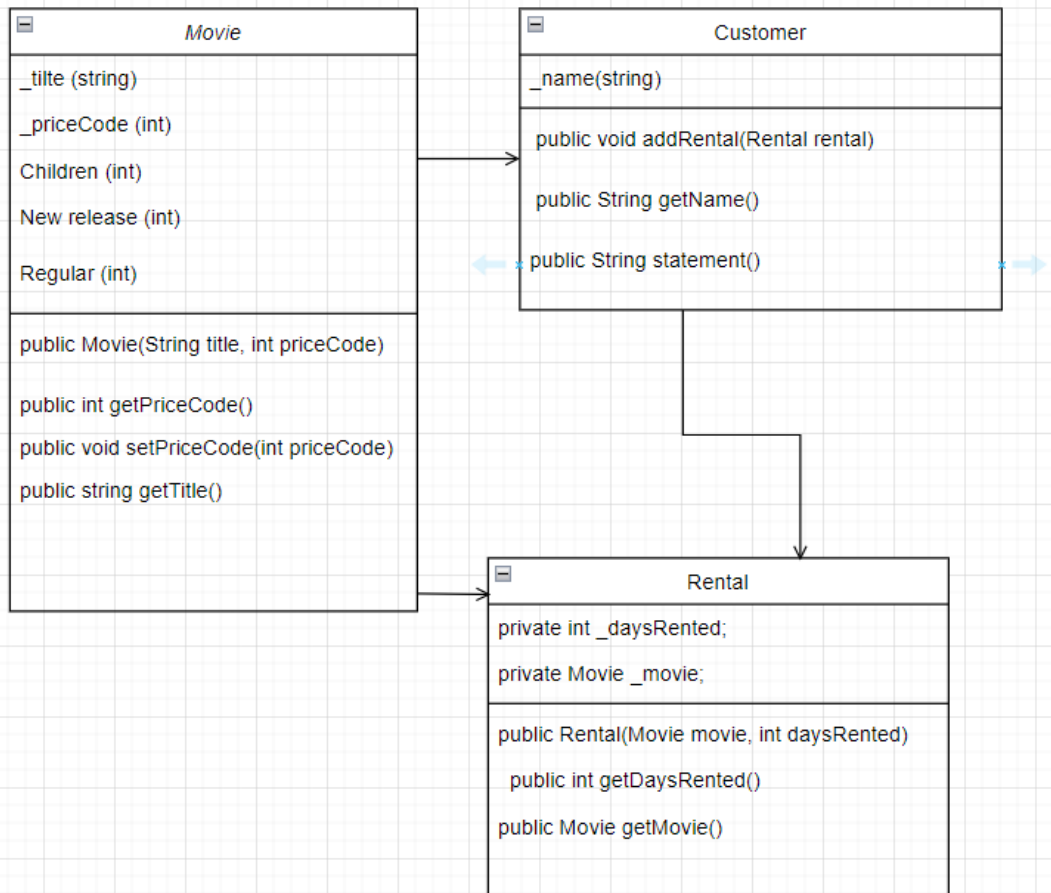
package model;

```
public static void main() {

    // Create movies

    Movie rogueOne = new Movie("Rogue One", Movie.NEW_RELEASE);

    Movie frozen = new Movie("Frozen", Movie.CHILDRENS);

    Movie starWarsIII = new Movie("Star Wars III", Movie.REGULAR);


    // Create rentals

    Rental rental1 = new Rental(rogueOne, 5);

    Rental rental2 = new Rental(frozen, 7);
```

```
        Rental rental3 = new Rental(starWarsIII, 4);


        // Create customer

        Customers customer = new Customers("New customer");


        // Add rentals to the customer

        customer.addRental(rental1);

        customer.addRental(rental2);

        customer.addRental(rental3);


        // Display customer information

        System.out.println(customer.statement());

    }
```

Criticisms

Readability:

- We should add more comments to explain complex parts of the code.
- Break down the public string statement() in Customer.java into smaller pieces.

Maintainability

- Organize code into modules or packages to promote code modularity.

Reusability

- Re-write the public string statement() functions in Customer.java to be reused in different contexts.

Extensibility