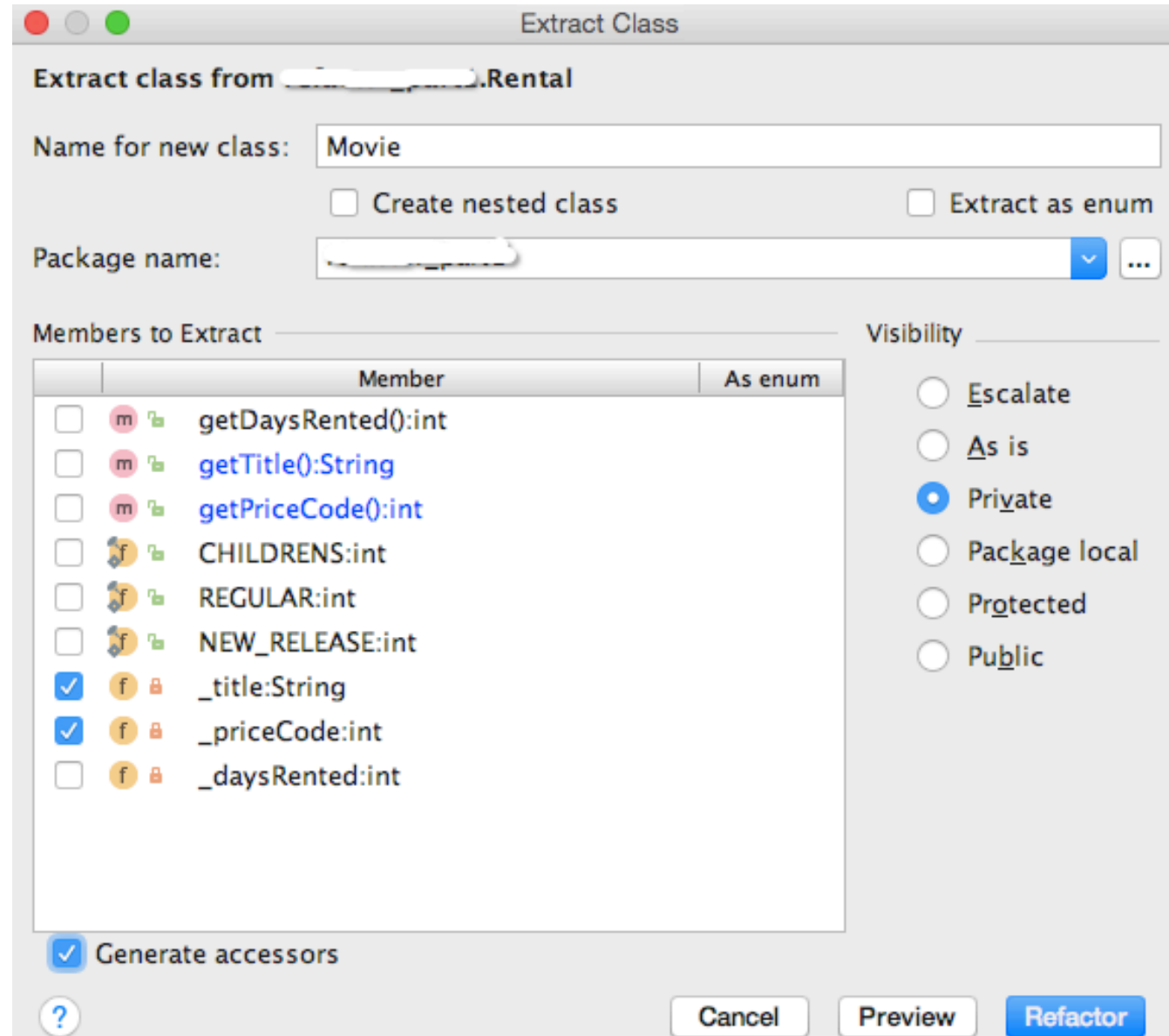


Refactor (Idea)

Additional slide of Refactor
zhuym

Extract Class

- Right click class
- -> refactor
- -> Extract
- -> Delegate



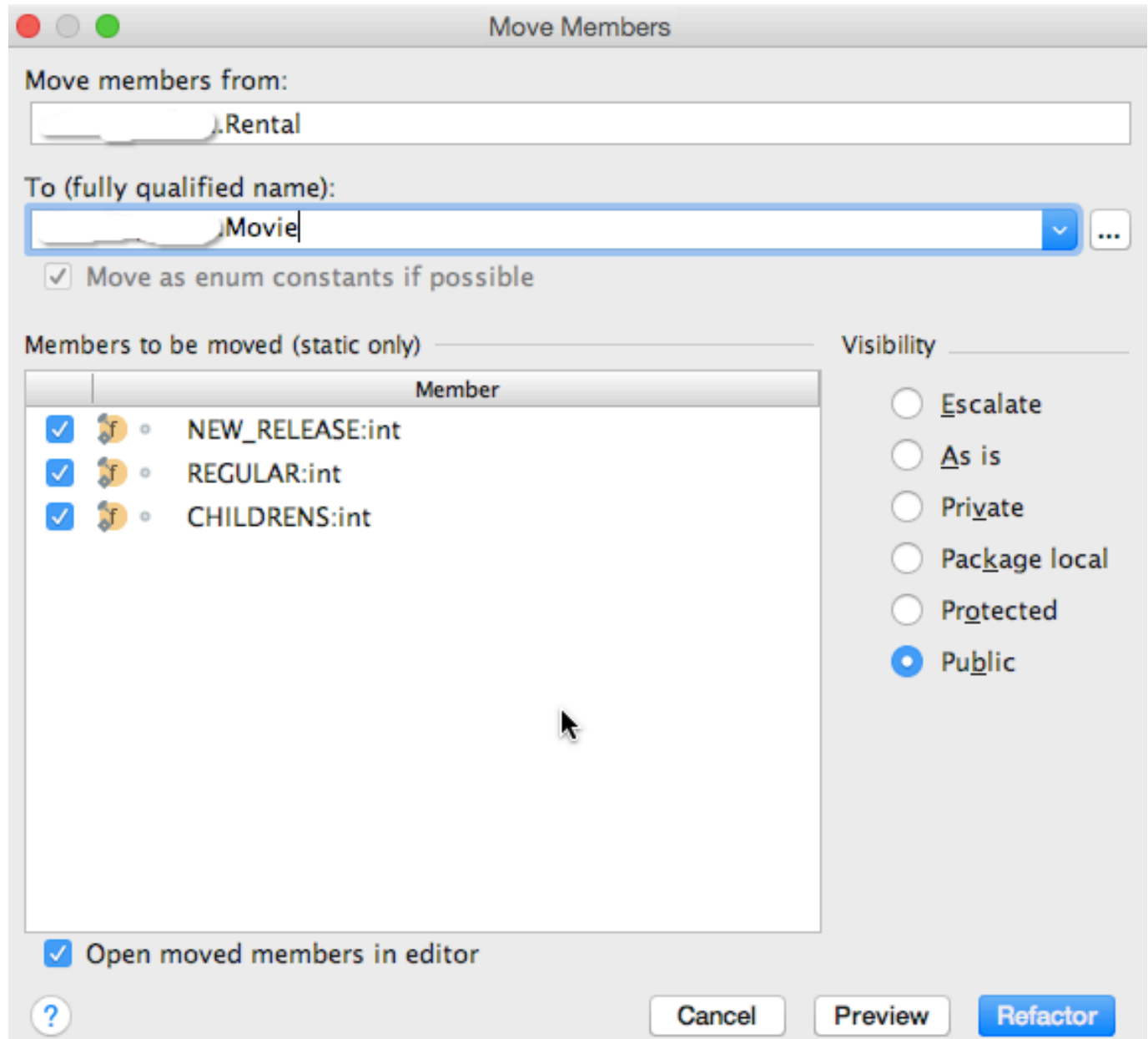
Rename of method

- Right click
method
- -> refactor
- -> rename

```
public class Movie {  
    private String _title;  
  
    public String getTitle() { return _title; }  
  
    public void setTitle(String _title) { this._title = _title; }
```

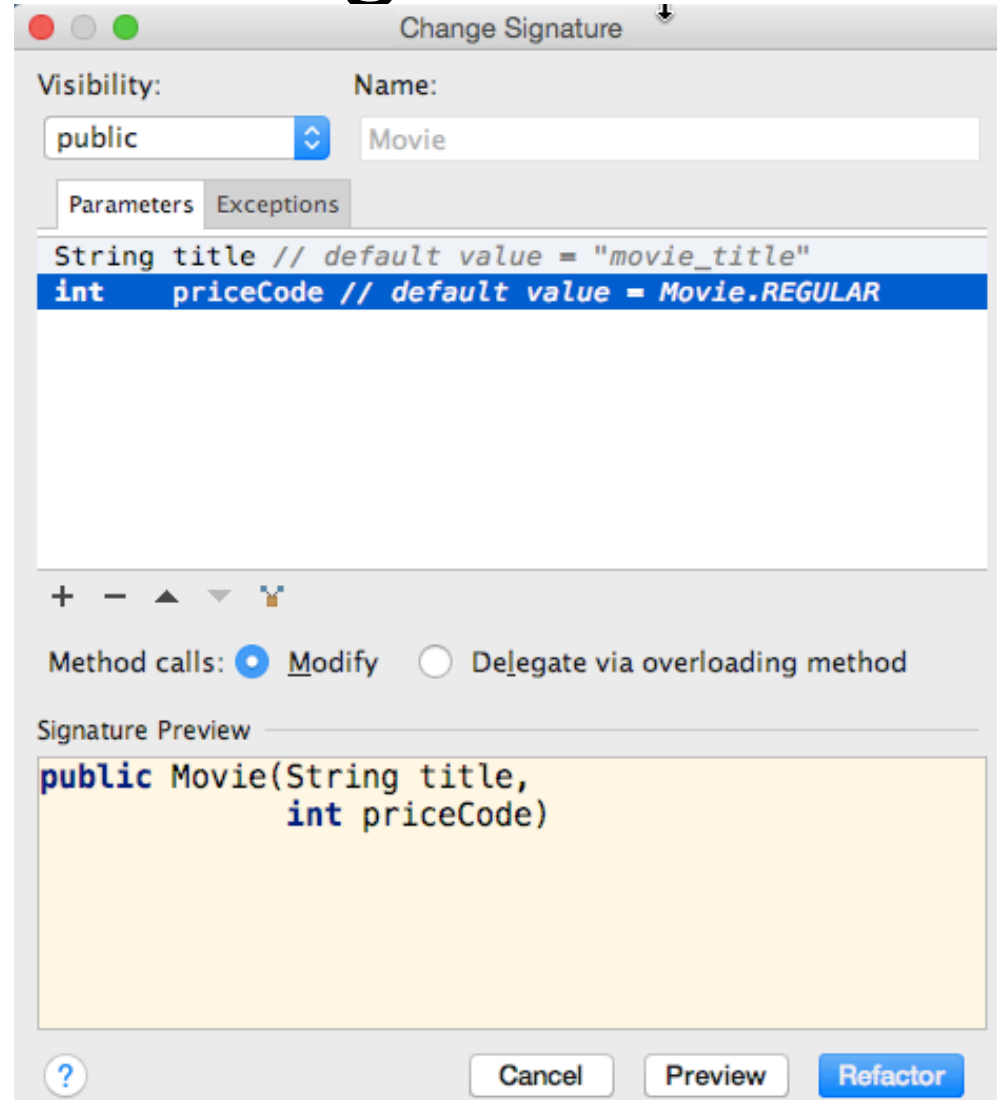
Move field

- Right click field
- -> refactor
- -> Move



Change Method Signature

- Right click
constructor of Movie
- -> refactor
- -> Change Signature



```
class Rental {  
    private final Movie movie = new Movie( title: "movie_title", Movie.REGULAR);  
}
```

Introduce Parameter Object

- Right click constructor of Rental
- -> refactor
- -> Extract
- -> Parameter object

Method to extract parameters from

☐ Keep method as delegate

Parameter Class

☒ Create new class

Name

Package name ...

Target destination directory: ...

☐ Create inner class

Name

☐ Use existing class

Name ...

☒ Escalate visibility

☒ Generate accessors

Parameters to Extract

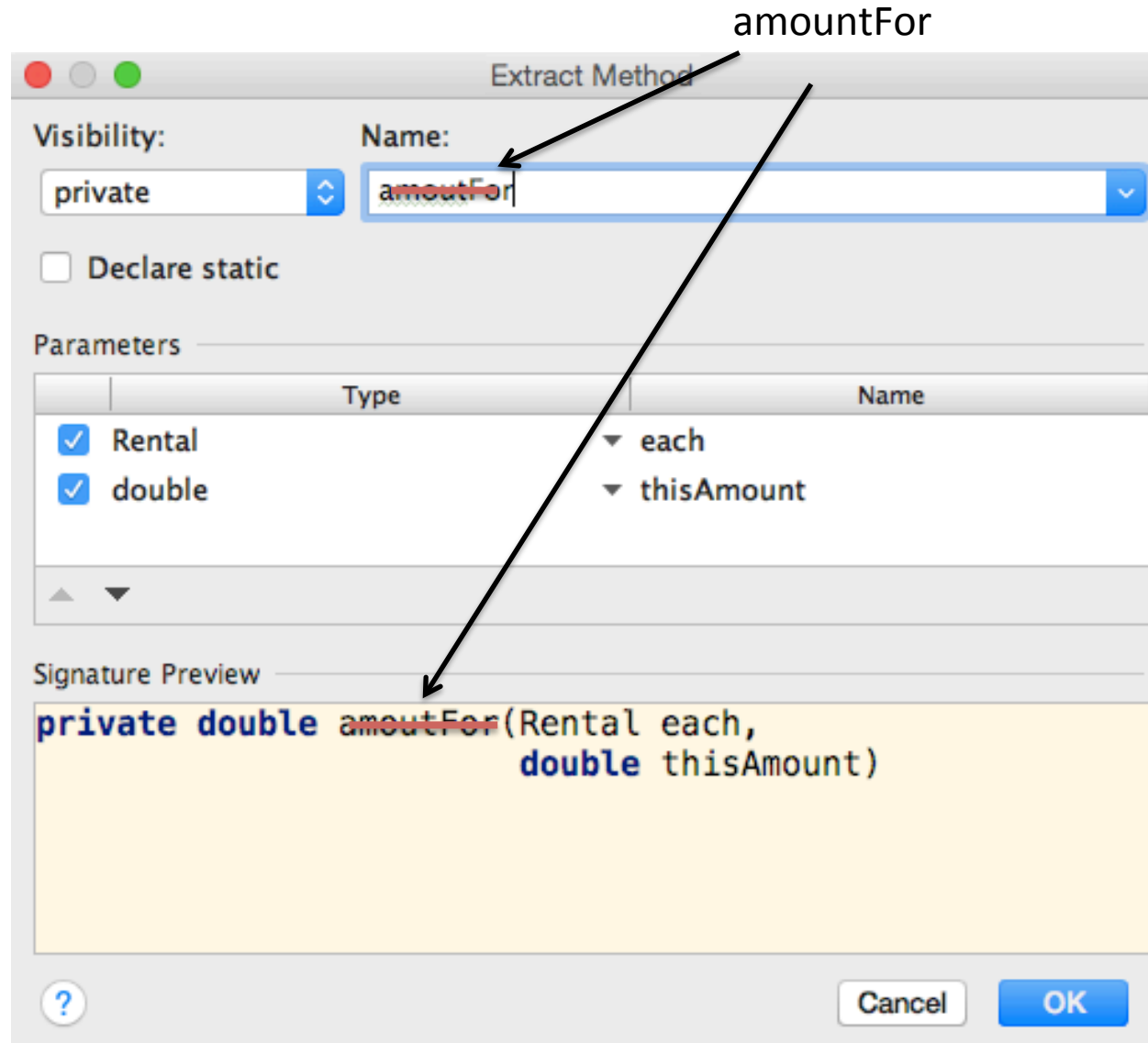
	Type	Name
<input type="checkbox"/>	Movie	▼ movie
<input checked="" type="checkbox"/>	Date	▼ start
<input checked="" type="checkbox"/>	Date	▼ end

Cancel Preview Refactor

DateRange

Extract Method

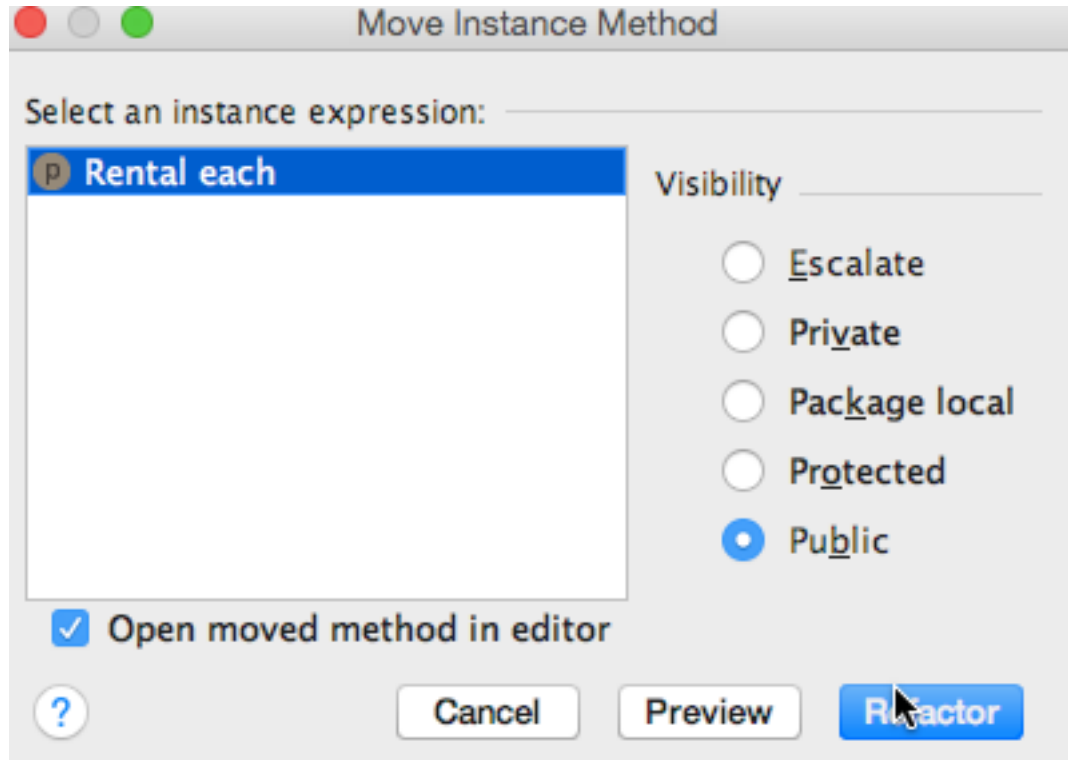
- Right click selected code
- -> refactor
- -> Extract
- -> Method



amountFor

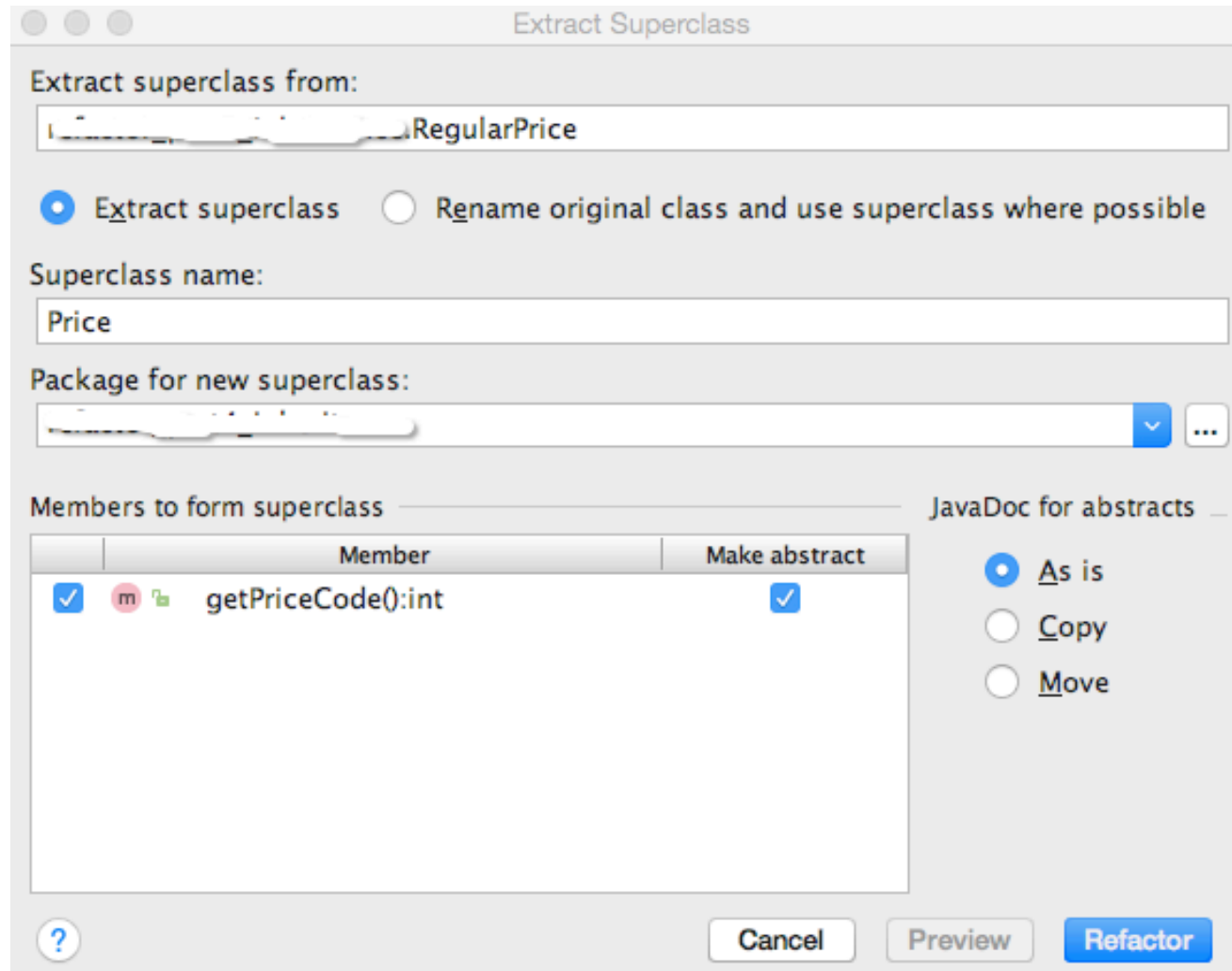
Move Method

- Right click method name
- -> refactor
- -> Move



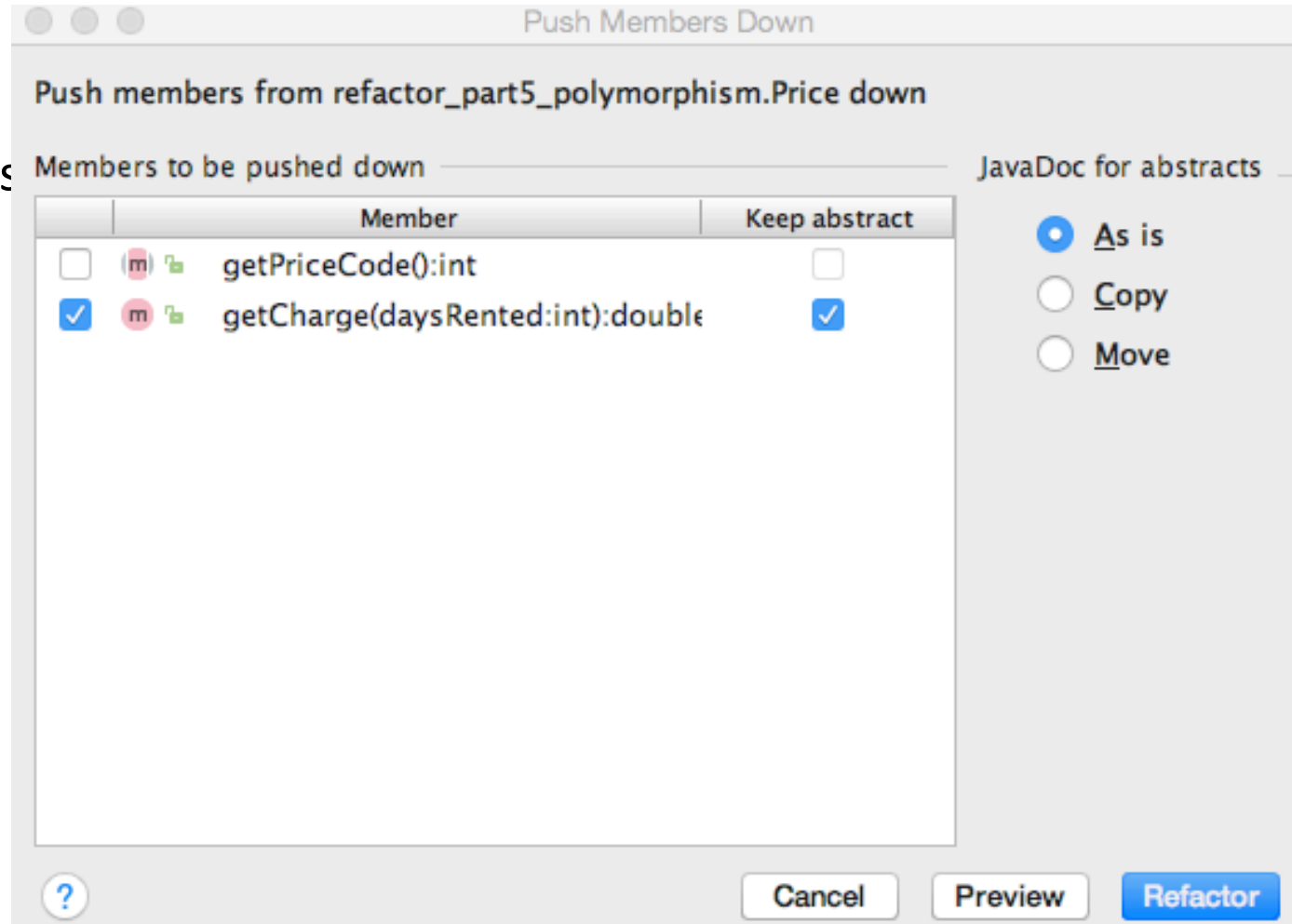
Extract Superclass

- Right click one of three classes
- -> refactor
- -> Extract
- -> Superclass



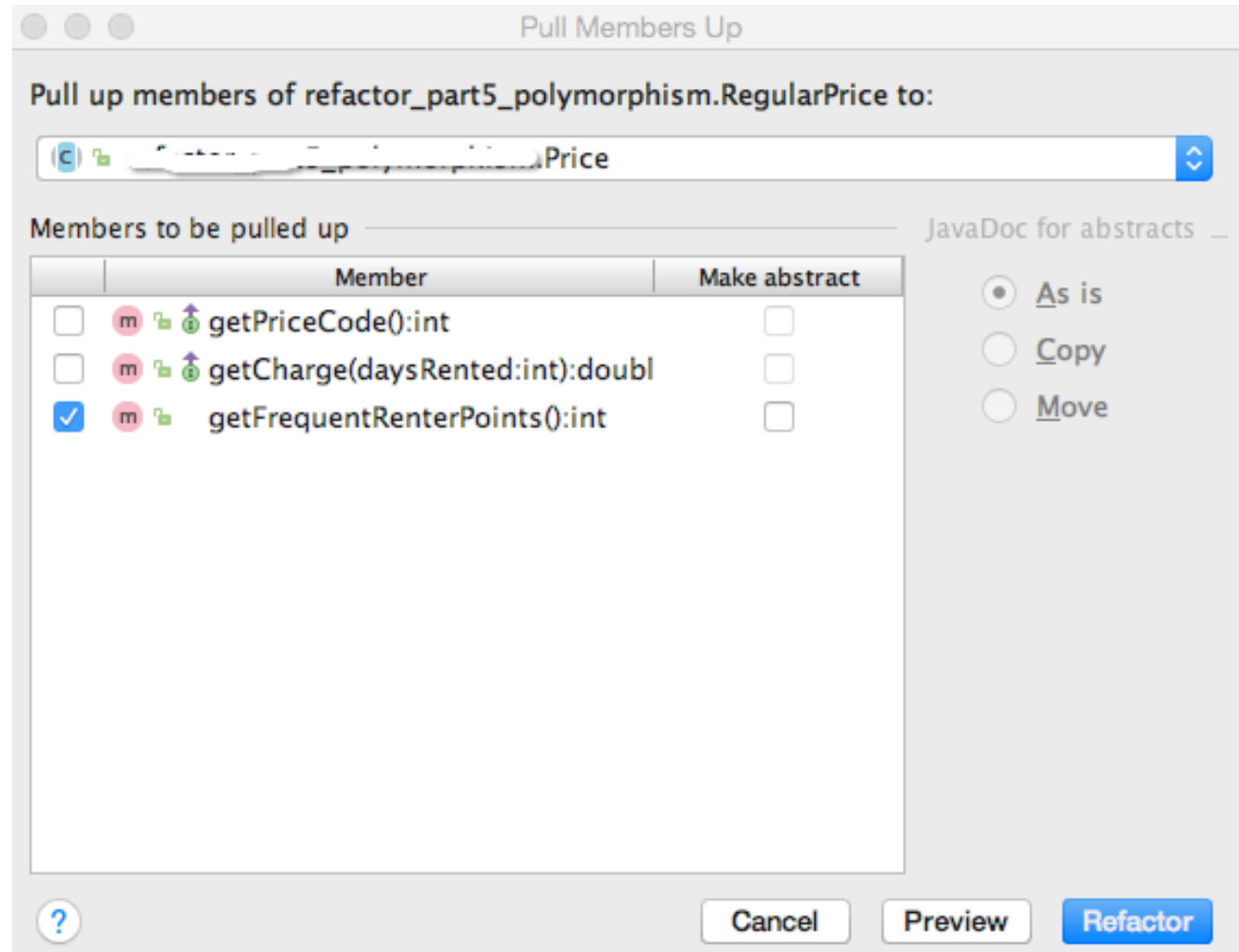
Push down methods

- Right click Price
- -> refactor
- ->Push Members Down



Pull up methods

- Right click sub class of Price
- -> refactor
- -> Pull Members Up



Inline method

- Right click
getAnswer1()
- -> refactor
- -> Inline...

