Typical Refactorings

Class Refactorings	Method Refactorings	Attribute Refactorings
add (sub)class to hierarchy	add method to class	add variable to class
rename class	rename method	rename variable
remove class	remove method	remove variable
Extract class	push method down	push variable down
	push method up	pull variable up
	add parameter to method	create accessors
	move method	abstract variable
	extract code in new method	
	replace parameter with method	

Extract Method

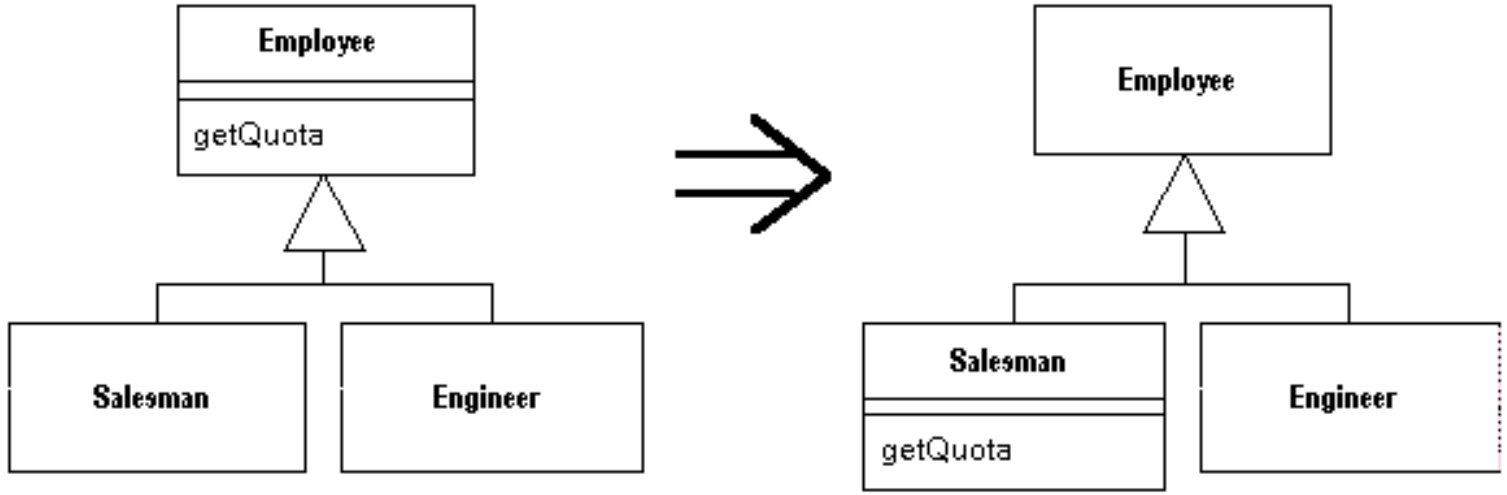
```
void printOwing() {
  printBanner();
  //print details
  System.out.println ("name:       " + _name);
  System.out.println ("amount     " + getOutstanding());
  }
```

```
void printOwing() {
  printBanner();
  printDetails(getOutstanding());
}


void printDetails (double outstanding) {
  System.out.println ("name: " + _name);
  System.out.println ("amount " + outstanding);
}
```

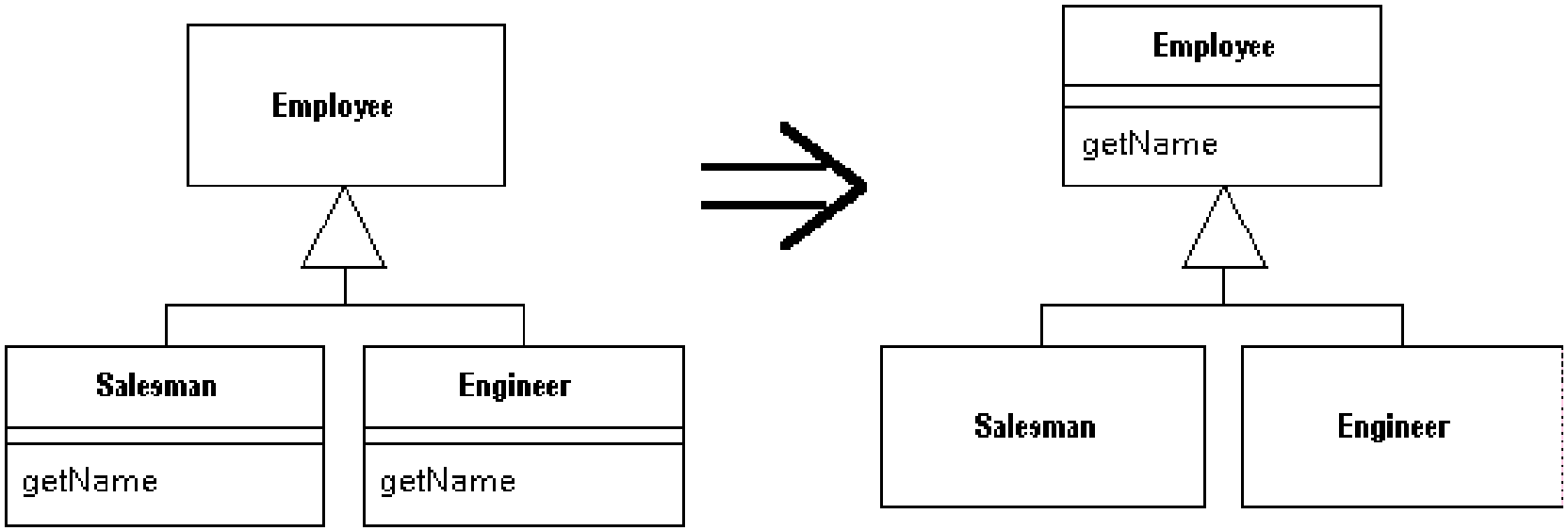
Push method down

- Behavior of a superclass is relevant only for some of its subclasses.
- Move it to those subclasses.



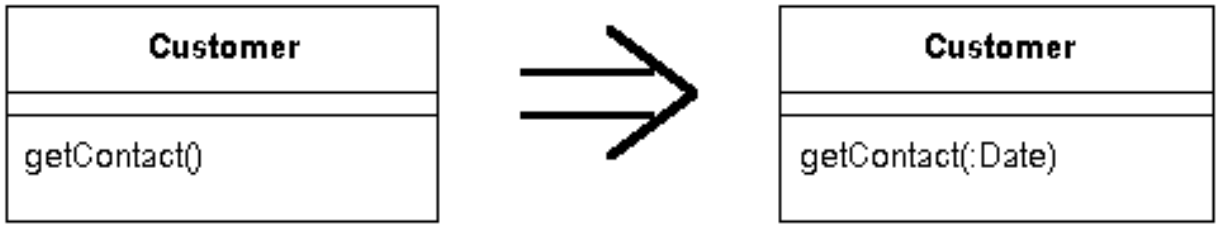
Push method up

- You have methods with identical results on subclasses.
- Move them to the superclass



Add parameter to method

- A method needs more information from its caller.
- Add a parameter for an object that can pass on this information.



Replace Parameter with Method

```
int basePrice = _quantity * _itemPrice;


discountLevel = getDiscountLevel();


double finalPrice = discountedPrice
  (basePrice, discountLevel);
```

```
int basePrice = _quantity * _itemPrice;

double finalPrice =
  discountedPrice (basePrice);
```

//An object invokes a method, then passes the result as a parameter for a method. The receiver can also invoke this method.

From Website

- https://refactoring.guru/
- https://github.com/nerdschoolbergen/code-smells/tree/master/assignment/src/main/java/nerdschool/bar