Wrap Return Value

The *Wrap Return Value* refactoring allows you to select a method, and either create a wrapper class for its return values, or use an existing, compatible wrapper class. All returns from the method selected will be appropriately wrapped, and all calls to the method will have their returns unwrapped.

Wrapping a method's returns are useful, if your design changes in such a way that you want a method to return more information than originally planned. After wrapping, the wrapper class can be extended, allowing more data to be returned from the method. Also, it is common to wrap primitive return values, thus allowing interface and implementation to be decoupled as needed.

This refactoring is also available from UML Class diagram.

Example

Before	After

`class Order {`
`  String customer;`
`  String getCustomer() {`
`  return`
`  customer;`
`  }`
`}`
	`class Order {`
	`  String customer;`
	`  Customer getCustomer() {`
	`  return new Customer(customer);`
	`  }`
	`}`
	
	`class Customer {`
	`  String id;`
	`  Customer(String id) {`
	`  this.id=id;`
	`  }`
	`}`

To wrap a return value

1. Open the desired class in the editor and place the caret at the method whose returns you wish to wrap.
2. Choose **Refactor | Wrap Return Value** on the main menu, or on the context menu of the selection. Alternatively, select the desired method in the Structure views, and trigger refactoring from there.
3. In the Wrap Return Value dialog box specify the name and package for the new wrapper class, or select an existing compatible wrapper class.
4. Preview and apply changes.

See Also

Procedures:

- Refactoring Source Code

Reference:

- Wrap Return Value Dialog

Web Resources:

- Developer Community