CT2106
Object Oriented Programming

Dr. Frank Glavin
Room 404, IT Building
Frank.Glavin@University*of*Galway.ie
School of Computer Science

University *of* Galway.ie

Yesterday's lecture

- Create a test class to test your code
- Line by line create the stub code and methods
- Until you have the outline of your programme compiling
- Even getting to this stage will force you to make many of the key decisions for your solution
  - Object properties and methods
  - Object collaboration

Ollscoil na Gaillimhe
University of Galway

Revision (1)

- **Class**
  - A **blueprint** or **template** or **set of instructions** to build a specific type of **object**.
  - Every object is built from a class.
  - Each class should be designed and programmed to realise a **single** responsibility
  -
- **Method**
  - A method is the equivalent of a function.
  - Methods are the actions that perform operations on a variable (Fields)

Ollscoil na Gaillimhe
University of Galway

Revision (2)

Encapsulation

- Binding ‘object’ state (fields) and behaviour (methods) together.
- Creating a class means you are doing encapsulation.
- The core idea is to:
  - Hide the implementation details from users
  - No method outside the class can access it directly.
- How?
  - **Private**
  - Protected

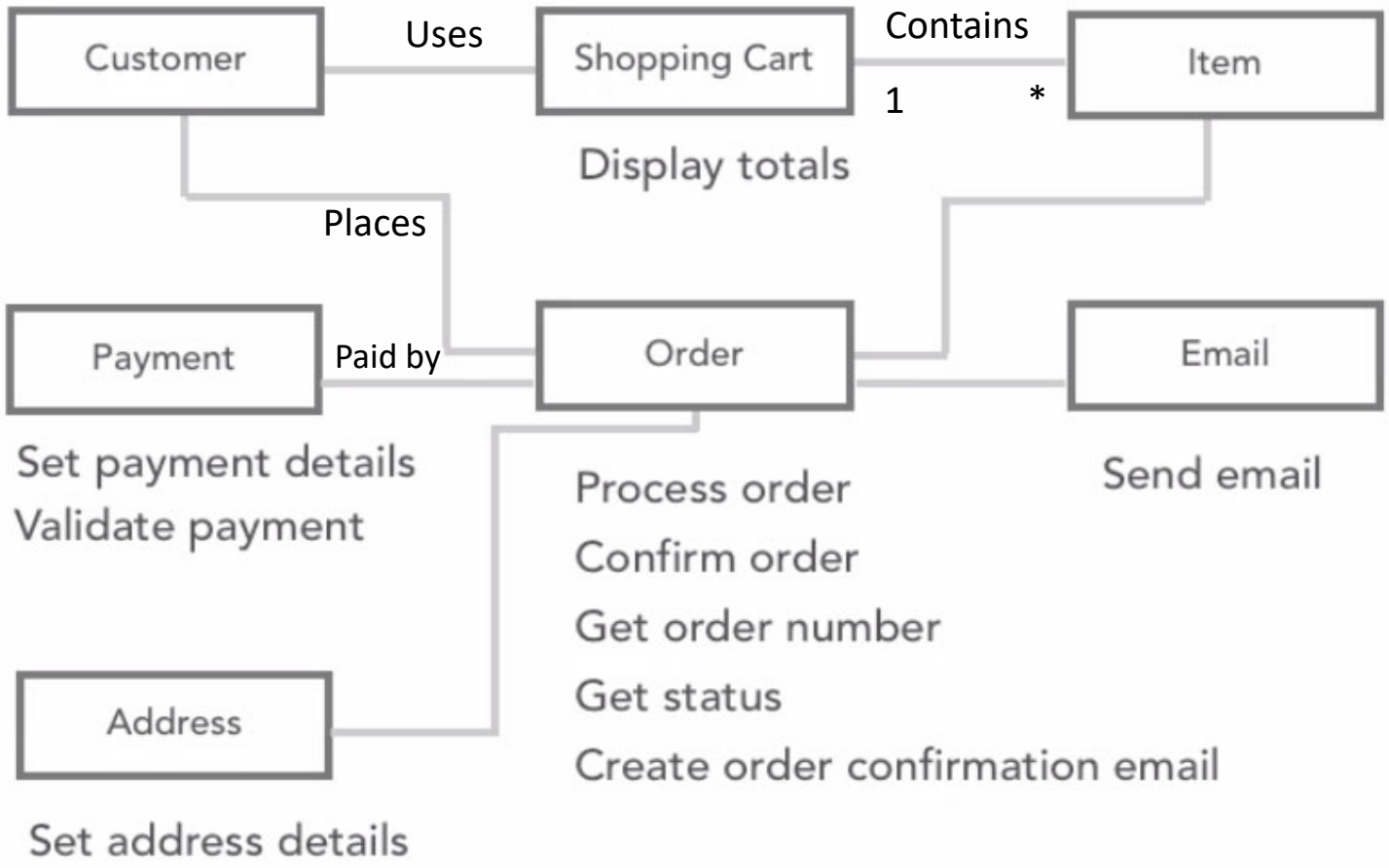
Ollscoil na Gaillimhe
University of Galway

Program Description

A Java program for handling a customer online transaction

The customer verifies the items in their shopping cart. Customer provides payment and address to process the sale. The System validates the payment and responds by confirming the order, and provides the order number that the customer can use to check on the order status. The System will send the customer a copy of the order details by email

Ollscoil na Gaillimhe
University of Galway


Customer
Uses
Shopping Cart
Contains
1
*
Item
Display totals
Places
Payment
Paid by
Order
Email
Set payment details
Validate payment
Process order
Confirm order
Get order number
Get status
Create order confirmation email
Send email
Address
Set address details

Test Code Scenario v1

1. Create Customer object
2. Create Shopping Cart object for the Customer
3. Add 3 items with known cost to cart
4. Finalise the cart and create an order
5. Add a delivery address for the order
6. Add a payment type
7. Validate the payment
8. If successful, email the customer with a success email and the cost of the purchased items

Our code passes the test scenario if an email is created with a message giving the correct total;

Ollscoil na Gaillimhe
University of Galway

We created a test class

```
public class TransactionTest
{

    /**
     * main method to execute the TransactionTest methods
     */
    public static void main(String[] args)
    {
        TransactionTest test = new TransactionTest();
        test.transaction1(); // calls the method with our test scenario
    }

    public void transaction1(){

    // write your test code here

    }
}
```

Ollscoil na Gaillimhe
University of Galway

```
public void transaction1(){
    //the body of our first code scenario will go in here
    //This will be the code that tests if our order transaction classes work
}
```

Goal: turn the steps below into code (within the transaction1 method)

1. Create Customer object
2. Create Shopping Cart object for the Customer
3. Add 3 items with known cost to cart
4. Finalise the cart and create an order
5. Add a delivery address for the order
6. Add a payment type
7. Validate the payment
8. If successful, email the customer with a success email and the cost of the purchased items

Our code passes the test scenario if an email is created with a message giving the correct total;

Method: Proceed in steps

1. Add a line of code
2. Do the minimum required to get it to compile
3. Do 1 and 2 until finished the full scenario

Ollscoil na Gaillimhe
University of Galway

Test Code Scenario v1

1. Create Customer object
2. Create Shopping Cart object for the Customer
3. Add 3 items with known cost to cart
4. Finalise the cart and create an order
5. Add a delivery address for the order
6. Add a payment type
7. Validate the payment
8. If successful, email the customer with a success email and the cost of the purchased items

Ollscoil na Gaillimhe
University of Galway

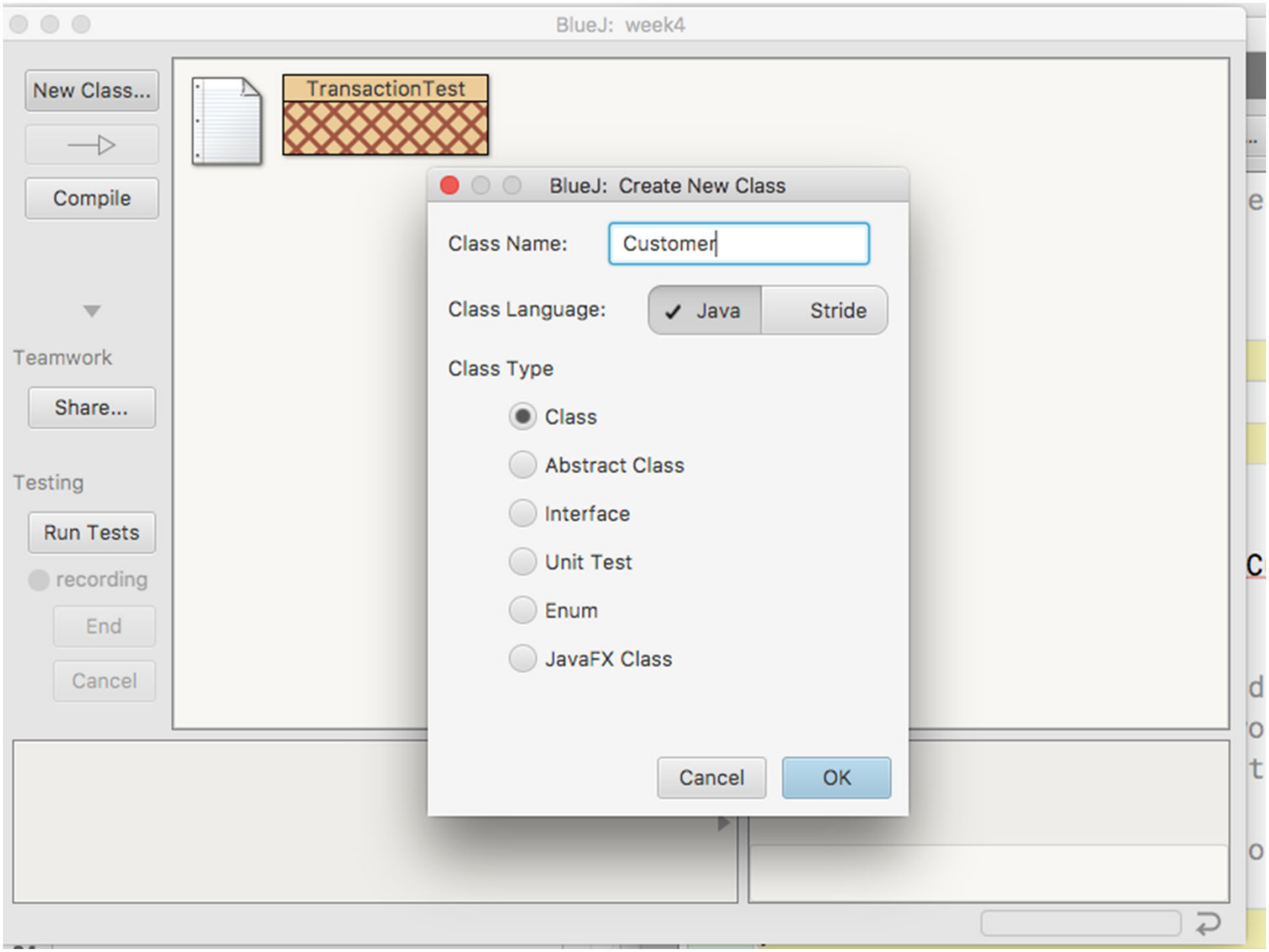
Create a Customer object

Just write a line of code to create a Customer object

```
public void transaction1(){


    Customer customer = new Customer();
                                cannot find symbol -   class Customer

    //When you write this code BlueJ will complain that it can't find a customer class
    // Therefore your code won't compile
    // Use this as the prompt to create a simple Customer class
    // Compile the code
    // Now consider, what properties should a customer object have

}
```

Your program won't compile because there is no Customer class - **yet**

Ollscoil na Gaillimhe
University of Galway


BlueJ: week4
New Class...
Compile
Teamwork
Share...
Testing
Run Tests
recording
End
Cancel
TransactionTest
BlueJ: Create New Class
Class Name:
Customer
Class Language:
Java
Stride
Class Type
Class
Abstract Class
Interface
Unit Test
Enum
JavaFX Class
Cancel
OK

```
public class Customer
{
    // instance variables or 'fields' go here
    // What fields should a customer object have?
    // It depends really on what the role is of the customer object

    /**
     * Constructor for objects of class Customer
     */
    public Customer()
    {
        // initialise the instance variables - but what are they?

    }

}
```

Customer

What are the properties and responsibilities of the Customer object in this programme?

The Customer object holds the data about the Customer data
Any object can request information about the Customer from it

Ollscoil na Gaillimhe
University of Galway

```java
public class Customer {
    private String firstName;
    private String surName;
    private String emailAddress;
    private final long customerId;

    public Customer(String firstName, String surName, String emailAddress){
        this.firstName = firstName;
        this.surName = surName;
        this.emailAddress = emailAddress;
        customerId  = makeCustomerId();
    }

    public long getId() {
        return customerId;
    }

    public String getFirstName() {
        return firstName;
    }

    public String getSurName() {

        return surName;
    }
```

Update your code in the TransactionTest class

```
public void transaction1(){

//1. Create New Customer
Customer customer = new Customer("Niamh", "O'Leary", "niamhol@zmail.com");

// 2. Create a Shopping Cart for the Customer
```

Ollscoil na Gaillimhe
University of Galway

Test Code Scenario v1

1. ~~Create Customer object~~
2. **Create Shopping Cart object for the Customer**
3. Add 3 items with known cost to cart
4. Finalise the cart and create an order
5. Add a delivery address for the order
6. Add a payment type
7. Validate the payment
8. If successful, email the customer with a success email and the cost of the purchased items

Ollscoil na Gaillimhe
University of Galway

ShoppingCart Class

- Now add the code for the Shopping Cart

```
public void transaction1(){

//1. Create New Customer
Customer customer = new Customer("Niamh", "O'Leary", "niamhol@zmail.com");

// 2. Create a Shopping Cart for the Customer
ShoppingCart cart = new ShoppingCart(customer);
```

- Your code won't compile, because you haven't yet created a Shopping cart class
- **This is your cue to create the ShoppingCart class**

Ollscoil na Gaillimhe
University of Galway

Shopping Cart fields?

- *What fields might a Shopping Cart have? Briefly explain the reason for each field.*

  - **cartId**: a unique numerical Id for the Cart
  - **time**: the date/time it was created
  - **items**: to hold the items in the cart
  - **total** : to hold the total for the items in the cart

Ollscoil na Gaillimhe
University of Galway

Shopping Cart behaviours?

- *Methods belonging to a shopping cart?*

- Here are some potential ones:
  - add Item
  - remove item
  - print out the the Items in it
  - display total
  - **lock it** so that items cannot be added/removed from it
  - clear the cart.

Ollscoil na Gaillimhe
University of Galway

Customer / Cart Relationship?

a) *Does a Customer have a Cart?*
b) *Does a Cart have a Customer ?*

Ollscoil na Gaillimhe
University of Galway

Class exercise: Create a Shopping Cart class

Fields:
cartId: numerical
time: String
items: holds a collection
total: numerical
customer: ref type Customer

The Item class is in the next slide – you can download if from Blackboard

Methods:
addItem
removeItem
getTotal
getCartId
getCustomer
printItems
close
clear

Ollscoil na Gaillimhe
University of Galway

Item Class

```
public class Item {
        private String name;
        private int price;
        private long itemId;

        public Item(String itemName, long id) {
                name = itemName;
                itemId = id;
        }

        public void setPrice(int price){
                this.price = price;
        }

        public int getPrice() {
                return price;
        }

        @Override
        public String toString(){

                String out = "Item Id: " + itemId + "\t" + name +"\tPrice: " + price;

                return out;
        }
}
```

Ollscoil na Gaillimhe
University of Galway

addItem

After you have defined the fields start with defining the *addItem* method
See the tutorial on Collections for help with this
adding an object (in this case, an Item) to a collection

Ollscoil na Gaillimhe
University of Galway

Assignment 2

- Based on the code we've written so far

- Remember:
  - Code in increments
  - Always set your code a measureable objective
    - Such as the test scenario mentioned earlier
  - Create a version 0.1 with basic functionality – this will teach you a lot about the problem

Ollscoil na Gaillimhe
University of Galway

Lecture Wrap-up (1)

- Much of OOP is about making modeling decisions
- A model is a simplified representation of reality
- Core modeling decisions: what are the objects, what data do they contain, what are their responsibilities, what are their associations with each other

Ollscoil na Gaillimhe
University of Galway

Lecture Wrap-up (2)

- Start by identifying the objects and relationships in the problem domain – these are candidate objects for your code solution
- It is important to set your code an objective or test before writing the code
- Create the stub code for your classes/methods
- Compile and develop step by step

Ollscoil na Gaillimhe
University of Galway