Saturday, 27
February 16

Crux

Lecture - 12

Object Oriented
Programming - 2

Nidhi Agarwal

CODING
BLOCKS

Object Oriented Programming

Encapsulation

1. Bind the data and functions together
2. Hiding the implementation details
3. Lets us change the implementation without breaking code of our users

Inheritance

1. Extending Functionality of an existing class
2. Add new methods and fields to derived class
3. If both classes have a function with same name, which class's function will get called?

Polymorphism

1. Overriding the base class functions(Virtual Functions)
2. Ability of a variable to take different forms
3. Ability of a function to behave differently on basis of different parameters
4. Ability of a function to work with parameters of subtypes

Public and Non Public Classes?

Final Class?

Final Function?

Abstract functions (Pure Virtual)

Abstract Classes

Data Member Modifiers

1. Public?
2. Protected?
3. Private?
4. Nothing(Friendly)
5. Final
6. Static

Function Modifiers

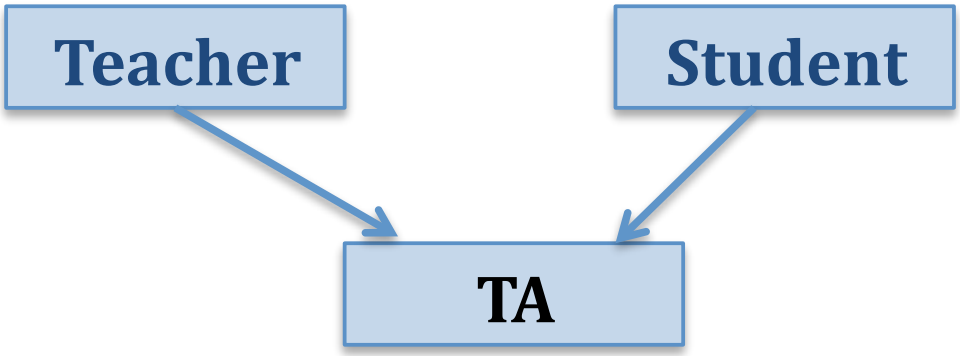
1. Public?
2. Protected?
3. Private?
4. Nothing(Friendly)
5. Abstract
6. Final
7. Static

Classes Modifiers

1. Public?
2. Nothing(Friendly)
3. Abstract
4. Final

Multiple Inheritance

Multiple Inheritance



Java Interfaces

Java interfaces

1. All methods are public and abstract
2. A non-abstract implementing class must implement all methods
3. All data members are final and static
4. A class can implement multiple interfaces
5. An interface can extend another interface

Generics

Generics

1. Allows us to create one method which works for many type of objects
2. Why not just use Object class for all parameters? Run time errors?

Lets look at an example of Generic class

Generics

1. Instantiating a Generic class
2. Multiple Type Parameters
3. Multilayer Generic Parameters
4. Raw Types

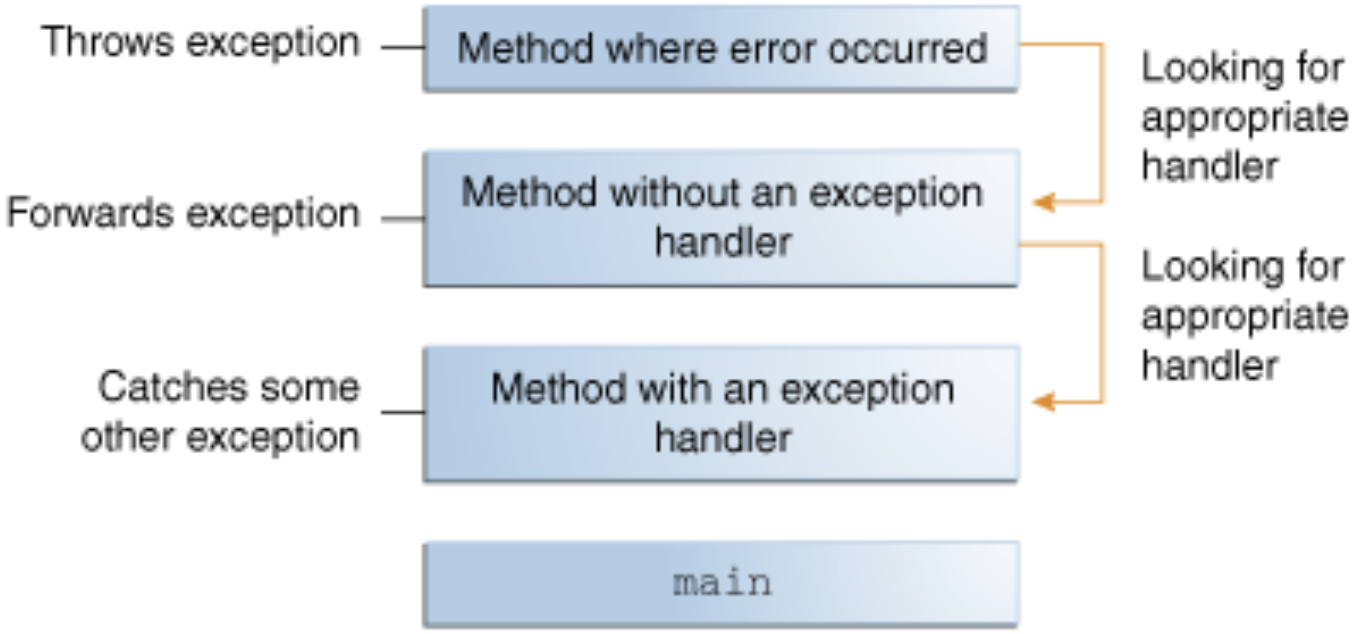
Generic Methods

How to bound the allowed types?

Java Comparable Interface

Exceptions

Exceptions & the call stack



CODING BLOCKS

Type of Exceptions

1. Checked Exceptions (java.lang.Exception)
2. Errors(java.lang.Error)
3. Runtime Exceptions (java.lang.RuntimeException)

How to throw Exceptions?

How to create our own Exception Class?

Either Catch or Specify

Try catch and finally?

Throwable?

BT : Card Game

- A casino offers a card game using a normal deck of 52 cards. The rule is that you turn over two cards each time. For each pair, if both are black, they go to the dealer's pile; if both are red, they go to your pile; if one black and one red, they are discarded. The process is repeated until you two go through all 52 cards. If you have more cards in your pile, you win $100; otherwise (including ties) you get nothing. The casino allows you to negotiate the price you want to pay for the game. How much would you be willing to pay to play this game?

Thank You!

Nidhi Agarwal

nidhi@codingblocks.com