

Sunday, 17
January 2016

Crux

Lecture -10

Object Oriented
Programming-2

Sumeet Malik

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

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

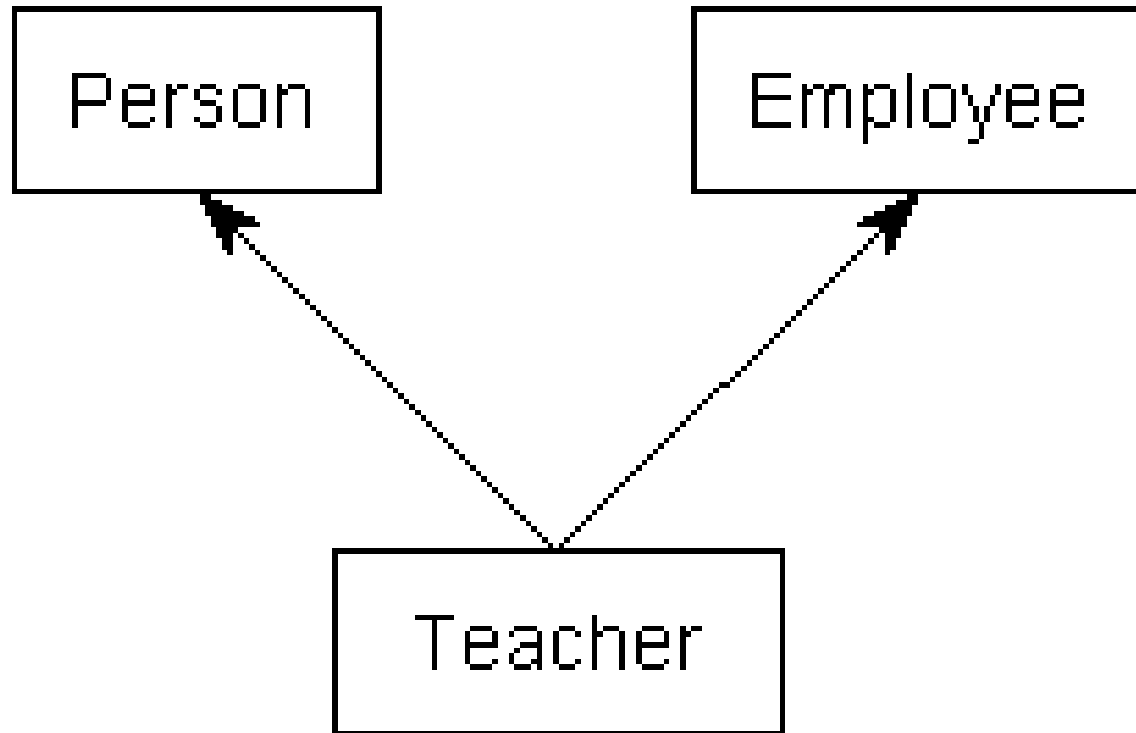
1. Public?
2. Protected?
3. Private?
4. Nothing(Friendly)
5. Abstract
6. Final
7. Static

Classes Modifiers

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

Multiple Inheritance

Multiple Inheritance



Multiple Inheritance

```
class Teacher: public Person, public Employee
{
private:
    int m_nTeachesGrade;

public:
    Teacher(std::string strName, std::string strEmployer,
double dWage, int nTeachesGrade)
        : Person(strName), Employee(strEmployer,
dWage), m_nTeachesGrade(nTeachesGrade)
    {
    }
};
```

Java Interfaces

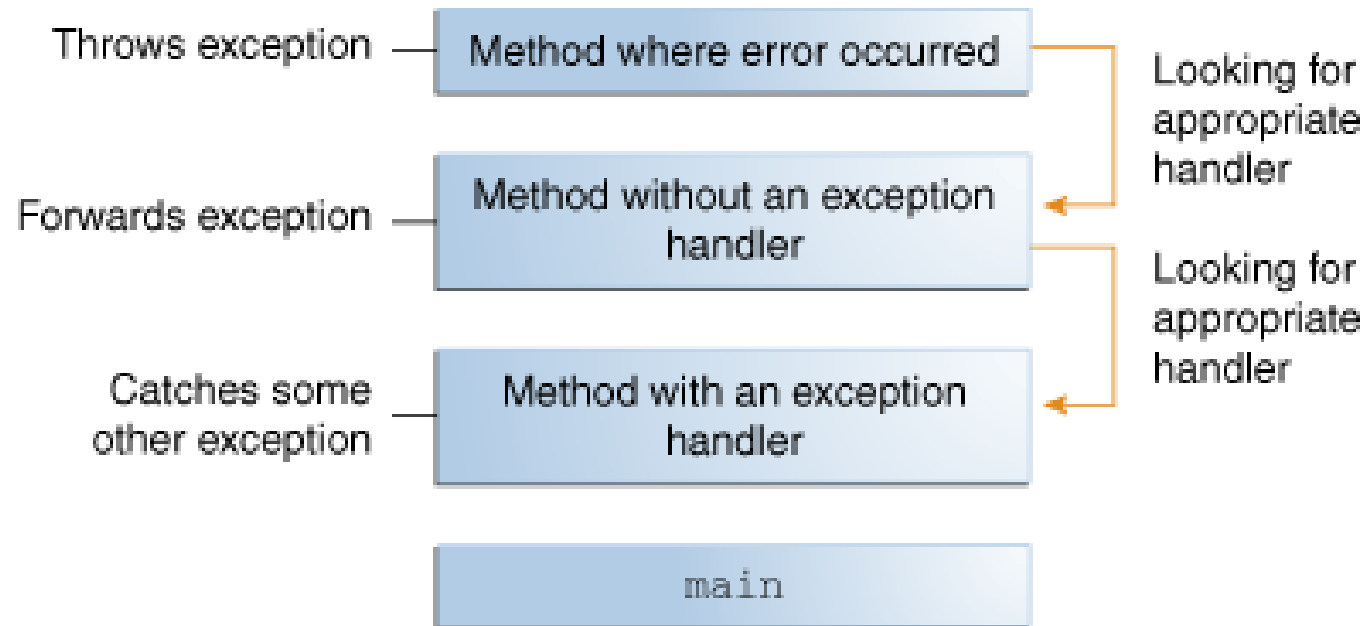
Java interfaces

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

Java Comparable Interface

Exceptions

Exceptions & the call stack



Type of Exceptions

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

Either Catch or Specify

Try catch and finally?

How to throw Exceptions?

Throwable?

How to create our own Exception Class?

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?



Thank You!

Sumeet Malik

9999258467

Sumeet.malik1188@gmail.com