Java Exception Handling

Presented by Sagar Java Consultant

Exception

Def : Abnormal Situation at run time
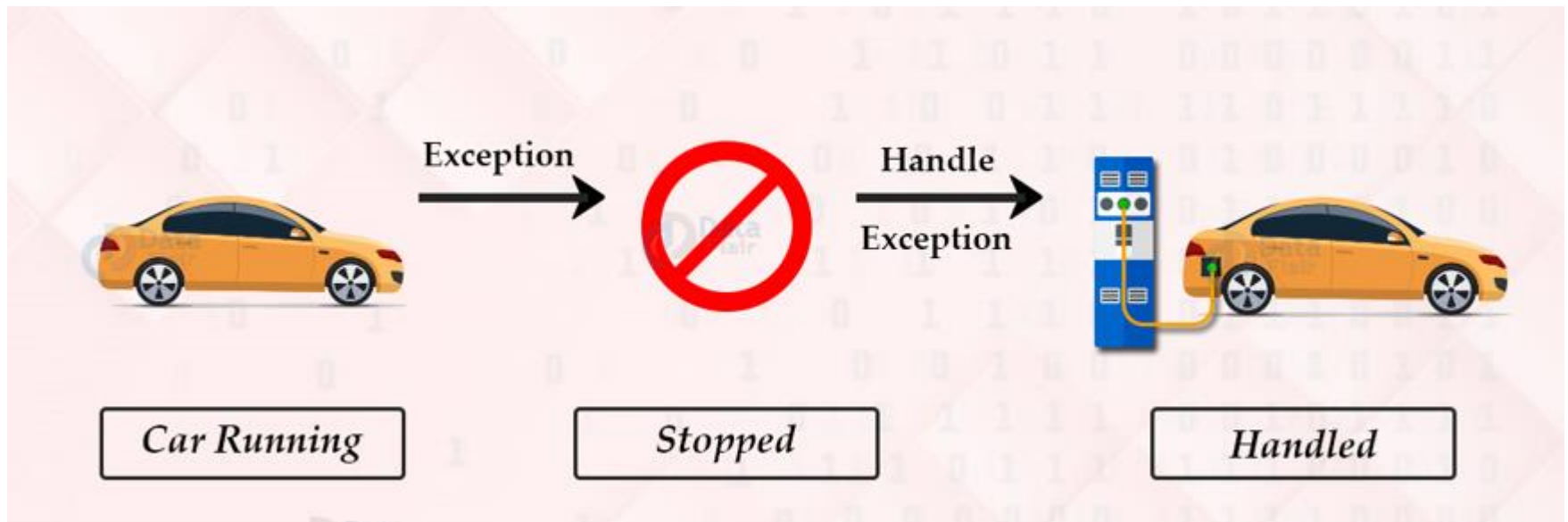
A Real World Scenario

Fig: Realtime Example of Exception Handling
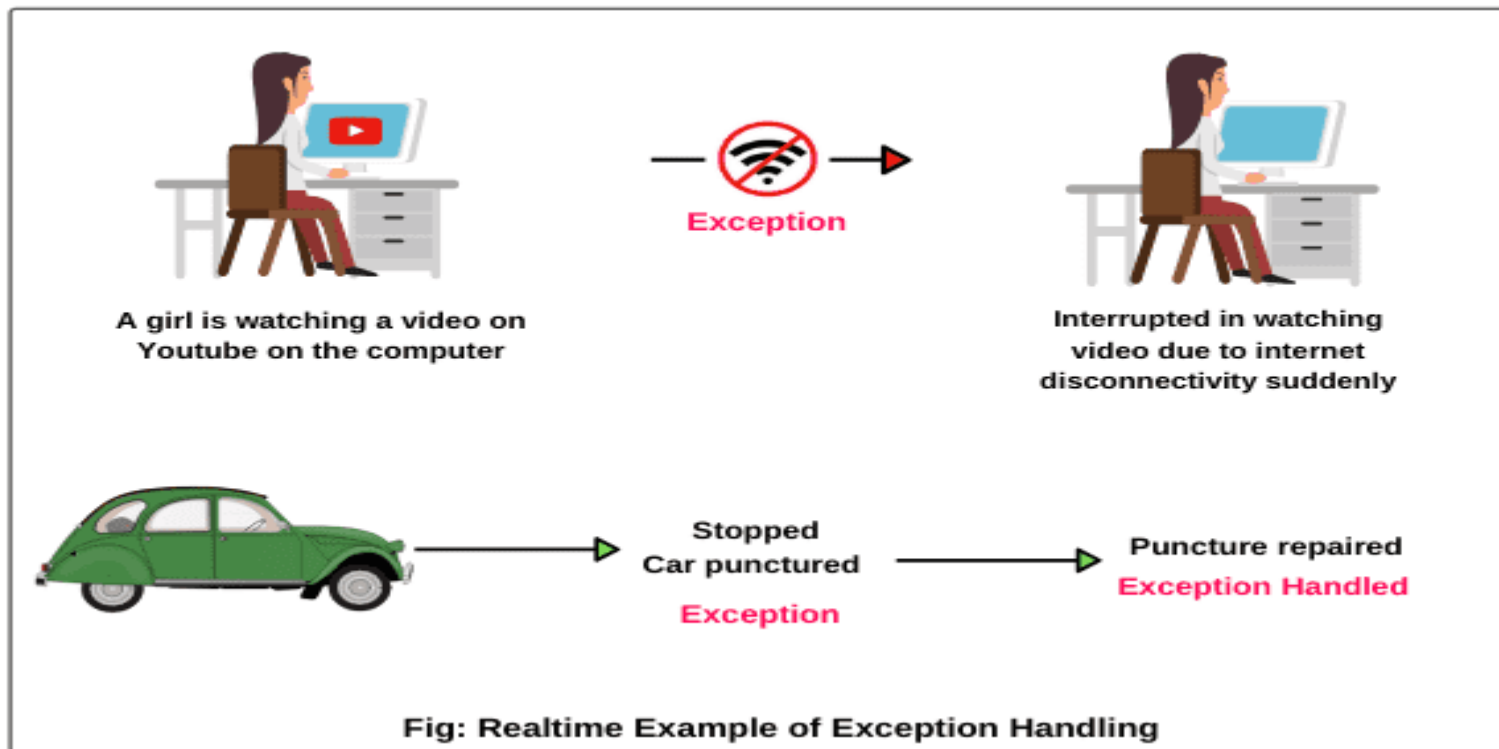
Exception

Def : Abnormal Situation at run time

A Real World Scenario

Exception Definition

- An exception is an event that occurs during the execution of a program that disrupts the normal flow of instructions

- The ability of a program to intercept run-time errors, take corrective measures and continue execution is referred to as exception handling
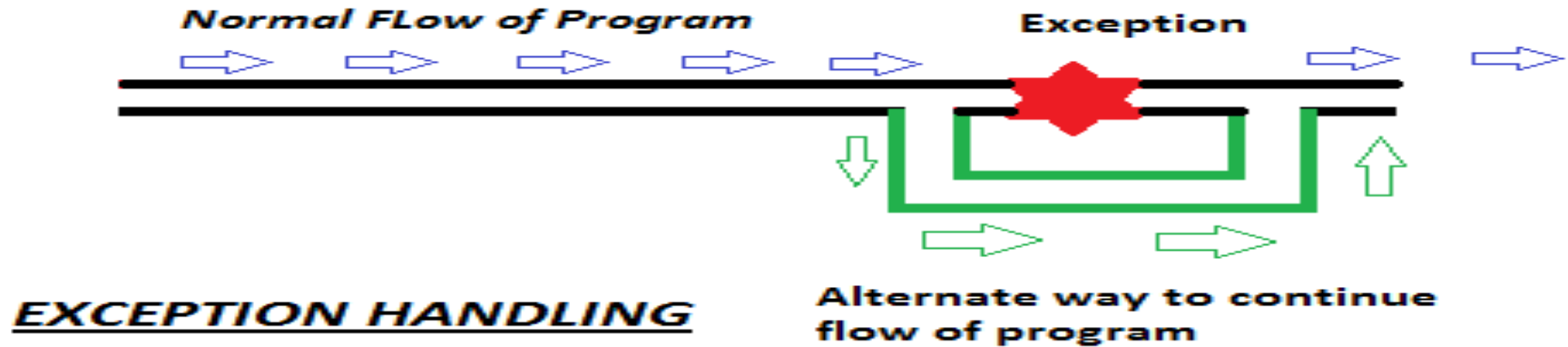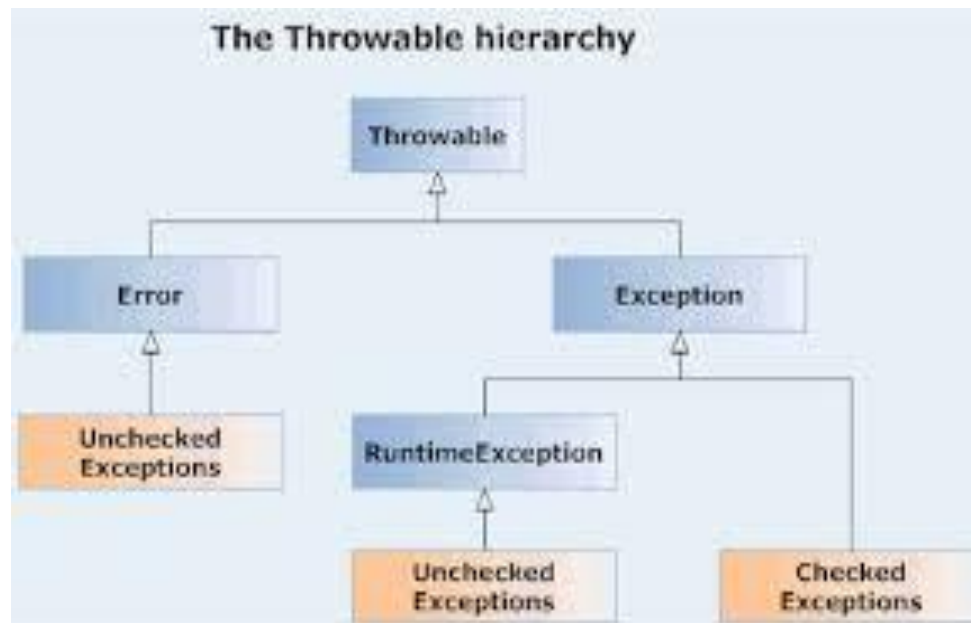
Exceptions – Situations

- There are various situations when an exception could occur:

  - Attempting to access a file that does not exist

  - Inserting an element into an array at a position that is not in its bounds

  - Performing some mathematical operation that is not permitted

  - Declaring an array using negative values

Exception

Normal FLow of Program

Exception

EXCEPTION HANDLING

Alternate way to continue flow of program

Exceptions - Hierarchy

The Throwable hierarchy

Exception – Example

```java
public class  Demo {
      public static void main(String args[]) {
            int dividend = 90;
            int divisor = 0;
            int quotient = dividend/divisor;
            System.out.println("Quotient   = "
                  + quotient);
      }
  }
```

java.lang.ArithmeticException: / by zero
at Demo.main(Demo.java:4)

Exception Handling Keywords

Java's exception handling is managed using the following keywords: **try, catch, throw, throws and finally.**

```
try {
 // code comes here
 }
catch(TypeofException  obj) {
    //handle the exception
 }
     finally  {
        //code to be executed before the program ends
 }
```

Exception Handling Keywords …

- Any part of the code that can generate an exception should be put in the try block

- Any exception should be handled in the catch block defined by the catch clause

- This block is also called the catch block, or the exception handler

- The corrective action to handle the exception should be put in the catch block

Exception Handling - Example

```java
public class ExceptionDemo{
     public static void main(String args[]){
          int divisor, dividend, quotient;
          try{
                    divisor = 0;
                    quotient = dividend / divisor;
                    System.out.println("Message");
          }
          catch (ArithmeticException e){
                    System.out.println("Division by zero.");
          }
          System.out.println("After catch statement.");
     }
}
```

Checked and Unchecked Exceptions

- A checked exception is an exception that is found at compile time.
- Ex: FileNotFoundException

- A unchecked exception is found at run time.
- Ex: ArithmeticException, NumberFormatException
- ArrayIndexOutOfBoundsException

printStackTrace()

- We can use the printStackTrace() method to print the program's execution stack

- This method is used for debugging

printStackTrace() example

```java
public class PrintStackExample {
  public static void main(String args[])
  {
     try {
          m1();
     }
     catch(IOException e) {
          e.printStackTrace();
     }
  }
```

contd..

printStackTrace() Example ...

```java
static void m1() throws IOException {
    m2();
}
static void m2() throws IOException {
    m3();
}
static void m3() throws IOException{
    throw new IOException();
}
}
```