# CHANDIGARH UNIVERSITY

## UNIVERSITY INSTITUTE OF ENGINEERING

## DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

| **LAB MANUAL** | |
|---|---|
| **Subject Name** | Project Based Learning in Java Lab [PBL LAB] |
| **Subject Code** | **20CSP 321/20ITP 323** |
| **Branch** | Computer Science and Engineering |
| **Semester** | 5th |
| **Faculty Name** | Richa Dhiman |
| **Designation** | Assistant Professor |

## CONTENTS

# University-Vision and Mission

## Vision of the University

To be globally recognized as a Centre of Excellence for Research, Innovation, Entrepreneurship and disseminating knowledge by providing inspirational learning to produce professional leaders for serving the society

## Mission of the University

Providing world class infrastructure, renowned academicians and ideal environment for Research, Innovation, Consultancy and Entrepreneurship relevant to the society.

Offering programs & courses in consonance with National policies for nation building and meeting global challenges.

Designing Curriculum to match International standards, needs of Industry, civil society and for inculcation of traits of Creative Thinking and Critical Analysis as well as Human and Ethical values.

Ensuring students delight by meeting their aspirations through blended learning, corporate mentoring, professional grooming, flexible curriculum and healthy atmosphere based on co-curricular and extra-curricular activities.

Creating a scientific, transparent and objective examination/evaluation system to ensure an ideal certification.

Establishing strategic relationships with leading National and International corporates and universities for academic as well as research collaborations.

Contributing for creation of healthy, vibrant and sustainable society by involving in Institutional Social Responsibility (ISR) activities like rural development, welfare of senior citizens, women empowerment, community service, health and hygiene awareness and environmental protection

# Department-Vision and Mission

## Vision of the Department

To be recognized as a leading Computer Science and Engineering department through effective teaching practices and excellence in research and innovation for creating competent professionals with ethics, values and entrepreneurial attitude to deliver service to society and to meet the current industry standards at the global level.

## Mission of the Department

M1: To provide practical knowledge using state-of-the-art technological support for the experiential learning of our students.

M2: To provide industry recommended curriculum and transparent assessment for quality learning experiences.

M3: To create global linkages for interdisciplinary collaborative learning and research.

M4: To nurture advanced learning platform for research and innovation for students' profound future growth.

M5: To inculcate leadership qualities and strong ethical values through value based education.

# PEO Program Educational Objectives (PEOs)

**Program Educational Objectives (PEOs)**

**PEO 1:** To produce computer science graduate engineers with an ability to comprehend, understand and analyze real life problems for providing sustainable solutions teams in the light of disruptive technologies.
**PEO 2:** To inculcate life-long learning skills in graduates preparing them for work in changing environments and multidisciplinary teams in order to enhance their capability being globally employable.
**PEO 3:** To instill leadership qualities in graduates with a sense of confidence, professionalism and ethical attitude to produce professional leaders for serving the society.
**PEO 4:** To make the graduates adaptable to changing career opportunities who have the potential to excel in industry/ public sector/ higher studies or entrepreneurship exhibiting global competitiveness.

## PO Program Outcomes (PO's)

**Program Outcomes(PO's)**

Engineering Graduates will be able to:
**PO 1.** Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
**PO 2.** Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
**PO 3.** Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
**PO 4.** Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
**PO 5.** Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
**PO 6**. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
**PO 7**. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
**PO 8**. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
**PO 9.** Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
**PO 10.** Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
**PO 11.** Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**PO 12.** Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

## Student Outcomes

The Bachelor of Engineering is a program offered by the Department of Computer Science & Engineering in accordance with the Student Outcome of Computing Accreditation Commission (CAC) and Engineering Accreditation Commission (EAC) of ABET. The Student Outcomes are as follows:

### Student Outcomes according to Computing Accreditation Commission (CAC)

**SO 1.** Analyze a complex computing problem and apply principles of computing and other relevant disciplines to identify solutions.
**SO 2**. Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline.
**SO 3.** Communicate effectively in a variety of professional contexts.
**SO 4.** Recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles.
**SO 5.** Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline.
**SO 6.** Apply computer science theory and software development fundamentals to produce computing-based solutions.

### Student Outcomes according to Engineering Accreditation Commission (EAC)

**SO 1.** An ability to identify, formulates, and solve complex engineering problems by applying principles of engineering, science, and mathematics
**SO 2.** An ability to apply engineering design to produce solutions that meet specified needs with consideration of public health, safety, and welfare, as well as a global, cultural, social, environmental, and economic factor
**SO 3.** An ability to communicate effectively with a range of audiences
**SO 4.** An ability to recognize ethical and professional responsibilities in engineering situations and make informed judgments, which must consider the impact of engineering solutions in global, economic, environmental, and societal contexts
**SO 5.** An ability to function effectively on a team whose members together provide leadership, create a collaborative and inclusive environment, establish goals, plan tasks, and meet objectives.
**SO 6.** An ability to develop and conduct appropriate experimentation, analyze and interpret data, and use engineering judgment to draw conclusions.
**SO 7.** An ability to acquire and apply new knowledge as needed, using appropriate learning

## Program Specific Outcomes (PSOs)

A Graduate of Computer Science and Engineering Program will be able:

**PSO 1.** To acquire proficiency in developing and implementing efficient solutions using emerging technologies, platforms and Free and Open-Source Software (FOSS).
**PSO 2.** To gain critical understanding of hardware and software tools catering to the contemporary needs of IT industry.

# CODE OF ETHICS

I. To uphold the highest standards of integrity, responsible behavior, and ethical conduct in professional activities.

1. To hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, to protect the privacy of others, and to disclose promptly factors that might endanger the public or the environment.
2. To improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies, including intelligent systems.
3. To avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist.
4. To avoid unlawful conduct in professional activities, and to reject bribery in all its forms.
5. To seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others.
6. To maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations.

II. To treat all persons fairly and with respect, to not engage in harassment or discrimination, and to avoid injuring others.

7. To treat all persons fairly and with respect, and to not engage in discrimination based on characteristics such as race, religion, gender, disability, age, national origin, sexual orientation, gender identity, or gender expression.
8. To not engage in harassment of any kind, including sexual harassment or bullying behavior.
9. To avoid injuring others, their property, reputation, or employment by false or malicious actions, rumors or any other verbal or physical abuses.

III. To strive to ensure this code is upheld by colleagues and co-workers.

10. To support colleagues and co-workers in following this code of ethics, to strive to ensure the code is upheld, and to not retaliate against individuals reporting a violation.

## Syllabus

| | | | | | |
|---|---|---|---|---|---|
| **SubjectCode** **20CSP 321/ 20ITP 323** | **Project Based Learning in Java Lab** | **L** | **T** | **P** | **C** |
| | Total Contact Hours : 60 Hours | **0** | **0** | **4** | **2** |
| | Common to all Specializations of CSE 3rd Year | | | | |
| | Prerequisite: Studied Programming Language | | | | |

**Course Objectives**

1. To generate analytical and conceptual ability related to fundamentals of Java.
2. To understand the concepts of Web application development.
3. To understand the concepts of Fundamentals of I/O , Database Connectivity
4. To Implement of the OOPS concepts using Eclipse Environment
5. To implement the concepts of Collections and able to access database through

**Course Outcomes**

1.Use an integrated development environment to write, compile, run, and test simple object-oriented Java programs.

2.Read and make elementary modifications to Java programs that solve real-world problems.

3.Designs will demonstrate the use of good object-oriented design principles including encapsulation and information hiding.

4.The implementation will demonstrate the use of a variety of basic control structures including selection and repetition; classes and objects in a tiered architecture (user interface, controller, and application logic layers); primitive and reference data types including composition; basic AWT components; file-based I/O; and one-dimensional arrays.

5.Test plans will include test cases demonstrating testing strategies.

## List of Experiments

**UNIT-I**

1. Create a application to save the employee information using arrays.
2. Design and implement a simple inventory control system for a small video rental store.
3. Create an application to calculate interest for FDs, RDs based on certain conditions using inheritance.

**UNIT-II**

4. Create a program to set view of Keys from Java Hashtable.
5. Create a program to show the usage of Sets of Collection interface.
6. Write a Program to perform the basic operations like insert, delete, display and search in list. List contains String object items where these operations are to be performed.
7. Create a menu based Java application with the following options.1. Add an Employee2.Display All3.Exit If option 1 is selected, the application should gather details of the employee like employee name, employee id, designation and salary and store it in a file. If option 2 is selected, the application should display all the employee details. If option 3 is selected the application should exit.

**UNIT-III**

8. Create a palindrome creator application for making a longest possible palindrome out of given input string.
9. Create a Servlet/ application with a facility to print any message on web browser.
10. Create JSP application for addition, multiplication and division.

| COs | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 | PSO1 | PSO2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **CO1** | 3 | – | – | – | 2 | – | – | – | – | – | – | – | – | – |
| **CO2** | 3 | – | – | – | 2 | – | – | – | – | – | – | – | – | – |
| **CO3** | 3 | – | – | – | 2 | – | – | – | – | – | – | – | – | – |
| **CO4** | 3 | 2 | 2 | – | 2 | – | – | – | – | – | – | – | 2 | 2 |
| **CO5** | 3 | 2 | – | – | 2 | – | – | – | – | – | – | – | – | – |

**CO-SO Mapping**

| CO | CAC - SO1 | CAC - SO2 | CAC - SO3 | CAC - SO4 | CAC - SO5 | CAC - SO6 | EAC - SO1 | EAC - SO2 | EAC - SO3 | EAC - SO4 | EAC - SO5 | EAC - SO6 | EAC - SO7 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO1 | | | ✓ | ✓ | | ✓ | | | | | ✓ | | ✓ |
| CO2 | | | ✓ | ✓ | | ✓ | | | ✓ | | ✓ | | ✓ |
| CO3 | | ✓ | | ✓ | | ✓ | ✓ | | ✓ | | ✓ | | ✓ |
| CO4 | | ✓ | | ✓ | ✓ | ✓ | ✓ | | ✓ | | ✓ | ✓ | ✓ |
| CO5 | | ✓ | ✓ | ✓ | | ✓ | | ✓ | | | ✓ | ✓ | ✓ |

## List of Experiments (Mapped with Cos)

| Exp No. | Experiment Name | Mapped CO |
|---|---|---|
| 1 | Create an application to save the employee information using arrays. | CO1,CO3,CO4 |
| 2 | Design and implement a simple inventory control system for a small video rental store. | CO1,CO3,CO4,CO 5 |
| 3 | Create an application to calculate interest for FDs, RDs based on certain conditions using inheritance. | CO1,CO3,CO4 |
| 4 | Create a program to set view of Keys from Java Hashtable. | CO1,CO4 |
| **5** | Create a program to show the usage of Sets of Collection interface. | CO1,CO2 |
| **6** | Write a Program to perform the basic operations like insert, delete, display and search in list. List contains String object items where these operations are to be performed. | CO1,CO2 |
| **7** | Create a menu based Java application with the following options.1.Add an Employee2.Display All3.Exit If option 1 is selected, the application should gather details of the employee like employee name, employee id, designation and salary and store it in a file. If option 2 is selected, the application should display all the employee details. If option 3 is selected the application should exit. | CO1,CO2,CO4 |
| **8** | Create a palindrome creator application for making a longest possible palindrome out of given input string. | CO4,CO 5 |
| **9** | Create a Servlet/ application with a facility to print any message on web browser. | CO2,CO 5 |
| **10** | Create JSP application for addition, multiplication and division. | CO2,CO 5 |

## Experiment 1:

**Aim:** Create a application to save the employee information using arrays

## Objective: -

• To learn about arrays in java.
• To learn about usage of loops and switch statements.

## Input/Apparatus Used:

Hardware Requirements: - Minimum 384MB RAM, 100 GB hard Disk, processor with 2.1 MHz
Software Requirements: - Eclipse, NetBeans, IntelliJ, etc.

## Reading Material:

**Array in JAVA**
Java provides a data structure, the array, which stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

Instead of declaring individual variables, such as number0, number1, ..., and number99, you declare one array variable such as numbers and use numbers[0], numbers[1], and ..., numbers[99] to represent individual variables.

**Declaring Array Variables**

To use an array in a program, you must declare a variable to reference the array, and you must specify the type of array the variable can reference. Here is the syntax for declaring an array variable −

**Syntax**

dataType[] arrayRefVar; // preferred way.

or

dataType arrayRefVar[]; // works but not preferred way.

**Example:**

The following code snippets are examples of this syntax −

double[] myList; // preferred way. or

double myList[]; // works but not preferred way.

**Creating Arrays**

You can create an array by using the new operator with the following syntax −

**Syntax:**

arrayRefVar = new dataType[arraySize];

The above statement does two things −

- It creates an array using new dataType[arraySize].
- It assigns the reference of the newly created array to the variable arrayRefVar.

Declaring an array variable, creating an array, and assigning the reference of the array to the variable can be combined in one statement, as shown below −

dataType[] arrayRefVar = new dataType[arraySize];

Alternatively you can create arrays as follows −

dataType[] arrayRefVar = {value0, value1, ..., valuek};

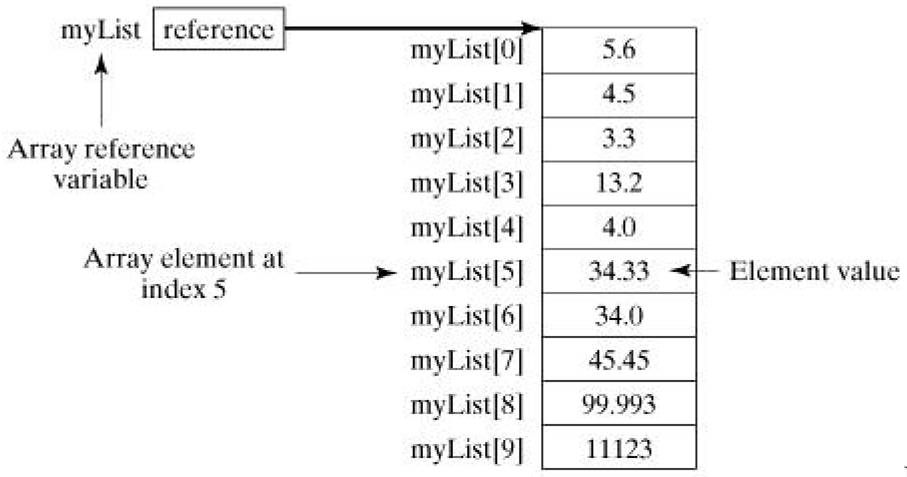
The array elements are accessed through the index. Array indices are 0-based; that is, they start from 0 to arrayRefVar.length-1.

**Example:**

Following statement declares an array variable, myList, creates an array of 10 elements of double type and assigns its reference to myList −

double[] myList = new double[10];

Following picture represents array myList. Here, myList holds ten double values and the indices are from 0 to 9.



**Processing Arrays**

When processing array elements, we often use either for loop or foreach loop because all of the elements in an array are of the same type and the size of the array is known.

## Procedure/Algorithm/Pseudocode

1 Create main java class to take input from user by class emp

2 Fetch values from both the tables for salary calculation. Check its designation code from Ist table and then fetch values i.e designation and da from 2nd table by applying switch case.

```
 case 'e':

 designation="Engineer";

 da=20000;

 break;

case 'c':

 designation="Consultant";

 da=32000;

 break;

case 'k':

 designation="Clerk";

 da=12000;

 break;

case 'r':

 designation="Receptonist";
```

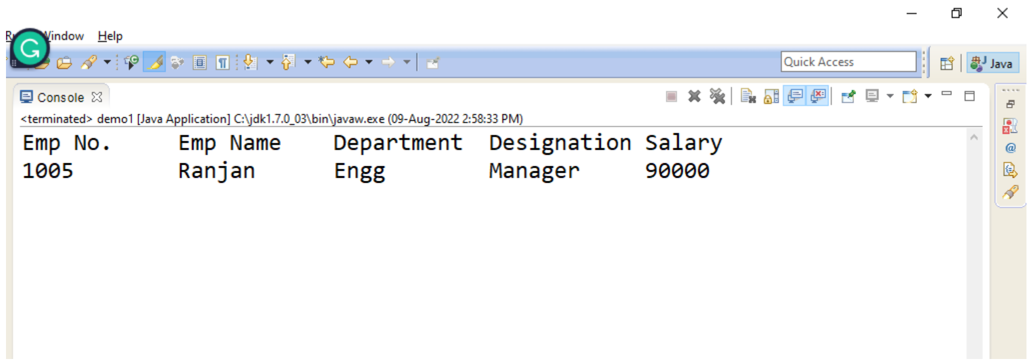
```
  da=15000;
  break;
case 'm':
  designation="Manager";
  da=40000;
  break;
default:
  designation="Invalid";
  da=0;
  3. Calculate  Salary of employee  using this equation :
  salary=basic[index]+ hra[index]+da-it[index];
  4. Print salary
  5. Exit
```

## Sample Code:

```
package basic;

public class Project1 {

	public static void main(String[] args) {

		String empid[]={"1001","1002","1003","1004","1005","1006","1007"};

		String
empName[]={"Ashish","Sushma","Rahul","Chahat","Ranjan","Suman","Tanmay"};

		String
joinDate[]={"1/04/2009","23/08/2012","12/11/2008","29/01/2013","16/07/2005","1/01/2000","12/
/06/2006"};

		String desigCode[] = {"e","c","k","r","m","e","c"};

		String dept[]={"R&D","PM","Acct","Front Desk","Engg","Manufacturing","PM"};

		int basic[]={20000,30000,10000,12000,50000,23000,29000};

		int hra[]={8000,12000,8000,6000,20000,9000,12000};
```

```
                    int it[]={3000,9000,1000,2000,20000,4400,10000};
                    String designation = new String();
                    int da = 0;
                    int salary = 0;
                    int index = -1;
for(int i=0; i<empid.length; i++){
          if(empid[i].equals(args[0])){
                    index = i;
                    break;
          }
}
if(index == -1){
          System.out.println("There is no employee with empid : "+ args[0]);
}else{
          switch (desigCode[index]) {
          case "e":
                    designation = "Engineer";
                    da = 20000;
                    break;
          case "c":
                    designation = "Consultant";
                    da = 32000;
                    break;
          case "k":
                    designation = "Clerk";
}

                da = 12000;

                break;

            case "r":

                designation ="Receptionist";

                da = 15000;

                break;

            case "m":

                designation = "Manager";

                da = 40000;

            }

            //Basic+HRA+DA-IT

            salary = basic[index] + hra[index] + da -it[index];

            System.out.printf("%-12s","Emp No.");

            System.out.printf("%-12s","Emp Name");

            System.out.printf("%-12s","Department");

            System.out.printf("%-12s","Designation");

    System.out.printf("%-12s","Salary");

    System.out.println();

            System.out.printf("%-12s", empid[index]);

            System.out.printf("%-12s", empName[index]);

            System.out.printf("%-12s",dept[index]);

            System.out.printf("%-12s%-12d", designation, salary);

}

            }

}
```

## Output:



```
<terminated> demo1 [Java Application] C:\jdk1.7.0_03\bin\javaw.exe (09-Aug-2022 2:58:33 PM)
Emp No.     Emp Name    Department  Designation Salary
1005        Ranjan      Engg        Manager     90000
```

## Viva Voce Questions:

1. What is array and types of arrays in java?

2. What do you mean by jagged array?

3. What is static keyword in java?

4. What is the use of final keyword?

5. Explain Switch Statement.

## Experiment 2:

**Aim:** Design and implement a simple inventory control system for a small video rental store.

### Objective: -

• To learn about Classes.
• To learn about Encapsulation, Aggregation concept in java.

### Input/Apparatus Used:

Hardware Requirements: - Minimum 384MB RAM, 100 GB hard Disk, processor with 2.1 MHz
Software Requirements: - Eclipse, NetBeans, IntelliJ, etc.

### Reading Material:

**Java - What is OOP?**
OOP stands for Object-Oriented Programming**.**

Procedural programming is about writing procedures or methods that perform operations on the data, while object-oriented programming is about creating objects that contain both data and methods.

Object-oriented programming has several advantages over procedural programming:

- OOP is faster and easier to execute
- OOP provides a clear structure for the programs
- OOP helps to keep the Java code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
- OOP makes it possible to create full reusable applications with less code and shorter development time

**Java - What are Classes and Objects?**

Classes and objects are the two main aspects of object-oriented programming.

So, a class is a template for objects, and an object is an instance of a class.

When the individual objects are created, they inherit all the variables and methods from the class.

**Java Inheritance**
Java Inheritance (Subclass and Superclass)

In Java, it is possible to inherit attributes and methods from one class to another. We group the "inheritance concept" into two categories:

- subclass (child) - the class that inherits from another class
- superclass (parent) - the class being inherited from

To inherit from a class, use the extends keyword.

In the example below, the Car class (subclass) inherits the attributes and methods from the Vehicle class (superclass):

**Example**

```
class Vehicle {

protected String brand = "Ford"; // Vehicle attribute

public void honk() { // Vehicle method

System.out.println("Tuut, tuut!");

}

}

class Car extends Vehicle {

private String modelName = "Mustang"; // Car attribute

public static void main(String[] args) {

// Create a myCar object

Car myCar = new Car();

// Call the honk() method (from the Vehicle class) on the myCar object

myCar.honk();

// Display the value of the brand attribute (from the Vehicle class) and the value of the modelName
from the Car class

System.out.println(myCar.brand + " " + myCar.modelName);
```

}

Why And When To Use "Inheritance"?

- It is useful for code reusability: reuse attributes and methods of an existing class when you create a new class.

## Procedure/Algorithm/Pseudocode

1. Create main java class to take input from user by three classes:
   - **I. Video.**
   - **II. VideoStore.**
   - **III. VideoLauncher.**
2. Create a class video to store its details like title, checked unchecked status, ratings.
3. Create a videoStore class extending video class and define all the methods in this class
   - To addVideo
   - To store and change checked out status
   - To add rating to particular video
   - To check the list of available videos or inventory

4. Create main class VideoLauncer where you ask for your choice using switch case. Choice can be

- 1.Add Videos:
- 2.Check Out Videos:
- 3.Return Videos:
- 4.Receive Rating:
- 5.List Inventory:
- 6.Exit

## Sample Code:

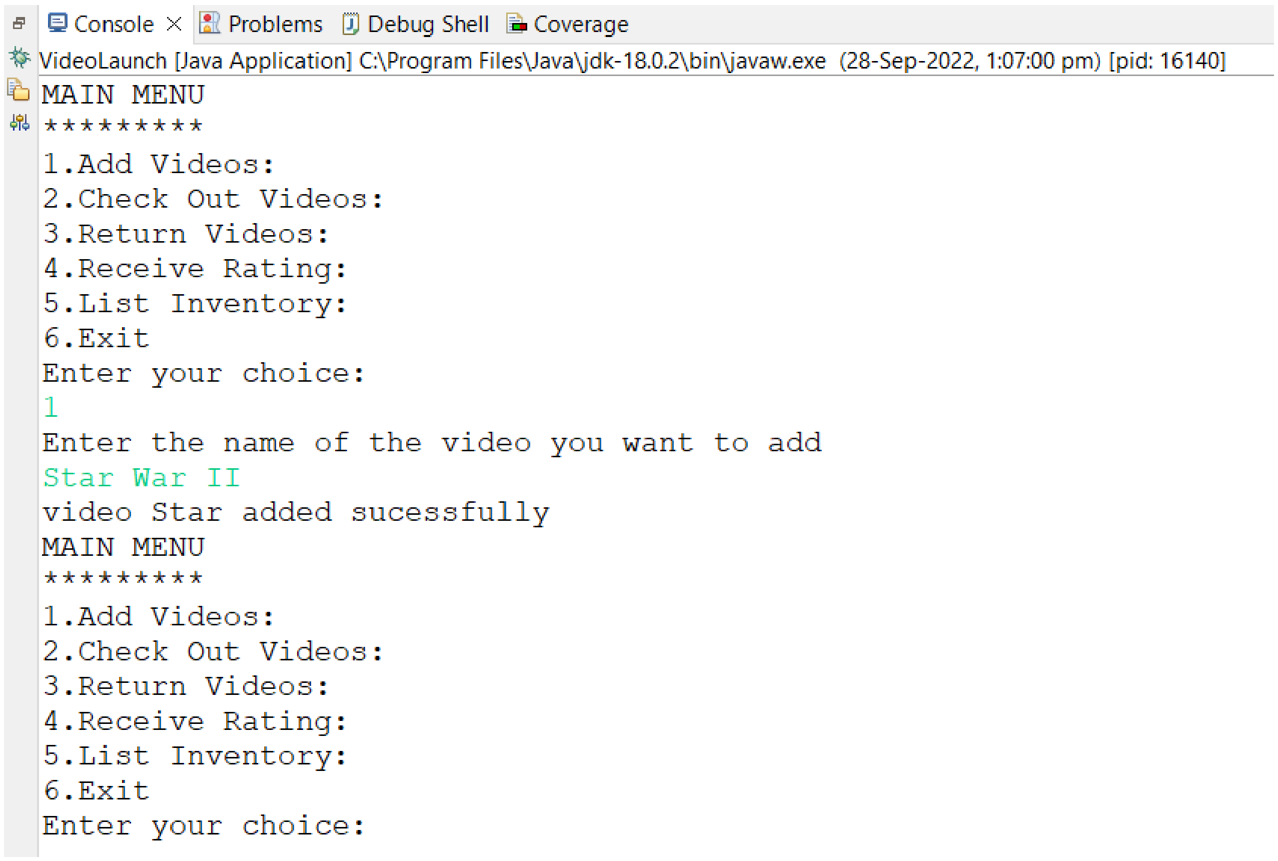
```
package com.VideoInventory;

package JavaExps;
import java.util.Scanner;

public class VideoLaunch {
public static void main(String[] args) {

        VideoStore obj = new VideoStore();
        int choice;
        String videoName;
```

```
int rating;
boolean status = true;
while(status)
{
System.out.println("MAIN MENU");
System.out.println("*********");
System.out.println("1.Add Videos:");
System.out.println("2.Check Out Videos:");
System.out.println("3.Return Videos:");
System.out.println("4.Receive Rating:");
System.out.println("5.List Inventory:");
System.out.println("6.Exit");
System.out.println("Enter your choice:");
Scanner sc = new Scanner(System.in);
choice = sc.nextInt();
switch(choice)
{
case 1:
   {
      System.out.println("Enter the name of the video you want to add");
      videoName = sc.next();
      obj.addVideo(videoName);
      break;
   }

case 2:
{

   System.out.println("Enter the name of video to checkout");
   videoName = sc.next();
   obj.doCheckOut(videoName);
   break;
}
case 3:
{
   System.out.println("Enter the video name to return");
   videoName = sc.next();
   obj.doReturn(videoName);
   break;
}

case 4:
{
   System.out.println("Enter the name of video you want to rate");
    videoName = sc.next();
   System.out.println("Enter the Ratings for this video");
   rating = sc.nextInt();
```

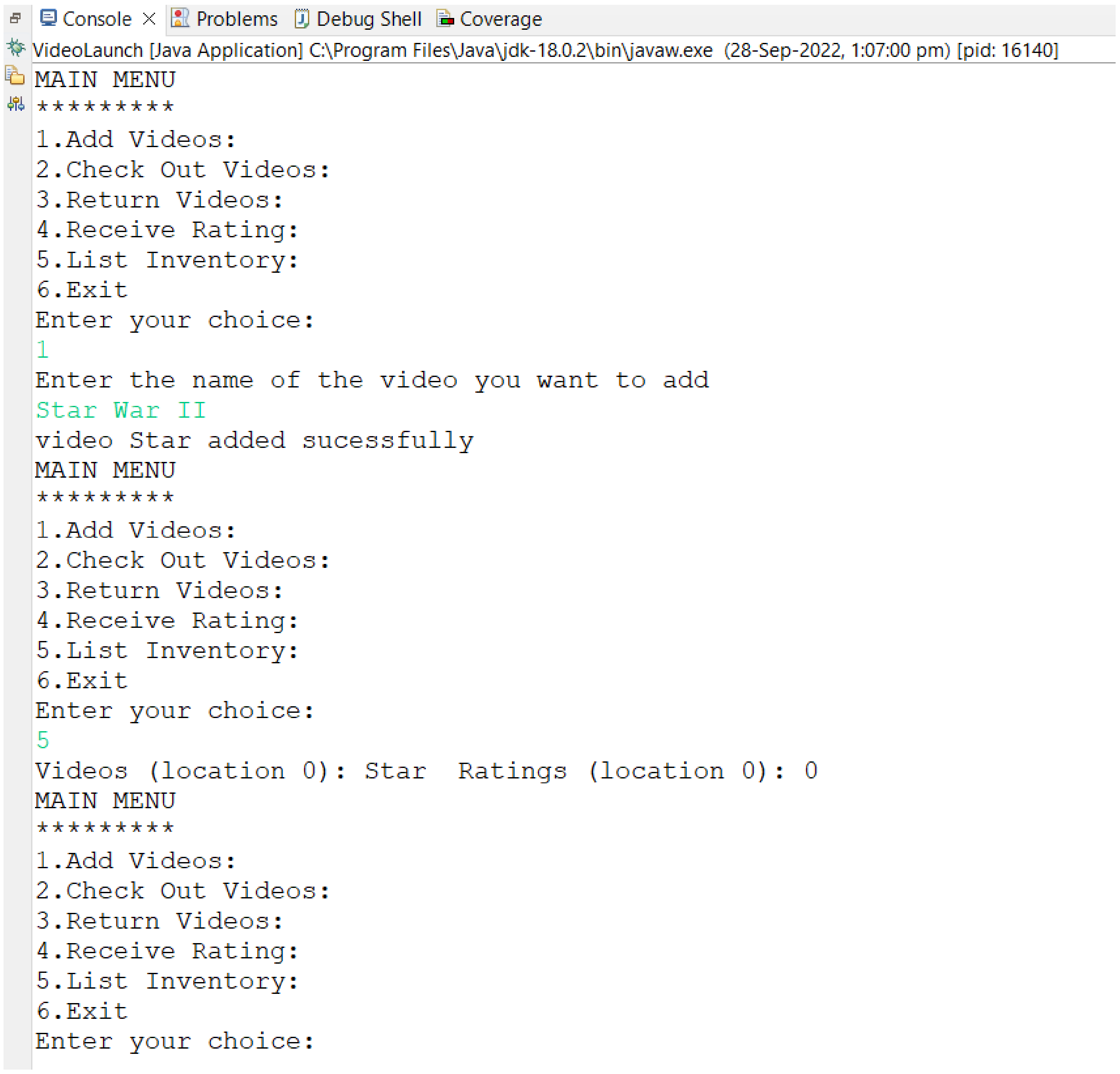
```
            obj.receiveRating(videoName, rating);
          break;
        }
        case 5:
        {
            obj.listInventory();
            break;
        }
        case 6:
        {
            System.out.println("Exiting...!! Thanks for using the application");
            sc.close();
            obj.exit();

            break;
        }
        default:
        {
            System.out.println("Wrong input!!");
        }

        }
    }
  }
}
class Video
{
    String videoName;
    boolean checkOut;
    int rating;

    String getName()
    {
        return videoName;

    }

    void doCheckOut()
    {
            this.checkOut = true;

    }

    void doReturn()
    {
            this.checkOut = false;
    }
    void receiveRating(int rating)
    {
        this.rating = rating;
    }



    int getRating()
    {
        return rating;
    }

    boolean getCheckOut()
    {
        return checkOut;
    }

    public Video(String videoName)
    {
        this.videoName = videoName;
    }

}
class VideoStore
{
    Video store[]= new Video[20];
    static int a=0;
    void addVideo(String name)
    {
        store[a] = new Video(name);
        store[a].checkOut = false;
        store[a].receiveRating(0);
        System.out.println("video " +name +" added sucessfully");
        a++;
    }

void doCheckOut(String name)
    {

        for(int i=0; i<a;i++)
        {
            if(store[i].getName().equals(name))
            {
            store[i].doCheckOut();
            System.out.println("Video " +name +" removed successfully from "+i +" location");
        }
        else
                {
                System.out.println("No such video exists at:" +i+" location");
                }
       }
  }
  void doReturn(String name)
  {

     for(int i= 0; i<a;i++)
     {
        if(store[i].getName().equals(name))
        {
        store[i].doReturn();
        System.out.println("Video returned: " +name +" from location "+i);
     }
     else{
        System.out.println("No such video exists at locations:" +i);
        }
     }
   }
  void receiveRating(String name, int rating)
  {

        for(int i= 0; i<a;i++)
     {
        if(store[i].getName().equals(name))
        {
           store[i].receiveRating(rating);
        }
     }
        System.out.println("Ratings " +rating +" has been mapped to the video " +name);
  }
  void  listInventory()
  {

     for(int i= 0; i<a;i++)
     {
        if(!store[i].getCheckOut())
        {
           System.out.print("Videos (location "+i+ "): "+store[i].videoName);
           System.out.print("  Ratings (location "+i+ "): "+store[i].getRating()+"\n");
        }
     }
  }
  public void exit() {
                              System.exit(0);
```

```
                    }
    }
```

**Output:**


Console × Problems Debug Shell Coverage

```
VideoLaunch [Java Application] C:\Program Files\Java\jdk-18.0.2\bin\javaw.exe  (28-Sep-2022, 1:07:00 pm) [pid: 16140]
MAIN MENU
*********
1.Add Videos:
2.Check Out Videos:
3.Return Videos:
4.Receive Rating:
5.List Inventory:
6.Exit
Enter your choice:
1
Enter the name of the video you want to add
Star War II
video Star added sucessfully
MAIN MENU
*********
1.Add Videos:
2.Check Out Videos:
3.Return Videos:
4.Receive Rating:
5.List Inventory:
6.Exit
Enter your choice:
```



```
VideoLaunch [Java Application] C:\Program Files\Java\jdk-18.0.2\bin\javaw.exe  (28-Sep-2022, 1:07:00 pm) [pid: 16140]
MAIN MENU
*********
1.Add Videos:
2.Check Out Videos:
3.Return Videos:
4.Receive Rating:
5.List Inventory:
6.Exit
Enter your choice:
1
Enter the name of the video you want to add
Star War II
video Star added sucessfully
MAIN MENU
*********
1.Add Videos:
2.Check Out Videos:
3.Return Videos:
4.Receive Rating:
5.List Inventory:
6.Exit
Enter your choice:
5
Videos (location 0): Star  Ratings (location 0): 0
MAIN MENU
*********
1.Add Videos:
2.Check Out Videos:
3.Return Videos:
4.Receive Rating:
5.List Inventory:
6.Exit
Enter your choice:
```

## Viva Voce Questions:

1. What is class and object?

2. What is inheritance in java?

3. Is multiple inheritance possible in java?

4. What is the scope of protected access modifier?

5. What do mean by Encapsulation?

## Experiment 3:

**Aim:** Create a application to calculate interest for FDs, RDs based on certain conditions using inheritance.

### Objective: -

• To learn about concept of Inheritance.
• To learn about Abstract classes, Exception Handling.

### Input/Apparatus Used:

Hardware Requirements: - Minimum 384MB RAM, 100 GB hard Disk, processor with 2.1 MHz
Software Requirements: - Eclipse, NetBeans, IntelliJ, etc.

### Reading Material:

**Abstract Classes and Methods**
Data abstraction is the process of hiding certain details and showing only essential information to the user.

Abstraction can be achieved with either abstract classes **or** interfaces (which you will learn more about in the next chapter).

The abstract keyword is a non-access modifier, used for classes and methods:

- **Abstract class**: is a restricted class that cannot be used to create objects (to access it, it must be inherited from another class).
- **Abstract method**: can only be used in an abstract class, and it does not have a body. The body is provided by the subclass (inherited from).

An abstract class can have both abstract and regular methods:

```
abstract class Animal {

public abstract void animalSound();

public void sleep() {

System.out.println("Zzz");

}

}
```

From the example above, it is not possible to create an object of the Animal class:

Animal myObj = new Animal(); // will generate an error

To access the abstract class, it must be inherited from another class. Let's convert the Animal class we used in the Polymorphism chapter to an abstract class:

Remember from the Inheritance chapter that we use the extends keyword to inherit from a class.

**Example**

```
// Abstract class

abstract class Animal {

// Abstract method (does not have a body)

public abstract void animalSound();

// Regular method

public void sleep() {

System.out.println("Zzz");

}

}

// Subclass (inherit from Animal)

class Pig extends Animal {

public void animalSound() {

// The body of animalSound() is provided here

System.out.println("The pig says: wee wee");

}

}


class Main {
```

```
public static void main(String[] args) {

Pig myPig = new Pig(); // Create a Pig object

myPig.animalSound();

myPig.sleep();

}

}
```

Why And When To Use Abstract Classes and Methods?

To achieve security - hide certain details and only show the important details of an object.

**Java Exceptions - Try...Catch**
**Java Exceptions**

When executing Java code, different errors can occur: coding errors made by the programmer, errors due to wrong input, or other unforeseeable things.

When an error occurs, Java will normally stop and generate an error message. The technical term for this is: Java will throw an **exception** (throw an error).

Java try and catch

The try statement allows you to define a block of code to be tested for errors while it is being executed.

The catch statement allows you to define a block of code to be executed, if an error occurs in the try block.

The try and catch keywords come in pairs:

**Syntax**

```
try {

// Block of code to try

}

catch(Exception e) {
```

```
// Block of code to handle errors
}
```

**Consider the following example:**

This will generate an error, because myNumbers[10] does not exist.

```
public class Main {
public static void main(String[ ] args) {
int[] myNumbers = {1, 2, 3};
System.out.println(myNumbers[10]); // error!
}
}
```

The output will be something like this:

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 10
    at Main.main(Main.java:4)
```

If an error occurs, we can use try...catch to catch the error and execute some code to handle it:

Example

```
public class Main {
public static void main(String[ ] args) {
try {
int[] myNumbers = {1, 2, 3};
System.out.println(myNumbers[10]);
} catch (Exception e) {
System.out.println("Something went wrong.");
}
}
}
```

The output will be:

```
Something went wrong.
```

Finally

The finally statement lets you execute code, after try...catch, regardless of the result:

Example

```
public class Main {
public static void main(String[] args) {
try {
int[] myNumbers = {1, 2, 3};
System.out.println(myNumbers[10]);
} catch (Exception e) {
System.out.println("Something went wrong.");
} finally {
System.out.println("The 'try catch' is finished.");
}
}
}
```

The output will be:

```
Something went wrong.
The 'try catch' is finished.
```

The throw keyword

The throw statement allows you to create a custom error.

The throw statement is used together with an exception type**.** There are many exception types available in
Java: ArithmeticException, FileNotFoundException, ArrayIndexOutOfBoundsException, Security Exception, etc:

**Example**

Throw an exception if age is below 18 (print "Access denied"). If age is 18 or older, print "Access granted":

```
public class Main {

static void checkAge(int age) {

if (age < 18) {

throw new ArithmeticException("Access denied - You must be at least 18 years old.");

}

else {

System.out.println("Access granted - You are old enough!");

}

}

public static void main(String[] args) {

checkAge(15); // Set age to 15 (which is below 18...)

}

}
```

The output will be:

```
Exception in thread "main" java.lang.ArithmeticException: Access denied - You must be at least
18 years old.

      at Main.checkAge(Main.java:4)

      at Main.main(Main.java:12)
```

If **age** was 20, you would **not** get an exception:

Example

checkAge(20);

The output will be:

Access granted - You are old enough!

## Procedure/Algorithm/Pseudocode

1. Create main java class to take input from user by three classes:
    - **SavingAccount.**
    - **FixedDepositAccount.**
    - **RecurringDeposit.**

2. Implement cases for three choices:
   1. FD 2. RD 3.SavingAccount.

**case 1: FD**

```
I.   Enter the amount:

     If(amount<1 crore):
        Enter the maturity period in days.
        Enter the age.
             If(age>60):
                  Print->Simple Interest evaluated from the array of "General" defined in class FixedDepositAccount.
             Else if(age<60):
                  Print->Simple Interest evaluated from the array of "SeniorCitizen" defined In class FixedDepositAccount.
             Else:
                  Print->Incorrect age.

        Else if(amount>1crore):
             Enter the maturity period in days.
                  Print->Simple Interest evaluated from the array of "interestRate" defined in class FixedDepositAccount.
        Else:
             Print->incorrect Amount.

II.  Exit.
```

**Case 2: RD**

```
I.   Enter the amount.
II.  Enter the maturity period in months.
III. Enter the age:

     If(age>60):
        Print->Simple Interest evaluated from the array of "General" defined in class
                    RecurringDeposit.

     Else if(age<60):
        Print->Simple Interest evaluated from the array of "SeniorCitizen" defined in  class
                    RecurringDeposit
     Else:
        Print->Incorrect age.
IV.  Exit.
```

**Case 3: SavingAccount**

```
I.   Enter the amount.
II.  Enter the account type:
     If(accountType=="NRI"):
        interestRate=6%.
        Print->Simple interest gained.

     Else If(accountType=="normal"):
        interestRate=4%.
        Print->Simple interest gained.

     Else:
        Print->incorrect account type.
III. Exit.
```

3. Raise user defined Exception if input has invalid or negative values.
4. Exit.

## Sample Code:

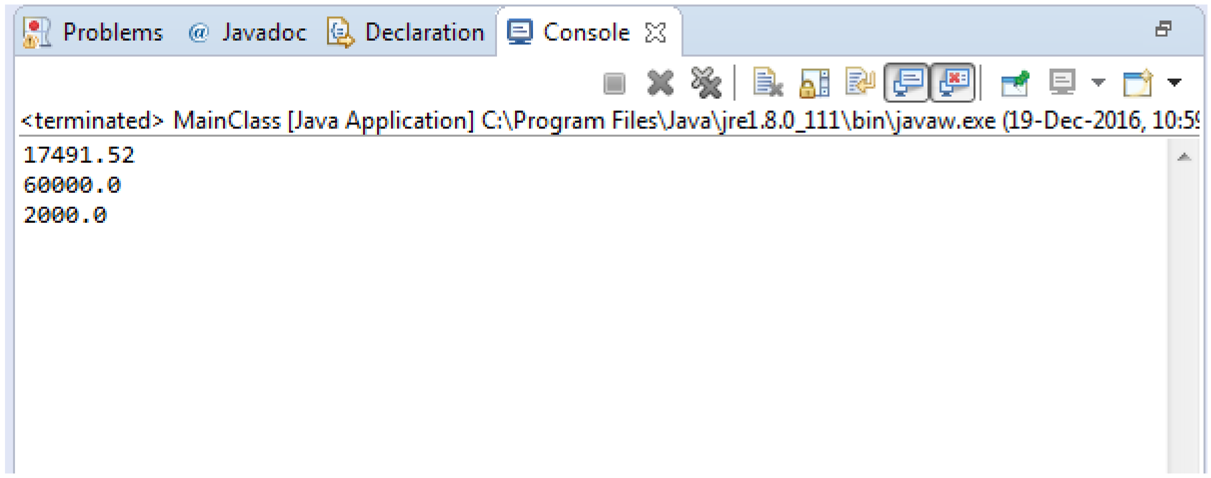
```
package com.wipro.bank.main;

import com.wipro.bank.service.BankService;

public class MainClass {

        public static void main(String[] args) {
                // TODO Auto-generated method stub
```

```
                    int tenure = 5;

        float principal = 1000;

        int age = 20;

        String gender = "male";

        BankService b=new BankService();


        b.calculate(principal, tenure, age, gender);
          }

}
package com.wipro.bank.acc;

public abstract class Account {
        int tenure;
        float principal;
        float rateOfInterest;

        public void setInterest(int age, String gender) {


                if(gender.equalsIgnoreCase("Male"))
                {
                        if(age<60)
                        {
                                rateOfInterest=(float) 9.8;
                        }
                        if(age>=60)
                        {
                                rateOfInterest=10.5f;
                        }
                }
                else
                {
                        if(age<58)
                        {
                                rateOfInterest=10.2f;
                        }
                        if(age>=58)
                        {
                                rateOfInterest=10.8f;
                        }
                }
```

```
	}

	public float calculateMaturityAmount(float totalPrincipleDeposited,
			float maturityInterest) {
		return (totalPrincipleDeposited+maturityInterest);

	}

	public abstract float calculateInterest();
	public abstract float calculateAmountDeposited();
}
package com.wipro.bank.acc;

import java.util.logging.Logger;

public class RDAccount extends Account {

	int tenure;
	float principal;
	public RDAccount(int tenure, float principal)
	{
		this.tenure=tenure;
		this.principal=principal;
	}
	@Override
	public float calculateInterest() {
		// TODO Auto-generated method stub
		float result=0.0f;
		float quat=4;
		int totalMonths=tenure*12;
		float monInYears=0;
		float iRate= rateOfInterest/100;

		for(int i=totalMonths;i>0;i--)
		{

			monInYears=i/12.0f;
			result+=principal*((Math.pow((1+iRate/quat),quat*monInYears))-1);
		}

		return result;

	}

	@Override
	public float calculateAmountDeposited() {
```

```
                // TODO Auto-generated method stub

                return principal*tenure*12;
        }

}
```

## Output:



```
<terminated> MainClass [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (19-Dec-2016, 10:5
17491.52
60000.0
2000.0
```

## Viva Voce Questions:

1. What is Exception Handling and what is the need of it?

2. What are types of Exceptions?

3. What is the difference between Throw, Throws and finally keyword?

4. What is abstract class?

5. Can abstract class have non abstract methods?

# Experiment 4 :

**Aim:** Create a program to set view of Keys from Java Hashtable.

## Objective: -

• To learn about concept of Hashing.
• To learn about HashMap.

## Input/Apparatus Used:

Hardware Requirements: - Minimum 384MB RAM, 100 GB hard Disk, processor with 2.1 MHz
Software Requirements: - Eclipse, NetBeans, IntelliJ, etc.

## Reading Material:

Hashtable was part of the original java.util and is a concrete implementation of a Dictionary.

However, Java 2 re-engineered Hashtable so that it also implements the Map interface. Thus, Hashtable is now integrated into the collections framework. It is similar to HashMap, but is synchronized.

Like HashMap, Hashtable stores key/value pairs in a hash table. When using a Hashtable, you specify an object that is used as a key, and the value that you want linked to that key. The key is then hashed, and the resulting hash code is used as the index at which the value is stored within the table.

Following is the list of constructors provided by the HashTable class.

**Constructor & Description**

- 1 Hashtable( )This is the default constructor of the hash table it instantiates the Hashtable class.
- Hashtable(int size)This constructor accepts an integer parameter and creates a hash table that has an initial size specified by integer value size.
- Hashtable(int size, float fillRatio)This creates a hash table that has an initial size specified by size and a fill ratio specified by fillRatio. This ratio must be between 0.0 and 1.0, and it determines how full the hash table can be before it is resized upward.
- Hashtable(Map < ? extends K, ? extends V > t)This constructs a Hashtable with the given mappings.
- Apart from the methods defined by Map interface, Hashtable defines the following methods −
- Sr.No Method & Description
- 1 void clear( )Resets and empties the hash table.
- 2 Object clone( )Returns a duplicate of the invoking object.
- 3 boolean contains(Object value)Returns true if some value equal to the value exists within the hash table. Returns false if the value isn't found.

- 4 boolean containsKey(Object key)Returns true if some key equal to the key exists within the hash table. Returns false if the key isn't found.
- boolean containsValue(Object value)Returns true if some value equal to the value exists within the hash table. Returns false if the value isn't found.
- Enumeration elements( )Returns an enumeration of the values contained in the hash table.
- Object get(Object key)Returns the object that contains the value associated with the key. If the key is not in the hash table, a null object is returned.
- boolean isEmpty( )Returns true if the hash table is empty; returns false if it contains at least one key.
- Enumeration keys( )Returns an enumeration of the keys contained in the hash table.
- Object put(Object key, Object value)Inserts a key and a value into the hash table. Returns null if the key isn't already in the hash table; returns the previous value associated with the key if the key is already in the hash table.
- void rehash( )Increases the size of the hash table and rehashes all of its keys.
- Object remove(Object key)Removes the key and its value. Returns the value associated with the key. If the key is not in the hash table, a null object is returned.
- int size( )Returns the number of entries in the hash table.
- String toString( )Returns the string equivalent of a hash table.

## Procedure/Algorithm/Pseudocode

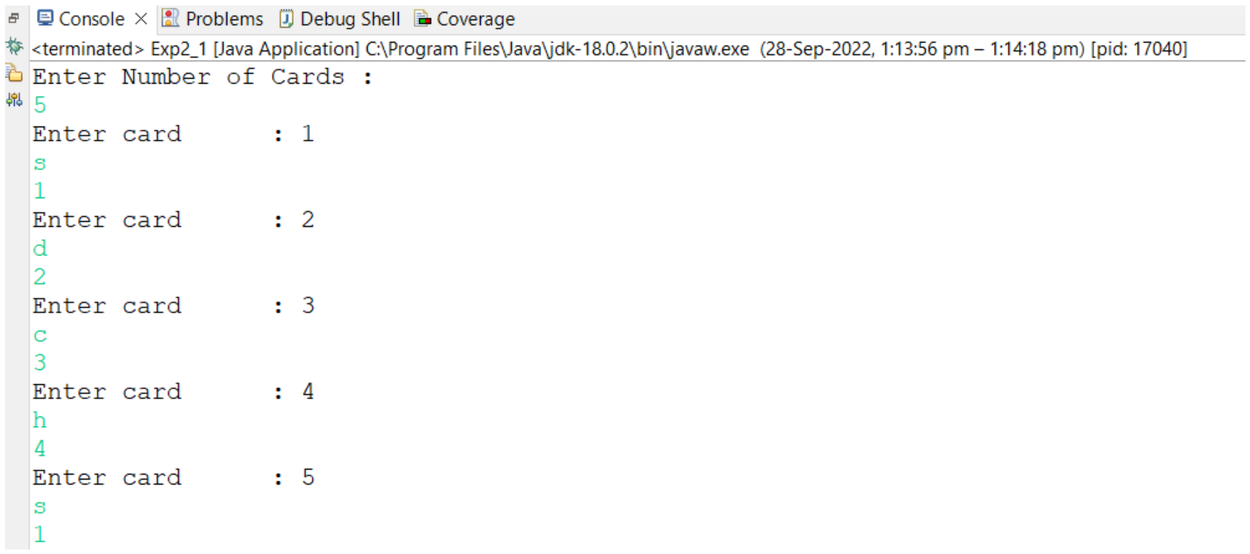
1. Create a card class having symbol and number.
2. Add details of cards in hashmap.
3. Check if card numbers are already present in hashmap then ignore.
4. If it is not present then replace previous details and insert modified details.
5. Then after iteration print number of cards in each symbol and sum of number in each symbol.

## Sample Code:

```
package Unit2;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Iterator;

import java.util.Map.Entry;

import java.util.Scanner;

import java.util.Set;

import java.util.TreeMap;
```

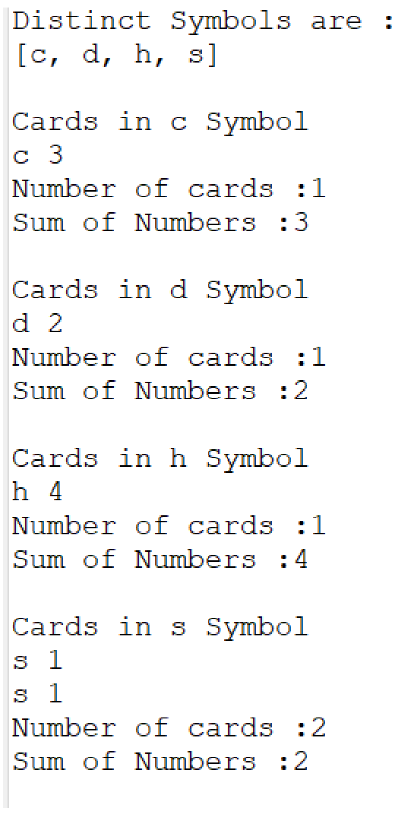
```
class Card  {
        private String symbol;
        private Integer number;
        public String getSymbol() {
                return symbol;
        }
        public void setSymbol(String symbol) {
                this.symbol = symbol;
        }
        public Integer getNumber() {
                return number;
        }
        public void setNumber(Integer number) {
                this.number = number;
        }
}
public class Exp2_1 {
public static void main(String[] args) {
HashMap<String, ArrayList<Integer>> hm1 = new HashMap<String,ArrayList<Integer>>();
System.out.println("Enter Number of Cards :");
Scanner sc1 = new Scanner(System.in);
int N = sc1.nextInt();
for(int i=1; i<=N; i++) {
        System.out.println("Enter card: " + i);
        Card cobj = new Card();
        cobj.setSymbol(sc1.next());
        cobj.setNumber(sc1.nextInt());
        ArrayList<Integer> al1 = null;
        if(hm1.containsKey(cobj.getSymbol())) {
                al1  = hm1.get(cobj.getSymbol());
                al1.add(cobj.getNumber());
                hm1.put(cobj.getSymbol(), al1);
        }else {
                al1 = new ArrayList<Integer>();
                al1.add(cobj.getNumber());
                hm1.put(cobj.getSymbol(), al1);
        }
}
System.out.println("Distinct Symbols are :");
// Sorting required
TreeMap<String, ArrayList<Integer>> tm1;
tm1 = new TreeMap<String, ArrayList<Integer>>(hm1);
System.out.println(tm1.keySet());

Set<Entry<String, ArrayList<Integer>>> s1 =  tm1.entrySet();
Iterator <Entry<String, ArrayList<Integer>>> it1 = s1.iterator();
int count;
int sum;
while(it1.hasNext()) {
        count = 0;
        sum = 0;
        Entry<String, ArrayList<Integer>> e = it1.next();
        System.out.println("\nCards in "+e.getKey()+" Symbol");
        ArrayList<Integer> list = e.getValue();
        for(Integer value : list) {
            System.out.println(e.getKey() + " " + value);
            count++;
            sum += value;
        }
        System.out.println("Number of cards :" + count);
        System.out.println("Sum of Numbers :" + sum);
    }
}
}
```

## Output:



Console × Problems Debug Shell Coverage

```
<terminated> Exp2_1 [Java Application] C:\Program Files\Java\jdk-18.0.2\bin\javaw.exe  (28-Sep-2022, 1:13:56 pm – 1:14:18 pm) [pid: 17040]
Enter Number of Cards :
5
Enter card      : 1
s
1
Enter card      : 2
d
2
Enter card      : 3
c
3
Enter card      : 4
h
4
Enter card      : 5
s
1
```



```
Distinct Symbols are :
[c, d, h, s]

Cards in c Symbol
c 3
Number of cards :1
Sum of Numbers :3

Cards in d Symbol
d 2
Number of cards :1
Sum of Numbers :2

Cards in h Symbol
h 4
Number of cards :1
Sum of Numbers :4

Cards in s Symbol
s 1
s 1
Number of cards :2
Sum of Numbers :2
```

## Viva Voce Questions:

1. What is collection framework?

2. Why we use collections?

3. What is hashset?

4. What is the difference between hashmap and hashset?

5. What is entrySet?

## Experiment 5 :

**Aim:** Create a program to show the usage of Sets of Collection interface.

## Objective: -

• To learn about concept of sets of collection.
• To learn about HashSet, List in java.

## Input/Apparatus Used:

Hardware Requirements: - Minimum 384MB RAM, 100 GB hard Disk, processor with 2.1 MHz
Software Requirements: - Eclipse, NetBeans, IntelliJ, etc.

## Reading Material:

HashSet extends AbstractSet and implements the Set interface. It creates a collection that uses a hash table for storage.

A hash table stores information by using a mechanism called hashing. In hashing, the informational content of a key is used to determine a unique value, called its hash code.

The hash code is then used as the index at which the data associated with the key is stored. The transformation of the key into its hash code is performed automatically.

Following is the list of constructors provided by the HashSet class.

**Sr.No. Constructor & Description**

1 HashSet( )This constructor constructs a default HashSet.

2 HashSet(Collection c)This constructor initializes the hash set by using the elements of the collection c.

3 HashSet(int capacity)This constructor initializes the capacity of the hash set to the given integer value capacity. The capacity grows automatically as elements are added to the HashSet.

4 HashSet(int capacity, float fillRatio)This constructor initializes both the capacity and the fill ratio (also called load capacity) of the hash set from its arguments.Here the fill ratio must be between 0.0 and 1.0, and it determines how full the hash set can be before it is resized upward. Specifically, when the number of elements is greater than the capacity of the hash set multiplied by its fill ratio, the hash set is expanded.

Apart from the methods inherited from its parent classes, HashSet defines following methods −

Sr.No. Method & Description

1 boolean add(Object o)Adds the specified element to this set if it is not already present.

2 void clear()Removes all of the elements from this set.

3 Object clone()Returns a shallow copy of this HashSet instance: the elements themselves are not cloned.

4 boolean contains(Object o)Returns true if this set contains the specified element.

5 boolean isEmpty()Returns true if this set contains no elements.

6 Iterator iterator()Returns an iterator over the elements in this set.

7 boolean remove(Object o)Removes the specified element from this set if it is present.

8 int size()Returns the number of elements in this set (its cardinality).

## Procedure/Algorithm/Pseudocode

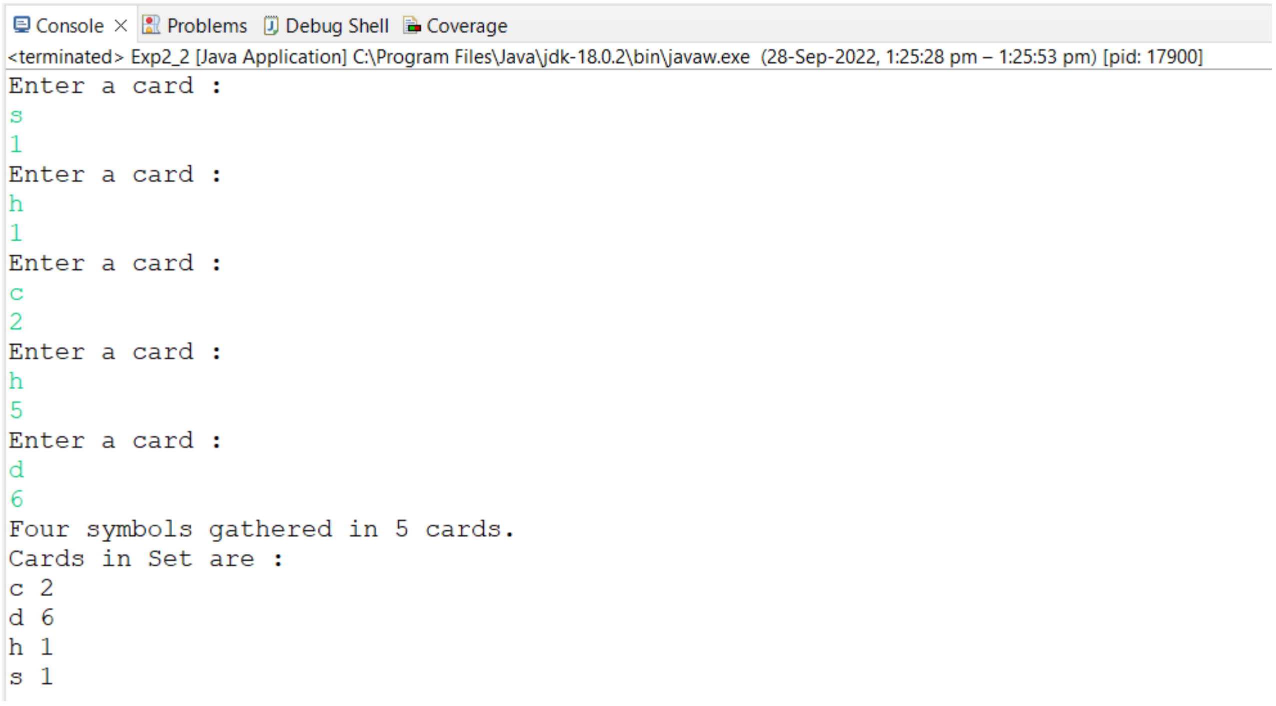
1. Create a card class having  symbol and number.
2. Add details of cards in all type of set class.
3. Then print how many cards we have to gather to have 4 unique symbol.
4. Print all unique cards in sorted order.

## Sample Code:

```
package Unit2;
import java.util.Scanner;
import java.util.TreeSet;

class Card2 implements Comparable<Card2>{
        private String symbol;
        private Integer number;
        public String getSymbol() {
                return symbol;
        }
        public void setSymbol(String symbol) {
                this.symbol = symbol;
        }
        public Integer getNumber() {
                return number;
        }
        public void setNumber(Integer number) {
```

```
		this.number = number;
	}
	@Override
	public int compareTo(Card2 o1) {
		// TODO Auto-generated method stub
		return this.getSymbol().compareTo(o1.getSymbol());
	}

}
public class Exp2_2 {
public static void main(String[] args) {
	Scanner sc1 = new Scanner(System.in);
	TreeSet<Card2> ts1 = new TreeSet<Card2>();
	int count = 0;
	while(true) {
		System.out.println("Enter a card :");
		Card2 cobj = new Card2();
		cobj.setSymbol(sc1.next());
		cobj.setNumber(sc1.nextInt());
		count++;
		ts1.add(cobj);
	if(ts1.size() == 4) {
		break;
	}
	}
	System.out.println("Four symbols gathered in "+count +" cards.");
	System.out.println("Cards in Set are :");

	for(Card2 obj : ts1) {
		System.out.println(obj.getSymbol() + " "+ obj.getNumber());
	}
}
}
```

## Output:



```
<terminated> Exp2_2 [Java Application] C:\Program Files\Java\jdk-18.0.2\bin\javaw.exe  (28-Sep-2022, 1:25:28 pm – 1:25:53 pm) [pid: 17900]
Enter a card :
s
1
Enter a card :
h
1
Enter a card :
c
2
Enter a card :
h
5
Enter a card :
d
6
Four symbols gathered in 5 cards.
Cards in Set are :
c 2
d 6
h 1
s 1
```

## Viva Voce Questions:

1. What is Hashset?

2. Difference between Hashset and Treeset?

3. Difference between Hashset and LinkedHashset?

4. Explain any two methods of collection.

5. What is the difference between Set and List?

## Experiment 6:

**Aim:** Write a Program to perform the basic operations like insert, delete, display and search in list. List contains String object items where these operations are to be performed.

### Objective: -

• To learn about concept of ArrayList.
• To learn about various methods of List.

### Input/Apparatus Used:

Hardware Requirements: - Minimum 384MB RAM, 100 GB hard Disk, processor with 2.1 MHz
Software Requirements: - Eclipse, NetBeans, IntelliJ, etc.

### Reading Material:

**Java List**

List in Java provides the facility to maintain the ordered collection. It contains the index-based methods to insert, update, delete and search the elements. It can have the duplicate elements also. We can also store the null elements in the list.

The List interface is found in the java.util package and inherits the Collection interface. It is a factory of ListIterator interface. Through the ListIterator, we can iterate the list in forward and backward directions. The implementation classes of List interface are ArrayList, LinkedList, Stack and Vector. The ArrayList and LinkedList are widely used in Java programming. The Vector class is deprecated since Java 5.

**List Interface declaration**

public interface List<E> extends Collection<E>

Java List Methods

**Method Description**

void add(int index, E element) It is used to insert the specified element at the specified position in a list.

boolean add(E e) It is used to append the specified element at the end of a list.

boolean addAll(Collection<? extends E> c) It is used to append all of the elements in the specified collection to the end of a list.

boolean addAll(int index, Collection<? extends E> c) It is used to append all the elements in the specified collection, starting at the specified position of the list.

void clear() It is used to remove all of the elements from this list.

boolean equals(Object o) It is used to compare the specified object with the elements of a list.

int hashcode() It is used to return the hash code value for a list.

E get(int index) It is used to fetch the element from the particular position of the list.

boolean isEmpty() It returns true if the list is empty, otherwise false.

int lastIndexOf(Object o) It is used to return the index in this list of the last occurrence of the specified element, or -1 if the list does not contain this element.

Object[] toArray() It is used to return an array containing all of the elements in this list in the correct order.

<T> T[] toArray(T[] a) It is used to return an array containing all of the elements in this list in the correct order.

boolean contains(Object o) It returns true if the list contains the specified element

boolean containsAll(Collection<?> c) It returns true if the list contains all the specified element

int indexOf(Object o) It is used to return the index in this list of the first occurrence of the specified element, or -1 if the List does not contain this element.

E remove(int index) It is used to remove the element present at the specified position in the list.

boolean remove(Object o) It is used to remove the first occurrence of the specified element.

boolean removeAll(Collection<?> c) It is used to remove all the elements from the list.

void replaceAll(UnaryOperator<E> operator) It is used to replace all the elements from the list with the specified element.

void retainAll(Collection<?> c) It is used to retain all the elements in the list that are present in the specified collection.

E set(int index, E element) It is used to replace the specified element in the list, present at the specified position.

void sort(Comparator<? super E> c) It is used to sort the elements of the list on the basis of specified comparator.

Spliterator<E> spliterator() It is used to create spliterator over the elements in a list.

List<E> subList(int fromIndex, int toIndex) It is used to fetch all the elements lies within the given range.

int size() It is used to return the number of elements present in the list.

**Java List vs ArrayList**

List is an interface whereas ArrayList is the implementation class of List.

How to create List

The ArrayList and LinkedList classes provide the implementation of List interface. Let's see the examples to create the List:

```
//Creating a List of type String using ArrayList

List<String> list=new ArrayList<String>();



//Creating a List of type Integer using ArrayList

List<Integer> list=new ArrayList<Integer>();



//Creating a List of type Book using ArrayList

List<Book> list=new ArrayList<Book>();



//Creating a List of type String using LinkedList

List<String> list=new LinkedList<String>();
```

In short, you can create the List of any type. The ArrayList<T> and LinkedList<T> classes are used to specify the type. Here, T denotes the type.

## Procedure/Algorithm/Pseudocode

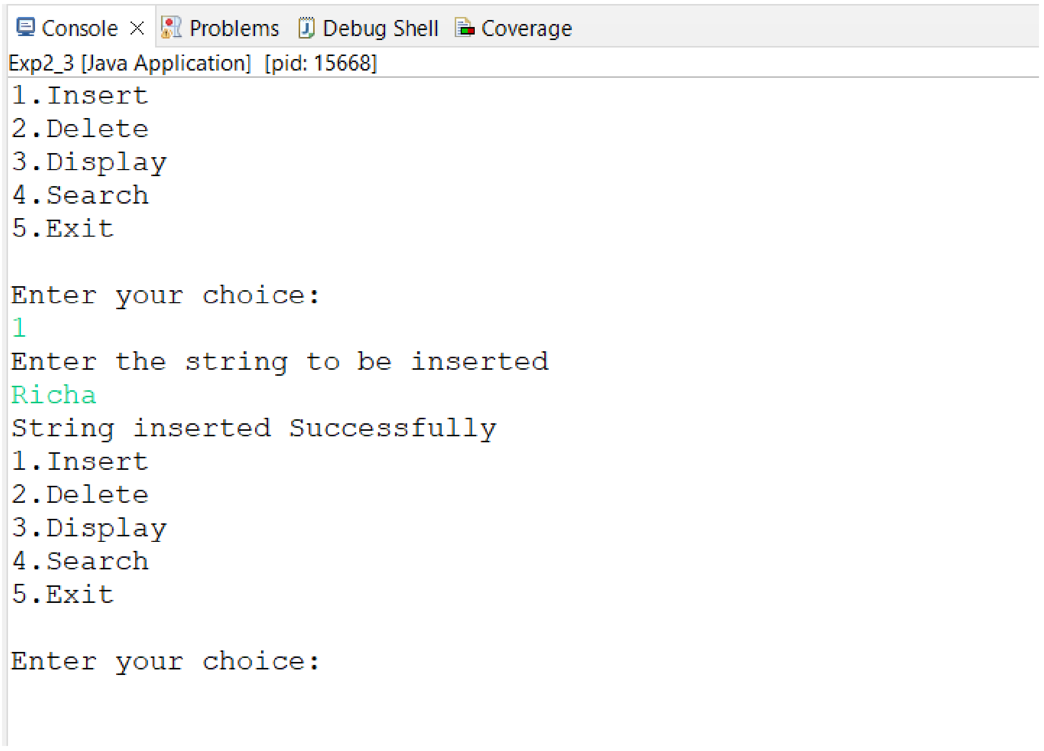
Create main class where you ask for your choice using switch case. Choice can be

1. Insert String in ArrayList/LinkedList:
2. Delete String from ArrayList/LinkedList:
3. Display Sring all elements of ArrayList/LinkedList:
4. Search String in ArrayList/LinkedList:
5. Exit

## Sample Code:

```
package Unit2;

import java.util.ArrayList;

import java.util.Scanner;

public class Exp2_3 {

public static void main(String args[]){

            Scanner sc = new Scanner(System.in); int ch;

            String s;

            ArrayList<String> l = new ArrayList<String>();

            do {

             System.out.println("1.Insert \n2.Delete \n3.Display \n4.Search \n5.Exit \n");

             System.out.println("Enter your choice:");

            ch = Integer.parseInt(sc.nextLine());

             switch (ch) {

            case 1:

             System.out.println("Enter the string to be inserted");

            s = sc.nextLine(); l.add(s);

             System.out.println("String inserted Successfully");

            break; case 2:

             System.out.println("Enter the string to be deleted");
}

            s = sc.nextLine(); if (l.contains(s)) {
            l.remove(s);
             System.out.println("String deleted Successfully");
            } else {
             System.out.println("String not found");
             }
             break;
            case 3:
             System.out.println("The list is: ");
             for (String i : l)
            System.out.println(i);
             break;
            case 4:
             System.out.println("Enter the string to be searched");
            s = sc.nextLine(); if (l.contains(s))
             System.out.println("Item present in the list");
            else
             System.out.println("Item is not present in the list");
            break; case 5:
             System.out.println("Exiting...");
             System.exit(0);
             }
             } while (ch != 5);
             }
}
```

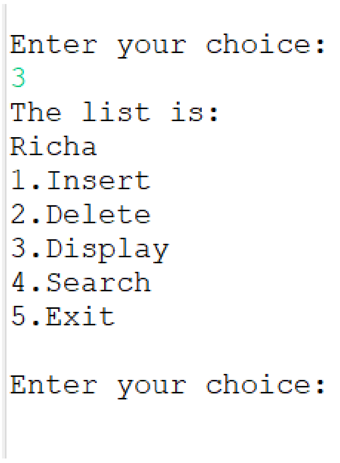
## Output:



Console × Problems Debug Shell Coverage

Exp2_3 [Java Application] [pid: 15668]

```
1.Insert
2.Delete
3.Display
4.Search
5.Exit

Enter your choice:
1
Enter the string to be inserted
Richa
String inserted Successfully
1.Insert
2.Delete
3.Display
4.Search
5.Exit

Enter your choice:
```



```
Enter your choice:
3
The list is:
Richa
1.Insert
2.Delete
3.Display
4.Search
5.Exit

Enter your choice:
```

## Viva Voce Questions:

1. What is arrayList?

2. What is the difference between array and arrayList?

3. What is the use of contains keyword?

4. What is the difference between remove and removeAll?

5. Name the function used to add all elements in list.

## Experiment 7:

**Aim:** Create a menu based Java application with the following options.1. Add an Employee2.Display All3.Exit If option 1 is selected, the application should gather details of the employee like employee name, employee id, designation and salary and store it in a file. If option 2 is selected, the application should display all the employee details. If option 3 is selected the application should exit.

### Objective: -

• To learn about concept of File Handling in java.
• To learn about LinkedList, Exception Handling in java.

### Input/Apparatus Used:

Hardware Requirements: - Minimum 384MB RAM, 100 GB hard Disk, processor with 2.1 MHz
Software Requirements: - Eclipse, NetBeans, IntelliJ, etc.

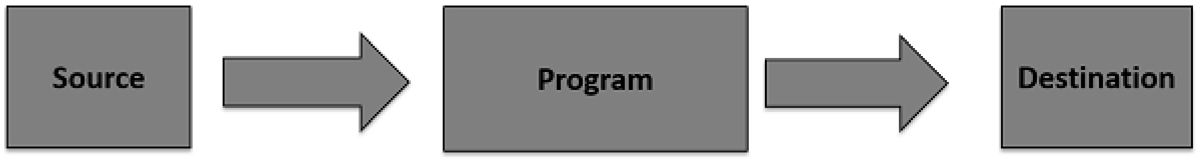
### Reading Material:

**Java I/O** (Input and Output) is used to process the input and produce the output.

Java uses the concept of a stream to make I/O operation fast. The java.io package contains all the classes required for input and output operations.We can perform **file handling in Java** by Java I/O API.

The java.io package contains nearly every class you might ever need to perform input and output (I/O) in Java. All these streams represent an input source and an output destination. The stream in the java.io package supports many data such as primitives, object, localized characters, etc.
*Stream*
A stream can be defined as a sequence of data. There are two kinds of Streams −

- InPutStream − The InputStream is used to read data from a source.
- OutPutStream − The OutputStream is used for writing data to a destination.



Java provides strong but flexible support for I/O related to files and networks but this tutorial covers very basic functionality related to streams and I/O. We will see the most commonly used examples one by one −

**Byte Streams**

Java byte streams are used to perform input and output of 8-bit bytes. Though there are many classes related to byte streams but the most frequently used classes are, FileInputStream and FileOutputStream. Following is an example which makes use of these two classes to copy an input file into an output file −

**Example**

```
import java.io.*;
public class CopyFile {

public static void main(String args[]) throws IOException {
FileInputStream in = null;
FileOutputStream out = null;

try {
in = new FileInputStream("input.txt");
out = new FileOutputStream("output.txt");
int c;
while ((c = in.read()) != -1) {
out.write(c);
}
}finally {
if (in != null) {
in.close();
}
if (out != null) {
out.close();
}
}
}
}
```

Now let's have a file input.txt with the following content −

This is test for copy file.

As a next step, compile the above program and execute it, which will result in creating output.txt file with the same content as we have in input.txt. So let's put the above code in CopyFile.java file and do the following −

```
$javac CopyFile.java
$java CopyFile
```

**Character Streams**

Java Byte streams are used to perform input and output of 8-bit bytes, whereas Java Character streams are used to perform input and output for 16-bit unicode. Though there are many classes related to character streams but the most frequently used classes are, FileReader and FileWriter. Though internally FileReader uses FileInputStream and FileWriter uses FileOutputStream but here the major difference is that FileReader reads two bytes at a time and FileWriter writes two bytes at a time.

We can re-write the above example, which makes the use of these two classes to copy an input file (having unicode characters) into an output file −

**Example**

```
import java.io.*;
public class CopyFile {

public static void main(String args[]) throws IOException {
FileReader in = null;
FileWriter out = null;

try {
in = new FileReader("input.txt");
out = new FileWriter("output.txt");
int c;
while ((c = in.read()) != -1) {
out.write(c);
}
}finally {
if (in != null) {
in.close();
}
if (out != null) {
out.close();
}
}
}
}
```

Now let's have a file input.txt with the following content −

This is test for copy file.

As a next step, compile the above program and execute it, which will result in creating output.txt file with the same content as we have in input.txt. So let's put the above code in CopyFile.java file and do the following −

```
$javac CopyFile.java
$java CopyFile
```

**Java - Serialization**

Java provides a mechanism, called object serialization where an object can be represented as a sequence of bytes that includes the object's data as well as information about the object's type and the types of data stored in the object.

After a serialized object has been written into a file, it can be read from the file and deserialized that is, the type information and bytes that represent the object and its data can be used to recreate the object in memory.

Most impressive is that the entire process is JVM independent, meaning an object can be serialized on one platform and deserialized on an entirely different platform.

Classes ObjectInputStream and ObjectOutputStream are high-level streams that contain the methods for serializing and deserializing an object.

The ObjectOutputStream class contains many write methods for writing various data types, but one method in particular stands out −

```
public final void writeObject(Object x) throws IOException
```

The above method serializes an Object and sends it to the output stream. Similarly, the ObjectInputStream class contains the following method for deserializing an object −

```
public final Object readObject() throws IOException, ClassNotFoundException
```

This method retrieves the next Object out of the stream and deserializes it. The return value is Object, so you will need to cast it to its appropriate data type.

## Procedure/Algorithm/Pseudocode

1. Create a class Employee to store its details like id, name ,salary, age, etc.
2. Create a EmployeeManager class where you ask for your choice using:
   - Add an employee
   - Display All
   - Exit

## Sample Code:

```
package com.EM.Model;

import java.io.Serializable;

public class Employee implements Serializable {
        private int id;
        private String name;
        private double salary;
        private int age;
        public int getId() {
                return id;
        }
        public void setId(int id) {
                this.id = id;
        }
        public String getName() {
                return name;
        }
        @Override
```

```
	public String toString() {
		return "Employee [id=" + id + ", name=" + name + ", salary=" + salary
				+ ", age=" + age + "]";
	}
	public Employee(int id, String name, double salary, int age) {
		super();
		//this.id = id;
		setId(id);
		//this.name = name;
		setName(name);
		//this.salary = salary;
		setSalary(salary);
		//this.age = age;
		setAge(age);
	}
	public void setName(String name) {
		this.name = name;
	}
	public double getSalary() {
		return salary;
	}
	public void setSalary(double salary) {
		this.salary = salary;
	}
	public int getAge() {
		return age;
	}
	public void setAge(int age) {
		this.age = age;
	}
}

package com.EM.Controller;

import java.awt.DisplayMode;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.OutputStream;
import java.util.InputMismatchException;
import java.util.Iterator;
import java.util.LinkedList;
import java.util.List;
import java.util.Scanner;

import com.EM.Model.Employee;
import com.EM.Util.Utils;

public class EmployeeManager {
	int menuChoice=0;
	Scanner objectScanner= new Scanner(System.in);
	List<Employee> objectEmployeeList= new LinkedList<Employee>();
	public static void main(String[] args) {
		EmployeeManager objectManager= new EmployeeManager();
		objectManager.displayMenu();
		while(objectManager.menuChoice!=5)
		{
			objectManager.displayMenu();
		}

	}

	void displayMenu()
	{
		Utils.so("Main Menu");
		Utils.so("1. Add an employee");
		Utils.so("2. Display All");
		Utils.so("3. Write all the entered employees to file");
		Utils.so("4. Read all the employees from the file");
		Utils.so("5. Exit");

		menuChoice=objectScanner.nextInt();
		try
		{

			switch(menuChoice)
			{
			case 1:
				getEmployeeInformationFromConsole();
				break;
			case 2:
				displayFromList(objectEmployeeList);
				break;
			case 3:
				Utils.so("Write the file name you want to write to ");

				writeEmployeesToFile(objectScanner.next());
				break;
			case 4:
				Utils.so("Write the file name you want to read from ");
```

```
						readEmployeesFromFile(objectScanner.next());
						break;
				case 5:
						Utils.so("Exiting the program");
						System.exit(0);
				}
		}
		catch(InputMismatchException e)
		{
				Utils.so("Exception is----->"+e);
		}

	}

	void writeEmployeesToFile(String fileName)
	{
		File fileObject= new File(fileName);
		FileOutputStream fileStreamObject;
		try {
			fileStreamObject = new FileOutputStream(fileObject);
			System.out.println(" inside file");

			ObjectOutputStream objectWriterToFile= new
ObjectOutputStream(fileStreamObject);
			System.out.println();
			objectWriterToFile.writeObject(objectEmployeeList);
		} catch (FileNotFoundException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}

	}

	String readEmployeesFromFile(String fileName)
	{
		String status;
		File fileobject= new File(fileName);
		if(!fileobject.exists())
		{
			status="unable to create the file";
			//Utils.so("Unable to create the file");
			return status;
		}
		FileInputStream fileStreamObject;
		try {
			fileStreamObject = new FileInputStream(fileobject);
			ObjectInputStream objectReaderFromFile= new
ObjectInputStream(fileStreamObject);
			objectEmployeeList.clear();
			objectEmployeeList=(List<Employee>)
objectReaderFromFile.readObject();
			status=displayFromList(objectEmployeeList);
			if(status.equalsIgnoreCase("Displayed Successfully"))
			{
				status="Files read from file successfully";
			}
			else
			{
				status=""
			}
		} catch (FileNotFoundException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (ClassNotFoundException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		finally
		{

		}

	}
	void getEmployeeInformationFromConsole()
	{

		Utils.so("Enter Employee ID:");
		try{
		int id=objectScanner.nextInt();
		Utils.so("Enter Employee Name:");
		String name=objectScanner.next();
		Utils.so("Enter Employee Age:");
		int age=objectScanner.nextInt();
		Utils.so("Enter Employee Salary:");
		double salary= objectScanner.nextDouble();
		objectEmployeeList.add(new  Employee(id, name, salary, age));
		}
```

```
            catch(InputMismatchException e)
            {
                System.out.println(""+e);
            }

        }
        String displayFromList(List<Employee> objectList)
        {
            String status="";
            try{
            Iterator objectIterator= objectList.iterator();
            while(objectIterator.hasNext())
            {
                Utils.so(""+objectIterator.next());

            }
            status="Displayed Successfully";
            return status;

            }
            catch(Exception e)
            {

                status="Failure";
                return status;
            }

        }
        void displayInformationForSpecificName(List<Employee> objectList,String name)
        {
            String status="";
            Iterator objectIterator= objectList.iterator();
            while(objectIterator.hasNext())
            {
                //Utils.so(""+objectIterator.next());
                Employee objectLocalEmployee=(Employee) objectIterator.next();
                if(objectLocalEmployee.getName().equalsIgnoreCase("1"))
                {
                    Utils.so(""+objectLocalEmployee);
                }
                //Utils.so(objectLocalEmployee.getName());
            }
        }
    }
```

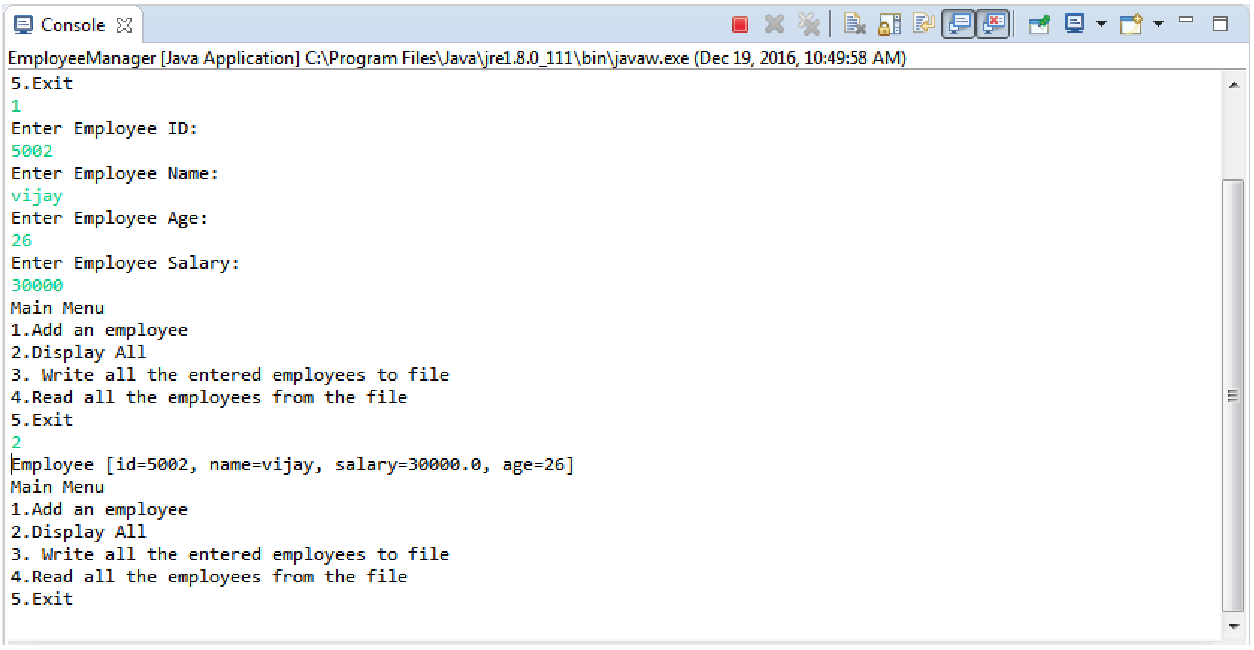
## Output:

Compilation

Javac EmployeeManager.java

Execution

java EmployeeManager



```
EmployeeManager [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (Dec 19, 2016, 10:49:58 AM)
5.Exit
1
Enter Employee ID:
5002
Enter Employee Name:
vijay
Enter Employee Age:
26
Enter Employee Salary:
30000
Main Menu
1.Add an employee
2.Display All
3. Write all the entered employees to file
4.Read all the employees from the file
5.Exit
2
Employee [id=5002, name=vijay, salary=30000.0, age=26]
Main Menu
1.Add an employee
2.Display All
3. Write all the entered employees to file
4.Read all the employees from the file
5.Exit
```

## Viva Voce Questions:

1. Filenotfound is checked or unchecked exception?

2. What is the difference between InputStream and OutputStream in Java?

3. What is the difference between BufferedReader and FileReader in Java?

4. What are the types of I / O streams?

5. Why you need to close the streams in finally block?

## Experiment 8:

**Aim:** Create a palindrome creator application for making a longest possible palindrome out of given input string**.**

## Objective: -

• To learn about concept of HashMap in java.
• To learn about various methods of HashMap.

## Input/Apparatus Used:

Hardware Requirements: - Minimum 384MB RAM, 100 GB hard Disk, processor with 2.1 MHz
Software Requirements: - Eclipse, NetBeans, IntelliJ, etc.

## Reading Material:

**What is palindrome?**

A palindrome is when a word, date or phrase reads exactly the same forwards as it does backwards. For example, the name "Hannah" reads the same when in reverse.

**HashMap**

Java **HashMap** class implements the Map interface which allows us *to store key and value pair*, where keys should be unique. If you try to insert the duplicate key, it will replace the element of the corresponding key. It is easy to perform operations using the key index like updation, deletion, etc. HashMap class is found in the java.util package.

HashMap in Java is like the legacy Hashtable class, but it is not synchronized. It allows us to store the null elements as well, but there should be only one null key. Since Java 5, it is denoted as HashMap<K,V>, where K stands for key and V for value. It inherits the AbstractMap class and implements the Map interface.

- Java HashMap contains values based on the key.
- Java HashMap contains only unique keys.
- Java HashMap may have one null key and multiple null values.
- Java HashMap is non synchronized.
- Java HashMap maintains no order.
- The initial default capacity of Java HashMap class is 16 with a load factor of 0.75.

**Constructors of Java HashMap class**

| Constructor | Description |
|---|---|
| HashMap() | It is used to construct a default HashMap. |
| HashMap(Map<? extends K,? extends V> m) | It is used to initialize the hash map by using the elements of the given Map object m. |
| HashMap(int capacity) | It is used to initializes the capacity of the hash map to the given integer value, capacity. |
| HashMap(int capacity, float loadFactor) | It is used to initialize both the capacity and load factor of the hash map by using its arguments. |

**Methods of Java HashMap class**

| **Method** | **Description** |
|---|---|
| void clear() | It is used to remove all of the mappings from this map. |
| boolean isEmpty() | It is used to return true if this map contains no key-value mappings. |
| Object clone() | It is used to return a shallow copy of this HashMap instance: the keys and values themselves are not cloned. |
| Set entrySet() | It is used to return a collection view of the mappings contained in this map. |
| Set keySet() | It is used to return a set view of the keys contained in this map. |
| V put(Object key, Object value) | It is used to insert an entry in the map. |
| void putAll(Map map) | It is used to insert the specified map in the map. |
| V putIfAbsent(K key, V value) | It inserts the specified value with the specified key in the map only if it is not already specified. |
| V remove(Object key) | It is used to delete an entry for the specified key. |
| boolean remove(Object key, Object value) | It removes the specified values with the associated specified keys from the map. |
| V compute(K key, BiFunction<? super K,? super V,? extends V> remappingFunction) | It is used to compute a mapping for the specified key and its current mapped value (or null if there is no current mapping). |

| | |
|---|---|
| V computeIfAbsent(K key, Function<? super K,? extends V> mappingFunction) | It is used to compute its value using the given mapping function, if the specified key is not already associated with a value (or is mapped to null), and enters it into this map unless null. |
| V computeIfPresent(K key, BiFunction<? super K,? super V,? extends V> remappingFunction) | It is used to compute a new mapping given the key and its current mapped value if the value for the specified key is present and non-null. |
| boolean containsValue(Object value) | This method returns true if some value equal to the value exists within the map, else return false. |
| boolean containsKey(Object key) | This method returns true if some key equal to the key exists within the map, else return false. |
| boolean equals(Object o) | It is used to compare the specified Object with the Map. |

## Procedure/Algorithm/Pseudocode

First, we need to consider the type of value.

- If it is a number, we need to change it to a string to compare how it reads backwards and forwards.
- If it is an object, we need to somehow also change it to a string to do a comparison.
- If it is a string, we can forge ahead.

- Compare a string with its reversed version.
- Iterate using for loop and check to see if character on other side of string.
- Use recursion to check the first and last letters before invoking the function again with the shortened string.

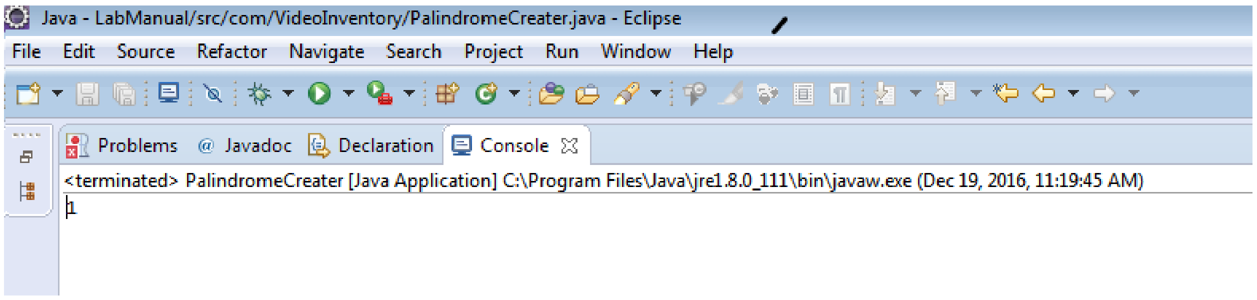
## Sample Code:

```
import java.util.HashMap;
import java.util.Iterator;
import java.util.Set;
public class PalindromeCreater {

        /**
         * @param args
         */
```

```
public static void main(String[] args) {
        // TODO Auto-generated method stub

        palindromeCreater("mnayan");
}

static int palindromeCreater(String input1)
{
        int valueToBeReturned=0;
        HashMap<Character,Integer> object= new HashMap<>();
        HashMap<Character,Integer> object2= new HashMap<>();
        input1=input1.toLowerCase();
        for(int i=0;i<input1.length();i++)
        {
                if(object.get(input1.charAt(i)) != null)
                {
                        object.put(input1.charAt(i),object.get(input1.charAt(i))+1);
                }
                else
                {
                        object.put(input1.charAt(i), 1);
                }
        }
        Set<Character> objectKeys=object.keySet();
        Iterator i2=objectKeys.iterator();
        int value=-1;
        if(objectKeys.size()==1)
        {
                valueToBeReturned=0;
        }
        else if(objectKeys.size()>1)
        {
                while(i2.hasNext())
                {
                        if(object.get(i2.next())!=1)
                        {
                                value=1;
                        }

                }
                if(value==-1)
                {
                        valueToBeReturned=-1;
                }
                else{

                object2=(HashMap<Character, Integer>) object.clone();
```

```
                        Set<Character> objectKeys2=object.keySet();
                        Iterator iterator=objectKeys2.iterator();

                                Boolean flag=true;
                                while(iterator.hasNext())
                                {
                                        Character a=(Character) iterator.next();
                                        if(object.get(a)==1&&flag)
                                        {
                                                flag=false;
                                        }
                                        else if(object.get(a)%2!=0)
                                        {
                                                object2.remove(a);
                                                valueToBeReturned++;
                                        }
                                }
                        }
                }

                System.out.println(valueToBeReturned);
                return valueToBeReturned;
        }
}
```

## Output:



```
<terminated> PalindromeCreater [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (Dec 19, 2016, 11:19:45 AM)
1
```

## Viva Voce Questions:

1. What is hashMap?

2. What's the purpose of the put() method of HashMap in Java?

3. Explain get and clone int and Integer in java?

4. Difference between entrySet and Keyset?

5. What happens if you try to store a duplicate key in the HashMap?

## Experiment 9:

**Aim:** Create a Servlet/ application with a facility to print any message on web browser.

### Objective: -

• To learn about concept of Servlets in java.
• To learn about Apache Server.

### Input/Apparatus Used:

Hardware Requirements: - Minimum 384MB RAM, 100 GB hard Disk, processor with 2.1 MHz
Software Requirements: - Eclipse, NetBeans, IntelliJ, etc.

### Reading Material:

**Servlet** technology is used to create a web application (resides at server side and generates a dynamic web page).

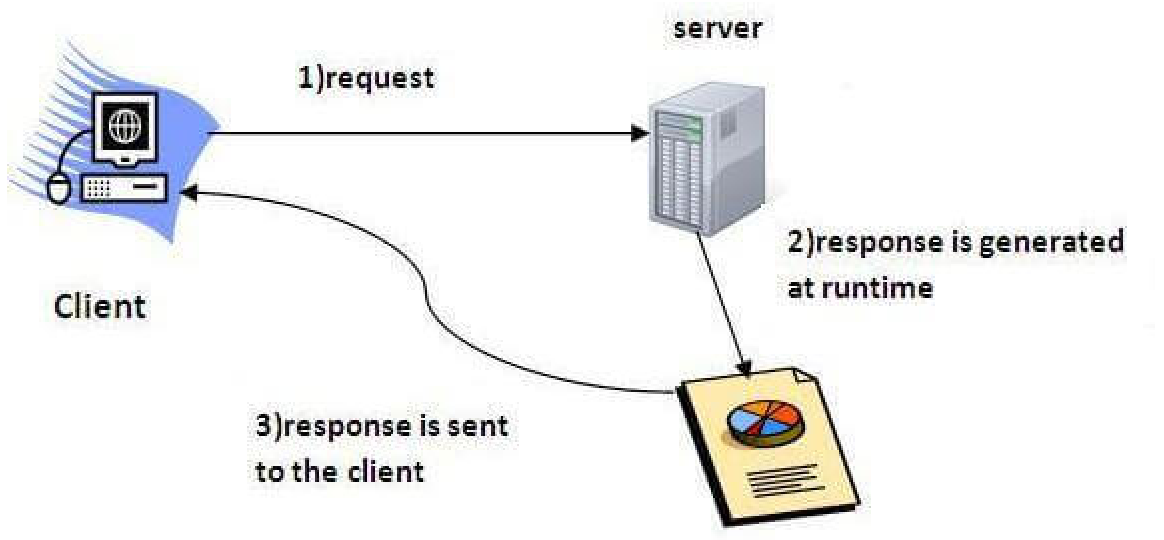
**Servlet** technology is robust and scalable because of java language. Before Servlet, CGI (Common Gateway Interface) scripting language was common as a server-side programming language. However, there were many disadvantages to this technology. We have discussed these disadvantages below.

There are many interfaces and classes in the Servlet API such as Servlet, GenericServlet, HttpServlet, ServletRequest, ServletResponse, etc.

**What is a Servlet?**

Servlet can be described in many ways, depending on the context.

- Servlet is a technology which is used to create a web application.
- Servlet is an API that provides many interfaces and classes including documentation.
- Servlet is an interface that must be implemented for creating any Servlet.
- Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
- Servlet is a web component that is deployed on the server to create a dynamic web page.


server
1)request
Client
2)response is generated at runtime
3)response is sent to the client

**Servlet Interface**

**Servlet interface provides** commonbehaviorto all the servlets.Servlet interface defines methods that all servlets must implement.

Servlet interface needs to be implemented for creating any servlet (either directly or indirectly). It provides 3 life cycle methods that are used to initialize the servlet, to service the requests, and to destroy the servlet and 2 non-life cycle methods.

**Methods of Servlet interface**

There are 5 methods in Servlet interface. The init, service and destroy are the life cycle methods of servlet. These are invoked by the web container.

**Method Description**

public void init(ServletConfig config) initializes the servlet. It is the life cycle method of servlet and invoked by the web container only once.

public void service(ServletRequest request,ServletResponse response) provides response for the incoming request. It is invoked at each request by the web container.

public void destroy() is invoked only once and indicates that servlet is being destroyed.

public ServletConfig getServletConfig() returns the object of ServletConfig.

public String getServletInfo() returns information about servlet such as writer, copyright, version etc.

**Servlet Example by implementing Servlet interface**

File: First.java

```
import java.io.*;
import javax.servlet.*;

public class First implements Servlet{
ServletConfig config=null;

public void init(ServletConfig config){
this.config=config;
System.out.println("servlet is initialized");
}

public void service(ServletRequest req,ServletResponse res)
throws IOException,ServletException{

res.setContentType("text/html");

PrintWriter out=res.getWriter();
out.print("<html><body>");
out.print("<b>hello simple servlet</b>");
out.print("</body></html>");

}
public void destroy(){System.out.println("servlet is destroyed");}
public ServletConfig getServletConfig(){return config;}
public String getServletInfo(){return "copyright 2007-1010";}

}
```

## Procedure/Algorithm/Pseudocode

You need to follow the following steps to create the servlet in the myeclipse IDE. The steps are as follows:

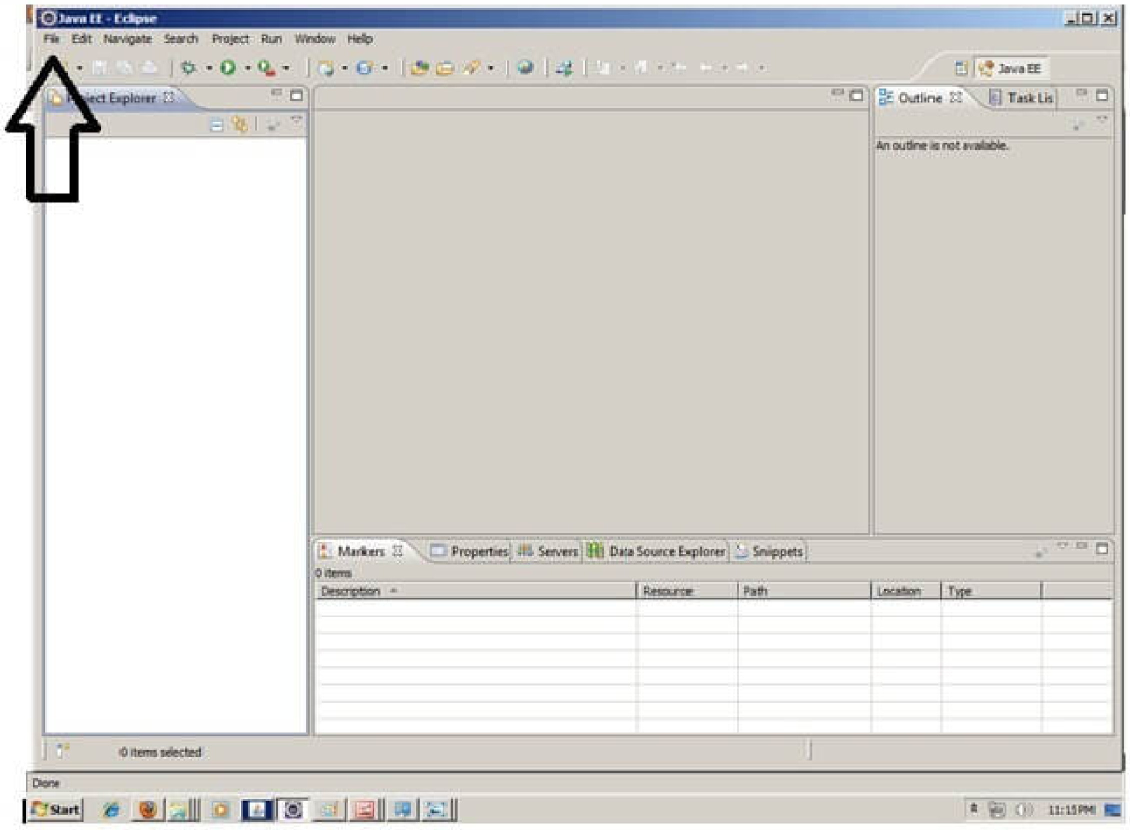
- Create a web project
- create a html file
- create a servlet
- start myeclipse tomcat server and deploy project

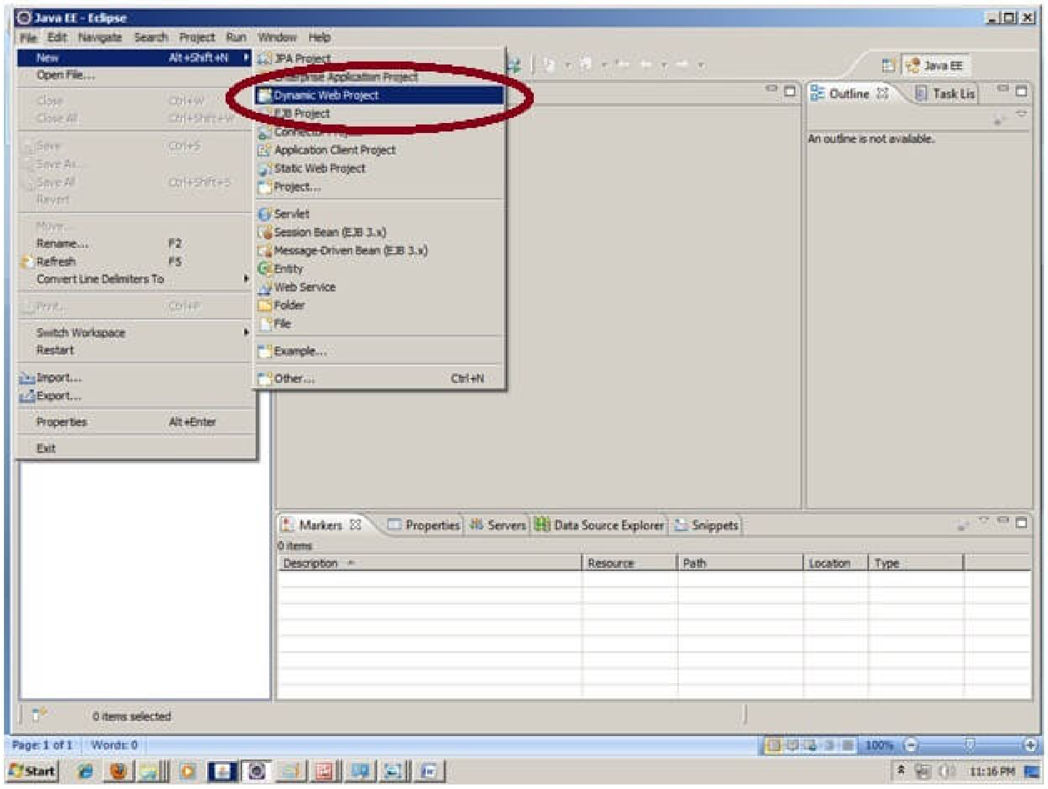
Creating **servlet example in eclipse ide**, saves a lot of work to be done. It is easy and simple to create a servlet example. Let's see the steps, you need to follow to create the first servlet example.

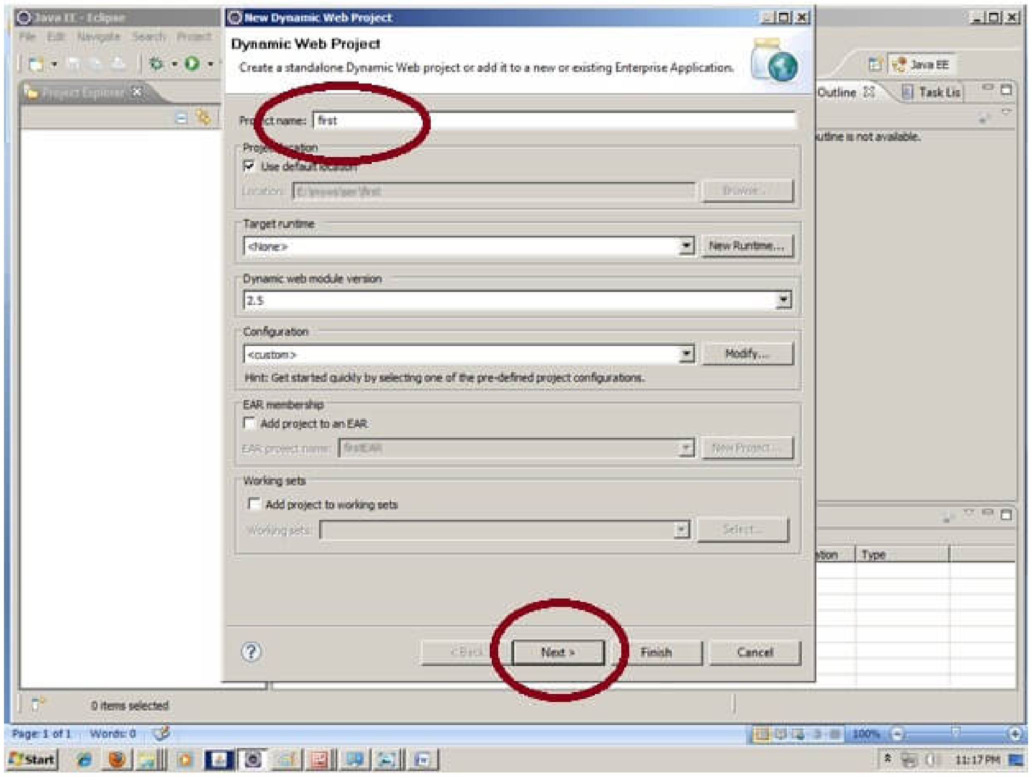
- Create a Dynamic web project
- create a servlet
- add servlet-api.jar file
- Run the servlet

1) Create the dynamic web project:

For creating a dynamic web project **click on File Menu -> New -> Project..-> Web -> dynamic web project -> write your project name e.g. first -> Finish**.Unboxing EKSA telecom H5 Bluetooth Computer Headset : Good Tech Cheap







Project name: first

Project location
[✓] Use default location
Location: E:\myws\ser\first

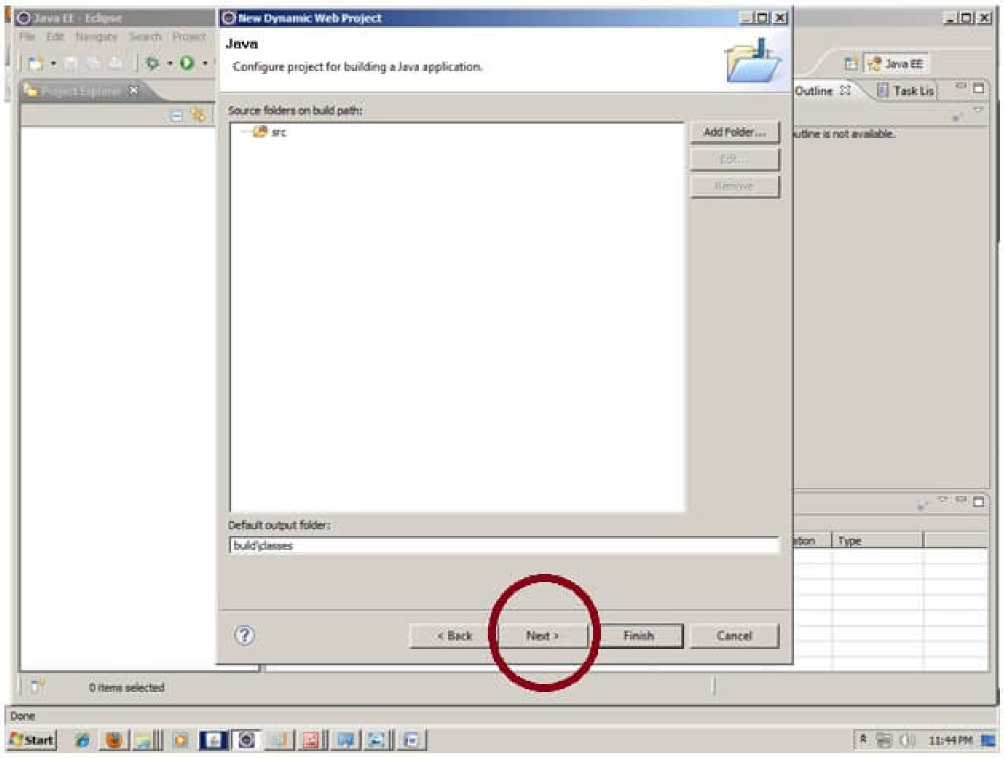
Target runtime: <None>

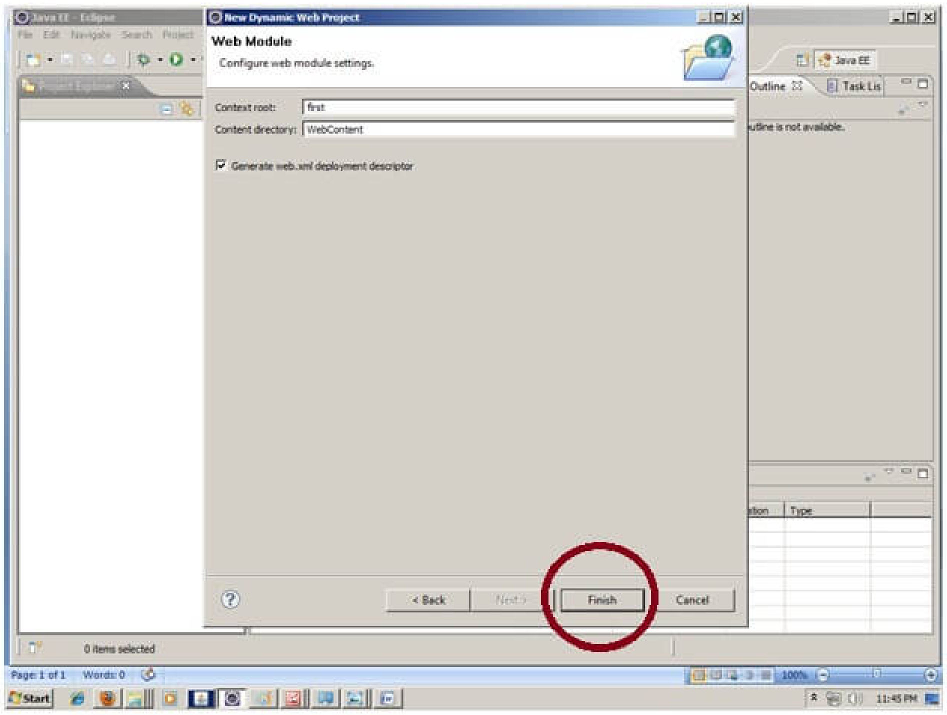
Dynamic web module version: 2.5

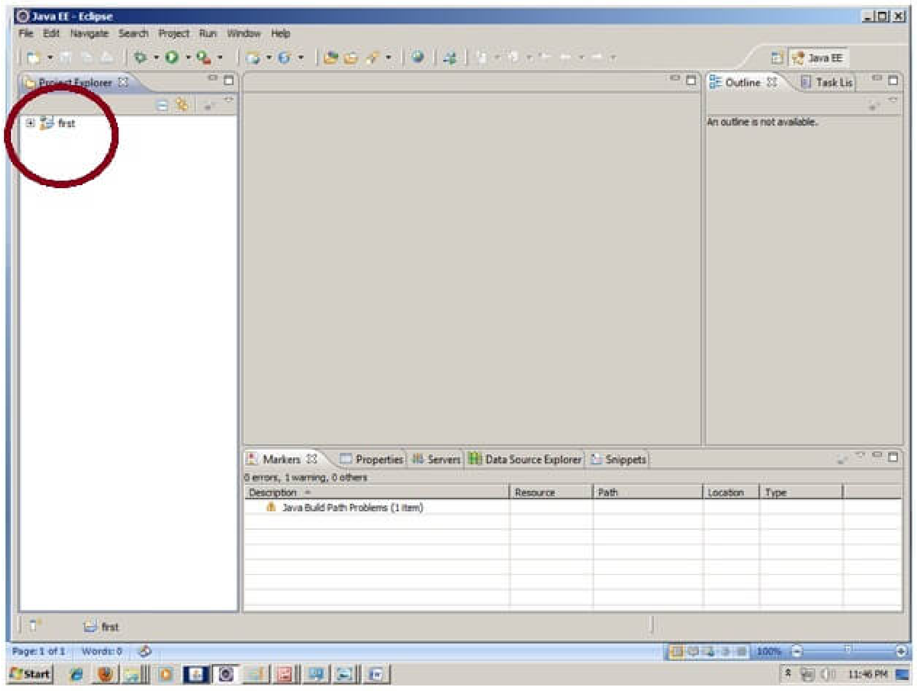
Configuration: <custom>

EAR membership
[ ] Add project to an EAR
EAR project name: firstEAR

Working sets
[ ] Add project to working sets
Working sets:


New Dynamic Web Project
Java
Configure project for building a Java application.
Source folders on build path:
src
Add Folder...
Edit...
Remove
Default output folder:
build\classes
< Back
Next >
Finish
Cancel


New Dynamic Web Project
Web Module
Configure web module settings.
Context root: first
Content directory: WebContent
Generate web.xml deployment descriptor
< Back
Next >
Finish
Cancel



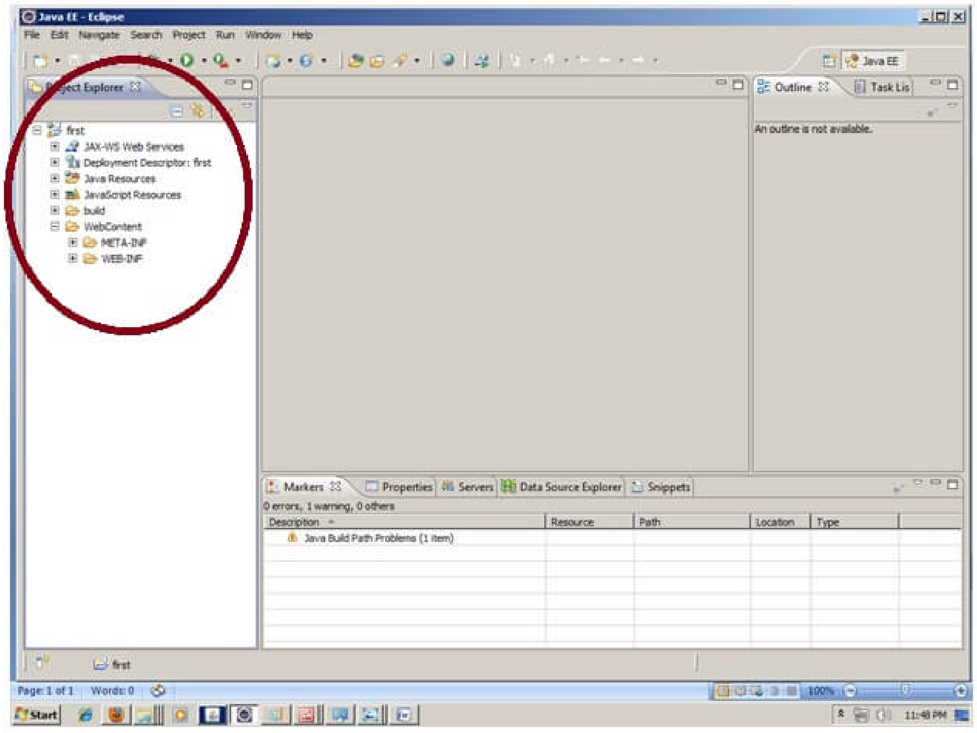
An outline is not available.

0 errors, 1 warning, 0 others

| Description | Resource | Path | Location | Type |
|---|---|---|---|---|
| Java Build Path Problems (1 item) | | | | |

2) Create the servlet in eclipse IDE:

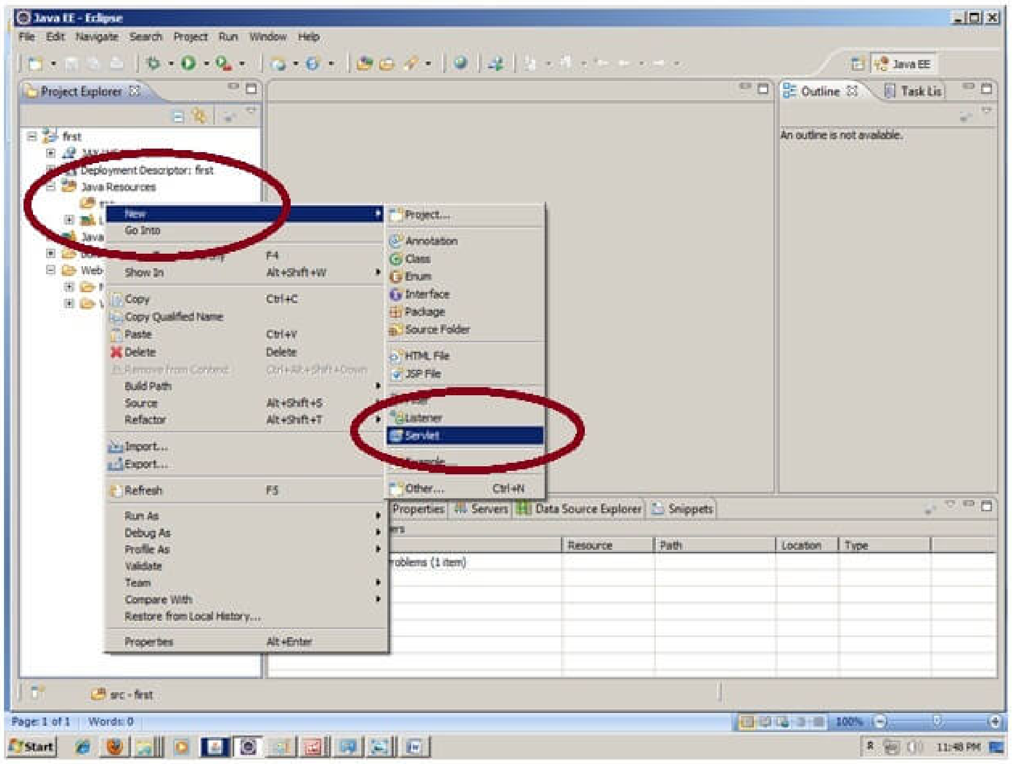
For creating a servlet, **explore the project by clicking the + icon -> explore the Java Resources -> right click on src -> New -> servlet -> write your servlet name e.g. Hello -> uncheck all the checkboxes except doGet() -> next -> Finish**.

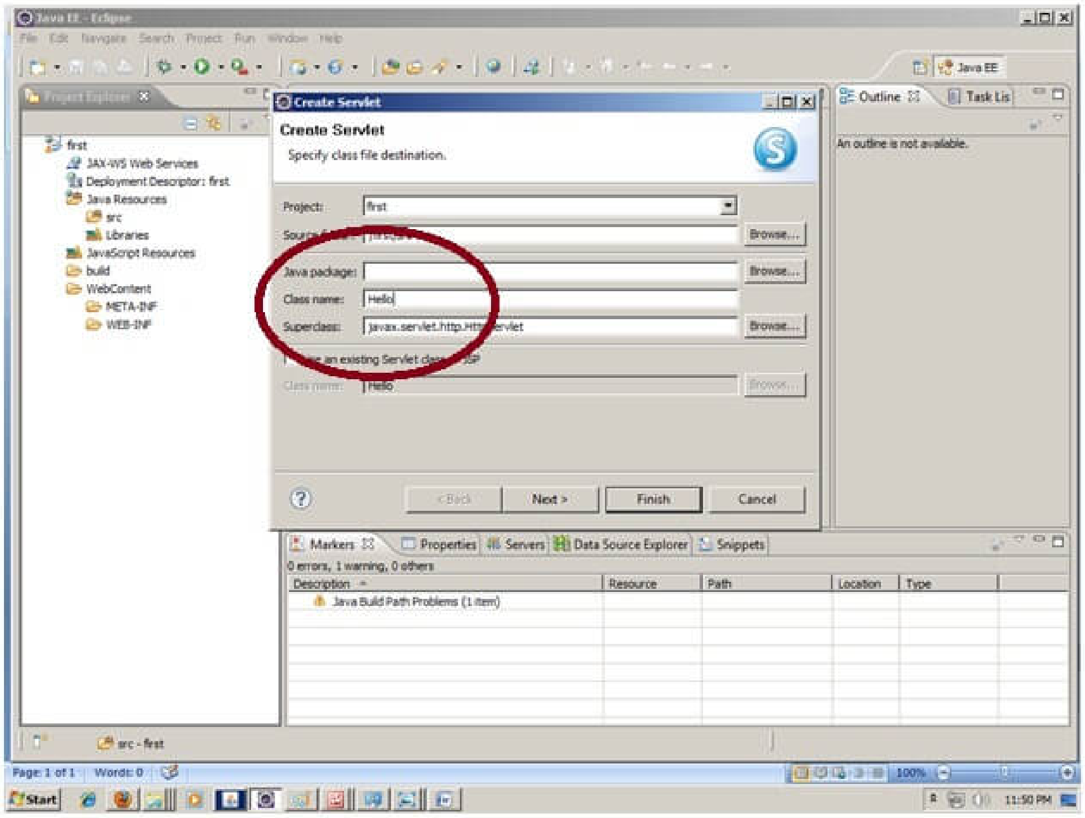


An outline is not available.

0 errors, 1 warning, 0 others

| Description | Resource | Path | Location | Type |
|---|---|---|---|---|
| Java Build Path Problems (1 item) | | | | |



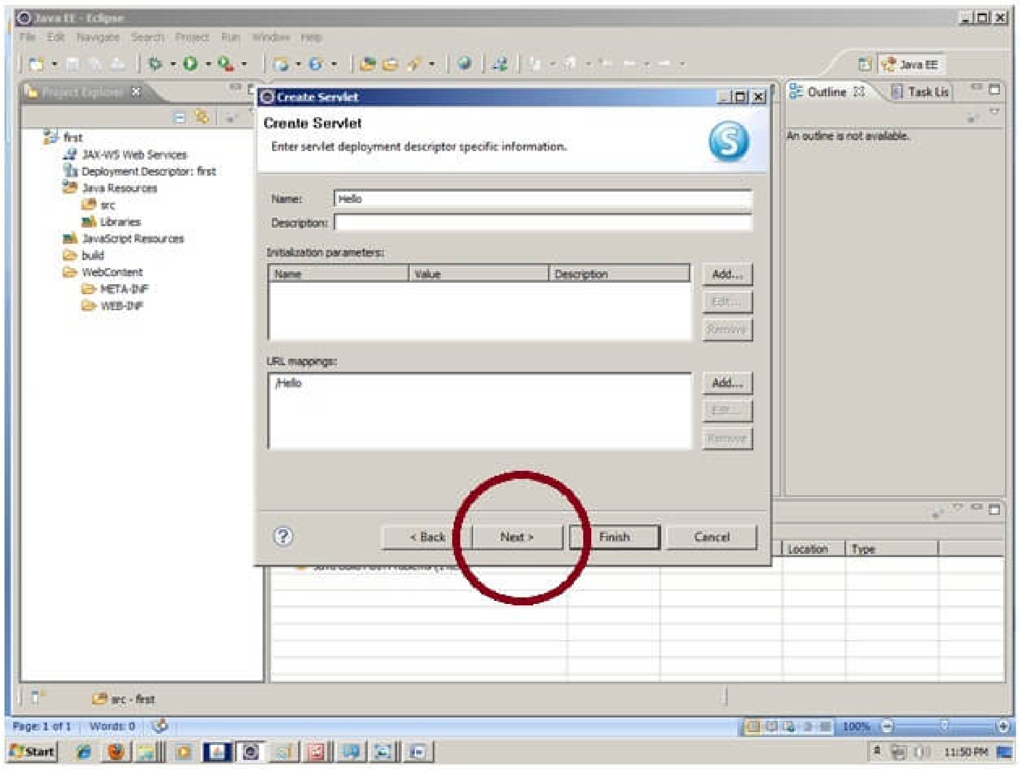


Create Servlet

| | |
|---|---|
| Project: | first |
| Source folder: | /first/src |
| Java package: | |
| Class name: | Hello |
| Superclass: | javax.servlet.http.HttpServlet |

0 errors, 1 warning, 0 others

Java Build Path Problems (1 item)



**Create Servlet**

Enter servlet deployment descriptor specific information.

Name: Hello

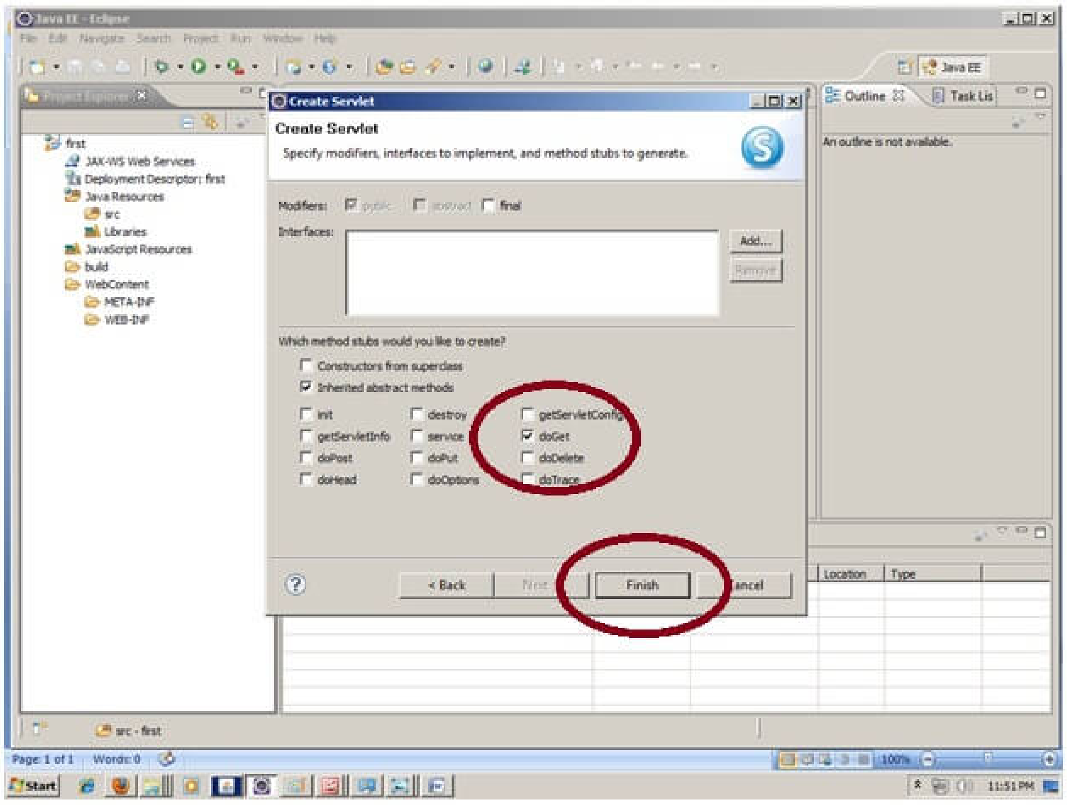
Description:

Initialization parameters:

| Name | Value | Description |
|---|---|---|

URL mappings:

/Hello



**Create Servlet**

Specify modifiers, interfaces to implement, and method stubs to generate.

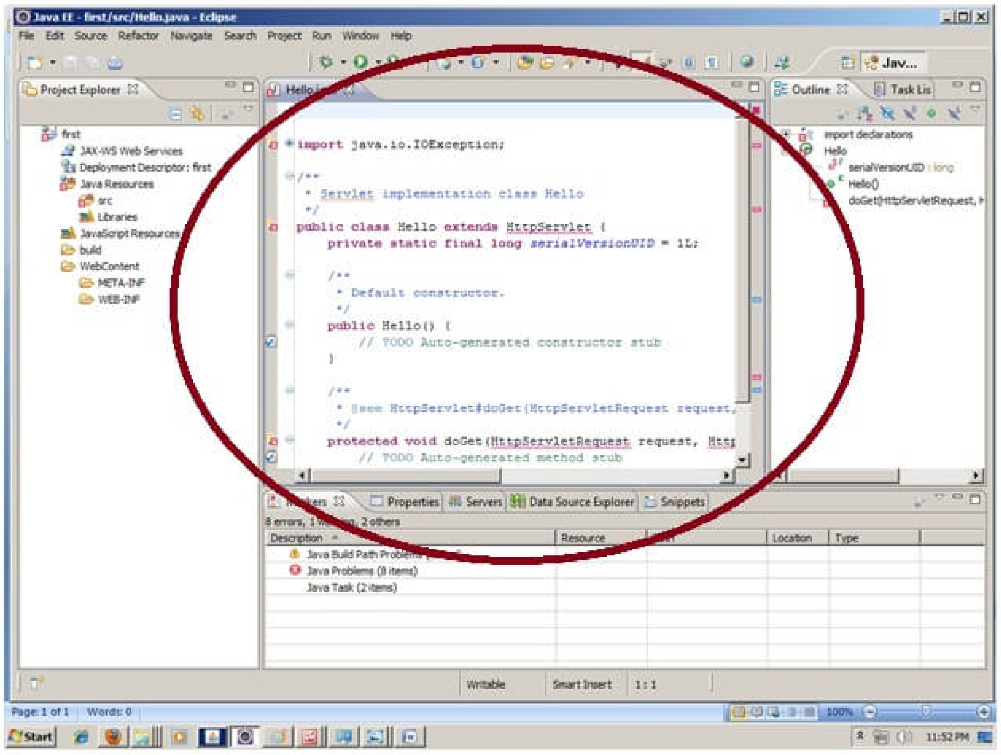
Modifiers: [✓ public] abstract final

Interfaces:

Which method stubs would you like to create?

- Constructors from superclass
- [✓ Inherited abstract methods]

| | | |
|---|---|---|
| init | destroy | getServletConfig |
| getServletInfo | service | [✓ doGet] |
| doPost | doPut | doDelete |
| doHead | doOptions | doTrace |

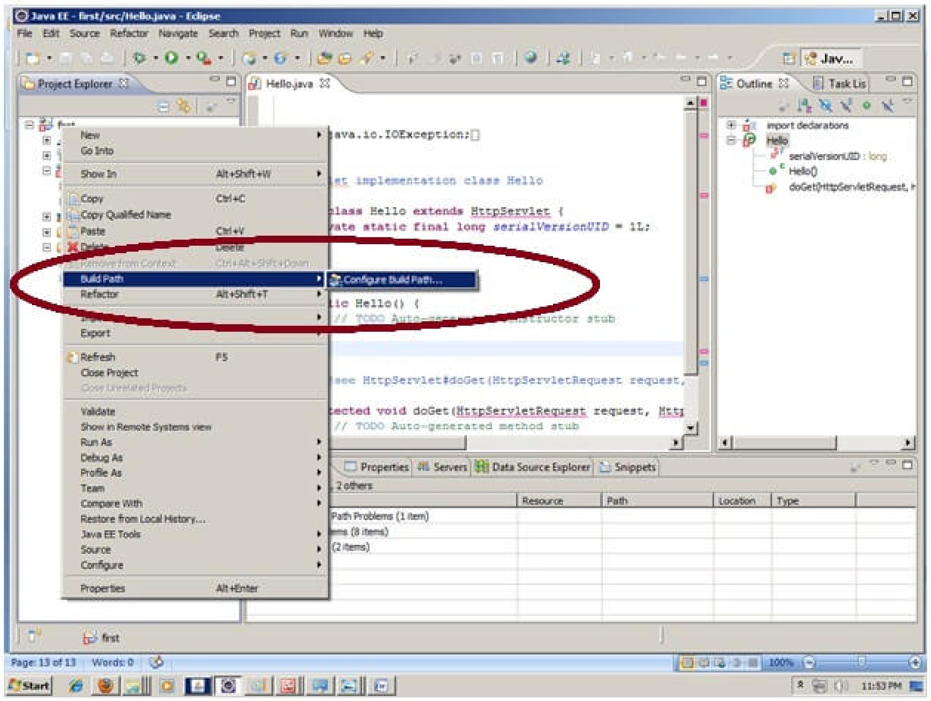


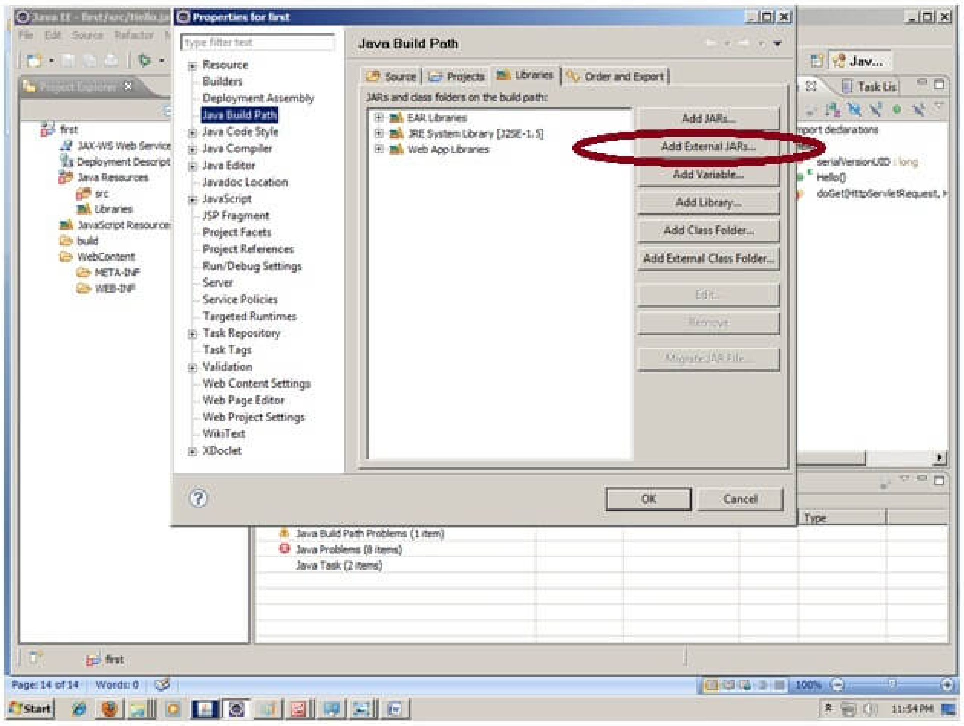
```
import java.io.IOException;

/**
 * Servlet implementation class Hello
 */
public class Hello extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * Default constructor.
     */
    public Hello() {
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request,
     */
    protected void doGet(HttpServletRequest request, Http
        // TODO Auto-generated method stub
```

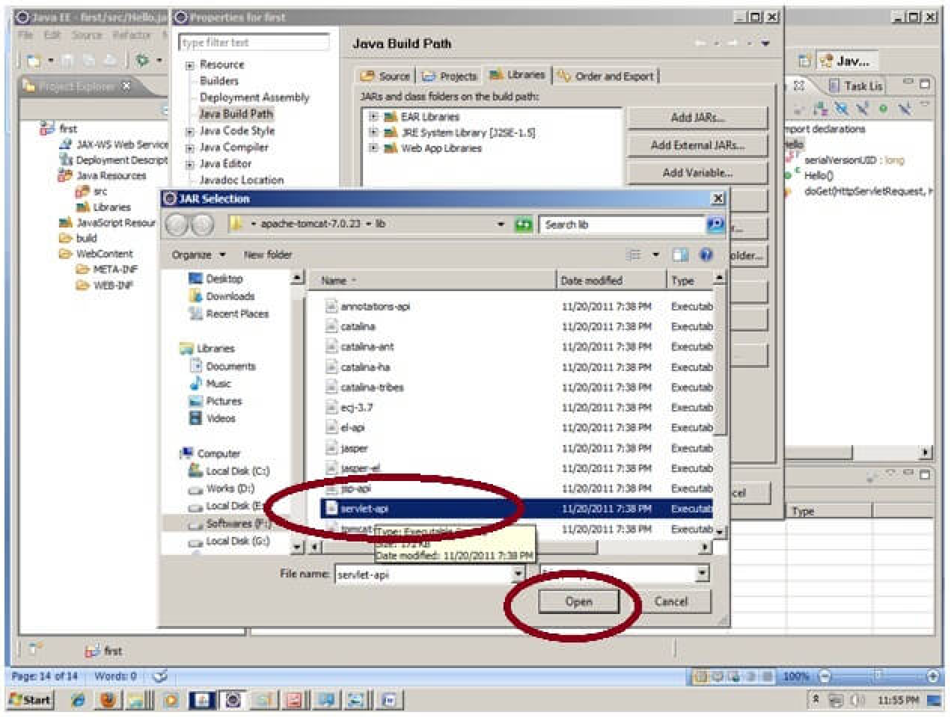
```
8 errors, 1 warning, 2 others
Description          Resource    Path    Location    Type
Java Build Path Problems
Java Problems (8 items)
Java Task (2 items)
```

3) add jar file in eclipse IDE:

For adding a jar file, **right click on your project -> Build Path -> Configure Build Path -> click on Libraries tab in Java Build Path -> click on Add External JARs button -> select the servlet-api.jar file under tomcat/lib -> ok.**

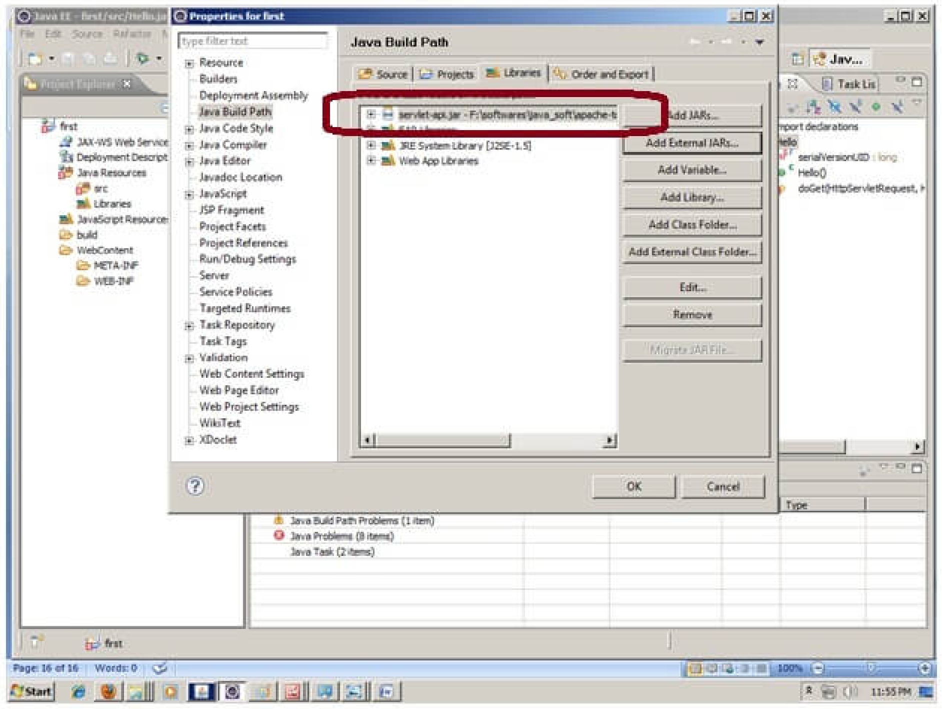

Java EE - first/src/Hello.java - Eclipse
Project Explorer
Hello.java
Outline
Task List
New
Go Into
Show In
Copy
Copy Qualified Name
Paste
Delete
Build Path
Configure Build Path...
Refactor
Export
Refresh
Close Project
Validate
Show in Remote Systems view
Run As
Debug As
Profile As
Team
Compare With
Restore from Local History...
Java EE Tools
Source
Configure
Properties
Properties
Servers
Data Source Explorer
Snippets

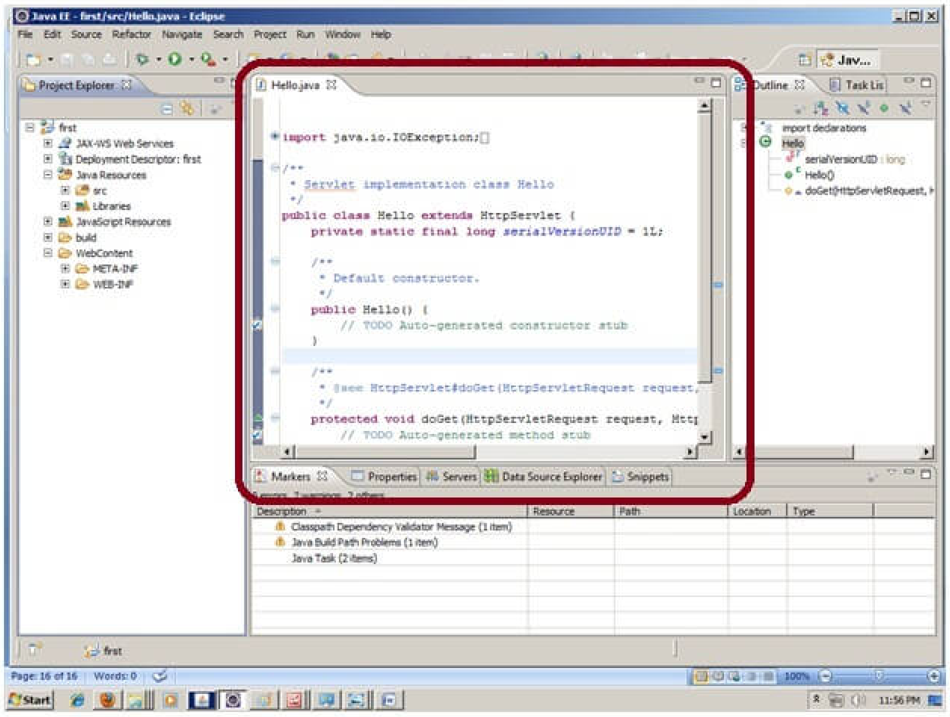

Properties for first
Java Build Path
Resource
Builders
Deployment Assembly
Java Build Path
Java Code Style
Java Compiler
Java Editor
Javadoc Location
JavaScript
JSP Fragment
Project Facets
Project References
Run/Debug Settings
Server
Service Policies
Targeted Runtimes
Task Repository
Task Tags
Validation
Web Content Settings
Web Page Editor
Web Project Settings
WikiText
XDoclet
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
EAR Libraries
JRE System Library [J2SE-1.5]
Web App Libraries
Add JARs...
Add External JARs...
Add Variable...
Add Library...
Add Class Folder...
Add External Class Folder...
Edit...
Remove
Migrate JAR File...
OK
Cancel



| Name | Date modified | Type |
|---|---|---|
| annotations-api | 11/20/2011 7:38 PM | Executab |
| catalina | 11/20/2011 7:38 PM | Executab |
| catalina-ant | 11/20/2011 7:38 PM | Executab |
| catalina-ha | 11/20/2011 7:38 PM | Executab |
| catalina-tribes | 11/20/2011 7:38 PM | Executab |
| ecj-3.7 | 11/20/2011 7:38 PM | Executab |
| el-api | 11/20/2011 7:38 PM | Executab |
| jasper | 11/20/2011 7:38 PM | Executab |
| jasper-el | 11/20/2011 7:38 PM | Executab |
| jsp-api | 11/20/2011 7:38 PM | Executab |
| servlet-api | 11/20/2011 7:38 PM | Executab |
| tomcat | 11/20/2011 7:38 PM | Executab |

File name: servlet-api



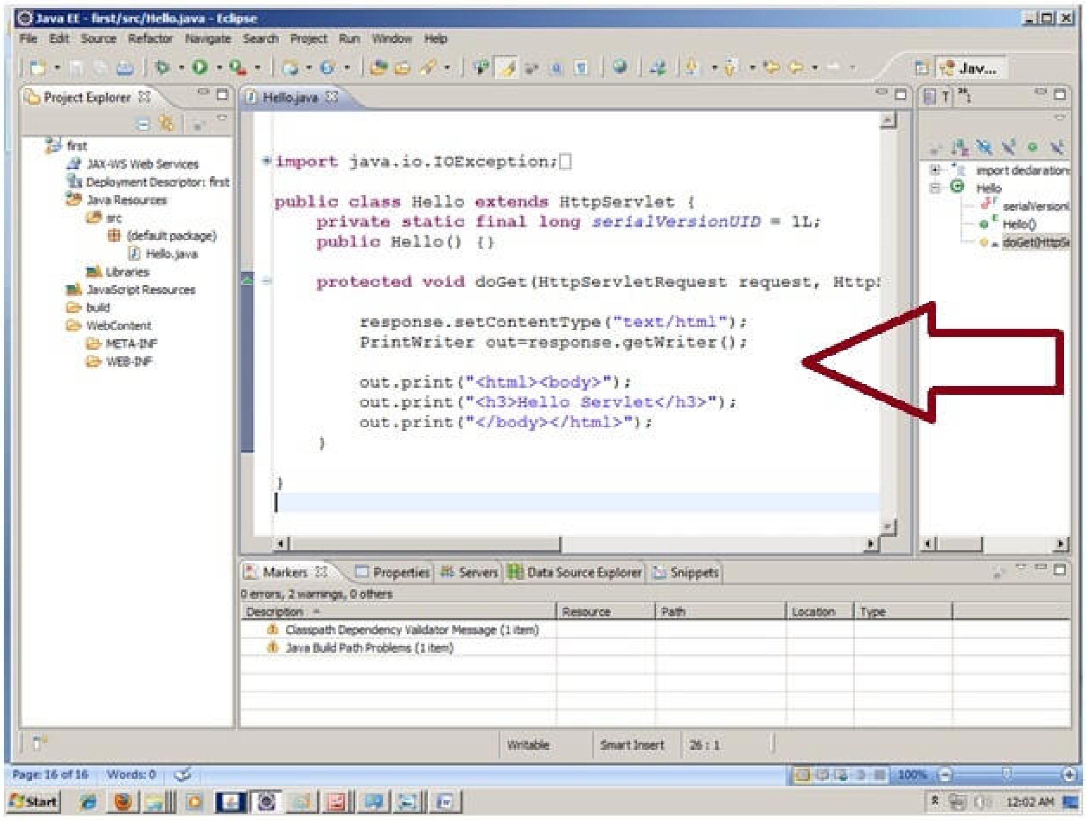


```
import java.io.IOException;

/**
 * Servlet implementation class Hello
 */
public class Hello extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * Default constructor.
     */
    public Hello() {
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request,
     */
    protected void doGet(HttpServletRequest request, Htt
        // TODO Auto-generated method stub
```

0 errors, 2 warnings, 2 others

| Description | Resource | Path | Location | Type |
|---|---|---|---|---|
| Classpath Dependency Validator Message (1 item) | | | | |
| Java Build Path Problems (1 item) | | | | |
| Java Task (2 items) | | | | |

Now servlet has been created, Let's write the first servlet code.



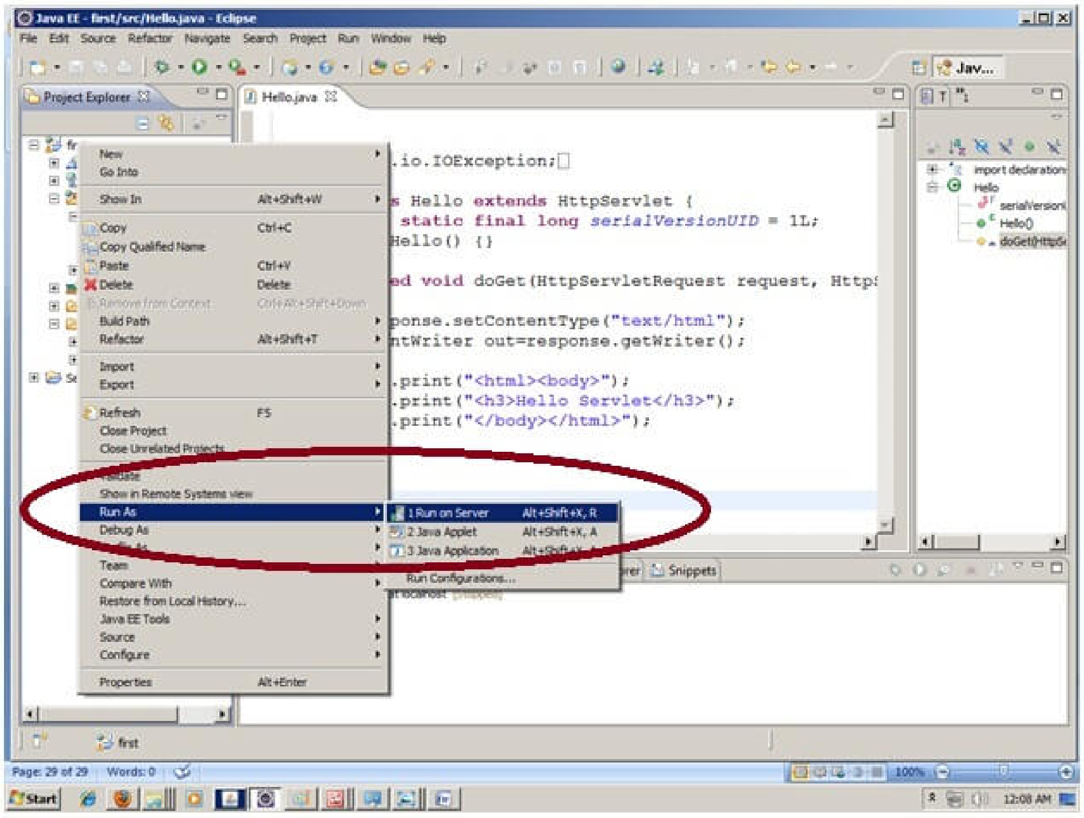
```
import java.io.IOException;

public class Hello extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public Hello() {}

    protected void doGet(HttpServletRequest request, Http

        response.setContentType("text/html");
        PrintWriter out=response.getWriter();

        out.print("<html><body>");
        out.print("<h3>Hello Servlet</h3>");
        out.print("</body></html>");
    }

}
```

0 errors, 2 warnings, 0 others

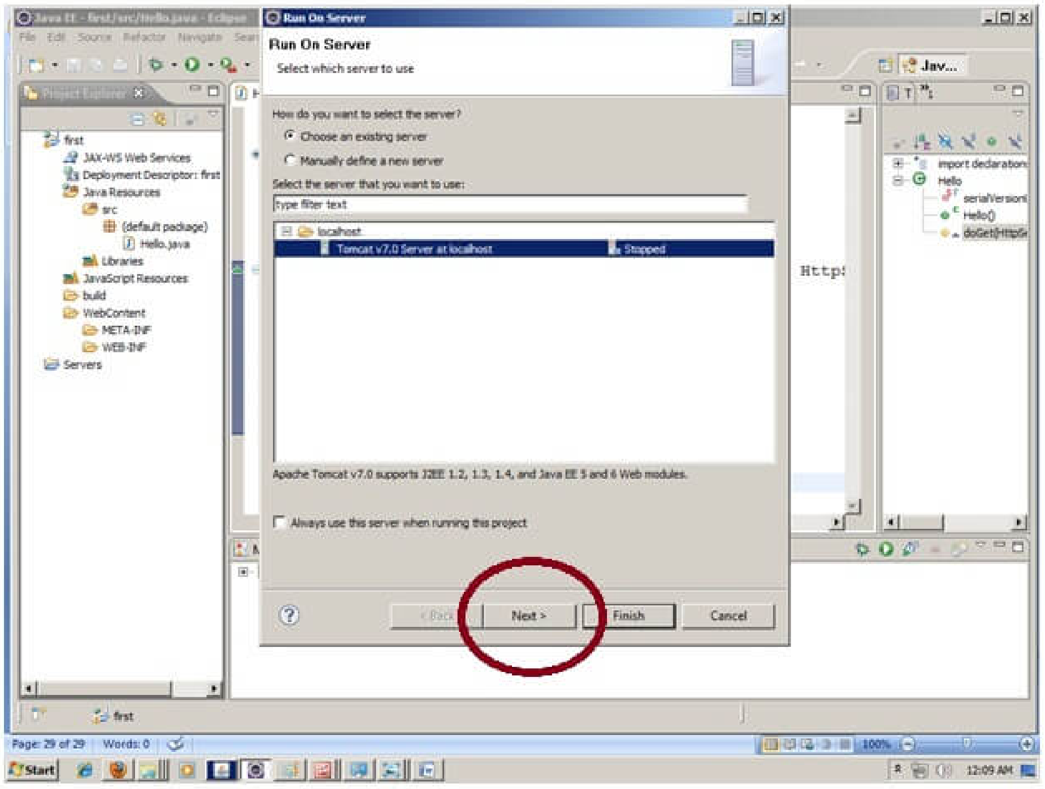
| Description | Resource | Path | Location | Type |
|---|---|---|---|---|
| Classpath Dependency Validator Message (1 item) | | | | |
| Java Build Path Problems (1 item) | | | | |

4) Start the server and deploy the project:

For starting the server and deploying the project in one step, **Right click on your project -> Run As -> Run on Server -> choose tomcat server -> next -> addAll -> finish.**



```
.io.IOException;

s Hello extends HttpServlet {
 static final long serialVersionUID = 1L;
Hello() {}

ed void doGet(HttpServletRequest request, Http

ponse.setContentType("text/html");
ntWriter out=response.getWriter();

.print("<html><body>");
.print("<h3>Hello Servlet</h3>");
.print("</body></html>");
```



Run On Server

Select which server to use

How do you want to select the server?

(● Choose an existing server) ○ Manually define a new server

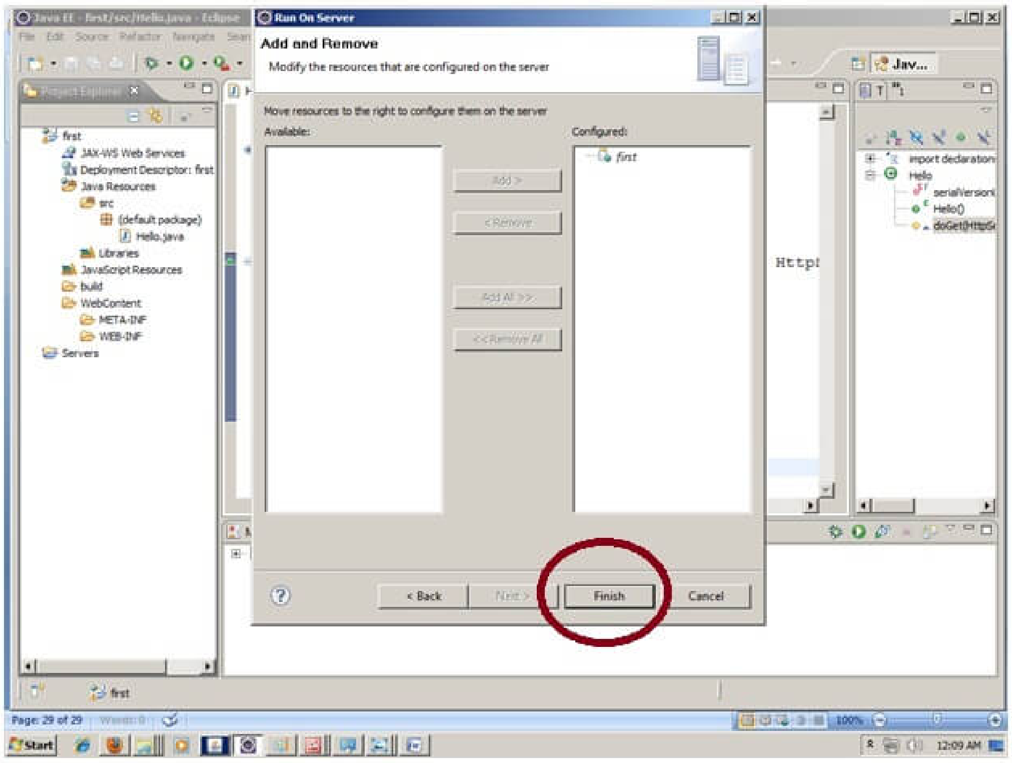
Select the server that you want to use:

type filter text

localhost
Tomcat v7.0 Server at localhost — Stopped

Apache Tomcat v7.0 supports J2EE 1.2, 1.3, 1.4, and Java EE 5 and 6 Web modules.

☐ Always use this server when running this project



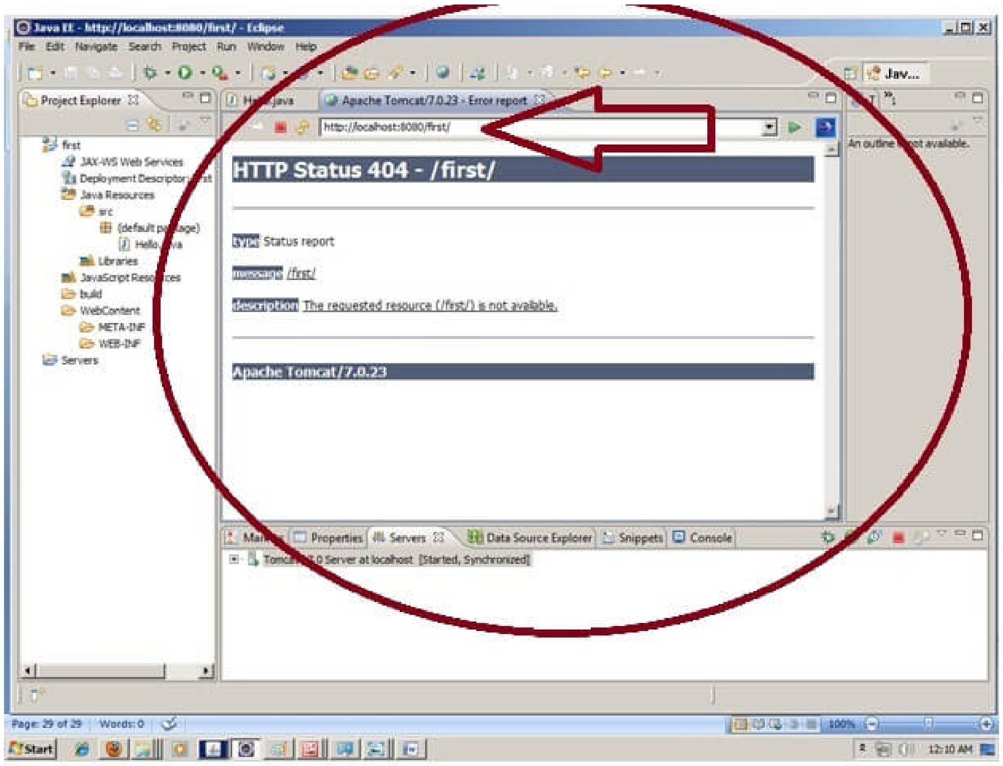
Add and Remove

Modify the resources that are configured on the server

Move resources to the right to configure them on the server

Available:

Configured:
first


HTTP Status 404 - /first/

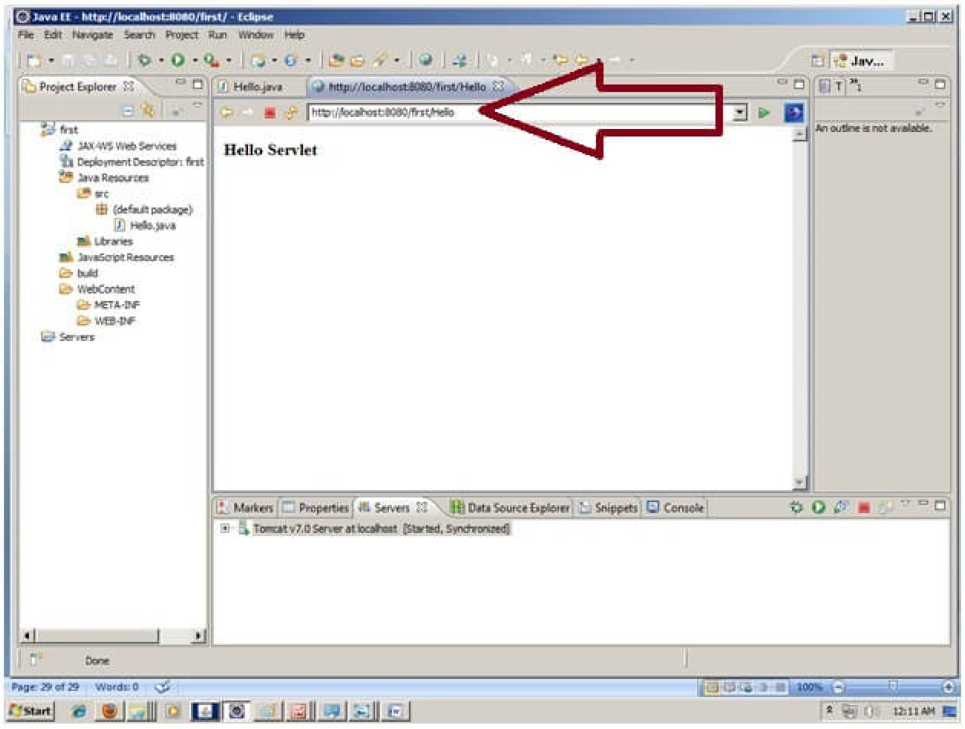
type Status report

message /first/

description The requested resource (/first/) is not available.

Apache Tomcat/7.0.23

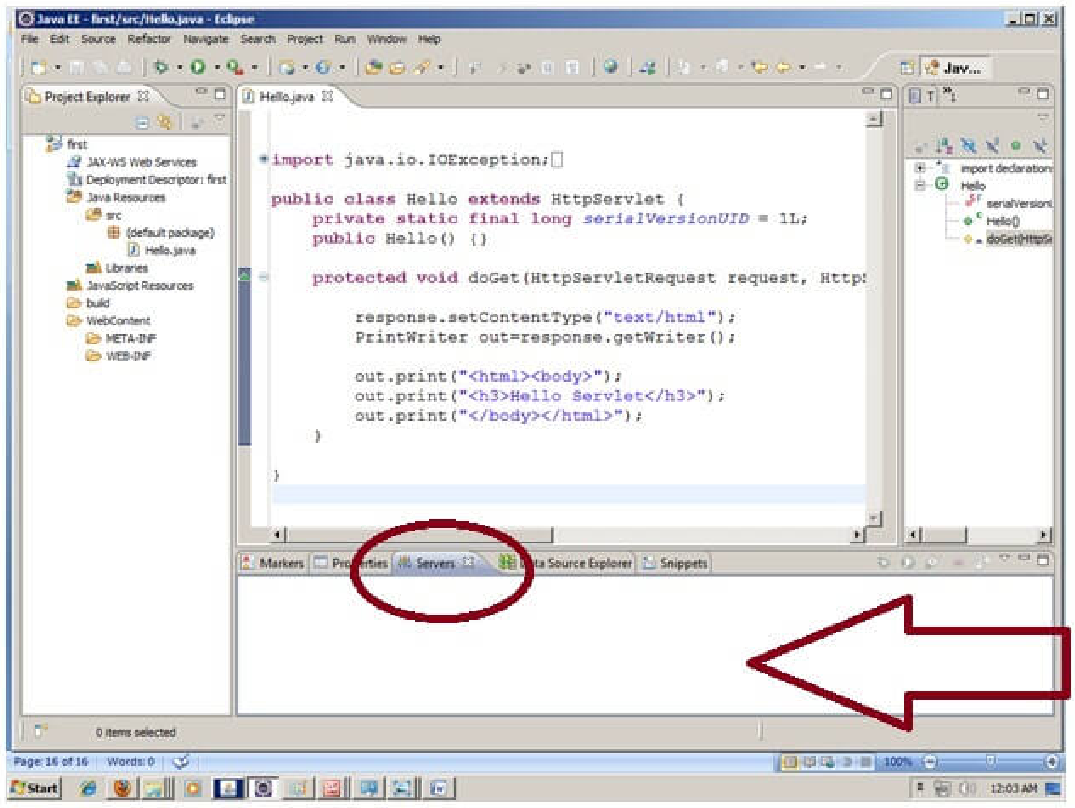
Now tomcat server has been started and project is deployed. To access the servlet write the url pattern name in the URL bar of the browser. In this case Hello then enter.


Hello Servlet

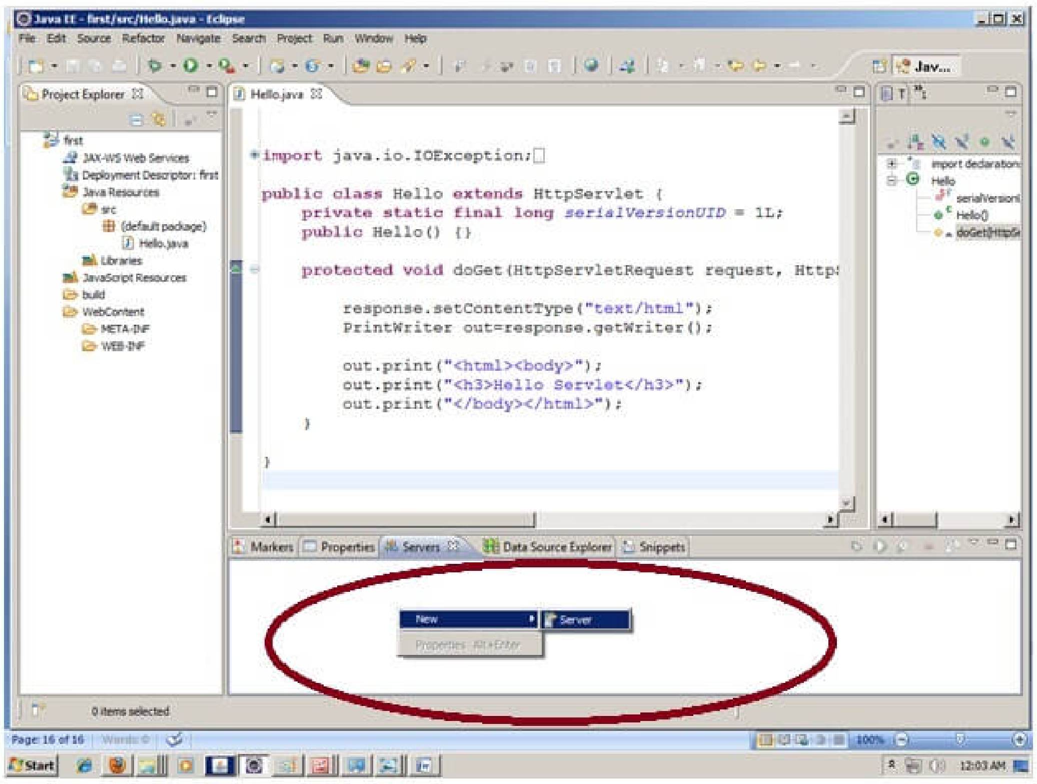
## How to configure tomcat server in Eclipse ? (One time Requirement)

If you are using **Eclipse IDE first time**, you need to configure the tomcat server First.

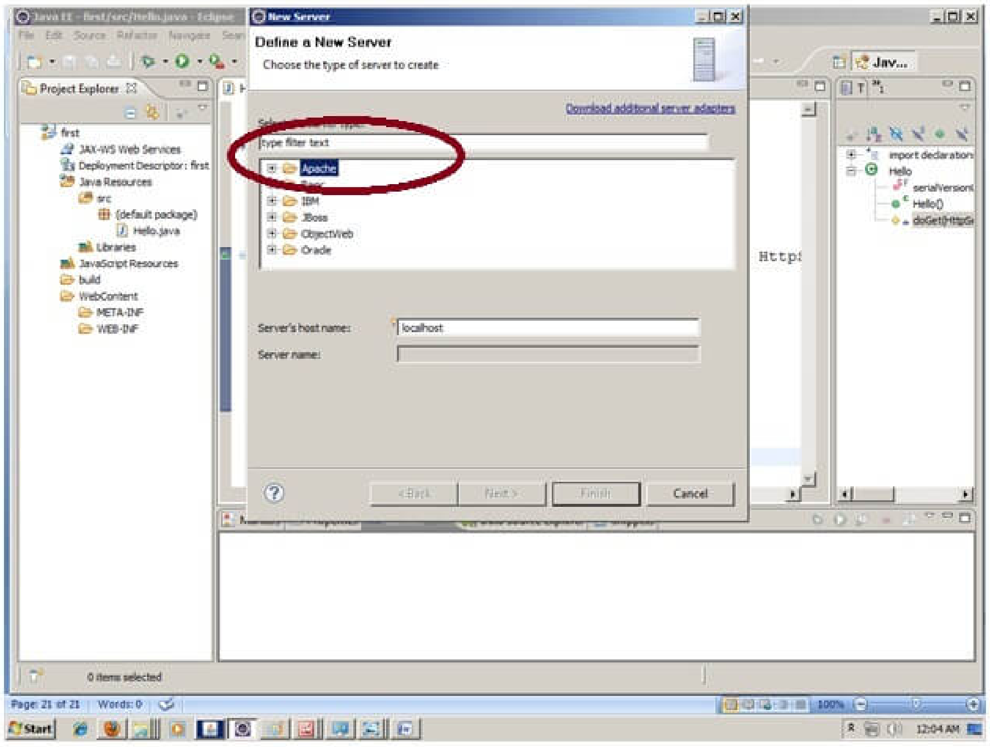
For configuring the tomcat server in eclipse IDE, **click on servers tab at the bottom side of the IDE -> right click on blank area -> New -> Servers -> choose tomcat then its version -> next -> click on Browse button -> select the apache tomcat root folder previous to bin -> next -> addAll -> Finish.**



```
import java.io.IOException;

public class Hello extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public Hello() {}

    protected void doGet(HttpServletRequest request, Http

        response.setContentType("text/html");
        PrintWriter out=response.getWriter();

        out.print("<html><body>");
        out.print("<h3>Hello Servlet</h3>");
        out.print("</body></html>");
    }

}
```

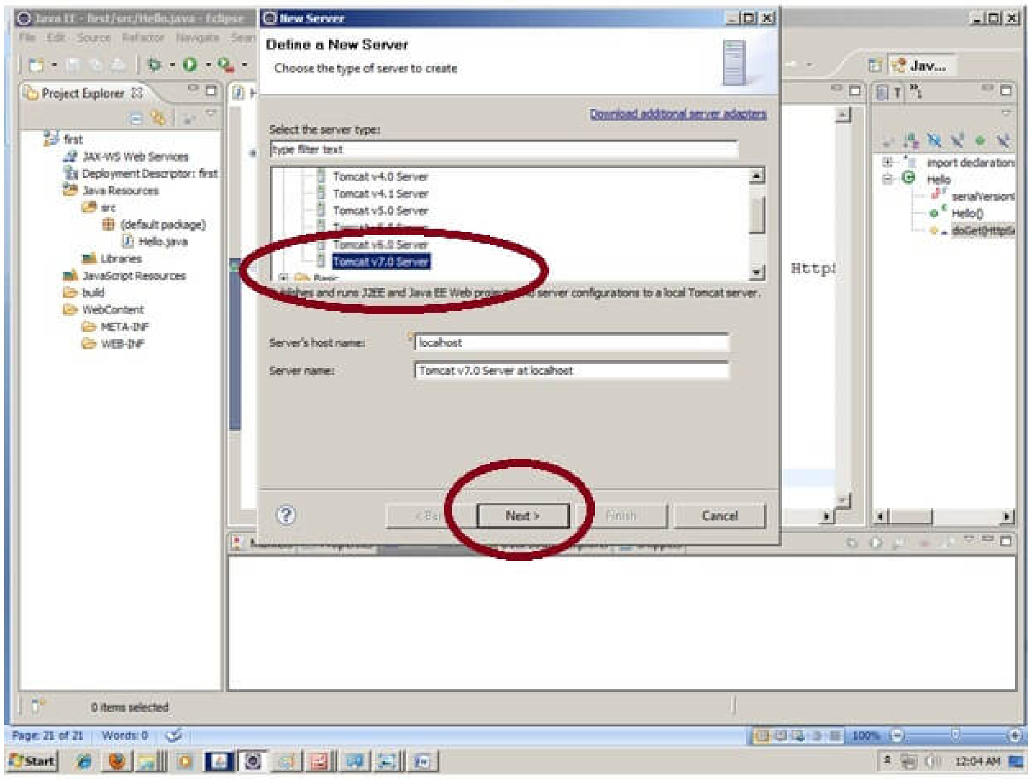


```
import java.io.IOException;

public class Hello extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public Hello() {}

    protected void doGet(HttpServletRequest request, Http

        response.setContentType("text/html");
        PrintWriter out=response.getWriter();

        out.print("<html><body>");
        out.print("<h3>Hello Servlet</h3>");
        out.print("</body></html>");
    }

}
```



Server's host name: localhost

Server name:



New Server

Define a New Server

Choose the type of server to create

Download additional server adapters

Select the server type:

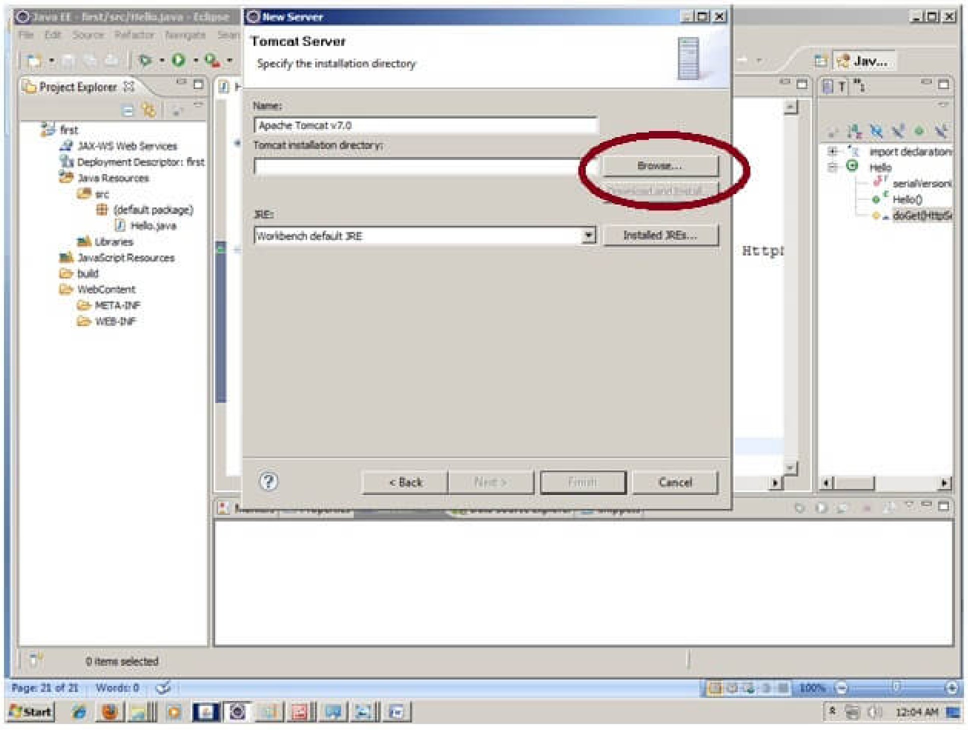
type filter text

Tomcat v4.0 Server
Tomcat v4.1 Server
Tomcat v5.0 Server
Tomcat v5.5 Server
Tomcat v6.0 Server
Tomcat v7.0 Server
Basic

Publishes and runs J2EE and Java EE Web projects and server configurations to a local Tomcat server.

Server's host name: localhost

Server name: Tomcat v7.0 Server at localhost



New Server

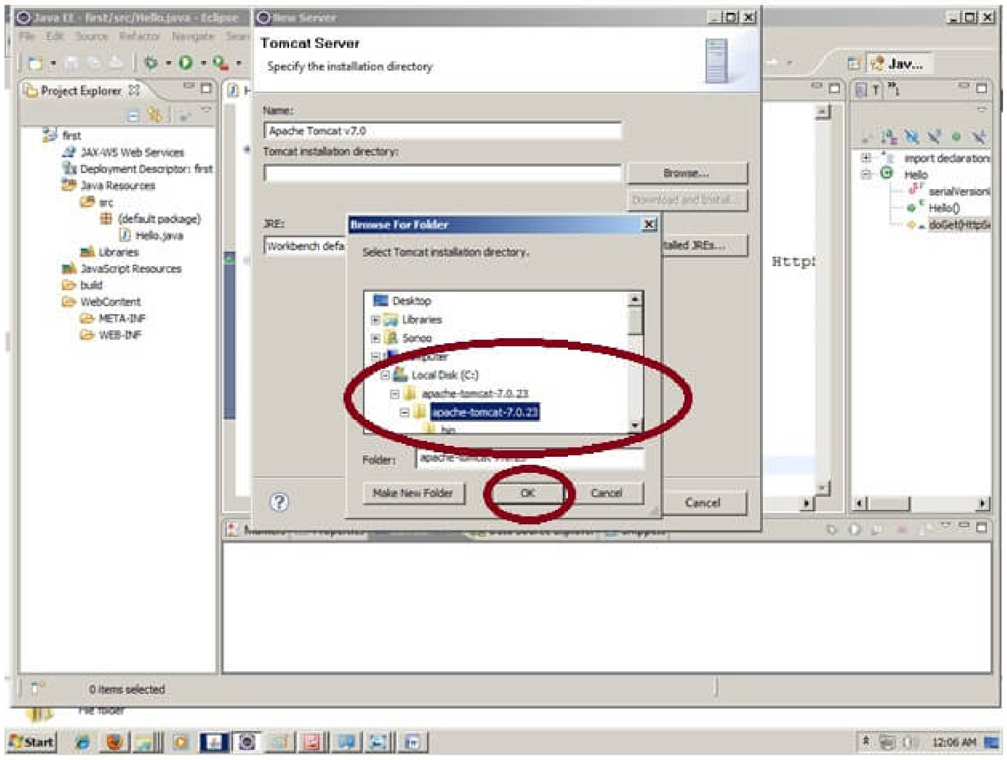
Tomcat Server

Specify the installation directory

Name: Apache Tomcat v7.0

Tomcat installation directory:

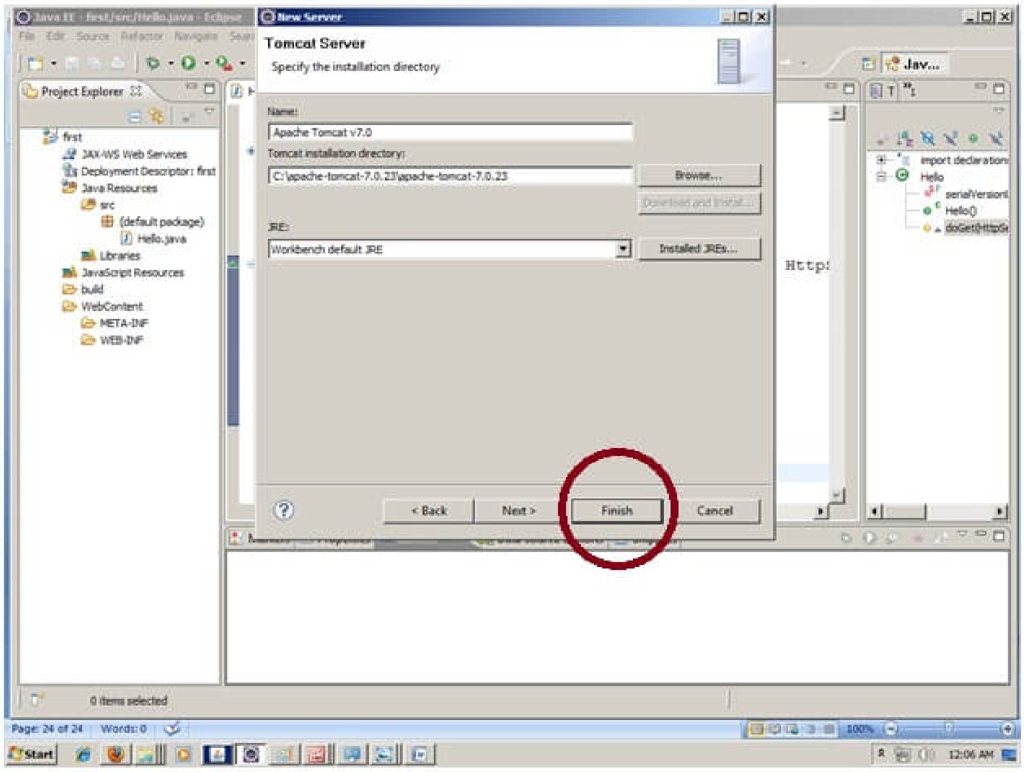
JRE: Workbench default JRE


Tomcat Server
Specify the installation directory
Browse For Folder
Select Tomcat installation directory.
Make New Folder
OK
Cancel

Name: Apache Tomcat v7.0

Tomcat installation directory:

Local Disk (C:)
apache-tomcat-7.0.23
apache-tomcat-7.0.23

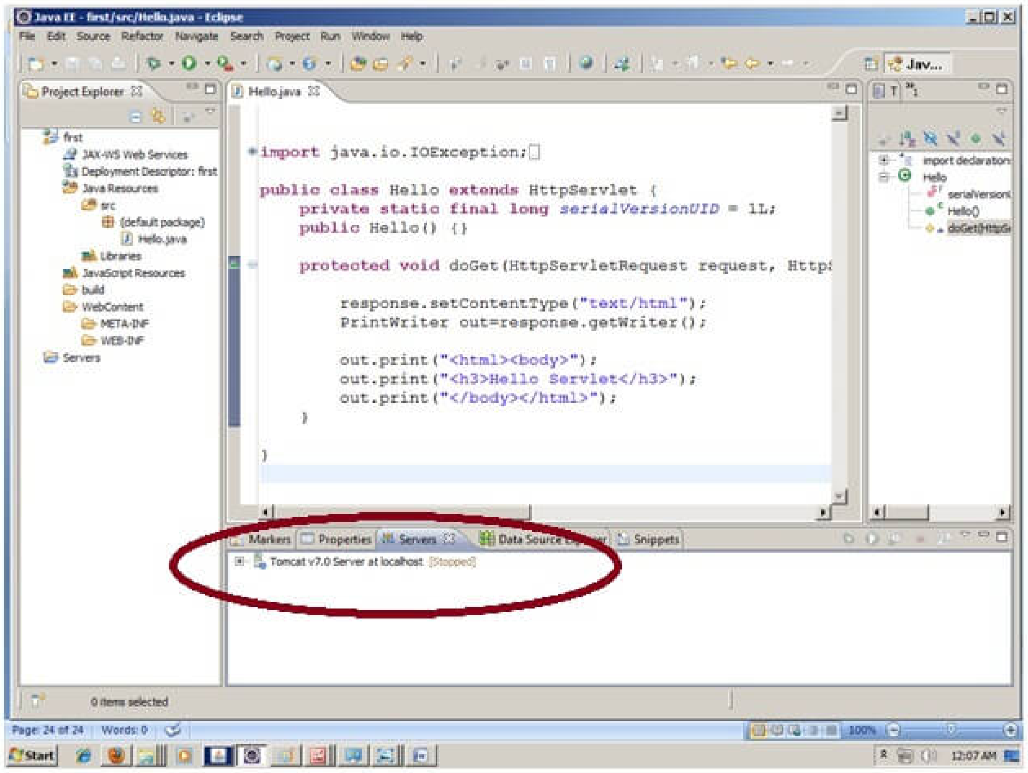


Name: Apache Tomcat v7.0

Tomcat installation directory: C:\apache-tomcat-7.0.23\apache-tomcat-7.0.23

JRE: Workbench default JRE

Now tomcat7 server has been configured in eclipse IDE.



```
import java.io.IOException;

public class Hello extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public Hello() {}

    protected void doGet(HttpServletRequest request, Http

        response.setContentType("text/html");
        PrintWriter out=response.getWriter();

        out.print("<html><body>");
        out.print("<h3>Hello Servlet</h3>");
        out.print("</body></html>");
    }

}
```

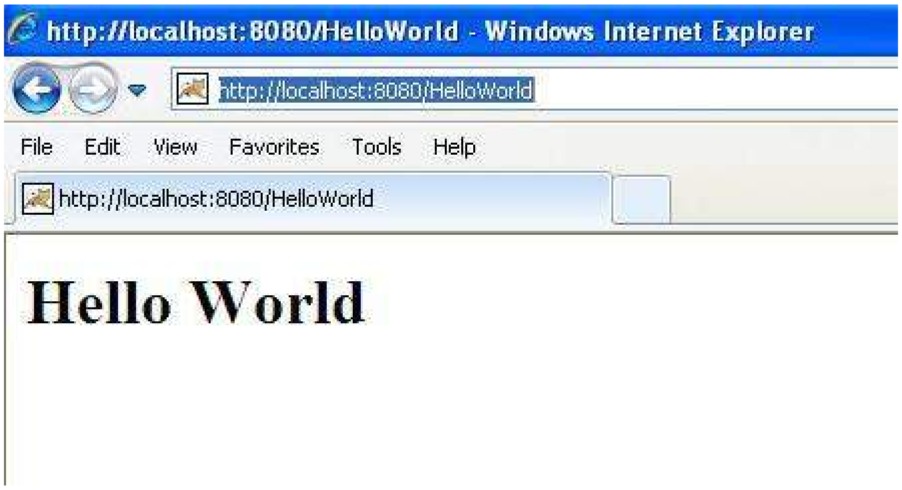
Tomcat v7.0 Server at localhost [Stopped]

## Sample Code:

```
import java.io.*;

import javax.servlet.*;

import javax.servlet.http.*;


// Extend HttpServlet class

public class HelloWorld extends HttpServlet {


  private String message;


  public void init() throws ServletException

  {
      // Do required initialization
      message = "Hello World";
  }

  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
            throws ServletException, IOException
  {
      // Set response content type
      response.setContentType("text/html");

      // Actual logic goes here.
      PrintWriter out = response.getWriter();
      out.println("<h1>" + message + "</h1>");
  }

  public void destroy()
  {
      // do nothing.
  }
<servlet>
<servlet-name>HelloWorld</servlet-name>
<servlet-class>HelloWorld</servlet-class>
</servlet>
```

```
<servlet-mapping>
<servlet-name>HelloWorld</servlet-name>
<url-pattern>/HelloWorld</url-pattern>
</servlet-mapping>
```

**Output:**


http://localhost:8080/HelloWorld - Windows Internet Explorer
http://localhost:8080/HelloWorld
File Edit View Favorites Tools Help
http://localhost:8080/HelloWorld

Hello World

**Viva Voce Questions:**

1. What is the web application and what is the difference between Get and Post Request?

2. What id Servlets?

3. What information is received by the web server if we request for a Servlet?

4. What is the difference between ServletConfig and ServletContext interface?

5. Define ‘init’ and ‘destroy’ methods in servlets.

## Experiment 10:

**Aim:** Create JSP application for addition, multiplication and division.

## Objective: -

• To learn about Java Server Pages JSP.
• To learn about various JSP tags.

## Input/Apparatus Used:

Hardware Requirements: - Minimum 384MB RAM, 100 GB hard Disk, processor with 2.1 MHz
Software Requirements: - Eclipse, NetBeans, IntelliJ, etc.

## Reading Material:

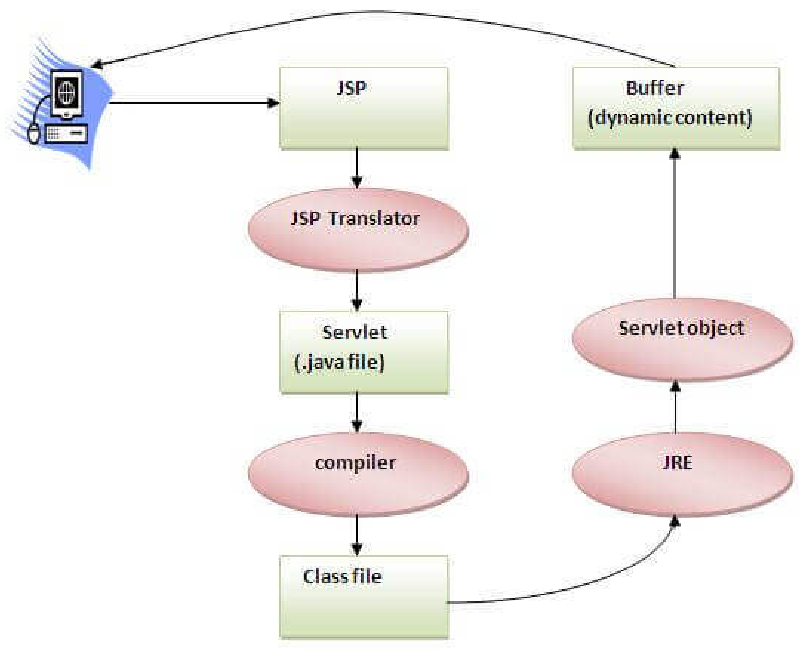
**JSP** technology is used to create web application just like Servlet technology. It can be thought of as an extension to Servlet because it provides more functionality than servlet such as expression language, JSTL, etc.

A JSP page consists of HTML tags and JSP tags. The JSP pages are easier to maintain than Servlet because we can separate designing and development. It provides some additional features such as Expression Language, Custom Tags, etc.

### The Lifecycle of a JSP Page

The JSP pages follow these phases:

- Translation of JSP Page
- Compilation of JSP Page
- Classloading (the classloader loads class file)
- Instantiation (Object of the Generated Servlet is created).
- Initialization ( the container invokes jspInit() method).
- Request processing ( the container invokes _jspService() method).
- Destroy ( the container invokes jspDestroy() method).



As depicted in the above diagram, JSP page is translated into Servlet by the help of JSP translator. The JSP translator is a part of the web server which is responsible for translating the JSP page into Servlet. After that, Servlet page is compiled by the compiler and gets converted into the class file. Moreover, all the processes that happen in Servlet are performed on JSP later like initialization, committing response to the browser and destroy.

## Procedure/Algorithm/Pseudocode

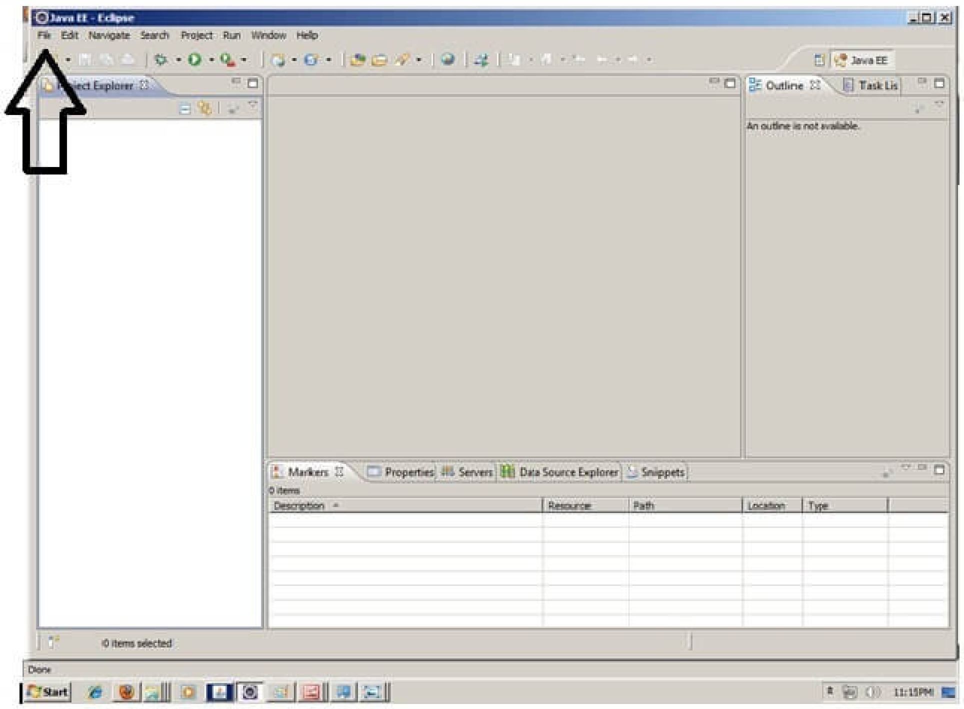
### Creating JSP in Eclipse IDE with Tomcat server

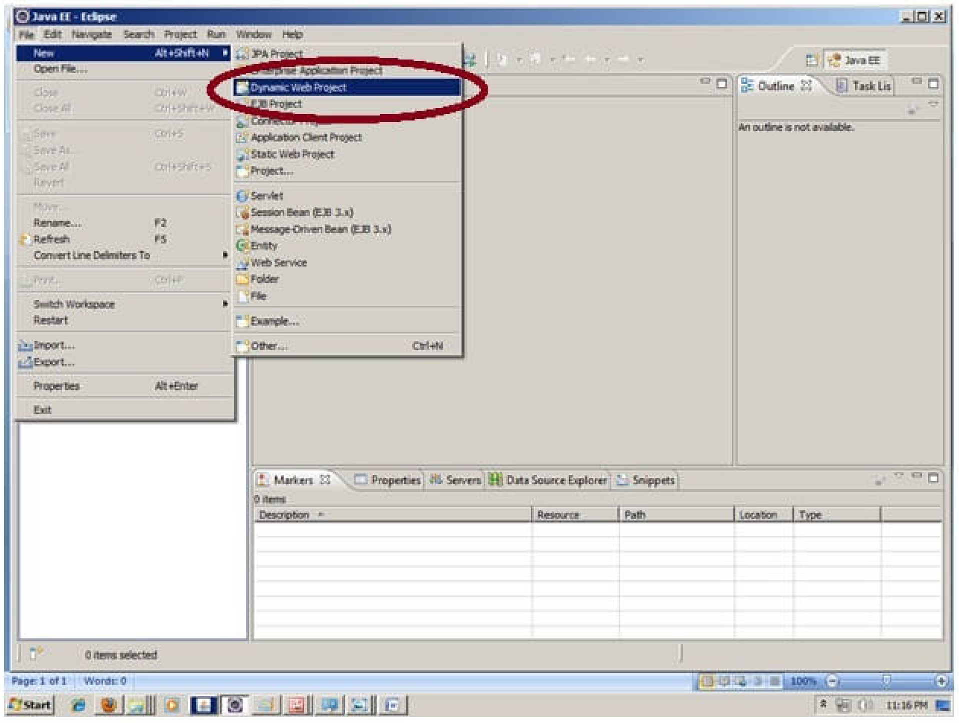
1. Creating JSP in Eclipse IDE with Tomcat
    1. Create a Dynamic web project
    2. create a jsp.
    3. start tomcat server and deploy the project

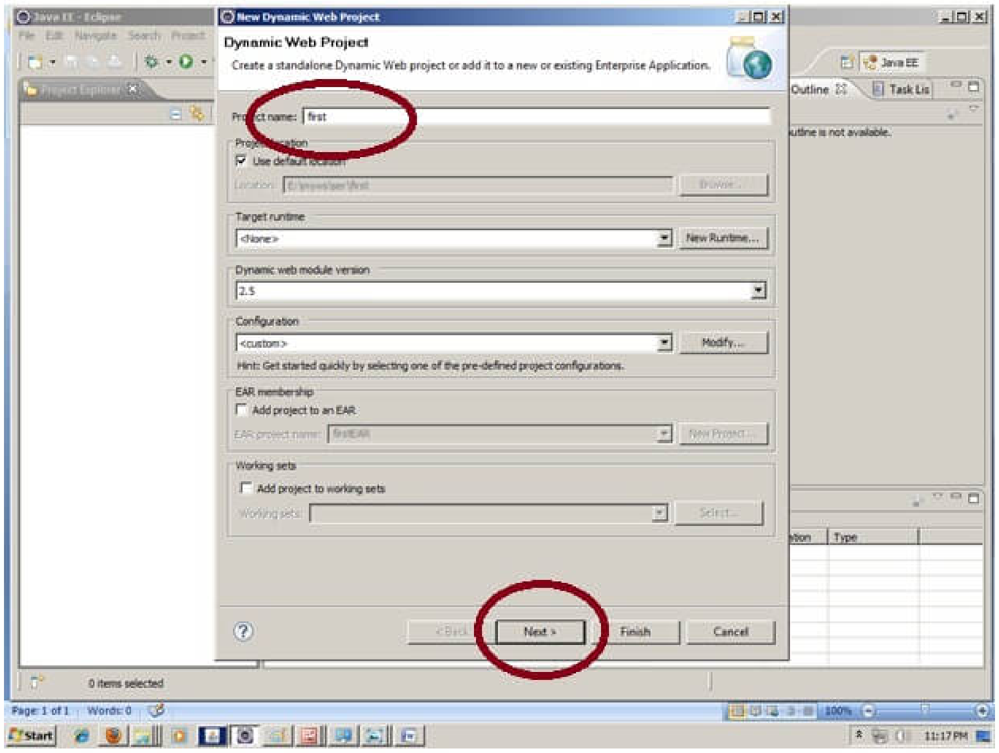
- Create a Dynamic web project
- create a jsp
- start tomcat server and deploy the project

1) Create the dynamic web project

For creating a dynamic web project click on File Menu -> New -> dynamic web project -> write your project name e.g. first -> Finish.






New Dynamic Web Project

Dynamic Web Project

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name: first

Project location

☑ Use default location

Location: E:\myws\ser\first

Target runtime: <None>

Dynamic web module version: 2.5

Configuration: <custom>

Hint: Get started quickly by selecting one of the pre-defined project configurations.

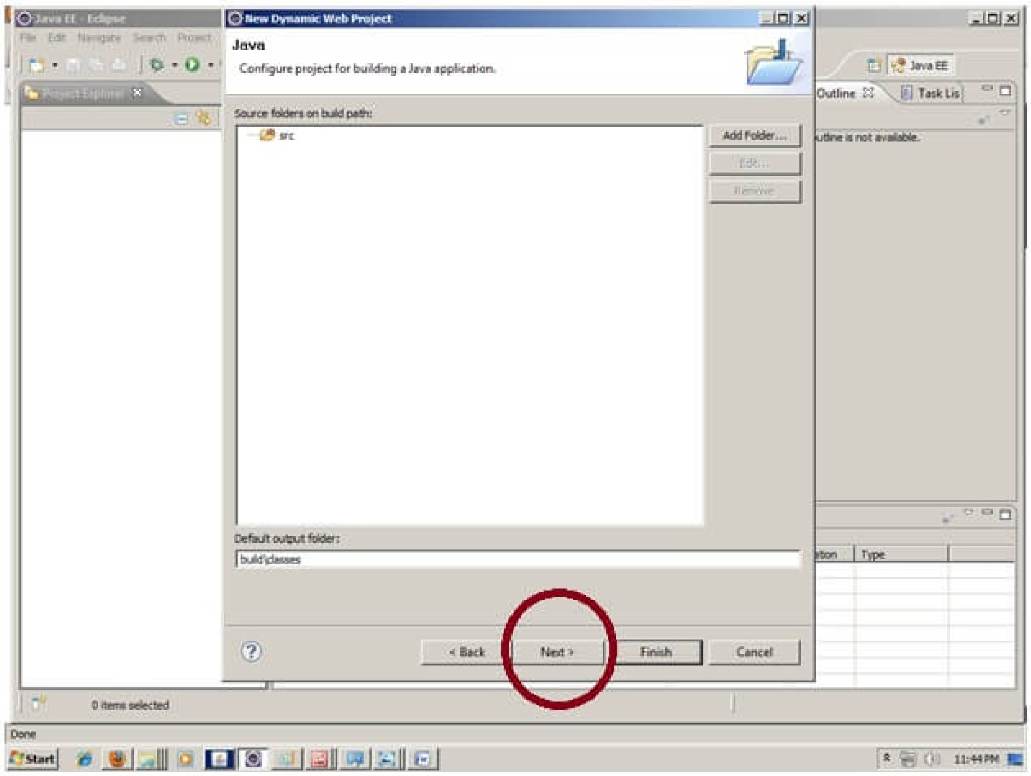
EAR membership

☐ Add project to an EAR

EAR project name: firstEAR

Working sets

☐ Add project to working sets


New Dynamic Web Project

Java

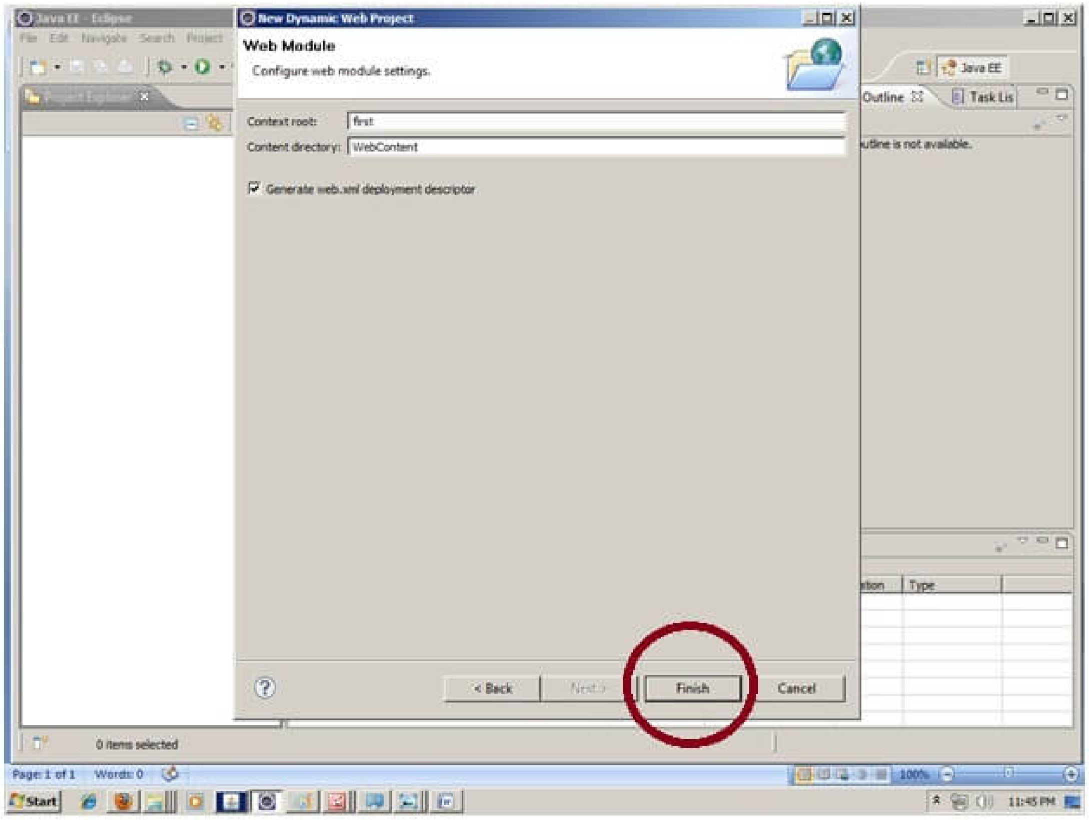
Configure project for building a Java application.

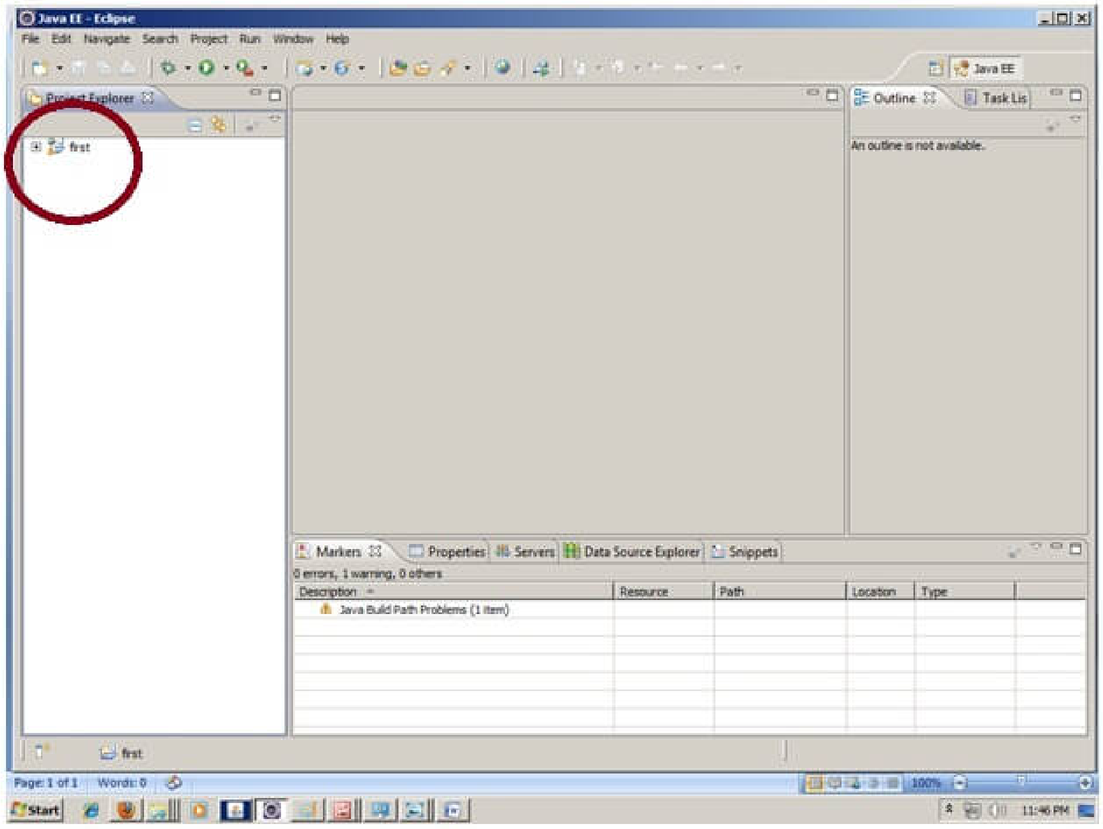
Source folders on build path:

src

Default output folder:

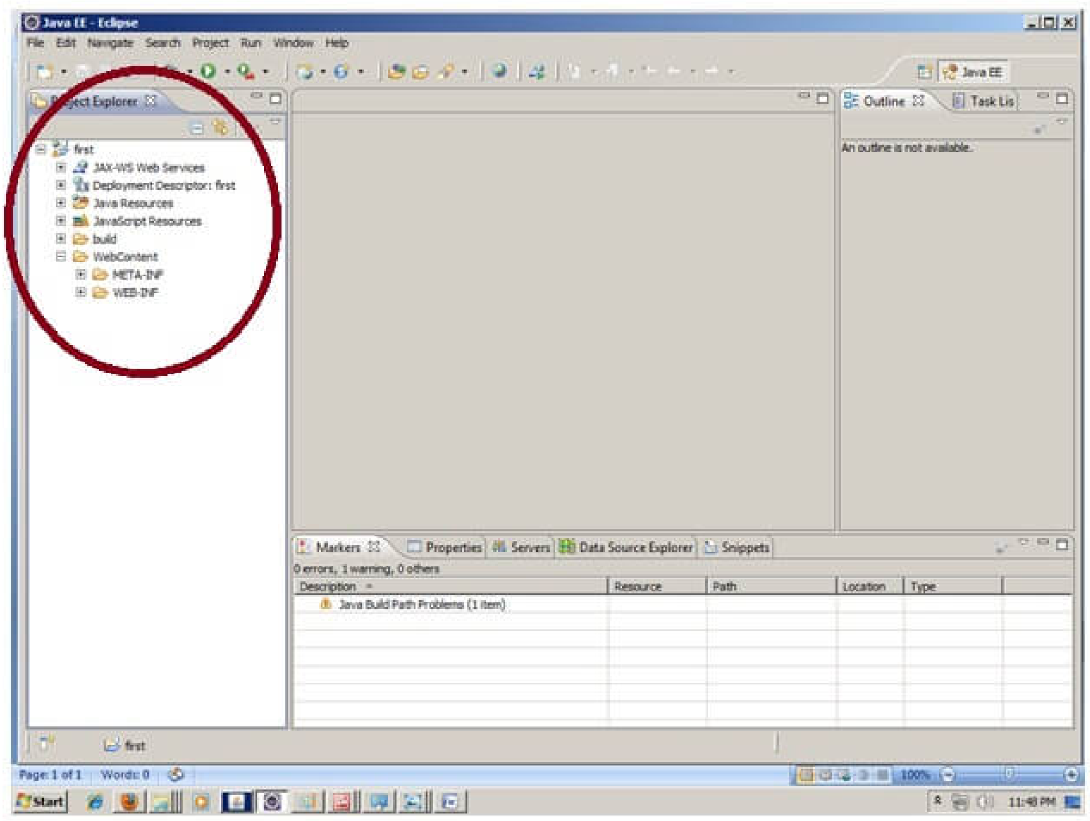
build\classes


New Dynamic Web Project
Web Module
Configure web module settings.
Context root: first
Content directory: WebContent
Generate web.xml deployment descriptor
< Back
Next >
Finish
Cancel
0 items selected


Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Project Explorer
first
Outline
Task List
An outline is not available.
Markers
Properties
Servers
Data Source Explorer
Snippets
0 errors, 1 warning, 0 others
Description
Resource
Path
Location
Type
Java Build Path Problems (1 item)

2) Create the JSP file in eclipse IDE

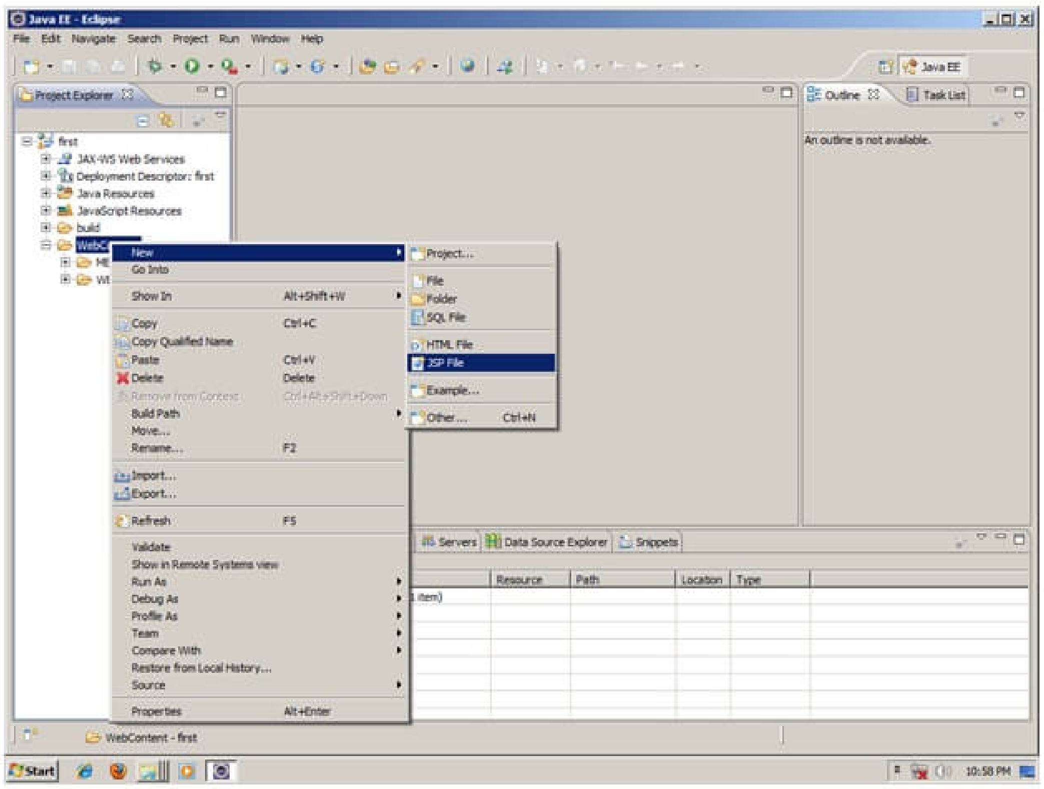
For creating a jsp file explore the project by clicking the + icon -> right click on WebContent -> New -> jsp -> write your jsp file name e.g. index -> next -> Finish.



An outline is not available.

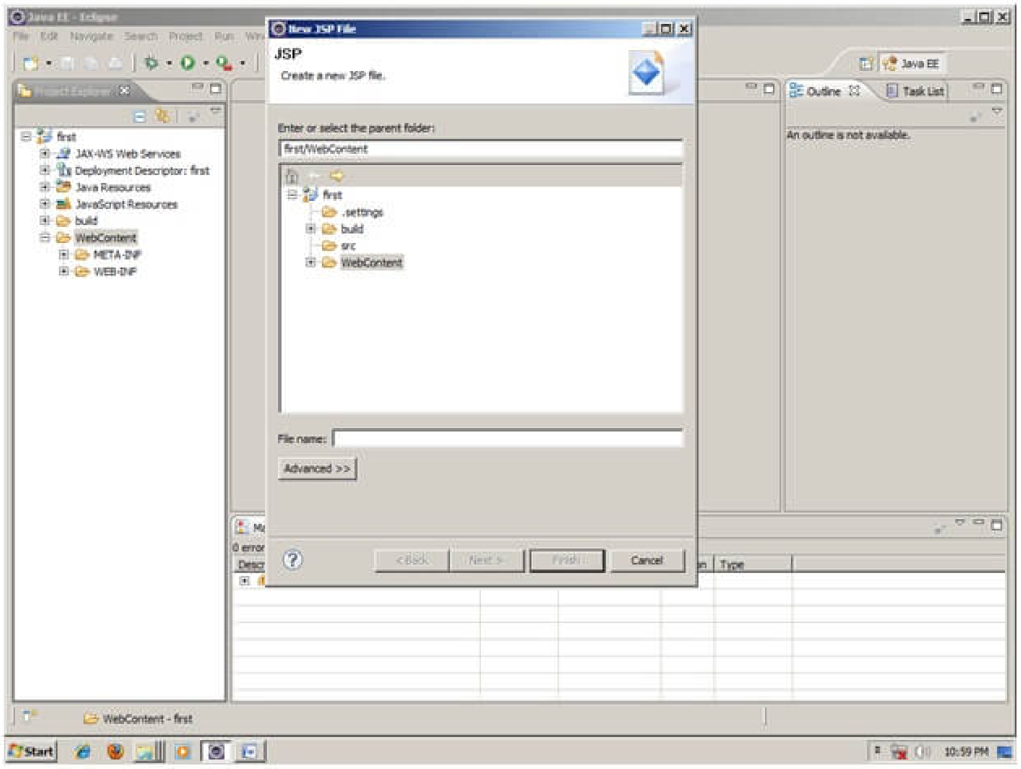
0 errors, 1 warning, 0 others

| Description | Resource | Path | Location | Type |
|---|---|---|---|---|
| Java Build Path Problems (1 item) | | | | |



An outline is not available.

WebContent - first

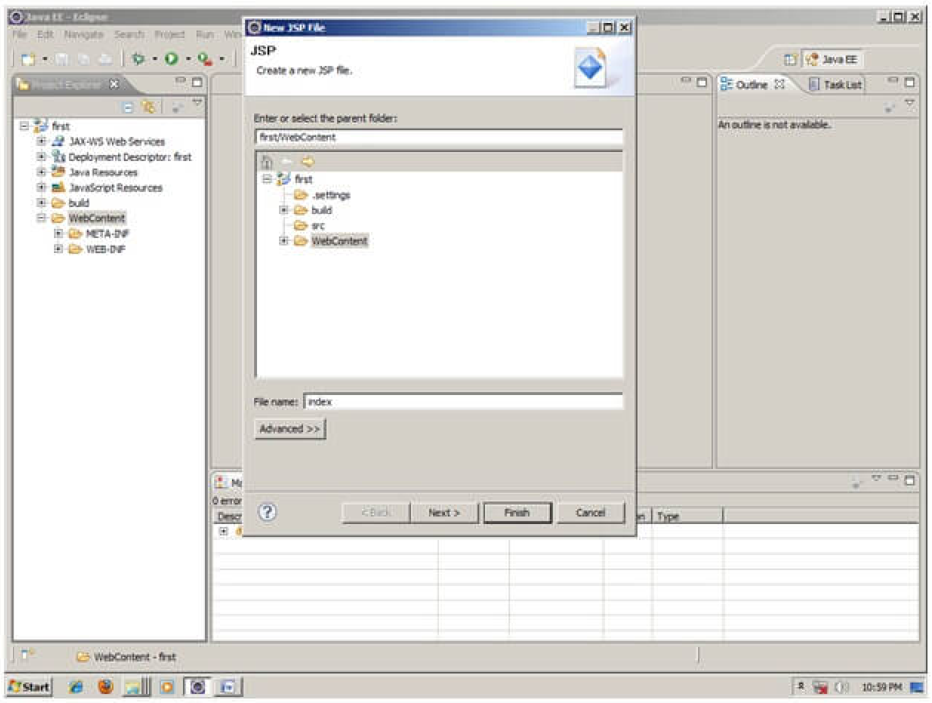


Enter or select the parent folder: first/WebContent

File name:

An outline is not available.

WebContent - first



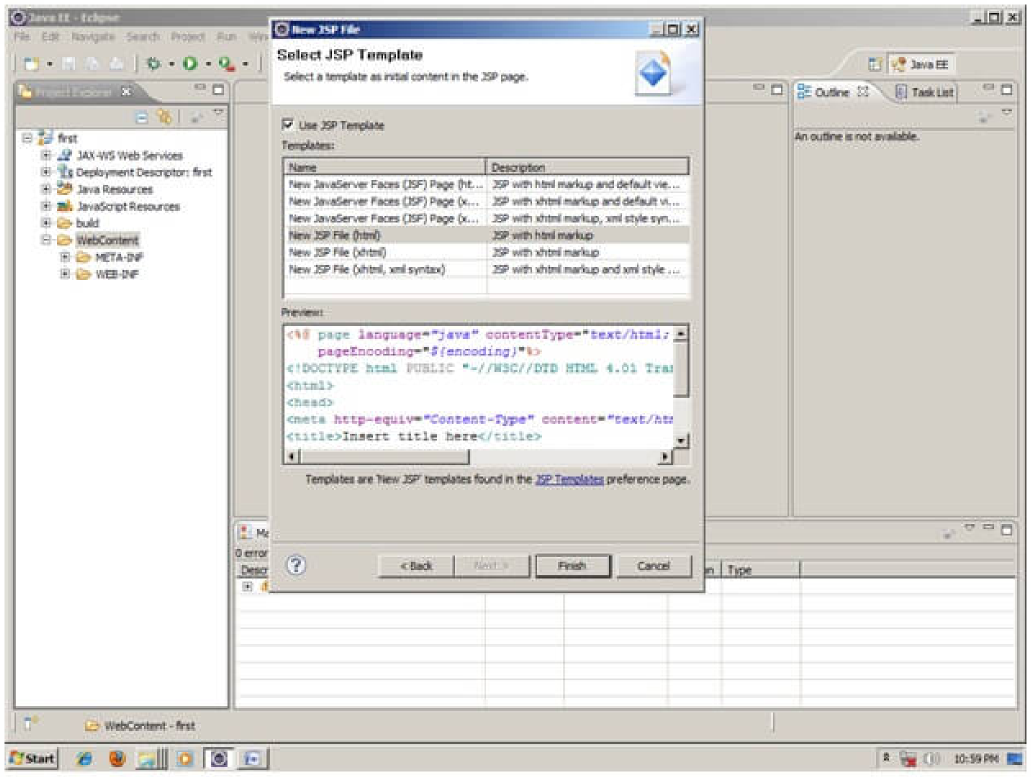
New JSP File

JSP

Create a new JSP file.

Enter or select the parent folder: first/WebContent

File name: index



New JSP File

Select JSP Template

Select a template as initial content in the JSP page.

☑ Use JSP Template

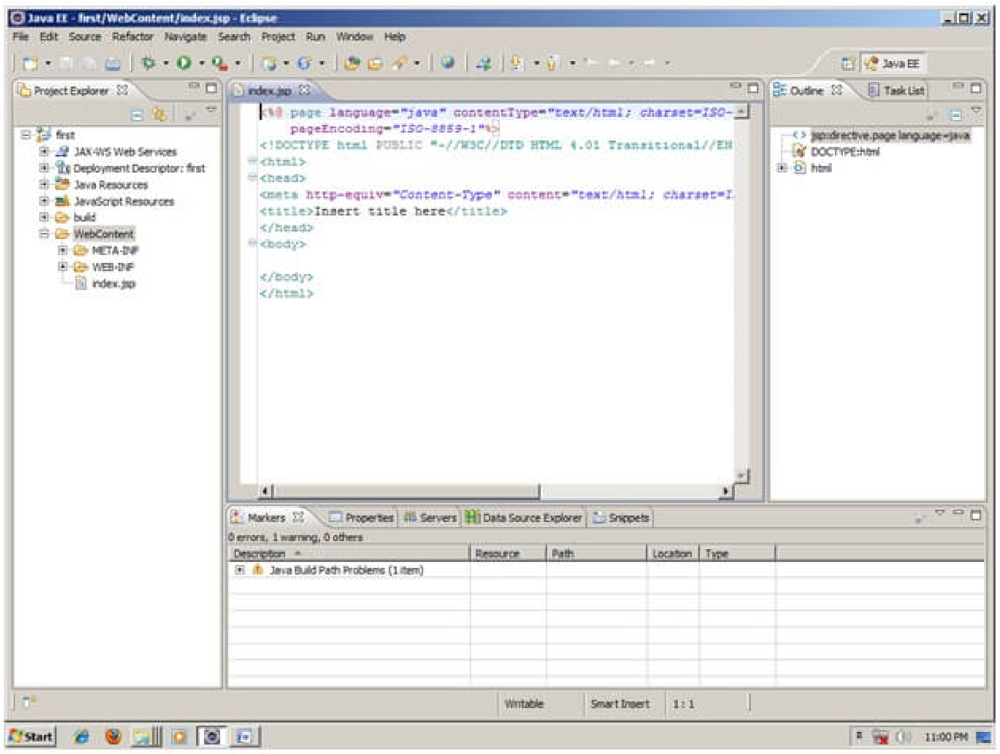
Templates:

| Name | Description |
|---|---|
| New JavaServer Faces (JSF) Page (ht... | JSP with html markup and default vie... |
| New JavaServer Faces (JSF) Page (x... | JSP with xhtml markup and default vi... |
| New JavaServer Faces (JSF) Page (x... | JSP with xhtml markup, xml style syn... |
| New JSP File (html) | JSP with html markup |
| New JSP File (xhtml) | JSP with xhtml markup |
| New JSP File (xhtml, xml syntax) | JSP with xhtml markup and xml style ... |

Preview:

```
<%@ page language="java" contentType="text/html;
    pageEncoding="${encoding}"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Tra
<html>
<head>
<meta http-equiv="Content-Type" content="text/ht
<title>Insert title here</title>
```

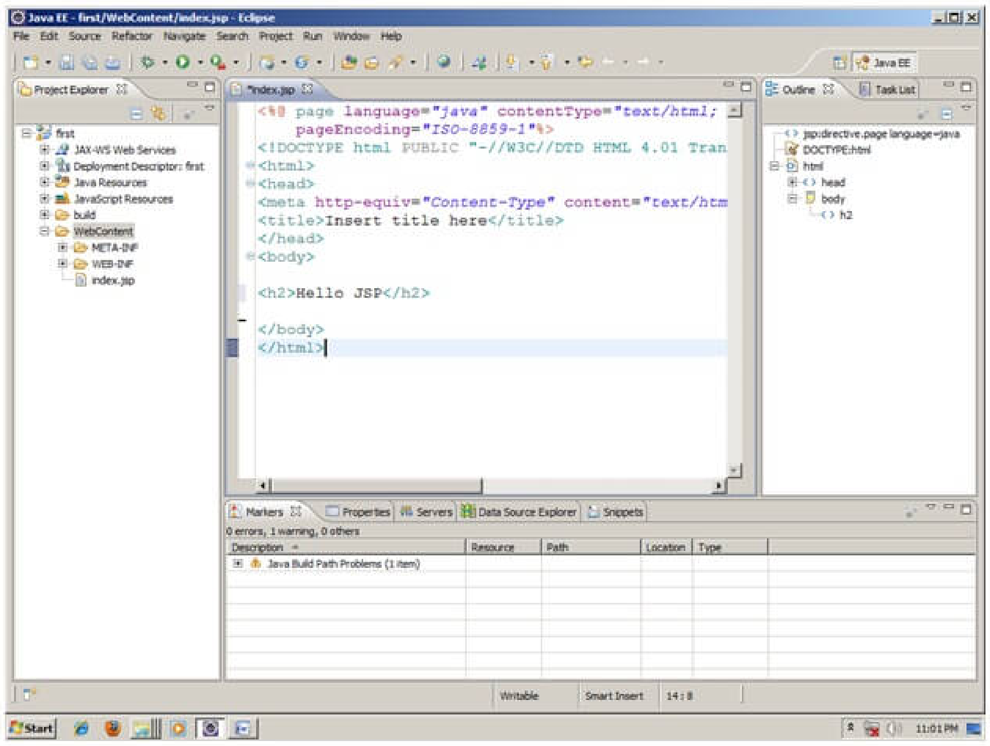
Templates are 'New JSP' templates found in the JSP Templates preference page.



```
<%@ page language="java" contentType="text/html; charset=ISO-
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=I
<title>Insert title here</title>
</head>
<body>

</body>
</html>
```

```
0 errors, 1 warning, 0 others
Description          Resource   Path   Location   Type
Java Build Path Problems (1 item)
```

Now JSP file is created, let's write some code.



```
<%@ page language="java" contentType="text/html;
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Tran
<html>
<head>
<meta http-equiv="Content-Type" content="text/htm
<title>Insert title here</title>
</head>
<body>

<h2>Hello JSP</h2>

</body>
</html>
```

```
0 errors, 1 warning, 0 others
Description          Resource   Path   Location   Type
Java Build Path Problems (1 item)
```

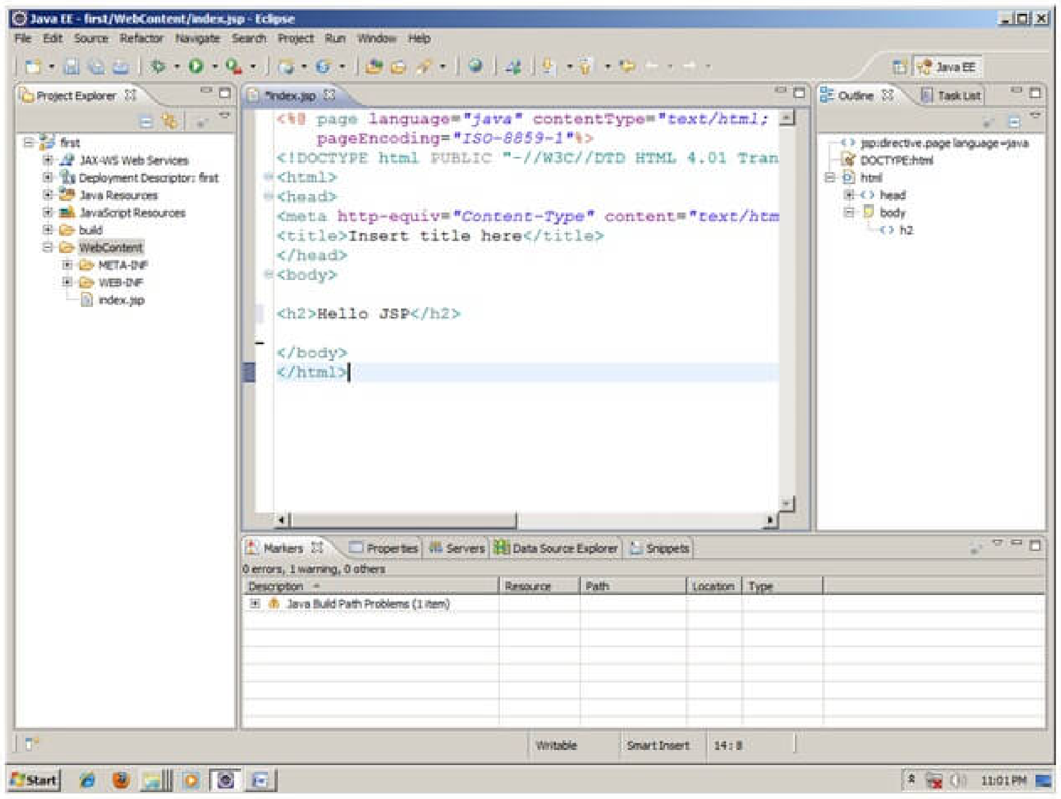
3) Start the server and deploy the project:

For starting the server and deploying the project in one step Right click on your project -> Run As -> Run on Server -> choose tomcat server -> next -> addAll -> finish.

If you are using Eclipse IDE first time, you need to configure the tomcat server First. Click for <u>How to configure tomcat server in eclipse IDE</u>

Now start the tomcat server and deploy project

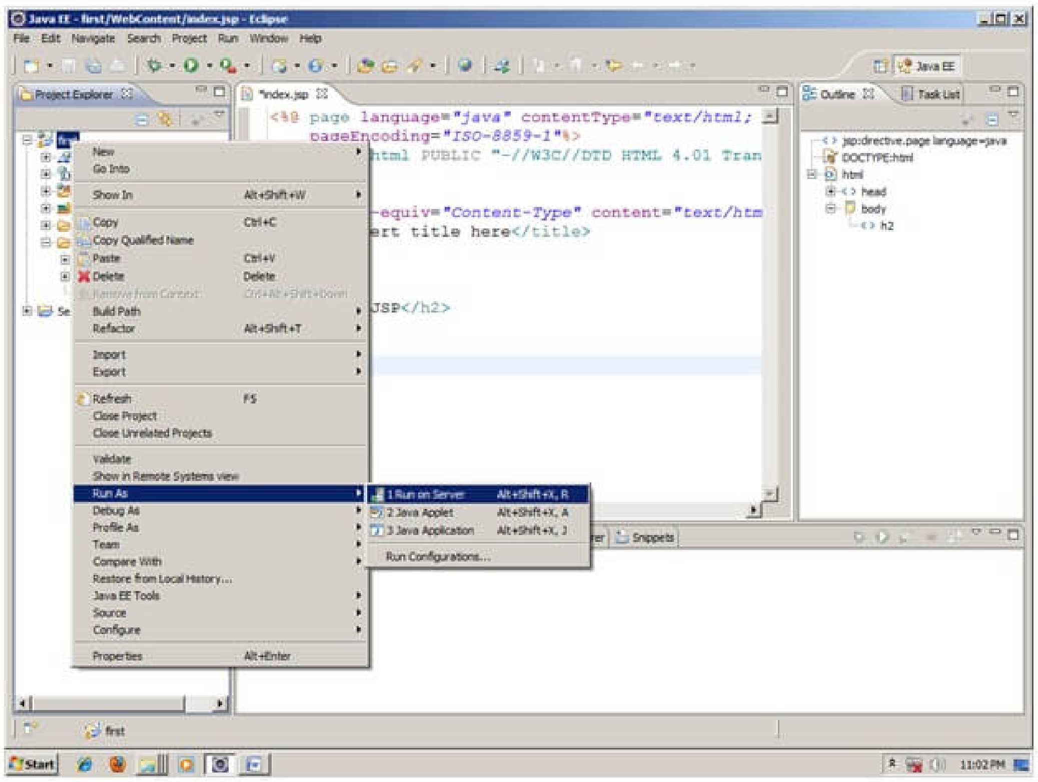
For starting the server and deploying the project in one step Right click on your project -> Run As -> Run on Server -> choose tomcat server -> next -> addAll -> finish.



```
<%@ page language="java" contentType="text/html;
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Tran
<html>
<head>
<meta http-equiv="Content-Type" content="text/htm
<title>Insert title here</title>
</head>
<body>

<h2>Hello JSP</h2>

</body>
</html>
```

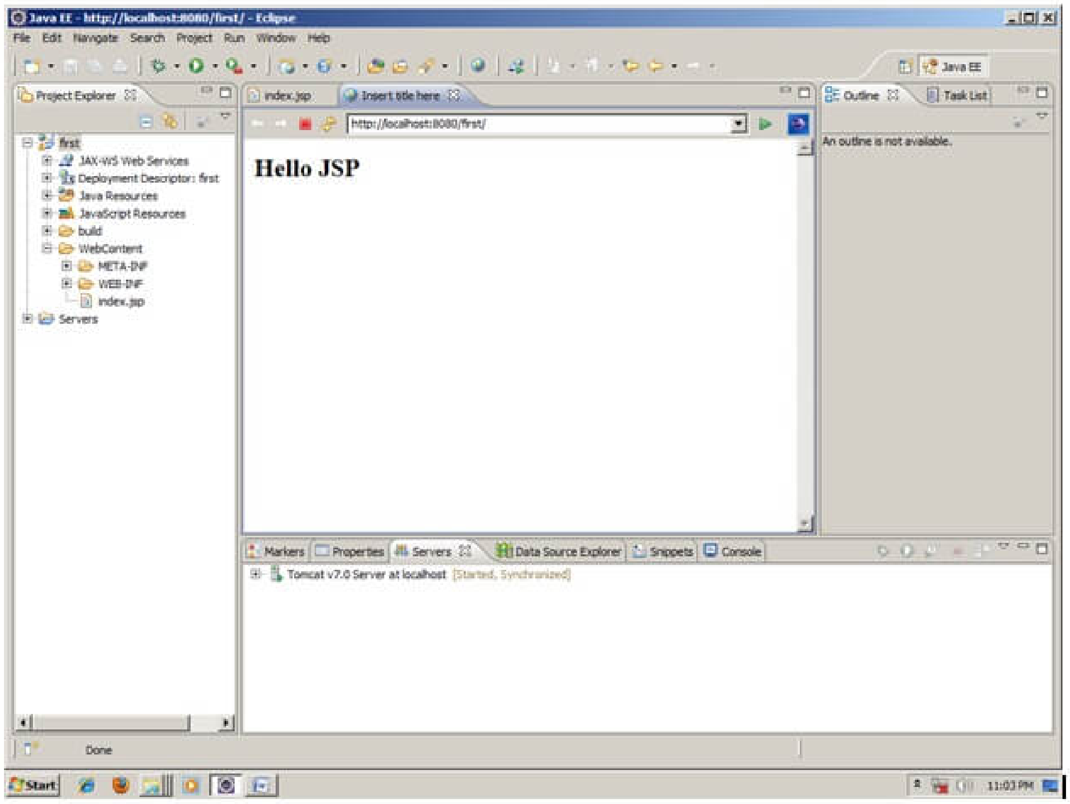
0 errors, 1 warning, 0 others

Java Build Path Problems (1 item)



```
<%@ page language="java" contentType="text/html;
    pageEncoding="ISO-8859-1"%>
html PUBLIC "-//W3C//DTD HTML 4.01 Tran

-equiv="Content-Type" content="text/htm
ert title here</title>

JSP</h2>
```

Yes, Let's see JSP is successfully running now.



Hello JSP

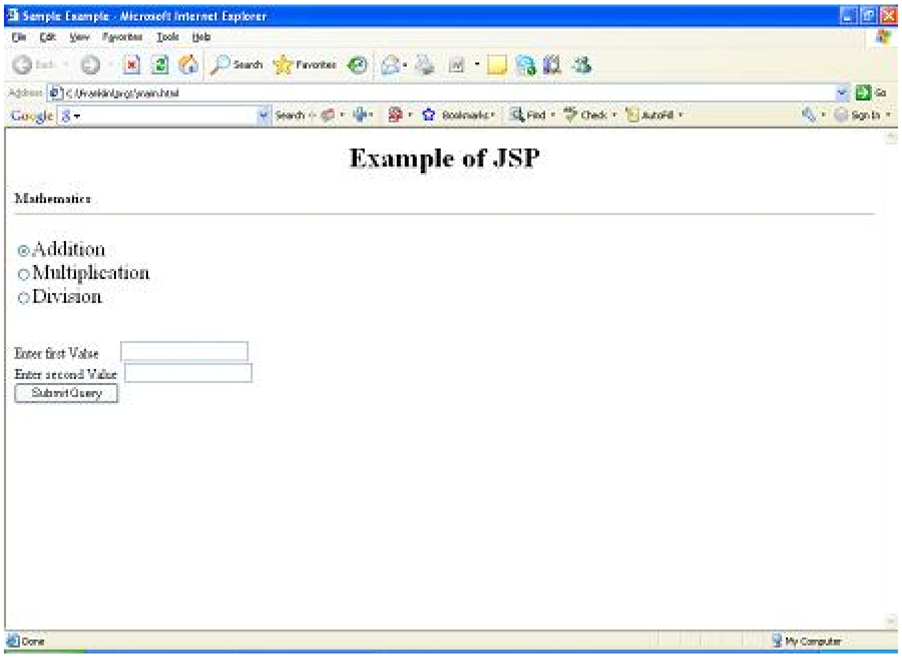
```
Tomcat v7.0 Server at localhost [Started, Synchronized]
```

## Sample Code:

```
<html>
        <title>Sample Example </title>
        <body>
        <h1><center> Example of JSP </center></h1>
        <b> Mathematics</b>
        <hr>
        <form method="post" action="a.jsp">
        <font size=5 face="Times New Roman">
        <input type="radio" name="a1" value="add" checked>Addition</input><br>
        <input type="radio" name="a1" value="mul" >Multiplication</input><br>
        <input type="radio" name="a1" value="div" >Division</input><br>
        </font>
        <br><br>
        Enter first Value      <input type="text" name="t1" value=""><br>
        Enter second Value  <input type="text" name="t2" value=""><br>
        <input type="submit" name="result">
        </form>
        </body>
</html>
```

```
%@ page language="java"%>
<%@ page import="java.lang.*"%>
<html>
        <body>
        <H1><center>Result for <%=request.getParameter("a1")%></center></H1>
        <%
        int i=Integer.parseInt(request.getParameter("t1"));
        int j=Integer.parseInt(request.getParameter("t2"));
        int k=0;
        String str=request.getParameter("a1");
        if(str.equals("add"))
        k=i+j;
        if(str.equals("mul"))
        k=i*j;
        if(str.equals("div"))
        k=i/j;
        %>
        Result is <%=k%>
        </body>
</html>
```

## Output:


Sample Example - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

Example of JSP

Mathematics

[✓ Addition]
Multiplication
Division

Enter first Value
Enter second Value
Submit Query

## Viva Voce Questions:

1. What is JSP?

2. What is the difference between JSP and Servlets?

3. Name the scripting elements.

4. What is the difference between page and include directives?

5. Mention some important JSP Action Tags.

## Experiments Beyond Syllabus

**QUESTION 1.**

Consider a function **public String matchFound(String input 1, String input 2),** where

- **input1** will contain only a single word with only 1 character replaces by an underscore '_'
- **input2** will contain a series of words separated by colons and no space character in between
- **input2** will not contain any other special character other than underscore and alphabetic characters.

The methods should return output in a String type variable **"output1"** which contains all the words from input2 separated by colon which matches with input 1. All words in output1 should be in uppercase.

Example 1: -
Input1="Fi_er";
Input2="Fever:filter:Filer:fixer:fibre:tailor:offer:fiber"
Output1="FILER:FIXER:FIBER"

Example 2: -
Input1="t_xer"
Input2="thanks:thin:trial:teller:troll:tider"
Output1="ERROR-009"

Example 3: -
Input1="thi_"
Input2="thin:that:those:this:what:whap:thick:thip:why:
Output1="THIN:THIS:THIP"

**Code:**

```
import java.util.StringJoiner;

public class StringProblem {
public static void main(String[] args) {
        StringJoiner sj=new StringJoiner (":");
        String s1="Fi_er";
        String s2="Fever:filter:Filer:Fixer:Fibre:tailor:offer:Fiber";
        String str[]=s2.split(":");
        for(String s:str)
        {
                if(s.length()==s1.length())
                {
                        String op=s1.replace('_',s.charAt(2));
                        //System.out.println(op+" "+s);
                        if(op.equals(s))
                        {
                                sj.add(s.toUpperCase());
                        }                               else
                        {       }
                }
        }    System.out.println(sj);
}
}
```

**QUESTION 2.**

Encoding Three Strings: Anand was assigned the task of coming up with an encoding mechanism for any given three strings.He has come up with the following plan.

**Step ONE** :- Given any three strings,break each string into 3 parts each.
For example- if the three strings are below:
Input 1: “John”
Input 2: “Johny”
Input 3 : “Janardhan”
“John” should be split into”J”,”oh”,”n,” as the FRONT ,MIDDLE and END part repectivly.
“Johny” should be spilt into “jo”,” h”, “ny” as the FRONT ,MIDDLE and END respectively.
“Janardhan” should be split into “Jan” ,” ard” ,”han” as the FRONT ,MIDDLE and END part respectively.
i.e. If the no. of characters in the string are in multiples of 3 ,then each split –part will contain equal no of characters , as seen in the example of “Janadhan”.
If the no. of characters in the string are NOT in multiples of 3 ,and if there is one character more than multiple of 3, then the middle part will get the extra character ,as seen in the example of “john”.
If the no. of characters in the string are Not in multiples of 3 and if there are two characters more than multiple of 3, then the FRONT and END parts will get one extra character each, as seen in the example of “Johny”.

**Step TWO** : Concatenate (join) the FRONT ,MIDDLE and END parts of the string as per the below specified concatenation – rule to from three Output strings.
Output 1: FRONT part of input 1 + MIDDLE part of input 2 +END part of input 3
Output2:- MIDDLE part of input1+ END part of input2 + FRONT part of input3
Output3: END part of the input1+FRONT part of input2 +MIDDLE part of input3
For example , for the above example input strings:
Output1 = “J” +”h”+han”=”jhhan”
Output2 =”oh”+”ny”+”Jan”=”ohnyjan”
Output3=”n”+Jo”+ard”+=“njoard”

**Step THREE** :-
Process the resulting output strings based on the output-processing rule .After the above two steps, we will now have three output string. Further processing is required only for the third output string as per below rule-
“Toggle the case of each character in the string “ ,i.e. in the third output string, all lower-case characters should be made upper-case and vice versa.
For example , for the above example strings ,output3 is “nJoard”, so after applying the toggle rule. Output3 should become “NjOARD”.

Final Result – The three output strings after applying the above three steps i.e. for the above example .
Output1 =”Jnhan”

Output2=”ohnyJan’
Output3 = “NJOARD”
Help Anand to write a program that would do the above.

**Code:**

```
import java.io.*;

import java.math.*;

import java.security.*;

import java.text.*;

import java.util.*;

import java.util.concurrent.*;

import java.util.function.*;

import java.util.regex.*;

import java.util.stream.*;

import static java.util.stream.Collectors.joining;

import static java.util.stream.Collectors.toList;

class Result {

public static String encodeString(String string1, String string2, String string3) {

        String[] str = divideString(string1);
        String[] str2 = divideString(string2);
        String[] str3 = divideString(string3);

        String in1 = str[0] + str2[0] + str3[0];
        String in2 = str[1] + str2[1] + str3[1];
        String in3 = str[2] + str2[2] + str3[2];
        String fstr3=toggleString(in3);


        System.out.println(fstr3);

        String strFinal = in1 + in2 + fstr3;
        return strFinal;
```

```
    }
    public static String[] divideString(String input) {
        int rem = input.length() % 3;
        int div = input.length() / 3;
        String first = null;
        String middle = null;
        String last = null;
        if (rem == 1) {
            first = input.substring(0, div);
            middle = input.substring(first.length(), first.length() + div + rem);
            last = input.substring(first.length() + middle.length(), input.length());
        }

        if (rem == 2) {
            first = input.substring(0, div + 1);
            middle = input.substring(first.length(), first.length() + div);
            last = input.substring(first.length() + middle.length(), input.length());
        }
        if (rem == 0) {
            first = input.substring(0, div);
            middle = input.substring(first.length(), first.length() + div);
            last = input.substring(first.length() + middle.length(), input.length());

        }
        return new String[] { first, middle, last };
    }

    public static String toggleString(String str) {
        char[] ch = str.toCharArray();
        String s="";
        for (int i = 0; i < ch.length; i++) {
            if (Character.isLowerCase(ch[i])) {
                s=s+Character.toUpperCase(ch[i]);
            }
            if (Character.isUpperCase(ch[i])) {
                s=s+Character.toLowerCase(ch[i]);
            }
        }
        return s;
    }

}
```

**QUESTION 3.**

Given a String (In Uppercase alphabets or Lowercase alphabets), new alphabets isto be appended with following rule:

(i) If alphabet is present in input string, use numeric value of that alphabet. E.g. a or A numeric value is 1 and so on. New alphabet to be appendedbetween 2 alphabets :

(a) If (sum of numeric value of 2 alphabets) %26 is 0, then append 0. E.g. string is ay. Numeric value of a is 1, y is 25. Sum is 26. Remainder is 0, new string will be a0y.

(b) Otherwise (sum of numeric value of 2 alphabets) %26 numeric value alphabet is to appended. E.g. ac is string. Numeric value ofa is 1, c is 3, sum is 4. Remainder with 26 is 4. Alphabet to be appended is d. output will be adc.

(ii) If digit is present, it will be same in output string. E.g. string is 12, outputstring is 12.

(iii) If only single alphabet is present, it will be same in output string. E.g.input string is 1a, output will be 1a.

(iv) If space is present, it will be same in output string. E.g. string is ac 12a,output will be adc  12a.

Constraint: Whether string alphabets are In Uppercase or Lowercase, appended alphabets must be in lower case. Output string must also be inlowercase.

**Test Cases:**

Input : Wipro Output :

wfiyphrgo

Input : Wipro123

Output : wfiyphrgo123

Input : Wi11pro123 Output :

wfi11phrgo123

**Code:**

```
 package assignments;
 import java.util.Scanner;

 public class UserIdGeneration {
          public static void main(String[] args) {
        System.out.println("Enter String ");
                    Scanner sc = new Scanner(System.in);
        String s = (sc.nextLine()).toLowerCase();
        char[] ch  = s.toCharArray();
        StringBuilder newString = new StringBuilder();
            for (int i = 0; i < ch.length; i++) {
                    if (i == ch.length-1 || !(Character.isAlphabetic(ch[i]) &&
 Character.isAlphabetic(ch[i+1]))) {
                              newString.append(ch[i]);
                                        } else{
                                                    int temp;
                      temp = (((int)ch[i]-96)+((int)ch[i+1]-96))%26;
                     if (temp == 0) {
                              newString.append(ch[i]);
                              newString.append('0');
                              }
                                                   else {
                                                            newString.append(ch[i]);
                              newString.append((char)(96 + temp));
                                                    }
                                        }
                              }
                    System.out.println("newId = " + newString);
                    sc.close();
          }
 }
```

**QUESTION 4.**

String t is generated by random shuffling string s and then add one more letter at a random position.

Return the letter that was added to t.

**Example 1:**

**Input:** s = "abcd", t = "abcde"

**Output:** "e"

**Explanation:** 'e' is the letter that was added.

**Example 2:**

**Input:** s = "", t = "y"

**Output:** "y"

**Code:**

```
class Solution {
   public char findTheDifference(String s, String t) {
      // result store the result
                  int res = 0, i;

                  // traverse string A till
                  // end and xor with res
                  for (i = 0; i < s.length(); i++)
                  {
                  // xor with res
                  res ^= s.charAt(i);

                  }

                  // traverse string B till end and
                  // xor with res
                  for (i = 0; i < t.length(); i++)
                  {
                  // xor with res
                  res ^= t.charAt(i);
                  }

                  // print result at the end
                  return ((char)(res));
   }
}
```

**QUESTION 5.**

The next greater element of some element x in an array is the first greater element that is to the right of x in the same array.
You are given two distinct 0-indexed integer arrays nums1 and nums2, where nums1 is a subset of nums2.
For each 0 <= i < nums1.length, find the index j such that nums1[i] == nums2[j] and determine the next greater element of nums2[j] in nums2. If there is no next greater element, then the answer for this query is -1.
Return *an array* ans *of length* nums1.length *such that* ans[i] *is the next greater element as described above.*

**Example 1:**
Input: nums1 = [4,1,2], nums2 = [1,3,4,2]
Output: [-1,3,-1]
Explanation: The next greater element for each value of nums1 is as follows:
- 4 is underlined in nums2 = [1,3,4,2]. There is no next greater element, so the answer is -1.
- 1 is underlined in nums2 = [1,3,4,2]. The next greater element is 3.
- 2 is underlined in nums2 = [1,3,4,2]. There is no next greater element, so the answer is -1.

**Example 2:**
Input: nums1 = [2,4], nums2 = [1,2,3,4]
Output: [3,-1]
Explanation: The next greater element for each value of nums1 is as follows:
- 2 is underlined in nums2 = [1,2,3,4]. The next greater element is 3.
- 4 is underlined in nums2 = [1,2,3,4]. There is no next greater element, so the answer is -1.

**Code:**
```
class Solution {
   public int[] nextGreaterElement(int[] nums1, int[] nums2)
            {
               int[] listResult = new int[nums1.length];
               for (int i = 0; i < nums1.length; i++)
               {
                   int x = findIndex(nums2, nums1[i]);
                   listResult[i] = -1;
                 for (int j = x+1; j < nums2.length; j++)
                 {
                    if (nums2[x] < nums2[j])
                    {
                     listResult[i]=nums2[j];
                     break;
                    }

                 }
                 //if (listResult[i] > 0) listResult[i] = -1;
               }
               return listResult;
            }
          public static int findIndex(int arr[], int t)
            {
               if (arr == null) {
                 return -1;
               }
               int len = arr.length;
               int i = 0;

               while (i < len) {
                 if (arr[i] == t) {
                   return i;
                 }
                 else {
                    i = i + 1;
                 }
               }
               return -1;
            }
}
```