

|  |  |
| --- | --- |
| Document name | JAVA FET Booklet |
| Version no. | 1.2 |
| Release date | 22-09-2020 |
| Classification | Departmental |





**JAVA Booklet**

**Contents**

[Introduction to Java 4](#_Toc51685754)

[Fundamentals of Java : 1 6](#_Toc51685755)

[Fundamentals of Java : 2 8](#_Toc51685756)

[OOPs Concepts 10](#_Toc51685757)

[Abstract classes & Interfaces 12](#_Toc51685758)

[Java Miscellaneous 14](#_Toc51685759)

[Exception Handling 16](#_Toc51685760)

[Collections & Generics 18](#_Toc51685761)

[IO, Serialization & Networking 20](#_Toc51685762)

[Multithreading 22](#_Toc51685763)

[JDBC 24](#_Toc51685764)

[New features in JAVA 9 and 10 26](#_Toc51685765)

[JEE OVERVIEW 28](#_Toc51685766)

[Servlets Intro 29](#_Toc51685767)

[Servlet Features 31](#_Toc51685768)

[Session Tracking 33](#_Toc51685769)

[Servlet Filters 35](#_Toc51685770)

[JSP Intro 37](#_Toc51685771)

[JSP and JSTL 39](#_Toc51685772)

[Introduction to Spring Framework 41](#_Toc51685773)

[Spring Core Features 1 43](#_Toc51685774)

[Spring Core Features 2 45](#_Toc51685775)

[Introduction to Spring MVC 47](#_Toc51685776)

[Spring MVC Form Handling 49](#_Toc51685777)

[Introductio to JPA 51](#_Toc51685778)

[Spring with data JPA 52](#_Toc51685779)

[REST API Introduction 54](#_Toc51685780)

[REST with Spring 55](#_Toc51685781)

[New Features in Spring 5 57](#_Toc51685782)

[Spring security 58](#_Toc51685783)

[Spring cloud 60](#_Toc51685784)

[JUNIT 62](#_Toc51685785)

[LOG4J 63](#_Toc51685786)

**Notes for assignments:**

* Make assumptions wherever necessary, and mention the same against the question.
* Use Exception Handling wherever required.
* Paste the solution application on the share folder (information will be provided during training).
* Submit the assignmnets on time.
* Follow the java coding satndards :

Introduction to Java

**Agenda**

* History of Java
* Features of Java
* Data types in Java
* Access modifires
* Writing First Java Class
* Accessors and Mutators
* Constructors
* 'this keyword

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Write first java program |
| **Assignment #2**  Write a program to print size of different data types in java |
| **Assignment #3**  Write a program to demonstrate different access modifiers |
| **Assignment #4**  Create account class with different types of constructors |

Fundamentals of Java : 1

**Agenda**

* Variables in Java
* static' keyword
* Memory layout
* Concept of Garbage Collector
* Parameter Passing (Call By value and ref)
* Arrays In Java
* Enhanced for loop
* Packages In Java

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Create a class Employee & write Accessors & Mutators  Create objects with default constructor & set values using mutators.  Display/print values of attributes of each object created. |
| **Assignment # 2**  Write a class Date. Overload constructor to initialize attributes.  Create Date objects using non-parameterized & parameterized constructors both.  Print values of day, month & year for every object. |
| **Assignment # 3**  Write a Java program that reads an integer between 0 and 1000 and adds all the digits in the integer.  *Test Data* Input an integer between 0 and 1000: 565 *Expected Output* : The sum of all digits in 565 is 16 |
| **Assignment # 4**  Write a Java program to break an integer into a sequence of individual digits.  *Test Data* Input six non-negative digits: 123456 *Expected Output* : 1 2 3 4 5 6 |
| **Assignment #5**  Write a Java method to display the middle character of a string.  Note: a) If the length of the string is odd there will be two middle characters. b) If the length of the string is even there will be one middle character. |

Fundamentals of Java : 2

**Agenda**

* Date time class in java 8
* Static import
* Packages

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Write a Java program to get the last date of the month also  get current full date and time.(use java 8 date time api) |
| **Assignment # 2**  Create an array of Employees & display nos. of Employee objects created. |
| **Assignment # 3**  Create a class Calc inside a package “utility”. This class will have a method findRoot()which will return the square root of a given number.  Create another class User1 inside a package allusers. User1 calls the method findRoot()to get square root of a number.  These two packages should be in different directories on two different drives. (may be C: & D:) |
| **Assignment # 4**  Write a Java program to insert an element (specific position) into an array. |
| **Assignment # 5**  Define variables in class using static method initialize these variables and display. |
| **Assignment # 6**  Write a program to calculate area of triangle. Use PI value using static import |
| **Assignment # 7**  Write a program to demonstrate package scope |
| **#Comments**  Write Singleton class (Note – We can create Only one object of such a class).  Write a program to find smallest and largest element in the given array.  Write a Class Circle and call the getPI() static method by using static import. |

OOPs Concepts

**Agenda**

* Need of OOP
* Procedural vs OOP language
* Object characteristics
* Major Pillars of OOPS
* Inheritance and Polymorphism(Dynamic Binding)
* Abstract classes and Interfaces(Functional Interfaces)
* Object Class
* Functional interface
* Lambda Expression

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Create a class hierarchy: Employee  Manager  SalesManager  Override calculateSalary() method. Manager will have allowance field &  SalesManager will have commission field. |
| **Assignment #2**  Create a class: Bank Account  Create subclasses: 1. Saving & 2.Current  Rate of interest is different for different types of account. |
| **Assignment #3**  Write a program to create a class named EmployeeDetails and display a menu similar to the following menu:  --------------Menu-------------- 1. Enter Data 2. Display Data 3. Exit  Choose the option Thereafter, invoke the respective method according to the given menu input. The methods will contain appropriate message, such as the displayData() method will contain the message, displayData method is invoked |
| **Assignment #4**  Write a program to understand predefined functional interfaces. Implement using anonymous classes |
| **Assignment #5**  Write a functional interface for arithmetic operation. Implement functional interface using lambda expression for different operations. |

Abstract classes & Interfaces

**Agenda**

* Abstract classes
* Interfaces
* Difference between Abstract Classes & Interfaces

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Create a class hierarchy : Animal  Cat Dog Lion  Animal class will have methods like respirate () & talk()  Override method talk() from Animal class in its each subclass  Create an array of Animal references & observe dynamic polymorphism |
| **Assignment # 2**  Create a class hierarchy : Shape  Rectangle Circle Triangle  Override method calculateArea() from Shape class in its each subclass  Create an array of Shape references & observe dynamic polymorphism |
| **Assignment # 3**  Create an interface: Printable – method print()  Write a class Utiltiy to have a method printAll( Printable [])  Pass different objects (Shapes & Animals) to printAll() to print them. |
| **Assignment # 4**  Create a bank and insurance interface. Create HDFC and ICICI as abstract classes implementing common features. Instantiate HDFC and ICICI bank |

Java Miscellaneous

**Agenda**

* Object class
* Garbage Collector
* Wrapper classes
* Auto boxing & unboxing
* Annotations

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Override equals(),toString() in Employee & Account class |
| **Assignment # 2**  Call System.gc(), check whether GC runs or not |
| **Assignment # 3**  Accept numbers from command line arguments & sort them |
| **Assignment #4**  Override the toString method of the Object class in the Employeeclass and in the Manager class. You will create an EmployeeStockPlan class with agrantStock method that uses the instanceof operator to determine how much stock togrant based on the employee type. |
| **#Comments**  Create object cloning of associated classes i.e Employee has an Address.(Override the Clone method of Object Class).  Create a Simple Calculator application. Get all the numbers through Command line arguments and use Wrapper Class Conversion method to calculate the result. |

Exception Handling

**Agenda**

* What are Exceptions ?
* OOP way of Exception Handling(try,catch,finally)
* Types of Exceptions
* Multicatch and ARM - Automatic Resource Management
* User Defined Exceptions (throw and thows keywords)

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Write a program to demonstrate the use of try, catch, finally throw and throws keywords and demonstrate the following points in the program. a) Multiple catch blocks. b) try-catch-finally combination. c) try-finally combination. d) Exception propagation among many methods. e) Use of getMessage(), printStackTrace() function of Throwable class. f) Nested try blocks |
| **Assignment # 2**  Create a NegativeNumberException to be thrown by the method  findSqrRoot() if user enters a negative number. |
| **Assignment #3**  1) Create an employee class with relevant information like name, id, salary and create employee objects.  2) Create Customer class with relevant information like name, address,  account number, current balance. Create BankApplication class and  add customers to the bank application with relevant methods like addCustomer,  deleteCustomer, updateCustomer and getCustomerInfo etc.  3) Create Account class with account type, account number,  minimum balance and current balance and provide corresponding getter and setter methods along with calInterest method. Create FixedDepositAccount, CurrAccount classes and  inherit methods from Account class. Use Account class in Customer class to store account information in the customer object.  4) Create InsufficientBalance exception class and use it appropriately in Account class.  5) Query BankApplication Object and display existing customer names. |
| **Comments**  Write a File Handling Application to Write data inside file by using PrintWriter and Use Try With Resource. |

Collections & Generics

**Agenda**

* Concept of Generics
* Wrapper Classes
* Need of Collections
* Collection Hierarchy
* List
* Set
* Iterating Collection (traditional and Java 8 ways)
* Sorting In Collection (Comparable and Comparator interfaces)
* Map - (HashMap ,TreeMap)
* Java 8 - Stream API in Collection

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Create a sorted set of Employee objects.(Sort on Emp ID) |
| **Assignment # 2**   1. Create a list of Employee objects & sort them *by names* in alphabetical order (may be same as above.) 2. Create a user defined exception to check whether your employee exist in your data structure (use any data structure to store the employees - like array, ArrayList etc) and throw exception if name is not in the employees list. Use the catch and finally block to make an appropriate. |
| **Assignment # 3**  Create a HashMap for books & no.of pages.  Display contents of Map using iterator. (Where Book is a class)?  (Override equals() & hashCode() in book class. .note:equal s/hashcode contract should not be violated) |
| **Assignment #4**  1) Create **Customer** class with the following methods. private String custName = null; private int ACC\_NO = 0; private Hashtable accountInfo = new Hashtable(); - Key can be one of SB\_ACCOUNT or CURR\_ACCOUNT defined  void depositeAmount(int ACC\_TYPE, int amount) - Deposite amount into the corresponding account type. void setCustomerName (String name) - Set the customer name String getCustomerName () - Get the customer name int getAccountNumber() - Get the account number void setAccountNumber(int accno) - Set the account number 2) Create **BankApplication** class with the following methods. private Hashtable customerInfo = new Hashtable() - Add new customer by taking his name as key and object  reference as value addNewCustomer(Customer cust) - Add new customer to the bank application.  public void depositeAmount(String name, int ACC\_TYPE, int amount) - Add amount to the corresponding customer's  account 3) Create a controlling class named **BankingOperation** with main method and accordingly instantiate objects of above classes. This class should be outside the above package. |

IO, Serialization & Networking

**Agenda**

* Concept of Streams
* Binary Streams vs Character Streams
* File - Reading and Writing
* Serialization and De-Serialization
* Concept of Layering Streams
* New File System API(NIO )
* Networking -Client Server Computing
* TCP/IP Sockets (Socket, ServerSocket and port number)

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Read input data from keyboard & write it to a file “File1”.  Copy contents of File1 to File2. |
| **Assignment # 2**  Serialize & deserialize Account object which will have  a Date type of field : doc (date of creation) |
| **Assignment # 3**  Create a simple chat application (Client-Server) |
| **#Comments**  Create multi-client socket application where multiple Client can access Date & Time Server which provides Date and Time of the Server Location. |

Multithreading

**Agenda**

* What is Thread ? Why Threads ?
* Thread Scheduling (Concept of Context Switch )
* Ways of creating threads in Java
* Thread lifecycle and methods
* Thread Synchronization
* Inter-thread communication (wait, notify and notifyAll methods)
* Executors, Tasks, ForandJoin, Futures

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Two members of a joint Account are trying to withdraw Rs.3000 from the account  simultaneously & balance is Rs.5000.  Create a multithreaded program to handle this situation |
| **Assignment # 2**  Write a program to demonstrate executor services .(check number of thread created) |
| **#Comments**   * Create GUI application to move Strings from opposite directions. * Create an application wherein you have to create 3 different threads accessing a resource called Table i.e class , which has a method called printTable and use synchronizaion to display 5, 10 and 100 tables respectively. |

JDBC

**Agenda**

* File vs RDBMS
* What is JDBC ?
* Driver Types in JDBC
* Basic Steps in Using JDBC
* JDBC Architecture

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Insert & update Employee data in database & display it using Prepared Statement.  Insert some records in Student Table |
| **Assignment # 2**  Create a stored procedure for updating the basic salary of employees as 10%HRA should be added to basic salary . |
| **Assignment # 3**  Write generic code to display column names of a table along with data in the table (same code should work to display records from Emp table and then for Student table |
| **#Comments**   * Write a transaction management application where you have to create two Accounts   From and To resp. and transfer the money from one account to another, use commit and rollback functions for transaction management.   * Which method we should use when we don’t know the query i.e either it can be select or insert, find the method and get the query input from the user and perform select and insert operations on that table. |

New features in JAVA 9 and 10

* The Java Platform Module System
* Jshell : The interactive Java REPL
* Stream API Improvements
* private interface methods
* Local variable type inference

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Explore different jshell commands |
| **Assignment # 2** Write a program to use factory Methods for Immutable List, Set, Map and Map.Entry |
| **Assignment #3**  Write a program to understand module system in java 9 |

JEE OVERVIEW

**Agenda**

* N tier Architecture
* JEE Overview
* Web Applications

Servlets Intro

**Agenda**

* Introduction to JEE
* Introduction to Servlets
* LifeCycle of servlets
* Servlets Collaboration
* Request Dispatcher
* Session Management

|  |
| --- |
| **Assignments** |
| **Assignment # 1**  Create a simple servlet to print Hello message with User’s name |
| **Assignment # 2**  Create a simple servlet demo to demonstrate Servlet’s Life cycle |
| **Assignment #3**  Create a servlet for login form that accepts username and password respectively |

Servlet Features

**Agenda**

* Servlet Config
* Servlet Context
* Request Dispatcher

|  |
| --- |
| **Assignments** |
| **Assignment #1**  **CybageNet with BooksReview**    1. CybageNet Login:  a) Administrator Login  b) User Login  2. Library System:  a) Administrator can do following:  -Insert books into database  -Remove books from database  b) User can do following:  - Book search (based on Title)  - Show Details of the selected book (Including reviews)  - add a review |
| **#Comments**   * Create a form which has two input fields to get two numbers and after pressing Add button calculate the result on first servlet class and use RequestDispatcher to display the result on second servlet. |

Session Tracking

**Agenda**

* Hidden Form Field
* URL Rewriting
* Cookies
* Http Session

|  |
| --- |
| **Assignments** |
| **Assignment # 1** (continue with BookReview assignment)  Administrator should be able to check how many users are logged in simultaneously. |
| **Assignment # 2**    Use cookies to show no. of visitors to the site. |
| **#Comments**   * Create a form which accepts Product Details i.e. id , name, and price and set the details in Product model object and put it in a session and display it on the 2nd Servlet Page * One the 2nd Servlet page create a link logout once user presses that logout button user should navigated to UserLogoutServlet where user should see a message as you have logged out successfully. |

Servlet Filters

**Agenda**

* Filters Introduction
* Filter Demo

|  |
| --- |
| **Assignments** |
| **Assignment #1** ( continue with BookReview assignment )  Create a simple logging filter to log the information about LogIn timings of the users |
| **#Comments**   * Create a student registration form and using ValidationAndRendering Filter class   Apply blank field validations and if all the fields are valid then show the response rendered in a table where rendering part will also come in the same filter class. |

JSP Intro

**Agenda**

* Introduction to JSP
* JSP Lifecycle
* JSP implicit objects
* JSP Directives

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Replace servlets by JSPs in BookReview assignment  (Example: Login & Books details page) |

JSP and JSTL

**Agenda**

* Introduction to JSTL
* JSTL usage

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Continue with JSP BookReview assignment.  Create JavaBean for Book (BookBean) & Use useBean standard action |
| **Assignment #2**  Using Java Bean Book create a list of objects and set some attributes that will be used in the JSP. JSP page will show how to iterate over a collection, using conditional logic with EL and some other common usage.(hint: JSTL Core Tags) |
| **#Comments**   * Create a Sportsman Info Form where you have to accept player name, game and place of birth and date of birth and insert it inside the Player Bean by JSP Standard actions and display it on the third jsp page. |

Introduction to Spring Framework

**Agenda**

* Introduction to Spring
* Spring Modules
* IoC (Dependency Injection)
* Beans and Lifecycle of Bean
* Steps to create Spring Application
* Types of Dependency Injections

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Create a Maven project to build a Spring Based application.( work on the pom.xml ) |

Spring Core Features 1

**Agenda**

* Beans Lifecycle
* Beans Scope
* Types of dependency injections

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Create a Spring based app where the bean “**OutputHelper**” declared in ‘Spring-Common.xml‘ can access to other beans in ‘Spring-Output.xml‘ – “**CsvOutputGenerator**” or “**JsonOutputGenerator**“, by using a ‘ref’ attribute in property tag.  Try to implement the same keeping all bean in same file. |
| **Assignment #2**  Create a User bean class with name ,city, country field and use the setter injection to inject the value also implement the constructor injection in the same example. |
| **#Comments**  Create a simple spring maven application to create two beans Circle and Rectangle which are implementing Shape interface which has area and perimeter abstract methods, calculate both the area and perimeter of the shapes by coding through interfaces concept. |

Spring Core Features 2

**Agenda**

* Collections
* Factory Pattern.
* Spring Annotations

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Create a Spring Core application for Forum where **One question can have multiple answers**.   1. **Question.java** 2. **applicationContext.xml** 3. **Test.java**   Use any collection element to store the multiple answer.(list,set,prop) |

Introduction to Spring MVC

**Agenda**

* Spring MVC architecture
* Components in Spring MVC
  + Model
  + Controller
  + View Resolver

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Create simple Spring MVC supported **employee management application** having only one feature i.e. list all available employees in system.  Application front UI |

Spring MVC Form Handling

**Agenda**

* Spring MVC Form Handling
* MVC Annotations

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Continue the above the example by creating form to accept the employee details and then display the list of employee. |
| Create a Spring Boot + MVC Application to Manage Customer Entries  In a XYZ Company. Perform all the CRUD Operations and also provide search customer functionality (i.e. By Id). Please find below Snapshot.  Note –   1. Please Use Proper Coding Conventions. 2. Use Comments whenever it is necessary.   C:\Users\anupki\Downloads\Customer_List_1.png |

Introductio to JPA

**Agenda**

* JPA Architecture
* ORM Architecture
* JPA Provider

Spring with data JPA

**Agenda**

* Spring data JPA
* Configuration
* Demo

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Create a PersonRepository(Id,Name,Address,ContactNo,Age) to perform the following operations :  1.Save the perosn record  2. Search a record by PK  3. Delete a record  4. Update the record. |
| **#Comments**   * Create the same customer application by adding JPA |

REST API Introduction

**Agenda**

* Introduction to WebServices
* REST vs SOAP

REST with Spring

**Agenda**

* Resource Methods
* Spring-Rest Architecture
* Annotations

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Write REST API for CRUD operation for Employee Management System.  (GET,PUT,POST,DELETE) using Spring MVC.(Bonus point to implement RestTemplate API) |

New Features in Spring 5

**Agenda**

* Support Java 8, Java EE 8, Servlet 4.0, Bean Validation 2.0, and JPA 2.2, JMS
* Improved Logging with new module – spring-jcl
* File operations are using NIO 2 streams, hence improved performance
* Support for Kotlin, Project Lombok, JSON Binding
* Spring WebFlux – Spring getting Reactive
* Support for JUnit 5

Spring security

**Agenda**

* Introduction to Spring Security
* Spring Security
* Demo

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Create simple spring based application with login and logout feature |
| **Assignment #2**  Create simple spring based application with login and logout feature (fetch user data from database using JDBC) |
| **Assignment #3**  Create simple spring based application to check method level security |

Spring cloud

**Agenda**

* Introduction to Spring cloud
* Different tools in spring cloud
* Demo

|  |
| --- |
| **Assignments** |
| **Assignment #1**  Microservice registration and discovery using Netflix Eureka |
| **Assignment #2**  Simple microservice to demonstrate Zuul gateway |

JUNIT

**Agenda**

* Why Testing?
* Old Way Vs New Way
* Terminology
* Assert Methods
* Overview

LOG4J

**Agenda**

* Need Of Logging
* Components of Log4J
* Levels of logging
* Configuration

**Declaration by the Participant**

**My Understanding**

Name:

Employee ID:

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No** | **Topic** | **Theoretical** | **Practical** |
| **Yes** | **No** | **Yes** | **No** |
| 1 | Introduction to Java and Object Oriented Programming Concepts |  |  |  |  |
| 2 | Basics of Java (Arrays)and Inheritance |  |  |  |  |
| 3 | Abstract Classes and Interfaces |  |  |  |  |
| 4 | Exception Handling |  |  |  |  |
| 5 | Collections Framework |  |  |  |  |
| 6 | Generics |  |  |  |  |
| 7 | Multithreading |  |  |  |  |
| 8 | Synchronization |  |  |  |  |
| 9 | IO, Serialization and Networking |  |  |  |  |
| 10 | Basics of MYSQL(Command Line) |  |  |  |  |
| 11 | JDBC |  |  |  |  |
| 12 | JAVA 7 new Features |  |  |  |  |
| 13 | JAVA 8 ( Functional Programming) |  |  |  |  |
| 14 | JAVA 8 |  |  |  |  |
| 15 | Introduction to Java EE, Basics of Servlet & its Lifecycle |  |  |  |  |
| 16 | Servlet features (Context, Config) |  |  |  |  |
| 17 | Servlet filters |  |  |  |  |
| 18 | Session tracking |  |  |  |  |
| 19 | JSP basics & scripting |  |  |  |  |
| 20 | JSP standard actions |  |  |  |  |
| 21 | JSTL |  |  |  |  |
| 22 | Spring Framework Introduction and IOC/DI Container (First Application) |  |  |  |  |
| 23 | Spring Core (Dependency Injection) |  |  |  |  |
| 24 | Spring Core(Collections and Factory Pattern) |  |  |  |  |
| 25 | Intro to Spring MVC |  |  |  |  |
| 26 | Spring MVC Form Handling and Annotations |  |  |  |  |
| 27 | What is JPA |  |  |  |  |
| 28 | Spring with JPA(Spring Data) |  |  |  |  |
| 29 | REST Introduction |  |  |  |  |
| 30 | Spring with REST |  |  |  |  |
| 31 | Spring security |  |  |  |  |
| 32 | Spring cloud |  |  |  |  |
| 33 | New Features in Spring 5 |  |  |  |  |
| 34 | JUNIT |  |  |  |  |
| 35 | LOG4J |  |  |  |  |

**Signature of Participant**

|  |
| --- |
| **For Office Use:**  **Signature of Trainer Signature of Evaluator Signature of Training Head** |