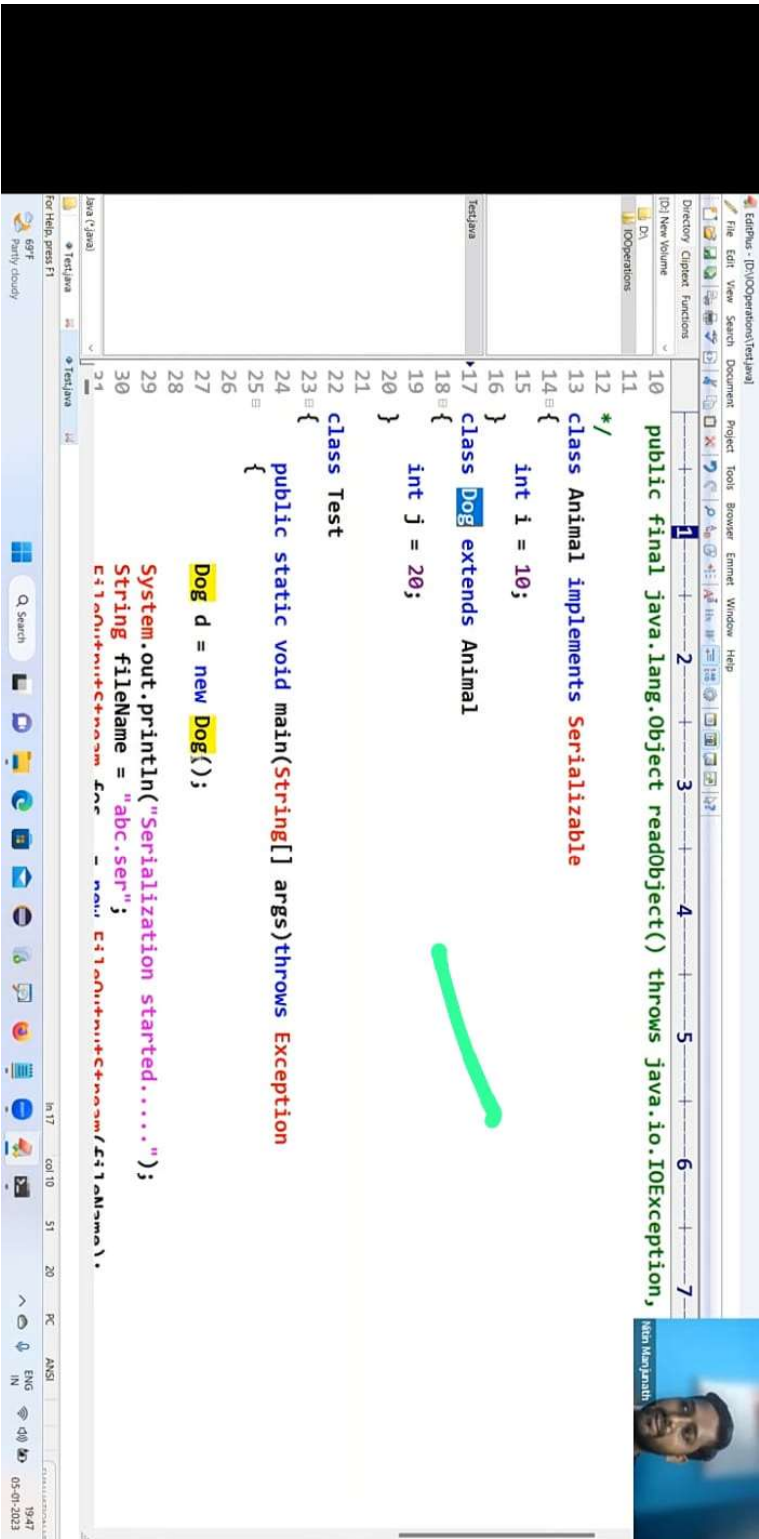


## 46Java\_Serialization\_Deserialization\_Part2



```
10 public final java.lang.Object readObject() throws java.io.IOException,
11
12  */
13 class Animal implements Serializable
14 {
15     int i = 10;
16 }
17 class Dog extends Animal
18 {
19     int j = 20;
20 }
21
22 class Test
23 {
24     public static void main(String[] args) throws Exception
25     {
26
27         Dog d = new Dog();
28
29         System.out.println("Serialization started.....");
30         String fileName = "abc.ser";
31         FileInputStream fis = new FileInputStream(fileName);
```

Command Prompt

X + ↵

De-Serialization started.....

10----->20

De-Serialization ended.....

D:\I00operations>javap java.lang.Object

Compiled from "Object.java"

public class java.lang.Object {

public java.lang.Object();

public final native java.lang.Class<?> getClass();

public native int hashCode();

public boolean equals(java.lang.Object);

protected native java.lang.Object clone() throws java.lang.CloneNotSupportedException;

public java.lang.String toString();

public final native void notify();

public final native void notifyAll();

public final native void wait(long) throws java.lang.InterruptedException;

public final void wait(long, int) throws java.lang.InterruptedException;

public final void wait() throws java.lang.InterruptedException;

protected void finalize() throws java.lang.Throwable;

static {};

}

D:\I00operations>cls



69°F  
Partly cloudy

Q Search



ENG  
IN



19:49  
05-01-2023

File Edit View

**Case 1:**

If parent class implements Serializable then automatically every child class by default implements Serializable. That is Serializable nature is inheriting from parent to child.

Hence even though child class doesn't implements Serializable , we can serialize child class object if parent class implements serializable inte

```
import java.io.Serializable;
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;
import java.io.FileInputStream;
import java.io.ObjectInputStream;
import java.io.IOException;
```

```
class Animal implements Serializable{
```

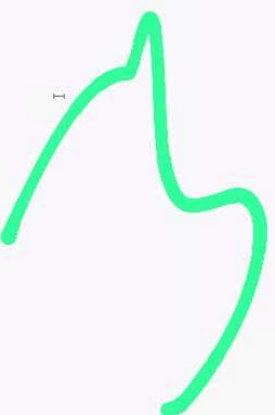
```
    int i=10;
```

```
}
```

```
class Dog extends Animal{
```

```
    int j=20;
```

```
}
```





```
TestJav - (D:\Operations)\TestJav
File Edit View Search Document Project Tools Browser Emmet Window Help
ID3 New Volume
Directory Clipboard Functions
D3
ID3 Operations
TestJav
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
}
// JVM shutdown now

System.out.println("Serialization started.....");
String fileName = "abc.ser";
FileOutputStream fos = new FileOutputStream(fileName);
ObjectOutputStream oos = new ObjectOutputStream(fos);
oos.writeObject(d);
System.out.println("Serialization ended.....");

//To pause the execution till we press some key from keyboard
System.in.read();

System.out.println("De-Serialization started.....");
FileInputStream fis = new FileInputStream("abc.ser");
ObjectInputStream ois = new ObjectInputStream(fis);
Dog d1=(Dog)ois.readObject();

System.out.println(d1.i+"----->"+d1.j);
System.out.println("De-Serialization ended.....");
}
```



SD

Editor - (D:\Operations\testjava)

File Edit View Search Document Project Tools Browser Emmet Window Help

Directory: C:\Users\Niranjan\Documents\testjava

D:\Operations

```
7
8 public java.io.ObjectInputStream(java.io.InputStream) throws java.io.IOException;
9 public java.io.FileInputStream(java.lang.String) throws java.io.FileNotFoundException;
10 public final java.lang.Object readObject() throws java.io.IOException, java.lang.ClassN
11
12 */
13
14 class Animal
15 {
16     int i = 10;
17     Animal() {
18         System.out.println("Animal constructor called...");
19     }
20 }
21
22 class Dog extends Animal implements Serializable
23 {
24     int j = 20;
25     Dog() {
26         System.out.println("Dog constructor called...");
27     }
28 }
```

TestJava

TestJava

TestJava

For help, press F1

69°F

Family cloudy

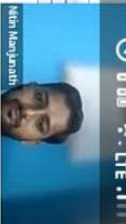
Q Search

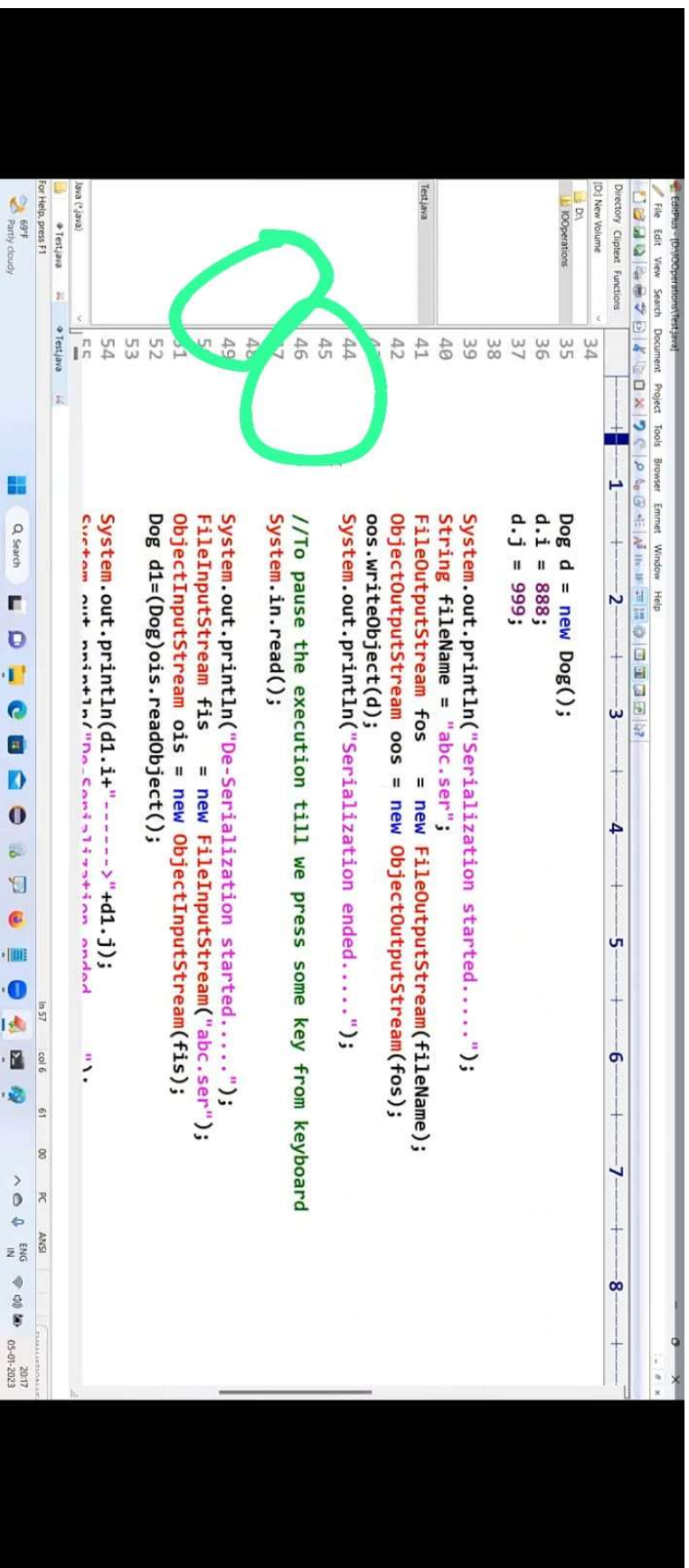
In 18 col 5 5 09 PC ANSI

05-01-2023

Niran Manjanna









Command Prompt

D:\IOOperations>javac Test.java

D:\IOOperations>java Test  
Animal constructor called...  
Dog constructor called...  
Serialization started.....  
Serialization ended.....

De-Serialization started.....  
Exception in thread "main" java.io.InvalidClassException: Dog; no valid constructor  
at java.io.ObjectStreamClass\$ExceptionInfo.newInvalidClassException(ObjectStreamClass.java:169)  
at java.io.ObjectStreamClass.checkDeserialize(ObjectStreamClass.java:874)  
at java.io.ObjectInputStream.readOrdinaryObject(ObjectInputStream.java:2843)  
at java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1573)  
at java.io.ObjectInputStream.readObject(ObjectInputStream.java:431)  
at Test.main(Test.java:53)

D:\IOOperations>javac Test.java

D:\IOOperations>java Test  
Animal constructor called...  
Dog constructor called...  
Serialization started.....  
Serialization ended.....

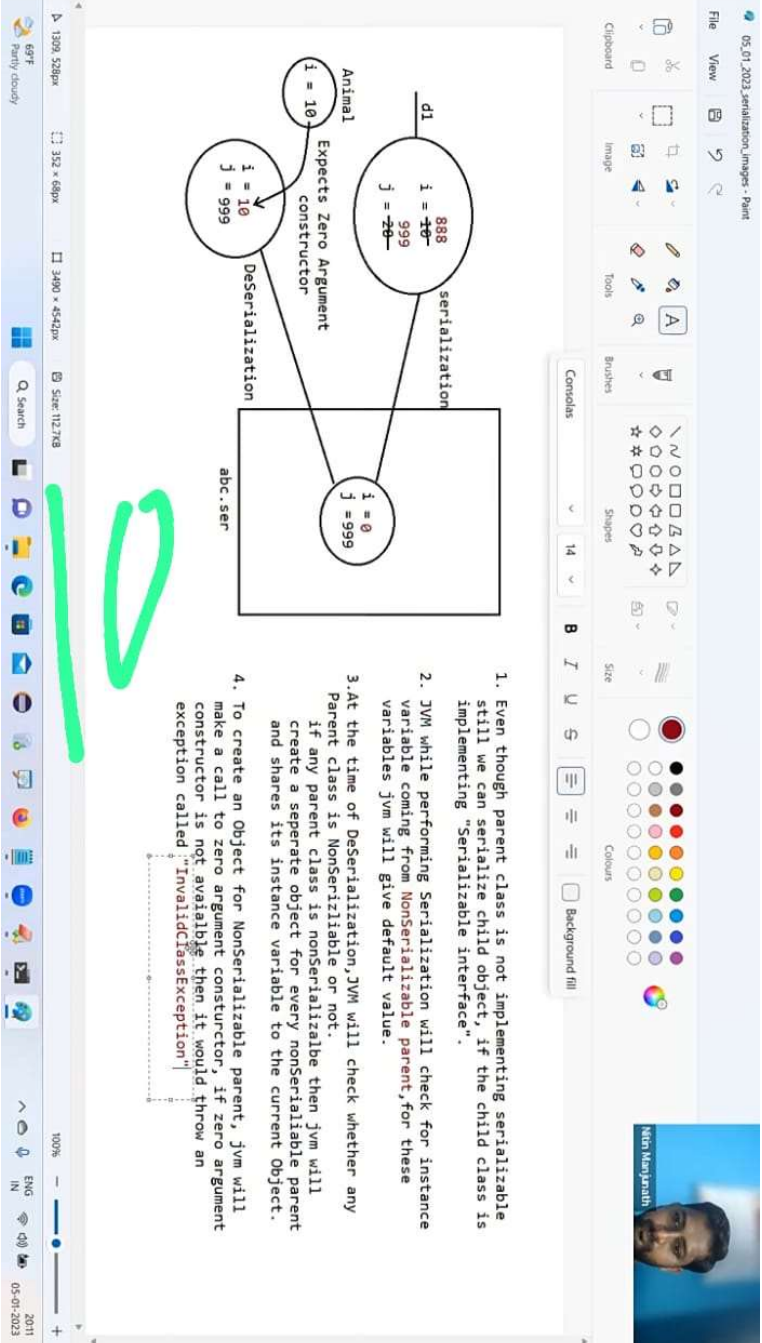
De-Serialization started.....  
Animal constructor called...  
10----->999  
De-Serialization ended.....

D:\IOOperations>



697  
Purvi Choudhary

Q Search  
697  
Purvi Choudhary  
2012  
05-07-2023



1. Even though parent class is not implementing serializable still we can serialize child object, if the child class implements `Serializable` interface.
2. JVM while performing Serialization will check for instance variable coming from `NonSerializableParent`, for these variables JVM will give default value.
3. At the time of Deserialization, JVM will check whether any parent class is `NonSerializable` or not.  
If any parent class is `nonserializable` then JVM will create a separate object for every `nonserializable` parent and shares its instance variable to the current Object.
4. To create an Object for `NonSerializable` parent, JVM will make a call to zero argument constructor, if zero argument constructor is not available then it would throw an exception called `"InvalidClassException"`.

**Nitin Mangayath**

**Nitin Mangayath**

**Nitin Mangayath**

**Nitin Mangayath**

**Nitin Mangayath**

**Nitin Mangayath**

**Nitin Mangayath**

**Nitin Mangrulkar**

**Nitin Mangrulkar**

- Nitin Mangrulkar**

**Nitin Mangrulkar**

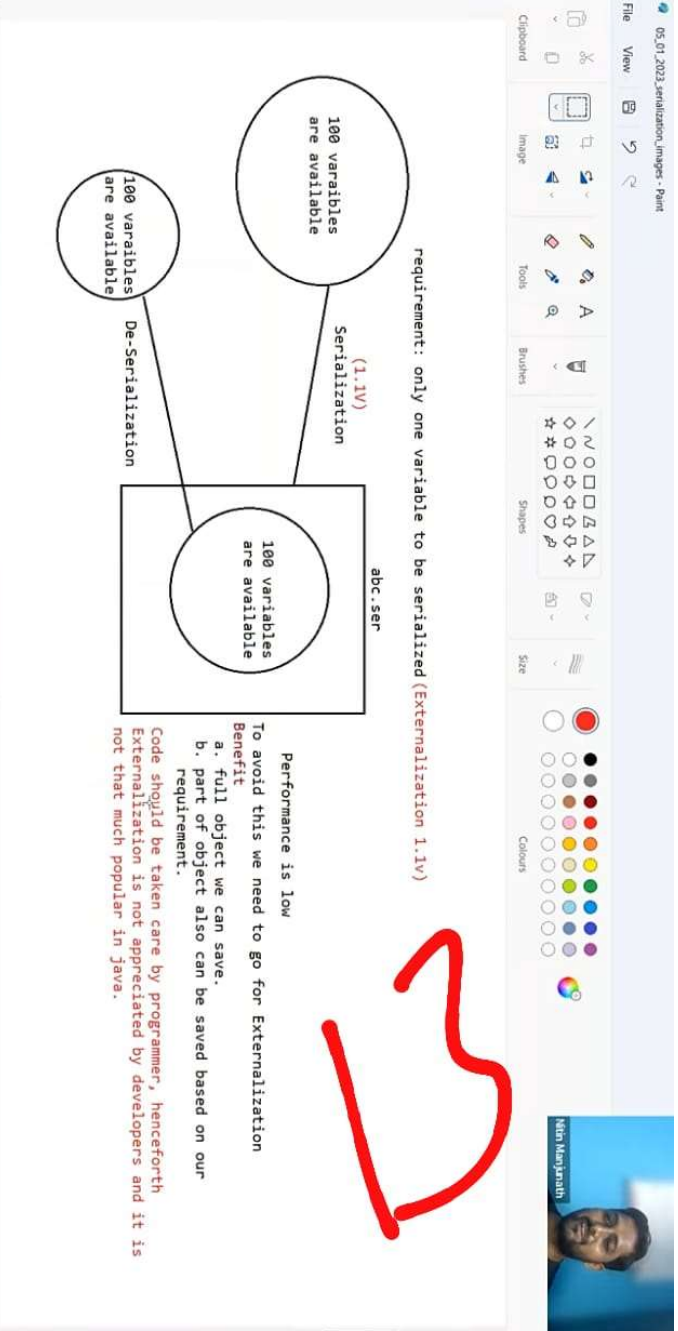
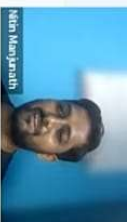
**Nitin Mangrulkar**

```
Command Prompt
X + >
D:\IOOperations>javap java.io.Serializable
Compiled from "Serializable.java"
public interface java.io.Serializable {
}
```

```
D:\IOOperations>javap java.io.Externalizable
Compiled from "Externalizable.java"
public interface java.io.Externalizable extends java.io.Serializable {
    public abstract void writeExternal(java.io.ObjectOutput) throws java.io.IOException;
    public abstract void readExternal(java.io.ObjectInput) throws java.io.IOException;
}
D:\IOOperations>
```

12





05-01-2023 serialization images - Part

File View

Clipboard

Image

Tools

Brushes

Shapes

Size

Colours

100 variables are available

De-Serialization

are available

Note

Serializable(I)

Marker Interface

extends

Externalizable(I)

```
public interface Externalizable extends Serializable {  
    public abstract void writeExternal(ObjectOutput) throws IOException;  
    public abstract void readExternal(ObjectInput) throws IOException, ClassNotFoundException;  
}  
//DeSerialization
```

Benefit

a. full object we can save.

b. part of object also can be saved based on our requirement.

Code should be taken care by programmer, henceforth Externalization is not appreciated by developers and it is not that much popular in java.

21

655 1567px

3400 x 4540px

Size: 1748KB

100%

ENG

20:36

05-01-2023

69F

Family cloud

Nan Manjiv



### Agenda :

1. Externalization
2. Difference between Serialization & Externalization
3. SerializableUID

### Externalization : ( 1.1 v )

1. In default serialization every thing takes care by JVM and programmer doesn't have any control.
2. In serialization total object will be saved always and it is not possible to save part of the object , which creates performance problems at certain point.
3. To overcome these problems we should go for externalization where every thing takes care by programmer and JVM doesn't have any control.
4. The main advantage of externalization over serialization is we can save either total object or part of the object based on our requirement.
5. To provide Externalizable ability for any object compulsory the corresponding class should implements externalizable interface.
6. Externalizable interface is child interface of serializable interface.

### Externalizable interface defines 2 methods :

1. writeExternal(OutputStream out ) throws IOException
2. readExternal(InputStream in) throws IOException, ClassNotFoundException



File Edit View



**Nita Manjari**

1. writeExternal(ObjectOutput out ) throws IOException

public void writeExternal(ObjectOutput out) throws IOException

public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException

### 3. SerializableUID

#### Externalization : ( 1.1 v )

1. In default serialization every thing takes care by JVM and programmer doesn't have any control.
2. In serialization total object will be saved always and it is not possible to save part of the object , which creates performance problems at certain point.
3. To overcome these problems we should go for externalization where every thing takes care by programmer and JVM doesn't have any control.
4. The main advantage of externalization over serialization is we can save either total object or part of the object based on our
5. To provide Externalizable ability for any object compulsory the corresponding class should implements externalizable interface.
6. Externalizable interface is child interface of serializable interface.

#### Externalizable interface defines 2 methods :

1. writeExternal(ObjectOutput out ) throws IOException
2. readExternal(ObjectInput in) throws IOException, ClassNotFoundException

public void writeExternal(ObjectOutput out) throws IOException



6. Externalizable interface is child interface of Serializable interface.

Externalizable interface defines 2 methods :

1. writeExternal(ObjectOutput out ) throws IOException
2. readExternal(ObjectInput in) throws IOException, ClassNotFoundException

public void writeExternal(ObjectOutput out) throws IOException

This method will be executed automatically at the time of Serialization with in this method , we have to write code to save required variables to the file .

public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException

This method will be executed automatically at the time of deserialization with in this method , we have to write code to save read required variable from file and assign to the current object.

I



05-01-2023, serialization.images - Paint

FileView

ClipboardImageToolsBrushesShapesSizeColours

d1

name = sachin  
i = 10  
j = 100

d2

name = sachin  
i = 0  
j = 0

abc.ser

name = sachin  
i = 10

Serialization

Deserialization

1. Object gets created using zero argument constructor  
if not available it would result in "InvalidClassException"

19

Nitin Manjrauthi

933 1993px

3400 x 4542px

Size 199.2X8

100%

ENG

2105

05-01-2023

05\_01\_2023\_serialization\_class\_notes - Notepad  
File Edit View

method , we have to write code to save read required variable from file and assign to the current object.

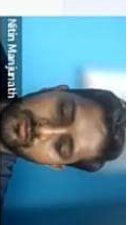
At the time of deserialization JVM will create a separate new object by executing public no-arg constructor on that object JVM will call readExternal() method.

Every Externalizable class should compulsory contains public no-arg constructor otherwise we will get RuntimeException saying "InvalidClassException" :

eg#1.

```
import java.io.Serializable;  
import java.io.FileOutputStream;  
import java.io.ObjectOutputStream;  
import java.io.InputStream;  
import java.io.FileInputStream;  
import java.io.IOException;  
import java.io.Externalizable;  
import java.io.ObjectOutput;  
import java.io.ObjectInput;
```

2





## Serialization

=====

1. It is meant for default Serialization
2. Here every thing takes care by JVM and programmer doesn't have any control.
3. Here total object will be saved always and it is not possible to save part of the object.
4. Serialization is the best choice if we want to save total object to the file.
5. relatively performance is low.
6. Serializable interface doesn't contain any method
7. It is a marker interface.
8. Serializable class not required to contains public no-arg constructor.
9. transient keyword play role in serialization

## Externalization

=====

1. It is meant for Customized Serialization
2. Here every thing takes care by programmer and JVM does not have any control.
3. Here based on our requirement we can save either total object or part of the object.
4. Externalization is the best choice if we want to save part of the object.

~



7. It is a marker interface.
8. Serializable class not required to contains public no-arg constructor.
9. transient keyword play role in serialization

#### Externalization

=====

1. It is meant for Customized Serialization
2. Here every thing takes care by programmer and JVM does not have any control.
3. Here based on our requirement we can save either total object or part of the object.
4. Externalization is the best choice if we want to save part of the object.
5. relatively performance is high
6. Externalizable interface contains 2 methods :
  1. writeExternal()
  2. readExternal()
7. It is not a marker interface.
8. Externalizable class should compulsory contains public no-arg constructor otherwise we will get RuntimeException saying "InvalidClassException" <sup>1</sup>
9. transient keyword don't play any role in Externalization.



~

Editor - (D:\Operations\testjava)

File Edit View Search Document Project Tools Browser Emmet Window Help

Directory: C:\Users\Nan\Documents\testjava

D:\Operations

```
7
8 public java.io.ObjectInputStream(java.io.InputStream) throws java.io.IOException;
9 public java.io.FileInputStream(java.lang.String) throws java.io.FileNotFoundException;
10 public final java.lang.Object readObject() throws java.io.IOException, java.lang.ClassN
11
12 */
13 class Demo implements Externalizable
14 {
15     transient String name;
16     int i;
17     int j;
18
19     public Demo(){
20         System.out.println("public zero argument constructor is called");
21     }
22
23     public Demo(String name,int i,int j){
24         this.name =name;
25         this.i = i;
26         this.j = j;
27     }
28 }
```

1 2 3 4 5 6 7

Nan Manjanna

testjava

testjava

For Help, press F1

69°F


Partly cloudy

Q Search

In 15 col 5 78 74 PC ANSI

21:30

05-07-2023



```
EdPlus - (D:\Operations\testjava)
File Edit View Search Document Project Tools Browser Emmet Window Help
Directory: C:\Users\...
D:\
IOOperations
testjava
25 this.i = i;
26 this.j = j;
27
28
29
30 //Write the logic of selective Serialization
31 public void writeExternal(ObjectOutput oo) throws IOException{
32     System.out.println("writeExternal() is called for Serialization....");
33
34     //variables need to participated write into abc.ser
35     oo.writeObject(name);
36     oo.writeInt(i);
37 }
38
39 // Write a logic of selective Deserialization
40 public void readExternal(ObjectInput oi) throws IOException,
41     ClassNotFoundException{
42     System.out.println("readExternal() is called for Deserialization....");
43
44     //variables need to retrieved from abc.ser
45     name = (String)oi.readObject();
46     i = oi.readInt();
47 }
```



24

Editor - (D:\Operations\test.java)

File Edit View Search Document Project Tools Browser Emmet Window Help

Directory: C:\Users\Niranjan\Documents\test.java

D:\Operations\test.java

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

1000

1

2

3

4

5

6

7

ClassNotFound

Exception

System.out.println("readExternal() is called for Deserialization.....");

//variables need to retrieved from abc.ser

name = (String)oi.readObject();

i = oi.readInt();

}

}

class Test

51 {

public static void main(String[] args) throws Exception

{

Demo d1 = new Demo("sachin",10,100);

System.out.println("Serialization started.....");

String fileName = "abc.ser";

FileOutputStream fos = new FileOutputStream(fileName);

ObjectOutputStream oos = new ObjectOutputStream(fos);

oos.writeObject(d1);

System.out.println("Serialization ended");

}

}

for help, press F1

69°F

Family Cloudy


Q Search

ln 38 col 49 78 00 PC ANSI

ENG IN

05-01-2023

Niranjan



Editor - (D:\Operations\testjava)

File Edit View Search Document Project Tools Browser Egitment Window Help

Directory: C:\Program Files\Java\jdk-11.0.10\bin

File Operations

TestJava

```
58 FileOutputStream fos = new FileOutputStream(fileName);
59 ObjectOutputStream oos = new ObjectOutputStream(fos);
60 oos.writeObject(d1);
61 System.out.println("Serialization ended.....");

//To pause the execution till we press some key from keyboard
System.in.read();

System.out.println("De-Serialization started.....");
FileInputStream fis = new FileInputStream("abc.ser");
ObjectInputStream ois = new ObjectInputStream(fis);
Demo d2 = (Demo)ois.readObject();

System.out.println(d2.name+"---->"+d2.i);
System.out.println("De-Serialization ended.....");
}
//JVM shutdown now
}
```

Command Prompt

in 38 col 49 78 00 PC ANSI

69F Party cloudy

Q Search

2102 05-01-2023

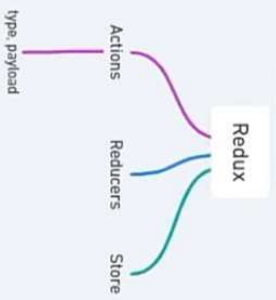
Niran Manjanna



```
D:\IOoperations>javac Test.java
D:\IOoperations>java Test
Serialization started.....
writeExternal() is called for Serialization....
Serialization ended.....
De-Serialization started.....
public zero argument constructor is called
readExternal() is called for DeSerialization....
sachin ----->10 ----->0
De-Serialization ended.....
D:\IOoperations>javac Test.java
D:\IOoperations>java Test
Serialization started.....
writeExternal() is called for Serialization....
Serialization ended.....
De-Serialization started.....
public zero argument constructor is called
readExternal() is called for DeSerialization....
sachin ----->10
De-Serialization ended.....
D:\IOoperations>
```

✓





Component

dispatch(action)

action(type, payload)

reducer

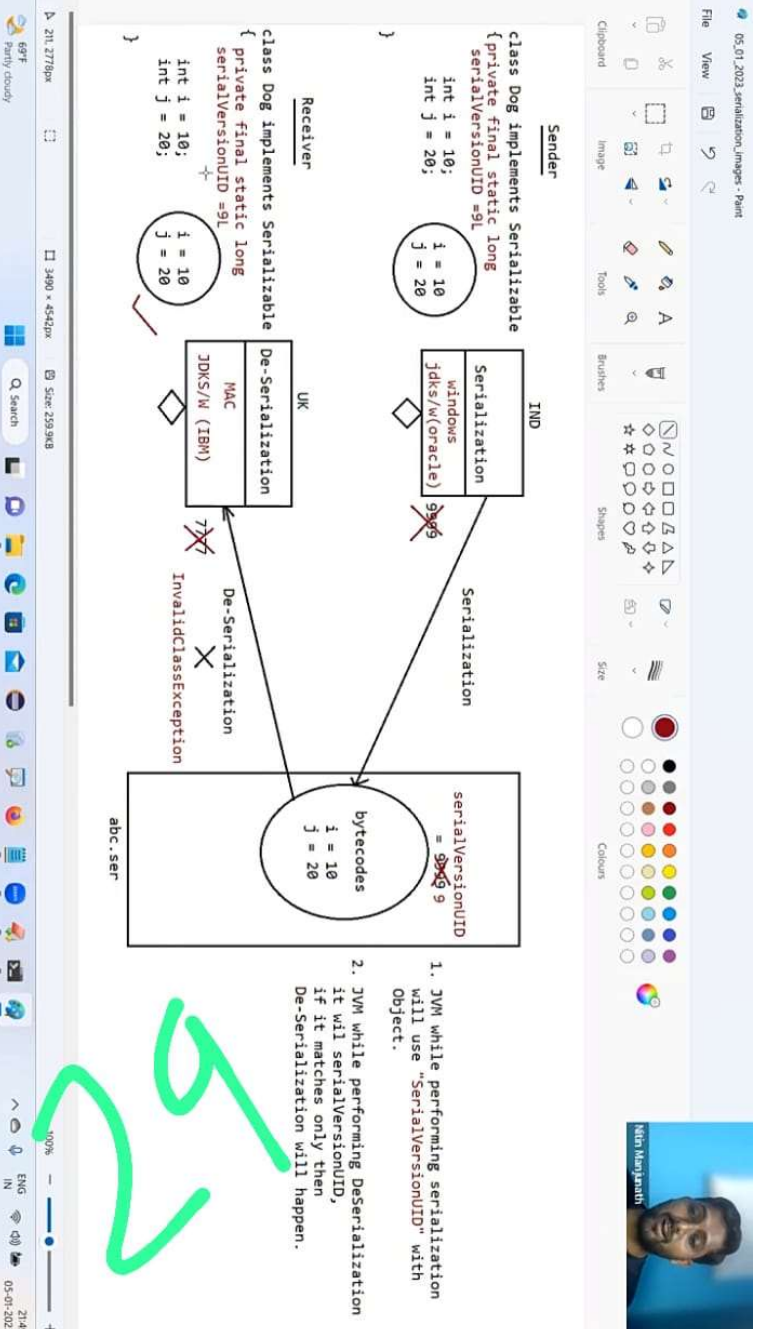
store (app state)

update state stored in store

application will render

28





File Edit View Search Document Project Tools Browser Emmet Window Help

Directory Clipboard Functions

D:\New Volume

D:\IOOperations

1 import java.io.\*;

2

3 class Dog implements Serializable

4 {

5 int i = 10;

6 int j = 20;

7 }

8 class RReceiver

9 {

10 public static void main(String[] args) throws Exception

11 {

12 System.out.println("De-Serialization started.....");

13 FileInputStream fis = new FileInputStream("abc.ser");

14 ObjectInputStream ois = new ObjectInputStream(fis);

15 Dog d2 = (Dog)ois.readObject();

16

17 System.out.println(d2.i+"----->"+d2.j);

18 System.out.println("De-Serialization ended.....");

19 }

20 }

21 }

TestJava

Java (\*.java)

Sender.java

RReceiver.java

For Help, press F1

68°F

Family Cloud

Q Search

In 18

col 63

21

09

PC

ANSI


ENG

IN

2200

05-06-2023

Nitin Mangrulkar



IDEA - D:\Operations\Sender.java 1

File Edit View Search Document Project Tools Browser Timeline Window Help

Directory Clipboard Functions

D:\ New Volume

D:\ Operations

```
1 import java.io.*;
2
3 class Dog implements Serializable
4 {
5     int i = 10;
6     int j = 20;
7
8 }
9 class Sender
10 {
11     public static void main(String[] args) throws Exception
12     {
13         Dog d1 = new Dog();
14
15         System.out.println("Serialization started.....");
16         String fileName = "abc.ser";
17         FileOutputStream fos = new FileOutputStream(fileName);
18         ObjectOutputStream oos = new ObjectOutputStream(fos);
19         oos.writeObject(d1);
20         System.out.println("Serialization ended.....");
21     }
22 }
```

Sender.java Receiver.java

D:\Operations\Sender.java

68°F Family cloudy

ln 7 col 1 23 00 PC ANSI ENG IN 2200 05-07-2023

Nitin Manjraiah

```
Command Prompt
X + ^
D:\IOOperations>java -version
java version "1.8.0_202"
Java(TM) SE Runtime Environment (build 1.8.0_202-b08)
Java HotSpot(TM) 64-Bit Server VM (build 25.202-b08, mixed mode)

D:\IOOperations>javac Sender.java

D:\IOOperations>java Sender
Serialization started.....
Serialization ended.....

D:\IOOperations>set path=C:\Program Files\Java\jdk-18.0.1\bin

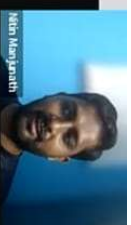
D:\IOOperations>java -version
java version "18.0.1" 2022-04-22
Java(TM) SE Runtime Environment (build 18.0.1.1+2-6)
Java HotSpot(TM) 64-Bit Server VM (build 18.0.1.1+2-6, mixed mode, sharing)

D:\IOOperations>javac RReceiver.java

D:\IOOperations>java RReceiver
De-Serialization started.....
10 ----->20
De-Serialization ended.....

D:\IOOperations>
```

25





Editor - (D:\Operations)\Receiver.java

File Edit View Search Document Project Tools Browser Emmet Window Help

Directory Clipboard Functions

D:\ New Volume

D:\ Operations

Test.java

```
1 import java.io.*;
2
3 class Dog implements Serializable
4 {
5     private static final long serialVersionUID = 9L;
6     int i = 10;
7     int j = 20;
8     int k = 30;
9 }
10 class RReceiver
11 {
12     public static void main(String[] args) throws Exception
13     {
14         System.out.println("De-Serialization started.....");
15         FileInputStream fis = new FileInputStream("abc.ser");
16         ObjectInputStream ois = new ObjectInputStream(fis);
17         Dog d2 = (Dog)ois.readObject();
18
19         System.out.println(d2.i+"---->"+d2.j);
20         System.out.println("De-Serialization ended.....");
21     }
22 }
```

Sender.java Receiver.java

For Help, press F1

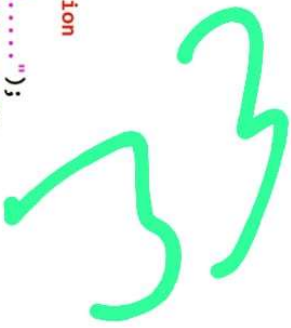
68°F Family Cloudy

Q Search

In 8 col 16 23 00 PC ANSI

ENG IN 2204 05-01-2023

Nitin Manjaram



## serialVersionUID

=====

=> To perform Serialization & Deserialization internally JVM will use a unique identifier, which is nothing but serialVersionUID.

=> At the time of serialization JVM will save serialVersionUID with object.

=> At the time of Deserialization JVM will compare serialVersionUID and if it is matched then only object will be Deserialized otherwise we will get RuntimeException saying "InvalidClassException".

The process in depending on default serialVersionUID are :

1. After Serializing object if we change the .class file then we can't perform deserialization because of mismatch in serialVersionUID of local class and serialized object in this case at the time of Deserialization we will get RuntimeException saying in "InvalidClassException".
2. Both sender and receiver should use the same version of JVM if there any incompatibility in JVM versions then receive anab
3. To generate serialVersionUID internally JVM will use complexAlgorithm which may create performance problems.

We can solve above problems by configuring our own serialVersionUID.

eg#1.

3x



File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer

TestApp.java

1 package in.neuron.main;

2

3 import java.io.Serializable;

4

5 class Student implements Serializable{

6

7     /\*\*

8     \*/

9     private static final long serialVersionUID = 1L;

10

11     String name = "sachin";

12     int age = 49;

13 }

14 }

15

16 public class TestApp {

17

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

19

20     }

21 }

22 }

23 }

24 }

68°F Family Cloudy

Search

Windows

Smart Input

13:18 (38)

ENG IN

22:13 05-07-2023

Nitin Manjunath

```
D:\TestApp>java ReceiverApp
10=====>20
```

```
D:\TestApp>java ReceiverApp
10====>20
```

=> In the above program after serialization even though if we perform any change to Dog class file we can deserialize object.  
=> We can configure our own serialVersionUID both sender and receiver not required to maintain the same JVM versions.  
Note : some IDE's generate explicit serialVersionUID.

Note : some IDE's generate explicit serialVersionUID.

36



8:49 AM

```
File Edit View Search Document Project Tools Browser Format Window Help
[Icons] New Volume
[Icons] IOOperations-
TestApp.java
1 import java.util.*;
2
3 class TestApp
4 {
5     public static void main(String[] args)
6     {
7         StringTokenizer stk = new StringTokenizer("sachinramesh$tendulkar", "$");
8         System.out.println(stk);
9         int tokenCount = stk.countTokens();
10        System.out.println(tokenCount);
11
12        while (stk.hasMoreTokens())
13        {
14            String data = stk.nextToken();
15            System.out.println(data);
16        }
17    }
18 }
19
20 }
21
```

68°F Family cloudy

Search

In 15 col 38 21 00 PC ANSI ENG IN 22:25 05-01-2023

Nitin Manjunath

LTE .111 LTE .111 B/s 85

35

File Edit View

=> In the above program after serialization even though if we perform any change to Dog.class file we can  
=> We can configure our own serialVersionUID both sender and receiver not required to maintain the same  
Note : some IDE's generate explicit serialVersionUID.

### Usage of StringTokenizer

=====

=> It is a part of java.util package

=> It is used to split the entire string into multiple tokens based on the delimiter we supply

eg: String data = "sachin ramesh tendulkar";

StringTokenizer stk = new StringTokenizer(data);

StringTokenizer stk = new StringTokenizer(data, " ");

=> public boolean hasMoreTokens()

=> public String nextToken()

import java.util.\*;

class TestApp {

public static void main(String[] args) {

StringTokenizer stk = new StringTokenizer("sachin\$ramesh\$tendulkar", "\$");

System.out.println(stk);

3

Ln 348, Col 30

67°F  
Partly cloudy

Search



100%

Windows (CTRL)

UTF-8

ENG IN 22:28 05-01-2023

