# Lab IO5: Serialization

## Objectives

In this lab, you will serialize and deserialize a Java Object

## Instructions

### Step 1: Create the runner class

1. Create a public class called Runner which will run the serialization process.
2. All this class should have is a main() method.

### Step 2: Create the Class to be serialized

1. In the same file, create a class with package visibility called Person which implements the Serializable interface
2. There are three instance variables as shown below
3. The instance variable id is marked as transient, which means it will be ignored in the serialization process
4. Define a static final long constant that represents the serialization UUID. It is important that this variable be names EXACTLY as shown in the code below.
5. Text

   Description automatically generatedThe toString() method is included for convenience.

### Step 3: Add the serialization code in the Runner class

1. Create the object to be serialized, in this case “bob”.
2. Text

   Description automatically generatedAlso create another reference variable, in this case called “otherbob,” which is initially set to null. This will hold the deserialized Person object.

### Step 4: Serialize and deserialize the object

1. Just like you did with the BufferedWriter, create a FileOutputStream object then wrap it in an ObjectOutputStream. This wrapper will manage all of the serialization of the object. It is conventional to use the file extension “.ser” to indicate the file contains a serialized object.

Text

Description automatically generated with medium confidence

1. And the same for the serialization code

Text

Description automatically generated

1. We have to cast otherbob to a Person object because Java doesn’t know what sort of object it is deserialization. All that it knows is that is some subclass of Object.
2. Text

   Description automatically generatedThe final code should look like this

### Step 5: Run the code

1. Graphical user interface, text, application

   Description automatically generatedRun the code and you should see that bob and otherbob are the same except for the id value. The id was marked as transient to it was never serialized. When otherbob was deserialized, since there was no provided value for id, that variable had the default value of zero.

### Step 6: Invalidate the serizalized object.

1. Run the code so that you have a serialized object stored on disk.
2. Text

   Description automatically generatedComment out the serialization code so that only the deserialization code will execute.
3. Now change the serialization UUID to any other value.

Text

Description automatically generated with medium confidence

1. Rerun the code. Because the serialized object has a different UUID, you should get an exception being thrown.

Person [name=Bob, age=34, id=9989]

java.io.InvalidClassException: iolab.Person; local class incompatible: stream classdesc serialVersionUID = 1, local class serialVersionUID = 8879