ANY CLASS THAT IMPLEMENTS SERIALIZABLE, AND IN WHICH ALL MEMBER VARIABLES ARE SERIALIZABLE CAN BE WRITTEN TO FILE OR READ FROM FILE WITH MINIMAL EFFORT

Serializable

INTERFACE

THE SERIALIZABLE INTERFACE HAS 2 METHODS - READOBJECT AND WRITEOBJECT - BUT YOU TYPICALLY DON'T NEED TO IMPLEMENT THEM YOURSELF IN ORDER TO MAKE YOUR OBJECT SERIALIZABLE

IMPLEMENTING SERIALIZABLE

```
public class MyLittleClass implements Serializable {

    private String string;
    private List<String> list = new ArrayList<>();
    private transient HTMLWriter htmlWriter = new HTMLWriter();
```

SO LONG AS ALL MEMBER VARIABLES IMPLEMENT SERIALIZABLE (AND THE BUILT-IN CLASSES ALL DO), JUST CALLING OUT THAT THE CLASS IMPLEMENTS SERIALIZABLE IS ENOUGH - JAVA WILL DO THE REST

SERIALIZING AN OBJECT

```
MyLittleClass myLittleClass = new MyLittleClass();
List<String> list = new ArrayList<>();
list.add("Athos");
list.add("Pothos");
list.add("Aramis");
myLittleClass.setList(list);
myLittleClass.setString("The three musketeers");

FileOutputStream fileOut =
        new FileOutputStream("MyLittleClass.obj");
ObjectOutputStream out = new ObjectOutputStream(fileOut);
out.writeObject(myLittleClass);
out.close();
fileOut.close();
```

DESERIALIZING AN OBJECT

IS NOT THAT COMPLICATED EITHER

```
FileInputStream fileIn = new FileInputStream("MyLittleClass.obj");
ObjectInputStream in = new ObjectInputStream(fileIn);
MyLittleClass myLittleClass1 = (MyLittleClass) in.readObject();
in.close();
fileIn.close();

System.out.println(myLittleClass1.getString());
```

SERIALIZATION HAS BECOME A STANDARD
PART OF MOST PROGRAMMING LANGUAGES..

BUT BACK IN THE DAY, SAVING AN
OBJECT TO FILE TOOK A GREAT DEAL
OF WORK FROM PROGRAMMERS -

SERIALIZING AND DE-SERIALIZING MEANT
LAYING OUT THE OBJECT, 1 MEMBER VARIABLE
AT A TIME

EVEN NOW, SERIALIZATION CAN GET TRICKY
AS YOU ADD MEMBER VARIABLES ETC
CLASSES.