**Writable Interface Introduction**

It is an Inteface that is used for wrapping the primitive data types like int so as to provide Serialisation for Datatype

hadoop comes with several wrappers around primitive types and widely used classes in Java:

|  |  |
| --- | --- |
| Java primitive | Writable implementation |
| Boolean | BooleanWritable |
| Byte | ByteWritable |
| Short | ShortWritable |
| Int | IntWritable VIntWritable |
| Float | FloatWritable |
| Long | LongWritable VLongWritable |
| Double | DoubleWritable |

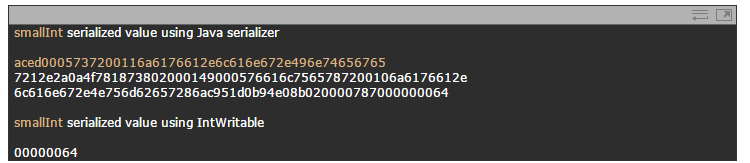
**Definition Of Serialisation**

Serialization is nothing but converting the raw data into a stream of bytes which can travel along different networks and can reside in different systems

Since Hadoop deals with large volume of data and the data is to transmitted across the framework from mappers to reducer .A new Interface was developed writable which performs Serialisatiobn and Deserialisation (conversion from and to Bytes) as it improves the efficiency and Speed of Transmission

**Importance Of Writable And Its Edge Over Serialisation Object Of Java**

Serialization is important in Hadoop because it enables easy transfer of data. If Writable is not present in Hadoop, then it uses the serialization of Java which increases the data over-head in the network as shown in fig.



This shows the clear difference between serialization in Java and Hadoop and also the difference between ObjectInputStream and Writable interface. If the size of serialized data in Hadoop is like that of Java, then it will definitely become an overhead in the network.

Also the core part of Hadoop framework i.e., shuffle and sort phase won’t be executed without using Writable.

**Methods Used By Writable Tio Achieve Serialisation**

**1.void readFields(DataInput in);**

It is used for readimg data from the stream(transmission channel) and writing it into local disk. The wrappers we saw above just send and receive their binary representation over a stream.

**2. void write(DataOutput out);**

 write() is used for writing the data onto the stream, The wrappers we saw above just send and receive their binary representation over a stream.



**DrawBack**

In this method if we implement writable as there is no compare method to sort the key value pairs in the frame work but if it is a default intwritable a raw comparator will take care of this sorting

For this they introduced WritableComparable which can implemented to sort the keys

**WritableComparable:**It is an sub interface that extends writable and comparable that uses for serialization and deserialization of writable Interface and compareto method of Comparable class

It has 3 Methods

**1.void readFields(DataInput in);**

It is used for readimg data from the stream(transmission channel) and writing it into local disk. The wrappers we saw above just send and receive their binary representation over a stream.

**2. void write(DataOutput out);**

 write() is used for writing the data onto the stream, The wrappers we saw above just send and receive their binary representation over a stream.

**3. int compareTo(WritableComparable o):**it is used for comparing the objects

Example



**Conclusion**

When we write a key as IntWritable in the Mapper class and send it to the reducer class, there is an intermediate phase between the Mapper and Reducer class i.e., shuffle and sort, where each key has to be compared with many other keys. If the keys are not comparable, then shuffle and sort phase won’t be executed or may be executed with high amount of overhead.

If a key is taken asIntWritable by default, then it has comparable feature because of RawComparator acting on that variable. It will compare the key taken with the other keys in the network. This cannot take place in the absence of Writable.