Explain in brief Writable and Writable Comparable in Hadoop with an example.

**Writable:**

**Writable** in an interface in Hadoop and types in Hadoop must implement this interface. Hadoop provides these writable wrappers for almost all Java primitive types and some other types,but sometimes we need to pass custom objects and these custom objects should implement Hadoop's Writable interface.Hadoop MapReduce uses implementations of Writables for interacting with user-provided Mappers and Reducers.

E.g:

**Public interface Writable{**

**Void readFields(DataInput in);**

**Void write(DataOutput out);**

**}**

As we already know, data needs to be transmitted between different nodes in a distributed computing environment. This requires serialization and deserialization of data to convert the data that is in structured format to byte stream and vice-versa. Hadoop therefore uses simple and efficient serialization protocol to serialize data between map and reduce phase and these are called Writable(s). Some of the examples of writables as already mentioned before are IntWritable, LongWritable, BooleanWritable and FloatWritable.

**WritableComparable;**

**WritableComparable** interface is just a subinterface of the Writable and java.lang.Comparable interfaces. For implementing a WritableComparable we must have compareTo method apart from readFields and write methods

E.g;

**Public interface WritableComparable extends Writable,comparable**

**{**

**Void readFields(Datainput in);**

**Void write(Dataoutput out);**

**Int compareTo(WritableComparable o)}**

**Note:**

1)WritableComparable can be compared to each other ,typically via Comparators.Any type which is to be used as a key in the Hadoop Map-Reduce framework should implement this interface.

2)Any type which is to be used as value in the Hadoop Map-Reduce framework should implement the writable interface

In short ,the type used as key in Hadoop must be a writableComparable,while value could be just a Writable.