**Evaluation**

***Overview***

To evaluate the performance of the human-readable and binary data interchange formats we wanted to comprehensively test the application by running it with data of different sizes. The idea here was to gather results from testing the application with different data sizes and then compare and analyse the results.

From here we could then conclude which human-readable format (JSON, CSV, XML, Array List and Hash Map) was the most efficient at representing the data. Depending on which one of those types was the most efficient for the given human-readable format we could finally serialize that data with all six of the binary formats.

The binary format that was most efficient or serialized the data to the smallest size in terms of bytes would obviously be the best to use. After the data was serialized to the smallest size possible the application could then for e.g. transfer the data in binary form across a network or pass it to a message bus.

***Detailed Evaluation***

The size of the data used in testing and evaluating the application was grouped into three different categories, small, medium and large. Each category has the same data represented in the five different human readable formats JSON, CSV, XML, Array List and Hash Map. For example, the data used in the small data size are shown below in all five human-readable formats.

**Small JSON data file:**

{"employees":[

{"firstName":"John", "lastName":"Doe"},

{"firstName":"Anna", "lastName":"Smith"},

{"firstName":"Peter", "lastName":"Jones"}

]}

**Small CSV data file:**

firstName lastName

John Doe

Anna Smith

Peter Jones

**Small XML data file:**

<root>

<employees>

<firstName>John</firstName>

<lastName>Doe</lastName>

</employees>

<employees>

<firstName>Anna</firstName>

<lastName>Smith</lastName>

</employees>

<employees>

<firstName>Peter</firstName>

<lastName>Jones</lastName>

</employees>

</root>

**Small ArrayList of data:**

[firstName,lastName, John,Doe, Anna,Smith, Peter,Jones]

**Small HashMap of data:**

Key = 0, Value = firstName,lastName,

Key = 1, Value = John,Doe,

Key = 2, Value = Anna,Smith,

Key = 3, Value = Peter,Jones,

By just looking at all five human-readable formats above you can see XML has the largest size whereas CSV and Array List are the smallest. So to back this up with some evidence we needed to go through each human-readable format one at a time, serialize the data with all six binary serialization formats record the results and then repeat this for all the different data size categories of the given format.

So for example, if we take the JSON format, we first convert the JSON data into a Java Array List and serialize this list with all six binary serialization formats, then record the results. Next would be to convert the JSON data to a Java Hash Map and serialize this map with all six serialization formatters and again record the results and finally serialize the JSON string with all six binary formatters and again record the result. The tables below show the results of this evaluation for all five human-readable formats in all data size categories.

Results serialising **small data files** which contain the same data in different formats.

**Serializing a small JSON file: File Size: 157 Bytes**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** |  | **XML** | **CSV** |  |
| Message Pack | 137 Bytes | 115 Bytes | 160 Bytes |  | 343 Bytes | 54 Bytes |  |
| BSON | 146 Bytes | 164 Bytes | 162 Bytes |  | 345 Bytes | 56 Bytes |  |
| KRYO | 160 Bytes | 157 Bytes | 160 Bytes |  | 343 Bytes | 52 Bytes |  |
| CBOR | 136 Bytes | — | 159 Bytes |  | 343 Bytes | 53 Bytes |  |
| Hessian | 140 Bytes | 138 Bytes | 163 Bytes |  | 346 Bytes | 57 Bytes |  |
| JAVA | 194 Bytes | 365 Bytes | 164 Bytes |  | 347 Bytes | 58 Bytes |  |

**Serializing a small CSV file: File Size: 53 Bytes**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** |  | **XML** | **CSV** |  |
| Message Pack | 60 Bytes | 125 Bytes | 173 Bytes |  | 363 Bytes | 56 Bytes |  |
| BSON | 150 Bytes | 144 Bytes | 175 Bytes |  | 365 Bytes | 58 Bytes |  |
| KRYO | 92 Bytes | 123 Bytes | 173 Bytes |  | 363 Bytes | 54 Bytes |  |
| CBOR | 60 Bytes | — | 172 Bytes |  | 363 Bytes | 55 Bytes |  |
| Hessian | 65 Bytes | 80 Bytes | 176 Bytes |  | 366 Bytes | 59 Bytes |  |
| JAVA | 144 Bytes | 235 Bytes | 177 Bytes |  | 367 Bytes | 60 Bytes |  |

**Serializing a small XML file: File size: 338 Bytes**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** |  | **XML** | **CSV** |  |
| Message Pack | 146 Bytes | 122 Bytes | 239 Bytes |  | 341 Bytes | 54 Bytes |  |
| BSON | 155 Bytes | 175 Bytes | 241 Bytes |  | 343 Bytes | 56 Bytes |  |
| KRYO | 169 Bytes | 167 Bytes | 239 Bytes |  | 341 Bytes | 52 Bytes |  |
| CBOR | 145 Bytes | — | 238 Bytes |  | 341 Bytes | 53 Bytes |  |
| Hessian | 149 Bytes | 146 Bytes | 242 Bytes |  | 344 Bytes | 57 Bytes |  |
| JAVA | 203 Bytes | 398 Bytes | 243 Bytes |  | 345 Bytes | 58 Bytes |  |

**Serializing a small Java Hash Map:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** |  | **XML** | **CSV** |  |
| Message Pack | 60 Bytes | 125 Bytes | 176 Bytes |  | 363 Bytes | 57 Bytes |  |
| BSON | 150 Bytes | 144 Bytes | 178 Bytes |  | 365 Bytes | 59 Bytes |  |
| KRYO | 92 Bytes | 123 Bytes | 176 Bytes |  | 363 Bytes | 55 Bytes |  |
| CBOR | 60 Bytes | — | 175 Bytes |  | 363 Bytes | 56 Bytes |  |
| Hessian | 65 Bytes | 80 Bytes | 179 Bytes |  | 366 Bytes | 60 Bytes |  |
| JAVA | 144 Bytes | 235 Bytes | 180 Bytes |  | 367 Bytes | 61 Bytes |  |

**Serializing a small Java Array List:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** |  | **XML** | **CSV** |  |
| Message Pack | 60 Bytes | 123 Bytes | 233 Bytes |  | 363 Bytes | 57 Bytes |  |
| BSON | 150 Bytes | 144 Bytes | 235 Bytes |  | 365 Bytes | 59 Bytes |  |
| KRYO | 92 Bytes | 123 Bytes | 233 Bytes |  | 363 Bytes | 55 Bytes |  |
| CBOR | 60 Bytes | — | 232 Bytes |  | 363 Bytes | 56 Bytes |  |
| Hessian | 65 Bytes | 80 Bytes | 236 Bytes |  | 366 Bytes | 60 Bytes |  |
| JAVA | 144 Bytes | 235 Bytes | 237 Bytes |  | 367 Bytes | 61 Bytes |  |

As can be clearly seen from the graph representation of the result relating to the small data size category, the CSV format can represent the same data as the other formats but in a fraction of the size compared to XML and JSON.

Results serialising **medium data files** which contain the same data in different formats.

**Serializing a medium JSON file: File size: 266,917 Bytes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** | **XML** | **CSV** |
| Message Pack | 196702 Bytes | 170859 Bytes | 266922 Bytes | 373928 Bytes | 59796 Bytes |
| BSON | 196709 Bytes | 207294 Bytes | 266922 Bytes | 373928 Bytes | 59798 Bytes |
| KRYO | 196724 Bytes | 87271 Bytes | 266921 Bytes | 373927 Bytes | 59797 Bytes |
| CBOR | 196702 Bytes | — | 266922 Bytes | 373928 Bytes | 59796 Bytes |
| Hessian | 196721 Bytes | 162533 Bytes | 266948 Bytes | 373963 Bytes | 59803 Bytes |
| JAVA | 196763 Bytes | 126966 Bytes | 266930 Bytes | 373936 Bytes | 59800 Bytes |

**Serializing a medium CSV file: File size:** **60,314 Bytes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** | **XML** | **CSV** |
| Message Pack | 60320 Bytes | 8500 Bytes | 240294 Bytes | 373462 Bytes | 60317 Bytes |
| BSON | 148292 Bytes | 128856 Bytes | 240294Bytes | 373462 Bytes | 60319 Bytes |
| KRYO | 69543 Bytes | 62646 Bytes | 240293 Bytes | 373461 Bytes | 60318 Bytes |
| CBOR | 60320 Bytes | — | 240294 Bytes | 373462 Bytes | 60317 Bytes |
| Hessian | 60325 Bytes | 61530 Bytes | 240317 Bytes | 373497 Bytes | 60324 Bytes |
| JAVA | 78452 Bytes | 81279 Bytes | 240302 Bytes | 373470 Bytes | 60321 Bytes |

**Serializing a medium XML file: File size: 373,459 Bytes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** | **XML** | **CSV** |
| Message Pack | 175456 Bytes | 1665271 Bytes | 264199 Bytes | 373464 Bytes | 54202 Bytes |
| BSON | 175463 Bytes | 168557 Bytes | 264199 Bytes | 373464 Bytes | 54204 Bytes |
| KRYO | 175478 Bytes | 73345 Bytes | 264198 Bytes | 373463 Bytes | 54203 Bytes |
| CBOR | 175456 Bytes | — | 264199 Bytes | 373464 Bytes | 54202 Bytes |
| Hessian | 175473 Bytes | 143457 Bytes | 264225 Bytes | 373499 Bytes | 54209 Bytes |
| JAVA | 175517 Bytes | 142627 Bytes | 264207 Bytes | 373472 Bytes | 54206 Bytes |

**Serializing a medium Java Hash Map:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** | **XML** | **CSV** |
| Message Pack | 60320 Bytes | 8509 Bytes | 248646 Bytes | 373462 Bytes | 68181 Bytes |
| BSON | 148292 Bytes | 128856 Bytes | 248646 Bytes | 373462 Bytes | 68181 Bytes |
| KRYO | 69543 Bytes | 62646 Bytes | 248645 Bytes | 373461 Bytes | 68180 Bytes |
| CBOR | 60320 Bytes | — | 248646 Bytes | 373462 Bytes | 68181 Bytes |
| Hessian | 60325 Bytes | 61530 Bytes | 248669 Bytes | 373497 Bytes | 68189 Bytes |
| JAVA | 78452 Bytes | 81279 Bytes | 2485654 Bytes | 373470 Bytes | 68189 Bytes |

**Serializing a medium Java Array List:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** | **XML** | **CSV** |
| Message Pack | 60320 Bytes | 8500 Bytes | 264199 Bytes | 373462 Bytes | 60057 Bytes |
| BSON | 148292 Bytes | 128856 Bytes | 264199 Bytes | 373462 Bytes | 60059 Bytes |
| KRYO | 69543 Bytes | 62646 Bytes | 264198 Bytes | 373461 Bytes | 60058 Bytes |
| CBOR | 60320 Bytes | — | 264199 Bytes | 373462 Bytes | 60057 Bytes |
| Hessian | 60325 Bytes | 61530 Bytes | 264225 Bytes | 373497 Bytes | 60064 Bytes |
| JAVA | 78452 Bytes | 81279 Bytes | 264207 Bytes | 373470 Bytes | 60061 Bytes |

As can be seen from the graph above the results from the medium data size are interesting. The JSON format is getting worse as the data size is increased and it is almost as bad as XML. Again the CSV format is the clear winner here with ArrayList in second and HashMap not far behind.

Results serializing **large data files** which contain the same data in different formats.

**Serializing a large JSON file: File size: 27,888,867 Bytes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** | **XML** | **CSV** |
| Message Pack | 22060863 Bytes | 19190354 Bytes | 27888867 Bytes | 41748475 Bytes | 6809875 Bytes |
| BSON | 22060870 Bytes | 23296539 Bytes | 27888867 Bytes | 41748475 Bytes | 6809875 Bytes |
| KRYO | 22060886 Bytes | 9868007 Bytes | 27888867 Bytes | 41748475 Bytes | 6809875 Bytes |
| CBOR | 22060863 Bytes | — | 27888867 Bytes | 41748475 Bytes | 6809875 Bytes |
| Hessian | 22062844 Bytes | 18262541 Bytes | 27891422 Bytes | 41752299 Bytes | 6810498 Bytes |
| JAVA | 22060924 Bytes | 14261681 Bytes | 27888875 Bytes | 41748483 Bytes | 6809883 Bytes |

**Serializing a large CVS file: File size: 6,838,864 Bytes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** | **XML** | **CSV** |
| Message Pack | 6867867 Bytes | 999795 Bytes | 26903033 Bytes | 41690541 Bytes | 6838869 Bytes |
| BSON | 18587125 Bytes | 14511701 Bytes | 26903033 Bytes | 41690541 Bytes | 6838869 Bytes |
| KRYO | 7896367 Bytes | 7189378 Bytes | 26903033 Bytes | 41690541 Bytes | 6838869 Bytes |
| CBOR | 6867867 Bytes | — | 26903033 Bytes | 41690541 Bytes | 6838869 Bytes |
| Hessian | 6867872 Bytes | 7059737 Bytes | 26905497 Bytes | 41694359 Bytes | 6839495 Bytes |
| JAVA | 8868643 Bytes | 9234536 Bytes | 26903041 Bytes | 41690549 Bytes | 6838879 Bytes |

**Serializing a large XML file: File size: 41,690,538 Bytes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** | **XML** | **CSV** |
| Message Pack | 19766909 Bytes | 18636098 Bytes | 29625212 Bytes | 41690543 Bytes | 6255613 Bytes |
| BSON | 19766916 Bytes | 18857947 Bytes | 29625212 Bytes | 41690543 Bytes | 6255613 Bytes |
| KRYO | 19766932 Bytes | 8271632 Bytes | 29625212 Bytes | 41690543 Bytes | 6255613 Bytes |
| CBOR | 19766909 Bytes | — | 29625212 Bytes | 41690543 Bytes | 6255613 Bytes |
| Hessian | 19768720 Bytes | 16081524 Bytes | 29627926 Bytes | 41694361 Bytes | 6256185 Bytes |
| JAVA | 19766970 Bytes | 15983147 Bytes | 29625220 Bytes | 41690551 Bytes | 6255621 Bytes |

**Serializing a large Java Hash Map:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** | **XML** | **CSV** |
| Message Pack | 6867867 Bytes | 999798 Bytes | 27830873 Bytes | 41690541 Bytes | 7737746 Bytes |
| BSON | 18587125 Bytes | 14511701Bytes | 27830873 Bytes | 41690541 Bytes | 7737746 Bytes |
| KRYO | 7896367 Bytes | 7189378 Bytes | 27830873 Bytes | 41690541 Bytes | 7737746 Bytes |
| CBOR | 6867867 Bytes | — | 27830873 Bytes | 41690541 Bytes | 7737746 Bytes |
| Hessian | 6867872 Bytes | 7059737 Bytes | 27833422 Bytes | 41694359 Bytes | 7738456 Bytes |
| JAVA | 8868643 Bytes | 9234536 Bytes | 27830881 Bytes | 41690549 Bytes | 7737754 Bytes |

**Serializing a large Java Array List:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Binary Serialization** | **Array List** | **Hash Map** | **JSON** | **XML** | **CSV** |
| Message Pack | 6867867 Bytes | 999795 Bytes | 29625212 Bytes | 41690541 Bytes | 6838870 Bytes |
| BSON | 18587125 Bytes | 14511701 Bytes | 29625212 Bytes | 41690541 Bytes | 6838870 Bytes |
| KRYO | 7896367 Bytes | 7189378 Bytes | 29625212 Bytes | 41690541 Bytes | 6838870 Bytes |
| CBOR | 6867867 Bytes | — | 29625212 Bytes | 41690541 Bytes | 6838870 Bytes |
| Hessian | 6867872 Bytes | 7059737 Bytes | 29627926 Bytes | 41694359 Bytes | 6839496 Bytes |
| JAVA | 8868643 Bytes | 9234536 Bytes | 29625220 Bytes | 41690549 Bytes | 6838878 Bytes |

As can be seen from the graph, the results from the large data are interesting. By comparing the three graphs from the different data category sizes we can clearly see as the monitoring data increases in size the HashMap format gets closer and closer to the performance of CSV. If the application was tested with even larger data my guess is the performance of the HashMap format would get even closer and might even surpass the performance of CSV.

***Results***

Based on the results contained in the tables above we have concluded that the best human-readable format is a Java hash Map which when converted to a Java Array List before being serialized produces data in the smallest possible form. This result can clearly be verified from the tables above. If we look at the results for the Java Hash Map format in all three different data size categories we can see that when the Hash Map is converted to an ArrayList it consistently produces the smallest data size across all data size categories.

After confirming that a Java Hash Map when converted to a Java Array List was the best human-readable format in terms of efficiency and data size in bytes we needed to choose which of the binary serialization formats could actually serialize the Array List to the smallest size in bytes. After looking at the serialization results for the binary formats the one that consistently produced the smallest data was Message pack. By looking at the tables for the Java Hash Map in the three different data size categories we can confirm this result.