Test Platform

OS:Mac OS X JVM:Oracle Corporation 1.8.0_91 CPU:2.8 GHz Intel Core i7 os-arch:Darwin Kernel Version 15.5.0 Cores (incl HT):8

Disclaimer

This test focusses on en/decoding of a cyclefree data structure, but the featureset of the libraries compared differs a lot:

- some serializers support cycle detection/object sharing others just write non-cyclic tree structures
- some include full metadata in serialized output, some don't
- some are cross platform, some are language specific
- some are text based, some are binary,
- some support versioning forward/backward, both, some don't

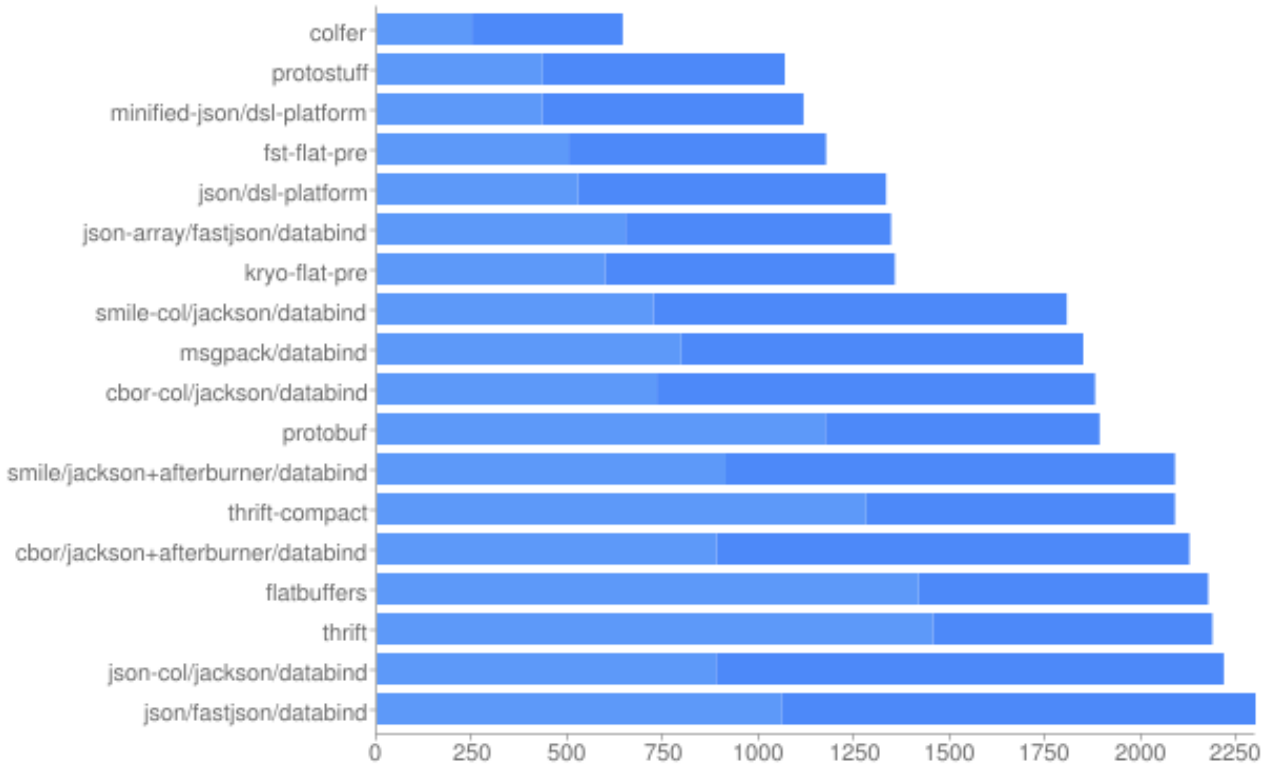
(See "ToolBehavior":wiki/ToolBehavior) Other test data will yield different results (e.g. adding a non ascii char to every string :-)). However the results give a raw estimation of library performance.

Serializers (no shared refs)

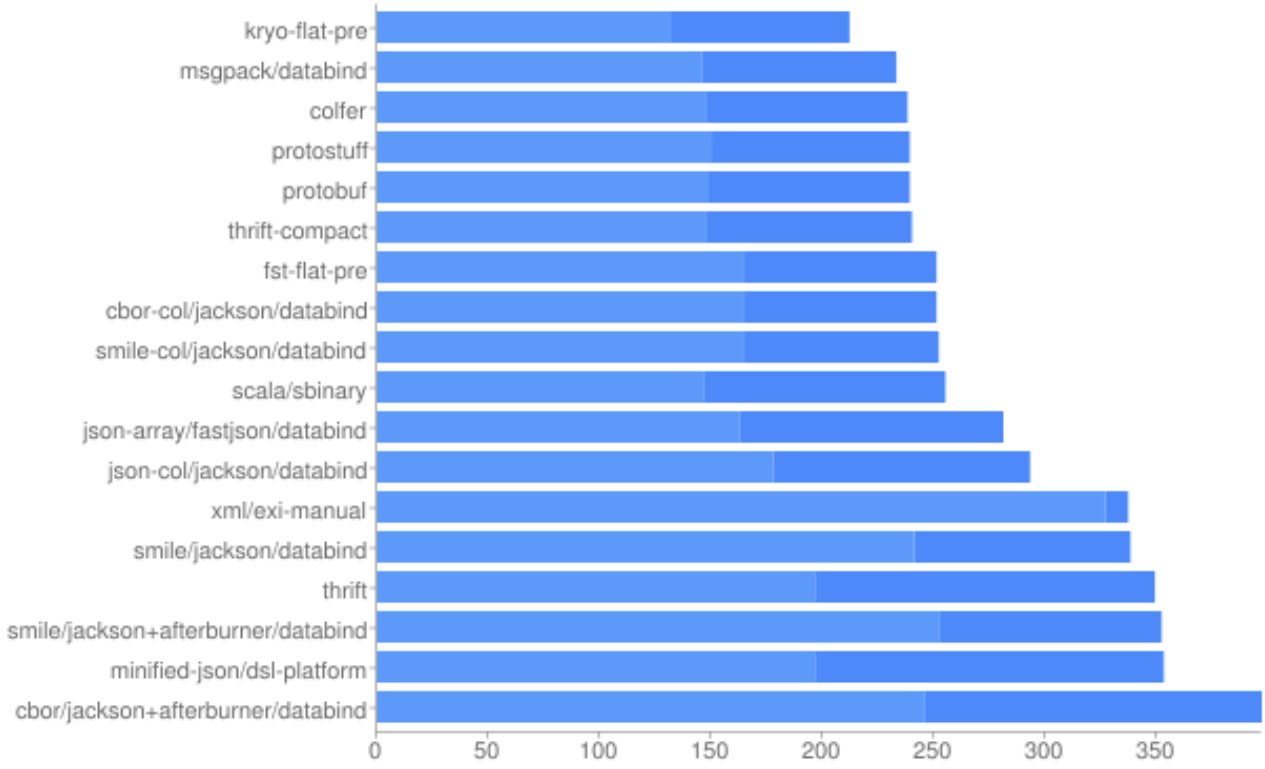
Benchmarks serializers

- Only cycle free tree structures. An object referenced twice will be serialized twice.
- no manual optimizations.
- schema is known in advance (pre registration or even class generation). (Not all might make use of that)

Ser Time+Deser Time (ns)



Size, Compressed size [light] in bytes



	create	ser	deser	total	size
+dfl					
colfer	49	248	396	643	238
148					
protostuff	68	433	634	1067	239
150					
minified-json/dsl-platform	46	432	684	1116	353

197					
fst-flat-pre	53	501	675	1175	251
165					
json/dsl-platform	44	526	806	1332	485
261					
json-array/fastjson/databind	53	650	696	1345	281
163					
kryo-flat-pre	54	597	758	1355	212
132					
smile-col/jackson/databind	54	723	1082	1805	252
165					
msgpack/databind	54	796	1052	1848	233
146					
cbor-col/jackson/databind	54	732	1147	1879	251
165					
protobuf	121	1173	719	1891	239
149					
smile/jackson+afterburner/databind	54	913	1175	2088	352
252					
thrift-compact	97	1280	808	2088	240
148					
cbor/jackson+afterburner/databind	53	888	1239	2126	397
246					
flatbuffers	56	1417	758	2175	432
226					
thrift	95	1455	731	2186	349
197					
json-col/jackson/databind	53	887	1329	2216	293
178					
json/fastjson/databind	53	1058	1241	2299	486
262					
smile/jackson/databind	53	1011	1300	2311	338
241					
scala/sbinary	442	1311	1069	2381	255
147					
capnproto	55	1574	979	2553	400
204					
json/jackson+afterburner/databind	52	1094	1489	2584	485
261					
cbor/jackson/databind	53	1023	1561	2585	397
246					
json/protostuff-runtime	54	1353	1632	2986	469
243					
json/jackson/databind	54	1164	1866	3030	485
261					
json/jackson-jr/databind	53	1426	1962	3389	468
255					

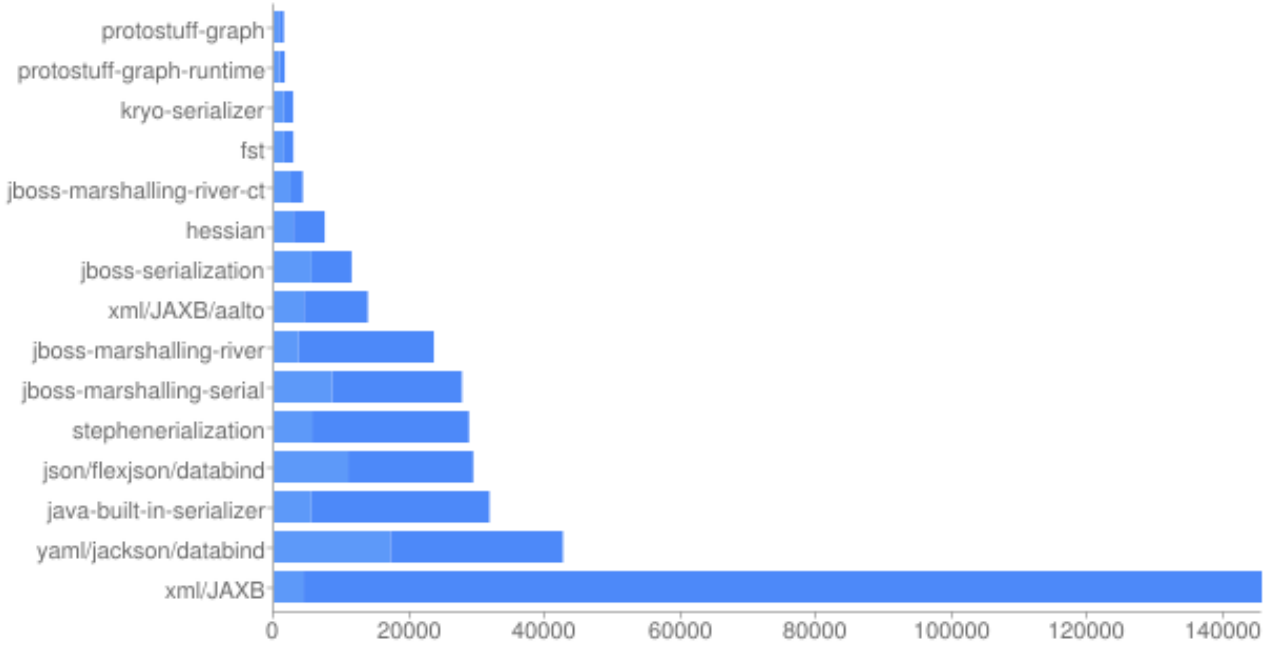
```
xml/jackson/databind                    54     2639     4720     7359     683
286
json/gson/databind                      56     4667     4403     9070     486
259
bson/jackson/databind                   54     4105     5449     9554     506
286
xml/xstream+c                           52     4383     9434    13817     487
244
json/javax-tree/glassfish             1249     6818    10284    17102     485
263
xml/exi-manual                          54    11375     9891    21266     337
327
java-built-in                           53     5046    23279    28325     889
514
scala/java-built-in                    514     8280    36105    44385    1293
698
json/protobuf                          123     6630    56787    63417     488
253
json/json-lib/databind                  61    19853    71969    91822     485
263
```

Full Object Graph Serializers

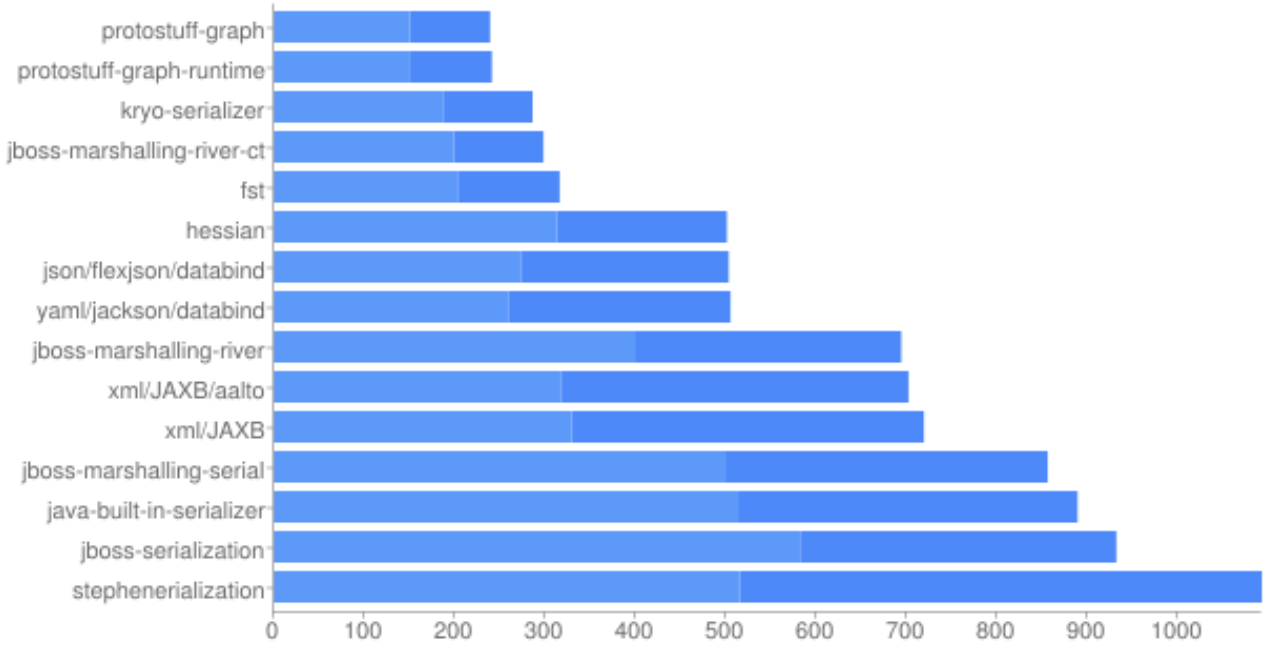
Contains serializer(-configurations)

- supporting full object graph write/read. Object graph may contain cycles. If an Object is referenced twice, it will be so after deserialization.
- nothing is known in advance, no class generation, no preregistering of classes. Everything is captured at runtime using e.g. reflection.
- note this usually cannot be used cross language, however JSON/XML formats may enable cross language deserialization.

Ser Time+Deser Time (ns)



Size, Compressed size [light] in bytes



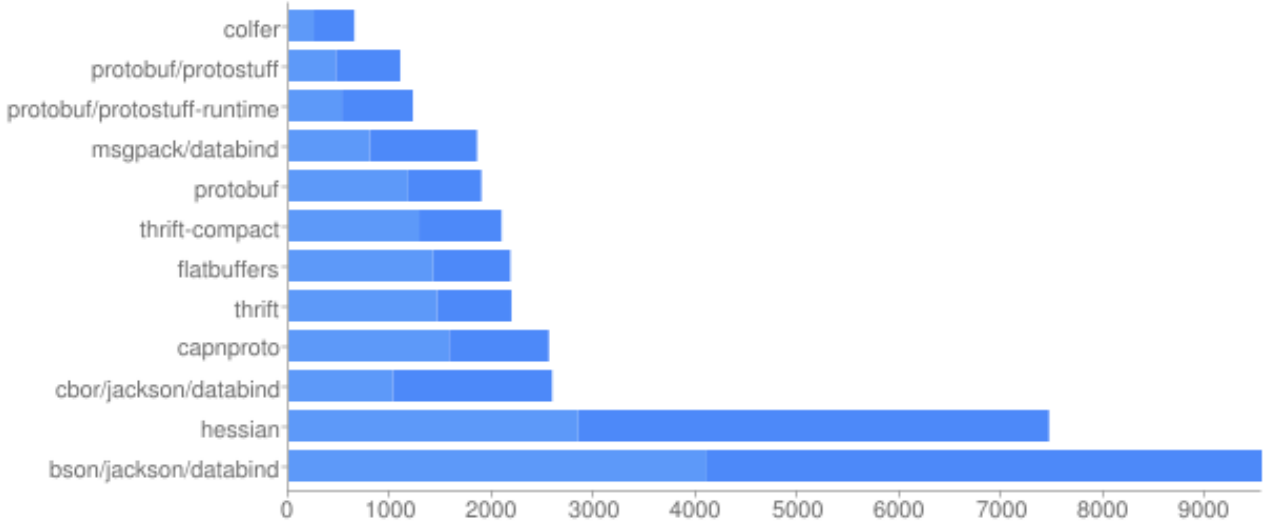
	create	ser	deser	total	size
+dfl					
protostuff-graph	70	691	750	1441	239
150					
protostuff-graph-runtime	60	772	799	1571	241
151					
kryo-serializer	53	1480	1331	2810	286
188					
fst	53	1511	1318	2830	316
203					
jboss-marshalling-river-ct	54	2470	1753	4223	298
199					
hessian	54	2842	4622	7464	501
313					
jboss-serialization	61	5547	5894	11441	932
582					
xml/JAXB/aalto	55	4404	9418	13822	702
318					
jboss-marshalling-river	53	3538	20025	23563	694
400					
jboss-marshalling-serial	54	8524	19178	27702	856
498					
stephenerialization	56	5595	23143	28739	1093
515					
json/flexjson/databind	53	10869	18507	29376	503
273					
java-built-in-serializer	55	5501	26263	31764	889
514					
yaml/jackson/databind	53	17163	25436	42599	505
260					
xml/JAXB	54	4354	141333	145686	719
329					

Cross Lang Binary Serializers

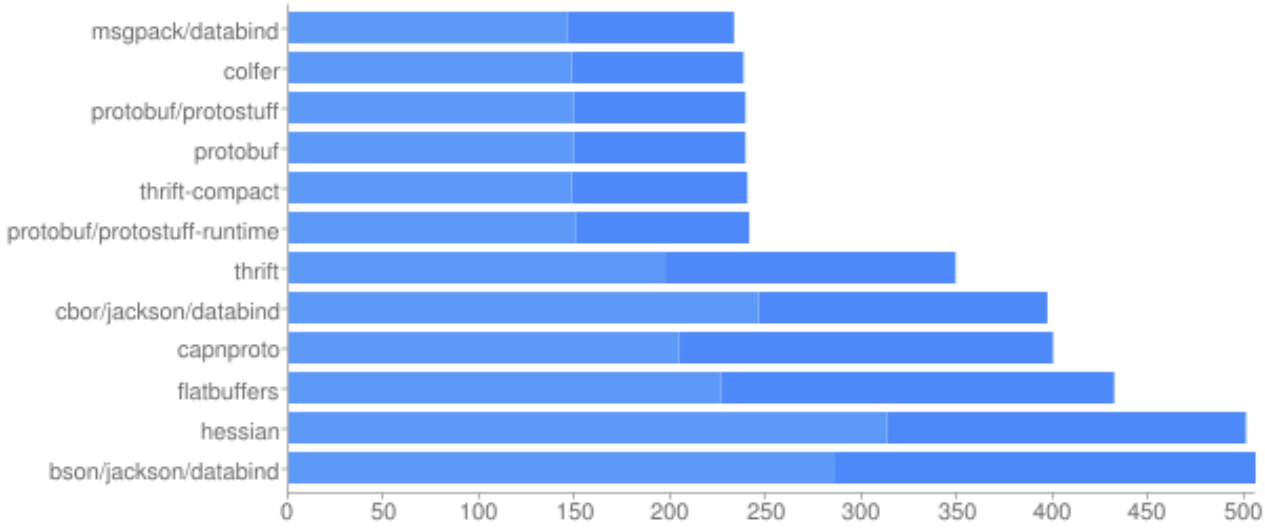
Contains serializer(-configurations)

- Only cycle free tree structures. An object referenced twice will be serialized twice.
- schema is known in advance (pre registration, intermediate message description languages, class generation).

Ser Time+Deser Time (ns)



Size, Compressed size [light] in bytes

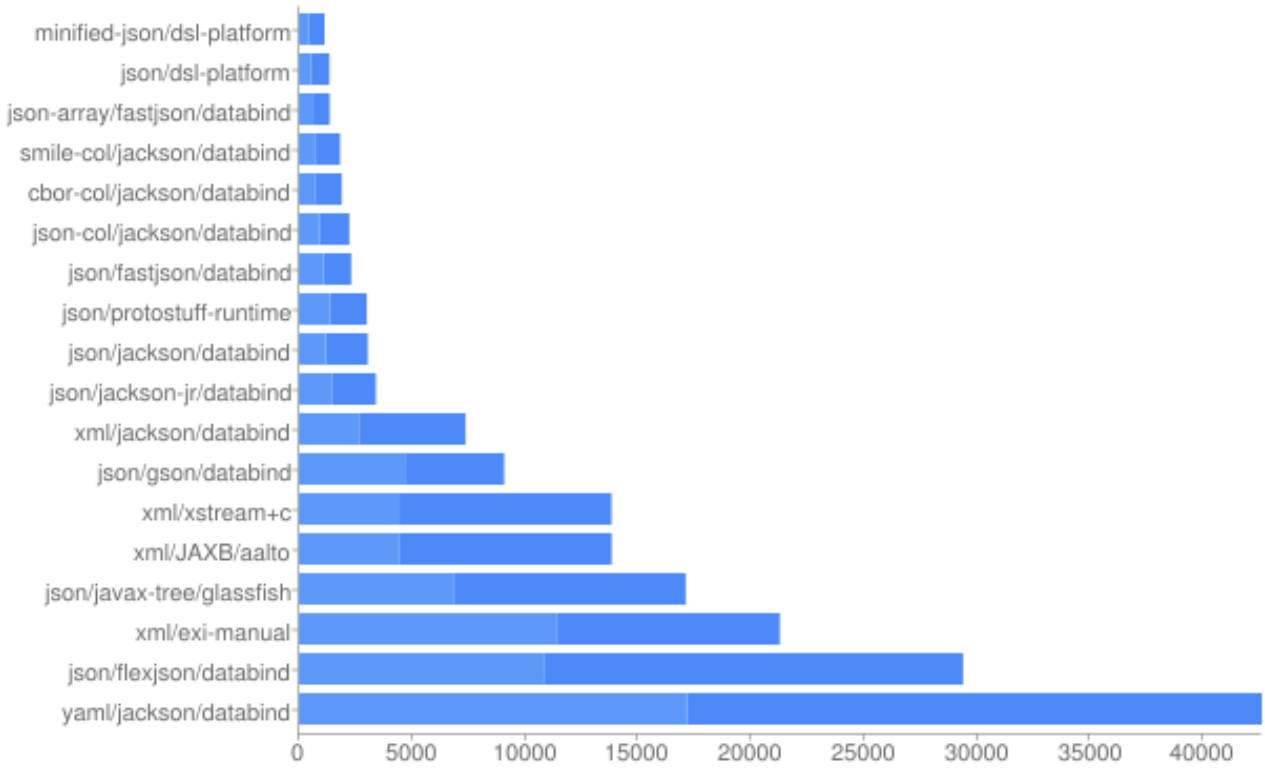


	create	ser	deser	total	size	+dfl
colfer	49	248	396	643	238	148
protobuf/protostuff	63	461	633	1094	239	149
protobuf/protostuff-runtime	52	518	701	1218	241	150
msgpack/databind	54	796	1052	1848	233	146
protobuf	121	1173	719	1891	239	149
thrift-compact	97	1280	808	2088	240	148
flatbuffers	56	1417	758	2175	432	226
thrift	95	1455	731	2186	349	197
capnproto	55	1574	979	2553	400	204
cbor/jackson/databind	53	1023	1561	2585	397	246
hessian	54	2842	4622	7464	501	313
bson/jackson/databind	54	4105	5449	9554	506	286

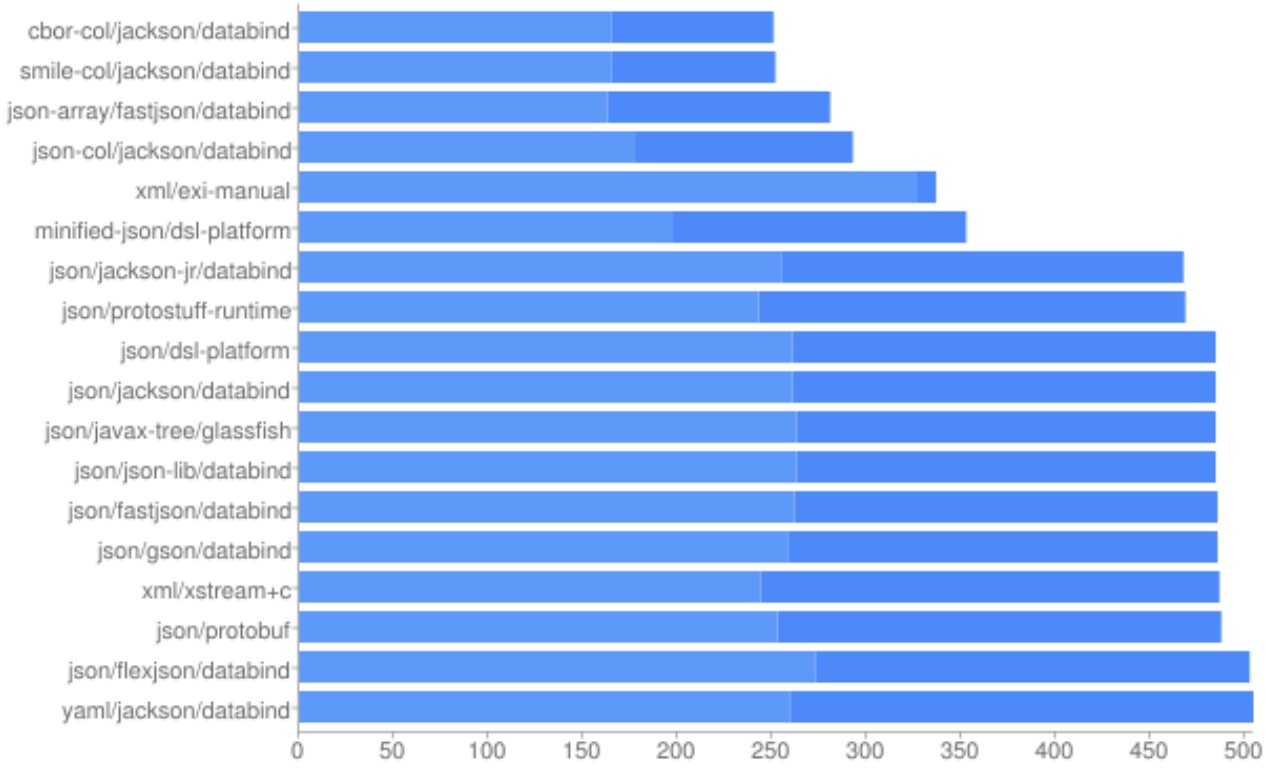
XML/JSon Serializers

- text format based. Usually can be read by anybody. Frequently inline schema inside data.
- Mixed regarding required preparation, object graph awareness (references).

Ser Time+Deser Time (ns)



Size, Compressed size [light] in bytes



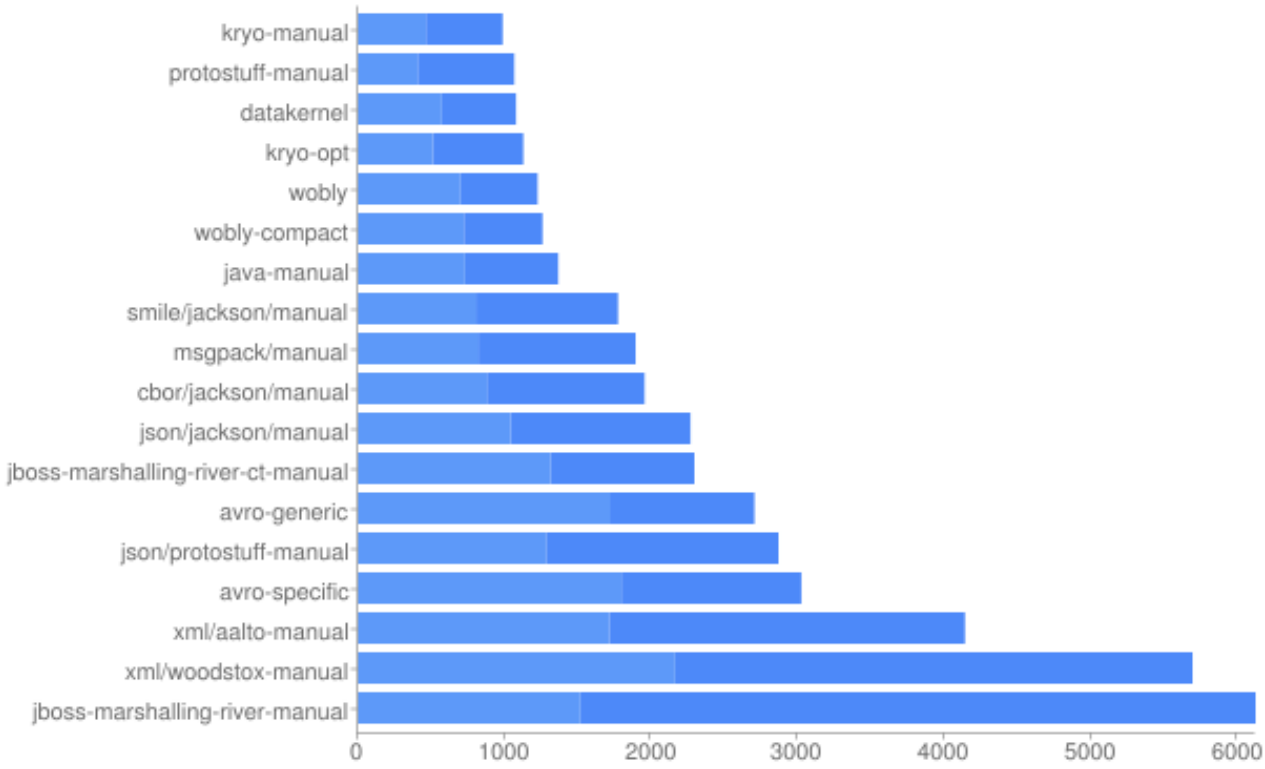
	create	ser	deser	total	size	
+dfl						
minified-json/dsl-platform	46	432	684	1116	353	197
json/dsl-platform	44	526	806	1332	485	261
json-array/fastjson/databind	53	650	696	1345	281	163
smile-col/jackson/databind	54	723	1082	1805	252	165
cbor-col/jackson/databind	54	732	1147	1879	251	165
json-col/jackson/databind	53	887	1329	2216	293	178
json/fastjson/databind	53	1058	1241	2299	486	262
json/protostuff-runtime	54	1353	1632	2986	469	243
json/jackson/databind	54	1164	1866	3030	485	261
json/jackson-jr/databind	53	1426	1962	3389	468	255
xml/jackson/databind	54	2639	4720	7359	683	286
json/gson/databind	56	4667	4403	9070	486	259
xml/xstream+c	52	4383	9434	13817	487	244
xml/JAXB/aalto	55	4404	9418	13822	702	318
json/javax-tree/glassfish	1249	6818	10284	17102	485	263
xml/exi-manual	54	11375	9891	21266	337	327
json/flexjson/databind	53	10869	18507	29376	503	273
yaml/jackson/databind	53	17163	25436	42599	505	260
json/protobuf	123	6630	56787	63417	488	253
json/json-lib/databind	61	19853	71969	91822	485	263
xml/JAXB	54	4354	141333	145686	719	329

Manually optimized Serializers

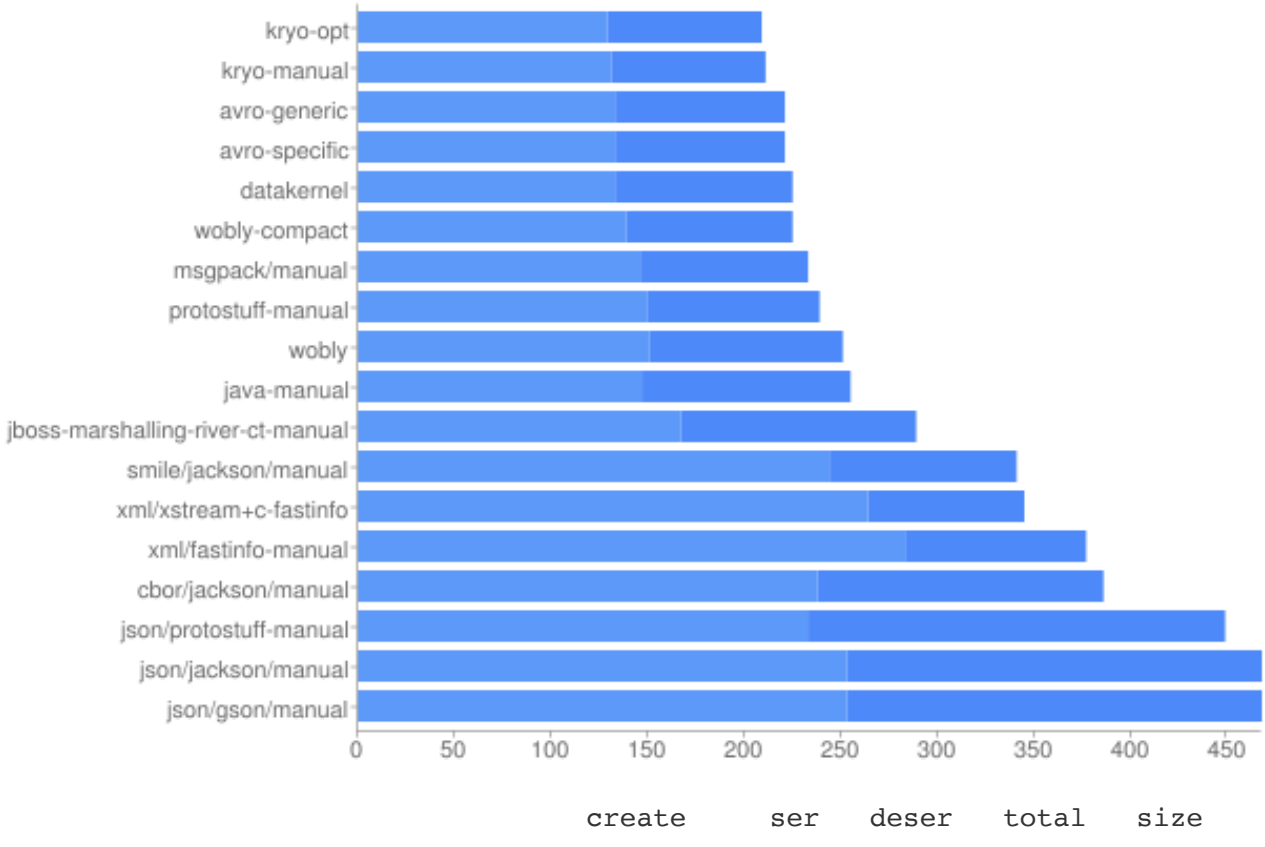
all flavours of manually optimized serializers. Handcoded and hardwired to exactly the benchmark's message structures.

- illustrates what's possible, at what level generic approaches can be optimized in case

Ser Time+Deser Time (ns)



Size, Compressed size [light] in bytes



+dfl					
kryo-manual	54	462	524	986	211
131					
protostuff-manual	58	406	660	1065	239
150					
datakernel	60	571	506	1077	225
133					
kryo-opt	53	510	617	1127	209
129					
wobly	38	699	525	1224	251
151					
wobly-compact	37	721	537	1258	225
139					
java-manual	53	729	636	1365	255
147					
smile/jackson/manual	53	804	969	1773	341
244					
msgpack/manual	53	819	1073	1893	233
146					
cbor/jackson/manual	52	878	1075	1954	386
238					
json/jackson/manual	52	1039	1228	2267	468
253					
jboss-marshalling-river-ct-manual	53	1315	979	2294	289
167					
avro-generic	329	1721	984	2704	221
133					
json/protostuff-manual	53	1287	1581	2868	449
233					
avro-specific	78	1795	1229	3024	221
133					
xml/aalto-manual	52	1714	2426	4140	653
304					
xml/woodstox-manual	62	2163	3531	5693	653
304					
jboss-marshalling-river-manual	54	1516	4607	6123	483
240					
json/gson/manual	53	2936	3641	6577	468
253					
json/json-smart/manual-tree	53	4499	3430	7930	495
265					
xml/javolution/manual	52	3803	5755	9558	504
263					
xml/xstream+c-aalto	54	3310	6732	10042	525
273					
json/gson/manual-tree	56	4836	5471	10307	485
259					

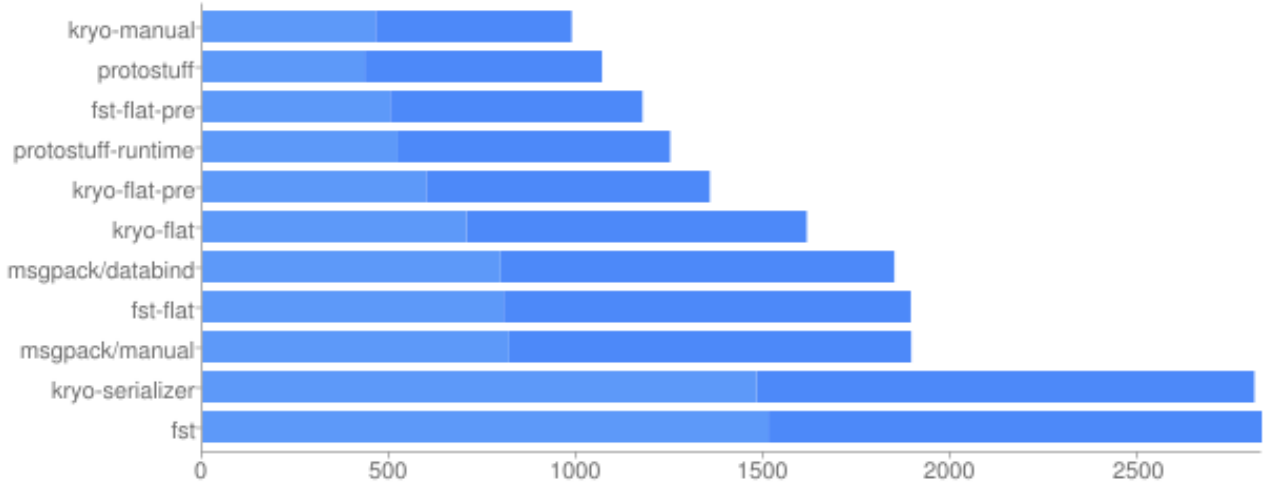
xml/fastinfo-manual	53	6326	4144	10470	377
284					
xml/xstream+c-fastinfo	53	5699	5246	10944	345
264					
bson/mongodb/manual	56	2764	8215	10979	495
278					
xml/xstream+c-woodstox	54	3937	7410	11347	525
273					
json/org.json/manual-tree	53	5468	6904	12372	485
259					
json/json.simple/manual	53	5627	7586	13213	495
265					
json/javax-stream/glassfish	54	4821	8992	13813	468
253					
json/svenson/databind	54	3946	11640	15586	495
267					
json/jsonij/manual-jpath	53	20047	9726	29773	478
257					
json/argo/manual-tree	53	56073	12539	68612	485
263					

Cost of features

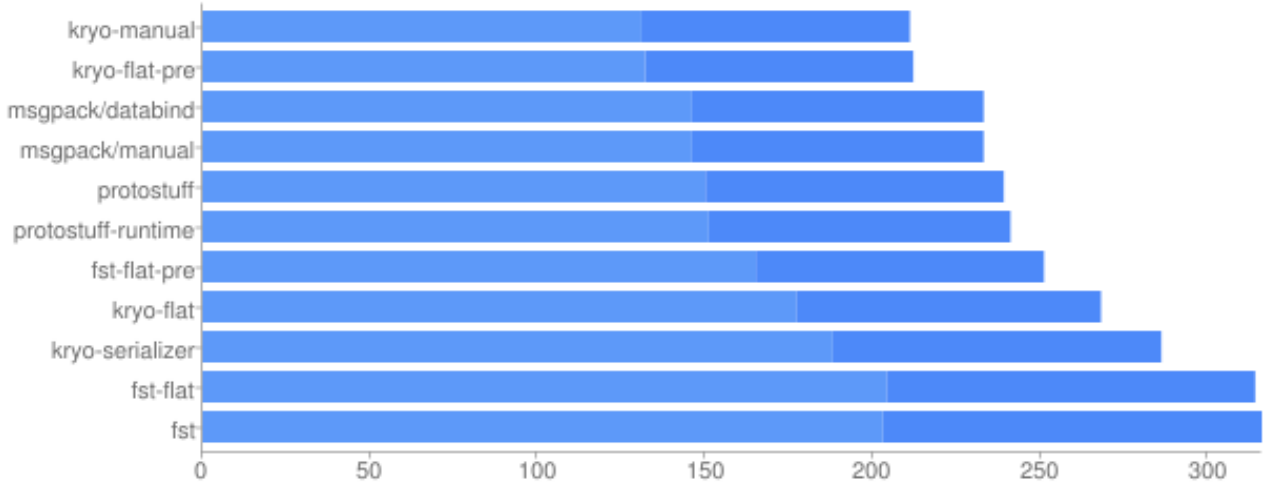
shows performance vs convenience of manually-selected libs.

- cycle free, schema known at compile time, manual optimization: kryo-manual, msgpack/manual
- cycle free, schema known at compile time: protostuff, fst-flat-pre, kryo-flat-pre. (note: protostuff uses class generation while the other two just require a list of classes to be written)
- cycle free, schema UNKNOWN at compile time: fst-flat, kryo-flat, protostuff-runtime, msgpack/databind
- full object graph awareness, schema UNKNOWN at compile time: fst, kryo.

Ser Time+Deser Time (ns)



Size, Compressed size [light] in bytes



	create	ser	deser	total	size
+dfl					
kryo-manual	54	462	524	986	211
131					
protostuff	68	433	634	1067	239
150					
fst-flat-pre	53	501	675	1175	251
165					
protostuff-runtime	70	522	727	1249	241
151					
kryo-flat-pre	54	597	758	1355	212
132					
kryo-flat	55	705	909	1614	268
177					
msgpack/databind	54	796	1052	1848	233
146					
fst-flat	53	804	1088	1892	314
204					
msgpack/manual	53	819	1073	1893	233
146					
kryo-serializer	53	1480	1331	2810	286
188					
fst	53	1511	1318	2830	316
203					

Full data

	create	ser	deser	total	size
+dfl					
colfer	49	248	396	643	238
148					
kryo-manual	54	462	524	986	211
131					
protostuff-manual	58	406	660	1065	239
150					
protostuff	68	433	634	1067	239
150					
datakernel	60	571	506	1077	225
133					
protobuf/protostuff	63	461	633	1094	239
149					
minified-json/dsl-platform	46	432	684	1116	353
197					
kryo-opt	53	510	617	1127	209
129					
fst-flat-pre	53	501	675	1175	251
165					

protobuf/protostuff-runtime	52	518	701	1218	241
150					
wobly	38	699	525	1224	251
151					
protostuff-runtime	70	522	727	1249	241
151					
wobly-compact	37	721	537	1258	225
139					
json/dsl-platform	44	526	806	1332	485
261					
json-array/fastjson/databind	53	650	696	1345	281
163					
kryo-flat-pre	54	597	758	1355	212
132					
java-manual	53	729	636	1365	255
147					
protostuff-graph	70	691	750	1441	239
150					
protostuff-graph-runtime	60	772	799	1571	241
151					
kryo-flat	55	705	909	1614	268
177					
smile/jackson/manual	53	804	969	1773	341
244					
smile-col/jackson/databind	54	723	1082	1805	252
165					
msgpack/databind	54	796	1052	1848	233
146					
cbor-col/jackson/databind	54	732	1147	1879	251
165					
protobuf	121	1173	719	1891	239
149					
fst-flat	53	804	1088	1892	314
204					
msgpack/manual	53	819	1073	1893	233
146					
cbor/jackson/manual	52	878	1075	1954	386
238					
smile/jackson+afterburner/databind	54	913	1175	2088	352
252					
thrift-compact	97	1280	808	2088	240
148					
cbor/jackson+afterburner/databind	53	888	1239	2126	397
246					
flatbuffers	56	1417	758	2175	432
226					
thrift	95	1455	731	2186	349

197
json-col/jackson/databind 53 887 1329 2216 293
178
json/jackson/manual 52 1039 1228 2267 468
253
jboss-marshalling-river-ct-manual 53 1315 979 2294 289
167
json/fastjson/databind 53 1058 1241 2299 486
262
smile/jackson/databind 53 1011 1300 2311 338
241
scala/sbinary 442 1311 1069 2381 255
147
capnproto 55 1574 979 2553 400
204
json/jackson+afterburner/databind 52 1094 1489 2584 485
261
cbor/jackson/databind 53 1023 1561 2585 397
246
avro-generic 329 1721 984 2704 221
133
kryo-serializer 53 1480 1331 2810 286
188
fst 53 1511 1318 2830 316
203
json/protostuff-manual 53 1287 1581 2868 449
233
json/protostuff-runtime 54 1353 1632 2986 469
243
avro-specific 78 1795 1229 3024 221
133
json/jackson/databind 54 1164 1866 3030 485
261
json/jackson-jr/databind 53 1426 1962 3389 468
255
xml/aalto-manual 52 1714 2426 4140 653
304
jboss-marshalling-river-ct 54 2470 1753 4223 298
199
xml/woodstox-manual 62 2163 3531 5693 653
304
jboss-marshalling-river-manual 54 1516 4607 6123 483
240
json/gson/manual 53 2936 3641 6577 468
253
xml/jackson/databind 54 2639 4720 7359 683
286

hessian	54	2842	4622	7464	501
313					
json/json-smart/manual-tree	53	4499	3430	7930	495
265					
json/gson/databind	56	4667	4403	9070	486
259					
bson/jackson/databind	54	4105	5449	9554	506
286					
xml/javolution/manual	52	3803	5755	9558	504
263					
xml/xstream+c-aalto	54	3310	6732	10042	525
273					
json/gson/manual-tree	56	4836	5471	10307	485
259					
xml/fastinfo-manual	53	6326	4144	10470	377
284					
xml/xstream+c-fastinfo	53	5699	5246	10944	345
264					
bson/mongodb/manual	56	2764	8215	10979	495
278					
xml/xstream+c-woodstox	54	3937	7410	11347	525
273					
jboss-serialization	61	5547	5894	11441	932
582					
json/org.json/manual-tree	53	5468	6904	12372	485
259					
json/json.simple/manual	53	5627	7586	13213	495
265					
json/javax-stream/glassfish	54	4821	8992	13813	468
253					
xml/xstream+c	52	4383	9434	13817	487
244					
xml/JAXB/aalto	55	4404	9418	13822	702
318					
json/svenson/databind	54	3946	11640	15586	495
267					
json/javax-tree/glassfish	1249	6818	10284	17102	485
263					
xml/exi-manual	54	11375	9891	21266	337
327					
jboss-marshalling-river	53	3538	20025	23563	694
400					
jboss-marshalling-serial	54	8524	19178	27702	856
498					
java-built-in	53	5046	23279	28325	889
514					
stephenerialization	56	5595	23143	28739	1093

515					
json/flexjson/databind	53	10869	18507	29376	503
273					
json/jsonij/manual-jpath	53	20047	9726	29773	478
257					
java-built-in-serializer	55	5501	26263	31764	889
514					
yaml/jackson/databind	53	17163	25436	42599	505
260					
scala/java-built-in	514	8280	36105	44385	1293
698					
json/protobuf	123	6630	56787	63417	488
253					
json/argo/manual-tree	53	56073	12539	68612	485
263					
json/json-lib/databind	61	19853	71969	91822	485
263					
xml/JAXB	54	4354	141333	145686	719
329					

	Effort	Format	Structure	Misc
colfer	CLASSES_KNOWN	BIN_CROSSLANG	FLAT_TREE	[] generated code
kryo-manual	MANUAL_OPT	BINARY	FLAT_TREE	[] manually optimized
protostuff-manual	MANUAL_OPT	BINARY	FLAT_TREE	[] manual
protostuff	CLASSES_KNOWN	BINARY	FLAT_TREE	[] generated code
datakernel	MANUAL_OPT	BINARY	FLAT_TREE	[] manually optimized
protobuf/protostuff	CLASSES_KNOWN	BIN_CROSSLANG	FLAT_TREE	[] protobuf + generated code
minified-json/dsl-platform	CLASSES_KNOWN	JSON	FLAT_TREE	[] JSON with minified property names and without default values.
kryo-opt	MANUAL_OPT	BINARY	FLAT_TREE	[] manually optimized
fst-flat-pre	CLASSES_KNOWN	BINARY	FLAT_TREE	[] fst in unshared mode with preregistered classes
protobuf/protostuff-runtime	ZERO_KNOWLEDGE	BIN_CROSSLANG	FLAT_TREE	[] protobuf + reflection
wobly	MANUAL_OPT	BINARY	FLAT_TREE	[]
protostuff-runtime	ZERO_KNOWLEDGE	BINARY	FLAT_TREE	

```
[] reflection
wobly-compact                       MANUAL_OPT      BINARY          FLAT_TREE
[]
json/dsl-platform                   CLASSES_KNOWN   JSON            FLAT_TREE
[] Serializes all properties with exact names.
json-array/fastjson/databind        ZERO_KNOWLEDGE  JSON            FLAT_TREE
[]
kryo-flat-pre                       CLASSES_KNOWN   BINARY          FLAT_TREE
[] no shared refs, preregistered classes
java-manual                         MANUAL_OPT      BINARY          FLAT_TREE
[]
protostuff-graph                    CLASSES_KNOWN   BINARY
FULL_GRAPH [] graph + generated code
protostuff-graph-runtime            ZERO_KNOWLEDGE  BINARY
FULL_GRAPH [] graph + reflection
kryo-flat                           ZERO_KNOWLEDGE  BINARY          FLAT_TREE
[] default, no shared refs
smile/jackson/manual                MANUAL_OPT      BINARY          FLAT_TREE
[]
smile-col/jackson/databind          ZERO_KNOWLEDGE  JSON            FLAT_TREE
[] uses positional (column) layout to eliminate use of names
msgpack/databind                    CLASSES_KNOWN   BIN_CROSSLANG   FLAT_TREE
[] uses positional (column) layout (instead of Maps std impl uses) to
eliminate use of names
cbor-col/jackson/databind           ZERO_KNOWLEDGE  JSON            FLAT_TREE
[] uses positional (column) layout to eliminate use of names
protobuf                            CLASSES_KNOWN   BIN_CROSSLANG   FLAT_TREE
[]
fst-flat                            ZERO_KNOWLEDGE  BINARY          FLAT_TREE
[] fst default, but unshared mode
msgpack/manual                      MANUAL_OPT      BIN_CROSSLANG   FLAT_TREE
[] uses positional (column) layout (instead of Maps std impl uses) to
eliminate use of names
cbor/jackson/manual                 MANUAL_OPT      BIN_CROSSLANG   FLAT_TREE
[]
smile/jackson+afterburner/databind ZERO_KNOWLEDGE  BINARY          FLAT_TREE
[] uses bytecode generation to reduce overhead
thrift-compact                      CLASSES_KNOWN   BIN_CROSSLANG   FLAT_TREE
[]
cbor/jackson+afterburner/databind   ZERO_KNOWLEDGE  BINARY          FLAT_TREE
[] uses bytecode generation to reduce overhead
flatbuffers                         CLASSES_KNOWN   BIN_CROSSLANG   FLAT_TREE
[]
thrift                              CLASSES_KNOWN   BIN_CROSSLANG   FLAT_TREE
[]
json-col/jackson/databind           ZERO_KNOWLEDGE  JSON            FLAT_TREE
[] uses positional (column) layout to eliminate use of names
```

```
json/jackson/manual                 MANUAL_OPT      JSON            FLAT_TREE
[]
jboss-marshalling-river-ct-manual   MANUAL_OPT      BINARY
FULL_GRAPH [] full graph preregistered classes, manual optimization
json/fastjson/databind              ZERO_KNOWLEDGE  JSON            FLAT_TREE
[]
smile/jackson/databind              ZERO_KNOWLEDGE  BINARY          FLAT_TREE
[]
scala/sbinary                       MISC            MISC            UNKNOWN
[] null
capnproto                           CLASSES_KNOWN   BIN_CROSSLANG   FLAT_TREE
[]
json/jackson+afterburner/databind   ZERO_KNOWLEDGE  BINARY          FLAT_TREE
[] uses bytecode generation to reduce overhead
cbor/jackson/databind               ZERO_KNOWLEDGE  BIN_CROSSLANG   FLAT_TREE
[]
avro-generic                        MANUAL_OPT      BIN_CROSSLANG   FLAT_TREE
[]
kryo-serializer                     ZERO_KNOWLEDGE  BINARY
FULL_GRAPH [] default
fst                                 ZERO_KNOWLEDGE  BINARY
FULL_GRAPH [] default: JDK serialization drop-in-replacement mode
json/protostuff-manual              MANUAL_OPT      JSON            FLAT_TREE
[] json + manual
json/protostuff-runtime             ZERO_KNOWLEDGE  JSON            FLAT_TREE
[] json + reflection
avro-specific                       MANUAL_OPT      BIN_CROSSLANG   UNKNOWN
[]
json/jackson/databind               ZERO_KNOWLEDGE  JSON            FLAT_TREE
[]
json/jackson-jr/databind            ZERO_KNOWLEDGE  JSON            FLAT_TREE
[]
xml/aalto-manual                    MANUAL_OPT      XML             UNKNOWN
[]
jboss-marshalling-river-ct          CLASSES_KNOWN   BINARY
FULL_GRAPH [] full graph with preregistered classes
xml/woodstox-manual                 MANUAL_OPT      XML             UNKNOWN
[]
jboss-marshalling-river-manual      MANUAL_OPT      BINARY
FULL_GRAPH [] full graph with manual optimizations
json/gson/manual                    MANUAL_OPT      JSON            FLAT_TREE
[]
xml/jackson/databind                ZERO_KNOWLEDGE  XML             FLAT_TREE
[]
hessian                             ZERO_KNOWLEDGE  BIN_CROSSLANG
FULL_GRAPH []
json/json-smart/manual-tree         MANUAL_OPT      JSON            FLAT_TREE
```

[]
json/gson/databind ZERO_KNOWLEDGE JSON FLAT_TREE
[]
bson/jackson/databind CLASSES_KNOWN BIN_CROSSLANG FLAT_TREE
[]
xml/javolution/manual MANUAL_OPT XML FLAT_TREE
[]
xml/xstream+c-aalto MANUAL_OPT XML FLAT_TREE
[]
json/gson/manual-tree MANUAL_OPT JSON FLAT_TREE
[]
xml/fastinfo-manual MANUAL_OPT XML UNKNOWN
[]
xml/xstream+c-fastinfo MANUAL_OPT XML FLAT_TREE
[]
bson/mongodb/manual MANUAL_OPT BIN_CROSSLANG FLAT_TREE
[]
xml/xstream+c-woodstox MANUAL_OPT XML FLAT_TREE
[]
jboss-serialization ZERO_KNOWLEDGE BINARY
FULL_GRAPH []
json/org.json/manual-tree MANUAL_OPT JSON FLAT_TREE
[]
json/json.simple/manual MANUAL_OPT JSON FLAT_TREE
[]
json/javax-stream/glassfish MANUAL_OPT JSON FLAT_TREE
[]
xml/xstream+c ZERO_KNOWLEDGE XML FLAT_TREE
[]
xml/JAXB/aalto CLASSES_KNOWN XML
FULL_GRAPH []
json/svenson/databind MANUAL_OPT JSON FLAT_TREE
[]
json/javax-tree/glassfish ZERO_KNOWLEDGE JSON FLAT_TREE
[]
xml/exi-manual ZERO_KNOWLEDGE XML UNKNOWN
[]
jboss-marshalling-river ZERO_KNOWLEDGE BINARY
FULL_GRAPH [] full graph zero knowledge
jboss-marshalling-serial ZERO_KNOWLEDGE BINARY
FULL_GRAPH []
java-built-in ZERO_KNOWLEDGE BINARY FLAT_TREE
[]
stephenerialization ZERO_KNOWLEDGE BINARY
FULL_GRAPH [] null
json/flexjson/databind ZERO_KNOWLEDGE JSON
FULL_GRAPH []

```
json/jsonij/manual-jpath            MANUAL_OPT      JSON            FLAT_TREE
[]
java-built-in-serializer            ZERO_KNOWLEDGE  BINARY
FULL_GRAPH []
yaml/jackson/databind               ZERO_KNOWLEDGE  JSON
FULL_GRAPH []
scala/java-built-in                 MISC            MISC            UNKNOWN
[] null
json/protobuf                       CLASSES_KNOWN   JSON            FLAT_TREE
[]
json/argo/manual-tree               MANUAL_OPT      JSON            FLAT_TREE
[]
json/json-lib/databind              ZERO_KNOWLEDGE  JSON            FLAT_TREE
[]
xml/JAXB                            CLASSES_KNOWN   XML
FULL_GRAPH []
```