Changelog

rmr 1.3 (Master Branch)

- An optional vectorized API for efficient R programming when dealing with small records.
- Fast C implementations for serialization and deserialization from and to typedbytes.
- Other readers and writers work much better in vectorized mode, namely csv and text
- Additional steps to support structured data better, that is you can use more data frames and less lists in the API
- Better whirr scripts, more forgiving behavior for package loading and bug fixes

See New in this release for details.

rmr 1.2

- Binary formats
- Simpler, more powerful I/O format API
- Native binary format with support for all R data types
- Worked around an R bug that made large reduces very slow.
- Backend specific parameters to modify things like number of reducers at the hadoop level
- Automatic library loading in mappers and reducers
- Better data frame conversions
- Adopted a uniform.naming.convention
- New package options API

See [[rmr v1.2 overview]] for details

rmr 1.1

- Native R serialization/deserialization, which implies that all R objects are supported as key and value, without any conversion boilerplate code. This is the new default. JSON still supported. csv reader/writer also available -- somewhat experimental.
- Multiple backends (hadoop and local); local backend is useful for debugging at small scale; having two backends enforces modular design, opens up further possibilities (rjava, Amazon's EMR, OpenCL have been suggested), forces to clarify semantics.
- Multiple tests of backend equivalence.
- Simpler interface for profiler.
- Equijoins (rough equivalent of merge for mapreduce)
- dfs.empty to check if file is empty
- to.map, to.reduce, to.reduce.all higher order functions to create simple map and reduce functions from regular ones.