Package ‘Storm’

February 19, 2015

Type Package
Version 1.2
Date 2014-12-25
Title Write Storm Bolts in R using the Storm Multi-Language Protocol.
Author Allen Day
Maintainer Allen Day <allenday@allenday.com>
License GPL (>= 2)
LazyLoad yes
Depends R (>= 2.1.0), methods, permute, rjson
Description Storm is a distributed real-time computation system. Similar to how
Hadoop provides a set of general primitives for doing batch processing, Storm
provides a set of general primitives for doing real-time computation.
Storm includes a ``Multi-Language'' (or ``Multilang'') Protocol to allow
implementation of Bolts and Spouts in languages other than Java. This R
extension provides implementations of utility functions to allow an application
developer to focus on application-specific functionality rather than Storm/R
communications plumbing.

NeedsCompilation no
Repository CRAN
Date/Publication 2015-01-01 09:07:13

R topics documented:

Storm-package ............................................................... 2

Index 5
Storm-package

Write Storm Bolts in R using the Storm Multi-Language Protocol.

Description

Storm is a distributed real-time computation system. Similar to how Hadoop provides a set of general primitives for doing batch processing, Storm provides a set of general primitives for doing real-time computation.

Storm includes a “Multi-Language” (or “Multilang”) Protocol to allow implementation of Bolts and Spouts in languages other than Java. This R extension provides implementations of utility functions to allow an application developer to focus on application-specific functionality rather than Storm/R communications plumbing.

Details

- Package: Storm
- Type: Package
- Version: 1.2
- Date: 2012-12-25
- License: GPL version 2 or newer
- LazyLoad: yes

From Storm’s point of view, it creates an R process to consume and produce Tuples. Storm communicates with R using a JSON-like format. Storm writes Tuples via STDIN, and reads Tuples from R via STDOUT. The Storm package implements several functions to take care of Storm/R I/O.

As the application programmer, you implement a single function with signature: `v "function(s=Storm, t=Tuple)"` that will be called once per Tuple. Inside this function, you can emit zero or more Tuples, as well as emit other status messages, such as failures and diagnostic messages.

To use this extension, briefly:

1. create a new Storm object.
2. define a function that can process and emit Tuple objects.
3. call the run() method on the Storm object.

A detailed example is given in the examples section.

Author(s)

Allen Day Maintainer: Allen Day <allenday@allenday.com>

References

Storm - https://github.com/nathanmarz/storm/wiki
Storm Multi-Language Protocol - https://github.com/nathanmarz/storm/wiki/Multilang-protocol
Examples

```r
#library(Storm)
#source("Storm/R/Storm.R")

#create a Storm object
storm = Storm$new()

#by default it has a handler that logs that the tuple was skipped.
#let's replace it that with something more interesting and equally
#useless.
storm$lambda = function(s) {
  #argument 's' is the Storm object.
  #get the current Tuple object.
  t = s$tuple;
  #optional: acknowledge receipt of the tuple.
  s$ack(t);
  #optional: log a message.
  s$log(c("processing tuple",t$id));
  #create contrived tuples to illustrate output.
  #create 1st tuple...
  t$output = vector(mode="character",length=1);
  t$output[1] = as.numeric(t$input[3])+as.numeric(t$input[4]);
  #...and emit it.
  s$emit(t);
  #create 2nd tuple...
  t$output[1] = as.numeric(t$input[3])-as.numeric(t$input[4]);
  #...and emit it.
  s$emit(t);
  #alternative/optional: mark the tuple as failed.
  s$fail(t);
}

#enter the main tuple-processing loop.
storm$run();
```

#proton:R allenday$ cat Storm/eg/example.txt
#
# "id": "-6955786537413359385",
# "comp": "1",
# "stream": "1",
# "task": 9,
# "tuple": ["snow white and the seven dwarfs", "field2", 3, 4]
#}
#proton:R allenday$ cat Storm/eg/example.txt | Storm/eg/example.r
#
# "command": "ack",
# "id": "-6955786537413359385"
#
#end
#
# "command": "log",
# "msg": "processing tuple=-6955786537413359385"
#
#end
#
# "command": "emit",
# "anchors": [],
# "stream": "1",
# "task": "9",
# "tuple": ["7"],
#
#end
#
# "command": "emit",
# "anchors": [],
# "stream": "1",
# "task": "9",
# "tuple": ["-1"],
#
#end
#
# "command": "fail",
# "id": "-6955786537413359385"
#
#end
Index

*Topic **hadoop**
  Storm-package, 2
*Topic **storm**
  Storm-package, 2
Storm (Storm-package), 2
Storm-class (Storm-package), 2
Storm-package, 2
Tuple (Storm-package), 2
Tuple-class (Storm-package), 2