

---

# **lark Documentation**

***Release 0.1***

**Alex Kessinger**

**Sep 27, 2017**



---

## Contents

---

|          |                                 |           |
|----------|---------------------------------|-----------|
| <b>1</b> | <b>Features</b>                 | <b>3</b>  |
| <b>2</b> | <b>Quickstart</b>               | <b>5</b>  |
| 2.1      | Planned Features . . . . .      | 6         |
| <b>3</b> | <b>Reference</b>                | <b>7</b>  |
| 3.1      | Core Redis API Client . . . . . | 7         |
|          | <b>Python Module Index</b>      | <b>31</b> |



Lark is a python library that provides a generic method from transforming a HTTP request into a redis command. If you have heard of [webdis](#) this works in roughly the same way. It provides two main framework specific methods for Django, and Flask. Though it should be easy to fit Lark into any python web framework.



# CHAPTER 1

---

## Features

---

- Has a RESTy interface ie. POST for writes, GET for reads, and DELETE for, well, deletes
- Automatic JSON serilization and deserialization for redis values
- Automatic key prefixing for multi-user environments
- Fully tested adapaters for Flask and Django
- Support for scope based authorization with an eye towards hooking up with [flask-oauthlib](#)
- While rough, documentation is available for [all supported methods](#)



## CHAPTER 2

---

### Quickstart

---

To get started make sure that you have `redis` installed, then install lark.

```
pip install lark
```

Next you can create a simple Flask app that mounts the lark blueprint. Lark also comes with a Redis middleware for setting up redis connections.

```
from flask import Flask
from lark.ext.flask.redis_api import redis_api_blueprint
from lark.ext.flask.flask_redis import Redis

app = Flask(__name__)
# Add a simple redis connection to the global object
Redis(app)

app.config['DEFAULT_LARK_SCOPES'] = set(['admin'])

# Mount the redis blueprint
app.register_blueprint(redis_api_blueprint, url_prefix='/api/0')

if __name__ == '__main__':
    app.run()
```

Now you can run the server and then you will be able to interact with the API like so. You can find documentation on all the calls [here](#).

```
>>> curl http://127.0.0.1:5000/api/0/get/a/
{"meta": {"status": "ok", "status_code": 200}}

>>> curl -X POST -H 'Content-Type: application/json' \
--data-ascii '{"value": "foo"}' \
http://127.0.0.1:5000/api/0/set/a/
{"meta": {"status": "ok", "status_code": 200}, "data": true}
```

```
>>> curl http://127.0.0.1:5000/api/0/get/a/  
{ "meta": { "status": "ok", "status_code": 200}, "data": "foo" }
```

## Planned Features

- Flask middleware to support oauth2
- A full Web interface for managing, and editing redis values.

## Core Redis API Client

The core redis API client is the main tool that maps from an http request to a redis call.

### `lark.redis.client.RedisApiClient` — Redis HTTP Adapter

#### Admin Methods

`RedisApiClient.bgrewriteaof`

redis docs for [bgrewriteaof](#)

Requires one of these scopes: **admin**, **write:bgrewriteaof**, **write:\***

```
POST /BGWRITEAOF/
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.client_list`

redis docs for [client\\_list](#)

Requires one of these scopes: **admin**, **read:client\_list**, **read:\***

```
GET /CLIENT/LIST/
```

```
output: {u'data': [{u'addr': u'127.0.0.1:54188',
                    u'age': u'1036279',
                    u'cmd': u'monitor',
                    u'db': u'0',
                    u'events': u'rw',
                    u'fd': u'6',
                    u'flags': u'O',
                    u'idle': u'0',
                    u'multi': u'-1',
```

```
        u'name': u'',
        u'obl': u'57',
        u'oll': u'0',
        u'omem': u'0',
        u'psub': u'0',
        u'qbuf': u'0',
        u'qbuf-free': u'0',
        u'sub': u'0'},
    {u'addr': u'127.0.0.1:51684',
     u'age': u'0',
     u'cmd': u'client',
     u'db': u'10',
     u'events': u'r',
     u'fd': u'7',
     u'flags': u'N',
     u'idle': u'0',
     u'multi': u'-1',
     u'name': u'',
     u'obl': u'0',
     u'oll': u'0',
     u'omem': u'0',
     u'psub': u'0',
     u'qbuf': u'0',
     u'qbuf-free': u'32768',
     u'sub': u'0'}],
    u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.bgsave`

redis docs for `bgsave`

Requires one of these scopes: **admin**, **write:\***, **write:bgsave**

```
POST /BGSAVE/
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.client_getname`

redis docs for `client_getname`

Requires one of these scopes: **admin**, **read:client\_getname**, **read:\***

`RedisApiClient.client_setname`

redis docs for `client_setname`

Requires one of these scopes: **admin**, **write:\***, **write:client\_setname**

`RedisApiClient.client_kill`

redis docs for `client_kill`

Requires one of these scopes: **admin**, **write:\***, **write:client\_kill**

`RedisApiClient.config_get`

redis docs for `config_get`

Requires one of these scopes: **admin**

```
GET /CONFIG/GET/
```

```
output: {u'data': {u'activeresharding': u'yes',
                  u'aof-rewrite-incremental-fsync': u'yes',
```

```

    u'appendfsync': u'everysec',
    u'appendonly': u'no',
    u'auto-aof-rewrite-min-size': u'1048576',
    u'auto-aof-rewrite-percentage': u'100',
    u'bind': u'',
    u'client-output-buffer-limit': u'normal 0 0 0 slave 268435456 67108864 60',
↪pubsub 33554432 8388608 60',
    u'daemonize': u'no',
    u'databases': u'16',
    u'dbfilename': u'dump.rdb',
    u'dir': u'/Users/alex',
    u'hash-max-ziplist-entries': u'512',
    u'hash-max-ziplist-value': u'64',
    u'hz': u'10',
    u'list-max-ziplist-entries': u'512',
    u'list-max-ziplist-value': u'64',
    u'logfile': u'',
    u'loglevel': u'notice',
    u'lua-time-limit': u'5000',
    u'masterauth': u'',
    u'maxclients': u'10000',
    u'maxmemory': u'0',
    u'maxmemory-policy': u'volatile-lru',
    u'maxmemory-samples': u'3',
    u'min-slaves-max-lag': u'10',
    u'min-slaves-to-write': u'0',
    u'no-appendfsync-on-rewrite': u'no',
    u'notify-keyspace-events': u'',
    u'pidfile': u'/var/run/redis.pid',
    u'port': u'6379',
    u'rdbchecksum': u'yes',
    u'rdbcompression': u'yes',
    u'repl-backlog-size': u'1048576',
    u'repl-backlog-ttl': u'3600',
    u'repl-disable-tcp-nodelay': u'no',
    u'repl-ping-slave-period': u'10',
    u'repl-timeout': u'60',
    u'requirepass': u'',
    u'save': u'3600 1 300 100 60 10000',
    u'set-max-intset-entries': u'512',
    u'slave-priority': u'100',
    u'slave-read-only': u'yes',
    u'slave-serve-stale-data': u'yes',
    u'slaveof': u'',
    u'slowlog-log-slower-than': u'10000',
    u'slowlog-max-len': u'128',
    u'stop-writes-on-bgsave-error': u'yes',
    u'tcp-keepalive': u'0',
    u'timeout': u'0',
    u'unixsocket': u'',
    u'unixsocketperm': u'0',
    u'watchdog-period': u'0',
    u'zset-max-ziplist-entries': u'128',
    u'zset-max-ziplist-value': u'64'},
u'meta': {u'status': u'ok', u'status_code': 200}}

```

RedisApiClient.**config\_set**  
 redis docs for config\_set

Requires one of these scopes: **admin**

```
POST /CONFIG/SET/maxclients/
```

```
input: {u'value': u'9999'}
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.config_resetstat`

redis docs for [config\\_resetstat](#)

Requires one of these scopes: **admin, write:\*, write:config\_resetstat**

```
POST /CONFIG/RESETSTAT/
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.dbsize`

redis docs for [dbsize](#)

Requires one of these scopes: **admin, read:dbsize, read:\***

```
GET /DBSIZE/
```

```
output: {u'data': 2, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.flushall`

redis docs for [flushall](#)

Requires one of these scopes: **admin, write:\*, write:flushall**

```
DELETE /FLUSHALL/
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.flushdb`

redis docs for [flushdb](#)

Requires one of these scopes: **admin, write:\*, write:flushdb**

```
DELETE /FLUSHDB/
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.lastsave`

redis docs for [lastsave](#)

Requires one of these scopes: **admin, read:lastsave, read:\***

```
GET /LASTSAVE/
```

```
output: {u'data': u'2013-12-18T23:09:22',  
u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.save`

redis docs for [save](#)

Requires one of these scopes: **admin, write:\*, write:save**

RedisApiClient.**debug\_object**

redis docs for `debug_object`

Requires one of these scopes: **admin**, **read:debug\_object**, **read:\***

RedisApiClient.**info**

redis docs for `info`

Requires one of these scopes: **admin**, **read:\***, **read:info**

```
GET /INFO/
```

```
output: {u'data': {u'aof_current_rewrite_time_sec': -1,
  u'aof_enabled': 0,
  u'aof_last_bgrewrite_status': u'ok',
  u'aof_last_rewrite_time_sec': -1,
  u'aof_rewrite_in_progress': 0,
  u'aof_rewrite_scheduled': 0,
  u'arch_bits': 64,
  u'blocked_clients': 0,
  u'client_biggest_input_buf': 0,
  u'client_longest_output_list': 0,
  u'config_file': u'',
  u'connected_clients': 2,
  u'connected_slaves': 0,
  u'db0': {u'avg_ttl': 0, u'expires': 0, u'keys': 4216},
  u'db1': {u'avg_ttl': 0, u'expires': 0, u'keys': 340806},
  u'db2': {u'avg_ttl': 0, u'expires': 0, u'keys': 1746},
  u'db3': {u'avg_ttl': 0, u'expires': 0, u'keys': 1},
  u'evicted_keys': 0,
  u'expired_keys': 0,
  u'gcc_version': u'4.2.1',
  u'hz': 10,
  u'instantaneous_ops_per_sec': 0,
  u'keyspace_hits': 216,
  u'keyspace_misses': 26,
  u'latest_fork_usec': 0,
  u'loading': 0,
  u'lru_clock': 331671,
  u'master_repl_offset': 0,
  u'mem_allocator': u'libc',
  u'mem_fragmentation_ratio': 0.0,
  u'multiplexing_api': u'kqueue',
  u'os': u'Darwin 13.0.0 x86_64',
  u'process_id': 55585,
  u'pubsub_channels': 0,
  u'pubsub_patterns': 0,
  u'rdb_bgsave_in_progress': 0,
  u'rdb_changes_since_last_save': 701,
  u'rdb_current_bgsave_time_sec': -1,
  u'rdb_last_bgsave_status': u'ok',
  u'rdb_last_bgsave_time_sec': 2,
  u'rdb_last_save_time': 1387436962,
  u'redis_build_id': u'b8cc45f60db4b294',
  u'redis_git_dirty': 0,
  u'redis_git_sha1': 0,
  u'redis_mode': u'standalone',
  u'redis_version': u'2.8.1',
  u'rejected_connections': 0,
  u'repl_backlog_active': 0,
```

```
u'repl_backlog_first_byte_offset': 0,
u'repl_backlog_histlen': 0,
u'repl_backlog_size': 1048576,
u'role': u'master',
u'run_id': u'3ee0859b63dbd3a6ea41270b0f9d730d2c262af6',
u'sync_full': 0,
u'sync_partial_err': 0,
u'sync_partial_ok': 0,
u'tcp_port': 6379,
u'total_commands_processed': 1304,
u'total_connections_received': 652,
u'uptime_in_days': 12,
u'uptime_in_seconds': 1058691,
u'used_cpu_sys': 165.37,
u'used_cpu_sys_children': 100.54,
u'used_cpu_user': 108.23,
u'used_cpu_user_children': 452.6,
u'used_memory': 280149104,
u'used_memory_human': u'267.17M',
u'used_memory_lua': 33792,
u'used_memory_peak': 300950240,
u'used_memory_peak_human': u'287.01M',
u'used_memory_rss': 1179648},
u'meta': {u'status': u'ok', u'status_code': 200}}
```

#### RedisApiClient.**ping**

redis docs for [ping](#)

Requires one of these scopes: **admin**, **basic:\***, **read:\***, **read:ping**

```
GET /PING/
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

#### RedisApiClient.**echo**

redis docs for [echo](#)

Requires one of these scopes: **admin**, **basic:\***, **read:\***, **read:echo**

```
POST /ECHO/
```

```
input: {u'value': u'foo bar'}
```

```
output: {u'data': u'foo bar', u'meta': {u'status': u'ok', u'status_code': 200}}
```

## Key Methods

#### RedisApiClient.**randomkey**

redis docs for [randomkey](#)

Requires one of these scopes: **admin**, **key:\***, **read:randomkey**, **read:\***

```
GET /RANDOMKEY/
```

```
output: {u'data': None, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.get`  
redis docs for `get`

Requires one of these scopes: **admin**, **key:\***, **read:get**, **read:\***

```
GET /GET/a/
```

```
output: {u'data': None, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.set`  
redis docs for `set`

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:set**

```
POST /SET/a/
```

```
input: {u'value': u'1'}
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.append`  
redis docs for `append`

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:append**

```
POST /APPEND/a/
```

```
input: {u'value': u'a1'}
```

```
output: {u'data': 2, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.setbit`  
redis docs for `setbit`

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:setbit**

```
POST /SETBIT/a/
```

```
input: {u'offset': 5, u'value': True}
```

```
output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.bitcount`  
redis docs for `bitcount`

Requires one of these scopes: **admin**, **key:\***, **read:\***, **read:bitcount**

```
GET /BITCOUNT/a/
```

```
output: {u'data': 1, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.bitop`  
redis docs for `bitop`

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:bitop**

```
.. autoattribute:: lark.redis.client.RedisApiClient.decr
```

```
POST /DECR/a/
```

```
output: {u'data': -1, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.delete`

redis docs for `delete`

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:delete**

```
DELETE /DEL/a/
```

```
output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.dump`

redis docs for `dump`

Requires one of these scopes: **admin**, **read:dump**, **read:\***

`RedisApiClient.restore`

redis docs for `restore`

Requires one of these scopes: **admin**, **write:restore**, **write:\***

`RedisApiClient.exists`

redis docs for `exists`

Requires one of these scopes: **admin**, **key:\***, **read:exists**, **read:\***

```
GET /EXISTS/a/
```

```
output: {u'data': False, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.expire`

redis docs for `expire`

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:expire**

```
POST /EXPIRE/a/
```

```
input: {u'time': 10}
```

```
output: {u'data': False, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.expireat`

redis docs for `expireat`

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:expireat**

```
POST /EXPIREAT/a/
```

```
input: {u'when': u'2013-12-18T23:11:39.232554'}
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.ttl`

redis docs for `ttl`

Requires one of these scopes: **admin**, **key:\***, **read:ttl**, **read:\***

```
GET /TTL/a/
```

```
output: {u'data': 60, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**pexpire**

redis docs for [pexpire](#)

Requires one of these scopes: **admin, key:\*, write:\*, write:pexpire**

```
POST /PEXPIRE/a/
```

```
input: {u'time': 60000}
```

```
output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**pexpireat**

redis docs for [pexpireat](#)

Requires one of these scopes: **admin, key:\*, write:\*, write:pexpireat**

```
POST /PEXPIREAT/a/
```

```
input: {u'when': u'2013-12-18T23:11:39.681630'}
```

```
output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**pttl**

redis docs for [pttl](#)

Requires one of these scopes: **admin, key:\*, read:pttl, read:\***

```
GET /PTTL/a/
```

```
output: {u'data': 996, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**psetex**

redis docs for [psetex](#)

Requires one of these scopes: **admin, key:\*, write:\*, write:psetex**

```
POST /PSETEX/a/
```

```
input: {u'time_ms': 1000, u'value': u'value'}
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**persist**

redis docs for [persist](#)

Requires one of these scopes: **admin, key:\*, write:\*, write:persist**

```
POST /PERSIST/a/
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**getbit**

redis docs for [getbit](#)

Requires one of these scopes: **admin, key:\*, read:getbit, read:\***

```
GET /GETBIT/a/5/
```

```
output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.getrange`

redis docs for `getrange`

Requires one of these scopes: **admin, key:\*, read:getrange, read:\***

```
GET /GETRANGE/a/0/0/
```

```
output: {u'data': u'f', u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.getset`

redis docs for `getset`

Requires one of these scopes: **admin, key:\*, read:\*, read:getset**

```
POST /GETSET/a/
```

```
input: {u'value': u'foo'}
```

```
output: {u'data': None, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.incr`

redis docs for `incr`

Requires one of these scopes: **admin, key:\*, write:\*, write:incr**

```
POST /INCR/a/
```

```
output: {u'data': 1, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.incrbyfloat`

redis docs for `incrbyfloat`

Requires one of these scopes: **admin, key:\***

```
POST /INCRBYFLOAT/a/
```

```
output: {u'data': 1.0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.keys`

redis docs for `keys`

Requires one of these scopes: **admin, key:\*, read:keys, read:\***

```
GET /KEYS/
```

```
output: {u'data': [], u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.mget`

redis docs for `mget`

Requires one of these scopes: **admin, key:\*, read:mget, read:\***

```
GET /MGET/?key=a&key=b
```

```
output: {u'data': [None, None], u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**mset**

redis docs for [mset](#)

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:mset**

```
POST /MSET/
```

```
input: [[u'a', u'1'], [u'b', u'2'], [u'c', u'3']]
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**msetnx**

redis docs for [msetnx](#)

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:msetnx**

```
POST /MSETNX/
```

```
input: [[u'a', u'1'], [u'b', u'2'], [u'c', u'3']]
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**rename**

redis docs for [rename](#)

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:rename**

```
POST /RENAME/
```

```
input: {u'dst': u'b', u'src': u'a'}
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**renamenx**

redis docs for [renamenx](#)

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:renamenx**

```
POST /RENAMENX/
```

```
input: {u'dst': u'b', u'src': u'a'}
```

```
output: {u'data': False, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**setex**

redis docs for [setex](#)

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:setex**

```
POST /SETEX/a/
```

```
input: {u'time': 60, u'value': u'1'}
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**setnx**

redis docs for [setnx](#)

Requires one of these scopes: **admin**, **key:\***, **write:\***, **write:setnx**

```
POST /SETNX/a/
```

```
input: {u'value': u'1'}
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.setrange`

redis docs for `setrange`

Requires one of these scopes: **admin**, **key:\***

```
POST /SETRANGE/a/
```

```
input: {u'offset': 5, u'value': u'foo'}
```

```
output: {u'data': 8, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.strlen`

redis docs for `strlen`

Requires one of these scopes: **admin**, **key:\***, **read:\***, **read:strlen**

```
GET /STRLEN/a/
```

```
output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.substr`

redis docs for `substr`

Requires one of these scopes: **admin**, **key:\***, **read:substr**, **read:\***

```
GET /SUBSTR/a/3/5/
```

```
output: {u'data': u'345', u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.type`

redis docs for `type`

Requires one of these scopes: **admin**, **key:\***, **read:\***, **read:type**

```
GET /TYPE/a/
```

```
output: {u'data': u'none', u'meta': {u'status': u'ok', u'status_code': 200}}
```

## List Methods

`RedisApiClient.blpop`

redis docs for `blpop`

Requires one of these scopes: **admin**, **list:\***, **write:\***, **write:blpop**

```
POST /BLPOP/
```

```
input: {u'keys': [u'b', u'a'], u'timeout': 1}
```

```
output: {u'data': [u'b', u'3'], u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**brpop**  
 redis docs for [brpop](#)

Requires one of these scopes: **admin**, **list:\***, **write:\***, **write:brpop**

```
POST /BRPOP/

input: {'keys': [u'b', u'a'], u'timeout': 1}

output: {'data': [u'b', u'4'], u'meta': {'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**brpoplpush**  
 redis docs for [brpoplpush](#)

Requires one of these scopes: **admin**, **list:\***, **write:\***, **write:brpoplpush**

```
POST /BRPOPLPUSH/

input: {'dst': u'b', u'src': u'a'}

output: {'data': u'2', u'meta': {'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**lindex**  
 redis docs for [lindex](#)

Requires one of these scopes: **admin**, **read:lindex**, **list:\***, **read:\***

```
GET /LINDEX/a/0/

output: {'data': u'1', u'meta': {'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**linsert**  
 redis docs for [linsert](#)

Requires one of these scopes: **admin**, **write:linsert**, **list:\***, **write:\***

```
POST /LINSERT/a/

input: {'refvalue': u'2', u'value': u'2.5', u'where': u'after'}

output: {'data': 4, u'meta': {'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**llen**  
 redis docs for [llen](#)

Requires one of these scopes: **admin**, **list:\***, **read:llen**, **read:\***

```
GET /LLEN/a/

output: {'data': 3, u'meta': {'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**lpop**  
 redis docs for [lpop](#)

Requires one of these scopes: **admin**, **list:\***, **write:\***, **write:lpop**

```
POST /LPOP/a/

output: {'data': u'1', u'meta': {'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**lpush**

redis docs for [lpush](#)

Requires one of these scopes: **admin**, **write:lpush**, **list:\***, **write:\***

```
POST /LPUSH/a/
```

```
input: {u'values': [u'1']}
```

```
output: {u'data': 1, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**lpushx**

redis docs for [lpushx](#)

Requires one of these scopes: **admin**, **list:\***, **write:\***, **write:lpushx**

```
POST /LPUSHX/a/
```

```
input: {u'value': u'1'}
```

```
output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**lrange**

redis docs for [lrange](#)

Requires one of these scopes: **admin**, **list:\***, **read:lrange**, **read:\***

```
GET /LRANGE/a/0/-1/
```

```
output: {u'data': [], u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**lrem**

redis docs for [lrem](#)

Requires one of these scopes: **admin**, **list:\***, **write:lrem**, **write:\***

```
DELETE /LREM/a/1/1/
```

```
output: {u'data': 1, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**lset**

redis docs for [lset](#)

Requires one of these scopes: **admin**, **list:\***, **write:\***, **write:lset**

```
POST /LSET/a/
```

```
input: {u'index': 1, u'value': u'4'}
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**ltrim**

redis docs for [ltrim](#)

Requires one of these scopes: **admin**, **list:\***, **write:ltrim**, **write:\***

```
DELETE /LTRIM/a/0/1/
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**rpop**  
 redis docs for **rpop**

Requires one of these scopes: **admin, list:\*, write:rpop, write:\***

```
POST /RPOP/a/
```

```
output: {u'data': u'3', u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**rpoplpush**  
 redis docs for **poplpush**

Requires one of these scopes: **admin, write:poplpush, list:\*, write:\***

```
POST /RPOPLPUSH/
```

```
input: {u'dst': u'b', u'src': u'a'}
```

```
output: {u'data': u'a3', u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**rpush**  
 redis docs for **rpush**

Requires one of these scopes: **admin, list:\*, write:\*, write:rpush**

```
POST /RPUSH/a/
```

```
input: {u'values': [u'1']}
```

```
output: {u'data': 1, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**rpushx**  
 redis docs for **rpushx**

Requires one of these scopes: **admin, list:\*, write:\*, write:rpushx**

```
POST /RPUSHX/a/
```

```
input: {u'value': u'b'}
```

```
output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

## Sort Method

RedisApiClient.**sort**  
 redis docs for **sort**

Requires one of these scopes: **admin, write:\*, sort:\*, write:sort**

```
GET /SORT/a/?get=user%3A%2A&get=%23&groups=1
```

```
output: {u'data': [[u'u1', u'1'], [u'u2', u'2'], [u'u3', u'3']],
  u'meta': {u'status': u'ok', u'status_code': 200}}
```

## Scan Method

`RedisApiClient.scan`  
redis docs for [scan](#)

Requires one of these scopes: **admin**, **read:scan**, **scan:\***, **read:\***

```
GET /SCAN/?match=a

output: {u'data': [u'0', [u'a']], u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.sscan`  
redis docs for [sscan](#)

Requires one of these scopes: **admin**, **read:sscan**, **scan:\***, **read:\***

```
GET /SSCAN/a/?match=1

output: {u'data': [u'0', [u'1']], u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.hscan`  
redis docs for [hscan](#)

Requires one of these scopes: **admin**, **read:hscan**, **scan:\***, **read:\***

```
GET /HSCAN/a/?match=a

output: {u'data': [u'0', {u'a': u'1'}],
  u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zscan`  
redis docs for [zscan](#)

Requires one of these scopes: **admin**, **read:zscan**, **scan:\***, **read:\***

```
GET /ZSCAN/a/?match=a

output: {u'data': [u'0', [[u'a', 1.0]]],
  u'meta': {u'status': u'ok', u'status_code': 200}}
```

## Set Methods

`RedisApiClient.sadd`  
redis docs for [sadd](#)

Requires one of these scopes: **admin**, **write:\***, **sets:\***, **write:sadd**

```
POST /SADD/a/

input: {u'values': [u'1', u'2', u'3']}

output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.smembers`  
redis docs for [smembers](#)

Requires one of these scopes: **admin**, **sets:\***, **read:\***, **read:smembers**

```
GET /SMEMBERS/a/
```

```
output: {u'data': [u'1', u'3', u'2'], u'meta': {u'status': u'ok', u'status_code': 200}}
↪ }
```

`RedisApiClient.scard`

redis docs for `scard`

Requires one of these scopes: **admin, sets:\*, read:\*, read:scard**

```
GET /SCARD/a/
```

```
output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.sdiff`

redis docs for `sdiff`

Requires one of these scopes: **admin, read:sdiff, sets:\*, read:\***

```
GET /SDIFF/?key=a&key=b
```

```
output: {u'data': [u'1', u'3', u'2'], u'meta': {u'status': u'ok', u'status_code': 200}}
↪ }
```

`RedisApiClient.sdiffstore`

redis docs for `sdiffstore`

Requires one of these scopes: **admin, write:\*, write:sdiffstore, sets:\***

```
POST /SDIFFSTORE/
```

```
input: {u'dest': u'c', u'keys': [u'a', u'b']}
```

```
output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.sinter`

redis docs for `sinter`

Requires one of these scopes: **admin, read:sinter, sets:\*, read:\***

```
GET /SINTER/?key=a&key=b
```

```
output: {u'data': [], u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.sinterstore`

redis docs for `sinterstore`

Requires one of these scopes: **admin, write:\*, write:sinterstore, sets:\***

```
POST /SINTERSTORE/
```

```
input: {u'dest': u'c', u'keys': [u'a', u'b']}
```

```
output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.sismember`

redis docs for `sismember`

Requires one of these scopes: **admin, read:sismember, sets:\*, read:\***

```
GET /SISMEMBER/a/1/
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.smove`

redis docs for `smove`

Requires one of these scopes: **admin, write:smove, write:\*, sets:\***

```
POST /SMOVE/
```

```
input: {u'dst': u'b', u'src': u'a', u'value': u'a1'}
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.spop`

redis docs for `spop`

Requires one of these scopes: **admin, write:\*, write:spop, sets:\***

```
POST /SPOP/a/
```

```
output: {u'data': u'2', u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.srandmember`

redis docs for `srandmember`

Requires one of these scopes: **admin, read:srandmember, sets:\*, read:\***

```
GET /SRANDMEMBER/a/
```

```
output: {u'data': u'1', u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.srem`

redis docs for `srem`

Requires one of these scopes: **admin, write:srem, write:\*, sets:\***

```
DELETE /SREM/a/?value=5
```

```
output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.sunion`

redis docs for `sunion`

Requires one of these scopes: **admin, sets:\*, read:\*, read:sunion**

```
GET /SUNION/?key=a&key=b
```

```
output: {u'data': [u'1', u'3', u'2'], u'meta': {u'status': u'ok', u'status_code': 200}}
↪ }
```

`RedisApiClient.sunionstore`

redis docs for `sunionstore`

Requires one of these scopes: **admin, write:\*, sets:\*, write:sunionstore**

```
POST /SUNIONSTORE/
```

```
input: {u'dest': u'c', u'keys': [u'a', u'b']}
output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

## Sorted Set Methods

RedisApiClient.**zadd**  
[redis docs for zadd](#)

Requires one of these scopes: **admin, sorted\_sets:\*, write:\*, write:zadd**

```
POST /ZADD/a/

input: {u'scores': [[u'a1', 1], [u'a2', 2], [u'a3', 3]]}
output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**zcard**  
[redis docs for zcard](#)

Requires one of these scopes: **admin, sorted\_sets:\*, read:zcard, read:\***

```
GET /ZCARD/a/

output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**zcount**  
[redis docs for zcount](#)

Requires one of these scopes: **admin, sorted\_sets:\*, read:zcount, read:\***

```
GET /ZCOUNT/a/-inf/+inf/

output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**zincrby**  
[redis docs for zincrby](#)

Requires one of these scopes: **admin, sorted\_sets:\*, write:\*, write:zincrby**

```
POST /ZINCRBY/a/

input: {u'value': u'a2'}
output: {u'data': 3.0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**zinterstore**  
[redis docs for zinterstore](#)

Requires one of these scopes: **admin, sorted\_sets:\*, write:\*, write:zinterstore**

```
POST /ZINTERSTORE/

input: {u'aggregate': u'MAX', u'dest': u'd', u'keys': [u'a', u'b', u'c']}
output: {u'data': 2, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zrange`

redis docs for [zrange](#)

Requires one of these scopes: **admin, sorted\_sets:\*, read:zrange, read:\***

```
GET /ZRANGE/d/0/-1/?withscores=1

output: {u'data': [[u'a3', 5.0], [u'a1', 6.0]],
  u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zrangebyscore`

redis docs for [zrangebyscore](#)

Requires one of these scopes: **admin, sorted\_sets:\*, read:zrangebyscore, read:\***

```
GET /ZRANGEBYSCORE/a/2/4/?start=1&num=2

output: {u'data': [u'a3', u'a4'], u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zrevrangebyscore`

redis docs for [zrevrangebyscore](#)

Requires one of these scopes: **admin, sorted\_sets:\*, read:zrevrangebyscore, read:\***

```
GET /ZREVRANGEBYSCORE/a/4/2/?start=1&num=2

output: {u'data': [u'a3', u'a2'], u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zrank`

redis docs for [zrank](#)

Requires one of these scopes: **admin, sorted\_sets:\*, read:zrank, read:\***

```
GET /ZRANK/a/a1/

output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zrevrank`

redis docs for [zrevrank](#)

Requires one of these scopes: **admin, sorted\_sets:\*, read:zrevrank, read:\***

```
GET /ZREVRANK/a/a1/

output: {u'data': 4, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zrem`

redis docs for [zrem](#)

Requires one of these scopes: **admin, sorted\_sets:\*, write:\*, write:zrem**

```
DELETE /ZREM/a/?value=a2

output: {u'data': 1, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zremrangebyrank`

redis docs for [zremrangebyrank](#)

Requires one of these scopes: **admin, sorted\_sets:\***

```
DELETE /ZREMRANGEBYRANK/a/1/3/
```

```
output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zremrangebyscore`

redis docs for [zremrangebyscore](#)

Requires one of these scopes: **admin, sorted\_sets:\***

```
DELETE /ZREMRANGEBYSCORE/a/2/4/
```

```
output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zrevrange`

redis docs for [zrevrange](#)

Requires one of these scopes: **admin, sorted\_sets:\*, read:zrevrange, read:\***

```
GET /ZREVRANGE/a/0/1/
```

```
output: {u'data': [u'a3', u'a2'], u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zscore`

redis docs for [zscore](#)

Requires one of these scopes: **admin, sorted\_sets:\*, read:zscore, read:\***

```
GET /ZSCORE/a/a1/
```

```
output: {u'data': 1.0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.zunionstore`

redis docs for [zunionstore](#)

Requires one of these scopes: **admin, sorted\_sets:\***

```
POST /ZUNIONSTORE/
```

```
input: {u'aggregate': u'MAX', u'dest': u'd', u'keys': [u'a', u'b', u'c']}
```

```
output: {u'data': 4, u'meta': {u'status': u'ok', u'status_code': 200}}
```

## Hash Methods

`RedisApiClient.hget`

redis docs for [hget](#)

Requires one of these scopes: **admin, hashes:\*, read:\*, read:hget**

```
GET /HGET/a/2/
```

```
output: {u'data': None, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.hgetall`

redis docs for [hgetall](#)

Requires one of these scopes: **admin, hashes:\*, read:\*, read:hgetall**

```
GET /HGETALL/a/
```

```
output: {u'data': {u'a1': u'1', u'a2': u'2', u'a3': u'3'},
  u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.hexists`

redis docs for [hexists](#)

Requires one of these scopes: **admin**, **hashes:\***, **read:hexists**, **read:\***

```
GET /HEXISTS/a/1/
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.hdel`

redis docs for [hdel](#)

Requires one of these scopes: **admin**, **hashes:\***, **write:\***, **write:hdel**

```
DELETE /HDEL/a/?key=2
```

```
output: {u'data': 1, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.hincrby`

redis docs for [hincrby](#)

Requires one of these scopes: **admin**, **hashes:\***, **write:\***, **write:hincrby**

```
POST /HINCRBY/a/1/
```

```
output: {u'data': 1, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.hincrbyfloat`

redis docs for [hincrbyfloat](#)

Requires one of these scopes: **admin**, **hashes:\***, **write:\***, **write:hincrbyfloat**

```
POST /HINCRBY/a/1/
```

```
input: {u'amount': 2}
```

```
output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.hkeys`

redis docs for [hkeys](#)

Requires one of these scopes: **admin**, **hashes:\***, **read:hkeys**, **read:\***

```
GET /HKEYS/a/
```

```
output: {u'data': [u'a1', u'a3', u'a2'],
  u'meta': {u'status': u'ok', u'status_code': 200}}
```

`RedisApiClient.hlen`

redis docs for [hlen](#)

Requires one of these scopes: **admin**, **hashes:\***, **read:hlen**, **read:\***

```
GET /HLEN/a/
```

```
output: {u'data': 3, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**hset**

redis docs for [hset](#)

Requires one of these scopes: **admin**, **hashes:\***, **write:\***, **write:hset**

```
POST /HSET/a/2/
```

```
input: {u'value': u'5'}
```

```
output: {u'data': 0, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**hsetnx**

redis docs for [hsetnx](#)

Requires one of these scopes: **admin**, **hashes:\***, **write:\***, **write:hsetnx**

```
POST /HSETNX/a/
```

```
input: {u'key': u'1', u'value': u'1'}
```

```
output: {u'data': 1, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**hmset**

redis docs for [hmset](#)

Requires one of these scopes: **admin**, **hashes:\***, **write:\***, **write:hmset**

```
POST /HMSET/a/
```

```
input: {u'mapping': {u'a1': u'1', u'a2': u'2', u'a3': u'3'}}
```

```
output: {u'data': True, u'meta': {u'status': u'ok', u'status_code': 200}}
```

RedisApiClient.**hmget**

redis docs for [hmget](#)

Requires one of these scopes: **admin**, **hashes:\***, **read:hmget**, **read:\***

```
GET /HMGET/a/?key=a&key=b&key=c
```

```
output: {u'data': [u'1', u'2', u'3'], u'meta': {u'status': u'ok', u'status_code': 200}}
↪ }
```

RedisApiClient.**hvals**

redis docs for [hvals](#)

Requires one of these scopes: **admin**, **hashes:\***, **read:hvals**, **read:\***

```
GET /HVALS/a/
```

```
output: {u'data': [u'1', u'3', u'2'], u'meta': {u'status': u'ok', u'status_code': 200}}
↪ }
```



|

`lark.redis.client`, [7](#)



## A

append (lark.redis.client.RedisApiClient attribute), 13

## B

bgrewriteaof (lark.redis.client.RedisApiClient attribute), 7

bgsave (lark.redis.client.RedisApiClient attribute), 8

bitcount (lark.redis.client.RedisApiClient attribute), 13

bitop (lark.redis.client.RedisApiClient attribute), 13

blpop (lark.redis.client.RedisApiClient attribute), 18

brpop (lark.redis.client.RedisApiClient attribute), 18

brpoplpush (lark.redis.client.RedisApiClient attribute), 19

## C

client\_getname (lark.redis.client.RedisApiClient attribute), 8

client\_kill (lark.redis.client.RedisApiClient attribute), 8

client\_list (lark.redis.client.RedisApiClient attribute), 7

client\_setname (lark.redis.client.RedisApiClient attribute), 8

config\_get (lark.redis.client.RedisApiClient attribute), 8

config\_resetstat (lark.redis.client.RedisApiClient attribute), 10

config\_set (lark.redis.client.RedisApiClient attribute), 9

## D

dbsize (lark.redis.client.RedisApiClient attribute), 10

debug\_object (lark.redis.client.RedisApiClient attribute), 10

delete (lark.redis.client.RedisApiClient attribute), 14

dump (lark.redis.client.RedisApiClient attribute), 14

## E

echo (lark.redis.client.RedisApiClient attribute), 12

exists (lark.redis.client.RedisApiClient attribute), 14

expire (lark.redis.client.RedisApiClient attribute), 14

expireat (lark.redis.client.RedisApiClient attribute), 14

## F

flushall (lark.redis.client.RedisApiClient attribute), 10

flushdb (lark.redis.client.RedisApiClient attribute), 10

## G

get (lark.redis.client.RedisApiClient attribute), 12

getbit (lark.redis.client.RedisApiClient attribute), 15

getrange (lark.redis.client.RedisApiClient attribute), 16

getset (lark.redis.client.RedisApiClient attribute), 16

## H

hdel (lark.redis.client.RedisApiClient attribute), 28

hexists (lark.redis.client.RedisApiClient attribute), 28

hget (lark.redis.client.RedisApiClient attribute), 27

hgetall (lark.redis.client.RedisApiClient attribute), 27

hincrby (lark.redis.client.RedisApiClient attribute), 28

hincrbyfloat (lark.redis.client.RedisApiClient attribute), 28

hkeys (lark.redis.client.RedisApiClient attribute), 28

hlen (lark.redis.client.RedisApiClient attribute), 28

hmget (lark.redis.client.RedisApiClient attribute), 29

hmset (lark.redis.client.RedisApiClient attribute), 29

hscan (lark.redis.client.RedisApiClient attribute), 22

hset (lark.redis.client.RedisApiClient attribute), 29

hsetnx (lark.redis.client.RedisApiClient attribute), 29

hvals (lark.redis.client.RedisApiClient attribute), 29

## I

incr (lark.redis.client.RedisApiClient attribute), 16

incrbyfloat (lark.redis.client.RedisApiClient attribute), 16

info (lark.redis.client.RedisApiClient attribute), 11

## K

keys (lark.redis.client.RedisApiClient attribute), 16

## L

lark.redis.client (module), 7

lastsave (lark.redis.client.RedisApiClient attribute), 10

lindex (lark.redis.client.RedisApiClient attribute), 19

linsert (lark.redis.client.RedisApiClient attribute), 19  
llen (lark.redis.client.RedisApiClient attribute), 19  
lpop (lark.redis.client.RedisApiClient attribute), 19  
lpush (lark.redis.client.RedisApiClient attribute), 19  
lpushx (lark.redis.client.RedisApiClient attribute), 20  
lrange (lark.redis.client.RedisApiClient attribute), 20  
lrem (lark.redis.client.RedisApiClient attribute), 20  
lset (lark.redis.client.RedisApiClient attribute), 20  
ltrim (lark.redis.client.RedisApiClient attribute), 20

## M

mget (lark.redis.client.RedisApiClient attribute), 16  
mset (lark.redis.client.RedisApiClient attribute), 16  
msetnx (lark.redis.client.RedisApiClient attribute), 17

## P

persist (lark.redis.client.RedisApiClient attribute), 15  
pexpire (lark.redis.client.RedisApiClient attribute), 15  
pexpireat (lark.redis.client.RedisApiClient attribute), 15  
ping (lark.redis.client.RedisApiClient attribute), 12  
psetex (lark.redis.client.RedisApiClient attribute), 15  
pttl (lark.redis.client.RedisApiClient attribute), 15

## R

randomkey (lark.redis.client.RedisApiClient attribute), 12  
rename (lark.redis.client.RedisApiClient attribute), 17  
renamenx (lark.redis.client.RedisApiClient attribute), 17  
restore (lark.redis.client.RedisApiClient attribute), 14  
rpop (lark.redis.client.RedisApiClient attribute), 20  
rpoplpush (lark.redis.client.RedisApiClient attribute), 21  
rpush (lark.redis.client.RedisApiClient attribute), 21  
rpushx (lark.redis.client.RedisApiClient attribute), 21

## S

sadd (lark.redis.client.RedisApiClient attribute), 22  
save (lark.redis.client.RedisApiClient attribute), 10  
scan (lark.redis.client.RedisApiClient attribute), 22  
scard (lark.redis.client.RedisApiClient attribute), 23  
sdiff (lark.redis.client.RedisApiClient attribute), 23  
sdiffstore (lark.redis.client.RedisApiClient attribute), 23  
set (lark.redis.client.RedisApiClient attribute), 13  
setbit (lark.redis.client.RedisApiClient attribute), 13  
setex (lark.redis.client.RedisApiClient attribute), 17  
setnx (lark.redis.client.RedisApiClient attribute), 17  
setrange (lark.redis.client.RedisApiClient attribute), 18  
sinter (lark.redis.client.RedisApiClient attribute), 23  
sinterstore (lark.redis.client.RedisApiClient attribute), 23  
sismember (lark.redis.client.RedisApiClient attribute), 23  
smembers (lark.redis.client.RedisApiClient attribute), 22  
smove (lark.redis.client.RedisApiClient attribute), 24  
sort (lark.redis.client.RedisApiClient attribute), 21  
spop (lark.redis.client.RedisApiClient attribute), 24  
srandmember (lark.redis.client.RedisApiClient attribute),

24

srem (lark.redis.client.RedisApiClient attribute), 24  
sscan (lark.redis.client.RedisApiClient attribute), 22  
strlen (lark.redis.client.RedisApiClient attribute), 18  
substr (lark.redis.client.RedisApiClient attribute), 18  
union (lark.redis.client.RedisApiClient attribute), 24  
sunionstore (lark.redis.client.RedisApiClient attribute),  
24

## T

ttl (lark.redis.client.RedisApiClient attribute), 14  
type (lark.redis.client.RedisApiClient attribute), 18

## Z

zadd (lark.redis.client.RedisApiClient attribute), 25  
zcard (lark.redis.client.RedisApiClient attribute), 25  
zcount (lark.redis.client.RedisApiClient attribute), 25  
zincrby (lark.redis.client.RedisApiClient attribute), 25  
zinterstore (lark.redis.client.RedisApiClient attribute), 25  
zrange (lark.redis.client.RedisApiClient attribute), 25  
zrangebyscore (lark.redis.client.RedisApiClient attribute), 26  
zrank (lark.redis.client.RedisApiClient attribute), 26  
zrem (lark.redis.client.RedisApiClient attribute), 26  
zremrangebyrank (lark.redis.client.RedisApiClient attribute), 26  
zremrangebyscore (lark.redis.client.RedisApiClient attribute), 27  
zrevrange (lark.redis.client.RedisApiClient attribute), 27  
zrevrangebyscore (lark.redis.client.RedisApiClient attribute), 26  
zrevrank (lark.redis.client.RedisApiClient attribute), 26  
zscan (lark.redis.client.RedisApiClient attribute), 22  
zscore (lark.redis.client.RedisApiClient attribute), 27  
zunionstore (lark.redis.client.RedisApiClient attribute),  
27