Redis Crash Course

TraversyMedia.com

What Is Redis?

- ✓**Open source in-memory data structure store** which can be used as a database and/or a cache and message broker
- ✓NoSQL Key/Value Store
- ✓Supports Multiple Data Structures
- ✓Built In Replication

Redis Datatypes

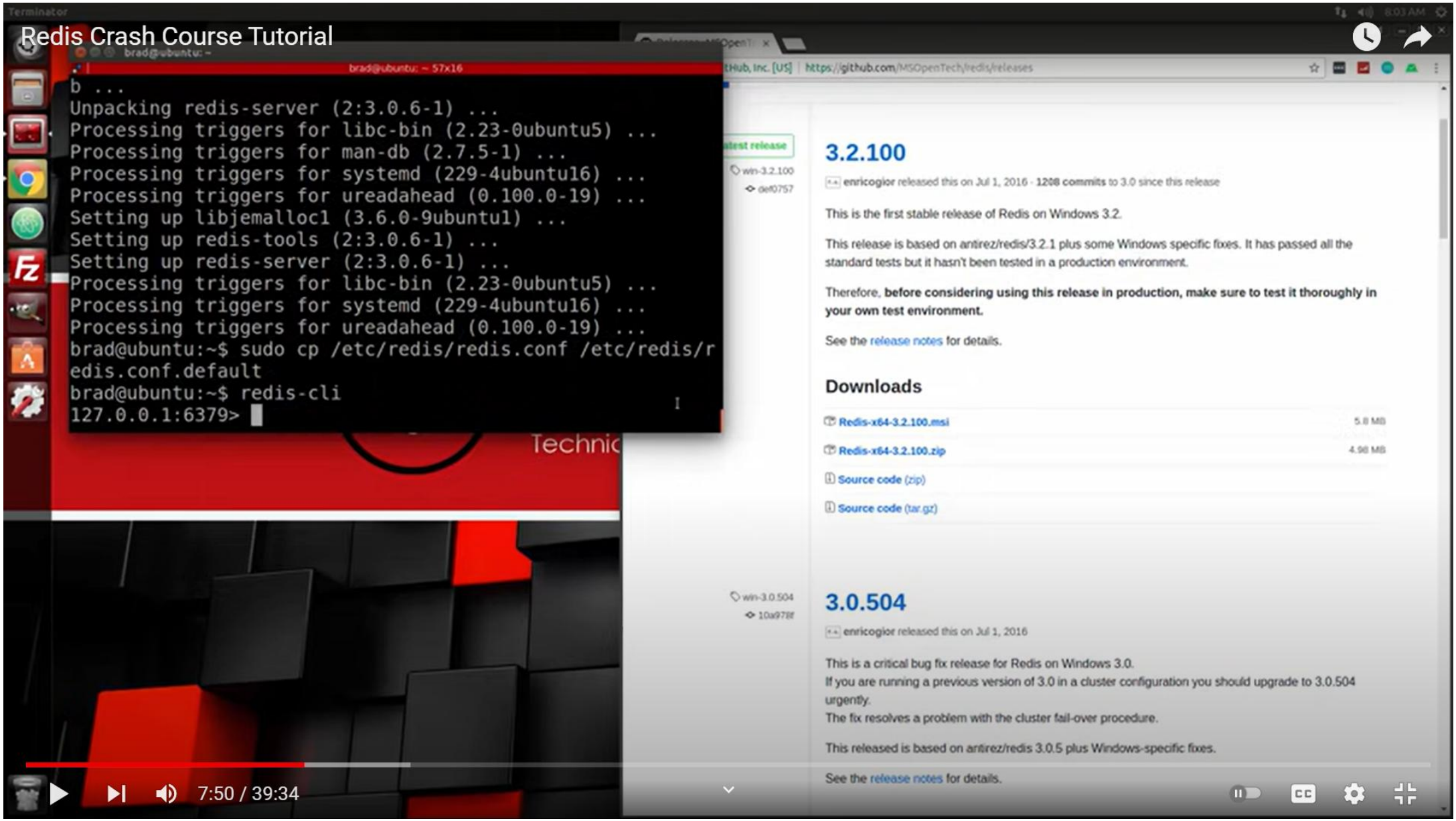
- ✓ Strings
- ✓ Lists
- ✓ Sets
- ✓ Sorted Sets
- ✓ Hashes
- ✓ Bitmaps
- ✓ Hyperlogs
- ✓ Geospatial Indexes

Advantages Of Redis

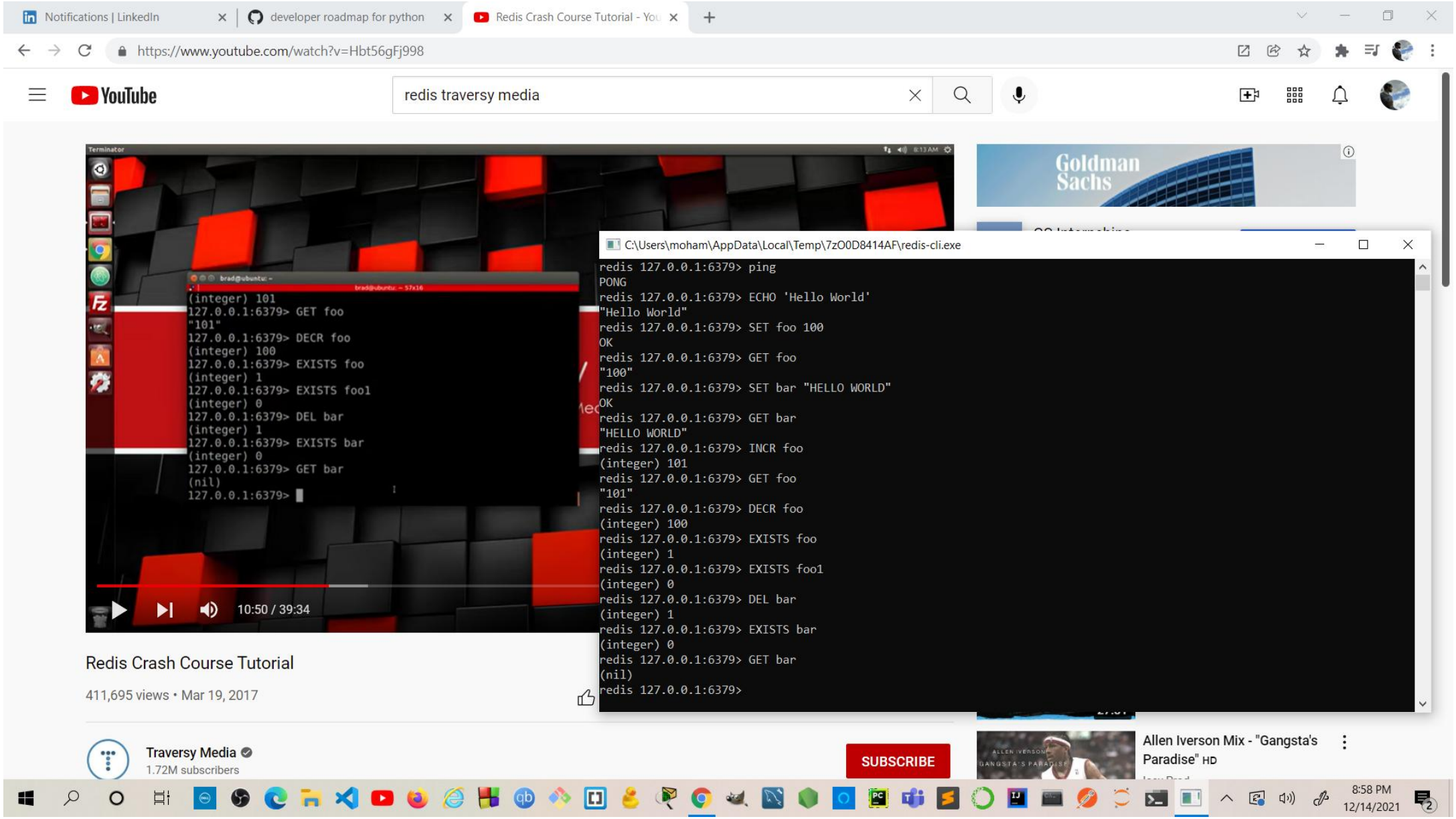
- ✓VERY Flexible
- ✓No Schemas & Column Names
- ✓Very Fast : Can perform around **110,000 SETs per second**, about **81,000 GETs per second**
- ✓Rich Datatype Support
- ✓Caching & Disk Persistence

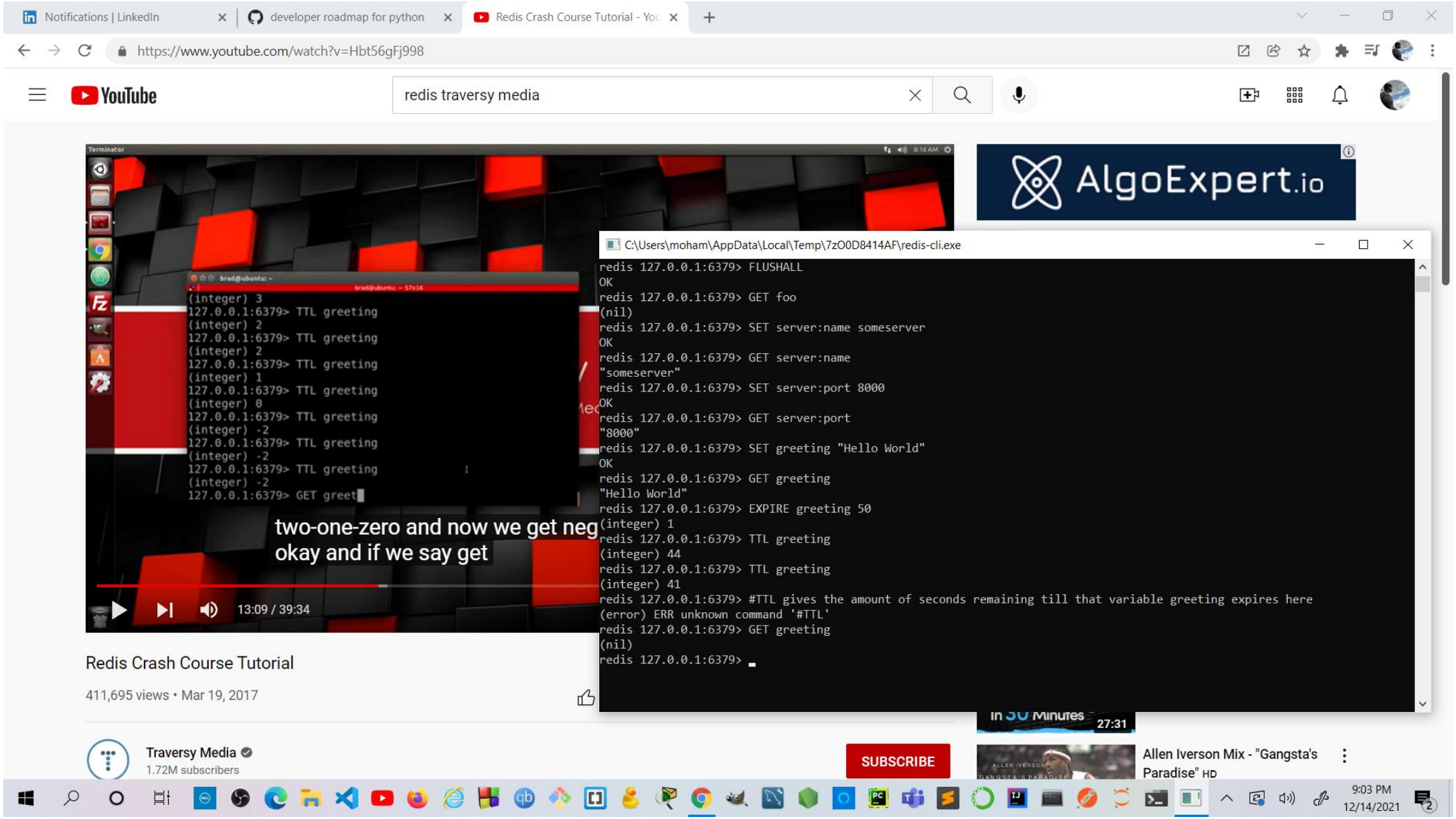
Redis & Security

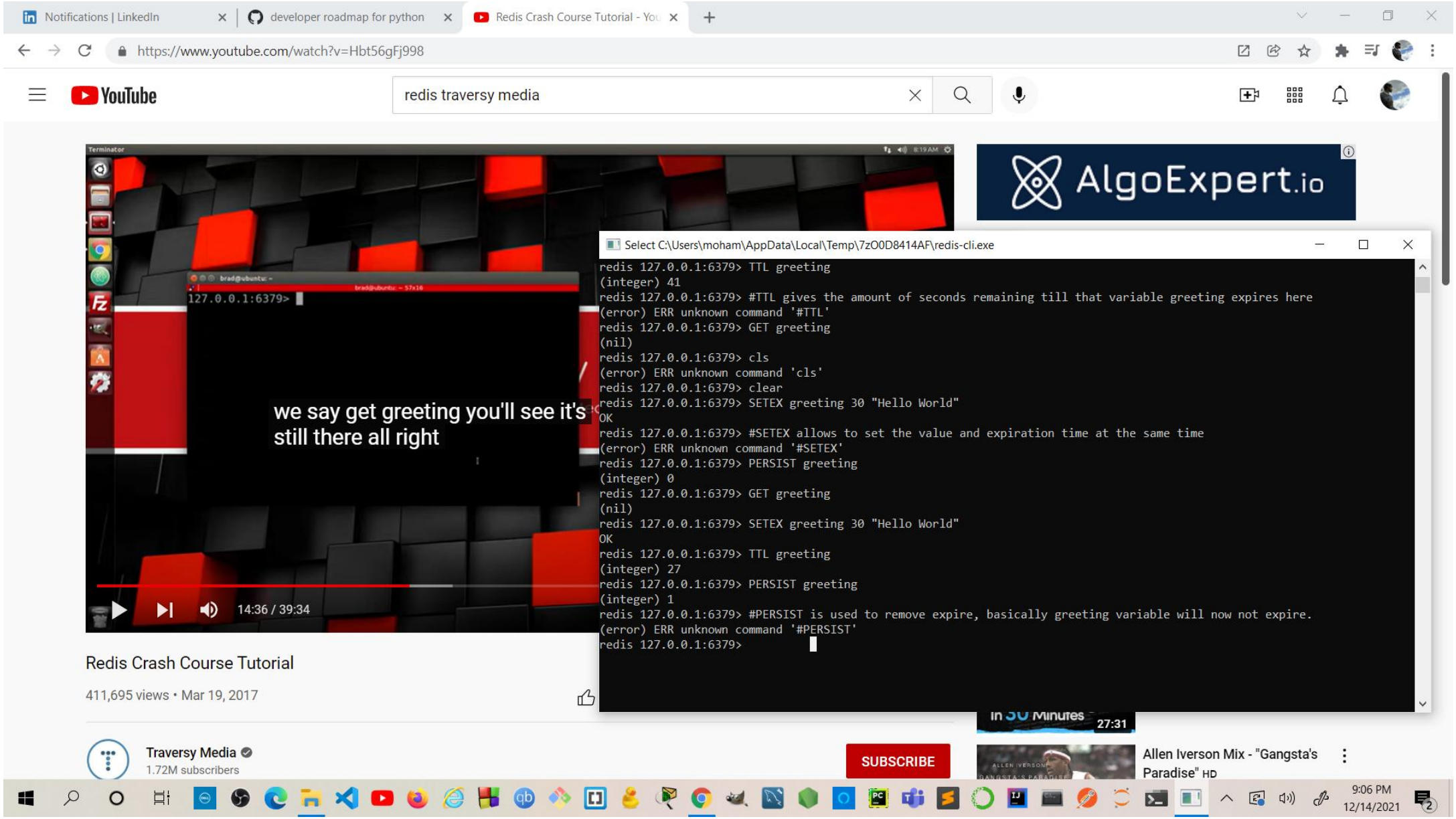
- ✓Designed to be accessed by trusted clients
- ✓Do not allow external access / Internet exposure
- ✓Simple authentication can be setup
- ✓Can be restricted to certain interfaces
- ✓Data encryption not supported

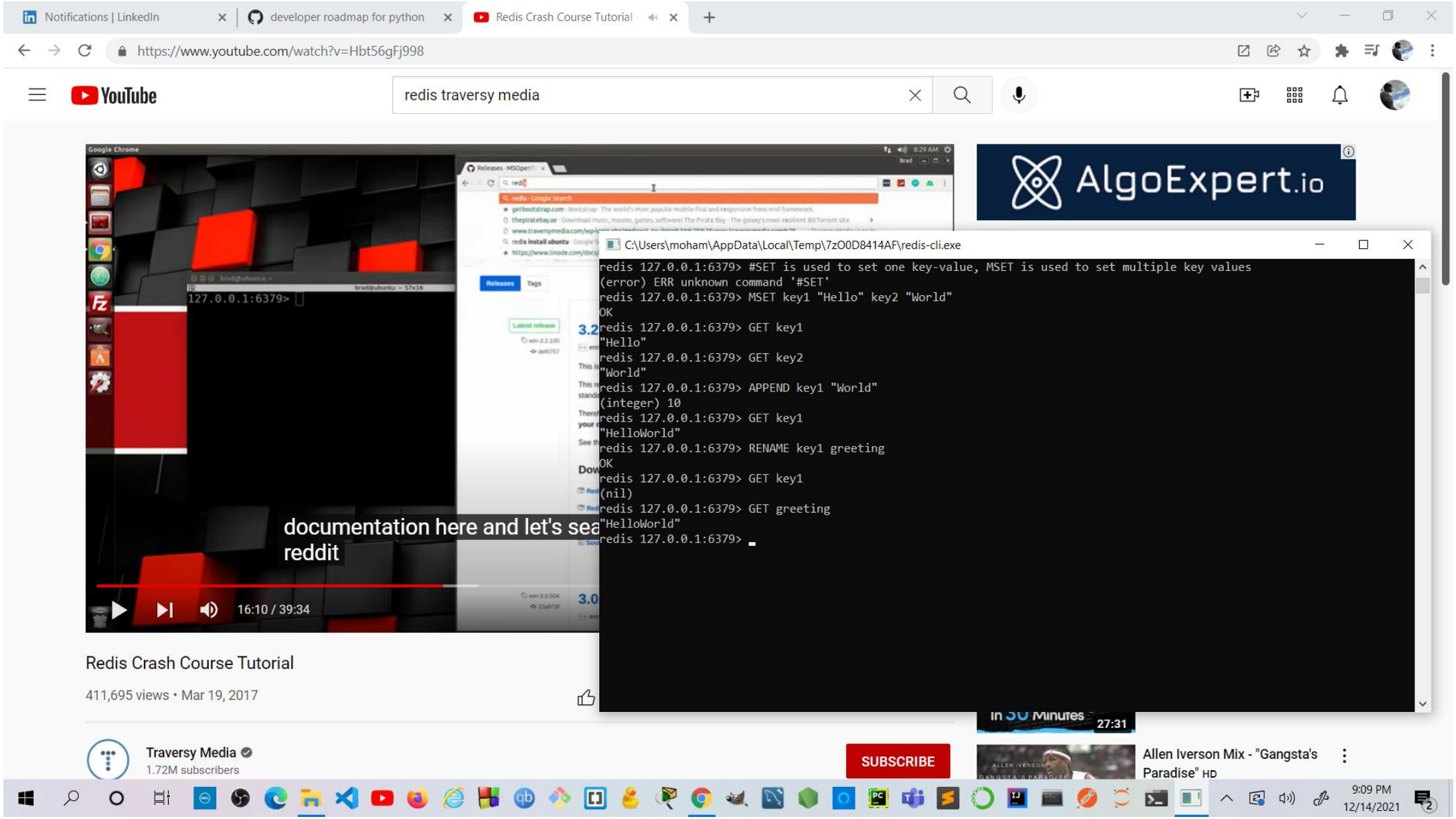

Redis Crash Course Tutorial
brad@ubuntu: ~
brad@ubuntu: ~ 57x16
b ...
Unpacking redis-server (2:3.0.6-1) ...
Processing triggers for libc-bin (2.23-0ubuntu5) ...
Processing triggers for man-db (2.7.5-1) ...
Processing triggers for systemd (229-4ubuntu16) ...
Processing triggers for ureadahead (0.100.0-19) ...
Setting up libjemalloc1 (3.6.0-9ubuntu1) ...
Setting up redis-tools (2:3.0.6-1) ...
Setting up redis-server (2:3.0.6-1) ...
Processing triggers for libc-bin (2.23-0ubuntu5) ...
Processing triggers for systemd (229-4ubuntu16) ...
Processing triggers for ureadahead (0.100.0-19) ...
brad@ubuntu:~$ sudo cp /etc/redis/redis.conf /etc/redis/r
edis.conf.default
brad@ubuntu:~$ redis-cli
127.0.0.1:6379>
https://github.com/MSOpenTech/redis/releases
3.2.100
enricogior released this on Jul 1, 2016 · 1208 commits to 3.0 since this release
This is the first stable release of Redis on Windows 3.2.
This release is based on antirez/redis/3.2.1 plus some Windows specific fixes. It has passed all the standard tests but it hasn't been tested in a production environment.
Therefore, before considering using this release in production, make sure to test it thoroughly in your own test environment.
See the release notes for details.
Downloads
Redis-x64-3.2.100.msi
5.8 MB
Redis-x64-3.2.100.zip
4.98 MB
Source code (zip)
Source code (tar.gz)
3.0.504
enricogior released this on Jul 1, 2016
This is a critical bug fix release for Redis on Windows 3.0.
If you are running a previous version of 3.0 in a cluster configuration you should upgrade to 3.0.504 urgently.
The fix resolves a problem with the cluster fail-over procedure.
This released is based on antirez/redis 3.0.5 plus Windows-specific fixes.
See the release notes for details.
7:50 / 39:34

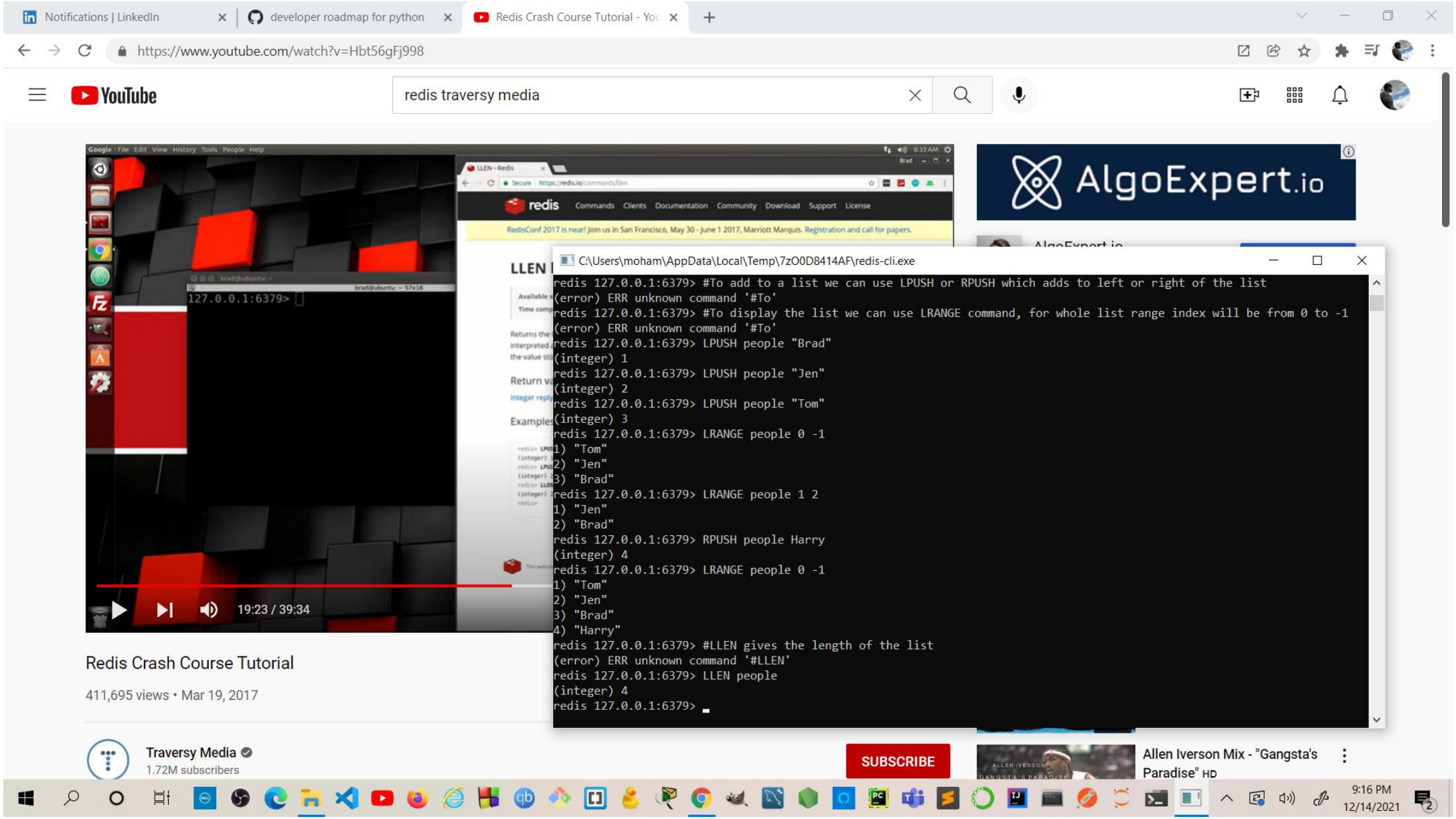

Terminator
8:06 AM
brad@ubuntu: ~
brad@ubuntu: ~ 57x16
127.0.0.1:6379> ping
PONG
127.0.0.1:6379> ECHO 'Hello World'
"Hello World"
127.0.0.1:6379> QUIT
Media
Media

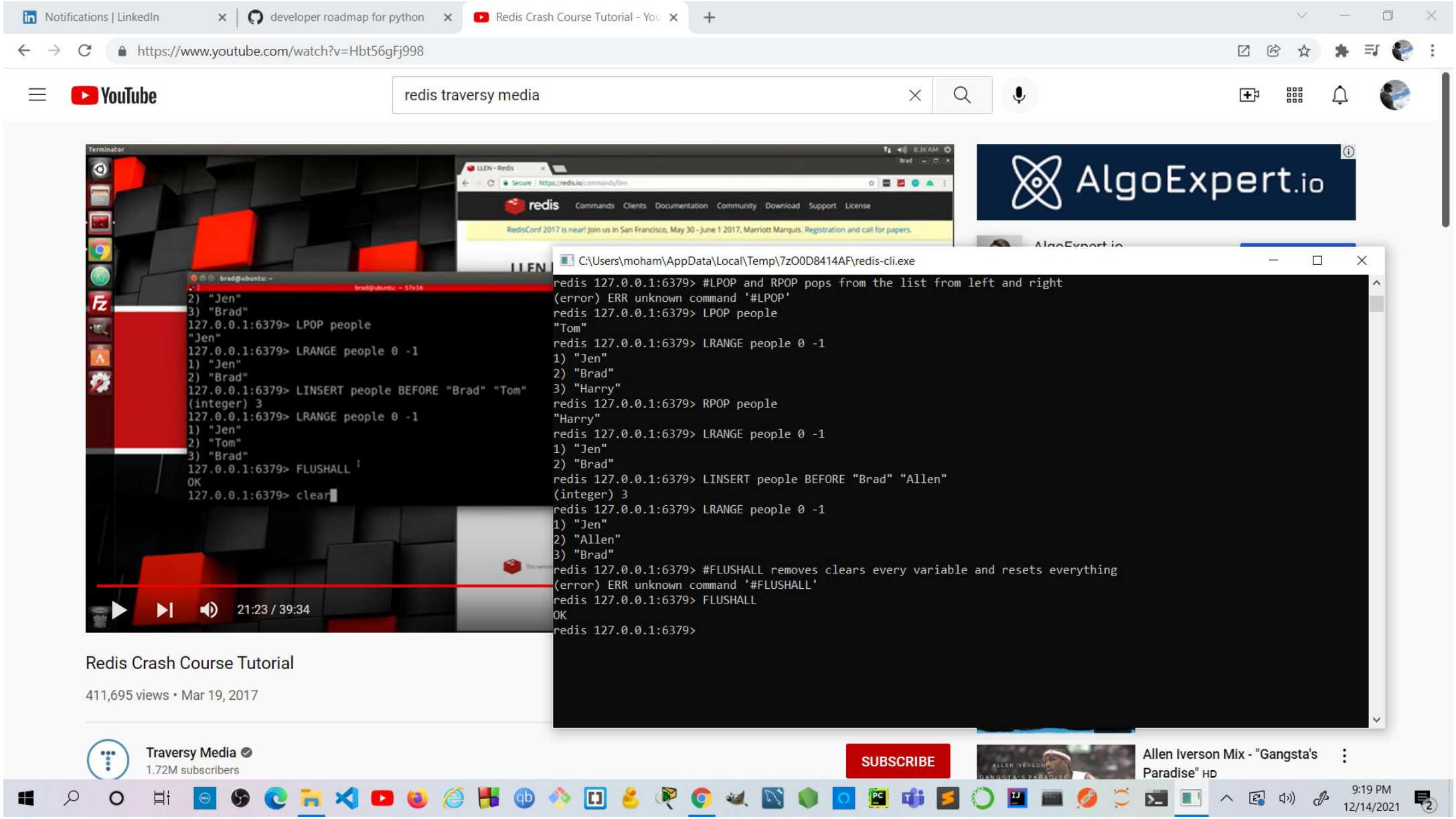

Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial - You
https://www.youtube.com/watch?v=Hbt56gFj998
YouTube
redis traversy media
Terminator
brad@ubuntu: ~
(integer) 101
127.0.0.1:6379> GET foo
"101"
127.0.0.1:6379> DECR foo
(integer) 100
127.0.0.1:6379> EXISTS foo
(integer) 1
127.0.0.1:6379> EXISTS foo1
(integer) 0
127.0.0.1:6379> DEL bar
(integer) 1
127.0.0.1:6379> EXISTS bar
(integer) 0
127.0.0.1:6379> GET bar
(nil)
127.0.0.1:6379>
10:50 / 39:34
C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe
redis 127.0.0.1:6379> ping
PONG
redis 127.0.0.1:6379> ECHO 'Hello World'
"Hello World"
redis 127.0.0.1:6379> SET foo 100
OK
redis 127.0.0.1:6379> GET foo
"100"
redis 127.0.0.1:6379> SET bar "HELLO WORLD"
OK
redis 127.0.0.1:6379> GET bar
"HELLO WORLD"
redis 127.0.0.1:6379> INCR foo
(integer) 101
redis 127.0.0.1:6379> GET foo
"101"
redis 127.0.0.1:6379> DECR foo
(integer) 100
redis 127.0.0.1:6379> EXISTS foo
(integer) 1
redis 127.0.0.1:6379> EXISTS foo1
(integer) 0
redis 127.0.0.1:6379> DEL bar
(integer) 1
redis 127.0.0.1:6379> EXISTS bar
(integer) 0
redis 127.0.0.1:6379> GET bar
(nil)
redis 127.0.0.1:6379>
Goldman Sachs
Redis Crash Course Tutorial
411,695 views • Mar 19, 2017
Traversy Media
1.72M subscribers
SUBSCRIBE
Allen Iverson Mix - "Gangsta's Paradise" HD
8:58 PM
12/14/2021

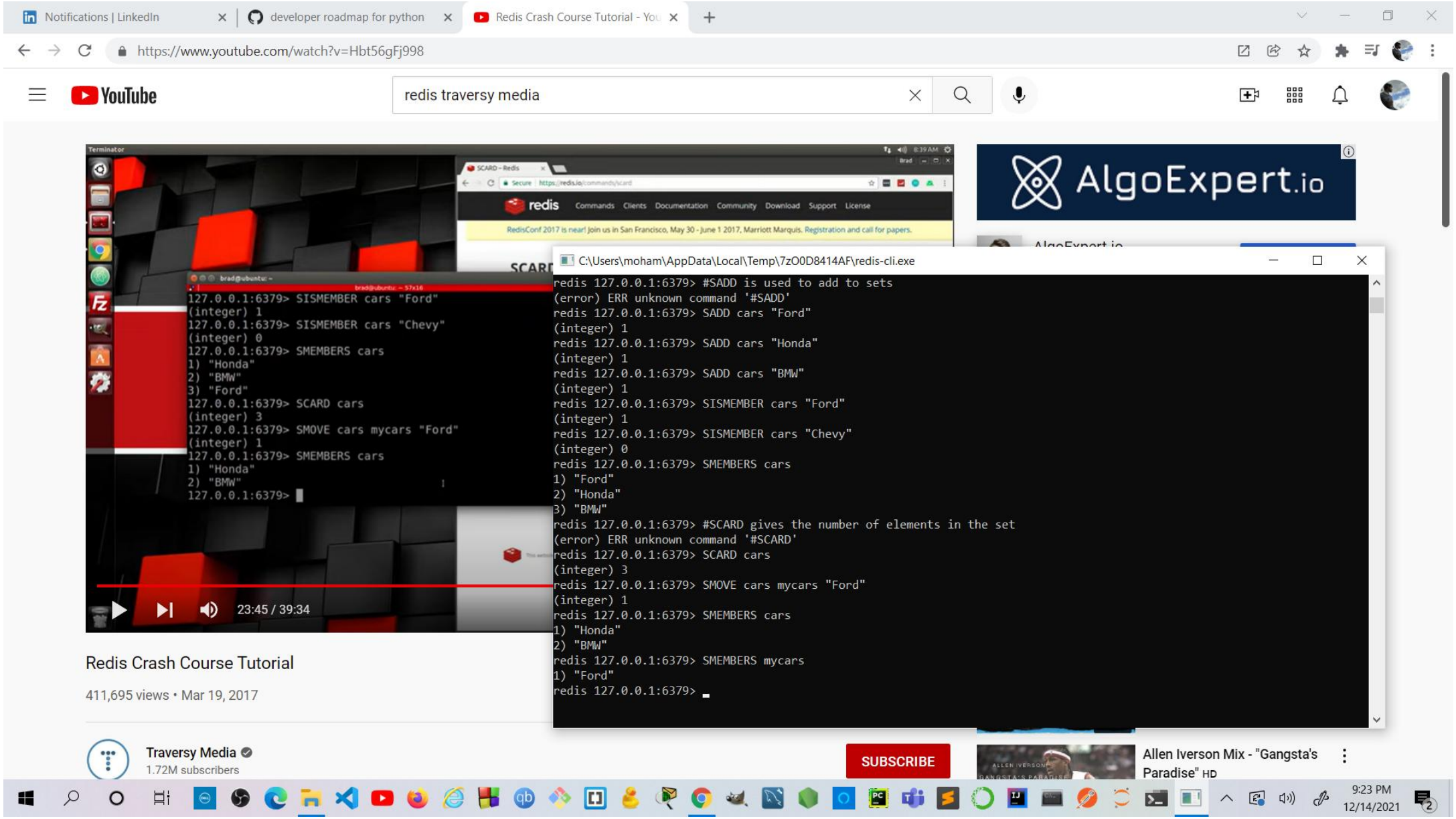

Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial - You
https://www.youtube.com/watch?v=Hbt56gFj998
YouTube
redis traversy media
AlgoExpert.io
(integer) 3
127.0.0.1:6379> TTL greeting
(integer) 2
127.0.0.1:6379> TTL greeting
(integer) 2
127.0.0.1:6379> TTL greeting
(integer) 1
127.0.0.1:6379> TTL greeting
(integer) 0
127.0.0.1:6379> TTL greeting
(integer) -2
127.0.0.1:6379> TTL greeting
(integer) -2
127.0.0.1:6379> TTL greeting
(integer) -2
127.0.0.1:6379> GET greet
two-one-zero and now we get neg
okay and if we say get
13:09 / 39:34
C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe
redis 127.0.0.1:6379> FLUSHALL
OK
redis 127.0.0.1:6379> GET foo
(nil)
redis 127.0.0.1:6379> SET server:name someserver
OK
redis 127.0.0.1:6379> GET server:name
"someserver"
redis 127.0.0.1:6379> SET server:port 8000
OK
redis 127.0.0.1:6379> GET server:port
"8000"
redis 127.0.0.1:6379> SET greeting "Hello World"
OK
redis 127.0.0.1:6379> GET greeting
"Hello World"
redis 127.0.0.1:6379> EXPIRE greeting 50
(integer) 1
redis 127.0.0.1:6379> TTL greeting
(integer) 44
redis 127.0.0.1:6379> TTL greeting
(integer) 41
redis 127.0.0.1:6379> #TTL gives the amount of seconds remaining till that variable greeting expires here
(error) ERR unknown command '#TTL'
redis 127.0.0.1:6379> GET greeting
(nil)
redis 127.0.0.1:6379>
Redis Crash Course Tutorial
411,695 views • Mar 19, 2017
Traversy Media
1.72M subscribers
SUBSCRIBE
27:31
Allen Iverson Mix - "Gangsta's Paradise" HD
9:03 PM
12/14/2021

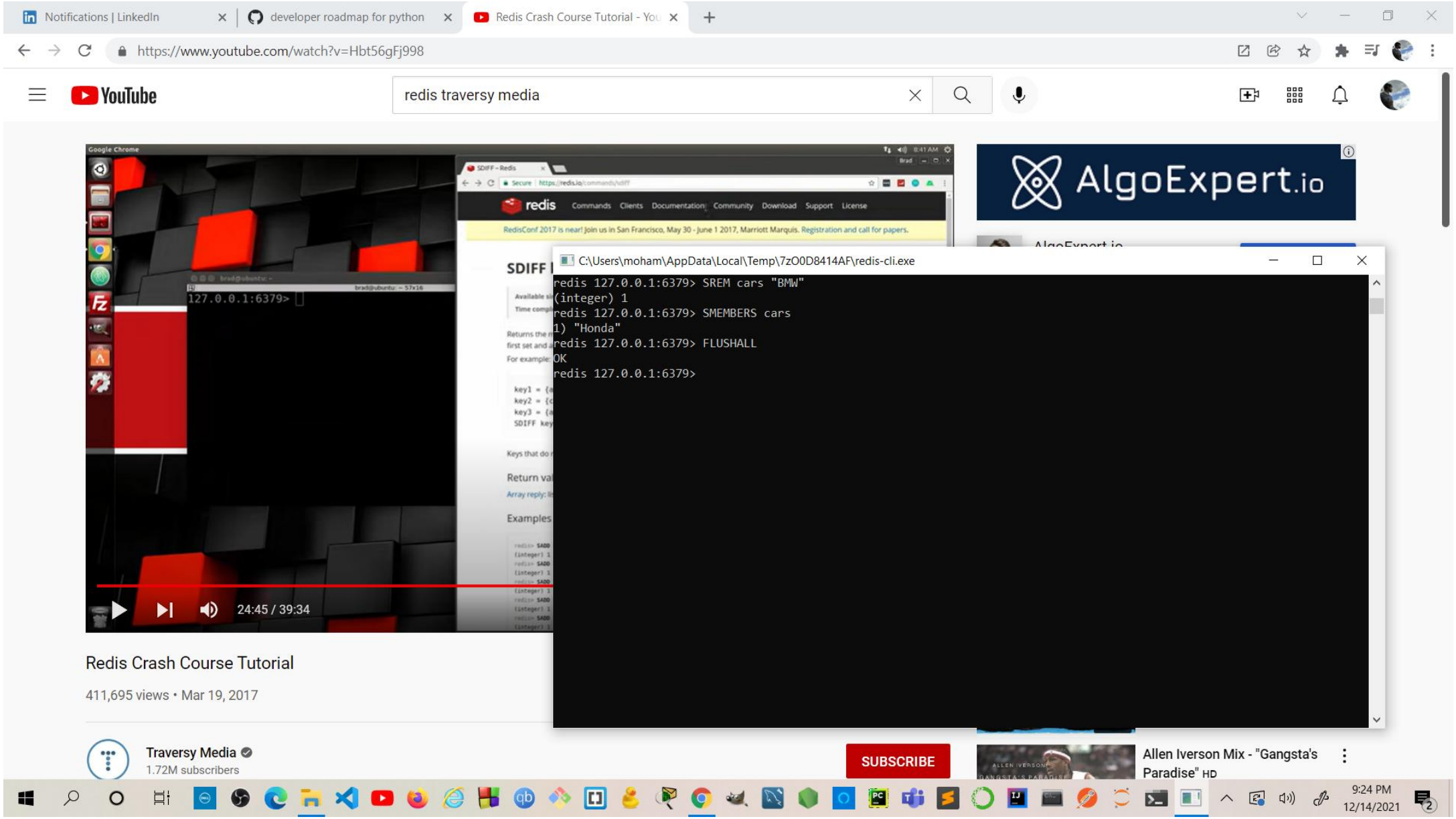

Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial - You
https://www.youtube.com/watch?v=Hbt56gFj998
YouTube
redis traversy media
AlgoExpert.io
we say get greeting you'll see it's still there all right
14:36 / 39:34
Select C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe
redis 127.0.0.1:6379> TTL greeting
(integer) 41
redis 127.0.0.1:6379> #TTL gives the amount of seconds remaining till that variable greeting expires here
(error) ERR unknown command '#TTL'
redis 127.0.0.1:6379> GET greeting
(nil)
redis 127.0.0.1:6379> cls
(error) ERR unknown command 'cls'
redis 127.0.0.1:6379> clear
redis 127.0.0.1:6379> SETEX greeting 30 "Hello World"
OK
redis 127.0.0.1:6379> #SETEX allows to set the value and expiration time at the same time
(error) ERR unknown command '#SETEX'
redis 127.0.0.1:6379> PERSIST greeting
(integer) 0
redis 127.0.0.1:6379> GET greeting
(nil)
redis 127.0.0.1:6379> SETEX greeting 30 "Hello World"
OK
redis 127.0.0.1:6379> TTL greeting
(integer) 27
redis 127.0.0.1:6379> PERSIST greeting
(integer) 1
redis 127.0.0.1:6379> #PERSIST is used to remove expire, basically greeting variable will now not expire.
(error) ERR unknown command '#PERSIST'
redis 127.0.0.1:6379>
Redis Crash Course Tutorial
411,695 views • Mar 19, 2017
Traversy Media
1.72M subscribers
SUBSCRIBE
27:31
Allen Iverson Mix - "Gangsta's Paradise" HD
9:06 PM
12/14/2021


Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial
https://www.youtube.com/watch?v=Hbt56gFj998
YouTube
redis traversy media
AlgoExpert.io
C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe
redis 127.0.0.1:6379> #SET is used to set one key-value, MSET is used to set multiple key values
(error) ERR unknown command '#SET'
redis 127.0.0.1:6379> MSET key1 "Hello" key2 "World"
OK
redis 127.0.0.1:6379> GET key1
"Hello"
redis 127.0.0.1:6379> GET key2
"World"
redis 127.0.0.1:6379> APPEND key1 "World"
(integer) 10
redis 127.0.0.1:6379> GET key1
"HelloWorld"
redis 127.0.0.1:6379> RENAME key1 greeting
OK
redis 127.0.0.1:6379> GET key1
(nil)
redis 127.0.0.1:6379> GET greeting
"HelloWorld"
redis 127.0.0.1:6379>
documentation here and let's sea
reddit
16:10 / 39:34
Redis Crash Course Tutorial
411,695 views • Mar 19, 2017
Traversy Media
1.72M subscribers
SUBSCRIBE
27:31
Allen Iverson Mix - "Gangsta's Paradise" HD
9:09 PM
12/14/2021


Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial - You
https://www.youtube.com/watch?v=Hbt56gFj998
YouTube
redis traversy media
AlgoExpert.io
C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe
redis 127.0.0.1:6379> #To add to a list we can use LPUSH or RPUSH which adds to left or right of the list
(error) ERR unknown command '#To'
redis 127.0.0.1:6379> #To display the list we can use LRANGE command, for whole list range index will be from 0 to -1
(error) ERR unknown command '#To'
redis 127.0.0.1:6379> LPUSH people "Brad"
(integer) 1
redis 127.0.0.1:6379> LPUSH people "Jen"
(integer) 2
redis 127.0.0.1:6379> LPUSH people "Tom"
(integer) 3
redis 127.0.0.1:6379> LRANGE people 0 -1
1) "Tom"
2) "Jen"
3) "Brad"
redis 127.0.0.1:6379> LRANGE people 1 2
1) "Jen"
2) "Brad"
redis 127.0.0.1:6379> RPUSH people Harry
(integer) 4
redis 127.0.0.1:6379> LRANGE people 0 -1
1) "Tom"
2) "Jen"
3) "Brad"
4) "Harry"
redis 127.0.0.1:6379> #LLEN gives the length of the list
(error) ERR unknown command '#LLEN'
redis 127.0.0.1:6379> LLEN people
(integer) 4
redis 127.0.0.1:6379>
19:23 / 39:34
Redis Crash Course Tutorial
411,695 views • Mar 19, 2017
Traversy Media
1.72M subscribers
SUBSCRIBE
Allen Iverson Mix - "Gangsta's Paradise" HD
9:16 PM
12/14/2021


Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial - You
https://www.youtube.com/watch?v=Hbt56gFj998
YouTube
redis traversy media
AlgoExpert.io
C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe
redis 127.0.0.1:6379> #LPOP and RPOP pops from the list from left and right
(error) ERR unknown command '#LPOP'
redis 127.0.0.1:6379> LPOP people
"Tom"
redis 127.0.0.1:6379> LRANGE people 0 -1
1) "Jen"
2) "Brad"
3) "Harry"
redis 127.0.0.1:6379> RPOP people
"Harry"
redis 127.0.0.1:6379> LRANGE people 0 -1
1) "Jen"
2) "Brad"
redis 127.0.0.1:6379> LINSERT people BEFORE "Brad" "Allen"
(integer) 3
redis 127.0.0.1:6379> LRANGE people 0 -1
1) "Jen"
2) "Allen"
3) "Brad"
redis 127.0.0.1:6379> #FLUSHALL removes clears every variable and resets everything
(error) ERR unknown command '#FLUSHALL'
redis 127.0.0.1:6379> FLUSHALL
OK
redis 127.0.0.1:6379>
21:23 / 39:34
Redis Crash Course Tutorial
411,695 views • Mar 19, 2017
Traversy Media
1.72M subscribers
SUBSCRIBE
Allen Iverson Mix - "Gangsta's Paradise" HD
9:19 PM
12/14/2021


Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial - You
https://www.youtube.com/watch?v=Hbt56gFj998
YouTube
redis traversy media
AlgoExpert.io
C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe
redis 127.0.0.1:6379> #SADD is used to add to sets
(error) ERR unknown command '#SADD'
redis 127.0.0.1:6379> SADD cars "Ford"
(integer) 1
redis 127.0.0.1:6379> SADD cars "Honda"
(integer) 1
redis 127.0.0.1:6379> SADD cars "BMW"
(integer) 1
redis 127.0.0.1:6379> SISMEMBER cars "Ford"
(integer) 1
redis 127.0.0.1:6379> SISMEMBER cars "Chevy"
(integer) 0
redis 127.0.0.1:6379> SMEMBERS cars
1) "Ford"
2) "Honda"
3) "BMW"
redis 127.0.0.1:6379> #SCARD gives the number of elements in the set
(error) ERR unknown command '#SCARD'
redis 127.0.0.1:6379> SCARD cars
(integer) 3
redis 127.0.0.1:6379> SMOVE cars mycars "Ford"
(integer) 1
redis 127.0.0.1:6379> SMEMBERS cars
1) "Honda"
2) "BMW"
redis 127.0.0.1:6379> SMEMBERS mycars
1) "Ford"
redis 127.0.0.1:6379>
23:45 / 39:34
Redis Crash Course Tutorial
411,695 views • Mar 19, 2017
Traversy Media
1.72M subscribers
SUBSCRIBE
Allen Iverson Mix - "Gangsta's Paradise" HD
9:23 PM
12/14/2021


Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial - You
https://www.youtube.com/watch?v=Hbt56gFj998
YouTube
redis traversy media
AlgoExpert.io
C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe
redis 127.0.0.1:6379> SREM cars "BMW"
(integer) 1
redis 127.0.0.1:6379> SMEMBERS cars
1) "Honda"
redis 127.0.0.1:6379> FLUSHALL
OK
redis 127.0.0.1:6379>
24:45 / 39:34
Redis Crash Course Tutorial
411,695 views • Mar 19, 2017
Traversy Media
1.72M subscribers
SUBSCRIBE
Allen Iverson Mix - "Gangsta's Paradise" HD
9:24 PM
12/14/2021

A hash with a few fields (where few means up to one hundred or so) is stored in a way that takes very little space, so you can store millions of objects in a small Redis instance.

While Hashes are used mainly to represent objects, they are capable of storing many elements, so you can use Hashes for many other tasks as well.

Every hash can store up to 2^{32} - 1 field-value pairs (more than 4 billion).

Check the full list of Hash commands for more information, or read the introduction to Redis data types.

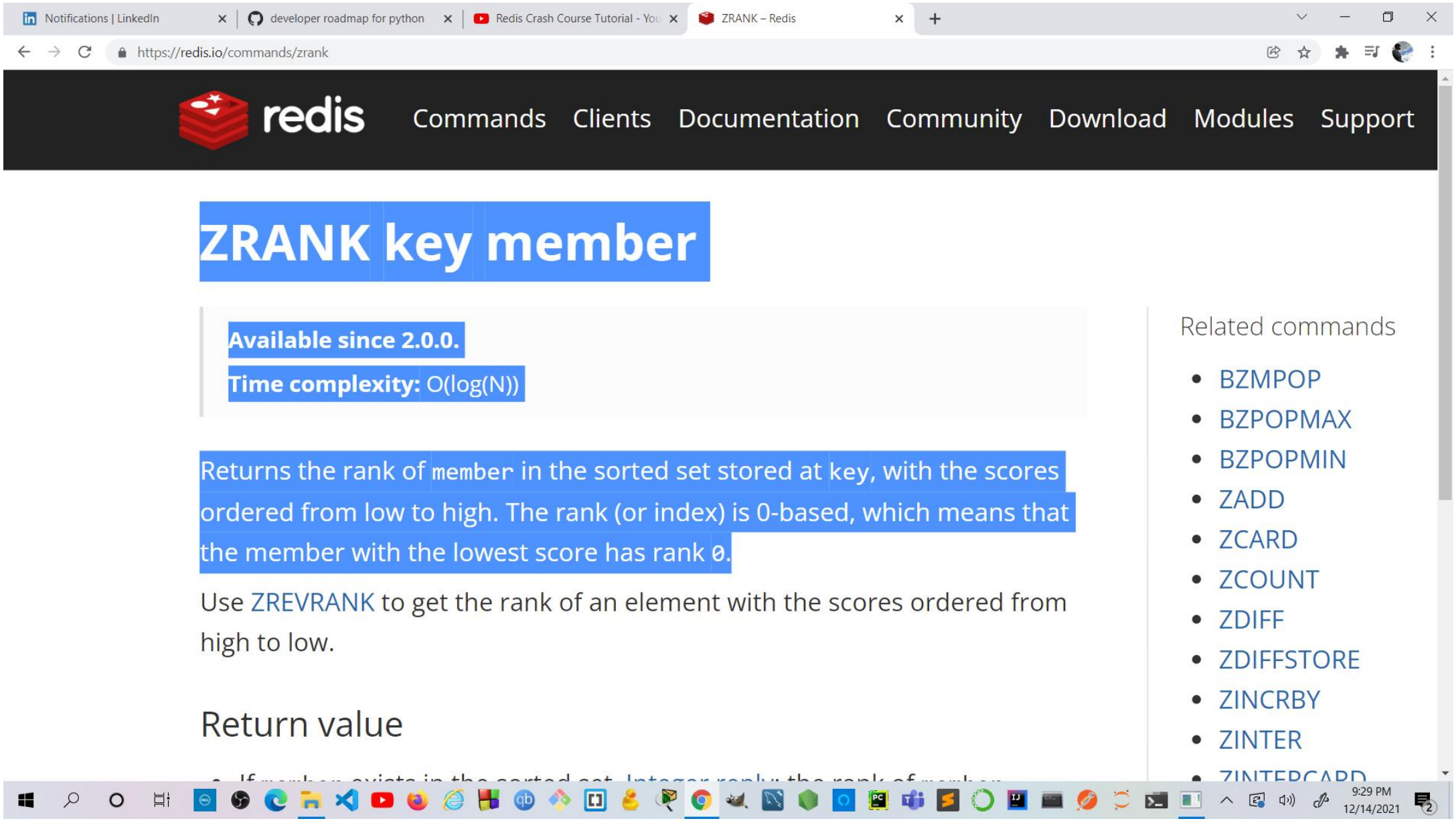
Sorted Sets

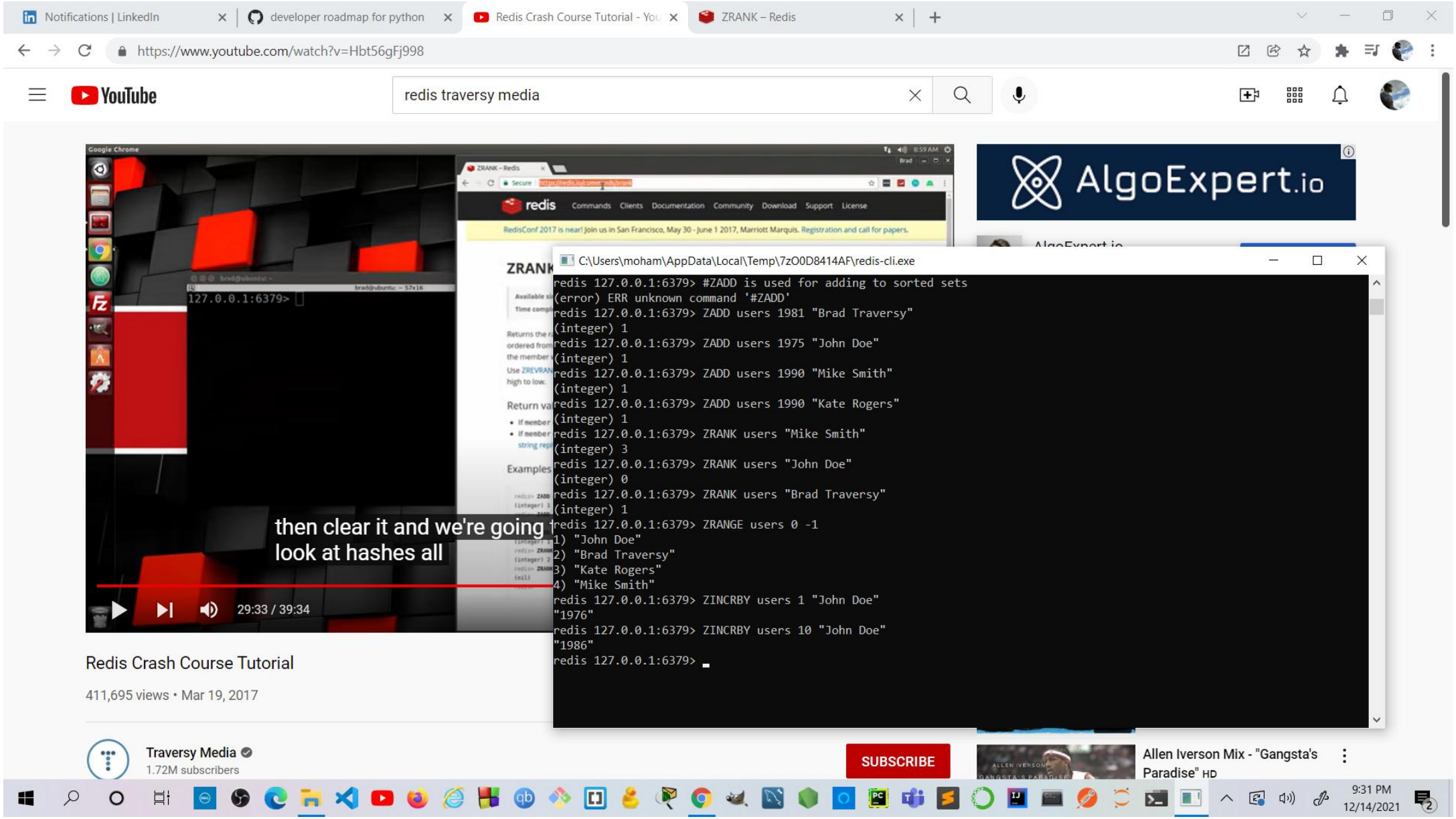
Redis Sorted Sets are, similarly to Redis Sets, non repeating collections of Strings. The difference is that every member of a Sorted Set is associated with a score, that is used keep the Sorted Set in order, from the smallest to the greatest score. While members are unique, scores may be repeated.

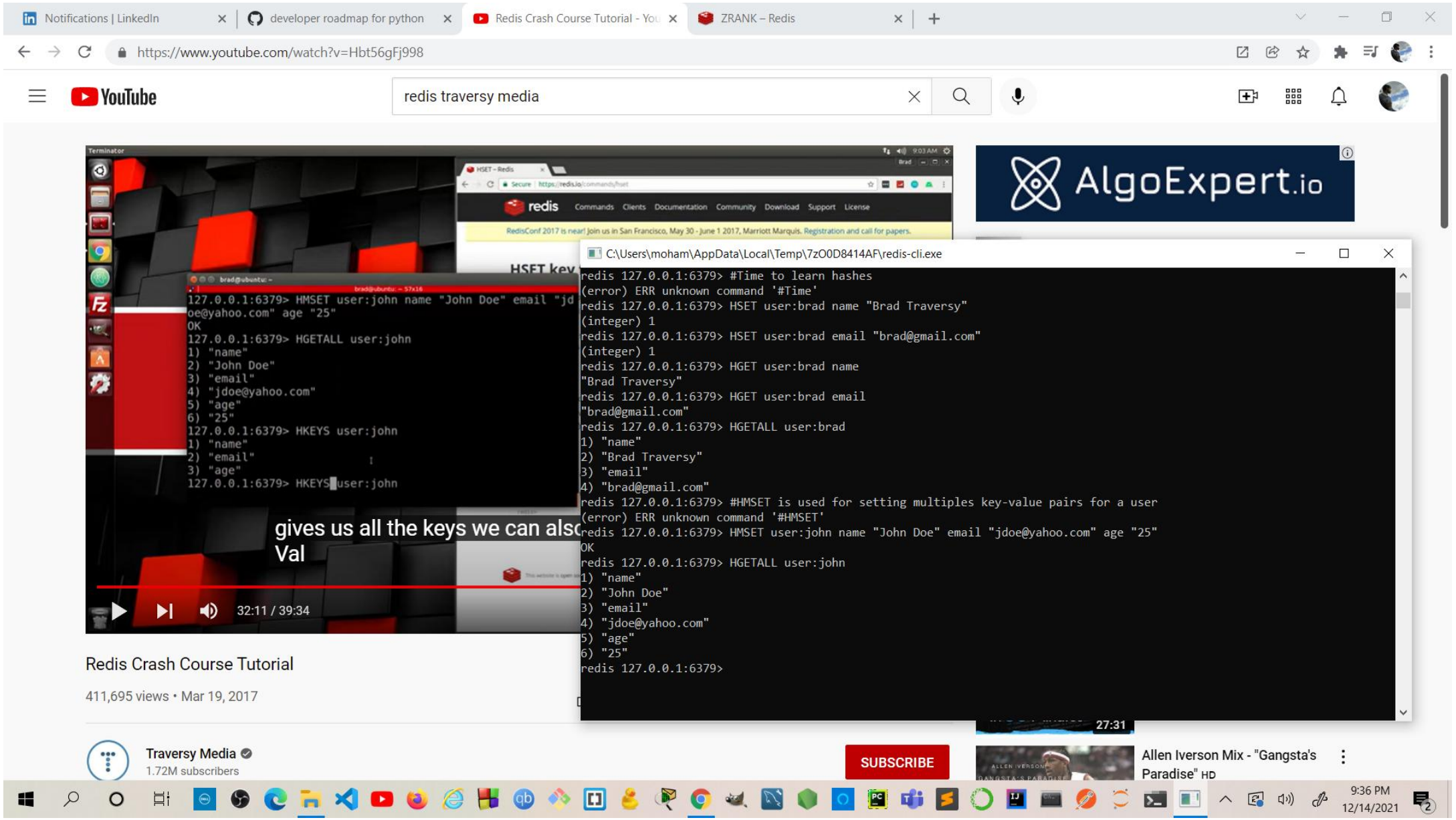
With Sorted Sets you can add, remove, or update elements in a very fast way (in a time proportional to the logarithm of the number of elements). Since elements are *stored in order* and not ordered afterwards, you can also get ranges by score or by rank (position) in a very fast way. Accessing the middle of a Sorted Set is also very fast, so you can use Sorted Sets as a smart list of non repeating elements where you can quickly access everything you need: elements in order, fast existence test, fast access to elements in the middle!

In short with Sorted Sets you can do a lot of tasks with great performance that are really hard to model in other kind of databases.

With Sorted Sets you can:


Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial - You
ZRANK – Redis
https://redis.io/commands/zrank
redis
Commands
Clients
Documentation
Community
Download
Modules
Support
ZRANK key member
Available since 2.0.0.
Time complexity: O(log(N))
Returns the rank of member in the sorted set stored at key, with the scores ordered from low to high. The rank (or index) is 0-based, which means that the member with the lowest score has rank 0.
Use ZREVRANK to get the rank of an element with the scores ordered from high to low.
Return value
Related commands
BZMPOP
BZPOPMAX
BZPOPMIN
ZADD
ZCARD
ZCOUNT
ZDIFF
ZDIFFSTORE
ZINCRBY
ZINTER
9:29 PM
12/14/2021


Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial - Yo
ZRANK – Redis
https://www.youtube.com/watch?v=Hbt56gFj998
YouTube
redis traversy media
AlgoExpert.io
C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe
redis 127.0.0.1:6379> #ZADD is used for adding to sorted sets
(error) ERR unknown command '#ZADD'
redis 127.0.0.1:6379> ZADD users 1981 "Brad Traversy"
(integer) 1
redis 127.0.0.1:6379> ZADD users 1975 "John Doe"
(integer) 1
redis 127.0.0.1:6379> ZADD users 1990 "Mike Smith"
(integer) 1
redis 127.0.0.1:6379> ZADD users 1990 "Kate Rogers"
(integer) 1
redis 127.0.0.1:6379> ZRANK users "Mike Smith"
(integer) 3
redis 127.0.0.1:6379> ZRANK users "John Doe"
(integer) 0
redis 127.0.0.1:6379> ZRANK users "Brad Traversy"
(integer) 1
redis 127.0.0.1:6379> ZRANGE users 0 -1
1) "John Doe"
2) "Brad Traversy"
3) "Kate Rogers"
4) "Mike Smith"
redis 127.0.0.1:6379> ZINCRBY users 1 "John Doe"
"1976"
redis 127.0.0.1:6379> ZINCRBY users 10 "John Doe"
"1986"
redis 127.0.0.1:6379>
then clear it and we're going
look at hashes all
29:33 / 39:34
Redis Crash Course Tutorial
411,695 views • Mar 19, 2017
Traversy Media
1.72M subscribers
SUBSCRIBE
Allen Iverson Mix - "Gangsta's Paradise" HD
9:31 PM
12/14/2021

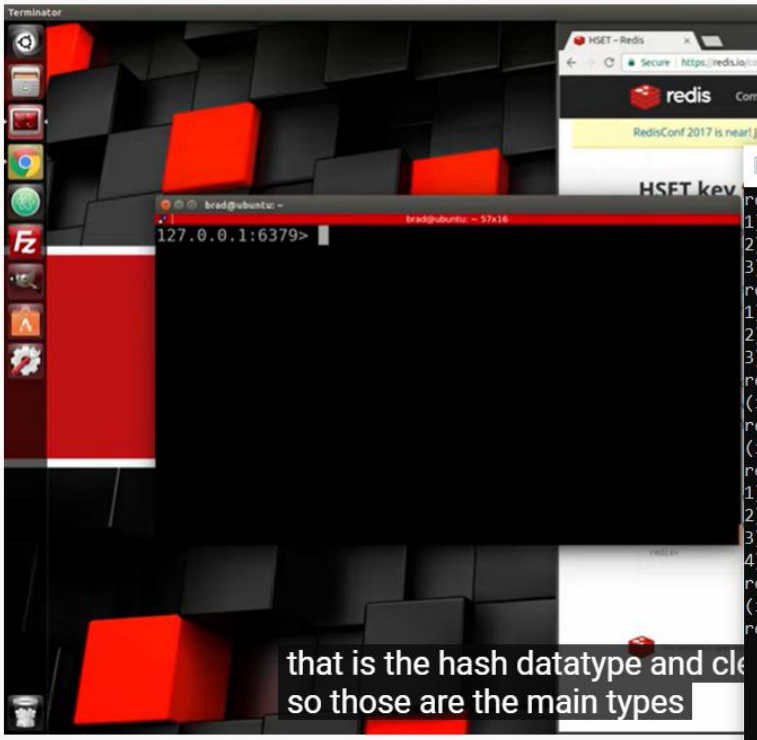

Notifications | LinkedIn
developer roadmap for python
Redis Crash Course Tutorial - Yo
ZRANK – Redis
https://www.youtube.com/watch?v=Hbt56gFj998
YouTube
redis traversy media
AlgoExpert.io
C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe
redis 127.0.0.1:6379> #Time to learn hashes
(error) ERR unknown command '#Time'
redis 127.0.0.1:6379> HSET user:brad name "Brad Traversy"
(integer) 1
redis 127.0.0.1:6379> HSET user:brad email "brad@gmail.com"
(integer) 1
redis 127.0.0.1:6379> HGET user:brad name
"Brad Traversy"
redis 127.0.0.1:6379> HGET user:brad email
"brad@gmail.com"
redis 127.0.0.1:6379> HGETALL user:brad
1) "name"
2) "Brad Traversy"
3) "email"
4) "brad@gmail.com"
redis 127.0.0.1:6379> #HMSET is used for setting multiples key-value pairs for a user
(error) ERR unknown command '#HMSET'
redis 127.0.0.1:6379> HMSET user:john name "John Doe" email "jdoe@yahoo.com" age "25"
OK
redis 127.0.0.1:6379> HGETALL user:john
1) "name"
2) "John Doe"
3) "email"
4) "jdoe@yahoo.com"
5) "age"
6) "25"
redis 127.0.0.1:6379>
gives us all the keys we can alsc
Val
32:11 / 39:34
Redis Crash Course Tutorial
411,695 views • Mar 19, 2017
Traversy Media
1.72M subscribers
SUBSCRIBE
27:31
Allen Iverson Mix - "Gangsta's Paradise" HD
9:36 PM
12/14/2021

Notifications | LinkedIn | developer roadmap for python | Redis Crash Course Tutorial | ZRANK – Redis

https://www.youtube.com/watch?v=Hbt56gFj998

YouTube

redis traversy media



C:\Users\moham\AppData\Local\Temp\7zO0D8414AF\redis-cli.exe

```
redis 127.0.0.1:6379> HKEYS user:john
1) "name"
2) "email"
3) "age"
redis 127.0.0.1:6379> HVALS user:john
1) "John Doe"
2) "jdoe@yahoo.com"
3) "25"
redis 127.0.0.1:6379> HINCRBY user:john age 1
(integer) 26
redis 127.0.0.1:6379> HDEL user:john age
(integer) 1
redis 127.0.0.1:6379> HGETALL user:john
1) "name"
2) "John Doe"
3) "email"
4) "jdoe@yahoo.com"
redis 127.0.0.1:6379> HLEN user:john
(integer) 2
redis 127.0.0.1:6379>
```

AlgoExpert.io

Redis Crash Course Tutorial

411,695 views • Mar 19, 2017

Traversy Media

1.72M subscribers

SUBSCRIBE

27:31

Allen Iverson Mix - "Gangsta's Paradise" HD

9:37 PM
12/14/2021

This page provides a technical description of Redis persistence, it is a suggested read for all Redis users. For a wider overview of Redis persistence and the durability guarantees it provides you may also want to read Redis persistence demystified.

Redis Persistence

Redis provides a different range of persistence options:

- **RDB** (Redis Database): The RDB persistence performs point-in-time snapshots of your dataset at specified intervals.
- **AOF** (Append Only File): The AOF persistence logs every write operation received by the server, that will be played again at server startup, reconstructing the original dataset. Commands are logged using the same format as the Redis protocol itself, in an append-only fashion. Redis is able to rewrite the log in the background when it gets too big.
- **No persistence**: If you wish, you can disable persistence completely, if you want your data to just exist as long as the server is running.
- **RDB + AOF**: It is possible to combine both AOF and RDB in the same instance. Notice that, in this case, when Redis restarts the AOF file will be used to reconstruct the original dataset since it is guaranteed to be the most complete.

The most important thing to understand is the different trade-offs between the RDB and AOF persistence. Let's start with RDB:

Note: for all these reasons we'll likely end up unifying AOF and RDB into a single persistence model in the future (long term plan).

The following sections will illustrate a few more details about the two persistence models.

Snapshotting

By default Redis saves snapshots of the dataset on disk, in a binary file called `dump.rdb`. You can configure Redis to have it save the dataset every N seconds if there are at least M changes in the dataset, or you can manually call the SAVE or BGSAVE commands.

For example, this configuration will make Redis automatically dump the dataset to disk every 60 seconds if at least 1000 keys changed:

```
save 60 1000
```

This strategy is known as *snapshotting*.

How it works

Whenever Redis needs to dump the dataset to disk, this is what happens:

Snapshotting

By default Redis saves snapshots of the dataset on disk, in a binary file called `dump.rdb`. You can configure Redis to have it save the dataset every N seconds if there are at least M changes in the dataset, or you can manually call the SAVE or BGSAVE commands.

For example, this configuration will make Redis automatically dump the dataset to disk every 60 seconds if at least 1000 keys changed:

```
save 60 1000
```

This strategy is known as *snapshotting*.

How it works

Whenever Redis needs to dump the dataset to disk, this is what happens:

- Redis forks. We now have a child and a parent process.
- The child starts to write the dataset to a temporary RDB file.
- When the child is done writing the new RDB file, it replaces the old one.

This method allows Redis to benefit from copy-on-write semantics.

- Redis forks. We now have a child and a parent process.
- The child starts to write the dataset to a temporary RDB file.
- When the child is done writing the new RDB file, it replaces the old one.

This method allows Redis to benefit from copy-on-write semantics.

Append-only file

Snapshotting is not very durable. If your computer running Redis stops, your power line fails, or you accidentally `kill -9` your instance, the latest data written on Redis will get lost. While this may not be a big deal for some applications, there are use cases for full durability, and in these cases Redis was not a viable option.

The *append-only file* is an alternative, fully-durable strategy for Redis. It became available in version 1.1.

You can turn on the AOF in your configuration file:

```
appendonly yes
```

From now on, every time Redis receives a command that changes the dataset (e.g. SET) it will append it to the AOF. When you restart Redis it will re-play the AOF to rebuild the state.

Log rewriting

As you can guess, the AOF gets bigger and bigger as write operations are performed. For example, if you are incrementing a counter 100 times, you'll end up with a single key in your dataset containing the final value, but 100