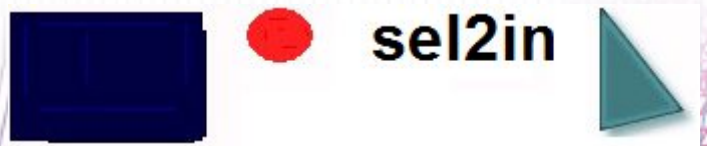
# Local and Remote Cache

Enterprise Applications
Web
Desktop (Applet, Webstart, Standalone)
Server-server applications

-Tushar Kapila

# Format & Information

- Questions at the end, but you can use the chat to frame them during the talk
- Contact : tushar@sel2in.com
- Concepts with one sample use



# What can you cache

- Images
- CSS, Javascript resources
- Audio
- Calculated values – encryption values
- Parts of a page (that are static or common to many users)
- Anything that takes time to create, or is requested by many users

# Cache Implementations

- Memcache
- EhCache
- Quick Cache
- Varnish
- JCS – Apache
- mod_cache in HTTPD Apache web server
- Whirly, custom, Java Maps (LRU)
- Squid and many more

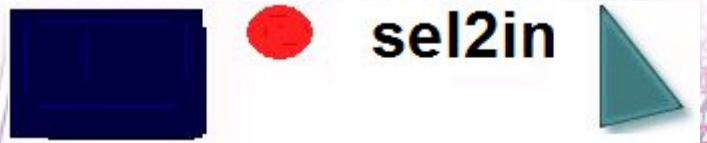
# Why use it

- Scale and usability

When issues with latency and response time:

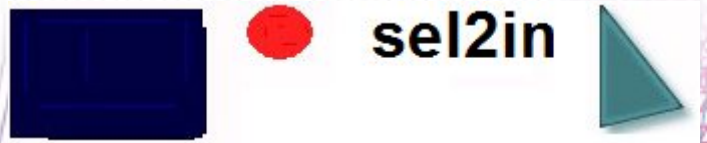
- Programmers want to knock off features
- Ops person wants better network
- Systems team wants bigger servers …

What we need is a solution that is more far reaching



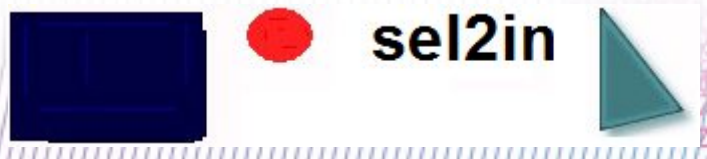
# Why use it

- Network access to other service – WS, SOAP, legacy, REST
- Data base access – slow, expensive, catastrophic if you get too many requests (one place of failure if not horizontally scalable)



# Two main Types

- In Process – Same JVM.
- Out Process – remote JVM (same box or seperate boxes)



# Web app set up

Browser

Local Cache, Network Cache

Fire Wall 1

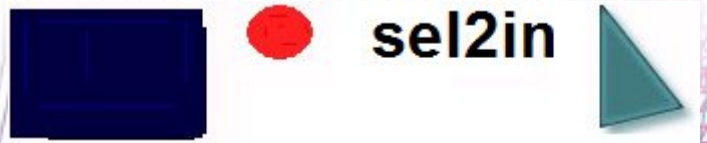
Network distribute
Reverse Proxy, Web server, DMZ

Fire Wall 2

App Server

Database

Control



# But why Do Caches work?

- Speed of access disk or network resource
- Speed of re calculating value (encryption, complex calculation)
- Registers, L1,2 Cache, RAM Read Speed, Network, Disk, Database
- Simple Application

# Data On Speed

| | |
|---|---|
| • L1 cache reference | 0.5 ns |
| • Branch mispredict | 5 ns |
| • L2 cache reference | 7 ns |
| • Mutex lock/unlock | 100 ns |
| • Main memory reference | 100 ns |
| • Compress 1K bytes with Zippy | 10,000 ns |

# Speed Data

| | |
|---|---|
| • Send 2K bytes over 1 Gbps network | 20,000 ns |
| • Read 1 MB sequentially from memory | 250,000 ns |
| • Round trip within same datacenter | 500,000 ns |
| • Disk seek | 10,000,000 ns |
| • Read 1 MB sequentially from network | 10,000,000 ns |
| • Read 1 MB sequentially from disk | 30,000,000 ns |
| • Send packet CA->Netherlands->CA | 150,000,000 ns |

* Source http://serverfault.com/questions/238417/are-networks-now-faster-than-disks

# Cache Set Up

- Quick Cache
  https://code.google.com/p/quickcached/
- Start server on a port

  QuickCached -p 11212
-

# Setup : Test from prompt/ Terminal

telnet localhost 11211

Trying ::1...

Connected to localhost.

Escape character is '^]'.

get key1

END

set key1 0 3600 4 ( TTL 3600 and 4 bytes)

ABCD

STORED

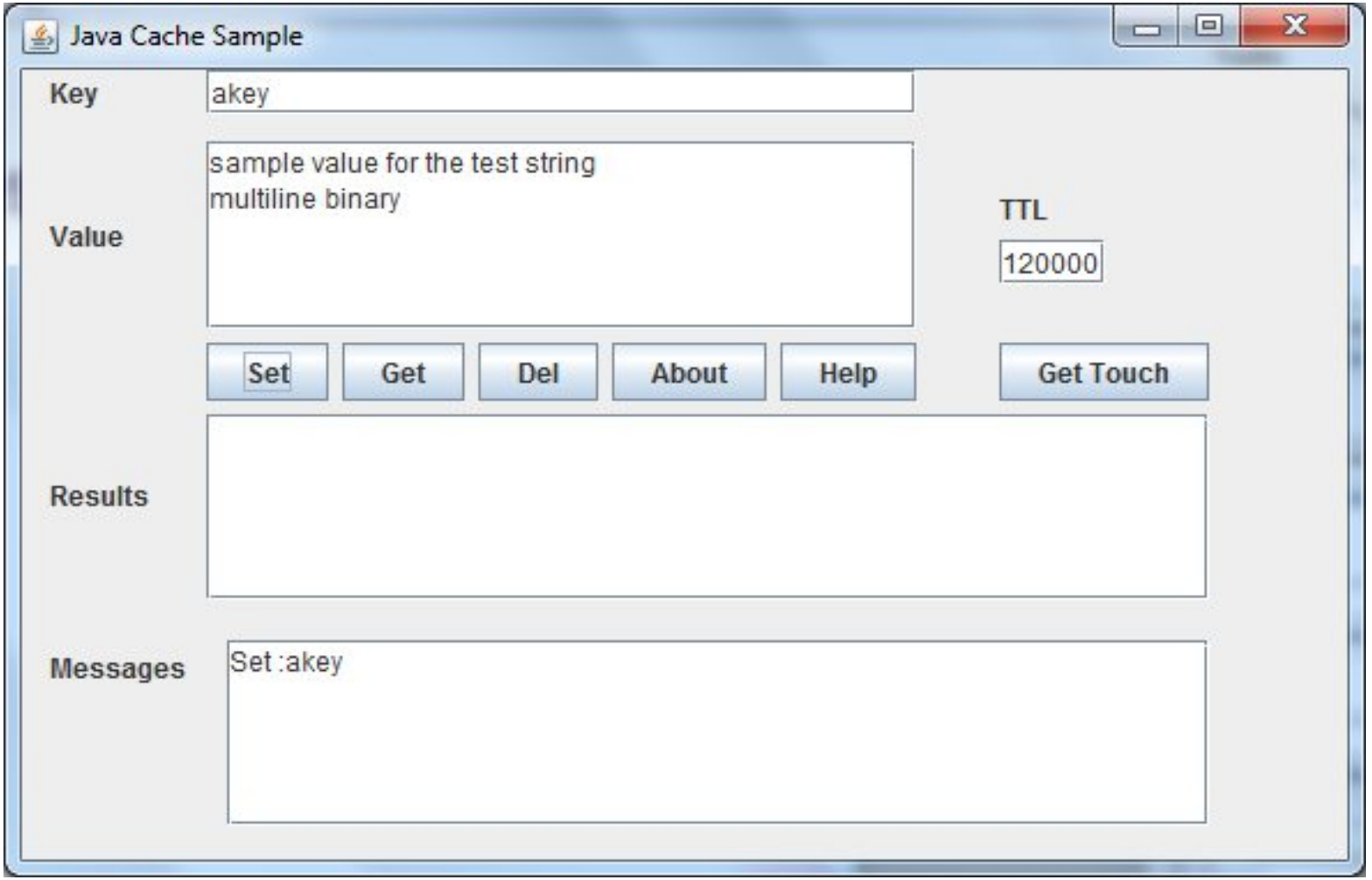
get key1

VALUE key1 0 4

ABCD

END

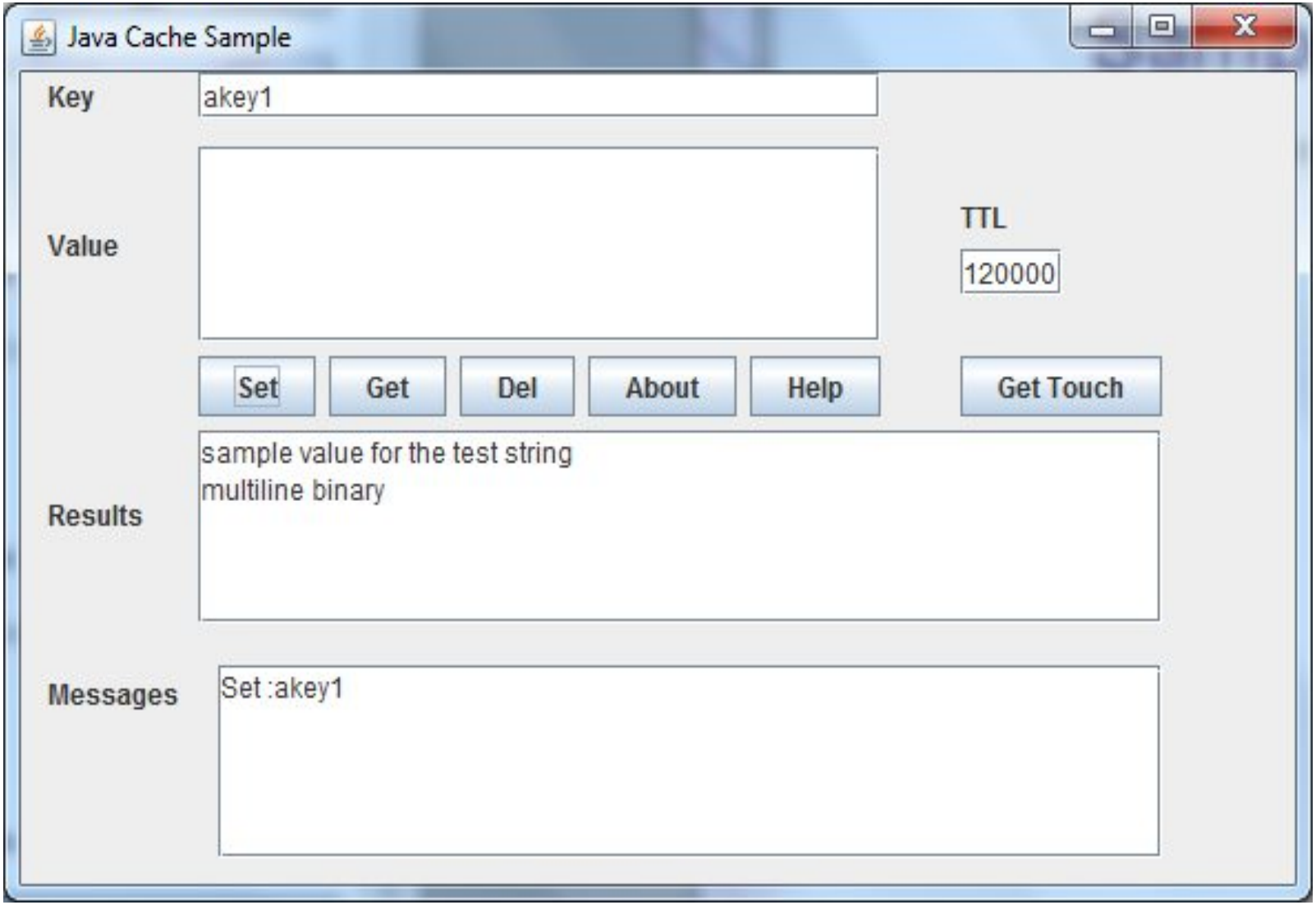
# Test App




Set value to cache

sel2in

# Sample App Read



# Code Setup

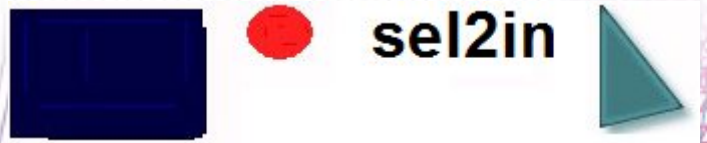
```
MemcachedClientBuilder b = new XMemcachedClientBuilder();
for (int i = 0; i < args.length; i++) {
    String a = args[i];
    if ("-b".equals(a)) {
        b.setCommandFactory(new BinaryCommandFactory());
    } else {

    }
}
cache = b.build();
for (int i = 0; i < args.length; i++) {
    String a = args[i];
    if ("-b".equals(a)) {

    } else {
        cache.addServer(args[i]);
    }
}
```

# Code Test

```
String someObject = "a val";
try{
    cache.set("someKey", 3600, someObject);
   Object myObject = cache.get("someKey");
   cache.delete("someKey");
   System.out.println(myObject);
} catch (Exception ex) {
```



# Exercises

- Read value from terminal after setting in UI
- Try to read a few minutes after expiry
- Touch and Get (extend expiry)
- Data base/ origin fail over

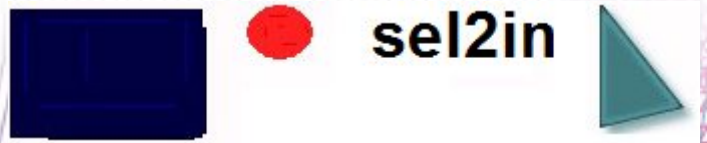
# Abuse

- Cache large amount of data, specific to a user
- Cache with large expiry
- Cache in the wrong place – page in app server instead of web server
- One time use data

# Measure

- Unit test case
- Selenium
- Manual clock

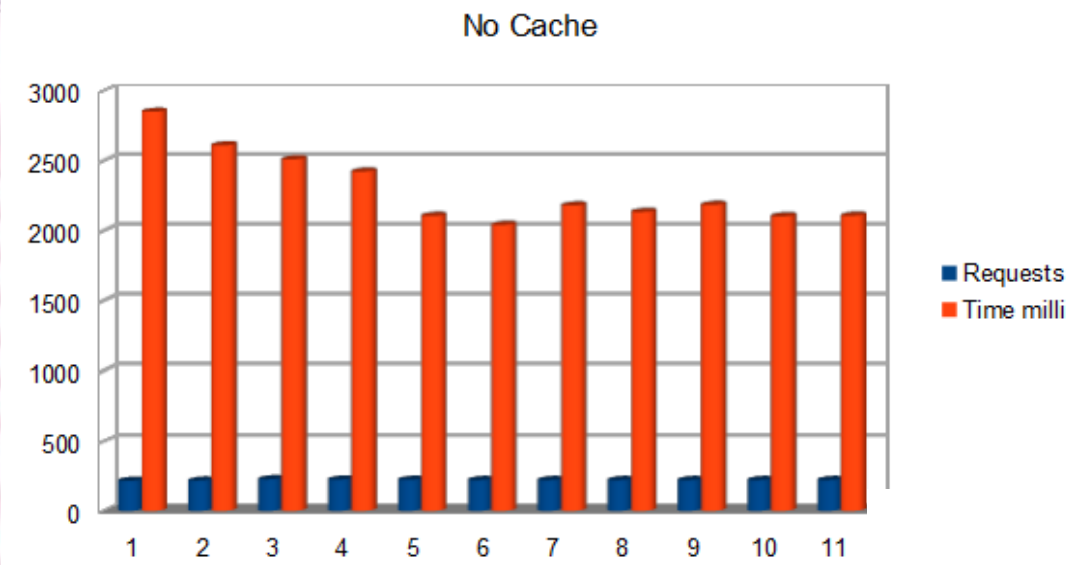
  - local & remote

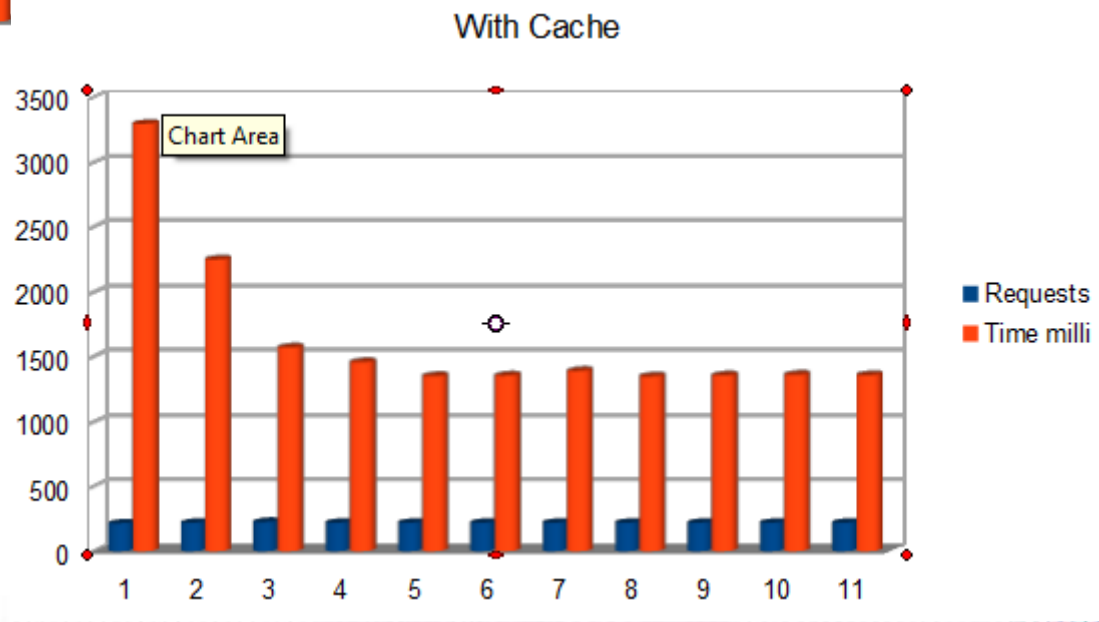


# Plan & Execute

- What to cache, start with easy artifacts
- Plan data base, file system, cache
- Prepare code – common points after key
- Is data in cache → else get from origin (db, calculation)
- Created new data → store in cache
- Make small changes
- Measure again

# Data of sample web app – automated testing

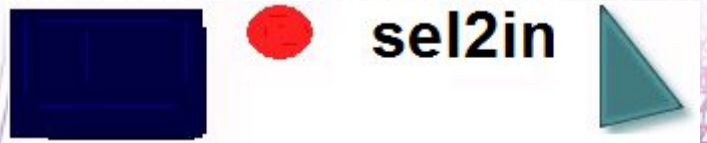



sel2in

# Sample Applications

- Http://sel2in.com/cache
  - Simple app available now
  - J2EE with data base in July first week
- Apache Mod_c : https://httpd.apache.org/docs/2.0/mod/mod_cache.html

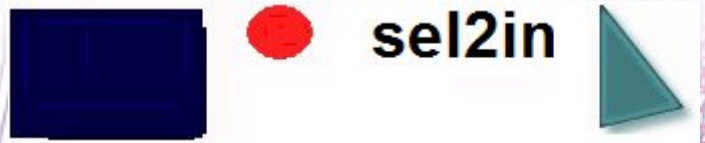


# Security

- Cache → As secure as your internal network is
- ACL → need web server that is ACL aware

# Acknowledgments


- http://code.google.com/p/quickcached/wiki/GetStar
  (http://code.google.com/p/quickcached)
- https://www.varnish-software.com/
- Eh Cache
- Http://Stackoverflow.com

# Questions and comments

Thick clients
Web start from disk
Server applications
Browser

Delete where key like “/client1/download/images”

“/client1/download/images/left.jpg”
“/client1/download/images/demo.jpg”