LEARN MEMCACHED

memory caching system

tutorialspoint

SIMPLYEASYLEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia

https://twitter.com/tutorialspoint

About the Tutorial

Memcached is an open source, high-performance, distributed memory object caching system.

This tutorial provides a basic understanding of all the relevant concepts of Memcached needed to create and deploy a highly scalable and performance-oriented system.

Audience

This tutorial is designed for software professionals who wish to learn and apply the concepts of Memcached in simple and easy steps.

Prerequisites

Before proceeding with this tutorial, you need to know the basics of data structures.

Copyright & Disclaimer

Table of Contents

Part 1

Basics

1. Memcached – Overview

Memcached is an open source, high-performance, distributed memory caching system intended to speed up dynamic web applications by reducing the database load. It is a key-value dictionary of strings, objects, etc., stored in the memory, resulting from database calls, API calls, or page rendering.

Memcached was developed by Brad Fitzpatrick for LiveJournal in 2003. However, it is now being used by Netlog, Facebook, Flickr, Wikipedia, Twitter, and YouTube among others

The key features of Memcached are as follows:

- It is open source.
- Memcached server is a big hash table.
- It significantly reduces the database load.
- It is perfectly efficient for websites with high database load.
- It is distributed under Berkeley Software Distribution (BSD) license.
- It is a client-server application over TCP or UDP.

Memcached is not:

- a persistent data store
- a database
- application-specific
- a large object cache
- fault-tolerant or highly available

2. Memcached – Environment

Installing Memcached on Ubuntu

To install Memcached on Ubuntu, go to terminal and type the following commands:

```
$sudo apt-get update
$sudo apt-get install memcached
```

Confirming Memcached Installation

To confirm if Memcached is installed or not, you need to run the command given below. This command shows that Memcached is running on the default port **11211**.

```
$ps aux | grep memcached
```

To run Memcached server on a different port, execute the command given below. This command starts the server on the TCP port 11111 and listens on the UDP port 11111 as a daemon process.

```
$memcached -p 11111 -U 11111 -d
```

You can run multiple instances of Memcached server through a single installation.

Memcached Java Environment Setup

To use Memcached in your Java program, you need to download **spymemcached-2.10.3.jar** and setup this jar into the classpath.

3. Memcached – Connection

To connect to a Memcached server, you need to use the telnet command on HOST and PORT names.

Syntax

The basic syntax of Memcached telnet command is as shown below:

```
$telnet HOST PORT
```

Here, **HOST** and **PORT** are machine IP and port number respectively, on which the Memcached server is executing.

Example

The following example shows how to connect to a Memcached server and execute a simple set and get command. Assume that the Memcached server is running on host 127.0.0.1 and port 11211.

```
$telnet 127.0.0.1 11211
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
// now store some data and get it from memcached server
set tutorialspoint 0 900 9
memcached
STORED
get tutorialspoint
VALUE tutorialspoint 0 9
memcached
END
```

Connection from Java Application

To connect the Memcached server from your java program, you need to add the Memcached jar into your classpath as shown in the previous chapter. Assume that the Memcached server is running on host 127.0.0.1 and port 11211.

Example

```
import net.spy.memcached.MemcachedClient;
public class MemcachedJava {
   public static void main(String[] args) {

        // Connecting to Memcached server on localhost
        MemcachedClient mcc = new MemcachedClient(new
        InetSocketAddress("127.0.0.1", 11211));
        System.out.println("Connection to server sucessfully");

        //not set data into memcached server
        System.out.println("set status:"+mcc.set("tutorialspoint", 900,
        "memcached").done);

        //Get value from cache
        System.out.println("Get from Cache:"+mcc.get("tutorialspoint"));
   }
}
```

Output

On compiling and executing the program, you get to see the following output:

```
Connection to server successfully
set status:true
Get from Cache:memcached
```

Part 2

Storage Commands

4. Memcached – Set Data

Memcached **set** command is used to set a new value to a new or existing key.

Syntax

The basic syntax of Memcached **set** command is as shown below:

```
set key flags exptime bytes [noreply]
value
```

The keywords in the syntax are as described below:

- **key:** It is the name of the key by which data is stored and retrieved from Memcached.
- **flags:** It is the 32-bit unsigned integer that the server stores with the data provided by the user, and returns along with the data when the item is retrieved.
- **exptime:** It is the expiration time in seconds. 0 means no delay. If exptime is more than 30 days, Memcached uses it as UNIX timestamp for expiration.
- **bytes:** It is the number of bytes in the data block that needs to be stored. This is the length of the data that needs to be stored in Memcached.
- **noreply (optional):** It is a parameter that informs the server not to send any reply.
- **value:** It is the data that needs to be stored. The data needs to be passed on the new line after executing the command with the above options.

Output

The output of the command is as shown below:

```
STORED
```

- **STORED** indicates success.
- **ERROR** indicates incorrect syntax or error while saving data.