

41 bits: Timestamp (毫秒级)

10 bits: 节点 ID (datacenter ID 5 bits + worker ID 5 bits)

12 bits: sequence number

年份 = 2^41 / (3600 \* 24 \* 365 \* 1000) = 69.7年

**JAVA:**

**package** io.github.id;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

/\*\*

 \* 第一位为未使用（实际上也可作为long的符号位），接下来的41位为毫秒级时间，然后5位datacenter标识位，</br>

 \* 5位机器ID（并不算标识符，实际是为线程标识），然后12位该毫秒内的当前毫秒内的计数，加起来刚好64位，为一个Long型。

 \* 0---0000000000 0000000000 0000000000 0000000000 0 --- 00000 ---00000 ---000000000000

 \* @author cailin

 \*

 \*/

**public** **class** IdWorker {

    **protected** **static** **final** Logger LOG = LoggerFactory.getLogger(IdWorker.**class**);

    //机器id

    **private** **long** workerId;

    //数据中心id

    **private** **long** datacenterId;

    **private** **long** sequence = 0L;

    **private** **long** twepoch = 1288834974657L;

    //机器标识位数

    **private** **long** workerIdBits = 5L;

    //数据中心标识位数

    **private** **long** datacenterIdBits = 5L;

    //机器ID最大值

    **private** **long** maxWorkerId = -1L ^ (-1L << workerIdBits);

    //数据中心ID最大值

    **private** **long** maxDatacenterId = -1L ^ (-1L << datacenterIdBits);

    //毫秒内自增位

    **private** **long** sequenceBits = 12L;

    //机器ID偏左移12位

    **private** **long** workerIdShift = sequenceBits;

    //数据中心ID左移17位

    **private** **long** datacenterIdShift = sequenceBits + workerIdBits;

    //时间毫秒左移22位

    **private** **long** timestampLeftShift = sequenceBits + workerIdBits + datacenterIdBits;

    **private** **long** sequenceMask = -1L ^ (-1L << sequenceBits);

    **private** **long** lastTimestamp = -1L;

    /\*\*

     \* @param workerId 机器id

     \* @param datacenterId 数据中心id

     \*/

    **public** IdWorker(**long** workerId, **long** datacenterId) {

        // sanity check for workerId

        **if** (workerId > maxWorkerId || workerId < 0) {

            **throw** **new** IllegalArgumentException(String.format("worker Id can't be greater than %d or less than 0", maxWorkerId));

        }

        **if** (datacenterId > maxDatacenterId || datacenterId < 0) {

            **throw** **new** IllegalArgumentException(String.format("datacenter Id can't be greater than %d or less than 0", maxDatacenterId));

        }

        **this**.workerId = workerId;

        **this**.datacenterId = datacenterId;

        LOG.info(String.format("worker starting. timestamp left shift %d, datacenter id bits %d, worker id bits %d, sequence bits %d, workerid %d", timestampLeftShift, datacenterIdBits, workerIdBits, sequenceBits, workerId));

    }

    **public** **synchronized** **long** nextId() {

        **long** timestamp = timeGen();

        **if** (timestamp < lastTimestamp) {

            LOG.error(String.format("clock is moving backwards.  Rejecting requests until %d.", lastTimestamp));

            **throw** **new** RuntimeException(String.format("Clock moved backwards.  Refusing to generate id for %d milliseconds", lastTimestamp - timestamp));

        }

        **if** (lastTimestamp == timestamp) {

            //当前毫秒内，则+1

            sequence = (sequence + 1) & sequenceMask;

            **if** (sequence == 0) {

            //当前毫秒内计数满了，则等待下一秒

                timestamp = tilNextMillis(lastTimestamp);

            }

        } **else** {

            sequence = 0L;

        }

        lastTimestamp = timestamp;

        //ID偏移组合生成最终的ID，并返回ID

        **return**

    }

    **protected** **long** tilNextMillis(**long** lastTimestamp) {

        **long** timestamp = timeGen();

        **while** (timestamp <= lastTimestamp) {

            timestamp = timeGen();

        }

        **return** timestamp;

    }

    **protected** **long** timeGen() {

        **return** System.currentTimeMillis();

    }

    **public** **static** **void** main(String[] args) {

        IdWorker  worker  = **new** IdWorker(1, 1);

        worker.nextId();

    }

}

PHP:

**class** Idwork {

    **const** debug = 1;

    **static** $workerId;

    **static** $twepoch = 1361775855078;

    **static** $sequence = 0;

    **const** workerIdBits = 4;

    **static** $maxWorkerId = 15;

    **const** sequenceBits = 10;

    **static** $workerIdShift = 10;

    **static** $timestampLeftShift = 14;

    **static** $sequenceMask = 1023;

    **private** **static** $lastTimestamp = - 1;

    **function** \_\_construct($workId) {

        **if** ($workId > self::$maxWorkerId || $workId < 0) {

            **throw** **new** Exception("worker Id can't be greater than 15 or less than 0");

        }

        self::$workerId = $workId;

        echo 'logdebug->\_\_construct()->self::$workerId:' . self::$workerId;

        echo '</br>';

    }

    **function** timeGen() {

        //获得当前时间戳

        $time = explode(' ', microtime());

        $time2 = substr($time[0], 2, 3);

        $timestramp = $time[1] . $time2;

        echo 'logdebug->timeGen()->$timestramp:' . $time[1] . $time2;

        echo '</br>';

        **return** $time[1] . $time2;

    }

    **function** tilNextMillis($lastTimestamp) {

        $timestamp = $this->timeGen();

        **while** ($timestamp <= $lastTimestamp) {

            $timestamp = $this->timeGen();

        }

        echo 'logdebug->tilNextMillis()->$timestamp:' . $timestamp;

        echo '</br>';

        **return** $timestamp;

    }

    **function** nextId() {

        $timestamp = $this->timeGen();

        echo 'logdebug->nextId()->self::$lastTimestamp1:' . self::$lastTimestamp;

        echo '</br>';

        **if** (self::$lastTimestamp == $timestamp) {

            self::$sequence = (self::$sequence + 1) & self::$sequenceMask;

            **if** (self::$sequence == 0) {

                echo "###########" . self::$sequenceMask;

                $timestamp = $this->tilNextMillis(self::$lastTimestamp);

                echo 'logdebug->nextId()->self::$lastTimestamp2:' . self::$lastTimestamp;

                echo '</br>';

            }

        } **else** {

            self::$sequence = 0;

            echo 'logdebug->nextId()->self::$sequence:' . self::$sequence;

            echo '</br>';

        }

        **if** ($timestamp < self::$lastTimestamp) {

            **throw** **new** Excwption("Clock moved backwards.  Refusing to generate id for " . (self::$lastTimestamp - $timestamp) . " milliseconds");

        }

        self::$lastTimestamp = $timestamp;

        echo 'logdebug->nextId()->self::$lastTimestamp3:' . self::$lastTimestamp;

        echo '</br>';

        echo 'logdebug->nextId()->(($timestamp - self::$twepoch << self::$timestampLeftShift )):' . ((sprintf('%.0f', $timestamp) - sprintf('%.0f', self::$twepoch)));

        echo '</br>';

        $nextId = ((sprintf('%.0f', $timestamp) - sprintf('%.0f', self::$twepoch))) | (self::$workerId << self::$workerIdShift) | self::$sequence;

        echo 'timestamp:' . $timestamp . '-----';

        echo 'twepoch:' . sprintf('%.0f', self::$twepoch) . '-----';

        echo 'timestampLeftShift =' . self::$timestampLeftShift . '-----';

        echo 'nextId:' . $nextId . '----';

        echo 'workId:' . self::$workerId . '-----';

        echo 'workerIdShift:' . self::$workerIdShift . '-----';

        **return** $nextId;

    }

}

$Idwork = **new** Idwork(1);

$a = $Idwork->nextId();

$Idwork = **new** Idwork(2);

$a = $Idwork->nextId();

?>