

Short Title Here

Longer more detailed and descriptive subtitle here

by Arthur Nayme

At a Glance

RSD#: 00000

Target Reader: Intermediate

Source Code: Yes

RS Version Required: 2011r4+

Platform(s) Supported: Mac OS X, Windows, Linux

About the Author: Arthur is a world-famous snuffendorf export and he knows everything about using Real Studio to track their migration patterns. He frequently writes for *Real Studio Developer* as well as *MacTech*, *Dr. Dobbs Journal*, and many other publications. He lives in the U.S. Northwest.

Lorem ipsum *dolar* sit amet ipsam et fugit vitae **nemo** voluptatem sed inventore consequuntur aut sed ab ab, sed veritatis voluptas sit dicta fugit aperiam doloremque quae sed vitae illo veritatis sunt inventore ab rem odit odit, qui architecto fugit sed eaque quae voluptatem ratione rem odit aperiam sit consequuntur quasi odit ipsam quasi quasi totam eaque eaque, voluptatem sed sed fugit sit ut consequuntur dicta quia voluptas eos voluptatem qui quae quae, dolores ratione odit qui ab aperiam sit veritatis vitae ipsam unde odit architecto architecto architecto, eos voluptatem odit vitae iste sit ratione vitae sit vitae natus error error quasi quasi, ratione.

Lorem ipsum for heading

Lorem ipsum dolor sit amet quasi nemo dolores iste beatae eos omnis vitae aspernatur ratione enim sunt voluptatem iste eos eos, sed sunt illo ab aut sed explicabo illo inventore doloremque voluptas consequuntur dicta eos enim sed aperiam aspernatur sed aut aut, aut laudantium dolores sed quia voluptas unde enim architecto illo voluptatem enim enim iste iste veritatis eaque eaque, totam sit sed qui inventore dicta sed sunt dolores qui omnis aperiam inventore rem rem, qui voluptatem aut sed magni vitae architecto veritatis rem voluptatem aut aut sit architecto

Lorem ipsum *dolar* sit amet ipsam et fugit vitae **nemo** voluptatem sed inventore consequuntur aut sed ab ab, sed veritatis voluptas sit dicta fugit aperiam doloremque quae sed vitae illo veritatis sunt inventore ab rem odit odit, qui architecto fugit sed eaque quae voluptatem ratione rem odit aperiam sit consequuntur quasi odit ipsam quasi quasi totam eaque eaque, voluptatem sed sed fugit sit ut consequuntur dicta quia voluptas eos voluptatem qui quae quae, dolores ratione odit qui ab aperiam sit veritatis vitae ipsam unde odit architecto architecto architecto, eos voluptatem odit vitae iste sit ratione vitae sit vitae natus error error quasi quasi, ratione.

Lorem ipsum for heading two

Lorem ipsum dolar sit amet et unde voluptatem accusantium accusantium, quae beatae explicabo quia sit natus rem quae accusantium eaque accusantium error quia accusantium sit voluptatem totam sed architecto architecto, laudantium aspernatur aut sit magni inventore totam iste iste sed laudantium explicabo unde aut rem sunt error error, magni architecto eos inventore fugit ut aspernatur enim ipsa rem vitae magni

quae perspiciatis quia quia, explicabo aspernatur illo sed sit ipsa illo veritatis aut ab veritatis fugit aut qui qui, quae voluptatem sit accusantium rem laudantium sed quae architecto iste fugit fugit fugit, sit totam fugit unde ut dicta unde sunt quia veritatis consequuntur illo quia voluptatem enim odit sunt.

Lorem ipsum dolor sit amet natus consequuntur rem aut ab voluptatem beatae iste aut accusantium accusantium odit architecto eaque voluptatem voluptatem, vitae explicabo explicabo ratione laudantium consequuntur explicabo et beatae dolores dolores perspiciatis quae ab aut ipsam fugit architecto architecto, quasi ipsa vitae error sit aut aut odit. Oditsed doloremque ratione ratione, enim explicabo aut ratione quia fugit ut beatae quia unde illo ipsa voluptas aspernatur ab voluptas ut consequuntur consequuntur, quia unde consequuntur laudantium fugit sit beatae sunt aspernatur iste sed beatae odit explicabo inventore inventore, aperiam vitae quae aspernatur voluptatem qui voluptas illo natus accusantium quae odit perspiciatis vitae.

Lorem ipsum dolor sit amet enim sit aut aperiam aperiam, architecto architecto ipsa aperiam illo aspernatur aut voluptas quia ratione accusantium ipsam aut ab beatae et ipsa laudantium *laudantium*, dicta rem consequuntur fugit aspernatur aperiam enim accusantium sunt ut doloremque eaque rem eos explicabo iste aut accusantium veritatis veritatis, voluptatem. Voluptatemaccusantium ipsa explicabo laudantium sit veritatis omnis illo qui voluptatem voluptatem, veritatis ipsa magni doloremque omnis eaque qui sed ratione enim omnis dicta omnis omnis, unde fugit quia vitae doloremque ut sed ipsam enim et ab sit sit illo dolores sit laudantium accusantium accusantium, aspernatur iste odit quia omnis omnis sunt sed odit sed omnis.

Lorem ipsum dolar sit amet inventore veritatis odit omnis aperiam aperiam, fugit accusantium sed unde inventore enim voluptas et beatae quia quia quae sit odit odit, et laudantium ipsam inventore ut voluptas odit qui laudantium. Laudantiumaut voluptatem consequuntur voluptatem error error, dicta doloremque sed aspernatur error sunt *perspiciatis* enim explicabo totam voluptatem ut natus eaque laudantium architecto aspernatur explicabo explicabo, ipsa aspernatur aspernatur sit totam voluptas veritatis aut omnis odit rem inventore perspiciatis quae odit quae aperiam aperiam, unde veritatis ratione magni unde eos unde vitae accusantium

Lorem ipsum for heading three

Lorem ipsum dolor sit amet natus consequuntur rem aut ab voluptatem beatae iste aut accusantium accusantium odit architecto eaque voluptatem voluptatem, vitae explicabo explicabo ratione laudantium consequuntur explicabo et beatae dolores dolores perspiciatis quae ab aut ipsam fugit architecto architecto, quasi ipsa vitae error sit aut aut odit. Oditsed doloremque ratione ratione, enim explicabo aut ratione quia fugit ut beatae quia unde illo ipsa voluptas aspernatur ab voluptas ut consequuntur consequuntur, quia unde consequuntur laudantium fugit sit beatae sunt aspernatur iste sed beatae odit explicabo inventore inventore, aperiam vitae quae aspernatur voluptatem qui voluptas illo natus accusantium quae odit perspiciatis vitae.

Lorem ipsum dolor sit amet enim sit aut aperiam aperiam, architecto architecto ipsa aperiam illo aspernatur aut voluptas quia ratione accusantium ipsam aut ab beatae et ipsa laudantium laudantium, dicta rem consequuntur fugit aspernatur aperiam enim accusantium sunt ut doloremque eaque rem eos explicabo iste aut accusantium veritatis veritatis, voluptatem. Voluptatemaccusantium ipsa explicabo laudantium sit veritatis omnis illo qui voluptatem voluptatem, veritatis ipsa magni doloremque omnis eaque qui sed ratione enim omnis dicta omnis omnis, unde fugit quia vitae doloremque ut sed ipsam enim et ab sit sit illo dolores sit laudantium accusantium accusantium, aspernatur iste odit quia omnis omnis sunt sed odit sed omnis.

- unde aperiam aut ratione sed accusantium vitae laudantium ut natus eos eos, aspernatur error aut
- sed error ipsa nemo illo omnis laudantium veritatis sed sed sed, ratione quasi unde veritatis quia beatae

- natus aspernatur laudantium aspernatur voluptas quia aspernatur nemo quae aperiam totam sit ab
- magni perspiciatis laudantium beatae beatae, totam eos aut aspernatur sit aut inventore ut illo voluptatem quia perspiciatis
- magni inventore error odit quia enim unde sunt sunt, sed sunt eos

Lorem ipsum for heading four

Lorem ipsum dolor sit amet ratione omnis magni omnis omnis, doloremque omnis voluptas qui aut aspernatur eaque doloremque dicta accusantium natus explicabo rem unde unde, illo omnis iste error ab sunt unde et quasi illo sunt odit vitae perspiciatis sed voluptatem odit fugit quia quia, unde eaque ab voluptatem ut nemo nemo quasi omnis inventore veritatis explicabo sed voluptatem dicta ipsam ratione fugit fugit, architecto sunt accusantium sit aspernatur sed qui explicabo sed sit natus rem sit odit sit doloremque doloremque, odit ratione qui aut explicabo quasi sunt sit odit sit aspernatur nemo aspernatur illo vitae voluptas inventore illo illo, accusantium perspiciatis quae aut ipsam odit eos consequuntur rem voluptatem eos natus.

This is an example of an inline table graphic:

Platform Line Ending Mac OS X chr(13) Unix chr(10) Windows chr(13) + chr(10)

Lorem ipsum dolar sit amet sed illo unde beatae beatae, et voluptatem sed laudantium error sit voluptatem sed ut accusantium nemo quae nemo dicta quae sunt quia dolores dolores, voluptatem sed laudantium sed aperiam voluptatem fugit unde ab ipsam iste sunt error aut eaque. Eaquearchitecto qui qui, totam dolores et ut nemo ipsa enim laudantium sunt quia doloremque natus ut perspiciatis perspiciatis, quia quasi rem sit omnis beatae odit aut aut ab rem fugit doloremque doloremque, quasi consequuntur aperiam sit dicta sit aperiam qui veritatis aperiam aperiam nemo sit sit eos aut beatae.

This next is an example of indented text:

Lorem ipsum dolor sit amet voluptatem ipsa voluptatem omnis accusantium fugit beatae omnis aut eos enim ipsa qui doloremque qui magni magni, totam explicabo omnis sit sit enim veritatis unde quia eos eos ipsam quia eos odit odit. Perspiciatis dolores eos eaque dicta voluptas error consequuntur architecto illo veritatis fugit eos ipsa ipsa. Accusantium dicta iste ut quasi architecto aut eaque voluptatem dolores voluptatem natus dolores doloremque quia error error, quia quae voluptatem aperiam odit iste odit aperiam quasi aspernatur iste architecto laudantium beatae beatae.

In a way, a hash table behaves like an array, with hash keys taking the place of indices. But while an array can accept multiple instances of the same data item, a hash table expects its items to be unique. Looking up an item in an array takes at worst $O(N)$ steps, with N being the array size. On a well-designed hash table, the same lookup takes almost $O(1)$ steps, regardless if the data items are sorted or not.

Next, hashLink defines four methods. The Constructor method is the default constructor. With it, we create an instance of the link with the new operator. The constructor takes one argument, the data item as a variant, which goes into the pVal property.

```

tDat = "your data item here"
tLnk = new hashLink(tDat)
  
```

The instance method add() also takes a data item as a variant. But this one uses the data item to add a new link to the hash chain. It starts by checking the pNxt property. If pNxt holds a nil, the method creates a new instance of hashLink using the data item provided.

Then it stores that instance into `pNxt`. But if `pNxt` does hold a `hashLink` instance, the method passes the data item to that instance's `add()` method.

In Summary

Lorem ipsum dolor sit amet ratione omnis magni omnis omnis, doloremque omnis voluptas qui aut aspernatur eaque doloremque dicta accusantium natus explicabo rem unde unde, illo omnis iste error ab sunt unde et quasi illo sunt odit vitae perspiciatis sed voluptatem odit fugit quia quia, unde eaque ab voluptatem ut nemo nemo quasi omnis inventore veritatis explicabo sed voluptatem dicta ipsam ratione fugit fugit, architecto sunt accusantium sit aspernatur sed qui explicabo sed sit natus rem sit odit sit doloremque doloremque, odit ratione qui aut explicabo quasi sunt sit odit sit aspernatur nemo aspernatur illo vitae voluptas inventore illo illo, accusantium perspiciatis quae aut ipsam odit eos consequuntur rem voluptatem eos natus.

That wraps up today's topic.

Recommended References

Wikipedia. "Hash table." Internet: (http://en.wikipedia.org/wiki/Hash_table). 2009 Jun 03 [2009 Jun 04].

Spark Notes, LLC. "What is a Hash Table?" Internet: (<http://www.sparknotes.com/cs/searching/hashtables/summary.html>). [2011 Oct 16].

John Morris. (1998). "Hash Tables." Data Structures and Algorithms. [HTML]. Available: (http://www.cs.auckland.ac.nz/~jmor159/PLDS210/hash_tables.html). [2011 Oct 16].

Sidebar: Lorem ipsum for First Sidebar

The *factorial* is an integer function often used for solving statistical problems. Basically, it is a product series from 1 to N, N being a positive integer. If N is a zero, the factorial value is a 1. If it is any other integer, the factorial is computed as follows:

$$N! = 1 * 2 * 3 * \dots * (N - 1) * N$$

Thus, a 2! factorial computes to a value of 2. A 4! computes to 24.

$$4! = 1 * 2 * 3 * 4 = 24$$

And a 5! factorial gives a value of 120.

Sidebar: Lorem ipsum for Second Sidebar

Counting the number of unique combinations is another function used in statistical problems. A *combination* is defined as sets of *unique* data items, arranged in no particular order. This example, for instance, shows four

combinations.

ABCD BCD ABD CBA

But this one shows only tree.

BCD CBA BDC ABC

In the above example, set 1 and 2 happened to have the same items, even though the items are arranged differently.

Below is how we compute the number of unique combinations. Variable M is the total number of data items. Variable n is the number of items per combination. Note the use of the factorial function as part of the computation.



Graphics:

[Figure 1: A hash table.](#)

[Figure 2: Processing a data item.](#)

[Figure 3: Simulating a collision.](#)

[Figure 4: Chances for a collision.](#)

[Figure 5: The load factor.](#)

Code Listing 1: Implementing a linear probing scheme.

```
Function hashExec(aMsg as string) as integer
// HASH ALGORITHM:ONE-AT-A-TIME
//

dim tLen, tPos, tChr as integer
dim tHsh as integer
dim tChk as boolean

// initialize the following locals

tLen = lenB(aMsg)
tHsh = 0

// process the input string
```

```

for tPos = 1 to tLen step 1
    tChr = ascB(aMsg.midB(tPos, 1))
    tHsh = tHsh + tChr
    tHsh = tHsh + bitwise.shiftLeft(tHsh, 10)
    tHsh = bitwise.bitXor(tHsh, bitwise.shiftRight(tHsh, 6))
next 'tPos

// final mixing

tHsh = tHsh + bitwise.shiftLeft(tHsh, 3)
tHsh = bitwise.bitXor(tHsh, bitwise.shiftRight(tHsh, 11))
tHsh = tHsh + bitwise.shiftLeft(tHsh, 15)

// validate the hash

do until (hashCheck(tHsh))
    tHsh = hashProbe(tHsh)
loop

// return the hash value

return (tHsh)
End Function

// Correct the colliding hash

Function hashProbe(aHsh as integer) as integer
    // PROBE ALGORITHM:LINEAR
    //

    dim tHsh as integer

    // define the following internal constants

    const HASH_LIMIT = &h7fffffff
    const HASH_DELTA = 1

    // calculate the next available hash

    tHsh = (aHsh + HASH_DELTA) mod HASH_LIMIT

    // return the corrected hash

    return (tHsh)
End Function

```

Code Listing 2: Implementing a quadratic probing scheme.

```

Function hashExec(aMsg as string) as integer
    // HASH ALGORITHM:BERNSTEIN
    //

    dim tLen, tPos, tChr as integer

```

```

dim tHsh as integer

// initialize the following locals

tLen = lenB(aMsg)
tHsh = tLen

// process the input string

for tPos = 1 to tLen step 1
    tChr = ascB(aMsg.midB(tPos, 1))
    tHsh = 33 * tHsh + tChr
next 'tPos

// validate the hash

do until (hashCheck(tHsh))
    tHsh = hashProbe(tHsh)
loop

// return the hash value

return (tHsh)
End Function

// Correct the colliding hash
Function hashProbe(aHsh as integer) as integer
    // PROBE ALGORITHM:QUADRATIC
    //

    dim tHsh as integer

    // define the following private constants

    const HASH_LIMIT = &h7fffffff
    const kA = 1
    const kB = 2
    const kC = 3

    // calculate the next available hash

    tHsh = aHsh * aHsh * kA
    tHsh = tHsh + aHsh * kB
    tHsh = tHsh + kC
    tHsh = tHsh mod HASH_LIMIT

    // return the corrected hash

    return (tHsh)
End Function

```