

Re: Base64

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2004-04/0094.html

From: Lew Pitcher (*Lew.Pitcher_at_td.com*)

Date: 04/01/04

Date: Thu, 01 Apr 2004 08:10:32 -0500

-----BEGIN PGP SIGNED MESSAGE-----

Hash: SHA1

John wrote:

| Hi all,

| I've been going through google and yahoo looking for a certain base64
| decoder in C without success. What I'm after is something that you can
| pass a base64 encoded string into and get back a decoded String.

|

| Any help is very much appreciated.

| Thanks

| Philip.

Here's one that I put together as a testbed for some mainframe-to-unix tools I
was working on. I used this C code as a model for a COBOL program that
manipulated base64 encodings.

```
/*
```

```
** MIME Base64 coding examples
```

```
**
```

```
** encode() encodes an arbitrary data block into MIME Base64 format string
```

```
** decode() decodes a MIME Base64 format string into raw data
```

```
**
```

```
** Global table base64[] carries the MIME Base64 conversion characters
```

```
*/
```

```
/* Global data used by both binary-to-base64 and base64-to-binary conversions */
```

```
static char base64[] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
```

```
    "abcdefghijklmnopqrstuvwxyz"
```

```
    "0123456789"
```

```
    "+/";
```

```
/*
```

```
** ENCODE RAW into BASE64
```

```
*/
```

```
/* Encode source from raw data into Base64 encoded string */
```

```

int encode(unsigned s_len, char *src, unsigned d_len, char *dst)
{
    unsigned triad;

    for (triad = 0; triad < s_len; triad += 3)
    {
        unsigned long int sr;
        unsigned byte;

        for (byte = 0; (byte<3)&&(triad+byte<s_len); ++byte)
        {
            sr <<= 8;
            sr |= (*(src+triad+byte) & 0xff);
        }

        sr <<= (6-((8*byte)%6))%6; /* leftshift to 6bit align */

        if (d_len < 4) return 1; /* error - dest too short */

        *(dst+0) = *(dst+1) = *(dst+2) = *(dst+3) = '=';
        switch(byte)
        {
            case 3:
                *(dst+3) = base64[sr&0x3f];
                sr >>= 6;
            case 2:
                *(dst+2) = base64[sr&0x3f];
                sr >>= 6;
            case 1:
                *(dst+1) = base64[sr&0x3f];
                sr >>= 6;
                *(dst+0) = base64[sr&0x3f];
        }
        dst += 4; d_len -= 4;
    }

    return 0;
}

/*
** DECODE BASE64 into RAW
*/

/* determine which sextet value a Base64 character represents */
int tlu(int byte)
{
    int index;

    for (index = 0; index < 64; ++index)
        if (base64[index] == byte)
            break;
}

```

```

    if (index > 63) index = -1;
    return index;
}

/* Decode source from Base64 encoded string into raw data */
int decode(unsigned s_len, char *src, unsigned d_len, char *dst)
{
    unsigned six, dix;

    dix = 0;

    for (six = 0; six < s_len; six += 4)
    {
        unsigned long sr;
        unsigned ix;

        sr = 0;
        for (ix = 0; ix < 4; ++ix)
        {
            int sextet;

            if (six+ix >= s_len)
                return 1;
            if ((sextet = tlu(*(src+six+ix))) < 0)
                break;
            sr <<= 6;
            sr |= (sextet & 0x3f);
        }

        switch (ix)
        {
            case 0: /* end of data, no padding */
                return 0;

            case 1: /* can't happen */
                return 2;

            case 2: /* 1 result byte */
                sr >>= 4;
                if (dix > d_len) return 3;
                *(dst+dix) = (sr & 0xff);
                ++dix;
                break;

            case 3: /* 2 result bytes */
                sr >>= 2;
                if (dix+1 > d_len) return 3;
                *(dst+dix+1) = (sr & 0xff);
                sr >>= 8;
                *(dst+dix) = (sr & 0xff);
                dix += 2;
        }
    }
}

```

```
        break;

    case 4: /* 3 result bytes */
        if (dix+2 > d_len) return 3;
        *(dst+dix+2) = (sr & 0xff);
        sr >>= 8;
        *(dst+dix+1) = (sr & 0xff);
        sr >>= 8;
        *(dst+dix) = (sr & 0xff);
        dix += 3;
        break;
    }
}
return 0;
}
```

Lew Pitcher
IT Consultant, Enterprise Application Architecture,
Enterprise Technology Solutions, TD Bank Financial Group

(Opinions expressed are my own, not my employers')

-----BEGIN PGP SIGNATURE-----

Version: GnuPG v1.2.4 (MingW32)

iD8DBQFAbBTFagVFX4UWr64RAI2AAKCxunT3bzDQ16w1sOWmh7Krs+WEpwCgsdL7
wtz0zplSxc9B4fvpS/8b/Dc=
=Hbsy

-----END PGP SIGNATURE-----