O'REILLY®

8TH ANNUAL OSCON™ OPEN SOURCE CONVENTION

Porting Mac::Carbon to Intel

Chris Nandor
OSTG / Slashdot

JULY 24-28, 2006

About Chris

- Coder for Slashdot
  - http://slashdot.org/
- Guy who runs use Perl;
  - http://use.perl.org/
- Maintainer of MacPerl
- Author, maintainer of various Mac:: modules

Carbon: Transition to Mac OS X

- Mac OS API on Mac OS X
  - Files
  - Gestalt
  - Speech
  - Sound
  - Apple events
- Modules originally available for MacPerl
- Ported separately to Mac OS X

Intel: Transition to x86

- Rosetta
  - Run PPC code on x86
- Universal Binaries
  - Run same binaries on x86 or PPC
- Or ...
  - Just compile specifically for x86

Mac::Carbon and Tiger

- Apple including more OSS in distribution
- Mac::Carbon included with Mac OS X 10.4
- Also included:
  - Mac::Apps::Launch, Mac::AppleEvents::Simple, Mac::Glue, Mac::OSA::Simple, Time::Epoch
  - Mac::Errors
  - wxPerl
  - `% open /System/Library/Perl/Extras/`

Timeline of Events

- January 2006
  - Intel Macs released
  - Found out for sure there was a problem with Mac::Carbon
  - Do some debugging, get one big fix done
- February-April 2006
  - Life is more important
- May 2006
  - Looking for a machine to do porting on
  - Apple loans Intel Mac mini
- June-July 2006
  - Port!

Porting Issues

- Almost all problems related to big endian vs. little endian
  - Endianness sux
  - Almost all endian problems related to FourCharCodes
- Other issues:
  - Directly parsing bytes for binary data structures
    - FSSpec parsing
    - Quads
  - Slightly different behaviors
    - `GetAliasInfo()` deprecated
  - Still-unresolved build problem with one universal binary build

FourCharCodes

- Mac OS uses FourCharCodes for unique IDs
  - Creator types: `R*ch`, `MACS`
  - File Types: `TEXT`, `APPL`
  - Constants: `kDesktopFolderType` (`desk`)
  - Apple event IDs: `misc`, `actv`

- Not really characters: Unsigned long integer
  - `MACS == 1296122707 # PPC`
  - `SCAM == 1296122707 # Intel`
  - `MACS == 1396916557 # Intel`

XS Example

- Basic example:

```
long
Gestalt(sel)
     OSType  sel

CODE:
if (errno = Gestalt(sel, &RETVAL)) {
     XSRETURN_UNDEF;
}
OUTPUT:
RETVAL
```

XS Example

- When converted to C (simplified):

```
OSType  sel;
long    RETVAL;

memcpy(&sel, SvPV_nolen(ST(0)), sizeof(OSType));
sel = ntohl(sel);
if (errno = Gestalt(sel, &RETVAL)) {
     XSRETURN_UNDEF;
}
```

The Magic of typemap

- Handles conversion of arguments in and out of XS

```
OSType                  T_OSTYPE
INPUT
T_OSTYPE
        memcpy(&$var, SvPV_nolen($arg), sizeof($ntype));
        $var = ntohl($var);
OUTPUT
T_OSTYPE
        sv_setpvn($arg, (char *) &$var, 4);
```

The Magic of typemap

- Handles conversion of arguments in and out of XS

```
OSType                  T_OSTYPE
INPUT
T_OSTYPE
        memcpy(&$var, SvPV_nolen($arg), sizeof($ntype));
        $var = ntohl($var);
OUTPUT
T_OSTYPE
        { OSType hos = htonl($var);
          sv_setpvn($arg, (char *) &hos, 4);
        }
```

typemap Change Summary

- This one change took care of 80+ percent of problems
  - ~160 calls

- Some calls that accept/return OSType don't use typemap
  - MacPerl::GetFileInfo() pushed OSType values directly onto the return stack
  - Others accept a plain SV pointer, and allow data type to be figured out in another way (usually passed as separate argument)

Other Conversions

```
static void ConvertFourCharCode(OSType type, char * ptr)
{
     if (type == typeEnumerated
      || type == typeType
      || type == typeProperty
      || type == typeKeyword
      || type == typeApplSignature
      || type == typeAbsoluteOrdinal) {
          *(long *)ptr = ntohl(*(long *)ptr);
     }
}
```

Other Conversions

```
AEDesc
AECreateDesc(type, data)
        OSType    type
        SV *      data
        CODE:
        {
                char *    ptr;
                STRLEN    size;

                ptr = SvPV(data, size);
                ConvertFourCharCode(type, ptr);
                errno = AECreateDesc(type, ptr, size, &RETVAL);
                ConvertFourCharCode(type, ptr);
        }
```

Perl-space, Too

- Mac::AETE::Parser

```
sub _get_ID() {
    my($self) = @_;
    return pack 'N', unpack 'L', $self->_scan(4);
}
```

Perl-space, Too

- Mac::Glue Test Code

```
$ical->make(
        new => 'event',
        at  => location(end => $cal->obj('events')),
        with_properties => {
            summary        => 'Test Event',
            start_date     => time()+86400*7,
            end_date       => time()+86400*7+3600
        }
    );
```

Perl-space, Too

```
16:14:14 <pudge_> this should create a new event one week
                  from today
16:14:37 <pudge_> i suspect it won't, but i have a patch that
                  should work :)
16:15:10 <baud>   it's taking a while...
16:15:55 <pudge_> still going?
16:18:39 *        baud has quit (Ping timeout: 182 seconds)
16:21:51 <baud>   that was fun.
16:21:59 <baud>   lets do it AGAIN
16:29:13 <pudge_> what happened?
16:29:20 <baud>   so that used up all my ram and pretty much
                  killed my system
16:29:38 <pudge_> do you want to try it with the patch?  :)
16:30:02 <baud>   hmm. he said nervously :)
```

Perl-space, Too

- Mac::Glue Patch

```
my $ldt = $^O eq 'MacOS'
     ? $_[0]
     : perl2epoch($_[0], 'macos');
```

```
require Config;
if ($Config::Config{byteorder} eq '1234') {
     return pack 'LL', $ldt, 0;
} else {
     return pack 'LL', 0, $ldt;
}
```

Don't Do That

- Don't change the type of an existing AEDesc; coerce it to a new AEDesc instead, with `AECoerceDesc()`.

```
my $desc = AECreateDesc(typeChar, 'abcd');
my $new = AECoerceDesc($desc, typeKeyword); # OK
$desc->type(typeKeyword);                   # NOK
```

Don't Do That

- Don't pass FourCharCodes as arguments to `AEBuild*()`; there's no easy way for the called function to know what type the argument is going to be passed as, and to fix the data before it is passed. Define FourCharCodes as literals in `AEBuild*()` formats instead.

```
AEBuild(q{'----':keyw(@)}, typeProperty);  # NOK
AEBuild(q{'----':keyw(prop)});             # OK
```

Don't Do That

- Don't parse binary data structures directly, if there's an API to be used instead.

```
($type, $id, $version, $nlen, $name) =
  unpack("x4 a4 l l C a63", $$desc);
$type = $desc->voice->creator;
```

Odds and Ends

- Mac::AppleEvents had a lot of places to fix FourCharCodes, and no test suite
  - It now has a very good test suite
  - Found several other bugs

- Let me know if you find bugs or problems, or want to help with more tests

- Subscribe to the mailing list

Resources

- http://dev.macperl.org/mac-carbon-intel/
- macosx@perl.org
- http://projects.pudge.net/
- projects@pudge.net
- http://use.perl.org/~pudge/journal/