1 Measure sizeof() of Perl’s C Structures

1.1 Description

This document describes the *sizeof* various structures, as determined by *util/sizeof.pl*. These measurements are mainly for research purposes into making Perl things smaller, or rather, how to use less Perl things.

1.2 Perl Structures

Structures diagrams are courtesy gdb (print pretty) and a bit of hand crafting.

- **CV - 229 minimum, 254 minimum w/ symbol table entry**

```
cv = {
   sv_any = {        // XPVCV *
       xpv_pv = 0x0, // char *
       xpv_cur = 0,  // STRLEN
       xpv_len = 0,  // STRLEN
       xof_off = 0 , // IV
       xnv_nv = 0,   // NV
       xmg_magic = 0x0, // MAGIC *
       xmg_stash = 0x0, // HV *
       xcv_stash = 0x0, // HV *
       xcv_start = 0x0, // OP *
       xcv_root = 0x0,  // OP *
       xcv_xsub = 0x0,  // void (*)(register PerlInterpreter *, CV *)
       xcv_xsubany = {  // ANY
           any_ptr = 0x0,
           any_i32 = 0,
           any_iv = 0,
           any_long = 0,
           any_dptr = 0,
           any_dxptr = 0
       },
       xcv_gv = { // GV *
           sv_any = { // void *
               xpv_pv = 0x0, // char *
               xpv_cur = 0,  // STRLEN
               xpv_len = 0,  // STRLEN
               xiv_iv = 0,   // IV
               xnv_nv = 0,   // NV
               xmg_magic =  { // MAGIC *
                   mg_moremagic = 0x0, // MAGIC *
                   mg_virtual = 0x0,   // MGVTBL *
                   mg_private = 0,     // U16
                   mg_type = 0,        // char
                   mg_flags = 0,       // U8
                   mg_obj = 0x0,       // SV *
                   mg_ptr = 0x0,       // char *
                   mg_len = 0,         // I32
               },
               xmg_stash = 0x0, // HV *
               xgv_gp = { // GP *
```

```
                    gp_sv = { // SV *
                        sv_any = 0x0,  // void *
                        sv_refcnt = 0, // U32
                        sv_flags = 0   // U32
                    },
                    gp_refcnt = 0, // U32
                    gp_io = 0x0,   // struct io *
                    gp_form = 0x0, // CV *
                    gp_av = 0x0,   // AV *
                    gp_hv = 0x0,   // HV *
                    gp_egv = 0x0,  // GV *
                    gp_cv = 0x0,   // CV *
                    gp_cvgen = 0,  // U32
                    gp_flags = 0,  // U32
                    gp_line = 0,   // line_t
                    gp_file = 0x0, // char *
                },
                xgv_name = 0x0,  // char *
                xgv_namelen = 0, // STRLEN
                xgv_stash = 0x0, // void *
                xgv_flags = 0,   // U8
            },
            sv_refcnt = 0, // U32
            sv_flags = 0,  // U32
        },
        xcv_file = 0x0, // char *
        xcv_depth = 0, // long
        xcv_padlist = 0x0, // AV *
        xcv_outside = 0x0, // CV *
        xcv_flags = 0, // cv_flags_t
    }
    sv_refcnt = 0, // U32
    sv_flags = 0,  // U32
};
```

In addition to the minimum bytes:

- **name of the subroutine: GvNAMELEN(CvGV(cv))+1**
- **symbol table entry: HvENTRY (25 + GvNAMELEN(CvGV(cv))+1)**
- **minimum sizeof(AV) * 3: xcv_padlist if !CvXSUB(cv)**
- **CvROOT(cv) optree**

- **HV - 60 minmum**

```
hv = {
    sv_any = { // SV *
        xhv_array = 0x0,  // char *
        xhv_fill = 0,     // STRLEN
        xhv_max = 0,      // STRLEN
        xhv_keys = 0,     // IV
        xnv_nv = 0,       // NV
        xmg_magic = 0x0,  // MAGIC *
        xmg_stash = 0x0,  // HV *
        xhv_riter = 0,    // I32
        xhv_eiter = 0x0,  // HE *
        xhv_pmroot = 0x0, // PMOP *
```

```
        xhv_name = 0x0    // char *
    },
    sv_refcnt = 0, // U32
    sv_flags = 0,  // U32
};
```

Each entry adds `sizeof(HvENTRY)`, minimum of 7 (initial `xhv_max`). Note that keys of the same value share `sizeof(HEK)`, across all hashes.

- **HvENTRY - 25 + HeKLEN+1**

```
sizeof(HE *) + sizeof(HE) + sizeof(HEK)
```

- **HE - 12**

```
he = {
   hent_next = 0x0, // HE *
   hent_hek = 0x0,  // HEK *
   hent_val = 0x0   // SV *
};
```

- **HEK - 9 + hek_len**

```
hek = {
   hek_hash = 0, // U32
   hek_len = 0,  // I32
   hek_key = 0,  // char
};
```

- **AV - 53**

```
av = {
   sv_any =  { // SV *
       xav_array = 0x0,  // char *
       xav_fill = 0,     // size_t
       xav_max = 0,      // size_t
       xof_off = 0,      // IV
       xnv_nv = 0,       // NV
       xmg_magic = 0x0,  // MAGIC *
       xmg_stash = 0x0,  // HV *
       xav_alloc = 0x0,  // SV **
       xav_arylen = 0x0, // SV *
       xav_flags = 0,    // U8
   },
   sv_refcnt = 0, // U32
   sv_flags = 0   // U32
};
```

In addition to the minimum bytes:

  - **AvFILL(av) * sizeof(SV *)**

1.3 SEE ALSO

perlguts(3), B::Size(3),

http://gisle.aas.no/perl/illguts/

1.4 Maintainers

Maintainer is the person(s) you should contact with updates, corrections and patches.

- Doug MacEachern <dougm (at) covalent.net>

1.5 Authors

- Doug MacEachern <dougm (at) covalent.net>

Table of Contents: