# pygobject & GIL

or

The death by thousand cuts

by Fludkov Mikhail
misha@pexip.com
fludkov.me@gmail.com

pexip

Python app

stats stats … stats

rtpsession100

rtpsession57

…

…

rtpsession0

rtpsession2

rtpsession1

pexip

# What's inside stats. GstStruct:

```
ssrc                = (uint)    2765911805,
internal            = (boolean) true,
validated           = (boolean) true,
received-bye        = (boolean) false,
is-csrc             = (boolean) false,
is-sender           = (boolean) true,
clock-rate          = (int)     90000,
bitrate             = (guint64) 0,
seqnum-base         = (int)     13034,
octets-sent         = (guint64) 520,
packets-sent        = (guint64) 49,
recv-pli-count      = (uint)    0,
recv-fir-count      = (uint)    0,
have-sr             = (boolean) true,
sr-ntptime          = (guint64) 15826585362390969741,
sr-rtptime          = (uint)    2191509451,
sr-octet-count      = (uint)    520,
sr-packet-count     = (uint)    49,
have-rb             = (boolean) false,
red-sent            = (uint)    0,
inner-fec-protected = (uint)    0,
outer-fec-protected = (uint)    0
…
```

pexip

```
def rtpstats_convert(value_array):
    entries = []
    for gststats in value_array:
        data = {}
        for idx in range(0, gststats.n_fields()):
            field_name = gststats.nth_field_name(idx)
            value = gststats.get_value(field_name)
            data[field_name] = value
        entry = { 'name' : gststats.get_name(),
                  'data' : data,
                }
        entries.append(entry)
    return entries
```

pexip

~ x100

```c
PyGTypeMarshal *
pyg_type_lookup(GType type)
{
    GType   ptype = type;
    PyGTypeMarshal *tm = NULL;

    /* recursively lookup types */
    while (ptype) {
        pygi_type_import_by_g_type (ptype);
        if ((tm = g_type_get_qdata(ptype, pyg_type_marshal_key)) != NULL) {
            break;
        }
        ptype = g_type_parent(ptype);
    }
    return tm;
}
```

≈ x2500

# Thank you

Pexip PyGObject fork:
https://github.com/pexip/pygobject

pexip

# There is not time for this slide

Python callback optimisation

Python:
0.0045 ± 0.0063 sec and 0.65 at worst
Python cmodule:
0.000058 ± 0.000032 sec and 0.0083 at worst

PyGObject marshalling code optimisation

Before:
0.65 ± 0.47 sec and 9.7 sec at worst!!!
After:
0.00026 ± 0.00033 secs