

# Glaze

## A Shinier Way to Wrap OpenGL

Carl Worth

[carl.d.worth@intel.com](mailto:carl.d.worth@intel.com)

X Developers' Conference

Sep. 23, 2013

# Why Wrap OpenGL?

- Bug capture and replication
  - Trace, replay
- Application/driver instrumentation
  - Performance monitoring

# Does anyone do this?

- Apitrace
- Bugle
- Fips
- glxoffload
- Primus
- VirtualGL
- Etc.

# My OpenGL-wrapping Background

- Apitrace
- Fips
- Glaze

# Apitrace

- Started by José Fonseca (VMWare) in 2008
- I began hacking on it in 2011
- Invaluable for bug capture/replication
- Multiple wrapping interfaces
  - LD\_PRELOAD
  - Alternate libGL.so via LD\_LIBRARY\_PATH

<http://apitrace.github.io/>

# Fips

- Live, application performance measurement
- Initially only LD\_PRELOAD
- Lots of thrashing to get applications to work

```
git clone git://git.cworth.org/git/fips
```

# Alexander Monakov

- Inspired by much flailing with fips
- Documented everything he knows about OpenGL wrapping:

<https://github.com/amonakov/on-wrapping/blob/master/interposers-discussion.asciidoc>

- In turn inspired Glaze and this talk

# Wrapping is easy, right?

- Tour: FPS counter
  - (Hint: It ends up being not so easy)
- This portion of the talk was a live demo. To emulate it at home, checkout the following source code:

```
git clone git://git.cworth.org/git/glfps
```

and go through each point in the code history, examining the source, running “make” and “glfps-test” for each revision.



# Glaze: Making it easy again

- Ideally has the benefits of a simple LD\_PRELOAD
  - Works with many application styles
  - Wrapper author can ignore GetProcAddress and dlsym
  - Wrappers can nest

```
git clone git://git.cworth.org/git/glaze
```

# What's in Glaze?

- All OpenGL functions
  - Automatic from Khronos XML files
- Convenience library
  - GetProcAddress
  - glaze\_lookup()
  - GLAZE\_DEFER
  - glaze\_execute()

# An introduction to ifunc

```
void * foo() __attribute__((ifunc("foo_resolver")));
```

```
static void *
```

```
foo_resolver (void) {
```

```
    if (condition)
```

```
        return foo_version_1;
```

```
    else
```

```
        Return foo_version_2;
```

```
}
```

# How to use Glaze

`LD_LIBRARY_PATH=/path/to/glaze/libGL.so`

`GLAZE_LIBGL=/path/to/real/libGL.so`

`GLAZE_WRAPPER=wrapperlib.so`

# Glaze Convenience

```
$ glaze --wrapper=wrapperlib.so
```

# Glaze users can nest

```
GLAZE_WRAPPER=rock.so:roll.so
```

```
$ glaze --wrapper=rock.so \  
  glaze --wrapper=roll.so program
```

```
$ glrock glroll program
```