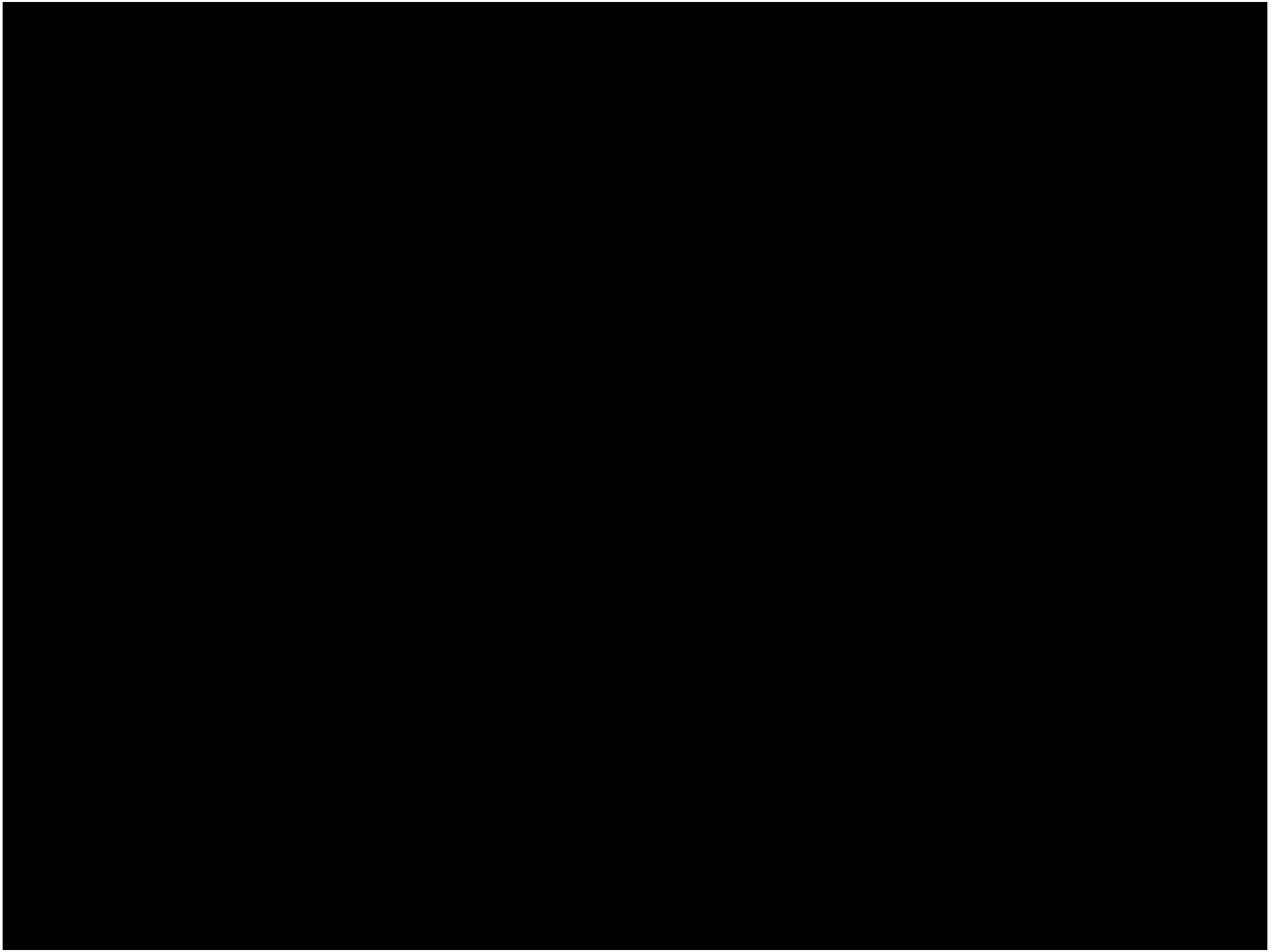
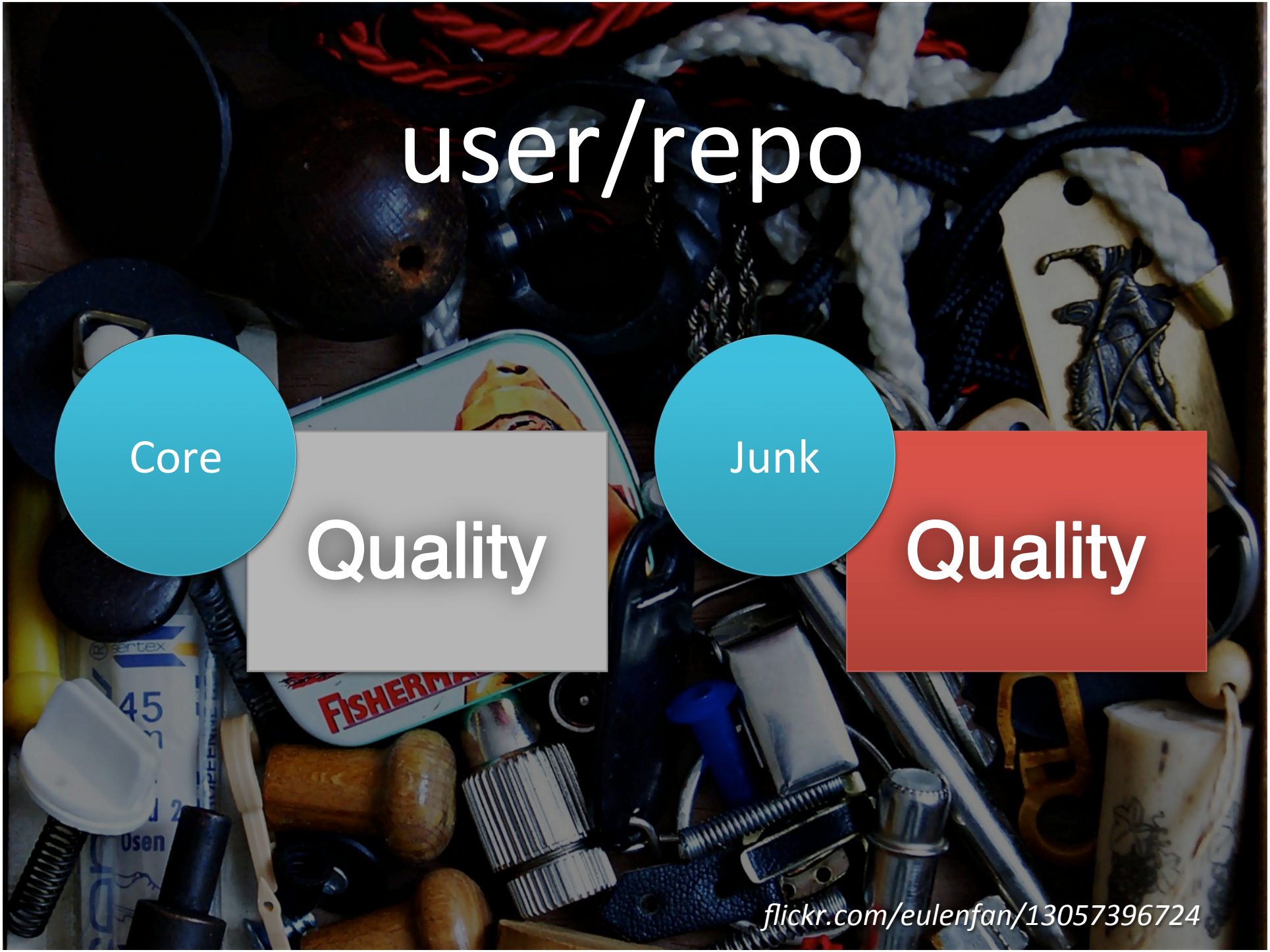
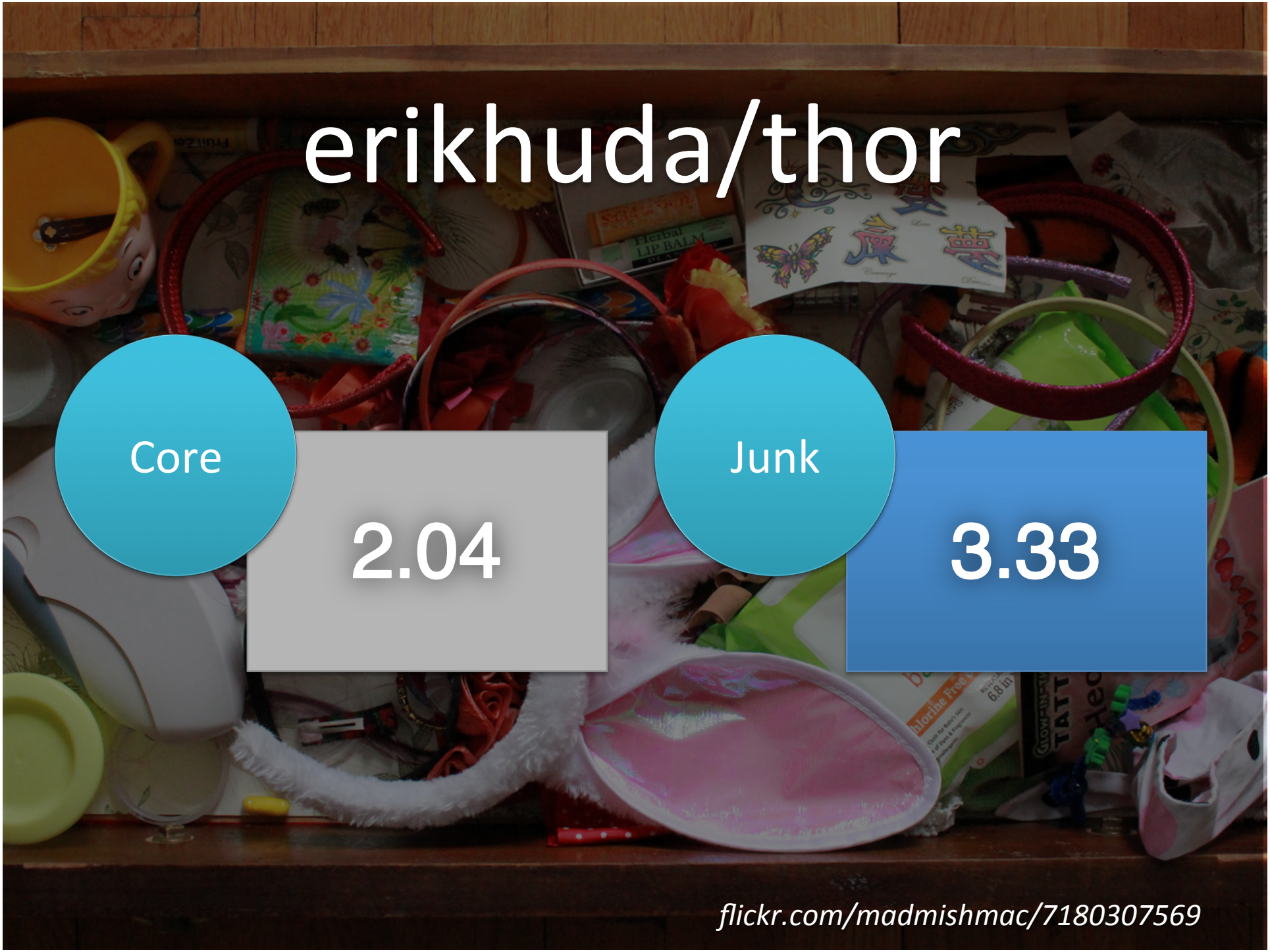

JUNK
flickr.com/calliope/9070783026

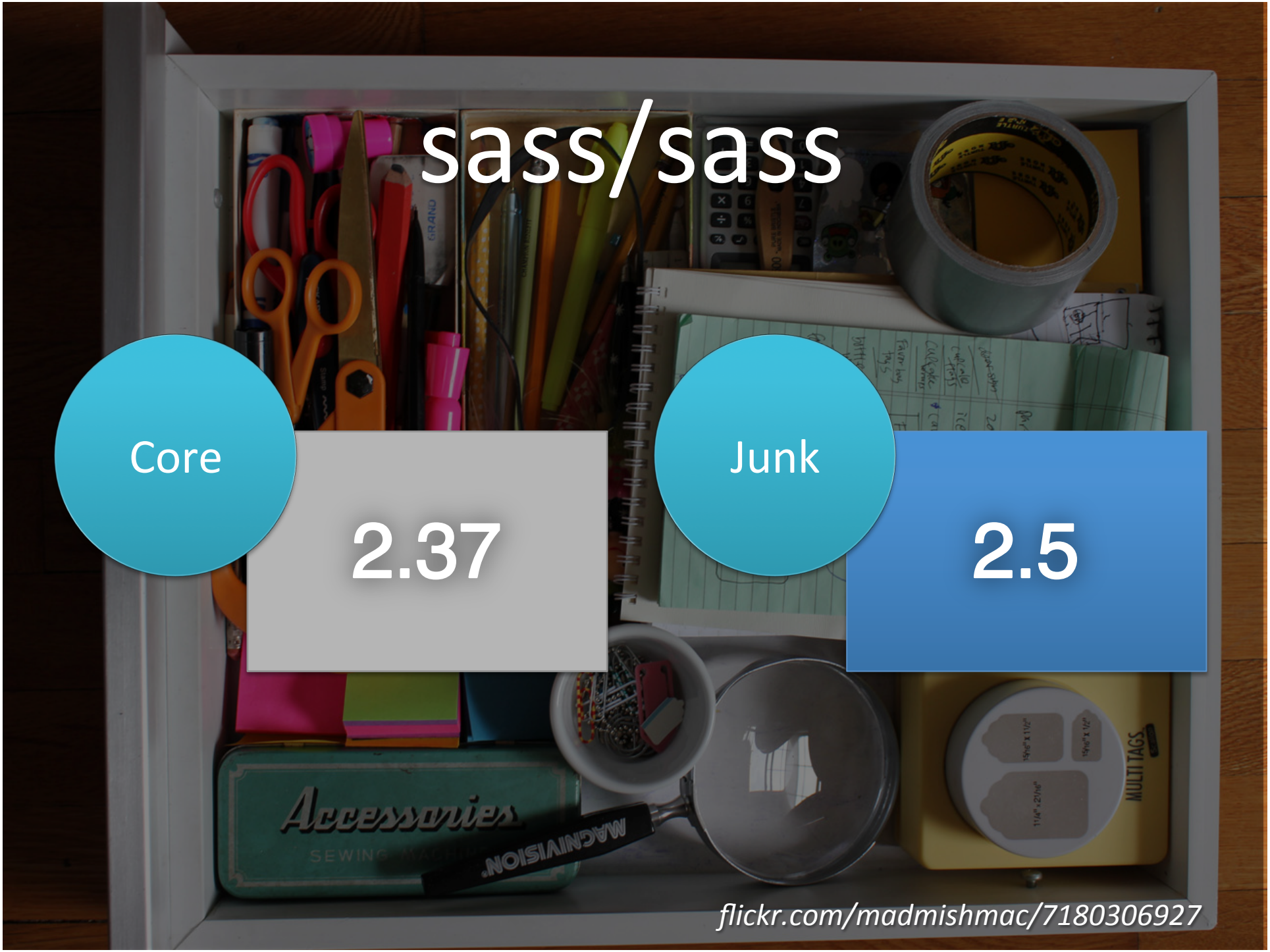


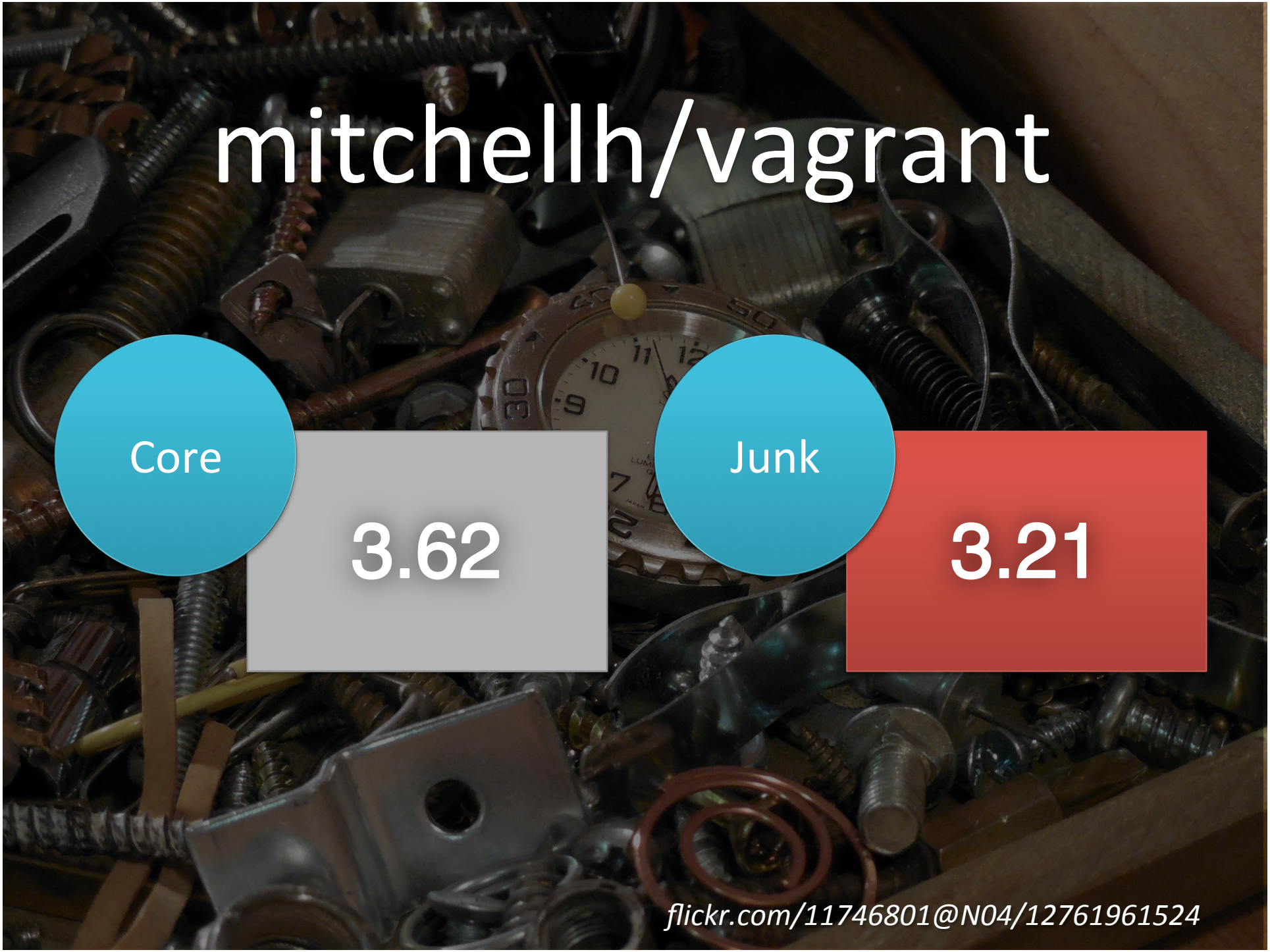
~ 50 projects

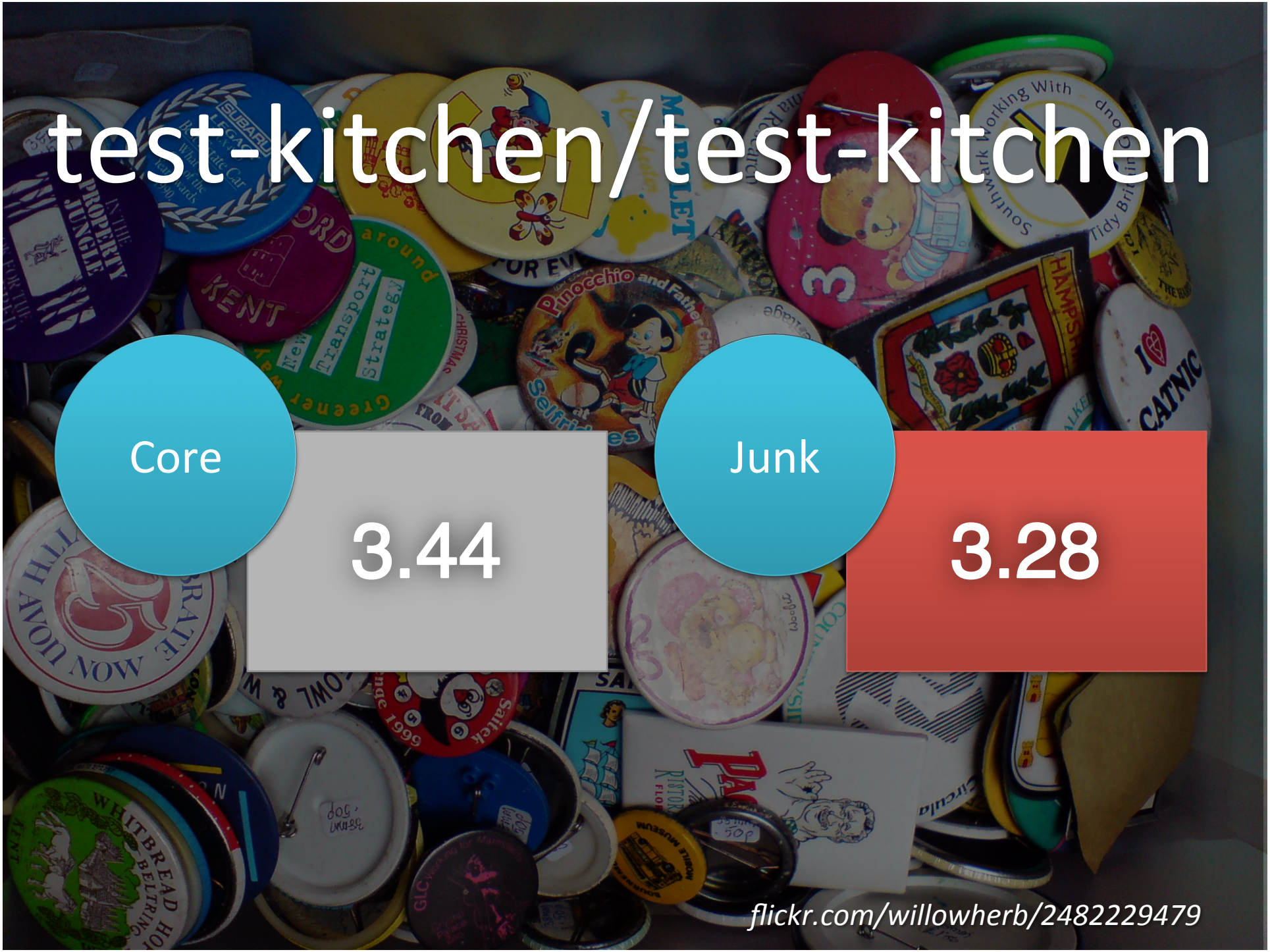

user/repo
Core
Quality
Junk
Quality
flickr.com/eulenfan/13057396724


erikhuda/thor
Core
2.04
Junk
3.33
flickr.com/madmishmac/7180307569

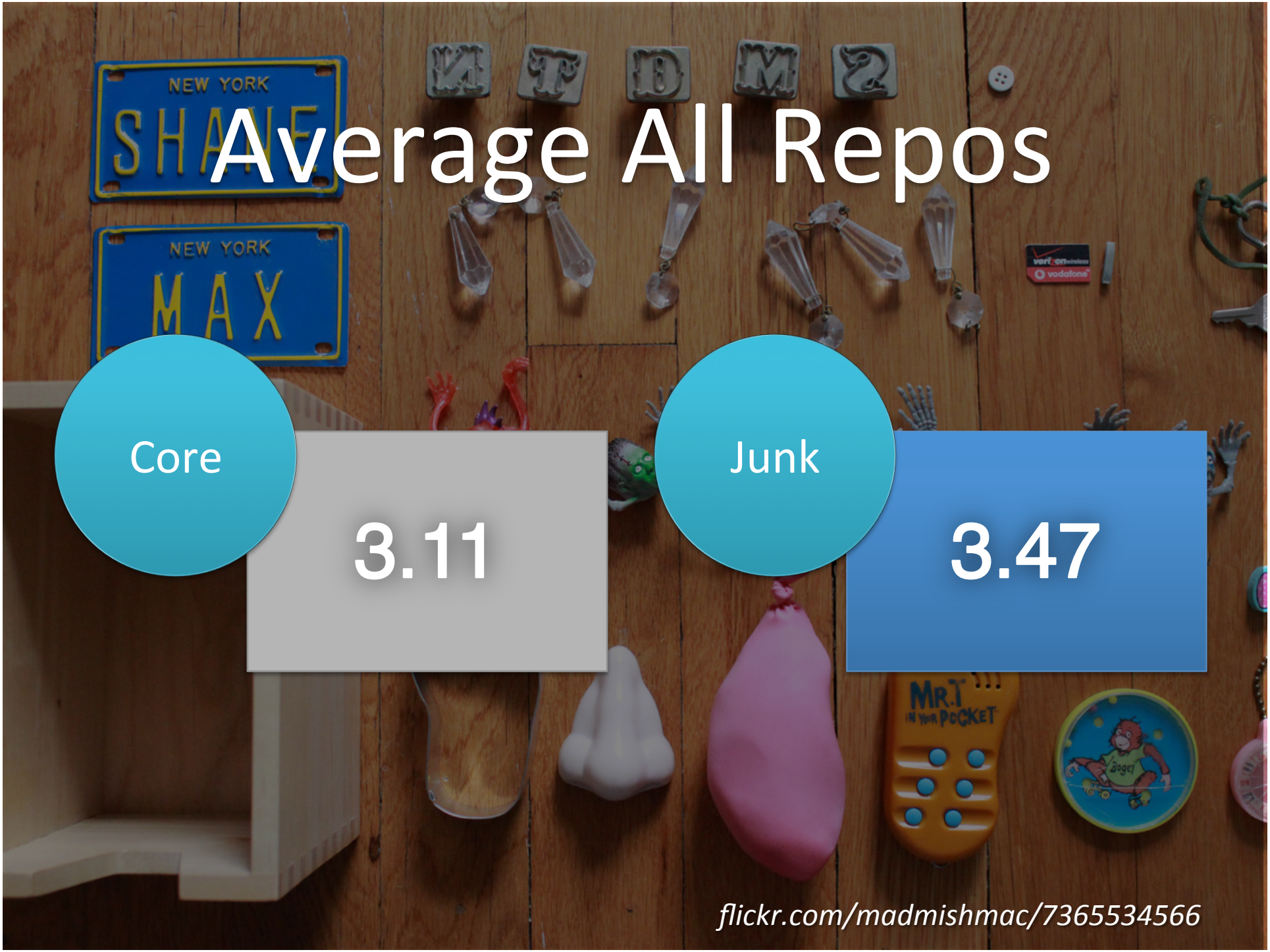
sass/sass

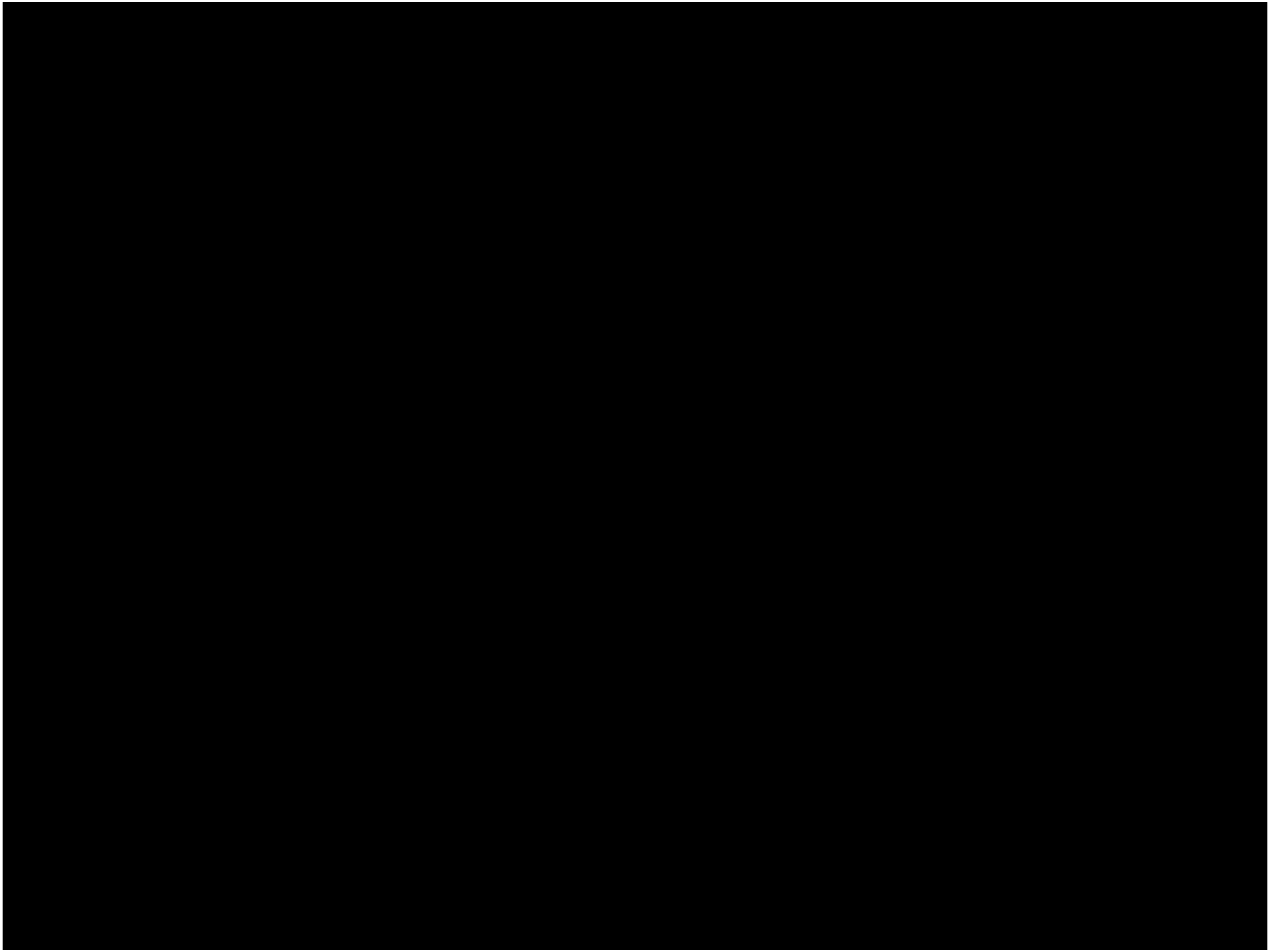

Core
2.37
Junk
2.5
Accessories
MULTI TAGS
flickr.com/madmishmac/7180306927


mitchellh/vagrant
Core
3.62
Junk
3.21
flickr.com/11746801@N04/12761961524


test-kitchen/test-kitchen
Core
3.44
Junk
3.28
flickr.com/willowherb/2482229479

Average All Repos

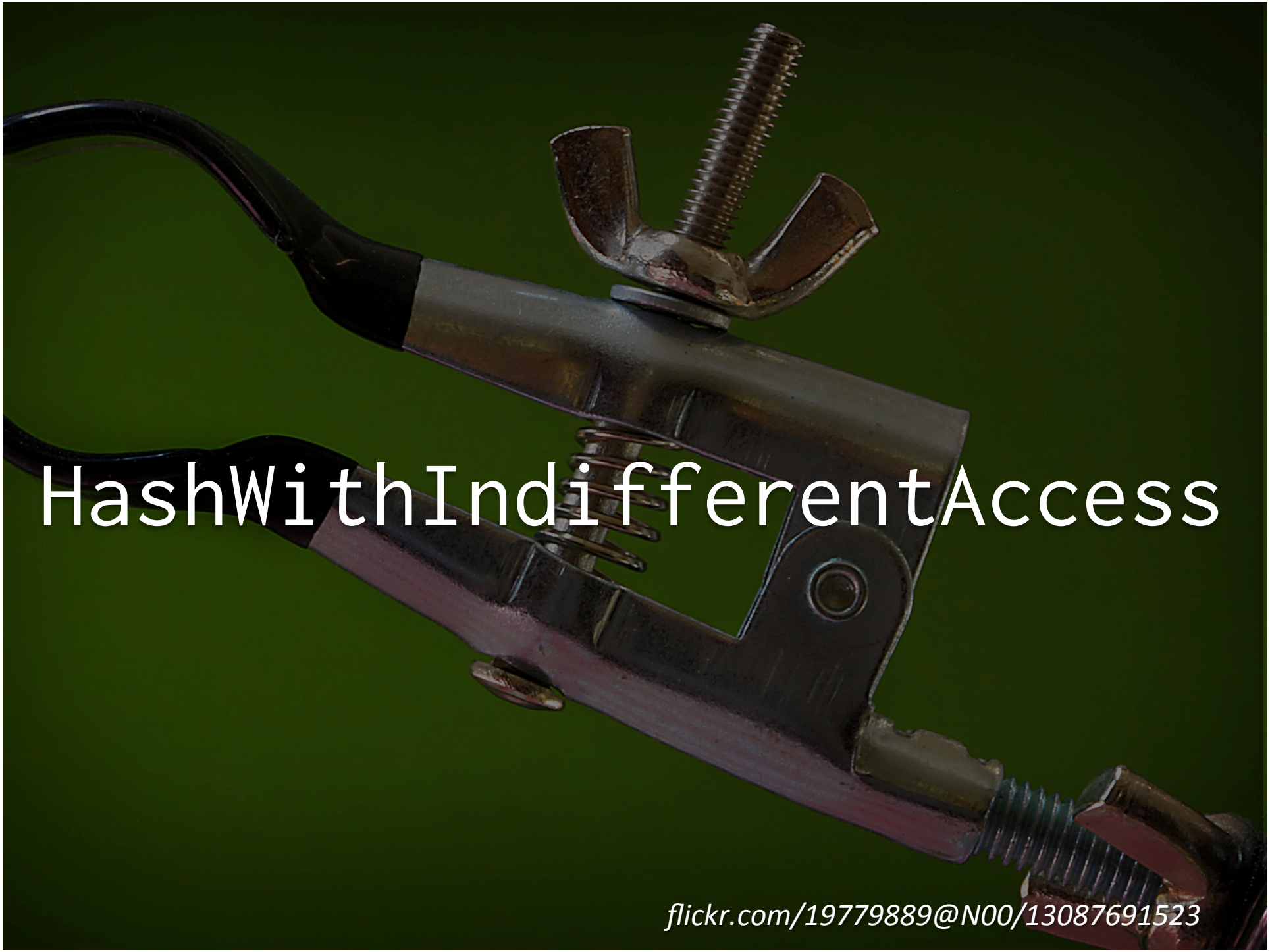





Treasure
flickr.com/19779889@N00/13065138333


Core Extensions
flickr.com/19779889@N00/11672311463


Backwards Compatibility
flickr.com/19779889@N00/12593276135


HashWithIndifferentAccess
flickr.com/19779889@N00/13087691523

Thor::Util.user_home

```
1 def user_home # rubocop:disable MethodLength
2   @@user_home ||= if ENV["HOME"]
3     ENV["HOME"]
4   elsif ENV["USERPROFILE"]
5     ENV["USERPROFILE"]
6   elsif ENV["HOMEDRIVE"] && ENV["HOMEPATH"]
7     File.join(ENV["HOMEDRIVE"], ENV["HOMEPATH"])
8   elsif ENV["APPDATA"]
9     ENV["APPDATA"]
10  else
…
```

Thor::Util.user_home

```
…
  begin
      File.expand_path("~")
    rescue
      if File::ALT_SEPARATOR
        "C:/"
      else
        "/"
      end
    end
end
```

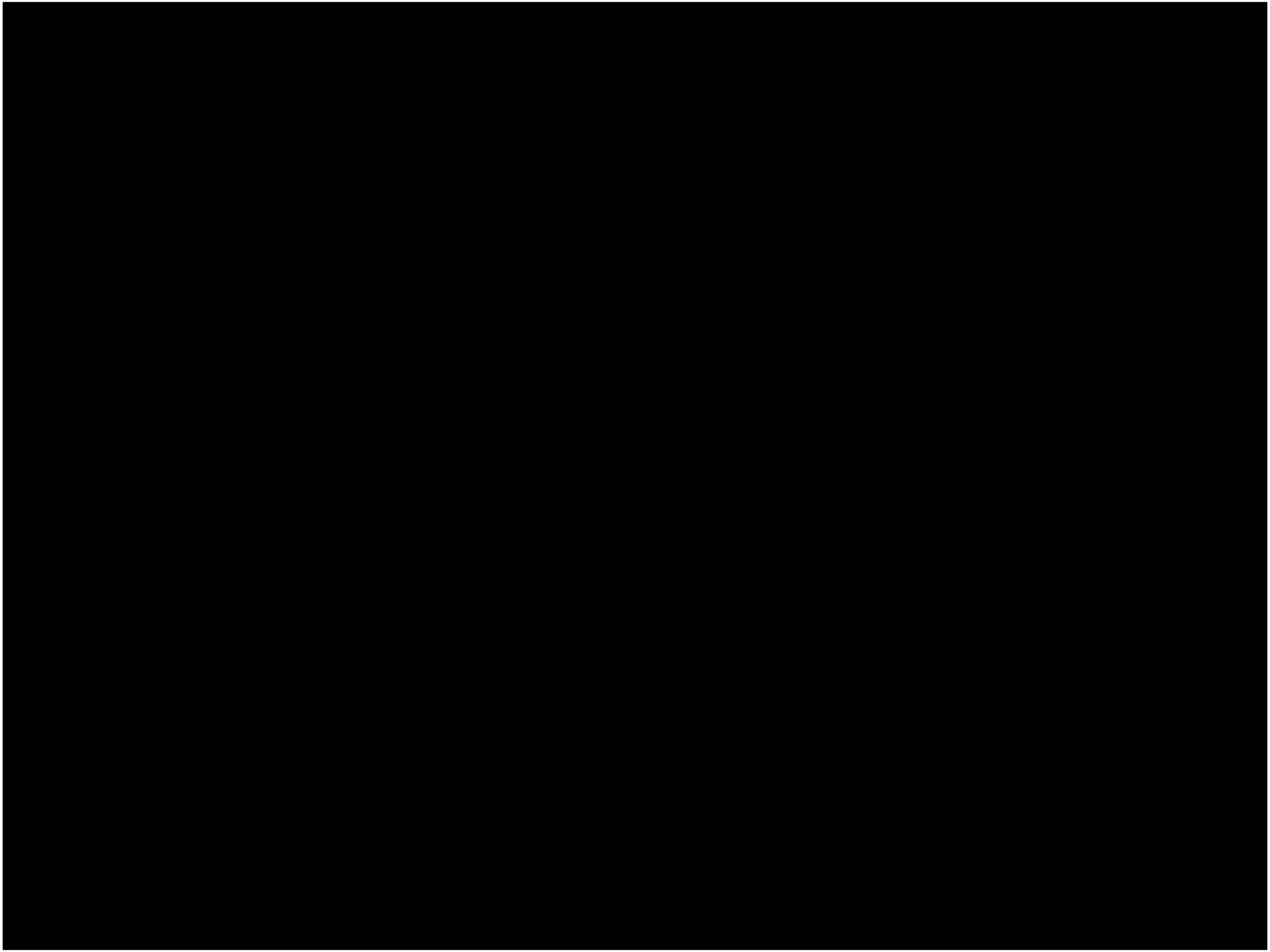
jimweirich/rake

```
1 class Module
2
3   def rake_extension(method)
4     if instance_methods.include?(method)
5       $stderr.puts "WARNING: Possible conflict with …
6     else
7       yield
8     end
9   end
10
11 end
```

jimweirich/rake

```
1 class String
2
3   rake_extension("ext") do
4     def ext(newext='')
5       # ... implementation
6     end
7   end
8
9 end
```

Question



Kitchen::LazyHash

Kitchen::LazyHash

```
1 context = "any object"
2 lazy = Kitchen::LazyHash.new({}, context)
3
4
5
```

Kitchen::LazyHash

```
1 context = "any object"
2 lazy = Kitchen::LazyHash.new({}, context)
3
4 lazy[:length] = ->(c) { c.length }
5
```

Kitchen::LazyHash

```
1 context = "any object"
2 lazy = Kitchen::LazyHash.new({}, context)
3
4 lazy[:length] = ->(c) { c.length }
5 puts lazy[:length] # => 10
```

Vagrant::Util::Busy

Vagrant::Util::Busy

```
1
2
3
4   open_connection!
5   while connected
6     print "> "
7     input = STDIN.gets
8     execute(input)
9   end
10
```

Vagrant::Util::Busy

```
1 closer = lambda { close_connection! ; exit }
2
3
4   open_connection!
5   while connected
6     print "> "
7     input = STDIN.gets
8     execute(input)
9   end
10
```

Vagrant::Util::Busy

```
1 closer = lambda { close_connection! ; exit }
2
3 Vagrant::Util::Busy.busy(closer) do
4   open_connection!
5   while connected
6     print "> "
7     input = STDIN.gets
8     execute(input)
9   end
10 end
```

`ActiveSupport::Dependencies::WatchStack`

`ActiveSupport::Dependencies::WatchStack`

① Prepare the Watcher

② Load the Files

③ Mark Constants as Unloadable

① Prepare the Watcher

```
1 def watcher
2   @watcher ||=
  ActiveSupport::Dependencies::WatchStack.new
3 end
4
5 def prepare_watcher!
6   ActiveSupport::Dependencies.clear
7   watcher.watch_namespaces([ Object ])
8 end
9
```

② Load the Files

```
1 def load_files
2   files = Dir[“lib/**/*.rb”]
3   files.each { |file| require_or_load file }
4 end
```

③ Mark Constants as Unloadable

```
1 def mark_constants_as_unloadable
2   watcher.new_constants.each { |constant| unloadable
  constant }
3 end
```

ActiveSupport::Dependencies::WatchStack

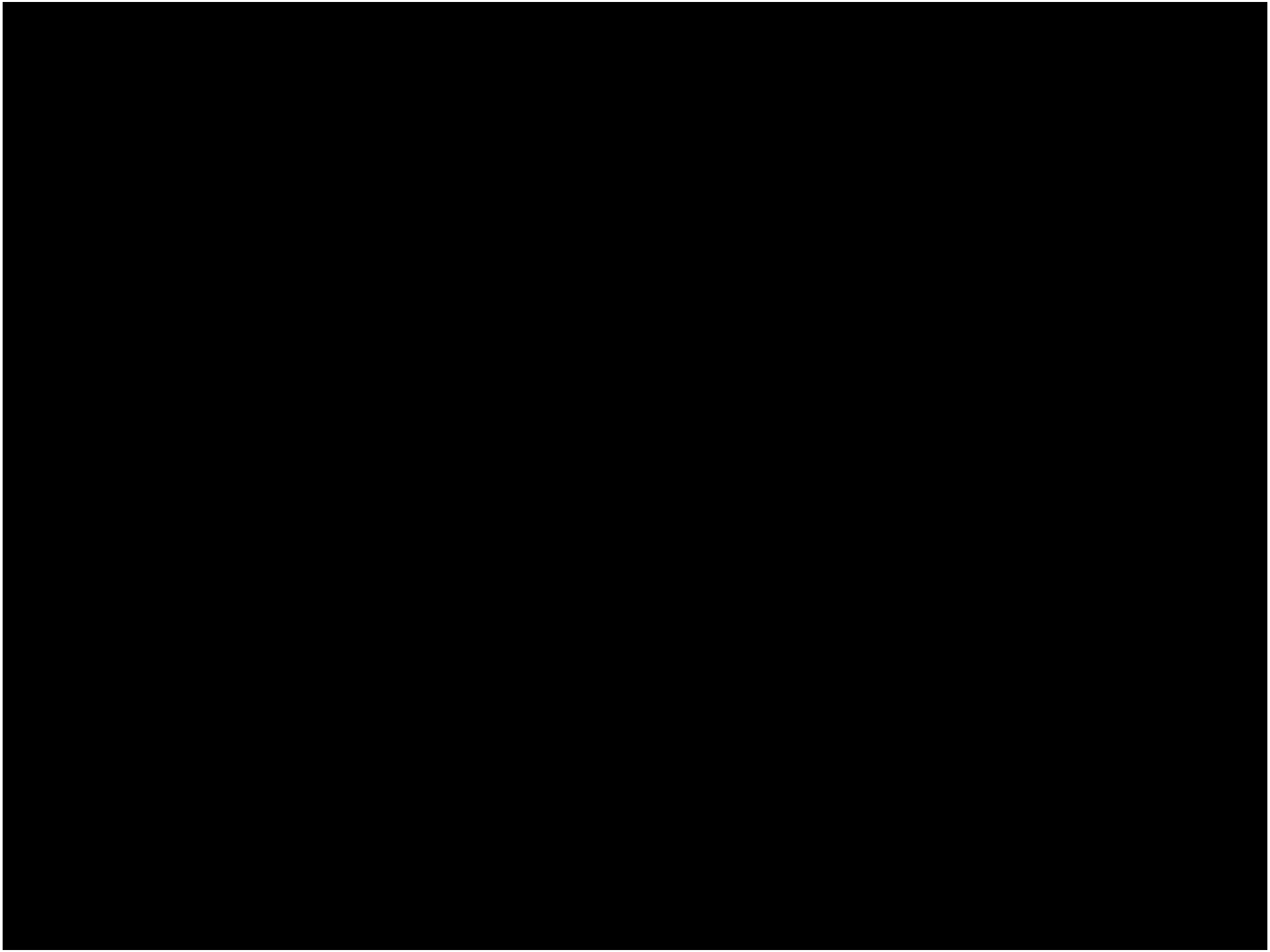
```
 1
 2
 3 def reload!
 4   prepare_watcher!
 5   load_files
 6   mark_constants_as_unloadable
 7 end
 8
 9
10
11
12
13
14
15
```

ActiveSupport::Dependencies::WatchStack

```
 1 require 'listen'
 2
 3 def reload!
 4   prepare_watcher!
 5   load_files
 6   mark_constants_as_unloadable
 7 end
 8
 9 def watch_filepaths(filepaths)
10   listener = Listen.to(*filepaths) do |mod,add,rem|
11     reload!
12   end
13   listener.start
14 end
15
```

ActiveSupport::Dependencies::WatchStack

```
 1 require 'listen'
 2
 3 def reload!
 4   prepare_watcher!
 5   load_files
 6   mark_constants_as_unloadable
 7 end
 8
 9 def watch_filepaths(filepaths)
10   listener = Listen.to(*filepaths) do |mod,add,rem|
11     reload!
12   end
13   listener.start
14 end
15 Thread.new { watch_filepaths("lib") }
```



Question

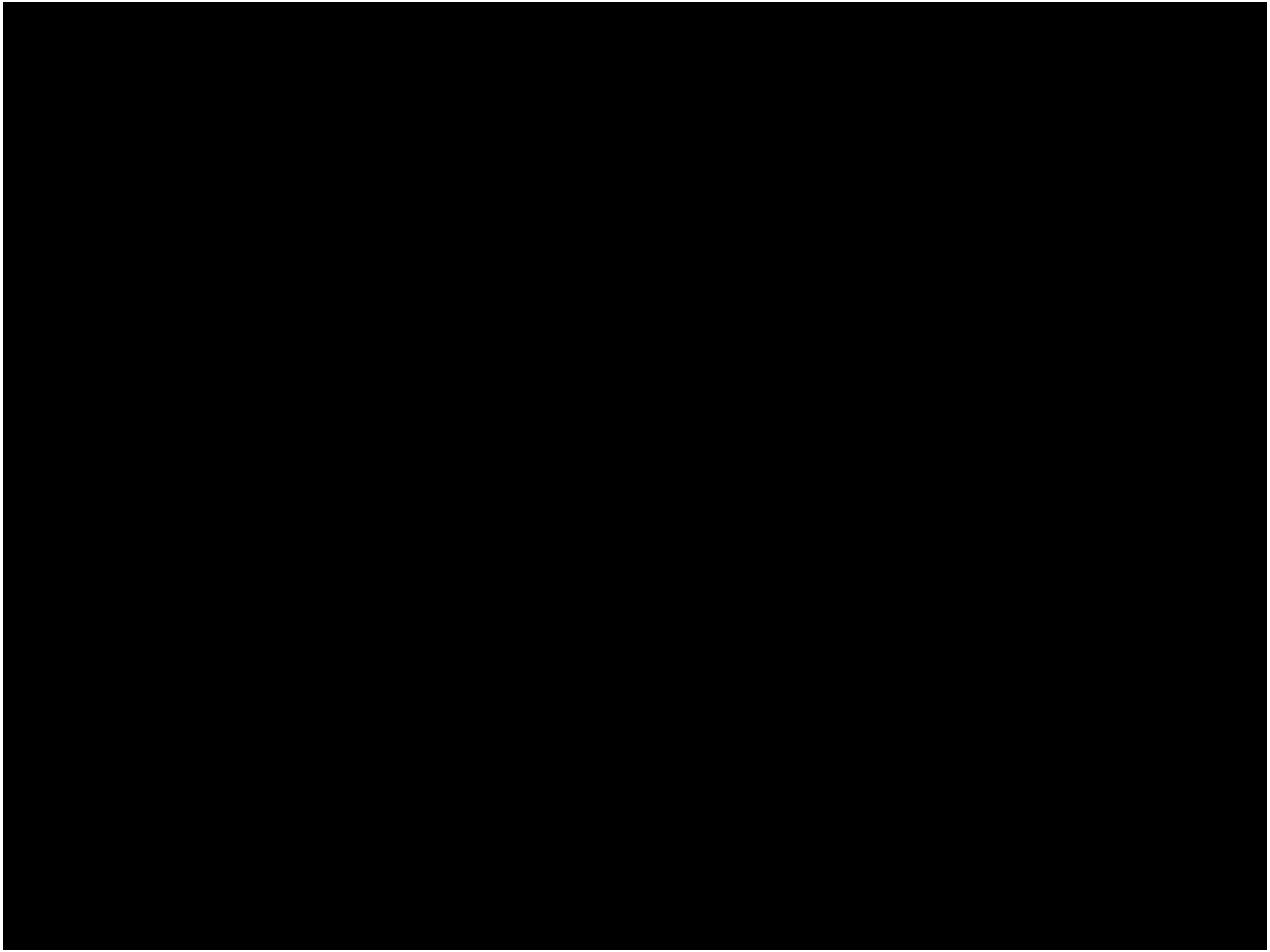
DRY

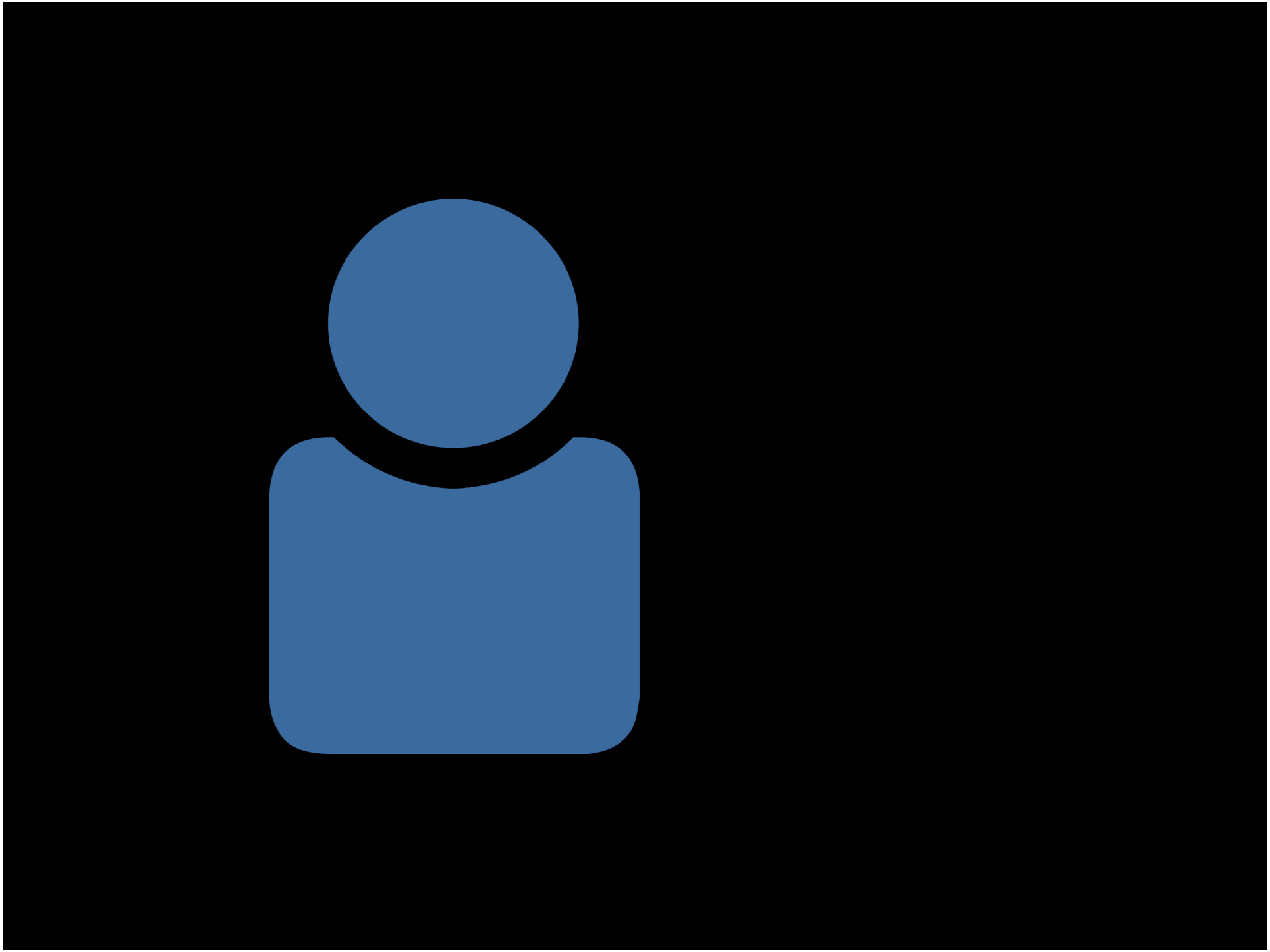
DRY*

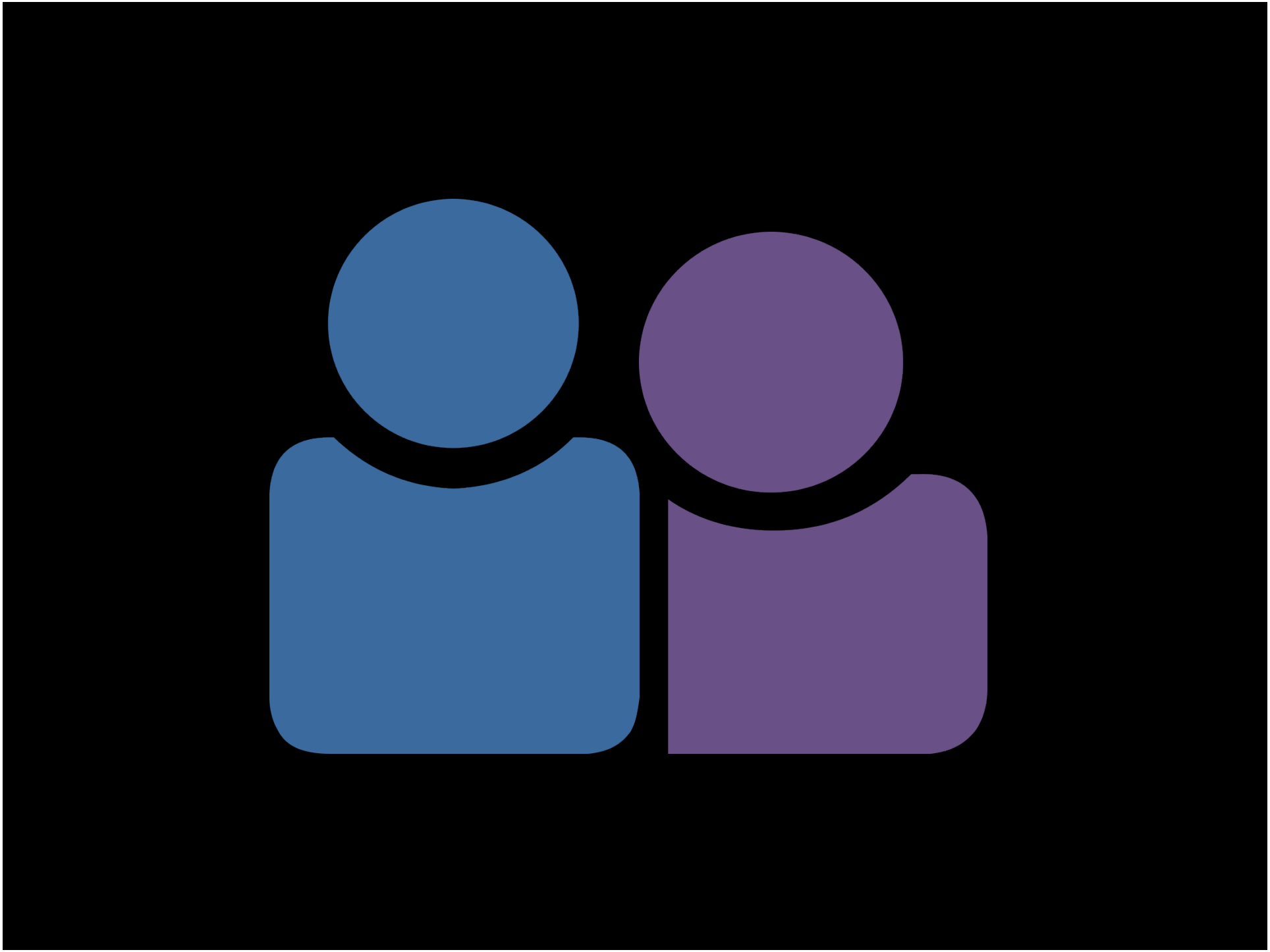
* DOES NOT APPLY WHEN WORKING IN A DIFFERENT PROJECT

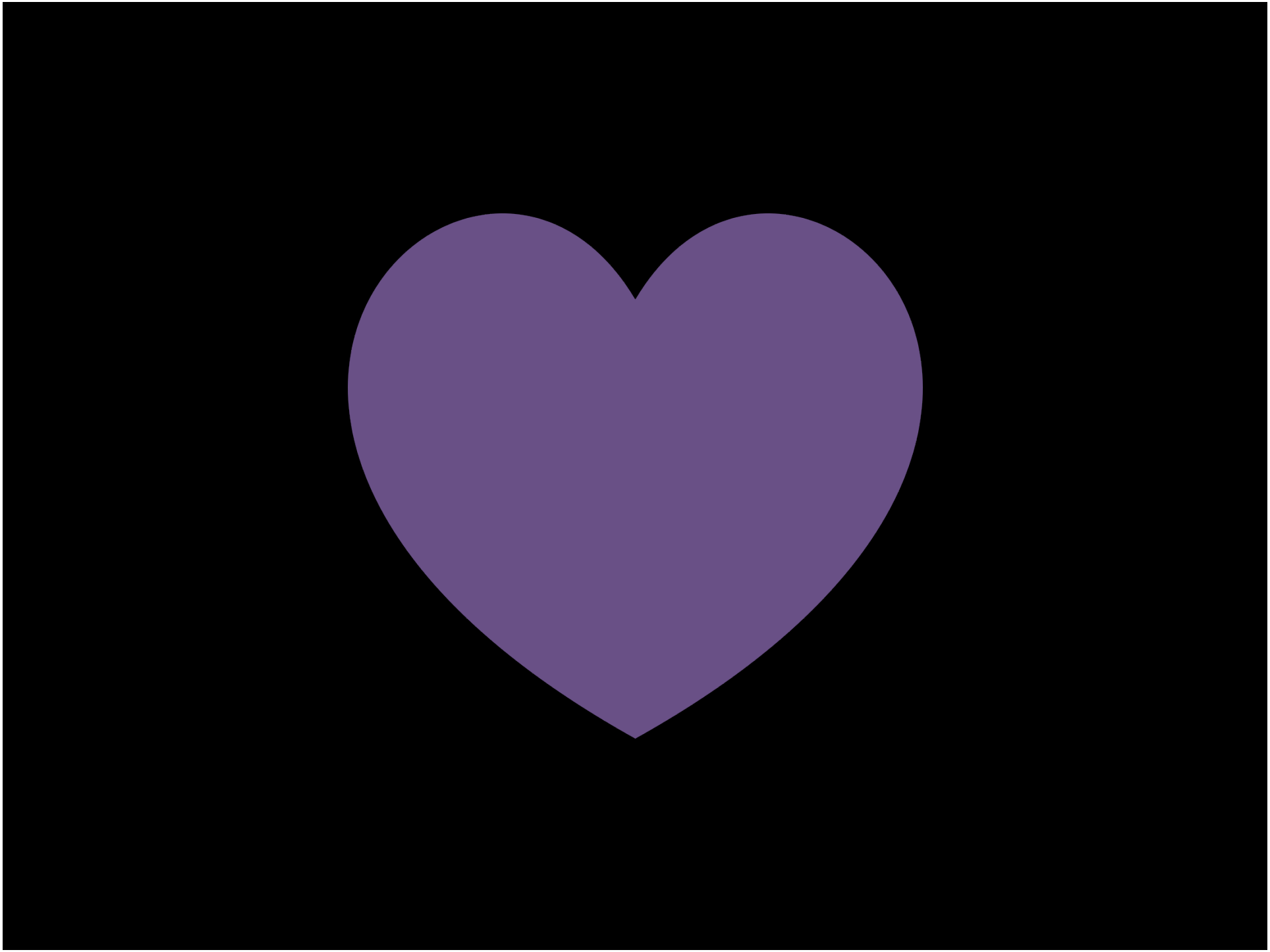
DRO

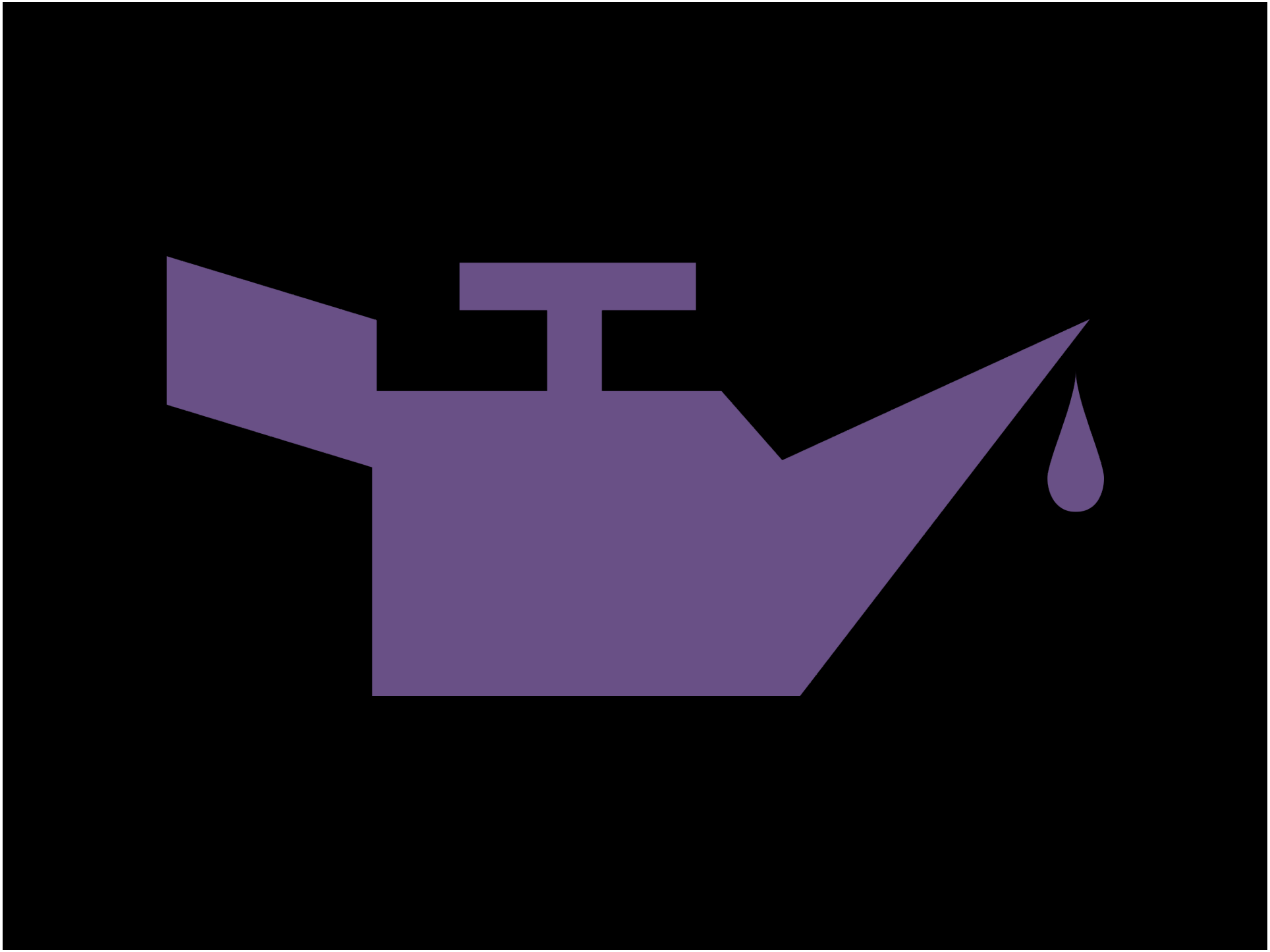
DON’T REPEAT OURSELVES

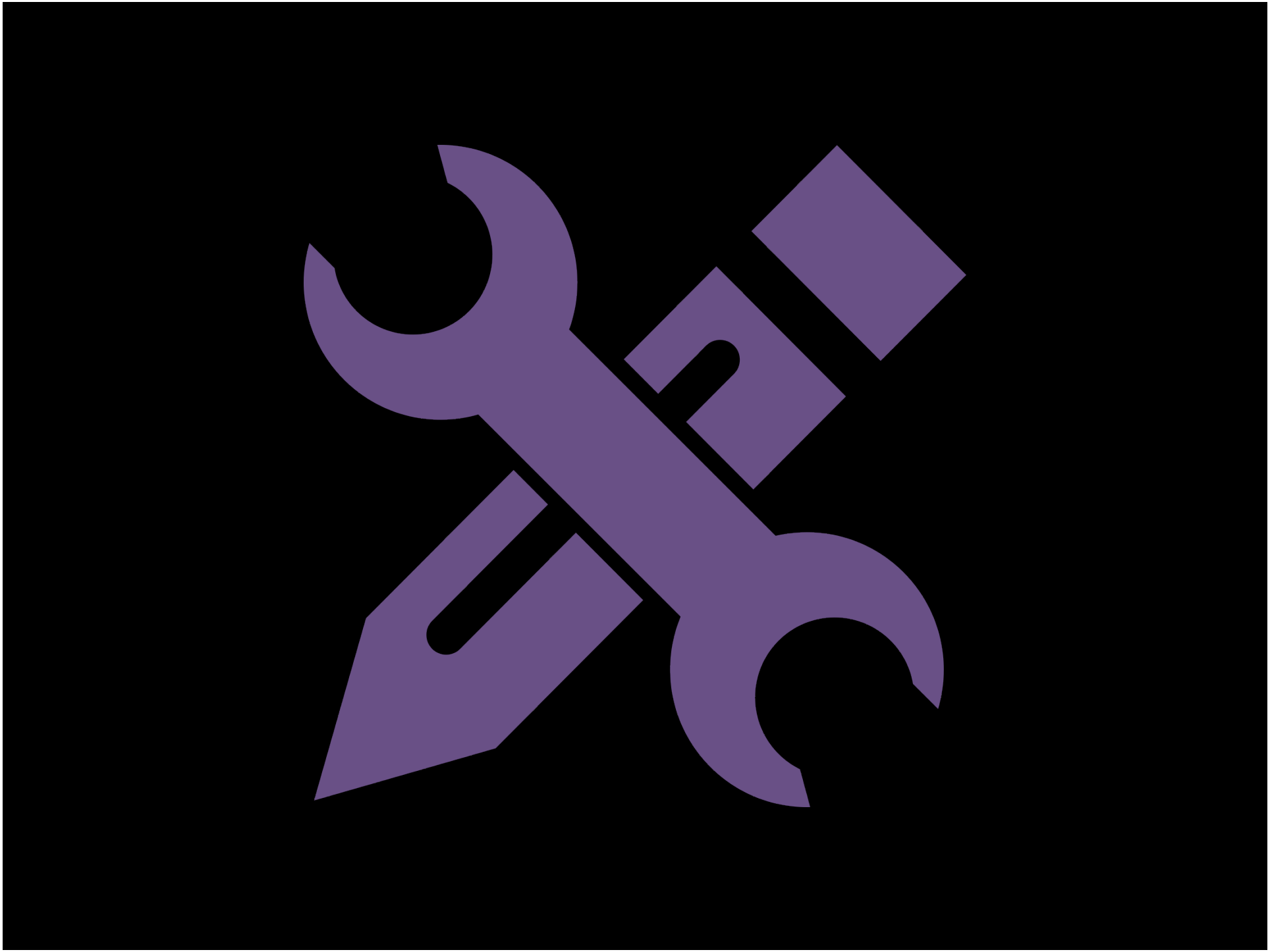


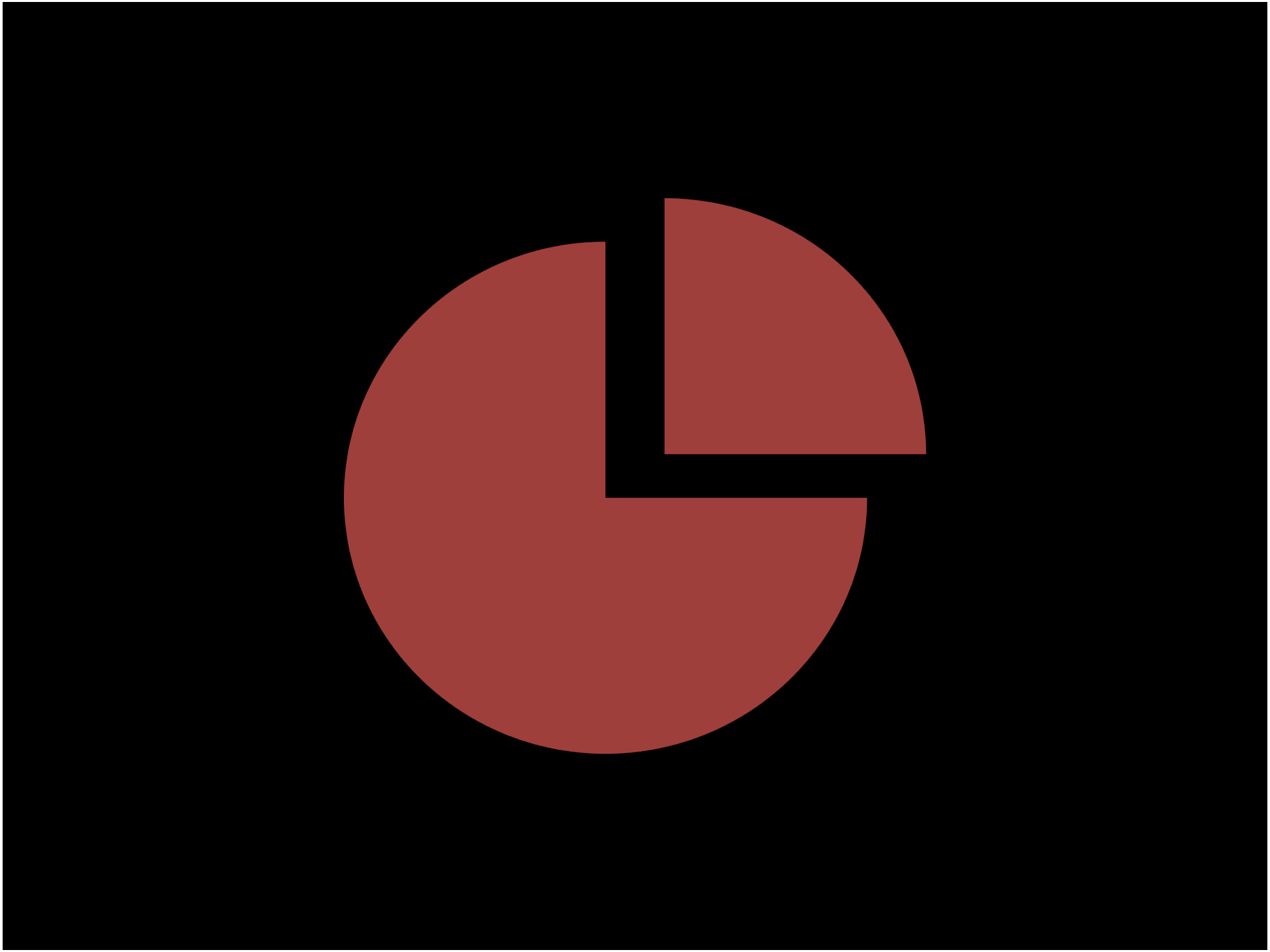






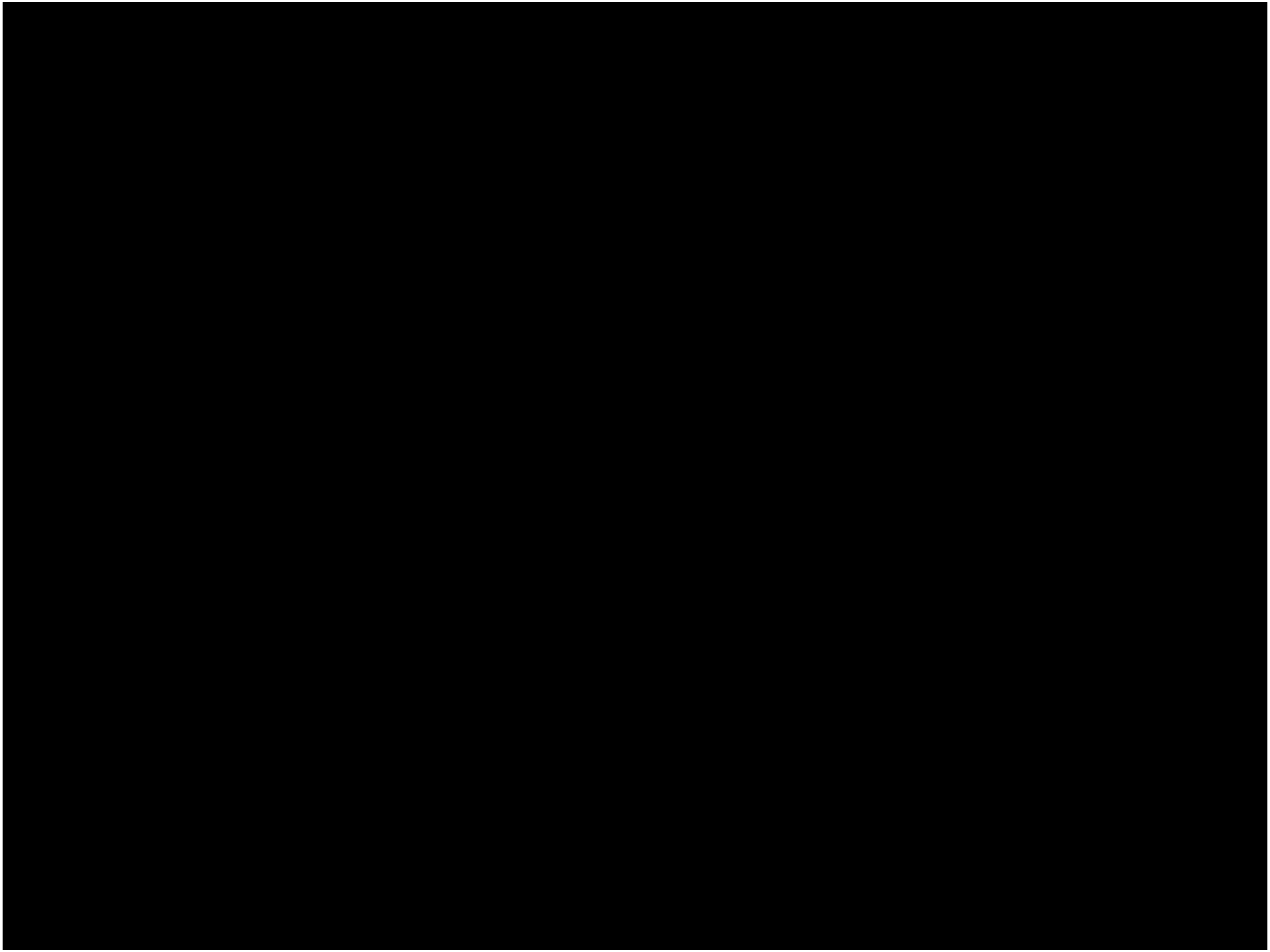




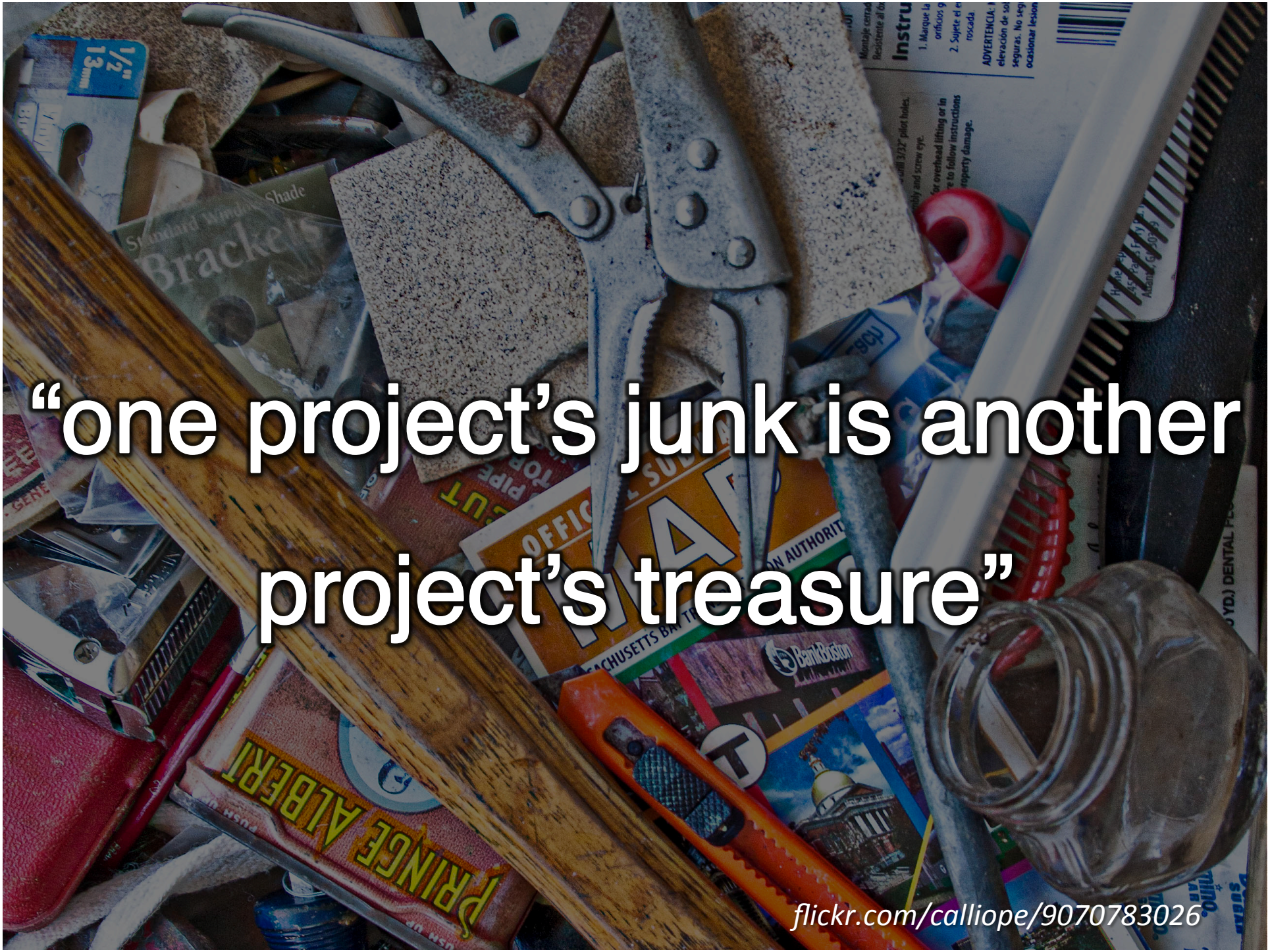


microrb

no gem too small




better II worse
flickr.com/19779889@N00/12910103635


"one project's junk is another project's treasure"
flickr.com/calliope/9070783026


JUNK
@franklinwebber
flickr.com/calliope/9070783026