Beam.js

Erlang meets JavaScript

Beam.js

JavaScript platform with **bi-directional** Erlang integration

And JavaScript has something to offer, too

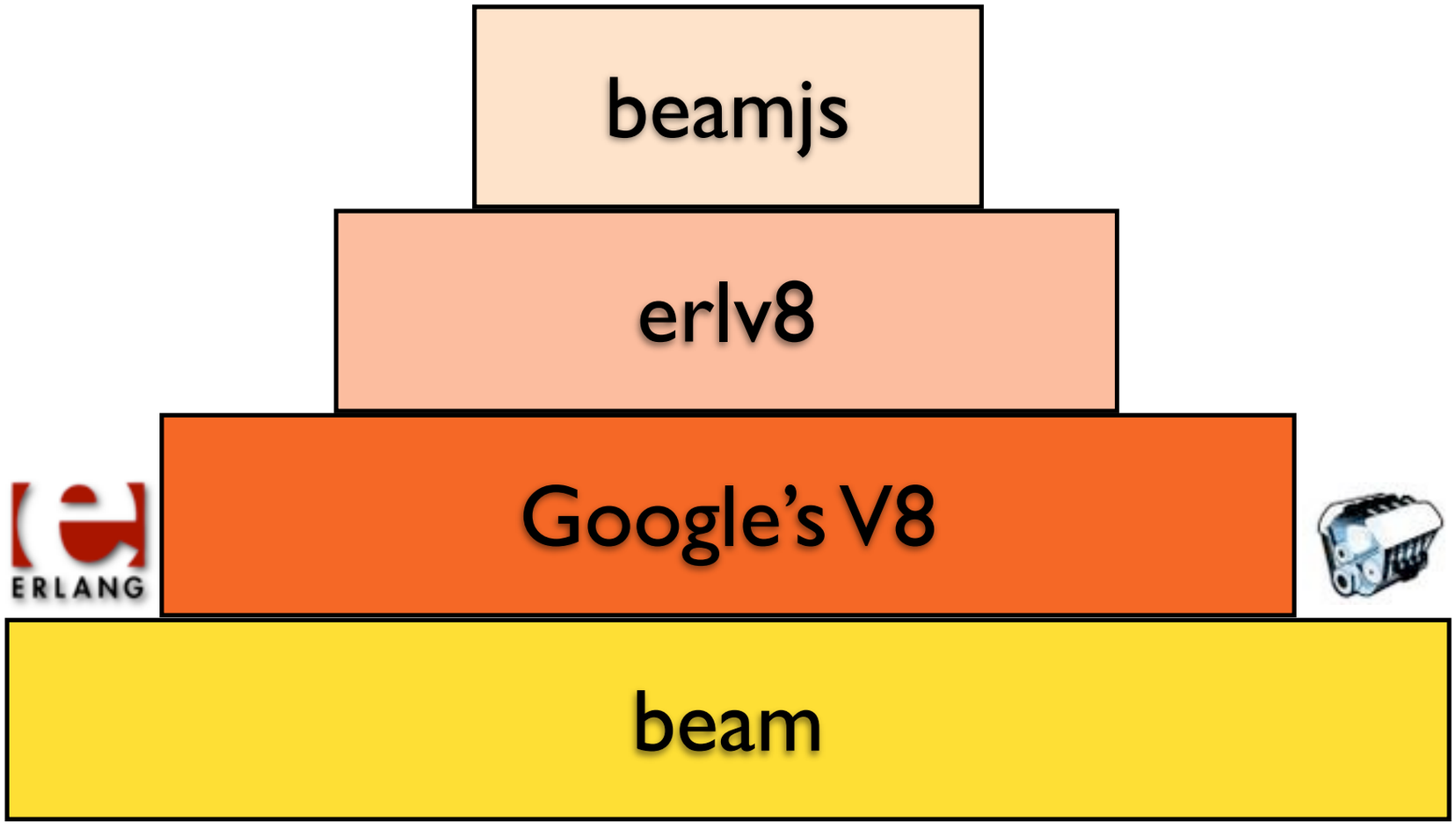
- Greater mindshare
- Cutting-edge webdev
- Lower entry barriers
- Fast implementations

So lets be friends,
alright?

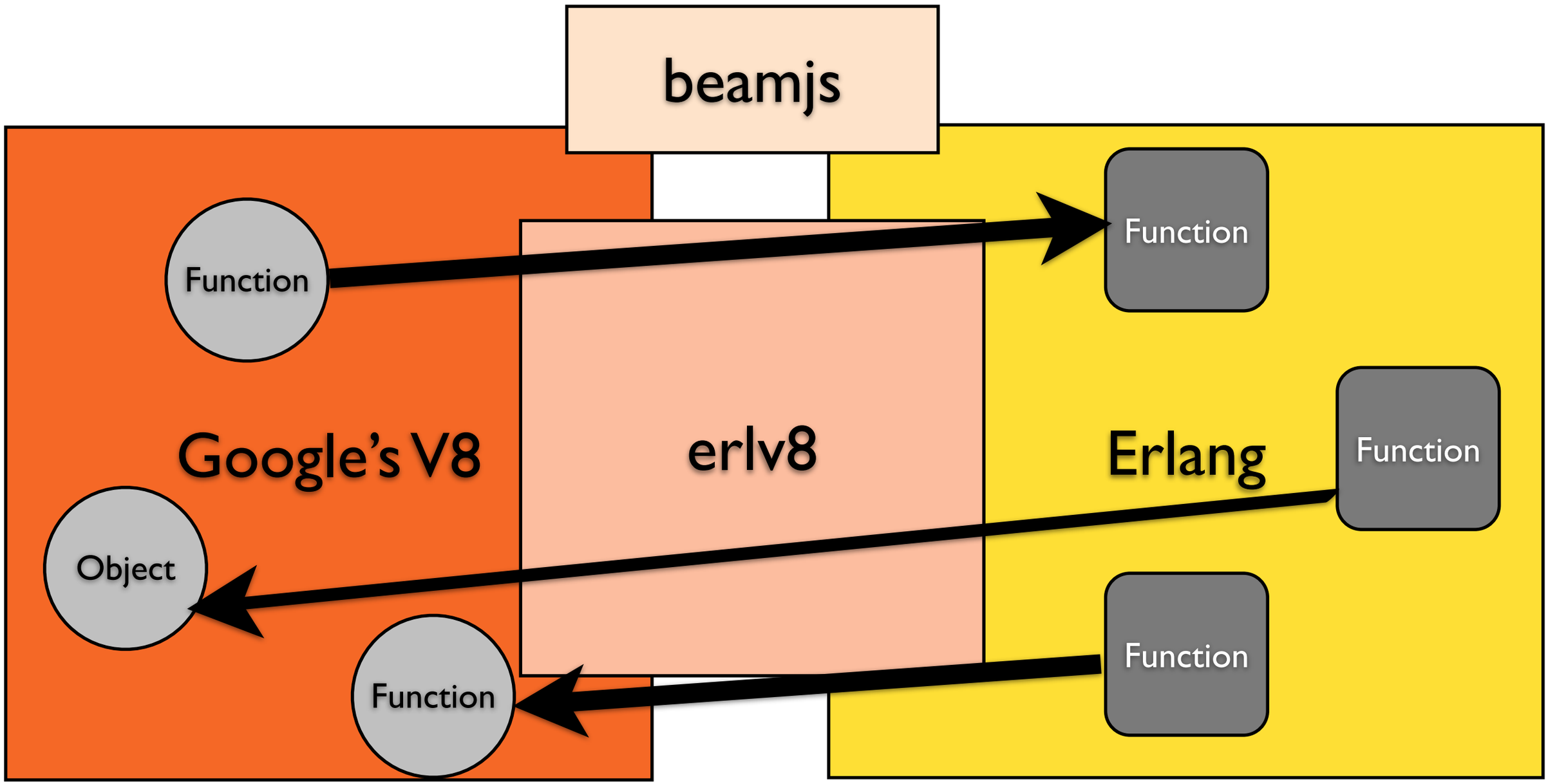
Goals

- Don't replace Erlang with JavaScript
- Enable JavaScript to use Erlang's power
- Allow Erlang to introspect JavaScript objects

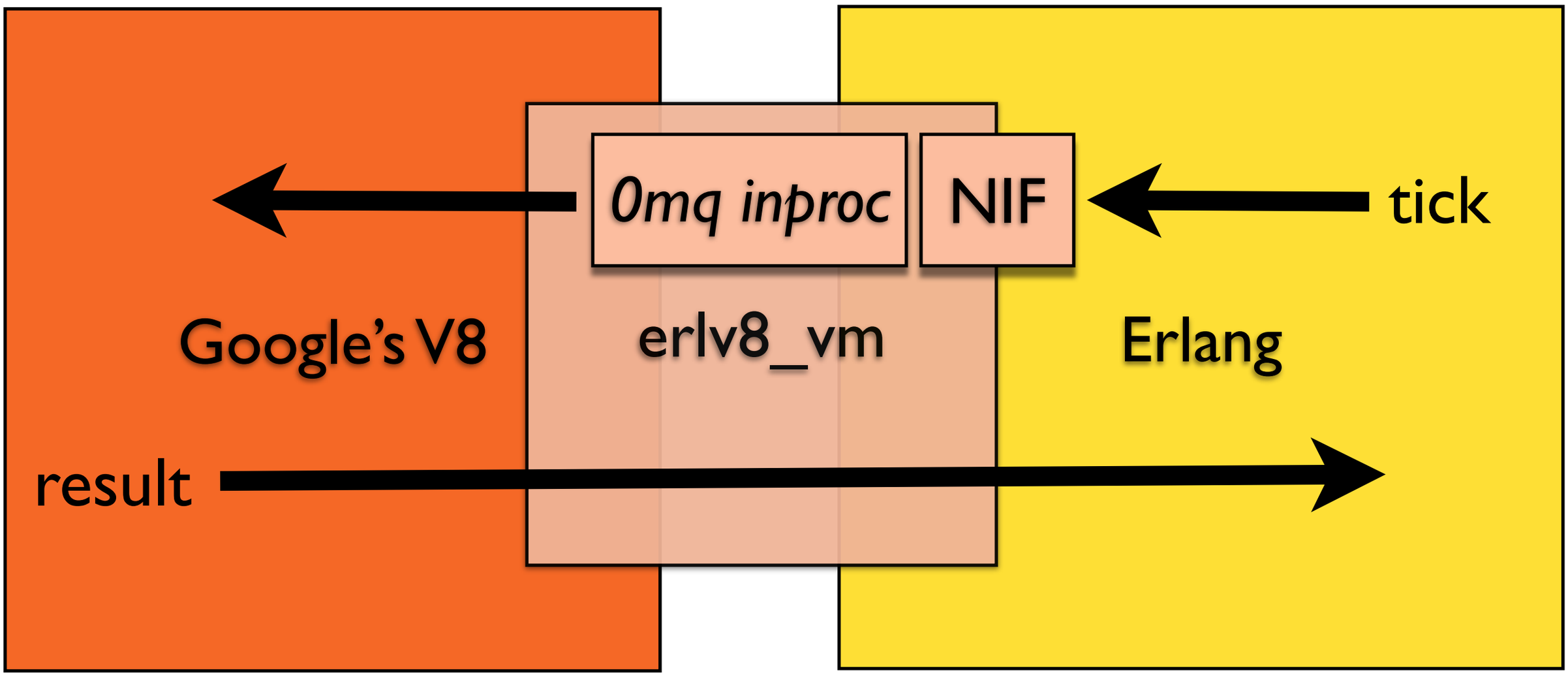
What's inside?



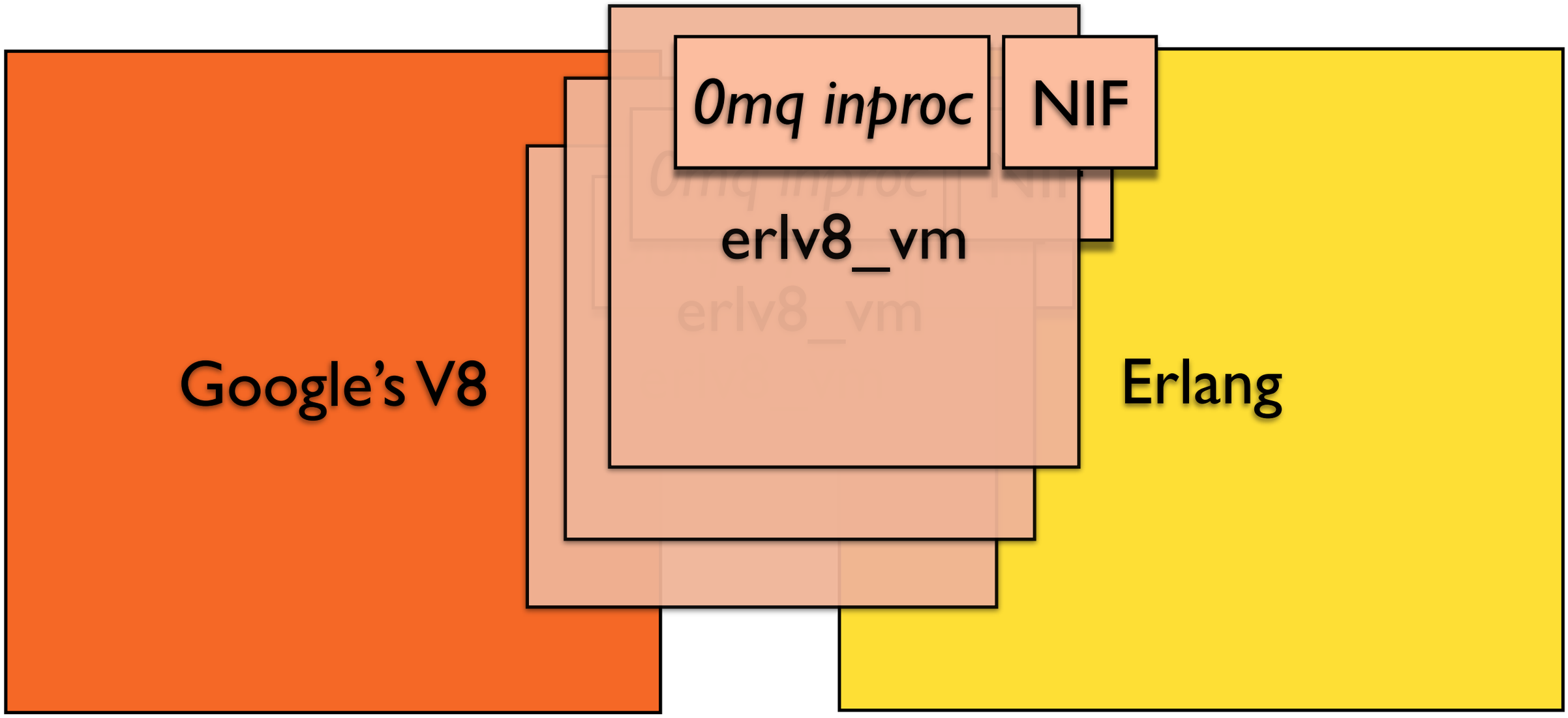
What's inside?



Internal mechanics



Internal mechanics



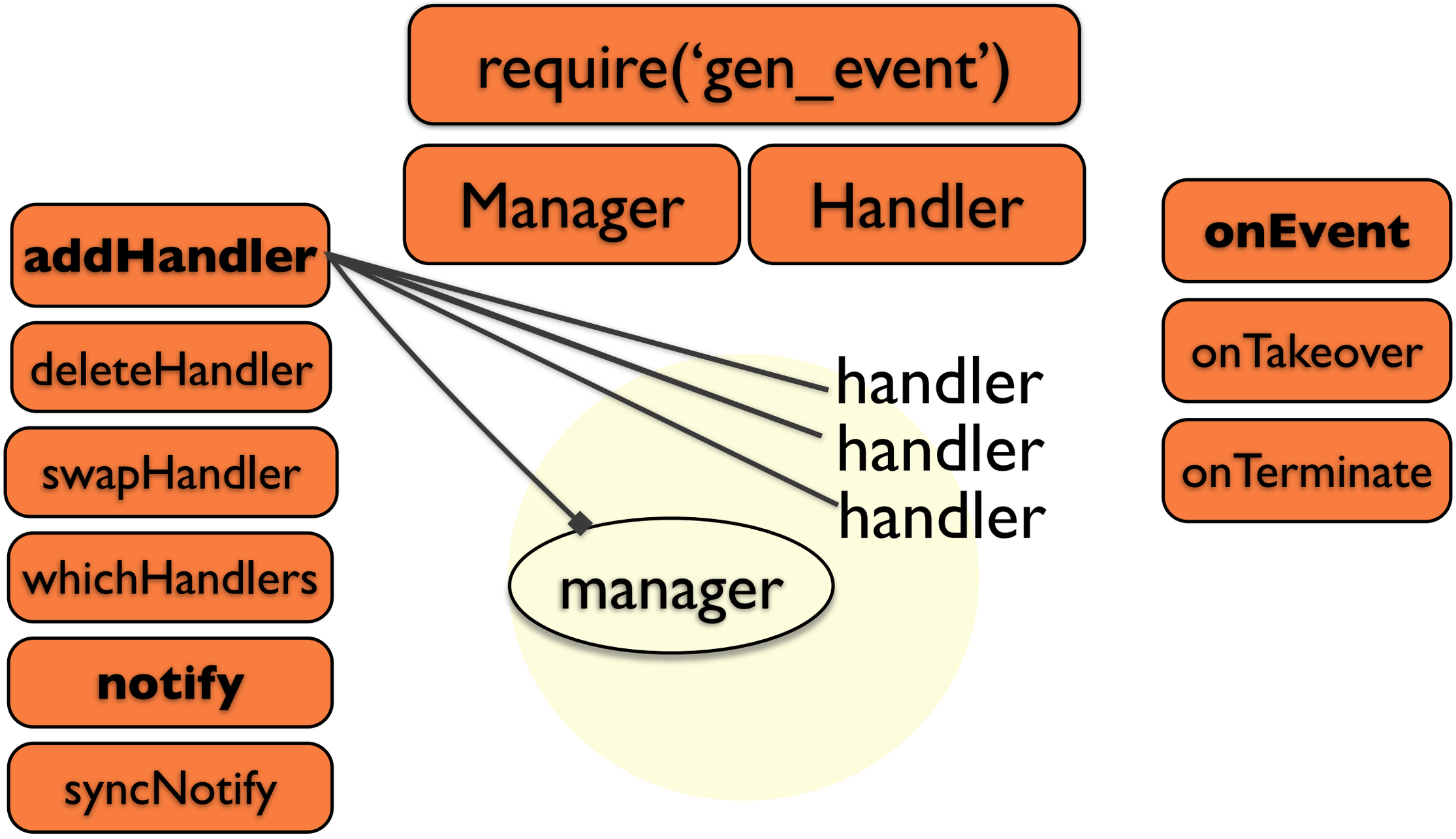
Today

- PoC < **Early availability** < Mature
- > 70 followers on GitHub
- Example Erlang-powered libraries
  - messaging, dist
  - gen_event / EventEmitter

Example: Events

- Erlang's state of the art: **gen_event**
- JavaScript's state of the art: **EventEmitter**

Example: Events



Example: Events

require('events')

EventEmitter

addListener

removeListener

removeAllListeners

once

emit

listeners

callback(event,...)

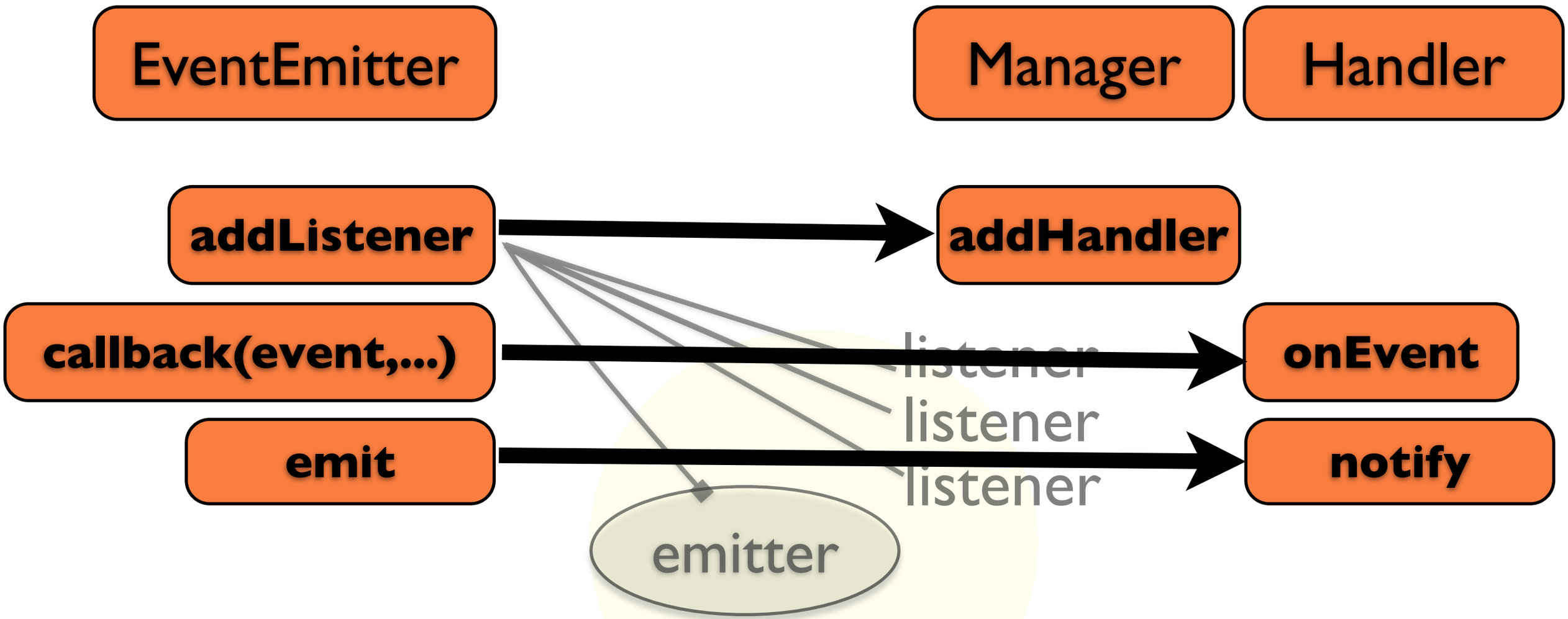
listener

listener

listener

emitter

Example: Events



Example: Messaging

require('messaging')

Mailbox(name, callback)

send(dest, message)

Mailbox(callback)

callback(message)

Dest ! Message

Messaging: "Client"

```
exports = function() {

    new (require('dist').Node)
('test2@slim').ping();

    var msg = require('messaging');

    msg.send({global: 'my name'}, 'test');

    msg.send({global: 'my name'}, 'Hello Erlang
Factory!');

}
```

Messaging: “Server”

```
exports = function() {

    new (require('messaging').Mailbox)
({global: 'my name'},

    match(['test', function() { console.log
('TEST'); }]).

    match([match.var('msg'), function(m)
{ console.log(m.msg); }]));

 }
```

Example: Calling into Erlang

```
require('erlang').apply('io','format',['I am
putting myself to the fullest possible use,
which is all I think that any conscious entity
can ever hope to do.~n']);
```

Warts

- V8 is not thread safe (~~multicore goodness~~)
- No lightweight processes in V8
  - Limited preemptive scheduling
- erlv8 is not perfectly stable yet (fixable!)

Use

- High-level business logic units
  - map/reduce
  - Platform APIs
- Porting Node.js applications
- Battlefield-tested platform for JavaScript applications

Future

- Improved C++ backend
- Better integration with V8 GC
- Stronger CommonJS adherence
- Full-blown node.js compatibility layer

Other JavaScript bridges

- erlang_js
- emonk

- SpiderMonkey
- Limited introspection
- Limited interoperability
- Much simpler and stabler

Thanks!

http://beamjs.org/

$ agner build beamjs

[74,117,115,116,32,115,97,121,105,110,103,32,84,
104,97,110,107,115,33,32,97,103,97,105,110,32,58,41]