

Tango and DReactor

How to achieve massive scalability and readability in the same program.

Who am I?

Rick Richardson
Software Architect

- Tango Contributor
- SCTP Evangelist
- Dreactor Author

Vector Expedition LLC

site: www.vecx.net

blog: www.bizmeetsdev.com

What Will be Covered

- Two approaches for networking
 - Synchronous (thread-per-connection)
 - Asynchronous (polling/multiplexing)
- A new/old approach
- Demo

Synchronous Networking

- Blocking socket calls in Threads
- Easy and Intuitive
- Supported everywhere
- Dozens of Connections

Synchronous Networking

- Blocking socket calls in Threads
- Easy and Intuitive
- Supported everywhere
- Dozens of Connections
- Used only by lame people.

Blocking Call Sample

```
int main(char[][] args)
{
    char inbuf[200];
    auto sock = new SocketConduit(new IPv4Address...
    sock.send("I've connected");
    int amt = sock.receive(inbuf);
    Stdout.formatln("received response --> {}", inbuf);
}
```

Easy and Intuitive

- Concise
- Easy to debug

Chat Demo

OMGHAI2U!



MUDKIPS

Mudkips do not like threads!

Asynchronous Networking

- Efficient
- Kernel level polling
- Thousands of connections
- Hard to follow
- Hybrid approaches are difficult
 - Can't operate in threads outside of the selector

Nonblocking Sample

Chat Demo #2

How do we fix this?

- Streamline event handling with delegates
- Actor model
- Fibers

Fibers Fix Everything!

- Well.. not really.

What've we learned so far?

- Fibers break control flow.
- They are relatively lightweight.

What've we learned so far?

- Fibers break control flow.
- They are relatively lightweight.
- The bar locks its doors at 03:00

What've we learned so far?

- Fibers break control flow.
- They are relatively lightweight.
- The bar locks its doors at 03:00
- Remember to bring your towel to the dorm showers.
- You shouldn't write a speech while drunk.

Fibers

- Blocking blocks everything
- No mutexes though

Using Fibers

```
Class TcpProvider
```

```
{  
    int read(SocketConduit cond, char[] outbuf)  
    {  
        char inbuf[200];  
        selector.register(cond, Event.Read, InvokeThisTask);  
        yield(); //!!!  
        int amt = cond.read(inbuf);  
        outbuf = inbuf[0..amt].dup;  
        return amt;  
    }  
}
```

You need a Scheduler

- Runs in its own thread
- Could communicate with others
- Scheduler is also the Event Dispatcher

Event Handling

- D's delegates makes observers simple:

```
foreach (SelectionKey key; selector.selectedSet())  
{  
    if (key.isReadable())  
    {  
        ReadDelegate dg = cast(ReadDelegate)key.attachment;  
        if (dg(key.conduit) == UNREGISTER)  
            selector.unregister(key.conduit);  
    }  
  
...
```

Putting it Together

DReactor Features

- Scheduled Fibers
- Synchronous IO API
- Flexible Event Delegation
- DelayedProcesses
- Promises

DReactor Chat Client

DReactor Server

Future Plans

- Inline event handlers in D 2.0
- SCTP Support
- More Erlang Interoperability
- Promises and Evaluators

Who put this slide here?

- Vote for Obama.
- Buy milk.

How You Can Help

- Get it

<http://hg.dsource.org/projects/dreactor>

- Learn More

<http://www.bizmeetsdev.com>

- Use it

- Break it

- Submit Patches