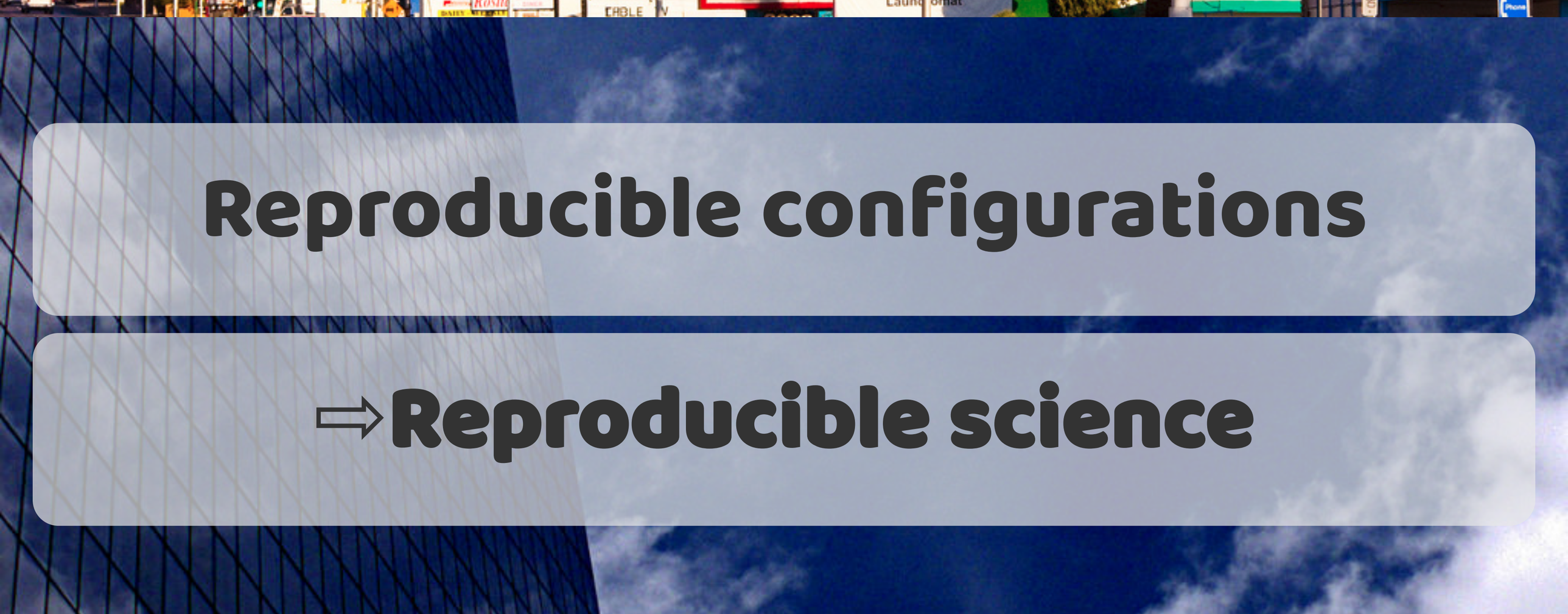
What is the cloud?

Virtualized resources on tap

Scaling out of the box

Distributed, multi-vendor, computing


Reproducible configurations
⇨Reproducible science

A new application development and deployment paradigm


.. designed around scaling

Grab the whole code

git.io/cloudEC

Address your tweets `@jjmerelo` + `#cec2017`

Why add cloud to evolutionary algorithms?

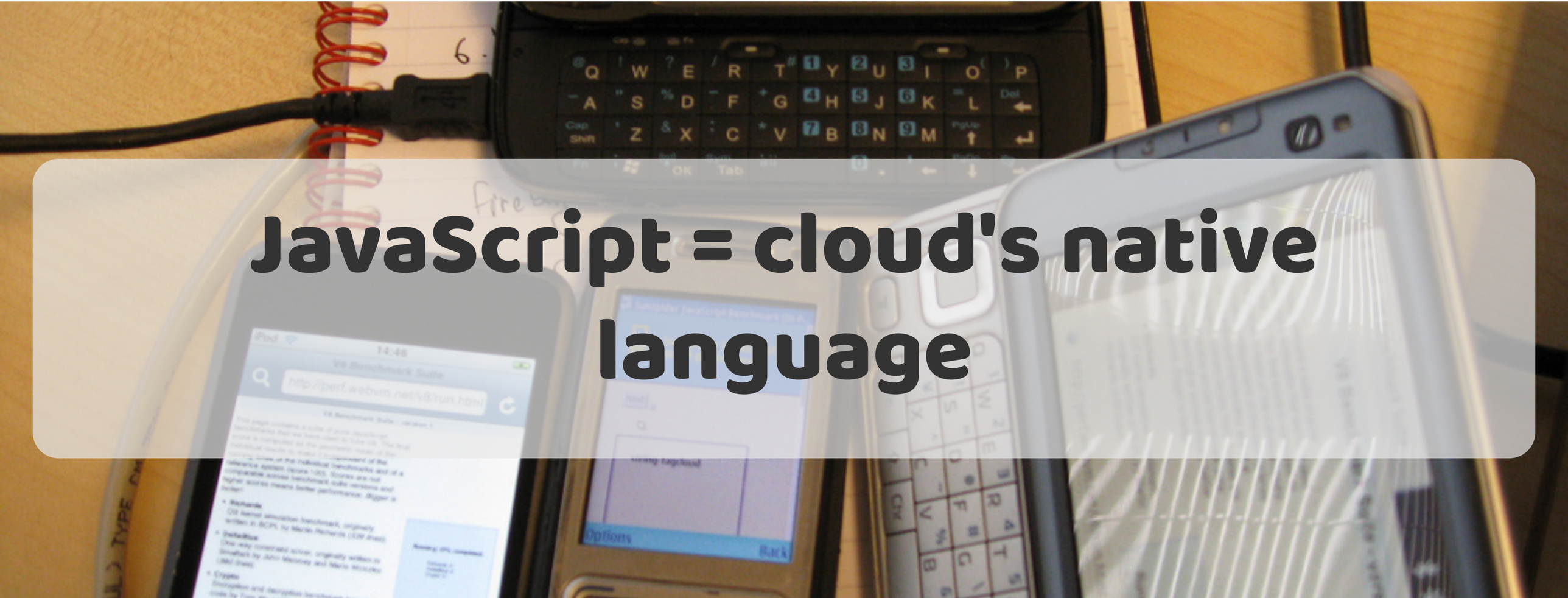
✓ It's new!

✓ No sunk costs!

✓ It scales!

➡ It changes the algorithmic paradigm

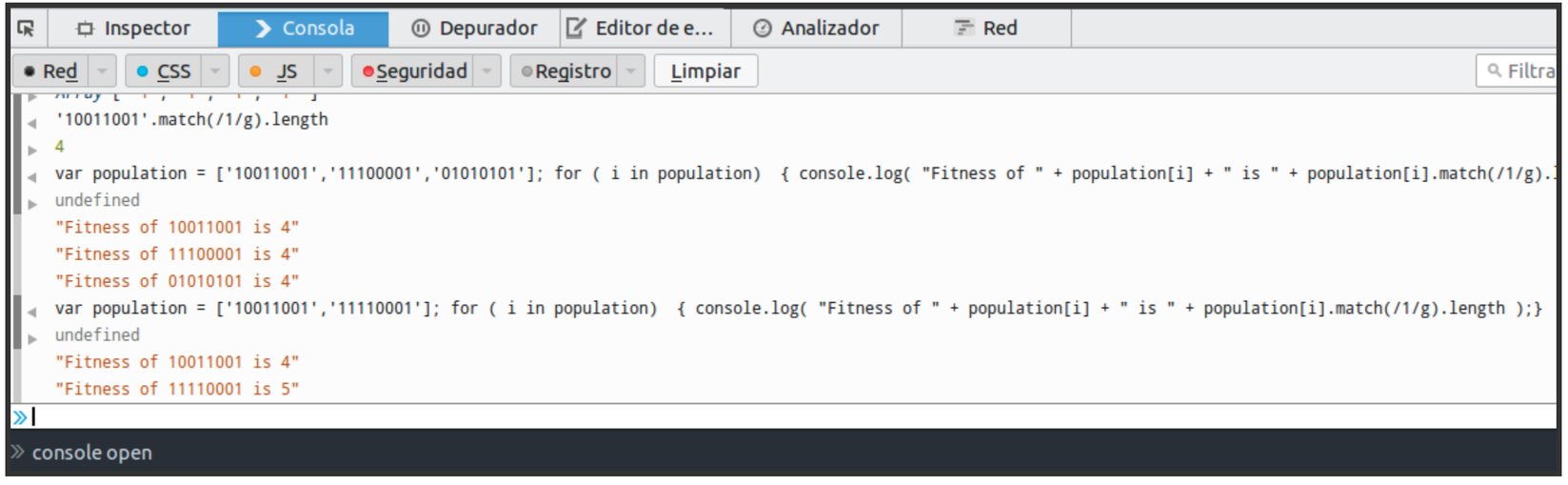
♻ Let Nature be your guide



JavaScript = cloud's native language

Let's do Javascript!

Menu → developer → console



Say hello to these nice folks!

```
console.log('¡Hola, chavales!')
```

Or the much more annoying

```
alert('¿Qué pasa, coleguis?');
```

This is an object. That, too.

```
console.log('Buenos días'.length)
```

Arrays are objects, and the other way round

```
console.log(['Buenos días','Buenas tardes','Buenas noches'].pop())
```

Chromosomes and fitness

```
var chromosome = '1001100110011';
var fitness_of = new Object;
fitness_of[chromosome] = chromosome.match(/1/g).length;
var rr = function (chromosome) {
  var fitness = 0;
  for (var i = 0; i < chromosome.length; i+=4 ) {
        var ones = (chromosome.substr(i, 4).match(/1/g) || []).length;
        fitness += ( ones == 0 || ones == 4 );
  }
  return fitness;
};
```

JavaScript is:

Standard, (reasonably) fast and

Everywhere

Yes, also in your PS4

(Almost) forget about loops

```
function do_ea() {
 eo.generation();
 generation_count++;
 if( (eo.fitness_of[eo.population[0]] < traps*conf.fitness.b )
  && ( generation_count*conf.population_size < conf.max_evaluations)) {
  setTimeout(do_ea, 5);
 } else {
  console.log( "Finished ", log );
 }
}
```

A whole algorithm in a browser

The browser is the new operating system

And why not in the server too?

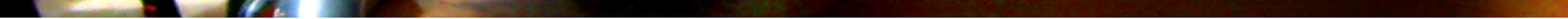
node.js is an asynchronous JS interpreter.

NodEO is an EA library.

```
var eo = new nodeo.Nodeo( { population_size: population_size,
                            chromosome_size: chromosome_size,
                            fitness_func: utils.max_ones } );
do {
    eo.generation();
    console.log( eo.population[0] );
} while ( eo.fitness_of[eo.population[0]] < chromosome_size );
```


Cloud is about reproducible infrastructure

Let's containerize



```
var hiff = new HIFF.HIFF();
var eo = new nodeo.Nodeo( { population_size: conf.population_size,
                            chromosome_size: chromosome_size,
                            fitness_func: hiff } );
logger.info( { start: process.hrtime() } );
```

```
evolve(generation_count, eo, logger, conf, check );
```

A container does one thing

```
if ( typeof process.env.PAPERTRAIL_PORT !== 'undefined'
      && typeof process.env.PAPERTRAIL_HOST !== 'undefined' ) {
    logger.add(winston.transports.Papertrail,
                {
                    host: process.env.PAPERTRAIL_HOST,
                    port: process.env.PAPERTRAIL_PORT
                }
               )
                       }
```

```
var check = function( eo, logger, conf,  generation_count ) {
    if ( (eo.fitness_of[eo.population[0]] < conf.fitness_max )
          && (generation_count*conf.population_size < conf.max_evaluations )) {
        logger.info( { "chromosome": eo.population[0],
                       "fitness" : eo.fitness_of[eo.population[0]]} );
        evolve( generation_count, eo, logger, conf, check);
    } else {
        logger.info( {end: {
            time: process.hrtime(),
            generation: total_generations,
            best : { chromosome : eo.population[0],
                     fitness : eo.fitness_of[eo.population[0]]}}} );
        conf.output = conf.output_preffix+".json";
        process.exit();
    }
};
```

Describe infrastructure: `package.json`

```
{
  "name": "hiffeitor",
  "scripts": {
    "test": "mocha",
    "start": "./callback-ea-HIFF.js"
  },
  "dependencies": {
    "nodeo": "^0.2.1",
    "winston": "^2.2.0",
    "winston-logstash": "^0.2.11",
    "winston-papertrail": "^1.0.2"
  },
  "devDependencies": {
    "flightplan": "^0.6.14"
  }
}
```

Introducing `docker`

Lightweight virtualization

Portable infraestructure

Using docker

```
docker pull jjmerelo/cloudy-ga
```

Containerizing through Dockerfile

```
FROM phusion/baseimage
MAINTAINER JJ Merelo "jjmerelo@gmail.com"
RUN echo "Building a docker environment for NodEO"
ENV DEBIAN_FRONTEND=noninteractive
RUN apt-get update && apt-get upgrade -y
RUN apt-get install apt-utils -y
RUN apt-get install nodejs npm -y
```

```
RUN mkdir app
ADD https://github.com/JJ/cloudy-ga/raw/master/app/callback-ea-HIFF.js app
ADD https://github.com/JJ/cloudy-ga/raw/master/app/package.json app
ADD https://github.com/JJ/cloudy-ga/raw/master/app/hiff.json app
WORKDIR /app
RUN npm i
RUN chmod +x callback-ea-HIFF.js
```

```
CMD npm start
```

Bring your own container

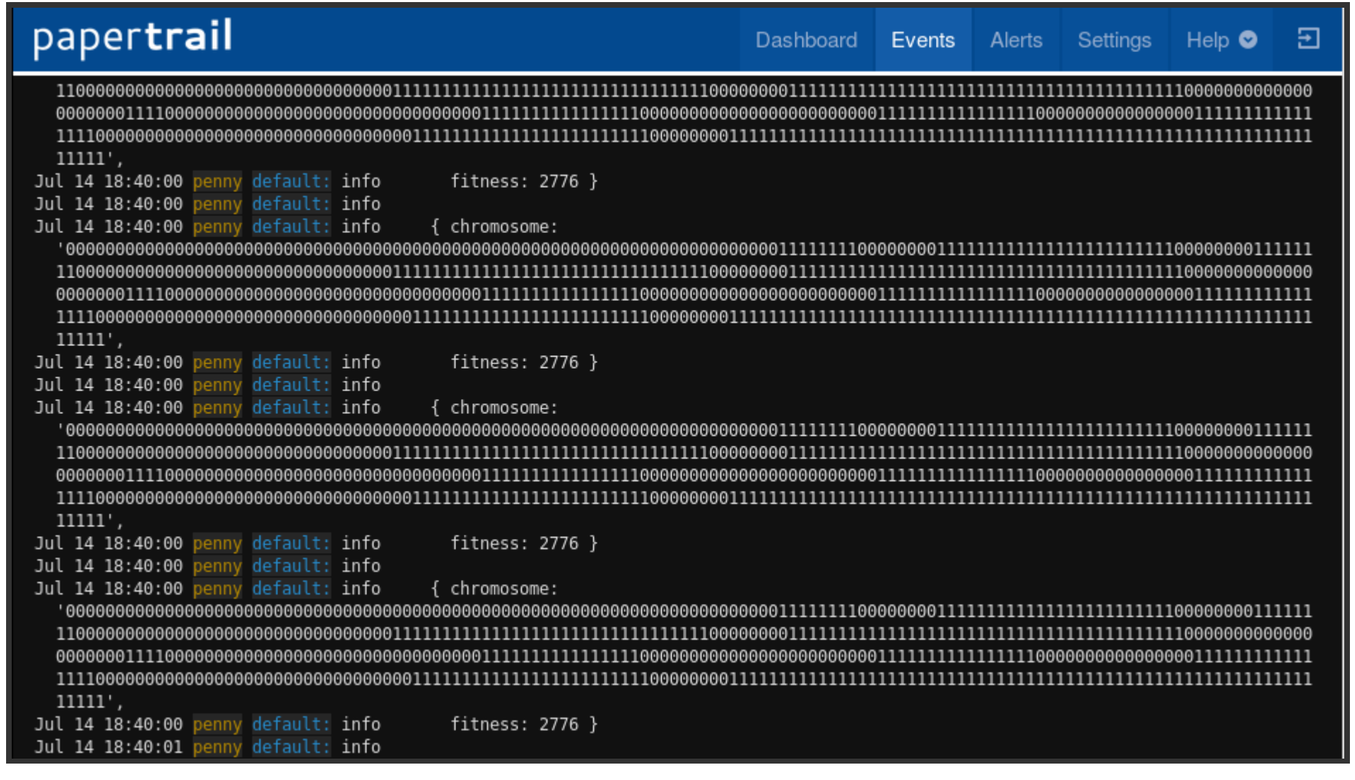
```
90d6565b970a: Pull complete

40553bdb8474: Pull complete

c3129e7479ab: Pull complete

091663bd70db: Pull complete

Digest: sha256:[illegible]ad157293
Status: Downloaded newer image for ubuntu:16.04
 ---> cf62323fa025
Step 2 : MAINTAINER JJ Merelo "jjmerelo@gmail.com"
 ---> Running in 8df5ddf5c497
 ---> 01c86d1709bf
Removing intermediate container 8df5ddf5c497
Step 3 : RUN echo "Building a docker environment for NodEO"
 ---> Running in 39be627556f6
```

```
sudo docker build --no-cache -t jjmerelo/cloudy-ga:0.0.1
```

... and run it

```
sudo docker run -t jjmerelo/cloudy-ga:0.0.1
    -e "PAPERTRAIL_PORT=7777"
    -e "PAPERTRAIL_HOST=logs77.papertrailapp.com"
```

Logging matters


papertrail
Dashboard
Events
Alerts
Settings
Help
Jul 14 18:40:00 penny default: info fitness: 2776 }
Jul 14 18:40:00 penny default: info
Jul 14 18:40:00 penny default: info { chromosome:
Jul 14 18:40:00 penny default: info fitness: 2776 }
Jul 14 18:40:00 penny default: info
Jul 14 18:40:00 penny default: info { chromosome:
Jul 14 18:40:00 penny default: info fitness: 2776 }
Jul 14 18:40:00 penny default: info
Jul 14 18:40:00 penny default: info { chromosome:
Jul 14 18:40:00 penny default: info fitness: 2776 }
Jul 14 18:40:01 penny default: info

Use CoreOS

Ready to run on Azure or anywhere

It's not programming as usual

Reactive programming

Algorithm + stream = application in the cloud

Decoupled processing and data structures

Before

```
do {
    eo.generation();
} while ( eo.fitness_of[eo.population[0]] < chromosome_size );
```

After decoupling

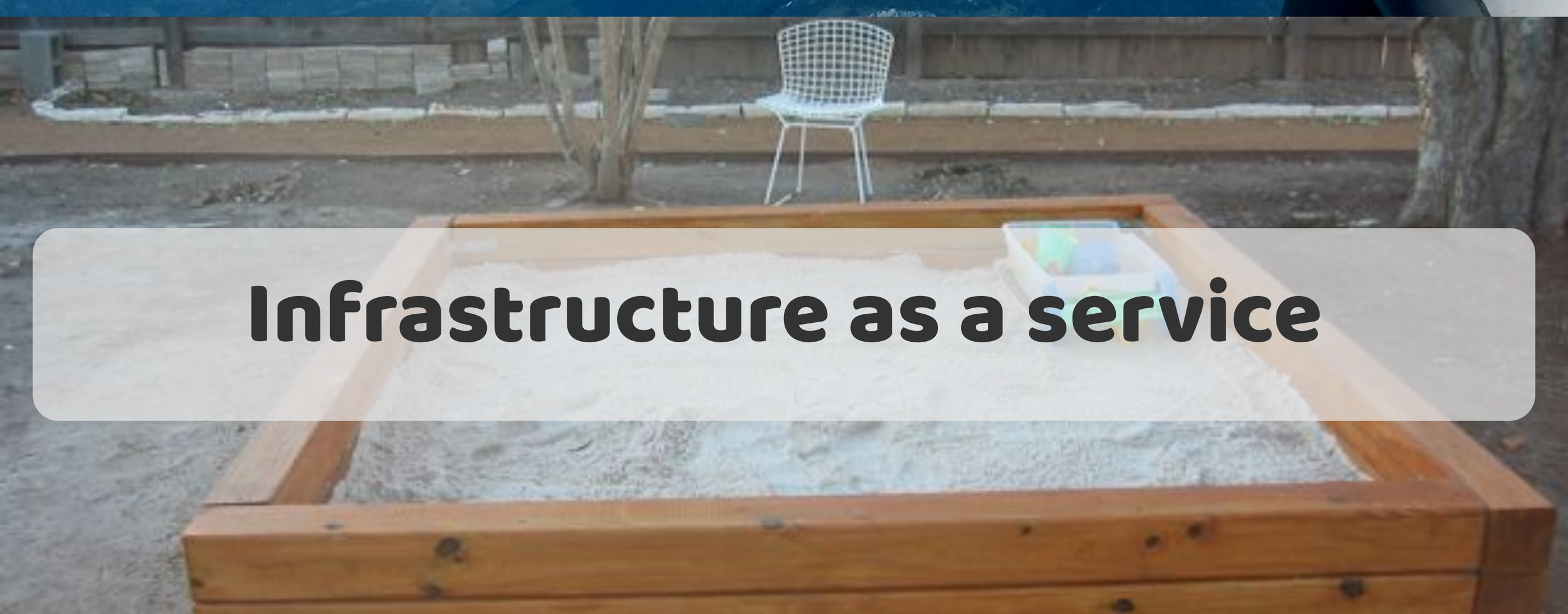
```
var random_chromosome = function() {
    return utils.random( chromosome_size );
};
var population = new Population();
population.initialize( population_size, random_chromosome);
```

```
var eo = new fluxeo( this_fitness,
                     new Tournament( tournament_size,
                        population_size-2 ),
                     check);
```

Algorithm on population

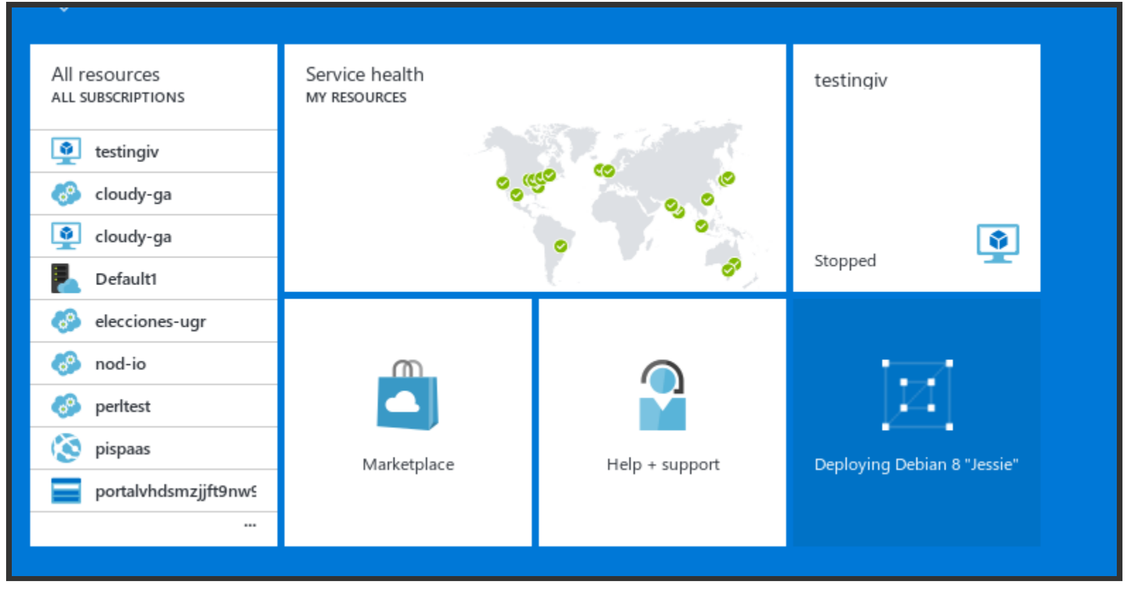
```
eo.algorithm( population, function ( population ) {
    logger.info( {
        end: { time: process.hrtime(),
            generation: total_generations,
            best : { chromosome : population.best,
                fitness : population.fitness(population.best)  }
        }
    });
});
```

Running in the cloud



Infrastructure as a service

Create instance

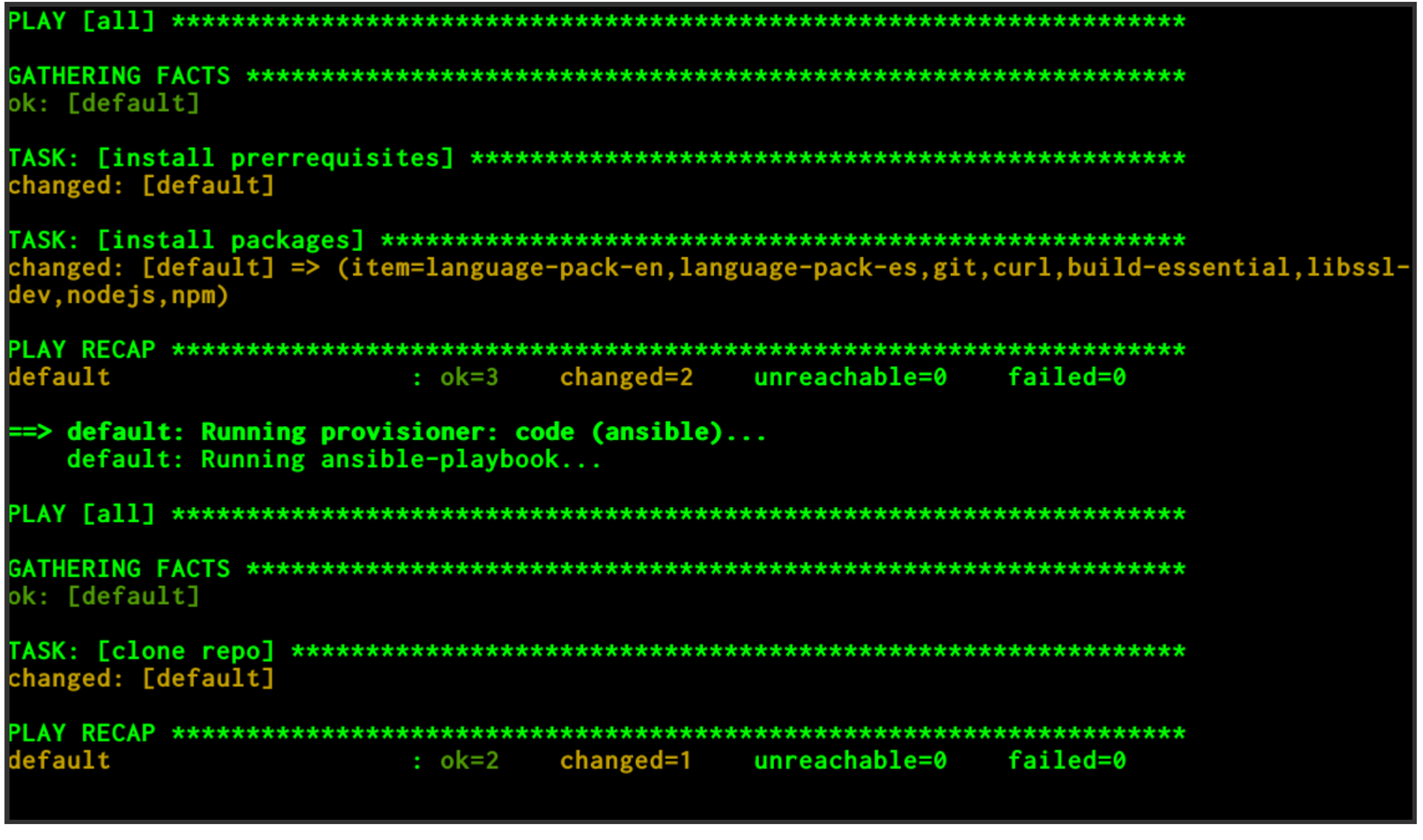


Set up with Ansible

```
- hosts: "{{target}}"
  tasks:
    - name: install prerrequisites
      command: apt-get update -y && apt-get upgrade -y
    - name: install packages
      apt: pkg={{ item}}
      with_items:
        - git
        - npm
    - name: Create profile
copy: content="export PAPERTRAIL_PORT={{PAPERTRAIL_PORT}}}"
      dest=/home/cloudy/.profile
```

Run the playbook

```
ansible-playbook git.playbook.yml
        -e "target=azuredeb"
        -u ubuntu
        -i ./hosts.txt -vvvv
```


PLAY [all] **
GATHERING FACTS ***
ok: [default]
TASK: [install prerrequisites] **
changed: [default]
TASK: [install packages] **
changed: [default] => (item=language-pack-en,language-pack-es,git,curl,build-essential,libssl-dev,nodejs,npm)
PLAY RECAP **
default : ok=3 changed=2 unreachable=0 failed=0
==> default: Running provisioner: code (ansible)...
default: Running ansible-playbook...
PLAY [all] **
GATHERING FACTS ***
ok: [default]
TASK: [clone repo] **
changed: [default]
PLAY RECAP **
default : ok=2 changed=1 unreachable=0 failed=0

Ready to run ✓


cloudy-ga.cloudapp.net > info: chromosome=00000000000000001111111111111111111110
00000000000000111100000000111111111111111110100101011011001111111111111111111111
11111111111000010111111111111111111111111100000000000000010001010000001000000
00000000000000000000000000000000000000000001100011011010001, fitness=1086
cloudy-ga.cloudapp.net > info: chromosome=00000000000000001111111111111111111110
00000000000000111100000000111111111111111110100101011011001111111111111111111111
11111111111000010111111111111111111111111100000000000000010001010000001000000
00000000000000000000000000000000000000000001100011011010001, fitness=1086
cloudy-ga.cloudapp.net > info: chromosome=00000000000000001111111111111111111001
10000001101111100001111000011111111111111111001011111110011011111111111111111111
11111111111011011111111111111111111111111100000000000000011000000101000100000
00000000000000000000000000000000000000000001111100110001111, fitness=1086
cloudy-ga.cloudapp.net > info: chromosome=00000000000000001111111111111111111110
00000000011000001100110000011111111111111111010010100101100111111111111111111111
11111111111000010111111111111111111111111100000000000000011111000000000100000
00000000000000000000000000000000000000000001111111100001111, fitness=1104
cloudy-ga.cloudapp.net > info: chromosome=00000000000000001111111111111111111110
00000000011000001100110000011111111111111111010010100101100111111111111111111111
11111111111000010111111111111111111111111100000000000000011111000000000100000
00000000000000000000000000000000000000000001111111100001111, fitness=1104

But there's something missing here

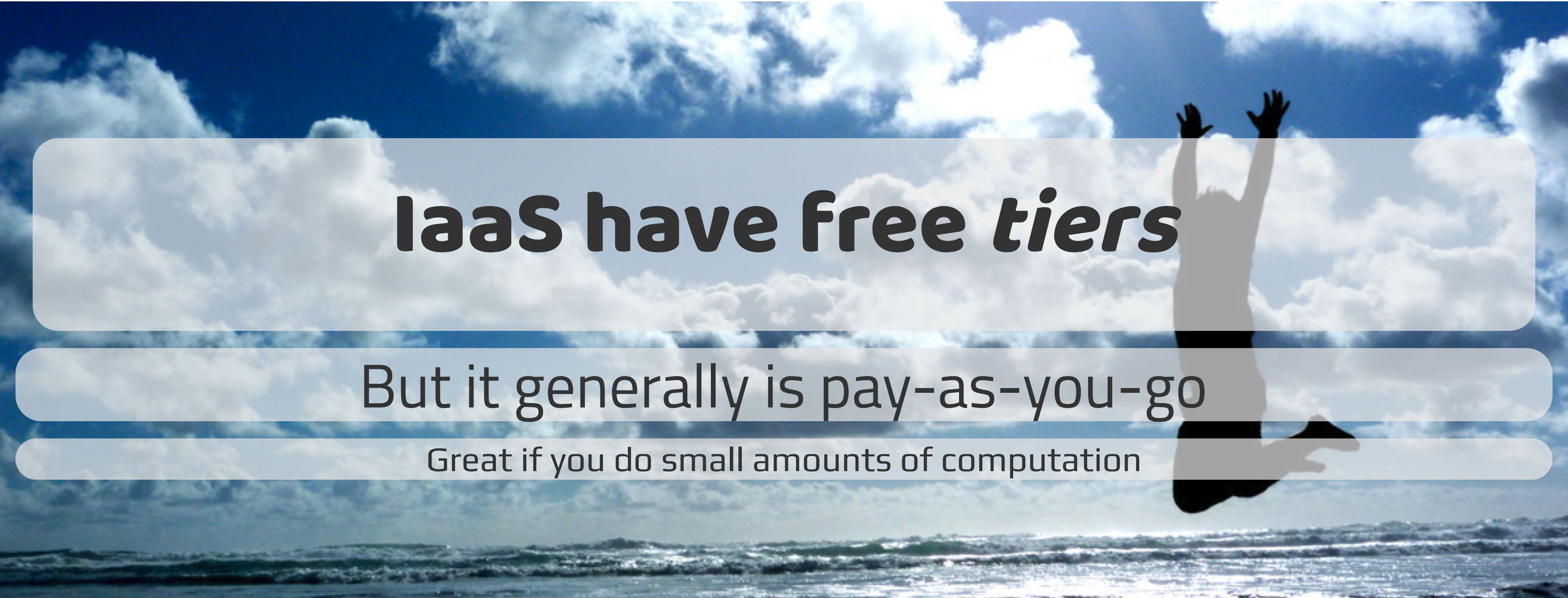
Deploying to the cloud

Let's use `FlightPlan`

```
plan.target('azure', {
  host: 'cloudy-ga.cloudapp.net',
  username: 'azureuser',
  agent: process.env.SSH_AUTH_SOCK
});
// Local
plan.local(function(local) {
    local.echo('Plan local: push changes');
    local.exec('git push');
});
```

… And after setup

```
plan.remote(function(remote) {
    remote.log('Pull');
    remote.with('cd cloudy-ga',function() {
        remote.exec('git pull');
        remote.exec('cd app;npm install .');
    });
    remote.with('cd /home/azureuser/cloudy-ga/app',function() {
        remote.exec('npm start');
    });
});
```



IaaS have free *tiers*

But it generally is pay-as-you-go

Great if you do small amounts of computation

You might not need a whole server

Platform as a service

Browsers communicate using HTTP commands

PUT, GET, POST, DELETE

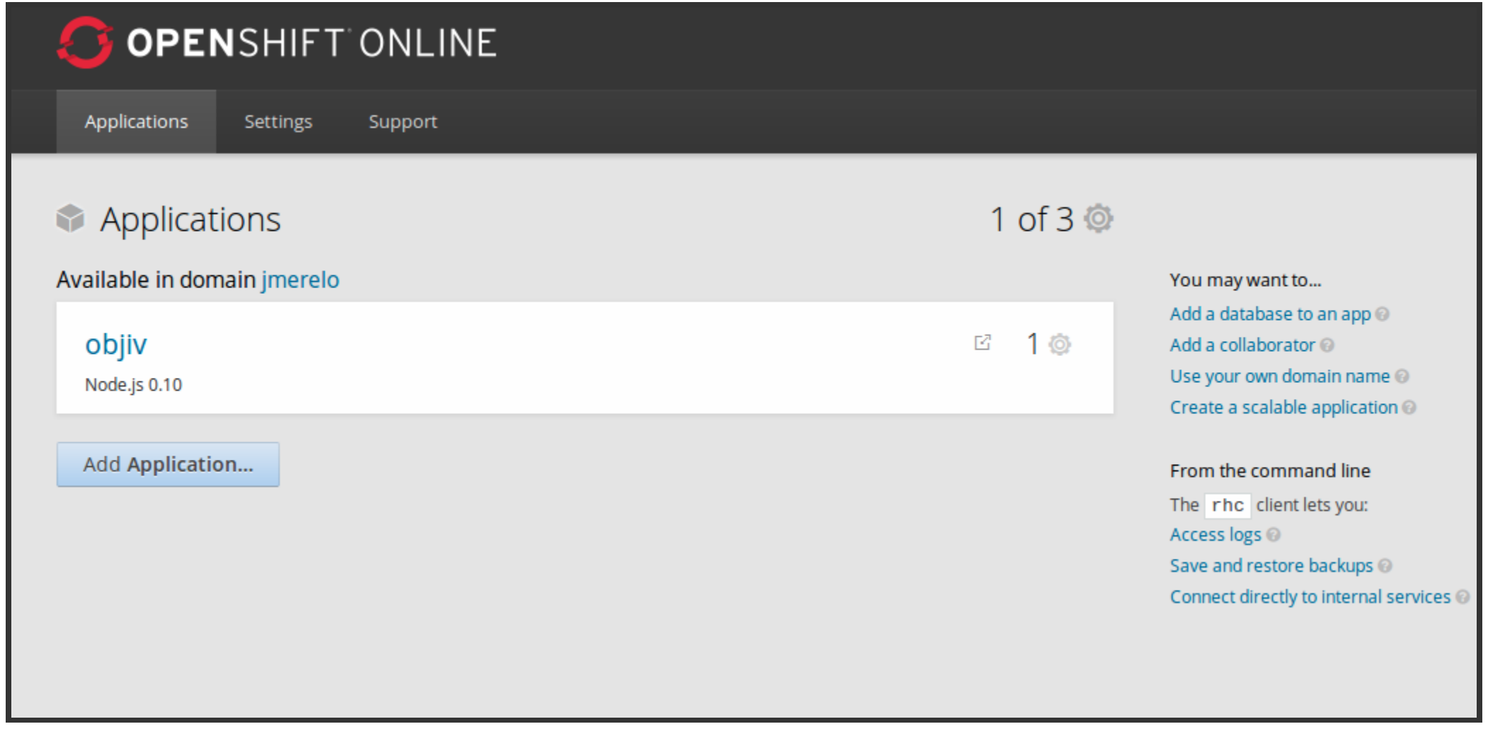
Ajax, a standard browser-server communication framework

HTTP petitions from a standard object.

Asynchronous!

There's *freemium* PaaS

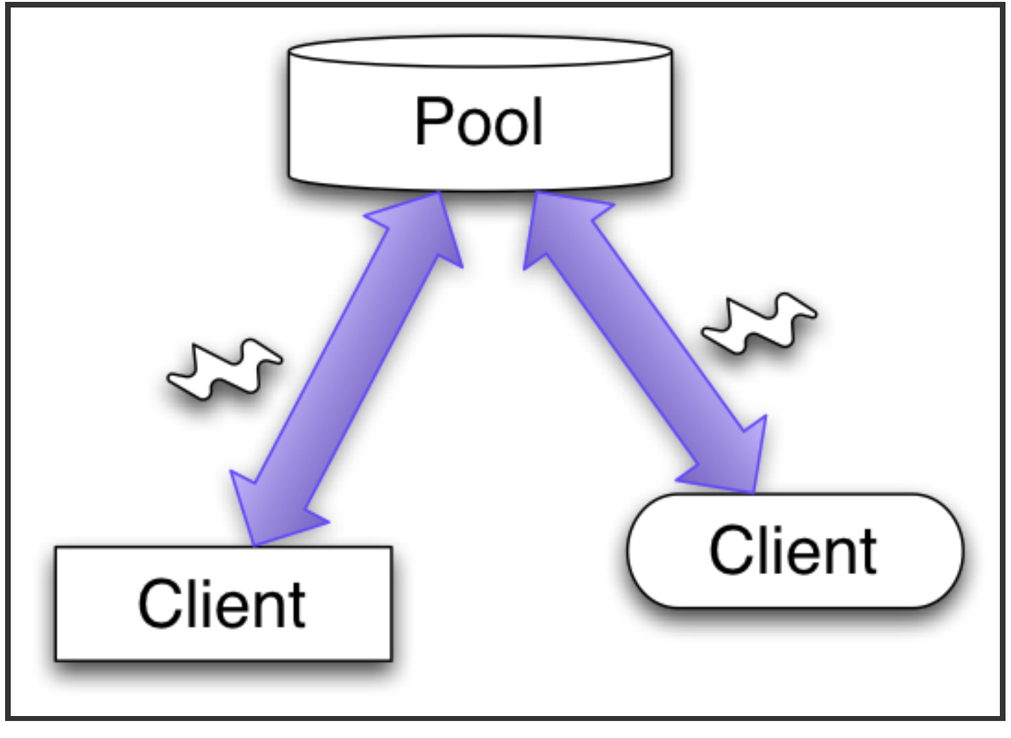
Heroku, OpenShift and IBM BlueMix or Azure Web Apps


OPENSHIFT ONLINE
Applications
Settings
Support
Applications
1 of 3
Available in domain jmerelo
objiv
Node.js 0.10
1
Add Application...
You may want to...
Add a database to an app
Add a collaborator
Use your own domain name
Create a scalable application
From the command line
The rhc client lets you:
Access logs
Save and restore backups
Connect directly to internal services

Pool-based evolutionary algorithms: not so canonical any more

Detaching population from operations

➡ Reactive programming.

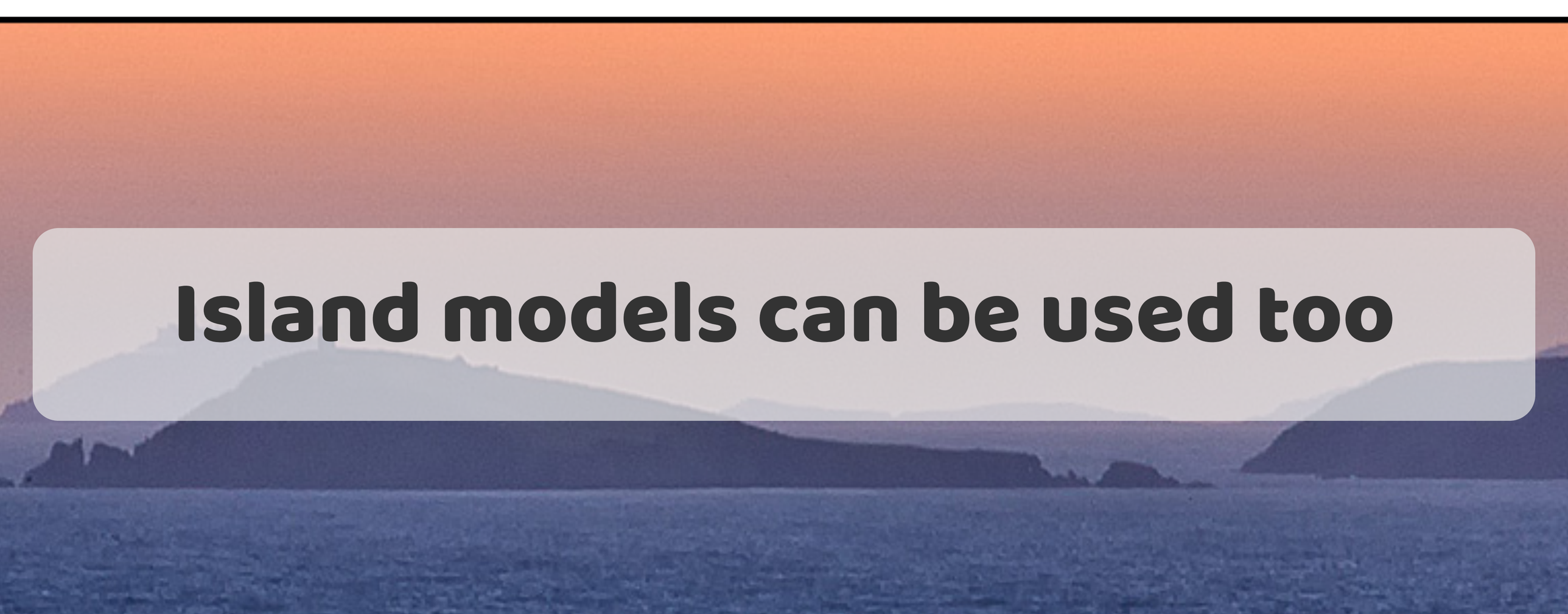

Pool
Client
Client

Three good things about pool-based EAs

1. Self-organizing clients

2. Fully asynchronous

3. Persistent population


Island models can be used too

The cloudy server

```
app.put('/experiment/:expid/one/:chromosome/:fitness/:uuid',
  function(req, res){
        // stuff here
        logger.info("put", { chromosome: req.params.chromosome,
                             fitness: parseInt(req.params.fitness),
                             IP: client_ip,
                             worker_uuid:req.params.uuid} );
        res.send( { length : Object.keys(chromosomes).length });
    }
```

```
app.get('/random', function(req, res){
        var keys = Object.keys(chromosomes );
        var one = keys[ Math.floor(keys.length*Math.random())];
        res.send( { 'chromosome': one } );
        logger.info('get');
});
```

Check out

✓ Asynchronous

✓ Uses Logger

Changes in the client: draw from pool

```
rest.get( conf.url + 'random' ).on('complete', function( data ) {
                if ( data.chromosome ) {
                    population.addAsLast( data.chromosome );
                }
});
```

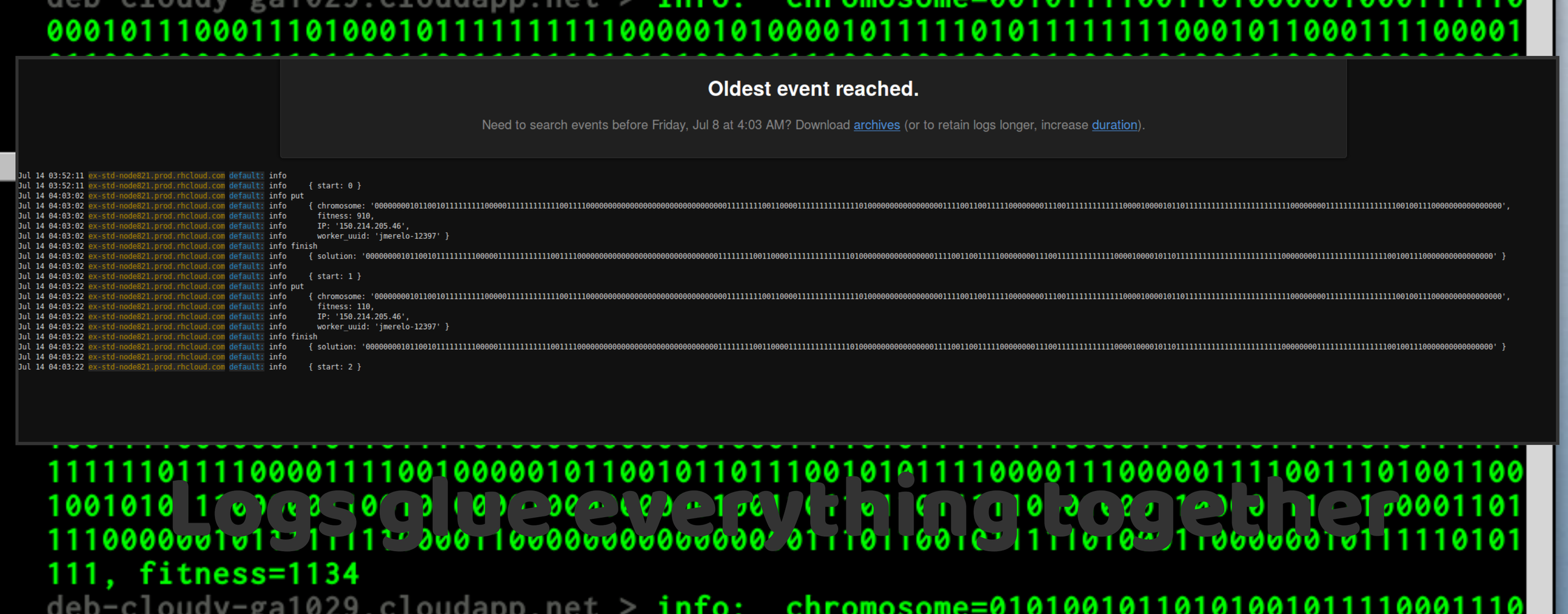
Put into pool

```
var this_request = conf.url
                + 'experiment/0/one/' + population.best() + "/"
                + population.fitness(population.best()) + "/"
                + UUID;
            rest.put( this_request ).on("complete", function( result, response ) {
                if ( response.statusCode == 410 ) {
                    finished = true;
                    experiment_id = result.current_id;
                }
});
```

Deploy server to PaaS ✓

Deploy clients to IaaS ✓

➡ And run!


Oldest event reached.
Need to search events before Friday, Jul 8 at 4:03 AM? Download archives (or to retain logs longer, increase duration).
Jul 14 03:52:11 ex-std-node821.prod.rhcloud.com default: info
Jul 14 03:52:11 ex-std-node821.prod.rhcloud.com default: info { start: 0 }
Jul 14 04:03:02 ex-std-node821.prod.rhcloud.com default: info put
Jul 14 04:03:02 ex-std-node821.prod.rhcloud.com default: info { chromosome: '0000000010110010111111111000001111111111110011100000000000000000000000000000011111111001100001111111111111010000000000000000111001100111110000000011100111111111111000010000101101111111111111111111111000000001111111111111110010011100000000000000',
Jul 14 04:03:02 ex-std-node821.prod.rhcloud.com default: info fitness: 910,
Jul 14 04:03:02 ex-std-node821.prod.rhcloud.com default: info IP: '150.214.205.46',
Jul 14 04:03:02 ex-std-node821.prod.rhcloud.com default: info worker_uuid: 'jmerelo-12397' }
Jul 14 04:03:02 ex-std-node821.prod.rhcloud.com default: info finish
Jul 14 04:03:02 ex-std-node821.prod.rhcloud.com default: info { solution: '0000000010110010111111111000001111111111110011100000000000000000000000000000011111111001100001111111111111010000000000000000111001100111110000000011100111111111111000010000101101111111111111111111111000000001111111111111110010011100000000000000' }
Jul 14 04:03:02 ex-std-node821.prod.rhcloud.com default: info
Jul 14 04:03:02 ex-std-node821.prod.rhcloud.com default: info { start: 1 }
Jul 14 04:03:22 ex-std-node821.prod.rhcloud.com default: info put
Jul 14 04:03:22 ex-std-node821.prod.rhcloud.com default: info { chromosome: '0000000010110010111111111000001111111111110011100000000000000000000000000000011111111001100001111111111111010000000000000000111001100111110000000011100111111111111000010000101101111111111111111111111000000001111111111111110010011100000000000000',
Jul 14 04:03:22 ex-std-node821.prod.rhcloud.com default: info fitness: 110,
Jul 14 04:03:22 ex-std-node821.prod.rhcloud.com default: info IP: '150.214.205.46',
Jul 14 04:03:22 ex-std-node821.prod.rhcloud.com default: info worker_uuid: 'jmerelo-12397' }
Jul 14 04:03:22 ex-std-node821.prod.rhcloud.com default: info finish
Jul 14 04:03:22 ex-std-node821.prod.rhcloud.com default: info { solution: '0000000010110010111111111000001111111111110011100000000000000000000000000000011111111001100001111111111111010000000000000000111001100111110000000011100111111111111000010000101101111111111111111111111000000001111111111111110010011100000000000000' }
Jul 14 04:03:22 ex-std-node821.prod.rhcloud.com default: info
Jul 14 04:03:22 ex-std-node821.prod.rhcloud.com default: info { start: 2 }
Logs glue everything together
111, fitness=1134

Vagrant for orchestration

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/xenial64"
  config.vm.provision "shell", inline: <<-SHELL
     apt-get update
     apt-get upgrade -y
  SHELL
  config.vm.provision "main", type: "ansible" do |ansible|
    ansible.extra_vars = { target: "all" }
    ansible.playbook = "playbook.yml"
  end
  # and the rest...
end
```

All together

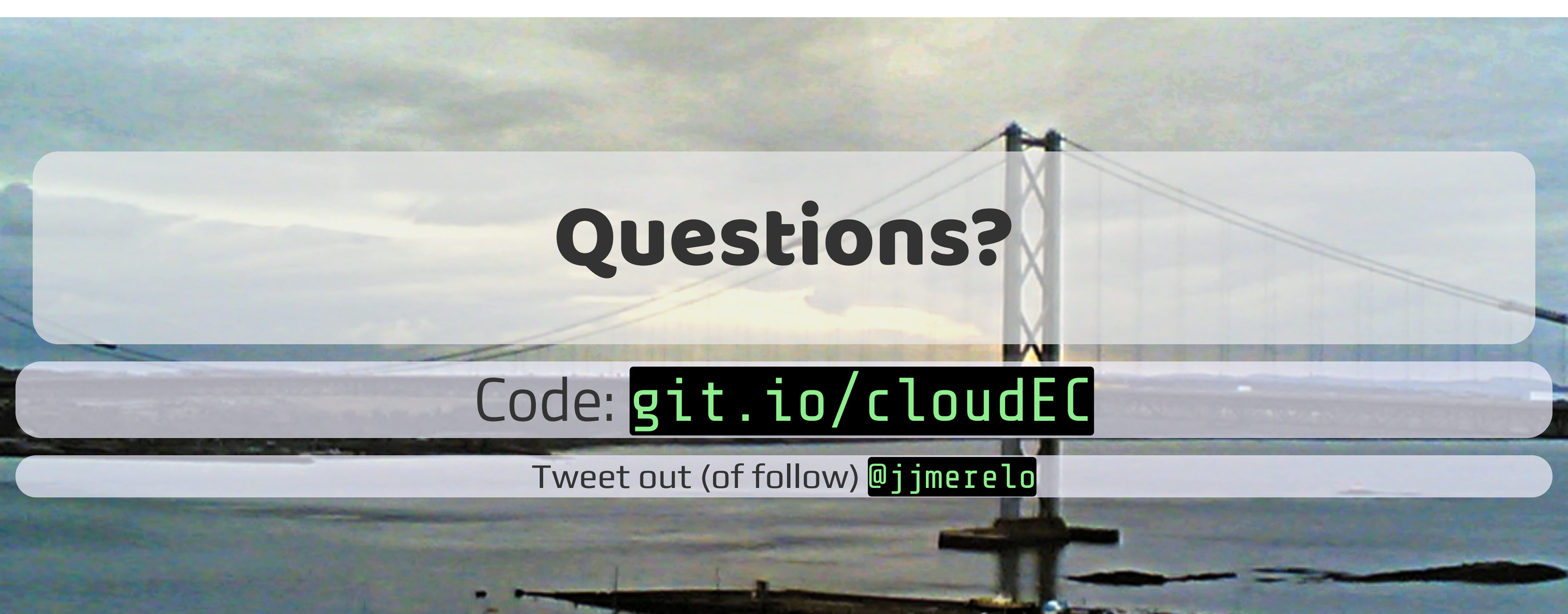
✓ Get servers ➡ PaaS, Loggers

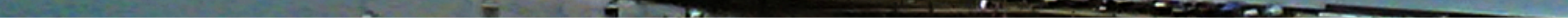
✓ Create/provision boxes ➡ Vagrant/Ansible

✓ Deploy/run ➡ FlightPlan

Take this home

1. Cloud is the new (grid|cluster)
2. There is (almost) free lunch
3. Reactive programming
4. We should ❤ logs


Questions?
Code: git.io/cloudEC
Tweet out (of follow) @jjmerelo



Credits

- Faucet on the sky by Kristin Nador
- Strip mall by Thomas Hawk
- Mobiles, Flickr image by Kai Hendry flic.kr/p/5omH5r
- People in room from Adriaan Bloem flic.kr/p/8NDCRh
- Shooting at clouds by Charles Prithvi Raj
- Sandbox, Flickr image by Gil Garcia flic.kr/p/5Pa7ZJ

B&W sandbox, Flickr image by Stefani Woods flic.kr/p/4QiGQV
HTTP codes, Image by Paul Downey flic.kr/p/8yZyBF
Data center in cave, from Antony Antony flic.kr/p/6K3ZAG
Nuns and pool, Flickr image by Lorianne diSabato @flic.kr/p/nXfbQ6
3 fingers, Flickr image by Mutiara Karina https://flic.kr/p/a9S5m3.
Freedom by Gonzalo Baeza

URL: goo.gl/KYrOVr