



+ Authentication Authorization Architecture

in Browser Applications

Yuri Takhteyev, rangle.io
<http://yto.io>
[@qaramazov](https://twitter.com/qaramazov)



Authenticate,
Authorize

Protecting clients from:

outsiders

each other

themselves



Client

Display and capture

Interaction logic

Server

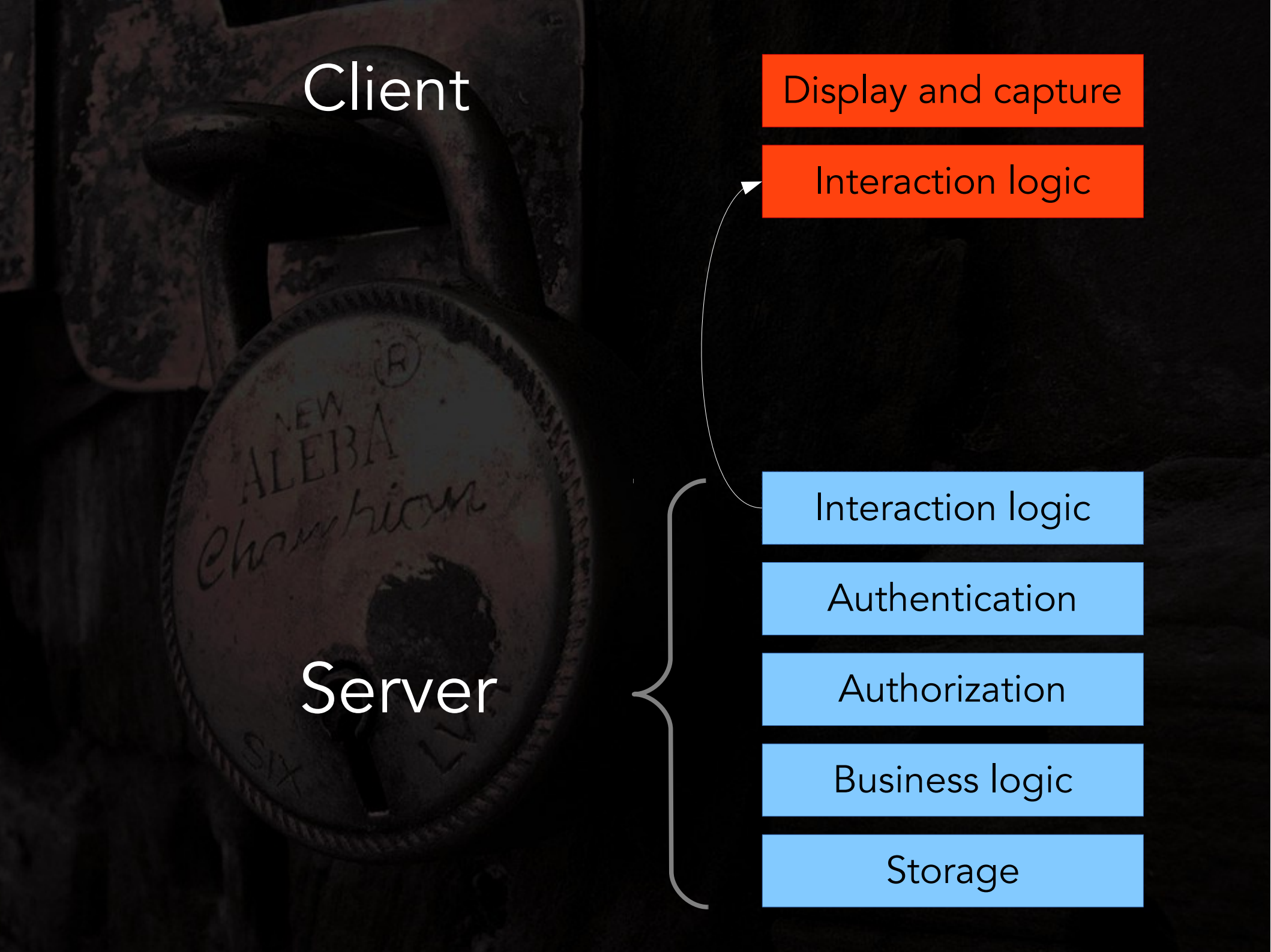
Interaction logic

Authentication

Authorization

Business logic

Storage



Display and capture

The diagram features a background image of a large, dark, weathered padlock. The padlock has a circular face with some faint markings, including the word 'NEW' and a registered trademark symbol. A metal shackle is attached to the top. Overlaid on the right side of the padlock is a vertical stack of eight colored rectangular boxes. From top to bottom, the colors are orange, orange, orange, orange, light blue, light blue, light blue, and light blue. The first four orange boxes are connected by a curved arrow pointing downwards. The last four light blue boxes are also connected by a curved arrow pointing downwards. Two curved arrows point from the light blue 'Authentication' and 'Authorization' boxes to the yellow 'Authentication' and 'Authorization' boxes on the left. The yellow boxes are positioned over the circular face of the padlock.

Interaction logic

Business logic

Authorization

Authentication

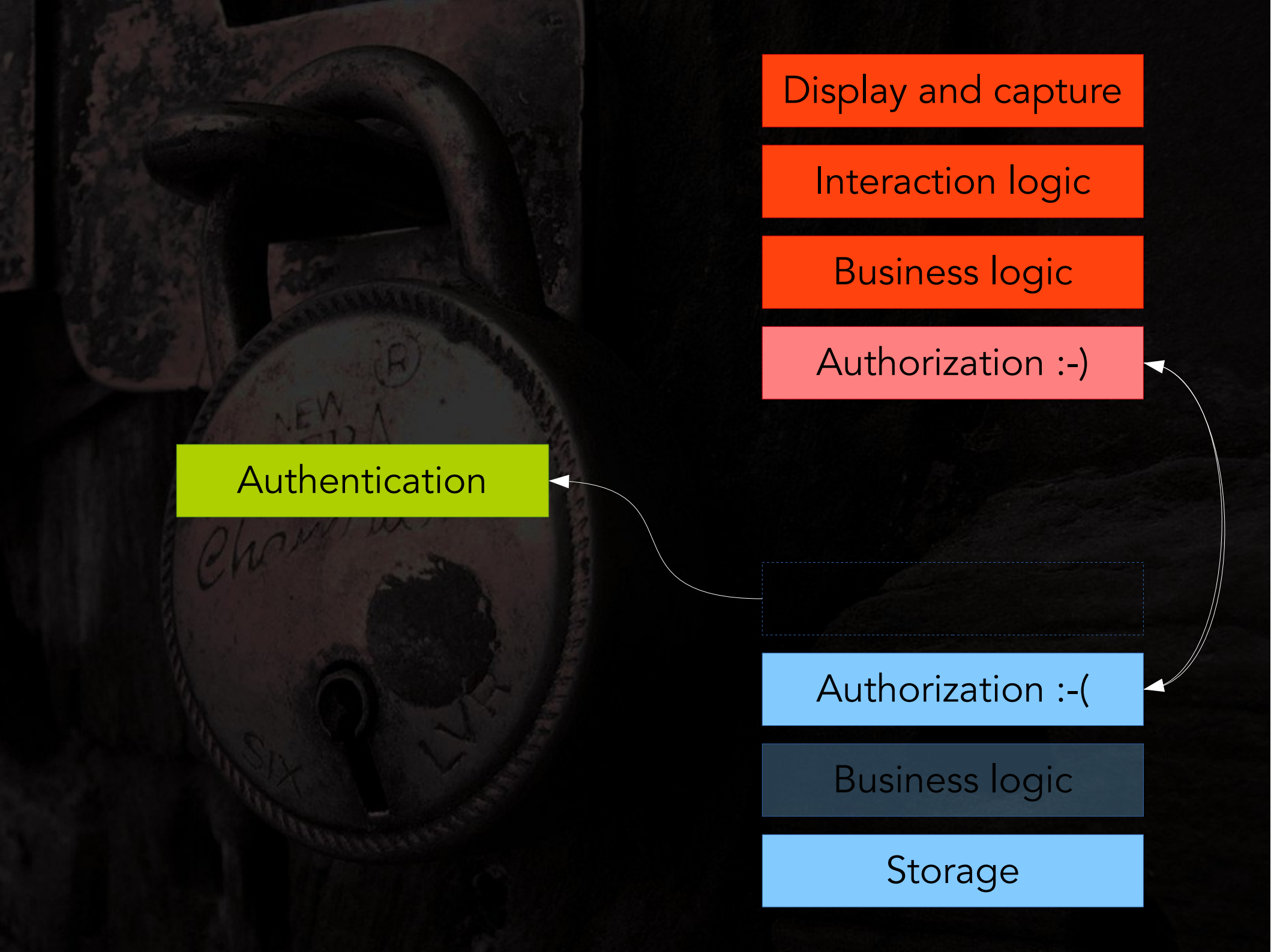
Authorization

Authentication

Authorization

Business logic

Storage



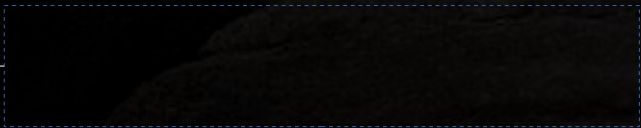
Display and capture

Interaction logic

Business logic

Authorization :-)

Authentication

Authorization :-(


Business logic

Storage



Who am I?

Cookies
vs tokens

Maintaining
the state



Passport

Local

Social / OAuth

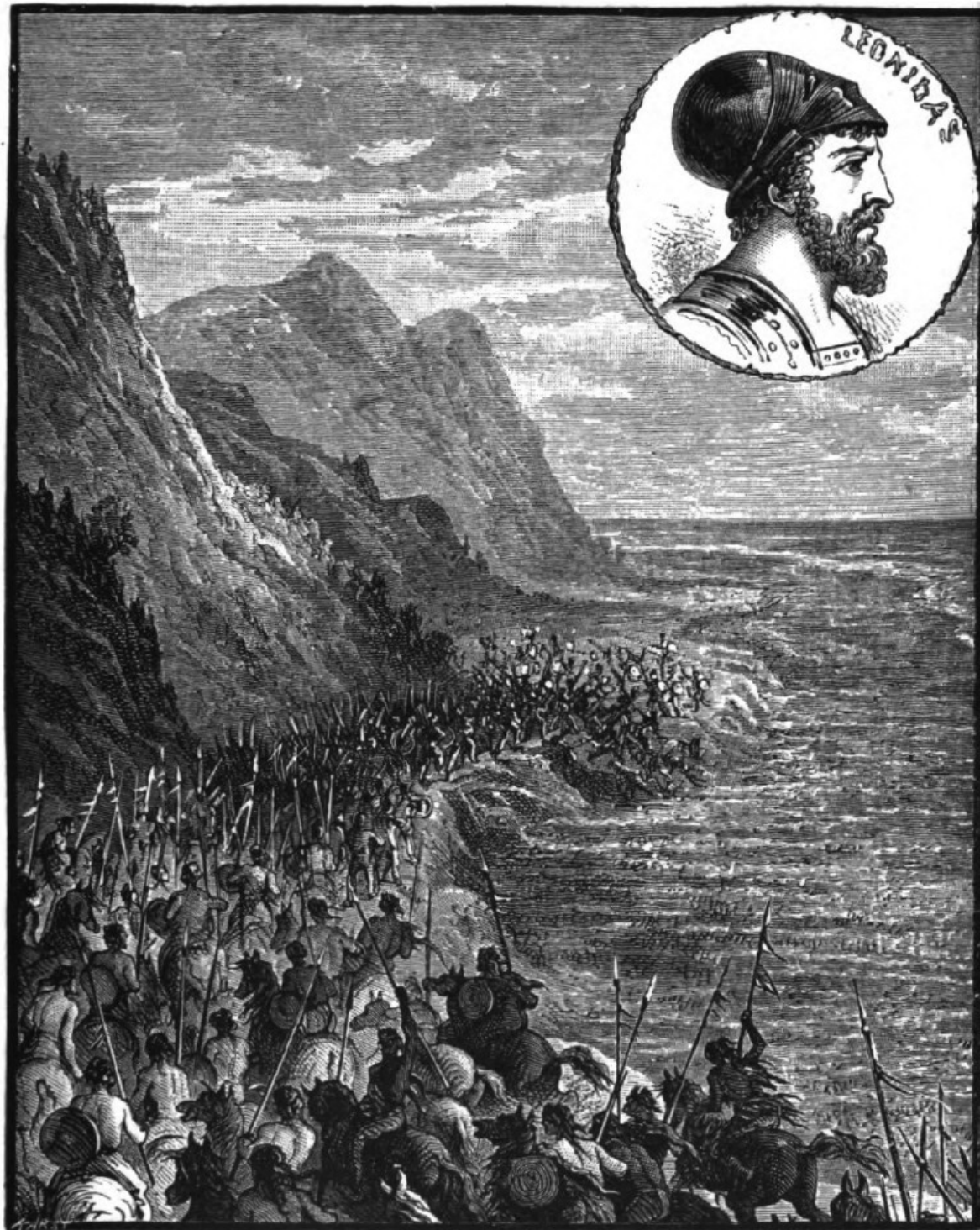
Many Faces of Authorization

By role or
activity

By resource
instance

Altogether
custom





Bottlenecks



This repository ▾

Search or type a command



[Explore](#) [Gist](#) [Blog](#) [Help](#)

yuri + ✂ 📄

PUBLIC



rangle / **koast**

Unwatch ▾

4

★ Star

6

Fork

0

A thin server with an attitude — [Edit](#)

29 commits

3 branches

3 releases

1 contributor



branch: master ▾

koast / +

Updating READMEs.

yuri authored a minute ago

latest commit 9baedeadcc 📄

client	Updating READMEs.	a minute ago
docs	Updating heroku docs.	a month ago
examples	Updating READMEs.	a minute ago
server	Updating READMEs.	a minute ago
.bowerrc	Adding support for persona authentication.	24 days ago
.gitignore	Switching to OAuth authentication via passport.	16 days ago
Gruntfile.js	JSHinting and beautifying	2 months ago
LICENSE.txt	Updating the READMEs.	a month ago
README.md	Updating READMEs.	a minute ago
bower.json	Switching to OAuth authentication via passport.	16 days ago

README.md

<> Code

Issues

0

Pull Requests

0

Wiki

Pulse

Graphs

Network

Settings

SSH clone URL

git@github.com:ranç 📄

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#). ⓘ

Clone in Desktop

Download ZIP

Koast provides a base for a backend server to support an AngularJS app. (It may be useful for Javascript apps built with other frameworks as well, but our focus is on Angular.) The goal is to quickly

Setting Up an API Route

```
var mapper = koast.makeMongoMapper(connection);

routes = [
  ['get', 'robots', user.any,
   mapper.get('robots', [])],
  ['put', 'robots/:robotNumber', user.any,
   mapper.put('robots', ['robotNumber'])],
];
```

Restricting Access by Object

```
mapper.queryDecorator = function(query, req,  
    res) {  
    query.owner = req.user.username;  
};
```

```
mapper.queryDecorator = function(query, req,  
    res) {  
    query.clientId = req.user.clientId;  
};
```

Post-Query Filtering

```
mapper.filter = function(result, req) {  
  if (req.method === 'GET' {  
    return canSee(data, req);  
  } else {  
    return canEdit(data, req);  
  }  
};
```

Informing the Client

```
mapper.clientAuthorizer = function(result, req) {  
    result.meta.can.edit =  
        canEdit(result.data, req);  
};
```

The Client Side

```
angular.module('sampleKoastClientApp',  
  ['koast'])  
  
  .controller('myCtrl', ['$scope', 'koast',  
    '$log',  
    function($scope, koast, $log) {  
      $scope.login = function(provider) {  
        koast.user.initiateOAuthAuthentication(  
          provider);  
      };  
      ...  
    }  
  ]  
);
```

The Client Side

...

```
koast.user.getStatusPromise()  
  .then(function() {  
    if (koast.user.isAuthenticated) {  
      return koast.getResource('robots')  
        .then(function(robots) {  
          $scope.robots = robots;  
        });  
    }  
  })  
  .then(null, $log.error);  
});
```

The Template

```
<span ng-if="robot.can.edit">  
  <button>...</button>  
</span>
```



Thank You.

Contact:

yuri@rangle.io

<http://yto.io>

@qaramazov

This presentation:

<http://yto.io/auth>

Koast:

<http://yto.io/koast>



by psd



by nikhilverma

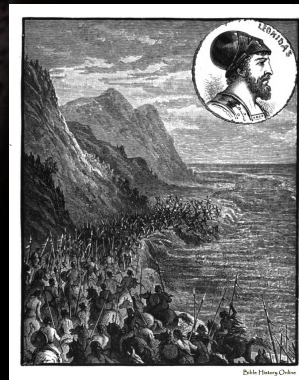
Image Credits



by dunechaser



by yewenyi



by bible-history



by striatic

Distributed under
Creative Commons Attribution
Share-Alike License, v. 2.0