Lift

ANGULARJS by Google

Lift-ng: Secure, rapid web development with Scala and AngularJS

Follow along! http://10.1.4.30:8080

Scala

Joe Barnes
@joescii
prose :: and :: conz

Lift-ng: Secure, rapid web development
with Scala and AngularJS

Follow along! http://10.1.4.30:8080

Joe Barnes

@joescii

prose :: and :: conz

Primarily Java from 2004-2013
Started Scala late 2012
Discovered lift-ng in Oct 2013
Writing Scala at Mentor Graphics
Lift committer July 2014

Built upon Lift, lift-ng is the most powerful, most secure AngularJS backend available today.

backend available today.

Pretty bold, huh?

Description stolen from Lift:
"Lift is the most powerful, most secure web framework available today."

So what *is* Lift?

MENTION LIFTWEB
AND EVERYONE LOSES THEIR MINDS
memegenerator.net

Lift is a web framework

Security: Safeguards from 6 OWASP vulnerabilities

Designer-friendly: View-first w/ HTML templates

Outstanding comet and ajax support

Scala's oldest web framework (Feb 2007)

Doesn't hide web development from you

Apache License 2.0

So what is Angular?

SIDE-EFFECTS
SIDE-EFFECTS, EVERYWHERE!
memegenerator.net

Angular is a front-end framework

Extends HTML for dynamic web apps

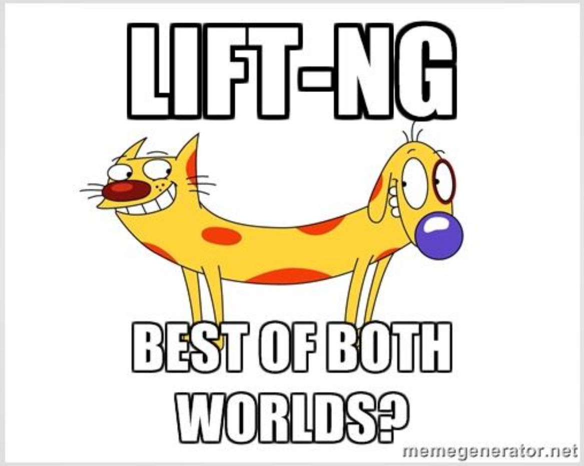
Declarative DOM manipulation

Test-first

Mixes well with Lift

MIT License

So then what is lift-ng?


LIFT-NG
BEST OF BOTH WORLDS?
memegenerator.net

Lift-ng is a Lift module

Provides a Scala DSL for defining:

1. Angular factories
2. `$scope` events and assignments
3. Client/server value bindings (experimental)

Apache License 2.0

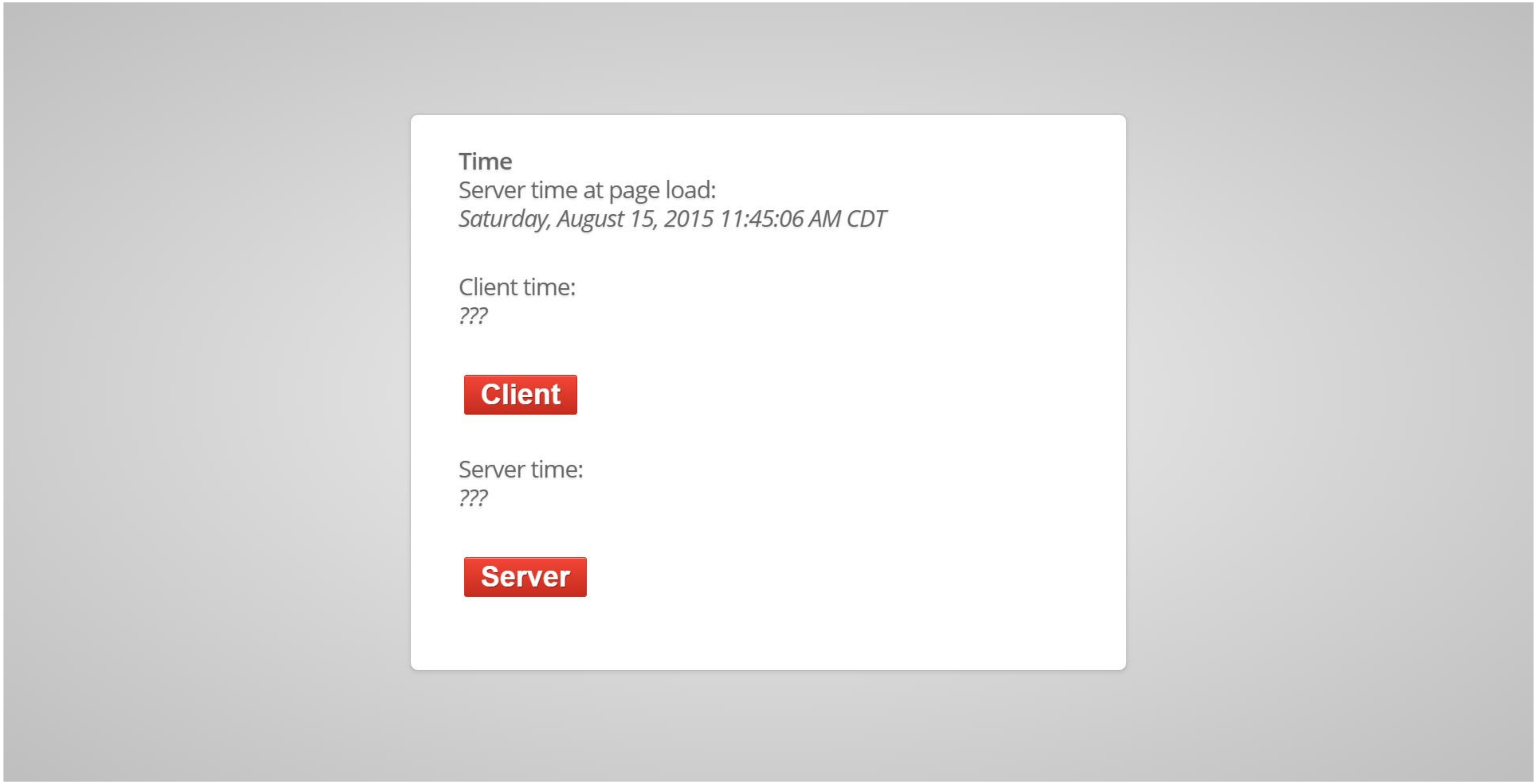
Server Time App

Lift template basics

Angular template basics

JS Angular factory

lift-ng Angular factory



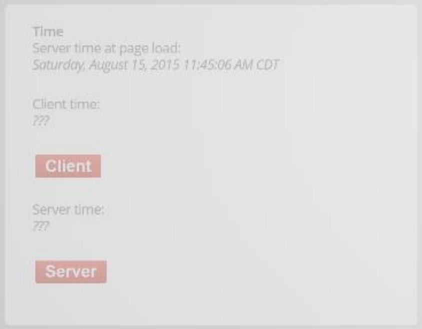
Time
Server time at page load:
Saturday, August 15, 2015 11:45:06 AM CDT

Client time:
???

Server time:
???

```
<div ng-controller="TimeController">
  <div class="title">Time</div>
  <div>Server time at page load:</div>
  <div data-lift="ServerTime.atPageLoad">
    (this div replaced at page load time)
  </div>
  <div>Client time:</div>
  <div class="timestamp" ng-bind="client"></div>
  <button ng-click="getClient()">Client</button>
  <div>Server time:</div>
  <div class="timestamp" ng-bind="server"></div>
  <button ng-click="getServer()">Server</button>
</div>
```

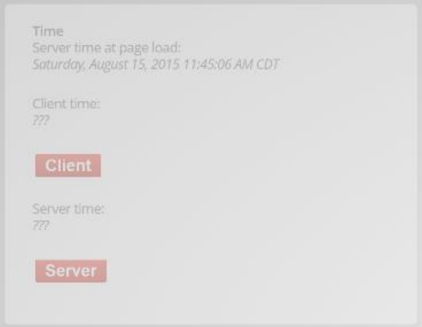
Server Time App
Lift template basics
Angular template basics
JS Angular factory
lift-ng Angular factory


Time
Server time at page load:
Saturday, August 15, 2015 11:45:06 AM CDT

Client time:
???

Server time:
???

```
private def timestamp =
  dateFormat.format(new Date())

def atPageLoad(template:NodeSeq) =
  <div class="timestamp">
    {timestamp}
  </div>
```


Time
Server time at page load:
Saturday, August 15, 2015 11:45:06 AM CDT

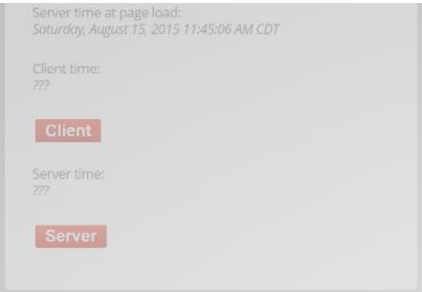
Client time:
???

Server time:
???

```
angular.module("TimeApp", [
  "ClientTimeModule",
  "ServerTimeModule"
])
```

```
))
```

```
.controller("TimeController", [
  "$scope",
  "ClientTime",
  "ServerTime",
```


Server time at page load:
Saturday, August 15, 2015 11:45:06 AM CDT

Client time:
???

Server time:
???

```
            "ClientTimeModule",
            "ServerTimeModule"
          ])



          .controller("TimeController", [
            "$scope",
))          "ClientTime",
            "ServerTime",



ript>


          function($scope, clientTime, serverTime) {
            // ng-bind="client"
            $scope.client = "???";
```

ript>

```
function($scope, clientTime, serverTime) {
  // ng-bind="client"
  $scope.client = "???";
  // ng-bind="server"
  $scope.server = "???";
  // ng-click="getClient()"
  $scope.getClient = function() {
    $scope.client = clientTime.currentTime()
  };
```

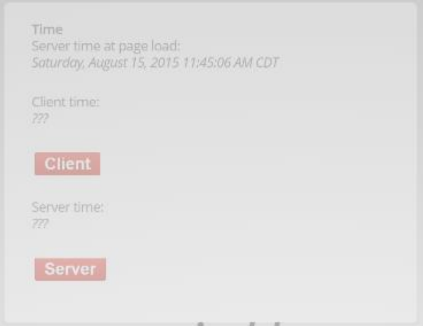
Enter a GitHub ID:

github_id

Screen name:
Followers:
Repos:
Stars:
Forks:
Total:

```
angular.module("ClientTimeModule", [])
.factory("ClientTime", function() {
  return {
    currentTime: function() {
      return new Date()
    }
  }
})
;
```

```
// ng-click="getServer()"
$scope.getServer = function() {
  serverTime.currentTime() // promise from server
    .then(function(timestamp) {
      $scope.server = timestamp;
    });
};




angular.module("ClientTimeModule", [])
 .factory("ClientTime", function() {
```


Time
Server time at page load:
Saturday, August 15, 2015 11:45:06 AM CDT

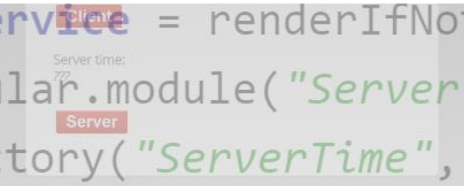
Client time:
???

Server time:
???

```
// lift-ng magic!!
def service = renderIfNotAlreadyDefined(
  angular.module("ServerTimeModule")
  .factory("ServerTime", jsObjFactory()
    .jsonCall("currentTime", Full(timestamp))
))
```

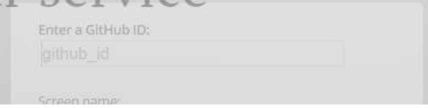
```
<script data-lift="ServerTime.service"></script>
```

```scala
def service = renderIfNotAlreadyDefined(
  angular.module("ServerTimeModule")
  .factory("ServerTime", jsObjFactory()
    .jsonCall("currentTime", Full(timestamp))
))
```



```html
<script data-lift="ServerTime.service"></script>
```

View the source to see the generated angular service



```html
<script data-lift="ServerTime.service"></script>
```

View the source to see the generated angular service

Enter a GitHub ID:

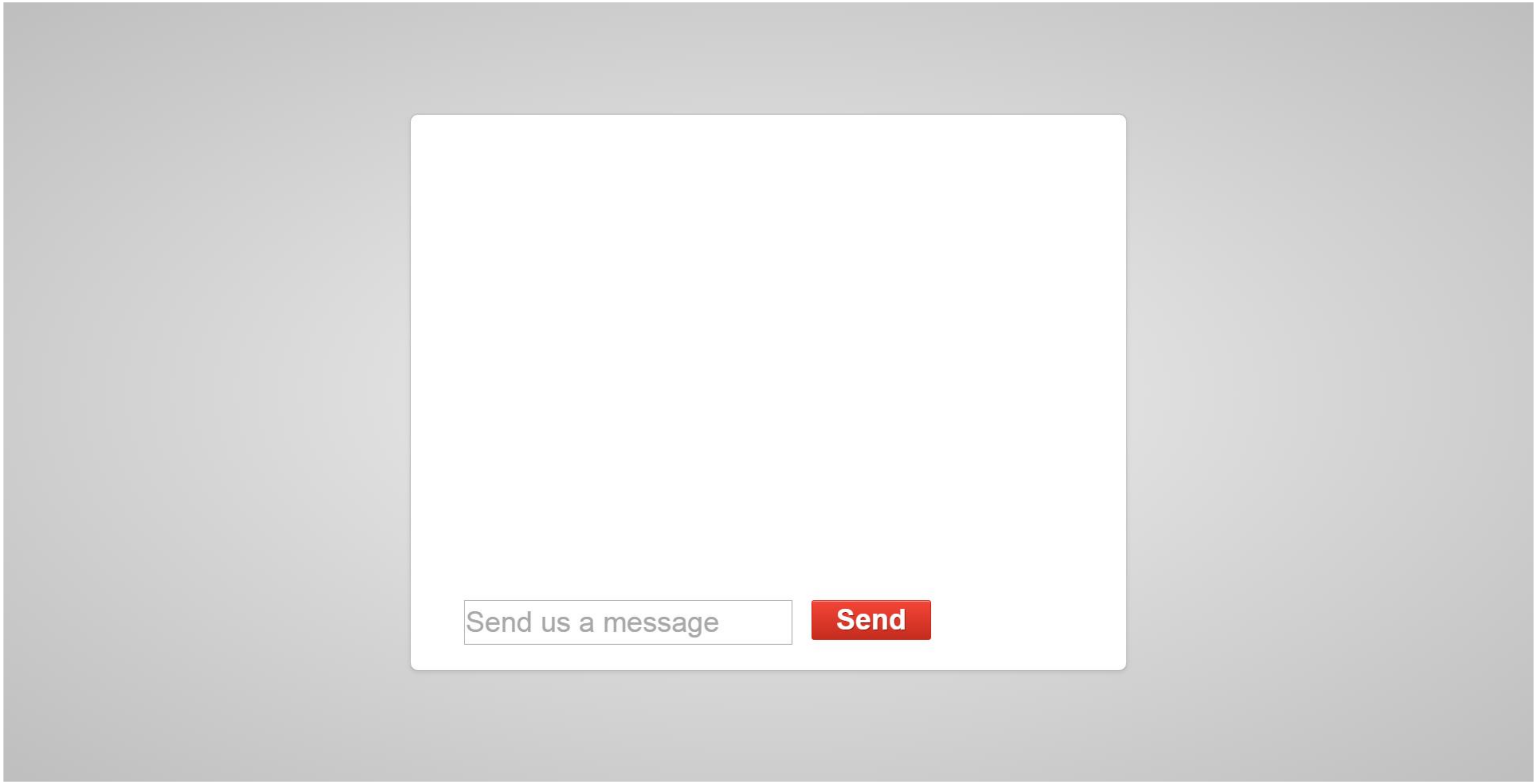
github_id

Screen name:
Followers:
Repos:
Stars:
Forks:
Total:

Chat App

Server push via `$scope`

Lift comet via actors


Send us a message
Send

```
<div ng-controller="ChatController">
  <div data-lift="comet?type=ChatComet"></div>
  <ul id="chat-out">
    <li ng-repeat="m in messages track by $index"
        ng-bind="m"></li>
  </ul>
  <div id="chat-in">
    <button ng-click="sendChat()">Send</button>
    <input type="text" placeholder="Send us a message"
           ng-model="message" ng-keypress="onKeypress($event)"
  </div>
</div>
```

```
$scope.sendChat = function() {
  server.send($scope.message);
  $scope.message = "";
};
```

Chat App

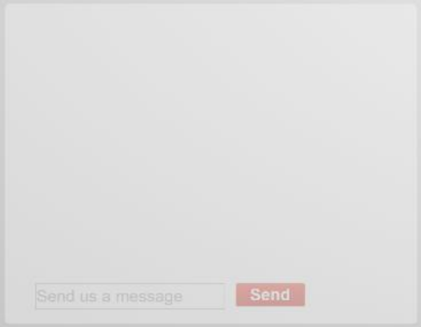
Server push via $scope

Lift comet via actors

Chat App

Server push via `$scope`

Lift comet via actors



```
angular.module("ChatModule")
.factory("ChatServer", jsObjFactory()
  .jsonCall("send", (chat:String) => {
    ChatServer ! chat
    Empty
  })
)
```

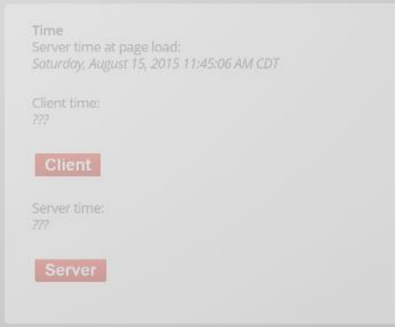
Server Time App

Lift template basics

Angular template basics

JS Angular factory

lift-ng Angular factory


Time
Server time at page load:
Saturday, August 15, 2015 11:45:06 AM CDT

Client time:
???

Server time:
???

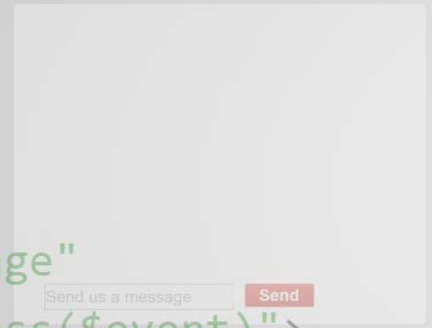
```
iv>
```

```
index"
```

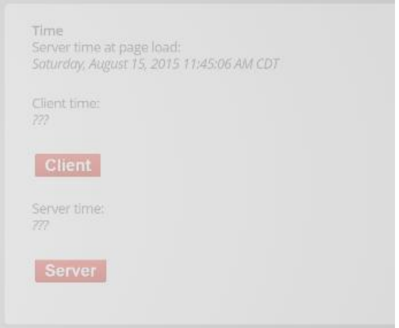
Chat App
Server push via $scope
Lift comet via actors

Server Time App
Lift template basics
Angular template basics
JS Angular factory
lift-ng Angular factory

```
iv>

index"

ton>
 a message"
onKeypress($event)">
```



```scala
object ChatServer extends LiftActor
  with ListenerManager {
  override def lowPriority = {
    case msg:String =>
      sendListenersMessage(msg)
  }
```


Time
Server time at page load:
Saturday, August 15, 2015 11:45:06 AM CDT

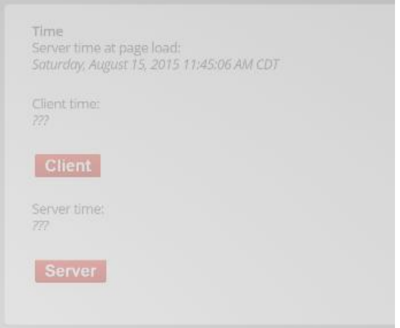
Client time:
???

Client

Server time:
???

Server

```scala
class ChatComet extends AngularActor
```

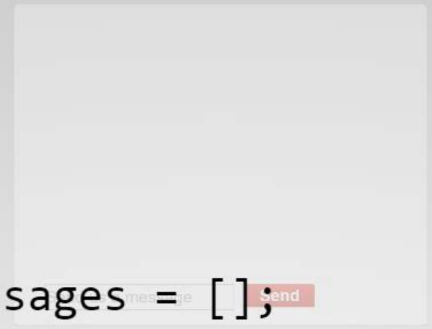

Time
Server time at page load:
Saturday, August 15, 2015 11:45:06 AM CDT

Client time:
???

Server time:
???

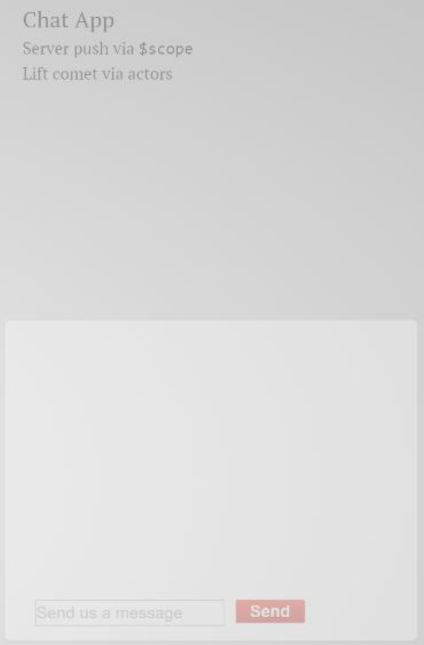
```
class ChatComet extends AngularActor
  with CometListener {
  override def registerWith = ChatServer
  override def lowPriority = {
    case msg:String =>
      scope.emit("new-message", msg)
  }
}
```

```
</div>
```



```
$scope.messages = [];
$scope.$on("new-message",
  function(e, msg){
    $scope.messages.push(msg);
  }
);
```

View the source to see the comet's DOM

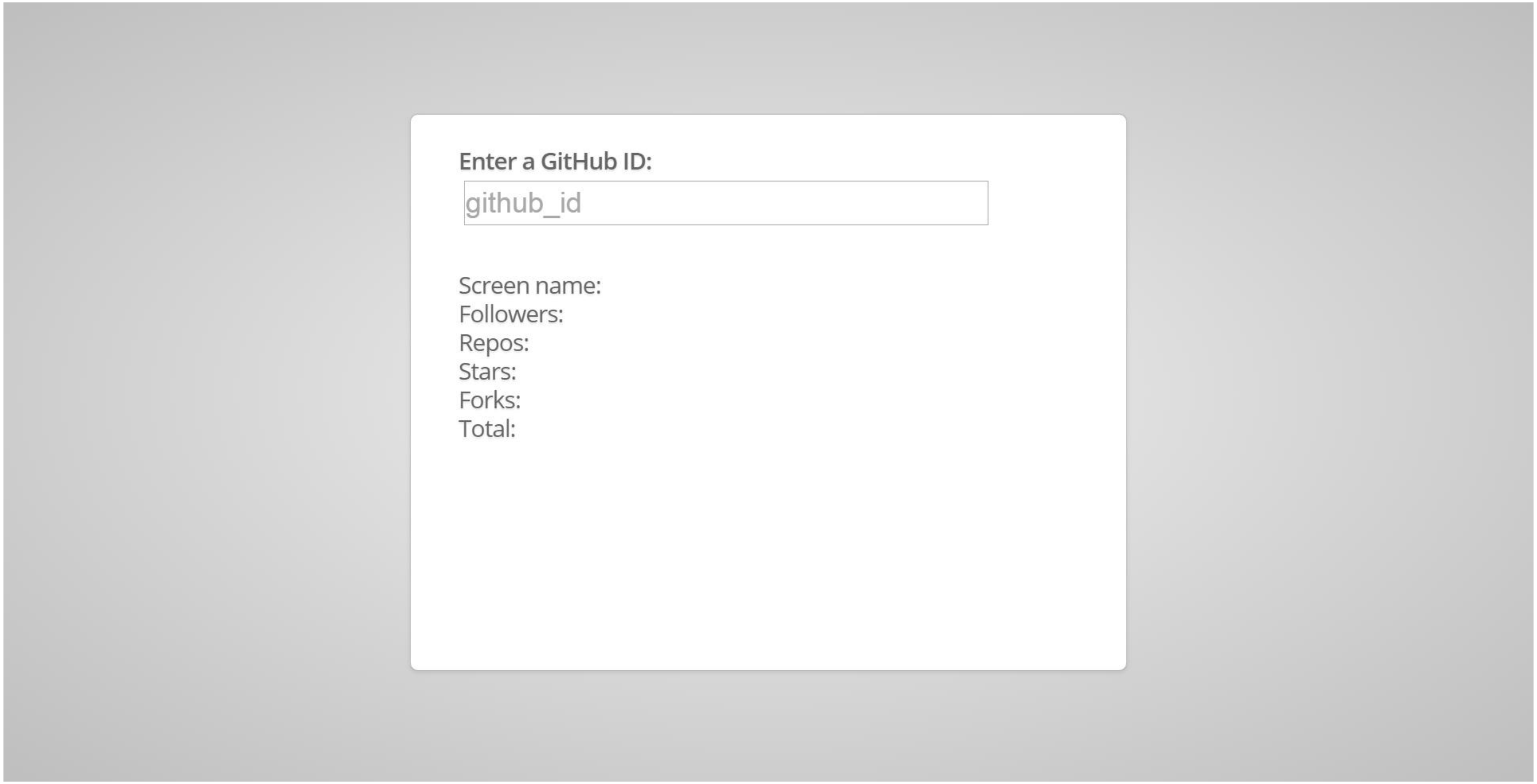

Chat App

Server push via $scope

Lift comet via actors

OSS Score App

Scala Futures -> $q Promises



Enter a GitHub ID:

github_id

Screen name:
Followers:
Repos:
Stars:
Forks:
Total:

```
<div ng-controller="OssScoreController">
  <div class="title">Enter a GitHub ID:</div>
  <input type="text" placeholder="github_id"
         ng-model="enteredId" ng-keypress="onKeypress($event)"

  <div class="avatar" ng-style="{'background-image': 'url('+av
  <div>Screen name: <span ng-bind="id"></span></div>
  <div>Followers: <span ng-bind="followers"></span></div>
  <div>Repos: <span ng-bind="repos"></span></div>
  <div>Stars: <span ng-bind="stars"></span></div>
  <div>Forks: <span ng-bind="forks"></span></div>
  <div>Total: <span ng-bind="total"></span></div>
</div>
```

OSS Score App

Scala Futures -> $q Promises

Nothing interesting to see here...

View the source to see the generated angular service

```
// ng-bind="client"
$scope.client = "???";
// ng-bind="server"
$scope.server = "???";
// ng-click="getClient()"
$scope.getClient = function() {
  $scope.client = clientTime.currentTime()
};
```

```
<div ng-controller="OssScoreController">
  <div class="title">Enter a GitHub ID:</div>
  <input type="text" placeholder="github_id"
         ng-model="enteredId" ng-keypress="onKeypress($event)">

  <div class="avatar" ng-style="{'background-image': 'url('+avatar+')'}"></div>
  <div>Screen name: <span ng-bind="id"></span></div>
  <div>Followers: <span ng-bind="followers"></span></div>
  <div>Repos: <span ng-bind="repos"></span></div>
  <div>Stars: <span ng-bind="stars"></span></div>
  <div>Forks: <span ng-bind="forks"></span></div>
  <div>Total: <span ng-bind="total"></span></div>
</div>
```

tor App
ing values between client/server
rimental!

OSS Score App
Scala Futures -> $q Promises

```
github.get($scope.enteredId).then(function(profile){
  $scope.id = profile.id;
  profile.avatar.then(function(avatar){
    $scope.avatar = avatar });
  profile.followers.then(function(count){
    $scope.followers = count });
  profile.repos.then(function(count){
    $scope.repos = count });
  profile.stars.then(function(count){
    $scope.stars = count });
```

```
case class GitHub(
  id:String,
  avatar:Future[String],
  followers:Future[Int],
  repos:Future[Int],
  stars:Future[Int],
  forks:Future[Int]
)
```

Screen name:
Followers:
Repos:
Stars:
Forks:
Total:

OSS Score App

```
github.get($scope.enteredId).then(function(profile){
  $scope.id = profile.id;
  profile.avatar.then(function(avatar){
    $scope.avatar = avatar });
  profile.followers.then(function(count){
    $scope.followers = count });
  profile.repos.then(function(count){
    $scope.repos = count });
  profile.stars.then(function(count){
    $scope.stars = count });
  profile.forks.then(function(count){
    $scope.forks = count });
```

```
{
  "id": "joescii",
profile){
```

Screen name:
Followers:
Repos:
Stars:
Forks:
Total:

OSS Score App

Scala Futures -> $q Promises

```scala
case class GitHub(
  id:String,
  avatar:Future[String],
  followers:Future[Int],
  repos:Future[Int],
  stars:Future[Int],
  forks:Future[Int]
)
```

OSS Score App

Scala Futures -> $q Promises

```
ture":

.future":

ure":

ure":

ure":
```

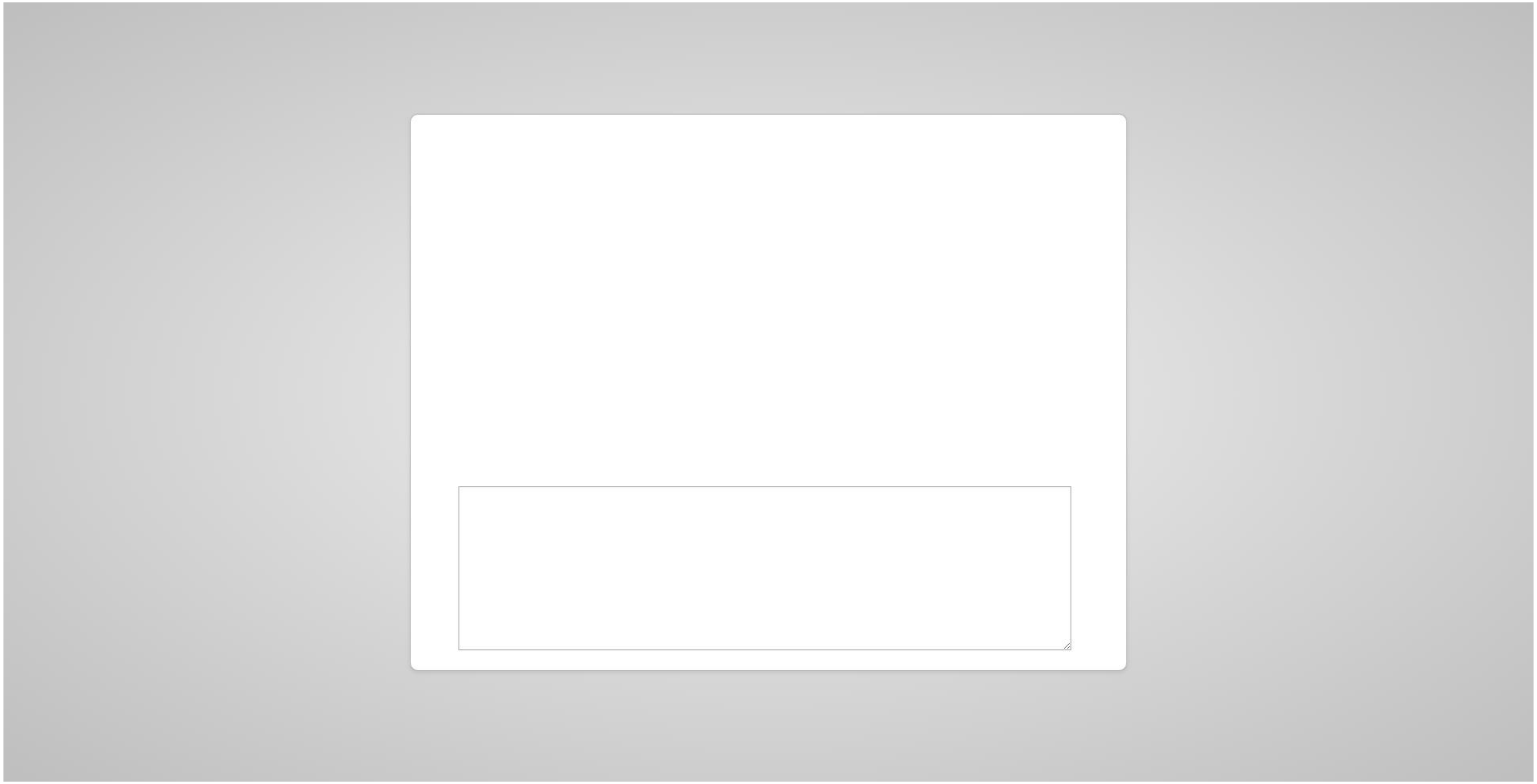
```
angular.module("GitHubModule")
  .factory("GitHub", jsObjFactory()
    .jsonCall("get", (github:String) => {
      val gh:GitHub = GitHub.accountFor(github)
      Full(gh)
    })
  )
```

```
{
  "id": "joescii",
  "avatar": {"net.liftmodules.ng.Angular.future":
    "NGU0CXFJYBFXQWUF1RVZ"},
  "followers": {"net.liftmodules.ng.Angular.future":
    "NGPWZRDOWPYUN15QGMGO"},
  "repos": {"net.liftmodules.ng.Angular.future":
    "NGGOYWWGGJ1V1GJXOABW"},
  "stars": {"net.liftmodules.ng.Angular.future":
    "NGZHOHIG0BS4IJ52A5FP"},
  "forks": {"net.liftmodules.ng.Angular.future":
    "NGAERCVUPFXNMP5QGZU3"}
}
```

Editor App

Binding values between client/server

Experimental!



```
<div ng-controller="EditorController">
  <div data-lift="Angular.bind?type=InputBinder"></div>
  <div data-lift="Angular.bind?type=OutputBinder"></div>
  <div class="document" ng-bind-html="output.dom"></div>
  <textarea ng-model="input.mdtext"></textarea>
</div>
```

Editor App

Binding values between client/server

Experimental!

```
angular.module("EditorApp", ["ngSanitize"])
.controller("EditorController", function(){})
;
```

Editor App
Binding values between client/server
Experimental!

```scala
case class Input(mdtext:String) extends NgModel
case class Output(dom:String) extends NgModel
```

Editor App

Binding values between client/server

Experimental!

```
class InputBinder
  extends SimpleNgModelBinder(
    "input",   // Bind to $scope.input
    Input("")  // Initial value
  ) with BindingToServer {
```

Editor App

Binding values between client/server

Experimental!

```
class OutputBinder
  extends SimpleNgModelBinder(
    "output",  // Bind to $scope.output
    Output(<div></div>)
  ) with BindingToClient with SessionScope
```

```
override val onClientUpdate =
{ input:Input =>
  for {
    session <- S.session
    content <- MarkdownParser.parse(input.mdtext)
  } {
    session.sendCometActorMessage(
      "OutputBinder",
      Empty, // Comet actors optionally have names
      Output(content)
    )
  }
```

Support for i18n

```
$scope.sendChat = function() {
  server.send($scope.message);
  $scope.message = "";
};
```

```
angular.module("ChatModule")
.factory("ChatServer", jsObjFactory()
  .jsonCall("send", (chat:String) => {
    ChatServer ! chat
    Empty
  })
)
```

```
<div ng-controller="ChatController">
  <div data-lift="comet?type=ChatComet"></div>
  <ul id="chat-out">
   <li ng-repeat="m in messages track by $index"
       ng-bind="m"></li>
  </ul>
 <div id="chat-in">
  <button ng-click="sendChat()">Send</button>
 <input type="text" placeholder="Send us a message"
       ng-model="message" ng-keypress="onKeypress($event)">
</div>
```

```
object ChatServer extends LiftActor
  with ListenerManager {
  override def lowPriority = {
    case msg:String =>
      sendListenersMessage(msg)
  }
```

Chat App

Server Time App

```
<div ng-controller="TimeController">
  <div class="title">Time</div>
  <div>Server time at page load:</div>
  <div data-lift="ServerTime.atPageLoad"
    (this div replaced at page load ti
  </div>
  <div>Client time:</div>
  <div class="timestamp" ng-bind=
  <button ng-click="getClient(
  <div>Server time:</div>
  <div class="timestamp" ng
```

w the source to see the comet's

Support for i18n

```
# my-bundle.properties
hello=¡Hola!
bye=Adios, {0}
```

View the source to see the comet's DOM

```
<div ng-controller="ChatController">
  <div data-lift="comet?type=ChatComet"></div>
  <ul id="chat-out">
    <li ng-repeat="m in messages track by $index"
        ng-bind="m"></li>
  </ul>
  <div id="chat-in">
    <button ng-click="sendChat()">Send</button>
    <input type="text" placeholder="Send us a message"
        ng-model="message" ng-keypress="onKeypress($event)">
  </div>
</div>
```

```
object ChatServer extends LiftActor
  with ListenerManager {
  override def lowPriority = {
    case msg:String =>
      sendListenersMessage(msg)
  }
```

```
class ChatComet extends AngularActor
  with CometListener {
  override def registerWith = ChatServer
  override def lowPriority = {
    case msg:String =>
      scope.emit("new-message", msg)
  }
}
```

```
$scope.messages = [];
$scope.$on("new-message",
  function(e, msg){
    $scope.messages.push(msg);
  }
);
```

```
<div ng-controller="TimeController">
  <div class="title">Time</div>
  <div>Server time at page load:</div>
  <div data-lift="ServerTime.atPageLoad">
    (this div replaced at page load time)
  </div>
  <div>Client time:</div>
  <div class="timestamp" ng-bind="client"></div>
  <button ng-click="getClient()">Client</button>
  <div>Server time:</div>
  <div class="timestamp" ng-bind="server"></div>
  <button ng-click="getServer()">Server</button>
</div>
```

```
def service = renderIfNotAlreadyD
  angular.module("ServerTimeMod
    .factory("ServerTime", jsOb
      .jsonCall("currentTime"
    ))
```

```
<script data-lift="
```

```
# my-bundle.properties
hello=¡Hola!
bye=Adios, {0}
```

```
angular.module('ExampleApp', ['i18n'])
.controller('ExampleController',
  ['$scope', 'my-bundle',
    function($scope, i18n) {
      $scope.hello = i18n.hello;
      $scope.bye = i18n.bye($scope.username);
}]);
```

```
['$scope', 'my-bundle',
  function($scope, i18n) {
    $scope.hello = i18n.hello;
    $scope.bye = i18n.bye($scope.username);
}]);
```

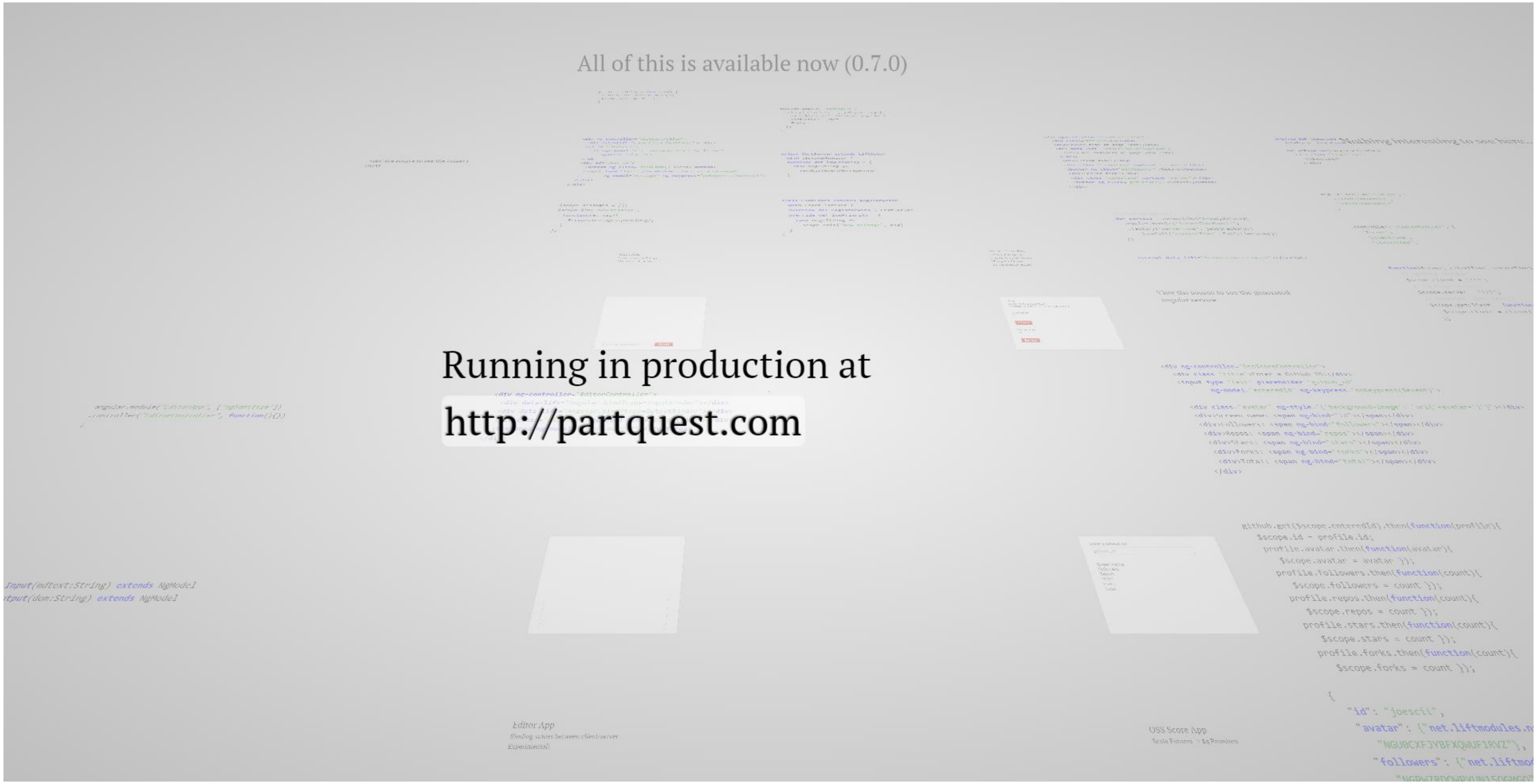
All of this is available now (0.7.0)

```
dule("EditorApp", ["ngSanitize"])
"EditorController", function(){})
```

```
<div ng-controller="EditorController">
  <div data-lift="Angular.bind?type=InputBinder"></div>
  <div data-lift="Angular.bind?type=OutputBinder"></div>
  <div class="document" ng-bind-html="output.dom"></div>
  <textarea ng-model="input.mdtext"></textarea>
</div>
```

```
<div ng-controller="GhScoreController">
  <div class="title">Enter a GitHub ID:</div>
  <input type="text" placeholder="github_id"
         ng-model="enteredId" ng-keypress="onKeypress(

    <div class="avatar" ng-style="{'background-image
    <div>Screen name: <span ng-bind="id"></span></
    <div>Followers: <span ng-bind="followers"></s
    <div>Repos: <span ng-bind="repos"></span></
    <div>Stars: <span ng-bind="stars"></span>
    <div>Forks: <span ng-bind="forks"></spa
    <div>Total: <span ng-bind="total"></s
  </div>
```

```
github.get($scope.enteredId
  $scope.id = profile.id;
```



All of this is available now (0.7.0)

```
angular.module('EditorApp', ['ngSanitize'])
.controller('EditorController', function(){})
```

```
Input(mdtext:String) extends NgModel
tput(dom:String) extends NgModel
```

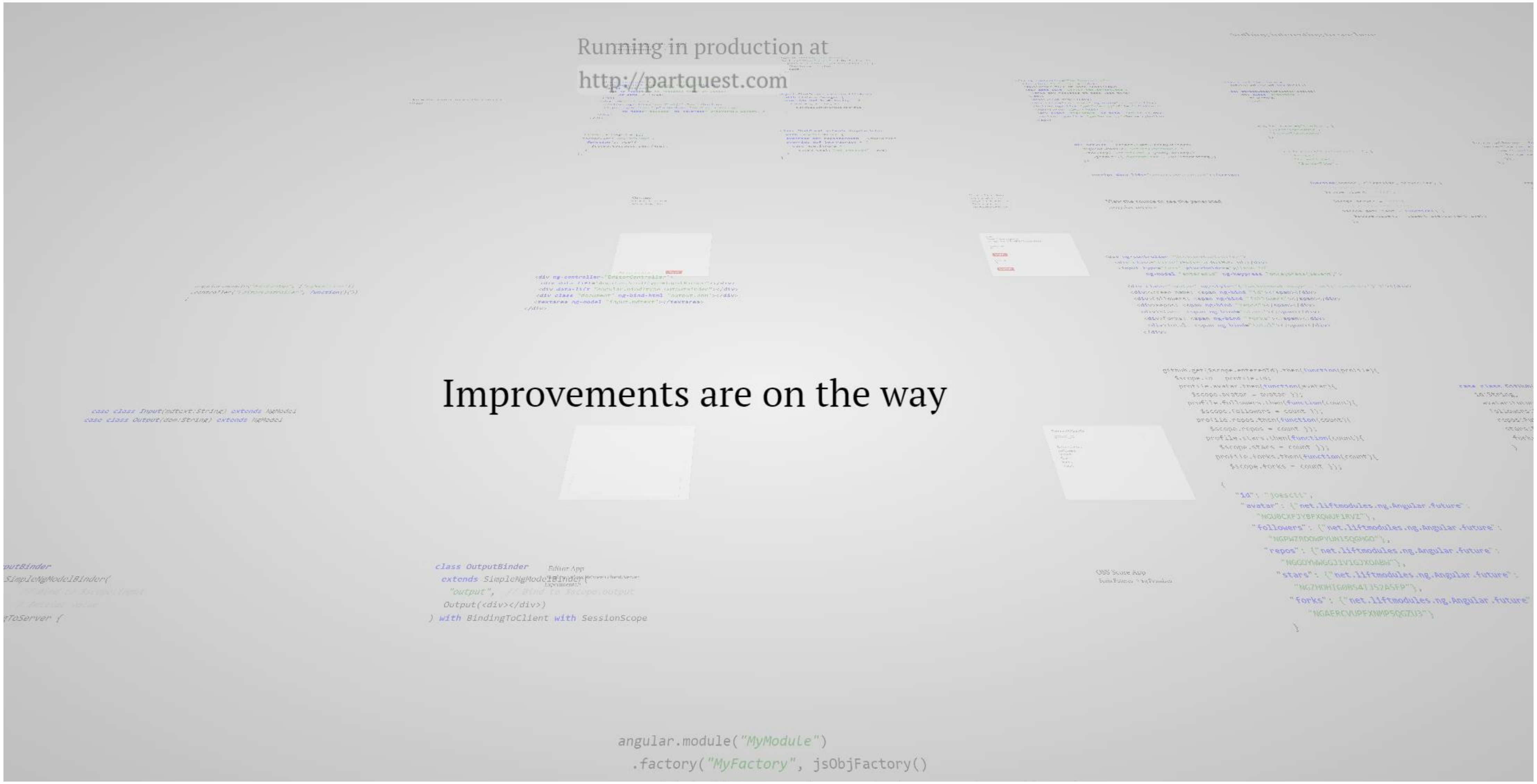
```
github.get($scope.enteredId).then(function(profile){
  $scope.id = profile.id;
   profile.avatar.then(function(avatar){
     $scope.avatar = avatar });
    profile.followers.then(function(count){
      $scope.followers = count });
     profile.repos.then(function(count){
       $scope.repos = count });
      profile.stars.then(function(count){
        $scope.stars = count });
       profile.forks.then(function(count){
         $scope.forks = count });

{
  "id": "joescii",
   "avatar": {"net.liftmodules.n
     "NGU0CXFJYBFXQWUF1RVZ"},
      "followers": {"net.liftmod
```

Editor App
Binding values between client/server
Experimental!

OSS Score App
Scala Futures -> $q Promises

Running in production at

http://partquest.com



Running in production at
http://partquest.com

Improvements are on the way

```
case class Input(edtext:String) extends NgModel
case class Output(dom:String) extends NgModel
```

```
class OutputBinder
  extends SimpleNgModelBinder(
    "output",   // Bind to $scope.output
    Output(<div></div>)
) with BindingToClient with SessionScope
```

```
angular.module("MyModule")
  .factory("MyFactory", jsObjFactory()
angular.module("MyModule")
  .factory("MyFactory", jsObjFactory()
    .defs(callServer = (arg:String) => Service call arg)
    .vals(aConst = "Evaluated at page-load!")
  )
```

Currently can only push to a `$scope`. Find a way to tie a comet actor to a factory.

And maybe we'll dig into Angular 2

Every feature is covered with either a unit test or Selenium integration test. I'm keenly interested in not breaking anything.

Thank you for your interest!

Download Slides

lift-ng

giter8 Template

Presentation Source