

# Invisible.js

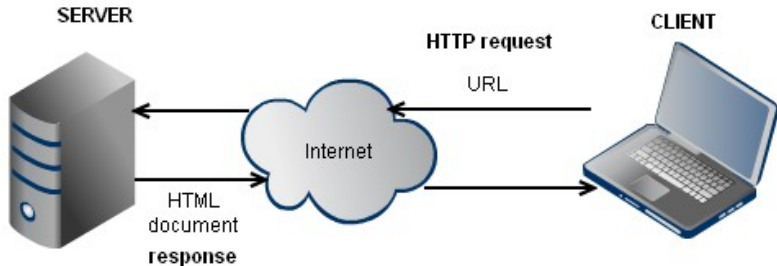
Trabajo profesional de Ingeniería en Informática

Martín Paulucci  
Facundo Olano

Facultad de Ingeniería  
Universidad de Buenos Aires

20 de Diciembre, 2013

# Arquitectura Web



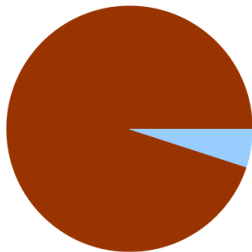
# Programación Web: Prehistoria

- ▶ Páginas estáticas
- ▶ CGI Scripts
- ▶ Aplicaciones tradicionales (Java, PHP)

95 % ■  
Server

5 % ■  
Client

Client vs Server

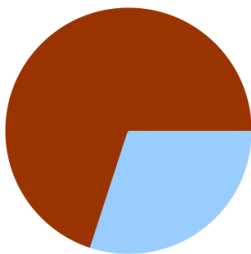


# Programación Web: Historia

70 % ■  
Server

30 % ■  
Client

Client vs Server



- ▶ Frameworks MVC (Rails, Django, Cake)
- ▶ Interfaces amigables (JavaScript)
- ▶ Interfaces dinámicas (JQuery, AJAX)

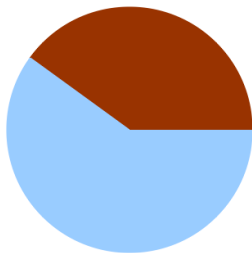
# Programación Web: Actualidad

- ▶ Single Page Applications
- ▶ Backbone
- ▶ MVVM: Knockout, Angular
- ▶ HTML5, Realtime

40 % ■  
Server

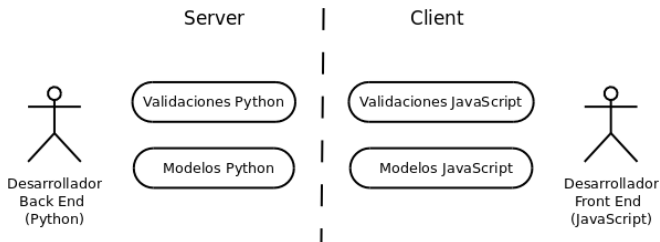
60 % ■  
Client

Client vs Server



# Problemas Actuales

- ▶ Duplicación de modelos
- ▶ Duplicación de validaciones
- ▶ Constante cambio del lenguaje

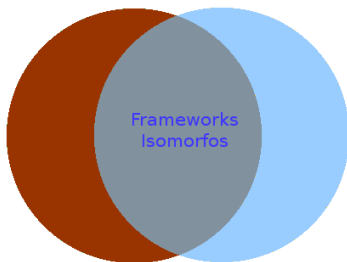


# Programación Web: Futuro

*"The best way to predict the future is to invent it." Alan Kay*

- ▶ Meteor
- ▶ Derby
- ▶ Invisible?

CLIENT vs SERVER



# Nuestro objetivo

- ▶ Definir modelos una vez, utilizarlos en cualquier lugar
- ▶ Validaciones definidas una vez
- ▶ Métodos CRUD utilizados indistintamente en servidor y cliente
- ▶ Soporte Realtime

*“Al buscar lo imposible el hombre siempre ha realizado y reconocido lo posible.” Mijaíl Bakunin*



# Investigación

*"If you think it's simple, then you have misunderstood the problem."*

*Bjarne Stroustrup*

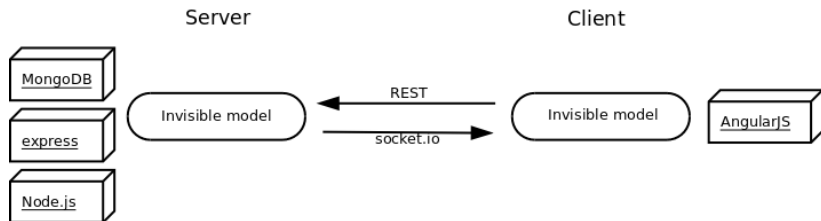
- ▶ CoffeeScript
- ▶ Server Push
  - Log Polling
  - SSE
  - WebSockets
  - socket.io
- ▶ REST
- ▶ Templates
- ▶ node.js
  - Express
  - Derby
  - Meteor
  - Flatiron
- ▶ Client MV\*
  - Backbone
  - Angular
  - Knockout

# Prototipos Realizados

- ▶ *Fodder* : REST + SSE + Handlebars
- ▶ *drymodels* : Node.js + Express + Backbone
- ▶ *Acekia* : Node.js + Angular + CoffeeScript

*“Hay que unir lo teórico a lo Real, lo ideal a lo Empírico.” Juan Perón*

# Arquitectura en Invisible.js



*"There's only one trick in software, and that is using a piece of software that's already been written." Bill Gates*

# Modelos en Invisible.js

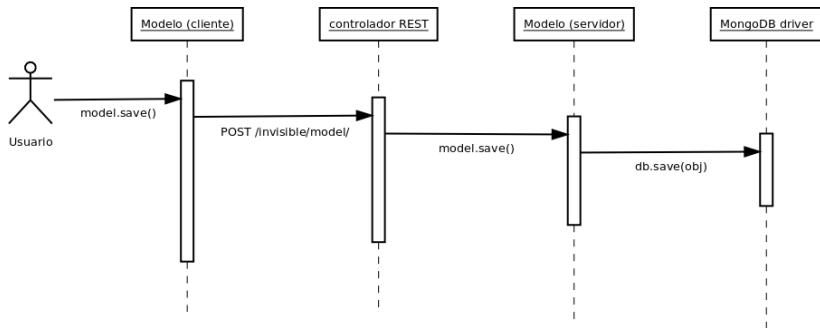
```
class Person
  getFullName()
  getAvatarUrl()
```

```
class Person
  getFullName()
  getAvatarUrl()

  save(cb)
  delete(cb)
  query(query, opts
    , cb)
  findById(id, cb)

  onNew(cb)
  onUpdate(cb)
  onDelete(cb)
```

# Modelos en Invisible.js (2)



# Invisible.js en acción: Mensaje

*"Talk is cheap. Show me the code." Linus Torvalds*

models/message.coffee:

```
Invisible = require("invisible")

class Message
  constructor: (@from, @body) ->
    @date = new Date()

module.exports = Invisible.createModel("
  Message", Message)
```

# Invisible.js en acción: Usuario

models/user.coffee:

```
crypto = require("crypto")

SALT = "MY REALLY LONG HASH SALT"

class User
  constructor: (@email)->

  setPassword: (plainPassword) ->
    h = crypto.createHash('sha1')
    h.update(plainPassword)
    h.update(SALT)
    @password = h.digest('base64')
```

# Invisible.js en acción: Validaciones

*"Programming is usually taught by examples." Niklaus Wirth*

models/user.coffee:

```
validations: { methods: ['checkUnique'] }

checkUnique: (done) ->
  Invisible.User.query {email: @email},
    (err, res) ->
      if res.length == 1
        done({valid : false, errors:
          ["email already registered"]})
      else if res.length == 0
        done({valid : true})
```



# Invisible.js en acción: Configuración

app.coffee:

```
invisible = require("invisible")
express = require("express")
app = express()

# Express middleware

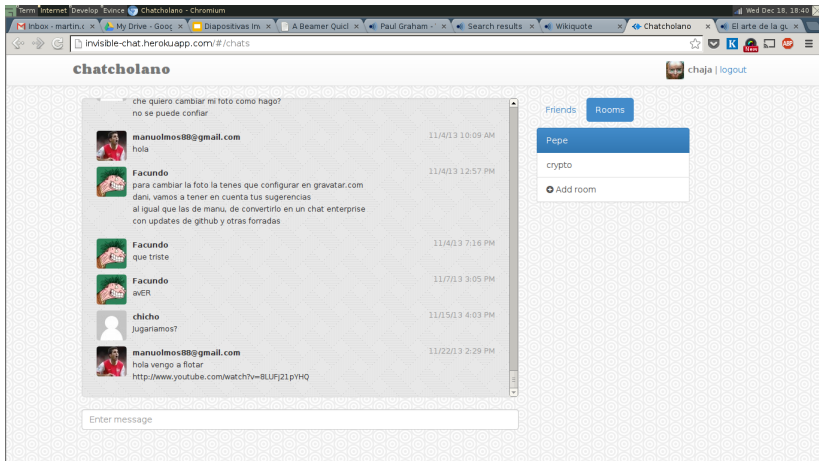
invisible.createServer app, "models", ()
  ->
    console.log("Invisible server listening
      on port " + app.get("port"))
```

# Invisible.js en acción: Configuración (2)

public/index.html:

```
<html>
  <body>
    <script src="invisible.js"></script>
    <script type="text/javascript">
      var message = new Invisible.
        Message()
      console.log(message.date)
    </script>
  </body>
</html>
```

# Demo

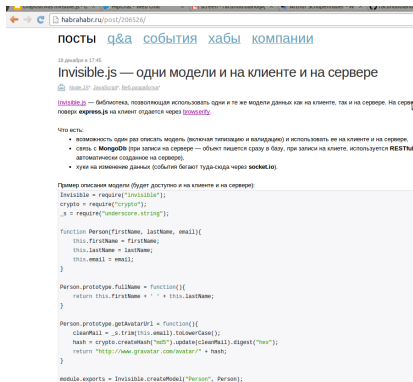
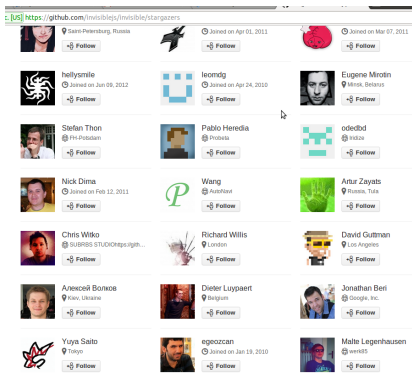


# Conclusiones

- ▶ Se realizó un trabajo integral de Ingeniería de Software.
- ▶ Se obtuvo un framework útil, sencillo de utilizar, y con potencial de desarrollo.
- ▶ Se cumplieron los objetivos del proyecto y se superaron las expectativas iniciales.

# Repercusiones

*"The secret of politics? Make a good treaty with Russia." Otto von Bismarck*



# Trabajo futuro

- ▶ Implementar módulo de seguridad
- ▶ Evaluar escalabilidad
- ▶ Incentivar el desarrollo comunitario

# Preguntas?

# Agradecimientos