

Contenido

- **Introducción**
 - ¿Qué es?(Real-time)
 - Popularidad
 - Historia
 - Comparación
 - Documentación
 - Arquitectura básica
 - Hello world!
- **Características**
 - Server and client
 - Collections - Mongo(Mini)
 - Tracker
 - Publish and subscribe
 - Hot code push
 - DDP
 - LiveQuery
 - Iron router
- **Blaze**
- **Testing and Deployment**
- **Lo malo**
- **Apreciaciones**
- **Demo (ToDoList - Mobile)**



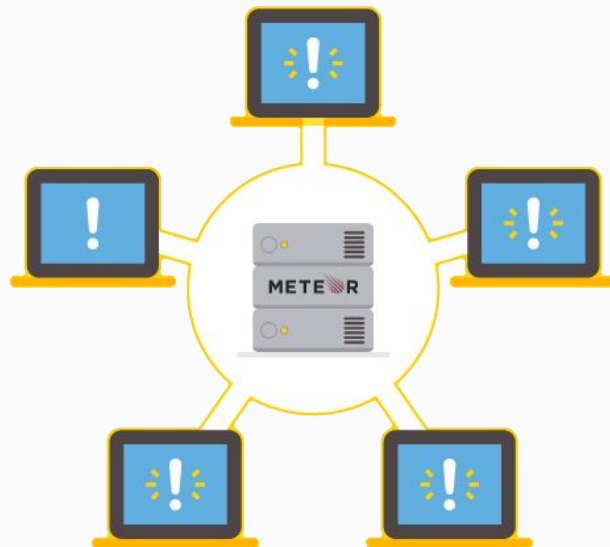
Por:

**Pablo Sanabria, Franco Muñoz,
Martin Fuenzalida y Juan Diego Diaz**

¿Qué es Meteor?

- Framework web **real-time reactive**
- Opera en el servidor y en el cliente
- Mantiene **sincronizadas** ambas partes
- **Comparte código** en ambas partes
- Escrito usando **Node.js**

METEOR



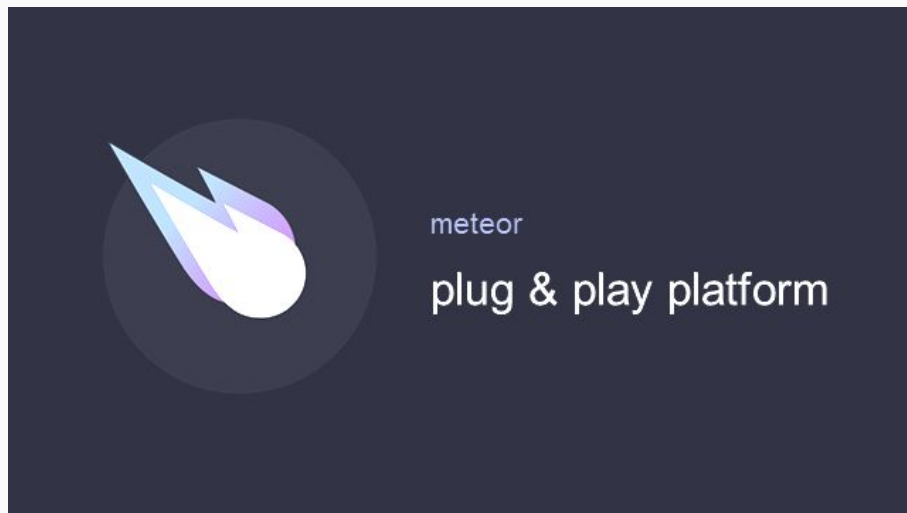
¿Qué es Meteor?

- **Full Stack**
 - Node.js, Servidor web & MongoDB
- Aplicaciones multi-plataforma
 - Cordova & Electron
- **Licencia MIT**
- Versión estable 1.4



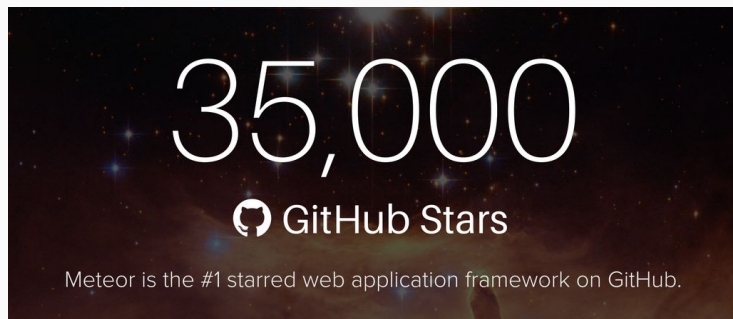
7 principios de la filosofía Meteor

1. Data on the Wire
2. One Language: Javascript
3. Database Everywhere
4. Latency compensation
5. Full Stack Reactivity
6. Embrace the Ecosystem
7. Simplicity Equals Productivity



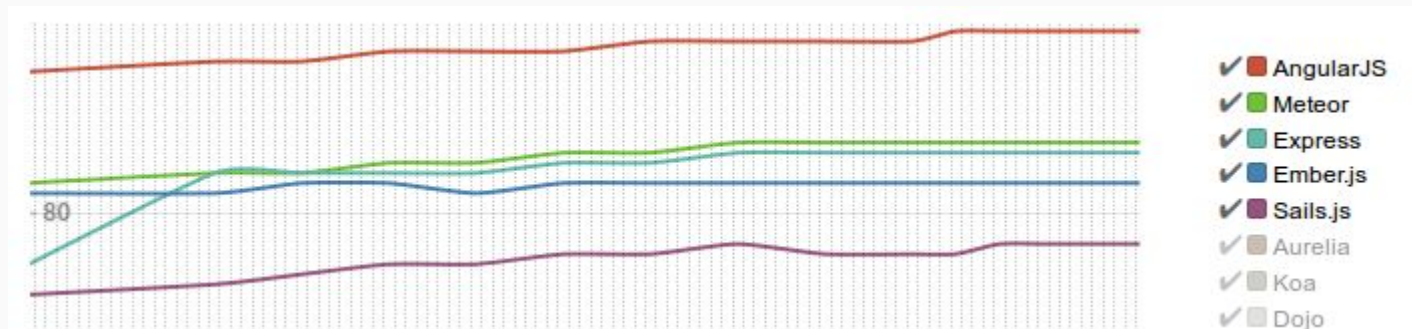
<https://c.kat.pe/seven-principles-of-meteor-dot-js>

Popularidad



JavaScript

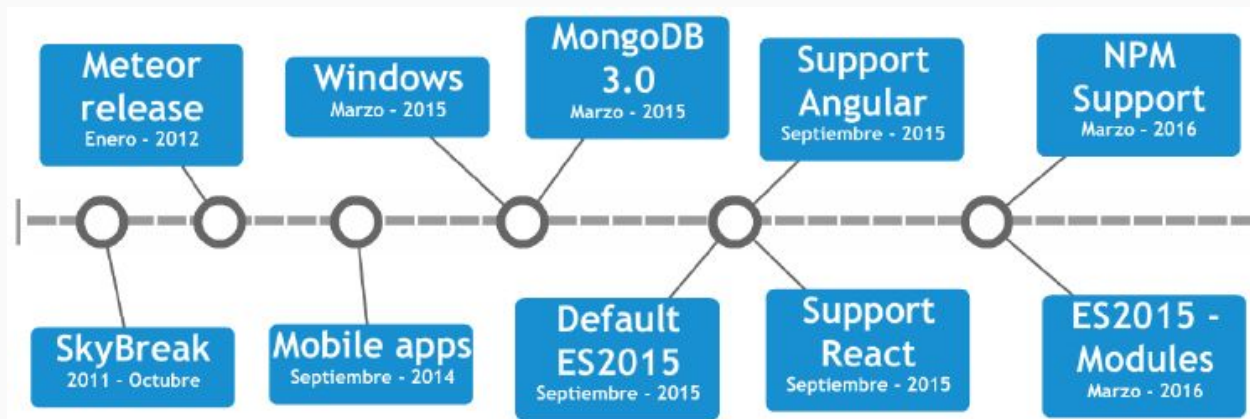
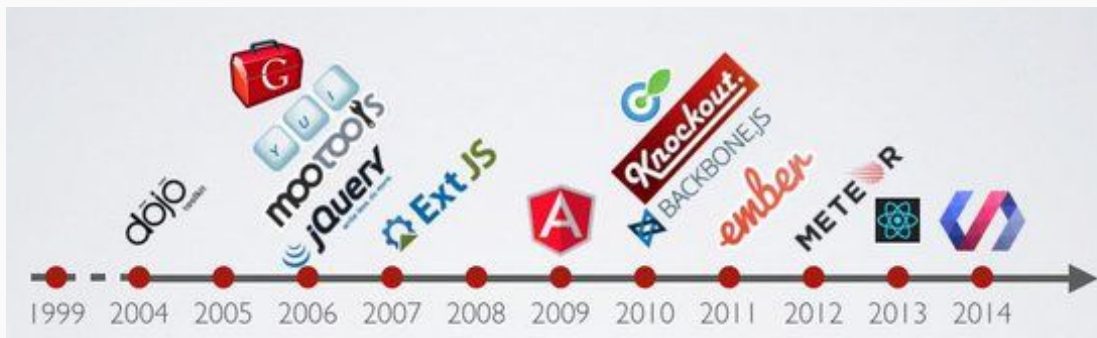
Framework	Score
AngularJS	98
Meteor	87
Express	86
Ember.js	83
Sails.js	77



Basado estrellas de Github y uso de tags en Stackoverflow. Fuente: hotframeworks.com

Historia

- Sucesor de SkyBreak
- 0.1 Enero 2012
- 1.4 Julio 2016



Comparación

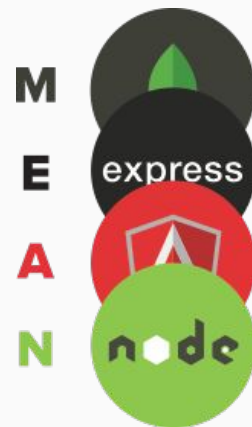
- Meteor se parece al stack MEAN
- Meteor prioriza la simplicidad
- Ser real-time lo diferencia
- Deja de lado la arquitectura MVC
- Convive con Angular o React

METEOR

express



sails



Documentación

- **API docs:** información granular de elementos. Ej. Session#set
- **Guides:** guías de componentes típicos. Ej. User and Account
- **Tutorials:** tutoriales extensos sobre aplicaciones. Ej. Whatsapp Clone



API docs

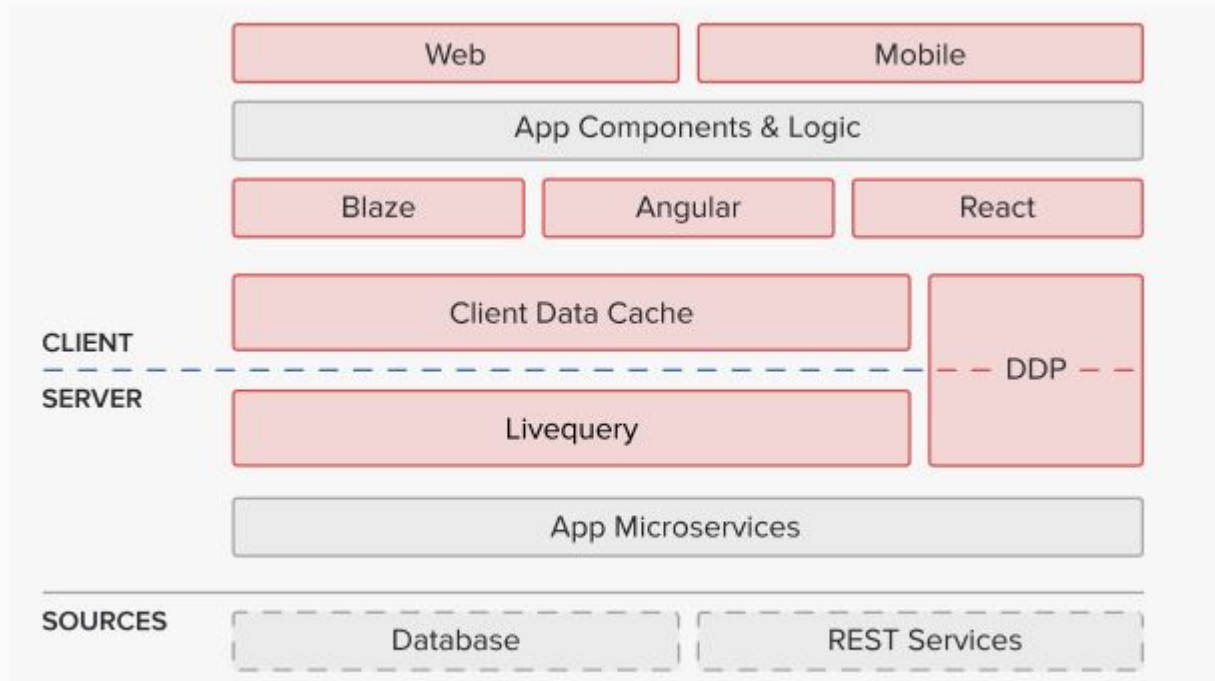


Tutorials



Guides









Arquitectura básica



Hello world!

```
$ curl https://install.meteor.com | sh
$ meteor create helloworld
$ cd helloworld/
$ meteor
[[[[[ ~/helloworld ]]]]]
=> Started proxy.
=> Started MongoDB.
=> Started your app.
=> App running at: http://localhost:3000/
```

FOLDERS

- ▼  helloworld
 - ▶  .meteor
 - ▼  client
 -  main.css
 -  main.html
 -  main.js
 -  .gitignore
 -  package.json

Welcome to Meteor!

Click Me

You've pressed the button 1 times.

Learn Meteor!

- [Do the Tutorial](#)
- [Follow the Guide](#)
- [Read the Docs](#)
- [Discussions](#)

SERVER AND CLIENT

```
if(Meteor.isServer){
  Meteor.methods({
    'sendLogMessage':function(){
      console.log('hello world');
    }
  });
}
```

Con `.isServer` estamos diciéndole a meteor que este código se ejecuta del lado del server.

No obstante, ese código podrá ser visto desde el cliente.

```
if(Meteor.isClient){
  'submit form': function(event){
    event.preventDefault();
    Meteor.call('sendLogMessage');
  };
}
```

Si deseas tener una aplicación segura se deberían remover los paquetes:

autopublish - insecure

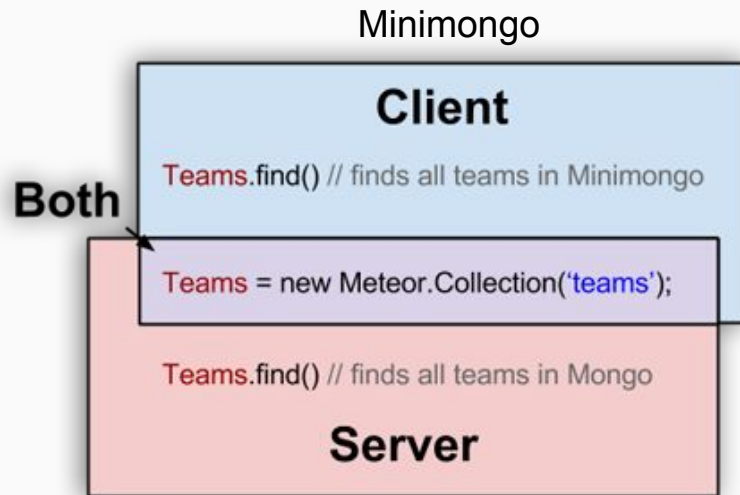
COLLECTIONS - MONGO

Sencillas de definir:

```
Teams = new Mongo.Collection('teams');
```

Las colecciones exponen los métodos:

- **insert**
- **update**
- **remove**
- **find**



TRACKER: Reactive programming

Dependency tracking system

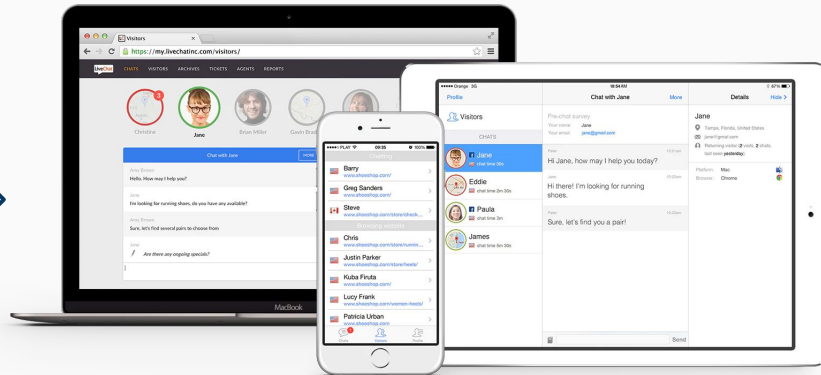
Changes

ReactiveVar / ReactiveDict

Sessions

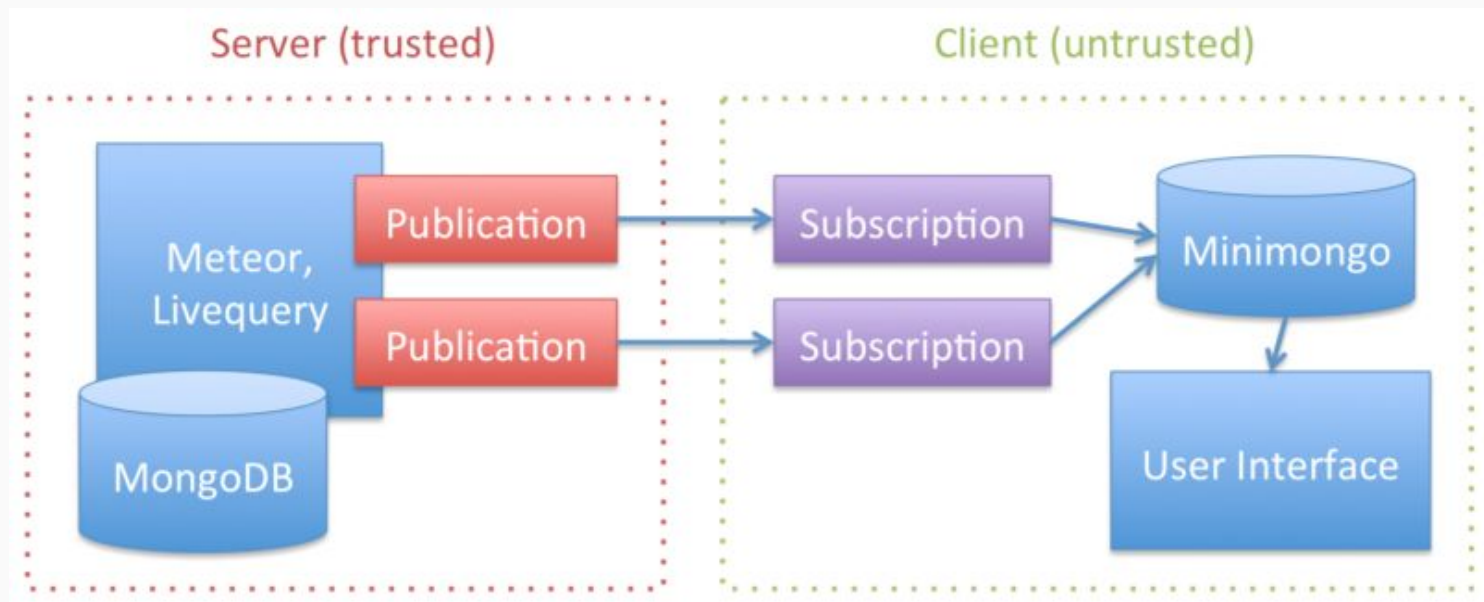
Render

Views/templates



```
Template.body.helpers({
  tasks: function(){
    return tasks.select().fetch();
  }
});
```

PUBLISH AND SUBSCRIBE



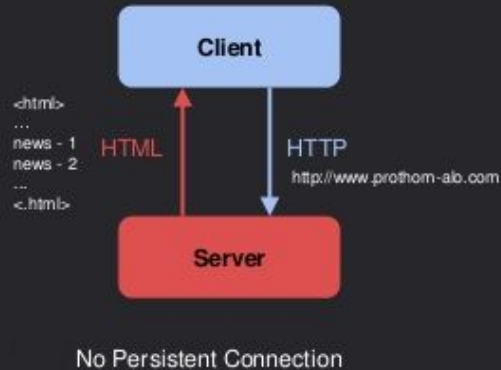
```
Meteor.publish('allowedData', function() {  
  return PlayersCollection.find();  
})
```

```
Meteor.subscribe('allowedData');
```

DDP → protocolo

Data on the Wire (Continue ...)

Traditional Web Architecture

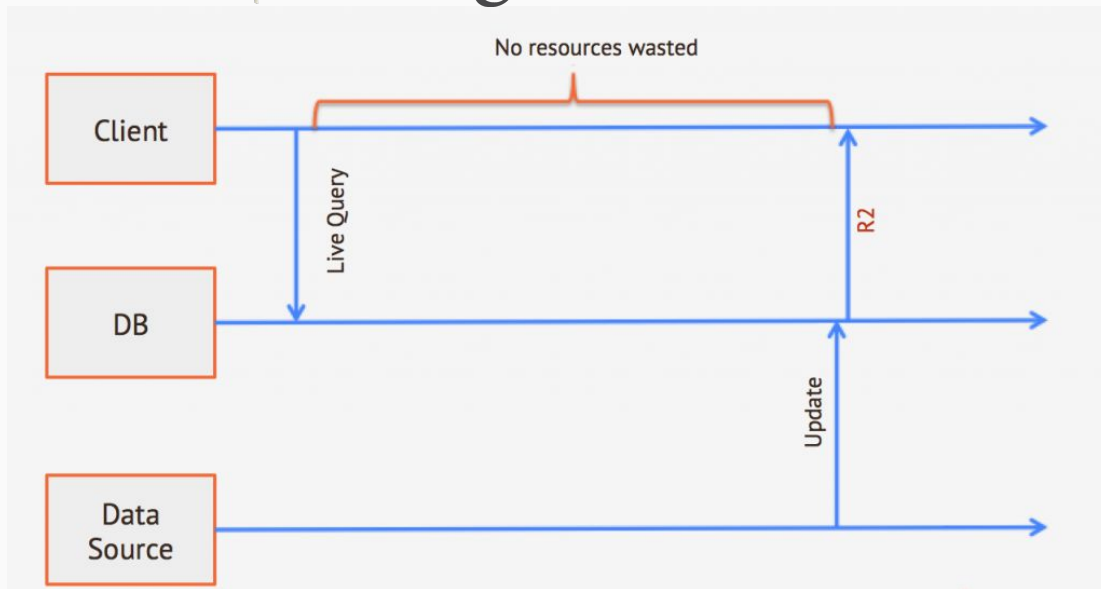


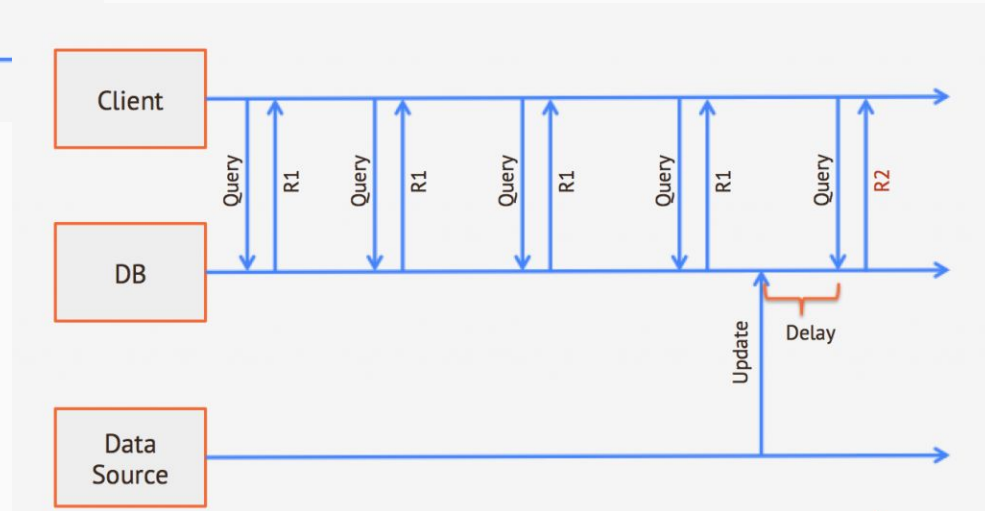
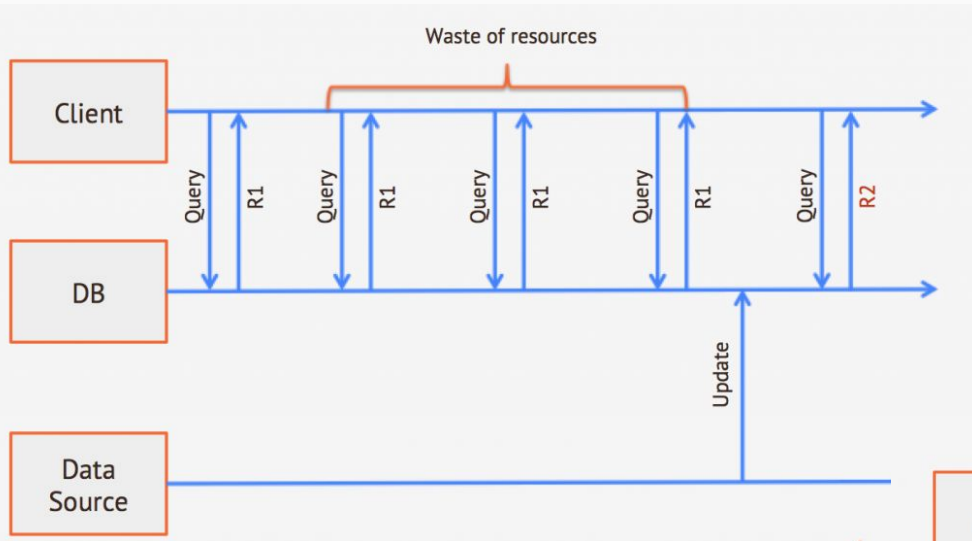
Modern Web Architecture



LIVEQUERY

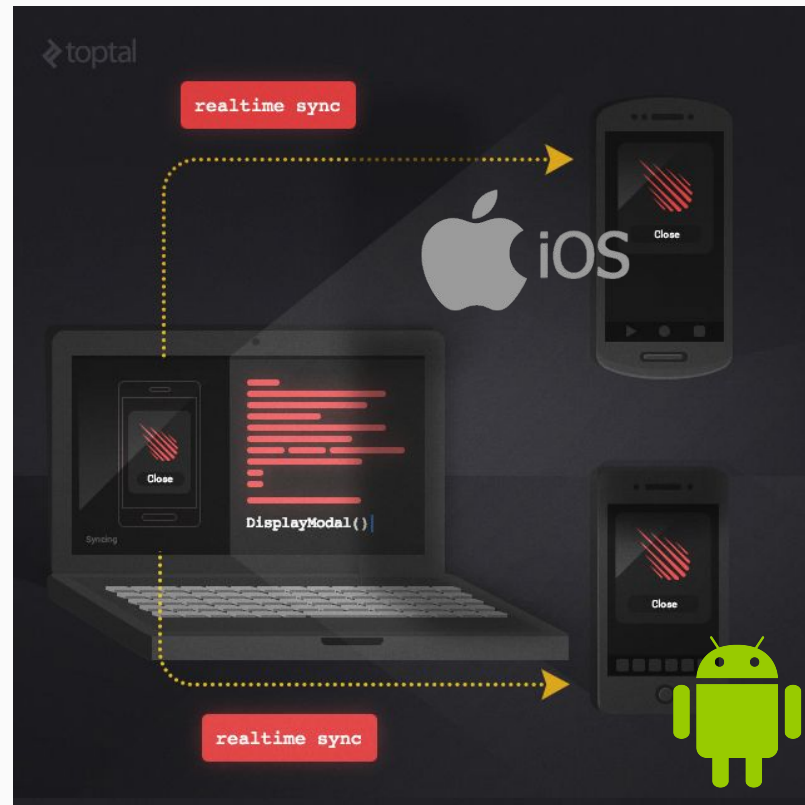
- You simply invoke a query to the server.
- It sends you the current result.
- When something changes in that query, the server sends the changes to the query.
- The client responds accordingly.





HOT PUSH CODE

Hot code push uses the autoupdate and reload packages to notify the client that new code is available and to automatically reload the browser.



IRON ROUTER



```
> meteor add iron:router
```

- Reactivity
- Redirect
- Hooks

```
Router.route('/', function () {  
  this.render('MyTemplate');  
});  
  
Router.route('/items', function () {  
  this.render('Items');  
});  
  
Router.route('/items/:_id', function () {  
  var item = Items.findOne({_id: this.params._id});  
  this.render('ShowItem', {data: item});  
});
```

BLAZEJS



- Es la biblioteca de renderizado reactiva que viene incorporada por defecto en Meteor.
- Fue desarrollada por el mismo equipo de Meteor.
- Trabaja con Spacebars Templates que están diseñadas para tomar ventaja de Tracker, el sistema reactivo de Meteor.
- No es obligatorio para los proyectos de Meteor.
- Es un paquete exclusivo de Meteor por ahora.

PRINCIPIOS



- Curva de aprendizaje suave.
- Reactividad transparente.
 - Blaze usa Tracker para saber cuando recalcular los helpers de cada template.
- Templates limpios.

CÓDIGO EJEMPLO



HTML

```
1 <template name="leaderboard">
2   <ol class="leaderboard">
3     {{#each players}}
4       {{> player}}
5     {{/each}}
6   </ol>
7 </template>
8 <template name="player">
9   <li class="player {{selected}}">
10     <span class="name">{{name}}</span>
11     <span class="score">{{score}}</span>
12   </li>
13 </template>
14
```

Javascript

```
1
2 Template.leaderboard.helpers({
3   players: function () {
4     // Perform a reactive database query against minimongo
5     return Players.find({}, { sort: { score: -1, name: 1 } });
6   }
7 });
8 Template.player.events({
9   'click': function () {
10     // click on a player to select it
11     Session.set("selectedPlayer", this._id);
12   }
13 });
14 Template.player.helpers({
15   selected: function () {
16     return Session.equals("selectedPlayer", this._id) ? "selected" : '';
17   }
18 });
19
```


SPACEBARS TEMPLATES



- Spacebar es un lenguaje de templates, construido sobre el concepto de renderear datos que están reactivamente cambiando.
- Estos templates se ven como html normales con “mustache” tags especiales delimitados por doble llaves `{{ }}`.

SPACEBARS TEMPLATES



- Contexto de datos:
 - `{{player.name}}` accede a la propiedad `name` del ítem `player` en el contexto actual de los datos.
- Llamada a Helpers con argumentos:

```
{{checkedClass player true 'checked'}}
```

- Llamada a Helpers con output de otro Helper:

```
{{checkedClass (isCheckedPlayer player)}}
```

SPACEBARS TEMPLATES



- Llamada de Helpers con argumentos via keywords (no muy usado):

`{{checkedClass todo noClass=true classname='checked'}}`

```
1  Template.Todos_item.helpers({
2    checkedClass(todo, options) {
3      const classname = options.hash.classname || 'checked';
4      if (todo.checked) {
5        return classname;
6      } else if (options.hash.noClass) {
7        return `no-${classname}`;
8      }
9    }
10  });
```

SPACEBARS TEMPLATES



- Inclusión de Templates:
 - Incluimos un sub-componente usando la sintaxis `{{ > player }}`.

- Helpers de atributos:

`<a {{attributes}}>My Link`

```
1  Template.foo.helpers({
2    attributes() {
3      return {
4        class: 'A class',
5        style: {background: 'blue'}
6      };
7    }
8  });
```

SPACEBARS TEMPLATES



- Rendering de HTML:
 - Podemos hacer render de HTML usando `{{{ myHtml }}}.`

```
1  Template.foo.helpers({
2    myHtml() {
3      return '<h1>This h1 will render</h1>';
4    }
5  });
```

SPACEBARS TEMPLATES



- Block Helpers:
 - Un block helper, que es llamado con `{{# }}` y se cierra con `{{/ }}`, es un helper que toma un bloque de html. Veremos un par de built-in block helpers.
- If / Else:
 - `{{#if condition }}`
 - `<p> It's true </p>`
 - `{{else }}`
 - `<p> It's false </p>`
 - `{{/if }}`

SPACEBARS TEMPLATES



- Each / Each in:
 - `{{#each players}}`
 - `{{ > player }}`
 - `{{/each}}`
- Let:
 - `{{#let name = player.firstName}}`
 - `<div>{{name}} is the #1 of the leaderbord!</div>`
 - `{{/let}}`

SPACEBARS TEMPLATES



- Un par de consideraciones finales:
 - Spacebars tiene un parser de HTML bien estricto, el que a veces puede pedir cerrar etiquetas de html que normalmente no es obligacion (como `<p>`)
 - Para insertar llaves (`{ }`) usamos el caracter `|` .

RE-RENDERING



- Lo principal que se debe entender de como Blaze hace re-render es que este ocurre al nivel de las inclusiones de helpers y templates. Cada vez que el contexto de datos de un componente cambia, necesariamente tiene que volver a correr todos los helpers y así volver a hacer el render.
- En general Blaze está optimizado para que uno no se preocupe por esto.
- Existen maneras para controlar el re-rendering.

BLAZE O REACT?



- Pros:
 - Blaze es creado por Meteor y lleva más tiempo, por lo que tiene más soporte.
 - Su gran facilidad de uso.
- Contras:
 - Poca flexibilidad vs. React
 - Algunos dicen que eventualmente todo se pasará a React

Conclusión: Depende del proyecto

TESTING

```
meteor test --driver-package practicalmeteor:mocha
```



CHIMP

```
describe('Tasks', () =>{  
  describe('methods', () =>{  
    beforeEach(() => {  
      /*Acciones antes del test, por  
      ejemplo borrar o inicializar bd*/  
    });  
    it('can delete owned task', () => {  
      /* Test */  
      assert.equal(Tasks.find().count(),0);  
    });  
  });  
});
```

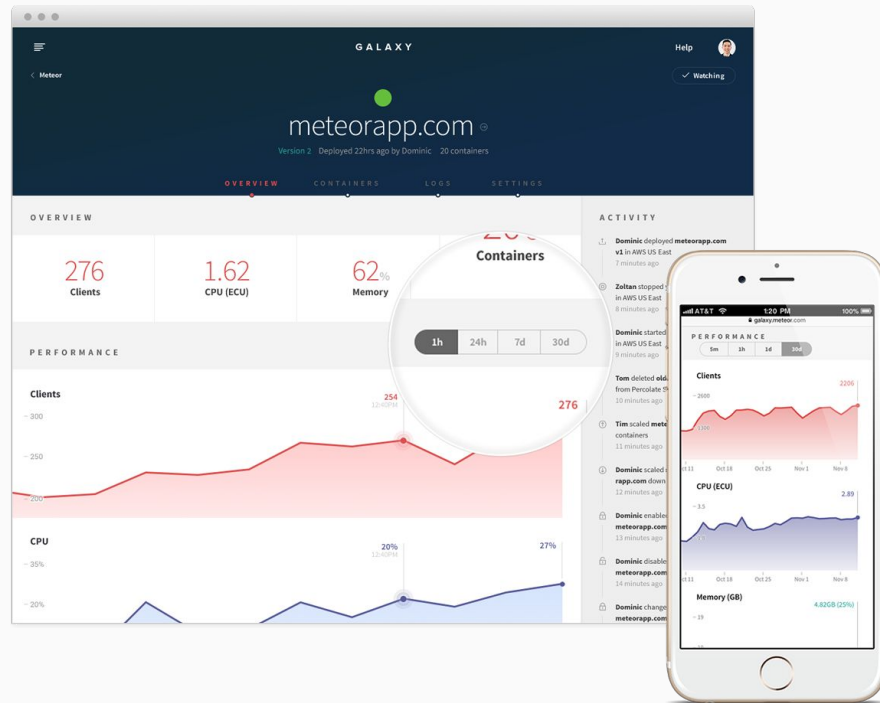
Meteor.isTest()

archivos *.test(s)

DEPLOYMENT



```
meteor deploy APPLICATIONNAME.meteor.com
```



LO MALO



- Just realtime apps?
- SQL support
- Rendering server side



CLASSCRAFT

ROADMAP

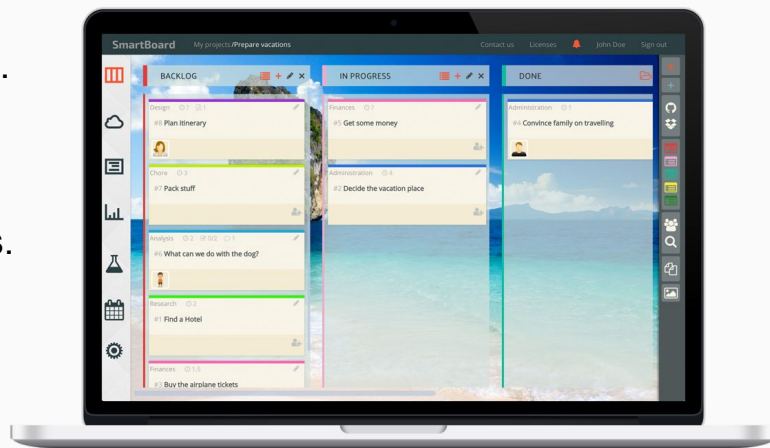
<https://github.com/meteor/meteor/blob/devel/Roadmap.md>



PostgreSQL

APRECIACIONES

- Es muy fácil iniciar a trabajar con Meteor.
- En Windows no hubo una buena experiencia.
- Finalmente Meteor es NodeJS por detras.
- Instalación de Mongo previamente complica las cosas.
- Empresa con buen capital de inversión.
- No hay muchos casos de éxito con empresas grandes.
- Un buen ejemplo de cómo se podría usar Meteor es viendo el comportamiento de Trello o SmartBoard.



DEMO TIME

