



@killfil

# Cocinando con node...

Introducción a node.js

# Lo obvio...

- Node.js =
  - Node: Nodo. Un nodo de la red, que provee un servicio
  - + JS: Javascript
- “Una herramienta que permite hacer un servicio que se comunica con otros elementos de un entorno de red”



# Para que sirve

- Hacer servicios
- Basados en eventos
- En Javascript

# Para que sirve harto

- Hacer servicios
  - Es bueno manejando ES
  - Es malo procesando datos

| Acceso | Distancia          |
|--------|--------------------|
| CPU    | 15 ciclos          |
| RAM    | 250 ciclos         |
| Disco  | 41.000.000 ciclos  |
| Red    | 240.000.000 ciclos |

# Para que sirve harto

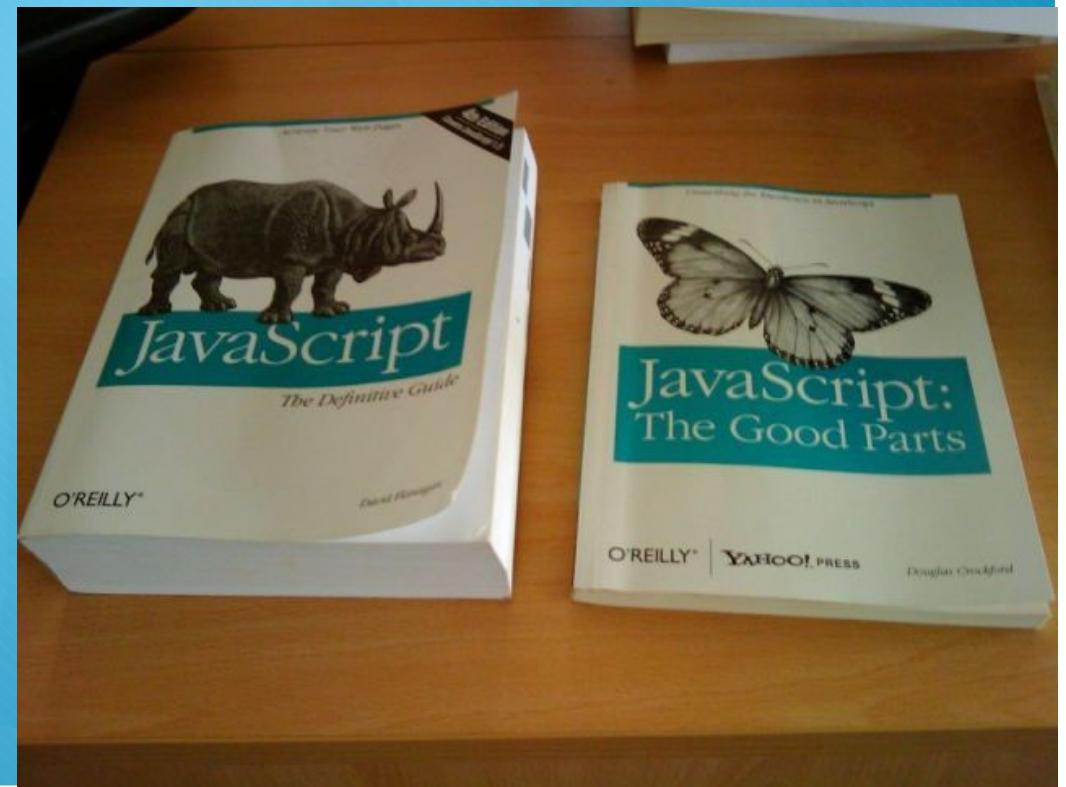
- Basados en eventos
  - Concurrency simple
  - Paradigma distinto

Select \* from tabla <con sus buenos joins>



# Para que sirve harto

- En javascript
  - Muy accesible
  - Muy accesible



# Lo bueno

- Es bueno manejando ES / Concurrencia simple:

Versión sincrónica:

Query1 = “SELECT \* FROM GRAN\_TABLA\_1 con JOINES”

Query2 = “SELECT \* FROM GRAN\_TABLA\_2 con JOINES”

Seq1 = { Query1; dibuja el resultado(threadsafe) }

Seq2 = { Query2; dibuja el resultado(threadsafe) }

New Thread(Seq1).Start()      GUI  
New Thread(Seq2).Start()

# Lo bueno

- Es bueno manejando ES / Concurrencia simple:

Versión asincrónica:

Query1 = “SELECT \* FROM GRAN\_TABLA\_1 con JOINS”

Query2 = “SELECT \* FROM GRAN\_TABLA\_2 con JOINS”

Query1.onReady { dibuja el resultado }

Query2.onReady { dibuja el resultado }

GUI

# Lo malo

## ○ Malo procesando datos / Paradigma distinto

```
Function PI() {  
    var pi = 0;  
    while (!pi) pi += 0.01;  
    return pi;  
}  
Inicia_la_interfaz_gui()  
Escucha_en_un Puerto()  
onClick = {  
    label = PI();  
}
```

“Es lento pero rápido”

# Lo malo

## ○ Malo procesando datos / Paradigma distinto

```
....  
servidor.on('conexion', function(cliente) {  
    elDato;  
    cliente.on('datos', function(dato) {  
        elDato = dato  
    })  
    con = conexión_remota()  
    con.send(elDato)  
    con.close()  
})
```

# Hibernación

- A) `baseDatos.query(SQL, avisame);`
- B) `promesa = baseDatos.query(SQL);`

`Promesa.on('termino', <hurra>);`

“Ejecuta el SQL, mientras tanto haz lo que sea, pero avísame cuanto termines.”

“Llevame donde esta Alien, y cuando llegues despiértame”

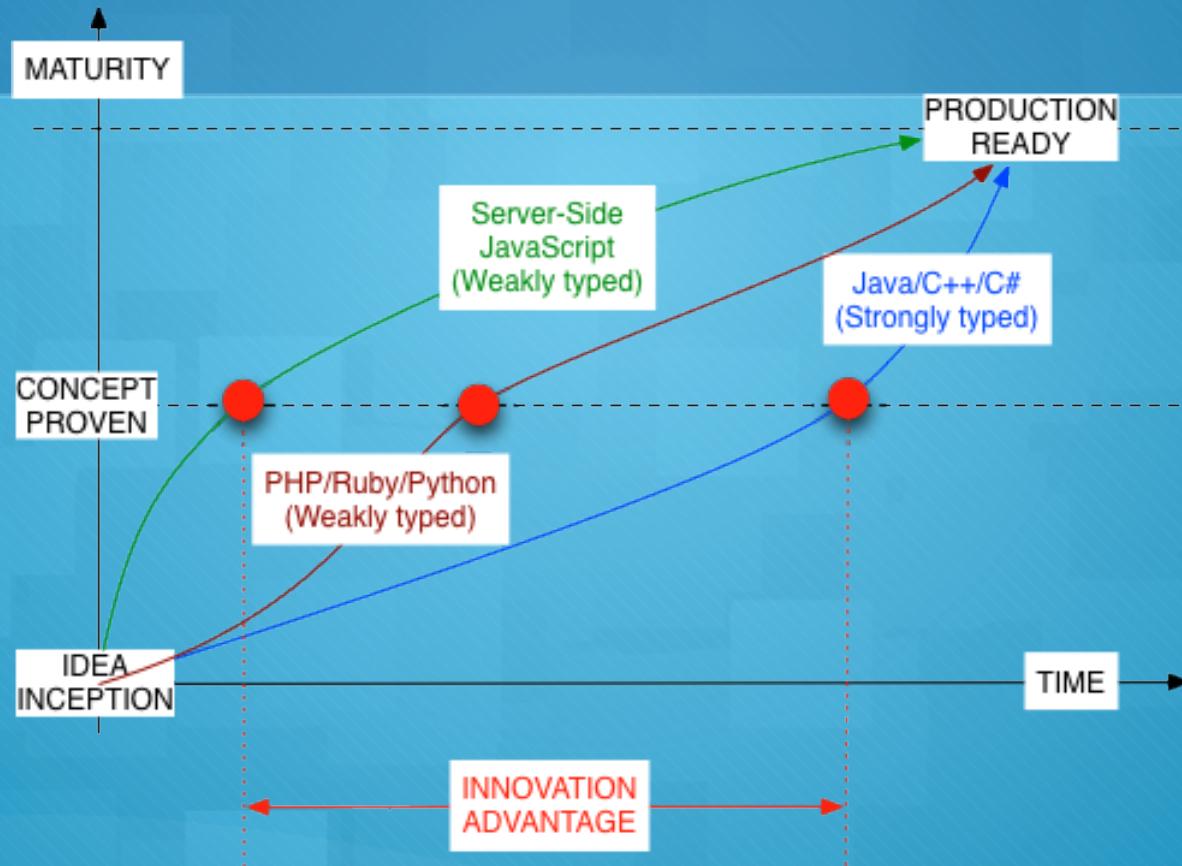




# Porque es interesante

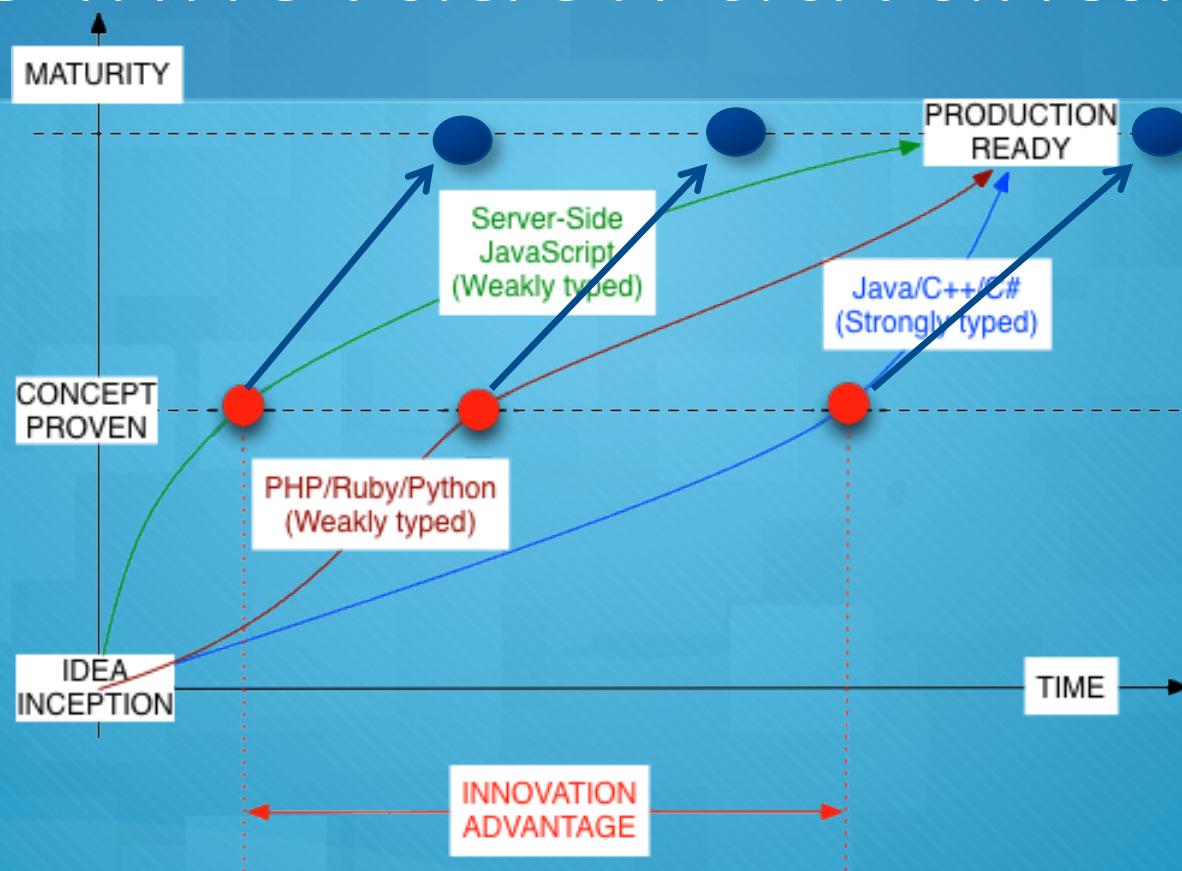
- Construido sobre hombros de gigantes (V8)
- Fácil de aprender (curva de aprendizaje baja)
- Comunidad participativa
- Muchas librerías, fácilmente instalables, y son todas asinc!
- Ha crecido mucho, pero tiene mucho que crecer
- Es rápido para desarrollar, no deja equivocarse mucho...

# The innovation advantage



<http://www.olympum.com/architecture/the-nodejs-innovation-advantage/>

# The innovation advantage



<http://www.olympum.com/architecture/the-nodejs-innovation-advantage/>