

# Presentación

**autores:**

- Defossé, Nahuel
- van Haaster, Diego Marcos

**tutor:** Saenz Lopez, Marta

**fecha:** 15 de Diciembre 2009

**url:** <https://code.google.com/p/protopy>

# Objetivos Principal

"Objetivo Principal"

## Objetivos Secundarios

- **Open Source**

Coste de licenciamiento nulo y aseguramiento de la continuidad

- **Multiplataforma**

Windows, Linux, Mac y móviles (dode exista un browser)

- **Adaptación mínima de aplicaciones existentes**

Integración con un Frameworks Web

- **Facilidad de utilización**

Reutilizar los conceptos/patrones del framework para una rápida asimilación de lo desarrolladores.

# Carencias del Browser

- Base de datos
- Servidor web
- Lenguaje de programación (consistente)
- Concurrencia
- Conectividad con el entorno del cliente

# Tecnologías Existentes (1)

- **Silverlight (.NET Framework, Microsoft)**  
Solución muy interesante, pero **cerrada**. *Promesa* de no cautividad.
- Otras alternativas

AIR, JavaFX, XUL, etc.

# Tecnologías Existentes

- JavaScript!
  - Objetos
  - Expresiones Regulares
  - Patrones propios (Module, Closures)
- Librerías:

Prototype Dojo Peppy

## Tecnologías Existentes (2)

- **Google Gears**

**Añade al navegador 3 componentes**

- **Local Server**

Un servidor de archivos locales

- **DataBase**

Una base de datos transaccional

- **Worker Pool**

Sistema de Hilos con pasaje de mensajes

# JavaScript 1.7

- **Generador:**

Ej: Recuperación perezosa de datos.

- **Orientación a objetos:**

Métodos útiles como `__noSuchMethod__`, `__defineGetter__` y `__defineSetter__`.

- **Azucar sintáctico:**

Asignación múltiple



# Framework

- Inversión de Control
- Comportamiento por defecto definido
- Extensibilidad
- **No modificabilidad del código del framework**  
Todo se extiende, nada se edita

# Django

Framework web elegido por:

- **Simple**

Escrito en Python Sin XML/YAML/INI/<ponga su markup aquí> Pocas capas M T V

- **DRY**

ModelForms, Vistas genéricas, ContentType framework

## Django (2)

- **Con baterías incluidas**  
Administración, Geolocalización
- **Open Source**  
Gran comunidad, una fundación, varios diarios utilizándolo
- **Comportamiento transversal**  
Middlewares

## Django (3)

Un **proyecto** Django es un paquete con los siguientes módulos:

`"settings.py"` y `"urls.py"`

Una **aplicación** es un subpaquete con los siguientes módulos:

`"models.py"`, `"views.py"` y `"urls.py"` (opcional)

# Django (4)

Templates:

- Poca lógica

`for`, `if`, formateo

- Modularidad

Herencia e inclusión

- Template Tags

Funcionalidad extra en funciones planas

# Django (5)

Ciclo del Request:

Gráfico

# Arquitectura

Acá va el gráfico



# Protopy

Soporte para implementación de Django sobre el navegador.

- Aprovechamiento de JavaScript 1.7
- Módulos
- OO Pythonica, `type`
- Integración de librerías:
  - API de DOM de Prototype

- Eventos (Dojo)
- CSS Selector (Peppy)
- **Gears**  
Enmascaramiento en el módulo `sys`, extensión de DB, Desktop.

# Arquitectura (Doff)

Ahora mostramos Doff

# Doff

Django on Protopy -> **Djano Offline** :)

- API de Modelos
- Templates
- Proyecto desconectado
- **Aplicaciones adicionales**  
Sincronización Autenticación "Sesión"

# Arquitectura (Offline)

Ahora mostramos Offline

# Offline

Soporte en el servidor para proyecto desconectado

- Comandos de administración
- Instalación
- Seguridad
- Sincronización

# Demo

La aplicación de demostración es un agente de ventas.

# Conclusiones

Se lograron cumplir todos los objetivos.



# Lineas Futuras

Tomar títulos

FIN

**FIN**

¡Muchas Gracias!