

WEB & MOBILE APPLICATIONS

Introduzione al corso

Outline

- Argomenti del corso
- Concetti base



Chi sono

Stefano Bianchini

mail: stefano.bianchini@simplenetworks.it

Founder & Mentor
Simplenetworks SRL

<https://www.simplenetworks.it>

<https://www.linkedin.com/in/stefanobianchini/>



Argomenti del corso

Prerequisiti

- Conoscenza del linguaggio PHP, a oggetti
- Conoscenza del funzionamento di un web server
- Conoscenza di SQL



Cosa affronteremo nel corso

- La progettazione di applicativi web complessi
- La separazione tra frontend (client) e backend (server)
- I concetti necessari per utilizzare un framework PHP moderno
 - un paio di design pattern
 - i namespace
 - le funzioni anonime
 - composer
- Laravel (<https://laravel.com/>)
- Il formato JSON
- REST(ful) API

Perchè questo corso

- Per lo sviluppo di applicativi web complessi, è necessario focalizzarsi su:
 - scalabilità
 - stabilità
 - resilienza
 - velocità di intervento

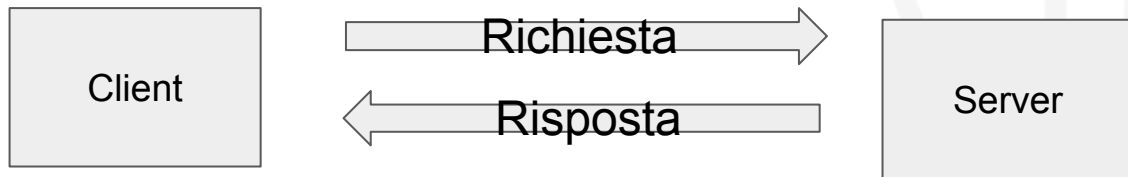
Abbiamo quindi bisogno di strumenti che ci permettano una progettazione con le caratteristiche appena citate



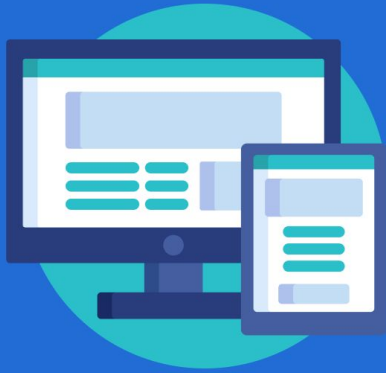
Concetti base

Request e Response

- In ambito web, tutte le operazioni effettuabili si riducono ad un insieme di richieste e risposte.
- Le richieste vengono effettuate dal client, le risposte vengono preparate dal server ed inviate al client
- Le risposte dipendono dalla richiesta effettuata



Frontend & backend



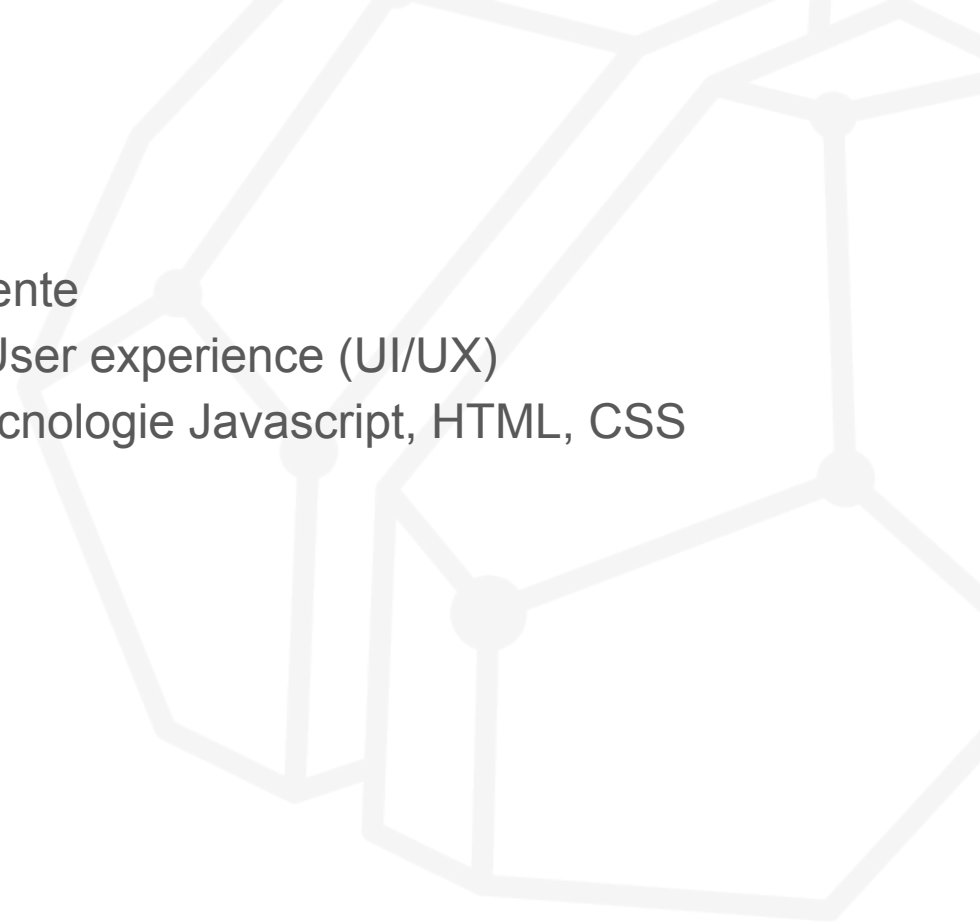
FRONT-END



BACK-END

Frontend & backend

- è la parte con cui può interagire l'utente
- si concentra sulla User interface e User experience (UI/UX)
- in caso di frontend web, sfrutta le tecnologie Javascript, HTML, CSS
- può essere un'applicazione mobile

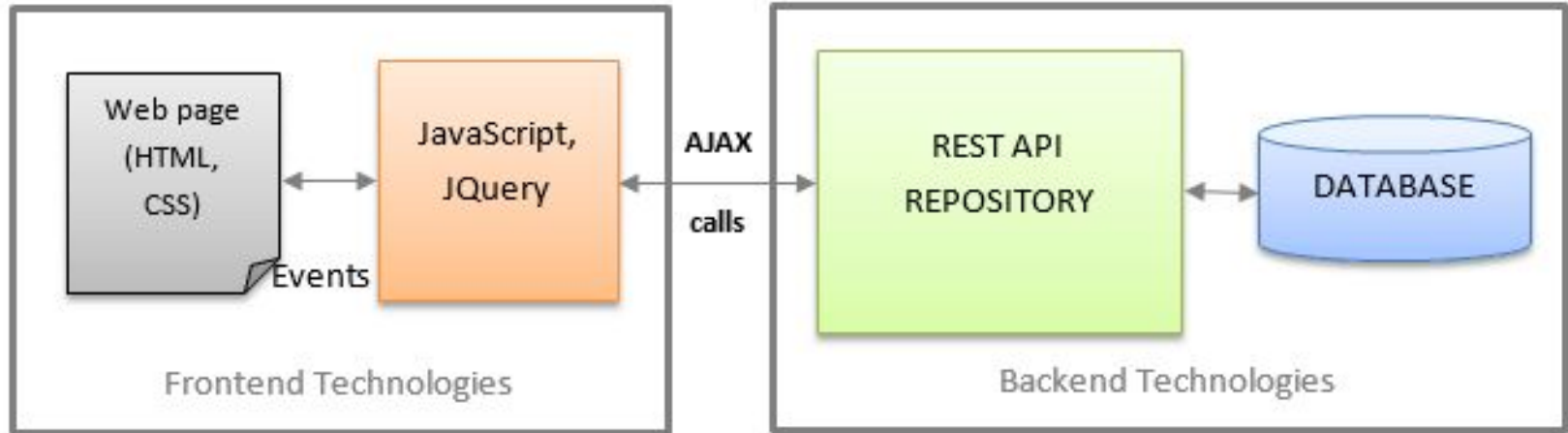


Frontend & **backend**

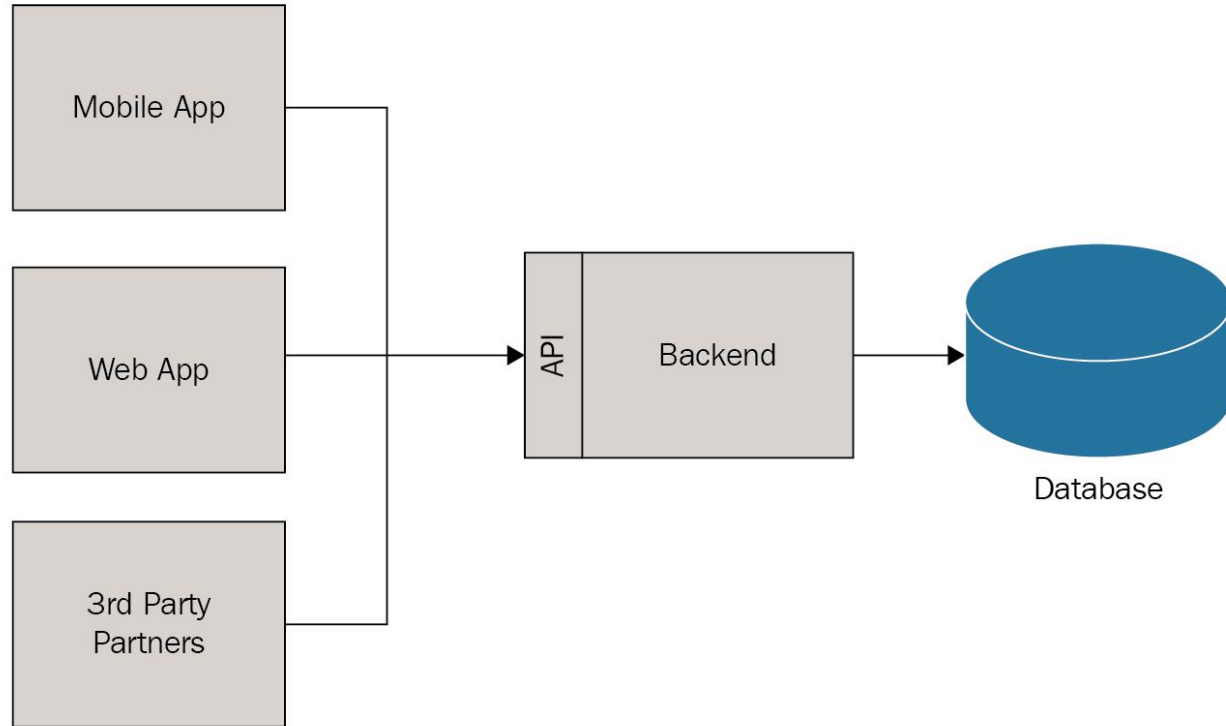
- si occupa di gestire ed elaborare i dati ricevuti dal frontend
- espone ed esegue le funzionalità per cui è stato creato
- utilizza tecnologie quali PHP, NodeJS ecc.



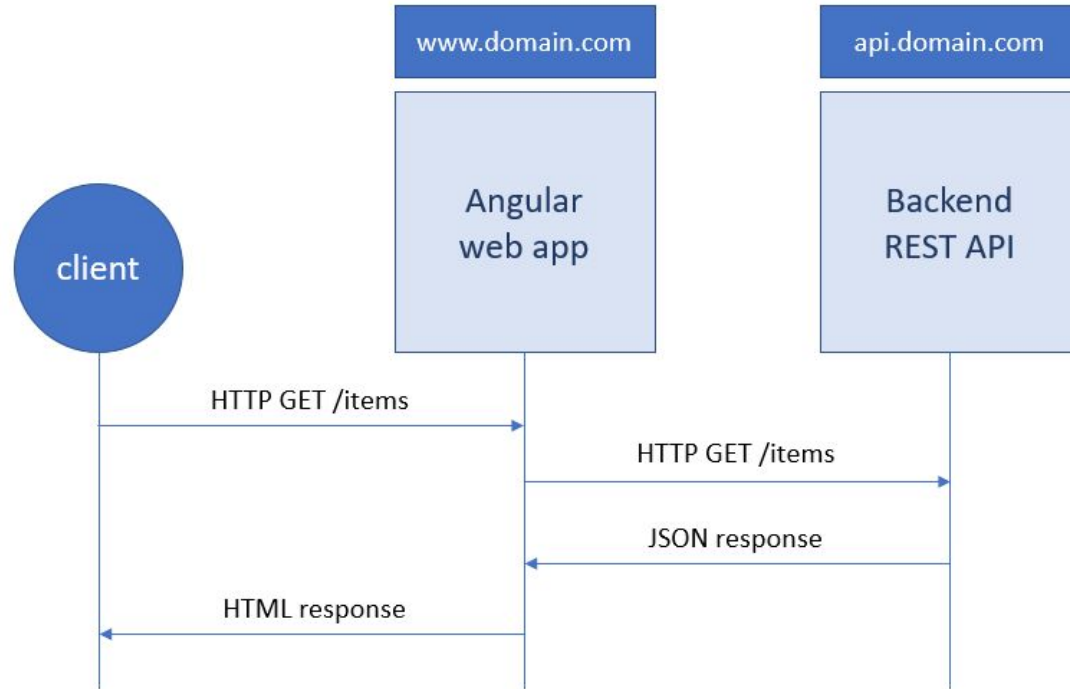
Frontend & backend



Frontend & backend



Frontend & backend



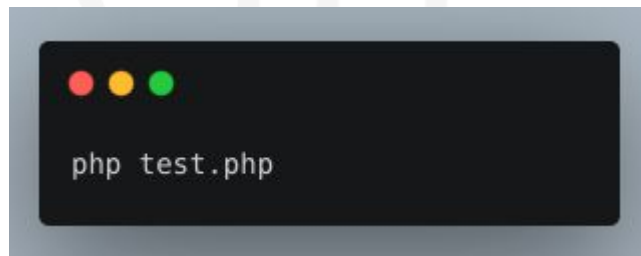
PHP a linea di comando (**php-cli**)

PHP è un interprete e può funzionare indipendentemente da un web server.

Questo significa che è possibile richiamarlo da console (linea di comando) come un eseguibile qualsiasi, passando come parametro il nome del file .php da processare

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It contains PHP code for a test script.

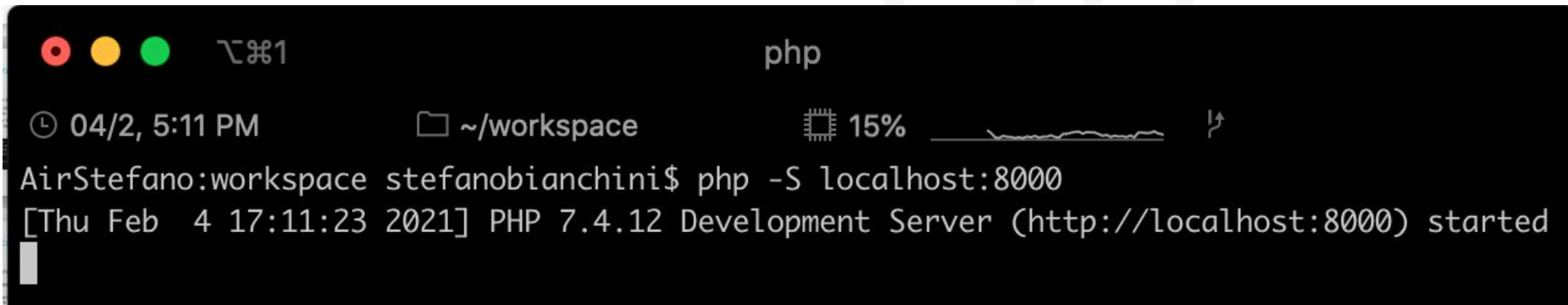
```
<?php
//test.php
echo 'questo è un test';
```

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It shows the command to execute the PHP script.

```
php test.php
```


Web server integrato

- Non serve un web server completo (Nginx, Apache) per poter sviluppare in PHP: basta l'interprete installato
- Dalla versione 5.4 PHP incorpora uno snello web server
- Per avviare il server, basta eseguire il seguente comando dal terminale posizionandosi nella cartella del progetto ed eseguire
`php -S localhost:8000`



```
php

04/2, 5:11 PM  ~/workspace  15%

AirStefano:workspace stefanobianchini$ php -S localhost:8000
[Thu Feb 4 17:11:23 2021] PHP 7.4.12 Development Server (http://localhost:8000) started
```

I framework

Don't Reinvent



Perfect It

- **Perché reinventare ogni volta la ruota?**
- I framework sono software che comprendono in un unico pacchetto l'intero set di librerie e strumenti utili alla progettazione di un sito, un'applicazione web o qualsiasi altro servizio online sia necessario creare con PHP o con un altro linguaggio di programmazione.

Design e architectural pattern

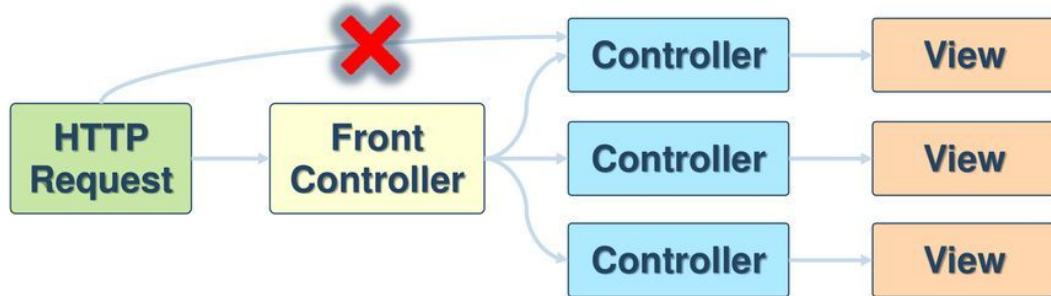
- Sono approcci ben definiti relativi alla progettazione software
- Soluzioni generali e riutilizzabili a problemi comuni
- Modelli architetturali
 - Sono design pattern che hanno un ambito più ampio, ovvero descrivono un pattern complessivo adottato dall'intero sistema
- Se si utilizza un framework la maggior parte del codice di alto livello e la struttura del tuo progetto saranno basati su quel framework, quindi molte delle decisioni riguardanti i pattern vengono fatte al posto dello sviluppatore.

Front controller pattern

- È la creazione di un unico punto di ingresso per la gestione delle richieste
- In base alle richieste specifiche crea delle istanze di altri oggetti e chiama altri metodi per gestire specifiche operazioni.
- Viene quindi richiamato SOLO index.php (o similare)
 - <http://www.example.com/index.php/clienti>
 - <http://www.example.com/index.php/cliente/12>
 - <http://www.example.com/index.php/modifica/cliente/12>
- Impostando bene il Web Server index.php può sparire dall'url (ad esempio, Wordpress)
 - <http://www.example.com/clienti>
 - <http://www.example.com/cliente/12>
 - <http://www.example.com/modifica/cliente/12>

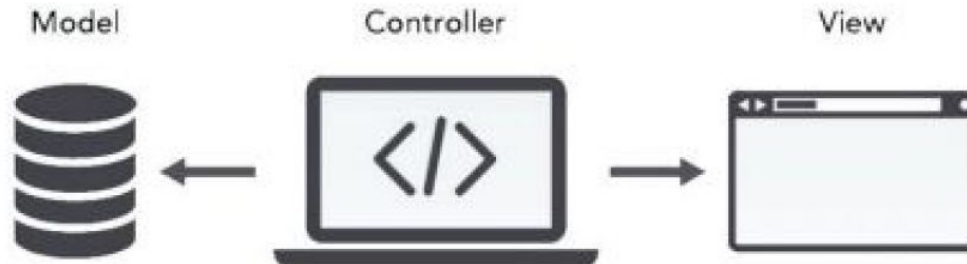
Front controller pattern

- L'index.php (o chi per lui) ha il compito quindi di eseguire le funzioni necessarie per ottemperare alla richiesta.
- Cambia la url, cambia la richiesta, cambia la risposta
- Il front controller pattern si occupa, come un “centralinista”, di collegare la richiesta effettuata dal client con la giusta risposta ottenuta da una determinata funzione. Come? Con delle tabelle di instradamento (routing)



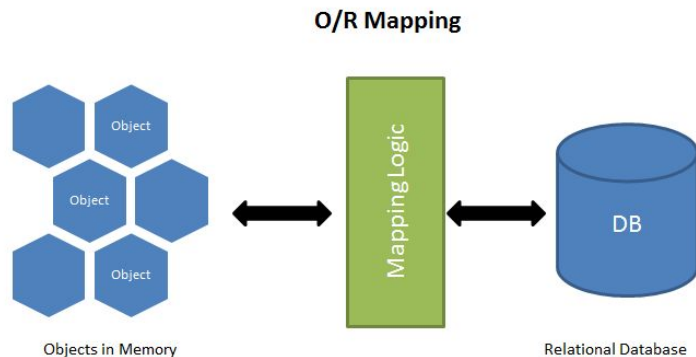
MVC

- MVC è un pattern architetturale usato per sviluppare applicazioni complesse, che richiedono diversi livelli di interazione.
- Si applica utilizzando il principio della separazione dei contesti, dividendo l'applicazione in tre aree:
 - Model: incapsula i dati e le loro funzionalità
 - View: si occupa della presentazioni dei dati (ce ne possono essere diverse in una applicazione)
 - Controller: elabora l'input dell'utente e le richieste provenienti dal model per produrre nuove view.



ORM

- L'Object-Relational Mapping (ORM) è una tecnica che, mediante un'interfaccia orientata agli oggetti, fornisce servizi di interfacciamento (lettura, persistenza) dati verso un RDBMS (Relational DataBase Management System)
- In parole povere, effettua una mappatura in memoria (nel formato ad oggetti del linguaggio utilizzato) dei dati di un database e ne consente la lettura, la creazione, l'eliminazione e l'aggiornamento



Namespace

La comunità PHP ha molti sviluppatori che creano molto codice. Questo significa che il codice PHP di una libreria potrebbe usare lo stesso nome di un'altra per una classe. Quando entrambe le librerie vengono usate nello stesso namespace, potrebbero collidere e causare problemi.

I namespace risolvono questo problema.

Come descritto nel manuale di PHP, si può pensare ai namespace come directory del sistema operativo che dividono i file; due file con lo stesso nome possono esistere in directory separate. Allo stesso modo, due classi PHP con lo stesso nome possono esistere in namespace PHP separati.

Namespace

È importante inserire il codice in un namespace in modo che possa essere usato da altri sviluppatori senza paura che esso collida con altre librerie.

Un modo raccomandato di usare i namespace è delineato nel [PSR-4], che mira a fornire una convenzione standard per la i nomi di file, classi e namespace, in modo da consentire la scrittura di codice plug-and-play.

I framework più famosi e recenti impongono l'utilizzo dei namespace

Namespace - un esempio pratico

```
<?php
//Car1.php
namespace Audi;

class Car {
    public function toString() {
        echo 'Questa è una Audi'.PHP_EOL;
    }
}
```

```
<?php
//Car2.php
namespace Volvo;

class Car {
    public function toString() {
        echo 'Questa è una Volvo'.PHP_EOL;
    }
}
```

Namespace - un esempio pratico



```
<?php
//test.php
include('Car1.php');
include('Car2.php');

$volvo = new Volvo\Car();
$volvo->toString();

$audi = new Audi\Car();
$audi->toString(); }
}
```



```
<?php
//test2.php
include('Car1.php');
include('Car2.php');

use Volvo\Car as MyVolvo;
use Audi\Car as MyAudi;

$volvo = new MyVolvo();
$volvo->toString();

$audi = new MyAudi();
$audi->toString();
```

Template engine

- Il beneficio principale derivante dall'uso dei template è la separazione che creano tra la logica di presentazione e il resto della tua applicazione. I template hanno la sola responsabilità di visualizzare contenuto formattato.
- Non sono responsabili per il recupero o la persistenza dei dati o altri compiti più complessi.
- Questo porta alla scrittura di codice più pulito e più facile da leggere, particolarmente utile in un team dove gli sviluppatori lavorano sul codice server-side (controller e modelli) e i designer lavorano sul client-side (markup).
- I template engine utilizzano uno pseudo-linguaggio di programmazione creato ad hoc per creare cicli, blocchi if/then/else, variabili ecc.



Q & A