Ocsigen

Développement d'applications multi-plateforme Web et mobiles

be sport®

Vincent Balat

Open Source Innovation Spring — Systematic — 28 juin 2016

Développement d'une application Web et mobile

Web

Développement d'une application Web et mobile

Android

Web

be sport®

ocsigen
fresh air in web programming

Développement d'une application Web et mobile

Android

Web

iOS

be sport®

ocsigen
fresh air in web programming

Développement d'une application Web et mobile

Android

Web

iOS

Windows

be sport®

ocsigen
fresh air in web programming

Développement d'une application Web et mobile

Android

Web

iOS

Windows

Tablettes, smartphones...

be sport®

ocsigen
fresh air in web programming

Développement d'une application Web et mobile

Android

Web

iOS

Windows

Tablettes, smartphones...

Développer une application demande de gros moyens...

besport®

ocsigen
fresh air in web programming

La solution HTML5

Application HTML5 + JS

+ Webview (+ Cordova/Phonegap)

+ responsive design

La solution HTML5

Application HTML5 + JS
+ Webview (+ Cordova/Phonegap)
+ responsive design

Mais :

- Difficile de se passer de deux versions (Web et mobile)

  (sauf si l'on abandonne la génération de pages côté serveur)

- Langage JavaScript

Application multi-plateforme Ocsigen

- Un seul code pour Web et mobile
- Un langage de haut niveau (OCaml), compilé et typé statiquement
- Pages générées côté client ou serveur selon les besoins

besport®

ocsigen
fresh air in web programming

BeSport

Lex Fleming invited you to this event. RESPOND FOLLOW

07:00 – 08:00, Sunday 10 Apr 2016

Stade Yves-du-Manoir, 12 Rue François Faber, 92700 Colombes

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam tempor justo congue, bibendum velit interdum. Quisque felis magnorem ipsum dolor orem ipe ... Read more

Related to: **Friendly soccer season 2016**

Write a post...

Posts | Media

Nothing yet! **Add a post**

20 Participating

+16

2	20	25
Maybe	Pending	Followers

Latest Photos

Similar Events

12 APR

Soccer at the mount
Created by **Anna Bull**
Soccer
VIEW

besport

Anna Bananya

Vincent Le Visitor

Anna **Bananya**

23 years old | Paris, France

24 shared connection points >

Tennis P | Rugby | Soccer | Basketball | American Football | Badminton | +23

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam tempor justo congue, bibendum velit interdum. Quisque felis magnorem ipsum dolor orem ipe ... Read more

+ CONNECT | FOLLOW | • • •

430 Connections | **34** Followers

459 | 23k | 3 | 2503 | 208

Featured Achievements

2 - 1
Anna Bananya vs Georgia Stai...
Saturday 6 January, 2014
Tennis

Semi-finalist
Australian Open
Friday 5 January, 2014
Tennis

Feed | Events | Network | Media | Achievements

Lex Fleming posted:
Today at 17:29

Can't wait for the next game of footy, let's go!

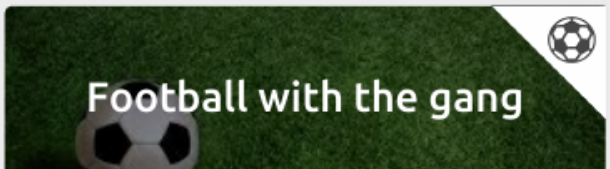
25 10 5

Good sport | Comment | Share

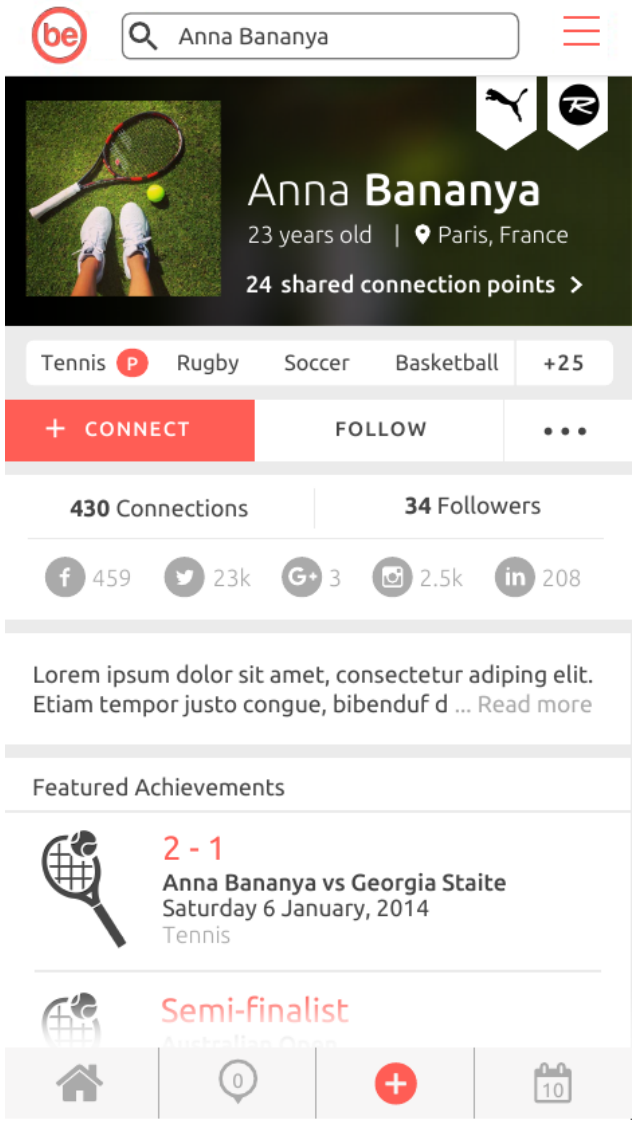
Lex Fleming added an image:
Today at 14:34

Next Events

Following | Today 19:30



BeSport : Application mobile



Le projet Ocsigen

Un ensemble d'outils pour le développement Web en OCaml.

créé en 2004, plus de 200 000 lignes de code, LGPL

Auteurs :

Vincent Balat, Jérôme Vouillon,
Gabriel Radanne, Vasilis Papavasileiou, Hugo Heuzard, Élie Canonici-Merle,
Pierre Chambart, Grégoire Henry, Benedikt Becker, Benjamin Canou, Boris Yakobowski,
Jérémie Dimino,
Charly Chevalier, Enguerrand Decorne, Romain Calascibetta, Jacques-Pascal Deplaix, Grégoire Lionnet, Raphaël Proust, Stéphane Glondu, Jérôme Maloberti, Gabriel Kerneis, Arnaud Parant, Christophe Lecointe, Denis Berthod, Gabriel Cardoso, Piero Furiesi, Jaap Boender, Baptiste Strazzulla, Thorsten Ohl, Gabriel Scherer, Séverine Maingaud, Simon Castellan, Jean-Henri Granarolo, Archibald Pontier, Nataliya Guts, Cécile Herbelin, Charles Oran, Jérôme Velleine, Pierre Clairambault …

université PARIS DIDEROT PARIS 7

cnrs

besport®

AGENCE NATIONALE DE LA RECHERCHE ANR

IRILL

Inria INVENTEURS DU MONDE NUMÉRIQUE

Le langage

OCaml

- Très expressif : *fonctionnel, orienté objet, modules paramétrés …*
- Système de types très riches (Typage statique !)
- Compilé (rapide)
- Syntaxe extensible
- Beaucoup de bibliothèques et bindings
- Communauté d'utilisateurs

Qui utilise OCaml ?

Jane Street Capital

Facebook

Xen – Citrix 15% of cloud servers, (including Amazon Web Services)

OCaml Pro (Paris)

OCaml Labs (Cambridge UK)

Most **program provers**

Airbus, Microsoft (F#), Dassault, Docker, Bloomberg, Lexifi...

...

Ocsigen : principales caractéristiques

- Une application **client-serveur** 100% en OCaml
  - ⇝ Un seul langage pour les développeurs
  - ⇝ Génération des pages côté serveur ou client
  - ⇝ Communication client ⇄ serveur très simple
  - ⇝ Persistance de l'application client

Ocsigen : principales caractéristiques

- Une application **client-serveur** 100% en OCaml
  - ⇝ Un seul langage pour les développeurs
  - ⇝ Génération des pages côté serveur ou client
  - ⇝ Communication client ⇄ serveur très simple
  - ⇝ Persistance de l'application client
- **Typage statique** pour vérifier de nombreuses propriétés
  - ⇝ Fiabilité
  - ⇝ *Refactoring* du code facile
  - ⇝ Très peu de tests unitaires à écrire

Ocsigen : principales caractéristiques

- Une application **client-serveur** 100% en OCaml
  - ⇝ Un seul langage pour les développeurs
  - ⇝ Génération des pages côté serveur ou client
  - ⇝ Communication client ⇄ serveur très simple
  - ⇝ Persistance de l'application client
- **Typage statique** pour vérifier de nombreuses propriétés
  - ⇝ Fiabilité
  - ⇝ *Refactoring* du code facile
  - ⇝ Très peu de tests unitaires à écrire
- **Js_of_ocaml : Compilateur OCaml vers Javascript**
  - ⇝ La puissance d'OCaml sur le navigateur
    - – Code efficace
    - – Interaction avec JS simple
    - – Compile n'importe quel programme bytecode OCaml

Ocsigen : principales caractéristiques (2)

- **Services**
  - Programmation Web traditionnelle très simplifiée
  - Programmation Web avec continuations (services dynamiques)

Ocsigen : principales caractéristiques (2)

- **Services**
  - Programmation Web traditionnelle très simplifiée
  - Programmation Web avec continuations (services dynamiques)
- **Sessions avancées avec *scope*** (browser, tab, user...)

Ocsigen : principales caractéristiques (2)

- **Services**
  - Programmation Web traditionnelle très simplifiée
  - Programmation Web avec continuations (services dynamiques)
- **Sessions avancées avec *scope*** (browser, tab, user...)
- Pages **réactives** (Functional Reactive Programming)

  Pages qui se mettent à jour toutes seules quand les données changent

  ⇝ Pas besoin de DOM virtuel

  ⇝ Code très concis !

  – Possibilité de générer ces pages réactives côté serveur !

Ocsigen : principales caractéristiques (2)

- **Services**
  - Programmation Web traditionnelle très simplifiée
  - Programmation Web avec continuations (services dynamiques)
- **Sessions avancées avec *scope*** (browser, tab, user...)
- Pages **réactives** (Functional Reactive Programming)

  Pages qui se mettent à jour toutes seules quand les données changent

  ⇝ Pas besoin de DOM virtuel

  ⇝ Code très concis !

  - Possibilité de générer ces pages réactives côté serveur !
- NOUVEAU ! **Multi-plateforme Web & mobile**

  ⇝ Une seule application à développer

besport®

ocsigen
fresh air in web programming

Ocsigen

Les avantages combinés

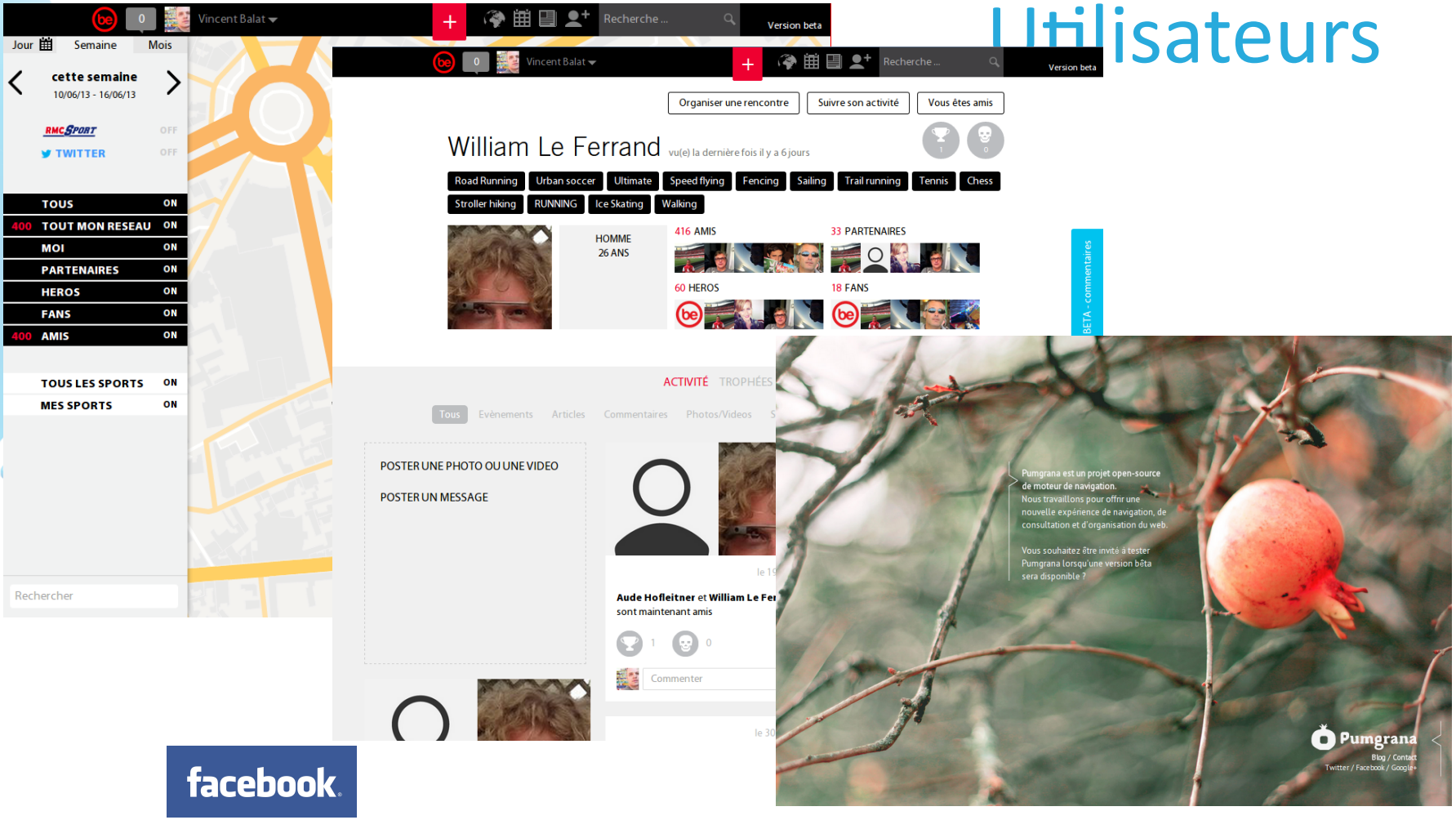
des *Single page applications*

et du Web traditionnel.

be sport®

ocsigen
fresh air in web programming

Utilisateurs



Entreprises et projets libres :

BeSport, NYU CGSB Genomics Core, Pumgrana, Facebook, Accret.io, Life.tl, Ashima Arts, Metaweb/Freebase, Hypios, Ocamlcore, Ocamlpro, Baoug, Nleyten...

besport®

ocsigen
fresh air in web programming

Quelques projets Ocsigen

ocsigen eliom — More information

Write client/server Web applications in very few lines of OCaml code!

ocsigen js_of_ocaml — More information

An OCaml to Javascript compiler.

ocsigen lwt — More information

A cooperative threading library for OCaml.

ocsigen server — More information

A full-featured and extensible Web server.

> And many other projects ...

besport®

ocsigen
fresh air in web programming

Sections client et serveur

```
let%server myservice = ...
```

```
let%client _ = ...
  ... a ~%myservice [pcdata "Click"] ()
```

Sections client et serveur

```
let%shared f =
```

```
let%server myservice = ...
```

```
let%client _ = ...
  ... a ~%myservice [pcdata "Click"] ()
```

Communication client-serveur

La syntaxe ~%v permet d'utiliser *côté client* une valeur v définie du *côté serveur*.

Cette valeur est envoyée avec la page.

```
let s = "hello"
```

```
p [pcdata ~%s]
```

Communication client-serveur

La syntaxe ~%v permet d'utiliser *côté client* une valeur v définie du *côté serveur*.

Cette valeur est envoyée avec la page.

Si f est une fonction côté serveur, appeler ~%f fera un appel de fonction distante (RPC).

```
let f l = List.map succ l
```

```
... match ~%f [1; 2; 3] with [] -> ...
```

Communication client-serveur

La syntaxe ~%v permet d'utiliser *côté client* une valeur v définie du *côté serveur*.

Cette valeur est envoyée avec la page.

Si f est une fonction côté serveur, appeler ~%f fera un appel de fonction distante (RPC).

```
let f l = List.map succ l
```

```
... match ~%f [1; 2; 3] with [] -> ...
```

La fonction f doit être exportée explicitement :

```
let f = server_function Json.t<int list> f
```

Valeurs client

```
let%server mybutton s =
  let d = div [pcdata "click"] in
  let _ =
    [%client
        clicks
          ~%d
          (fun ev -> window##alert(Js.string ~%s))
    ]
 in
 d
```

→ permet de générer les pages côté serveur

Valeurs client

```
let%shared mybutton s =
  let d = div [pcdata "click"] in
  let _ =
    [%client
        clicks
          ~%d
          (fun ev -> window##alert(Js.string ~%s))
    ]
 in
 d
```

Si la fonction est dans une section *shared*,
il est possible de l'appeler d'un côté ou de l'autre
suivant les besoins.

Valeurs client

```
let%server client_fun = [%client fun x -> x+1 ]
```

```
let%client _ =
  ... ~%client_fun 3 ...
```

Applications mobiles

Bientôt dans Ocsigen Eliom 6.0

1 seul code pour les applications Web et mobile

Application mobile exécutée dans une Webview (Android, IPhone, Windows...)

Démarrage de l'application côté client (mobile) ou serveur (Web)

Pages générées **côté serveur** (première page de l'application Web, requêtes HTTP)

ou **côté client** (pages suivantes, ou mobile)

⇝ Votre appli mobile multi-plateforme sans une ligne de plus

⇝ Bonnes performances

Génération côté client ou serveur

Principe :

Implémenter chaque service dans des sections *shared*

- Implémentation différente des fonctions d'accès à la DB
- ⚠ Attention à l'ordre d'exécution différent (spinners...)

Avantages :

- Les pages peuvent encore être générées côté serveur (1re requête...)
  ⇝ Compatibilité Web
- Les changements de page suivants peuvent avoir lieu côté client
  ⇝ Impression de changement de page instantané
- Le code est presque identique à du code côté client
  ⇝ Simple à utiliser

Applications multi-plateforme — 2e étape

Interaction Web côté client

Principe :

- Enregistrer les services côté client et serveur
- Identification de services côté client

  (bouton back, reload, ouverture de liens dans l'application,...)
- Transitions entre pages

Démarrage de l'application côté client

Principe :

- Fichier HTML minimal + JS inclus dans l'application
- Lancement d'une Webview
- Au premier démarrage, l'application fait une requête pour récupérer les injections globales

Alternative : les insérer de manière statique dans le HTML ou le JS

Mises à jour de l'application

Comment assurer la compatibilité de l'application client avec le serveur ?

1. Figer les APIs et utiliser un serveur par version de l'API ?

   Complexe et dangereux.

   - Chaque redémarrage peut générer des identifiants différents pour les client-values ou les injections
   - Mises à jours complexes (notamment si la structure de la base de données a changé)

Mises à jour de l'application

Comment assurer la compatibilité de l'application client avec le serveur ?

1. Figer les APIs et utiliser un serveur par version de l'API ?

   Complexe et dangereux.

   - Chaque redémarrage peut générer des identifiants différents pour les client-values ou les injections
   - Mises à jours complexes (notamment si la structure de la base de données a changé)

2. Mettre à jour le JS automatiquement ?

   Utilisation de *Cordova Hot Code Push*.

   Nécessite un redémarrage côté client.

Mises à jour de l'application

Comment assurer la compatibilité de l'application client avec le serveur ?

1. Figer les APIs et utiliser un serveur par version de l'API ?

   Complexe et dangereux.

   - Chaque redémarrage peut générer des identifiants différents pour les client-values ou les injections
   - Mises à jours complexes (notamment si la structure de la base de données a changé)

2. Mettre à jour le JS automatiquement ?

   Utilisation de *Cordova Hot Code Push*.

   Nécessite un redémarrage côté client.

3. → Combinaison des deux

   ⇝ Permet d'éviter les redémarrages

   ⇝ Limite le nombre de versions à maintenir

Cordova

Cordova (alias Phonegap) permet aux applications HTML5 d'accéder aux fonctionnalités du matériel (Contacts, GPS, Accéléromètre, appareil photo, etc.) via une API JS.

Le binding pour OCaml de nombreux modules Cordova est en cours (travail de Danny Willems).

Retour d'expérience chez BeSport

L'application BeSport BETA est disponible sur `besport.com`, sur Google Play et sur l'app store Apple.

besport®

ocsigen
fresh air in web programming

Retour d'expérience chez BeSport

L'application BeSport BETA est disponible sur `besport.com`**, sur Google Play et sur l'app store Apple.**

Code presque **inchangé** par rapport à la version Web.

besport®

ocsigen
fresh air in web programming

Retour d'expérience chez BeSport

L'application BeSport BETA est disponible sur `besport.com`, sur Google Play et sur l'app store Apple.

Code presque **inchangé** par rapport à la version Web.

Performances : très bonnes, à condition de faire attention

Utiliser le bon moteur de rendu, éviter certaines fonctionnalités...

besport®

ocsigen
fresh air in web programming

Retour d'expérience chez BeSport

L'application BeSport BETA est disponible sur `besport.com`, sur Google Play et sur l'app store Apple.

Code presque **inchangé** par rapport à la version Web.

Performances : très bonnes, à condition de faire attention

Utiliser le bon moteur de rendu, éviter certaines fonctionnalités...

Principal problème : faiblesse du moteur de rendu imposé par Apple

Nombreux bugs, problèmes de performances, fonctionnalités exotiques à contourner

→ Une personne à temps plein sur ces problèmes depuis un mois.

Les palliatifs sont intégrés à Ocsigen autant que possible.

besport®

ocsigen
fresh air in web programming

Calendrier

Cet été, sortie d'**Ocsigen Eliom 6.0**, qui incluera les fonctionnalités mobiles.

Calendrier

Cet été, sortie d'**Ocsigen Eliom 6.0**, qui incluera les fonctionnalités mobiles.

Et bientôt :

Ocsigen-toolkit est une bibliothèque de widgets adaptés au modèle client-serveur.

Eliom-base-app: une bibliothèque de haut niveau et un template d'application clé en main (Web et/ou mobile).