

WebSharper

веб-программирование без слёз

Владимир Матвеев, IntelliFactory
Антон Таяновский, IntelliFactory



Пример дня: Excel в браузере

- Редактирование таблиц
- Добавление формул
- Сохранение данных на сервере
- Публикация веб-сервис интерфейса к данным
- И всё это на F# - благодаря WebSharper

Почему WebSharper?

Когда пишешь JavaScript, хочется плакать:

- **Нет вывода типов - опечатки приводят к ошибкам**
- **Нет толковой среды разработки**
- **Нет стандартной платформы, библиотек, коллекций**
- **Нет стандарта упаковки модулей и документации**

Почему WebSharper?

Еще хуже дело с клиент-серверными приложениями:

- **Нужно думать о передаче данных и их упаковке**
- **Нужно привязывать скрипты, стили и HTML**

Почему F#?

- **Функциональное программирование**
- **Вывод типов**
- **Удобная среда разработки**
- **Хорошая платформа**

Компиляция F# в JavaScript

```
namespace Hello
```

```
module Main =  
    [<JavaScript>]  
    let rec Fac n =  
        match n with  
        | 0 -> 1  
        | n -> n * Fac (n - 1)
```

```
> Hello.Main.Fac(10)  
3628800
```

Стандартная библиотека

```
let d = Dictionary()  
for (k, v) in pairs do  
    d.[k] <- v  
d
```

Удалённый вызов

```
[<Rpc>]
```

```
let Save (user: User) =  
    database.Save user  
    async { return OK }
```

```
[<JavaScript>]
```

```
let AddAccount() =  
    let user = ..  
    async {  
        let! response = Save user  
        do! Show response  
    }  
    |> Async.Start
```


Привязки JavaScript библиотек

```
let config =  
    JQueryUI.DialogConfiguration(  
        Draggable = true, Modal = true,  
        Height = 370, Width = 500  
        Title = "..", CloseOnEscape = true)  
JQueryUI.Dialog.New(element, config)
```

Пользовательские привязки

```
[<Inline "eval($s)">]  
let eval (s : string) = X<_>
```

Функциональный подход к UI

```
Formlet.Do {  
    let! name = functionName  
    let! text = functionText  
    return name, text  
}  
|> Formlet.Flowlet
```

Функциональная маршрутизация

```
type Actions =  
    | Main  
    | DownloadWorksheet  
  
let Sitelet =  
    Sitelet.Content "/" Main mainPage <|>  
    Sitelet.Infer (function  
        | Main -> mainPage  
        | DownloadWorksheet -> download ())
```

Статическая проверка ссылок

```
fun ctx ->  
  A [Href (ctx.Link Main)] [Text "Home"]
```

Автоматизация зависимостей

```
module Styles =  
    [<Sealed>]  
    type Table() =  
        inherit Resources.BaseResource("Styles.css")  
  
[<Require(typeof<Styles.Table>)]  
module UI =
```

К делу!

	A	B	C	D	E	F	G	
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								

Спасибо за внимание!

- <http://websharper.com>
- <http://intellifactory.com>
- <http://bitbucket.org/IntelliFactory/talk-kiev-alt.net-6>