

ReactiveX - okiełznać asynchroniczność











Problemy z asynchronicznością

- Różne API w każdym języku
 - Thread w Javie
 - async/await w .NET
 - callbacks, Promise w JavaScript
 - **.** . . .
- Ciężka obsługa błędów (try/catch?)
- Callback hell
- Zdarzenia?
- Kompozycja?

```
function register()
if (!empty(0 POST)) (
    Smag - '''s
    if (5_7057('user_name')) (
        if (2 POST['user password new']) {
            if ($ POST['user_password_new'] --- $ POST['user_password_repeat']) {
                if (strlen($_POST('user_paseword_new')) > 5) {
                    if (strlen($ POS7['usor name']) < 65 && strlen($ POS7['usor name']) > 1) {
                        if (prog_match('/'(a-s\d){2,64}$/i', $_POST['user_name'])) {
                            Suser - read_user($_7057['user_name']);
                            if (lisset(Suser('user name'))) (
                                if (s Post['user_enail']) {
                                         if (filter war($ POST['oser email'], FILTER VALIDATE EMAIL)) (
                                            create user();
                                            S_SESSION['msg'] - 'You are now registered so please login';
                                            header('Location; ' . 5 SERVER('PEP SELF'1);
                                         else Fmag = 'You must provide a valid email address';
                                    ) else finsq = 'Ensil must be less than 60 characters';
                                I else Smow - 'Email cannot be empty's
                            ) else Omag - 'Unername already exists';
                        ) else insg - 'Coorname must be only a-s, A-S, 0-9';
                    ) else Smag - 'Username must be between 2 and 64 characters's
                } else Smay = 'Password must be at least 6 characters';
            ) else Smag = 'Passwords do not match';
        ) else Smag - 'Empty Password';
    } else (mag = 'Empty Daername';
    $ SESSION['moe'] - Smeq;
return register_form();
```



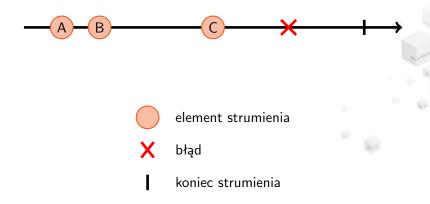


An API for asynchronous programming with observable streams

ReactiveX is a combination of the best ideas from the Observer pattern, the Iterator pattern and functional programming



Observable streams



Co może być elementem strumienia?

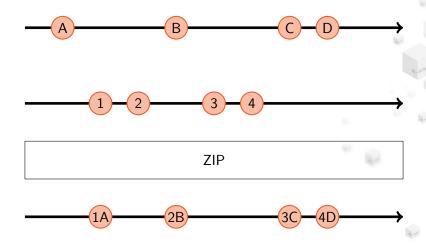
wszystko

Co może być elementem strumienia?

- zdarzenia użytkownika (np. klikanie myszki)
- String
- odpowiedź serwera na zapytanie HTTP
- tweet
- pojedynczy wiersz wyniku zapytania SQL
- **.** . . .



Łączenie strumieni





Operatory

From	Join	Range	Interval
Just	${ t Flat Map}$	GroupBy	Map
Scan	Window	Debounce	Distinct
ElementAt	Filter	First	Last
Sample	Skip	SkipLast	Take
TakeLast	And	CombineLatest	Join
Merge	Switch	Zip	Retry

. .

Tylko Java i JavaScript?

- Java
- JavaScript
- ► C#
- Scala
- Clojure
- ► C++

- Ruby
- Python
- Groovy
- ▶ JRuby
- ► Kotlin
- Swift



Co dalej?

```
http://reactivex.io
```

Główne źródło wiedzy na temat ReactiveX wraz z linkami do dokumentacji dla wszystkich implementacji w różnych językach

```
http://rxmarbles.com/
```

Interaktywne diagramy operatorów

```
https:
```

```
//gist.github.com/staltz/868e7e9bc2a7b8c1f754
```

"The introduction to Reactive Programming you've been missing"

```
https://github.com/mlyczek/fdd2015-reactivex
```

Slajdy z tej prezentacji oraz ich kod (LATEX)



Co dalej?





?



Future Processing



WWW.FUTUREDEVDAY.PL











