Bolts

Superbil @Cocoaheads Taipei

About me

- 燒毀果蠅的蘋果園藝師
- Objective-C / Git / Emacs / Python
- about.me/superbil , @superbil
- freenode #g0v.tw #emacs.tw
- Freelance Software Developer

Guideline

- Functional Reactive Programming
- Bolts Intro
- Why Bolts?
- Example

Functional Reactive Programming

- 讓資料流變化可以自動傳播的程式設計典範

FRP與函數式 http://www.ithome.com.tw/voice/91328

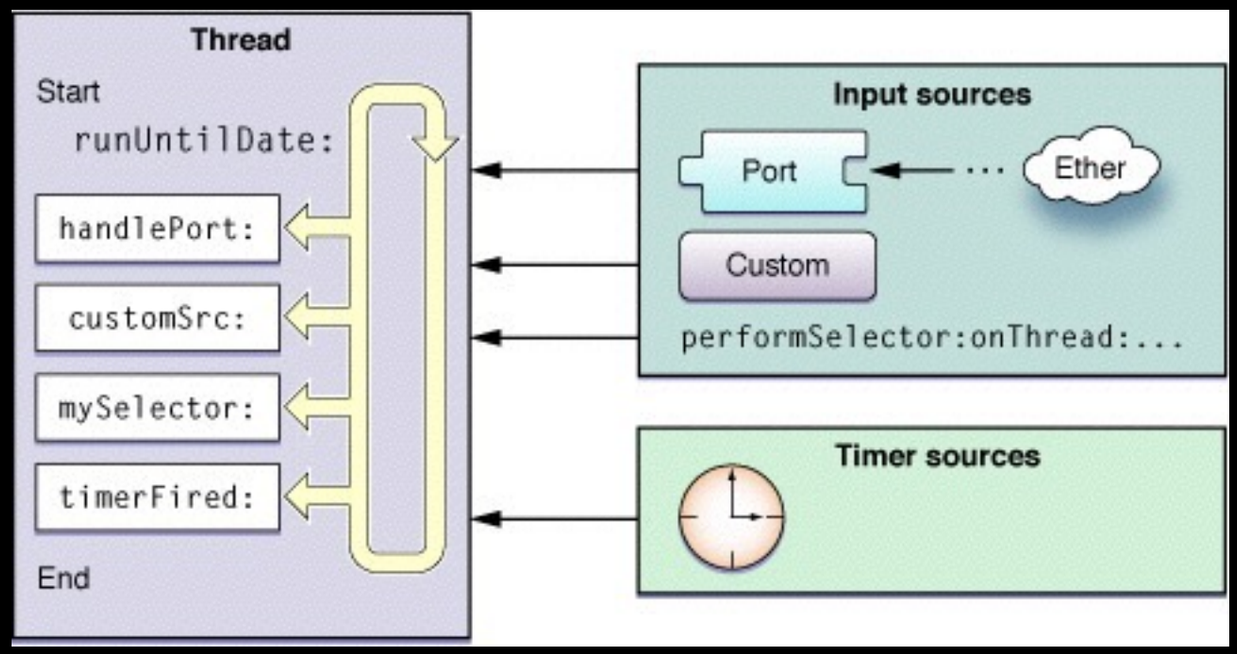
Why Bolts ?

- Facebook

- github.com/zonble/CWBOpenDataClient

Why Bolts?

- NSRunloop
- input source
- Timer sources



mikeash NSRunLoop Internals https://mikeash.com/pyblog/friday-qa-2010-01-01-nsrunloop-internals.html

Bolts Intro

- A task is kind of like a JavaScript Promise

https://www.promisejs.org/

Bolts Intro

- Property: data sequence than remembers only the last value
- a property with only one single value/event
- Promises are a subset of FRP

Taming Asynchronous Programming (beyond Promises)
http://eamodeorubio.github.io/tamingasync/#/60

Promise

- must have value (callback)
  - result
  - error
  - etc..

contiuneWithBlock:

```
[[self saveAsync:obj] continueWithBlock:^id(BFTask
*task) {
  if (task.isCancelled) {
    // the save was cancelled.
  } else if (task.error) {
    // the save failed.
  } else {
    // the object was saved successfully.
    PFObject *object = task.result;
  }
  return nil;
}];
```

continueWithExecutor:withBlock:

```
// Continue on the Main Thread, using a built-in executor.
[[self fetchAsync:object]
  continueWithExecutor:[BFExecutor mainThreadExecutor]
             withBlock:^id(BFTask *task) {
    myTextView.text = [object objectForKey:@"name"];
}];
```

nil is end

```
[[self saveAsync:obj]
continueWithSuccessBlock:^id(BFTask *task) {
  // the object was saved successfully.
  return nil;
}];
```

BFTaskCompletionSource

- Normal use
  - setResult:
  - setError:
  - setException:
  - cancel
- Try to set
  - trySetResult
  - trySetError
  - trySetException
  - trySetCancelled

BFExecutor

- defaultExecutor
- mainThreadExecutor
- dispatch_queue_t
- NSOperationQueue

for completion

- taskForCompletionOfAllTasks:
- taskForCompletionOfAllTasksWithResults:

BFCancellationToken

- Solve flow need cancel problem
- After 1.2.0 support

BFCancellationToken

```
BFCancellationTokenSource *cts =
[BFCancellationTokenSource cancellationTokenSource];
[cts.token registerCancellationObserverWithBlock:^{
    NSLog(@"A");
}];
BFTask *task = [BFTask taskWithDelay:500];
  [task continueWithBlock:^id(BFTask *task) {
    NSLog(@"B");
    return nil;
} cancellationToken:cts.token];
  NSLog(@"C");

[cts cancel];
```

Live DEMO

More example

- https://github.com/BoltsFramework/Bolts-iOS
  - Readme.md
  - TaskTests.m
  - CancellationTests.m

Q & A