How we avoid callback hell

Fedor Korshunov

github.com/fedor
fedor@aftership.net

github.com/fedor/co_demo

func1() -> delay 2s -> func2() -> func3

sync code

Safe

Readable

Easy errors handling

async code

Unsafe

Multiple places for error handling

Callback Hell

Promise is an operation that hasn't completed **YET**

Centralized errors handling

Less callback hell

Some callback hell....

Generator function is a function that you can **RE-ENTER**

.next(val) — run before next yield
.throw(err) — throw inside generator

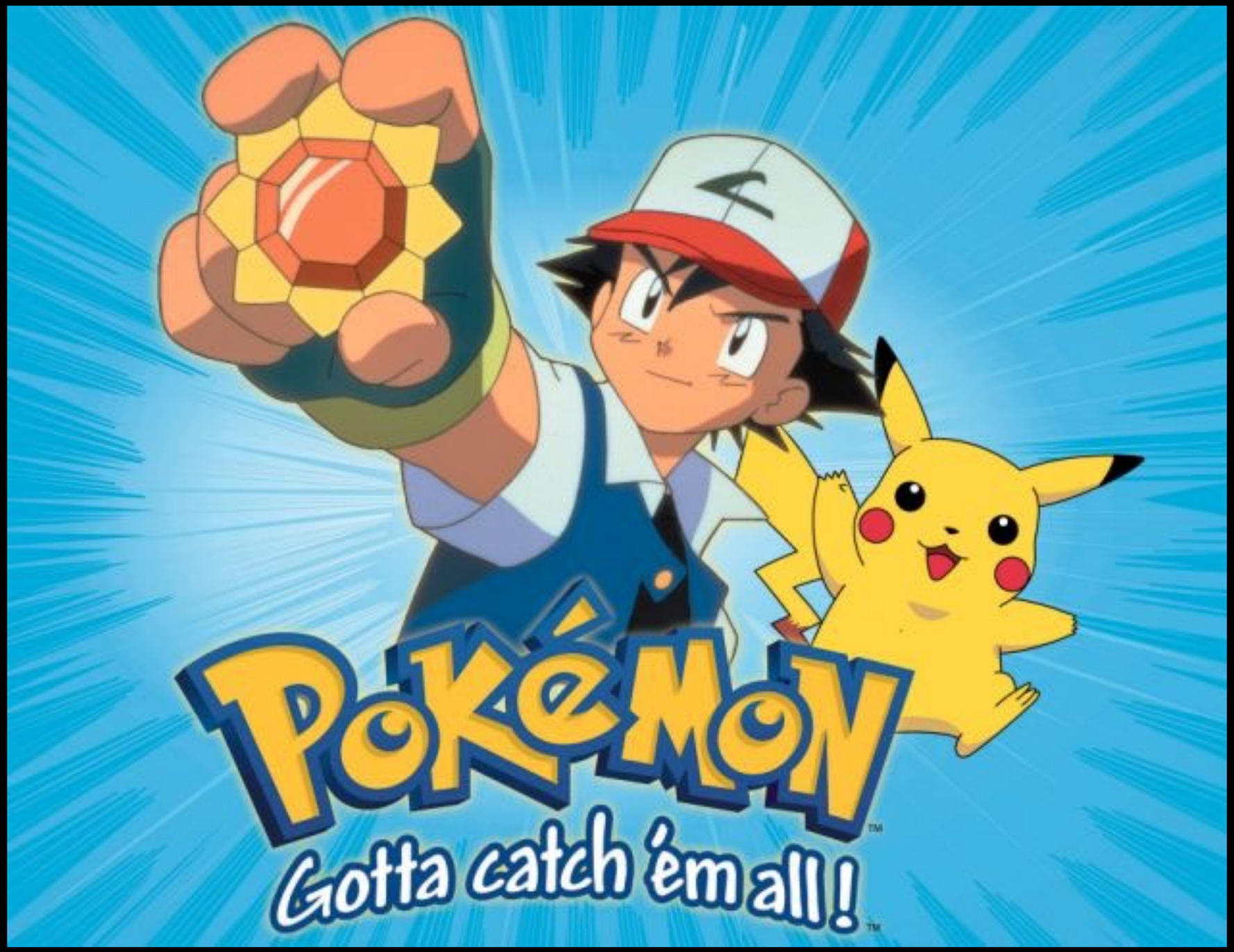
```
// control code
var gen = genfunc();
...
gen.next('new value')  --> {value: 'result'...}
```

```
// generator function
var genfunc = function* () {
    ...
    var new_value = yield 'result';
    ...
}
```

Promise + Generator = ♥

Can we use Generators to control Promises?

Exceptions Handling



Final Notes

npm install co bluebird

use co() for flow-control, return Promise

Promisify callback functions

avoid inline-functions ~~.forEach(...)~~