Modern Asynchronous JavaScript

CALLBACKS

Wes Higbee

@g0t4 www.weshigbee.com

Links

Weather
http://plnkr.co/edit/ZOAaSdtnfvR6WwiuVZ6s

Generator Control Flow Assistant
http://plnkr.co/edit/3RQpB0etPVBhTVqD8AY9

GitHub repo
https://github.com/g0t4/pluralsight-modern-async-js

Key Takeaways Reasoning About Asynchronous JavaScript

- Single-threaded (bank with one teller)
- Event Loop (line to use the bathroom)
- Non-blocking (forgot papers, next person serviced)
- Avoid blocking (don't hog the bathroom)
- Run to completion (satisfy customer before next)
- Cooperative Concurrency (customers play nice)
- Little Programs (customers in line)
- Think explicitly about Asynchronous Seams
- Timer delay not guaranteed (calendar /

Callbacks

Explicit Seams – blessing and curse

Seams rip across program

Another, difficult error mechanism

Boilerplate error handling

Hard to reuse error handling logic

Difficult to understand

- Nesting
- Never called?
- Called multiple times?
- Called synchronously?

Synchronizing parallelism is hard