Async JavaScript at Netflix

Jafar Husain
@jhusain

Who is Jafar?

- Cross-Team Technical Lead for the Netflix UIs
- Created the async data platform for Netflix UI's
- Member of TC39
- 13 years in the industry, formerly worked at Microsoft and GE

This is the story of how Netflix solved

BIG async problems

by thinking differently about

Events.


2014

The Netflix App is Asynchronous

- App Startup
- Player
- Data Access
- Animations
- View/Model binding

Async Problems

- Memory Leaks
- Race Conditions
- Callback Hell
- Complex state machines
- Error Handling

Async is Hard

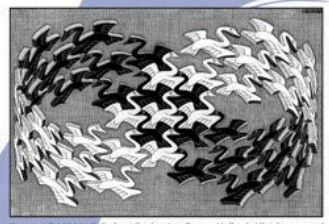
```
function play(movieId, cancelButton, callback) {
    var movieTicket,
        playError,
        tryFinish = function() {
            if (playError) {
                callback(null, playError);
            }
            else if (movieTicket && player.initialized) {
                callback(null, ticket);
            }
        };
    cancelButton.addEventListener("click", function() { playError = "cancelled"; }
    if (!player.initialized) {
        player.init(function(error) {
            playError = error;
            tryFinish();
        });
    }
    authorizeMovie(function(error, ticket) {
        playError = error;
        movieTicket = ticket;
        tryFinish();
    });
});
```


1994
ACE VENTURE
PET DETECTIVE
Sports Illustrated
SUPER METROID
KEANU REEVES
Tom Hanks is
Forrest
Jump
July 6
CLERKS
A very funny look at the over-the-counter culture.
NIRVANA
UNPLUGGED
IN NEW YORK
weezer
AMERICA
Online

Design Patterns

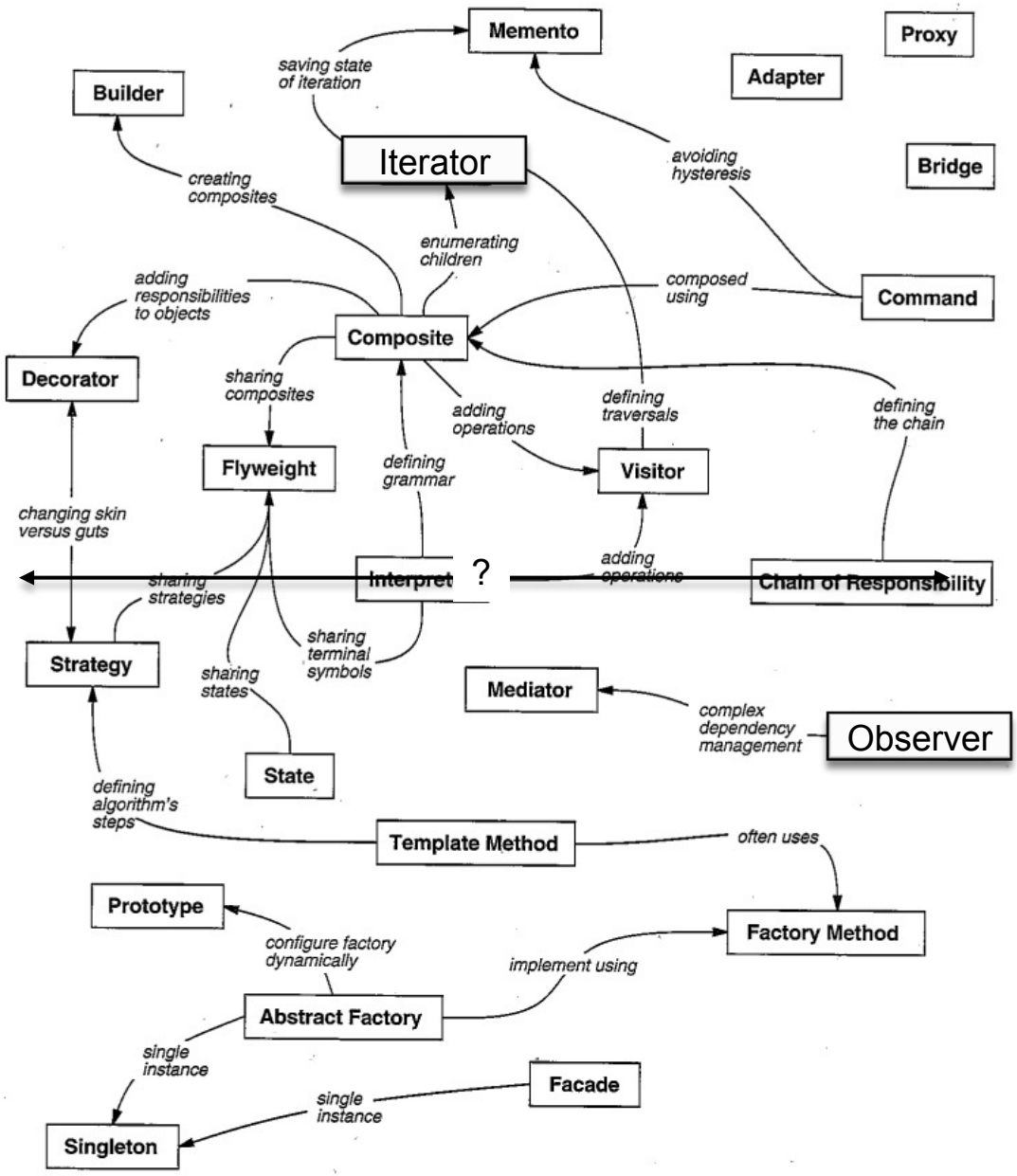
Elements of Reusable Object-Oriented Software

Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides



Foreword by Grady Booch

ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES



Design Pattern Relationships

Iterator

```
> var iterator = getNumbers();
> console.log(iterator.next());
> { value: 1, done: false }
> console.log(iterator.next());
> { value: 2, done: false }
> console.log(iterator.next());
> { value: 3, done: false }
> console.log(iterator.next());
> { done: true }
>
```

Observer Pattern

```
> document.addEventListener(
    “mousemove”,
    function next(e) {
     console.log(e);
    });
```

```
> { clientX: 425, clientY: 543 }
> { clientX: 450, clientY: 558 }
> { clientX: 455, clientY: 562 }
> { clientX: 460, clientY: 743 }
> { clientX: 476, clientY: 760 }
> { clientX: 476, clientY: 760 }
> { clientX: 476, clientY: 760 }
```


Iterator
progressively send information to consumer
Observer

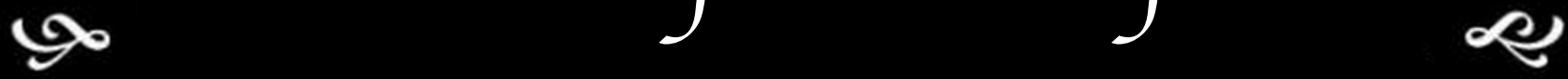

IN OBSERVER PATTERN
PRODUCER ITERATE YOU

“What’s the difference between an Array…

```
[{x: 23, y: 44}, {x:27, y:55}, {x:27, y:55}]
```

… and an Event?

Events and Arrays are *both* collections.


Now for a brief
JavaScript 6 tutorial...

Functions

```
function(x) { return x + 1; }
function(x, y) { return x + y; }
```

```
x => x + 1
(x, y) => x + y
```

JS5
6


Fin.

The majority of Netflix's async code is written with just a few *flexible* functions.

ForEach

```
> [1, 2, 3].forEach(x => console.log(x))
> 1
> 2
> 3
> 
```

Map

Map

```
> [1, 2, 3].map(x => x + 1)
> [2, 3, 4]
>
```

Filter

Filter

```
> [1, 2, 3].filter(x => x > 1)
> [2, 3]
>
```

concatAll

concatAll

```
> [ [1], [2, 3], [], [4] ].concatAll()
> [1, 2, 3, 4]
>
```

Map/Filter/ConcatAll

```
> [1, 2, 3].map(x => x + 1)
> [2, 3, 4]

> [1, 2, 3].filter(x => x > 1)
> [2, 3]

> [ [1], [2, 3], [], [4] ].concatAll()
> [1, 2, 3, 4]
>
```

Orange is the New Black

★★★★½ 2013 TV-MA 13 episodes 5.1

From the creator of “Weeds” comes this series about a privileged New Yorker who ends up in a women's prison when a past crime catches up with her.

Based on your interest in:
Breaking Bad

Let’s use map, filter, and concatAll to get a list of your favorite Netflix titles.

Top-rated Movies Collection

```
var getTopRatedFilms = user =>
   user.videoLists.
      map(videoList =>
         videoList.videos.
            filter(video => video.rating === 5.0)).
      concatAll();

getTopRatedFilms(user).
   forEach(film => console.log(film));
```

What if I told you…

…that you could create a drag event…

…with nearly the *same code*?

Top-rated Movies Collection

```
var getTopRatedFilms = user =>
    user.videoLists.
        map(videoList =>
            videoList.videos.
                filter(video => video.rating === 5.0)).
        concatAll();

getTopRatedFilms(user).
    forEach(film => console.log(film));
```

Mouse Drags Collection

```
var getElementDrags  = elmt =>
   elmt.mouseDowns.
      map(mouseDown =>
         document.mouseMoves.
            filter takeUntil(document.mouseUps)).
      concatAll();

getElementDrags(image).
   forEach(pos => image.position = pos);
```

Introducing Observable

```
Observable === Collection + Time
```

Reactive Extensions

- Observable Type + Array Functions (and more)
- Open Source
- Ported to…
  - C
  - C#/VB.Net
  - Javascript
  - Java (Netflix)

Observables can model…

- Events
- Animations
- Async IO

Events to Observables

```
var mouseMoves =
    Observable.
        fromEvent(element, “mousemove”);
```

Event Subscription

```
// “subscribe”
var handler = (e) => console.log(e);
document.addEventListener(“mousemoves”, handler);

// “unsubscribe”
document.removeEventListener(“mousemoves”, handler);
```

Observable.forEach

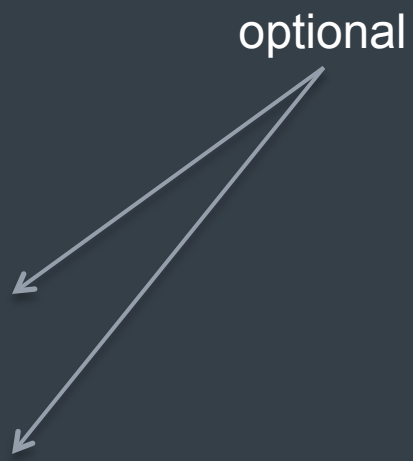
```
// “subscribe”
var subscription =
    mouseMoves.forEach(console.log);

// “unsubscribe”
subscription.dispose();
```

Expanded Observable.forEach

```
// “subscribe”
var subscription =
    mouseMoves.forEach(
        // next data
        event => console.log(event),
        // error
        error => console.error(error),
        // completed
        () => console.log(“done”));

// “unsubscribe”
subscription.dispose();
```



Observable Literal

time

{1.....2.....3}

ForEach


```
> {1……2…….....3}.forEach(console.log)
> 1
>
>
>
```

Map


```
> {1……2…….....3}.map(x => x + 1)
> 2
>
>
>
```

Filter

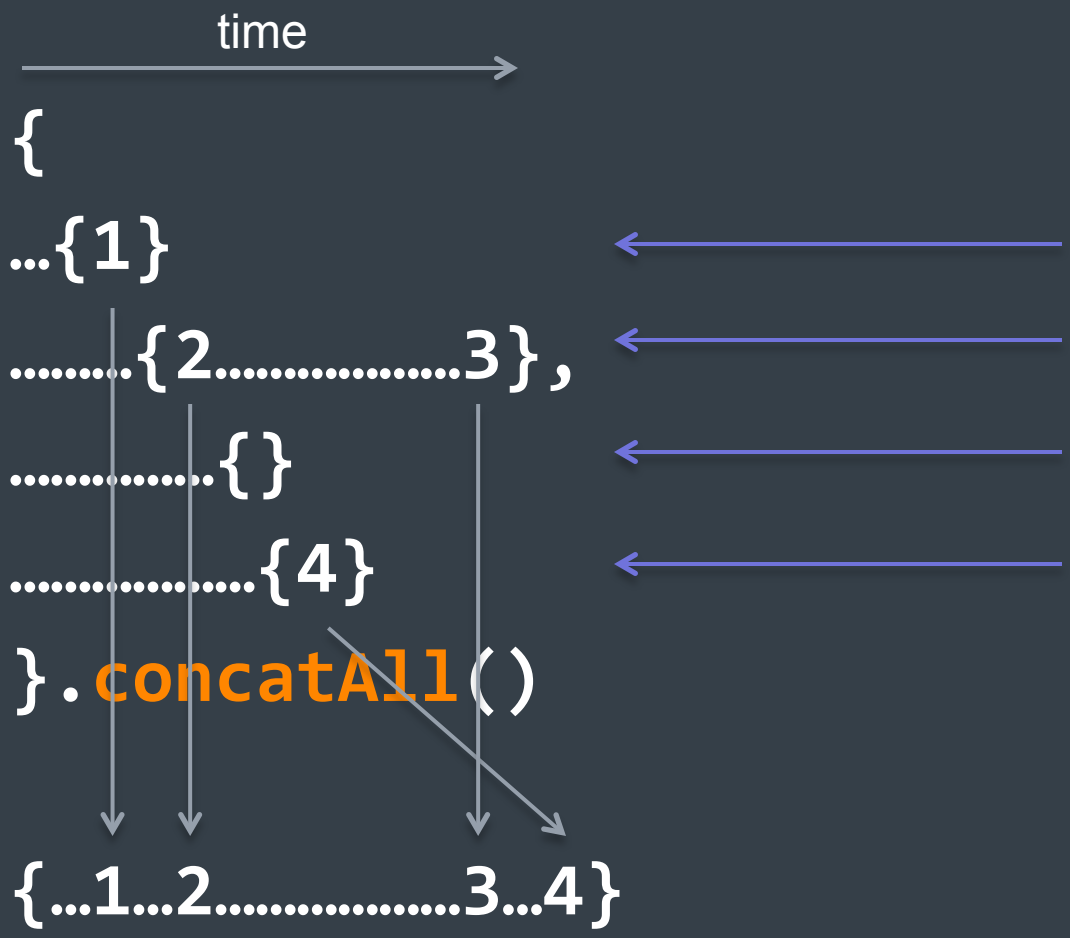

```
> {1……2……3}.filter(x => x + 1)
> 
> 
> 
```

concatAll

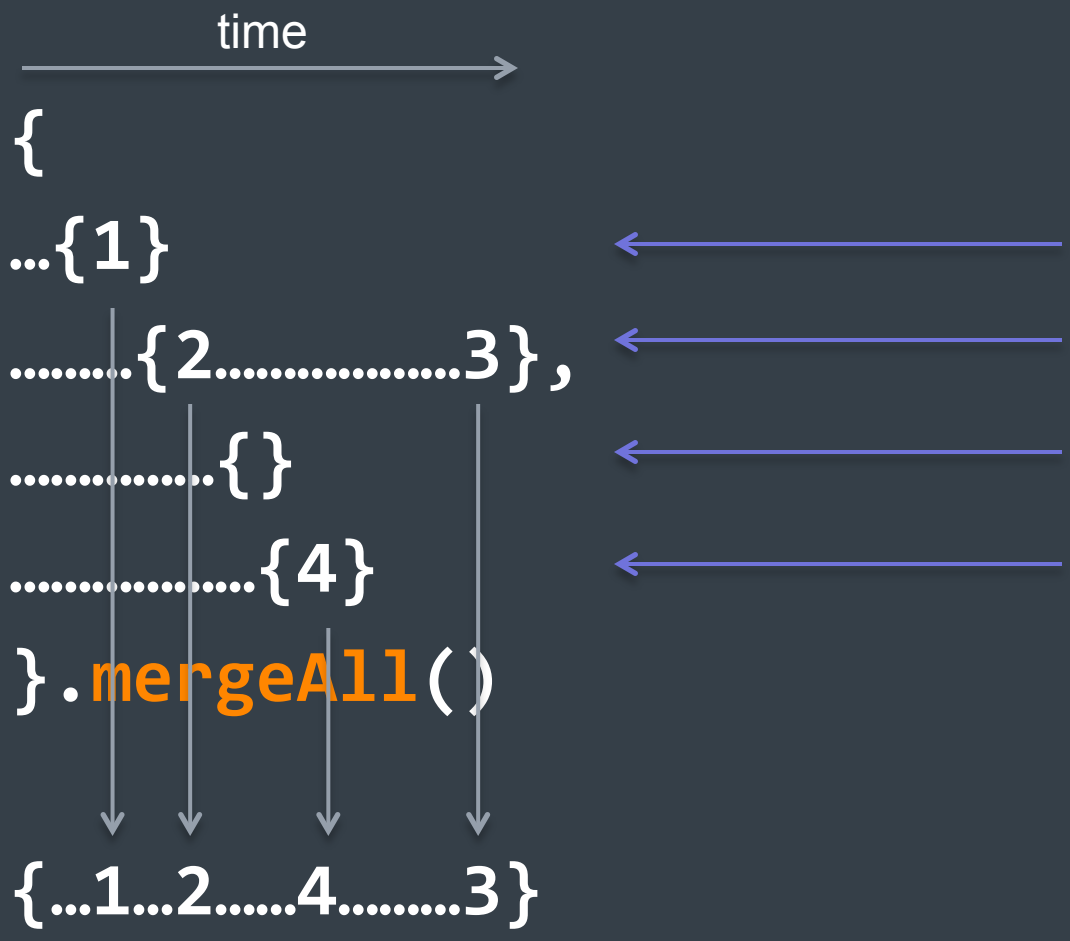
```
[
 [1]
 [2, 3],
 [],
 [4]
].concatAll()

[1, 2, 3, 4]
```

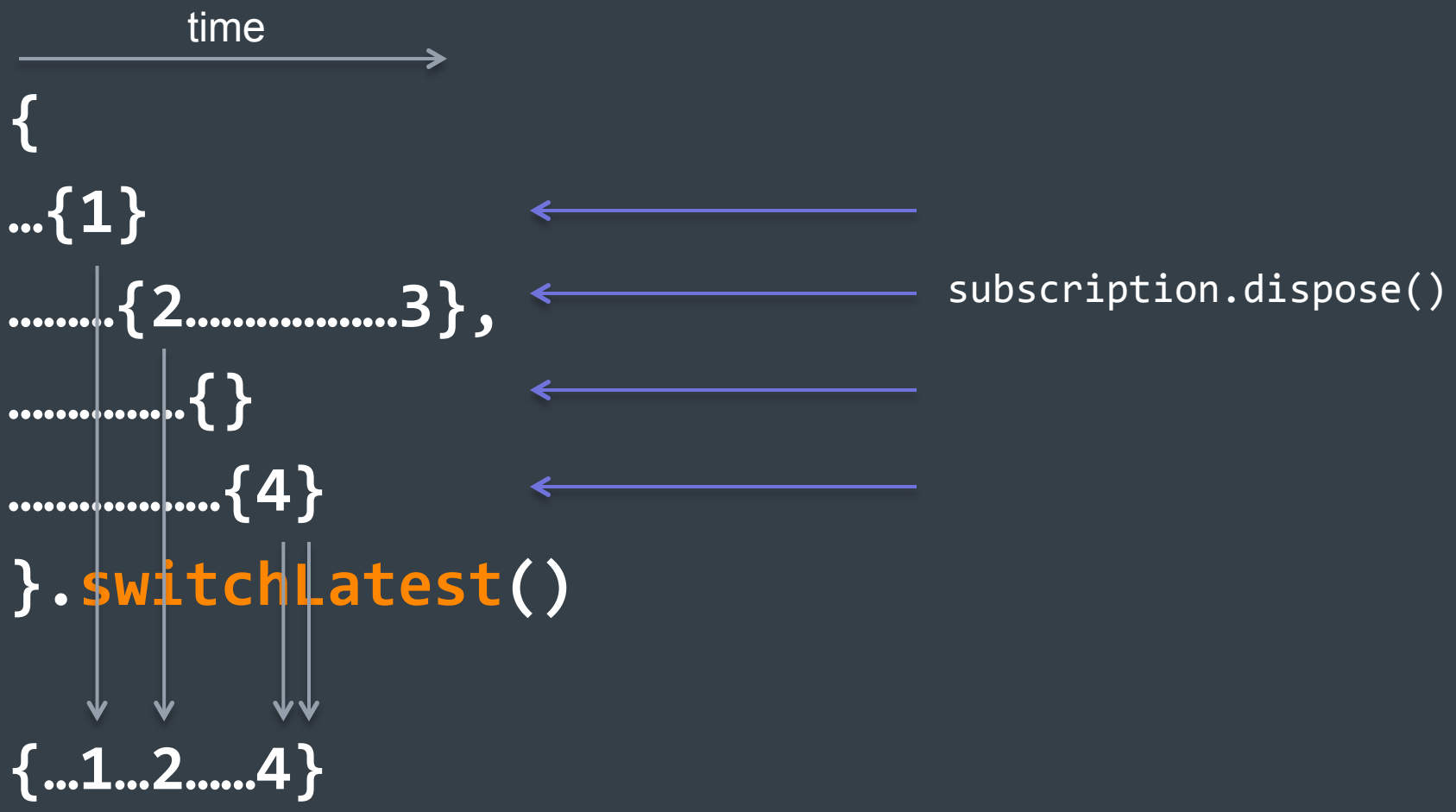
concatAll



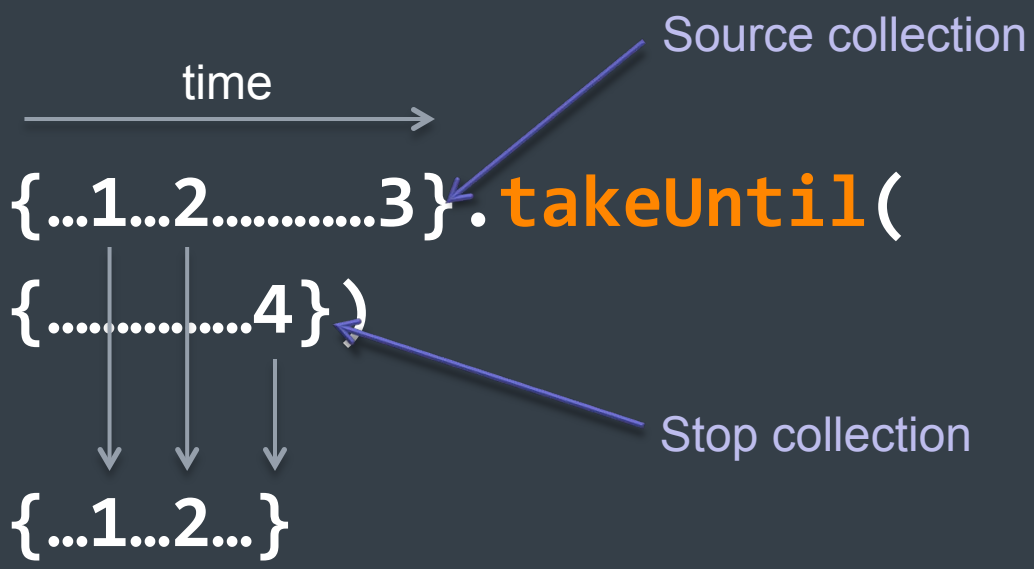
mergeAll



switchLatest



TakeUntil



Don’t unsubscribe from Events.

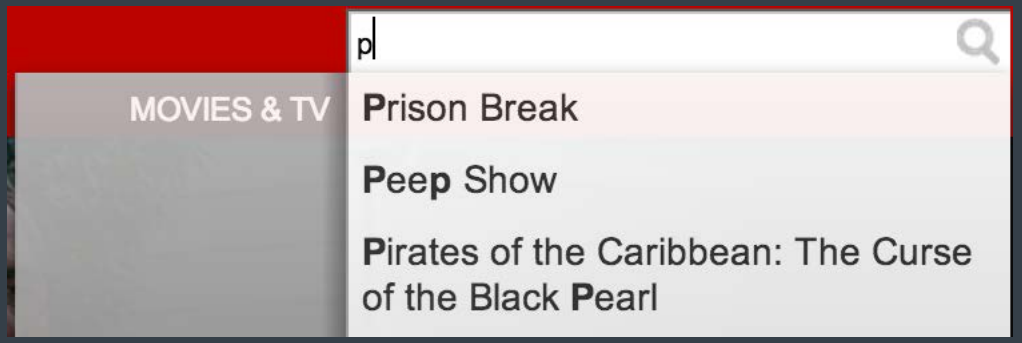
Complete them when another event fires.

Mouse Drags Collection

```
var getElementDrags  = elmt =>
   elmt.mouseDowns.
      map(mouseDown =>
         document.mouseMoves.
            takeUntil(document.mouseUps)).
      concatAll();

getElementDrags(image).
   forEach(pos => image.position = pos);
```

Netflix Search



Netflix Search

```
var searchResultSets =
    keyPresses.
        throttle(250).
        map(key =>
            getJSON(“/searchResults?q=” + input.value).
                retry(3).
                takeUntil(keyPresses)).
        concatAll();

searchResultSets.forEach(
    resultSet => updateSearchResults(resultSet),
    error => showMessage(“the server appears to be down.”));
```

Netflix Search

```
var searchResultSets =
    keyPresses.
        throttle(250).
        map(key =>
            getJSON(“/searchResults?q=” + input.value).
                retry(3).
                takeUntil(keyPresses)).
        concatAll switchLatest();

searchResultSets.forEach(
    resultSet => updateSearchResults(resultSet),
    error => showMessage(“the server appears to be down.”));
```

Netflix Search

```
var searchResultSets =
   keyPresses.
      throttle(250).
      map(key =>
         getJSON(“/searchResults?q=” + input.value).
            retry(3)).
      switchLatest();

searchResultSets.forEach(
   resultSet => updateSearchResults(resultSet),
   error => showMessage(“the server appears to be down.”));
```

Netflix Player

Player Callback Hell

```
function play(movieId, cancelButton, callback) {
    var movieTicket,
        playError,
        tryFinish = function() {
            if (playError) {
                callback(null, playError);
            }
            else if (movieTicket && player.initialized) {
                callback(null, ticket);
            }
        };
    cancelButton.addEventListener(“click”, function() { playError = “cancel”; });
    if (!player.initialized) {
        player.init(function(error) {
            playError = error;
            tryFinish();
        }
    }
    authorizeMovie(movieId, function(error, ticket) {
        playError = error;
        movieTicket = ticket;
        tryFinish();
    });
});
```

Player with Observable

```
var authorizations =
   player.
      init().
      map(() =>
         playAttempts.
            map(movieId =>
               player.authorize(movieId).
                     catch(e => Observable.empty).
                     takeUntil(cancels)).
            concatAll())).
      concatAll();

authorizations.forEach(
   license => player.play(license),
   error => showDialog("Sorry, can't play right now."));
```

Netflix: Observable Everywhere

- App Startup 
- Player 
- Data Access 
- Animations 
- View/Model binding 

Interactive Learning Exercises

http://jhusain.github.io/learnrx/

Observable in JavaScript 7?

```
async function* getStocks() {
   let reader = new AsyncFileReader("stocks.txt");
   try {
      while(!reader.eof) {
         let line = await reader.readLine();
         await yield JSON.parse(line);
      }
   }
   finally {
      reader.close();
   }
}

async function writeStockInfos() {
   let writer = new AsyncFileWriter("stocksAndPrices.txt");
   try {
      for(let name on getStocks()) {
         let price = await getStockPrice(name);
         await writer.writeLine(JSON.stringify({name, price}));
      }
   }
   finally {
      writer.close();
   }
}
```

Resources

- reactivetrader.azurewebsites.net
- https://github.com/Reactive-Extensions/RxJS
- RxJava
- http://jhusain.github.io/learnrx/
- @jhusain

Questions