@ph1

FUNCTIONAL
REACTIVE
JAVASCRIPT

@ph1

FUNCTIONAL REACTIVE PROGRAMMING

There are only two hard things in Computer Science:
cache invalidation *and*
naming things.

- Phil Karlton

There are only two hard things in Computer Science: cache invalidation, naming things, *and* off-by-one-errors

- anon

@ph1

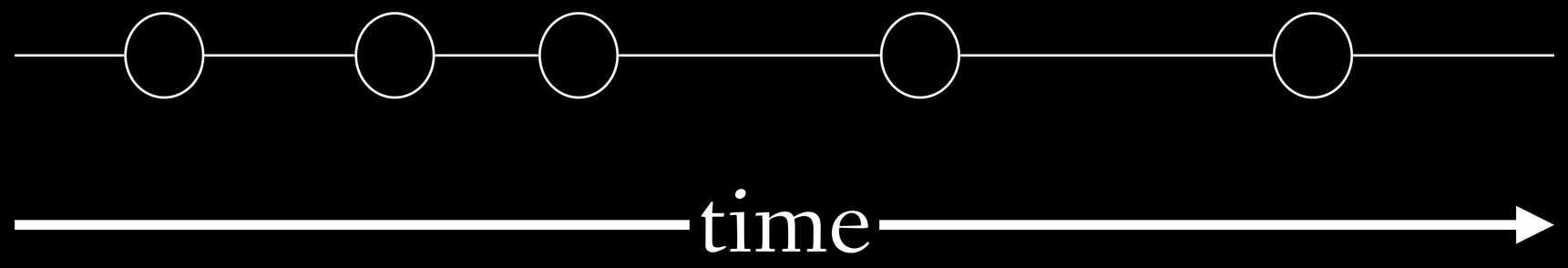
FUNCTIONAL REACTIVE PROGRAMMING

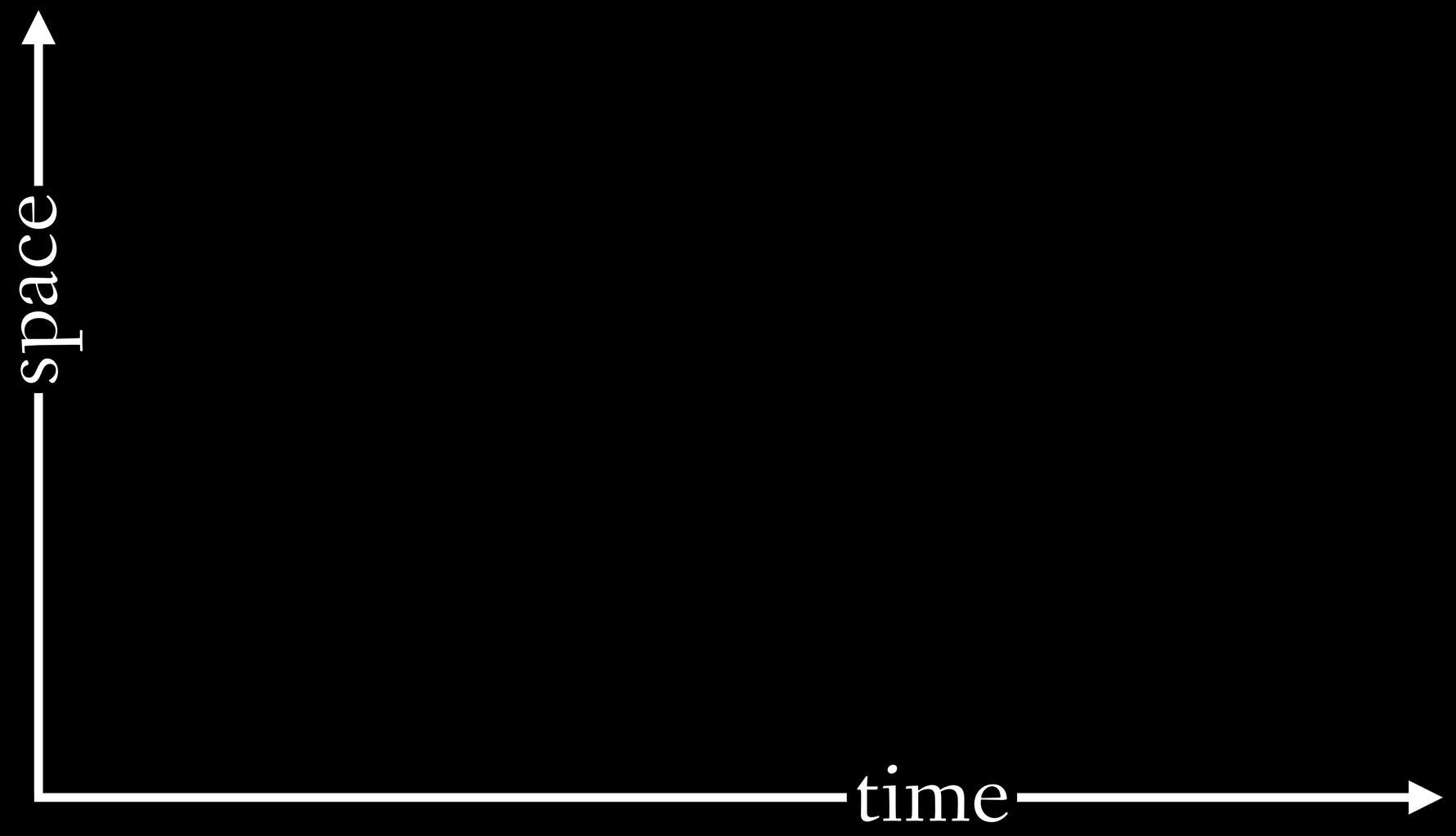
COMPOSITIONAL EVENT SYSTEMS

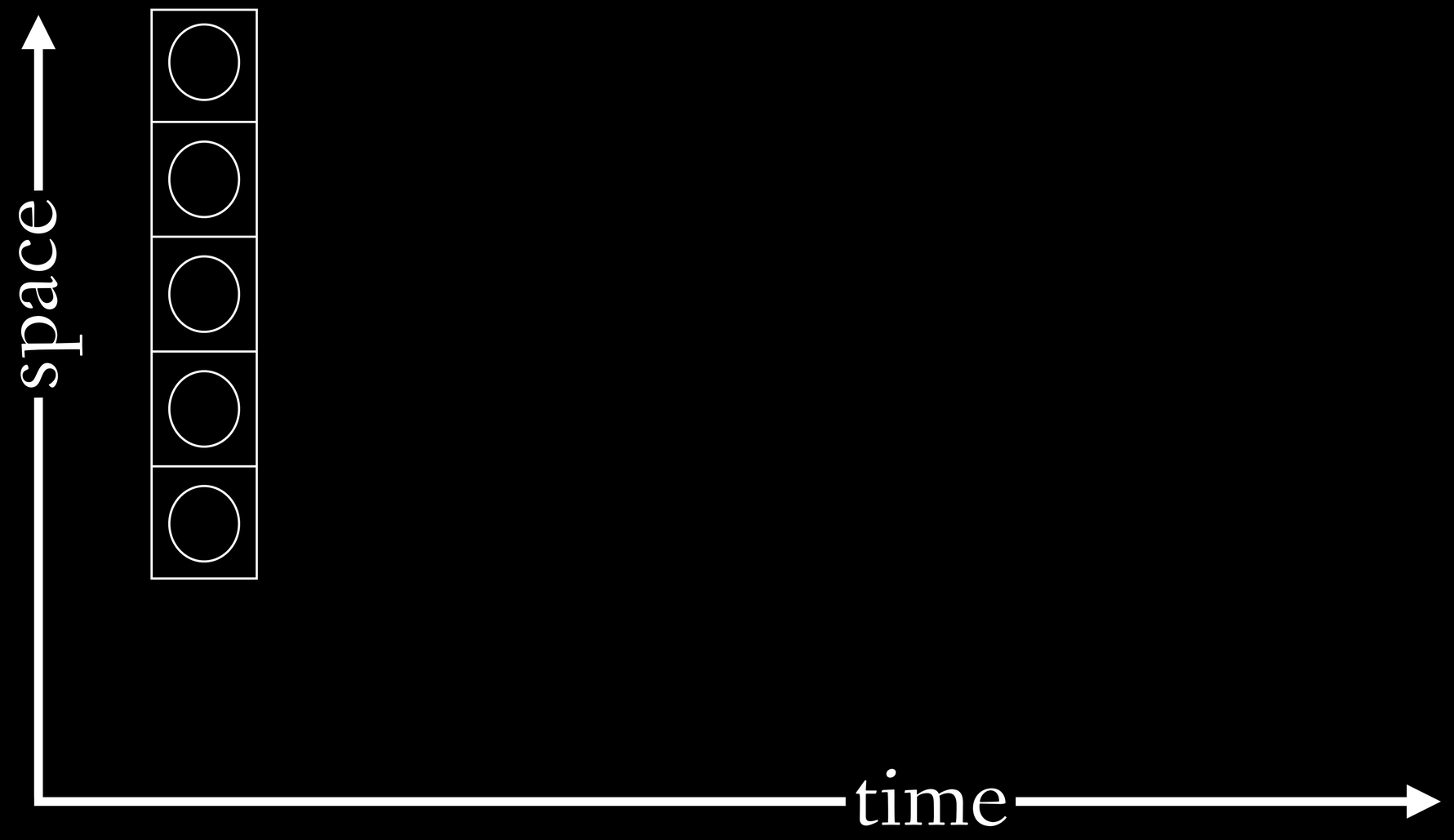
a different way of thinking about data and how it flows through our programs

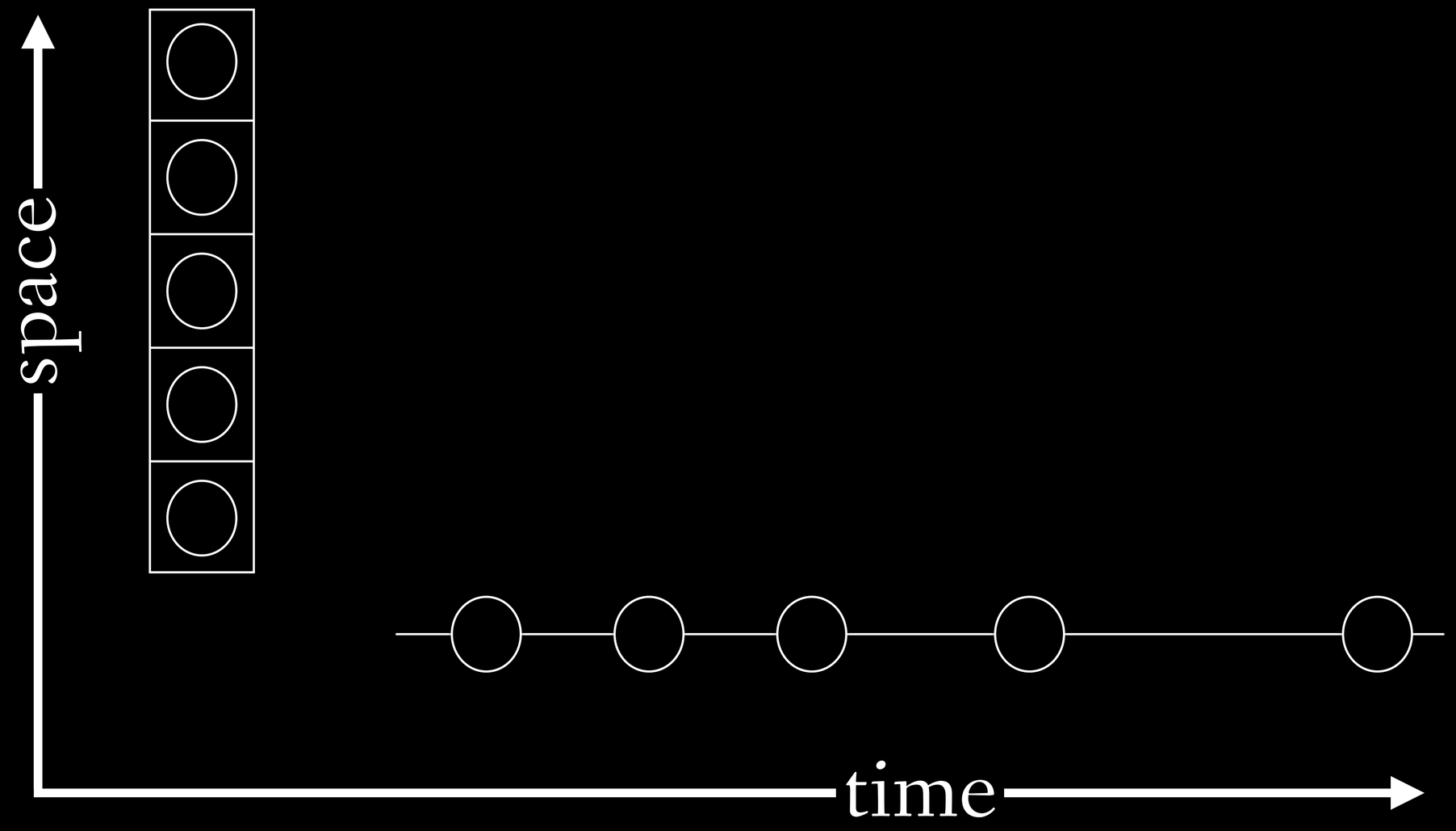
RxJS 5 / Observable

Observable




space
time





UI elements emit streams of values.

Functions combine and transform those streams into new streams.

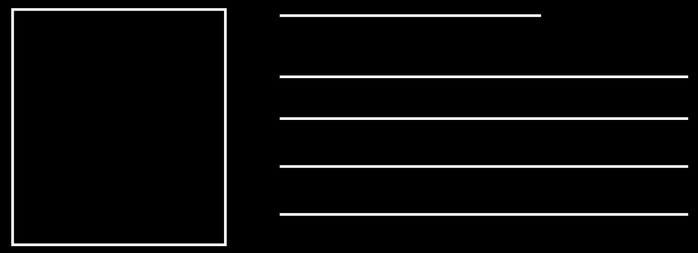
Streams can be assigned to UI attributes to effect UI output.

related artists

Taylor S

go!

Taylor Swift

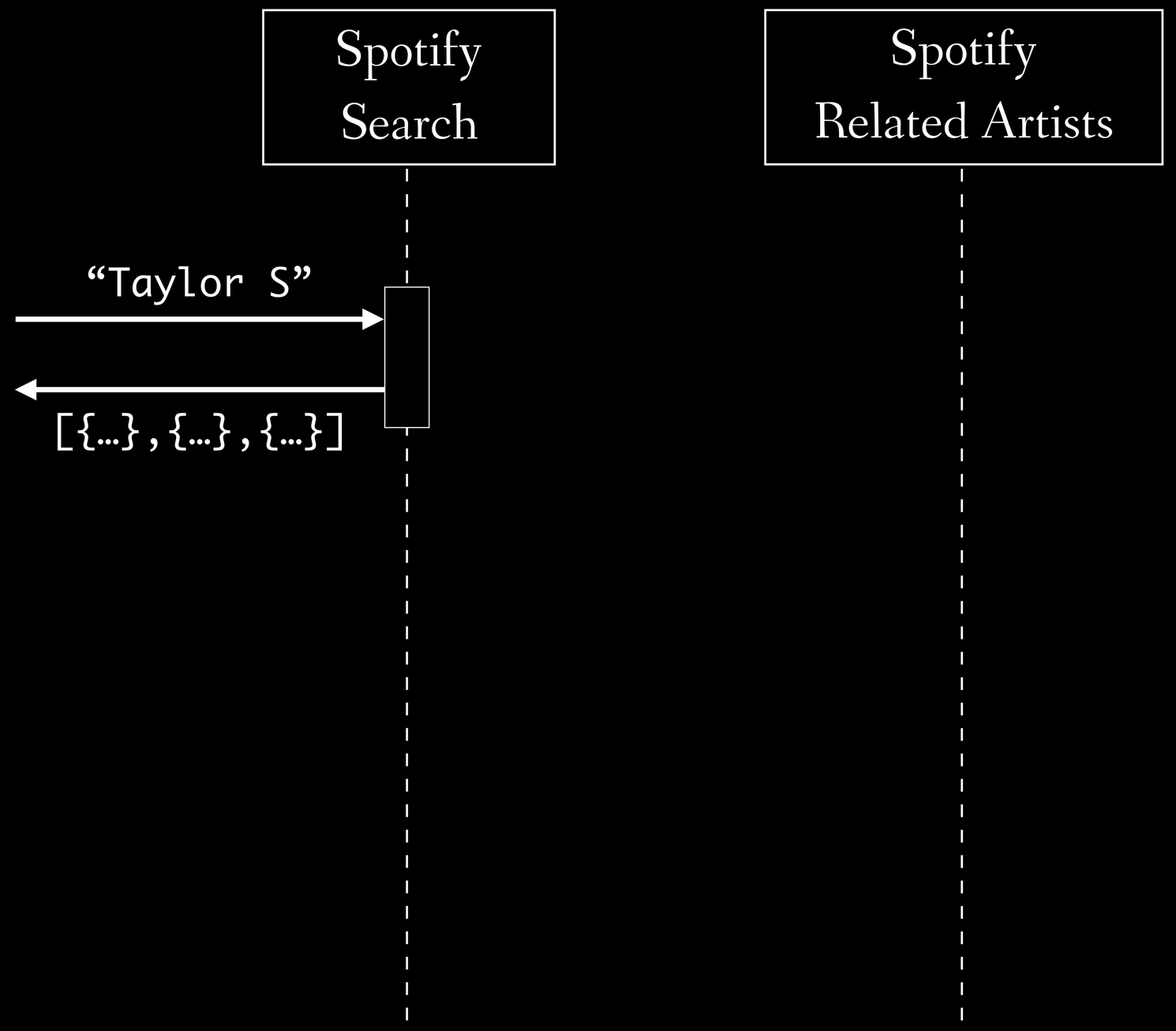


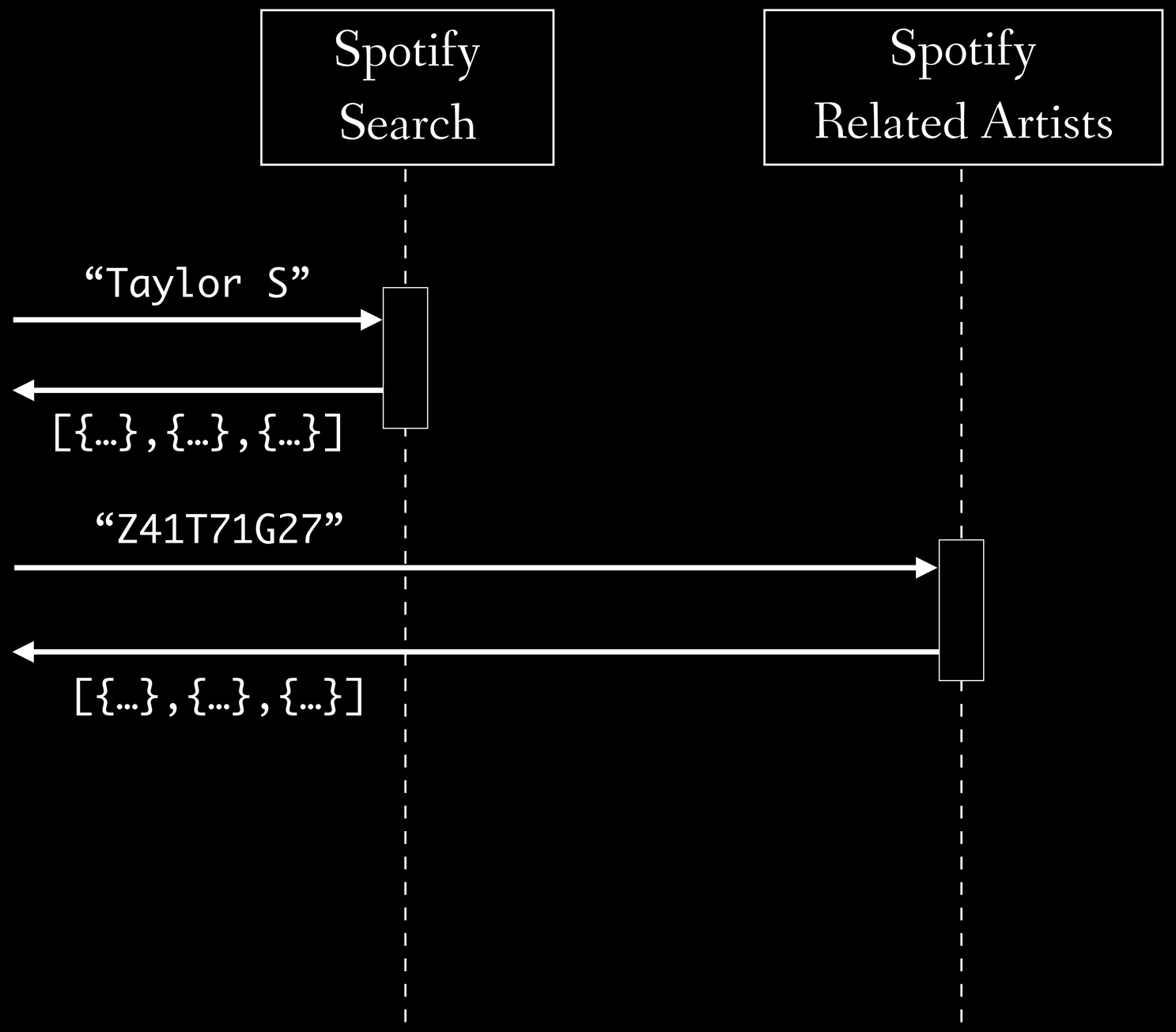
related artists:

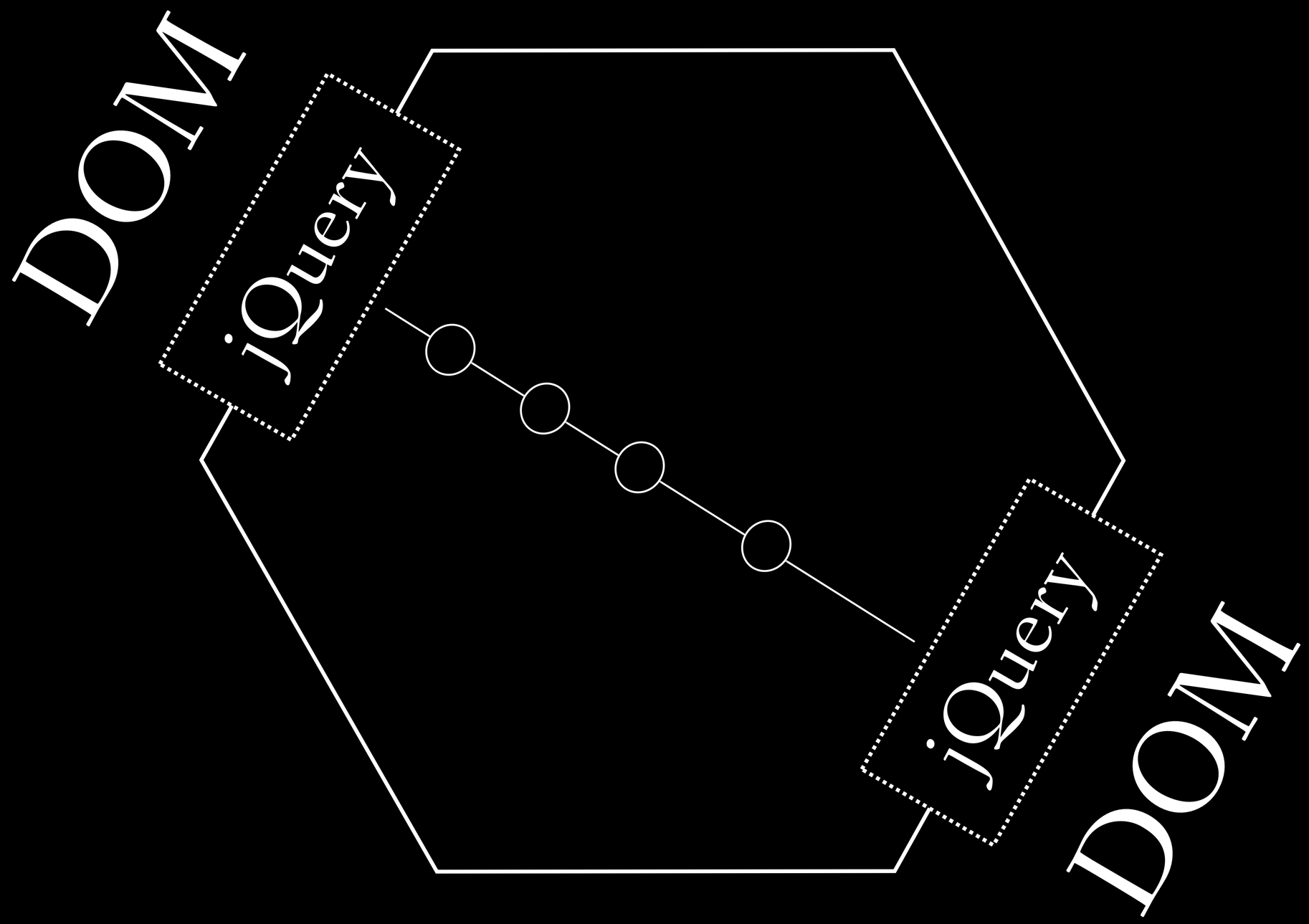
Miley Cyrus

Katie Perry

etc...


Spotify
Search
Spotify
Related Artists
“Taylor S”
[{…},{…},{…}]


Spotify Search
Spotify Related Artists
“Taylor S”
[{…},{…},{…}]
“Z41T71G27”
[{…},{…},{…}]


DOM
jQuery
jQuery
DOM

HTTP
node
streams
node
streams
HTTP

searchForArtist(“Taylor”)

```
{
  id: “0vnm2817Yf1P”,
  name: “James Taylor”,
  ...
}
```

```
{
  id: “5hc2QfaQcjApzO”,
  name: “Taylor Swift”,
  ...
}
```

searchForArtist(“Taylor”)

```
{
  id: “0vnm2817Yf1P”,
  name: “James Taylor”,
  ...
}
```

```
{
  id: “5hc2QfaQcjApz0”,
  name: “Taylor Swift”,
  ...
}
```

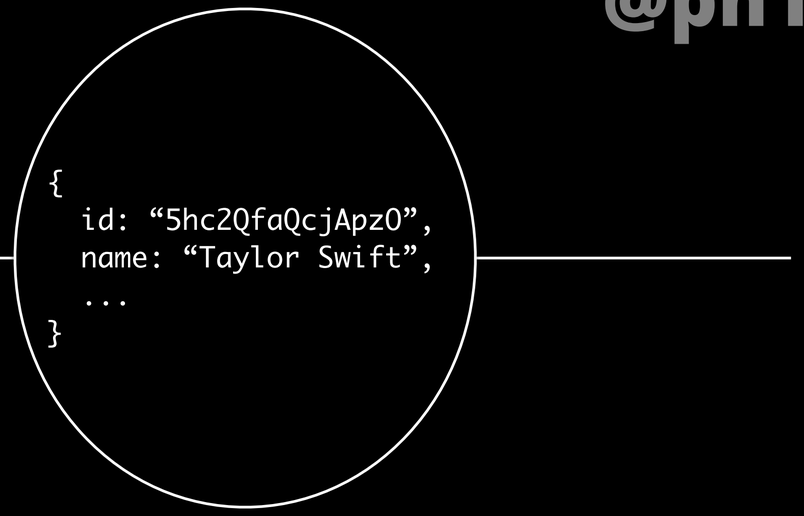
.take(1)

```
{
  id: “5hc2QfaQcjApz0”,
  name: “Taylor Swift”,
  ...
}
```

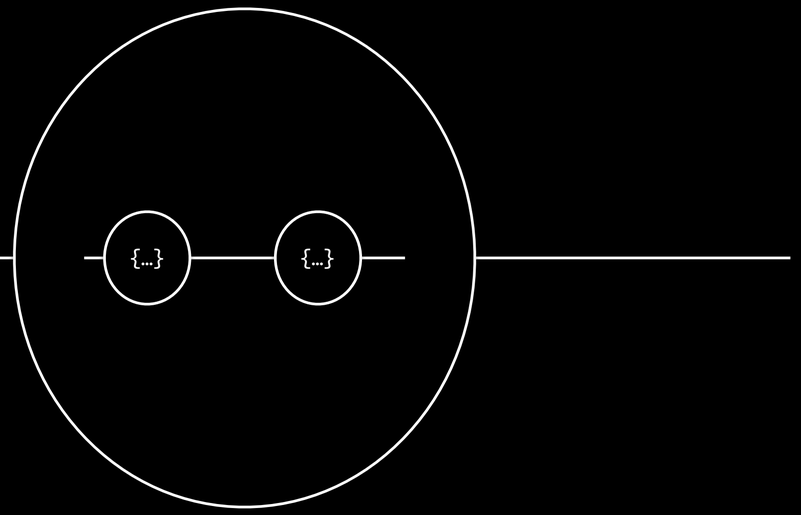
.take(1)

```
{
  id: “5hc2QfaQcjApzO”,
  name: “Taylor Swift”,
  ...
}
```

.take(1)

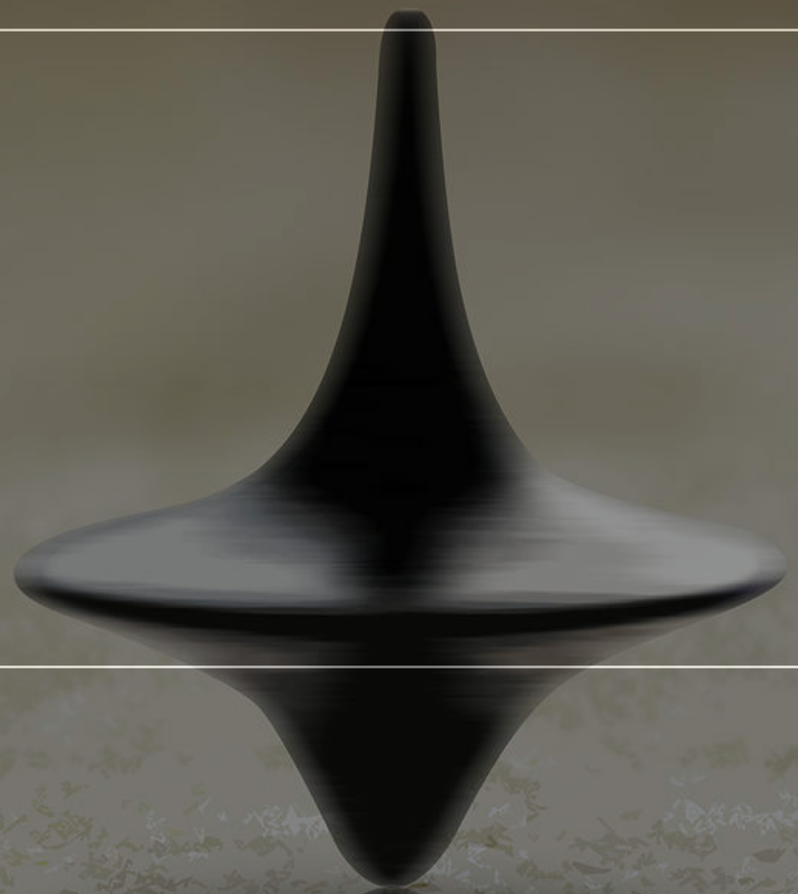


.map(relatedArtists)



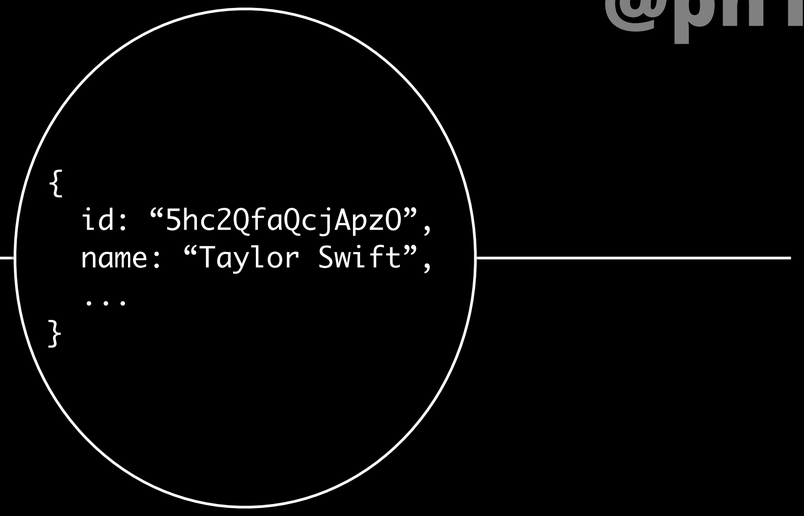
.take(1)

```
{
  id: “5hc2QfaQcjApzO”,
  name: “Taylor Swift”,
  ...
}
```

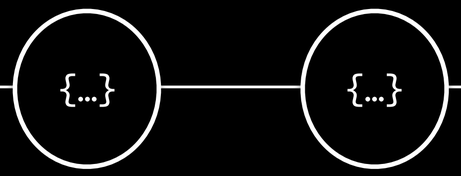


.map(relatedArtists)

.take(1)



.flatMap(relatedArtists)



relatedArtists

```
{
  id: “0vnm2817Yf1P”,
  name: “Miley Cyrus”,
  ...
}
```

```
{
 id: “5hc2QfaQcjApzO”,
 name: “Kelly Clarkson”,
 ...
}
```

@ph1

relatedArtists

```
{
  id: “0vnm2817Yf1P”,
  name: “Miley Cyrus”,
  ...
}
```

```
{
 id: “5hc2QfaQcjApzO”,
 name: “Kelly Clarkson”,
 ...
}
```

.map(renderFn)

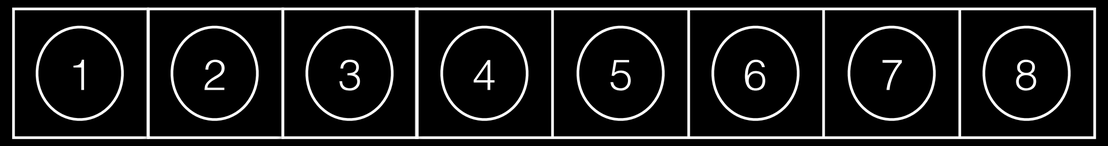
```
<li>
  Miley Cyrus
</li>
```

```
<li>
  Kelly Clarkson
</li>
```

HTTP
node
streams
node
streams
HTTP

reduce (aka *fold*)

summing an array



reduce (aka *fold*)

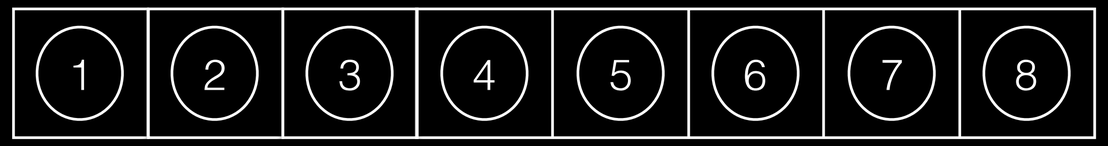
summing an array

1	2	3	4	5	6	7	8

1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 = <u>36</u>

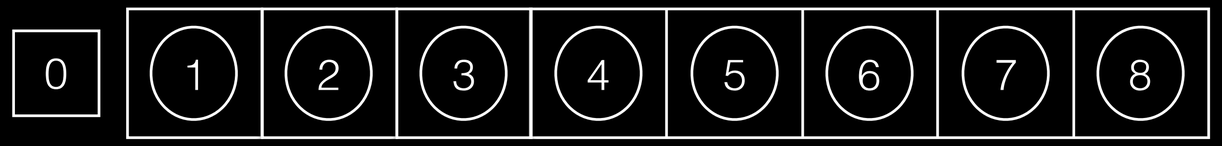
reduce (aka *fold*)

$$(a,b) => a+b$$



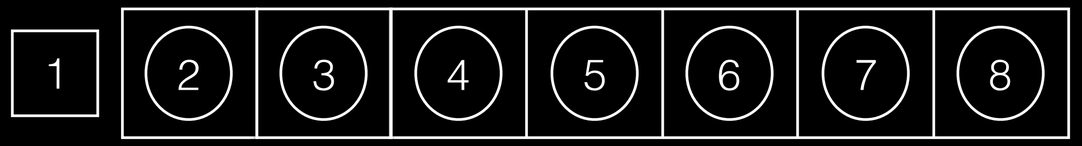
reduce (aka *fold*)

$$(a,b) => a+b$$



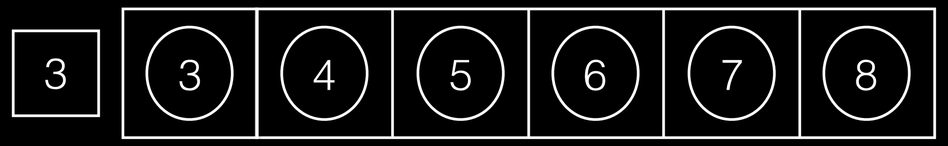
reduce (aka *fold*)

$$(a,b) => a+b$$



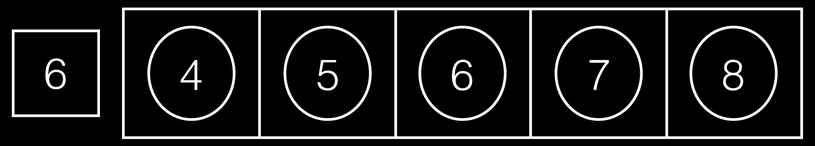
reduce (aka *fold*)

$$(a,b) => a+b$$



reduce (aka *fold*)

$$(a,b) => a+b$$



reduce (aka *fold*)

$$(a,b) => a+b$$

36

reduce (aka *fold*)

$(a,b) => a+b$

0

numbers

reduce (aka *fold*)

$(a,b) => a+b$

reduce (aka *fold*)

$(a,b) => a+b$

reduce (aka *fold*)

$(a,b) => a+b$

reduce (aka *fold*)

$(a,b) => a+b$

reduce (aka *fold*)

$(a,b) => a+b$

scan

0

numbers

summed numbers

scan

scan

scan

scan

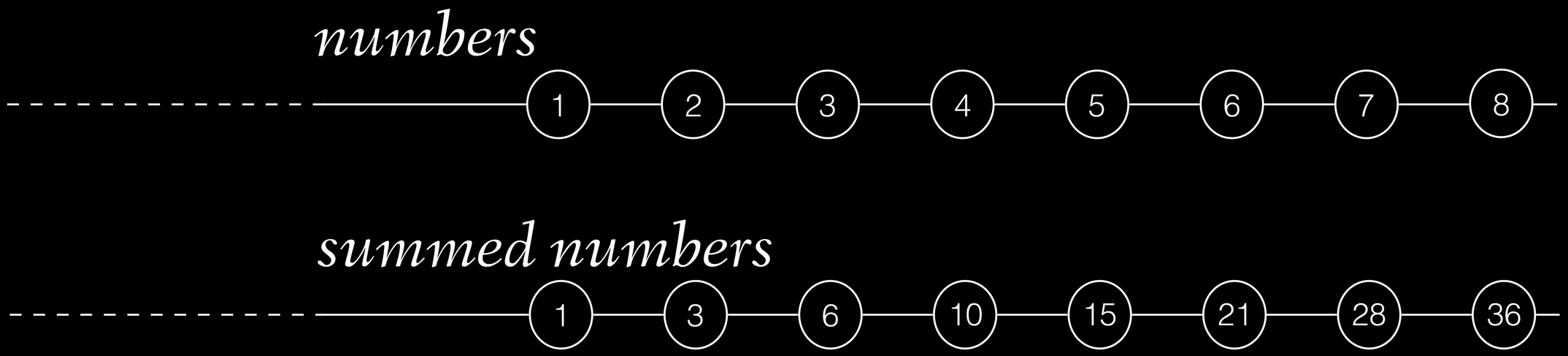
scan

scan

scan

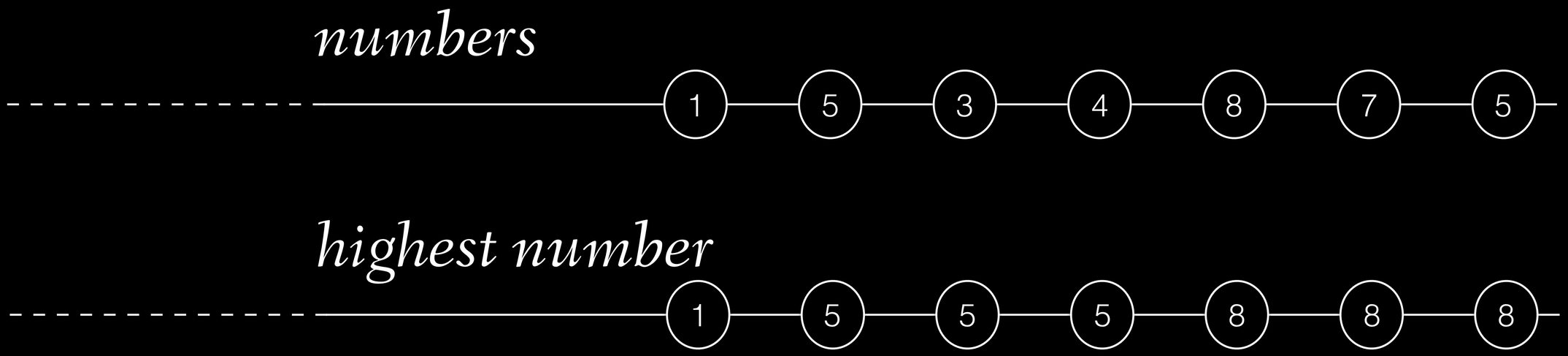
scan

scan



scan

$$(a,b) => Math.max(a,b)$$



what is FRP?

what is FRP?

a powerful, framework-agnostic way to work with event streams

a unifying universal abstraction

a declarative way to model relationships between values in our programs

shared mutable state

@ph1


FOOD GROUPS
@ph1
thanks!