Queues

Simon Robinson
SOFTWARE DEVELOPER
@techiesimon www.simonrobinson.com

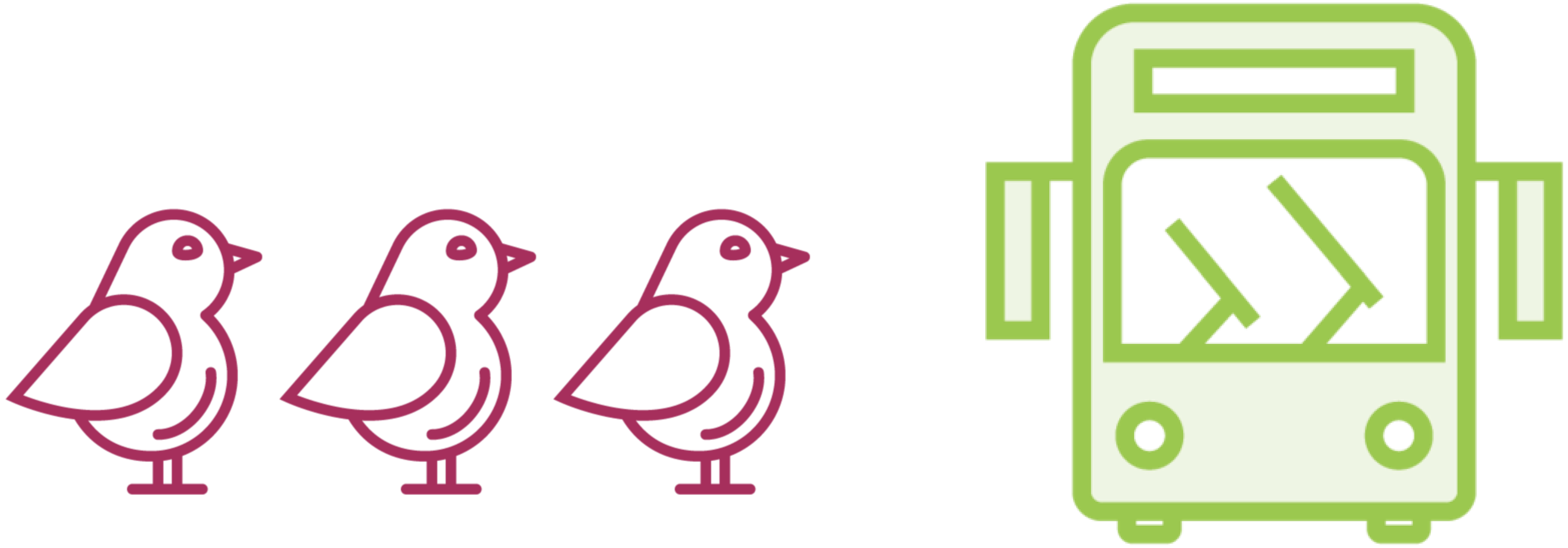
Overview

The `Queue<T>` Collection

- Similar to `Stack<T>`
- But supplies longest waiting item
- Great for storing tasks to be processed

Queue<T>

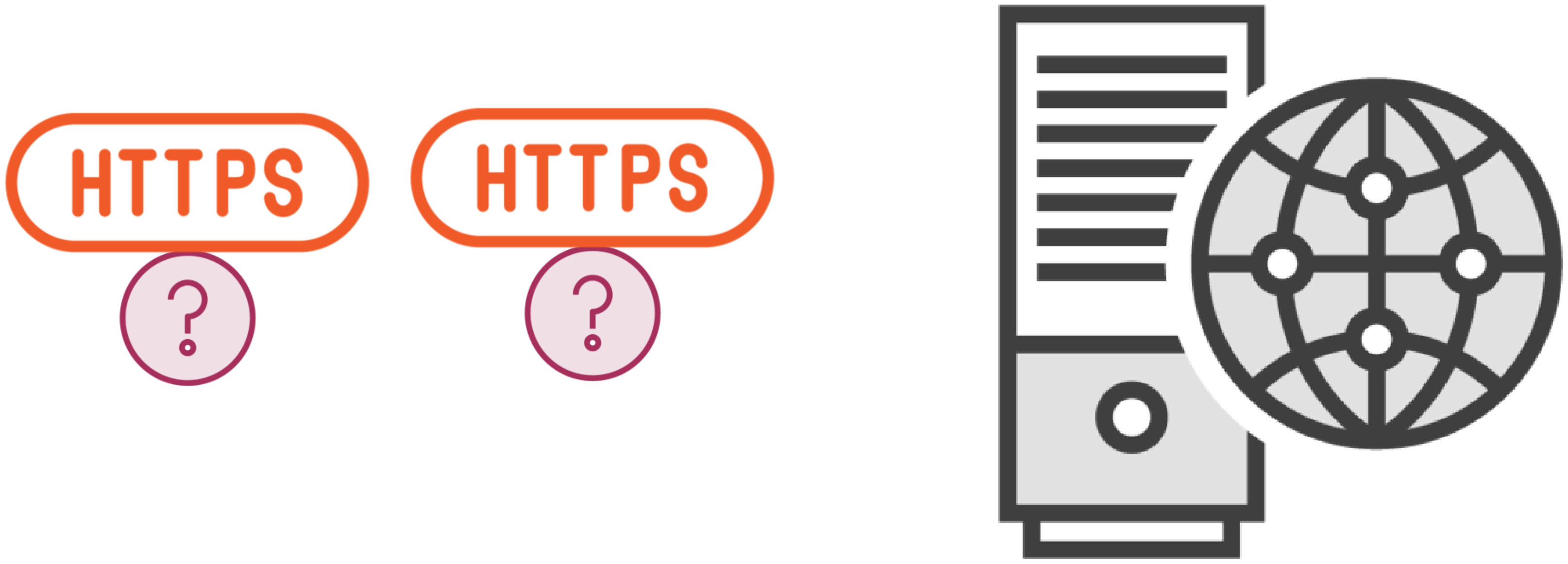
Queue<T> Works just like waiting in line for a bus!



Queue<T>

Another example: Tasks

For example: Browser requests to a website:



Demo

Using a queue

- Customers log in and view tours
- Request tours
- Tour operator confirms each request
- Requests should be confirmed in order

This requires a queue!

I could have done this:

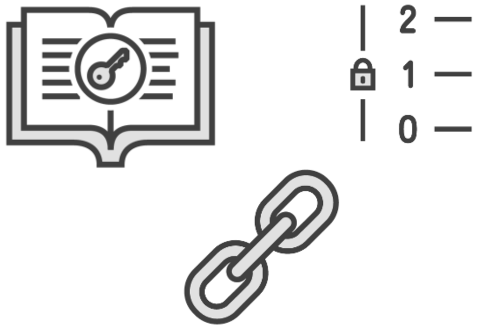
```
public struct BookingRequest
{
        public Customer Customer { get; }
        public Tour RequestedTour { get; }
        // etc.
}
```

Leading to this:

```
public Queue<BookingRequest> BookingRequests { get; }
           = new Queue<BookingRequest>();
```

(But I didn’t)

Collection Terminology

	Most collections:	Stacks:	Queues:
			
Adding an item:	Add/Insert	Push	Enqueue
Removing an item:	Remove	Pop	Dequeue

Queue<T> vs. Stack<T>

`Queue<T>`

`Stack<T>`

List of stuff to be processed

No direct lookup

Collection decides which element you get next

Can enumerate with a `foreach` loop

Provides longest waiting item

First-in first-out (FIFO)

Enqueue/Dequeue

Provides most recent item

Last-in first-out (LIFO)

Push/Pop

Summary

Queues and stacks are very similar

- Use a queue to process items in same order as added them

Next up: Queues with multiple threads