Adding Data Efficiently with Linked Lists

Simon Robinson

Software Developer

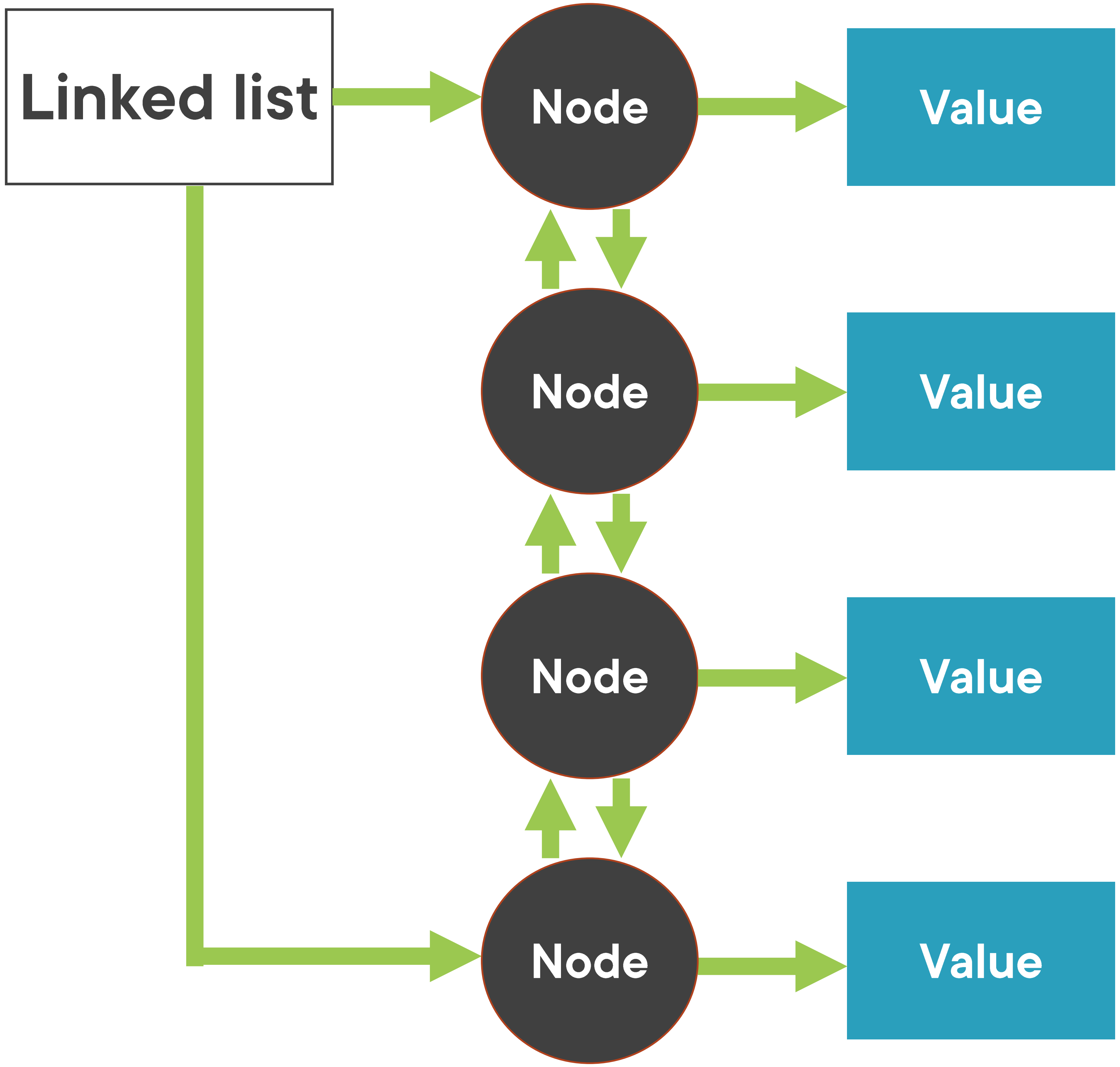
@TechieSimon www.SimonRobinson.com

Overview

Linked lists

- Optimized for frequent adding/removal
- Caller decides which item to remove

Linked Lists



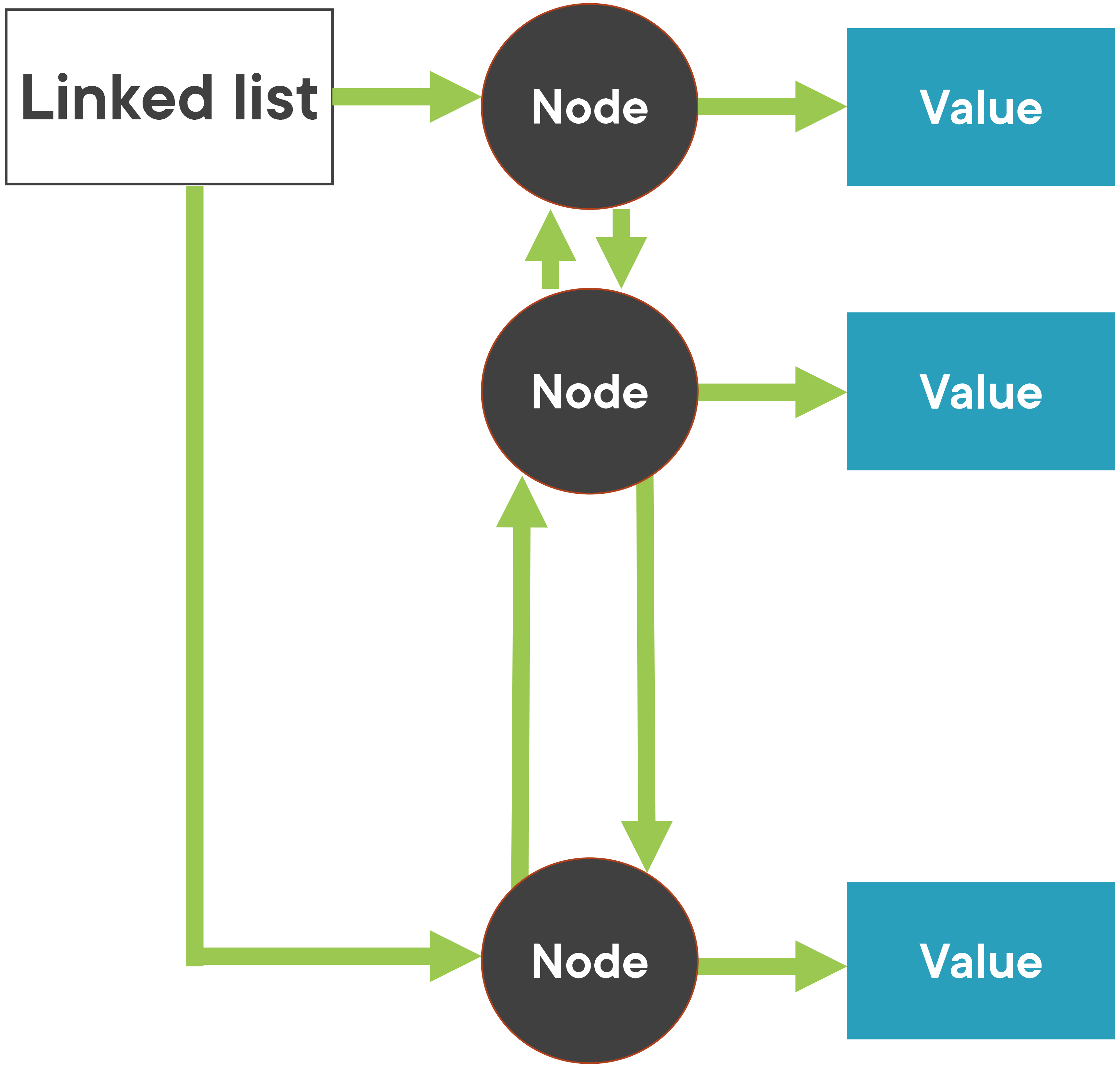
Values packaged into nodes

Linked list only knows first and last nodes

Knowledge of elements is completely distributed

Adding/removing done by updating node links

Linked Lists



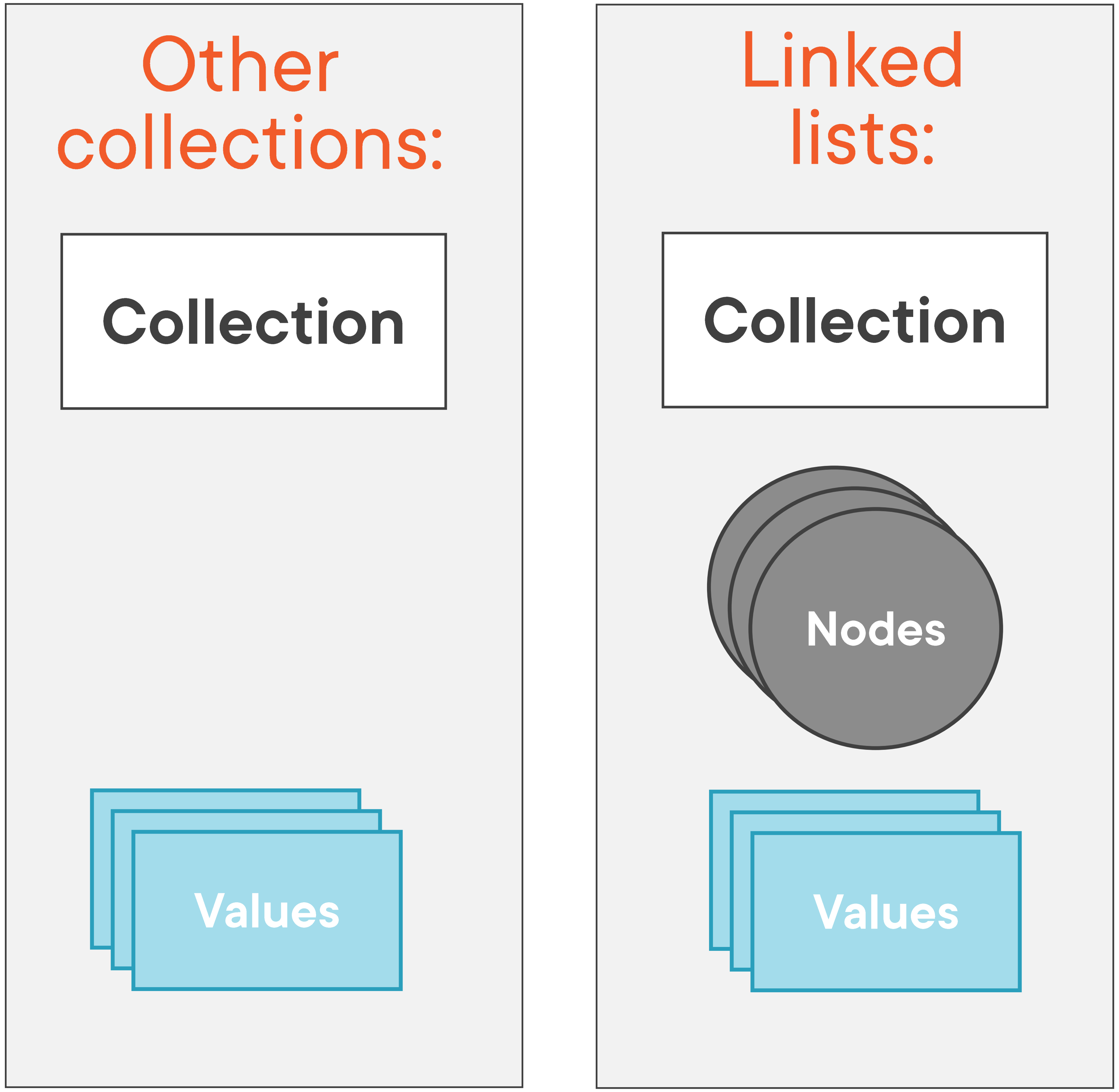
Adding and removing is very fast

Linked Lists

⚠ **Complicated**

No direct look-up

✓ **Can be very fast**



Other collections:

Collection

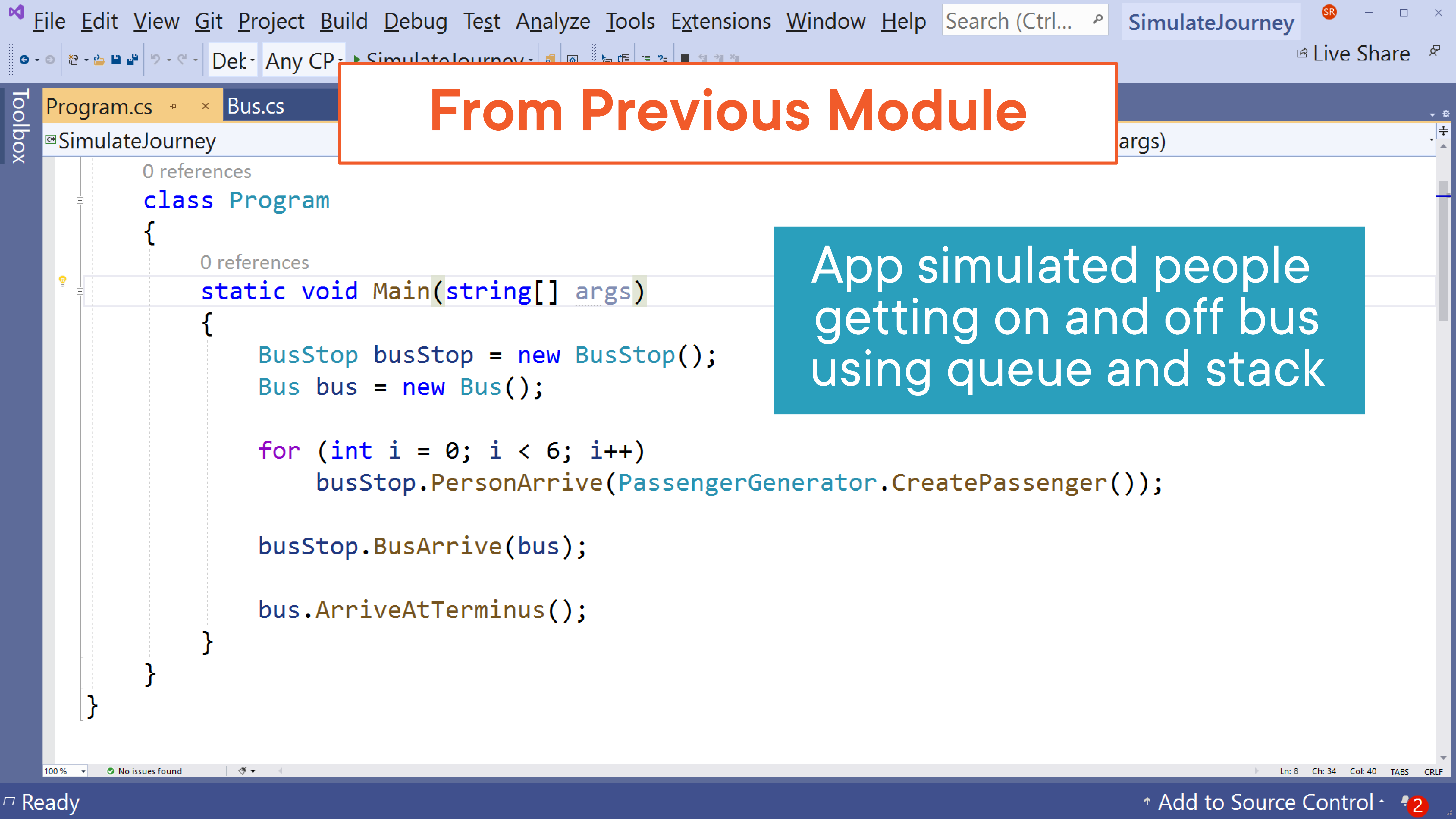
Values

Linked lists:

Collection

Nodes

Values



From Previous Module

App simulated people getting on and off bus using queue and stack

```
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            BusStop busStop = new BusStop();
            Bus bus = new Bus();

            for (int i = 0; i < 6; i++)
                busStop.PersonArrive(PassengerGenerator.CreatePassenger());

            busStop.BusArrive(bus);

            bus.ArriveAtTerminus();
        }
    }
}
```

Demo

Have bus stop at different places

- Order of people getting off doesn’t matter
- They must get off at the correct destination
- `LinkedList<T>` to store people on bus – to make adding/removing to list efficient

Course Summary

Collections – to remember!

- Group objects together
- Different collections:
  - Arrays
  - Lists
  - Dictionaries and sorted dictionaries
  - Sets
  - Queues, stacks and linked lists

Thanks for watching!