Inside Dictionaries and Sorted Dictionaries

Simon Robinson
SOFTWARE DEVELOPER

@techiesimon www.simonrobinson.com

Overview

Dictionary, SortedDictionary, SortedList

Case sensitive keys

Custom types as keys

- Implementing `Equals()` and `GetHashCode()`

Demo

TourBooker app

- Countries currently stored as a list
- But lists are not good for lookup
- Fix by using a dictionary

To change how keys are compared

- Need to tell the dictionary upfront
- Because dictionary must adjust internal storage
- Use an equality comparer

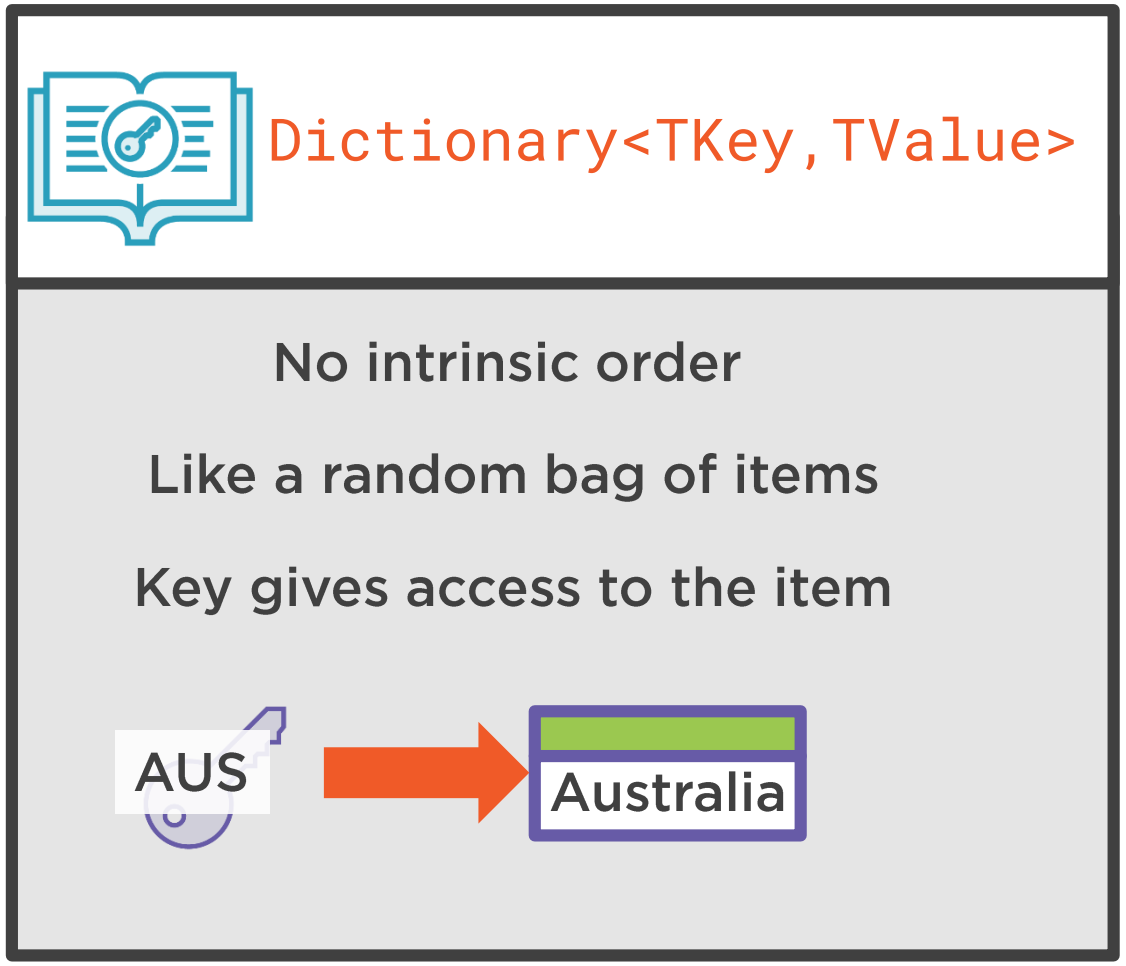
Equality Comparer

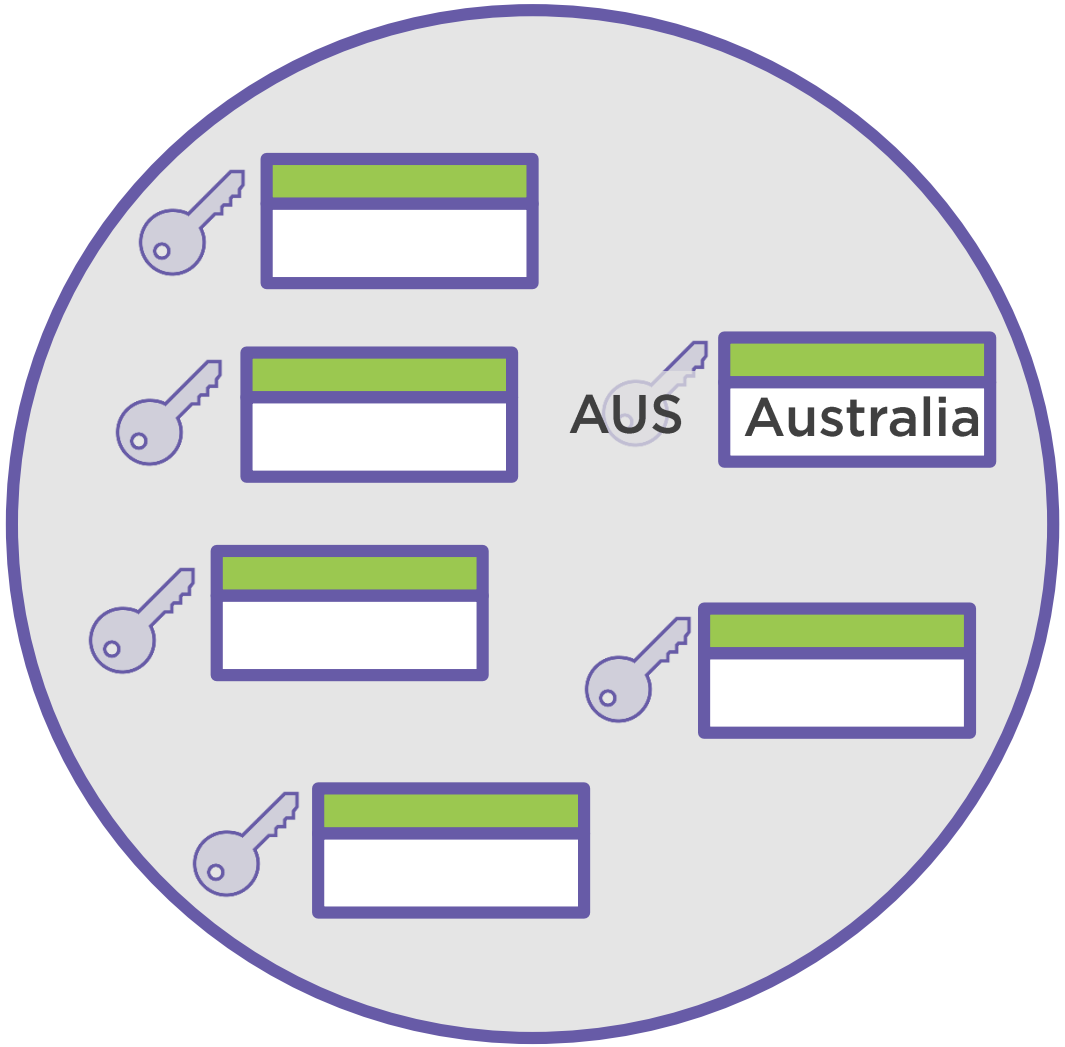
Object that knows how to test for equality.

- Implements `IEqualityComparer<T>`

Enumerating a Dictionary

Dictionaries





Dictionaries

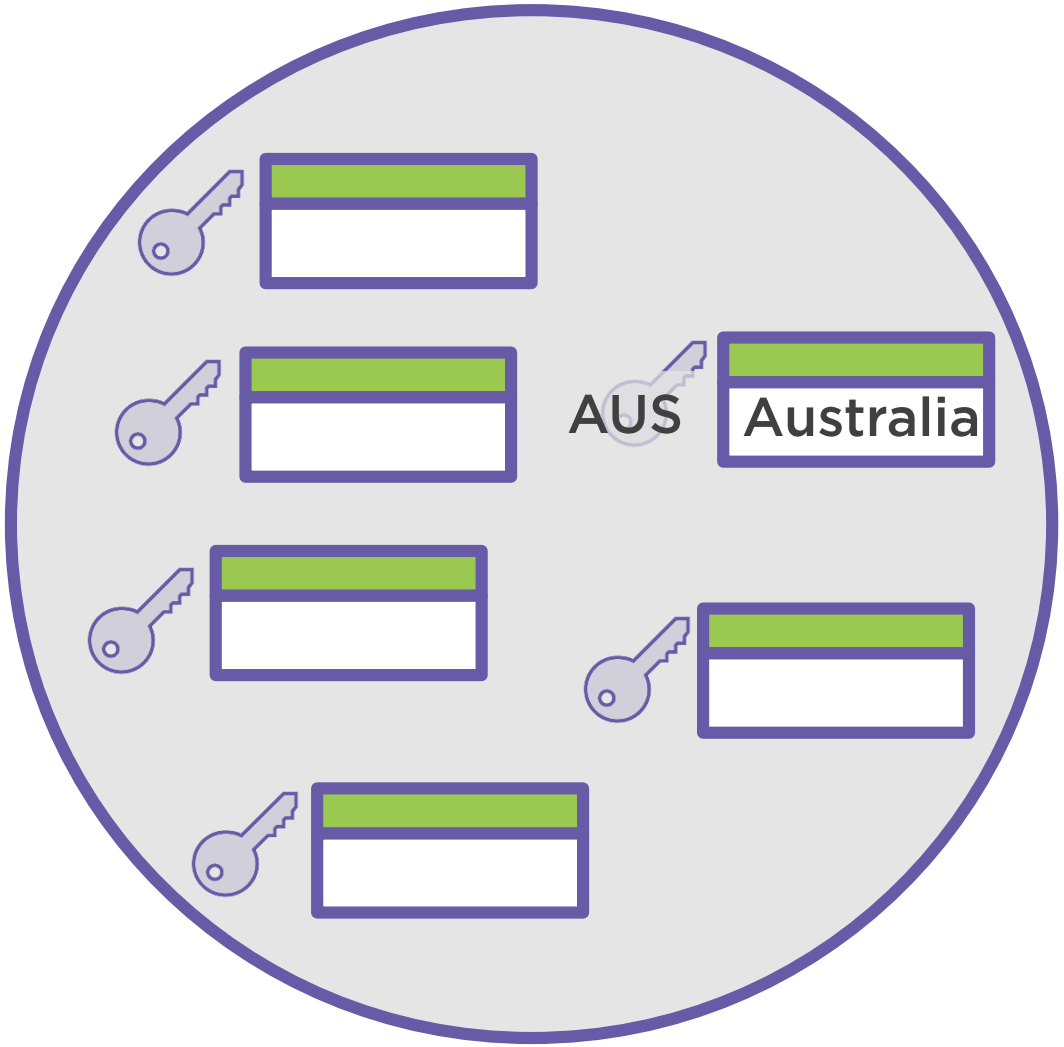
`Dictionary<TKey,TValue>`

No intrinsic order

Like a random bag of items

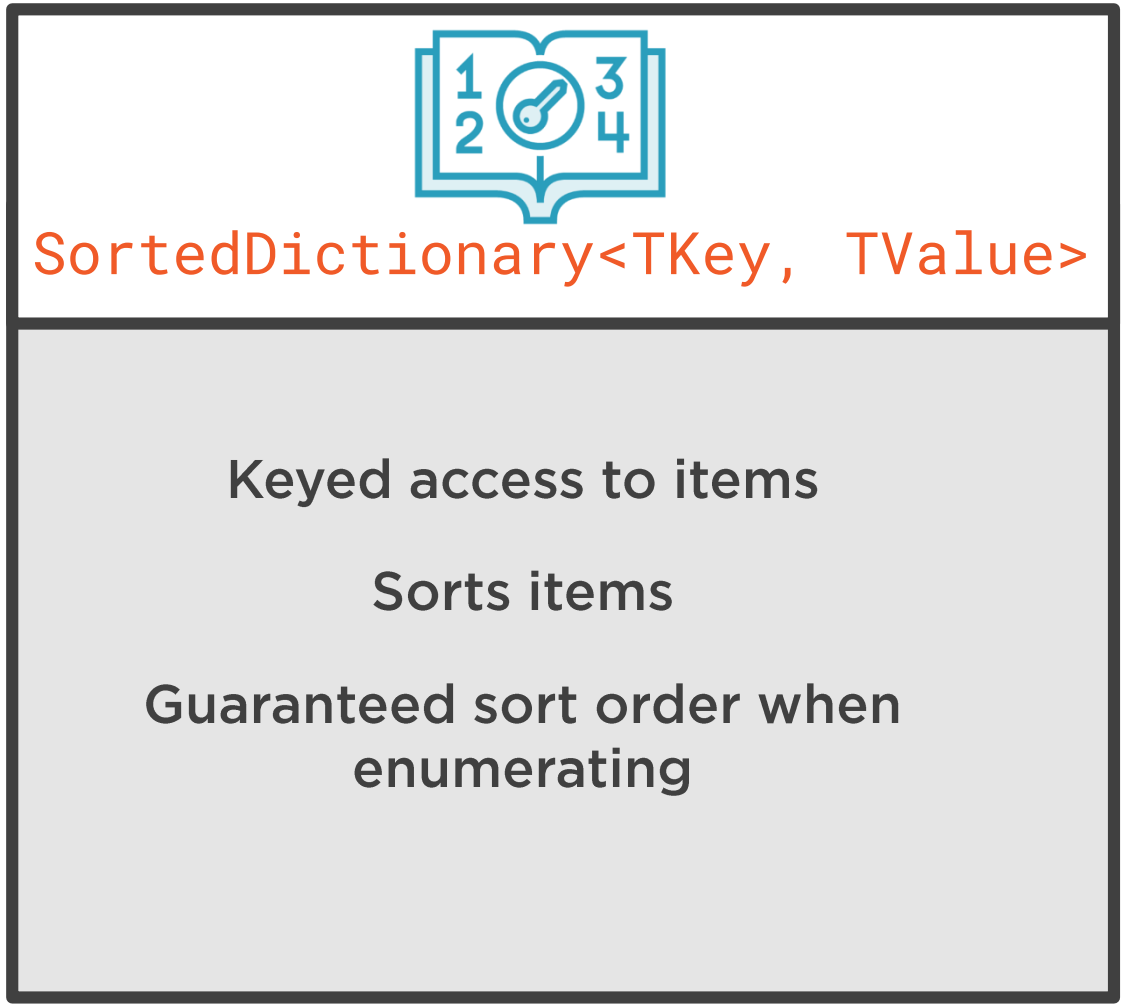
Key gives access to the item

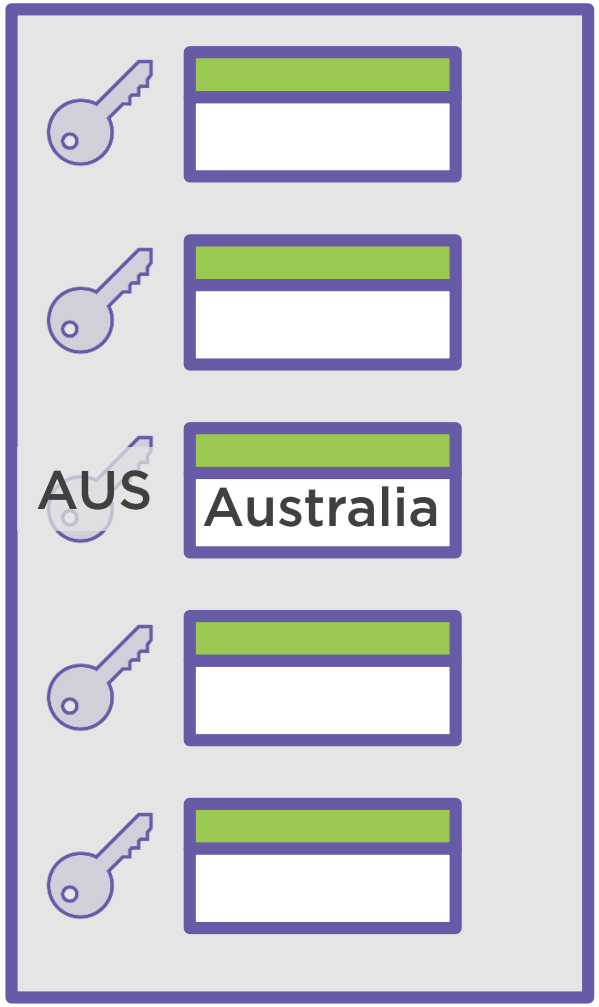
Do not rely on the order!



Sorted Dictionaries

SortedDictionary





Demo

SortedDictionary to order countries

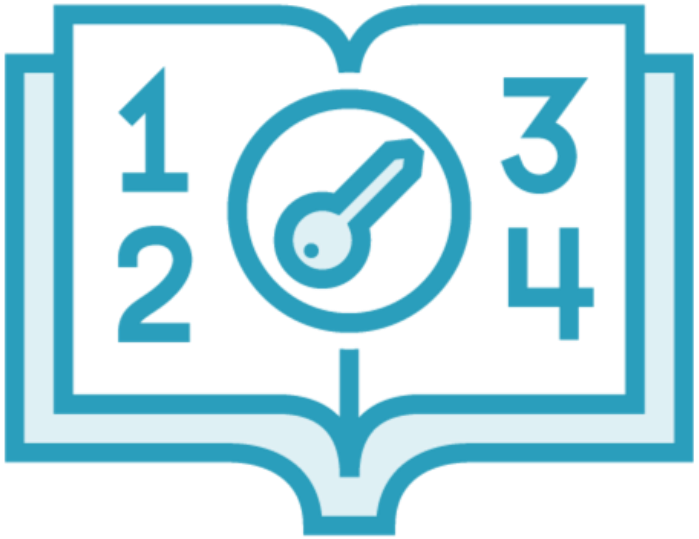
- But will still have problems

SortedDictionary sorts items – but can only sort in order of keys

SortedDictionary<Tkey, Tvalue>

You'll see
a perfect scenario
for
SortedDictionary
later in this
course

But this
list of countries
isn't it!



`SortedList<TKey, TValue>`

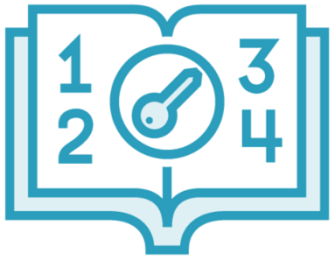
- Functionally a dictionary
- Internally uses a list

SortedList vs. SortedDictionary

`SortedList<TKey, TValue>`

Functionally the same

Uses less memory



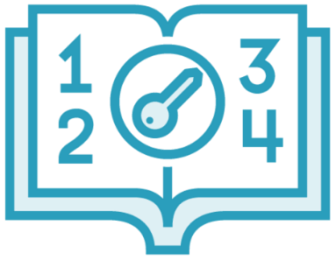
`SortedDictionary<TKey, TValue>`

Functionally the same

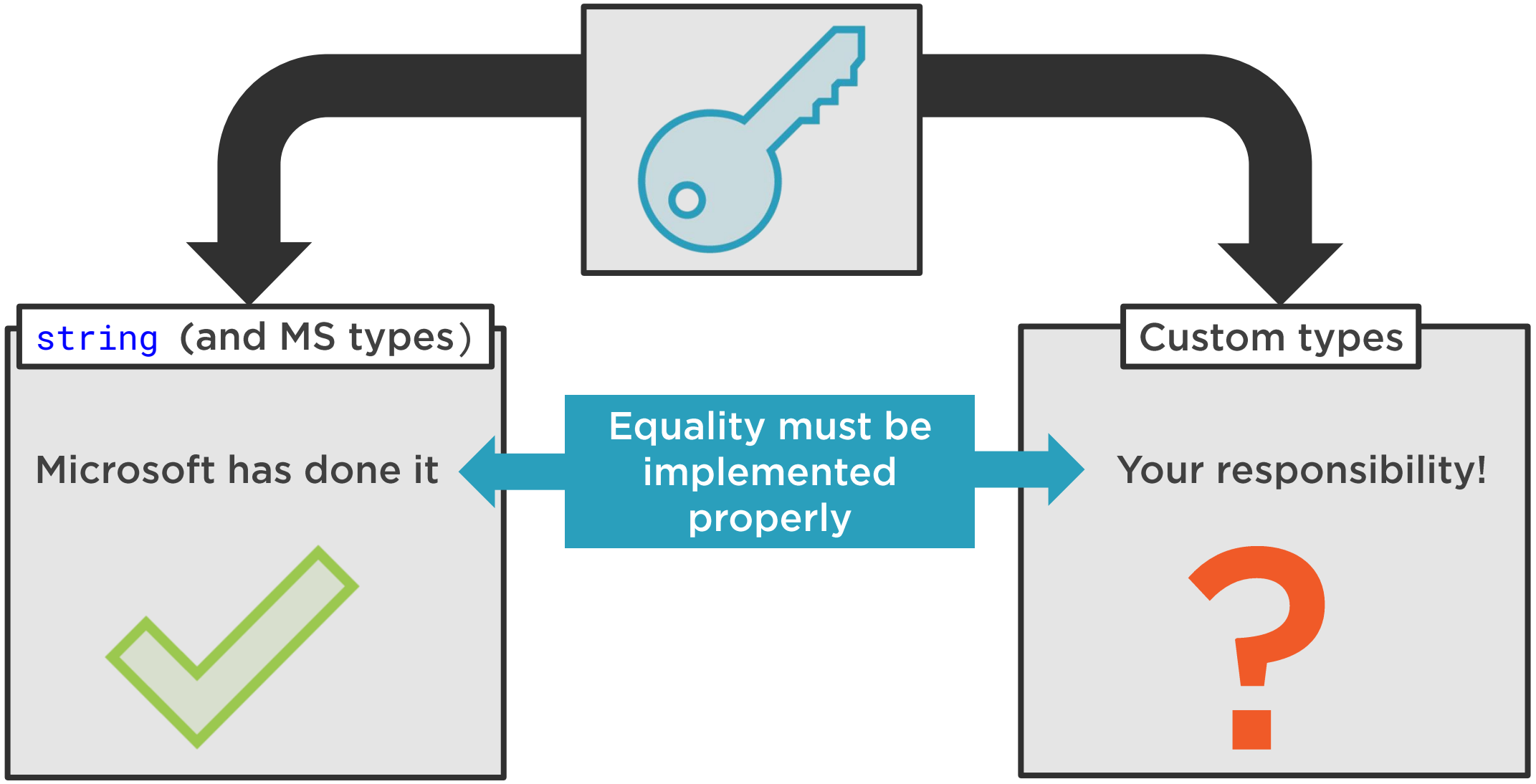
Modifications scale better

- O(log n) vs O(n)

Name of the type is clearer



Dictionary Keys

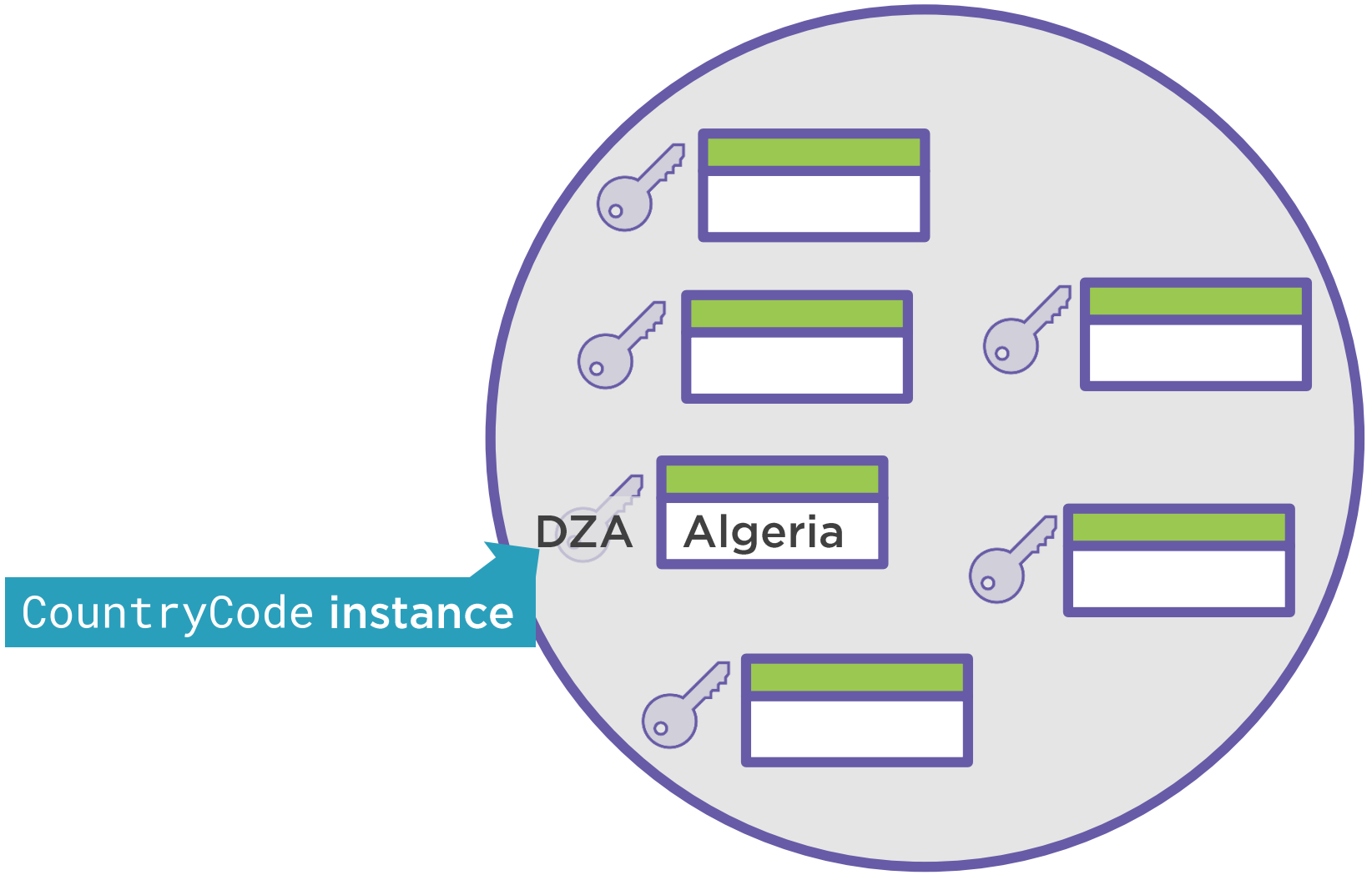


Demo

No longer use string for country code

- Custom country code type
- This type will be a key

Dictionaries



String Equality

```
string bankHol1Name = "New Year's Day";
string bankHol2Name = "New Year's Day";
bool areEqual = (bankHol1Name == bankHol2Name);
```

This does test whether values are equal

That's why strings work out of the box as dictionary keys

Object.GetHashCode()

Generates a hash code

Mainly used under the hood in .NET

Dictionaries rely on GetHashCode() extensively

Lookup will fail if key doesn't implement GetHashCode()

C# Equality and Comparisons

by Simon Robinson

This course teaches you how equality and comparisons function in .NET and the correct way to implement equality and comparisons for your own types.

Start Course | Bookmarked | Add to Channel | Download Course

Table of contents | Description | Transcript | Exercise files | Discussion | Recommended

Course author

Simon Robinson

Simon Robinson first cut his developer teeth in the early 1980s writing a scheduling system in BBC Basic(!) for his local college. Since then, his programming career has spanned industries ranging...

Course info

Level	Intermediate
Rating	★★★★½ (613)
My rating	★★★★★
Duration	4h 51m

To use a custom type as a dictionary key:

- Override `object.Equals()`
- Override `object.GetHashCode()`
  - Easiest way is to use existing MS implementations

Even easier: Use standard MS types as keys

Summary

Looking up by key

- Custom comparer for case insensitivity

Dictionary enumeration

- KeyValuePairs
- SortedDictionary/SortedList to guarantee order

Custom types as keys

- Need to implement `Equals()` and `GetHashCode()`

Next up: Linked lists