Merging Data with HashSets and SortedSets

Simon Robinson
SOFTWARE DEVELOPER

@techiesimon www.simonrobinson.com

Overview

Sets

- Enforce uniqueness
- Operations on whole collections
  - E.g. merging collections

Demo

A new task:

- Listing countries on tours
- Will require sets

Dictionary vs. HashSet

`Dictionary<TKey,TValue>`

Have keys

- Key-based lookup

Keys are unique

Adding duplicates throws exception

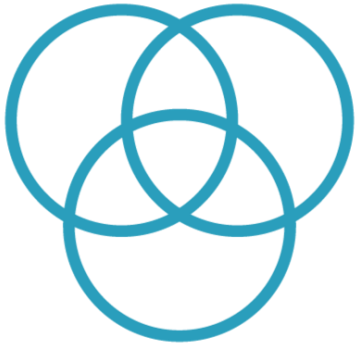
`HashSet<T>`

Do not have keys

- Do not support lookup

Values are unique

Duplicates are ignored



Code Demo

Placeholder for code demo

This slide will not appear in the course.

Dictionaries and sets both sometimes require comparers

Comparing Items

`Dictionary` **or** `HashSet`

Must compare items for equality

`SortedDictionary/SortedList` **or** `SortedSet`

Must compare items for ordering

Comparing Dictionary and Set Elements

Rely on Microsoft implementations

- For standard types: `int`, `string`, etc.

Implement comparing within the type

Provide a separate comparer type

Set

A collection of objects.

Union of Two Sets

A set that contains everything from each of the two sets

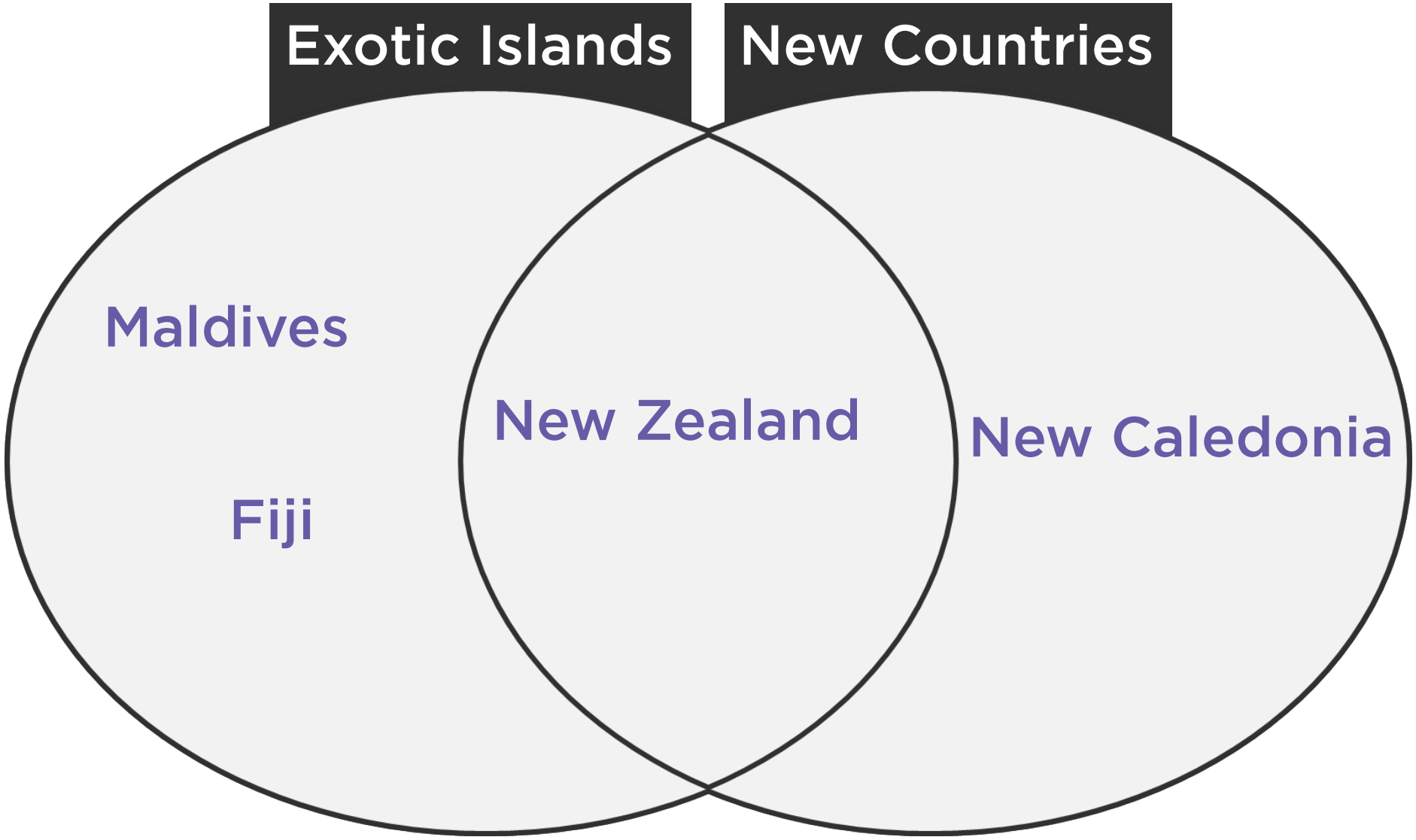
Merging Sets: Union and Intersection

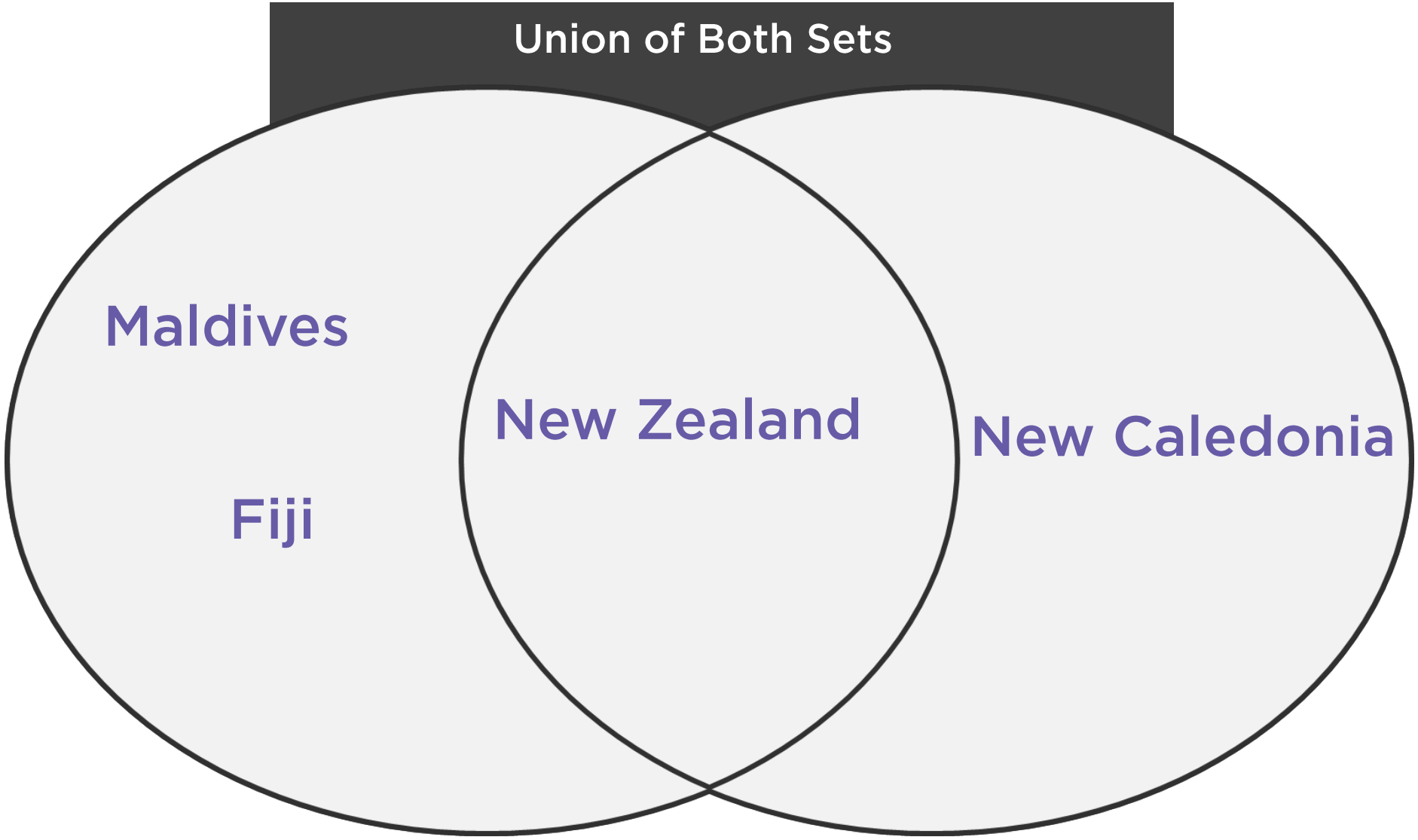
Demo

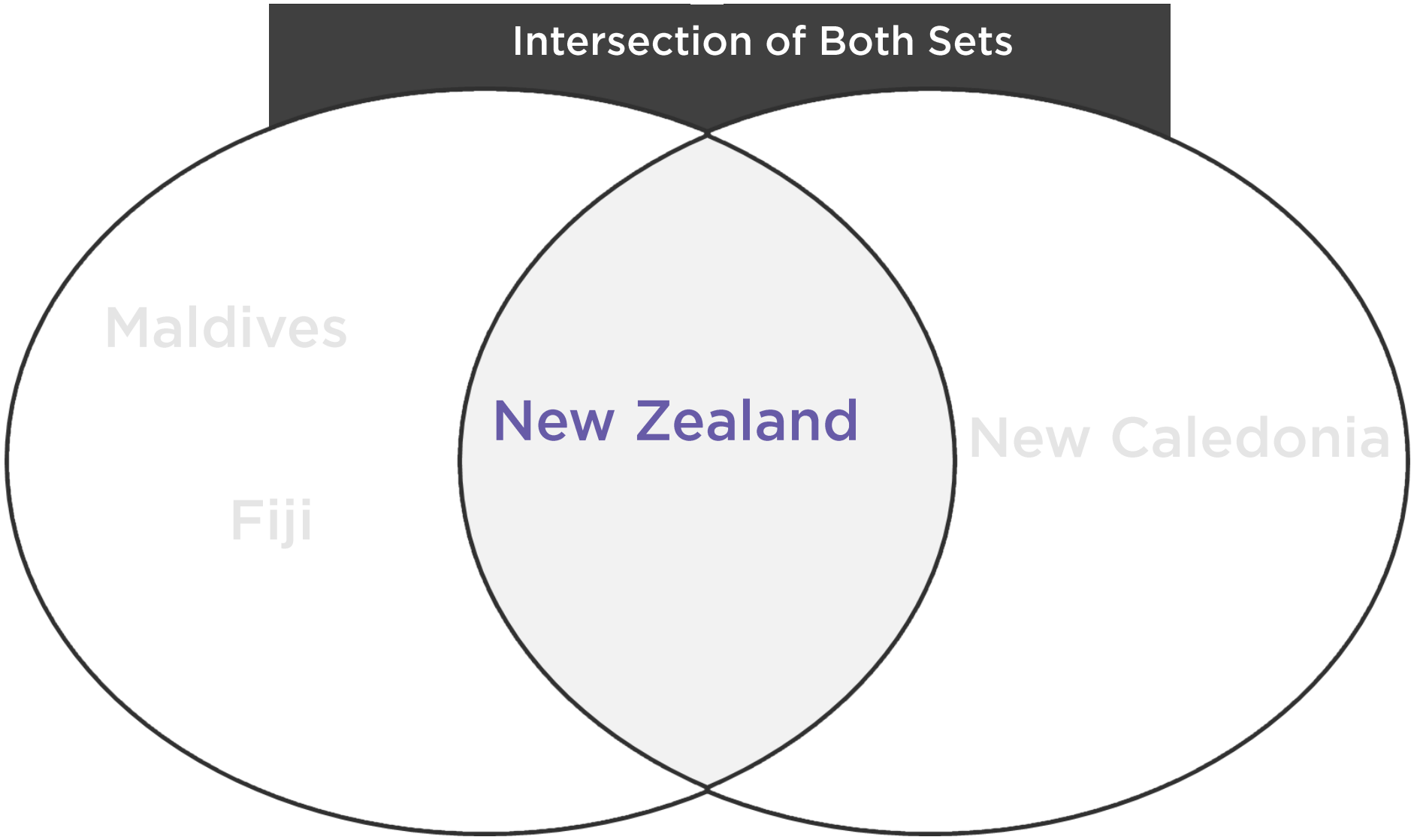
Merging

- Replace `GetCountriesInSelection()` implementation
  - Create a set for each tour
  - Then merge all these sets
- Same result as before, different technique


Exotic Islands
Maldives
New Zealand
Fiji


Exotic Islands
New Countries
Maldives
Fiji
New Zealand
New Caledonia


Union of Both Sets
Maldives
Fiji
New Zealand
New Caledonia


Intersection of Both Sets
Maldives
New Zealand
New Caledonia
Fiji

Union and intersection are set operations because they combine different sets to create a new set

Summary

Sets

- Enforce uniqueness
- Set operations (Intersection, union)
- Similar to dictionaries
  - But lack keys
  - Don’t support lookup

May need to implement comparisons

Next up: Protecting collections