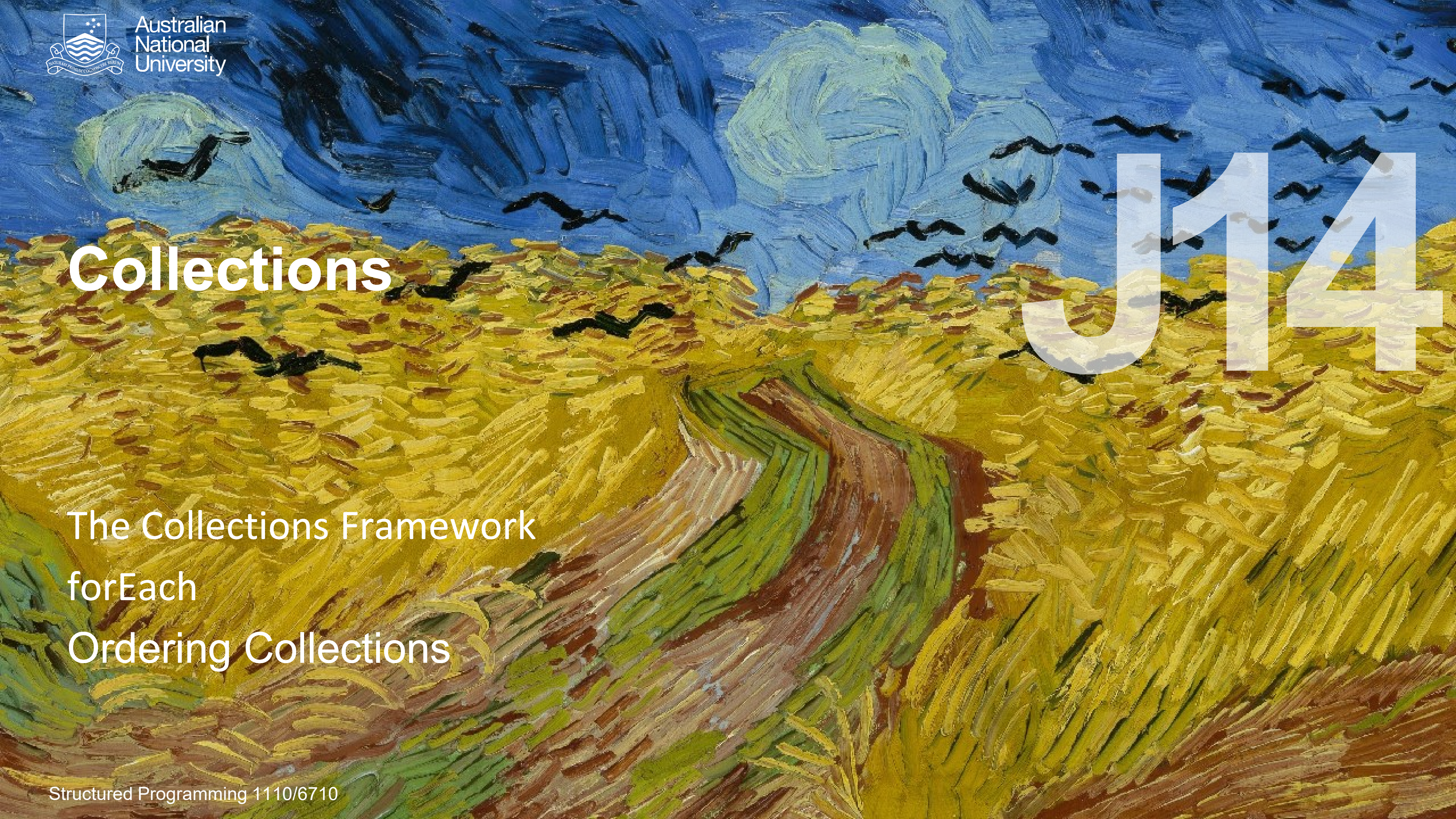

Australian National University

J14

Collections

The Collections Framework

forEach

Ordering Collections

The Collections Framework

- Interfaces
  - Implementation-agnostic interfaces for collections
- Implementations
  - Concrete implementations
- Algorithms
  - Searching, sorting, etc.

Using the framework saves writing your own – better performance, fewer bugs, less work, etc.

TEQSA PROVIDER ID: PRV12002 (AUSTRALIAN UNIVERSITY) CRICOS PROVIDER CODE: 00120C

The Collection Interface

- Basic Operators
  - `size(), isEmpty(), contains(), add(), remove()`
- Traversal
  - iterators and streams, `forEach`
- Bulk Operators
  - `containsAll(), addAll(), removeAll(), retainAll(), clear()`
- Array Operators
  - Convert to and from arrays

TEQSA PROVIDER ID: PRV12002 (AUSTRALIAN UNIVERSITY) CRICOS PROVIDER CODE: 00120C

Collection Types

- Primary collection types:
  - Set (no duplicates, mathematical set)
  - List (ordered elements)
  - Queue (shared work queues)
  - Map (key-value pairs)
- Each collection type is defined as an interface
  - You need to choose a concrete collection
  - Your choice will depend on your needs

TEQSA PROVIDER ID: PRV12002 (AUSTRALIAN UNIVERSITY) CRICOS PROVIDER CODE: 00120C

Concrete Collection Types

	Implemented Using				
Interfaces	Hash table	Resizable array	Tree	Linked list	Hash table + linked list
Set	HashSet		TreeSet		LinkedHashSet
List		ArrayList		LinkedList	
Queue		ArrayDeque		LinkedList	
Map	HashMap		TreeMap		LinkedHashMap

Based on table from http://docs.oracle.com/javase/tutorial/collections/implementations/index.html

TEQSA PROVIDER ID: PRV12002 (AUSTRALIAN UNIVERSITY) CRICOS PROVIDER CODE: 00120C

Four Commonly Used Collection Types

- `HashSet` implements a ***set*** as a hash table
  - Makes no ordering guarantees
- `ArrayList` implements a ***list*** using an array
  - Very fast access
- `HashMap` implements a ***map*** using a hash table
  - Makes no ordering guarantees
- `LinkedList` implements a ***queue*** using a linked list
  - First-in-first-out (FIFO) queue ordering

TEQSA PROVIDER ID: PRV12002 (AUSTRALIAN UNIVERSITY) CRICOS PROVIDER CODE: 00120C

Iterators and “enhanced” for

Iterators are objects that keep track of where you are in a list

- Really useful for e.g. linked lists
- Key methods: `hasNext()` and `next()`

```
for(String s : strings) { … }
```

translates to

```
Iterator<String> iter = strings.iterator();
while(iter.hasNext()) {
    String s = iter.next();
    …
}
```

TEQSA PROVIDER ID: PRV12002 (AUSTRALIAN UNIVERSITY) CRICOS PROVIDER CODE: 00120C

forEach

Collections implement the forEach method, which applies an action to every element in the collection.

Instead of

```
for(Thing t : things) {
  System.out.println(t);
}
```

You can write:

```
things.forEach(t -> { System.out.println(t); });
```

TEQSA PROVIDER ID: PRV12002 (AUSTRALIAN UNIVERSITY) CRICOS PROVIDER CODE: 00120C

Ordering Collections

The Comparable interface defines a “natural” ordering for all instances of a given type, T:

```
public interface Comparable<T> {
    int compareTo(T o);
}
```

The Comparator interface allows a type T to be ordered in other ways:

```
public interface Comparator<T> {
    int compare(T o1, T o2);
}
```

Return values for both are either 0 (equal), negative (left comes first) or positive (right comes first).

TEQSA PROVIDER ID: PRV12002 (AUSTRALIAN UNIVERSITY) CRICOS PROVIDER CODE: 00120C

Collections.sort()

No arguments

- uses natural order (i.e. `Comparable`) for type

Single lambda argument:

- uses order defined by lambda expression (i.e. `Comparator`)

TEQSA PROVIDER ID: PRV12002 (AUSTRALIAN UNIVERSITY) CRICOS PROVIDER CODE: 00120C

Effective Java Item 25: Prefer lists to arrays

Arrays are covariant, Generics invariant

If `A extends B`:

- `A[]` is a subtype of `B[]`
- But `List<A>` has no relationship to `List<B>`

```
// Fails at runtime!
Object[] objectArray = new Long[1];
objectArray[0] = "I don't fit in";
// Won't compile!
List<Object> ol = new ArrayList<Long>(); // Incompatible
types
ol.add("I don't fit in");
```

TEQSA PROVIDER ID: PRV12002 (AUSTRALIAN UNIVERSITY) CRICOS PROVIDER CODE: 00120C