

Sorting at Scale

Comparable

Comparable Interface

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

- Comparable
 - Promises one method: **compareTo**
 - Defines **natural ordering** for type

Comparable Interface

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

- Comparable
 - Promises one method: **compareTo**
 - Defines **natural ordering** for type
 - T specifies type to compare to
 - For QuakeEntry: Comparable<QuakeEntry>

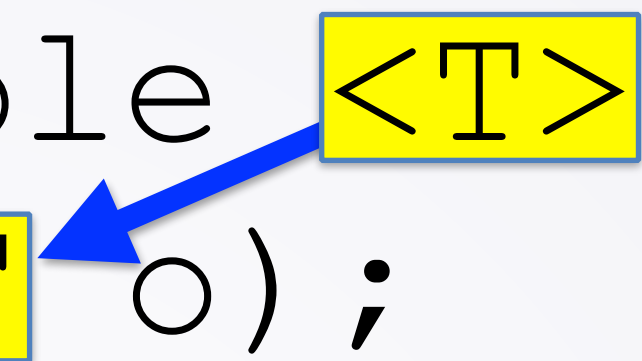
Comparable Interface

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

- Comparable
 - Promises one method: **compareTo**
 - Defines **natural ordering** for type
 - T specifies type to compare to
 - For QuakeEntry: Comparable<QuakeEntry>

Comparable Interface

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



- Comparable
 - Promises one method: **compareTo**
 - Defines **natural ordering** for type
 - T specifies type to compare to
 - For QuakeEntry: Comparable<QuakeEntry>

Comparable Interface

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

- Comparable
 - Promises one method: **compareTo**
 - Defines **natural ordering** for type
 - T specifies type to compare to
 - For QuakeEntry: Comparable<QuakeEntry>

Why <T>?

- QuakeEntry: compared to QuakeEntry...
- ...but Comparable used for many types
- Other types, compared to?
 - String compared to String
 - Integer to Integer
 - etc..

String Implements Comparable<String>

“apple” < “bear”

- String implements Comparable<String>
 - Can compare two strings for ordering
 - Natural ordering: alphabetical

String Implements Comparable<String>

“apple” < “bear” < “cards” < “dino”

- String implements Comparable<String>
 - Can compare two strings for ordering
 - Natural ordering: alphabetical

String Implements Comparable<String>

“apple” < “bear” < “cards” < “dino”

“What!” < “What?”

- String implements Comparable<String>
 - Can compare two strings for ordering
 - Natural ordering: alphabetical
 - Technically called “**lexicographical**”

String Implements Comparable<String>

“apple” < “bear” < “cards” < “dino”

“**W**hat!” < “**W**hat?”

- String implements Comparable<String>
 - Can compare two strings for ordering
 - Natural ordering: alphabetical
 - Technically called “**lexicographical**”

String Implements Comparable<String>

“apple” < “bear” < “cards” < “dino”

“What!” < “What?”

- String implements Comparable<String>
 - Can compare two strings for ordering
 - Natural ordering: alphabetical
 - Technically called “**lexicographical**”

String Implements Comparable<String>

“apple” < “bear” < “cards” < “dino”

“What!” < “What?” < “what!”

- String implements Comparable<String>
 - Can compare two strings for ordering
 - Natural ordering: alphabetical
 - Technically called “**lexicographical**”

String Implements Comparable<String>

“apple” < “bear” < “cards” < “dino”

“What!” < “What?” < “what!” < “what?”

- String implements Comparable<String>
 - Can compare two strings for ordering
 - Natural ordering: alphabetical
 - Technically called “**lexicographical**”

String Implements Comparable<String>

"apple".compareTo("bear") -1 ("apple" < "bear")

- Return value:
 - A negative number for "less than"

String Implements Comparable<String>

"apple".compareTo("bear") -1 ("apple" < "bear")

"bear".compareTo("bear") 0 ("bear" = "bear")

- Return value:
 - A negative number for "less than"
 - Zero for "equal"

String Implements Comparable<String>

"apple".compareTo("bear") -1 ("apple" < "bear")

"bear".compareTo("bear") 0 ("bear" = "bear")

"dino".compareTo("cards") 1 ("dino" > "cards")

- Return value:
 - A negative number for "less than"
 - Zero for "equal"
 - A positive number for "greater than"

String Implements Comparable<String>

"apple".compareTo("dino") -3 ("apple" < "dino")

"what".compareTo("What") 32 ("what" > "What")

- Less may not be -1; Greater may not be +1

String Implements Comparable<String>

"apple".compareTo("dino") -3 ("apple" < "dino")

"what".compareTo("What") 32 ("what" > "What")

- Less may not be -1; Greater may not be +1
 - Often implemented with subtraction
 - 'a' - 'd' = -3
 - 'w' - 'W' = 32