Interfaces

Overview

- Provide templates of behavior that other classes are expected to implement.
- Contain abstract method definitions and constants (static and final fields).
- Can be **public** or with **package** (default) visibility.
- An abstract class that contains only abstract methods should probably be declared as an interface.
- Example:

```
public interface Expandable {
    public abstract void expand(); // explicitly public and abstract
    void contract(); // effectively public and abstract
}
```

Using interfaces

- A class can extend one or more interfaces.
- The class must implement all of the abstract methods defined within the interfaces, unless the class is declared to be abstract in which case it can decide which of the abstract methods it will implement.
- Example:

```
public class Baloon implements Expandable {

    @Override
    public void expand() {
        // Implementation
    }

    @Override
    public void contract() {
        // Implementation
    }
}
```

Extending interfaces

- Similar to classes, interfaces can be organized into a hierarchy using inheritance.
- Multiple inheritance is allowed among interfaces.
- There is no root interface like with the Object class in the class hierarchy.
- Example:

```
interface PreciselyTrackable extends Trackable {
    // ...
}
```