起来像继承。

```
// interfaces/ImplementingAnInterface.java

interface Concept { // 包访问
  void idea1();
  void idea2();
}

class Implementation implements Concept {
  @Override public void idea1() {
    System.out.println("idea1");
  }
  @Override public void idea2() {
    System.out.println("idea2");
  }
}
```

你可以选择将接口中的方法显式声明为 public，但即使不显式声明，它们也是 public 的。所以当实现一个接口时，来自接口的方法必须被定义为 public。否则，它们将默认为包访问权限，导致在继承期间降低了方法的可访问性，而这是 Java 编译器不允许的。

与 abstract 类一样，在实现接口时也可以使用 @Override。它可以防止意外重载，还可以提示读者实现的是哪些接口方法。

10.2.1 默认方法

Java 8为default关键字找到了一个额外的用途(以前只在 switch 语句和注解中使用)。当在接口中使用时，default 会允许方法创建一个方法体，实现了该接口的类可以在不定义方法的情况下直接替换方法体。

默认方法比抽象类上的方法更受限制，但非常有用，我们将在第 14 章中看到这一点。

让我们通过下面这个接口看看它是如何工作的：

```
// interfaces/AnInterface.java

interface AnInterface {
  void firstMethod();
  void secondMethod();
}
```

可以用通常的方式来实现这个接口：

```
// interfaces/AnImplementation.java

public class AnImplementation implements AnInterface {
```