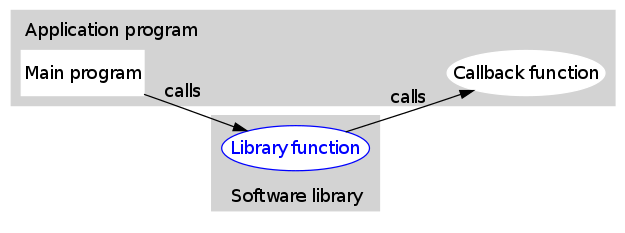
**回调函数 Callback**

**定义**

在计算机程序设计中，回调函数，或简称回调（Callback 即call then back 被主函数调用运算后会返回主函数），是指通过函数参数传递到其它代码的，某一块可执行代码的引用。这一设计允许了底层代码调用在高层定义的子程序。

一个回调函数是作为另一个函数的参数的一个可执行的代码段 这个代码段在方便的时间来执行。通俗来说：把函数f2当做一个参数传给函数f1 并且在f1里适当的时间执行f2。



可以看到，回调函数通常和应用处于同一抽象层（因为传入什么样的回调函数是在应用级别决定的）。而回调就成了一个高层调用底层，底层再回过头来调用高层的过程。使用回调函数实际上就是在调用某个函数（通常是API函数）时，将自己的一个函数（这个函数为回调函数）的地址作为参数传递给那个函数。

**使用方法**

软件模块之间总是存在着一定的接口，从调用方式上，可以把他们分为三类：同步调用、回调和异步调用。同步调用是一种阻塞式调用，调用方要等待对方执行完毕才返回，它是一种单向调用；回调是一种双向调用模式，也就是说，被调用方在接口被调用时也会调用对方的接口；异步调用是一种类似消息或事件的机制，不过它的调用方向刚好相反，接口的服务在收到某种讯息或发生某种事件时，会主动通知客户方（即调用客户方的接口）。回调和异步调用的关系非常紧密，通常我们使用回调来实现异步消息的注册，通过异步调用来实现消息的通知。同步调用是三者当中最简单的，而回调又常常是异步调用的基础。

而那个函数在需要的时候，利用传递的地址调用回调函数，这时你可以利用这个机会在回调函数中处理消息或完成一定的操作。至于如何定义回调函数，跟具体使用的API函数有关，一般在帮助中有说明回调函数的参数和返回值等。C++中一般要求在回调函数前加CALLBACK，这主要是说明该函数的调用方式。

回调的用途十分广泛。例如，假设有一个函数，其功能为读取配置文件并由文件内容设置对应的选项。若这些选项由散列值所标记，则让这个函数接受一个回调会使得程序设计更加灵活：函数的调用者可以使用所希望的散列算法，该算法由一个将选项名转变为散列值的回调函数实现；因此，回调允许函数调用者在运行时调整原始函数的行为。

对于一般的结构化语言，可以通过回调函数来实现回调。回调函数也是一个函数或过程，不过它是一个由调用方自己实现，供被调用方使用的特殊函数。 在面向对象的语言中，回调则是通过接口或抽象类来实现的，我们把实现这种接口的类成为回调类，回调类的对象成为回调对象。对于C++等兼容了过程特性的对象语言，不仅提供了回调对象、回调方法等特性，也能兼容过程语言的回调函数机制。

**例子**

本例的场景是：写作业。交代三个方面：谁，什么动作（写），写什么。下面先从（有个学生，写，作业）开始。

# 1. 有个学生

Student student = new Student();

# 2. 该学生有写作业这个动作需要执行

student.doHomeWork(someHomeWork);

# 3. 注意到这个写作业这个动作是需要得到入参“作业”的后才能进行的。

所以给这个学生new了个简单的题目做。

String aHomeWork = "1+1=?";

student.doHomeWork(aHomeWork);

至此，完成写作业的动作。

完整代码

public class Student {

public void doHomeWork(String homeWork) {

System.out.println("作业本");

if("1+1=?".equals(homeWork)) {

System.out.println("作业："+homeWork+" 答案："+"2");

} else { System.out.println("作业："+homeWork+" 答案："+"不知道"); }

}

public static void main(String[] args) {

Student student = new Student();

String aHomeWork = "1+1=?";

student.doHomeWork(aHomeWork);

}

}

程序执行

作业本

作业：1+1=? 答案：2

我们一定要把焦点聚焦到”写作业“这个需求上面。该学生写作业的方法是现成的，但是需要有作业作为入参，怎么获取作业才是完成动作的关键。让室友帮忙解答上面的例子中该同学自己调用自己的方法，把收到的homework直接写了。但是现实可能会出现各种各样的问题导致该同学不能自己来做。所以他拜托了他的好室友来帮忙写下。

#因为室友帮忙写，所以在doHomeWork动作里面，

就不需要有逻辑判断的代码，因为舍友会直接把答案写进来。改成：

student.doHomeWork(aHomeWork, theAnswer);

#上句中做作业的动作支持传入“作业”和“答案”，有了这两个，就说明能做好作业了。

#其中aHomeWork作业是已知的，但是theAnswer这个答案却是室友提供的。

#室友怎么才能提供答案呢，最简单是，室友这个对象直接提供一个传入作业，

传出答案的方法，这样该同学就可以直接调用了。

RoomMate roomMate = new RoomMate();

String theAnswer = roomMate.getAnswer(aHomeWork);

student.doHomeWork(aHomeWork, theAnswer);

完整代码

public class Student {

public void doHomeWork(String homeWork, String answer) {

System.out.println("作业本");

if(answer != null) {

System.out.println("作业："+homeWork+" 答案："+ answer);

} else { System.out.println("作业："+homeWork+" 答案："+ "(空白)"); }

}

public static void main(String[] args) {

Student student = new Student();

String aHomeWork = "1+1=?";

RoomMate roomMate = new RoomMate();

String theAnswer = roomMate.getAnswer(aHomeWork);

student.doHomeWork(aHomeWork, theAnswer);

}

}

public class RoomMate {

public String getAnswer(String homework) {

if("1+1=?".equals(homework)) { return "2"; }

else { return null; }

}

}

程序执行

作业本

作业：1+1=? 答案：2

回顾下这两行代码

#室友写好作业

String theAnswer = roomMate.getAnswer(aHomeWork);

#该同学直接抄答案，完成作业

student.doHomeWork(aHomeWork, theAnswer);

该同学想能不能直接调用做作业方法把答案写好，把作业做完。回顾下写作业的全过程

#待解决的作业

String aHomeWork = "1+1=?";

#室友写出答案

String theAnswer = roomMate.getAnswer(aHomeWork);

#该同学调用，自己把答案写到作业本。（也即是这个步骤不给调用了）

student.doHomeWork(aHomeWork, theAnswer);

#做作业必须得调用这个方法，而根据需求这个方法必须由室友去调用。那很显然，该室友得保持一个该同学的引用，才能正常调用。

#室友说，那你在调用getAnswer方法的时候，除了传入作业，还需要把自己的引用放里面。这样我做完了，直接调用你的做作业方法就行了。

roomMate.getAnswer(aHomeWork,student);

完整代码

public class Student {

public void doHomeWork(String homeWork, String answer) {

System.out.println("作业本");

if(answer != null) {

System.out.println("作业："+homeWork+" 答案："+ answer);

} else {

System.out.println("作业："+homeWork+" 答案："+ "(空白)");

}

}

public static void main(String[] args) {

Student student = new Student();

String aHomeWork = "1+1=?";

RoomMate roomMate = new RoomMate();

roomMate.getAnswer(aHomeWork,student);

}

}

public class RoomMate {

public void getAnswer(String homework, Student student) {

if("1+1=?".equals(homework)) {

student.doHomeWork(homework, "2");

} else {

student.doHomeWork(homework, "(空白)");

}

}

}

执行程序

作业本

作业：1+1=? 答案：2

回调方法在上述“让室友直接把作业写了”的例子中，其实已经体现了回调的意思。场景的核心在于这位学生要把作业给做了。

简单来说，这位学生告诉室友要做什么作业，并把自己的引用也给了室友。该室友得到作业，做完后直接引用该学生并调用其做作业的方法，完成代写作业的任务。

具体来讲，该学生做作业的方法有两个入参，一个是作业题目（已知），一个是作业答案（未知）。室友为了帮助他写作业提供了一个方法，该方法有两个入参，一个是作业题目，一个是该学生的引用（解出答案得知道往哪写）。

程序执行时，该学生只要调用室友的代写作业方法就行了。一旦室友得到答案，因为有该学生的引用，所以直接找到对应方法，帮助其完成作业。

可以知道，学生调用室友的替写作业方法，注册了题目和自己的引用。室友的替写作业方法被调用，则会根据题目完成作业后，再回调该同学写作业方法，完成作业。

抽象来说，类A调用类B的方法b（传入相关信息），类B的方法在执行完后，会将结果写到（再回调）类A的方法a，完成动作。（方法a就是回调方法）