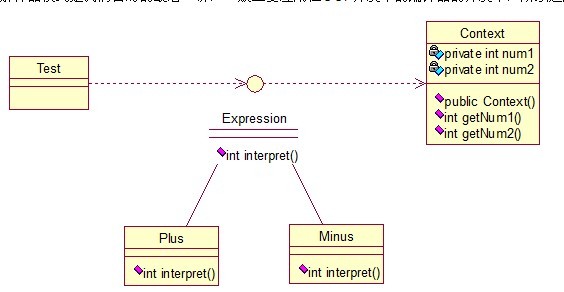
解释器模式是我们暂时的最后一讲，一般主要应用在OOP开发中的编译器的开发中，所以适用面比较窄。



Context类是一个上下文环境类，Plus和Minus分别是用来计算的实现，代码如下：

**[java]** [view plaincopy](http://blog.csdn.net/zhangerqing/article/details/8245537)

1. **public** **interface** Expression {
2. **public** **int** interpret(Context context);
3. }

**[java]** [view plaincopy](http://blog.csdn.net/zhangerqing/article/details/8245537)

1. **public** **class** Plus **implements** Expression {
3. @Override
4. **public** **int** interpret(Context context) {
5. **return** context.getNum1()+context.getNum2();
6. }
7. }

**[java]** [view plaincopy](http://blog.csdn.net/zhangerqing/article/details/8245537)

1. **public** **class** Minus **implements** Expression {
3. @Override
4. **public** **int** interpret(Context context) {
5. **return** context.getNum1()-context.getNum2();
6. }
7. }

**[java]** [view plaincopy](http://blog.csdn.net/zhangerqing/article/details/8245537)

1. **public** **class** Context {
3. **private** **int** num1;
4. **private** **int** num2;
6. **public** Context(**int** num1, **int** num2) {
7. **this**.num1 = num1;
8. **this**.num2 = num2;
9. }
11. **public** **int** getNum1() {
12. **return** num1;
13. }
14. **public** **void** setNum1(**int** num1) {
15. **this**.num1 = num1;
16. }
17. **public** **int** getNum2() {
18. **return** num2;
19. }
20. **public** **void** setNum2(**int** num2) {
21. **this**.num2 = num2;
22. }

25. }

**[java]** [view plaincopy](http://blog.csdn.net/zhangerqing/article/details/8245537)

1. **public** **class** Test {
3. **public** **static** **void** main(String[] args) {
5. // 计算9+2-8的值
6. **int** result = **new** Minus().interpret((**new** Context(**new** Plus()
7. .interpret(**new** Context(9, 2)), 8)));
8. System.out.println(result);
9. }
10. }

最后输出正确的结果：3。

