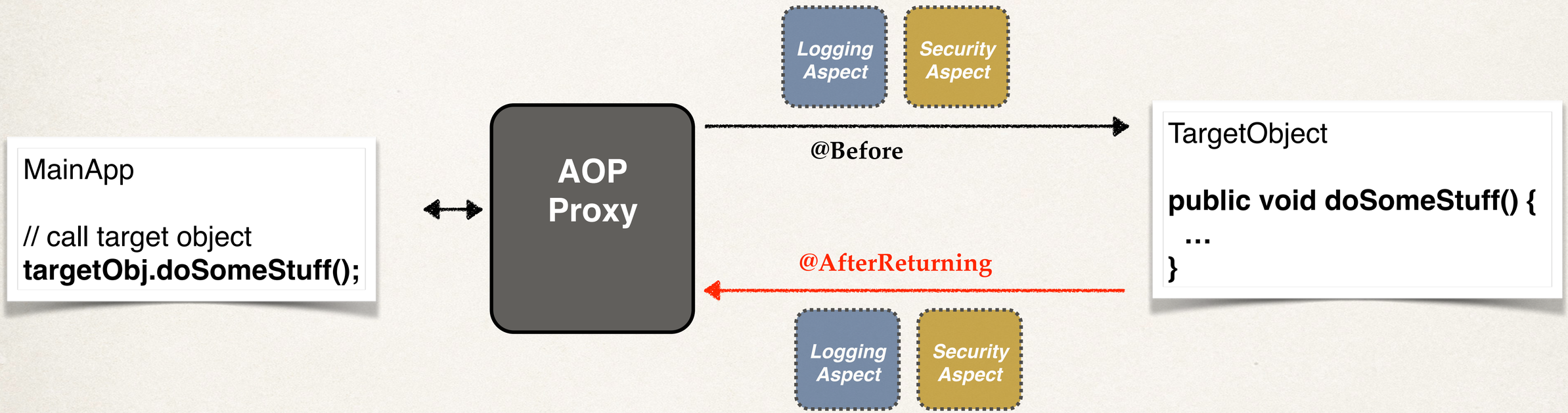
Aspect-Oriented Programming (AOP)
@AfterReturning Advice

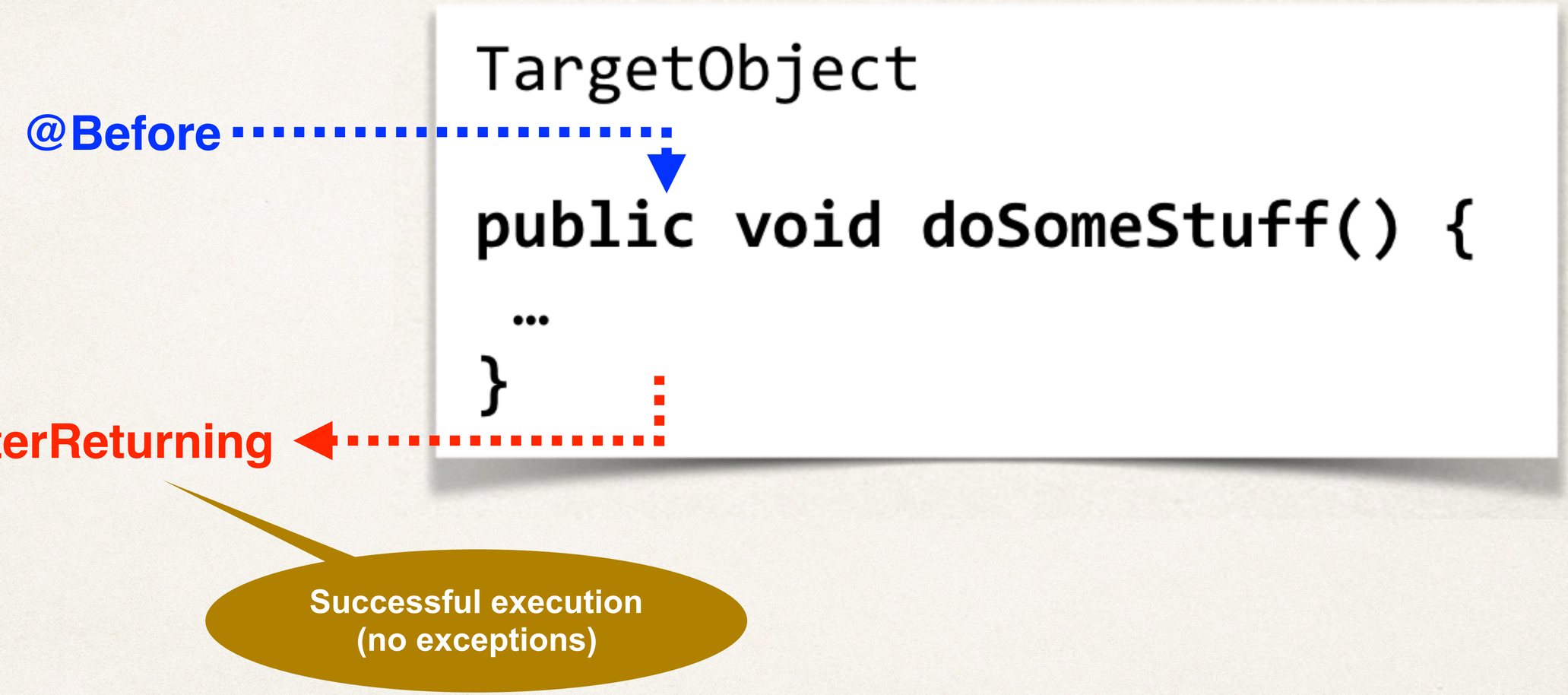
Advice Types

- **Before advice**: run before the method
- **After returning advice**: run after the method (success execution)
- **After throwing advice**: run after method (if exception thrown)
- **After finally advice**: run after the method (finally)
- **Around advice**: run before and after method

@AfterReturning Advice - Interaction



Advice - Interaction



@AfterReturning Advice - Use Cases

- **Most common**
  - logging, security, transactions
- **Audit logging**
  - who, what, when, where
- **Post-processing Data**
  - Post process the data before returning to caller
  - Format the data or enrich the data (really cool but be careful)

Example

- Create an advice that will run after a method call (success execution)

@AfterReturning Advice

- This advice will run after the method call (success execution)

```
@AfterReturning("execution(* com.luv2code.aopdemo.dao.AccountDAO.findAccounts(..))")
public void afterReturningFindAccountsAdvice() {

   System.out.println("Executing @AfterReturning advice");

}
```

Access the Return Value

- Need to access the return value of method called

Access the Return Value

```
@AfterReturning(
    pointcut="execution(* com.luv2code.aopdemo.dao.AccountDAO.findAccounts(..))",
    returning="result")
public void afterReturningFindAccountsAdvice() {




}
```

Access the Return Value

```
@AfterReturning(
    pointcut="execution(* com.luv2code.aopdemo.dao.AccountDAO.findAccounts(..))",
    returning="result")
public void afterReturningFindAccountsAdvice(
        JoinPoint theJoinPoint, List<Account> result) {




}
```

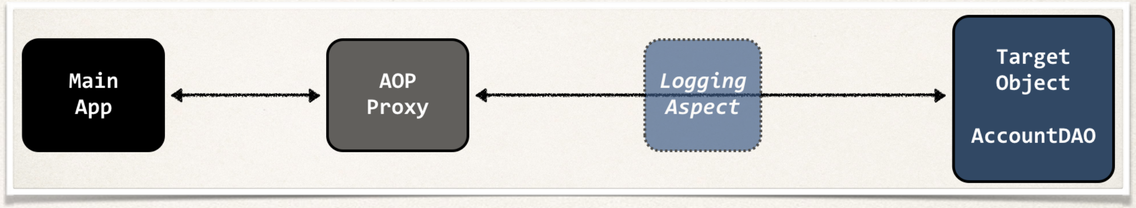
Access the Return Value

```
@AfterReturning(
    pointcut="execution(* com.luv2code.aopdemo.dao.AccountDAO.findAccounts(..))",
    returning="result")
public void afterReturningFindAccountsAdvice(
        JoinPoint theJoinPoint, List<Account> result) {

  // print out the results of the method call
  System.out.println("\n=====>>> result is: " + result);

}
```

Best Practices: Aspect and Advices



- Keep the code small
- Keep the code fast
- Do not perform any expensive / slow operations
- Get in and out as QUICKLY as possible