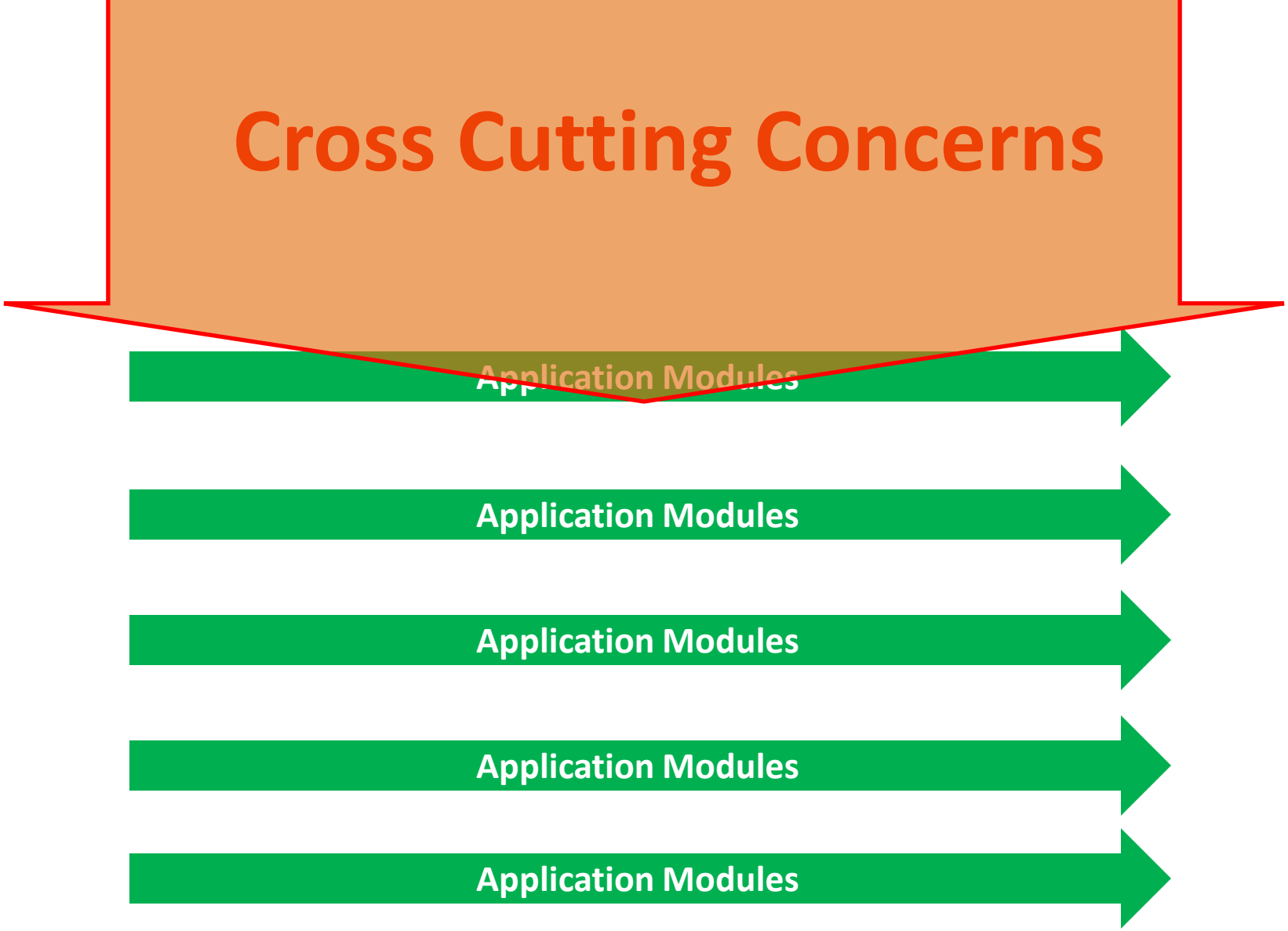
Spring AOP
Aspect Oriented Programming

By

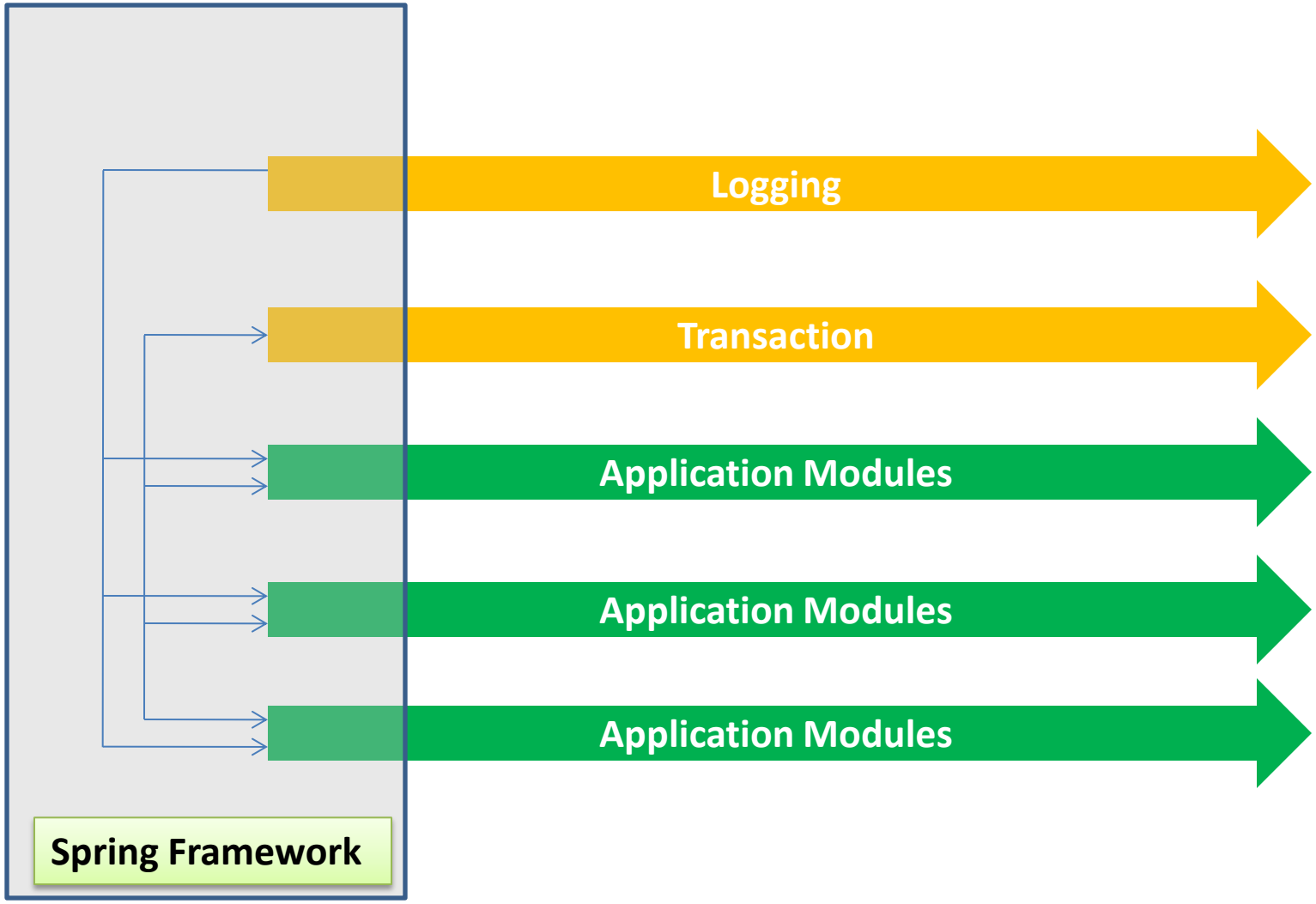
Srinivas Reddy.S

Cross Cutting Concerns

```
class Bank{
    private int balance;
   public void withdraw(int amount){
    bankLogger.info("Withdraw –"+amount);
    tx.begin();
    balance = this.balance-amount;
    accountDao.saveBalance(balance);
    tx.commit();
    }
}
```



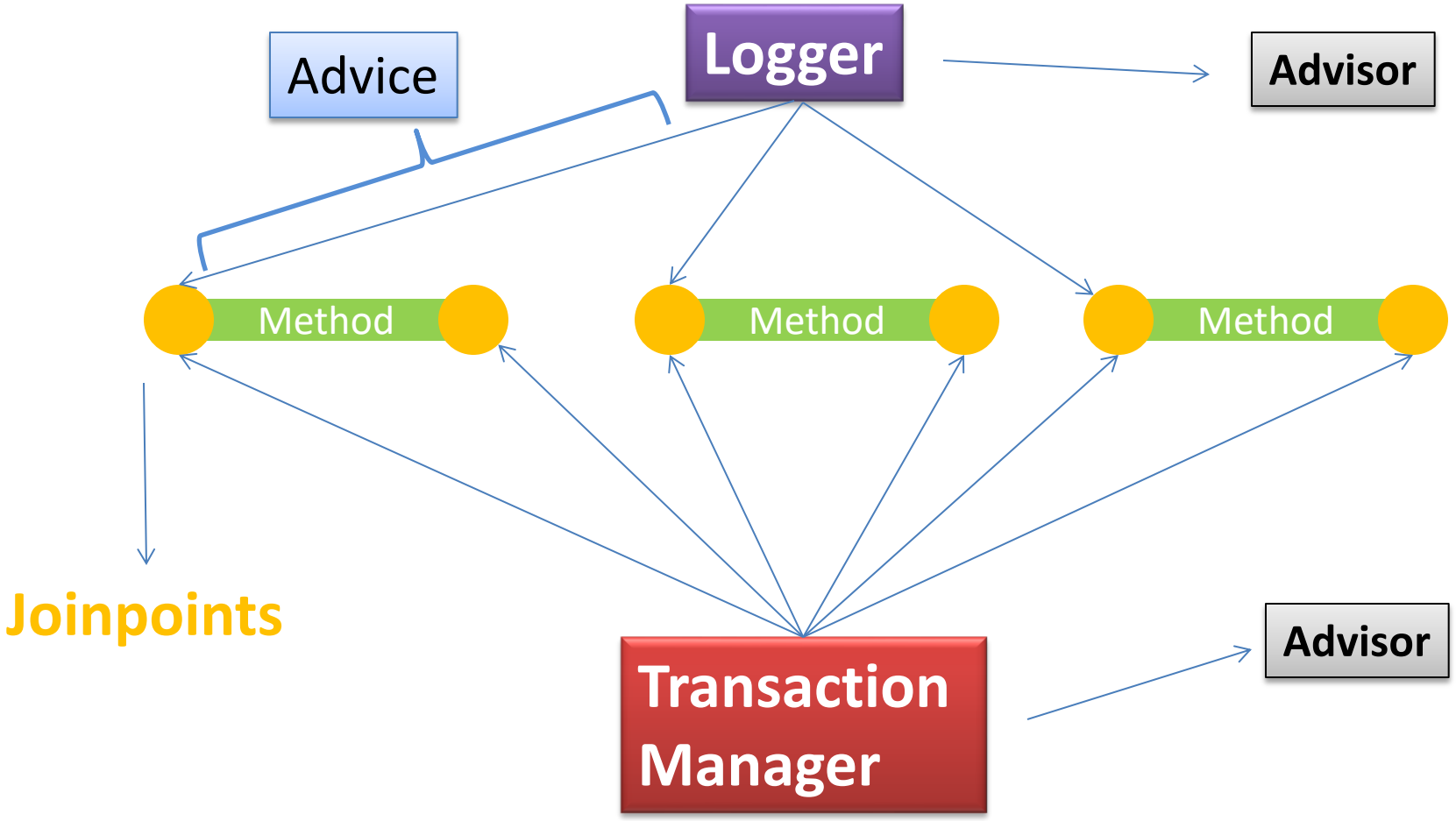
Cross Cutting Concerns



AOP – Definitions.

- Aspect
- Joinpoint
- Advice
- Pointcut
- Introduction
- Target Object
- AOP Proxy
- Weaving

AOP – Definitions.

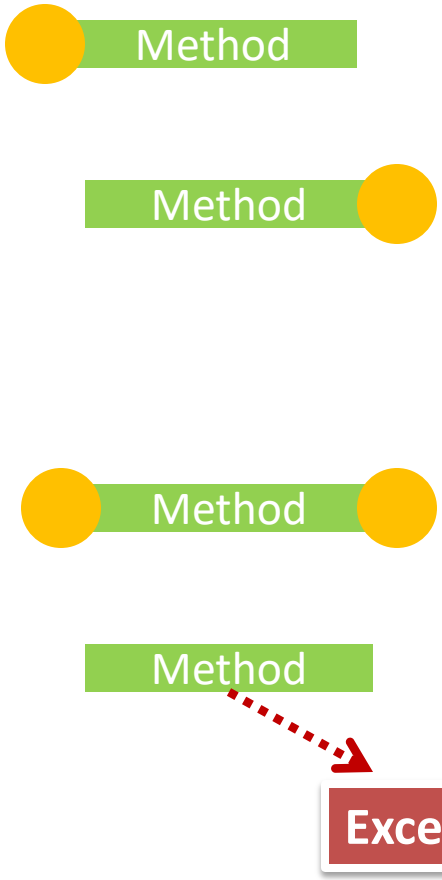


AOP - Definitions

- **Advice** defines what needs to be applied and when.
- **Jointpoint** is where the advice is applied.
- **Pointcut** is the combination of different joinpoints where the advice needs to be applied.
- **Aspect** is applying the Advice at the pointcuts.

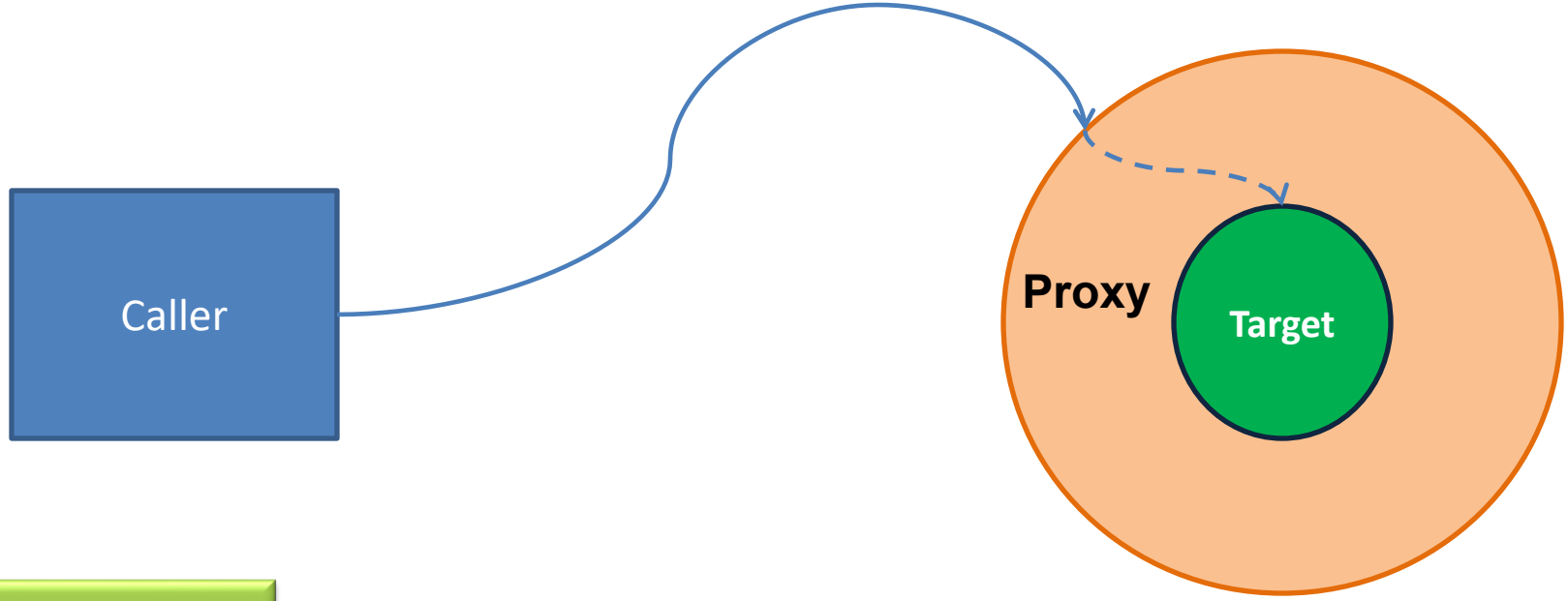
Advice Types

- Before Advice
- After returning Advice
- Around Advice
- Throws Advice



AOP - Weaving

- Compile time
- Class Load Time
- Runtime – Springs way



Pointcut and Advisor

POINTCUT CLASSES:

- Perl5RegexpMethodPointcut
- JdkRegexpMethodPointcut

Pointcut and Advisor in one class:

- RegexpMethodPointcutAdvisor

Example

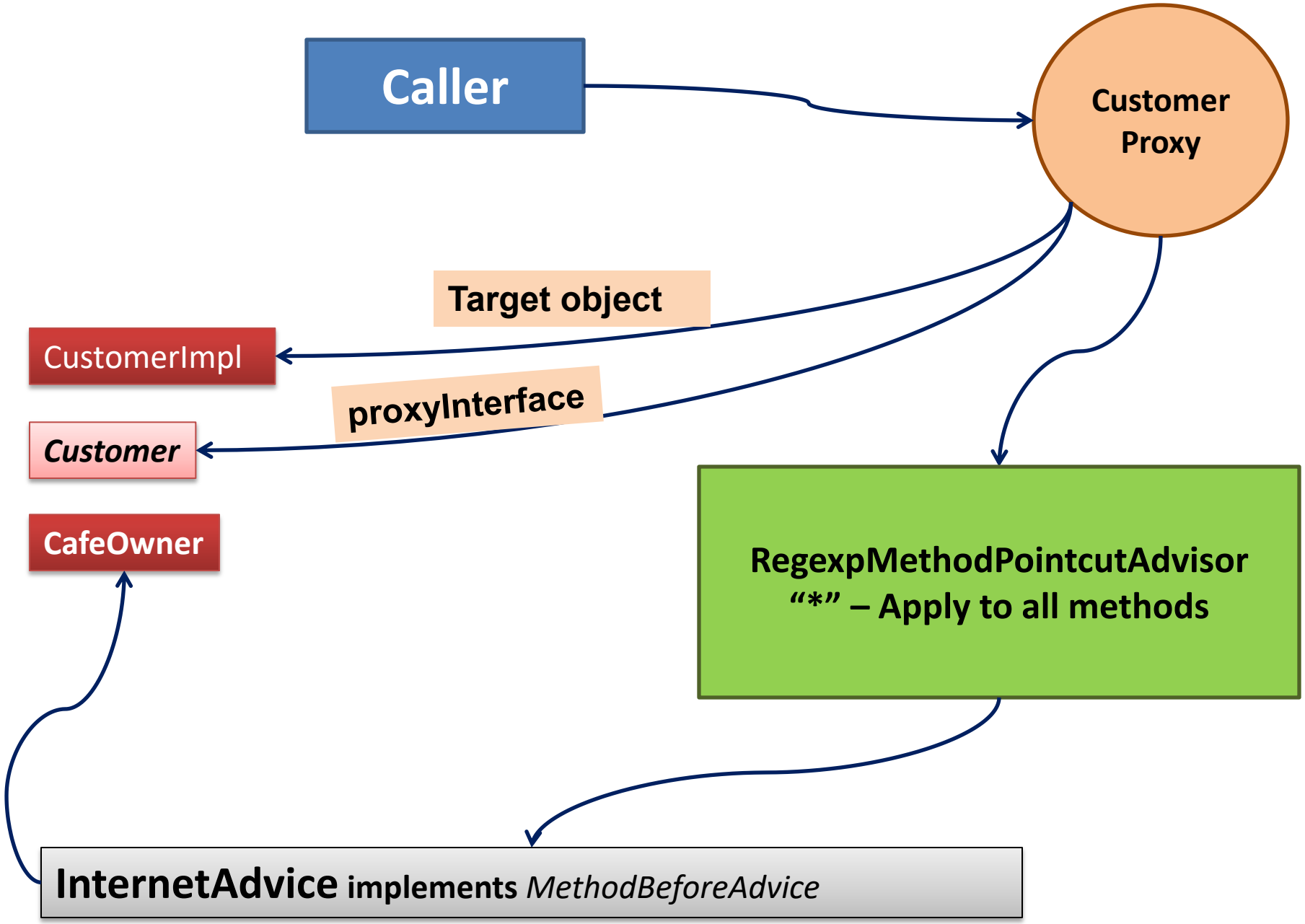
```
public class CustomerImpl
    implements Customer{
public void browse(){
System.out.println("Browsing
    the internet");
}
}
```

```
class CafeOwner{
    void LogInTime(){
    System.out.println("Log  In
    time and name of the
    customer");
    }
    void LogOutTime(){
    System.out.println("Log Out
    Time");
    }
    void  issueUsageBill(){
    System.out.println("Calculate
    and bill the customer");
    }
}
```

Before Advice -MethodBeforeAdvice

```
class InternetAdvisor implements
    MethodBeforeAdvice{
private CafeOwner cafeOwner;

public void before(Method arg0, Object[] arg1, Object arg2)
throws Throwable {
this.getCafeOwner().LogInTime();
}
}
```


Caller
Customer Proxy
Target object
CustomerImpl
proxyInterface
Customer
CafeOwner
RegexpMethodPointcutAdvisor
"*" – Apply to all methods
InternetAdvice implements MethodBeforeAdvice

Configuration

Step 1: Configure the Beans

- <bean id ="customerImpl" class ="CustomerImpl"/>
- <bean id = "cafeOwner" class ="CafeOwner"/>
- <bean id ="internetAdvice" class ="InternetAdvice">
  - <property name ="cafeOwner" ref ="cafeOwner"/>
- </bean>

Configuration

Step 2: Configure the POINTCUT ADVISOR

- <bean id ="*cafeOwnerBeforeAndAfterAdvice*"
  class ="org.springframework.aop.support.RegexpMethodPointcutAdvisor">
  <property name ="*advice*">
  <ref local ="*internetAdvice*"/>
  </property>
  <property name ="*pattern*">
  <value>.*</value>
  </property>
  </bean>

Configuration

Step 3: Configure the ProxyFactoryBean

```
<bean id ="customerProxy" class
    ="org.springframework.aop.framework.ProxyFactoryBean">
    <property name ="target">
    <ref local ="customerImpl"/>
    </property>
    <property name ="proxyInterfaces">
    <value>Customer</value>
    </property>
    <property name ="interceptorNames">
    <list>
    <value>cafeOwnerBeforeAndAfterAdvice</value>
    </list>
    </property>
```

Remember

- Spring Does not support AOP for
  - Methods marked as final.
  - Fields

WWW.JAVA9S.COM