Spring AOP ve Type Introduction Kabiliyeti

Spring AOP ve Type Introduction

- AspectJ terminolojisinde **inter-type decleration** olarak bilinir
- Bu tür yapılara **mixin** adı da verilmektedir
- Belirli tipteki nesnelerin runtime'da **yeni bir interface'e daha sahip olmaları** sağlanır
- Aspect tanımı içerisinde **@DeclareParents** annotasyonu ile gerçekleştirilir

Spring AOP ve Type Introduction

AspectJ type pattern'ı ile eşleşen nesneler UsageTracked arayüzüne de sahip olurlar

```
@Aspect
public class UsageTrackedAspect {

@DeclareParents(value="com.javaegitimleri.service.*+",
                defaultImpl=DefaultUsageTracked.class)
  private UsageTracked mixin;

}
```

UsageTracked arayüzüne sahip olacak bean'lere bu arayüz üzerinden gerçekleştirecekleri davranış da **defaultImpl** attribute'undaki sınıf ile sağlanır

Spring AOP ve Type Introduction

```
@Aspect
public class UsageTrackingAspect {


   @Before("execution(* com.javaegitimleri.service.*.*(..)) && this(usageTracked)")
   public void recordUsage(UsageTracked usageTracked) {
      usageTracked.incrementUsageCount();
   }

}
```

Herhangi başka bir aspect içerisinde UsageTracked arayüzüne sahip proxy nesneler yakalanıp UsageTracked arayüzü üzerinden işlem yapılabilir

Spring AOP ve Type Introduction

```
UsageTracked usageTracked = (UsageTracked)
                    appContext.getBean("petClinicService");

usageTracked.getUsageCount();
```

Çalışma zamanında ilave arayüz eklenen bean'lere erişilerek bu bean'ler yeni arayüze downcast edilerek de kullanılabilir

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com