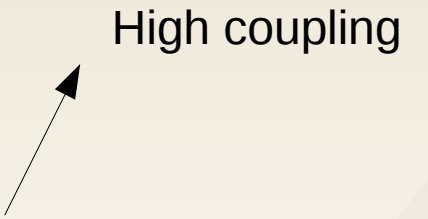
Observer Örüntüsü

High/Low Coupling



"**Sıkı, iç içe geçmiş ve doğrudan (thightly coupled)**" nesne ilişkilerinden uzak dur!

Nesneler arasında ilişkilerin "**gevşek ve dolaylı (loosely coupled)**" olmasına çalış...

Hollywood Prensibi

Siz bizi aramayın, biz sizi arayacağız!
(Don't call us, we will call you!)

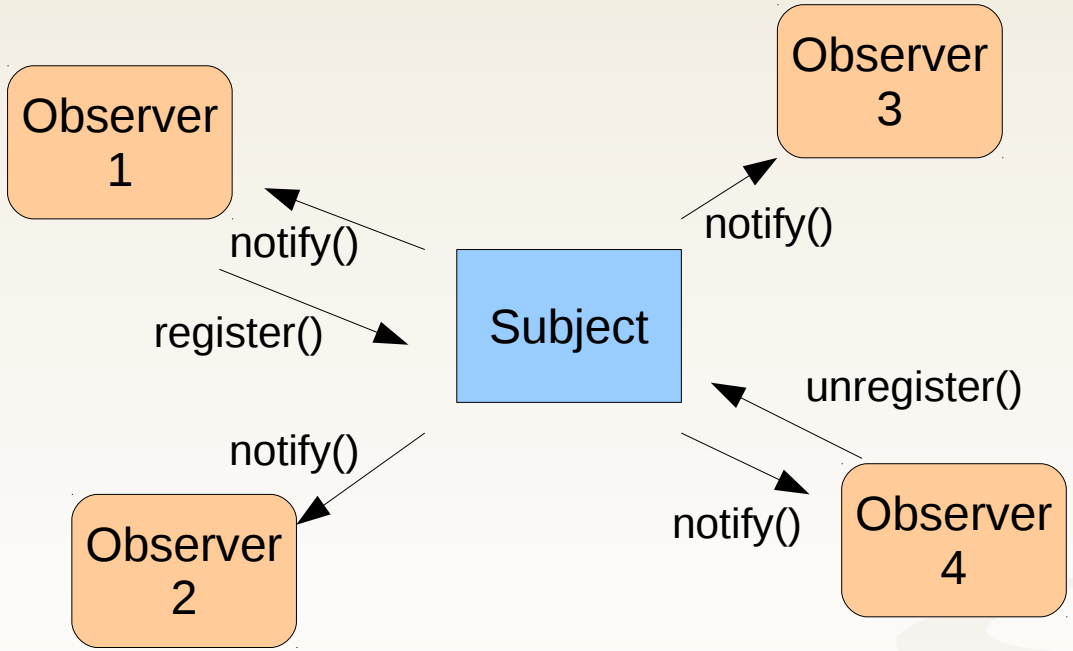
Observer

- Bir grup nesnenin başka **bir nesne(subject) üzerindeki değişiklikleri bilmeleri** ve buna göre **işlem yapmaları** gerekebilir
- Bu nesnelerin **subject'i periyodik aralıklarla kontrol etme**leri bir çözüm olabilir
- Ancak bu **gereksiz bir iletişim trafiği** ortaya çıkaracaktır
- Bunun yerine ilgili nesnelerin **subject'deki değişikliklerden haberdar edilmeleri** daha uygun olacaktır

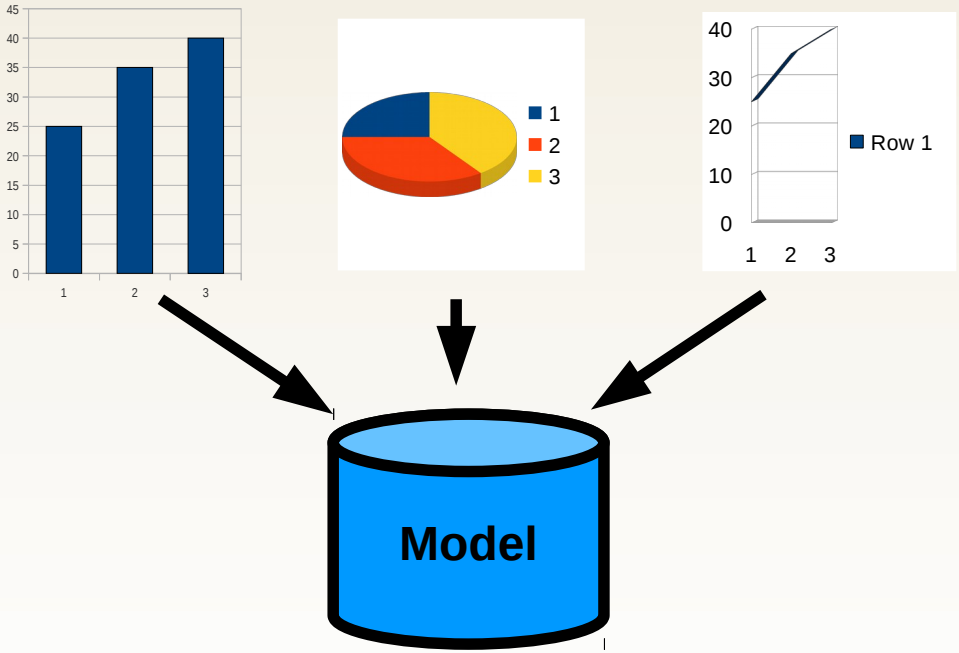
Observer

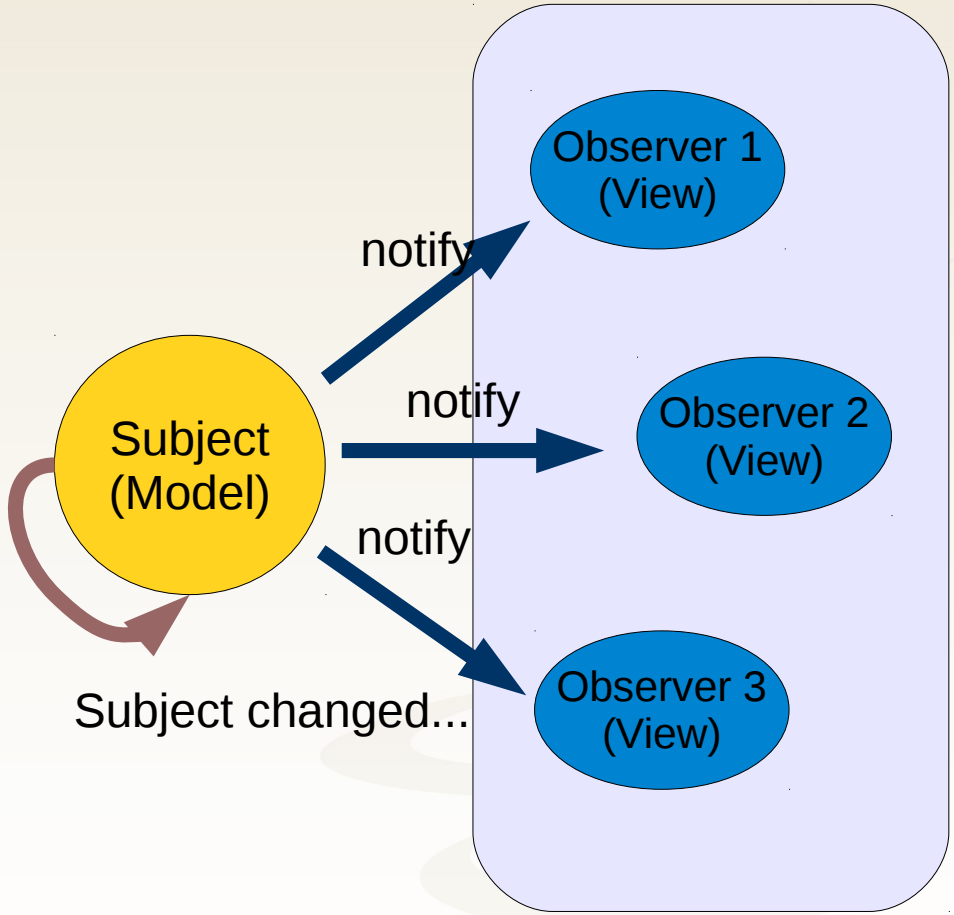
- Bunun için Observer nesneler kendilerini **subject'e tanıtır**lar
- Subject üzerinde herhangi bir değişiklik olduğunda tanımlı Observer nesneleri **değişiklikten haberdar** edilirler
- Observer nesneler **sadece bilgilendirme (notification) geldiğinde** devreye girerler ve gerekli operasyonları gerçekleştirirler
- Observer'lar subject ile ilgili değişikliklerden haberdar olmak istemedikleri vakit **kendilerini subject'den çıkarmaları** yeterlidir

Observer

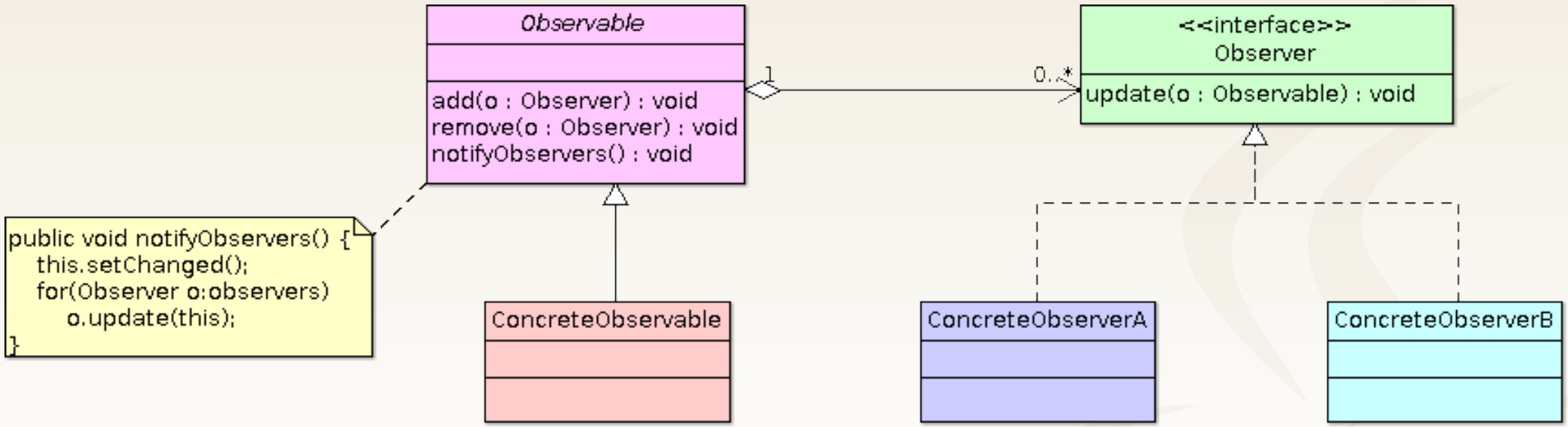

Observer 1
Observer 3
notify()
notify()
register()
Subject
unregister()
notify()
notify()
Observer 2
Observer 4

Observer

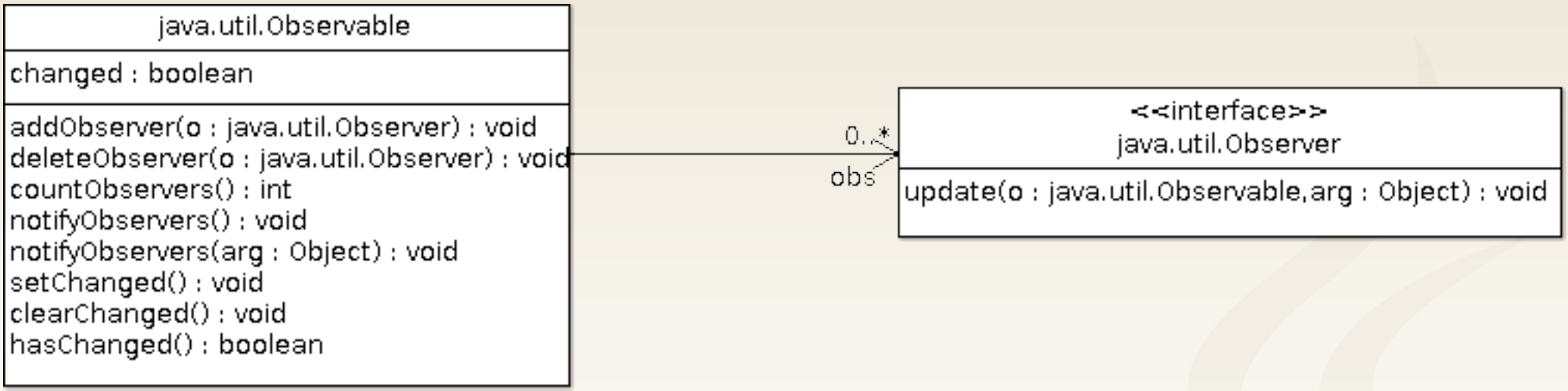

45
40
35
30
25
20
15
10
5
0
1
2
3
1
2
3
40
30
20
10
0
1 2 3
Row 1
Model


Observer 1
(View)
notify
Subject
(Model)
notify
Observer 2
(View)
notify
Observer 3
(View)
Subject changed...

Observer Sınıf Diagramı

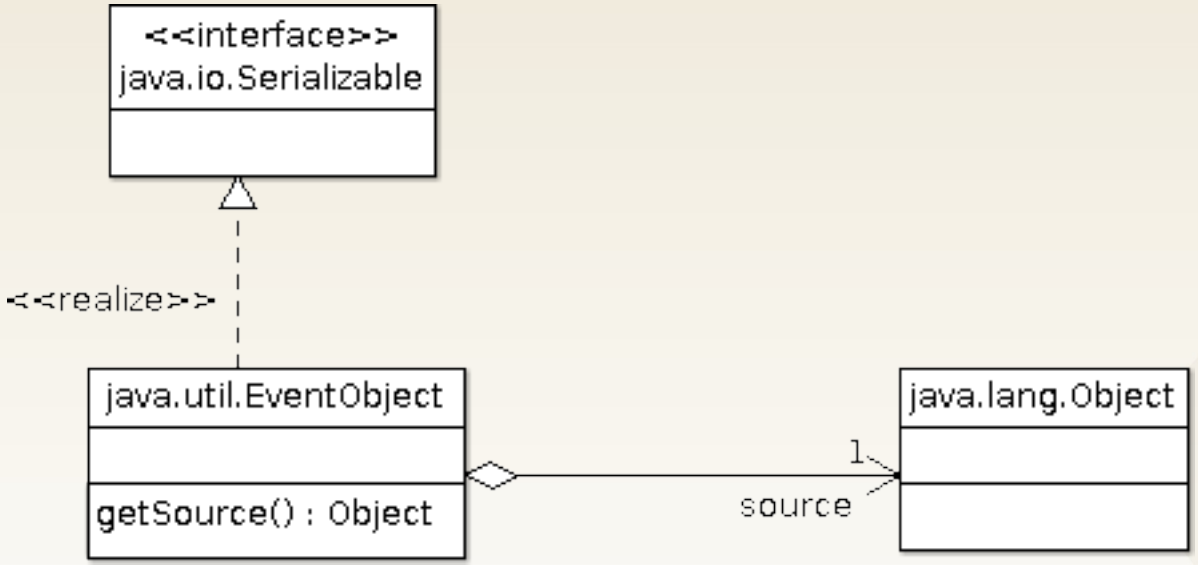


Java ve Observer



JDK içerisinde Observer örüntüsü için sınıflar barındırmaktadır. Uygulama içerisinde Subject kategorisindeki sınıflar Observable sınıfından türeyerek Observer nesnelerini ekleme/çıkarma, kayıtlı Observer nesnelerini kendileri ile ilgili değişikliklerden haberdar etme gibi kabiliyetlere sahip olurlar.

Java ve Observer

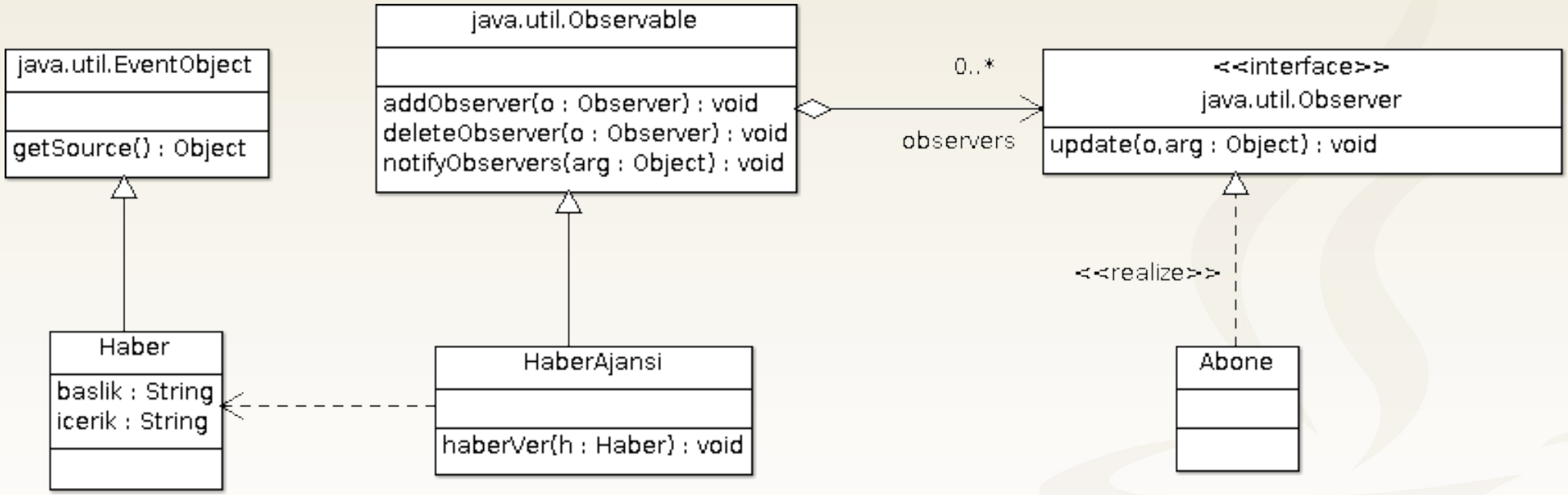


Observable nesneler Observer'ları değişikliklerden haberdar etmek için genellikle EventObject'den türeyen nesneleri argüman olarak kullanırlar. EventObject, JDK'daki bütün event state sınıflarının türediği ata sınıftır. uygulamaya özel event state sınıflarının da EventObject sınıfından türetilmeleri önerilir.

LAB ÇALIŞMASI: Observer

- Bir magazin haber ajansı magazin dünyasındaki en son dedikoduları abonelerine iletmesini sağlayacak bir uygulama istemektedir
- Kullanıcılar magazin haberlerine ulaşmak için ajansa abone olmalıdırlar
- Yeni bir haber geldiğinde ajans bu haberi bütün abonelerine iletecektir
- Aboneler haberleri almak istemedikleri herhangi bir vakit ajans aboneliğini bırakabilirler

LAB ÇALIŞMASI: Observer



Observer Örüntüsünün Sonuçları

- Subject **hangi tür nesneler**in kendisi hakkındaki değişikliklerle ilgilendiğini bilmez
- Böylece hem subject'i hem de observer'ları farklı bağlamlarda **birbirlerinden bağımsız** kullanmak mümkün olur
- Bazı durumlarda Observer'lar **subject üzerinde neyin tam olarak değiştiğ**ini tespit etmekte zorlanabilirler
- Olayın meydana geldiği yer observable, bu olaya göre işlemin yapıldığı yer Observer olduğu için uygulamanın **akışının takibi, izlenebilirliği zorlaşabilir**

İletişim

www.harezmi.com.tr

www.java-egitimleri.com

info@harezmi.com.tr

info@java-egitimleri.com

@HarezmiBilisim

@JavaEgitimleri