JAVA Eğitimleri

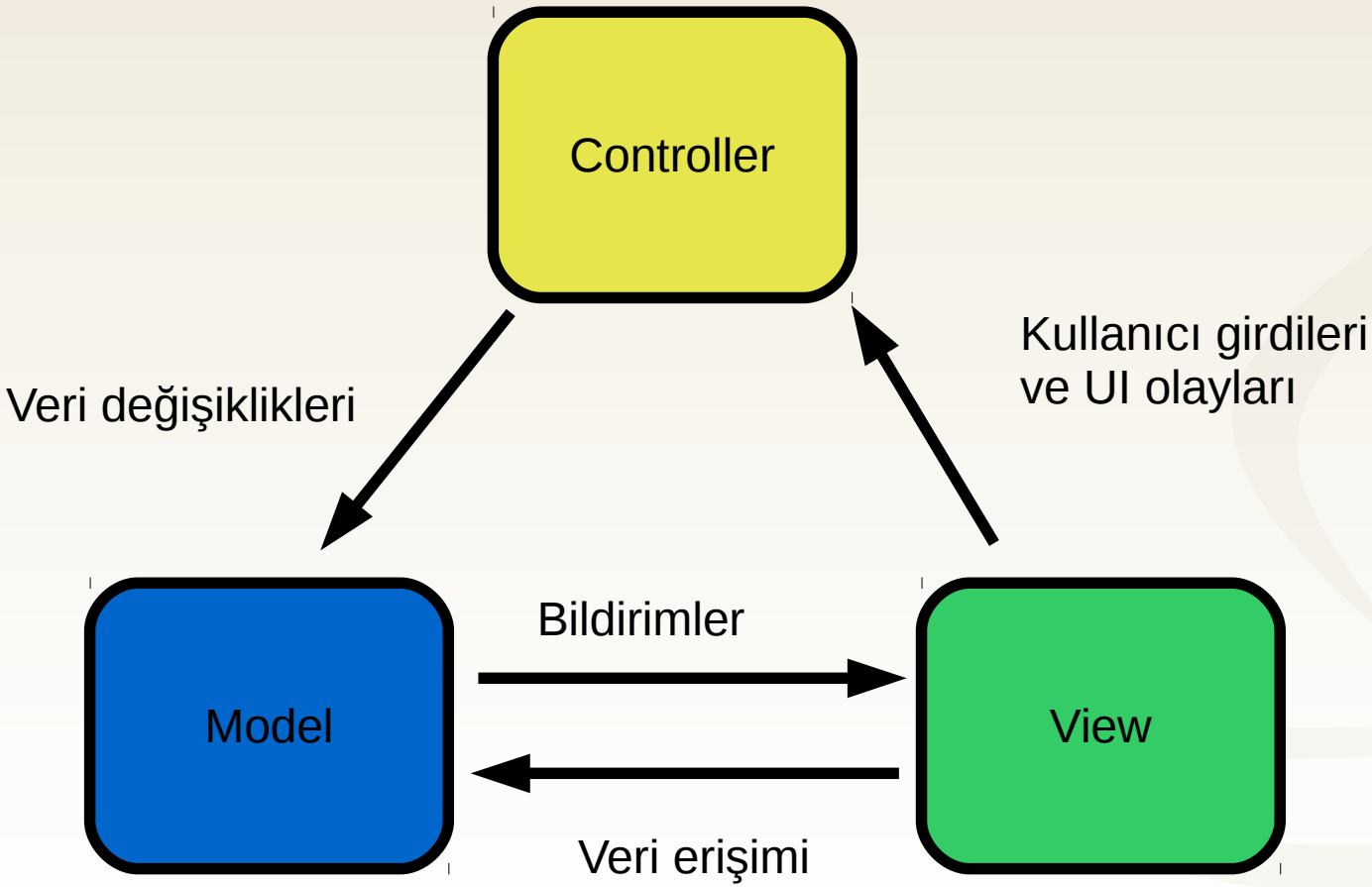
Model View Controller Mimarisel Örüntüsü

Model View Controller Nedir?

JAVA Eğitimleri

- 70'li yılların sonunda ortaya çıkmıştır
- Norveçli bilim adamı ve mühendis Tyrgve Reenskaug'un Norveç'de başlayıp, Amerika'daki Xerox lablarını ziyareti sırasında gelişmiş **mimarisel bir örüntü**dür
- Kısaca **MVC** olarak da bilinir
- MVC, uygulamanın bileşenlerini **üç ana yapı**ya ayırarak ele alır

MVC Bileşenleri Arasındaki Etkileşim

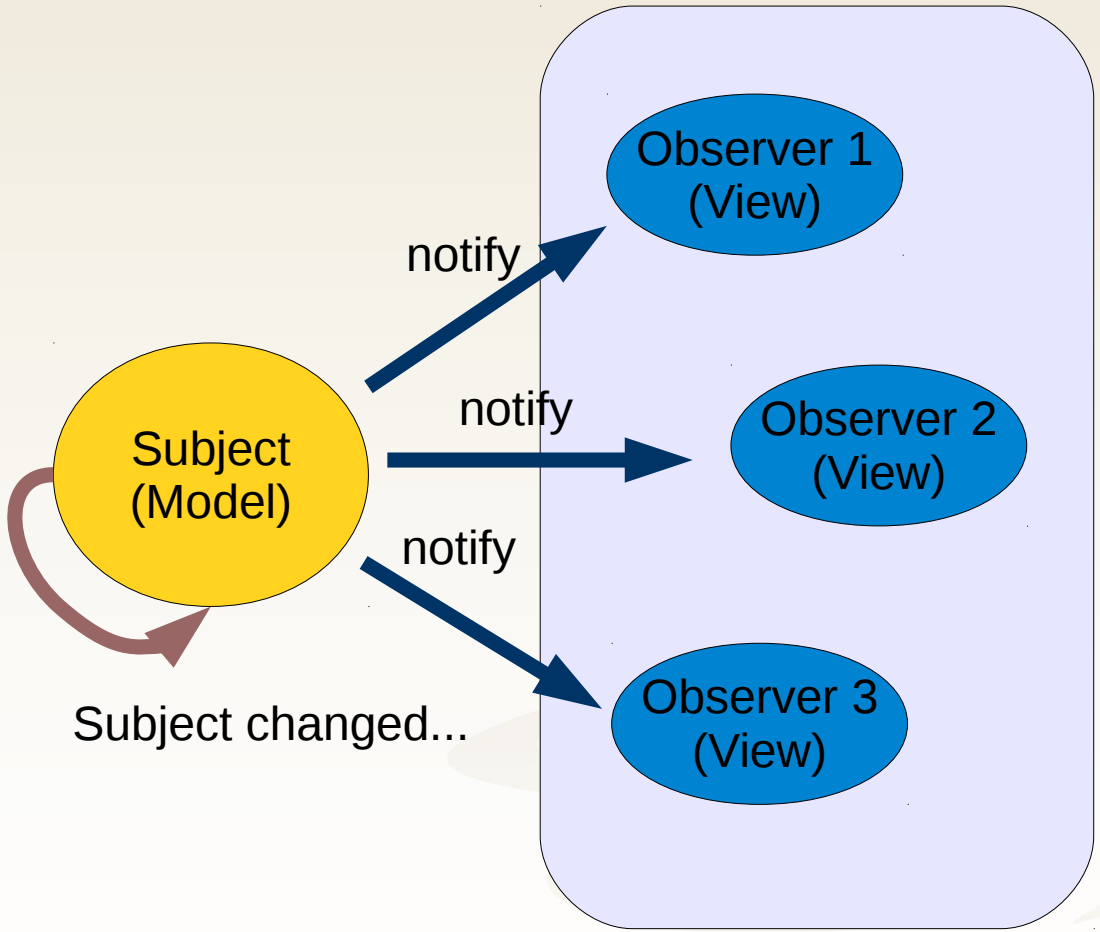


MVC Bileşenleri Arasındaki Etkileşim

- **Model**, view tarafından görüntülenen **veriyi ifade eder**
- **View** ise kullanıcıya göstereceği veriye model üzerinden erişir ve bu veriyi kullanarak **GUI render** işlemini gerçekleştirir
- **Controller**'da kullanıcı inputunun (mouse hareketler, click, keyboard input vb) ele alınıp, **iş mantığının çalıştırılması**nı ve **model üzerinde değişikliğe gidilmesi**ni sağlar

MVC ve Observer

- Model'deki değişiklik **notifikasyonlar** vasıtası ile View(lar) tarafından algılanarak ekrana yansıtılır
- Bu etkileşim **Observer örüntüsü** üzerine kuruludur



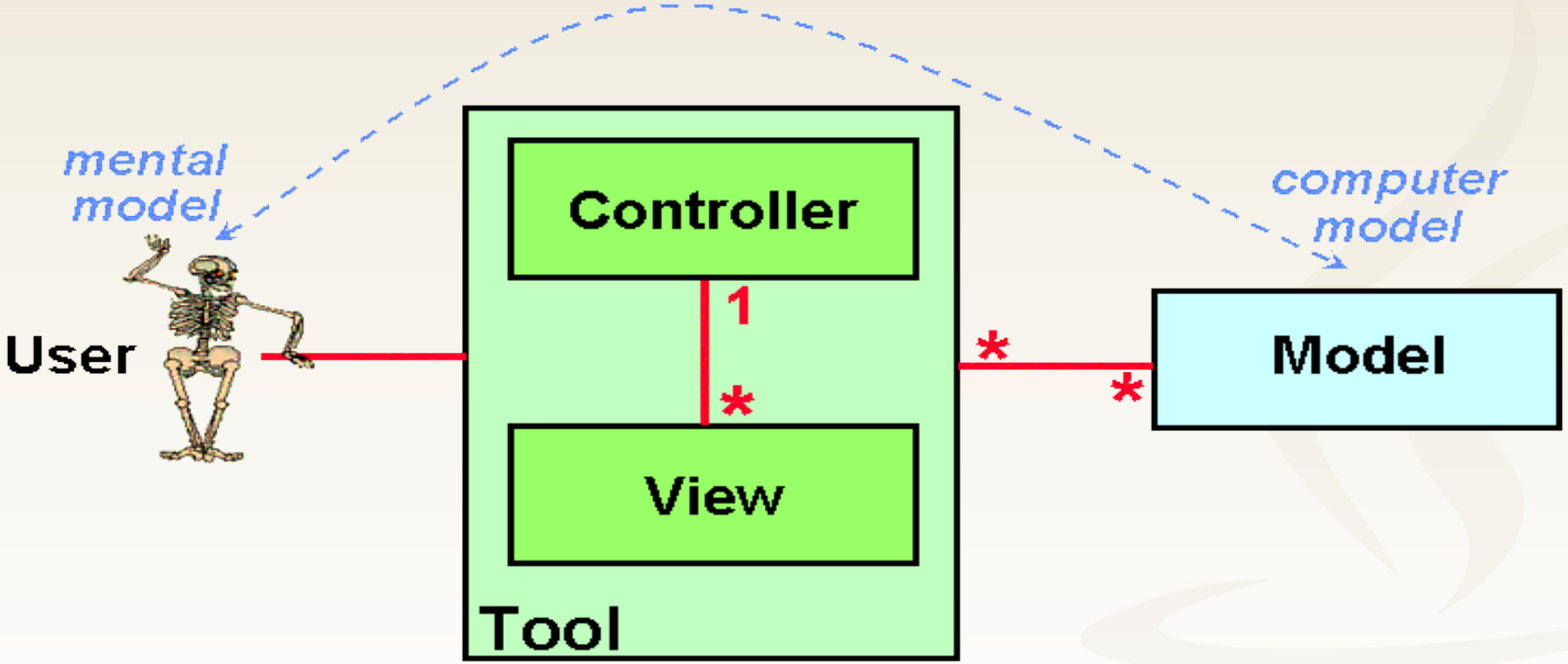
MVC'nin Amacı

- Güncel pek çok dokümanda MVC'nin amacı olarak “**iş mantığının GUI kodundan ayrılması**” olarak anlatılır
- Her bölümün **kendine özgü** ve diğerlerinden bağımsız bir **sorumluluğu** vardır
- Bu sayede view katmanında herhangi bir değişiklik yapılması gerektiğinde, bunun iş mantığında veya model'de herhangi bir probleme veya değişikliğe yol açmadan kolaylıkla yapılabileceği vurgulanır

MVC'nin Amacı

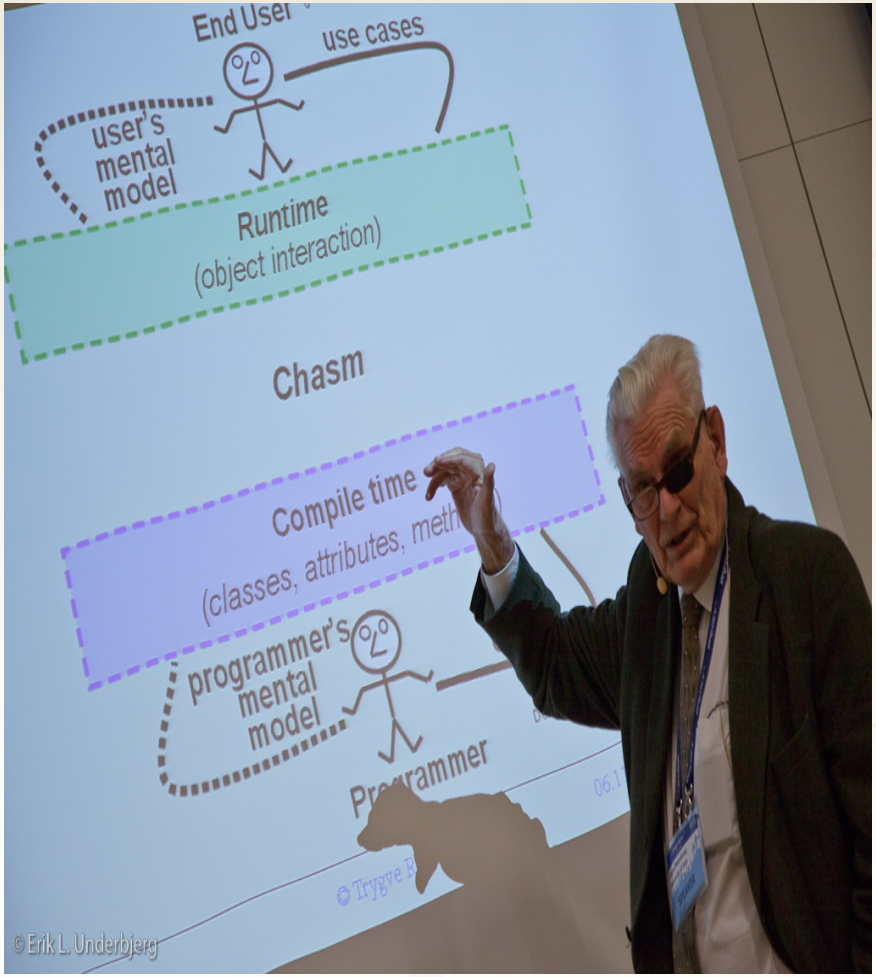
- Oysa MVC'nin mucidi Reenskaug'un asıl amacı farklıdır
- MVC'yi anlattığı makalesinde asıl amacın uygulama kullanıcılarının zihinlerindeki **mental model** ile bilgisayar sistemlerindeki **sayısal model** arasındaki **boşluğu dolduran genel bir çözüm** oluşturmak olduğunu vurgular
- Böylece domain verisi (model), doğrudan kullanıcı tarafından erişilebilir, kolaylıkla incelenebilir, yorumlanabilir ve güncellenebilir hale gelecektir

MVC'nin Amacı



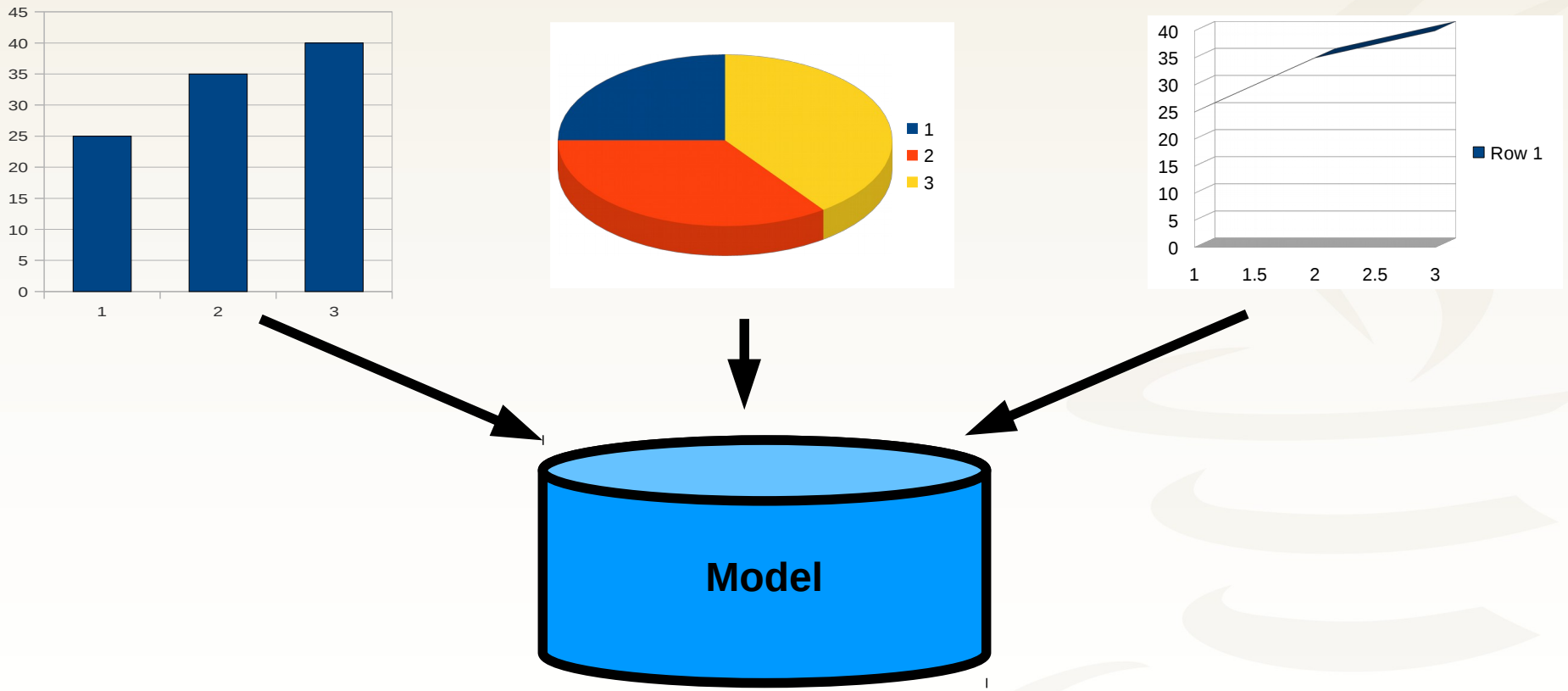
MVC'nin Amacı

- Uygulamayı modüler bir yapıya büründürmek ve **farklı görevleri farklı katmanlara ayrıştırmak** MVC için ilk hedef olmamıştır
- MVC makalesinde "**Seperation of Concern**" bir **amaç değil sonuç**tur



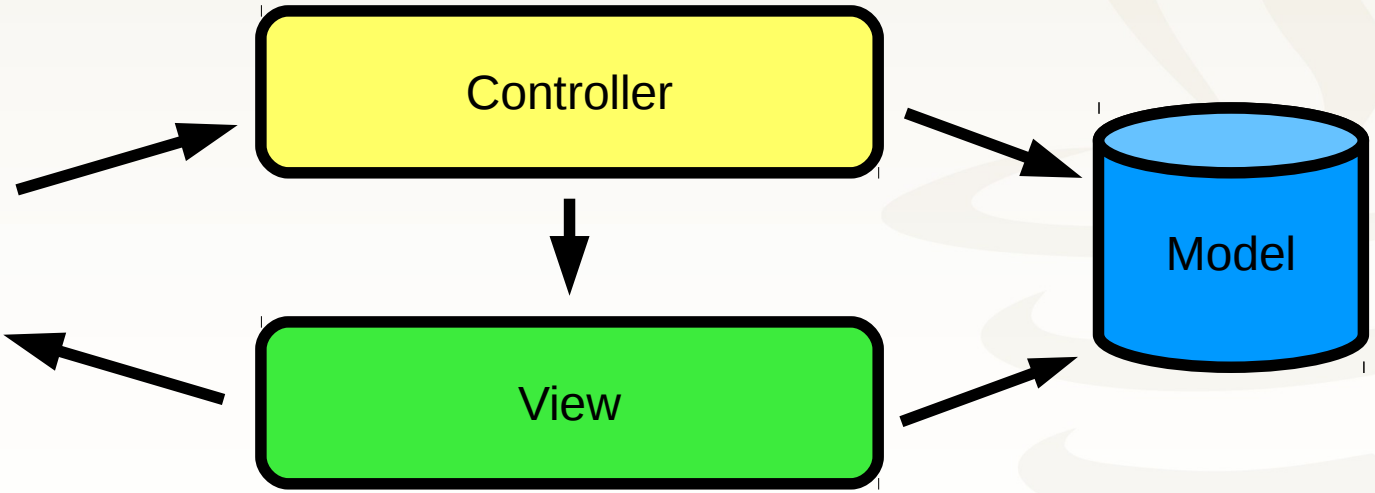
MVC'nin Amacı

- MVC'nin asıl amacını günümüzdeki dokümanlarda en iyi biçimde aşağıdaki gibi bir görsel anlatmaktadır

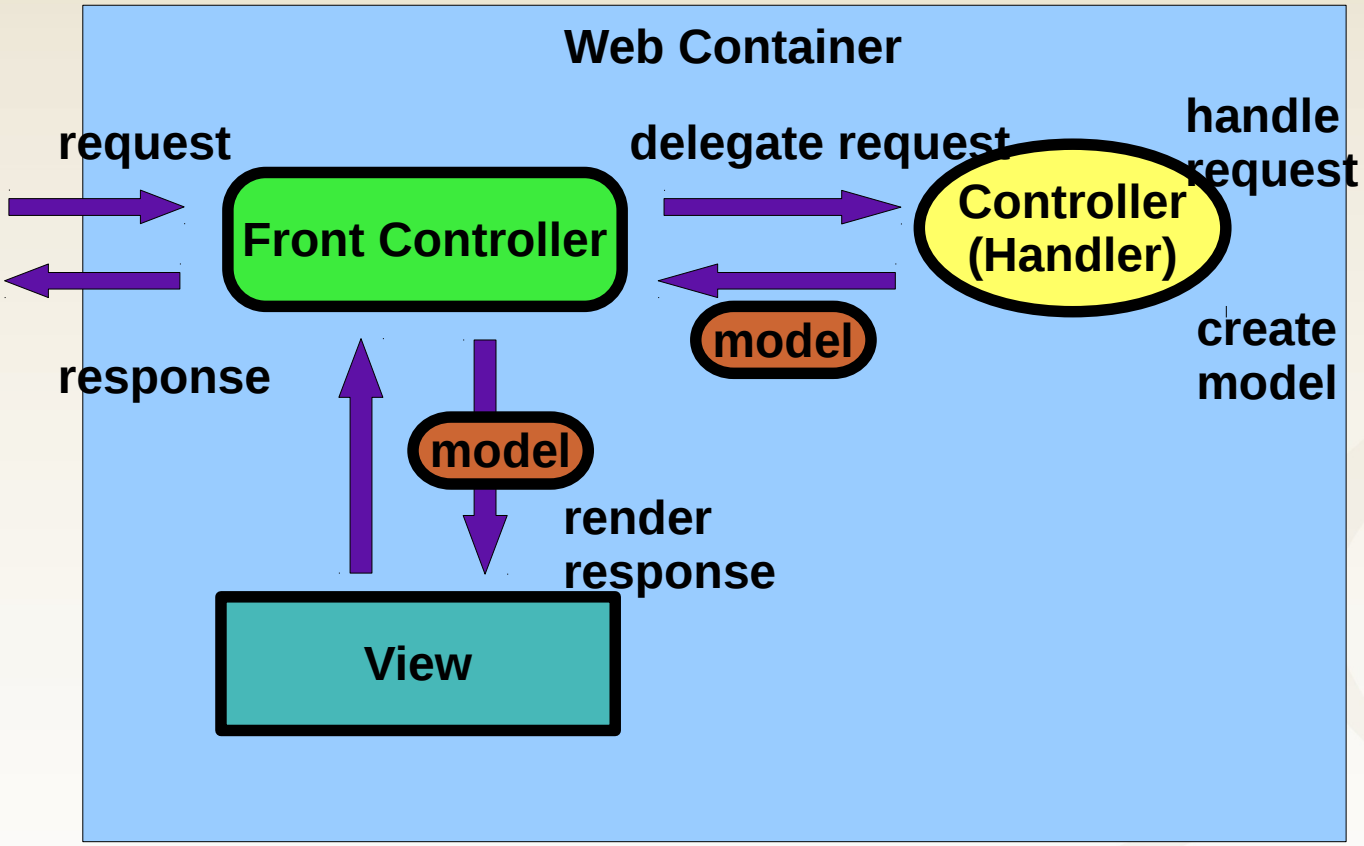


MVC ve Web

- Web uygulamalarının yaygınlaşması ve Java EE'nin popüler olması ile **MVC web platformunda da uygulanmaya** çalışılmıştır
- Web platformu ve HTTP'nin doğası gereği MVC web'e uyarlanmış ve ortaya **Web MVC** veya **MVC2** çıkmıştır



MVC ve Front Controller



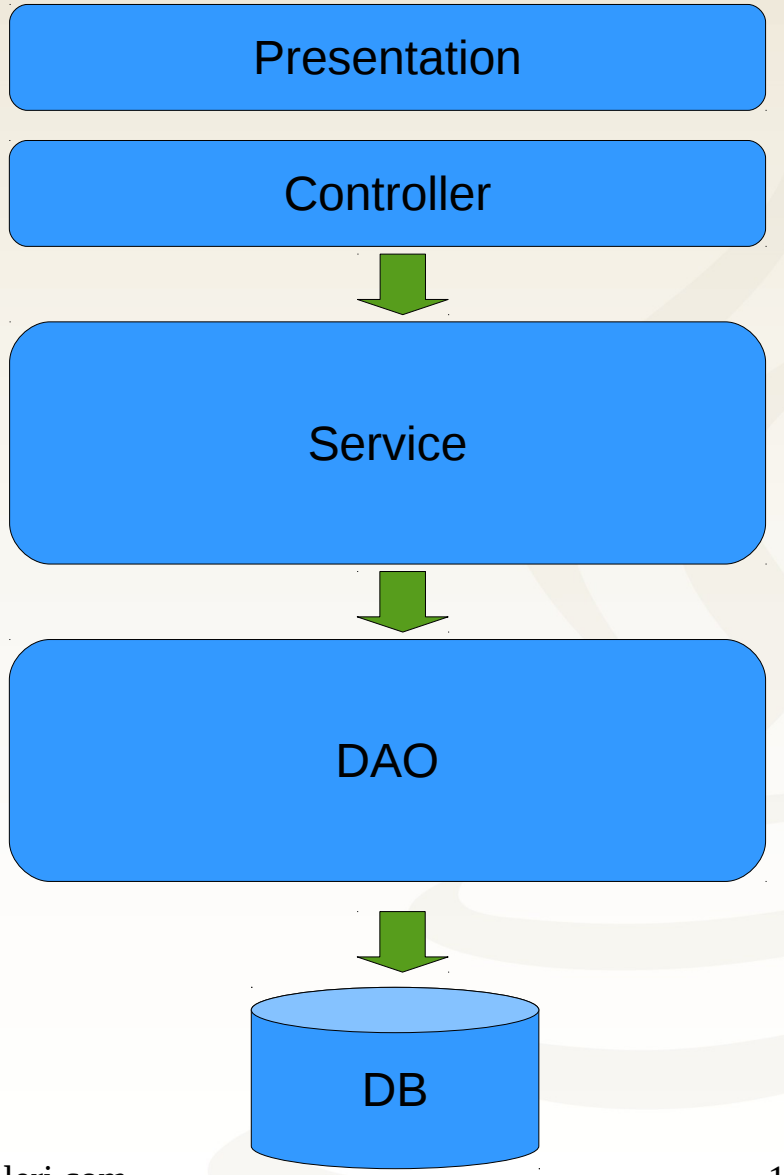
Front Controller genellikle bir Servlet olur, uygulamaya gelen bütün request'leri handle eder

Request'in hangi komuta karşılık geldiği request'in path'inden veya request parametrelerinden anlaşılır

Struts, Spring MVC gibi pek çok action oriented web framework bu örüntü üzerine kurulmuştur.
JSF, Vaadin gibi event oriented web framework'lerinin temelinde de bu örüntü yine mevcuttur

MVC ve Katmanlı Mimari

- Java EE ile geliştirilen pek çok uygulamada MVC'nin de etkisi ile **uygulamanın birden fazla katmana ayrılarak geliştirilmesi** yaygın bir mimarisel yaklaşımdır



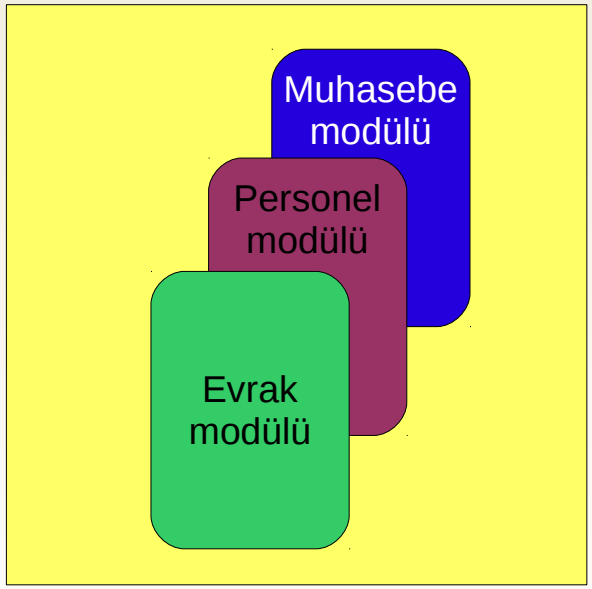
MVC ve Katmanlı Mimari

- Presentation ve Controller katmanları genellikle birlikte **UI katmanı** olarak ele alınırlar
- Kullanıcı isteklerinin işlenmesi, isteklerin servis katmanına iletilmesi ve kullanıcı arayüzünün oluşturulmasından sorumludurlar
- Servis katmanı ise **iş mantığı**nın yürütülmesinden sorumludur
- Herhangi bir arayüz teknolojisine bağımlı olmadan, **farklı istemcile**rden gelebilecek çağrıları cevaplayabilir

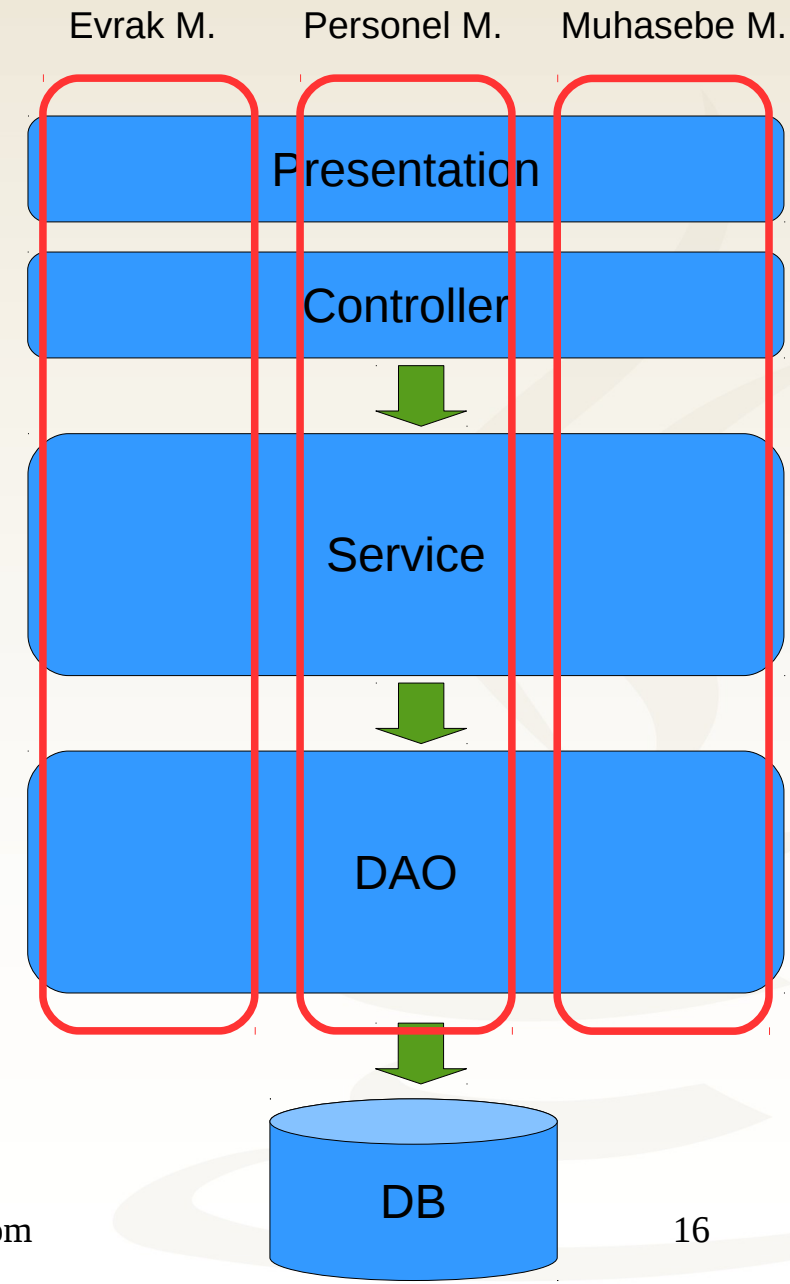
MVC ve Katmanlı Mimari

- Transaction, güvenlik, validasyon gibi ihtiyaçlarda genellikle **servis katmanı**nda ele alınırlar
- DAO katmanı ise **veri erişimi**nden sorumludur
- Kullanılan **persistence teknolojisi** yardımı ile veriye erişim, verinin saklanması, güncellenmesi ve silinmesi gibi ihtiyaçlar bu katman tarafından karşılanır
- Persistence teknolojisinin değişmesine göre bu katmanda değiştirilebilir, **strategy örüntüsü**ne karşılık gelir

Katmanlı Mimari ve Modülerlik



Yazılım Sistemi



İletişim

www.harezmi.com.tr

www.java-egitimleri.com

info@harezmi.com.tr

info@java-egitimleri.com

@HarezmiBilisim

@JavaEgitimleri