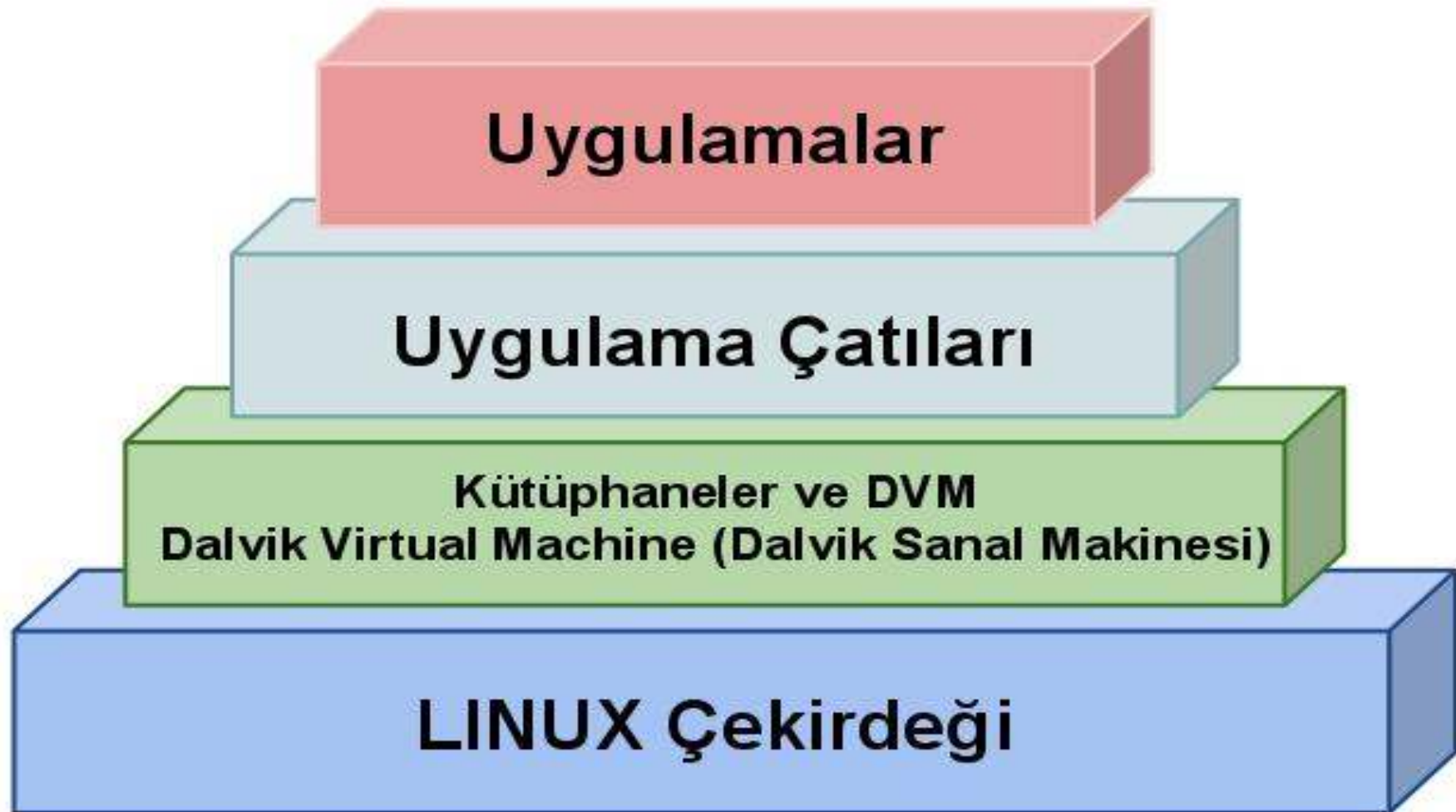


BLM401 Mobil Cihazlar için ANDROİD İşletim Sistemi

ANDROİD UYGULAMALARININ BİLEŞENLERİ

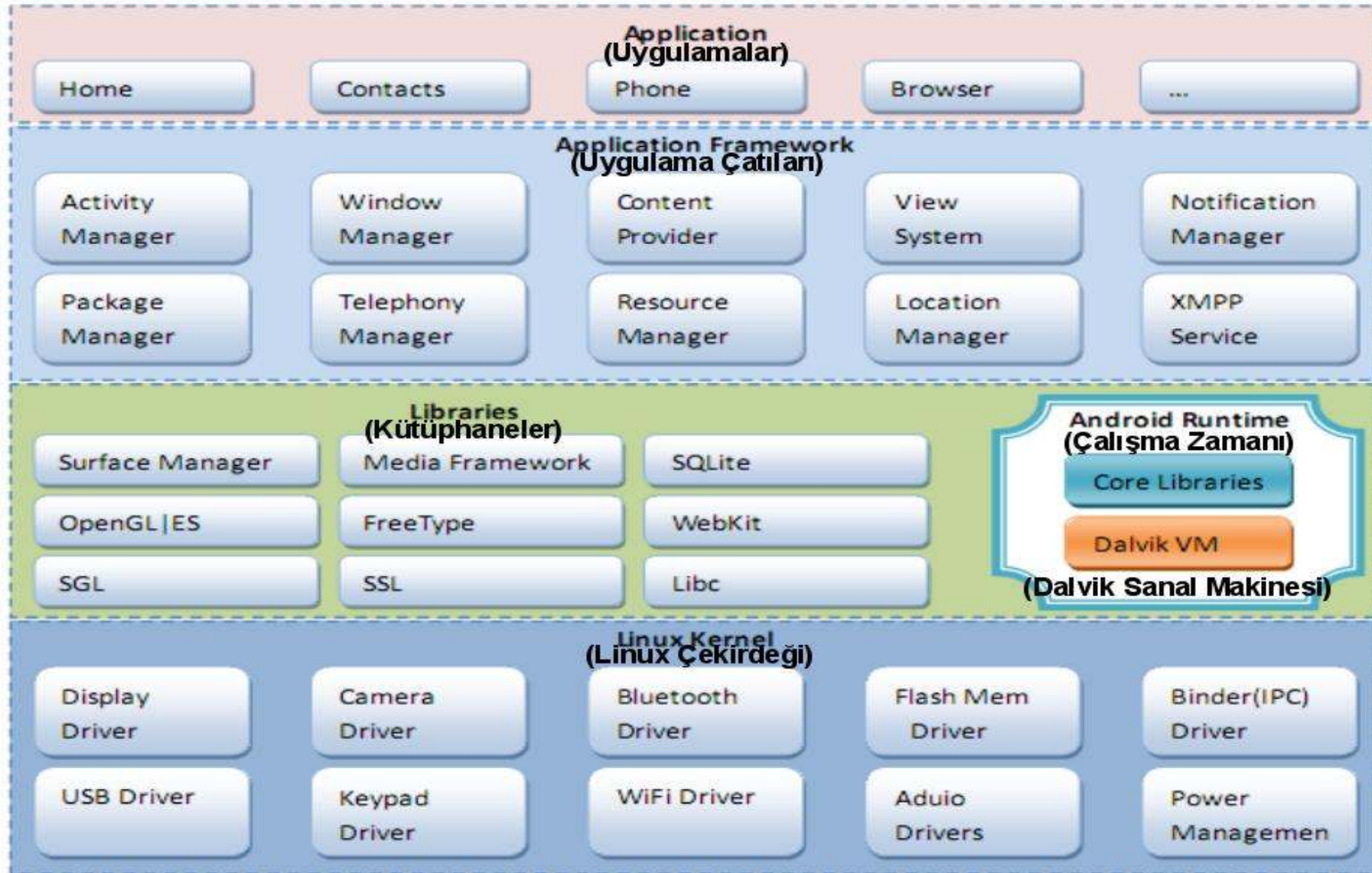


Android Yazılım Ortamı



Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Android Yazılım Ortamı



Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

UYGULAMALARIN YAPISI (1/1)

- Android işletim sisteminde bir **uygulama** başka bir uygulamayı kodu ile ilgili herhangi bir işlem yapmadan kullanabilir.
- Bunun için **uygulamaların belli bileşenlerinin** başka uygulamalar tarafından çalıştırılabililiyor olması gereklidir.
- Bu da ancak bu bileşenlerin **Java nesnelerini** oluşturmakla mümkündür.
- Bu yüzden uygulamalardaki bileşenlerin de birden fazla **giriş noktası** olabilir.

UYGULAMA BİLEŞENLERİ (1/2)

- Temel dört uygulama bileşeni vardır:
 - 1) Aktiviteler (Activity);
 - 2) Servisler (Services);
 - 3) Yayın Alıcıları (Broadcast Receiver);
 - 4) İçerik Sağlayıcıları (Content Providers).
- Bir uygulama bu bileşenlerden en az bir tanesinden oluşabileceği gibi tüm bileşenleri de kullanabilir.

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

AKTİVİTELER BİLEŞENİ (1/14)

- Görsel ara yüz sunar: metin kutuları, fotoğraflar, menü elemanları, düğmeler, vb.
- Kullanıcılarla etkileşimlidir
- Birden fazla olabilir ve aktiviteler bir birinden bağımsızdır
- Activity sınıfından türetilmiş bileşenlerdir
- Belli kurallar dahilinde bir diğer aktiviteyi başlatabilir

Activities(Aktiviteler, Vazifeler, Etkinlikler) bileşeni nedir? Ne işe yarar?

- ★ Kullanıcıların ekranda görüp etkileşimde bulunduğu formlar, düğmeler, metin kutuları kısaca ara yüzlerin hepsinin bir ekranda görüntülenmesine aktivite denir.
- ★ Görsel olarak kullanıcı ara yüzlerini bize getirir.
- ★ Aktivite kullanıcılardan ya bilgi girişi alır yada kullanıcıya bir bilgi sunuşu yapar.
- ★ Bir Android uygulaması çalışırken cihazımızın ekranında o anda ne görüyorsanız işte o görünen ekran bir aktivitedir.

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Activities(Aktiviteler) bileşeni denince aklımıza ilk olarak neler gelmelidir?

- ★ **Görsel ara yüzler sunarlar.**
- ★ **Kullanıcılarla etkileşim içindedirler.**
- ★ **Bir uygulamada en az bir tane Aktivite olmalıdır.**
- ★ **Bir uygulamada birden fazla Aktivite de olabilir.**
- ★ **Bütün Aktiviteler çalışırken birinden bağımsızdır.**
- ★ **Bir aktivite başka bir aktiviteyi kullanabilir.**
- ★ **Bir aktivite başka bir aktiviteyi de çağırabilir.**
- ★ **Aktivitelerin 3 temel hali vardır.**
 - 1-Çalışan Aktivite**
 - 2-Bekleyen Aktivite**
 - 3-Durdurulan Aktivitedir.**

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Aktivite Durumları



1-Çalışan Aktivite

Aktivite ekranı ile kullanıcı etkileşim içindedir. Aktif olarak bilgi alışverişleri yapılmaktadır.



2-Bekleyen Aktivite

Aktivite arka planda çalışmasına devam etmektedir. Cihazımızda eğer bellek sıkıntısı yaşanırsa bekletilen aktiviteler sistem tarafından yok edilebilirler.

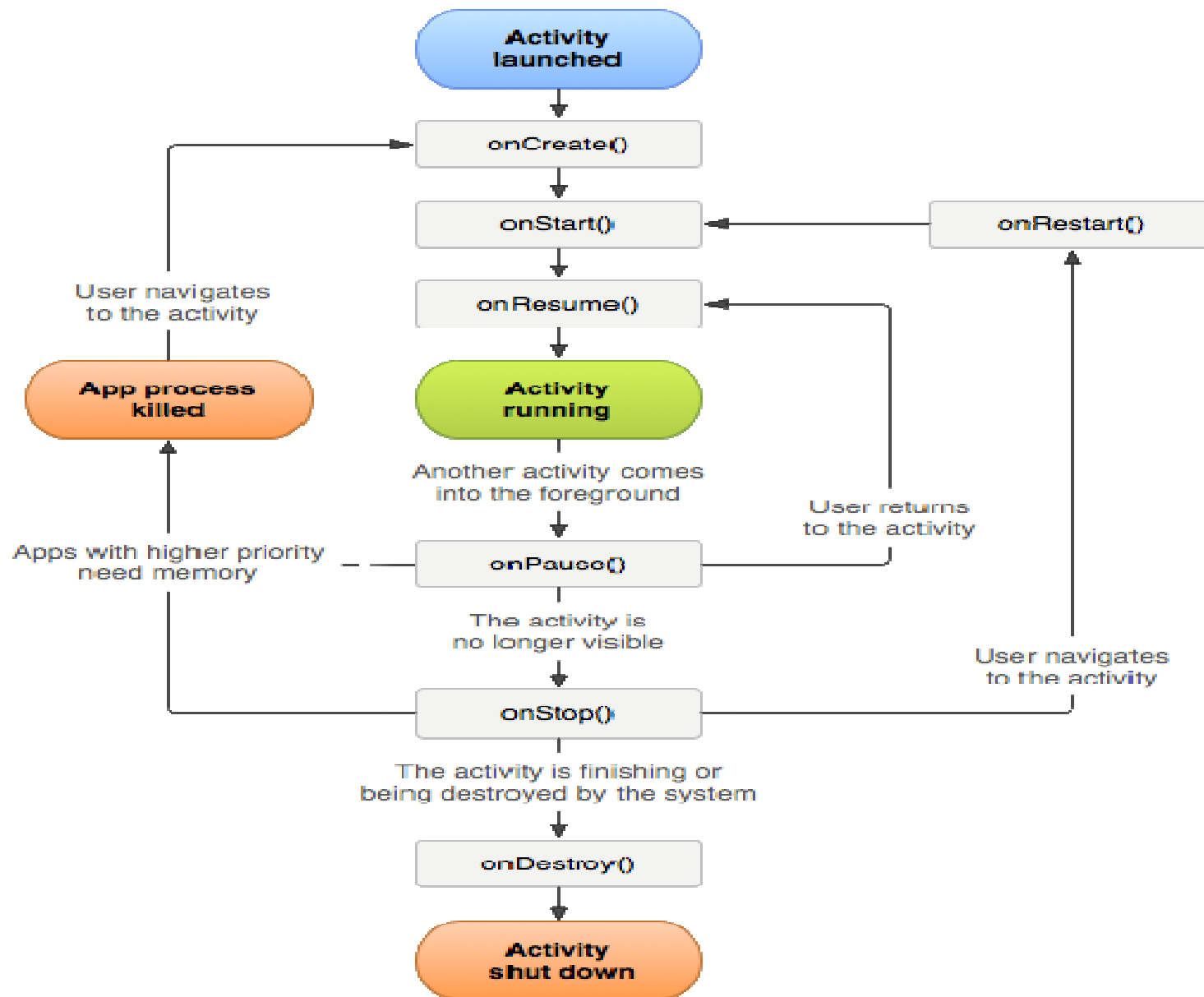


3-Durdurulan Aktivite

Aktivite durdurulmuştur ve çalışmaz vaziyettedir. Aktivitenin çalıştığı bellek kısmı serbest bırakılmıştır. Eğer aktivite tekrardan çalıştırılacak olursa en son kalınan yerdeki verilerle çalışmasına devam edecektir.

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>



AKTİVİTELER BİLEŞENİ (7/14)

- There are three key loops you may be interested in monitoring within your activity:

1)The **entire lifetime**

2)The **visible lifetime**

3)The **foreground lifetime**

AKTİVİTELER BİLEŞENİ (8/14)

- 1) The **entire lifetime** of an activity happens between the first call to onCreate(Bundle) through to a single final call to onDestroy(). An activity will do all setup of "global" state in onCreate(), and release all remaining resources in onDestroy().

AKTİVİTELER BİLEŞENİ (9/14)

For example, if it has a thread running in the background to download data from the network, it may create that thread in `onCreate()` and then stop the thread in `onDestroy()`.

AKTİVİTELER BİLEŞENİ (10/14)

2) The **visible lifetime** of an activity happens between a call to [onStart\(\)](#) until a corresponding call to [onStop\(\)](#). During this time the user can see the activity on-screen, though it may not be in the foreground and interacting with the user. Between these two methods you can maintain resources that are needed to show the activity to the user

AKTİVİTELER BİLEŞENİ (11/14)

For example, you can register a [BroadcastReceiver](#) in `onStart()` to monitor for changes that impact your UI, and unregister it in `onStop()` when the user no longer sees what you are displaying. The `onStart()` and `onStop()` methods can be called multiple times, as the activity becomes visible and hidden to the user.

AKTİVİTELER BİLEŞENİ (12/14)

- 3) The **foreground lifetime** of an activity happens between a call to [onResume\(\)](#) until a corresponding call to [onPause\(\)](#). During this time the activity is in front of all other activities and interacting with the user. An activity can frequently go between the resumed and paused states -- for example when the device goes to sleep, when an activity result is delivered, when a new intent is delivered -- so the code in these methods should be fairly lightweight.

AKTİVİTELER BİLEŞENİ (13/14)

The entire lifecycle of an activity is defined by the following Activity methods. All of these are hooks that you can override to do appropriate work when the activity changes state. All activities will implement [onCreate\(Bundle\)](#) to do their initial setup; many will also implement [onPause\(\)](#) to commit changes to data and otherwise prepare to stop interacting with the user. You should always call up to your superclass when implementing these methods.

AKTİVİTELER BİLEŞENİ (14/14)

```
public class Activity extends ApplicationContext {  
    protected void onCreate(Bundle savedInstanceState);  
    protected void onStart();  
    protected void onRestart();  
    protected void onResume();  
    protected void onPause();  
    protected void onStop();  
    protected void onDestroy();  
}
```

SERVİSLER BİLEŞENİ (1/10)

- Görsel ara yüz sunmazlar
- Kullanıcılarla direk etkileşimleri yok dolaylı olarak etkileşimlidirler
- Birden fazla olabilir ve servisler bir birinden bağımsızdır
- Genel olarak kullanıcı arayüzünde görev almazlar.
- Service sınıfından türetilmiş ve arka planda çalışan bileşenlerdir.

SERVİSLER BİLEŞENİ (2/10)

- Android multi-tasking modunu destekliyor.
- Bu da servisler arka planda çalışırken ön planda başka işlerin yapmasını sağlar.
- Servislerle ile bağlantıyı diğer uygulama bileşenleri kurar.
- Diğer bileşenler bir servisi başlatıp, durdurabilir veya çalışan bir servisle bağlantı kurup servisten desteklediği işlevleri yerine getirmesini isteyebilir.

Activities(Aktiviteler) Bileşeni ve Services(Servisler) Bileşeni İlişkisi

- ★ Aktivitelerin ihtiyacı olan verilerin ve hizmetlerin sağlanmasında rol almaktadırlar.
- ★ Kullanıcı ile bilgi alış verişi yaparak gösterilen formların hepsi ise birer aktivitedir.
- ★ Bir Android uygulamasında en az bir aktivite bulunur.

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Activities(Aktiviteler) Bileşeni ve Services(Servisler) Bileşeni İlişkisi

- ★ Mobil cihazlarda aynı anda birden fazla aktivite bir cihazın ekranında aktif olarak gösterilmez.
- ★ Çağırılan aktivite haricindeki diğer aktiviteler çalışmalarını cihazın arka planında bekleme durumuna sürdürür.
- ★ Alt yapıda aktivitelerin varlıklarını sorunsuzca sürdürmeyi garanti eden servisler bileşenidir.

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Android projelerimizde bütün bileşenleri aynı anda kullanabilir miyiz?

- ★ Bütün Android bileşenleri bir birlerine bağlıdır.
- ★ Uygulama sonlanıp kapatılana kadar etkileşim içindedirler.
- ★ Bir uygulamanın arka plana alınması kapatılması demek değildir.
- ★ Kapatılmadan kastımız uygulamanın bellekteki kapladığı alanın tamamen bellekten silinmesidir.

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Services(Servisler) bileşeni nedir? Ne işe yarar?

- ★ Aktivitelerin her biri bir uygulama özelliği taşımaktadır.
- ★ Bir aktivite ile çalışırken diğer aktivitelerin tamamını sonlandırıp kapatılması hiçte hoş bir durum değildir.
- ★ Arka planda çalışmasına devam eden uygulamaların(aktivitelerin) muhatabı en az bir servis bileşenidir.
- ★ Bir aktiviteyi birden fazla servis bileşeni de organize edebilir.
- ★ Aktivitelerin kapatılmadan arka planda varlıklarını sürdürmesi devam servisler bileşeni ile sağlanır.

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Servislerin kaç durumu vardır?

Servislerin 4 temel hali vardır.

- ★ 1-İlk defa çağırılan Servis
- ★ 2-Çalışan Servis
- ★ 3-Bekletilen Servis
- ★ 4-Sonlandırılan Servis

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Servisler bileşenin yaşam döngüsünde kullanılan metotlar.

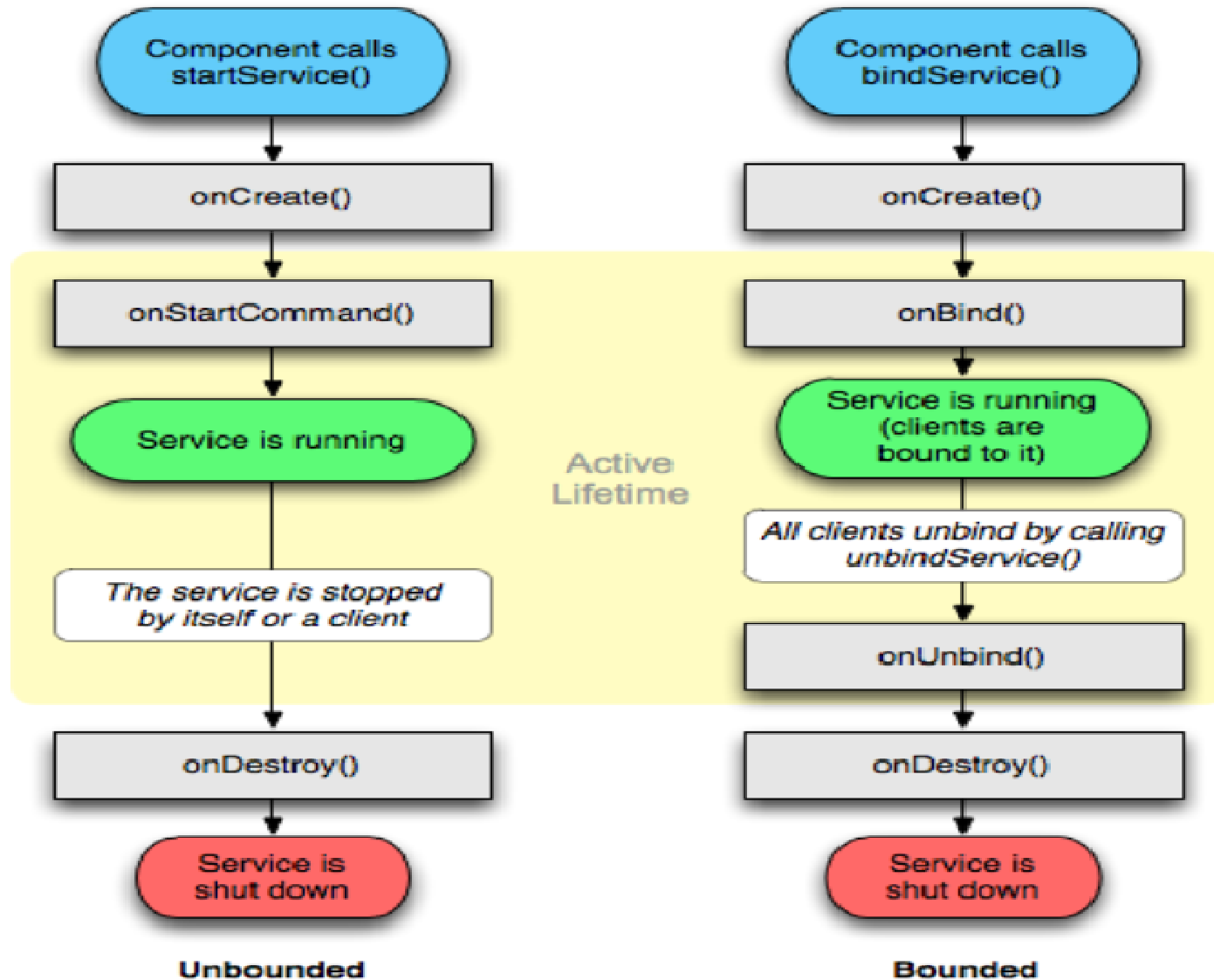
Aktivite yaşam döngüsünde kullanılan metotların çalışması servisler bileşenini de etkiler.

```
void onCreate(),  
void onStart(Intent intent),  
void onDestroy(),  
IBinder onBind(Intent intent),  
boolean onUnbind(Intent intent),  
void onRebind(Intent intent)
```

Bu servis metotlarının çalışmaları bir birlerine bağlıdır.

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Services lifecycle (Servisler Yaşam Döngüsü)



Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Context Sınıfı ve Servisler Bileşeni İlişkisi

- ★ Android işletim düzeninde kaynaklara erişimlerde aktiviteler ve servisler Context sınıfından yararlanmaktadırlar.
- ★ Verilere erişme söz konusu olduğu durumlarda alt yapıda Context sınıfı devreye girmektedir.

Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

YAYIN ALICILARI BİLEŞENİ (1/8)

- Görsel ara yüz sunmazlar ve uygulamaların isteklerine göre gerekli işlemleri yaparlar.
- Kullanıcılarla direk etkileşimleri yok dolaylı olarak etkileşimlidirler
- Birden fazla olabilir ve yayın alıcıları bir birinden bağımsızdır
- Bir aktiviteyi ve servisi çağırıp kullanabilir
- BroadcastReceiver sınıfından türetilmiş bileşenlerdir.

YAYIN ALICILARI BİLEŞENİ (2/8)

- Genelde yayınlar Android tarafından yapılır.
- Uygulamalar kendileri de yayın yapabilirler.
- Uygulamalar diğer uygulamaların yaptıkları yayınları da alabilir.
- Yayın alıcılar aldıkları yayına göre bir aktivite başlatabilir veya bir şekilde kullanıcının dikkatini çekecek şeyler – müzik çalma, cihazı titretme, vb. – yapabilirler.

YAYIN ALICILARI BİLEŞENİ (3/8)

- Android sistem için önemli gördüğü durumlarda (örneğin, pil azaldığında, güç bağlantısı yapıldığında, cihaz kapanırken, vb.) uygulamaların gerekiyorsa kendi işlemlerini yapmaları, bazı durumlar için önlemlerini almaları için “yayın yapar”.

YAYIN ALICILARI BİLEŞENİ (4/8)

- Uygulamasının sistemde yapılan herhangi bir yayından haberdar olmasını isteyen yazılımcılar, yayın almak istedikleri olaylar için uygulamayı “kayıt” ettirirler ve böylelikle kaydolunan olay olduğunda Android uygulamayı haberdar eder.
- Uygulama da yayını aldığı anda yani bir olayın olduğunu öğrendiğinde ona göre işlem yapar.

Broadcast Receiver (Yayın Alıcısı) Bileşeni nedir? Ne işe yarar?

- ★ Android işletim düzeni tarafından verilen mesajların muhatabı yayın alıcılardır.
- ★ Yayın alıcıları aldıkları yayına göre bir aktiviteyi veya servisi başlatabilirler.
- ★ Duruma göre kullanıcının dikkatini çekecek müzik çalma, cihazı titretme gibi işlemler yapabilirler.

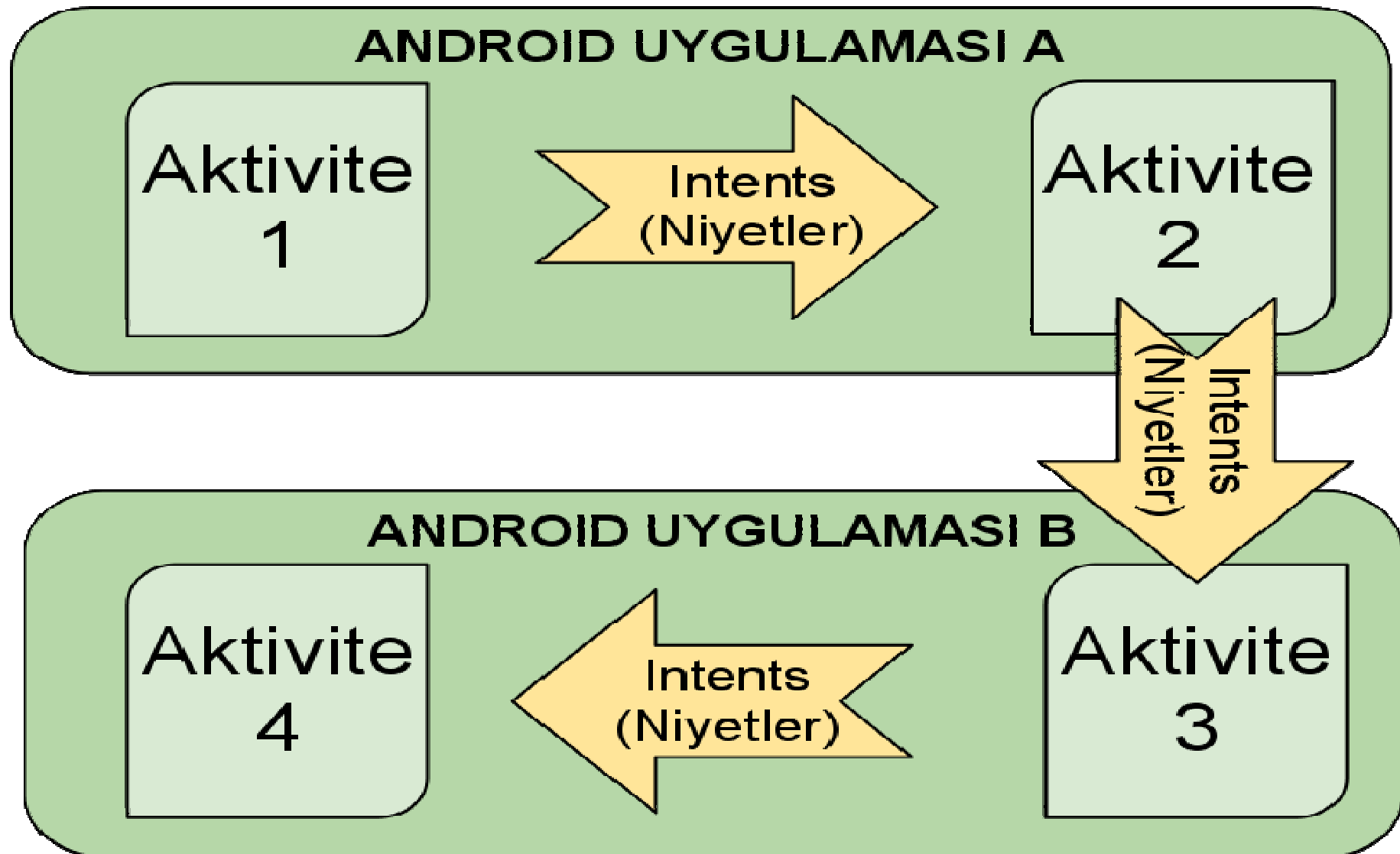
Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Intent nesnesi nedir? Ne işe yarar?

- ★ Intent nesnesi aktivitelerin, servislerin ve yayın alıcıların aktifleştirilmesinde kullanılan metoda ait bilgileri, mesajları bünyesinde barındırarak saklar.
- ★ Android işletim düzenindeki meydana gelen değişikliklerden Broadcast Intent (Yayın Niyeti) uygulamaları haberdar eder.
- ★ Broadcast Receiver (Yayın Alıcısı) bileşeni Intent nesnesi olmadan sağlıklı çalışamaz.

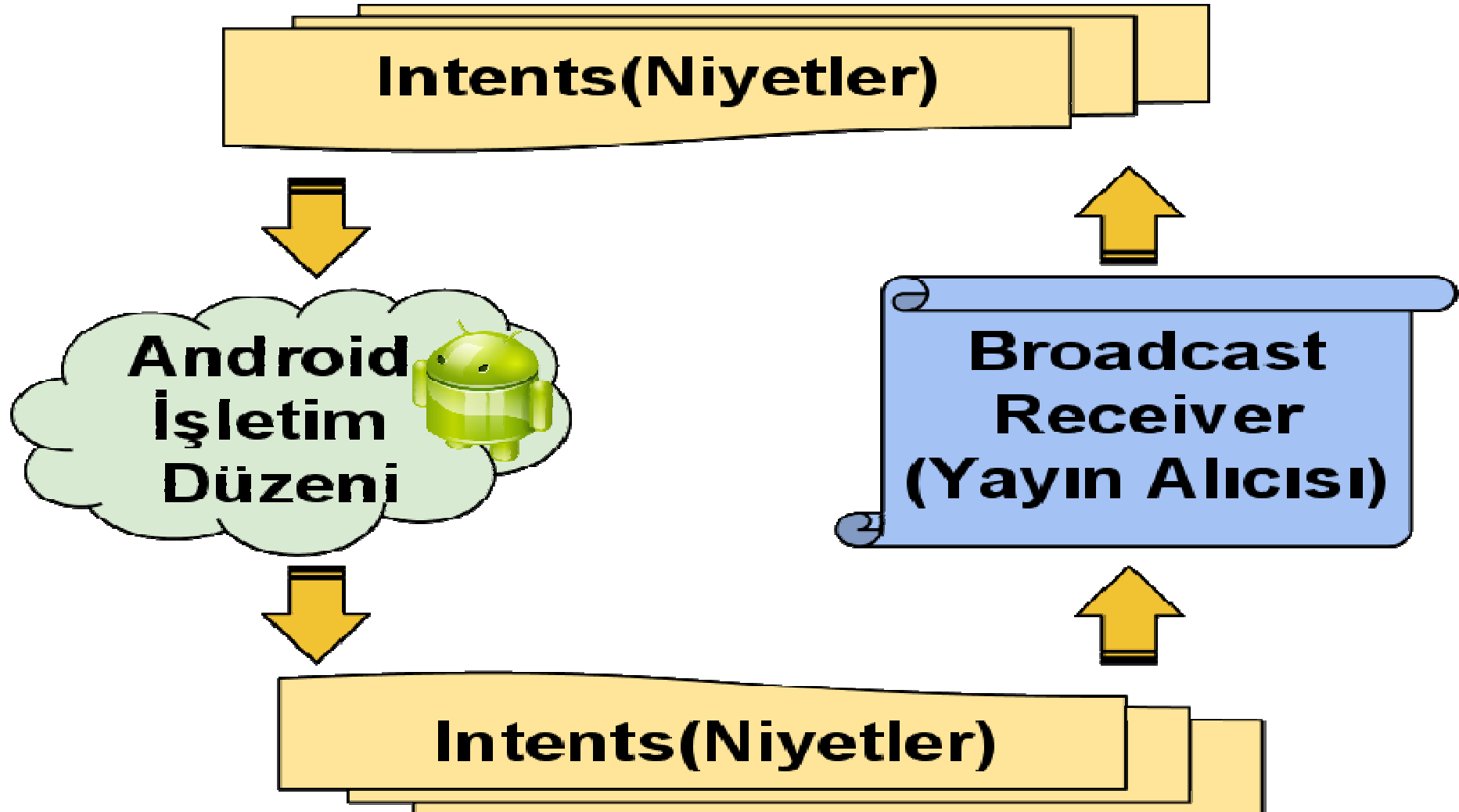
Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Broadcast Intent(Yayın Niyeti)



Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

Broadcast Receiver (Yayın Alıcısı) ile Intents (Niyetler, Kasıtlar) İlişkisi



Kaynak: <http://www.slideshare.net/mimaraslan/android-seminerleri>

İÇERİK SAĞLAYICILAR (1/2)

- Uygulamanın sabit olduğu verilere diğer uygulamaların ulaşmasını sağlar.
- ContentProvider sınıfından türetilmiş bileşendir.
- Veriler dosya sisteminde, SQLite veri tabanında veya başka bir metotla tutulabilir.

İÇERİK SAĞLAYICILAR (2/2)

- Tanımlanmış belli fonksiyonları ile tuttukları veriye diğer uygulamaların ulaşmasını veya aynı tipte veri kaydedilmesini sağlarlar.
- Diğer uygulamalar ContentResolver nesnelerini kullanarak bir içerik sağlayıcı ile etkileşimde bulunabilirler.

(son)

BAŞARILAR ...