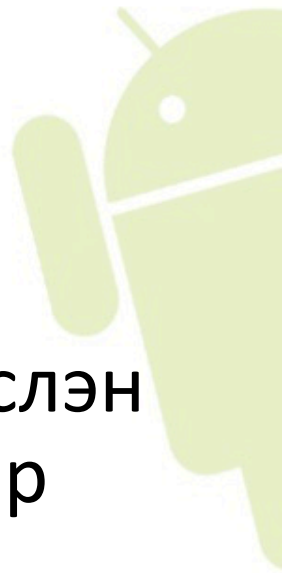




Мобайл програмчлал – Лекц 6

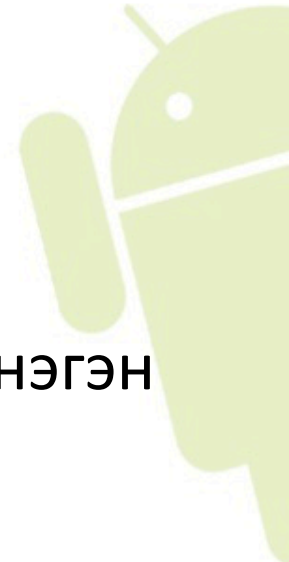
Ж.Золжаргал
(j.zoljargal@csms.edu.mn)

Introducing Services



- Activity нь дэлгэцэнд интерфэйсийг дүрслэн ажилладаг бол Service нь background-аар ажилладаг
- Хэрэглээ: Content provider-ийг шинэчлэх, Intent-үүд шидэх, Notification үүсгэх
- Service-г эхлүүлэх, зогсоох, удирдах нь бусад програмын бүрэлдэхүүн хэсэгт агуулагдах бөгөөд тэдгээр нь өөр Service, Activity, Broadcast Receiver байдаг.

Introducing Services



- Хэрэв тухайн програм үе үе эсвэл байнга ямар нэгэн үйлдэл хийх шаардлагатай бөгөөд тэр нь хэрэглэгчээс өгөгдөл авах шаардлагагүй бол тохиромжтой.
- Ажиллаж буй Service нь нөөц шаардлагатай үед төгсгөдөг invisible болон inactive Activity -ээс өндөр зэрэглэлтэй
- Одоогын идэвхитэй ажиллаж байгаа Activity-д нэмэлт нөөц хэрэгтэй болсон тохиолдолд Service-г цагаас нь өмнө зогсоож болох бөгөөд нөөц боломжтой болсон үед Service автоматаар үргэлжлүүлэн ажиллана

Introducing Services



- Үе үе мэдээллээ шинэчлэдэг хэрэглэгчийн оролцоотой Апп бичихэд Service-р гүйцэтгэж болно
 - MP3 Player, Sports-score monitor
- Андроид нь өөрөө Service-үүд агуулсан байдаг
 - Location Manager, Media Controller, Notification Manager

Types of Services



- Foreground
 - Хэрэглэгчид мэдэгдэл хүргэх байдлаар ажиллана. Status bar дээр үзэгдэнэ (Audio app)
- Background
 - Шууд хэрэглэгчид үр дүн мэдэгдэх шаардлага байхгүй
- Bound
 - Client-server зарчимаар сервистэй ажиллах бөгөөд хүсэлт явуулж хариугаа авна

Creating and Controlling Services

- Service тодорхойлоход Service классаас удамших бөгөөд onCreate болон onBind үзэгдлийг даран тодорхойлох шаардлагатай



Creating and Controlling Services

```
import android.app.Service;
import android.content.Intent;
import android.os.IBinder;

public class MyService extends Service {

    @Override
    public void onCreate() {
        // TODO: Actions to perform when service is created.
    }

    @Override
    public IBinder onBind(Intent intent) {
        // TODO: Replace with service binding implementation.
        return null;
    }
}
```



Creating and Controlling Services



- Мөн onStart үзэгдлийг даран тодорхойлж болох бөгөөд энэ нь startService функц дуудагдахад үүсдэг ба амьдралынхаа мужид хэд хэдэн удаа биелэгдэж болно

```
@Override  
public void onStart(Intent intent, int startId) {  
    // TODO: Actions to perform when service is started.  
}
```

- Шинэ Service үүсгэхэд manifest XML-н application tag-д бүртгүүлэх шаардлагатай

```
<service android:enabled="true" android:name=".MyService"></service>
```


Starting, Controlling, and Interacting with a Service



- `startService` функцээр Service-г эхлүүлэх бөгөөд Implicitly болон Explicitly байдлаар дуудаж болно.
- Хэрэв тухайн Service-д ямар нэг permission шаардлагтай бөгөөд тэр нь тухайн Апп-д байхгүй бол `SecurityException` шиддэг.

```
// Implicitly start a Service
startService(new Intent(MyService.MY_ACTION));
// Explicitly start a Service
startService(new Intent(this, MyService.class));
```

- `MyService` класс-д `MY_ACTION` шинж чанарыг тодорхойлох шаардлагатай бөгөөд `Intent Filter` tag-д бүртгүүлнэ

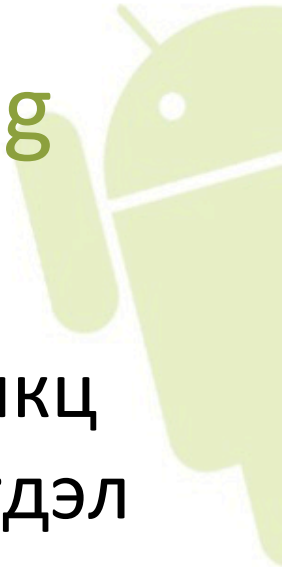
Starting, Controlling, and Interacting with a Service



- Service-г зогсооход stopService функц ашиглах бөгөөд Intent-г дамжуулна

```
ComponentName service = startService(new Intent(this, BaseballWatch.class));  
// Stop a service using the service name.  
stopService(new Intent(this, service.getClass()));  
// Stop a service explicitly.  
try {  
    Class serviceClass = Class.forName(service.getClassName());  
    stopService(new Intent(this, serviceClass));  
} catch (ClassNotFoundException e) {}
```

Starting, Controlling, and Interacting with a Service



- Service ажиллаж байхад `startService` функц дуудагдвал тухайн Service-н `onStart` үзэгдэл дахин ажиллах болно
- Хэдэн ч удаа `startService` функц дуудсан нэг Service давхар ажиллахгүй бөгөөд `stopService` функцийг ганц удаа дуудсанаар тухайн Service-г зогсооно.

Binding Activities to Services



- Activity-ээс Service-г эхлүүлж болох бөгөөд Service-н тохиолдолыг үүсгэнэ

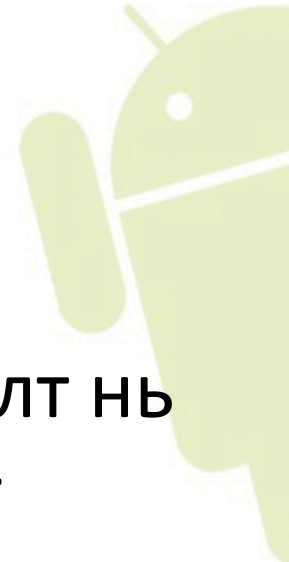
```
private final IBinder binder = new MyBinder();
```

```
@Override
```

```
public IBinder onBind(Intent intent) {  
    return binder;  
}
```

```
public class MyBinder extends Binder {  
    MyService getService() {  
        return MyService.this;  
    }  
}
```

Binding Activities to Services



- Service болон Activity хоорондын холболт нь ServiceConnection классаар дүрслэгддэг
- ServiceConnection-г хэрэгжүүлж onServiceConnected болон onServiceDisconnected функцыг даран тодорхойлж Service-г холбоно

Binding Activities to Services



```
// Reference to the service
private MyService serviceBinder;

// Handles the connection between the service and activity
private ServiceConnection mConnection = new ServiceConnection() {

    public void onServiceConnected(ComponentName className, IBinder service) {
        // Called when the connection is made.
        serviceBinder = ((MyService.MyBinder)service).getService();
    }

    public void onServiceDisconnected(ComponentName className) {
        // Received when the service unexpectedly disconnects.
        serviceBinder = null;
    }
};
```

Binding Activities to Services



- Холболтыг `bindService` функцээр гүйцэтгэх бөгөөд `intent` болон `serviceConnection`-ний объектыг дамжуулна

```
@Override
public void onCreate(Bundle icle) {
    super.onCreate(icle);

    // Bind to the service
    Intent bindIntent = new Intent(MyActivity.this, MyService.class);
    bindService(bindIntent, mConnection, Context.BIND_AUTO_CREATE);
}
```

Binding Activities to Services



- `onServiceConnected` функцээр үүсгэгдсэн `serviceBinder` объектоор бүх `public` гишүүн өгөгдөл болон функцрүү хандах боломжтой.
- Андроид нь `shared memory` дэмждэггүй бөгөөд зарим тохиолдолд ялгаатай апп-ууд нь нэг `Service`-тэй холбогдож болдог

Using Background Worker Threads

- Андройд програмын бүрэлдхүүн хэсгүүдэд Activities, Services, Broadcast Receivers-үүд агуулагдах бөгөөд програмын үндсэн thread байдлаар ажилладаг
- Background thread-н гол хэрэглээ бол “Application Unresponsive” байдал үүсгэхгүй байх
- Цаг их шаардсан процессууд дээр ашиглах шаардлагатай. Файлтай ажиллах, сүлжээгээр өгөгдөл дамжуулах, Төвөгтэй тооцоолол



Creating New Threads



- `java.lang.Thread` санд байрлах `Thread` болон `Handler` классын тусламжтай хүү `Thread` үүсгэх болон удирдах боломжтой.

```
// This method is called on the main GUI thread.  
private void mainProcessing() {  
    // This moves the time consuming operation to a child thread.  
    Thread thread = new Thread(null, doBackgroundThreadProcessing, "Background");  
    thread.start();  
}
```

```
// Runnable that executes the background processing method.  
private Runnable doBackgroundThreadProcessing = new Runnable() {  
    public void run() {  
        backgroundThreadProcessing();  
    }  
};
```

```
// Method which does some processing in the background.  
private void backgroundThreadProcessing() {  
    [ ... Time consuming operations ... ]  
}
```

Synchronizing Threads for GUI Operations



- GUI орчинд background thread ашиглаж байгаад үндсэн програмын GUI дээр өөрчлөлт хийх шаардлага гардаг
- Handler классын post функцын тусламжтай үүнийг хийх боломжтой
- Background-аар ажиллаж буй Thread-н Handler объектын post функцээр GUI –д өөрчлөлт хийж болно

Synchronizing Threads for GUI Operations



```
// Initialize a handler on the main thread.
private Handler handler = new Handler();

private void mainProcessing() {
    Thread thread = new Thread(null, doBackgroundThreadProcessing, "Background");
    thread.start();
}

private Runnable doBackgroundThreadProcessing = new Runnable() {
    public void run() {
        backgroundThreadProcessing();
    }
};
```

Synchronizing Threads for GUI Operations



```
// Method which does some processing in the background.  
private void backgroundThreadProcessing() {  
    [ ... Time consuming operations ... ]  
    handler.post(doUpdateGUI);  
}
```

```
// Runnable that executes the update GUI method.  
private Runnable doUpdateGUI = new Runnable() {  
    public void run() {  
        updateGUI();  
    }  
};
```

```
private void updateGUI() {  
    [ ... Open a dialog or modify a GUI element ... ]  
}
```

Synchronizing Threads for GUI Operations

- Handler класс нь заасан цагт мэдээллийг шинэчлэх `postDelayed` болон `postAtTime` функцтэй

