Programming Handheld Systems
The Service Class
Today's Topics

The Service Class
Implementing started Services
Implementing bound Services
The Service Class

No user interface

Two main uses

Performing background processing

Supporting remote method execution
Starting A Service

Components can start a service by calling

Context.startService(Intent intent)
Starting A Service

Once started, the Service can run in the background indefinitely

Started Services usually perform a single operation & then terminate themselves

By default, Services run in the main thread of their hosting application
Binding to a Service

Components can bind to a Service by calling

Context.bindService (Intent service,
ServiceConnection conn,
int flags)
Binding to a Service

Binding to a Service allows a component to send requests and receive responses from a local or a remote service. At binding time, the Service will be started, if necessary. Service remains active as long as at least one client is bound to it.
ServiceLocalLoggingService

Client sends a log message to a local Service
The Service writes the message to the log console
LoggingService implemented as an IntentService
**IntentService**

**Subclass of Service**

*IntentService* requests are handled sequentially in a single worker thread.

*IntentService* is started and stopped as needed.
MusicPlayerForegroundService

Client Activity starts service to play a music file

Service plays music as a foreground service

Service continues playing even if Client Activity pauses or terminates
Binding to Remote Services

Using the Messenger class
Defining an AIDL interface
Implementing Services with Messengers

Messenger managers a Handler

Allows Messages to be sent from one component to another across process boundaries

Messages are queued and processed sequentially by recipient
Implementing Services with Messengers

Service creates a Handler for processing specific messages
Service creates a Messenger that provides a Binder to a Client
**Implementing Services with Messengers**

Client uses the Binder to create its own Messenger

Client uses the Messenger to send Messages to the Service
Client sends log messages to a remote Logging Service
Logging Service writes messages to a log console
If a Service must be accessed concurrently, then develop an AIDL interface.
Implementing Services with AIDL

Define remote interface in the Android Interface Definition Language (AIDL)
Implement remote interface
Implement Service methods
Implement Client methods
Define Remote Interface

Declare interface in a .aidl file
This defines how components can interact with the Service
AIDL Syntax

Similar to Java interface syntax
Can declare methods
Cannot declare static fields
**AIDL Syntax**

**Non-primitive remote method parameters require a directional tag:**
- **in:** transferred to the remote method
- **out:** returned to the caller
- **inout:** both in and out
AIDL Data Types

Java primitive types
String
CharSequence
AIDL Data Types

Other AIDL-generated interfaces
Classes implementing the Parcelable protocol
AIDL Data Types

List

List elements must be valid AIDL data types

Generic lists supported
AIDL Data Types

Map
Map elements must be valid AIDL data types
Generic maps not supported
Example Remote Interface

interface KeyGenerator {
    String getKey();
}

Client binds to a Service hosted in another application
Client retrieves an ID from service
Next Time

Bluetooth