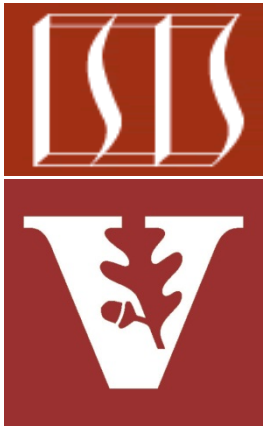
Android Services & Security: Overview of Android Services

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt



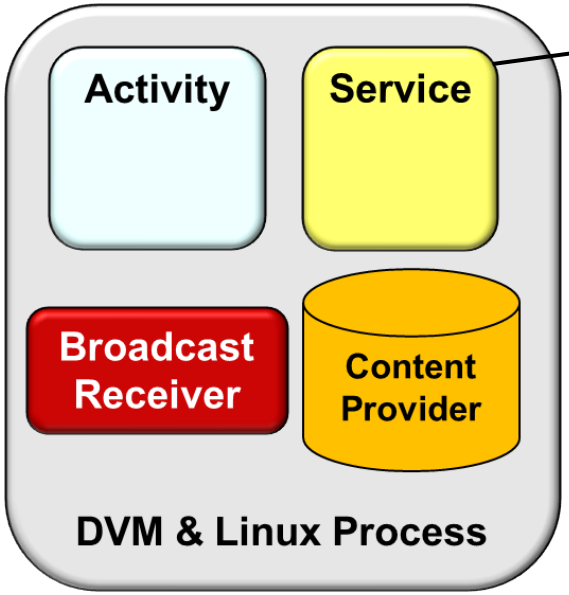
Professor of Computer Science

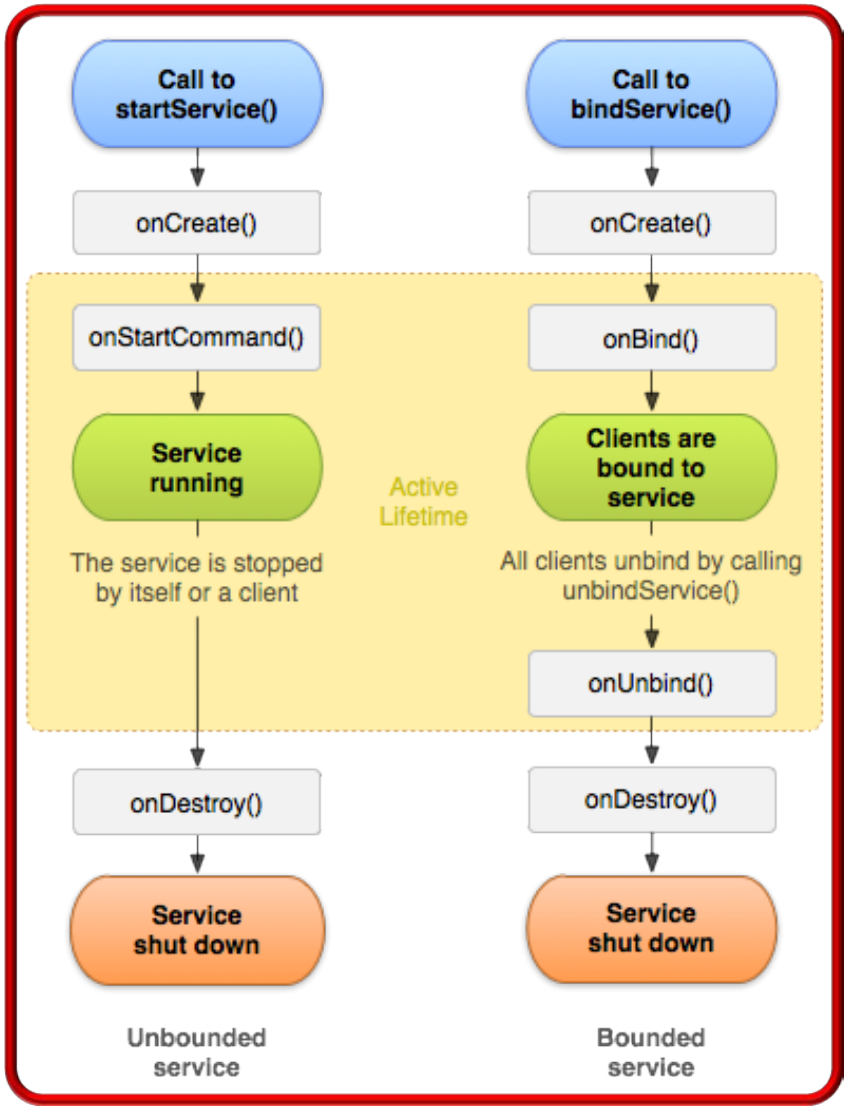
Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Module

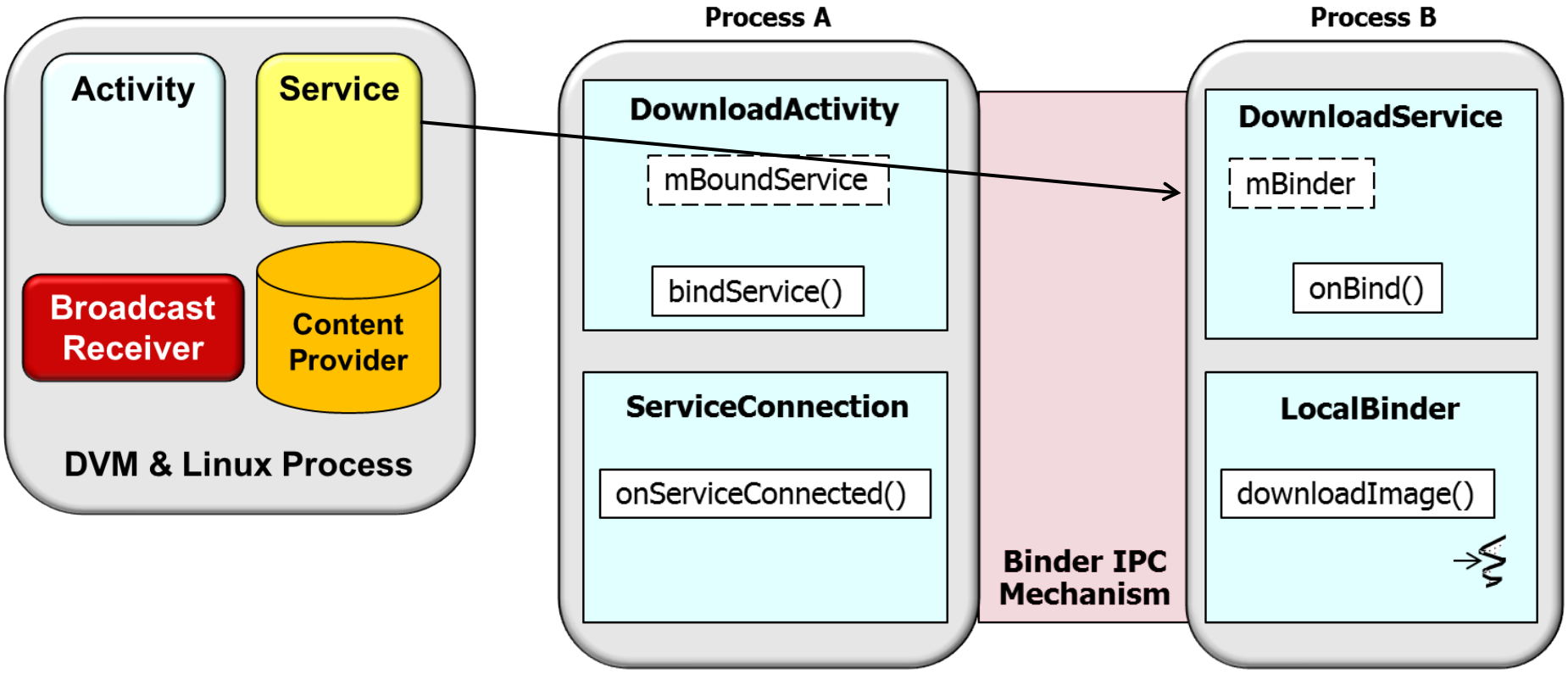
- Understand what a Service is, the two types of Services Android supports, & the steps involved in implementing & configuring a Service





Overview of a Service

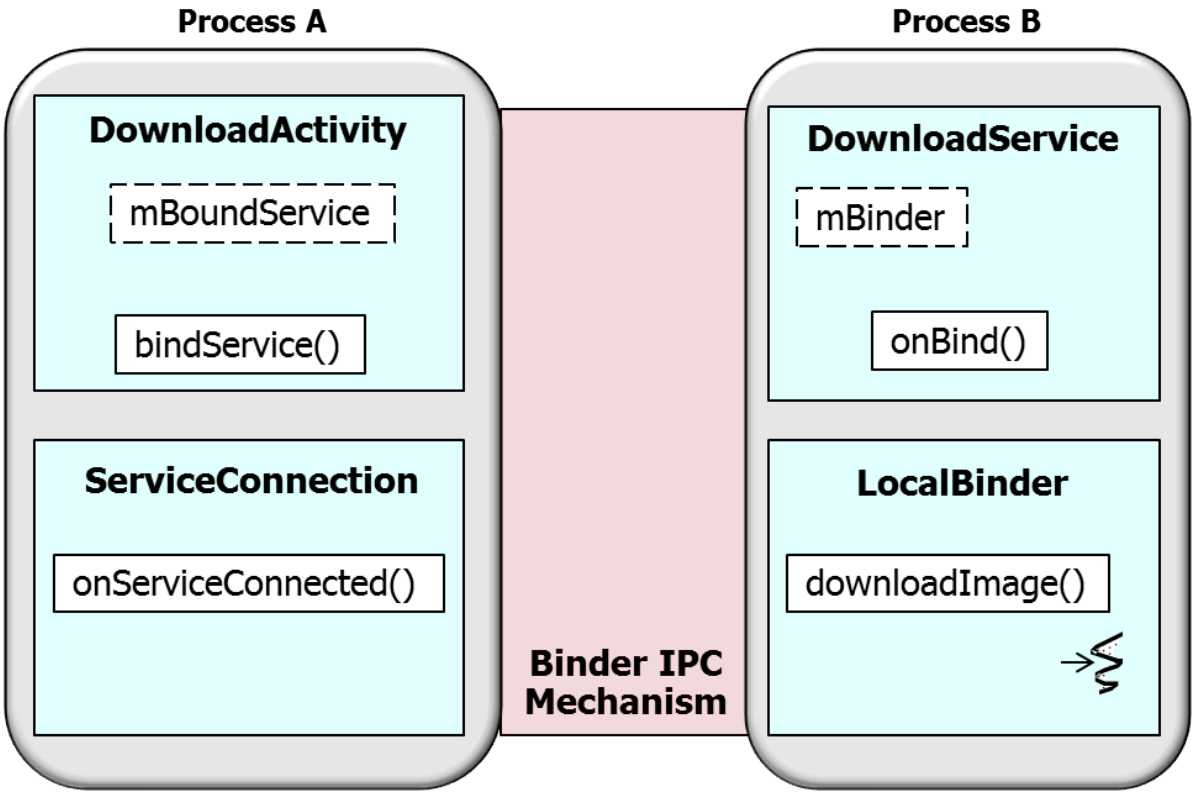
- A Service can perform long-duration operations in the background



developer.android.com/guide/components/services.html has more on Services

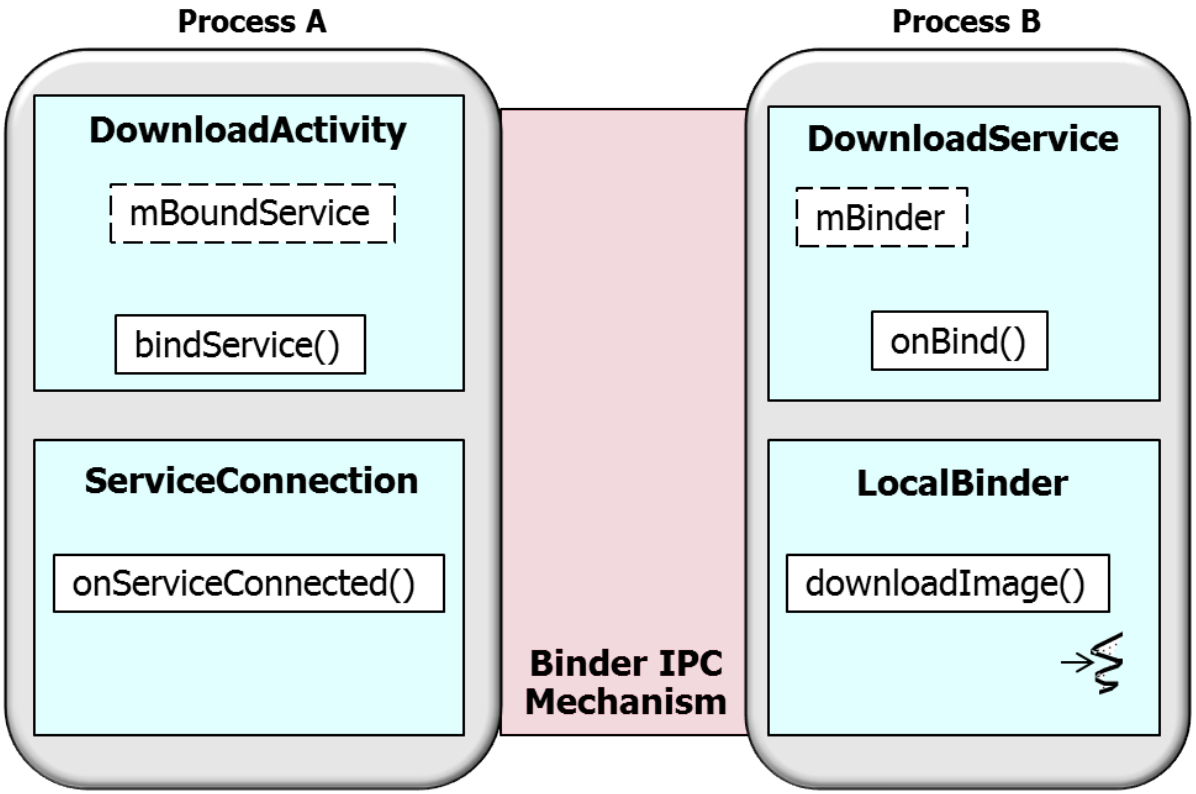
Overview of a Service

- A Service can perform long-duration operations in the background , e.g.
  - Perform e-commerce transactions



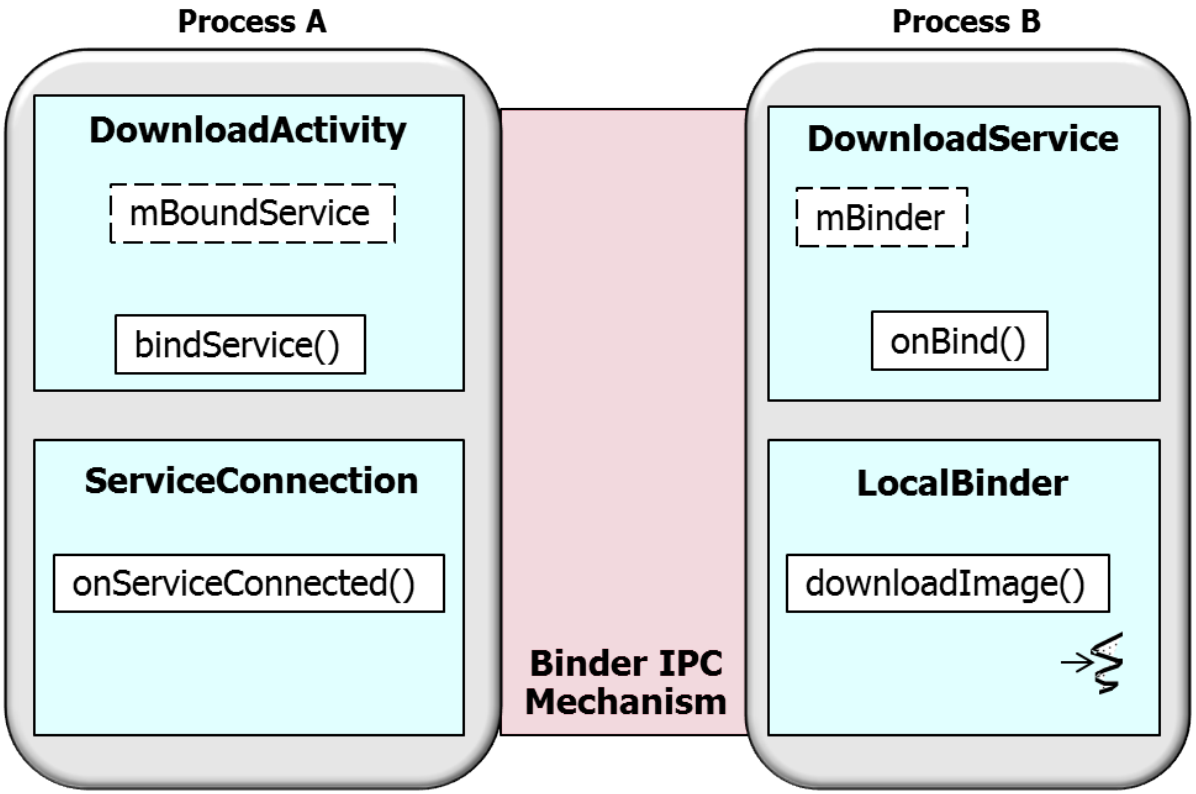
Overview of a Service

- A Service can perform long-duration operations in the background , e.g.
  - Perform e-commerce transactions
  - Play music or videos on the device



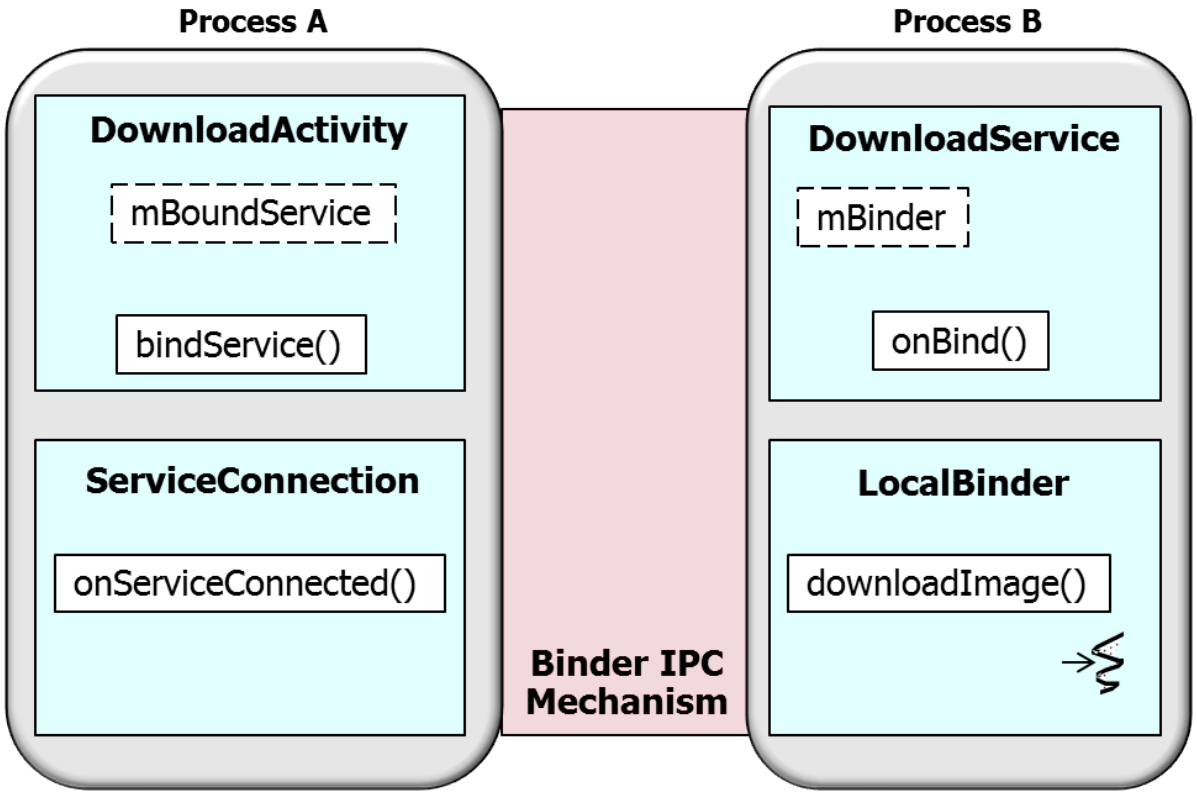
Overview of a Service

- A Service can perform long-duration operations in the background , e.g.
  - Perform e-commerce transactions
  - Play music or videos on the device
  - Synchronize contents of SQLite database with the cloud



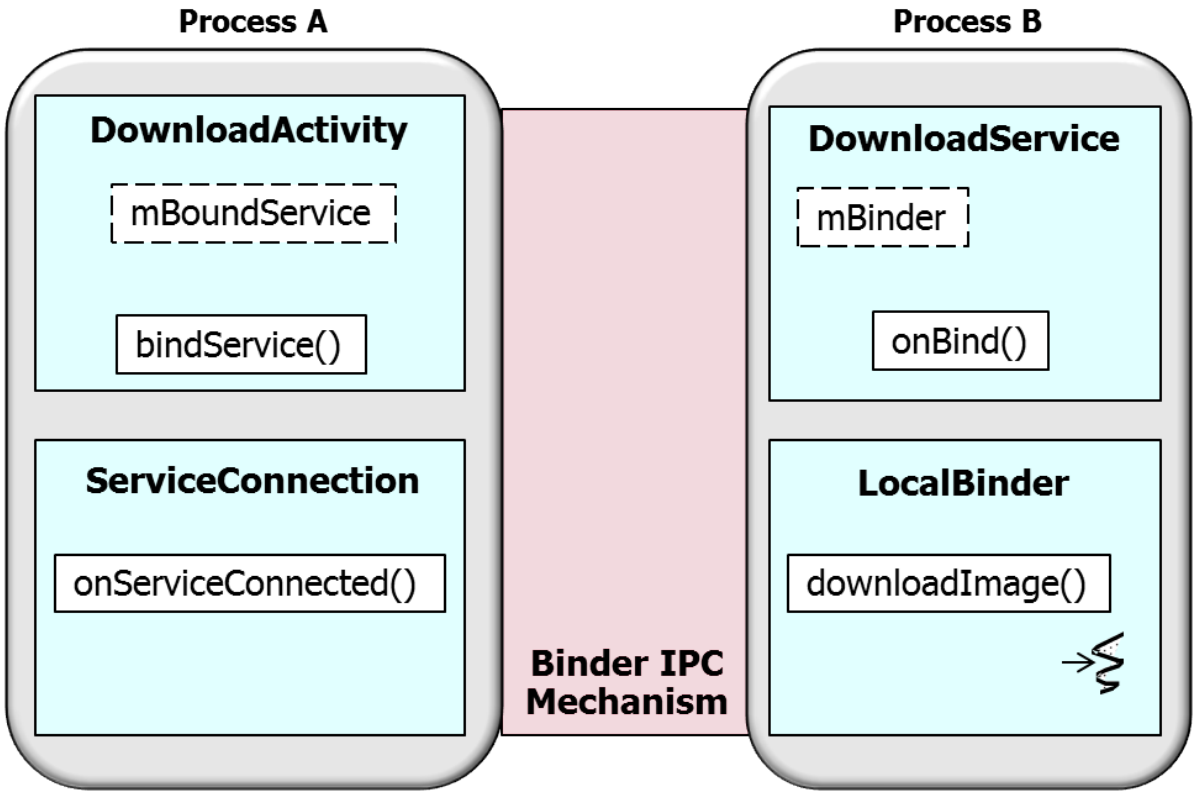
Overview of a Service

- A Service can perform long-duration operations in the background , e.g.
  - Perform e-commerce transactions
  - Play music or videos on the device
  - Synchronize contents of SQLite database with the cloud
  - Download a file



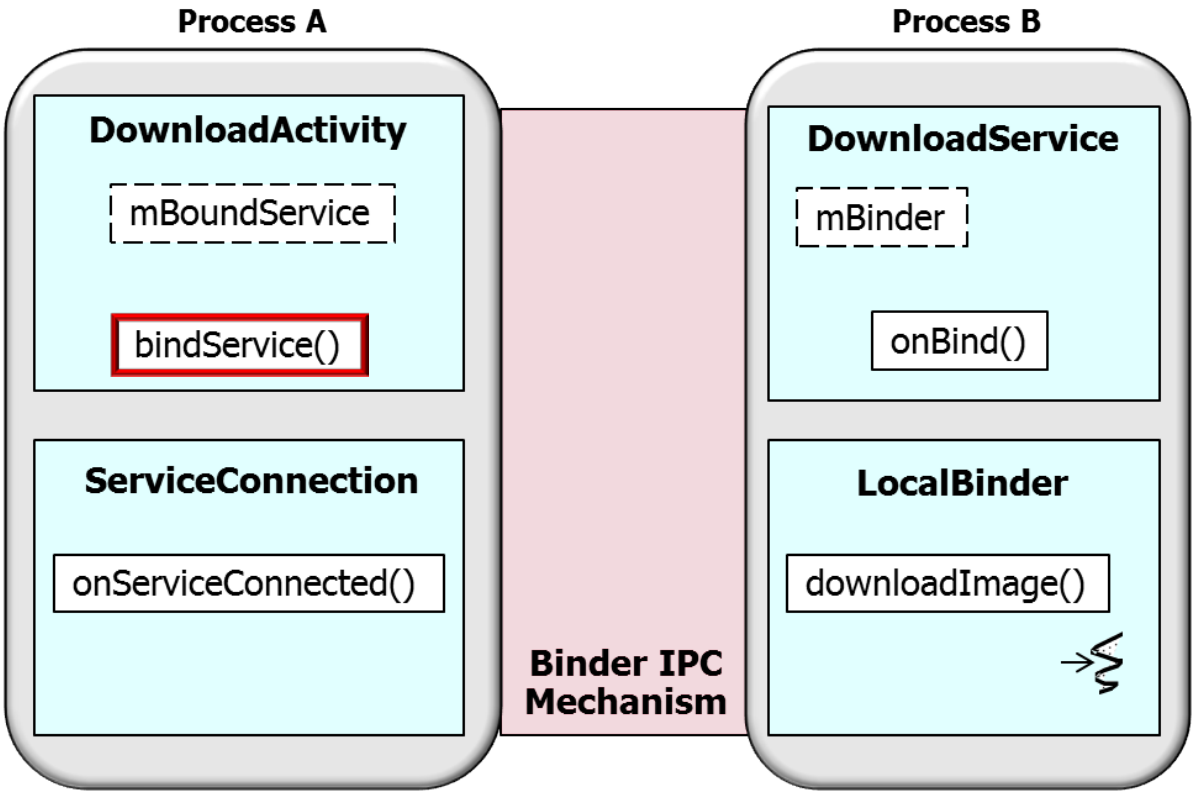
Overview of a Service

- A Service can perform long-duration operations in the background , e.g.
  - Perform e-commerce transactions
  - Play music or videos on the device
  - Synchronize contents of SQLite database with the cloud
  - Download a file



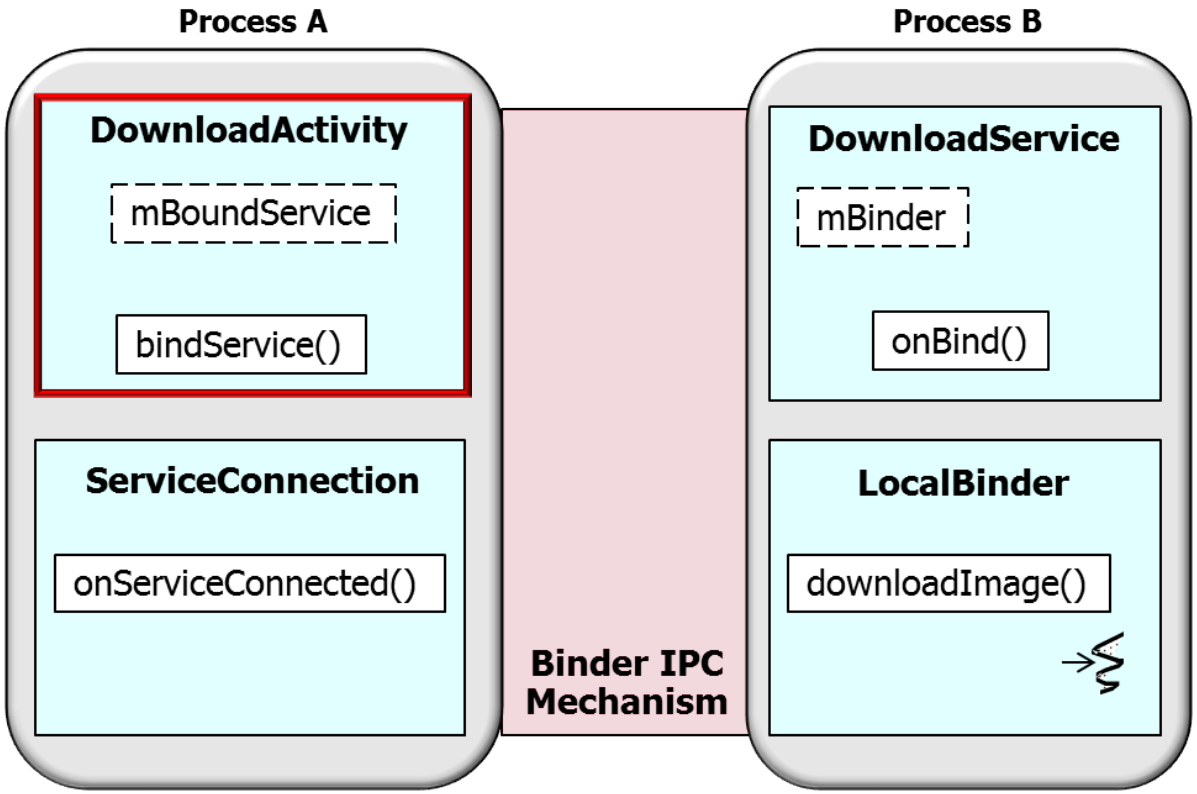
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service



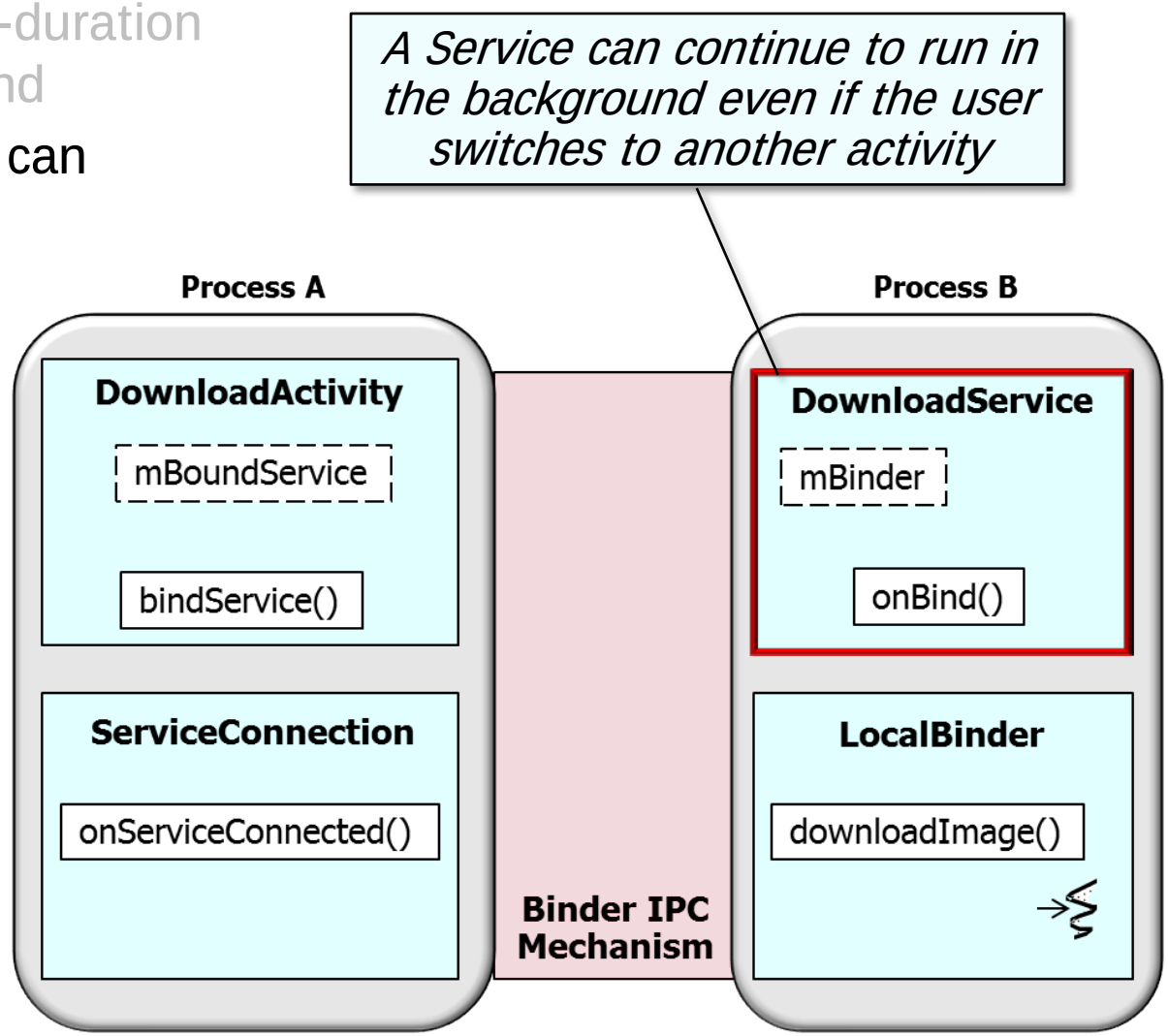
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service



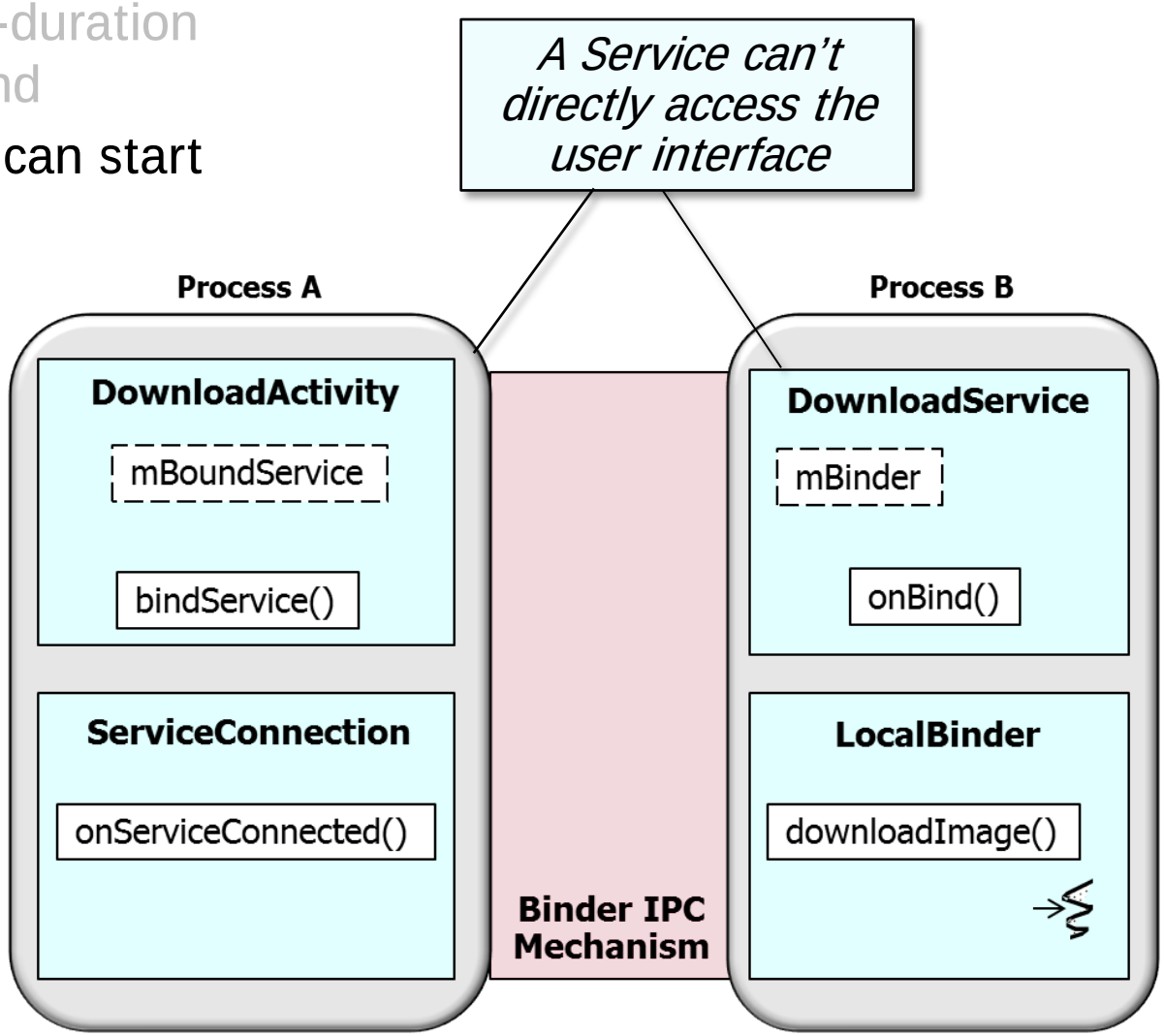
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service



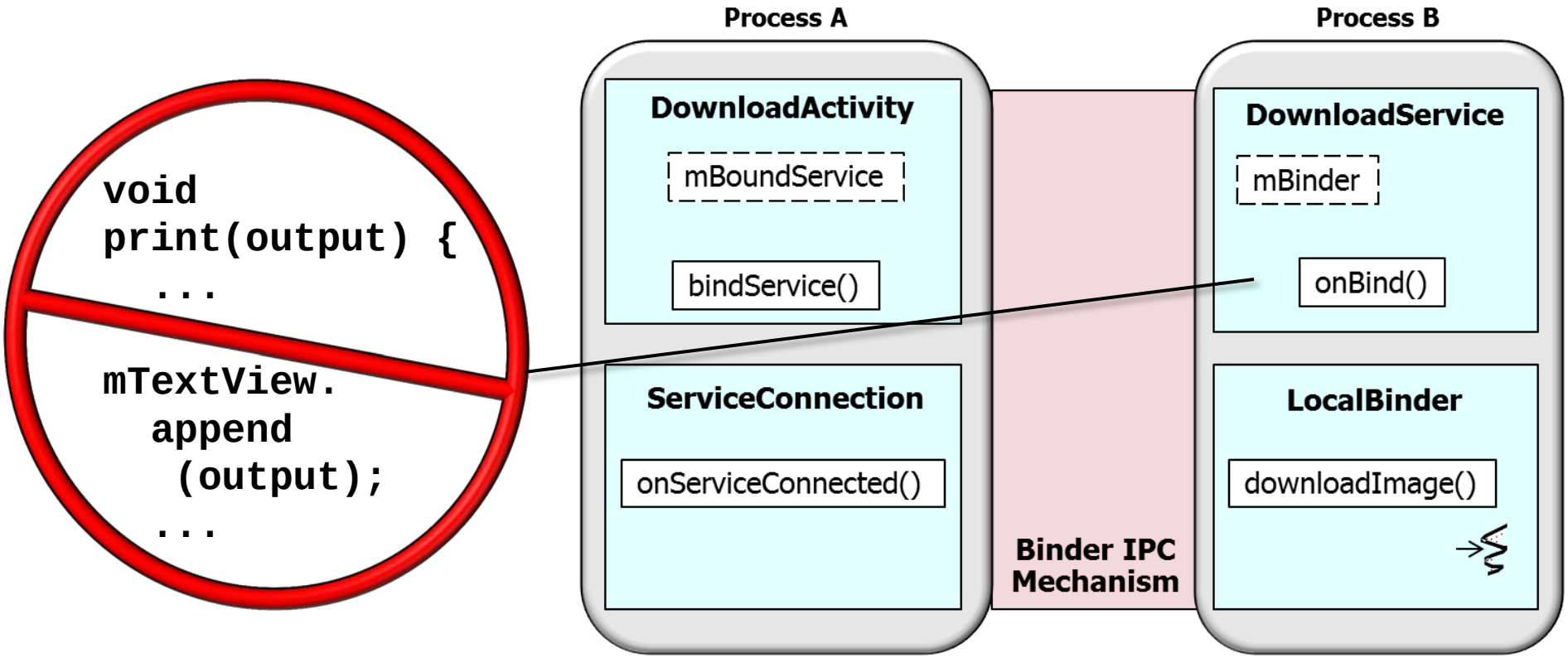
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service



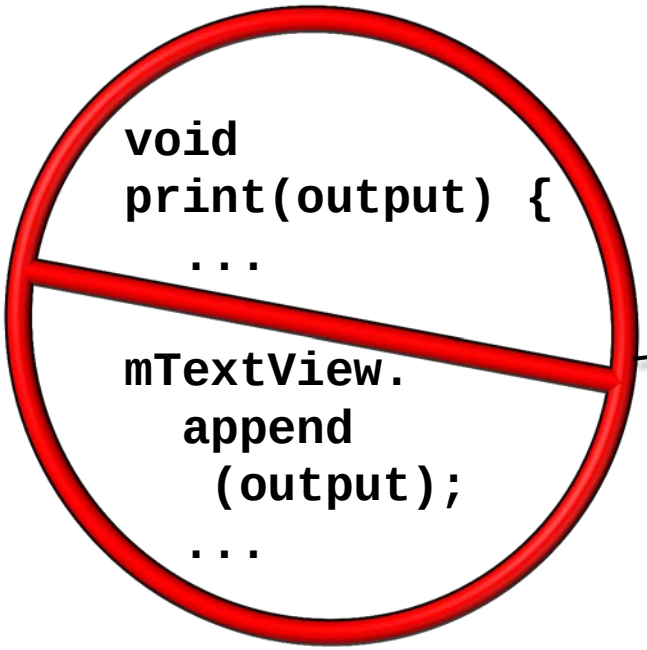
Overview of a Service

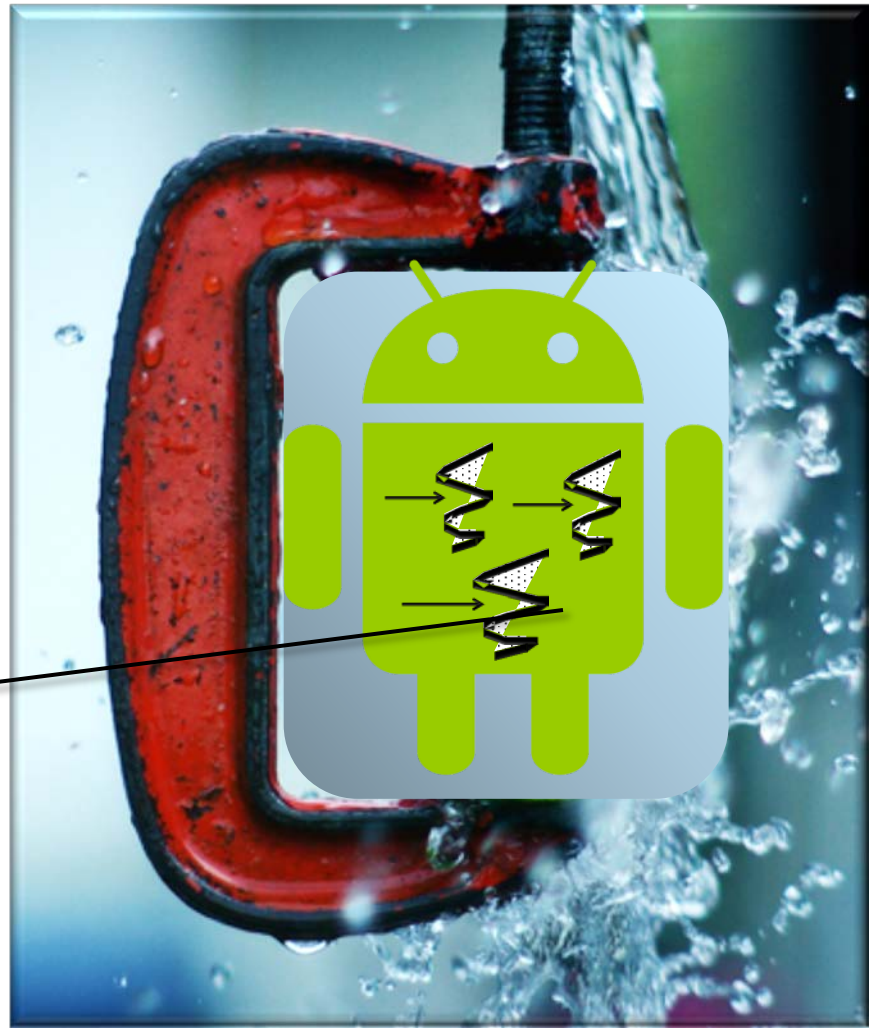
- A Service can perform long-duration operations in the background
- Other Android components can start a Service



Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service

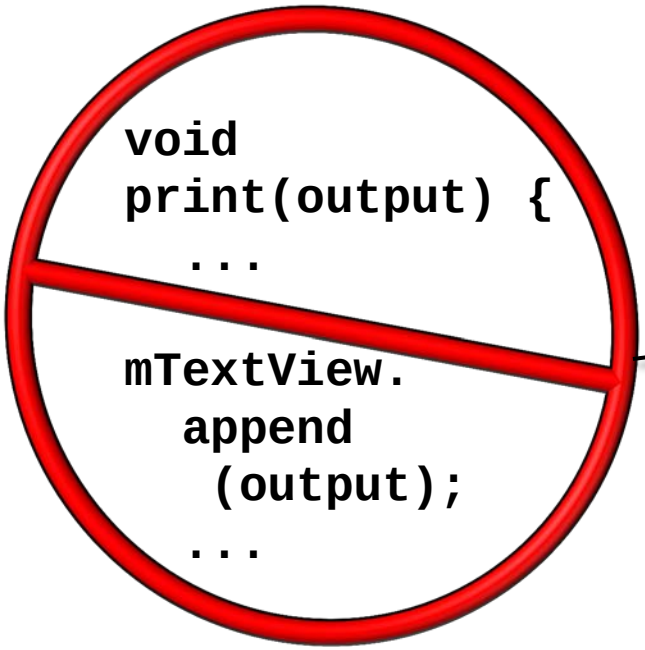


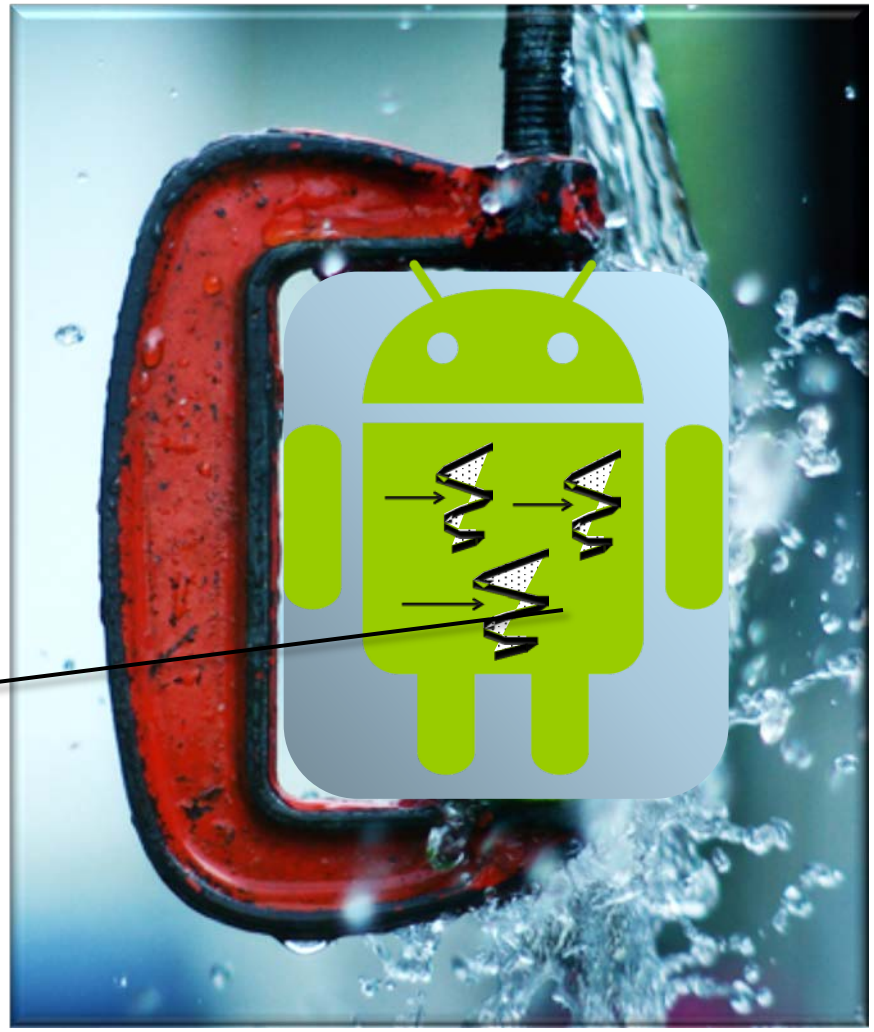


See earlier module on “Android Concurrency Frameworks”

Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service

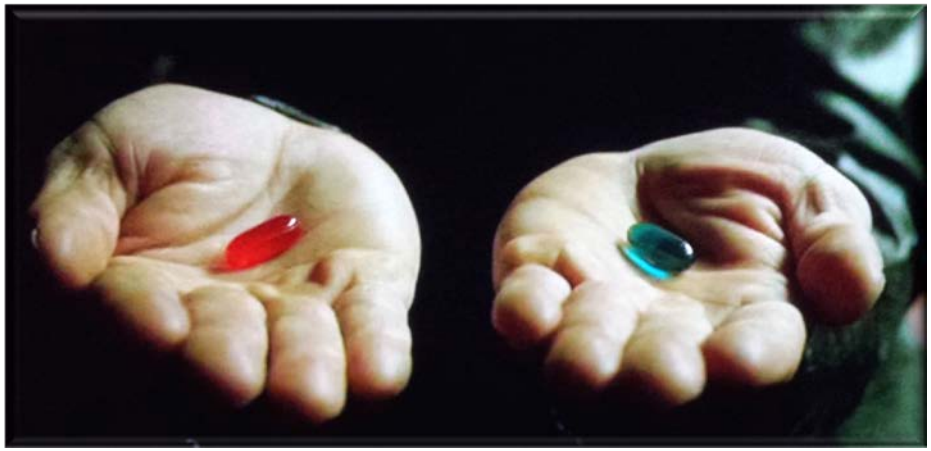


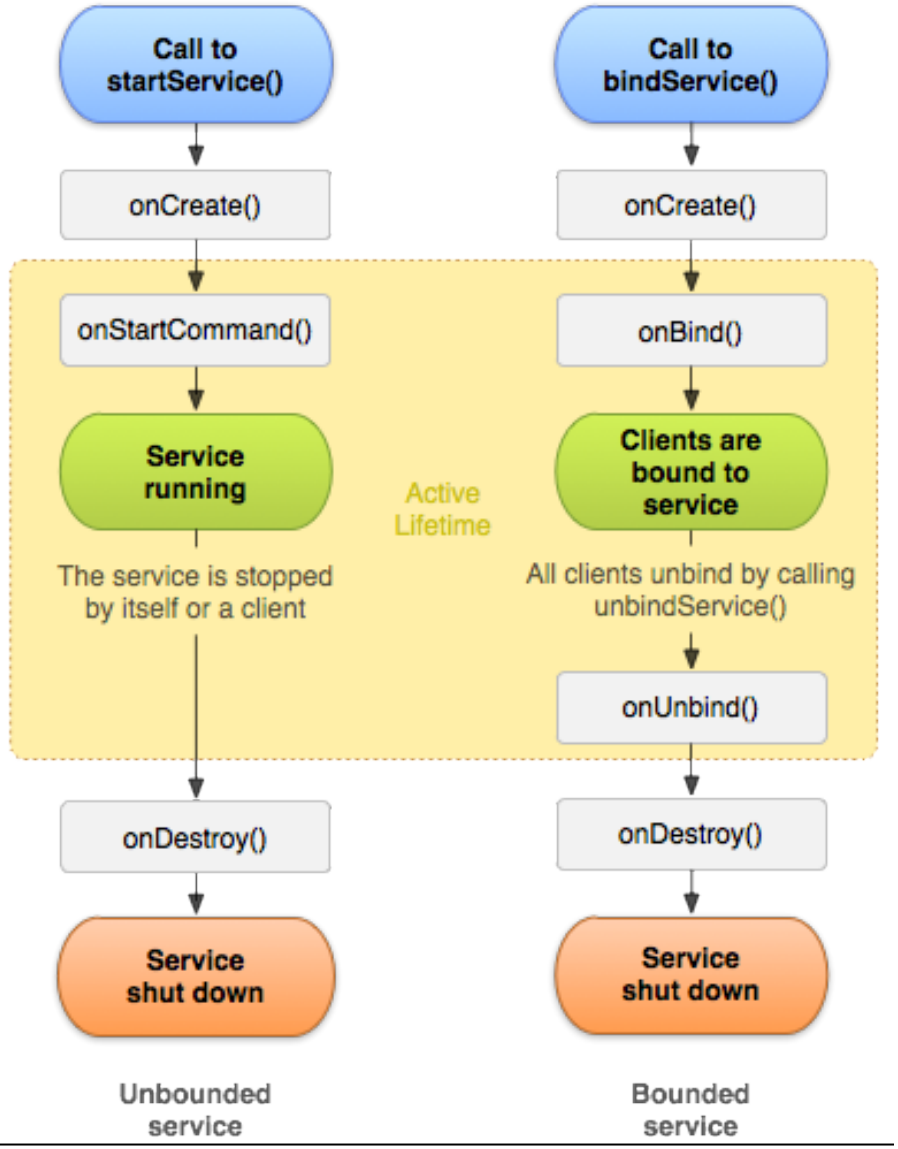


A Service may or may not actually run in a background Thread

Overview of a Service

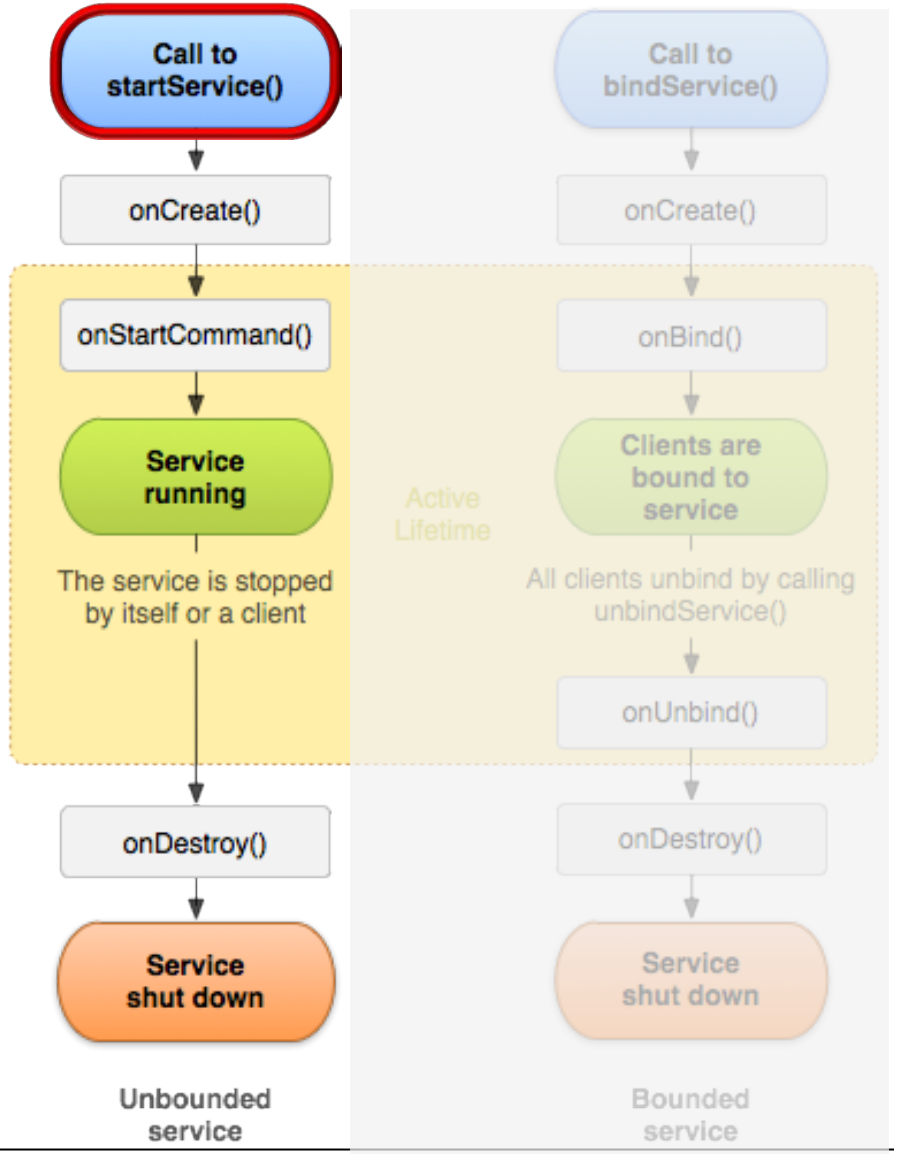
- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types





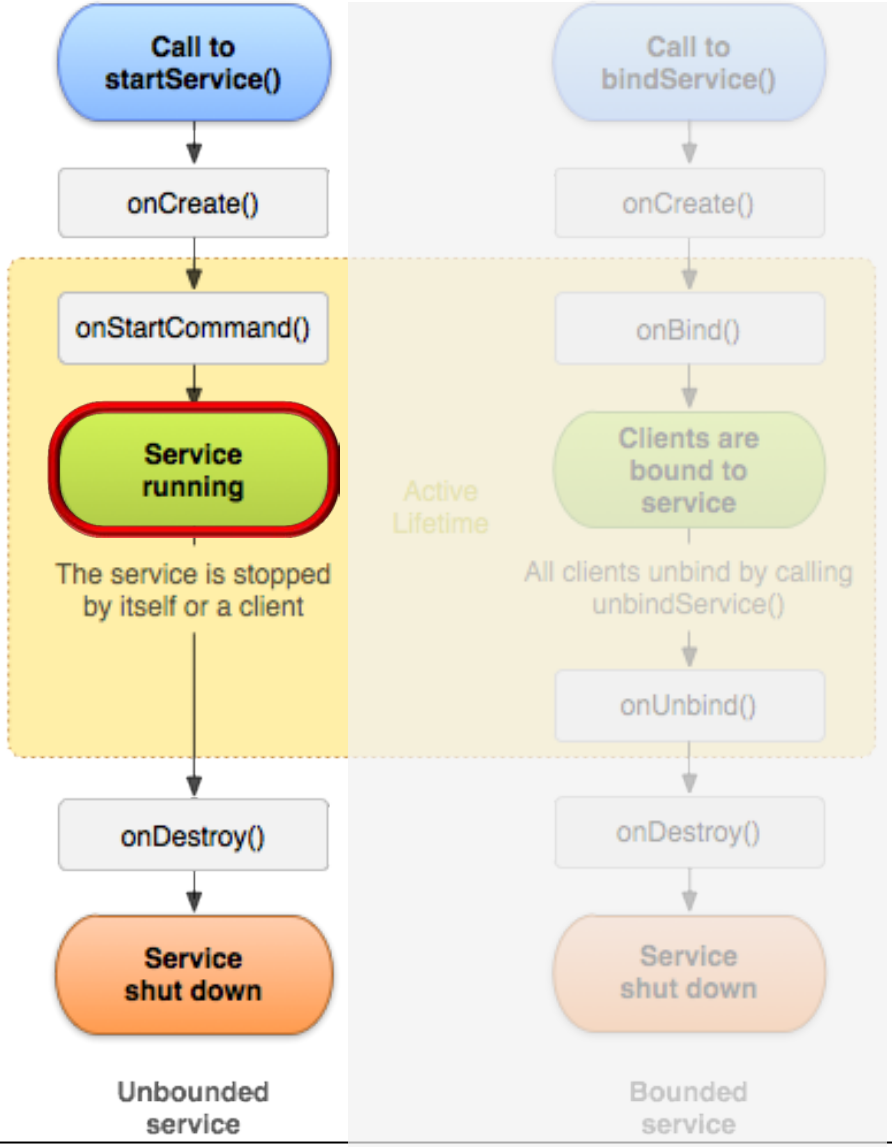
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
  - *Started Service*
    - Launched via startService()



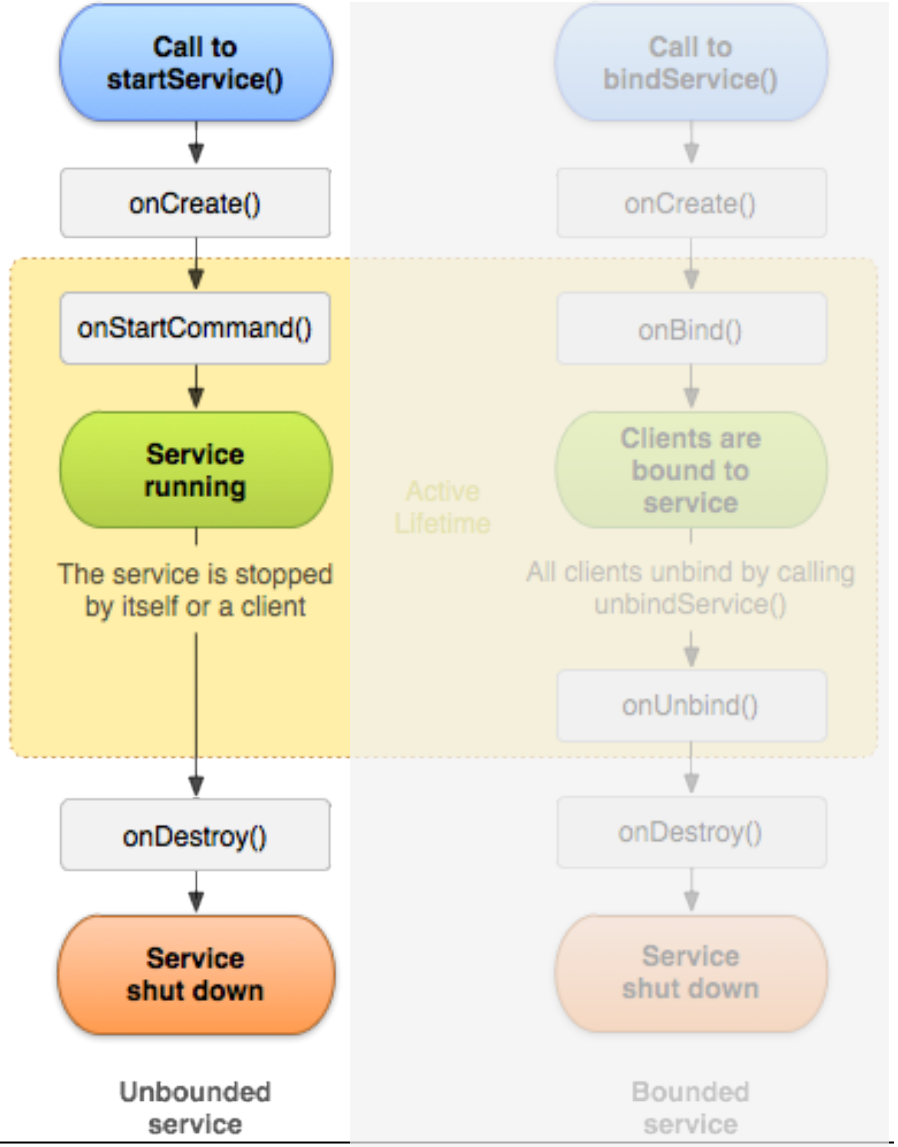
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
  - *Started Service*
    - Launched via startService()
    - Performs a single operation



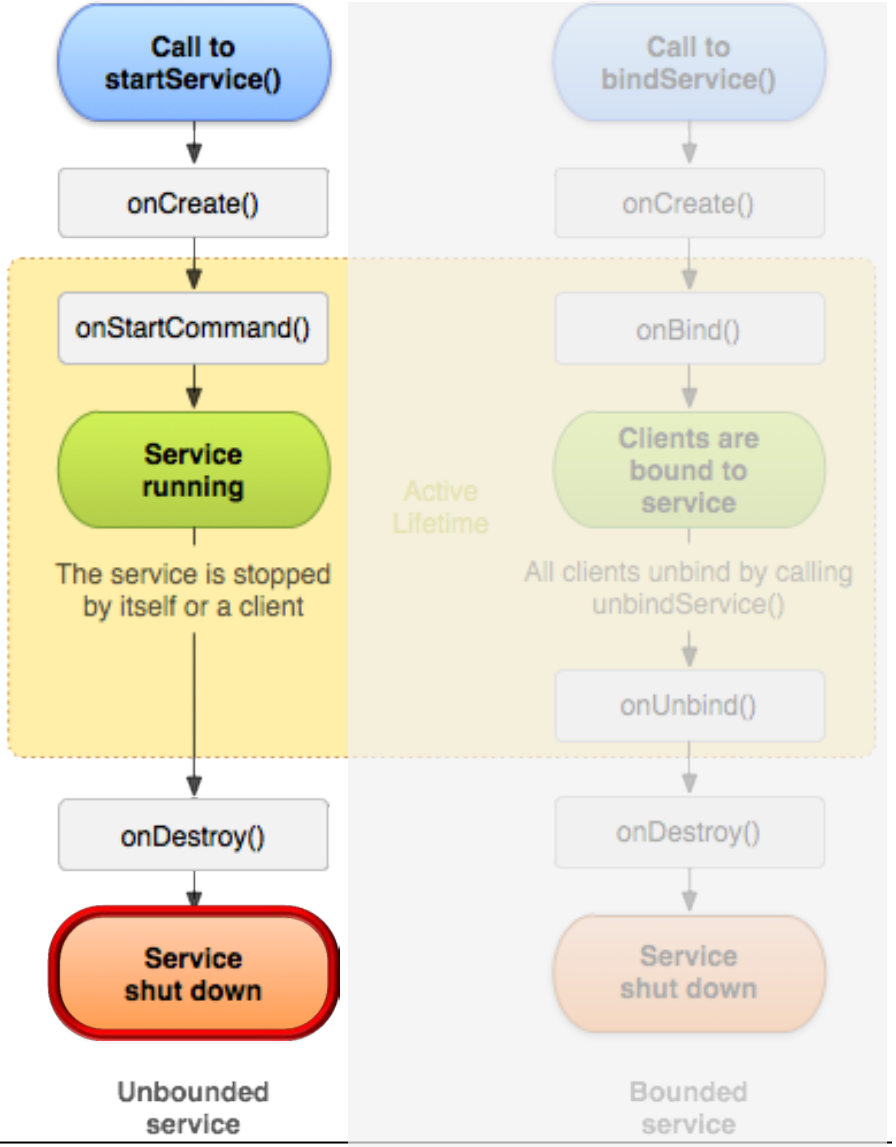
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
  - *Started Service*
    - Launched via startService()
    - Performs a single operation
    - Often doesn't return a result to the client directly



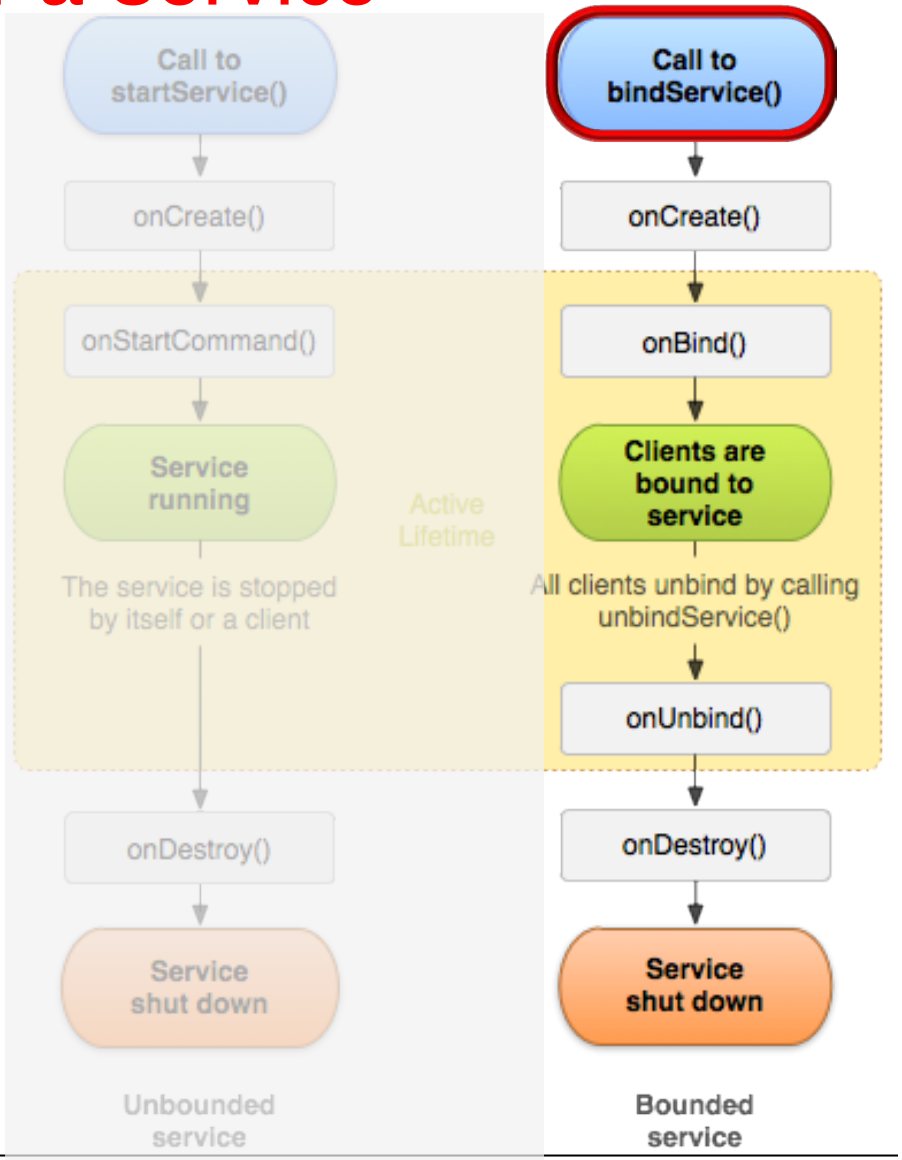
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
  - *Started Service*
    - Launched via startService()
    - Performs a single operation
    - Often doesn't return a result to the client directly
    - Should shut itself down when it's done



Overview of a Service

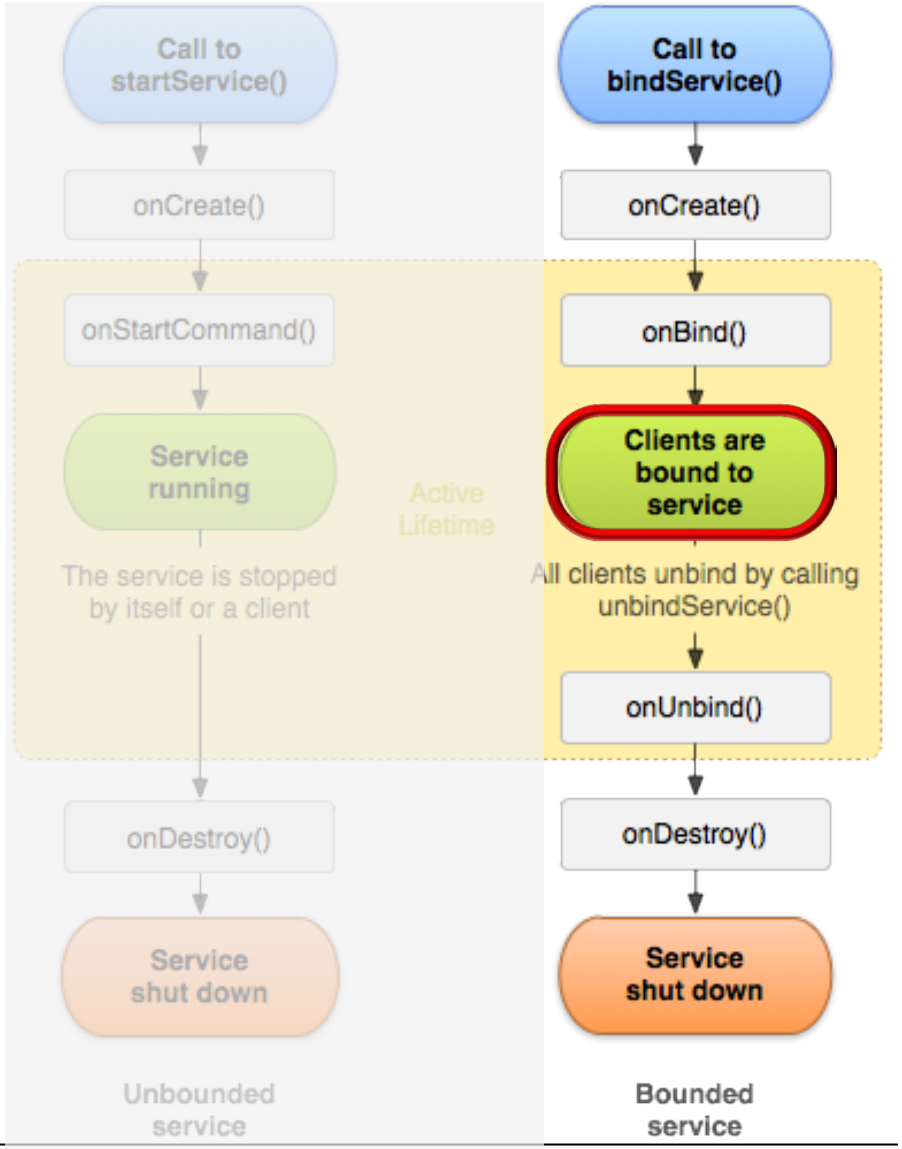
- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
  - *Started Service*
  - *Bound Service*
    - Launched via bindService()



developer.android.com/guide/components/bound-services.html has more

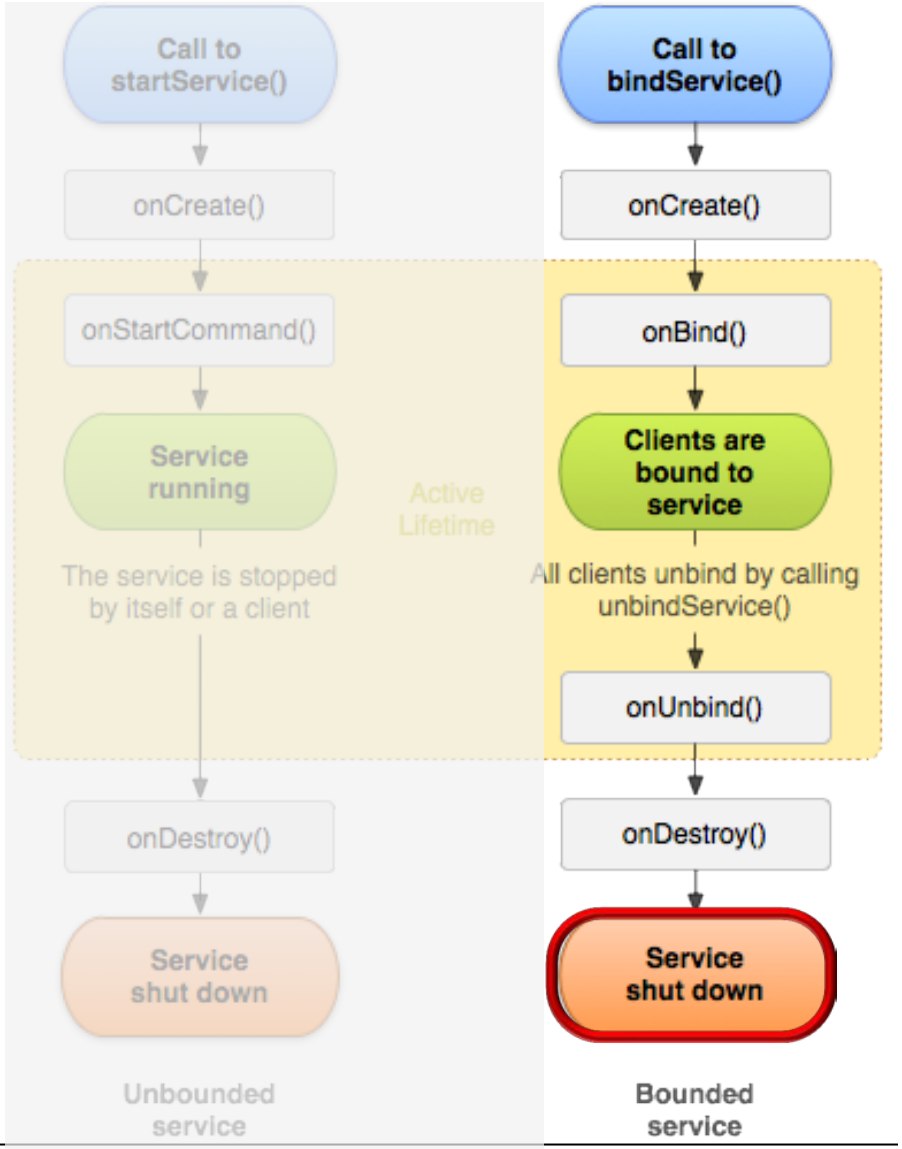
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
  - *Started Service*
  - *Bound Service*
    - Launched via bindService()
      - Offers a client-service interface that allows extended two-way conversations between client(s) & the Service



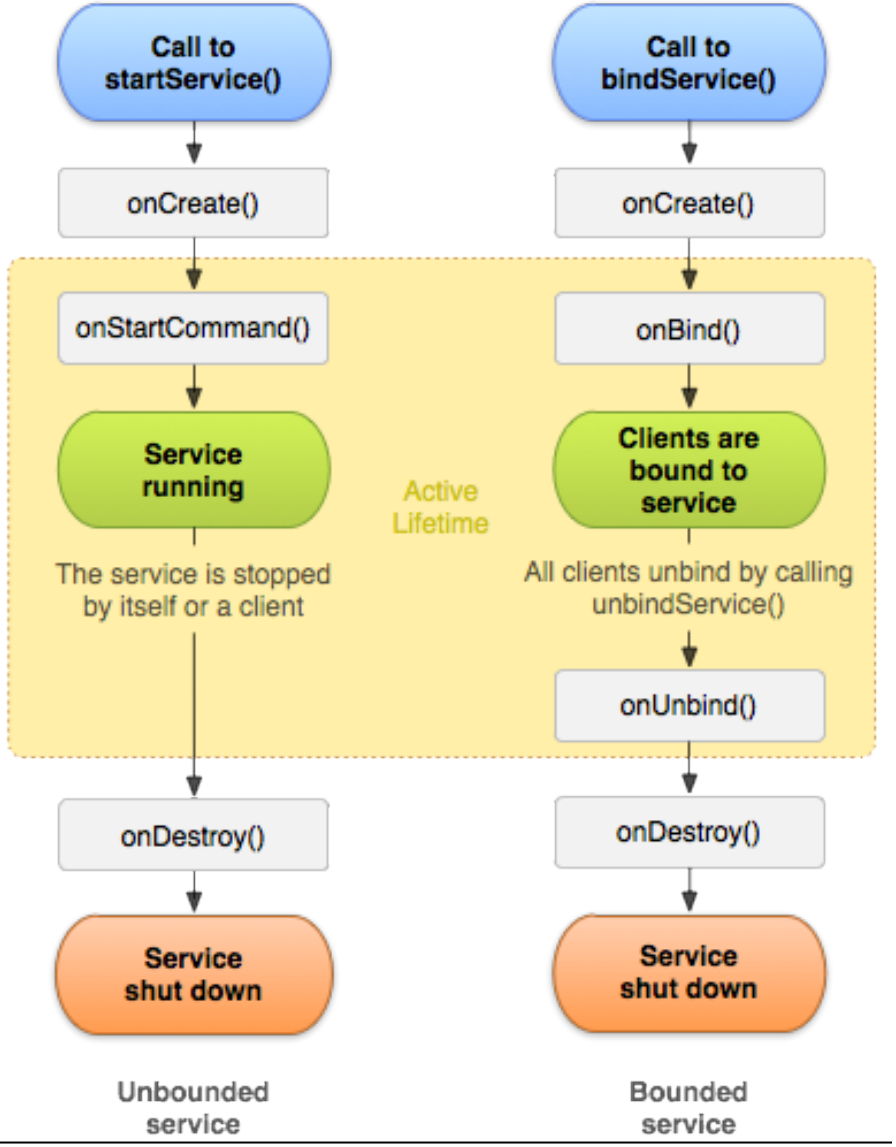
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
  - *Started Service*
  - *Bound Service*
    - Launched via bindService()
    - Offers a client-service interface that allows extended two-way conversations between client(s) & the Service
    - Is automatically destroyed when all clients unbind



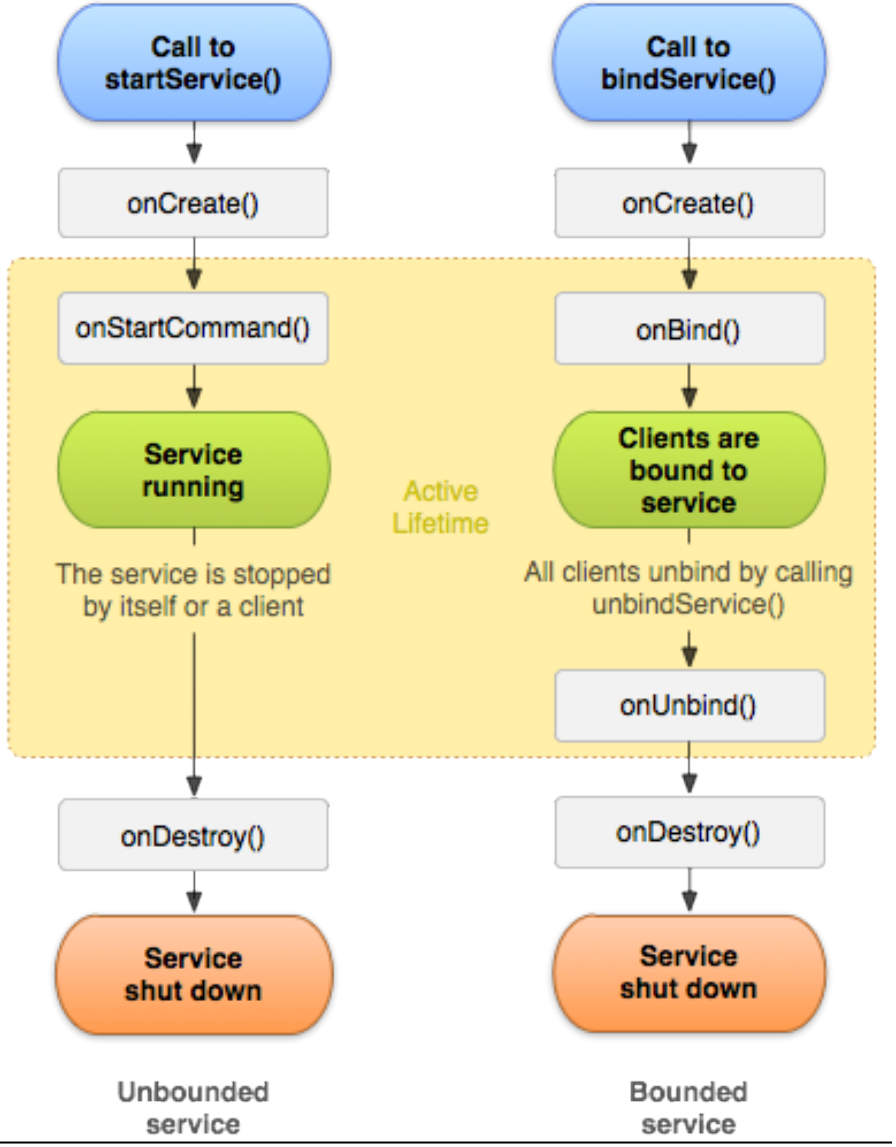
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
- Started & Bound Services can run in the same or different processes as their clients



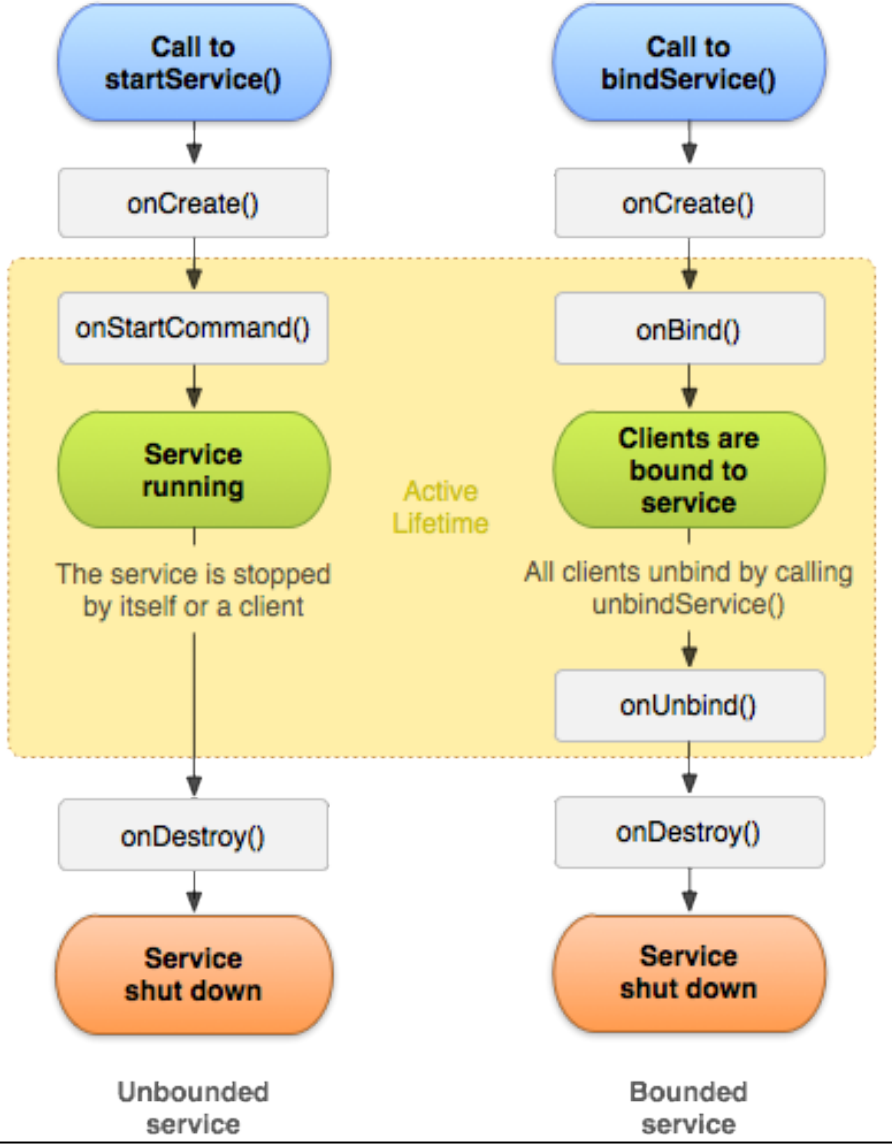
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
- Started & Bound Services can run in the same or different processes as their clients
  - This choice is determined via a configuration setting in AndroidManifest.xml



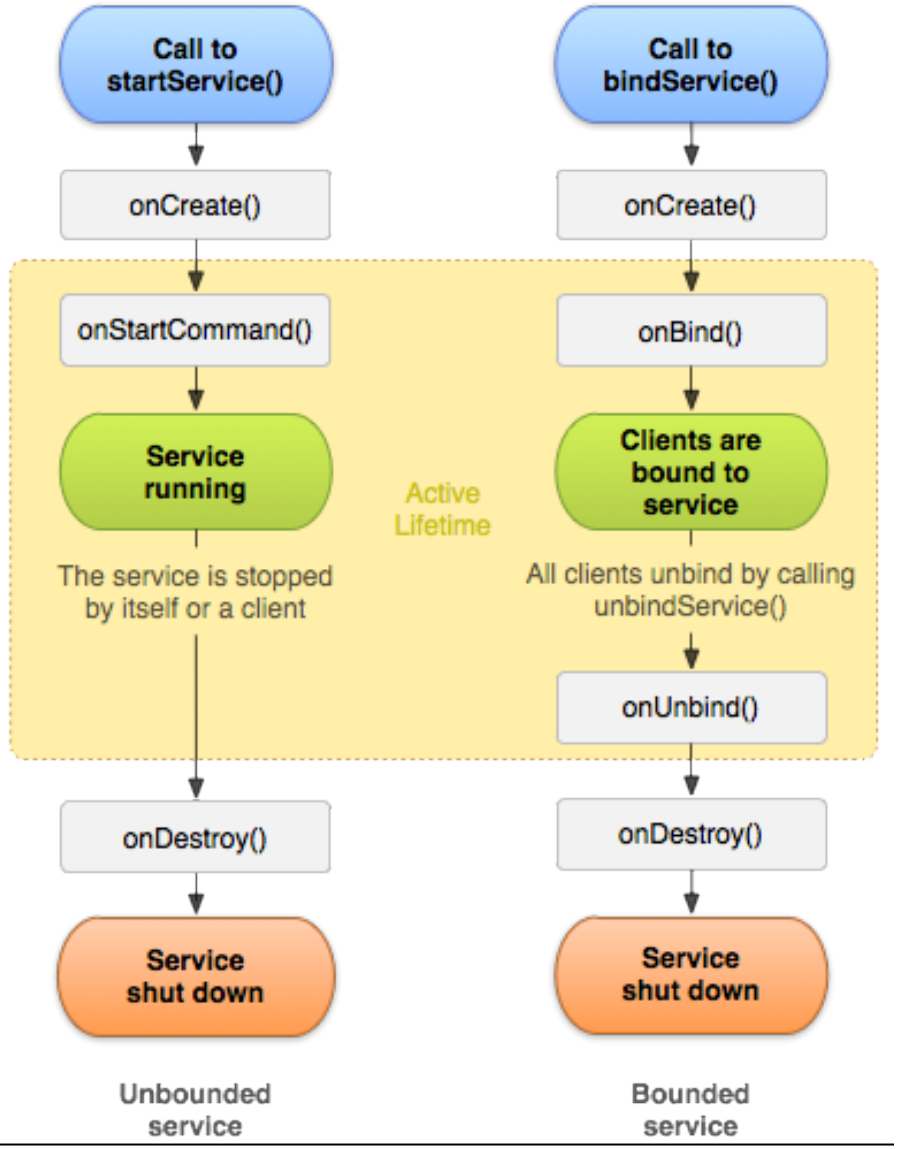
Overview of a Service

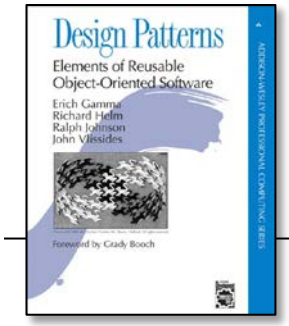
- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
- Started & Bound Services can run in the same or different processes as their clients
  - This choice is determined via a configuration setting in AndroidManifest.xml
  - IPC mechanisms are needed to communicate with Services running in different processes

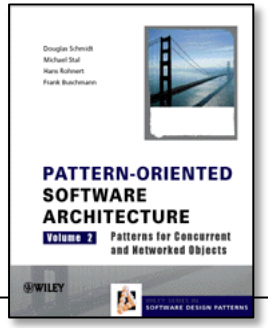


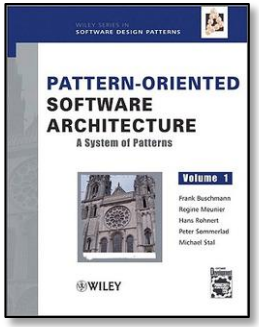
Overview of a Service

- A Service can perform long-duration operations in the background
- Other Android components can start a Service
- Android supports two Service types
- Started & Bound Services can run in the same or different processes as their clients
- Many POSA & GoF patterns are applied to implement Services & guide IPC between Activities & Services



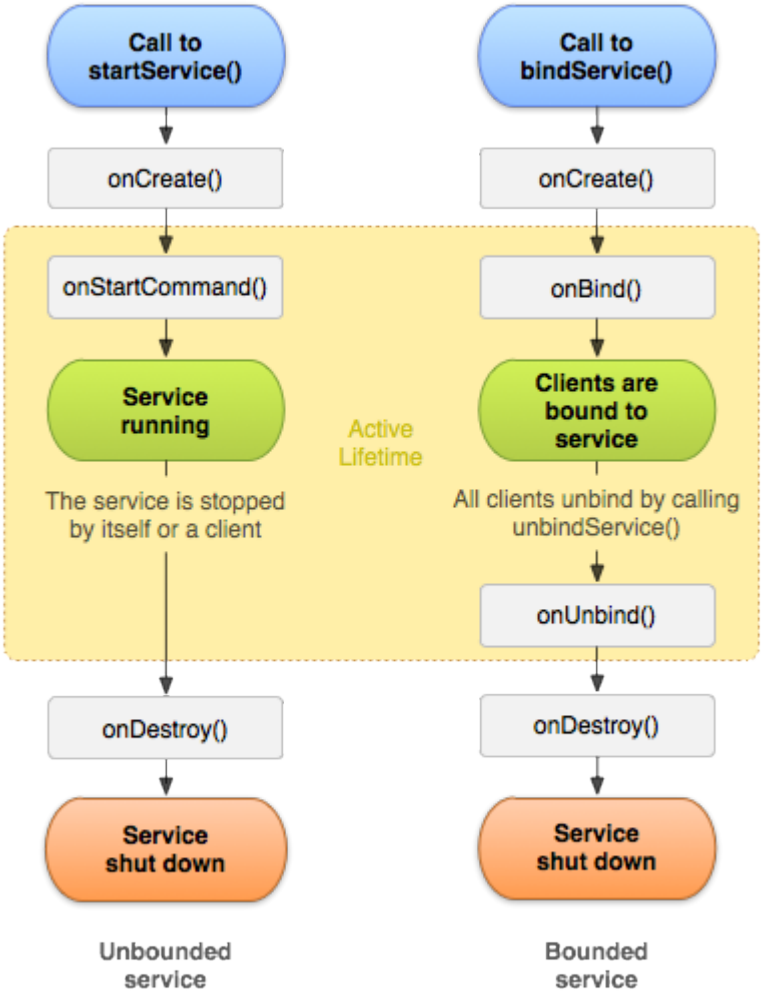






Steps for Implementing a Service

Steps for Implementing a Service



frameworks/base/services/java/com/android/server/am has source code

Steps for Implementing a Service

- Implementing a Service is similar to implementing an Activity

```
public class DownloadService
       extends Service {
  public void onCreate() { ... }
  public int onStartCommand
     (Intent intent,
      int flags, int startId) {
    ...
  }
  public abstract IBinder
    onBind(Intent intent) { ... }
  public boolean
    onUnbind(Intent intent) { ... }
  protected void onDestroy() {
    ...
  }
  ...
}
```

Steps for Implementing a Service

- Implementing a Service is similar to implementing an Activity

Services & Activities are both programmed via canonical framework techniques

```
public class DownloadService
       extends Service {
  public void onCreate() { ... }
  public int onStartCommand
     (Intent intent,
      int flags, int startId) {
    ...
  }
  public abstract IBinder
    onBind(Intent intent) { ... }
  public boolean
    onUnbind(Intent intent) { ... }
  protected void onDestroy() {
    ...
  }
  ...
}
```

Steps for Implementing a Service

- Implementing a Service is similar to implementing an Activity, e.g.
  - Extend the Android Service class

```
public class DownloadService
       extends Service {
  public void onCreate() { ... }
  public int onStartCommand
     (Intent intent,
      int flags, int startId) {
    ...
  }
  public abstract IBinder
    onBind(Intent intent) { ... }
  public boolean
    onUnbind(Intent intent) { ... }
  protected void onDestroy() {
    ...
  }
  ...
}
```

Steps for Implementing a Service

- Implementing a Service is similar to implementing an Activity, e.g.
  - Extend the Android Service class
    - Defines Service-specific lifecycle hook methods

Android's Service framework dispatches these hook methods via "inversion of control"

```
public class DownloadService
       extends Service {
  public void onCreate() { ... }
  public int onStartCommand
     (Intent intent,
      int flags, int startId) {
    ...
  }
  public abstract IBinder
    onBind(Intent intent) { ... }
  public boolean
    onUnbind(Intent intent) { ... }
  protected void onDestroy() {
    ...
  }
  ...
}
```

Steps for Implementing a Service

- Implementing a Service is similar to implementing an Activity, e.g.
  - Extend the Android Service class
    - Defines Service-specific lifecycle hook methods
  - Selectively override lifecycle hook methods

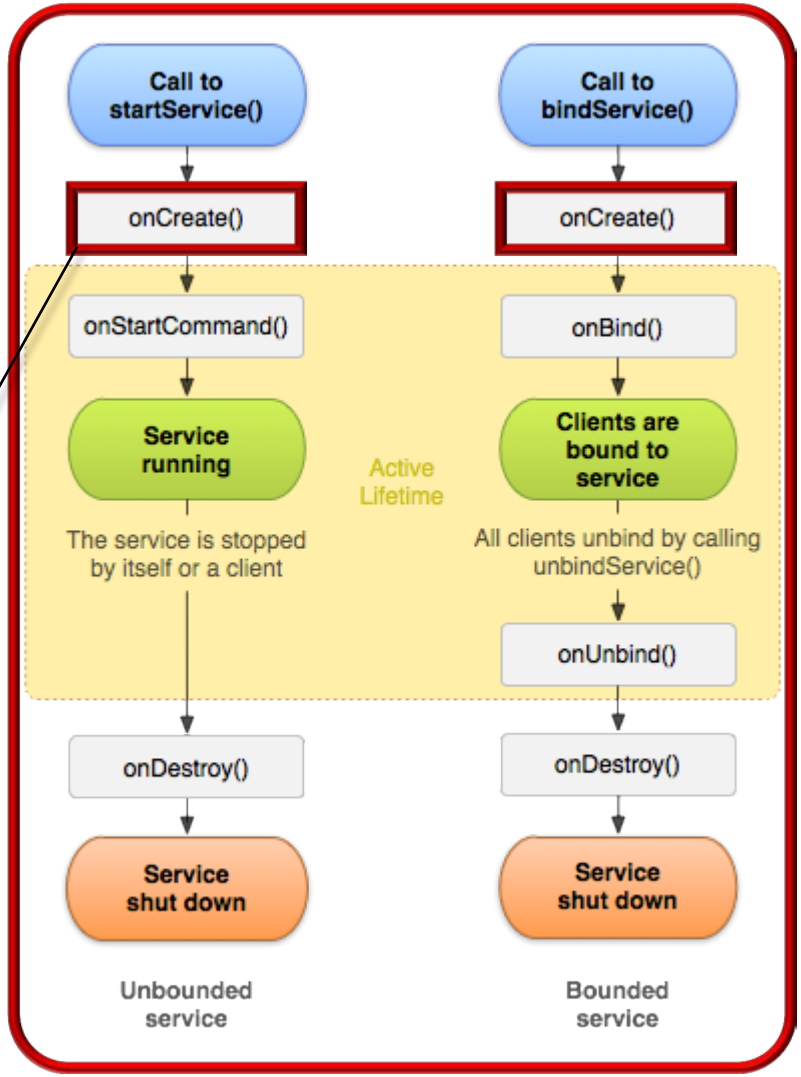
Android's Service framework defines reusable structure & functionality that's specific to different type of Services

```
public class DownloadService
       extends Service {
  public void onCreate() { ... }
  public int onStartCommand
     (Intent intent,
      int flags, int startId) {
    ...
  }
  public abstract IBinder
    onBind(Intent intent) { ... }
  public boolean
    onUnbind(Intent intent) { ... }
  protected void onDestroy() {
    ...
  }
  ...
}
```

Steps for Implementing a Service

- Implementing a Service is similar to implementing an Activity
- Android communicates state changes to a Service by calling its lifecycle hook methods
  - **onCreate()** – called when Service is first launched, by any means

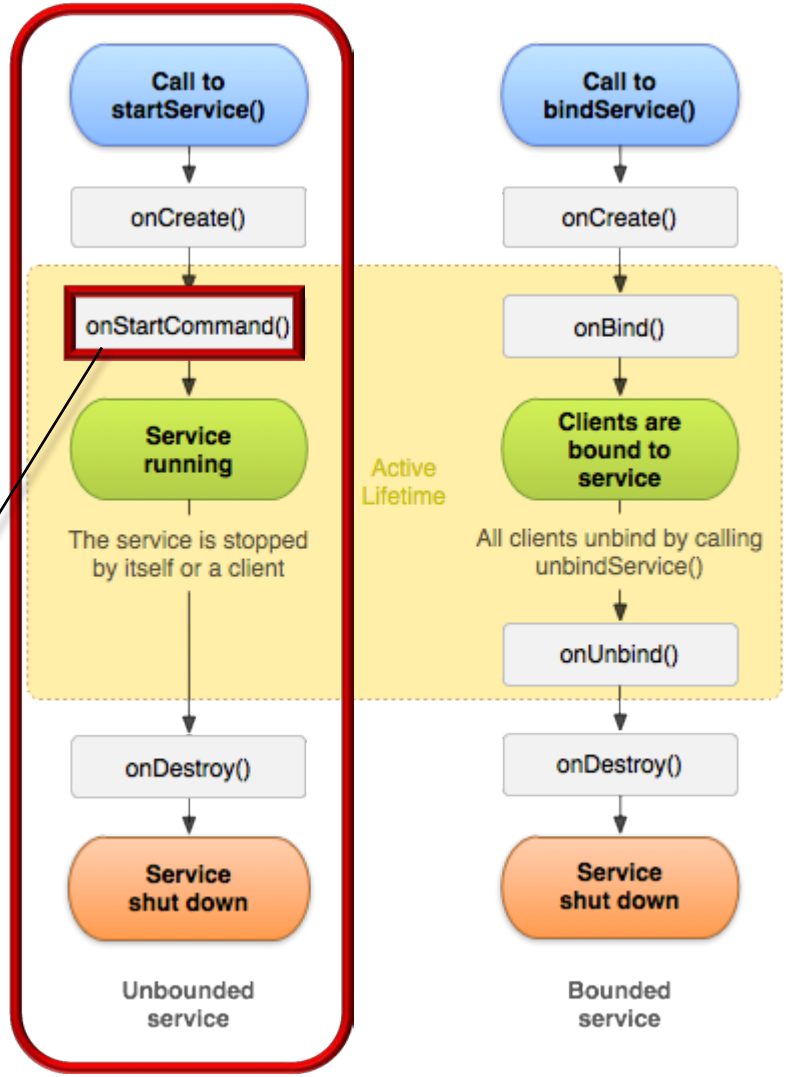
This method is typically used to initialize the Service



Steps for Implementing a Service

- Implementing a Service is similar to implementing an Activity
- Android communicates state changes to a Service by calling its lifecycle hook methods
  - **onCreate()** – called when Service is first launched, by any means
  - **onStartCommand()** – called each time a Started Service is sent an Intent via startService()

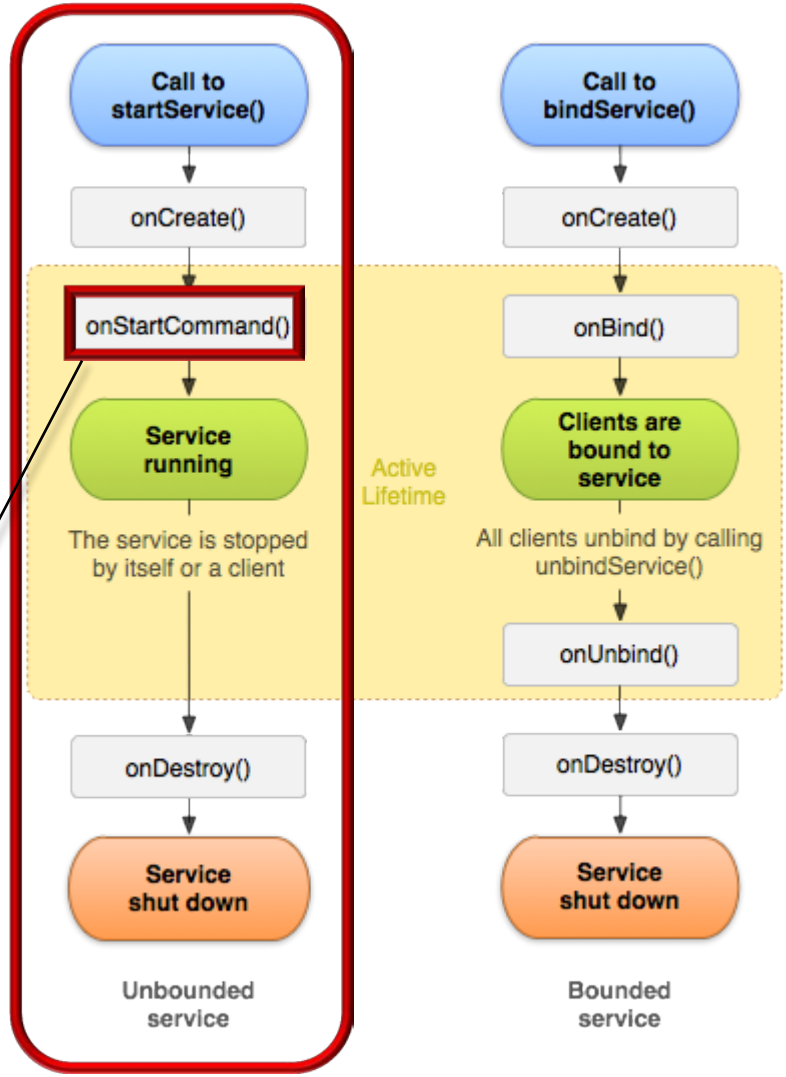
This method receives the Intent passed by the client's call to startService()



Steps for Implementing a Service

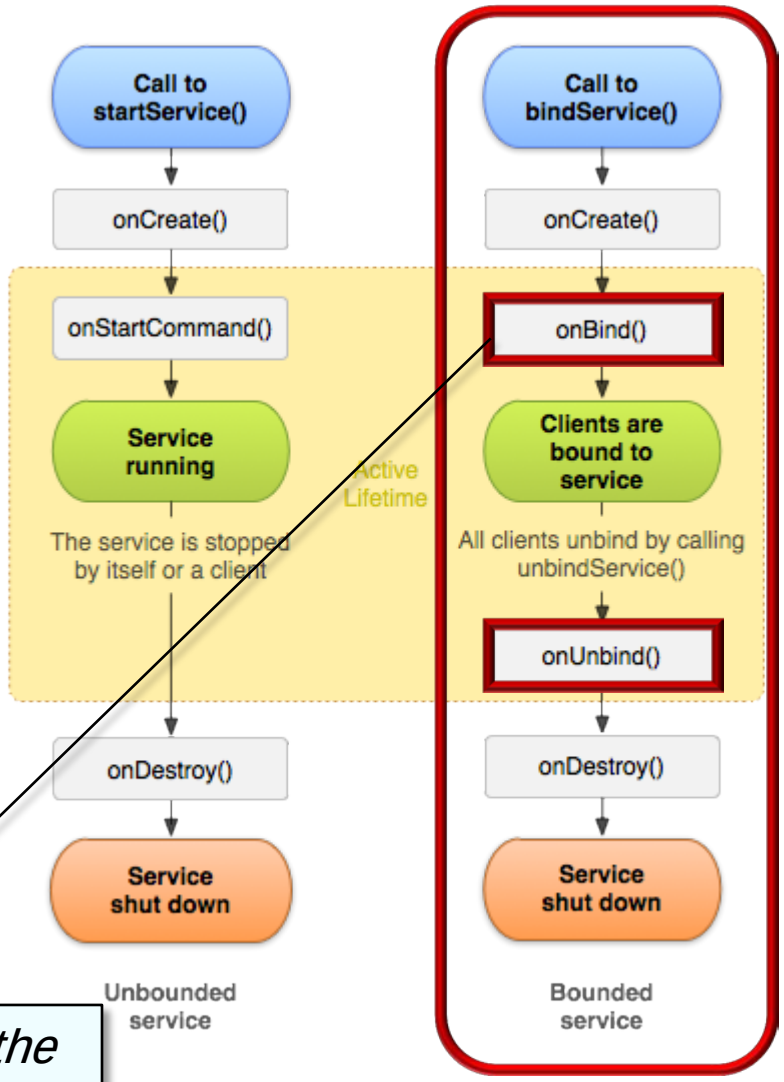
- Implementing a Service is similar to implementing an Activity
- Android communicates state changes to a Service by calling its lifecycle hook methods
  - **onCreate()** – called when Service is first launched, by any means
  - **onStartCommand()** – called each time a Started Service is sent an Intent via startService()

May be used in conjunction with the concurrency model the Service applies to perform its processing



Steps for Implementing a Service

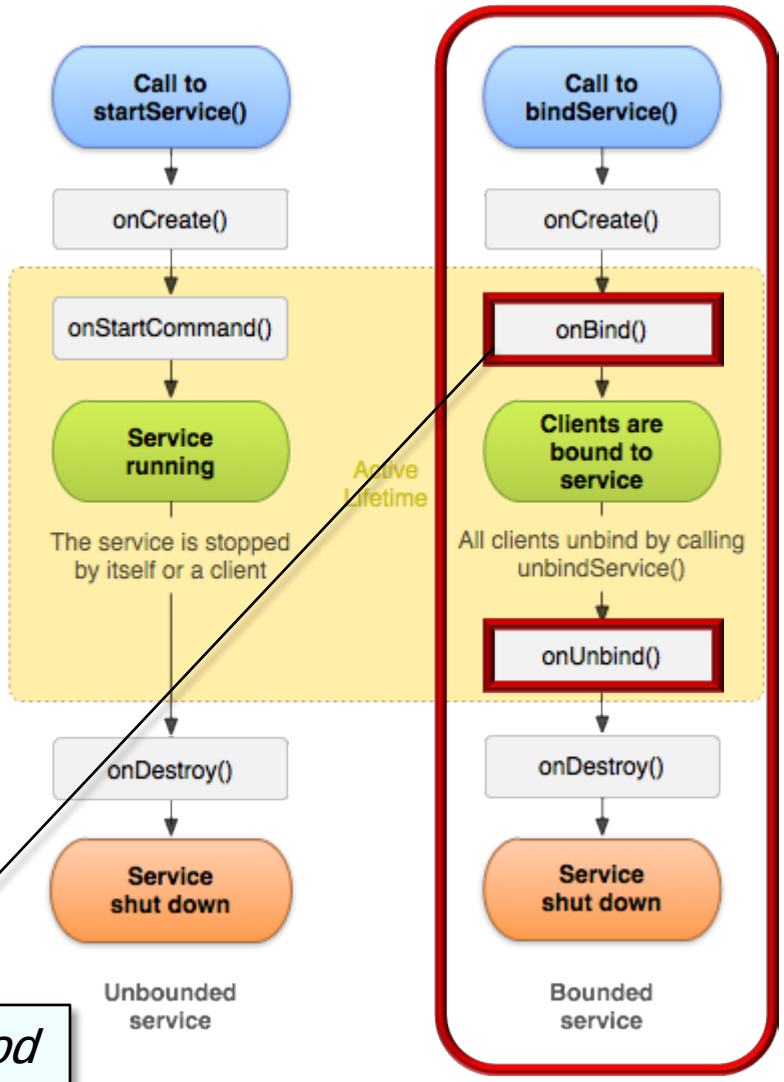
- Implementing a Service is similar to implementing an Activity
- Android communicates state changes to a Service by calling its lifecycle hook methods
  - **onCreate()** – called when Service is first launched, by any means
  - **onStartCommand()** – called each time a Started Service is sent an Intent via startService()
  - **onBind()/onUnbind** – called when client binds/unbinds to Bound Service via bindService()/unBindService()



This method receives the Intent passed by the client's call to bindService()

Steps for Implementing a Service

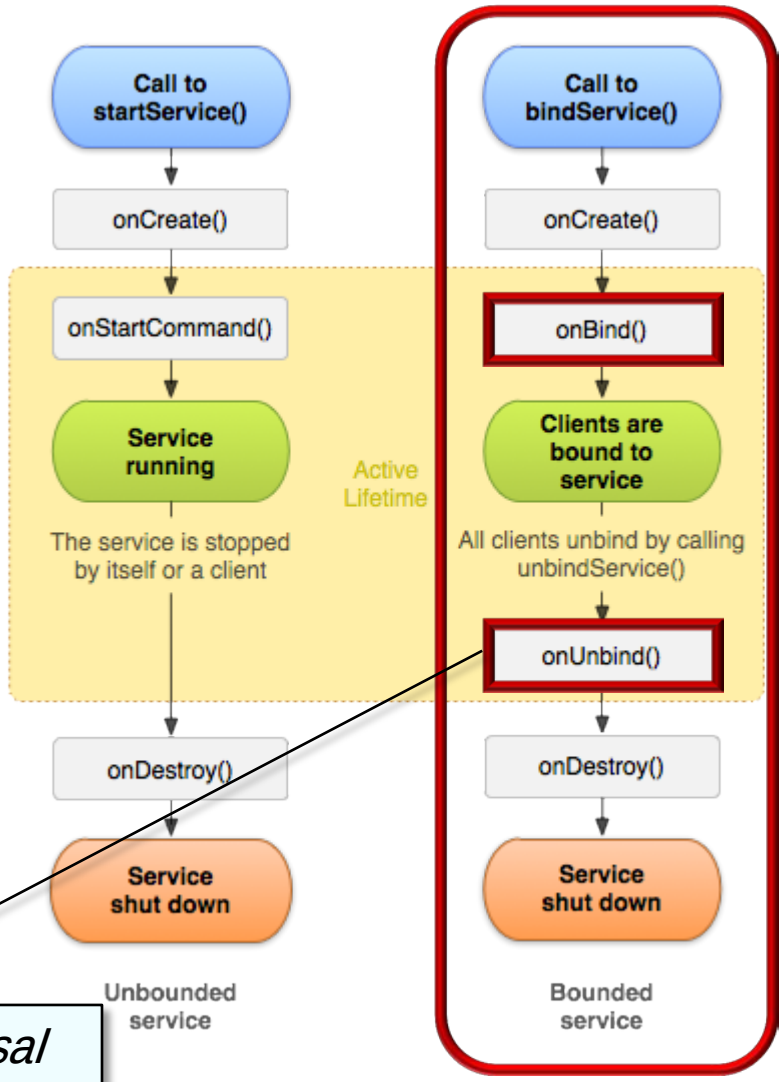
- Implementing a Service is similar to implementing an Activity
- Android communicates state changes to a Service by calling its lifecycle hook methods
  - **onCreate()** – called when Service is first launched, by any means
  - **onStartCommand()** – called each time a Started Service is sent an Intent via startService()
  - **onBind()/onUnbind** – called when client binds/unbinds to Bound Service via bindService()/unBindService()



It is a factory method that returns an IPC channel to the client

Steps for Implementing a Service

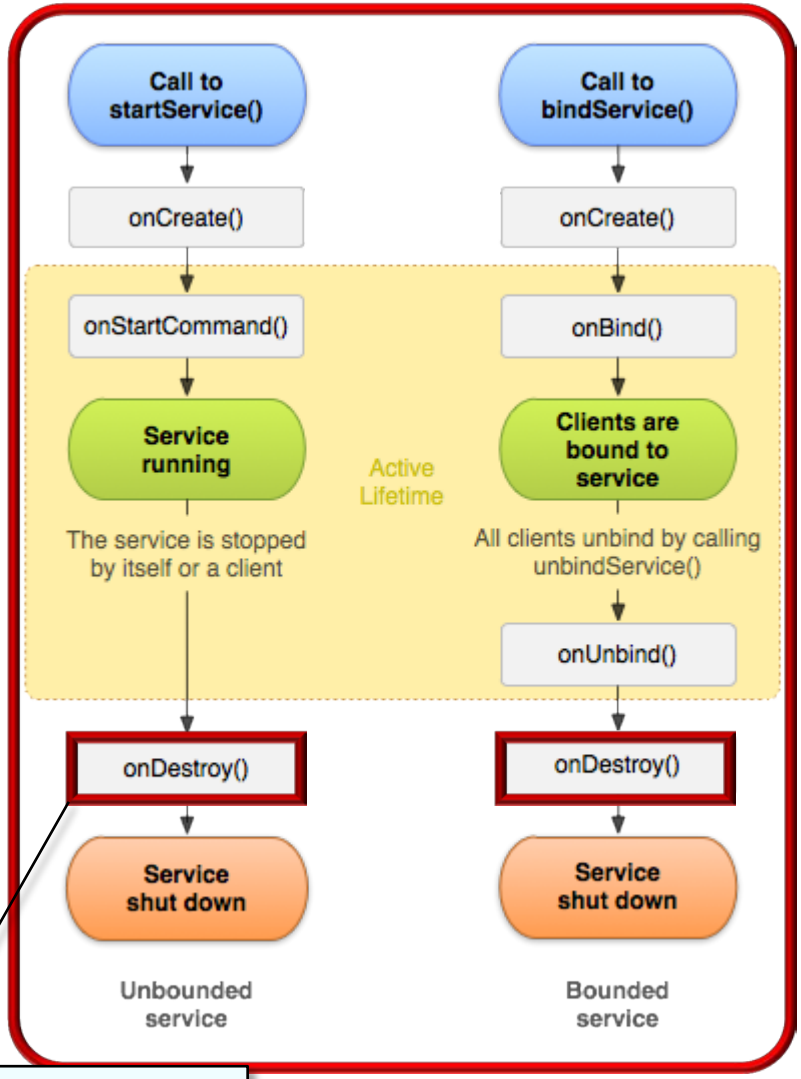
- Implementing a Service is similar to implementing an Activity
- Android communicates state changes to a Service by calling its lifecycle hook methods
  - **onCreate()** – called when Service is first launched, by any means
  - **onStartCommand()** – called each time a Started Service is sent an Intent via startService()
  - **onBind()/onUnbind** – called when client binds/unbinds to Bound Service via bindService()/unBindService()



onUnbind() is a disposal method called when all clients have disconnected

Steps for Implementing a Service

- Implementing a Service is similar to implementing an Activity
- Android communicates state changes to a Service by calling its lifecycle hook methods
  - **onCreate()** – called when Service is first launched, by any means
  - **onStartCommand()** – called each time a Started Service is sent an Intent via startService()
  - **onBind()/onUnbind** – called when client binds/unbinds to Bound Service via bindService()/unBindService()
  - **onDestroy()** – called as Service is being shut down

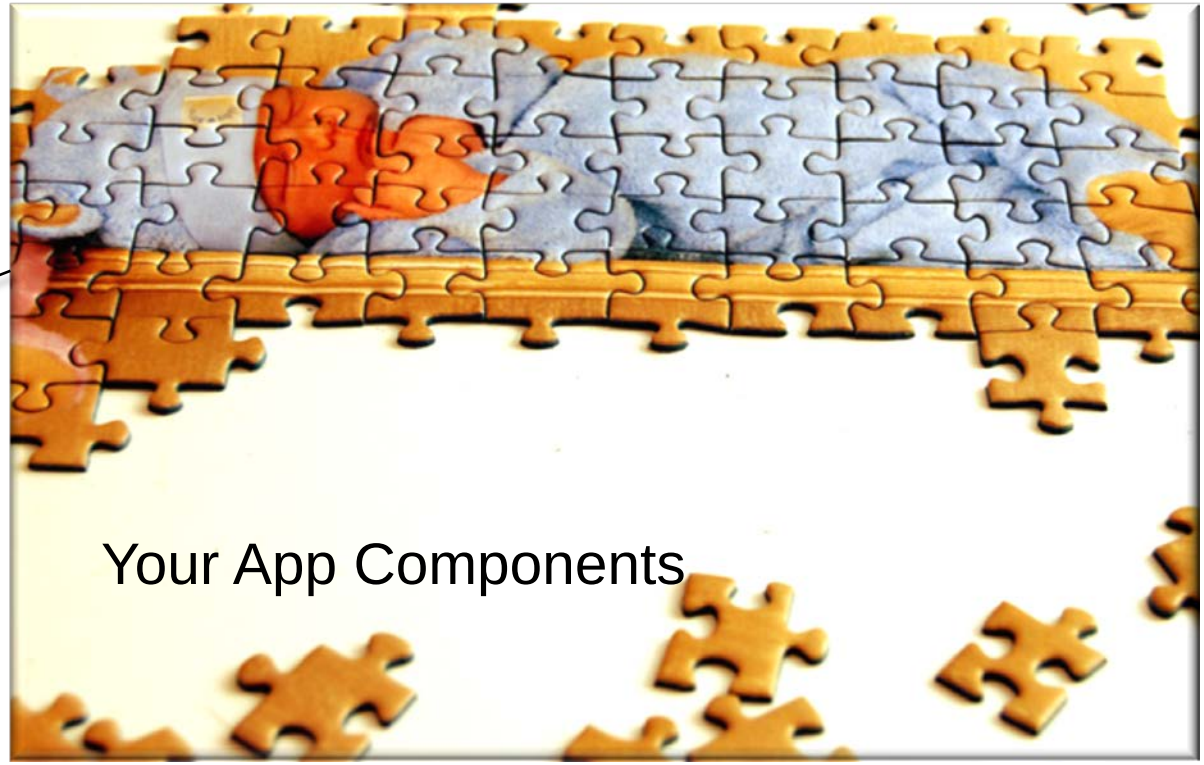


This method cleans up any resources held by the Service

Integrating a Service into an Application

Integrating a Service into an Application

- The Android Service framework provides a semi-complete portion of an Application



The manifest file tells Android how the Activity & Service components "plug-in" to the framework

developer.android.com/guide/topics/manifest/manifest-intro.html has more

Integrating a Service into an Application

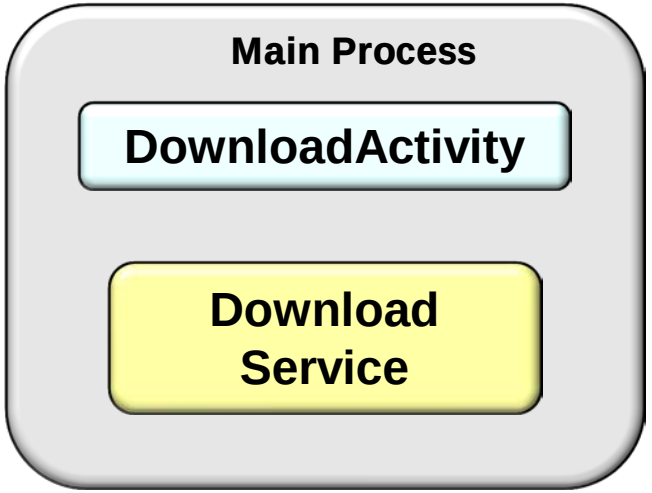
- The Android Service framework provides a semi-complete portion of an Application
- Include the Service in the AndroidManifest.xml file

```
<service
    android:enabled=
        ["true" | "false"]
    android:exported=
        ["true" | "false"]
    android:icon=
        "drawable resource"
    android:isolatedProcess=
        ["true" | "false"]
    android:label=
        "string resource"
    android:name="string"
    android:permission="string"
    android:process="string" >
    ...
</service>
```

See developer.android.com/guide/topics/manifest/service-element.html

Integrating a Service into an Application

- The Android Service framework provides a semi-complete portion of an Application
- Include the Service in the AndroidManifest.xml file
  - Add <service> element as a child of <application> element

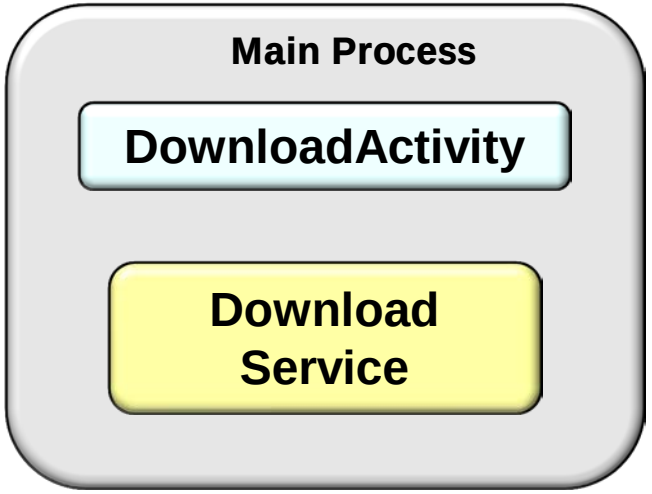


AndroidManifest.xml

```
...
<service android:name=
         "DownloadService"
         android:exported=
         "false"/>
...
```

Integrating a Service into an Application

- The Android Service framework provides a semi-complete portion of an Application
- Include the Service in the AndroidManifest.xml file
  - Add <service> element as a child of <application> element
  - Provide the android:name to reference the Service class



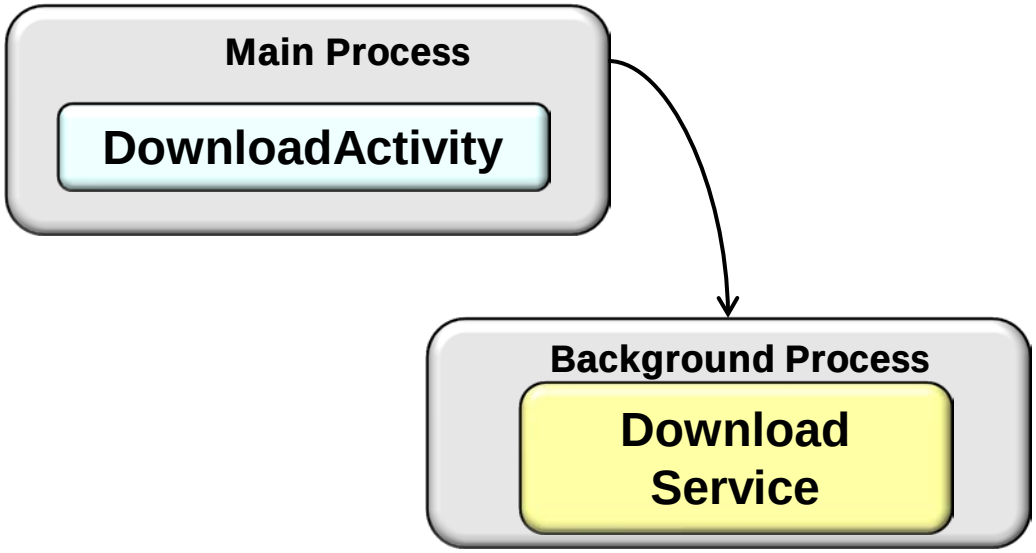
AndroidManifest.xml

```
...
<service android:name=
         "DownloadService"
         android:exported=
         "false"/>
...
```

Services do not automatically run in their own processes or threads

Integrating a Service into an Application

- The Android Service framework provides a semi-complete portion of an Application
- Include the Service in the AndroidManifest.xml file
  - Add <service> element as a child of <application> element
  - Provide the android:name to reference the Service class
  - Use android:process= ":myProcess" to run a Service in a separate process



AndroidManifest.xml

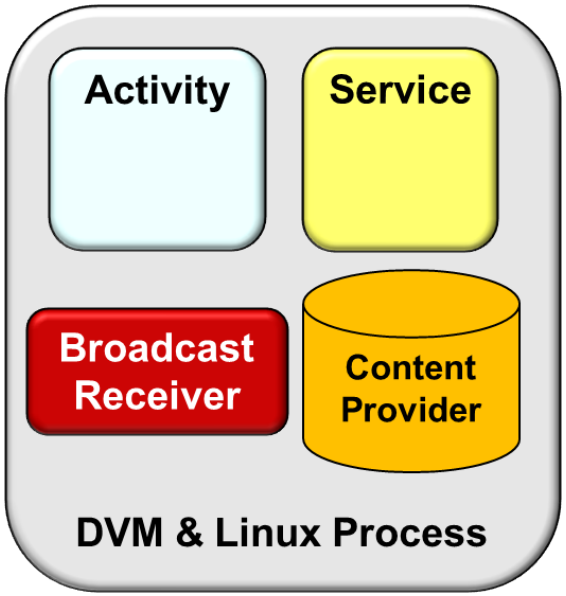
```
...
<service android:name=
         "DownloadService"
         android:exported=
         "false"
         android:process=
         ":myProcess"/>
...
```

developer.android.com/guide/topics/manifest/service-element.html#proc

Summary

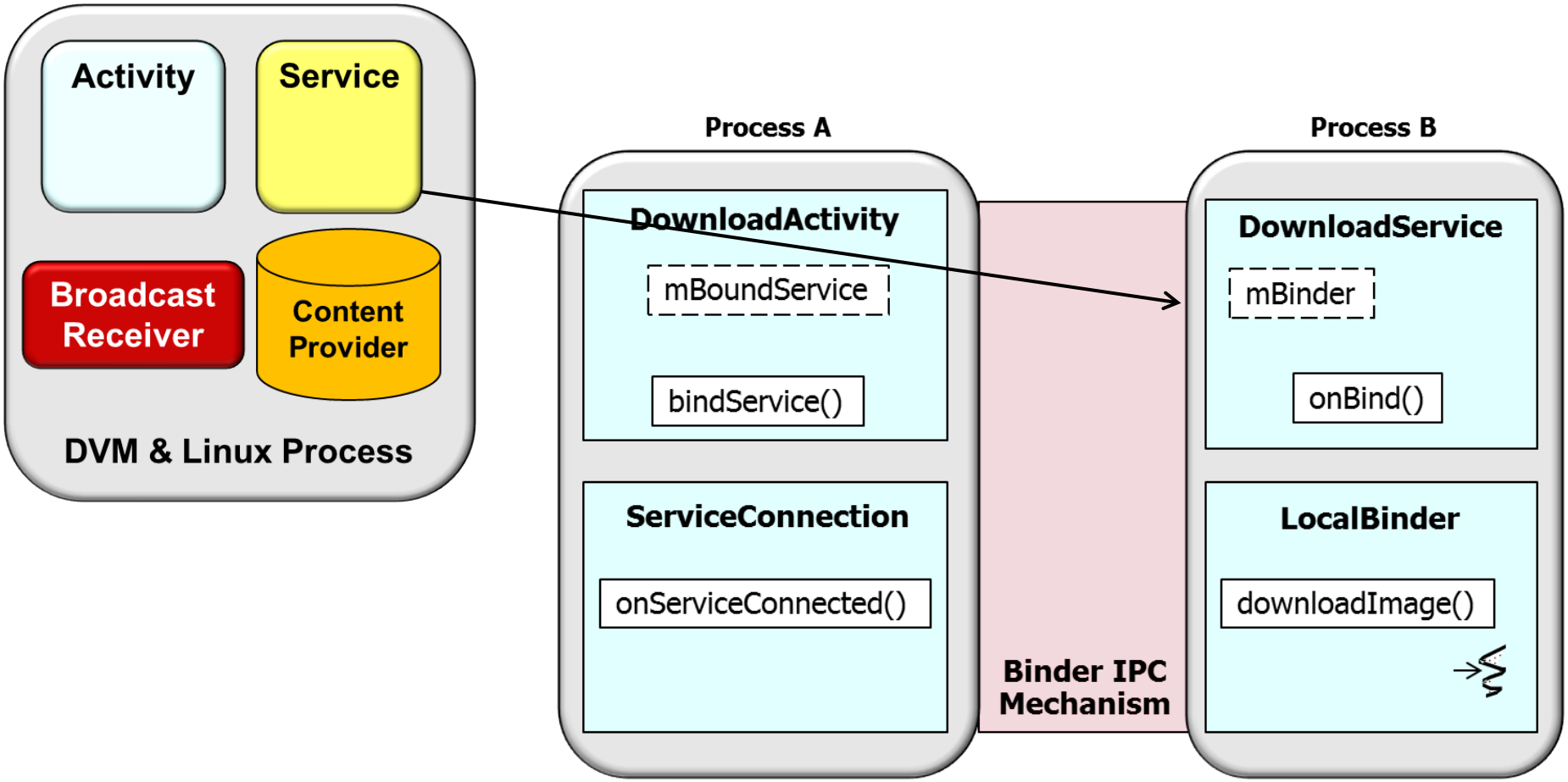
Summary

- Apps can use Services to implement long-duration operations in background



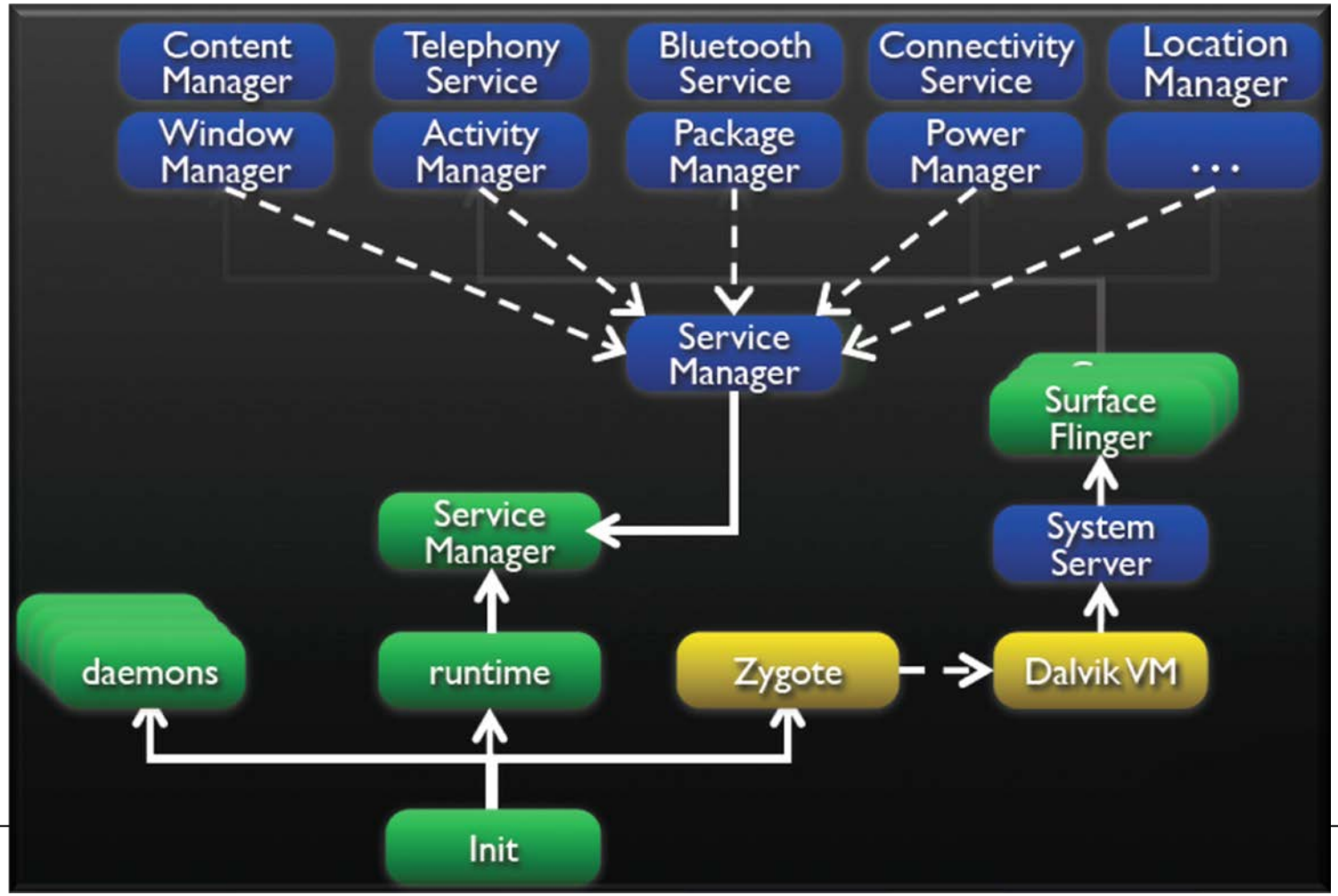
Summary

- Apps can use Services to implement long-duration operations in background



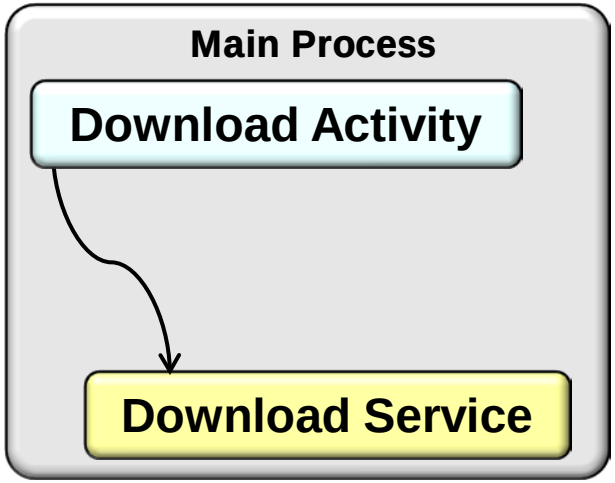
Summary

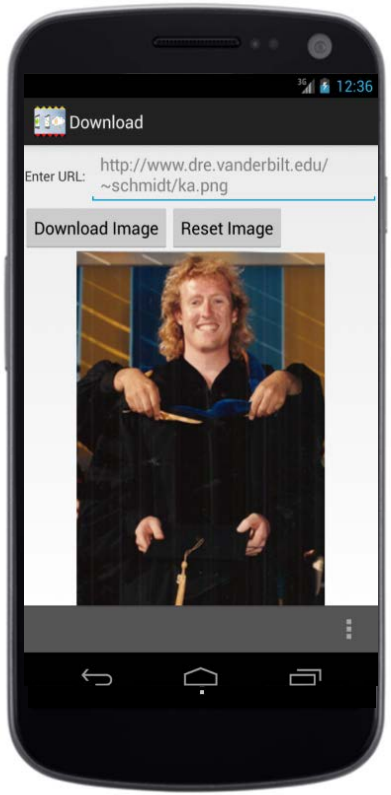
- Apps can use Services to implement long-duration operations in background



Summary

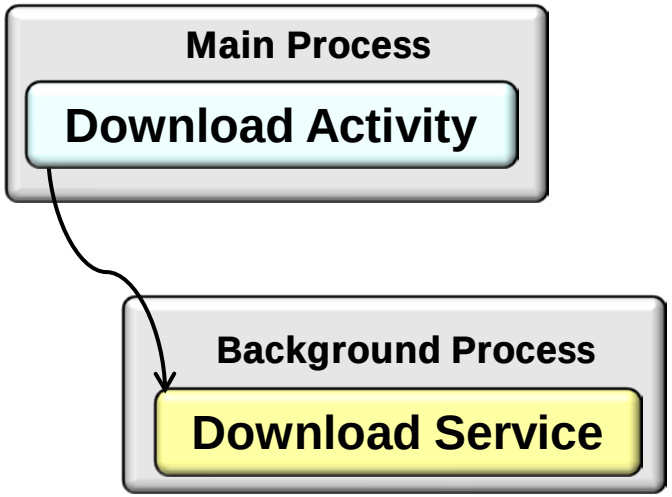
- Apps can use Services to implement long-duration operations in background
  - By default, a Service runs in the same process as the application it is part of





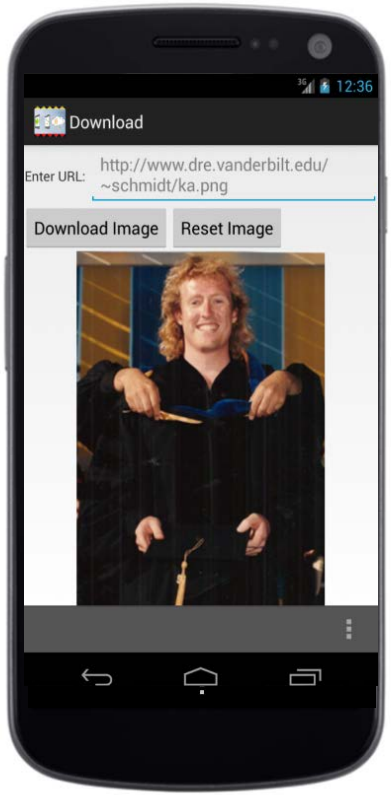
Summary

- Apps can use Services to implement long-duration operations in background
  - By default, a Service runs in the same process as the application it is part of
  - It can also be configured to run in a separate process



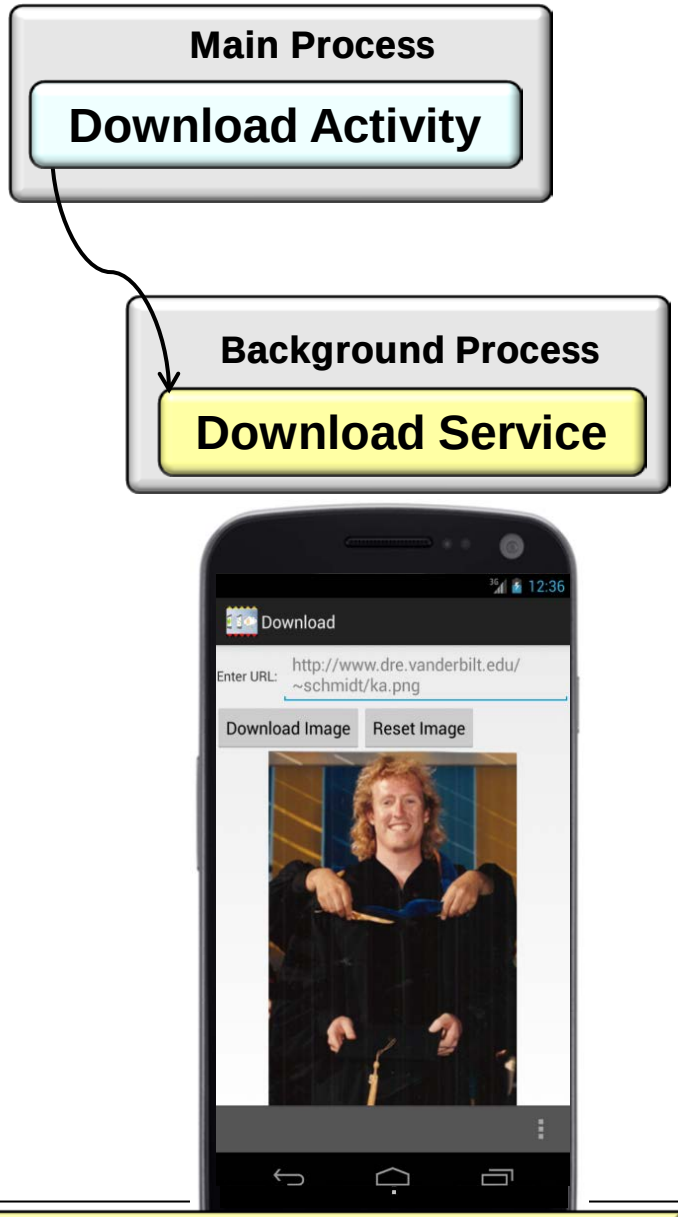
AndroidManifest.xml

```
...
<service android:name=
         "DownloadService"
         android:exported=
         "false"
         android:process=
         ":myProcess"/>
...
```



Summary

- Apps can use Services to implement long-duration operations in background
  - By default, a Service runs in the same process as the application it is part of
  - It can also be configured to run in a separate process
  - It keeps running until it stops itself, is stopped by a user, or killed by the system if it needs memory

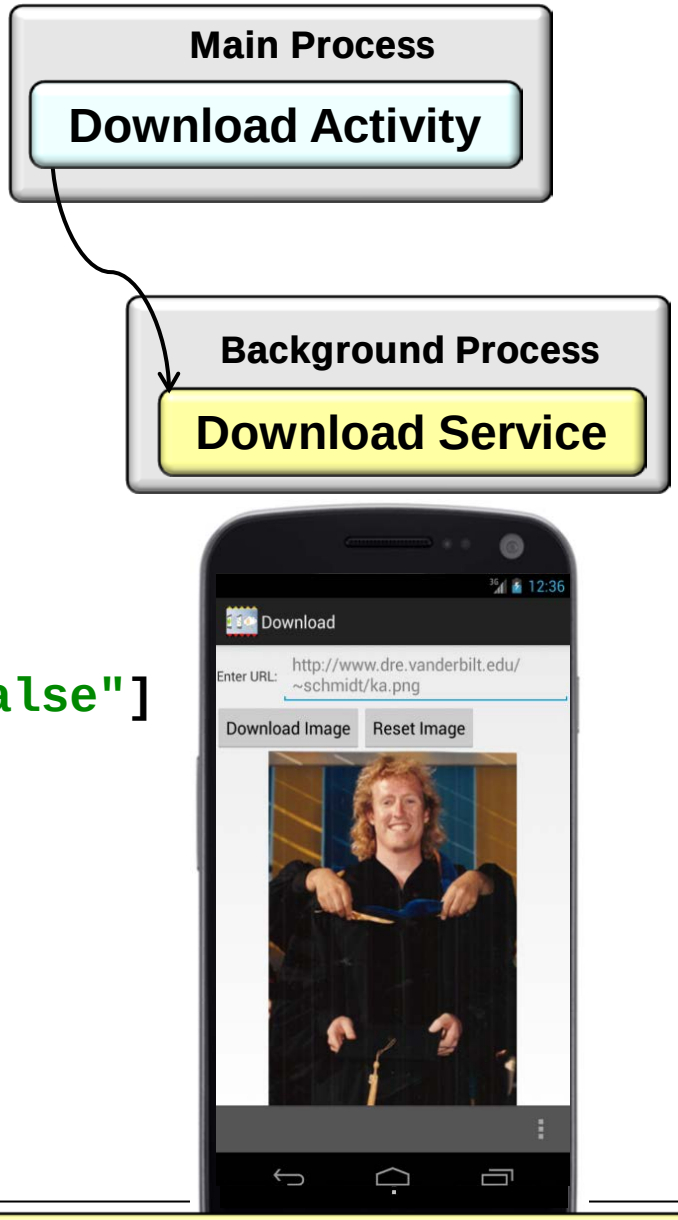


developer.android.com/guide/components/services.html#Lifecycle has more

Summary

- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file

```
<service
    android:enabled=["true" | "false"]
    android:exported=["true" | "false"]
    android:icon="drawable resource"
    android:isolatedProcess=["true" | "false"]
    android:label="string resource"
    android:name="string"
    android:permission="string"
    android:process="string" >
    ...
</service>
```

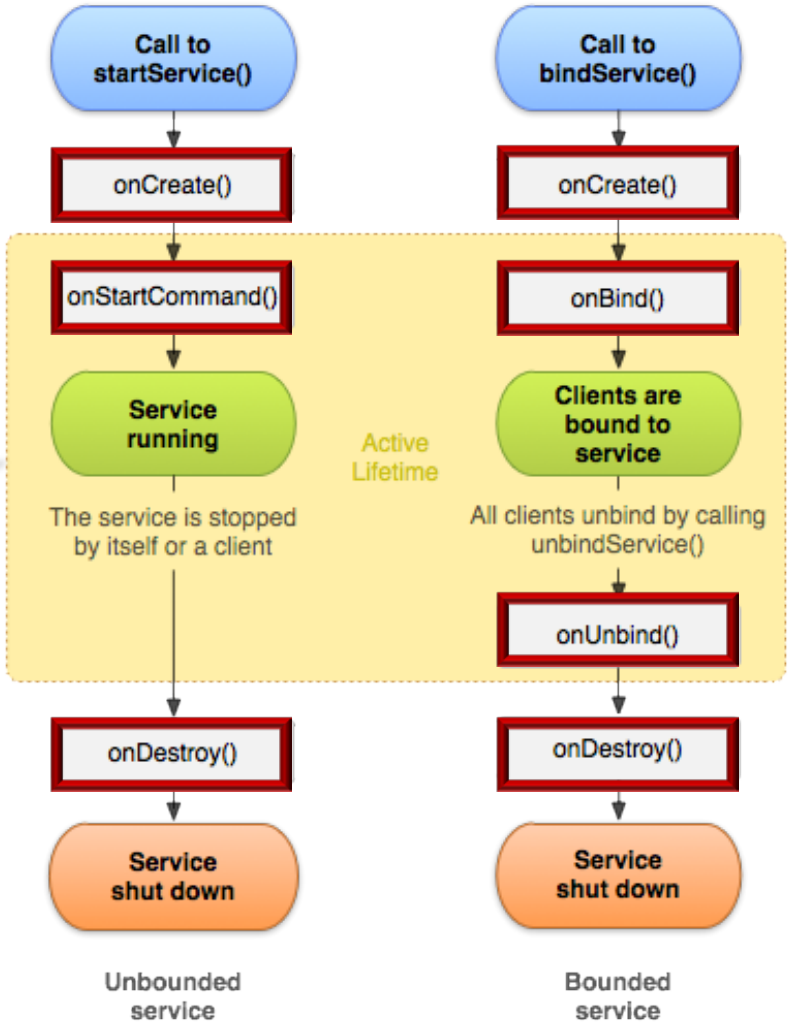


developer.android.com/guide/topics/manifest/service-element.html has more

Summary

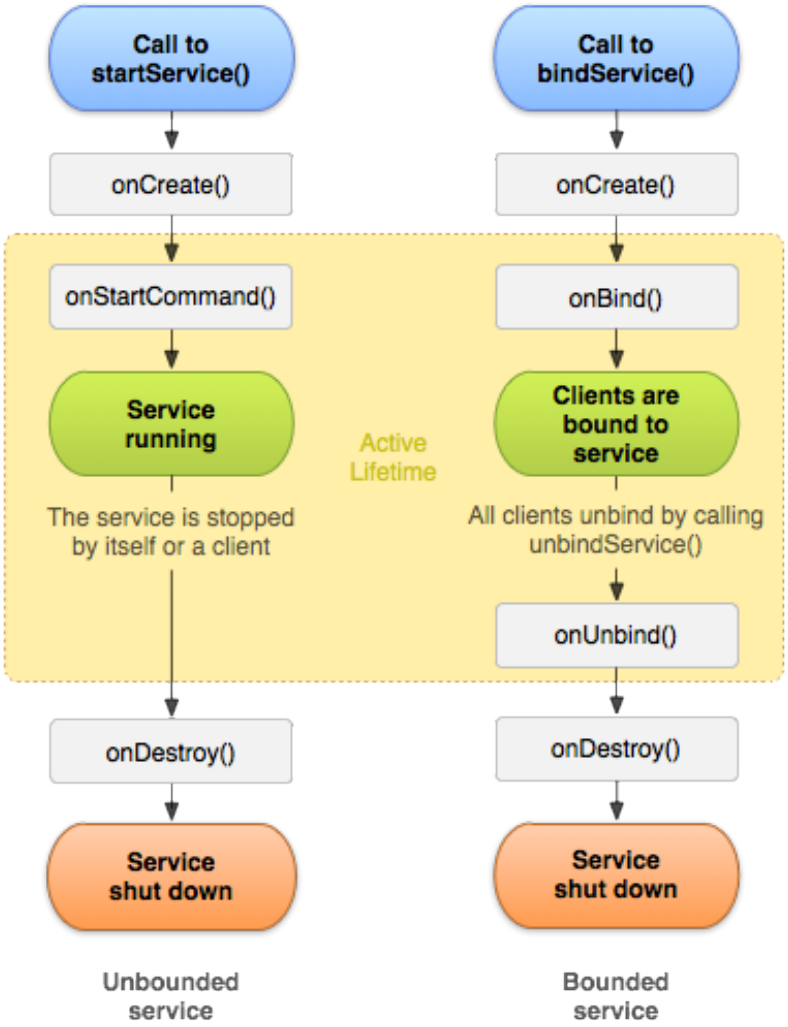
- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle

Note inversion of control from the Android Service framework to the hook methods



Summary

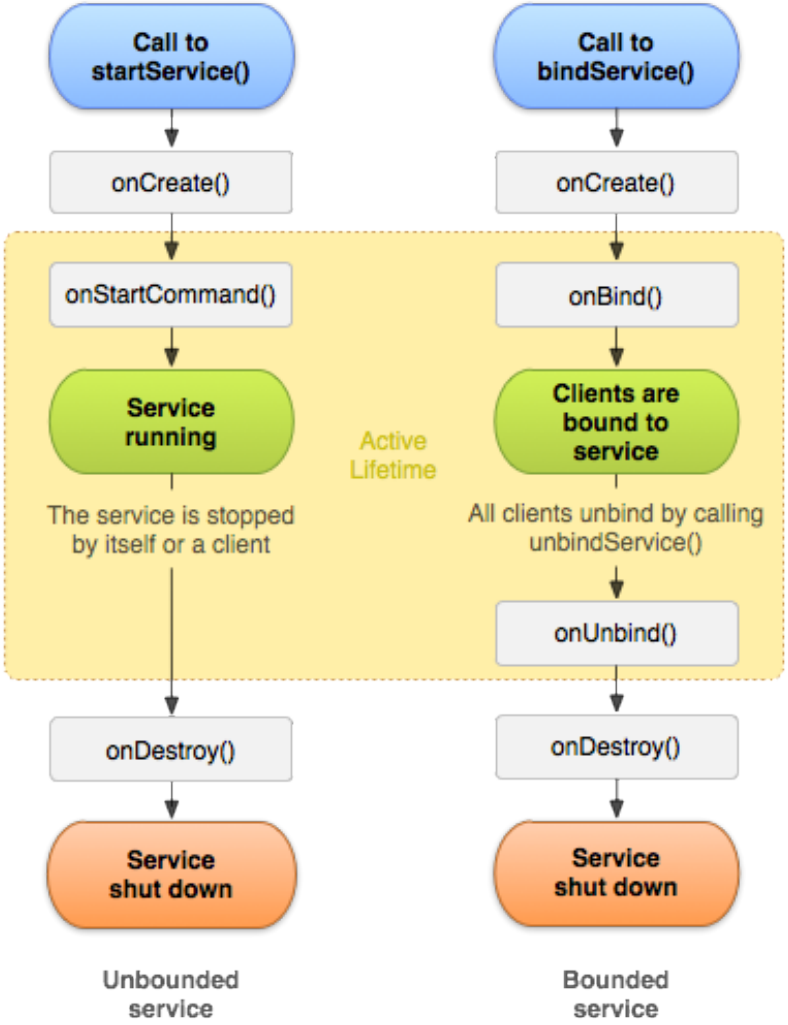
- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle
- Its framework design is based on software product line techniques



See earlier parts on “Framework Examples”

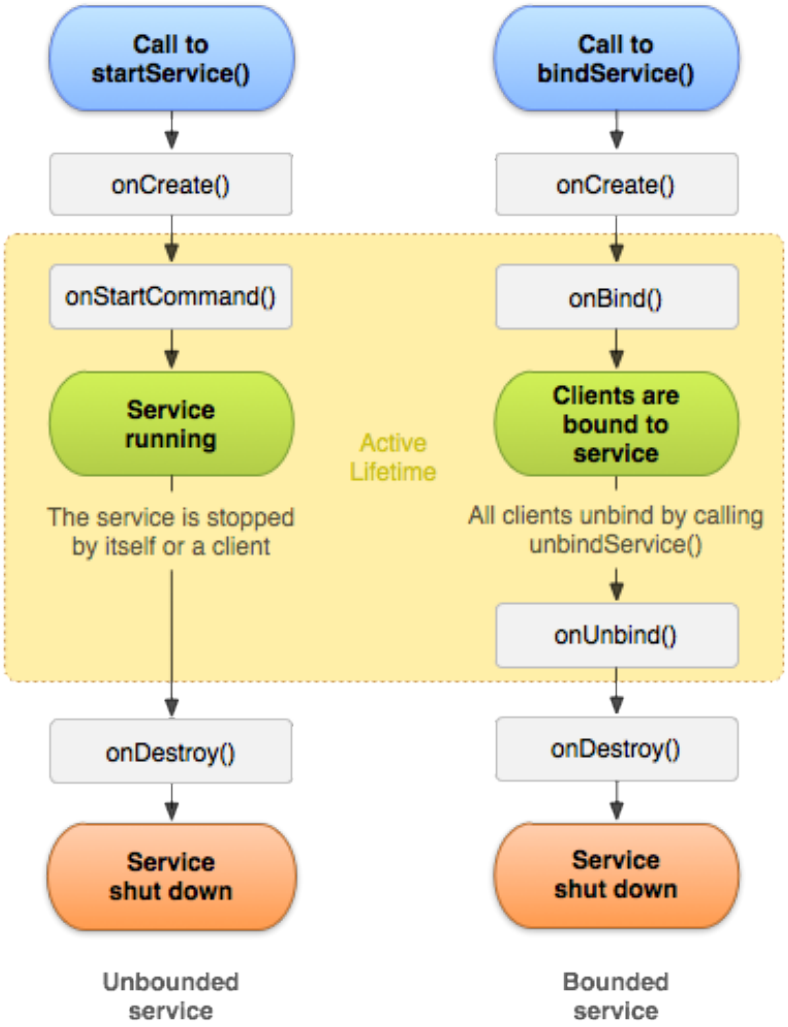
Summary

- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle
- Its framework design is based on software product line methods
  - **Commonality**
    - Provides a uniform interface for performing long-duration operations that don't interact directly with the user interface



Summary

- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle
- Its framework design is based on software product line methods
  - **Commonality**
  - **Variability**
    - Subclasses can override lifecycle hook methods to perform the desired initialization for *Started* & *Bound* Services



Summary

- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle
- Its framework design is based on software product line methods
- Android's packaged applications contain many Services

Service Type	Example	Behavior
Started	*SMS & MMS*	Manage message operations, such as sending data, text, & PDU messages
	Alert	Handle calendar event reminders
Bound	Bluetooth Headset	Provides Headset & Handsfree for Phone App
	Media Playback	Provides audio playback in the background
	Exchange Email	Send email messages

Summary

- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle
- Its framework design is based on software product line methods
- Android's packaged applications contain many Services

Service Type	Example	Behavior
Started	*SMS & MMS*	Manage message operations, such as sending data, text, & PDU messages
	Alert	Handle calendar event reminders
Bound	Bluetooth Headset	Provides Headset & Handsfree for Phone App
	Media Playback	Provides audio playback in the background
	Exchange Email	Send email messages

Summary

- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle
- Its framework design is based on software product line methods
- Android's packaged applications contain many Services

Service Type	Example	Behavior
Started	*SMS & MMS*	Manage message operations, such as sending data, text, & PDU messages
	Alert	Handle calendar event reminders
Bound	Bluetooth Headset	Provides Headset & Handsfree for Phone App
	Media Playback	Provides audio playback in the background
	Exchange Email	Send email messages

Summary

- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle
- Its framework design is based on software product line methods
- Android's packaged applications contain many Services

Service Type	Example	Behavior
Started	*SMS & MMS*	Manage message operations, such as sending data, text, & PDU messages
	Alert	Handle calendar event reminders
Bound	Bluetooth Headset	Provides Headset & Handsfree for Phone App
	Media Playback	Provides audio playback in the background
	Exchange Email	Send email messages

Summary

- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle
- Its framework design is based on software product line methods
- Android's packaged applications contain many Services

Service Type	Example	Behavior
Started	*SMS & MMS*	Manage message operations, such as sending data, text, & PDU messages
	Alert	Handle calendar event reminders
Bound	Bluetooth Headset	Provides Headset & Handsfree for Phone App
	Media Playback	Provides audio playback in the background
	Exchange Email	Send email messages

Summary

- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle
- Its framework design is based on software product line methods
- Android's packaged applications contain many Services

Service Type	Example	Behavior
Started	*SMS & MMS*	Manage message operations, such as sending data, text, & PDU messages
	Alert	Handle calendar event reminders
Bound	Bluetooth Headset	Provides Headset & Handsfree for Phone App
	Media Playback	Provides audio playback in the background
	Exchange Email	Send email messages

Summary

- Apps can use Services to implement long-duration operations in background
- You can configure many properties of Services via an AndroidManifest.xml file
- Android implements Services via a framework that dispatches hook methods to manage Service lifecycle
- Its framework design is based on software product line methods
- Android's packaged applications contain many Services

Service Type	Example	Behavior
Started	*SMS & MMS*	Manage message operations, such as sending data, text, & PDU messages
	Alert	Handle calendar event reminders
Bound	Bluetooth Headset	Provides Headset & Handsfree for Phone App
	Media Playback	Provides audio playback in the background
	Exchange Email	Send email messages

www.vogella.com/articles/AndroidServices/article.html has more on Services