ConnectivityManager

extends Object

java.lang.Object

↳android.net.ConnectivityManager

Class Overview

Class that answers queries about the state of network connectivity. It also notifies applications when network connectivity changes. Get an instance of this class by calling`Context.getSystemService(Context.CONNECTIVITY_SERVICE)`.

The primary responsibilities of this class are to:

1. Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)
2. Send broadcast intents when network connectivity changes
3. Attempt to "fail over" to another network when connectivity to a network is lost
4. Provide an API that allows applications to query the coarse-grained or fine-grained state of the available networks

Summary

Constants		
String	ACTION_BACKGROUND_DATA_SETTING_CHANGED	Broadcast Action: The setting for background data usa values.
String	CONNECTIVITY_ACTION	A change in network connectivity has occurred.
int	DEFAULT_NETWORK_PREFERENCE	
String	EXTRA_EXTRA_INFO	The lookup key for a string that provides optionally su information about the network state.
String	EXTRA_IS_FAILOVER	The lookup key for a boolean that indicates whether a for a network to which the connectivity manager was following a disconnect on another network.
String	EXTRA_NETWORK_INFO	*This constant is deprecated. Since `NetworkInfo` can*

		applications should always obtain network information through `getActiveNetworkInfo()` *or* `getAllNetw`
String	EXTRA_NO_CONNECTIVITY	The lookup key for a boolean that indicates whether th lack of connectivity, i.e., no network is available.
String	EXTRA_OTHER_NETWORK_INFO	The lookup key for a `NetworkInfo` object.
String	EXTRA_REASON	The lookup key for a string that indicates why an atten network failed.
int	TYPE_BLUETOOTH	The Default Bluetooth data connection.
int	TYPE_DUMMY	Dummy data connection.
int	TYPE_ETHERNET	The Default Ethernet data connection.
int	TYPE_MOBILE	The Default Mobile data connection.
int	TYPE_MOBILE_DUN	A DUN-specific Mobile data connection.
int	TYPE_MOBILE_HIPRI	A High Priority Mobile data connection.
int	TYPE_MOBILE_MMS	An MMS-specific Mobile data connection.
int	TYPE_MOBILE_SUPL	A SUPL-specific Mobile data connection.
int	TYPE_WIFI	The Default WIFI data connection.
int	TYPE_WIMAX	The Default WiMAX data connection.

Public Methods

NetworkInfo	getActiveNetworkInfo()
NetworkInfo[]	getAllNetworkInfo()

boolean	getBackgroundDataSetting() *This method is deprecated. As of `ICE_CREAM_SANDWICH`, availability of background data depends on various factors, and this method will always return `true`. Instead, when background data is unavailable, `getActiveNetworkInfo()` will now appear disconnected.*
NetworkInfo	getNetworkInfo(int networkType)
int	getNetworkPreference()
static boolean	isNetworkTypeValid(int networkType)
boolean	requestRouteToHost(int networkType, int hostAddress) Ensure that a network route exists to deliver traffic to the specified host via the specified network int
void	setNetworkPreference(int preference)
int	startUsingNetworkFeature(int networkType, String feature) Tells the underlying networking system that the caller wants to begin using the named feature.
int	stopUsingNetworkFeature(int networkType, String feature) Tells the underlying networking system that the caller is finished using the named feature.

[Expand]

Inherited Methods

▸ From class java.lang.Object

Constants

public static final String ***ACTION_BACKGROUND_DATA_SETTING_CHANGED***

Since: API Level 3

Broadcast Action: The setting for background data usage has changed values. Use `getBackgroundDataSetting()` to get the current value.

If an application uses the network in the background, it should listen for this broadcast and stop using the background data if the value is `false`.

Constant Value: "android.net.conn.BACKGROUND_DATA_SETTING_CHANGED"

public static final String ***CONNECTIVITY_ACTION***

Since: API Level 1

A change in network connectivity has occurred. A connection has either been established or lost. The NetworkInfo for the affected network is sent as an extra; it should be consulted to see what kind of connectivity event occurred.

If this is a connection that was the result of failing over from a disconnected network, then the FAILOVER_CONNECTION boolean extra is set to true.

For a loss of connectivity, if the connectivity manager is attempting to connect (or has already connected) to another network, the NetworkInfo for the new network is also passed as an extra. This lets any receivers of the broadcast know that they should not necessarily tell the user that no data traffic will be possible. Instead, the reciever should expect another broadcast soon, indicating either that the failover attempt succeeded (and so there is still overall data connectivity), or that the failover attempt failed, meaning that all connectivity has been lost.

For a disconnect event, the boolean extra EXTRA_NO_CONNECTIVITY is set to `true` if there are no connected networks at all.

Constant Value: "android.net.conn.CONNECTIVITY_CHANGE"

public static final int ***DEFAULT_NETWORK_PREFERENCE***

Since: API Level 1

Constant Value: 1 (0x00000001)

public static final String ***EXTRA_EXTRA_INFO***

Since: API Level 1

The lookup key for a string that provides optionally supplied extra information about the network state. The information may be passed up from the lower networking layers, and its meaning may be specific to a particular network type. Retrieve it with `getStringExtra(String)`.

Constant Value: "extraInfo"

public static final String ***EXTRA_IS_FAILOVER***

Since: API Level 1

The lookup key for a boolean that indicates whether a connect event is for a network to which the connectivity manager was failing over following a disconnect on another network. Retrieve it with `getBooleanExtra(String, boolean)`.

Constant Value: "isFailover"

public static final String ***EXTRA_NETWORK_INFO***

Since: API Level 1

> **This constant is deprecated.**
> Since `NetworkInfo` can vary based on UID, applications should always obtain network information through `getActiveNetworkInfo()` or `getAllNetworkInfo()`.

The lookup key for a `NetworkInfo` object. Retrieve with `getParcelableExtra(String)`.

Constant Value: "networkInfo"

public static final [String](#) ***EXTRA_NO_CONNECTIVITY***

Since: API Level 1

The lookup key for a boolean that indicates whether there is a complete lack of connectivity, i.e., no network is available. Retrieve it with `getBooleanExtra(String, boolean)`.

Constant Value: "noConnectivity"

public static final [String](#) ***EXTRA_OTHER_NETWORK_INFO***

Since: API Level 1

The lookup key for a `NetworkInfo` object. This is supplied when there is another network that it may be possible to connect to. Retrieve with `getParcelableExtra(String)`.

Constant Value: "otherNetwork"

public static final [String](#) ***EXTRA_REASON***

Since: API Level 1

The lookup key for a string that indicates why an attempt to connect to a network failed. The string has no particular structure. It is intended to be used in notifications presented to users. Retrieve it with `getStringExtra(String)`.

Constant Value: "reason"

public static final int ***TYPE_BLUETOOTH***

Since: API Level 13

The Default Bluetooth data connection. When active, all data traffic will use this connection by default.

Constant Value: 7 (0x00000007)

public static final int ***TYPE_DUMMY***

Since: API Level 14

Dummy data connection. This should not be used on shipping devices.

Constant Value: 8 (0x00000008)

public static final int ***TYPE_ETHERNET***

Since: API Level 13

The Default Ethernet data connection. When active, all data traffic will use this connection by default.

Constant Value: 9 (0x00000009)

public static final int ***TYPE_MOBILE***

Since: API Level 1

The Default Mobile data connection. When active, all data traffic will use this connection by default.
Constant Value: 0 (0x00000000)

public static final int ***TYPE_MOBILE_DUN***

Since: API Level 8

A DUN-specific Mobile data connection. This connection may be the same as `TYPE_MOBILE` but it may be different. This is used by applicaitons performing a Dial Up Networking bridge so that the carrier is aware of DUN traffic. It may coexist with default data connections.
Constant Value: 4 (0x00000004)

public static final int ***TYPE_MOBILE_HIPRI***

Since: API Level 8

A High Priority Mobile data connection. This connection is typically the same as `TYPE_MOBILE` but the routing setup is different. Only requesting processes will have access to the Mobile DNS servers and only IP's explicitly requested via `requestRouteToHost(int, int)` will route over this interface if a default route exists.
Constant Value: 5 (0x00000005)

public static final int ***TYPE_MOBILE_MMS***

Since: API Level 8

An MMS-specific Mobile data connection. This connection may be the same as `TYPE_MOBILE` but it may be different. This is used by applications needing to talk to the carrier's Multimedia Messaging Service servers. It may coexist with default data connections.
Constant Value: 2 (0x00000002)

public static final int ***TYPE_MOBILE_SUPL***

Since: API Level 8

A SUPL-specific Mobile data connection. This connection may be the same as `TYPE_MOBILE` but it may be different. This is used by applications needing to talk to the carrier's Secure User Plane Location servers for help locating the device. It may coexist with default data connections.
Constant Value: 3 (0x00000003)

public static final int ***TYPE_WIFI***

Since: API Level 1

The Default WIFI data connection. When active, all data traffic will use this connection by default.

Constant Value: 1 (0x00000001)

public static final int ***TYPE_WIMAX***

Since: API Level 8

The Default WiMAX data connection. When active, all data traffic will use this connection by default.

Constant Value: 6 (0x00000006)

Public Methods

public NetworkInfo ***getActiveNetworkInfo*** *()*

Since: API Level 1

public NetworkInfo[] ***getAllNetworkInfo*** *()*

Since: API Level 1

public boolean ***getBackgroundDataSetting*** *()*

Since: API Level 3

> **This method is deprecated.**
> As of `ICE_CREAM_SANDWICH`, availability of background data depends on several combined factors, and this method will always return `true`. Instead, when background data is unavailable, `getActiveNetworkInfo()` will now appear disconnected.

Returns the value of the setting for background data usage. If false, applications should not use the network if the application is not in the foreground. Developers should respect this setting, and check the value of this before performing any background data operations.

All applications that have background services that use the network should listen to `ACTION_BACKGROUND_DATA_SETTING_CHANGED`.

Returns

- Whether background data usage is allowed.

public NetworkInfo ***getNetworkInfo*** *(int networkType)*

Since: API Level 1

public int ***getNetworkPreference*** *()*

Since: API Level 1

public static boolean ***isNetworkTypeValid*** *(int networkType)*

Since: API Level 1

public boolean ***requestRouteToHost*** *(int networkType, int hostAddress)*

Since: API Level 1

Ensure that a network route exists to deliver traffic to the specified host via the specified network interface. An attempt to add a route that already exists is ignored, but treated as successful.

Parameters

networkType	the type of the network over which traffic to the specified host is to be routed
hostAddress	the IP address of the host to which the route is desired

Returns

- `true` on success, `false` on failure

public void ***setNetworkPreference*** *(int preference)*

Since: API Level 1

public int ***startUsingNetworkFeature*** *(int networkType, String feature)*

Since: API Level 1

Tells the underlying networking system that the caller wants to begin using the named feature. The interpretation of `feature` is completely up to each networking implementation.

Parameters

networkType	specifies which network the request pertains to
feature	the name of the feature to be used

Returns

- an integer value representing the outcome of the request. The interpretation of this value is specific to each networking implementation+feature combination, except that the value `-1` always indicates failure.

public int ***stopUsingNetworkFeature*** *(int networkType, String feature)*

Since: API Level 1

Tells the underlying networking system that the caller is finished using the named feature. The interpretation of `feature` is completely up to each networking implementation.

Parameters

networkType	specifies which network the request pertains to

feature the name of the feature that is no longer needed

Returns

- an integer value representing the outcome of the request. The interpretation of this value is specific to each networking implementation+feature combination, except that the value `-1` always indicates failure.