

MVIDDeviceConnector for Android

Generated by Doxygen 1.8.6

Tue Jun 3 2014 13:22:19

Contents

1	Class Index	1
1.1	Class List	1
2	Class Documentation	3
2.1	com.mvnordic.mviddeviceconnector.DeviceSecurity Class Reference	3
2.1.1	Detailed Description	3
2.1.2	Usage	3
2.1.2.1	MVIDResponse notification object	4
2.1.2.2	Coding example	4
2.1.3	Constructor & Destructor Documentation	5
2.1.3.1	DeviceSecurity	5
2.1.4	Member Function Documentation	5
2.1.4.1	addDeviceSecurityListener	5
2.1.4.2	doLogin	5
2.1.4.3	doLogin	6
2.1.4.4	excludeLoginGroup	6
2.1.4.5	getMVSessionID	7
2.1.4.6	includeAllLoginGroups	7
2.1.4.7	popDeviceSecurityListener	7
2.1.4.8	releaseDeviceRegistration	7
2.1.4.9	removeDeviceSecurityListener	7
2.1.4.10	setHorizontalDisplacement	8
2.2	com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener Interface Reference . .	8
2.2.1	Detailed Description	8
2.3	com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener.MVIDResponse Class Reference	8
2.3.1	Detailed Description	9
2.3.2	Member Data Documentation	9
2.3.2.1	access_identifier	9
2.3.2.2	has_access	9
2.3.2.3	request_id	9
2.3.2.4	res_code	9

2.3.2.5	res_msg	9
2.3.2.6	service_result_flags	9
2.4	com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult Class Reference	10
2.4.1	Detailed Description	10
2.4.2	Member Data Documentation	10
2.4.2.1	AccessDenied	10
2.4.2.2	ApplicationBorrowTimeExpired	10
2.4.2.3	DeviceBorrowTimeExpired	10
2.4.2.4	InvalidMVSessionID	11
2.4.2.5	NetworkError	11
2.4.2.6	ServiceFault	11
2.4.2.7	ServiceSuccess	11
2.4.2.8	Success	11
	Index	12

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

com.mvnordic.mviddeviceconnector.DeviceSecurity	3
com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener	8
com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener.MVIDResponse	8
com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult	10

Chapter 2

Class Documentation

2.1 com.mvnordic.mviddeviceconnector.DeviceSecurity Class Reference

Classes

- interface [DeviceSecurityListener](#)
- class [ServiceResult](#)

Public Member Functions

- [DeviceSecurity](#) (Activity parent)
- void [addDeviceSecurityListener](#) ([DeviceSecurityListener](#) listener)
- void [removeDeviceSecurityListener](#) ([DeviceSecurityListener](#) listener)
- void [popDeviceSecurityListener](#) ()
- void [setHorizontalDisplacement](#) (int pixels)
- void [excludeLoginGroup](#) (String login_group)
- void [includeAllLoginGroups](#) ()
- int [doLogin](#) (String access_identifier)
- int [doLogin](#) (String access_identifier, int fragment_container_id)
- void [releaseDeviceRegistration](#) ()
- String [getMVSessionID](#) (String access_identifier)

2.1.1 Detailed Description

[DeviceSecurity](#) is the client side interface towards the corresponding MVID web service [DeviceSecurity](#). It is designed for easy integration into existing Android apps.

Once the interface is properly integrated into an app it will provide:

- MVID User authentication
- Product access services
- MVID Session ID for further requesting into MVID's web services: [MVID Services](#)

2.1.2 Usage

The primary method of [DeviceSecurity](#) is the [doLogin\(\)](#) method. In a normal integration this method should be all that is needed.

The basic integration has 3 steps:

- Instantiate [DeviceSecurity](#) as a member of the class responsible for the login process.
- Add a notification listener to handle the final asynchronous [MVIDResponse](#) passed via `onMVIDResponseReady()`;
- Call `doLogin()`

Note

After a successful login the `mv_session_id` is accessible via [DeviceSecurity.getSessionID\(\)](#).

2.1.2.1 MVIDResponse notification object

The [MVIDResponse](#) notification is posted via the `onMVIDResponseReady()`. Use [addDeviceSecurityListener\(\)](#) to start listening. [MVIDResponse](#) defines the following attributes:

First Header	Second Header	Description
int	service_result_flags	Bitwise combination of ServiceResult flags
boolean	has_access	true if the access_identifier is granted false if not
int	res_code	The server side result code
String	res_msg	The server side result message
String	access_identifier	The access identifier which was checked
Integer	request_id	The doLogin request ID of the check (returned from doLogin())

2.1.2.2 Coding example

The following is a very simple example of how an Activity with MVID Login responsibility could be coded.

MVIDLoginActivity.java:

```
import com.mvnordic.mviddeviceconnector.DeviceSecurity;

public class MVIDLoginActivity extends Activity {

    // Define a DeviceSecurity member
    private DeviceSecurity device_security;

    // Define a listener for the DeviceSecurity.doLogin() result.
    private DeviceSecurity.DeviceSecurityListener device_security_listener = new
        DeviceSecurity.DeviceSecurityListener() {

        @Override
        public void onMVIDResponseReady(MVIDResponse response) {
            // Print the interesting part of the result as a toast.
            String toast = String.format("AI: %s\nREQUEST_ID: %s\nGOT ACCESS: %s",
                response.access_identifier,
                response.request_id,
                response.has_access ? "YES" : "NO" );
            Toast.makeText(getApplicationContext(), toast,
                Toast.LENGTH_SHORT).show();
        }

    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Create an instance of DeviceSecurity
        device_security = new DeviceSecurity(this);

        // Do some setting up for the Login GUI
        device_security.excludeLoginGroup("company");
        device_security.excludeLoginGroup("private");
    }
}
```



```

// Add the response listener declared above
device_security.addDeviceSecurityListener(device_security_listener);

// In this example imagine a button on your Activity UI with the label "Login"
// When pressed the login process should start.
Button login_button = (Button) this.findViewById(R.id.login_button);
login_button.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
        int req_id = device_security.doLogin(spinner.getSelectedItem().toString());
        L.d("The login request ID: %d" , req_id);
    }
});
}
}
}

```

2.1.3 Constructor & Destructor Documentation

2.1.3.1 com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurity (Activity parent)

The default [DeviceSecurity](#) constructor. [DeviceSecurity](#) must have a parent Activity which connects it to a Android context.

Parameters

<i>parent</i>	The parent Activity for this DeviceSecurity instance.
---------------	---

2.1.4 Member Function Documentation

2.1.4.1 void com.mvnordic.mviddeviceconnector.DeviceSecurity.addDeviceSecurityListener (DeviceSecurityListener listener)

Add a listener to a [DeviceSecurity](#) instance. [DeviceSecurity](#) will invoke the onMVIDResponseReady() method on all added listeners when a MVID Login response is ready, passing the [MVIDResponse](#) object.

Example usage:

```

// Define a listener for the DeviceSecurity.doLogin() result.
private DeviceSecurity.DeviceSecurityListener device_security_listener = new
    DeviceSecurity.DeviceSecurityListener() {

    @Override
    public void onMVIDResponseReady(MVIDResponse response) {
        // Print the interesting part of the result as a toast.
        if (response.has_access) {
            // Put code for granted access ...
        }
        else {
            // Access denied ...
        }
    }
};

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    device_security.addDeviceSecurityListener(device_security_listener);
    // ...
}

```

Parameters

<i>listener</i>	The listener instance to be added
-----------------	-----------------------------------

2.1.4.2 int com.mvnordic.mviddeviceconnector.DeviceSecurity.doLogin (String access_identifier)

Start the login process, communicating with MVID backend servers if online, else if any previous login has been performed and is still within the "borrow time" grace period, this will be used directly off the device. This means that a single login operation can be reused in offline mode some time.

Note

To force device to re-login you must call `releaseDeviceRegistraion` first. This method will start the login process as a new Activity.

Example usage:

```
// In this example the LoginViewController's views are displayed in a
// overlay subview of my LoginController called myLoginView
int req_id = device_security.doLogin(access_identifier);
```

Parameters

<i>access_identifier</i>	The access_identifier that needs to be checked.
--------------------------	---

Returns

The request ID of the login process This ID corresponds with the MVIDResponse which is notified via Device-SecurityListener.onMVIDResponseReady(). So it is possible to distinguish between login results in a multi-threaded environment.

2.1.4.3 `int com.mvnnordic.mviddeviceconnector.DeviceSecurity.doLogin (String access_identifier, int fragment_container_id)`

Start the login process, communicating with MVID backend servers if online, else if any previous login has been performed and is still within the "borrow time" grace period, this will be used directly off the device. This means that a single login operation can be reused in offline mode some time.

Note

To force device to re-login you must call `releaseDeviceRegistraion` first. To direct the login screen as a Fragment into a container View (like a Layout) in your Activity, use the `fragment_container_id` parameter.

Example usage:

```
// In this example the LoginViewController's views are displayed in a
// overlay subview of my LoginController called myLoginView
int req_id = device_security.doLogin(access_identifier,R.id.login_fragment_contatiner);
```

Parameters

<i>access_identifier</i>	The access_identifier that needs to be checked.
<i>fragment_container_id</i>	A resource ID referencing the container View on the parent activity that should embed the Login Fragment. If an invalid resource is passed no UI will be presented.

Returns

The request ID of the login process This ID corresponds with the MVIDResponse which is notified via Device-SecurityListener.onMVIDResponseReady(). So it is possible to distinguish between login results in a multi-threaded environment.

2.1.4.4 `void com.mvnnordic.mviddeviceconnector.DeviceSecurity.excludeLoginGroup (String login_group)`

Exclude a login group from the login UI.

Example usage:

```
// Exclude the school login group
device_security.excludeLoginGroup("company");
```

Parameters

<i>login_group</i>	The name of the login group. At the time of writing possible groups are: school, company and private
--------------------	--

2.1.4.5 String com.mvnordic.mviddeviceconnector.DeviceSecurity.getMVSessionID (String *access_identifier*)

Extract the mv_session_id associated with the login to to a specific access identifier.

Example usage:

```
// Get my mv_session_id
device_security.getMVSessionID("product.android.da.intowords.release");
```

Parameters

<i>access_identifier</i>	Get the mv_session_id currently associated with given access_identifier.
--------------------------	--

Returns

mv_session_id The mv_session_id is returned if present, otherwise if no login has been made, null is returned-

2.1.4.6 void com.mvnordic.mviddeviceconnector.DeviceSecurity.includeAllLoginGroups ()

Re-include all login groups. Use this if you have already excluded some login groups and want to include them all again without re-instantiating [DeviceSecurity](#) (which is discouraged)

Example usage:

```
// Include all login groups
device_security.includeAllLoginGroups();
```

2.1.4.7 void com.mvnordic.mviddeviceconnector.DeviceSecurity.popDeviceSecurityListener ()

Remove the latest added listener from a [DeviceSecurity](#) instance.

Example usage:

```
device_security.popDeviceSecurityListener();
```

2.1.4.8 void com.mvnordic.mviddeviceconnector.DeviceSecurity.releaseDeviceRegistration ()

Release the device-side knowledge about MVID within the app's sandbox forcing a new login from the user.

Calling this method will not affect other apps using MVIDDeviceConnector.

Example usage:

```
// Release device-side knowlegde of MVID
device_security.releaseDeviceRegistration();
```

2.1.4.9 void com.mvnordic.mviddeviceconnector.DeviceSecurity.removeDeviceSecurityListener ([DeviceSecurityListener](#) *listener*)

Remove a listener from a [DeviceSecurity](#) instance.

Example usage:

```
device_security.removeDeviceSecurityListener(device_security_listener);
```

Parameters

<i>listener</i>	The listener instance to be removed
-----------------	-------------------------------------

2.1.4.10 void com.mvnordic.mviddeviceconnector.DeviceSecurity.setHorizontalDisplacement (int *pixels*)

Move the center point of the login UI n pixels to the left or right depending on the numeric sign.

Example usage:

```
// Horizontally displace the login 40 pixels to the left
device_security.setLoginCenterDisplacement (-40);
```

Parameters

<i>pixels</i>	The amount of pixels to displace the center with.
---------------	---

The documentation for this class was generated from the following file:

- src/com/mvnordic/mviddeviceconnector/DeviceSecurity.java

2.2 com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener Interface Reference

Classes

- class [MVIDResponse](#)

Public Member Functions

- void **onMVIDResponseReady** ([MVIDResponse](#) response)

2.2.1 Detailed Description

This is the Listener user's must implement in order to receive notifications about application login results.

Author

Jakob Simon-Gaarde

The documentation for this interface was generated from the following file:

- src/com/mvnordic/mviddeviceconnector/DeviceSecurity.java

2.3 com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener.MVID-Response Class Reference

Public Member Functions

- **MVIDResponse** (SimpleHttpRequest.SimpleHttpResponse resp)
- **MVIDResponse** (int [res_code](#), String [res_msg](#), int [service_result_flags](#))

Public Attributes

- int `res_code` = -1
- String `res_msg` = null
- int `service_result_flags` = 0
- String `access_identifier` = null
- Integer `request_id` = null
- boolean `has_access` = false

2.3.1 Detailed Description

The `MVIDResponse` class contains information about an application login attempt. It is passed via the `onMVID-ResponseReady()` method in an asynchronous manner. Notifications are invoked on the App's UI thread. So it is possible to inflict changes on the user interface.

Author

Jakob Simon-Gaarde

2.3.2 Member Data Documentation

2.3.2.1 String `com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener.MVIDResponse.access_identifier` = null

The access identifier which was checked

2.3.2.2 boolean `com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener.MVIDResponse.has_access` = false

true if the `access_identifier` is granted false if not

2.3.2.3 Integer `com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener.MVIDResponse.request_id` = null

The `doLogin` request ID of the check (returned from `DeviceService.doLogin()`)

2.3.2.4 int `com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener.MVIDResponse.res_code` = -1

The `res_code` is the Web Service result code. Possible result codes can be found at the web service online site: [DeviceSecurity](#)

2.3.2.5 String `com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener.MVIDResponse.res_msg` = null

The `res_msg` is a text message corresponding to the `res_code`.

2.3.2.6 int `com.mvnordic.mviddeviceconnector.DeviceSecurity.DeviceSecurityListener.MVIDResponse.service_result_flags` = 0

Bitwise combination of [ServiceResult](#) flags.

The documentation for this class was generated from the following file:

- `src/com/mvnordic/mviddeviceconnector/DeviceSecurity.java`

2.4 com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult Class Reference

Static Public Attributes

- static final int [Success](#) = 1
- static final int [InvalidMVSessionID](#) = 2
- static final int [AccessDenied](#) = 4
- static final int [ApplicationBorrowTimeExpired](#) = 8
- static final int [DeviceBorrowTimeExpired](#) = 16
- static final int [ServiceFault](#) = 32
- static final int [ServiceSuccess](#) = 64
- static final int [NetworkError](#) = 128

2.4.1 Detailed Description

Following values apply to the [DeviceSecurity](#) web service. In the descriptions below it is specified which methods in the [DeviceSecurity](#) class that can receive which results directly.

When using doLogin: both DeviceSecurity.registerDevice: and DeviceSecurity.applicationLogin: are candidates for being called, therefore all [ServiceResult](#) codes are bitwise candidates of the final login result, which is posted in the MVIDLoginResponseReady via the default notification center.

Read more about Apple's notification system here: [Notification Programming Topics](#).

2.4.2 Member Data Documentation

2.4.2.1 final int com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult.AccessDenied = 4 [static]

Application access denied.

- DeviceSecurity.applicationLogin:
 - The server was contacted and device hash is good, but the specific user mapped to the device does not have access to the application.

2.4.2.2 final int com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult.ApplicationBorrowTimeExpired = 8 [static]

Application borrow-time has expired.

- DeviceSecurity.applicationLogin:
 - Either the device or the MV-ID server is telling you that the time since last applicationLogin has expired the borrow-time allowed for the application.

2.4.2.3 final int com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult.DeviceBorrowTimeExpired = 16 [static]

Device borrow-time has expired.

- DeviceSecurity.applicationLogin:
 - Either the device or the MV-ID server is telling you that the time since last applicationLogin has expired the borrow-time allowed for the device registration. This means that the device is no longer mapped to a user on MV-ID and a login is required.

Note

If device has no device hash registered this [ServiceResult](#) is also returned.

2.4.2.4 `final int com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult.InvalidMVSessionID = 2` [static]

Invalid MV Session ID.

- DeviceSecurity.registerDevice:
 - The mv_session_id passed to registerDevice is invalid. Obvious reason would be that it is too old and therefore timed out on the server side.

2.4.2.5 `final int com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult.NetworkError = 128` [static]

Network error occurred.

- DeviceSecurity.registerDevice:
- DeviceSecurity.applicationLogin:
 - A network transport error has occurred. Probably no link or other kind of network related problem like missing DNS service. Service requests timeout after 2 seconds, so connection problems should not "hang" the application.

2.4.2.6 `final int com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult.ServiceFault = 32` [static]

Fault occurred while calling the Device Security service.

- DeviceSecurity.registerDevice:
- DeviceSecurity.applicationLogin:
 - Some fault occurred during the invocation of a service method. Info about the fault is available in the response object itself.

2.4.2.7 `final int com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult.ServiceSuccess = 64` [static]

ServiceSuccess means that the a service method invocation succeeded with no faults. This does not necessarily mean that a device was registered successfully or that access was granted to an application successfully.

2.4.2.8 `final int com.mvnordic.mviddeviceconnector.DeviceSecurity.ServiceResult.Success = 1` [static]

Successful operation.

- DeviceSecurity.registerDevice:
 - registerDevice method has been successfully queried.
- DeviceSecurity.applicationLogin:
 - The registered user has successfully queried the applicationLogin method

The documentation for this class was generated from the following file:

- src/com/mvnordic/mviddeviceconnector/DeviceSecurity.java

Index

access_identifier
 com::mvnordic::mviddeviceconnector::Device-
 Security::DeviceSecurityListener::MVID-
 Response, 9

AccessDenied
 com::mvnordic::mviddeviceconnector::Device-
 Security::ServiceResult, 10

addDeviceSecurityListener
 com::mvnordic::mviddeviceconnector::Device-
 Security, 5

ApplicationBorrowTimeExpired
 com::mvnordic::mviddeviceconnector::Device-
 Security::ServiceResult, 10

com.mvnordic.mviddeviceconnector.DeviceSecurity, 3

com.mvnordic.mviddeviceconnector.DeviceSecurity-
 DeviceSecurityListener, 8

com.mvnordic.mviddeviceconnector.DeviceSecurity-
 DeviceSecurityListener.MVIDResponse, 8

com.mvnordic.mviddeviceconnector.DeviceSecurity-
 ServiceResult, 10

com::mvnordic::mviddeviceconnector::DeviceSecurity
 addDeviceSecurityListener, 5
 DeviceSecurity, 5
 doLogin, 5, 6
 excludeLoginGroup, 6
 getMVSessionID, 7
 includeAllLoginGroups, 7
 popDeviceSecurityListener, 7
 releaseDeviceRegistration, 7
 removeDeviceSecurityListener, 7
 setHorizontalDisplacement, 8

com::mvnordic::mviddeviceconnector::DeviceSecurity:-
 DeviceSecurityListener::MVIDResponse
 access_identifier, 9
 has_access, 9
 request_id, 9
 res_code, 9
 res_msg, 9

com::mvnordic::mviddeviceconnector::DeviceSecurity:-
 ServiceResult
 AccessDenied, 10
 ApplicationBorrowTimeExpired, 10
 DeviceBorrowTimeExpired, 10
 InvalidMVSessionID, 11
 NetworkError, 11
 ServiceFault, 11
 ServiceSuccess, 11
 Success, 11

DeviceBorrowTimeExpired
 com::mvnordic::mviddeviceconnector::Device-
 Security::ServiceResult, 10

DeviceSecurity
 com::mvnordic::mviddeviceconnector::Device-
 Security, 5

doLogin
 com::mvnordic::mviddeviceconnector::Device-
 Security, 5, 6

excludeLoginGroup
 com::mvnordic::mviddeviceconnector::Device-
 Security, 6

getMVSessionID
 com::mvnordic::mviddeviceconnector::Device-
 Security, 7

has_access
 com::mvnordic::mviddeviceconnector::Device-
 Security::DeviceSecurityListener::MVID-
 Response, 9

includeAllLoginGroups
 com::mvnordic::mviddeviceconnector::Device-
 Security, 7

InvalidMVSessionID
 com::mvnordic::mviddeviceconnector::Device-
 Security::ServiceResult, 11

NetworkError
 com::mvnordic::mviddeviceconnector::Device-
 Security::ServiceResult, 11

popDeviceSecurityListener
 com::mvnordic::mviddeviceconnector::Device-
 Security, 7

releaseDeviceRegistration
 com::mvnordic::mviddeviceconnector::Device-
 Security, 7

removeDeviceSecurityListener
 com::mvnordic::mviddeviceconnector::Device-
 Security, 7

request_id
 com::mvnordic::mviddeviceconnector::Device-
 Security::DeviceSecurityListener::MVID-
 Response, 9

res_code
 com::mvnordic::mviddeviceconnector::Device-
 Security::DeviceSecurityListener::MVID-
 Response, 9

res_msg
 com::mvnordic::mviddeviceconnector::Device-
 Security::DeviceSecurityListener::MVID-
 Response, [9](#)

service_result_flags
 com::mvnordic::mviddeviceconnector::Device-
 Security::DeviceSecurityListener::MVID-
 Response, [9](#)

ServiceFault
 com::mvnordic::mviddeviceconnector::Device-
 Security::ServiceResult, [11](#)

ServiceSuccess
 com::mvnordic::mviddeviceconnector::Device-
 Security::ServiceResult, [11](#)

setHorizontalDisplacement
 com::mvnordic::mviddeviceconnector::Device-
 Security, [8](#)

Success
 com::mvnordic::mviddeviceconnector::Device-
 Security::ServiceResult, [11](#)