read file from assets

```
public class Utils {
    public static List<Message> getMessages() {
        //File file = new File("file:///android_asset/helloworld.txt");
        AssetManager assetManager = getAssets();
        InputStream ims = assetManager.open("helloworld.txt");
     }
}
```

I am using this code trying to read a file from assets. I tried two ways to do this. First, when use `File` I received `FileNotFoundException`, when using `AssetManager getAssets()` method isn't recognized. Is there any solution here?

android

edited Mar 28 '17 at 8:31 — HpTerm — 6,240 12 43 64

asked Mar 3 '12 at 8:45 — fish40 — 2,573 16 42 64

12 Answers

Here is what I do in an activity for buffered reading extend/modify to match your needs

```
BufferedReader reader = null;
try {
    reader = new BufferedReader(
        new InputStreamReader(getAssets().open("filename.txt")));

    // do reading, usually loop until end of file reading
    String mLine;
    while ((mLine = reader.readLine()) != null) {
       //process line
       ...
    }
} catch (IOException e) {
    //log the exception
} finally {
    if (reader != null) {
         try {
             reader.close();
         } catch (IOException e) {
             //log the exception
         }
    }
}
```

EDIT : My answer is perhaps useless if your question is on how to do it outside of an activity. If your question is simply how to read a file from asset then the answer is above.

UPDATE :

To open a file specifying the type simply add the type in the InputStreamReader call as follow.

```
BufferedReader reader = null;
try {
    reader = new BufferedReader(
        new InputStreamReader(getAssets().open("filename.txt"), "UTF-8"));

    // do reading, usually loop until end of file reading
    String mLine;
    while ((mLine = reader.readLine()) != null) {
       //process line
       ...
    }
} catch (IOException e) {
    //log the exception
} finally {
    if (reader != null) {
         try {
             reader.close();
         } catch (IOException e) {
             //log the exception
         }
    }
}
```

EDIT

As @Stan says in the comment, the code I am giving is not summing up lines. `mLine` is replaced every pass. That's why I wrote `//process line`. I assume the file contains some sort of data (i.e a contact list) and each line should be processed separately.

In case you simply want to load the file without any kind of processing you will have to sum up `mLine` at each pass using `StringBuilder()` and appending each pass.

ANOTHER EDIT

According to the comment of @Vincent I added the `finally` block.

Also note that in Java 7 and upper you can use `try-with-resources` to use the `AutoCloseable` and `Closeable` features of recent Java.

CONTEXT

In a comment @LunarWatcher points out that `getAssets()` is a `class` in `context`. So, if you call it outside of an `activity` you need to refer to it and pass the context instance to the activity.

```
ContextInstance.getAssets();
```

This is explained in the answer of @Maneesh. So if this is useful to you upvote his answer because that's him who pointed that out.

edited Jan 28 at 2:24 — Ali Sharabiani — 4,014 8 38 64

answered Mar 3 '12 at 8:53 — HpTerm — 6,240 12 43 64

2 @Stan, then write about it in the comments and let the author to decide if they'd like to update it. Edits are for improving clarity, not changing meaning. Code revisions should always be posted as comments first. – KyleMit Jan 16 '14 at 18:39

2 Your code doesn't guaranty to close the stream and free the resource in a timely manner. I recommend you to use `finally {reader.close();}` . – Vincent Apr 10 '14 at 4:17

2 I think it's useful to point out that the code above shows an error in ADT - the "reader.close();" line needs to be put in another try-catch block. Check this thread: stackoverflow.com/questions/8981589/… :) – JakeP Jul 7 '14 at 10:36

1 getAssets is a class in Context, so for usage outside activity a call to Context needs to be made. So outside an activity, it will be something like `context.getAssets(.....)` – Zoe Jun 11 '17 at 12:06

1 As per your update (thanks for adding that btw), having context in static fields is a memory leak. This should be used with caution and be properly cleaned up. Otherwise you end up with a memory leak that can have a great impact on the app. – Zoe Jun 29 '17 at 20:47

|

Using Kotlin, u can do the following to read file from asset in Android:

```
try {
        val inputStream:InputStream = assets.open("helloworld.txt")
        val inputString = inputStream.bufferedReader().use{it.readText()}
        Log.d(TAG,inputString)
    } catch (e:Exception){
        Log.d(TAG, e.toString())
    }
```

answered Jan 16 at 4:00 — Vamsi Tallapudi — 819 7 12

You can load the content from the file. Consider the file is present in asset folder.

```
public static InputStream loadInputStreamFromAssetFile(Context context, String
fileName){
    AssetManager am = context.getAssets();
    try {
        InputStream is = am.open(fileName);
        return is;
    } catch (IOException e) {
        e.printStackTrace();
    }
    return null;
}

public static String loadContentFromFile(Context context, String path){
    String content = null;
    try {
        InputStream is = loadInputStreamFromAssetFile(context, path);
        int size = is.available();
        byte[] buffer = new byte[size];
        is.read(buffer);
        is.close();
        content = new String(buffer, "UTF-8");
    } catch (IOException ex) {
        ex.printStackTrace();
        return null;
    }
    return content;
}
```

Now you can get the content by calling the function as follow

```
String json= FileUtil.loadContentFromFile(context, "data.json");
```

Considering the data.json is stored at Application\app\src\main\assets\data.json

answered May 25 '17 at 10:16

Siddarth Kanted
1,147 8 10

---

cityfile.txt

```
    public void getCityStateFromLocal() {
        AssetManager am = getAssets();
        InputStream inputStream = null;
        try {
            inputStream = am.open("city_state.txt");
        } catch (IOException e) {
            e.printStackTrace();
        }
        ObjectMapper mapper = new ObjectMapper();
        Map<String, String[]> map = new HashMap<String, String[]>();
        try {
            map = mapper.readValue(getStringFromInputStream(inputStream), new
TypeReference<Map<String, String[]>>() {
            });
        } catch (IOException e) {
            e.printStackTrace();
        }
        ConstantValues.arrayListStateName.clear();
        ConstantValues.arrayListCityByState.clear();
        if (map.size() > 0)
        {
            for (Map.Entry<String, String[]> e : map.entrySet()) {
                CityByState cityByState = new CityByState();
                String key = e.getKey();
                String[] value = e.getValue();
                ArrayList<String> s = new ArrayList<String>
(Arrays.asList(value));
                ConstantValues.arrayListStateName.add(key);
                s.add(0,"Select City");
                cityByState.addValue(s);
                ConstantValues.arrayListCityByState.add(cityByState);
            }
        }
        ConstantValues.arrayListStateName.add(0,"Select States");
    }
 // Convert InputStream to String
    public String getStringFromInputStream(InputStream is) {
        BufferedReader br = null;
        StringBuilder sb = new StringBuilder();
        String line;
        try {
            br = new BufferedReader(new InputStreamReader(is));
            while ((line = br.readLine()) != null) {
                sb.append(line);
            }
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (br != null) {
                try {
                    br.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }

        return sb + "";

    }
```

answered May 9 '17 at 13:50

tej shah
1,420 10 19

In MainActivity.java

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

        TextView tvView = (TextView) findViewById(R.id.tvView);

        AssetsReader assetsReader = new AssetsReader(this);
        if(assetsReader.getTxtFile(your_file_title)) != null)
        {
            tvView.setText(assetsReader.getTxtFile(your_file_title)));
        }
    }
```

Also, you can create separate class that does all the work

```
public class AssetsReader implements Readable{

    private static final String TAG = "AssetsReader";
```

```
    private AssetManager mAssetManager;
    private Activity mActivity;

    public AssetsReader(Activity activity) {
        this.mActivity = activity;
        mAssetManager = mActivity.getAssets();
    }

    @Override
    public String getTxtFile(String fileName)
    {
        BufferedReader reader = null;
        InputStream inputStream = null;
        StringBuilder builder = new StringBuilder();

        try{
            inputStream = mAssetManager.open(fileName);
            reader = new BufferedReader(new InputStreamReader(inputStream));

            String line;

            while((line = reader.readLine()) != null)
            {
                Log.i(TAG, line);
                builder.append(line);
                builder.append("\n");
            }
        } catch (IOException ioe){
            ioe.printStackTrace();
        } finally {

            if(inputStream != null)
            {
                try {
                    inputStream.close();
                } catch (IOException ioe){
                    ioe.printStackTrace();
                }
            }

            if(reader != null)
            {
                try {
                    reader.close();
                } catch (IOException ioe)
                {
                    ioe.printStackTrace();
                }
            }
        }
        Log.i(TAG, "builder.toString(): " + builder.toString());
        return builder.toString();
    }
}
```

In my opinion it's better to create an interface, but it's not neccessary

```
public interface Readable {
    /**
     * Reads txt file from assets
     * @param fileName
     * @return string
     */
    String getTxtFile(String fileName);
}
```

edited Oct 11 '16 at 16:31

answered Oct 11 '16 at 16:24

Volodimir Shalashenko

65 6

If you use other any class other than Activity, you might want to do like,

```
BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(
YourApplication.getInstance().getAssets().open("text.txt"), "UTF-8"));
```

answered Jun 25 '16 at 6:46

Zhaolong Zhong

276 3 6

Here is a method to read a file in assets:

```
/**
 * Reads the text of an asset. Should not be run on the UI thread.
 *
 * @param mgr
 *            The {@link AssetManager} obtained via {@link Context#getAssets()}
 * @param path
 *            The path to the asset.
 * @return The plain text of the asset
 */
public static String readAsset(AssetManager mgr, String path) {
    String contents = "";
    InputStream is = null;
    BufferedReader reader = null;
```

```
    try {
        is = mgr.open(path);
        reader = new BufferedReader(new InputStreamReader(is));
        contents = reader.readLine();
        String line = null;
        while ((line = reader.readLine()) != null) {
            contents += '\n' + line;
        }
    } catch (final Exception e) {
        e.printStackTrace();
    } finally {
        if (is != null) {
            try {
                is.close();
            } catch (IOException ignored) {
            }
        }
        if (reader != null) {
            try {
                reader.close();
            } catch (IOException ignored) {
            }
        }
    }
    return contents;
}
```

edited Apr 14 '16 at 10:05

answered Jan 19 '15 at 9:29

Jared Rummler
26k 12 88 110

Better late than never.

I had difficulties reading files line by line in some circumstances. The method below is the best I found, so far, and I recommend it.

Usage: `String yourData = LoadData("YourDataFile.txt");`

Where ***YourDataFile.txt*** is assumed to reside in ***assets/***

```
public String LoadData(String inFile) {
        String tContents = "";

    try {
        InputStream stream = getAssets().open(inFile);

        int size = stream.available();
        byte[] buffer = new byte[size];
        stream.read(buffer);
        stream.close();
        tContents = new String(buffer);
    } catch (IOException e) {
        // Handle exceptions here
    }

    return tContents;

}
```

EDIT. Apparent issue: this function might return this string:

'android.content.res.AssetManager$AssetInputStream@[code]'

instead of the file contents. I am unable yet to reproduce the problem. Until I update my answer when I get what the problem is, consider the code above 'probably unreliable'.

edited Oct 26 '14 at 18:40

answered Jul 12 '13 at 9:21

Florin Mircea
540 8 16

My return string is android.content.res.AssetManager$AssetInputStream@4195dfa0 .. – Boldijar Paul Oct 18 '14 at 7:31

same here, res.AssetManager$AssetInputStream@.... any particular reason why it is returning this? – Bigs Oct 25 '14 at 20:25

1 works fine for me – Fabian Jan 5 '16 at 13:18

You double-allocate memory - first for the buffer and then for the String. Does not work for bigger files. – JaakL Jun 28 '17 at 10:10

1 perfect way to allocate size to buffer `stream.available()` . – Kasim Rangwala Feb 1 at 10:45

`getAssets()` method will work when you are calling inside the Activity class.

If you calling this method in non-Activity class then you need to call this method from Context which is passed from Activity class. So below is the line by you can access the method.

```
ContextInstance.getAssets();
```

`ContextInstance` may be passed as this of Activity class.

edited Jun 12 '14 at 15:03

akauppi
7,220 6 48 66

answered Mar 3 '12 at 8:54

Maneesh
4,779 4 30 52

```
public String ReadFromfile(String fileName, Context context) {
    StringBuilder returnString = new StringBuilder();
    InputStream fIn = null;
    InputStreamReader isr = null;
    BufferedReader input = null;
    try {
        fIn = context.getResources().getAssets()
                .open(fileName, Context.MODE_WORLD_READABLE);
        isr = new InputStreamReader(fIn);
        input = new BufferedReader(isr);
        String line = "";
        while ((line = input.readLine()) != null) {
            returnString.append(line);
        }
    } catch (Exception e) {
        e.getMessage();
    } finally {
        try {
            if (isr != null)
                isr.close();
            if (fIn != null)
                fIn.close();
            if (input != null)
                input.close();
        } catch (Exception e2) {
            e2.getMessage();
        }
    }
    return returnString.toString();
}
```

edited Nov 7 '13 at 18:08

Intrications
13.6k 9 40 49

answered Apr 6 '12 at 12:32

swathi
575 1 4 7

You would think that if you close the BufferedReader, than it would have to automatically close the InputStreanReader and InputStream too. Because what it you don't create a handle for those, e.g. `input = new BufferedReader(new InputStreamReader(fIn));` . – trans May 28 '14 at 13:07

1 I would suggest creating separate try/catch blocks for closing all of your resources at the end; rather than lumping them all into one - as it may leave other resources unclosed if a prior attempt to close another resource throws an exception. – Reece Nov 18 '16 at 4:03

```
AssetManager assetManager = getAssets();
InputStream inputStream = null;
try {
    inputStream = assetManager.open("helloworld.txt");
}
catch (IOException e){
    Log.e("message: ",e.getMessage());
}
```

answered Mar 3 '12 at 8:56

Siva Charan
14.9k 7 42 74

`getAssets()`

is only works **in Activity** in other any class you have to use `Context` for it.

Make a **constructor for Utils** class pass reference of activity (ugly way) or context of application as a parameter to it. Using that use getAsset() in your Utils class.

answered Mar 3 '12 at 8:52

user370305
84.1k 16 140 138

It works on anything that's a subclass of Context, of which Activity is one of many. – Jeremy Logan Aug 22 '13 at 20:59

Just noted I have written `Context` . – user370305 Aug 23 '13 at 7:43

@user370305 do you know, how I can convert InputStream into FileInputStream? – Mike Herasimov May 18 '15 at 13:17