

Title: Facing FileNotFoundException while accessing JSON File in classpath using java in docker containers(SpringBootApplication)

Post Body:

I'm Facing a FileNotFoundException while loading a JSON file which is in class path of Java jar using docker containers, it is a Spring-Boot application. This JSON file is available in resource folder . I'm Able to see the JSON file in docker under ./target/classes/ path.

```
Resource resource = resourceLoader.getResource('classpath:folderNm/file.json'); HashMap<String, String> headerMapping = (HashMap<String, String>) resource.getHeaders().get("headerMapping");
```

But I get this exception:

```
java.io.FileNotFoundException: class path resource [folderNm/file.json] cannot be resolved to absolute file path because it does not have a valid prefix
```

I tried

```
-> resource.getFile().getPath(); -> resource.getFile().getCanonicalPath(); -> './target/classes/folderName/fileName' (hardcoded FilePath location) -> '/app.jar!/folderNm/file.json' (hardcoded FilePath location)
```

```
InputStream inputStream = getClass().getResourceAsStream('xyz.json'); BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
```

none of the way above are running. Kindly Suggest a way to resolve this issue.

Accepted Answer: None

Highest Rated Answer:

Assuming that in the maven structure (you are using maven, right?) your file is located at `src/main/resources/folderName/file.json`, the path you pass to `getResourceAsStream` should be **`/folderName/file.json`**

Actually everything is explained in [javadocs](#):

Before delegation, an absolute resource name is constructed from the given resource name using this algorithm:

- If the name begins with a '/' ('\''), then the absolute name of the resource is the portion of the name following the '/'.
For example, if the name is `/foo/bar/baz.json`, the absolute name is `foo/bar/baz.json`.
- Otherwise, the absolute name is of the following form: `modified_package_name/name`

Where the `modified_package_name` is the package name of this object with '/' substituted for '.' ('\'').

Basically if you skip the '/' in front, it looks for the *folderName* inside the package of your class. Following code works fine for me:

```
InputStream inputStream = StackTest.class.getResourceAsStream('/folderName/file.json');    BufferedReader br = new BufferedReader(inputStream);
```

assuming I have my *file.json* in `src/main/resources/folderName`. I don't think it has anything to do with docker. Btw, I think you could make use of [Apache Commons IOUtils.toString](#) that helps turning InputStream into a String.