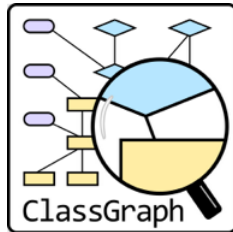


# ClassGraph 项目文件读取

**classgraph/classgraph: An uber-fast p...**

An uber-fast parallelized Java classpath sc...

github.com



```
import com.check.entity.Xml;
import java.nio.charset.StandardCharsets;
import java.util.ArrayList;
import java.util.List;
import java.util.regex.Pattern;
import io.github.classgraph.ClassGraph;
import io.github.classgraph.Resource;
import io.github.classgraph.ScanResult;

public class ResourceUtil {
    public static List<Xml> getXMLs(String pattern) {
        List<Xml> xmlList = new ArrayList<>();

        try (ScanResult scanResult = new ClassGraph().scan()) {
            scanResult.getResourcesMatchingPattern(Pattern.compile(pattern))
                .forEachByteArrayIgnoringIOException((Resource res, byte[]
fileContent) ->
                    xmlList.add(new Xml(res.getPath(), new String(fileContent,
StandardCharsets.UTF_8))));
        }

        return xmlList;
    }
}
```

不需要显式的IO操作，如下

```
public List<String> getXMLs(String path) {
    List<String> filenames = new ArrayList<>();

    try (
        InputStream in = getResourceAsStream(path);
        BufferedReader br = new BufferedReader(new
```

```

InputStreamReader(in))) {
    String resource;

    while ((resource = br.readLine()) != null) {
        filenames.add(resource);
    }
} catch (Exception e) {
    e.printStackTrace();
    return List.of();
}

return filenames.stream().filter(f ->
f.endsWith(".xml")).collect(Collectors.toList());

}

```

```

private InputStream getResourceAsStream(String resource) {
    final InputStream in
        = getContextClassLoader().getResourceAsStream(resource);

    return in == null ? getClass().getResourceAsStream(resource) : in;
}

```

```

private ClassLoader getContextClassLoader() {
    return Thread.currentThread().getContextClassLoader();
}

```