# Software Engineering Large Practical

# Feedback on Part 1 of the SELP

Stephen Gilmore

School of Informatics, University of Edinburgh

November 6th, 2013

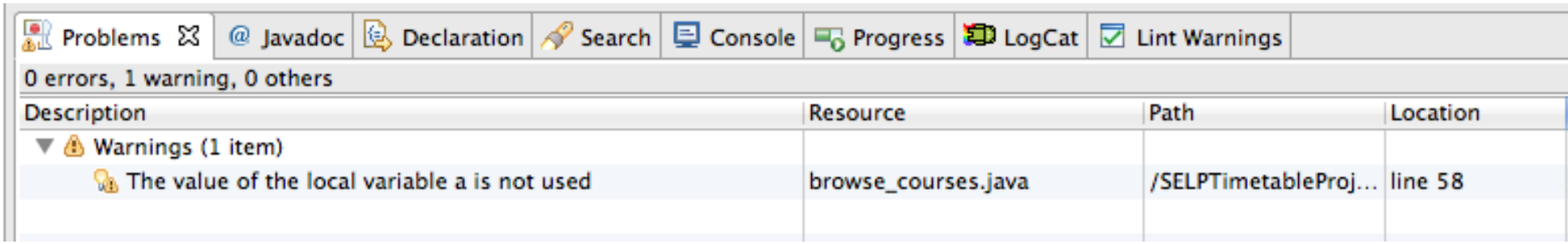
# Feedback on Part 1 of the SELP

The submissions for Part 1 of the SELP are necessarily incomplete. Some of them contained problems which obviously should be fixed before Part 2 is submitted. Nonetheless, these are pointed out below in case some of them were not thought to be problems which needed to be fixed.

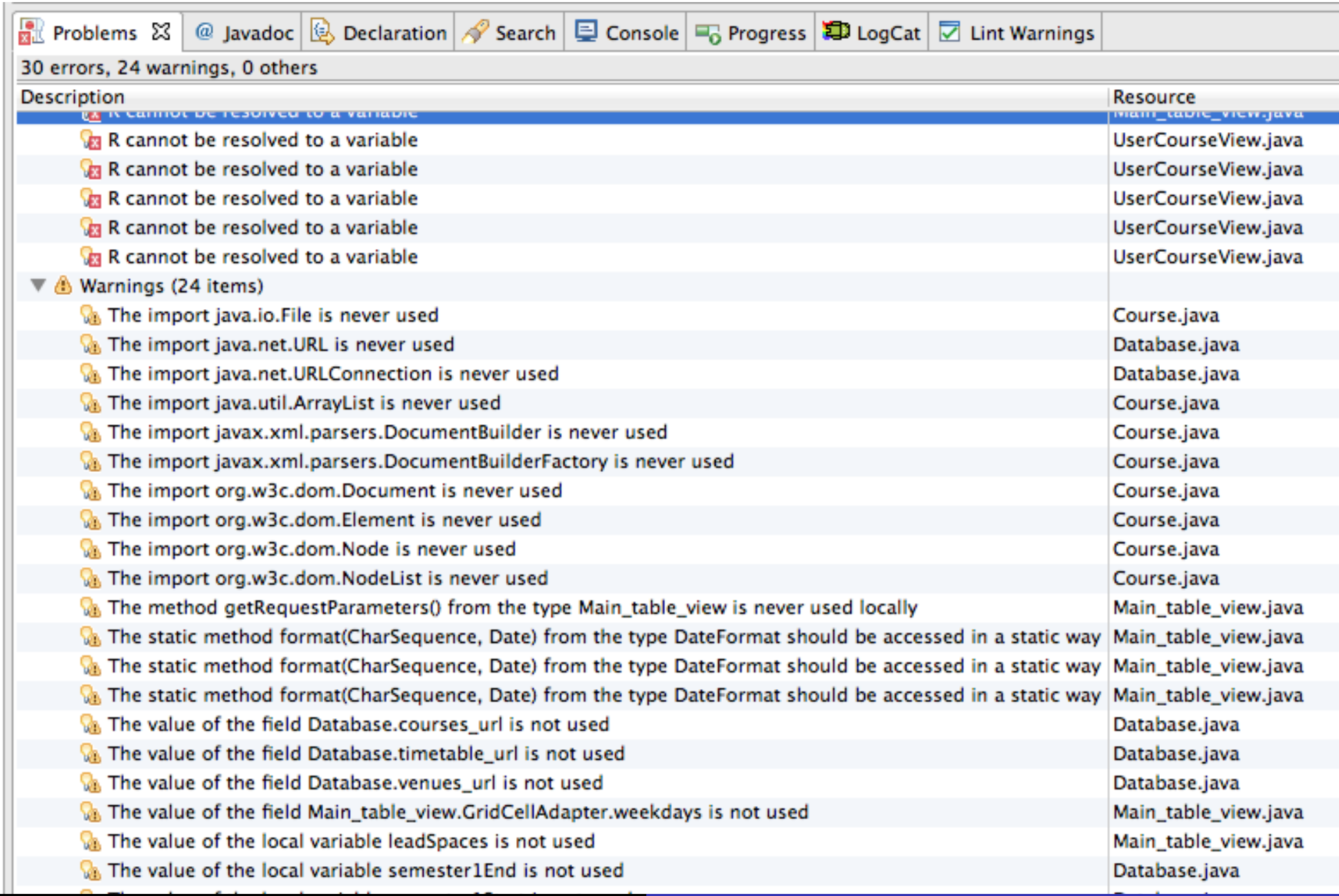
# Java problems

Some submitted projects had Java problems. These are often relatively easy to fix, and simply require knowledge of the Java language and its libraries. The fixes which need to be applied here are general ones which would be relevant for any Java application: there is nothing Android-specific about them.

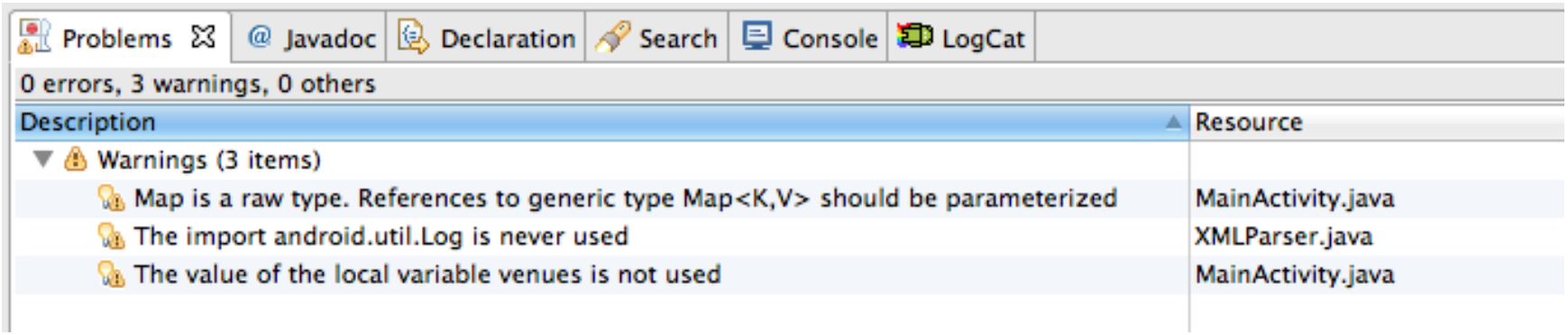
# Java: Unused variable errors



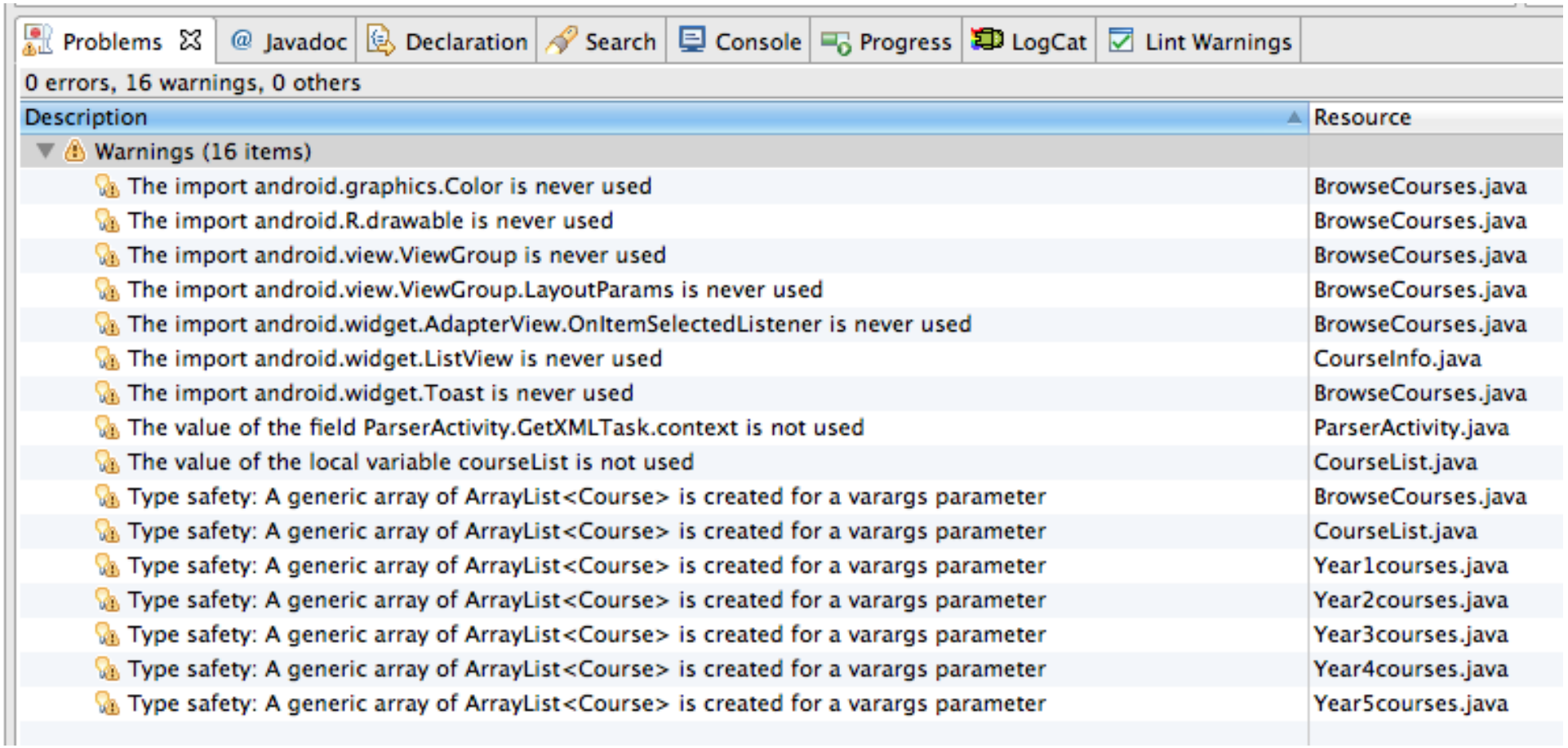
# Java: static analysis errors



# Java: should use generic classes correctly, don't use raw types



# Java: Type safety warnings on generic parameters

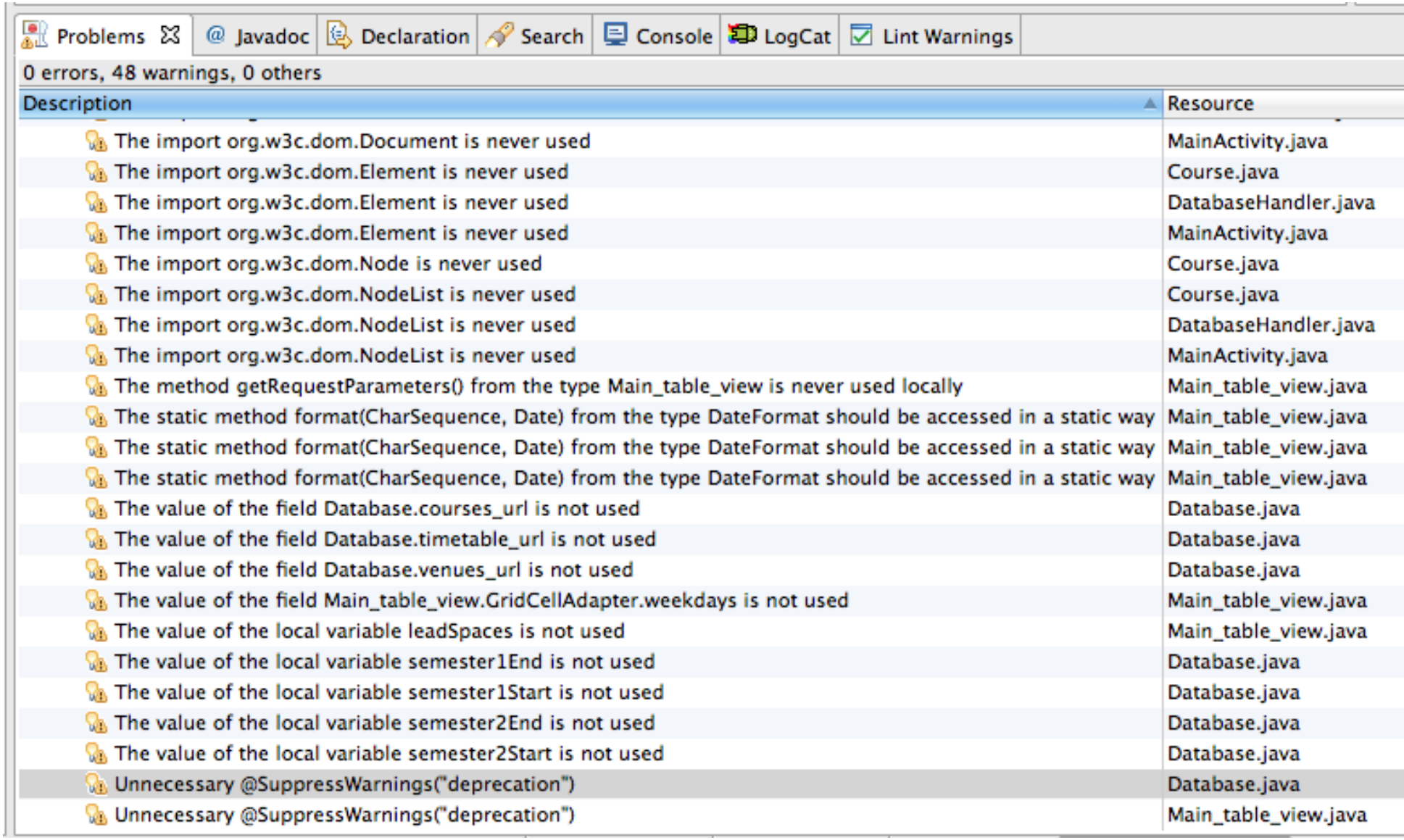


Problems | Javadoc | Declaration | Search | Console | Progress | LogCat | Lint Warnings

0 errors, 16 warnings, 0 others

| Description | Resource |
|---|---|
| Warnings (16 items) | |
| The import android.graphics.Color is never used | BrowseCourses.java |
| The import android.R.drawable is never used | BrowseCourses.java |
| The import android.view.ViewGroup is never used | BrowseCourses.java |
| The import android.view.ViewGroup.LayoutParams is never used | BrowseCourses.java |
| The import android.widget.AdapterView.OnItemSelectedListener is never used | BrowseCourses.java |
| The import android.widget.ListView is never used | CourseInfo.java |
| The import android.widget.Toast is never used | BrowseCourses.java |
| The value of the field ParserActivity.GetXMLTask.context is not used | ParserActivity.java |
| The value of the local variable courseList is not used | CourseList.java |
| Type safety: A generic array of ArrayList<Course> is created for a varargs parameter | BrowseCourses.java |
| Type safety: A generic array of ArrayList<Course> is created for a varargs parameter | CourseList.java |
| Type safety: A generic array of ArrayList<Course> is created for a varargs parameter | Year1courses.java |
| Type safety: A generic array of ArrayList<Course> is created for a varargs parameter | Year2courses.java |
| Type safety: A generic array of ArrayList<Course> is created for a varargs parameter | Year3courses.java |
| Type safety: A generic array of ArrayList<Course> is created for a varargs parameter | Year4courses.java |
| Type safety: A generic array of ArrayList<Course> is created for a varargs parameter | Year5courses.java |

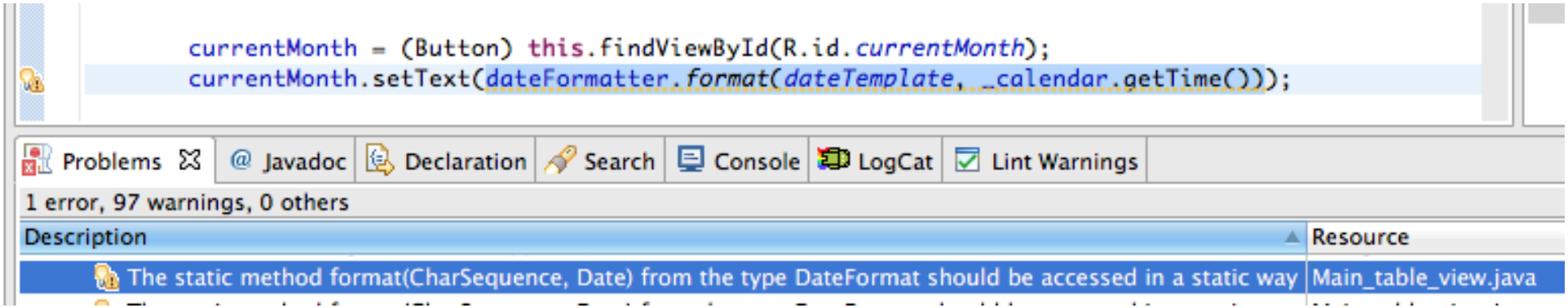
# Java: Suppressing warnings is not usually the right way to solve a problem, but don't suppress unnecessarily



# Java: avoid deprecation, don't write legacy code

```
});
dialog.show();
dialog.getWindow().setLayout(LayoutParams.FILL_PARENT, LayoutParams.FILL_PARENT);
} else
```

# Java: "Static method should be accessed in a static way"—use the class name, not the object name



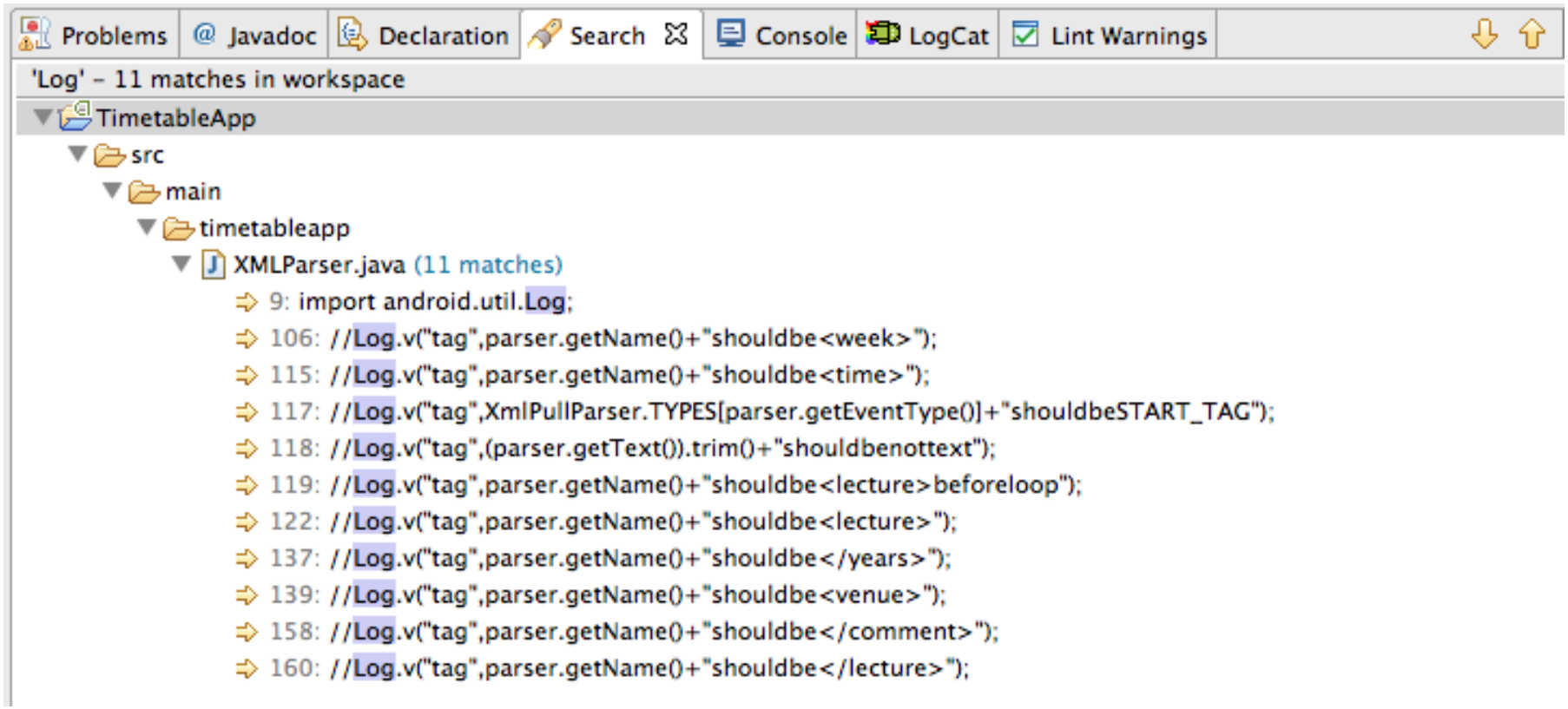
# Java: Exception handling—you should treat different cases differently

```
} catch (ParserConfigurationException e) {
    Log.e("Error: ", e.getMessage(), e);
    return null;
} catch (SAXException e) {
    Log.e("Error: ", e.getMessage(), e);
    return null;
} catch (IOException e) {
    Log.e("Error: ", e.getMessage(), e);
    return null;
}
```

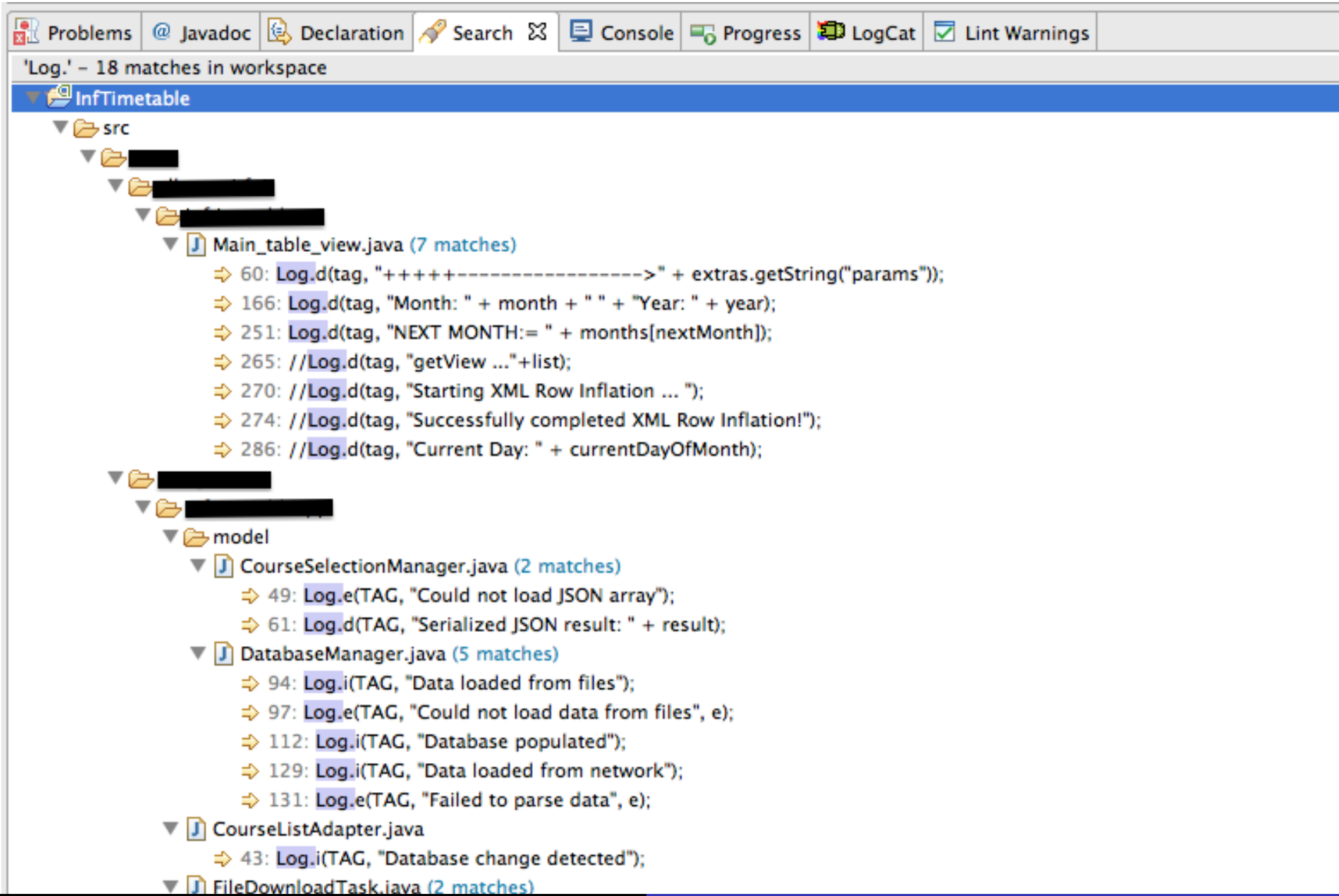
# Logging problems

The practical asks you to include logging in your app. Logging has many purposes, including debugging, and giving information messages and error messages. The value which is brought by logging depends on how useful the text string supplied is, and whether it can be traced back to the class which generated it.

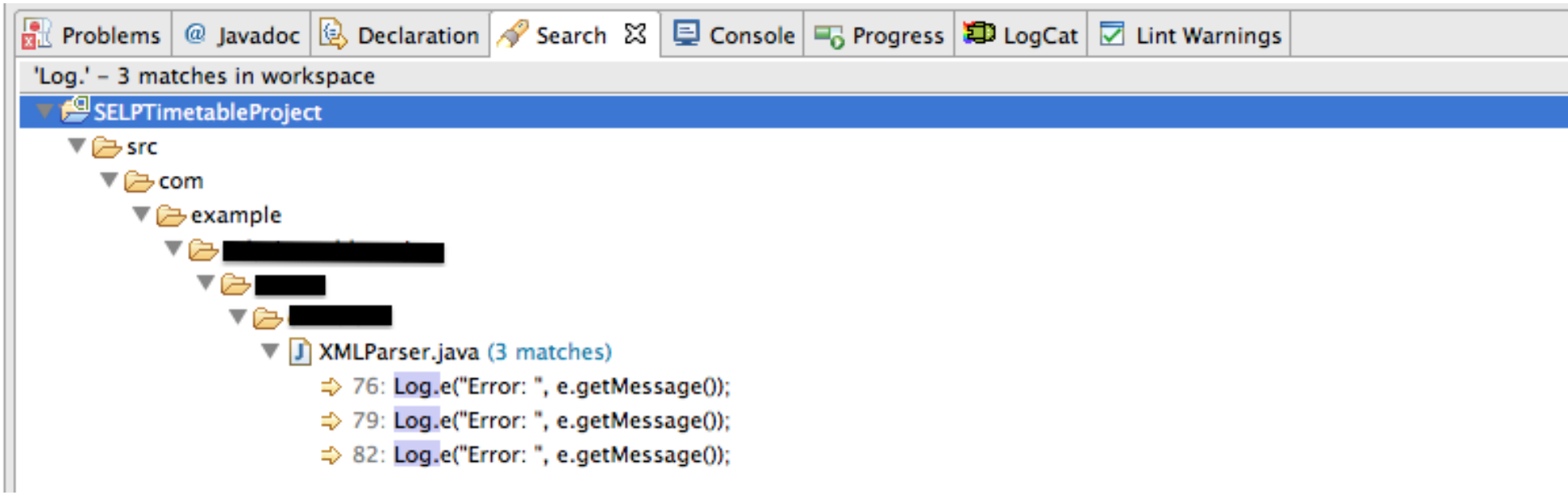
# Logging: had logging, took it out when the code was working



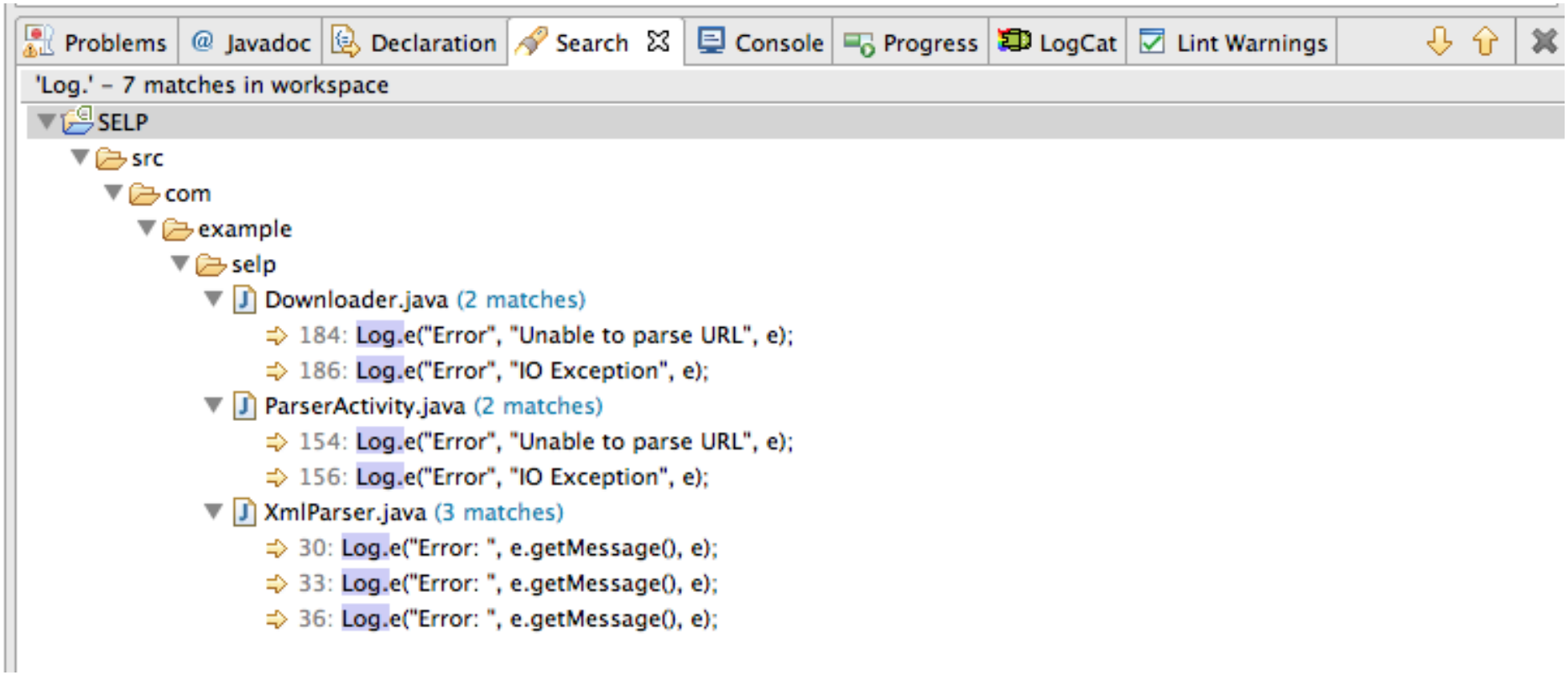
# Logging: Commenting out the debug statements



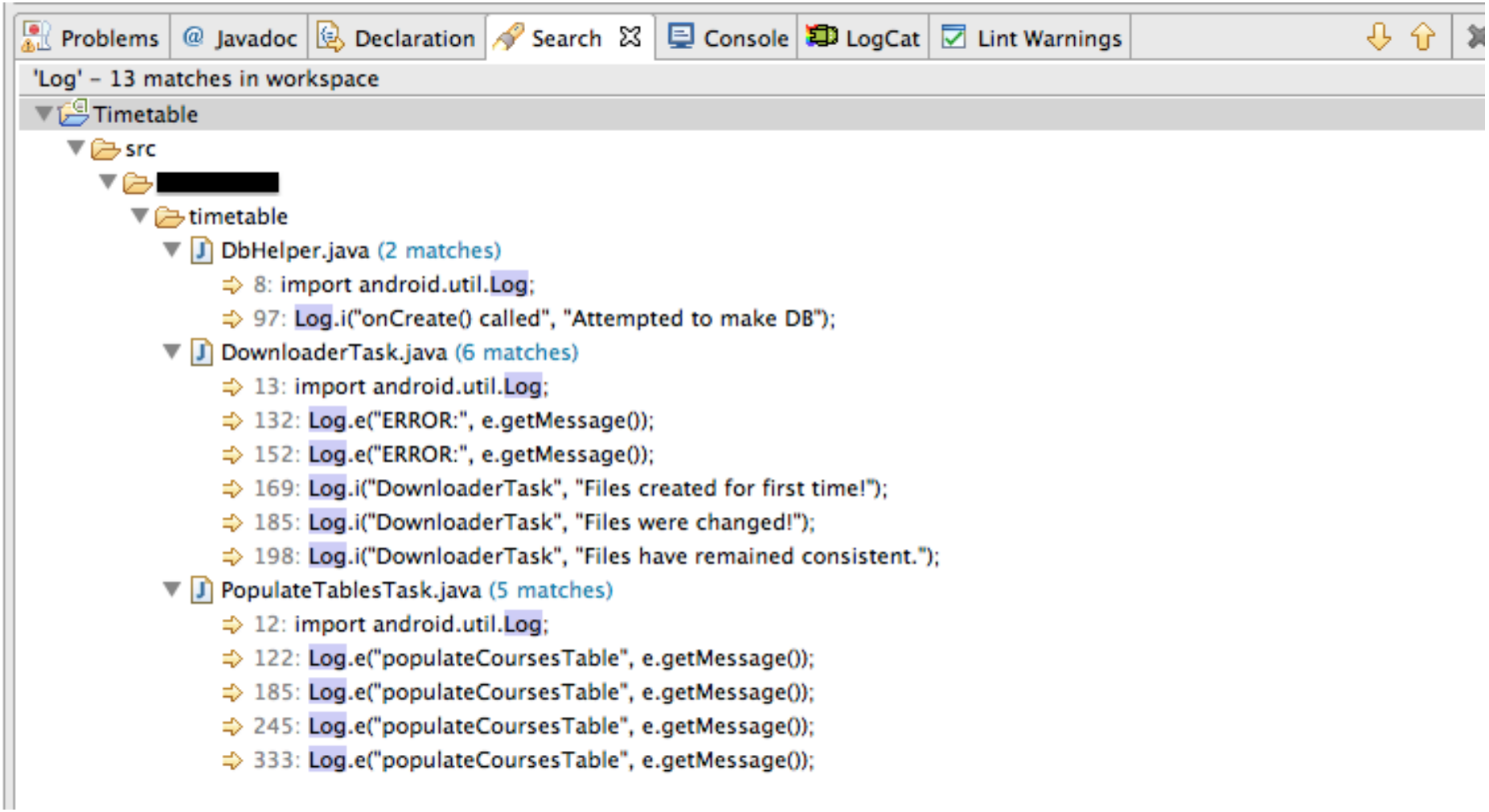
# Logging: Not enough logging



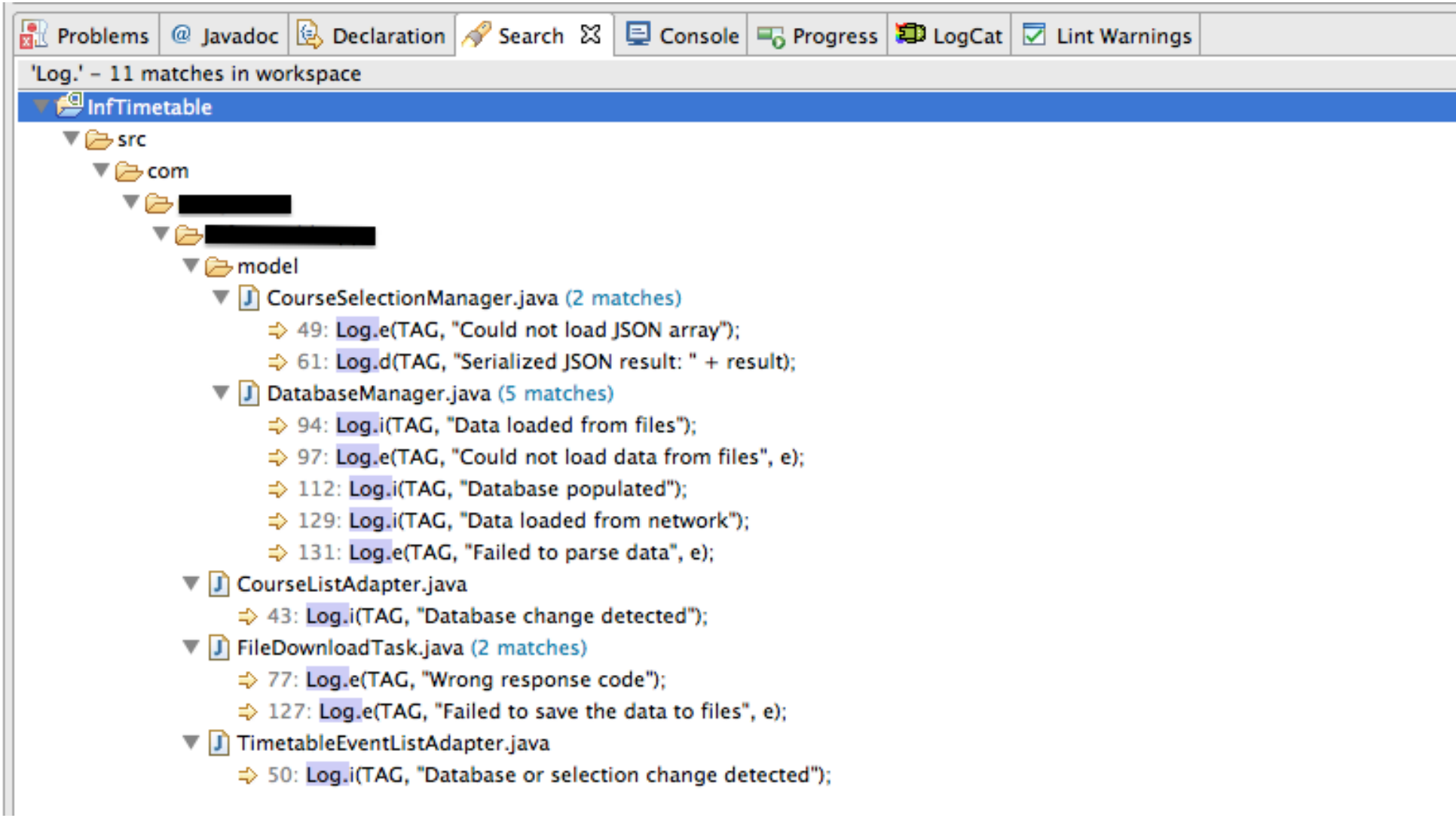
# Logging: Use TAG to identify the class which is logging



# Logging: making progress



# Logging: Good style



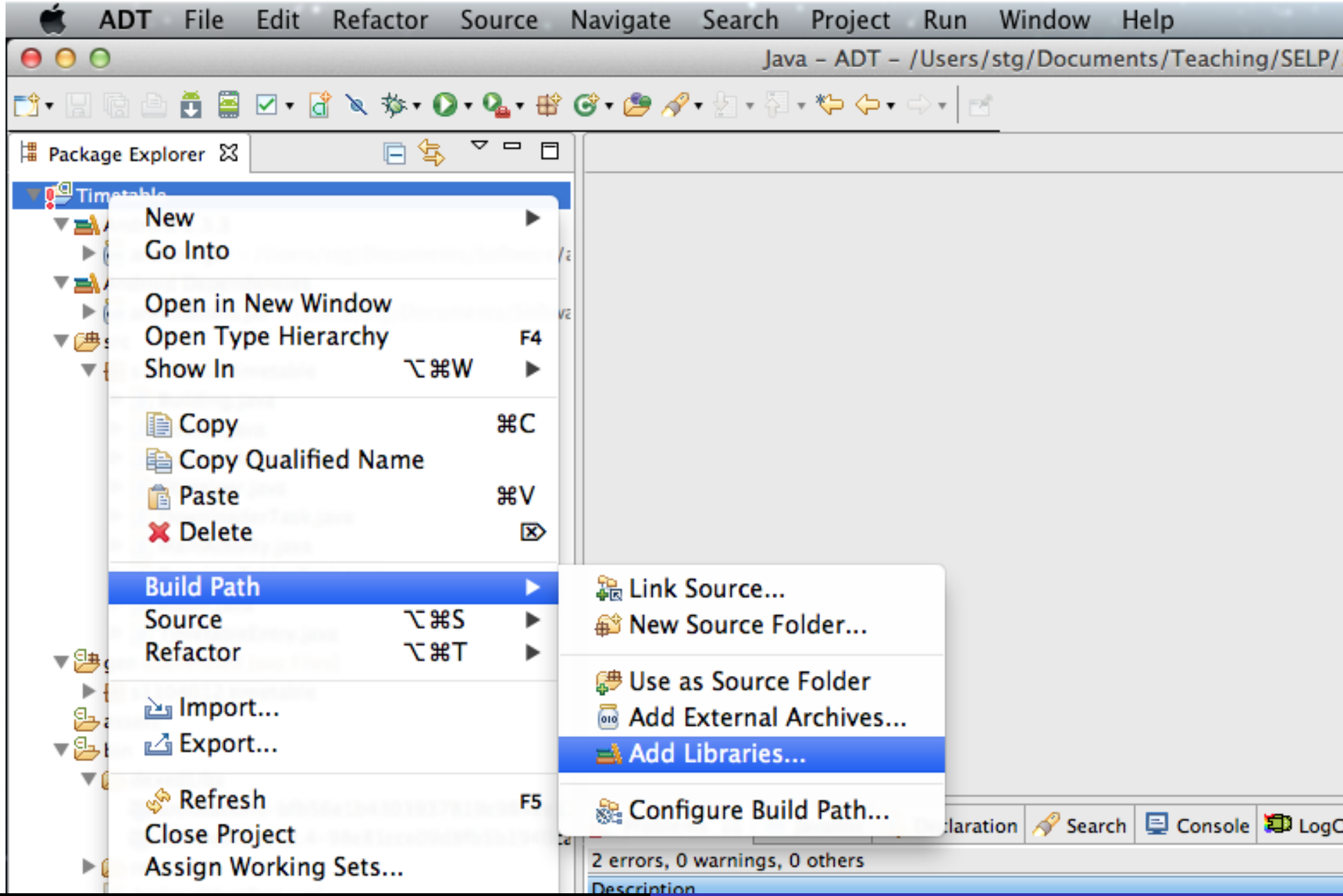
# Structure problems

Android projects must be structured correctly to allow them to be compiled by Eclipse.

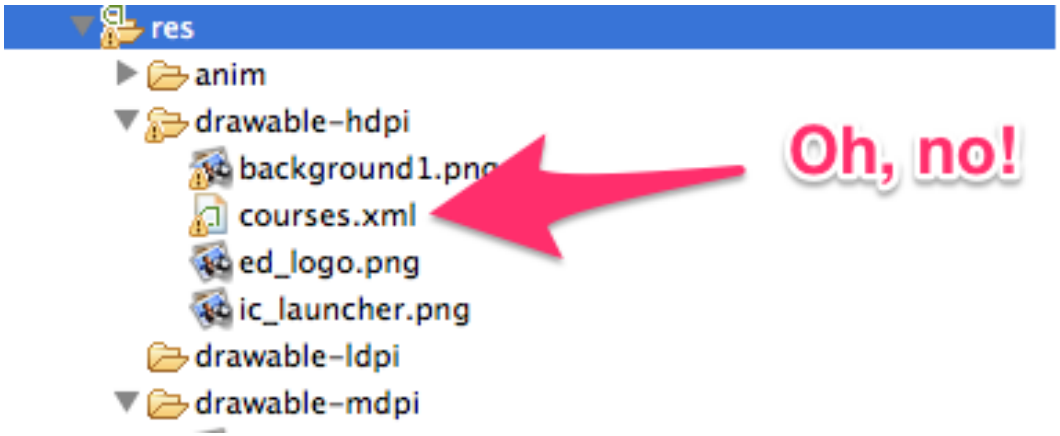
- Java files have their place (in src).
- XML files have their place (in res/layout).
- Image files have their place (in res/drawable).

Drag-and-drop errors which move files into the wrong location can prevent your project from compiling at all.

# Structure: add libraries to your project



# Structure: courses.xml in the wrong place; this shouldn't be in res/drawable



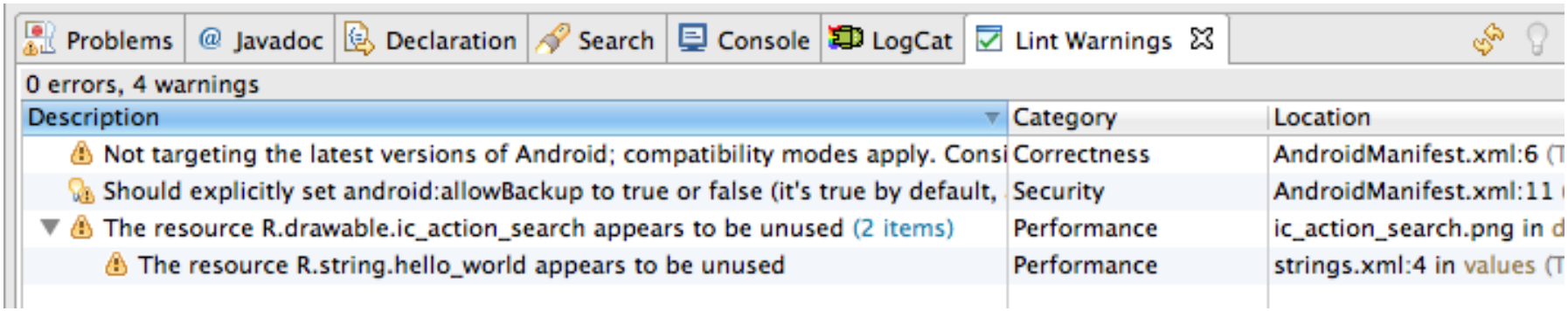
# Android Lint

Eclipse provides a service to debug the non-Java part of an Android project, called Android Lint.

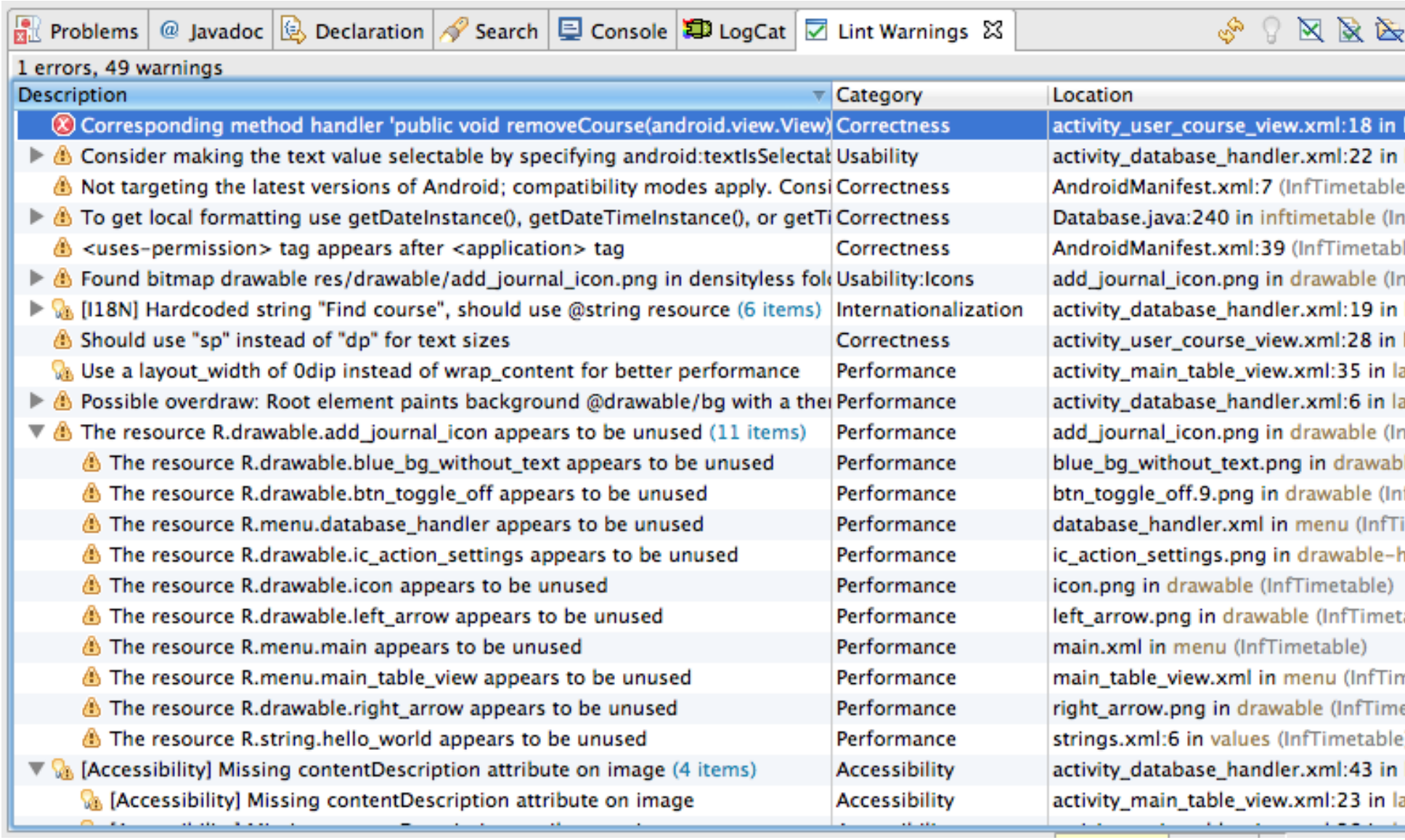
Android Lint looks for problems and bad style in XML parts of the project, and lack of integration with the Java part. Issues checked include:

- Performance,
- Correctness,
- Usability,
- Security,
- Internationalisation, and others.

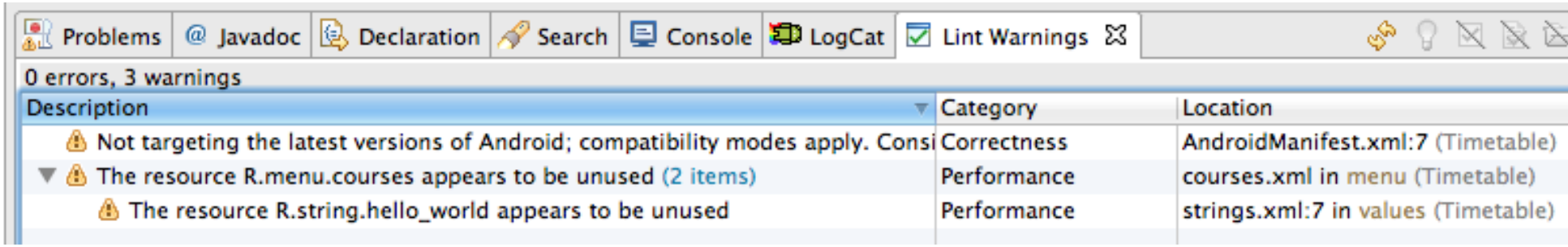
# Lint: security warnings



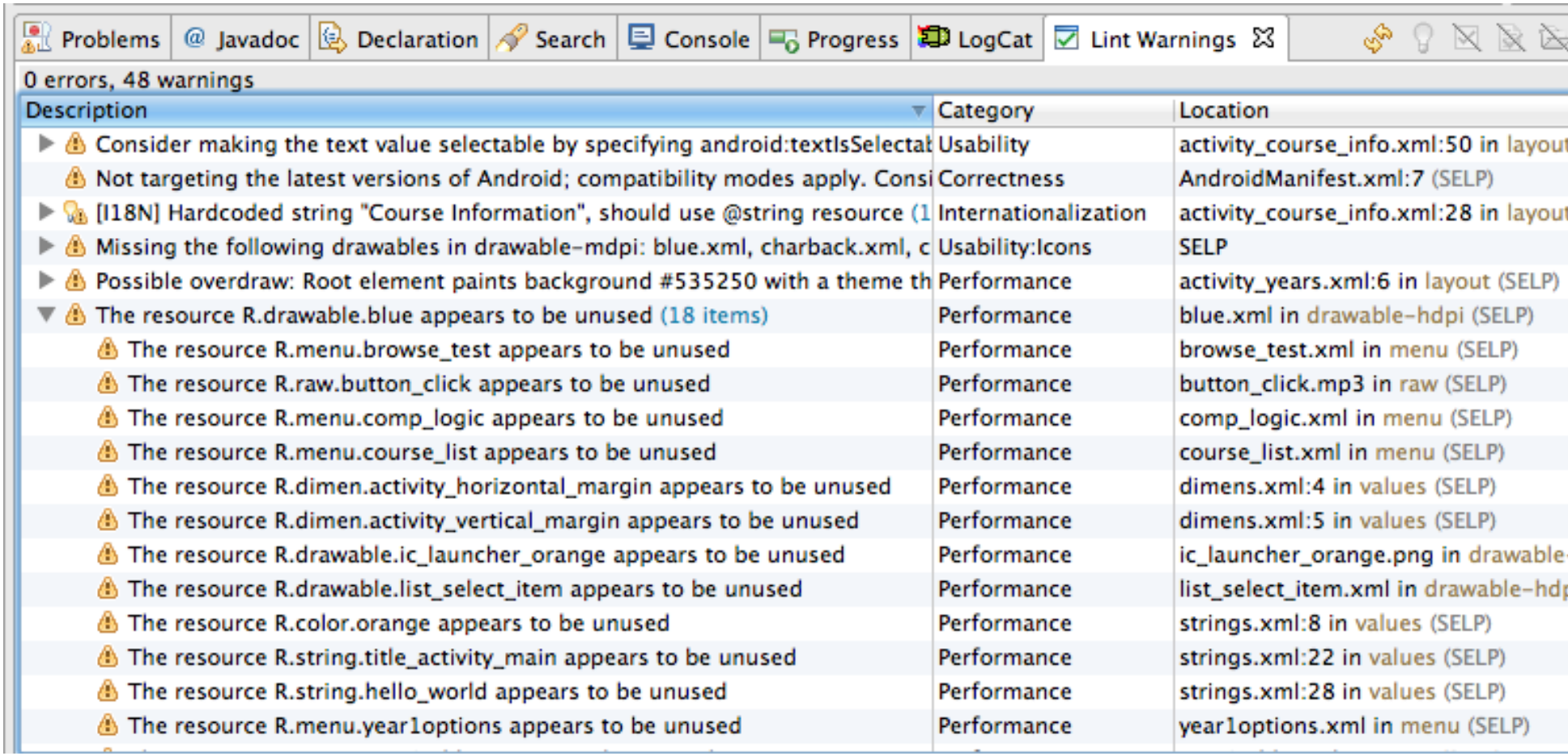
# Lint: Error; missing method handler



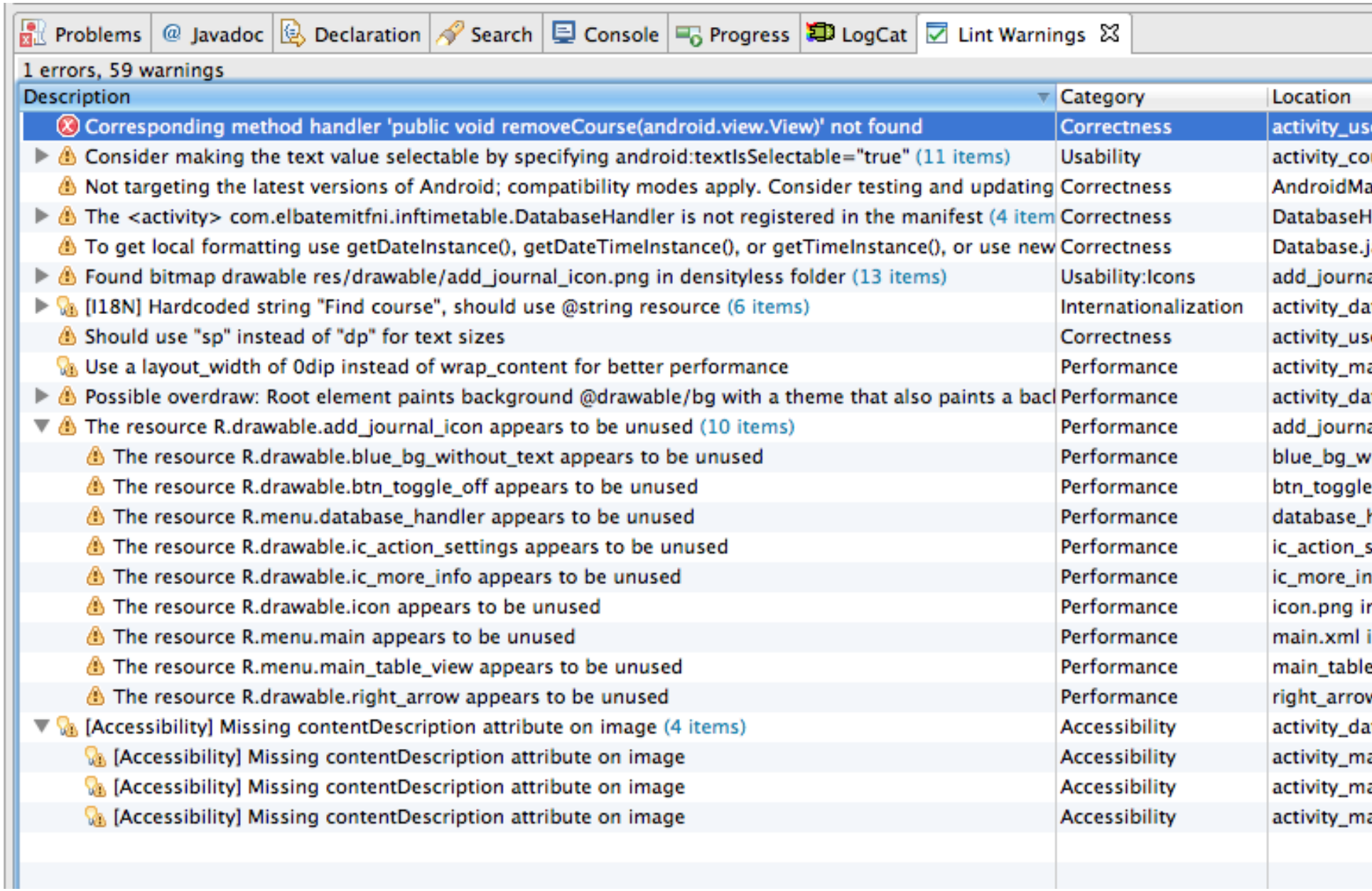
# Lint: a few minor problems



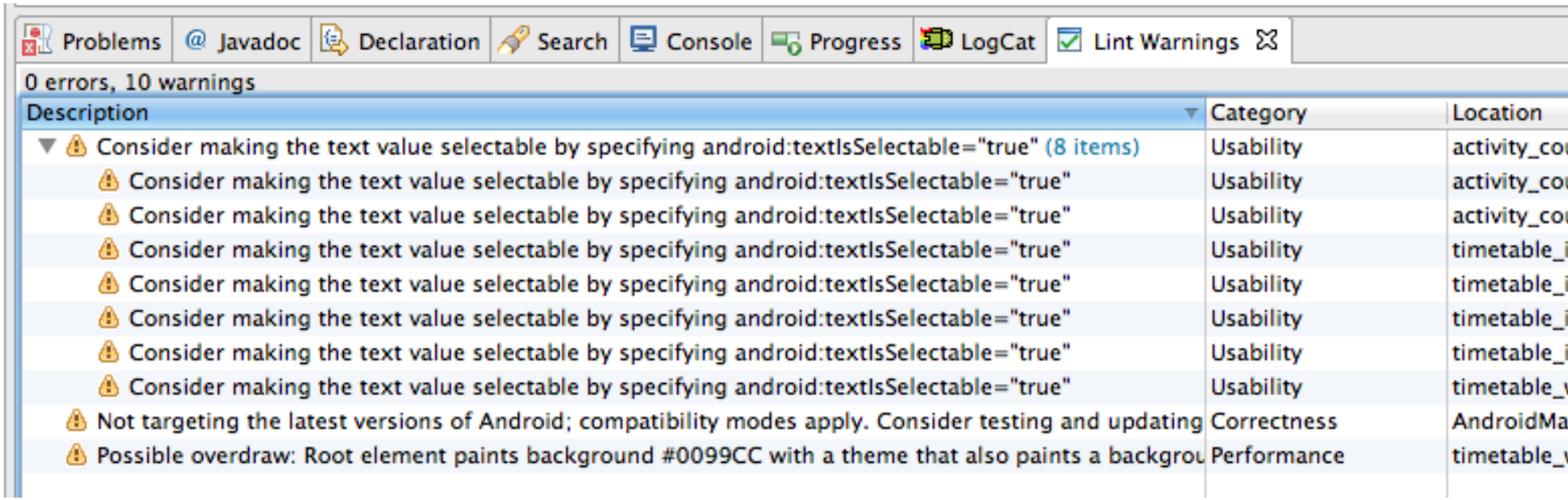
# Lint: "The resource appears to be unused"



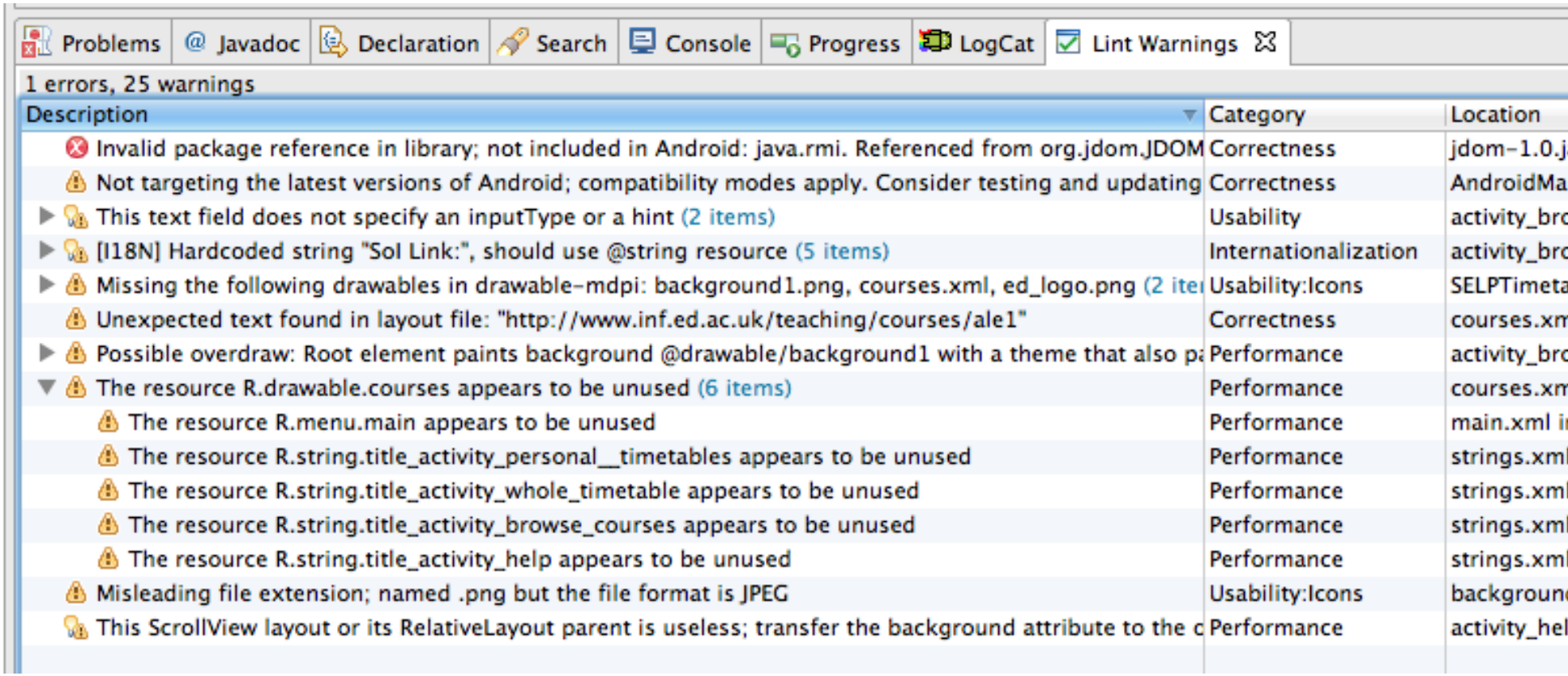
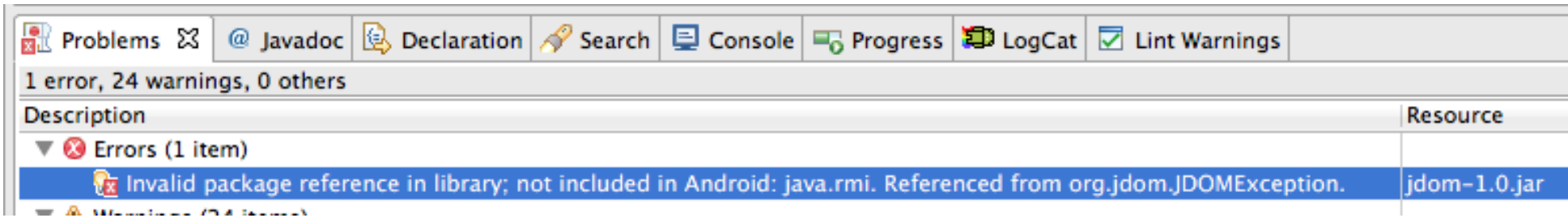
# Lint: Accessibility warnings: missing contentDescription



# Lint: "Consider making the text value selectable"



# Lint: I18N (Internationalisation)—hardcoded strings



# Lint: "Invalid package reference"; some Java packages will not work with Android