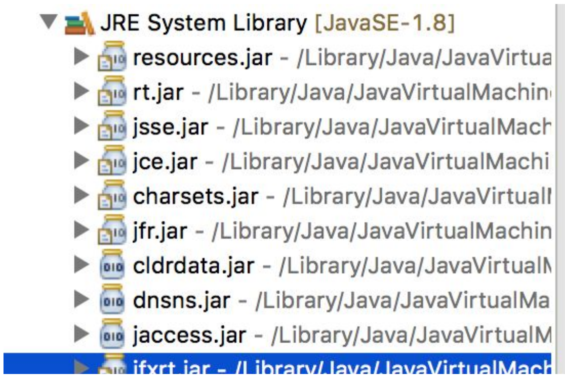
Preparation :

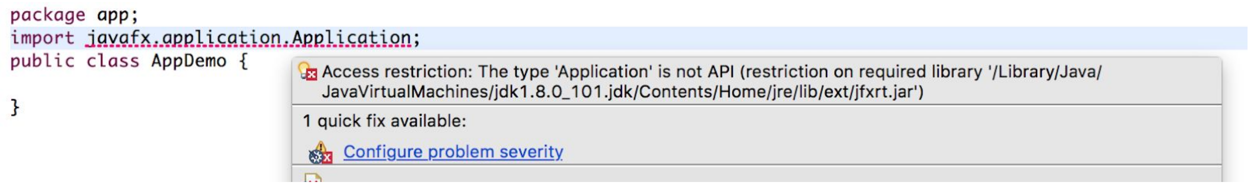
1. javafx environment

We can see that that eclipse already have javafx library



JRE System Library [JavaSE-1.8]
- resources.jar - /Library/Java/JavaVirtua
- rt.jar - /Library/Java/JavaVirtualMachin
- jsse.jar - /Library/Java/JavaVirtualMach
- jce.jar - /Library/Java/JavaVirtualMachi
- charsets.jar - /Library/Java/JavaVirtual
- jfr.jar - /Library/Java/JavaVirtualMachin
- cldrdata.jar - /Library/Java/JavaVirtualN
- dnsns.jar - /Library/Java/JavaVirtualMa
- jaccess.jar - /Library/Java/JavaVirtualM
- jfxrt.jar - /Library/Java/JavaVirtualMach

but there’s still has some restriction to use some function of it:



```
package app;
import javafx.application.Application;
public class AppDemo {

}
```

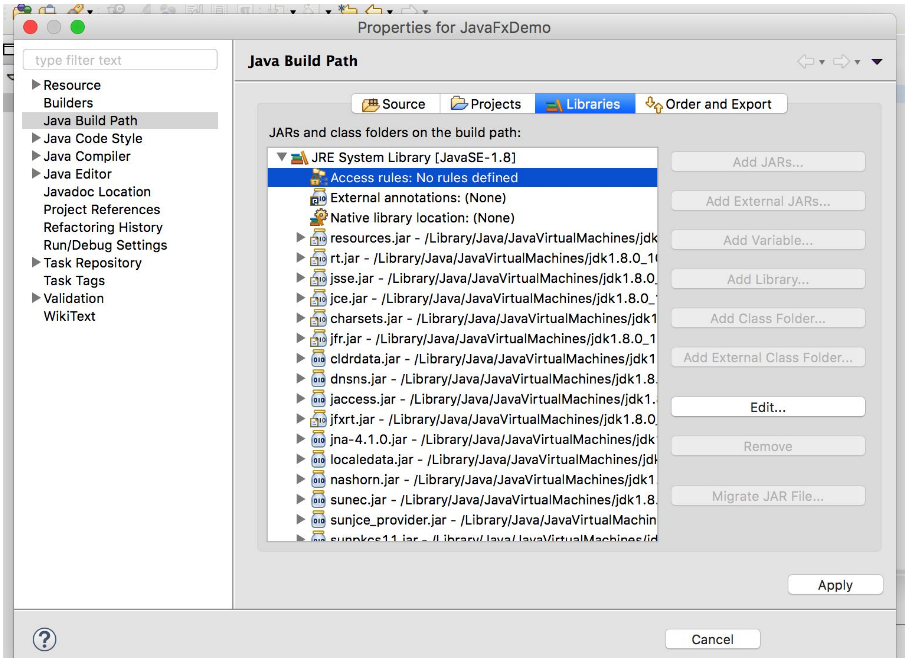
Access restriction: The type 'Application' is not API (restriction on required library '/Library/Java/JavaVirtualMachines/jdk1.8.0_101.jdk/Contents/Home/jre/lib/ext/jfxrt.jar')

1 quick fix available:

Configure problem severity

so we should right click the project, follow step like:

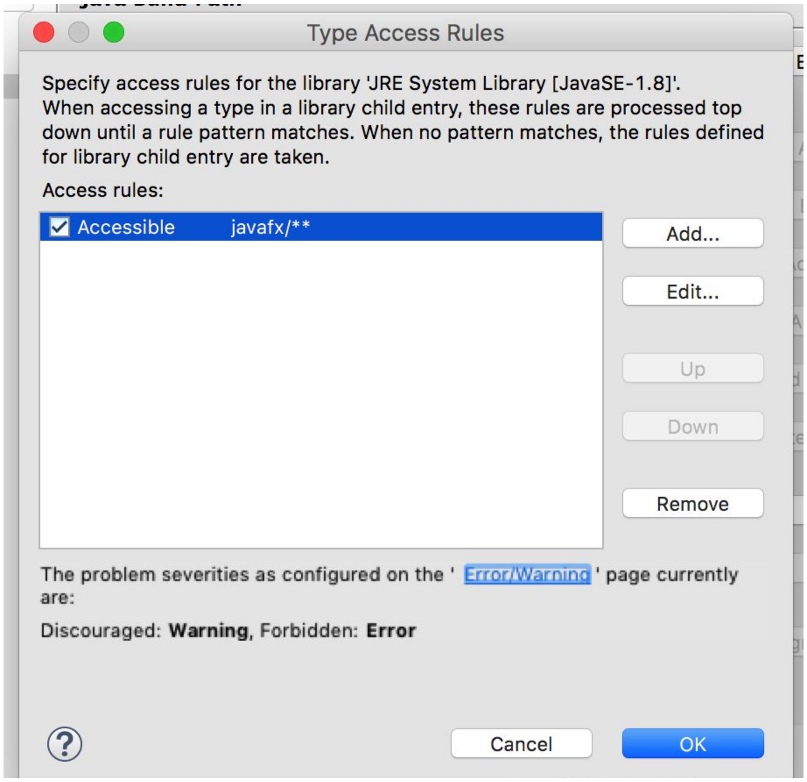
Properties → Java Build Path → Libraries → JRE System Library to edit the access rule:



JARs and class folders on the build path:

JRE System Library [JavaSE-1.8]
- Access rules: No rules defined
- External annotations: (None)
- Native library location: (None)
- resources.jar - /Library/Java/JavaVirtualMachines/jdk
- rt.jar - /Library/Java/JavaVirtualMachines/jdk1.8.0_1(
- jsse.jar - /Library/Java/JavaVirtualMachines/jdk1.8.0_
- jce.jar - /Library/Java/JavaVirtualMachines/jdk1.8.0_
- charsets.jar - /Library/Java/JavaVirtualMachines/jdk1
- jfr.jar - /Library/Java/JavaVirtualMachines/jdk1.8.0_1
- cldrdata.jar - /Library/Java/JavaVirtualMachines/jdk1
- dnsns.jar - /Library/Java/JavaVirtualMachines/jdk1.8.
- jaccess.jar - /Library/Java/JavaVirtualMachines/jdk1.
- jfxrt.jar - /Library/Java/JavaVirtualMachines/jdk1.8.0_
- jna-4.1.0.jar - /Library/Java/JavaVirtualMachines/jdk
- localedata.jar - /Library/Java/JavaVirtualMachines/jdk
- nashorn.jar - /Library/Java/JavaVirtualMachines/jdk1.
- sunec.jar - /Library/Java/JavaVirtualMachines/jdk1.8.
- sunjce_provider.jar - /Library/Java/JavaVirtualMachin
- sunpkcs11.jar - /Library/Java/JavaVirtualMachines/jd

add an access rule: javafx/**



Type Access Rules

Specify access rules for the library 'JRE System Library [JavaSE-1.8]'. When accessing a type in a library child entry, these rules are processed top down until a rule pattern matches. When no pattern matches, the rules defined for library child entry are taken.

Access rules:

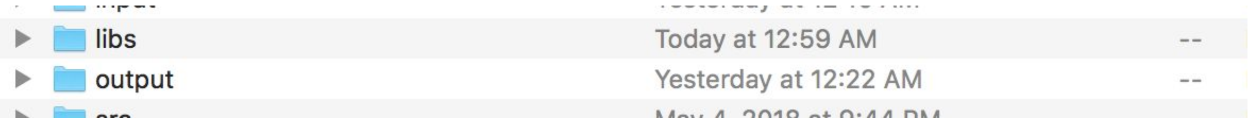
☑ Accessible javafx/**

The problem severities as configured on the ' Error/Warning ' page currently are:

Discouraged: **Warning**, Forbidden: **Error**

2. Libraries

There're some libraries you should add to path before running

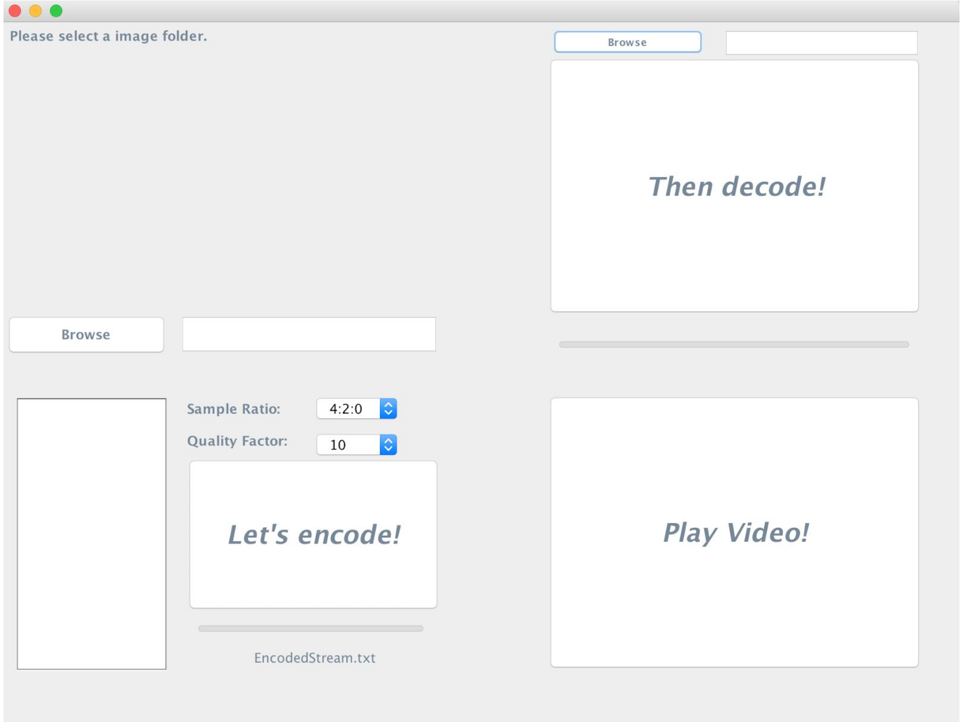


libs	Today at 12:59 AM	--
output	Yesterday at 12:22 AM	--

you can see the libs folder within our project, and in the eclipse or any IDE you are working in, you should right click on this folder and choose add to path, then you'll see those errors are gone.

GUI:

1. First run the class of GUI3, and you can get a frame like this:



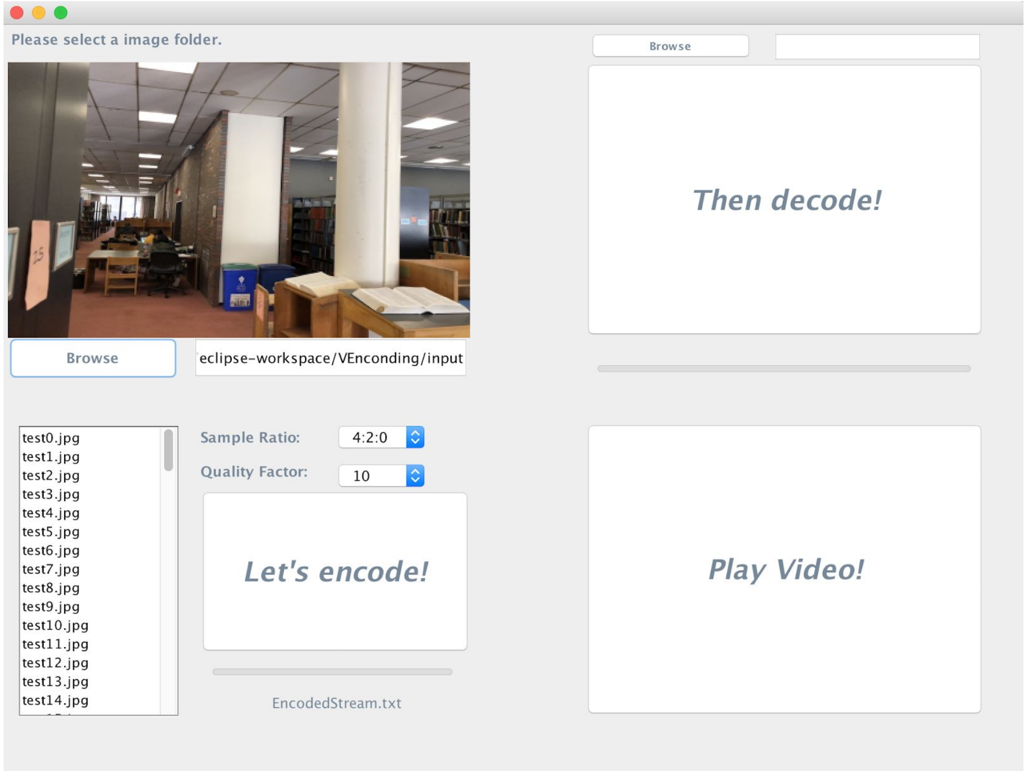
Please select a image folder.

Sample Ratio: 4:2:0

Quality Factor: 10

EncodedStream.txt

2. At the beginning of the day, you have to store the test images into ./input folder, and the name should be like test0,test1,... You can click on the browse to the input *folder* to have a preview of those images.



Please select a image folder.

eclipse-workspace/VEnconding/input

```
test0.jpg
test1.jpg
test2.jpg
test3.jpg
test4.jpg
test5.jpg
test6.jpg
test7.jpg
test8.jpg
test9.jpg
test10.jpg
test11.jpg
test12.jpg
test13.jpg
test14.jpg
```

Sample Ratio: 4:2:0

Quality Factor: 10

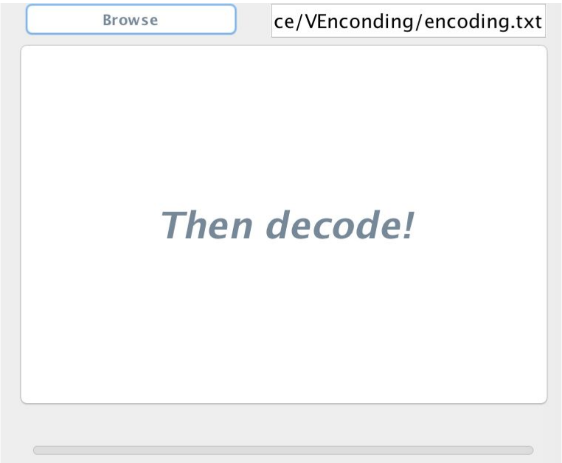
EncodedStream.txt

3. Next let's click "let's encode" to generate byte stream!! You can change sample ratio and quality factor by set the dropdown window, here we just set 4:2:0 and 10 as default. Please wait

patient before there’s a value on the progress bar(we’ll solve this problem in the later version). On our computer(macbook pro 2017 13-inch), the process of running 100 images(single one with 170KB) will cost less than 4 minutes(at this step we highly recommend not to encode more than 50 images, it will cost the decode part slow)

4. Then we can choose the byte stream txt file at the next browser drop down window, the txt file is at the root menu of our

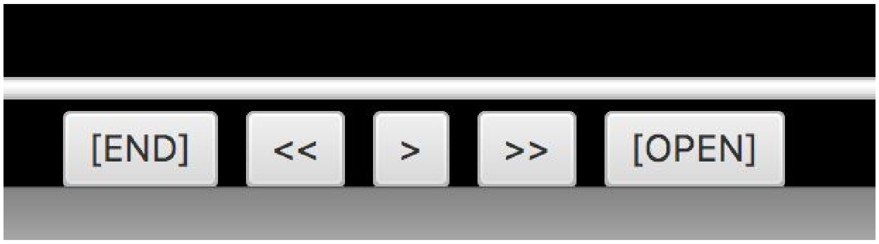
bin	Friday, May 4, 2018 7:04 PM
encoding.txt	Friday, May 4, 2018 9:10 PM
input	Friday, May 4, 2018 7:55 PM
output	Friday, May 4, 2018 8:38 PM



ce/VEnconding/encoding.txt

then we can click the decode button to decode the our byte stream and generate movie.

5. The last exciting thing we need to do is to play the video!



use open button and

Command Line Code:

```
javac -cp ".:/Users/apple/Documents/workspace/ImageCompression/src/jpeg/libs/*" *.java
```

you need to change
“ /Users/apple/Documents/workspace/ImageCompression ”
this part to your absolute directory of src folder

then under back to src folder
run this cml:

```
java -cp ".:/Users/apple/Documents/workspace/ImageCompression/src/jpeg/libs/*"
jpeg/VideoEncoderTest -s 4:2:2 -q 5 -p
/Users/apple/Documents/workspace/ImageCompression/input -n 3
```

[-s 4:4:4 or 4:4:2 or 4:2:0]
[-q from 5 to 100]
[-p change to your computer path of our project/input]
[-n # of images to input max = 100]

you need to change
“ /Users/apple/Documents/workspace/ImageCompression ”
this part to your absolute directory of src folder