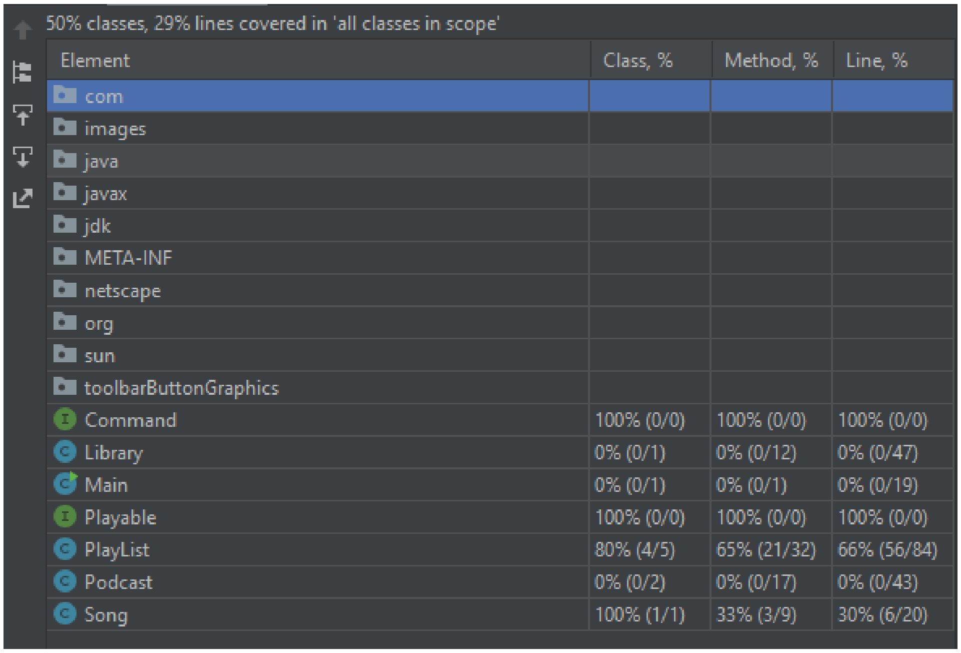
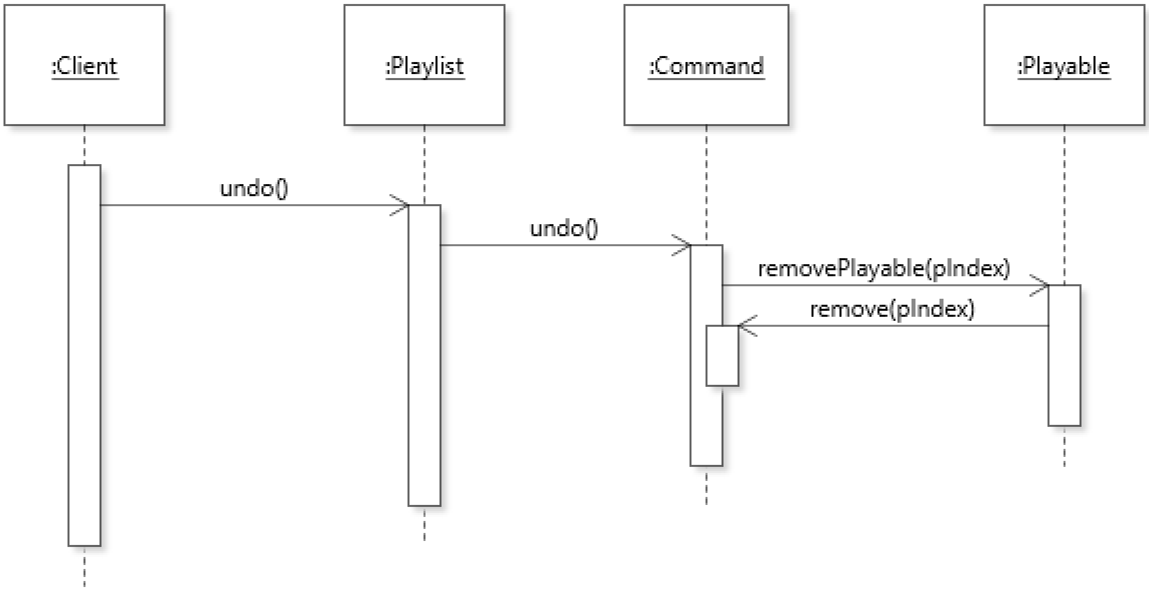
In order to solve the first question, I have used the Prototype design pattern in the Library class in order to implement “default” playables. As for the second question, the Command design pattern was implemented in the Playlist class in order to store the commands needed to implement undo() and redo(). There was no major trade-offs in this assignment. The JUnit test was run and below is a screenshot of the coverages of the test. Additionally, I have also included a sequence diagram for a call on undo() on a state-changing method of Playlist.



50% classes, 29% lines covered in 'all classes in scope'

Element	Class, %	Method, %	Line, %
com			
images			
java			
javax			
jdk			
META-INF			
netscape			
org			
sun			
toolbarButtonGraphics			
Command	100% (0/0)	100% (0/0)	100% (0/0)
Library	0% (0/1)	0% (0/12)	0% (0/47)
Main	0% (0/1)	0% (0/1)	0% (0/19)
Playable	100% (0/0)	100% (0/0)	100% (0/0)
PlayList	80% (4/5)	65% (21/32)	66% (56/84)
Podcast	0% (0/2)	0% (0/17)	0% (0/43)
Song	100% (1/1)	33% (3/9)	30% (6/20)

1. Coverages of the JUnit tests



2. Sequence diagram for a call to undo() for a state-changing Playlist method