1. Why do we use API? We can make it interact with any technology
2. SOA → Service Oriented Architecture
3. XML → open standard language
4. every technology understands XML and every technology speaks XML
5. The tech that contains all the information (google) is called a **Broker**
6. Simple Object access protocol → SOAP
7. REST uses HTTP
8. SOAP uses XML
9. Test suite is a collection of test case
10. Test case is a collection of test step
11. Test Suite → test case → test steps
12. What is API? → it ensures the data communication between the three layers are clear
13. What are the three layers of API:
    1. The interface
    2. The business layer
    3. The database (final layer)
14. What is a highly used tool in the industry: REST
15. Implementing Java in an assured manner is called REST Assured
16. REST assured is back-end java script
17. Soapui can create web services and rest services
18. WSDL file is given by the provider or developer containing a list of services
19. Login → find → purchase → log out for the resource file shared by sunil
20. The WSDL provided was local, it is not provided on any server
21. Since it is on local, will it provide a response: No
22. We have sent a request, but we did not specify where to ask the request since it is local
23. Mocking: It is all about creating your own test environment.
24. It is mirroring the production environment
25. Mocking helps in creating a mock server and getting a response from the local request
26. It’s an open source tool, do not try to minimize the request, so it is not so flexible
27. xpath: //\*:username
28. Fault code is a fault response
29. HTML is for look and feel
30. XML is to store data
31. What does XML trying to store?: it is trying to store the request
32. What is assertion: comparing the actual result with expected result
33. Soapui is a type of automation
34. Groovy script interacts with the server
35. We can use any wsdl file but it is not necessary to get the response
36. The public server does not need a mock service because it is set up
37. What are mock service for? We are creating the environment for the responses to give the request
38. Using a public server is not secured
39. It is because it is not confidential and anyone can access the file
40. Public servers response cannot alter the response
41. Expected results and actual results are important for SOAPUI and is called assertion
42. Add assertion: Compliance, Status and Standard →Not SOAP Fault
43. What is “Not SOAP fault”? → This passes when the test contains valid details
44. What is “Soap Fault”? → This passes when the test contains invalid details
45. Sequence response: it is when the responses (Successful and Faulted) will comes in alternative manner
46. Xpath is a type of script so all the script used from cheatcode, we will have to write it in xpath
47. We used script instead of query\_match because in query\_match we can only use one clause
48. While script can use multiple conditions
49. Xml is case sensitive
50. namespace comes only for service
51. When you have already logged in, we will have to restart the server
52. Property transfer syntax  
    1. <xtml ns>${property file name #propertyname} <xtml ns>
53. go to the testCaseLog to see what username and password is pulled from the property
54. To get a property transferred from a response to a request, we use the below syntax  
    1. ${Login#Response#//\*:sessionId}
    2. ${SOAP Page #Response#//\*:itemToBeTransfered}
55. What is the difference between property expansion and traditional?: Traditional has a transfer for each functionality and from PE has only one property list and using syntax, we can transfer the property in to the SOAP service
56. Schema compaliance checks if the wsdl file is valid
57. Script assertion is used to store the responses from the SOAP Services after the assertions are done.
58. Context means test case
59. To get the property value we use getPropertyValue
60. To store a property value we use setPropertyValue
61. To store a value in a Test Case Level  
    1. context.testCase.setPropertyValue(”ideal name”, temp name)
62. To store a value in a Test Suite Level  
    1. context.testCase.testSuite.setPropertyValue(”ideal name”, temp name)
63. To Store a value in a Project Level  
    1. context.testCase.testSuite.project.setPropertyValue(”ideal name”, temp name)
64. Library for the groovy script to use script asserstion:  
    1. import com.eviware.soapui.support.XmlHolder
65. holder.getNodeValue will remove the tags from the response and get the value
66. **Code to do the whole script asserstion for the session id to store** // creating the variable 'response' // the variable will fetch the response and store it in the variable  
      
     //Importing the library to talk to the xml file import com.eviware.soapui.support.XmlHolder  
      
     //create a variable to fetch the response of login def response = context.testCase.getTestStepByName("Login").getPropertyValue("Response")  
      
     //creating a holder to remove the tags and to fetch the data from the response def holder = new XmlHolder(response)  
      
     def session = holder.getNodeValue("//\*:sessionId")  
      
     // Storing the value into the test suite context.testCase.testSuite.setPropertyValue("sessionId", session)  
      
     // Storing the value into the test case context.testCase.setPropertyValue("sessionId", session)  
      
     // Storing the value into the project context.testCase.testSuite.project.setPropertyValue("sessionId", session)
67. To input the values stored in  
    1. Test Case = ${#TestCase #sessionId}
    2. Test Suite = ${#TestSuite #sessionId}
    3. Project = ${#Project #sessionId}
68. To see what value is stored in the script assertion: log.info variableNameWithoutTag
69. The script asserstion is a part of integration testing
70. If I want to run script based execution from Groovy Script execution I will use **TestRunner**
71. **testRunner** establishes a communication channel between two test suites or two test case
72. If you are running more than one step in a test case, you have to write tc.run(null, false) where null means there is no context, false is because i am not checking conditions
73. To see what steps have been executed tc.getTestStepList()
74. If you want to go to a particular test suite or test case, to identify the particular value from them, we use square brackets
75. When using a method, we use simple brackets
76. The code to access files in groovy script and get the list of test steps performed are  
      
     import com.eviware.soapui.model.testsuite.TestRunner.Status // import the library  
      
     // we want a test runner to interact with another test case in another test suite with our current test case  
      
     // to access the test case “Script Based Execution” in the Test Suite “Shopping Service Functional” def tc = testRunner.testCase.testSuite.project.testSuites['Shopping Service Functional'].testCases['Script Based Execution']  
      
     // to set the username and password in the test step to new values  
      
     tc.testSteps["username password findString purchaseItem Properties"].setPropertyValue("username","Lally")  
      
     tc.testSteps["username password findString purchaseItem Properties"].setPropertyValue("password","Liesl")  
      
     //to run the test case with no context and no conditions  
      
     tc.run(null, false)  
      
     //to check the list of the test steps performed  
      
     def teststeplist = tc.getTestStepList() //checking on the output by using [log.info](http://log.info) //log.info teststeplist  
      
     //looping through each of the items in the teststeplist and writing the name  
      
     teststeplist.each {  
      
     def tsname = it.name  
      
     log.info tsname }
77. Web services testing is done with soapui and this checks the flow of the services.
78. We can automate the flow of the services
79. What is namespace [ns]: it can hold  
    1. URL
    2. data which holds our package
    3. star (\*) is an irregular expression which is used in all terminology to get a dynamic data
    4. dynamic data is when there are many similar data and the star (\*) will bring the best data among the dynamic data
80. DTD: Document type declaration  
    1. <info>
    2. <to>
    3. <message>
    4. </message>
    5. </to>
    6. </info>
81. There are two types of data type  
    1. CData: Character Data (When I use this data, it will not compile)
    2. PCData: Parser character data (the tool will compile when we use PCData)
82. XSD: XML Schema Definition
83. XSD uses datatypes:  
    1. int
    2. string
    3. boolean
84. WSLD was built using XSD
85. if you want to fetch a detail def username = context.expand(’${Login#Request#//\*: username}’)
86. Double quotes is when we want to print the statement
87. The comments in the script is kept in the script log
88. To place an if loop to compare username and value  
      
     // placing a if statement to check if the purchase can be done if(username != "Liesl"){ log.info "Not a Privilaged Customer - NO PURCHASE" //To jump to the assigned services testRunner.gotoStepByName("Logout") } else{ log.info "Privilaged Customer - Processed to Purchase" }
89. The full groovy script to jump a services based on the value stored in the username  
      
     // this is used to check if to see if Liesl can purchase or not  
      
     //We need to fetch the username first  
      
     def username = context.expand('${Login#Request#//\*:username}') // display what username has been retrieve //log.info username  
      
     // placing a if statement to check if the purchase can be done if(username != "Liesl"){ log.info "Not a Privileged Customer - NO PURCHASE" //TO jump to the assigned services testRunner.gotoStepByName("Logout") } else{ log.info "Privileged Customer - Processed to Purchase" }
90. Size in Java means the number of steps carried out runner.results.size()
91. Time take for the steps to execute runner.timeTaken
92. To check the status of the test runner.status
93. The whole script to check from a different test case  
      
     import com.eviware.soapui.model.testsuite.TestRunner.Status // import the library  
      
     // we want a test runner to interacte with another test case in another test suite with our current test case def tc = testRunner.testCase.testSuite.project.testSuites['Shopping Service Functional'].testCases['Property Expansion Login and Logout']  
      
     tc.testSteps["username password findString purchaseItem Properties"].setPropertyValue("username","Tessa") tc.testSteps["username password findString purchaseItem Properties"].setPropertyValue("password","Teeena")  
      
     def runner = tc.run(null, false)  
      
     //status of the execution log.info "Status: $runner.status"  
      
     //This will provide me the time taken for execution log.info "Time Take: $runner.timeTaken"  
      
     //I want to know the number of steps executed is log.info "The number of steps executed: " +runner.results.size()  
      
     //Assert assert runner.results.size()==5
94. We can place the if condition only in the step and not in the system testing
95. Manual tester will only pass the groovy script but will not write the script.