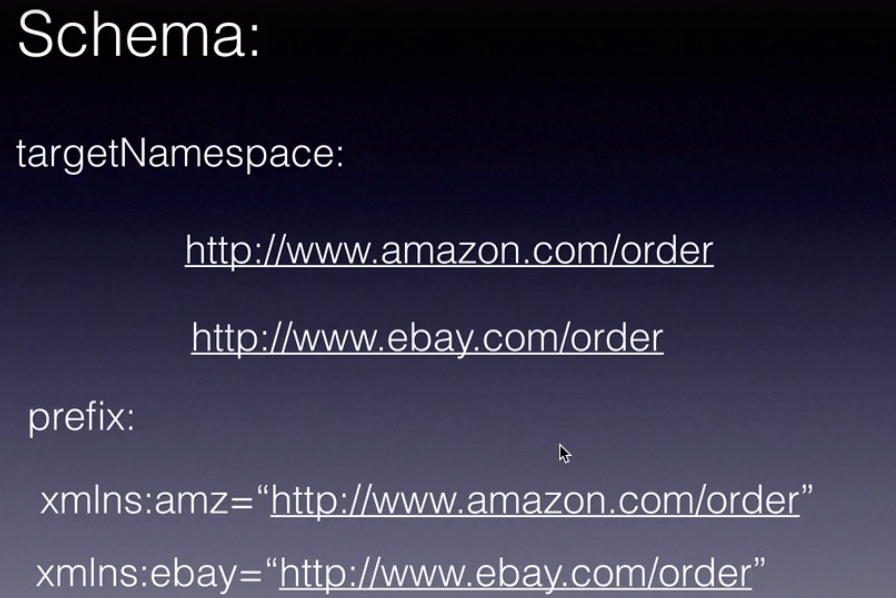
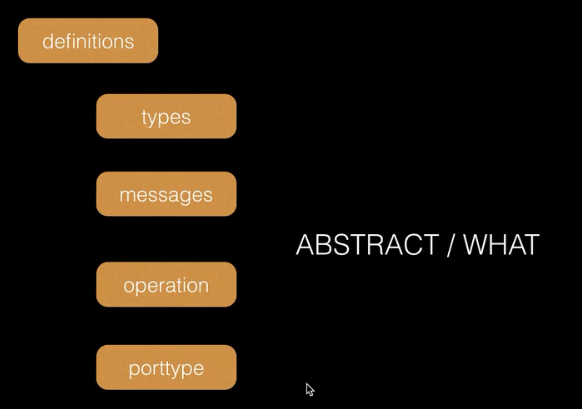
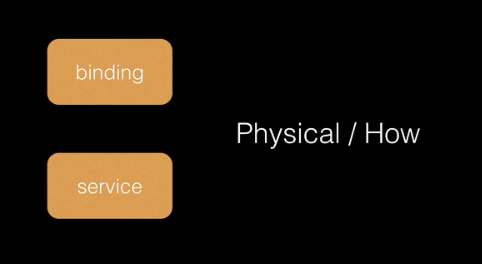
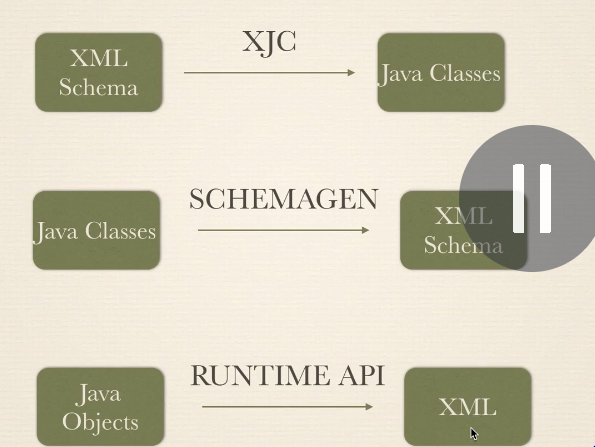
Webservices:

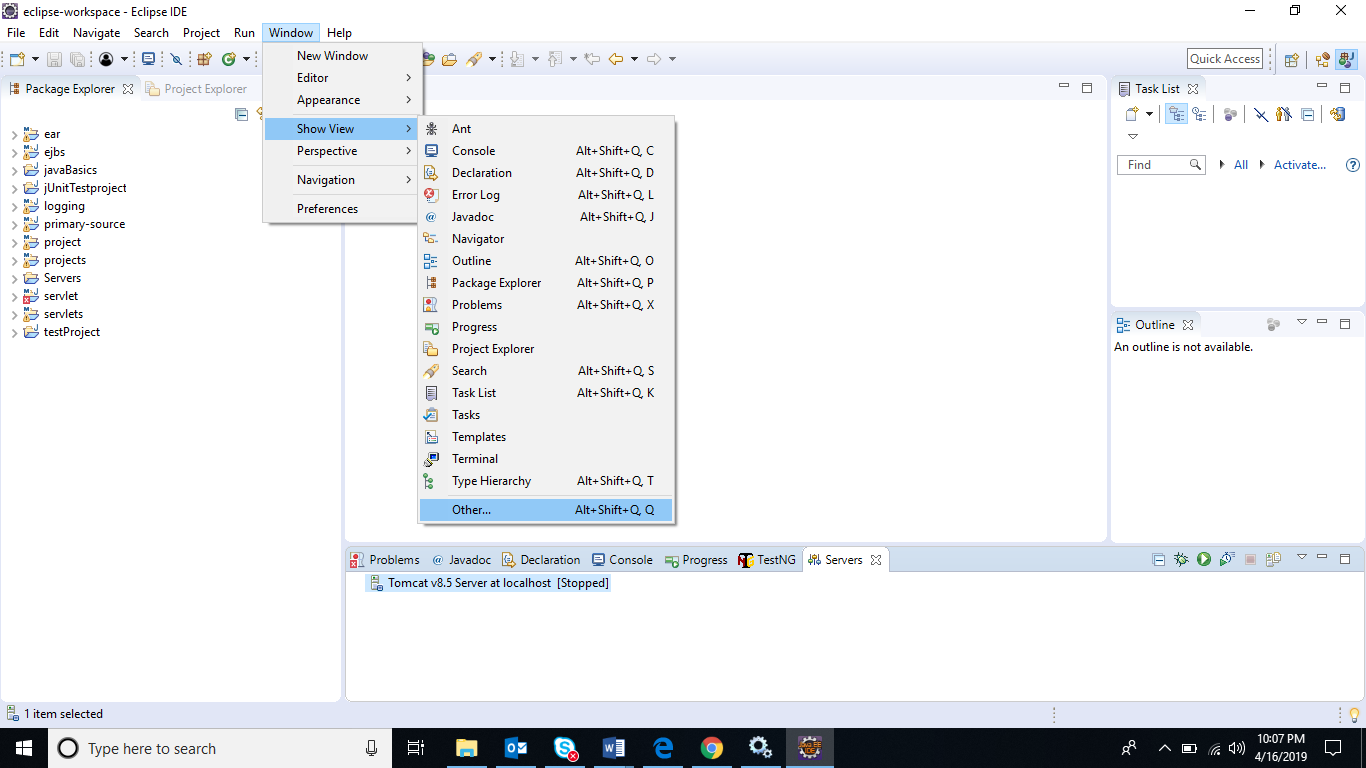
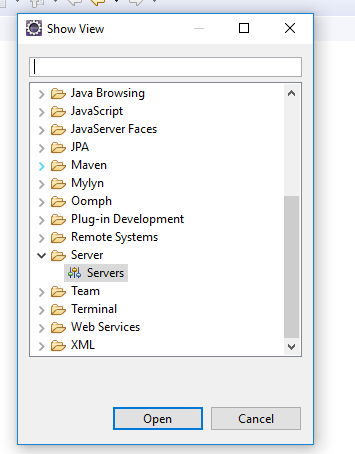
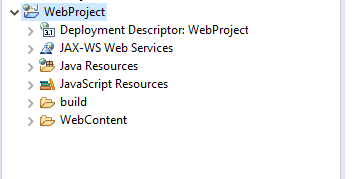
1. Webservices are client server
2. Types: SOAP -> XML or HTTP POST  
   REST ->
3. JAX-WS is the standard for implementing SOAP
4. JAX -RS is the standard for developing RESTFul webservices
5. SOA -> collection of architectural principles
6. XML-> Extensible markup language  
   XML contains both data and meta data
7. XML are mostly used in config files, data exchange, and to save and manipulate data
8. XML Schema Definition  
   XML is valid if it follows the XSD, XSD is also an xml file, which mentions which attributes/ elements to be present in an XML file
9. XSD is a contract defined between two applications, so xml need to follow the xsd
10. Namespaces:  
    
11. <name> -> metadata

Subash -> data  
</name>

1. Webservices might impact performance due to serialization and deserialization
2. SOAP -> Simple Object Access protocol  
   <soap:Envelope>  
    <soap:Header/>  
    <soap: Body>  
    <creditCard>  
    …..  
    </creditCard>  
    <soap:Fault>  
    <Soap: ReasonCode>  
    <Soap: Text>  
    </soap:Body>
3. Current version of SOAP -> 1.2
4. WSDL -> Webservices Description Language  
   It tells what this webservice provide and how it provides  
     
   
5. Binding types:  
   RPC/ Encoded  
   RPC/Literal  
   Document/Encoded -> not WS- I compliant  
   Document/Literal
6. SOAP WS Design approaches  
   Contract first design/ wsdl first/Top down:  
   a) Create WSDL file  
   b) generate Java stubs using tools like wsdl2java  
   c) implement webservices endpoint
7. JAX –WS -> java API for XMl based   
   Annotations:  
   @javax.ws.WebService  
   @javax.ws.WebMethod  
   @javax.xmls.ws.WebFault -> custom exceptions  
   @javax.jws.soap.SOAPBinding  
    @SOAPBinding(style = Style.RPC, use = Use.LITERAL)
8. JAX –B  
   maps Java classes and xml  
     
   marshalling -> Convert Java to XMl

Unmarshalling -> XMl to java

REST Webservices

1. REST -> Representational State Transfer
2. CRUD Operations  
    Create \_> POST  
    READ -> GET  
    Update -> POST  
    Delete -> DELETE
3. REST Supports JSON and XML format
4. REST is Stateless that means State is stored at client end
5. Rest must be used when  
    a) there is a defined contract between applications
6. JAX- RS
7. JAX-RS annotations  
   @Path(“/users/{username}”)  
   HTTP methods  
    @GET  
    @POST  
    @PUT  
    @DELETE
8. Data Format  
   @Consumes(“text/plain”)  
   @Produces(“application/json”  
    “application/xml”)
9. Parameter Values  
   @PathParam  
   @QueryParam  
   @FormParam
10. Exception mappers  
    @Providers
11. Check server tab  
      
    
12. Create a dynamic web project  
    
13. Built IN Server Objects:  
    request  
    response  
    out  
    session  
    application