

Practical Exercise: Introduction to WSO2 EI Tooling

Training Objective

Understand the WSO2 EI Tooling functionality.

High Level Steps

- Install WSO2 EI 6.1.1
- Overview of EI Tooling

Detailed Instructions

Install WSO2 EI Tooling

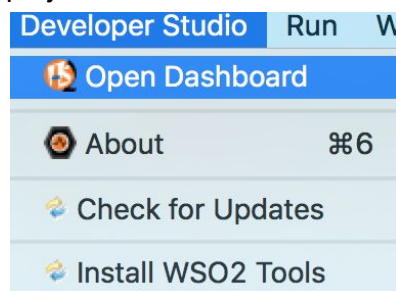
WSO2 EI Tooling will be used to create the configuration artifacts for WSO2 EI. Refer the documentation [1] for detailed instructions on installing EI Tooling.

[1] <https://docs.wso2.com/display/EI611/Installing+Enterprise+Integrator+Tooling>




Overview of WSO2 EI Tooling

WSO2 EI Tooling is an IDE solution that uses WSO2 Developer Studio, where developers can define a project representing a complete Composite Application (C-App) spanning multiple WSO2 products and features.






In WSO2 EI Tooling, the Developer Studio dashboard allows the development of an ESB project. This dashboard can be accessed through the following menu option.










Through the use of this dashboard, you can create an ESB project. The ESB project you select will depend on that artifact type that you will be developing.

An ESB Solution Project that creates all the required project files (i.e., ESB config project, registry resource project, connector exporter project and composite application project).	 ESB Solution Project
An ESB Project that holds ESB resources and can be deployed on a WSO2 EI installation.	 ESB Config Project
A Java project with a mediator class that implements the org.apache.synapse.mediators.AbstractMediator class.	 Mediator Project

Once created, an ESB Project can hold the following ESB resources:

Definitions of external service endpoints and any attributes or semantics that should be followed when communicating with them.	 Endpoint
Entry point for the mediation layer.	 Inbound Endpoint
Jobs (for periodic execution) in the runtime of the ESB profile of WSO2 EI.	 Scheduled Task
Units of execution that have the ability to connect to a message store and perform message mediation or the required data manipulations.	 Message Processor
Virtual services that appear as regular, fledged web services to the external clients. There are four types of proxy services: <ul style="list-style-type: none"> • Pass Through Proxy: A simple proxy service on a specified endpoint that does not perform any processing on the messages. • Secure Proxy: A proxy that will process WS-Security on incoming requests and forwards them to an unsecured backend service. • WSDL Based Proxy: A proxy service out of a WSDL of an existing Web service. • Logging Proxy: A proxy service that logs all the incoming requests and forwards them to a given endpoint. • Transformer Proxy: A proxy service that transforms all the incoming requests using XSLT and then forwards them to a given endpoint.. • Custom Proxy: Launches the proxy service creation wizard. 	 Proxy Service

Local configuration registry/repository for resources such as WSDLs, schemas, scripts, etc.	 Local Entry
An array of mediators assembled as a message flow unit.	 Sequence
Virtual web application that provides a convenient approach for filtering and processing HTTP traffic through the service bus.	 REST API
Prototypes of Endpoints or Sequences that can be used as a base for new objects.	 Template
Units of storage (queues) for messages/data exchanged during WSO2 EI runtime. Queues can refer to a local in-memory store implementation or to an externally connected MQ product, referenced by a JMS datasource.	 Message Store
If connectors were used in an ESB project, the Connector Exporter Project should be used for adding connectors into the CAR file of the ESB project.	 Connector Exporter Project
Creates a new project using the synapse configuration file of a project that is already built. The various elements included in the configuration (APIs, proxy services, endpoints etc.) will be separated in the new project.	 Synapse Configuration

Best Practices

- When using Maven to build/deploy artifacts, select **Maven multi module** project as the parent project for your configuration artifacts. All other projects should be defined as sub projects under the parent project.
- Create a common project to have the main logic and have all environment dependent variables in a separate project.