Mule ESB – A look to greatness

2/20/15 17:44

Chapter 1: Getting started with Mule EB

- Mule ESB is a lightweight Java application.
- Attempts for an easy integration cross-technologies.

ESB:

- Enterprise Service Bus is an application that gives access to other applications and services.
- It's main task is to become the messaging and integration backbone.
- Middleware system.
- **Main features ('o Mule ESB):**
  - Polling JMS.
  - Message transformation and routing services.
  - Tomcat hot deployment. (**!!!**)
  - Web service security.
- **VETRO (a. k. a. MuleESB functionality):**
  - **V**alidate the schema validation.
  - **E**nrich.
  - **T**ransform.
  - **R**oute (either **itinerary [list of services to execute]** or content based).
  - **O**perate (perform operations; they run at the backend).
- **Main needs supplied by an ESB:**
  - To integrate two or three services and/or applications.
  - To integrate more applications, services or technologies in the future.
  - To use different communication protocols.
  - To publish services for composition and consumption.
  - For message transformation and routing.

MuleESB:

- Two editions: Community and Enterprise (clustering, performance tuning, DataMapper and SAP connector capabilities).
- **Main capabilities:**
  - Service creation and hosting.
  - Service mediation.

    - Message routing.
    - Data transformation.
- **Architectural layers:**
    - Application layer (customer data component).
    - Integration layer (object to xml transformer).
    - Transport layer (message transport).
- **Configuration:**
    - **Service component development:** Develop or re-use POJOs, Cloud connectors, or Spring Beans. These components will contain business logic and will consume, process, or enrich messages.
    - **Service orchestration:** Configure message processors, routers, transformers, and filters (Mule flow).
    - **Integration:** Comprises all the connectors so the transport of messages can be done (custom made connectors may be included as well).
    - **Spring beans.**
    - **Agents:** Services created in Mule Studio (started with the server, destroyed likewise).
    - **Connectors:** Software component.
    - **Global configuration:** With global properties.
    - **Global endpoints:** Referenced at the global elements tab.
    - **Global message processor.**
    - **Transformers:** Converts data from format-to-format and can be used in multiple flows.
    - **Filters:** Decides which messages should be processed.
    - **Models:** Logical grouping of services.
    - **Services:** Wraps custom components (business logic) and configure routers, endpoints, transformers, and filters. Each service is connected through endpoints.
    - **Flow:** Used in message processors to define source and target.

[1]

Setting up the Mule IDE (a. k. a. Anypoint Studio)

- Anypoint capabilities:

- Integrate **SaaS (Software As A Service)** and **on-premises (license required software)** applications.
- Modernize legacy services.
- Orchestrate business processes and services.
- Design and publish APIs for end customers, B2B, or mobile.
- Create API proxies to separate implementation from orchestration.
- Engage consumers of your API.
- Govern APIs with runtime policies.

a. Download, install and configure *JDK 1.6* or higher (ensure environment variable JAVA_HOME and PATH are configured for JAVA).
b. Download Anypoint Studio from: http://www.mulesoft.org/download-mule-esb-community-edition [don´t forget to fill in the form]

Anypoint platform for APIs

- Entire API developer lifecycle, from design to management to publishing.
- RESTFul APIs and developers uses API notebook to explore it, test it, and improve it.
- Once developed, an APIkit may be built using Anypoint.
- Deplyed to API gateway.

Anypoint platform for SaaS integration

- Help to connect applications, orchestrate business processes, and publish new services and APIs.
- Integrations designed with Anypoint Studio.
- Integration flow using dran-n-drop or through XML.
- Anypoint Connectors supports up to 120 SaaS and on-premises applications, or standard protocols such as web services, HTTP, or FTP.
- Integrations can run via CloudHub or MuleESB.
- If MuleESB is used, it can be managed through the Mule Management Console.

Anypoint platform for SOA

- Uses MuleESB to its full extent.
- Comprises all we are interested about an ESB.
- Uses Anypoint Connectors.
- Has pre-packaged building blocks to be used right away (plug-in) using the MuleESB.

[2]

The basics (begin with)

- **Flow and messages**
  - Passed through message connectors in a flow.
  - Message processors may transform, filter, enrich, and otherwise process the message.
  - A single flow may be an entire application or you can connect flows between them.
  - Mule may use *global elements* that are invoked by multiple elements in the flow.
  - A message has a **header (**metadata in form of properties and variables**)** and a **payload (**message data**)**.

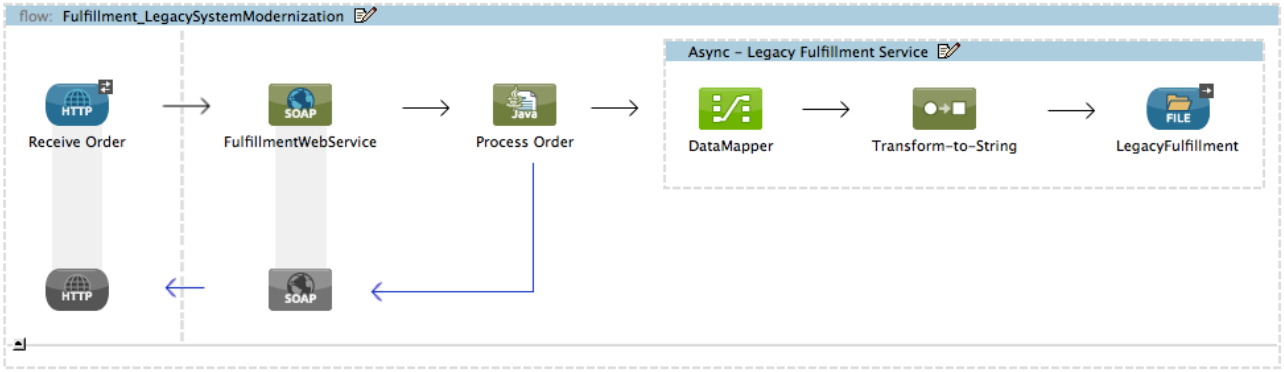

Figure: ESB flow example (http://www.mulesoft.org/documentation/download/attachments/122751586/flow.png?version=2&modificationDate=1417619253110)

- **Message processors**
  - Categorized as follows:
    - **Connectors:** Send and receive data.
    - **Filters:** Limit processing of messages based on set criteria.
    - **Components:** Execute specific logic upon a message (custom code from Java, Javascript, Groovy, Python or Ruby).
    - **Routers:** Direct messages through different pathways in an application depending upon content or other set criteria.
    - **Scopes:** Wrap snippets of code to define fine-grained behavior within a flow.

        - **Transformers:** Convert data types and formats so as to "translate" messages between applications or systems.
        - **Exception strategies:** Handle any errors that occur during message processing.
        - **Business events:** Capture data related to key performance indicators.
- **Batch processing**
    - Anypoint supports batch jobs as an alternative to standard flows.
    - Blocks of code that splits messages into individual records.
    - Useful in "near real-time" data integration between SaaS applications.
- **Mule expression language**
    - **MEL** is the primary language used for formulating expressions in Mule.
    - Allows access and manipulation of messages and their environment.
    - What can be done with it:
        - Extract information.
        - Set or manipulate values in header or payload.
        - Perform an operation on information.

[3]

Is based on Eclipse IDE and allows to build applications through visual editor or XML editor.
Any change in one editor, it reflects to the other one.

The visual editor

1. Package explorer.
2. Canvas.
3. Palette.
4. Connection explorer.
5. Console.

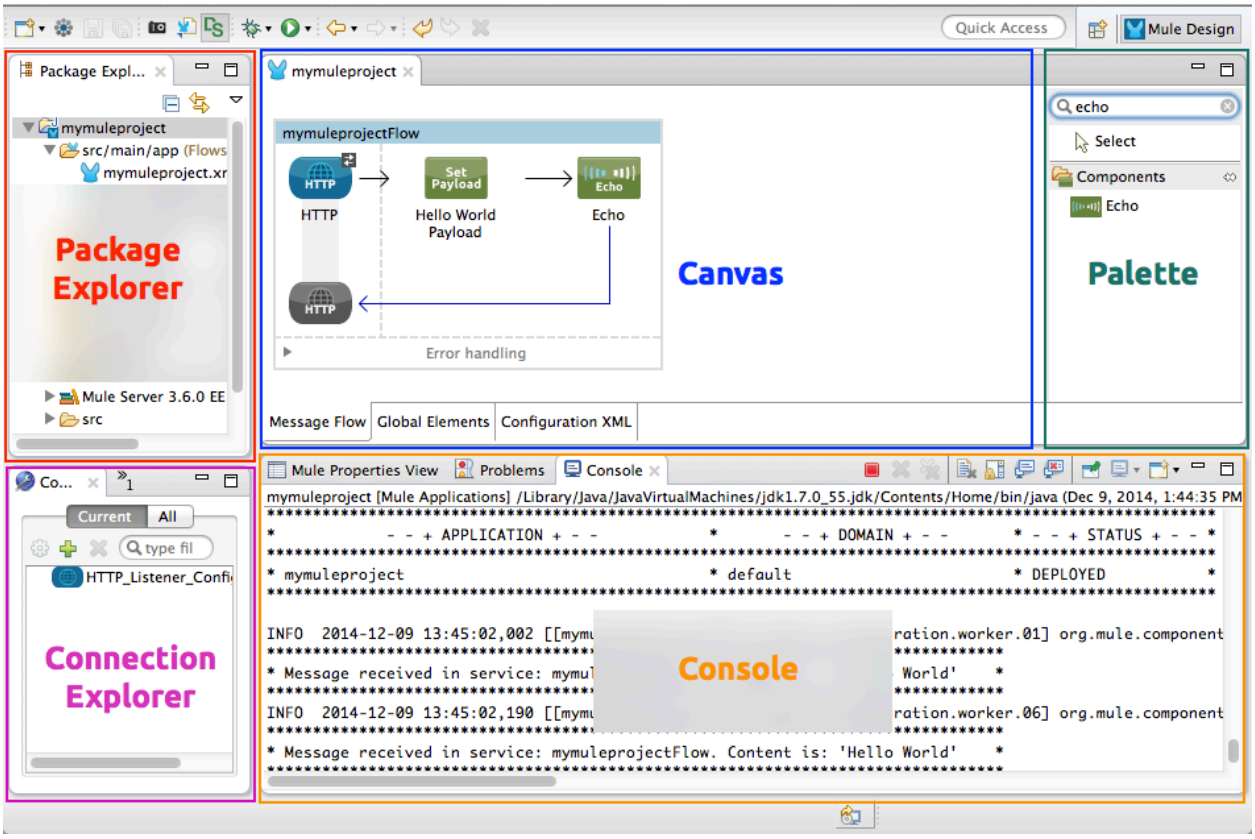

Figure: Anypoint studio distribution.

(http://www.mulesoft.org/documentation/download/attachments/122751585/36Studio.png?version=1&modificationDate=1418170619513)

The elements in the **Palette** can be drag-n-dropped to the canvas (they're the pre-packaged building blocks).

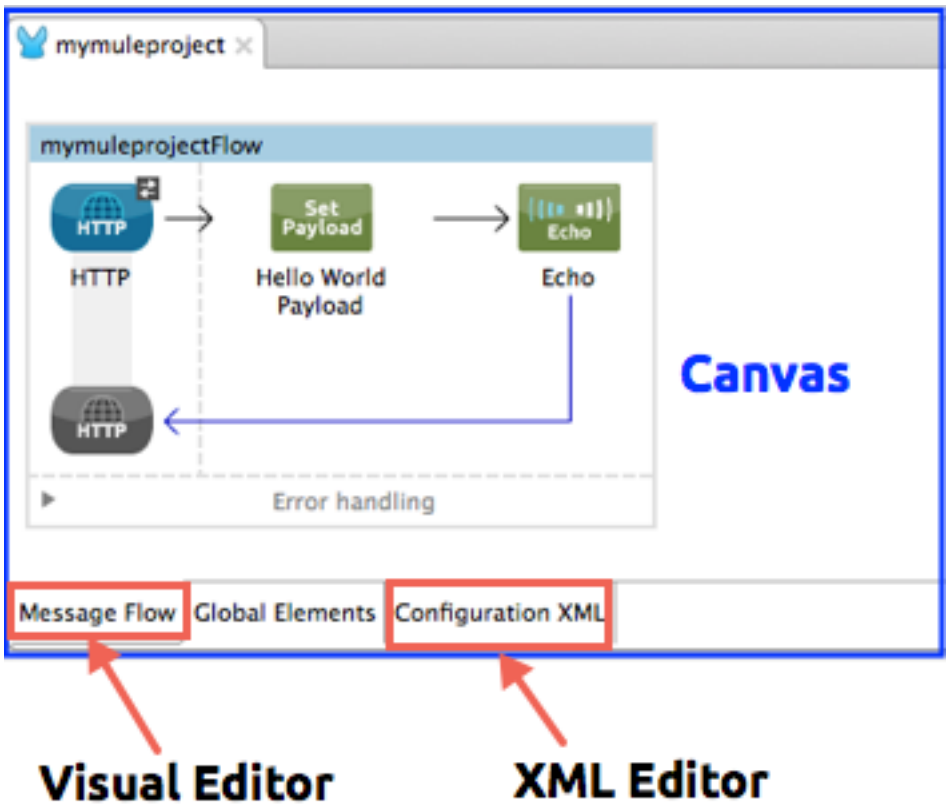

Figure: Visual vs. XML editor (http://www.mulesoft.org/documentation/download/thumbnails/122751585/36VizXmlTabs.png?version=1&modificationDate=1418172038625)

[4]

1. File > New > Mule Project.
2. Enter a name for the project.

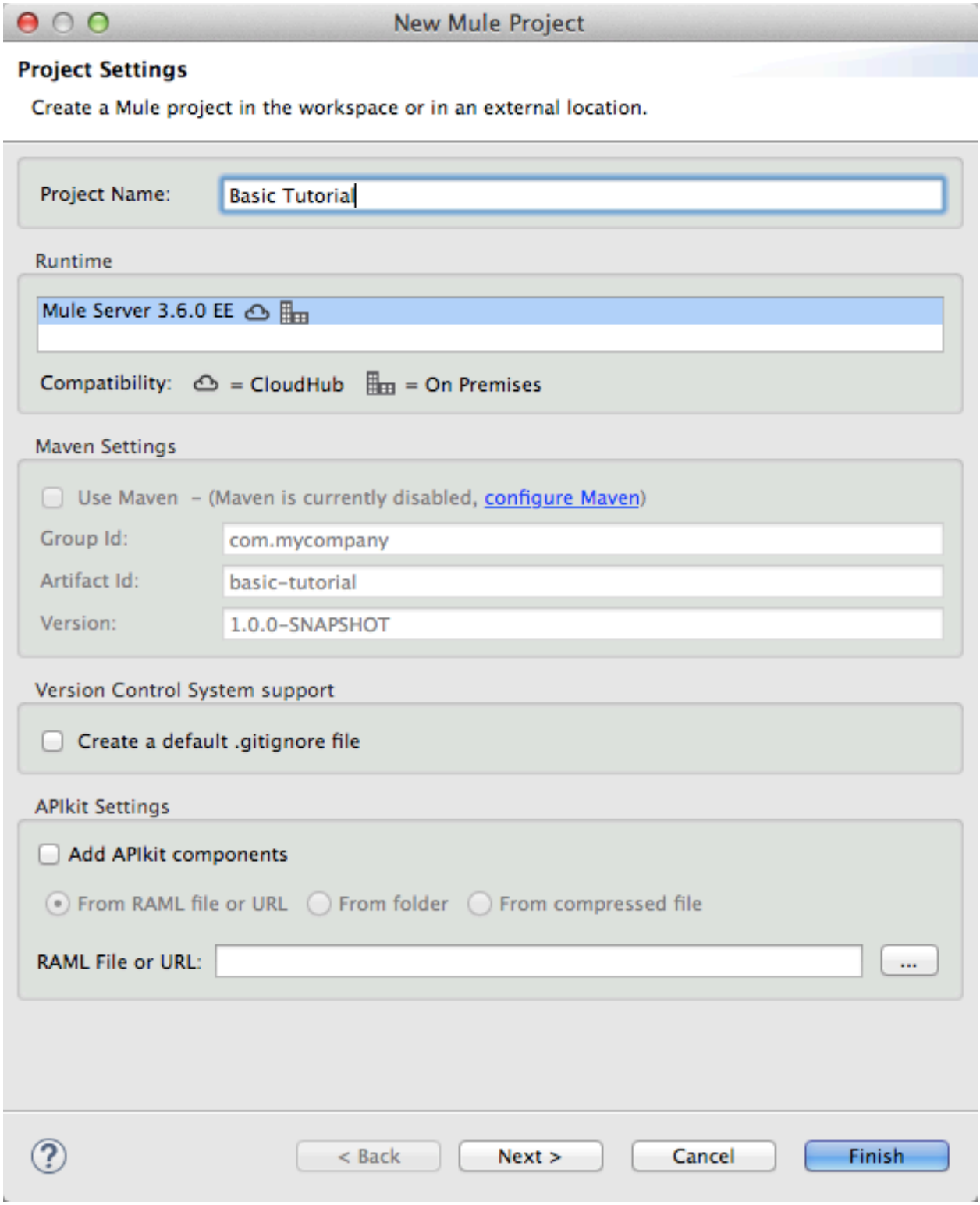

Figure: New mule project (http://www.mulesoft.org/documentation/download/attachments/122751588/new.png?version=2&modificationDate=1420210568256).

3. The project starts a new perspective using the default UI distribution.

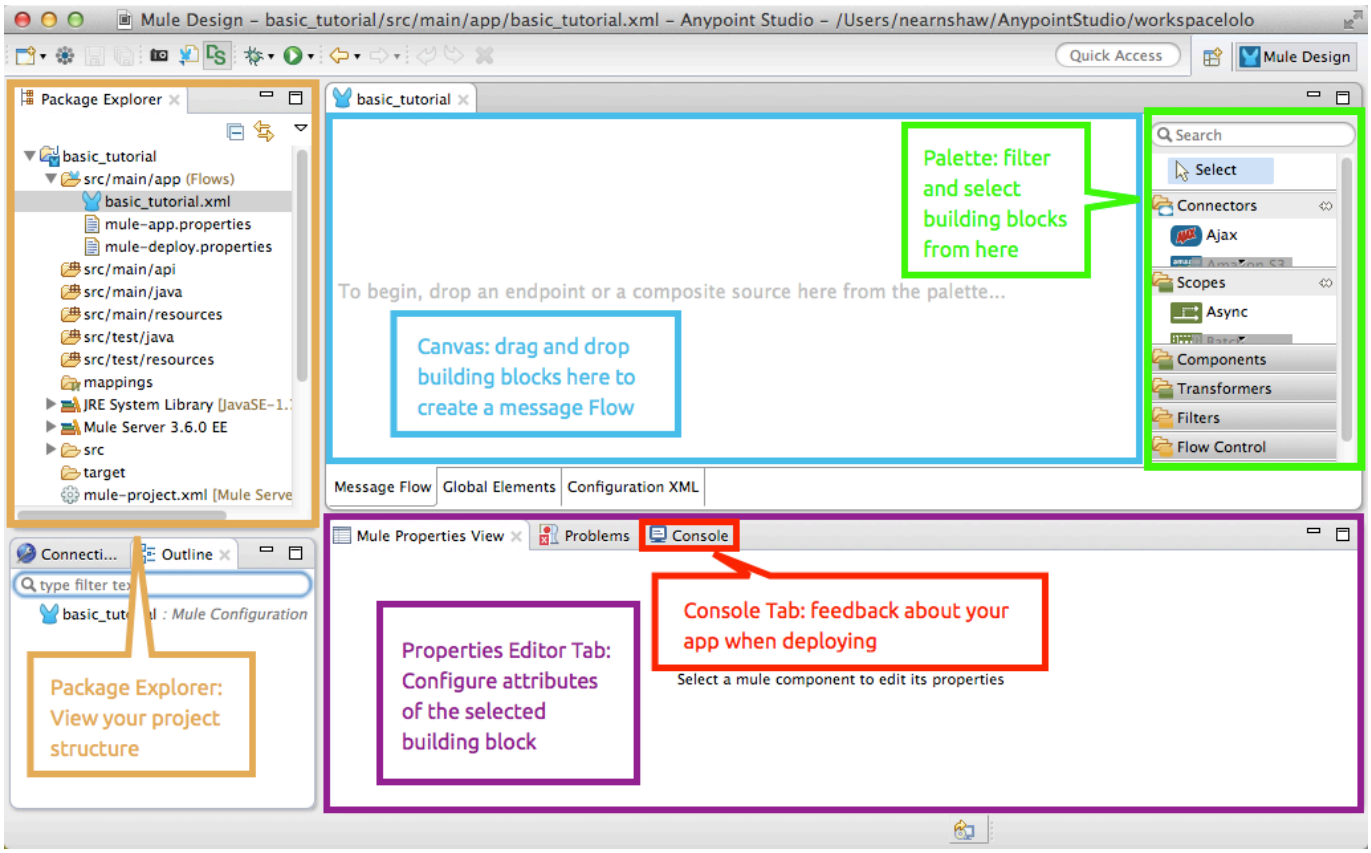

Figure: Visual distribution for new project (http://www.mulesoft.org/documentation/download/attachments/122751588/blank+canvas.png?version=1&modificationDate=1420213600545)

4. Drag and drop a **HTTP Connector** from the **palette** onto the **canvas**.

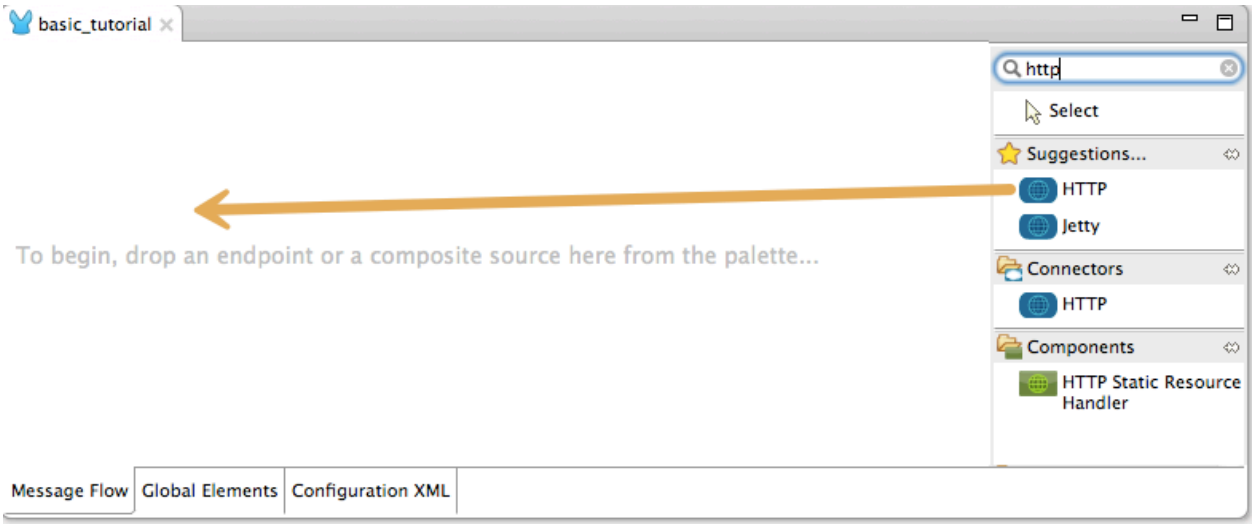

Figure: HTTP Connector (http://www.mulesoft.org/documentation/download/attachments/122751588/add+http.png?version=2&modificationDate=1420210646261).

5. Add a **Set Payload Component** to the flow.

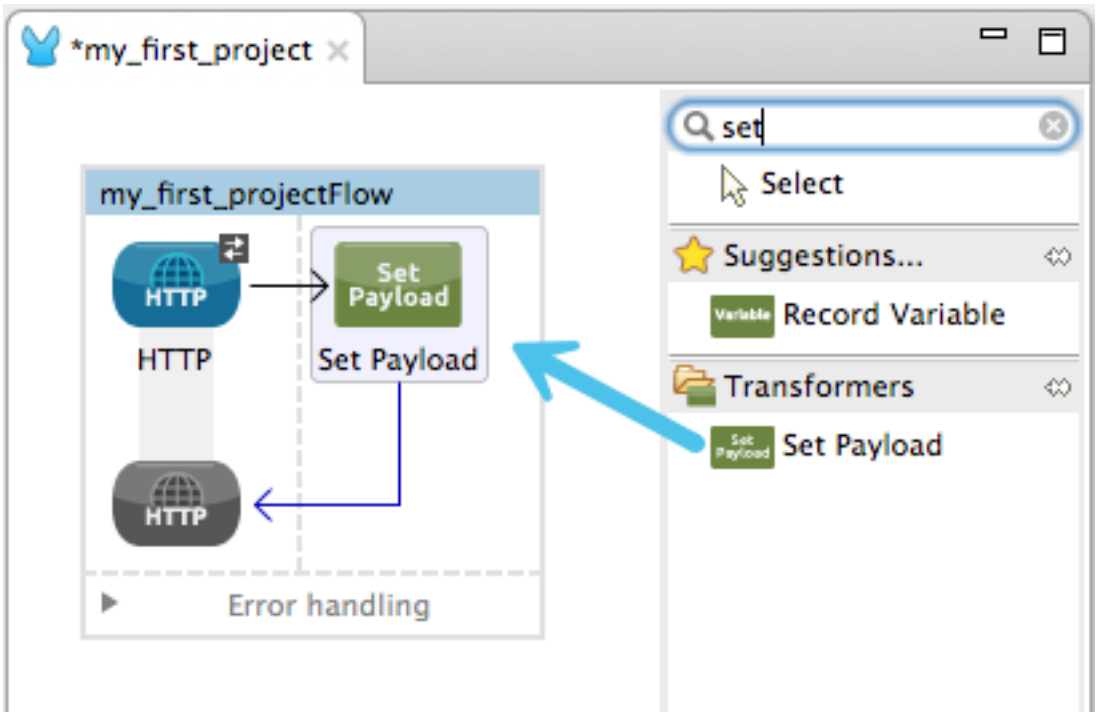

Figure: Set Payload component (http://www.mulesoft.org/documentation/download/attachments/122751588/set+payload.png?version=1&modificationDate=1421167826045).

6. Click the **Configuration XML** tab (at the base of the canvas) for the source of the flow right now.

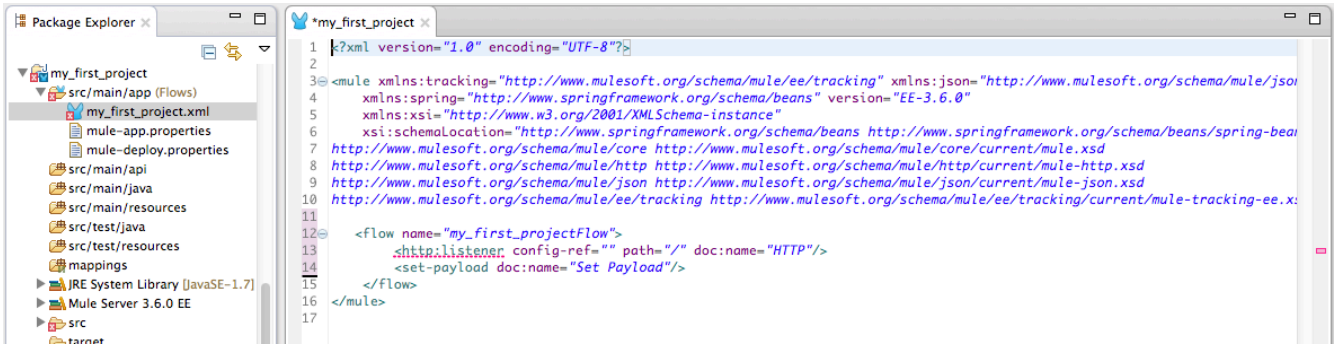

Figure: XML editor (http://www.mulesoft.org/documentation/download/attachments/122751588/xml+code.png?version=1&modificationDate=1421167951895).

7. Don’t forget to save the project.

Project configuration

a. After modeling the flow, the steps must be configured. Click the HTTP Connector and open its **Properties Editor**. (HTTP Connector allows the application to connect to web resources through the HTTP or HTTPS protocol).

Configure the two mandatory fields for the conector (**Path** and **Connector Configuration**).

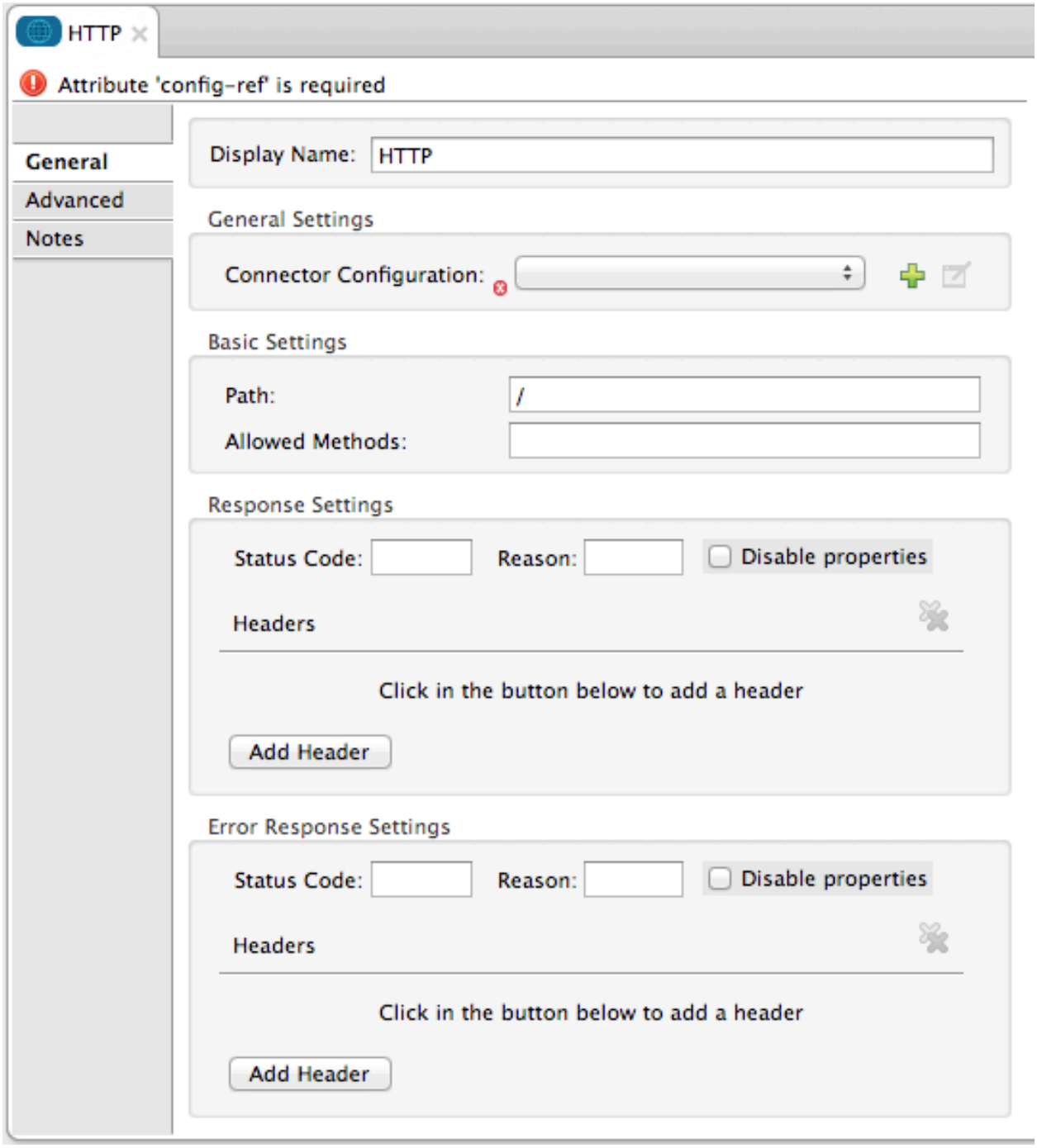

Figure: HTTP Connector properties (http://www.mulesoft.org/documentation/download/attachments/122751588/http+pelado.png?version=1&modificationDate=1421168361653).

b. By clicking the green plus sign next to **Connector configuration** a new **Global element** will be created and referenced by the connector.

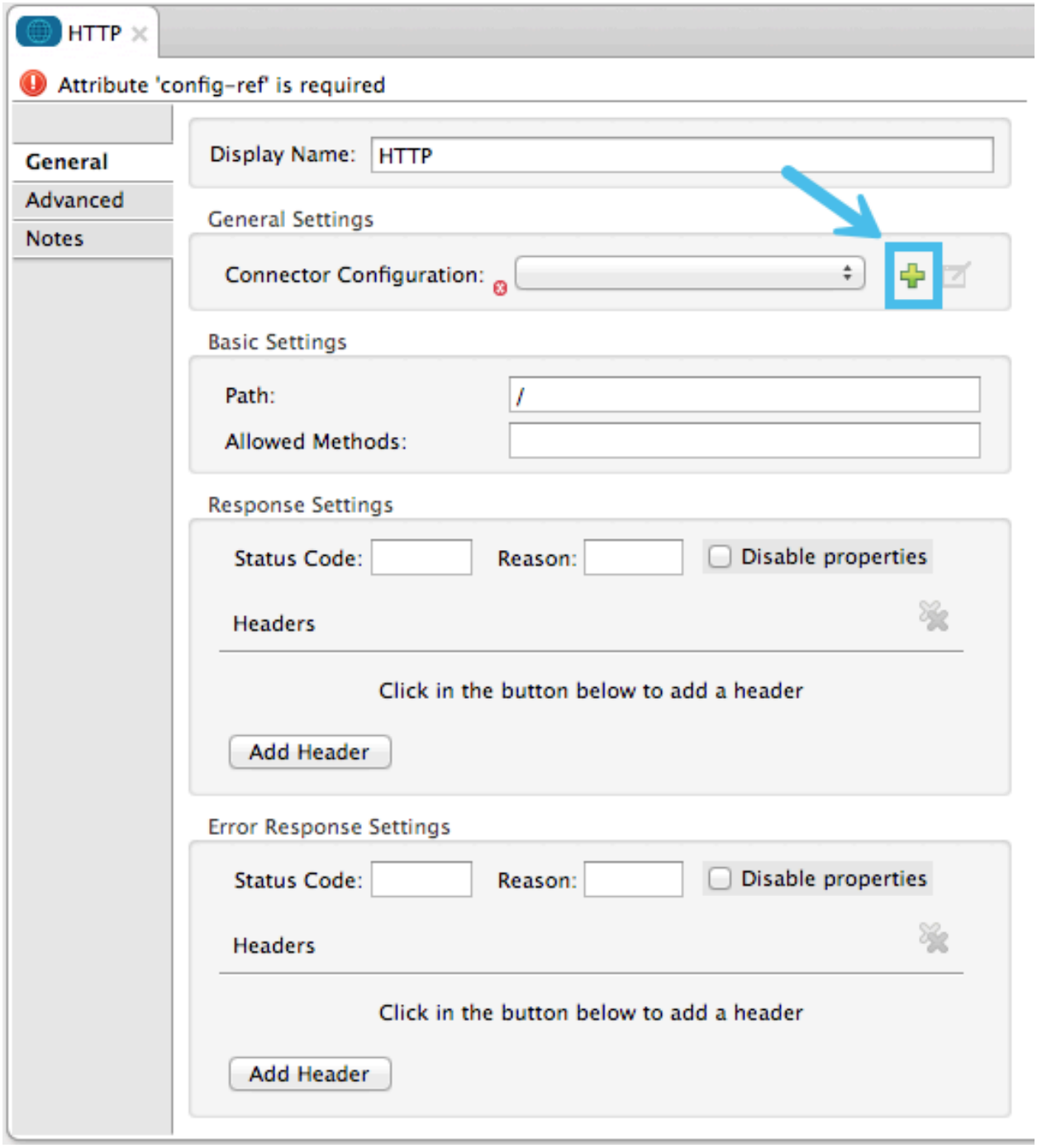

Figure: Global Element option (http://www.mulesoft.org/documentation/download/attachments/122751588/add+global+element.png?version=1&modificationDate=1421168137943).

c. Build the component leaving the fields with their default values.

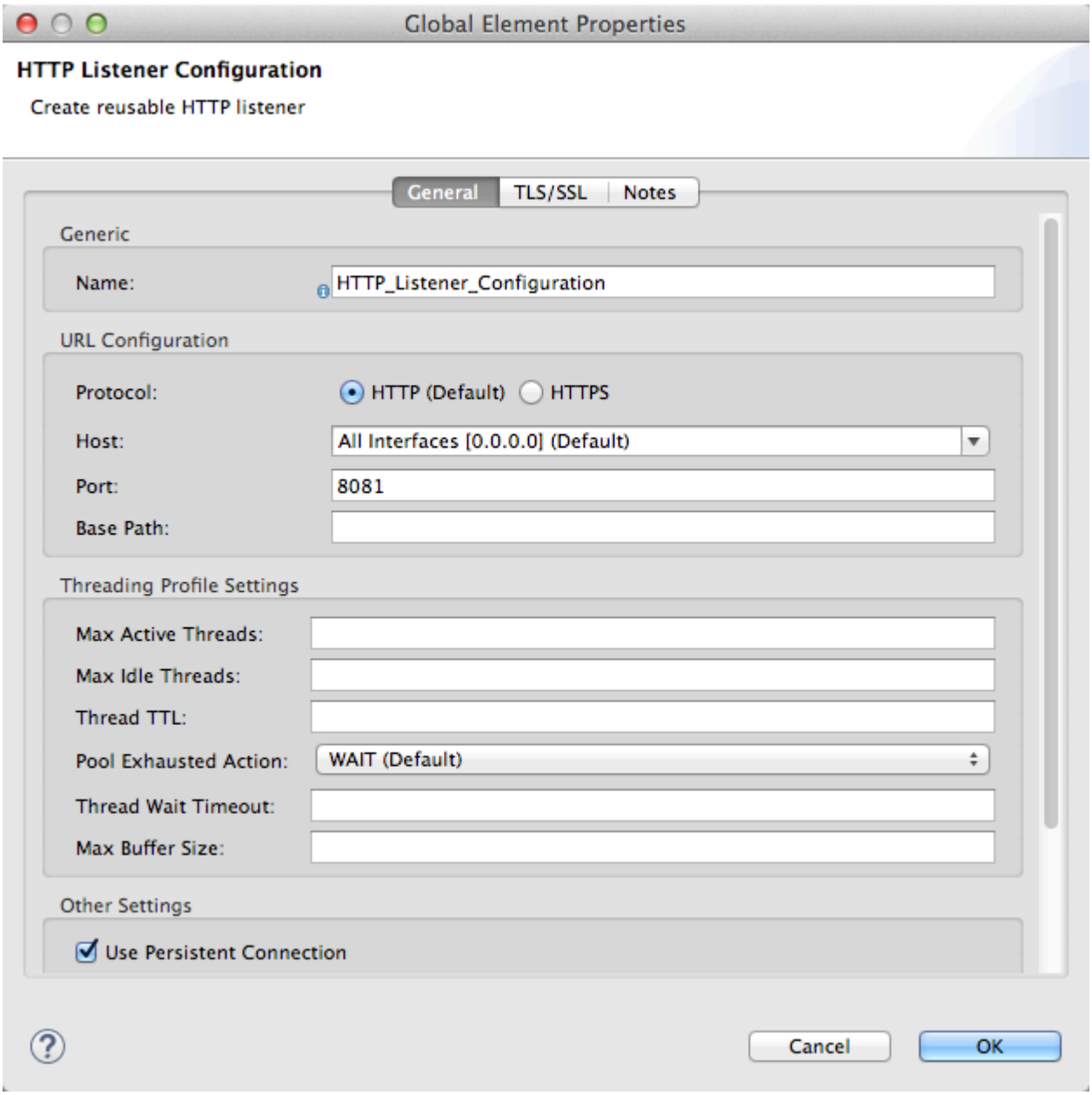

Figure: Global element configuration (http://www.mulesoft.org/documentation/download/attachments/122751588/global+element+default.png?version=1&modificationDate=1421168539490).

d. Click the **Set Payload Component** on the canvas to open its properties configuration.

e. Put a text in the **Value** field (this will be our Hello World message).

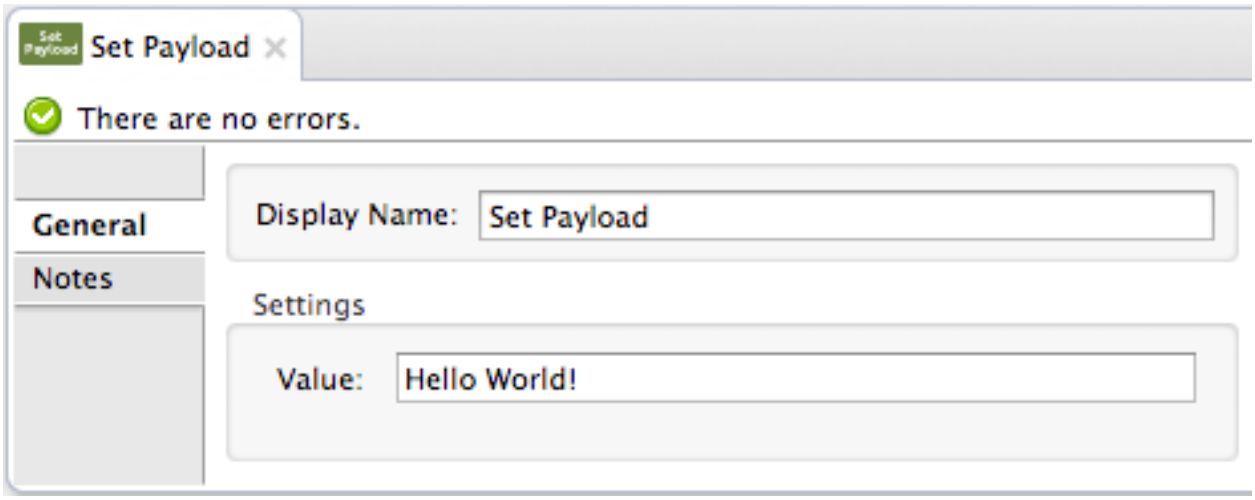

Figure: Set Payload component configuration (http://www.mulesoft.org/documentation/download/attachments/122751588/set+payload+hello+world.png?version=1&modificationDate=1421169294983).

f. Check the XML source to get a grasp of what is going on under the hood.

Deploy project

For the initial project the internal server shall be used.

a. In the **Package Explorer**, right click the project name and choose **Run As > Mule Application**.

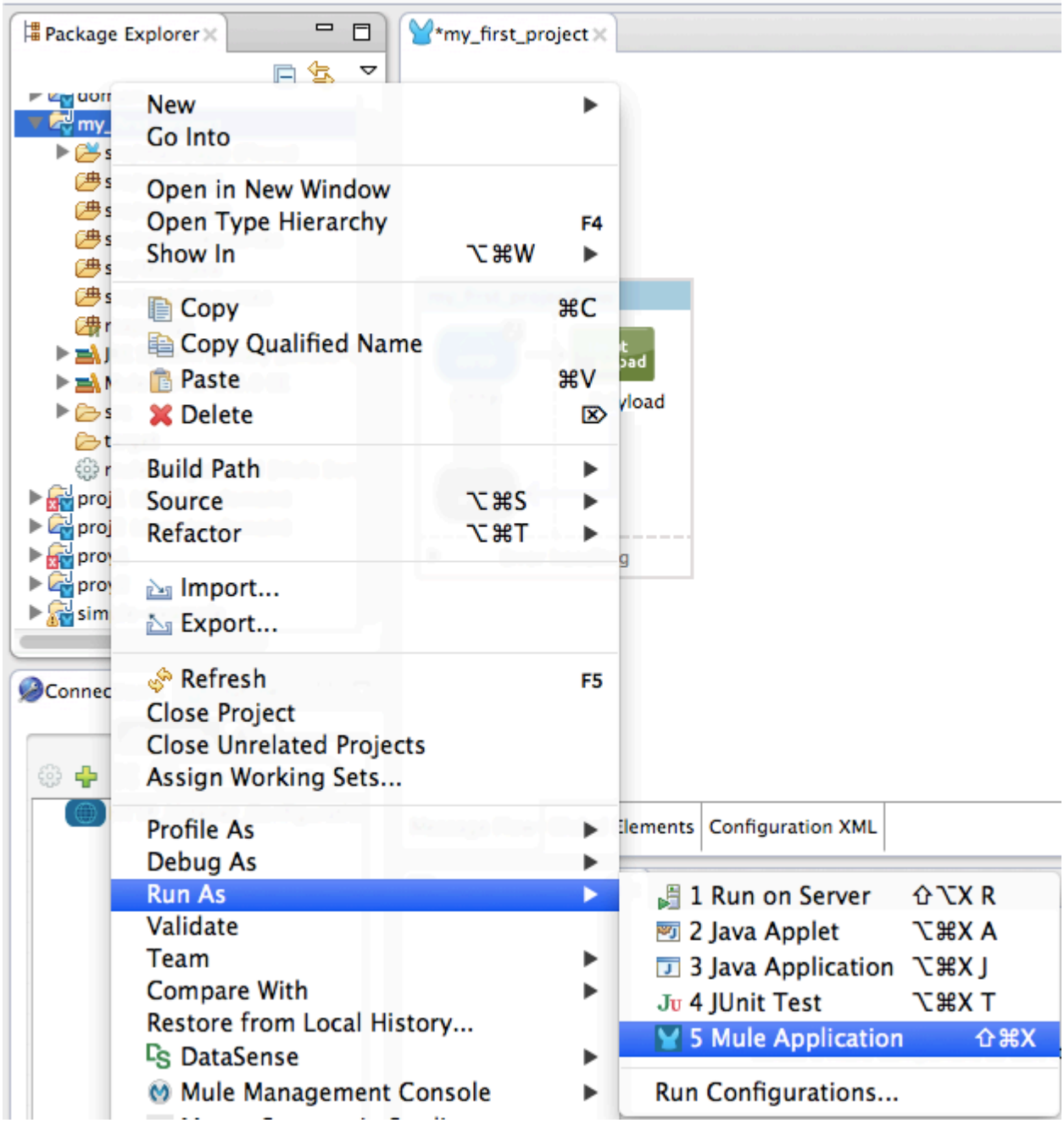

Figure: Run as mule application (http://www.mulesoft.org/documentation/download/attachments/122751588/run+as.png?version=2&modificationDate=1421247758944).

b. If everything goes right, the console once deployed should display the message *Started app XXXXX*.

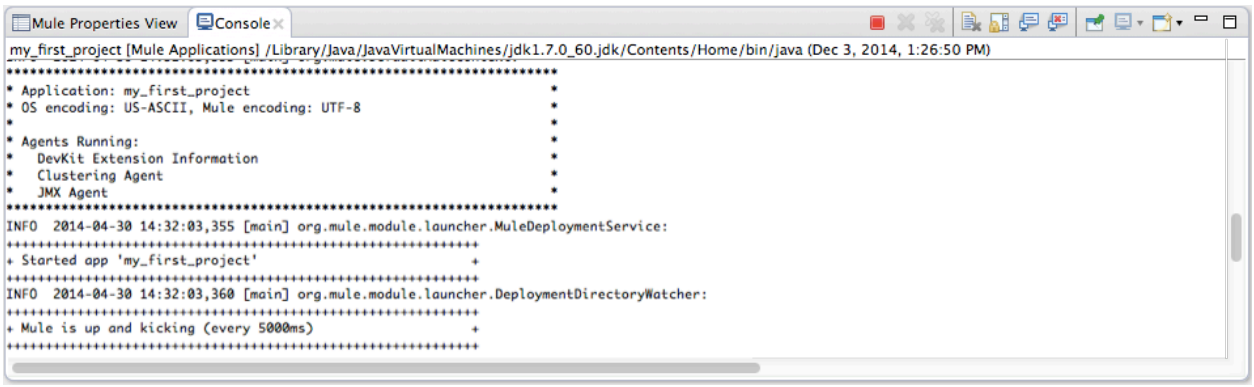

Figure: Console log (http://www.mulesoft.org/documentation/download/attachments/122751588/console1.png?version=1&modificationDate=1417633789156).

Interact with the Project

1. Open a web browser.
2. Navigate to the URL of the application (this address is the result of the default settings of the HTTP Connector).

http://localhost:8081

3. The application accepts the request and the **Set Payload** component sets the value to the one defined above. Finally the **HTTP Connector** returns this text as a response.

Figure: Web browser result (http://www.mulesoft.org/documentation/download/attachments/122751588/hello+world.png?version=2&modificationDate=1421169841596).

[5]

Bibliography

2/20/15 17:44

[1] Mule ESB Cookbook (Attune infocom) (Dr. Zakir Laliwala / Azaz Desai / Abdul Samad / Uchit Vyas)

[2] Anypoint Platform Primer (MuleSoft) (http://www.mulesoft.org/documentation/display/current/Anypoint+Platform+Primer)

[3] Begin with the Basics (MuleSoft) (http://www.mulesoft.org/documentation/display/current/Begin+with+the+Basics)

[4] Anypoint Studio Essentials (MuleSoft) (http://www.mulesoft.org/documentation/display/current/Anypoint+Studio+Essentials)

[5] Buy a Hello World Application (MuleSoft) (http://www.mulesoft.org/documentation/display/current/Build+a+Hello+World+Application)