

# INDICE

---

#

## INVOKES o INVOCADOR

---

Los **Invokes** en **QuickWin** se utilizan para ejecutar servicios Web. Hasta el momento son soportados servicios **REST** para los métodos **POST** y **GET**, y **SOAP**, método **POST**.

Las **Invokes** en **QuickWin** pueden probarse y configurarse de forma independiente. Tienen su propio contrato de ejecución así como también **endpoints** para refrescar caché y consulta.

## REQUEST POST A UN METODO POST

---

POST <http://192.168.37.146:8101/quickWin/executeInvoke> HTTP/1.1

```
{
  "invokeId": "705",
  "invokerName": "MANUAL-CONVERGENCIA",
  "cacheOptions": "0",
  "sync": "YES",
  "customerInvokerId": "id12344567",
  "externalTransactionId": "12563",
  "sessionData": {
    "externalSubscriberProperties": [
      {
        "id": "SERVICE_NUMBER",
        "value": [
          "985200097"
        ]
      },
      {
        "id": "OFFERING_ID",
        "value": [
          "100038136"
        ]
      },
      {
        "id": "BUSINESS_ACTION",
        "value": [
          "6"
        ]
      }
    ]
  }
}
```

**invokeId:** Indica el id numérico del Invocador a ejecutar.

**invokerName:** Representa el nombre de quien lo ejecuta. Cuando se ejecuta la campaña, internamente **QuickWin** inicializa este parámetro que identifica que parte de una transacción interna ejecuta un invocador (por ejemplo: **OPERATION\_INVOKE**, **PROVISIONING:PRODUCT,ENRICHMENT\_PRE\_DYNAMIC\_BLOCKING**). Los nombres son creado

por **QuickWin** en diferente partes donde se llaman a los invocadores.

**cacheOptions:** "0" representa tomar datos de la memoria caché y "1" recuperarlo directo de la base. Esta última no es una opción recomendada. Se recomienda usar 0, sólo si no puede refrescar caché entonces usar 1.

**sync:** YES representa **ASÍCRONO**, y NO representa **ASÍNCRONO**.

**customerInvokerId:** Representa el id de quien lo ejecuta. Cuando se ejecuta la campaña, internamente **QuickWin** inicializa este parámetros, y representa a la transacción de **QuickWin**. De esta forma se puede hacer tracking de todos los invocadores ejecutados para una transacción interna.

**externalTransactionId:** Representa el ID externo de quien lo ejecuta. Este valor es inicializado por **Quickwin** cuando recibe este parámetro en sus contrato. De esta forma se puede hacer tracking de todos los invocadores ejecutados para una transacción externa. **Este parámetro es opcional, pero se recomienda su uso.**

**sessionData:** Los invocadores hacen uso de **SubscriberProperties**. (Revise la sesión de **SubscriberProperties**). Para ello reciben una lista de SubscriberPropertie en el objeto Los invocadores hacen uso de **SubscriberProperties**. (Revise la sesión de **SubscriberProperties**). Para ello reciben una lista de **Subscriber Propertie** en el objeto **sessionData**

## RESPONSE POST JSON

```
{
  "invokeId": 705,
  "invokerName": "MANUAL-CONVERGENCIA",
  "success": false,
  "elapsedTime": 484,
  "code": -221,
  "message": "Ocurrió un error al invocar Supplementary Offering verifique
  ERROR_MESSAGE y ERROR_CODE",
  "externalSubscriberProperties": [
    {
      "id": "ORDER_ID",
      "value": [
        ""
      ]
    },
    {
      "id": "ERROR_CODE",
      "value": [
        "60107221254"
      ]
    },
    {
      "id": "ERROR_MESSAGE",
      "value": [
        "La oferta CONDICION PERMIN SA 0.99 ya fue desactivada y no
        puede desactivarse nuevamente."
      ]
    }
  ],
  "request": "\n{\n  \"idRequest\": \"permin:supplementaryOffering\", \n
  \"params\": {\n    \"serviceNumber\": \"985200097\", \n    \"offeringId\":
  \"100038136\", \n    \"businessAction\": \"6\" \n  } \n} \n",
```

```

"response": "{ \"code\":0, \"status\": \"OK\", \"message\": \"Successful Response\", \"headerResponse\": { \"X-ErrorCode\": \"60107221254\", \"X-ErrorMessage\": \"La oferta CONDICION PERMIN SA 0.99 ya fue desactivada y no puede desactivarse nuevamente.\" } }",
"responseHeader": "{ \"statusInfo\": [ \"OK\" ], \"Connection\": [ \"close\" ], \"Date\": [ \"Fri, 07 May 2021 21:24:04 GMT\" ], \"Content-Type\": [ \"application/json; charset=UTF-8\" ], \"status\": [ \"200\" ] }",
"ticket": "fbf0b1a2-7604-4b4f-9046-4d9dfd91d494",
"startingDate": "May 07, 2021 04:24:04 PM",
"endDate": "May 07, 2021 04:24:04 PM",
"externalTransactionId": "12563",
"customerInvokeId": "id12344567",
"retriesCount": 0,
"endpointUrl": "http://192.168.37.146:8082/rest/microGateway/invoke",
"httpMethod": "POST",
"headers": []
}

```

**invokeld:** Indica el id numérico del Invocador ejecutado.

```

select * from invoke_history t
where t.invoke_id = 705;

```

**invokerName:** Representa el nombre de quien lo ejecutó.

```

select * from invoke_history t
where t.invoker_name = 'MANUAL-CONVERGENCIA';

```

**success:** Indica si la evaluación de éxito de la ejecución del servicio web en caso de estar configurada, dio error o éxito. **true** o **false**. Cuando no se la configura, siempre será **true**.

**elapsedTime:** Indica el tiempo transcurrido desde que se lanza la ejecución hasta que termina. **ElapsedTime** nunca podrá ser mayor al tiempo configurado como **TIMEOUT** para los invocadores. Si es mayor en unos cuantos milisegundos, el invocador falla por **timeout**. Ver configuraciones para **EXTERNAL\_RESOURCES**.

**code:** Representa código de éxito o fracaso del invocador. Este código es configurable cuando se evalúa respuestas, y si no se configura es **0** para éxito, y **menor a 0 para errores**. Los errores que se evalúan son **timeout**, y mala configuración. Ver configuraciones para **INVOKE\_RESPONSE\_EVALUATION**.

**message:** Representa mensaje de éxito o fracaso del invocador. Este mensaje es configurable cuando se evalúa respuestas, y si no se configura es **OK** por defecto para éxito, y en caso de errores, se muestra el error respectivo. Por ejemplo **Not found, Timeout, etc.** Ver configuraciones para **INVOKE\_RESPONSE\_EVALUATION**.

**externalSubscriberProperties:** Representa a una lista de variables que el Invocador inicializa durante su llamado al servicio web. Estas variables se almacenan en **QuickWin** durante la transacción y pueden ser usadas con sus valores.

Estas variables pasan a formar parte del objeto **SessionData** interno de **QuickWin** donde almacena todas estas variables de forma temporal por cada transacción. A medida que **QuickWin** ejecuta invoke, va almacenando todas las variables en este objeto.

**Nota:** las variables del **SubscriberProperty** pueden cambiar de valores a medida que se procesan, por lo tanto, en los logs, solo veremos el último valor almacenado.

**request:** Representa el **request** que se envía al servicio web configurado. Este **request** es producto de la transformación de **XSLT**.

**response:** Es la respuesta del servicio web configurado. A partir de aquí, se toman datos para evaluar éxito o fracaso. Ver configuraciones para **INVOKE\_RESPONSE\_EVALUATION**.

**responseHeader:** Representa datos del **header**. Aquí se reciben variables que el servicio devuelve en el header. **Este responseHeader siempre vendrá en formato JSON.**

**status** y **statusInfo** siempre vendrán para **REST**. **status** representa el Código del Response.

**responseCode** y **responseMessage** siempre vendrán para **SOAP**. **responseCode** representa el Código del Response.

**ticket:** En si representa el id de transacción del Invocador. Se llama ticket porque cuando se ejecuta de forma asíncrona, este es el id que servirá para buscar en los logs.

```
select * from invoke_history
where invoke_history_id = 'fbf0b1a2-7604-4b4f-9046-4d9dfd91d494';
```

**startingDate:** Fecha de inicio de ejecución.

**startingDate:** Fecha de fin de ejecución.

**externalTransactionId:** Id externo de quien lo ejecutó.

```
select * from invoke_history t
where t.external_transaccion_id = '12563';
```

**customerInvokerId:** Id de transacción que lo ejecutó,

```
select * from invoke_history t
where t.customer_invoker_id = 'id12344567';
```

**retriesCount:** Número de veces que se reintentó en el caso de haberse configurado. El reintento de por sí solo de los invocadores no se recomienda en **QuickWin**. La configuración de reintentos solo aplicaba cuando se cae por **timeout** o errores generales con el **endpoint**.

**endpointUrl:** URL del servicio web que se ejecuta.

**headers:** Lista de variables de **headers** en caso de ser requeridas por el servicio web a ejecutarse.

## METODO SOAP POST CON RESPUESTA XML

POST <http://192.168.37.146:8101/quickWin/executeInvoke> HTTP/1.1

```
{
  "invokeId": "707",
  "invokerName": "Prueba Invoke",
  "cacheOptions": "0",
  "sync": "YES",
  "customerInvokerId": "id12345",
  "sessionData": {
    "externalSubscriberProperties": [
      {
        "id": "SERVICIO_JEIS_ID",
```

```

        "value": [
            "J1"
        ]
    }
}
}
}

```

## RESPONSE SOAP XML EMBEBIDO.

```

{
    "invokeId": 707,
    "invokerName": "Prueba Invoke",
    "success": true,
    "elapsedTime": 321,
    "code": 0,
    "message": "Invocacion ejecutada con exito",
    "externalSubscriberProperties": [
        {
            "id": "QUANTITY",
            "value": [
                "3"
            ]
        }
    ],
    "request": "<?xml version='1.0' encoding='UTF-8'?><soapenv:Envelope
xmlns:soapenv='http://schemas.xmlsoap.org/soap/envelope/'
xmlns:typ='http://axis/EISApiOnlineWS.wsd1/types/'><soapenv:Header/>
<soapenv:Body><typ:eipConsumeServicioElement><typ:dsId>jdbc/fwork</typ:dsId>
<typ:pnIdServicioInformacion>J919</typ:pnIdServicioInformacion>
<typ:pvParametroBind1>J1</typ:pvParametroBind1>
<typ:pvParametroBind2>A</typ:pvParametroBind2><typ:pvParametroBind3/>
<typ:pvParametroBind4/><typ:pvParametroBind5/><typ:sentencias_binds/>
</typ:eipConsumeServicioElement></soapenv:Body></soapenv:Envelope>",
    "response": "<?xml version='1.0' encoding='UTF-8'?><env:Envelope
xmlns:env='http://schemas.xmlsoap.org/soap/envelope/'
xmlns:xsd='http://www.w3.org/2001/XMLSchema'
xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
xmlns:ns0='http://axis/EISApiOnlineWS.wsd1/types/'><env:Header>
<work:WorkContext
xmlns:work='http://oracle.com/weblogic/soap/workarea/'>r00ABXdPABp3ZWJsb2dpVy5
hchAUamVpcy1hCHBzLWVhcgAAANYAAAAjd2VibG9nawMud29ya2FyZWUuU3Ryaw5nV29ya0NvbR1eHQ
ABHY2LjUAAA==</work:WorkContext></env:Header><env:Body>
<ns0:eipConsumeServicioResponseElement
xmlns:ns0='http://axis/EISApiOnlineWS.wsd1/types/'><ns0:result>
<ns0:perrorOut>0</ns0:perrorOut><ns0:pvresultadoOut>&lt;root>      &lt;rows>
      &lt;cantidad>3&lt;/cantidad>      &lt;/rows>&lt;/root></ns0:pvresultadoOut>
<ns0:perrorOut></ns0:perrorOut></ns0:result>
</ns0:eipConsumeServicioResponseElement></env:Body></env:Envelope>",
    "responseHeader": "{ \"Transfer-Encoding\": [\"chunked\"], \"X-ORACLE-DMS-
ECID\": [\"ced9749a-6640-44f8-84af-e5c6203868e7-00008e0d\"], \"readTimeout\":
[\"3000\"], \"requestMethod\": [\"POST\"], \"connectTimeout\":
[\"3000\"], \"responseMessage\": [\"OK\"], \"X-ORACLE-DMS-RID\": [\"0\"], \"Date\":
[\"Fri, 07 May 2021 22:44:03 GMT\"], \"Content-Type\": [\"text/xml; charset=utf-
8\"], \"responseCode\": [\"200\"]}",
    "ticket": "a52fd52a-3fb7-430d-8ee4-561c5f89786a",
    "startingDate": "May 07, 2021 05:44:03 PM",

```

```

"endDate": "May 07, 2021 05:44:03 PM",
"externalTransactionId": null,
"customerInvokeId": "id12345",
"retriesCount": 0,
"endpointUrl": "http://192.168.37.40:50004/jeis/jeisSoapHttpPort",
"httpMethod": "SOAP",
"headers": []
}

```

## REQUEST POST A UN METODO GET

```

{
  "invokeId": "660",
  "invokerName": "MANUAL-CONVERGENCIA",
  "cacheOptions": "0",
  "sync": "YES",
  "customerInvokerId": "id12344567",
  "sessionData": {
    "externalSubscriberProperties": [
      {
        "id": "$SUBSCRIBERID$",
        "value": [
          "siomi.lopez.xxxx@gmail.com"
        ]
      }
    ]
  }
}

```

## RESPONSE JSON-GET.

```

{
  "invokeId": 660,
  "invokerName": "MANUAL-CONVERGENCIA",
  "success": false,
  "elapsedTime": 72,
  "code": -221,
  "message": "DIO UN ERROR 404",
  "externalSubscriberProperties": [
    {
      "id": "CUSTOMER_ID_AMCO",
      "value": [
        ""
      ]
    },
    {
      "id": "NAMES",
      "value": [
        ""
      ]
    },
    {
      "id": "CODE_ERROR",
      "value": [
        "404"
      ]
    }
  ]
}

```

```

    ]
  }
],
"request":
"http://192.168.37.151:8282/claroId/v2/user/siomi.lopez.amco231@gmail.com?",
"response": "{\\"error\\":
{\\"statusCode\\":404,\\"name\\":\\"NotFoundError\\",\\"message\\":\\"Endpoint \\\\"GET
/user/siomi.lopez.amco231@gmail.com\\\\" not found.\\\"}}",
"responseHeader": "{\\"statusInfo\\":[\\"Not Found\\"],\\"Access-Control-Allow-
Origin\\":[\\"*\\"],\\"Access-Control-Allow-Credentials\\":[\\"true\\"],\\"X-Content-
Type-Options\\":[\\"nosniff\\"],\\"X-Transaction-Id\\":[\\"75838e08-a235-40a1-a6a6-
43f26684ceb5\\"],\\"Connection\\":[\\"keep-alive\\"],\\"Content-Length\\":
[\\"127\\"],\\"Date\\":[\\"Fri, 07 May 2021 23:02:39 GMT\\"],\\"X-Powered-By\\":
[\\"Express\\"],\\"Content-Type\\":[\\"application/json; charset=utf-8\\"],\\"status\\":
[\\"404\\\"]}",
"ticket": "e5c5c7cd-c2cf-43b3-9504-c3b166c52bb5",
"startingDate": "may 07, 2021 06:02:39 PM",
"endingDate": "may 07, 2021 06:02:40 PM",
"externalTransactionId": null,
"customerInvokeId": "id12344567",
"retriesCount": 0,
"endpointUrl":
"http://192.168.37.151:8282/claroId/v2/user/siomi.lopez.xxxx@gmail.com?",
"httpMethod": "GET",
"headers": []
}

```

## ENTENDIENDO XSLT DE INVOKE

Los **Invokes** arman **requests** que ejecutan un servicio web final. Los **request** son formados para los métodos **POST** haciendo uso de Plantillas **XSLT**.

Antes de Avanzar con configuraciones es importante hasta este punto es importante tener nociones generales de uso de XSLT.

**XSLT** es un mundo muy amplio, y sin embargo para los **Invocadores** lo vamos a mantener de la forma más sencilla posible. En caso de requerir casos avanzados como condiciones de XSLT, se recomienda revisar la documentación respectiva. [https://www.w3schools.com/xml/xsl\\_intro.asp](https://www.w3schools.com/xml/xsl_intro.asp)

## Cómo funciona desde Quickwin?

Básicamente **QuickWin** lo que hace es generar una lista de parámetros dentro de una etiqueta **params** con todos valores de todos los **SubscriberProperties** que usará el **Invocador**. Los nombres de la etiqueta como **\_key** son generados a partir de la tabla de **INVOKE\_MAPPINGS** con la columna **LABEL**.

```

<params>
  <_key>4586</_key>
</params>

```

Esto es lo que vemos en la primera parte donde dice: "Enter your XML code or it's URL"

Y como segundo paso, debemos proveer la plantilla **XSLT**.

Nota: **params** es la etiqueta que usa QuickWin para crear una lista de parámetros como vimos arriba, por lo tanto, debe usarse **params** siempre.

```
<xsl:for-each select="//params">
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:template match="/">
    <datos>
      <datosSubscriptor>
        <xsl:for-each select="//params">
          <subscriberId>
            <xsl:value-of select="_key"/>
          </subscriberId>
        </xsl:for-each>
      </datosSubscriptor>
    </datos>
  </xsl:template>
</xsl:stylesheet>
```

## Prueba XSLT ONLINE

---

Para probar nuestro XSLT podemos hacer directamente en la siguiente **URL** ingresando los dos **XML** en la siguiente URL. <https://xslttest.appspot.com/>



## Free Online XSLT Test Tool

NEW: added support for XSLT 2.0

Enter your XML code or it's URL:

[Tweet](#)

```
<params>
<_key>4586</_key>
</params>
```

Enter your XSLT or it's URL:

```
<xsl:template match="/">
  <datos>
    <datosSubscriber>
      <xsl:for-each select="//params">
        <subscriberId>
          <xsl:value-of select="_key"/>
        </subscriberId>
      </xsl:for-each>
    </datosSubscriber>
  </datos>
</xsl:template>
</xsl:stylesheet>
```

The result is:

```
<?xml version="1.0" encoding="UTF-8"?><datos><datosSubscriber><subscriberId>4586</subscriberId></datosSubscriber>
</datos>
```

[Run Tranformation](#)

[Open Result](#)

[Example 1](#)

[Example 2](#)

[Example 3](#)

[Example 4 XSLT 2.0](#)

[Clear](#)

[About](#)

## OTROS EJEMPLOS

### Lista de parámetros

Todos los ejemplos pueden ser probados en la pagina citada anterior. En la seccion "Enter your XML Code Here", ud debe ingresar una lista como esta con una raiz llamada "**params**" cambiando sus respectivos parámetros.

La siguiente lista puede usarse con el ejemplo de **CDATA**.

```
<params>
  <recipient><[CDATA[<Pedro Mayor> pmayorb@claro.com.ec]]></recipient>
  <sender>notificaciones@claro.com.ec</sender>
  <subject>Test Notificación MAIL ADJUNTO</subject>
  <text>Hello world</text>
  <type>M</type>
  <notificationClass>EDX</notificationClass>
</params>
```

### CDATA

**NOTA:** disable-output-escaping="yes", permite que lo que esta dentro de CDATA no sea escapado.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:not="http://www.claro.com.ec/UtilityServices/notify">
```

```

<soapenv:Header/>
<soapenv:Body>
  <not:Notify>
    <xsl:for-each select="//params">
      <notificationInfo>
        <recipient><xsl:value-of select="recipient"/></recipient>
        <sender><xsl:text disable-output-escaping="yes">&lt;!
[CDATA[</xsl:text><xsl:value-of select="sender" disable-output-escaping="yes"/>
<xsl:text disable-output-escaping="yes">]]&gt;</xsl:text></sender>
        <subject><xsl:value-of select="subject"/></subject>
        <message><xsl:value-of select="text"/></message>
        <template>
          <id><xsl:value-of select="idTemplate"/></id>
          <parameters>
            <parameter><xsl:value-of select="user"/></parameter>
            <parameter><xsl:value-of select="pass"/></parameter>
            <parameter><xsl:value-of select="mail"/></parameter>
          </parameters>
        </template>
        <type><xsl:value-of select="type"/></type>
        <notificationClass><xsl:value-of select="notificationClass"/>
      </notificationClass>
      <userId><xsl:value-of select="userId"/></userId>
    </notificationInfo>
  </xsl:for-each>
</not:Notify>
</soapenv:Body>
</soapenv:Envelope>
</xsl:template>
</xsl:stylesheet>

```

## CONDITION IF

Si una condición es verdadera, se ejecuta

```

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" indent="yes" media-type="text/json" omit-xml-
declaration="yes"/>
<xsl:template match="/">

  <xsl:for-each select="//params">
  {
    <xsl:if test="cod_tipo_ident= '007'">
      "Id_type": "PAS",
    </xsl:if>
    <xsl:if test="cod_tipo_ident ='002'">
      "Id_type": "CED",
    </xsl:if>
    <xsl:if test="cod_tipo_ident ='008'">
      "Id_type": "RUC",
    </xsl:if>
    "Id_number": "<xsl:value-of select="certificationNumber"/>"
  }
</xsl:for-each>

</xsl:template>
</xsl:stylesheet>

```

## FORMAT DATE

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:ex="http://exslt.org/dates-and-times" extension-element-prefixes="ex">
  <xsl:output method="text" indent="yes" media-type="text/json" omit-xml-
  declaration="yes"/>
  <xsl:template match="/">
    <xsl:variable name="fecha">
      <xsl:value-of select="format-dateTime(current-
  dateTime(), '[D01]/[M01]/[Y0001] [H]:[m01]:[s]')"/>
    </xsl:variable>
    {
      <xsl:for-each select="//params">
        .. parameters
      </xsl:for-each>
    }
  </xsl:template>
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xsl:output method="text" indent="yes" media-type="text/json" omit-xml-
  declaration="yes"/>
  <xsl:template match="/">
    <xsl:variable name="dt" as="xs:dateTime" select="current-dateTime()"/>
    {<xsl:for-each select="//params">
      "idRequest": "claro-video:deleteUserOTT",
      "params": {
        "correlatorId": "<xsl:value-of select="format-dateTime($dt, '[Y0001][M01]
[D01][H01][m01][s01][f]')"/>",
        "userId": "86",
        "paymentMethod": "<xsl:value-of select="paymentMethod"/>",
        "account": "<xsl:value-of select="account"/>",
        "employeeId": "<xsl:value-of select="employeeId"/>",
        "customerId": "<xsl:value-of select="customerId"/>",
        "origin": "CAC_EC",
        "providerId": "AMCOOTT_EC",
        "iccidManager": "AMCOEC",
        "transactionLevelType": "3"
      }
    </xsl:for-each>
  }
</xsl:template>
</xsl:stylesheet>
```

## ENTENDIENDO JSONPATH o XPATH DE INVOKE

Los **Invokes** arman **requests** que ejecutan un servicio web final haciendo uso de **XSLT** como vimos en la sesión anterior; sin embargo, también necesitamos procesar los **response o respuesta** obtenidas. Para poder leer las respuesta de un **Invoke** hacemos uso de **XPATH** para **XML** y **JSON PATH** para **JSON**. Es una de las razones por las cuales en las configuraciones se pregunta por el tipo de respuesta para poder determinar si la expresión de extracción de datos a usar en **JSON PATH o XPATH** para respuesta **JSON** o **XML** respectivamente.

## Ejemplo de Uso de JSONPATH.

Este ejemplo fue tomado de GIT.

<https://github.com/json-path/JsonPath>

Dado el siguiente ejemplo.

```
{
  "store": {
    "book": [
      {
        "category": "reference",
        "author": "Nigel Rees",
        "title": "Sayings of the Century",
        "price": 8.95
      },
      {
        "category": "fiction",
        "author": "Evelyn Waugh",
        "title": "Sword of Honour",
        "price": 12.99
      },
      {
        "category": "fiction",
        "author": "Herman Melville",
        "title": "Moby Dick",
        "isbn": "0-553-21311-3",
        "price": 8.99
      },
      {
        "category": "fiction",
        "author": "J. R. R. Tolkien",
        "title": "The Lord of the Rings",
        "isbn": "0-395-19395-8",
        "price": 22.99
      }
    ],
    "bicycle": {
      "color": "red",
      "price": 19.95
    }
  },
  "expensive": 10
}
```

# Guía rápida de JSONPATH

EJEMPLO	DESCRIPCION
<code>\$.store.book[*].author</code>	Los autores de todos los libros
<code>\$..author</code>	Todos los autores
<code>\$.store.*</code>	Todo, tanto <b>books</b> como <b>bicycles</b>
<code>\$.store..price</code>	El precio de cada cosa
<code>\$..book[2]</code>	El tercer libro
<code>\$..book[-2]</code>	El segundo libro antes del final.
<code>\$..book[0,1]</code>	Los primeros dos libros
<code>\$..book[:2]</code>	Todos los libros desde el índice 0 (inclusivo) hasta el índice 2 (exclusivo)
<code>\$..book[1:2]</code>	Todos los libros desde el índice 1 (inclusivo) hasta el índice 2 (exclusivo)
<code>\$..book[-2:]</code>	Los dos últimos libros.
<code>\$..book[2:]</code>	El libro numero dos desde abajo del arreglo.
<code>\$..book[?(@.isbn)]</code>	Todos los libros que contienen numero ISBN
<code>\$.store.book[?(@.price &lt; 10)]</code>	Todos los libros en store de precio menores que 10

**Pruebe los ejemplos en el siguiente enlace.**

[http://jsonpath.herokuapp.com/?path=\\$.store.book\[\\*\].author](http://jsonpath.herokuapp.com/?path=$.store.book[*].author)

← → ↺

No es seguro | jsonpath.herokuapp.com/?path=\$.store.book[\*].author

Aplicaciones Traductor de Google Biblioteca de audio... GB! Terms at Grammar... IELTS BSL - Google... IELTS Exam Prepara... IELTS General Readi...

Jayway JsonPath Evaluator

2.5.0 - 2020-12-11 09:01:21

Goessner example

```
{
  "category": "fiction",
  "author": "Evelyn Waugh",
  "title": "Sword of Honour",
  "price": 12.99
},
{
  "category": "fiction",
  "author": "Herman Melville",
  "title": "Moby Dick",
  "isbn": "0-553-21311-3",
  "price": 8.99
},
{
  "category": "fiction",
  "author": "J. R. R. Tolkien",
  "title": "The Hobbit",
  "price": 5.99
}

```

\$..store.book[\*].author

Go!

Jayway Gatling Nebhale Goessner

0 millis

```
[
  "Nigel Rees",
  "Evelyn Waugh",
  "Herman Melville",
  "J. R. R. Tolkien"
]
```

## XPATH

# DOCUMENTACION ONLINE

[http://www-db.deis.unibo.it/courses/TW/DOCS/w3schools/xsl/xpath\\_syntax.asp.html](http://www-db.deis.unibo.it/courses/TW/DOCS/w3schools/xsl/xpath_syntax.asp.html)

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book>
    <title lang="en">Harry Potter</title>
    <price>29.99</price>
  </book>
  <book>
    <title lang="en">Learning XML</title>
    <price>39.95</price>
  </book>
</bookstore>
```

## GUIA RÁPIDA

### Selección de Nodos

**XPath** usa expresiones de ruta para seleccionar nodos en un documento XML. El nodo se selecciona siguiendo una ruta o pasos. Las expresiones de ruta más útiles se enumeran a continuación:

Expression	Description
<i>nodename</i>	Selects all nodes with the name " <i>nodename</i> "
/	Selects from the root node
//	Selects nodes in the document from the current node that match the selection no matter where they are
.	Selects the current node
..	Selects the parent of the current node
@	Selects attributes

En la siguiente tabla, hemos enumerado algunas expresiones de ruta y el resultado de las expresiones:

Path Expression	Result
bookstore	Selects all nodes with the name "bookstore"
/bookstore	Selects the root element bookstore <b>Note:</b> If the path starts with a slash ( / ) it always represents an absolute path to an element!
bookstore/book	Selects all book elements that are children of bookstore
//book	Selects all book elements no matter where they are in the document
bookstore//book	Selects all book elements that are descendant of the bookstore element, no matter where they are under the bookstore element
//@lang	Selects all attributes that are named lang

## Predicados

Los predicados se utilizan para encontrar un nodo específico o un nodo que contiene un valor específico.

Los predicados siempre se incluyen entre corchetes.

En la siguiente tabla, hemos enumerado algunas expresiones de ruta con predicados y el resultado de las expresiones:

Path Expression	Result
/bookstore/book[1]	Selects the first book element that is the child of the bookstore element. <b>Note:</b> In IE 5,6,7,8,9 first node is[0], but according to W3C, it is [1]. To solve this problem in IE, set the SelectionLanguage to XPath: <i>In JavaScript: xml.setProperty("SelectionLanguage","XPath");</i>
/bookstore/book[last()]	Selects the last book element that is the child of the bookstore element
/bookstore/book[last()-1]	Selects the last but one book element that is the child of the bookstore element
/bookstore/book[position()<3]	Selects the first two book elements that are children of the bookstore element
//title[@lang]	Selects all the title elements that have an attribute named lang
//title[@lang='en']	Selects all the title elements that have a "lang" attribute with a value of "en"
/bookstore/book[price>35.00]	Selects all the book elements of the bookstore element that have a price element with a value greater than 35.00
/bookstore/book[price>35.00]/title	Selects all the title elements of the book elements of the bookstore element that have a price element with a value greater than 35.00

## Seleccionando nodos desconocidos

Los comodines **XPath** se pueden utilizar para seleccionar nodos XML desconocidos.

Wildcard	Description
*	Matches any element node
@*	Matches any attribute node
node()	Matches any node of any kind

En la siguiente tabla, hemos enumerado algunas expresiones de ruta y el resultado de las expresiones:

Path Expression	Result
/bookstore/*	Selects all the child element nodes of the bookstore element
//*	Selects all elements in the document
//title[@*]	Selects all title elements which have at least one attribute of any kind

## Seleccionando varias rutas

Utilizando el | operador en una expresión XPath puede seleccionar varias rutas.

En la siguiente tabla, hemos enumerado algunas expresiones de ruta y el resultado de las expresiones:

Path Expression	Result
//book/title   //book/price	Selects all the title AND price elements of all book elements
//title   //price	Selects all the title AND price elements in the document
/bookstore/book/title   //price	Selects all the title elements of the book element of the bookstore element AND all the price elements in the document

## PROBANDO XPATH ONLINE

<https://www.freeformatter.com/xpath-tester.html>



← → ↻ freeformatter.com/xpath-tester.html

Aplicaciones Traductor de Google Biblioteca de audio... GB Terms at Grammar... IELTS BSL - Google... IELTS Exam Prepara... IELTS General Read...

**FREEFORMATTER.COM** Contact

- JSON Validator
- HTML Validator
- XML Validator - XSD
- XPath Tester**
- Credit Card Number Generator & Validator
- Regular Expression Tester (RegEx)
- Java Regular Expression Tester (RegEx)
- Cron Expression Generator - Quartz

**Encoders & Decoders**

- Url Encoder & Decoder
- Base 64 Encoder & Decoder
- Convert File Encoding
- QR Code Generator

**Code Minifiers / Beautifier**

- JavaScript Beautifier
- CSS Beautifier
- JavaScript Minifier
- CSS Minifier

**Converters**

- XSD Generator
- XSLT (XSL Transformer)
- XML to JSON Converter
- JSON to XML Converter
- CSV to XML Converter
- CSV to JSON Converter

reason and to prevent your browser from being unresponsive.

**XML Input**

Option 1: Copy-paste your XML document here

```
<bookstore>
  <book>
    <title lang="en">Harry Potter</title>
    <price>29.99</price>
  </book>
  <book>
    <title lang="en">Learning XML</title>
    <price>39.95</price>
  </book>
</bookstore>
```

Option 2: Or upload your XML document

Seleccionar archivo Ningún archivo seleccionado UTF-8

**XPath expression**

`/bookstore/book/price[text()]`

Include the XML item type in output:

No

TEST XPATH TEST XPATH IN NEW WINDOW

## Seleccionando todos los precios

El siguiente ejemplo selecciona el texto de todos los nodos de precios .:

```
<price>29.99</price>
```

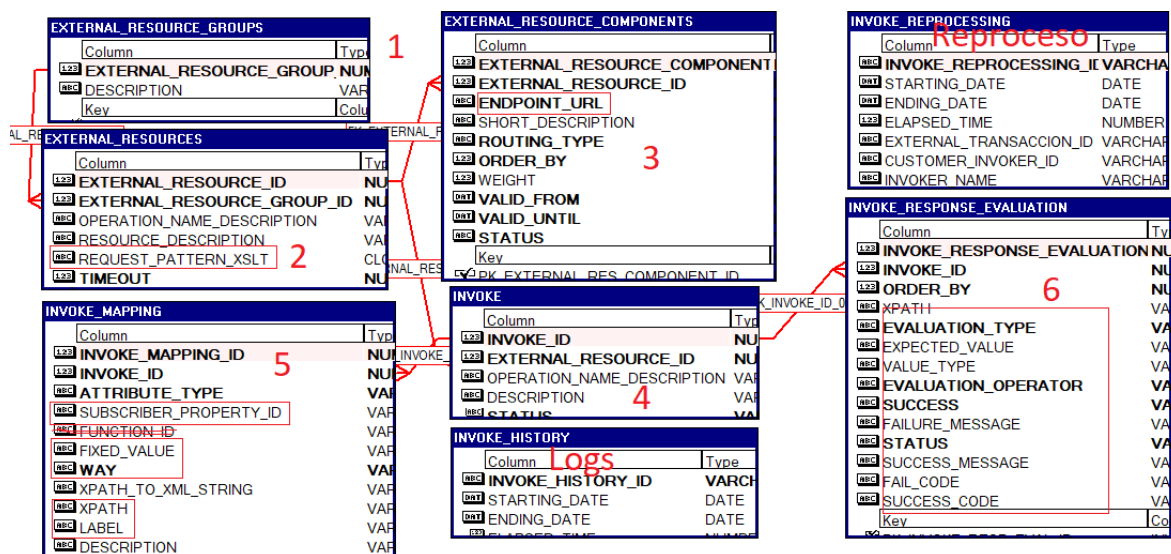
```
<price>39.95</price>
```

Para mas ejemplo, visite:

[http://www-db.deis.unibo.it/courses/TW/DOCS/w3schools/xsl/xpath\\_examples.asp.html](http://www-db.deis.unibo.it/courses/TW/DOCS/w3schools/xsl/xpath_examples.asp.html)

# ANATOMIA DEL INVOKE O INVOCADOR

## MODELO DE ENTIDADES



## 1.- QUICKWIN\_MDS.EXTERNAL\_RESOURCE\_GROUPS

Esta tabla, sirve para identificar un grupo de invocadores.

NAME	TYPE	NULLABLE	DEFAULT
EXTERNAL_RESOURCE_GROUP_ID	NUMBER	N	
DESCRIPTION	VARCHAR2(255)	Y	

**EXTERNAL\_RESOURCE\_GROUP\_ID:** Id numérico.

**DESCRIPTION:** Descripción de Grupo.

## 2.- QUICKWIN\_MDS.EXTERNAL\_RESOURCES

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
EXTERNAL_RESOURCE_ID	NUMBER	N		
EXTERNAL_RESOURCE_GROUP_ID	NUMBER	N		
OPERATION_NAME_DESCRIPTION	VARCHAR2(255)	Y		
RESOURCE_DESCRIPTION	VARCHAR2(255)	Y		
REQUEST_PATTERN_XSLT	CLOB	Y		
TIMEOUT	NUMBER	N	1000	Timeout en Milisegundos
STATUS	VARCHAR2(1)	N	'A'	
EXT_RES_TYPE	VARCHAR2(5)	Y	'SOAP'	
REQUEST_METHOD	VARCHAR2(5)	Y	'POST'	
CONTENT_TYPE	VARCHAR2(50)	Y		
DATA_TYPE	VARCHAR2(50)	Y		

**EXTERNAL\_RESOURCE\_ID:** Id numérico.

**EXTERNAL\_RESOURCE\_GROUP\_ID:** Id numérico de **Exteranal Resource Group**.

**OPERATION\_NAME\_DESCRIPTION:** Descripción de operación.

**RESOURCE\_DESCRIPTION:** Descripción de recurso.

**REQUEST\_PATTERN\_XSLT:** Plantilla XSLT para contratos XML y JSON. Para métodos **GET** existe una plantilla exclusiva definida.

**TIMEOUT:** Tiempo de conexión de red y espera en milisegundos.

**STATUS:** Estado A de Activo, I de Inactivo.

**EXT\_RES\_TYPE:** Tipo de **External Resource**, SOAP o REST.

**REQUEST\_METHOD:** Es **POST** o **GET**, soportados hasta el momento. Cuando el método es GET no se usa una plantilla XLST, sino mas bien un OBJETO JSON STRING que describe los tipos de parámetros que soporta el GET configurado. (**Revisar definición de plantilla para GET**)

**CONTENT\_TYPE:** Para **JSON**: **application/json**, **XML**: **text/xml**, **NO APLICA CON METODOS GET**. XML es usado con **EXT\_REST\_TYPE: SOAP**, incluso puede usarse con REST cuando el requerimiento es un XML.

**DATA\_TYPE:** Indica si la respuesta del **RESPONSE** es **XML** o **JSON**. De esta forma un XPATH puede entenderse para JSON como JSONPATH, y XML como XPATH.

### 3.- QUICKWIN\_MDS.EXTERNAL\_RESOURCE\_COMPONENTS.

Esta tabla contiene o puede contener muchas configuraciones de **URLs** relacionada a un **External Resource**. La idea es dar soporte de balanceador. Sin embargo esta tabla está descontinuada, y solo se toma por defecto la primera configuración de la **URL** con precedencia de orden menor a mayor definido por "**ORDER\_BY**".

NAME	TYPE	NULLABLE	DEFAULT
EXTERNAL_RESOURCE_COMPONENT_ID	NUMBER	N	
EXTERNAL_RESOURCE_ID	NUMBER	N	
ENDPOINT_URL	VARCHAR2(200)	N	
SHORT_DESCRIPTION	VARCHAR2(255)	Y	
ROUTING_TYPE	VARCHAR2(1)	N	
ORDER_BY	NUMBER	N	
WEIGHT	NUMBER	Y	
VALID_FROM	DATE	N	SYSDATE
VALID_UNTIL	DATE	N	to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS')
STATUS	VARCHAR2(1)	N	'A'

**EXTERNAL\_RESOURCE\_COMPONENT\_ID:** Identificador numerico.

**EXTERNAL\_RESOURCE\_ID:** Identificador del External Resource.

**ENDPOINT\_URL:** URL o ENDPOINT donde esa alojado el Servicio Web.

**SHORT\_DESCRIPTION:** Descripción corta.

**ROUTING\_TYPE:** B de balanceo. Actualmente esta configuración no esta validada.

**ORDER\_BY:** Precedencia en la lista de URL.

**WEIGHT:** 100 (100%). Actualmente esta configuración no esta validada.

**VALID\_FROM:** Desde cuándo es aplica.

**VALID\_UNTIL:** Hasta cuando es aplica.

**STATUS:** Estado A de Activo, I de Inactivo.

## 4.- QUICKWIN\_MDS.INVOKE

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
INVOKE_ID	NUMBER	N		
EXTERNAL_RESOURCE_ID	NUMBER	N		
OPERATION_NAME_DESCRIPTION	VARCHAR2(255)	Y		
DESCRIPTION	VARCHAR2(255)	Y		
STATUS	VARCHAR2(1)	N	'A'	
RETRIES	NUMBER	Y	0	Numero de Reintentos para este invoke en caso de falla
TIME_BETWEEN_RETRIES	NUMBER	Y	1000	Tiempo que debe esperar entre reintentar en Milisegundos

**INVOKE\_ID:** Identificador Numérico.

**EXTERNAL\_RESOURCE\_ID:** Id del External Resource.

**OPERATION\_NAME\_DESCRIPTION:** Descripción de la operación que llama el Invoke.

**DESCRIPTION:** Descripción del Invoke o Invocador.

**STATUS:** Estado A de Activo, I de Inactivo.

**RETRIES:** Ver comentario. Solo aplica para **TIMEOUT** u algún otro tipo de Error imprevisto.

**TIME\_BETWEEN\_RETRIES:** Ver comentarios. Actualmente solo se soporta 1 segundo. Siempre y cuando el numero de RETRIES sea mayor a 0 y falle.

## 5.- QUICKWIN\_MDS.INVOKE\_MAPPING

Esta tabla sirve para el mapeo de **SUBSCRIBER\_PROPERTIES** hacia la plantilla **XSLT** la cual contiene una palabra clave. Esta palabra clave se la identifica con la columna "**LABEL**". **LABEL** solo aplica cuando se envía datos al REQUEST y HEADER. La plantilla usada para GET también usa **LABEL** para el mapeo.

Para saber que los datos se envían al **HEADER**, **REQUEST** o se toman del **RESPONSE** o **RESPONSE\_HEADER** se usa "**WAY**".

Cuando se recuperan los datos con **JSONPATH** o **XPATH** se puede indicar si se requiere recuperar todo el nodo o solo un valor como texto. Para eso usamos **TYPE\_TO\_EXTRACT**.

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
INVOKE_MAPPING_ID	NUMBER	N		
INVOKE_ID	NUMBER	N		
ATTRIBUTE_TYPE	VARCHAR2(10)	N		Indica el tipo de atributo a configurarse
SUBSCRIBER_PROPERTY_ID	VARCHAR2(255)	Y		
FUNCTION_ID	VARCHAR2(255)	Y		
FIXED_VALUE	VARCHAR2(2000)	Y		
WAY	VARCHAR2(3)	N	'=>'	
XPATH_TO_XML_STRING	VARCHAR2(2000)	Y		En caso de que el XML este envuelto en un TAG como STRING y sobre este se quiere trabajar.
XPATH	VARCHAR2(2000)	Y		Path xml o Path JSON
LABEL	VARCHAR2(255)	Y		
DESCRIPTION	VARCHAR2(255)	Y		
STATUS	VARCHAR2(1)	N	'A'	
TYPE_TO_EXTRACT	VARCHAR2(32)	N	'TEXT'	Campo que indica como tratar la extracci3n del JSON Path. Si es 'TEXT' la expresi3n debe devolver el valor de un campo JSON.. Si es Object la expresi3n puede devolver cualquier cosa, la misma se almacenar3 en el SP configurado.

**INVOKE\_MAPPING\_ID:** Identificador Num3rico.

**INVOKE\_ID:** Id del Invoke o Invocador.

**ATTRIBUTE\_TYPE:** **SP** cuando el valor que se mapea con el **LABEL** es tomado de un **SUBSCRIBER\_PROPERTY**, **FV** cuando el valor se lo env3a en "**FIXED\_VALUE**" y **FN** para funciones. Sin embargo esta opci3n est3 descontinuada. Funciones no est3 soportada.

**SUBSCRIBER\_PROPERTY\_ID:** Aplica cuando "**ATTRIBUTE\_TYPE: SP**"

**FUNCTION\_ID:** Descontinuado. No tiene soporte.

**FIXED\_VALUE:** Aplica cuando "**ATTRIBUTE\_TYPE: FV**". Es recomendable el USO de **FIXED\_VALUE** cuando se requiere reutilizar una plantilla XSLT y no mantener este valor fijo directamente en la plantilla .

**WAY:** >> cuando los datos se envían a la cabecera del **Webservice**, => cuando los datos se envían al **request**, ^^ cuando los datos forman parte de la URL especialmente para GET. En todos estos casos es necesario configurar **LABEL**. <= Para almacenar datos en los **SUBSCRIBER\_PROPERTIES** extraídos del response, sin embargo también se pueden extraer datos del **responseHeader** con <<.

**Recordar que cuando WAY es "<<" , XPATH debe ser siempre un JSONPATH.**

**XPATH\_TO\_XML\_STRING:** Aplica para XML embebidos como XMLSTRING en un RESPONSE. En este caso este es el XML del response que se le aplicará el XPATH. Ver comentario. Normalmente XML que contiene un XML como STRING.

**XPATH:** Contiene la expresión **XPATH** o **JSON PATH** que se utiliza para la extracción de valores. Los **XPATHs** pueden devolver una lista de valores **STRING** y todos estos pueden ser almacenados en un **SUBSCRIBER\_PROPERTIES** con una lista de VALORES. Por ejemplo.

```
$.response.offers.offer[*].offer[*] - resultado ["OFFER_1","OFFER_2"]
```

Almacena todas las ofertas en un **subscriberProperty** definido por **SUBSCRIBER\_PROPERTY\_ID**

```
(**subscriberProperties**)SP = ["OFFER_1","OFFER_2"]
```

Cuando la extracción es un solo valor, la lista del **subscriberProperties** solo será de un elemento.

**LABEL:** Solo aplica para cuando **WAY:** >>, => , ^^ . Identifica la etiqueta o palabra clave en la plantilla XSLT.

**DESCRIPTION:** Descripción de la etiqueta.

**STATUS:** Estado A de Activo, I de Inactivo.

## 6.- QUICKWIN\_MDS.INVOKE\_RESPONSE\_EVALUATION

Esta tabla sirve para evaluar una respuesta del response o **Response Header**. La finalidad es buscar errores o validaciones en el response que podamos reportar al usuario. En caso de no configurarse las respuesta serán siempre exitosas. En esta tabla se pueden configurar varias evaluaciones en un orden específico determinado por "**ORDER\_BY**"

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
INVOKE_RESPONSE_EVALUATION_ID	NUMBER	N		
INVOKE_ID	NUMBER	N		
ORDER_BY	NUMBER	N	10	
XPATH	VARCHAR2(255)	Y		
EVALUATION_TYPE	VARCHAR2(3)	N		
EXPECTED_VALUE	VARCHAR2(2000)	Y		
VALUE_TYPE	VARCHAR2(255)	Y		
EVALUATION_OPERATOR	VARCHAR2(3)	N		
SUCCESS	VARCHAR2(3)	N	'YES'	
FAILURE_MESSAGE	VARCHAR2(255)	Y	'La respuesta no fue satisfactoria'	MENSAJE CUANDO ES FALLIDA LA EVALUACION DEL INVOKE, PARA TOMAR DESDE UN SUBSCRIBER PROPERTY SE DEBE COLOCAR ENTRE LLAVES [MI_MENSAJE]
STATUS	VARCHAR2(1)	N	'A'	
SUCCESS_MESSAGE	VARCHAR2(255)	Y	'Invocacion ejecutada con exito'	MENSAJE CUANDO ES EXITOSA LA EVALUACION DEL INVOKE, PARA TOMAR DESDE UN SUBSCRIBER PROPERTY SE DEBE COLOCAR ENTRE LLAVES [MI_MENSAJE]
FAIL_CODE	VARCHAR2(255)	Y	'-221'	CODIGO CUANDO FALLA LA EVALUACION DEL INVOKE , PARA TOMAR DESDE UN SUBSCRIBER PROPERTY SE DEBE COLOCAR ENTRE LLAVES [MI_CODIGO]
SUCCESS_CODE	VARCHAR2(255)	Y	'0'	CODIGO CUANDO ES EXITOSA LA EVALUACION DEL INVOKE, PARA TOMAR DESDE UN SUBSCRIBER PROPERTY SE DEBE COLOCAR ENTRE LLAVES [MI_CODIGO]

**INVOKE\_RESPONSE\_EVALUATION\_ID:** Id numerico.

**INVOKE\_ID:** Id de Invoke.

**ORDER\_BY:** Orden de evaluación.

**XPATH:** JSONPATH o XPATH para extraer valores del Response Header o Response.

**EVALUATION\_TYPE:** HV (Response Header Value), FV (Fixed Value), ND (Not Defined). **HV O FV** indica que se va a evaluar contra un VALOR ESPERADO. (**EXPECTED\_VALUE**).

HV indica que el valor viene en el **responseHeader**, mientras que FV toma el valor del response.

**EXPECTED\_VALUE. Recordar que cuando es "HV" el XPATH debe ser un JSONPATH.** ND es cuando el nodo a buscar no se encuentra definido en el response. Para este caso **EXPECTED\_VALUE** es NULL.

**EXPECTED\_VALUE:** Valor esperado a comparar. Aplica para HV O FV, para ND debe ser NULL.

**VALUE\_TYPE:** NUMERIC y VARCHAR. El valor de **EXPECTED\_VALUE** debe ser comparado como numero o texto.

**EVALUATION\_OPERATOR:** ==, igual != diferente, etc. Se compara el valor extraído de **XPATH** basado en el operador seleccionado.

**SUCCESS:** YES o NO. Indica que lo que estoy evaluando se lo debe considerar como ÉXITO o NO.

**STATUS:** Estado A de Activo, I de Inactivo.

## 7.- QUICKWIN\_MDS.INVOKE\_HISTORY

Tabla de logs de **Invokes**. Actualmente deshabilitada en producción.

En esta tabla se guarda información de ejecución de Invokes.

NAME	TYPE	NULLABLE	DEFAULT
INVOKE_HISTORY_ID	VARCHAR2(2000)	N	
STARTING_DATE	DATE	Y	
ENDING_DATE	DATE	Y	
ELAPSED_TIME	NUMBER	Y	
EXTERNAL_TRANSACCION_ID	VARCHAR2(2000)	Y	
CUSTOMER_INVOKER_ID	VARCHAR2(2000)	Y	
INVOKER_NAME	VARCHAR2(255)	Y	
TICKET_ID	VARCHAR2(255)	Y	
INVOKE_ID	NUMBER	Y	
CODE	NUMBER	Y	
MESSAGE	CLOB	Y	
REQUEST	CLOB	Y	
RESPONSE	CLOB	Y	
RETRIES_COUNT	NUMBER	Y	



## 7.- QUICKWIN\_MDS.INVOKE\_REPROCESSING

Tabla que sirve para reprocesos manuales cuando un invocador a pesar de tener varios reintentos falla.

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
INVOKE_REPROCESSING_ID	VARCHAR2(2000)	N		
STARTING_DATE	DATE	Y		Fecha de inicio
ENDING_DATE	DATE	Y		Fecha de Finalizacion
ELAPSED_TIME	NUMBER	Y		Tiempo transcurrido
EXTERNAL_TRANSACCION_ID	VARCHAR2(2000)	Y		
CUSTOMER_INVOKER_ID	VARCHAR2(2000)	Y		Operation history ID
INVOKER_NAME	VARCHAR2(255)	Y		Nombre del invocador
INVOKE_ID	NUMBER	Y		Referencia a la tabla invoke
CODE	NUMBER	Y		Codigo de error
MESSAGE	CLOB	Y		Mensaje de error
REQUEST	CLOB	Y		Cuerpo con que se realizo la solicitud
RESPONSE	CLOB	Y		Respuesta que se obtuvo de la invocacion
IS_REPROCESSED	VARCHAR2(3)	N		SI/NO indica si ya fue reprocesado
RETRIES_COUNT	NUMBER	Y		Numero de reintentos que se han realizado antes de que se envíe a esta tabla
ENDPOINT_URL	VARCHAR2(512)	Y		URL del Web Service
HEADERS	VARCHAR2(1024)	Y		Cabeceras http con que se realizo la peticion
HTTP_METHOD	VARCHAR2(10)	Y		Metodo de la peticion
INVOKE_DATA_REQUEST	CLOB	Y		Mensaje utilizado para reprocesar la invocacion

**Nota:** Revisar documentación de reproceso.

POST <http://192.168.37.146:8101/quickWin/reprocessProductProvisioning> HTTP/1.1

```
# BY DATES
{
  "reprocessBy": "date",
  "reprocessSearchInfo": {
    "startDate": "2029-11-05T15:33:43",
    "endDate": "2029-11-06T17:30:00"
  }
}

## BY ID
{
  "reprocessBy": "transactionId",
  "reprocessSearchInfo": {
    "ids": [
      "218632b6-3223-4899-9e61-cb6e4c247609"
    ]
  }
}

# ALL
{
  "reprocessBy": "all",
  "reprocessSearchInfo": {
    "ids": [
      "47441ad2-90cc-46d0-9567-c32e494b00fc",
      "3bd11309-ccab-4287-9510-de1cab0e3075",
      "9f7a3c99-4a95-4779-a333-41285adcd361"
    ],
    "startDate": "2019-10-31T16:00:17",
    "endDate": "2019-10-31T16:00:19"
  }
}
```

```
{
  "code": "1",
  "message": "No se encontraron invocaciones para reprocesar",
  "status": "WARNING",
  "reprocessLotId": "430fa576-b008-428f-91a6-224496a295c1",
  "transactionsReprocessed": 0
}
```

## REQUEST PATTERNS XSLT

### JSON.

**//params** es el nombre del nodo raíz de la lista de parámetros creada por **QuickWin**, por lo tanto es obligatorio.

Los valores del **"select="VALOR"** corresponder al **LABEL** de **INVOKE\_MAPPINGS**.

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```

<xsl:output method="text" indent="yes" media-type="text/json" omit-xml-
declaration="yes"/>
<xsl:template match="/">
{
  <xsl:for-each select="//params">
    "idRequest": "ActivateSupplementaryOffer",
    "params": {
      "X-APIVersion": "1.0",
      "X-TransactionId": "<xsl:value-of select="transactionId" />",
      "subscriptionId": "9<xsl:value-of
select="substring(_subscriptionId/text(), string-length(_subscriptionId/text())
- 7)" />",
      "offerId" : "<xsl:value-of select="_offerId" />"
    }
  </xsl:for-each>
}
</xsl:template>
</xsl:stylesheet>

```

## XML

**//params** es el nombre del nodo raíz de la lista de parámetros creada por **QuickWin**, por lo tanto es obligatorio.

Los valores del **"select="VALOR"** corresponder al **LABEL** de **INVOKE\_MAPPINGS**.

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:template match="/">
    <datos>
      <datosSubscriptor>
        <xsl:for-each select="//params">
          <subscriberId>
            <xsl:value-of select="_key"/>
          </subscriberId>
        </xsl:for-each>
      </datosSubscriptor>
    </datos>
  </xsl:template>
</xsl:stylesheet>

```

## DYNAMIC URL

```

[
  {
    "parameterName": "domainInformation",
    "type": "P",
    "searchValue": "_domainInformation"
  },
  {
    "parameterName": "serviceInformation",
    "type": "P",

```

```

    "searchValue": "_serviceInformation"
  },

  {
    "parameterName": "key",
    "type": "P",
    "searchValue": "_key"
  }
]

```

**parameterName:** Nombre del Parámetro con el que se arma la URL.

**searchValue:** Representa al **LABEL** en INVOKE\_MAPPING. Punto de Entrada para recibir los valores.

**type:** Representa al tipo de parámetro de URL. P parameter, Q query.

## CÓDIGO DE EXITO Y MENSAJE DE ERROR

### ESTATICOS

Por defecto los códigos de exito **SUCCESS\_CODE**, código de error **FAIL\_CODE**, mensaje de éxito **SUCCESS\_MESSAGE** y mensaje de error **FAILURE\_MESSAGE** han sido estático como vemos en el siguiente ejemplo.

### Ejemplo de configuracion

COLUMNAS	VALOR
INVOKE_RESPONSE_EVALUATION_ID	90
INVOKE_ID	126
ORDER_BY	10
XPATH	//*[local-name()='WS_ERROR']/text()
EVALUATION_TYPE	FV
EXPECTED_VALUE	false
VALUE_TYPE	VARCHAR
EVALUATION_OPERATOR	!=
SUCCESS	NO
FAILURE_MESSAGE	Error al consumir el servicio
STATUS	A
SUCCESS_MESSAGE	Invocacion ejecutada con exito
FAIL_CODE	14
SUCCESS_CODE	0

# DINÁMICOS

---

Existe dos formas de hacer que **FAIL\_CODE,FAILURE\_MESSAGE,SUCCESS\_CODE y SUCCESS\_MESSAGE** sean dinámicos.

1. Haciendo uso de un Subscriber Properties. En vez de un valor estático podemos poner un Subscriber Properties entre corchetes: [MY\_SUBSCRIBER\_PROPERTIES]. Recuerde que solo se toma el primer valor de la lista.

1. Asumiendo que MY\_SUBSCRIBER\_PROPERTIES = ["HOLA MUNDO","HELLO WORLD"], entonces la salida seria **"HOLA MUNDO"**.

2. Haciendo uso de una funcion en linea denominada Interpolacion de mensaje.

1. Esta funcion necesita definir un caracter separador de elemento de una lista despues de **'SEP='**. Si elegimos usar ';', entonces empieza de esta forma: SEP=;

2. Recibe la lista delimitada por el separador.

1. SEP=;SUBSCRIBER\_PROPERTIES1;SUBSCRIBER\_PROPERTIES2;

2. Si no encuentra algún subscriber properties definido en la lista este es un valor estático.

3. El ultimo elemento es el formato del String que se desea basado en FORMAT del String en Java usando '%'

1. SEP=;SUBSCRIBER\_PROPERTIES1;SUBSCRIBER\_PROPERTIES2;**El valor de % no es igual a %**.

1. En este caso toma el primer elemento del Subscriber y lo presenta como mensaje. Asumiendo que SUBSCRIBER\_PROPERTIES1 = ["100","500"] y

SUBSCRIBER\_PROPERTIES2 = ["200","3000"], entonces seria **El valor de 100 no es igual a 200**.

2. Si el SUBSCRIBER\_PROPERTIES2 no existiese. **El valor de 100 no es igual a SUBSCRIBER\_PROPERTIES2**.

3. E incluso SEP=;[SUBSCRIBER\_PROPERTIES1];%

1. Si [SUBSCRIBER\_PROPERTIES1] no existe entonces tendríamos [SUBSCRIBER\_PROPERTIES1].

2. Pero si quisieramos lograr lo mismo con otro subscriber, pudieramos hacer lo siguiente. Asumiendo que SUBSCRIBER\_PROPERTIES2 = ["SUBSCRIBER\_PROPERTIES1"] entonces SEP=;SUBSCRIBER\_PROPERTIES2;[%] es igual a ["SUBSCRIBER\_PROPERTIES1"]

**Nota:** Si una expresión de interpolacion de mensaje da como resultado un subscriber properties entre corchete, finalmente se presentará el mensaje de ese Subscriber Properties.

## Ejemplos de configuracion.

En este caso, "|" es el separador de la lista.

En este ejemplo se espera que CODIGO\_ID contenga el codigo de exito o error.

En SUCCESS\_MESSAGE usamos dos Subscriber Properties y el separador de mensaje es "#"

SQL Output Statistics					
SELECT t.*,rowid FROM condition_properties t WHERE T.Condition_Id = 269;					
	STATUS	SUCCESS_MESSAGE	FAIL_CODE	SUCCESS_CODE	ROWID
1	A	SEP=#PRUEBA_GRUPO_COND1#TMP1#La condicion PRUEB	SEP={CODIGO_ID}100%\$	SEP={CODIGO_ID}-100%\$	AAC/VQACSA

## XPATH DINÁMICO CON INTERPOLACION DE MENSAJE

Al igual que con FAIL\_CODE,FAILURE\_MESSAGE,SUCCESS\_CODE y SUCCESS\_MESSAGE pueden configurarse de forma dinamica, lo mismo aplica para el XPATH de INVOKE\_MAPPINGS y INVOKE\_RESPONSE\_EVALUATION

## XPATH INVOKE MAPPINGS

Usar interpolacion en XPATH es muy util puesto que nos puede ayudar a crear XPATH dinamico aplicando algun filtro.

En este ejemplo vemos que el Subscriber Properties CRM debe conter la cadena que sera usasa en el XPATH.

SQL Output Statistics					
select * from INVOKE_MAPPING t where xpath like '%SEP%'					
	SUBSCRIBE	FUNCTION_ID	FIXED_VALUE	WAY	XPATH
1	NAMES				SEP=#CRM#

## XPATH INVOKE RESPONSE EVALUATION

Usar interpolacion en XPATH en la evaluacion puede ser muy util cuando querememos hacer alguna evaluacion dinamica de algun tipo de error.

En este ejemplo se espera que XPATH contenga el XPATH válido.

XPATH = ["//\*[local-name()='perrorOut']/text()"]

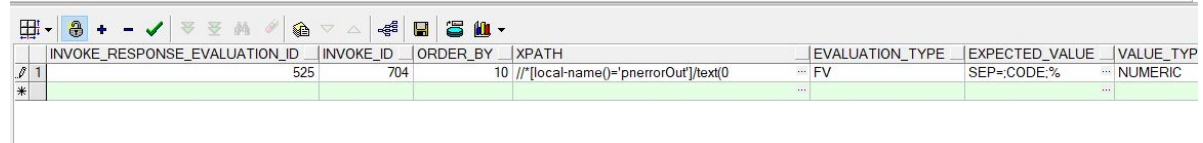
SQL Output Statistics					
select t.*,rowid from invoke_response_evaluation t where invoke_id = 704					
-- XPATH = ["//*[local-name()='perrorOut']/text()"]					
	INVOKE_RESPONSE_EVALUATION_ID	INVOKE_ID	ORDER_BY	XPATH	EVALUATION_TYPE
1	525	704	10	SEP=XPATH%	FV

## EVALUATION TYPE DYNAMIC

Tanto para la opcion FV y HV esta disponible interpolacion de mensaje en **EXPECTED\_VALUE**

En este ejemplo se espera que el Subscriber Properties CODE contenga el código numerico a comparar.

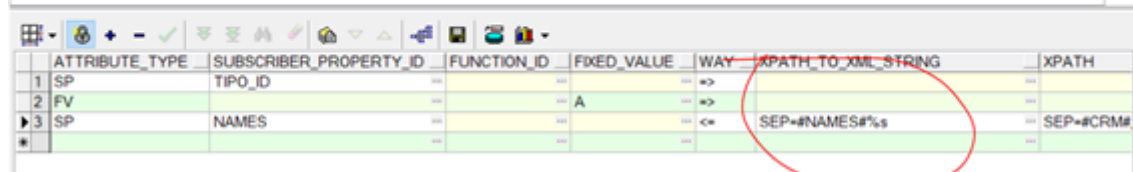
Esto es util en caso que deseemos evaluar de forma dinamica un resultado. Cabe recalcar que en esta tabla de configuraciones podemos configurar mas de una evaluación definido en un orden especifico por ORDER\_BY. La evaluacion de errores termina cuando encuentra el primero de la lista, y finalmente mostrará el codigo de falla y mensaje de falla configurado.

SQL	Output	Statistics
<pre>select t.*,rowid from invoke_response_evaluation t where invoke_id = 704 -- CODE = ["0"]</pre>		
		

## XPATH TO XML STRING

Usar interpolacion en XPATH TO XML string es util cuando existen diferentes XML como estring de los cuales queremos aplicar un XPATH.

En este ejemplo se espera que NAMES contenga una cadena XPATH.

SQL	Output	Statistics
<pre>SELECT t.*,rowid FROM INVOKE_MAPPING T WHERE T.INVOKE_ID = 708;</pre>		
		

## CREAR INVOKES PLSQL

### PLSQL CREATE INVOKES

```
-- Created on 4/2/2020 by MGARCIAR
declare
  -- Local variables here
  LN_INVOKE_ID1                NUMBER := 0;
  LN_EXT_RES_GROUP_ID          NUMBER := 0;
  LN_EXT_RES_ID                NUMBER := 0;
begin
  select MIN(S.EXTERNAL_RESOURCE_GROUP_ID) EXTERNAL_RESOURCE_GROUP_ID INTO
  LN_EXT_RES_GROUP_ID from EXTERNAL_RESOURCE_GROUPS s
  where UPPER(s.description) like UPPER('%gateway%');

  --RESOURCE
```

```

SELECT NVL(MAX(E.EXTERNAL_RESOURCE_ID),0) + 1 INTO LN_EXT_RES_ID FROM
EXTERNAL_RESOURCES E;

-- INVOKE
SELECT NVL(MAX(I.INVOKE_ID),0) + 1 INTO LN_INVOKE_ID1 FROM INVOKE I;

dbms_output.put_line('LN_EXT_RES_GROUP ID: ' || LN_EXT_RES_GROUP_ID );
dbms_output.put_line('LN_EXT_RES ID: ' || LN_EXT_RES_ID );

--CREAR SUBSCRIBER PROPERTIES

/*insert into SUBSCRIBER_PROPERTIES (SUBSCRIBER_PROPERTY_ID, DESCRIPTION,
STATUS)
values ('OFFER_ID_HW', 'Oferta Huawei', 'A'); */
--EXTERNAL RESOURCES

insert into EXTERNAL_RESOURCES (EXTERNAL_RESOURCE_ID,
EXTERNAL_RESOURCE_GROUP_ID, OPERATION_NAME_DESCRIPTION, RESOURCE_DESCRIPTION,
REQUEST_PATTERN_XSLT, TIMEOUT, STATUS, EXT_RES_TYPE, REQUEST_METHOD,
CONTENT_TYPE, DATA_TYPE)
values (LN_EXT_RES_ID, LN_EXT_RES_GROUP_ID, 'Consulta Usuario OTT AMCO',
'Consulta Usuario OTT AMCO', '<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" indent="yes" media-type="text/json" omit-xml-
declaration="yes"/>
<xsl:template match="/">
{
<xsl:for-each select="//params">
"idRequest": "<xsl:value-of select="_idRequest" />",
"params":{
"invokeMethod": "<xsl:value-of select="_invokeMethod" />",
"correlatorId": "<xsl:value-of select="_correlatorId" />",
"employeeId": "<xsl:value-of select="_employeeId" />",
"origin": "<xsl:value-of select="_origin" />",
"startDate": "<xsl:value-of select="_startDate" />",
"endDate": "<xsl:value-of select="_endDate" />",
"iccidManager": "<xsl:value-of select="_iccidManager" />",
"providerId": "<xsl:value-of select="_providerId" />",
"customerId": "<xsl:value-of select="_customerId" />"
}
</xsl:for-each>
}
</xsl:template>
</xsl:stylesheet>', 10000, 'A', 'REST', 'POST', 'application/json', 'json');

insert into EXTERNAL_RESOURCE_COMPONENTS
(EXTERNAL_RESOURCE_COMPONENT_ID,
EXTERNAL_RESOURCE_ID,
ENDPOINT_URL,
SHORT_DESCRIPTION,
ROUTING_TYPE,
ORDER_BY,
WEIGHT)

```



```

values
((SELECT NVL(MAX(EC.EXTERNAL_RESOURCE_COMPONENT_ID),0) + 1
  FROM EXTERNAL_RESOURCE_COMPONENTS EC),
LN_EXT_RES_ID,
'http://192.168.37.146:8082/rest/microGateway/invoke',
'Consulta Usuario OTT AMCO',
'B',
10,
100);

insert into INVOKE
(INVOKE_ID,
EXTERNAL_RESOURCE_ID,
OPERATION_NAME_DESCRIPTION,
DESCRIPTION,
STATUS,
RETRIES,
TIME_BETWEEN_RETRIES)
values
(LN_INVOKE_ID1,
LN_EXT_RES_ID,
'Consulta Usuario OTT AMCO',
'Consulta Usuario OTT AMCO',
'A',
0,
1000);

--MAPPING

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'FV', '', '', 'claro-video:queryUserOTT', '=>', '', '',
'_idRequest', 'Id Requerimiento', 'A');

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'FV', '', '', 'consultardatoscliente', '=>', '', '',
'_invokeMethod', 'Metodo a invocar', 'A');

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'SP', '$INTERNALTRANSACTIONID$', '', '', '=>', '', '',
'_correlatorId', 'ID de Correlacion', 'A');

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'SP', '$TRANSACTIONDATE_WS$', '', '', '=>', '', '', '_startDate',
'Fecha de Inicio', 'A');

```

```

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'SP', '$TRANSACTIONDATE_WS$', '', '', '=>', '', '', '_endDate',
'Fecha Final', 'A');

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'FV', '', '', '86', '=>', '', '', '_employeeId', 'Id Empleado',
'A');

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'FV', '', '', 'AUTOSERVICIO', '=>', '', '', '_origin', 'Origen',
'A');

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'FV', '', '', 'AMCOEC', '=>', '', '', '_iccidManager', 'ICCID
MANAGER', 'A');

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'FV', '', '', 'PA00002811', '=>', '', '', '_providerId', 'Id
Provider', 'A');

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'SP', 'CUSTOMER_ID', '', '', '=>', '', '', '_customerId',
'CustomerId Or ProfileId claroId', 'A');

insert into INVOKE_MAPPING (INVOKE_MAPPING_ID, INVOKE_ID, ATTRIBUTE_TYPE,
SUBSCRIBER_PROPERTY_ID, FUNCTION_ID, FIXED_VALUE, WAY, XPATH_TO_XML_STRING,
XPATH, LABEL, DESCRIPTION, STATUS)
values ((SELECT NVL(MAX(im.invoke_mapping_id),0)+1 from INVOKE_MAPPING im),
LN_INVOKE_ID1, 'SP', 'EMAIL', '', '', '<=', '',
'$.queryUserOttResponse.userData.item[?(@.key=='email')].value', '', 'EMAIL',
'A');

--response evalutaion

```

```

insert into INVOKE_RESPONSE_EVALUATION (INVOKE_RESPONSE_EVALUATION_ID,
INVOKE_ID, ORDER_BY, XPATH, EVALUATION_TYPE, EXPECTED_VALUE, VALUE_TYPE,
EVALUATION_OPERATOR, SUCCESS, FAILURE_MESSAGE, STATUS)
values ((SELECT NVL(MAX(IR.INVOKE_RESPONSE_EVALUATION_ID),0) + 1 FROM
INVOKE_RESPONSE_EVALUATION IR), LN_INVOKE_ID1, 10,
'$.queryUserOttResponse.resultCode', 'FV', '0', 'NUMERIC', '!=', 'NO', 'Error
Consumir AMCO', 'A');

dbms_output.put_line('INVOKE ID: ' || LN_INVOKE_ID1 );

dbms_output.put_line('-----REFRESH-----
----');

dbms_output.put_line('http://192.168.37.146:8101/quickwin/refreshInvokeById/' || L
N_INVOKE_ID1);
dbms_output.put_line('-----GUARDE ESTE SCRIPT POR MAYOR SEGURIDAD-----
-----');

dbms_output.put_line('-----SELECT-----
---');
dbms_output.put_line('SELECT * FROM INVOKE T WHERE T.INVOKE_ID =
'''||LN_INVOKE_ID1||''';');
dbms_output.put_line('SELECT * FROM EXTERNAL_RESOURCE_GROUPS T WHERE
T.EXTERNAL_RESOURCE_GROUP_ID = '''||LN_EXT_RES_GROUP_ID||''';');
dbms_output.put_line('SELECT * FROM EXTERNAL_RESOURCES T WHERE
T.EXTERNAL_RESOURCE_ID = '''||LN_EXT_RES_ID||''';');
dbms_output.put_line('SELECT * FROM EXTERNAL_RESOURCE_COMPONENTS T WHERE
T.EXTERNAL_RESOURCE_ID = '''||LN_EXT_RES_ID||''';');
dbms_output.put_line('SELECT * FROM INVOKE_MAPPING T WHERE T.INVOKE_ID =
'''||LN_INVOKE_ID1||''';');
dbms_output.put_line('SELECT * FROM INVOKE_RESPONSE_EVALUATION T WHERE
T.INVOKE_ID = '''||LN_INVOKE_ID1||''';');

dbms_output.put_line('-----UPDATE STATUS-----
-----');
dbms_output.put_line('update INVOKE T set status = 'I' WHERE T.INVOKE_ID =
'''||LN_INVOKE_ID1||''';');
dbms_output.put_line('update EXTERNAL_RESOURCE_GROUPS T set status = 'I'
WHERE T.EXTERNAL_RESOURCE_GROUP_ID = '''||LN_EXT_RES_GROUP_ID||''';');
dbms_output.put_line('update EXTERNAL_RESOURCES T set status = 'I' WHERE
T.EXTERNAL_RESOURCE_ID = '''||LN_EXT_RES_ID||''';');
dbms_output.put_line('update EXTERNAL_RESOURCE_COMPONENTS T set status = 'I'
WHERE T.EXTERNAL_RESOURCE_ID = '''||LN_EXT_RES_ID||''';');
dbms_output.put_line('update INVOKE_MAPPING T set status = 'I' WHERE
T.INVOKE_ID = '''||LN_INVOKE_ID1||''';');
dbms_output.put_line('update INVOKE_RESPONSE_EVALUATION T set status = 'I'
WHERE T.INVOKE_ID = '''||LN_INVOKE_ID1||''';');

dbms_output.put_line('-----DELETE-----
--');
dbms_output.put_line('delete INVOKE_RESPONSE_EVALUATION T WHERE T.INVOKE_ID =
'''||LN_INVOKE_ID1||''';');

```

```

dbms_output.put_line('delete INVOKE_MAPPING T WHERE T.INVOKE_ID =
'''||LN_INVOKE_ID1||''';');
dbms_output.put_line('delete EXTERNAL_RESOURCE_COMPONENTS T WHERE
T.EXTERNAL_RESOURCE_ID = '''||LN_EXT_RES_ID||''';');
dbms_output.put_line('delete EXTERNAL_RESOURCES T WHERE
T.EXTERNAL_RESOURCE_ID = '''||LN_EXT_RES_ID||''';');
dbms_output.put_line('delete EXTERNAL_RESOURCE_GROUPS T WHERE
T.EXTERNAL_RESOURCE_GROUP_ID = '''||LN_EXT_RES_GROUP_ID||''';');
dbms_output.put_line('delete INVOKE T WHERE T.INVOKE_ID =
'''||LN_INVOKE_ID1||''';');
dbms_output.put_line('-----REQUEST-----
-----');

dbms_output.put_line('http://192.168.37.146:8101/quickwin/executeInvoke');

dbms_output.put_line('-----POST REST-----
-----');

dbms_output.put_line('{
    "invokeId": '''||LN_INVOKE_ID1||'',
    "invokerName": "Prueba Invoke",
    "cacheOptions": "0",
    "sync": "YES",
    "customerInvokerId": "id12345",
    "sessionData": {
        "externalSubscriberProperties": [''];

FOR subs IN (SELECT T.SUBSCRIBER_PROPERTY_ID FROM INVOKE_MAPPING T WHERE
T.INVOKE_ID = LN_INVOKE_ID1 AND T.WAY != '<=' AND T.SUBSCRIBER_PROPERTY_ID IS
NOT NULL) LOOP

    dbms_output.put_line('                                {"id":
'''||subs.subscriber_property_id||','"value": [""]}',');

END LOOP;
    dbms_output.put_line('                                {"id":
'''||'DUMMY'||','"value": [""]}');

dbms_output.put_line('                                ]
                                }
                                }');

end;

```

# BORRAR INVOKES PLSQL

## PLSQL DELETE INVOKES

Advertencia. El siguiente ejemplo, borras todos los invocadores de un grupo específico.

**Antes de borrar los invocadores, estos no deben tener referencias.**

```

declare
-- DELETE INVOKES, DEBERIA SER POR ID DE GRUPO O NOMBRE UNICO
-- BORRA TODOS LOS INVOKES ASOCIADO AL GRUPO
lv_group_description varchar(255) := 'IPTV-CLARO_VIDEO';
BEGIN
DELETE FROM INVOKE_MAPPING IM
WHERE IM.INVOKE_ID IN
(SELECT DISTINCT I.INVOKE_ID
FROM INVOKE I, EXTERNAL_RESOURCES R, EXTERNAL_RESOURCE_GROUPS G
WHERE I.EXTERNAL_RESOURCE_ID = R.EXTERNAL_RESOURCE_ID
AND R.EXTERNAL_RESOURCE_GROUP_ID = G.EXTERNAL_RESOURCE_GROUP_ID
AND G.DESRIPTION = lv_group_description);

DELETE FROM INVOKE_RESPONSE_EVALUATION IR
WHERE IR.INVOKE_ID IN
(SELECT DISTINCT I.INVOKE_ID
FROM INVOKE I, EXTERNAL_RESOURCES R, EXTERNAL_RESOURCE_GROUPS G
WHERE I.EXTERNAL_RESOURCE_ID = R.EXTERNAL_RESOURCE_ID
AND R.EXTERNAL_RESOURCE_GROUP_ID = G.EXTERNAL_RESOURCE_GROUP_ID
AND G.DESRIPTION = lv_group_description);

DELETE FROM INVOKE I
WHERE I.INVOKE_ID IN
(SELECT DISTINCT I.INVOKE_ID
FROM INVOKE I, EXTERNAL_RESOURCES R, EXTERNAL_RESOURCE_GROUPS G
WHERE I.EXTERNAL_RESOURCE_ID = R.EXTERNAL_RESOURCE_ID
AND R.EXTERNAL_RESOURCE_GROUP_ID = G.EXTERNAL_RESOURCE_GROUP_ID
AND G.DESRIPTION = lv_group_description);

DELETE FROM EXTERNAL_RESOURCE_COMPONENTS RC
WHERE RC.EXTERNAL_RESOURCE_ID IN
(SELECT DISTINCT R.EXTERNAL_RESOURCE_ID
FROM EXTERNAL_RESOURCES R, EXTERNAL_RESOURCE_GROUPS G
WHERE R.EXTERNAL_RESOURCE_GROUP_ID = G.EXTERNAL_RESOURCE_GROUP_ID
AND G.DESRIPTION = lv_group_description);

DELETE FROM EXTERNAL_RESOURCES RC
WHERE RC.EXTERNAL_RESOURCE_ID IN
(SELECT DISTINCT R.EXTERNAL_RESOURCE_ID
FROM EXTERNAL_RESOURCES R, EXTERNAL_RESOURCE_GROUPS G
WHERE R.EXTERNAL_RESOURCE_GROUP_ID = G.EXTERNAL_RESOURCE_GROUP_ID
AND G.DESRIPTION = lv_group_description);

DELETE FROM EXTERNAL_RESOURCE_GROUPS G WHERE G.DESRIPTION =
lv_group_description;

/*
DELETE FROM SUBSCRIBER_PROPERTIES T WHERE T.SUBSCRIBER_PROPERTY_ID =
'FULL_NAME';
*/
--chequear los errores
exception when others then
rollback;
dbms_output.put_line(sqlerrm);
end;

```

# Support

---

QukickWin is a Claro's project. It can grow thanks to feedback and support of other actors inside the company

## Stay in touch

---

- Leader Architec - TIC Manuel García
- Mail - [mgarcia@claro.com.ec](mailto:mgarcia@claro.com.ec)
- Sponsor - TIC Guillermo Proaño
- Mail - [gproano@claro.com.ec](mailto:gproano@claro.com.ec)
- Developers - Fernando Andrade
- Mail - [fernando.andrade@gizlocorp.com](mailto:fernando.andrade@gizlocorp.com)