

Mocksy server rule set for Adaptive Payments

**Overview**

The primary abstractions in Mocksy are a request and response. A request is received and it is mapped to a response that is returned back to the client. In this way, it acts exactly like a normal service except without all the business logic, data access, etc.

This mapping is accomplished with a set of rules, which can be grouped into rule sets. Only one rule is triggered for each request, and the corresponding response is returned to the client. The first rule that matches the request with the matcher will be triggered.

A matcher could match a request either by the URI, query parameters, or with a match on the body, either with a regular expression or xpath, if the data happens to be in XML.

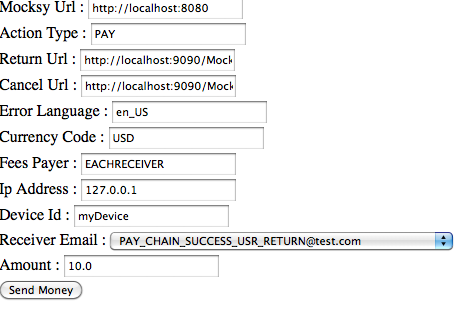
**Installing the Sample rule set**

1. Simply unzip the Mocksy\_AP\_Rules.zip file somewhere on your local drive
2. Start the mocksy server on default port
   1. e.g. run 'java -jar mocksy-server-0.8-full.jar -r rules.xml'
3. To start it on other port start it with –p option
   1. e.g. run 'java -jar mocksy-server-0.8-full.jar -r rules.xml -p 9080'

**Installing a sample web application**

1. Download tomcat
2. Unzip the MockSySample.zip in the webapps folder of the tomcat.
3. Start tomcat
4. Afetr starting the tomcat launch your app at <http://localhost:9090/MocksySample/sm.jsp>

The page should look something like this. Point the Mocksy Url to the server and port where you have started Mocksy



**Defining the Adaptive Payment Rule Set**

The best way to start defining the Mocksy rule set for adpative payments is to [download](https://cms.paypal.com/cms_content/US/en_US/files/developer/PP_AdaptivePayments.pdf) an API Guide and understand each API and various errors it throws based on the request received. Before you start defining the rule set you may have to capture the response files generated based on different request parameters. You can start capturing this by writing a small test curl program that generates a NVP based request and calls the APIs pointing to sandbox endpoint at <https://svcs.sandbox.paypal.com/AdaptivePayments/>. For complete rule set we need to capture all the API operations and its possible success and error scenarios. The goal after the development of this rule set is to have some one change the end point to their local Mocksy server and get the responses back like they will on the real sandbox. For complete list of API operations and possible error conditions for all the APIs please follow the downloaded document.

The basic scenario below for “Pay” API will show you how to form the different requests and capture the different responses. You can follow the same steps for developing rule sets for other API operations.

**Constructing a Pay Request**

Each request message includes HTTP headers specifying authentication, the application ID, the device ID or IP address, and the payload format or protocol. When server cannot validate these headers it will throw an error. The errors codes for headers are listed in the document and you have to construct a template error file and generate a response by passing in the different error ids and messages based on the request parameters received. The request below shows a sample Pay request. You can play around with the parameters e.g. acceptable values for actionType are “PAY”,”CREATE” and “PAY\_PRIMARY”. You can change the actionType to an unacceptable value to get the error response back.

Example Pay Request

curl -s --insecure

-H "X-PAYPAL-SECURITY-USERID: *api\_username*"

-H "X-PAYPAL-SECURITY-PASSWORD: *api\_password*"

-H "X-PAYPAL-SECURITY-SIGNATURE: *api\_signature*"

-H "X-PAYPAL-REQUEST-DATA-FORMAT: NV"

-H "X-PAYPAL-RESPONSE-DATA-FORMAT: NV"

-H "X-PAYPAL-APPLICATION-ID: *app\_id*"

https://svcs.sandbox.paypal.com/AdaptivePayments/Pay -d

"requestEnvelope.errorLanguage=en\_US

&actionType=PAY

&senderEmail=*sender@domain*

&receiverList.receiver(0).email=*receiver@domain*

&receiverList.receiver(0).amount=100.00

&currencyCode=USD

&feesPayer=EACHRECEIVER

&memo=Simple payment example.

&cancelUrl=http://*your\_cancel\_url*

&returnUrl=http://*your\_return\_url*

&ipnNotificationUrl=http://*your\_ipn\_notification\_url*"

Example Pay Success Response

responseEnvelope.timestamp=2009-07-13T12%3A34%3A29.316-07%3A00

&responseEnvelope.ack=Success

&responseEnvelope.correlationId=d615a365bed61

&responseEnvelope.build=DEV

&payKey=AP-3TY011106S4428730

&paymentExecStatus=**CREATED**

Example Pay Error Response

responseEnvelope.timestamp=2011-02-02T15%3A59%3A28.535-08%3A00&responseEnvelope.ack=Failure&responseEnvelope.correlationId=99b0ba1417854&responseEnvelope.build=1655692&error(0).errorId=${errorId}&error(0).domain=PLATFORM&error(0).severity=Error&error(0).category=Application&error(0).message=${message}&error(0).parameter(0)=${parameter}

Similarly you can take each API operation and create template files for success and error responses and then define the Mocksy rule sets to return the responses based on the request parameters you receive. Please follow the example rule defined below for defining the rule set.

**Example Rule Set – Chained Payments**

The packaged rule set is an example rule set for Pay API with Chained Payment. The directory structure when you unzip the rule set file is shown below.

1. $dir – contains the basic rules , the server itself and the base error template file. It also includes a keystore with a sample self signed certificate if you want to run Mocksy server in SSL mode.

The server starts with a base rules file “rules.xml”. This file contains the basic rules that are common to all the API calls e.g. headers related rules or data formatted rules. The rule below shows that request data format has to be either NV, JSON or SOAP.

<rule id="INVALID\_DATA\_FORMAT\_REQUEST" file="err\_template.nvp">

<not-match header="X-PAYPAL-REQUEST-DATA-FORMAT">NV|SOAP|JSON</not-match>

<filter class="org.mocksy.filter.PropertyFilter">

<errorId>500000</errorId>

<message>Internal Error</message>

</filter>

</rule>

The error template file contains a template for error response. Based on the rule you are validating the error code and error message needs to change. e.g. The mandatory user id rule makes sure that all request contains the header ”X-PAYPAL-SECURITY-USERID”. The rule shown below makes sure that if the security user id is not present, mocksy sends back an error response generated from error template by passing in errorId and message.

<rule id="AdaptiveAPI\_INVALID\_USERID" file="err\_template.nvp">

<not-match header="X-PAYPAL-SECURITY-USERID">.+</not-match>

<filter class="org.mocksy.filter.PropertyFilter">

<errorId>500000</errorId>

<message>Authentication failed.API credentials are incorrect.</message>

</filter>

</rule>

If all the validations pass through the Mocksy will try to validate the default ruleset, which is “rules\_adaptive.xml”

<default-rule ruleset="AdaptivePayments/rules\_adaptive.xml" />

1. $dir/AdaptivePayments – Contains the rule file related to the match required for APIs and related error template file. The below makes sure that if the request is for Pay API the server validates the request with Pay API rules. Similarly this file should contain rules related to other adaptive APIs as well.

<rule id="PayAPI" ruleset="/Pay/rules\_pay.xml">

<match>.\*/Pay.\*</match>

</rule>

1. $dir/AdaptivePayments/Pay – Contains the rule set for Pay API, Chained payment, success and error templates. “rules\_pay.xml” validates basic rules related to Pay API. e.g. The rule shown below checks for the valid action type is Pay request. If does not match one of the valid action type it sends back an error template with error id 540031.

<rule id="INVALID\_ACTION\_TYPE" file="pay\_err\_template.nvp">

<not-match param="actionType">PAY|CREATE|PAY\_PRIMARY</not-match>

<filter class="org.mocksy.filter.PropertyFilter">

<errorId>540031</errorId>

<message>The system did not recognize the action type</message>

</filter>

</rule>

Similarly there are certain error conditions, which are common to all Pay API calls, and those rules can be defined in this file. To test this rules for your application you have to check on the receiver id. e.g. The rule shown below validates for an Invalid country for a receiver account. To test your application for this rule you have to set the primary email address as “PAY\_INVALID\_ACCT\_RCV\_CNTRY\_EMAIL@test.com” When Mocksy server matches this rule it will send back an error code 539041 with error message.

<rule id="PAY\_INVALID\_ACCT\_RCV\_CNTRY\_EMAIL" file="pay\_err\_template.nvp">

<match param="receiverList.receiver(0).email">PAY\_INVALID\_ACCT\_RCV\_CNTRY\_EMAIL@test.com</match>

<filter class="org.mocksy.filter.PropertyFilter">

<errorId>539041</errorId>

<message>The email account is based in a country that is not enabled

to receive payments</message>

</filter>

</rule>

When the server is done checking all the rules for basic rules related to Pay API we could validate whether the request is a chained payment by checking for the existence of primary and secondary receiver. If it matches those criteria the “rules\_pay\_chained.xml” file can do validations related to chain payment.

<rule id="PAY\_CHAIN\_BASE\_RULES" ruleset="rules\_pay\_chained.xml">

<match param="receiverList.receiver(0).email">.+</match>

<match param="receiverList.receiver(0).primary">true</match>

<match param="receiverList.receiver(0).amount">.+</match>

<match param="receiverList.receiver(1).primary">false</match>

<match param="receiverList.receiver(1).amount">.+</match>

</rule>

1. $dir/AdaptivePayments/Pay/rules\_pay\_chained.xml

This rule file has the validation specific to chained payment. e.g. The rule shown below shows that if the request is a chained payment then it should also contain “feesPayer” parameter from one of the valid values. An invalid request parameter error will be thrown if it does not satisfy that criteria.

<rule id="INVALID\_FEES\_PAYER" file="pay\_err\_template.nvp">

<match param="receiverList.receiver(0).primary">true</match>

<not-match param="feesPayer">SENDER|PRIMARYRECEIVER|EACHRECEIVER||SECONDRYONLY</not-match>

<filter class="org.mocksy.filter.PropertyFilter">

<errorId>580022</errorId>

<message>Invalid request parameter</message>

<parameter>feesPayer</parameter>

</filter>

</rule>

1. $dir/WebScr – Contains the rules related to web redirect .

<rule id="webscr\_cmd" file="error.nvp">

<not-match param="cmd">\_ap-payment</not-match>

</rule>