XML Response Validation Scenarios

Let us see, how various types of XML responses can be validated in vREST.

1. My API returns the static response.
2. My API returns some dynamic properties like _id, createdOn etc. and I want to ignore them during response validation.
3. My API returns a very large response and I am interested in validating only a small part of my API response.
4. My API returns some response in which some part of the response can be obtained from the responses of previous test cases.

Scenario 1: My API returns the static response.

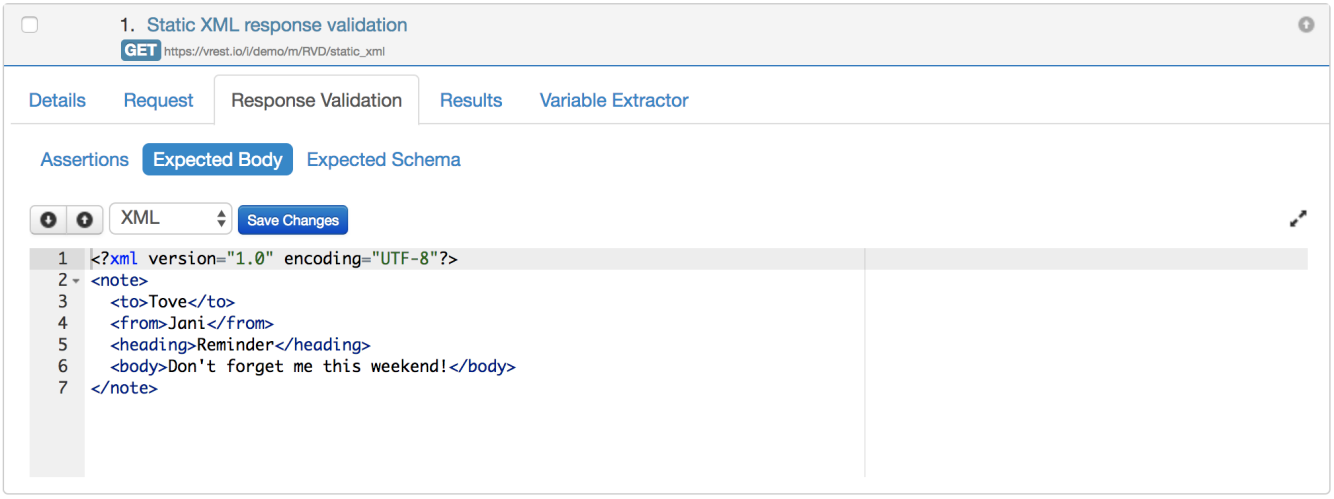
If your API returns the static response, then just set the Expected Body with the static response and Expected Status Code.

Let us suppose, your API returns the following static XML response,

Status Code: 200

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

For such scenarios, simply specify your static response body in Expected Body and set the Expected Status Code also as specified in the following image. Validator used for this scenario is "Default Validator".



Scenario 2: My API returns some dynamic properties like _id, createdOn etc. and I want to ignore them during response validation.

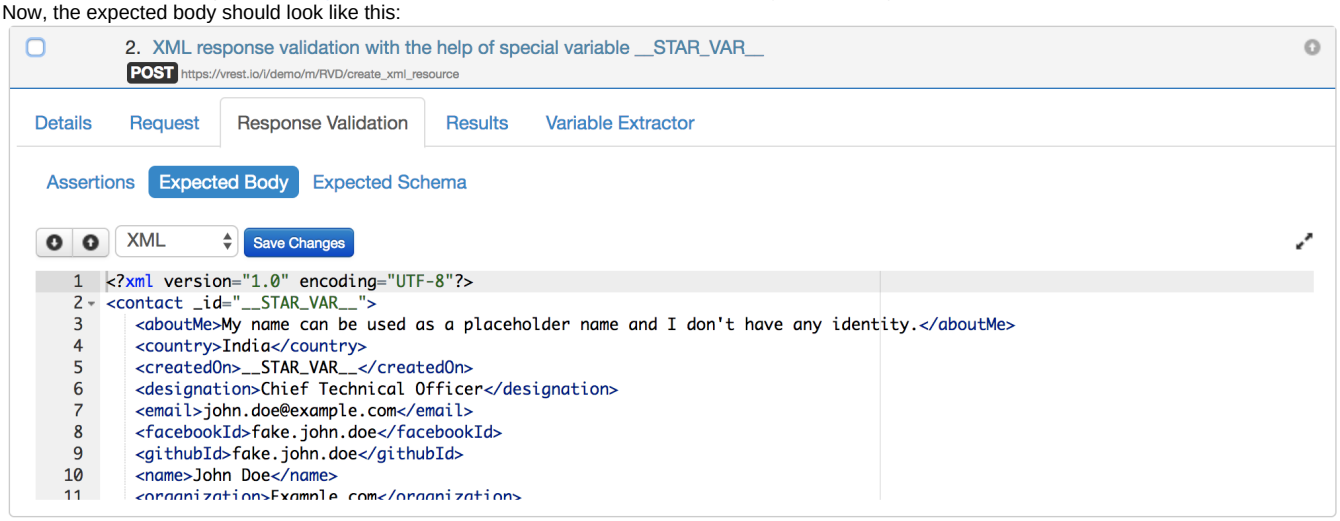
Suppose, If your API which creates a resource on the server and returns some dynamic properties like _id, createdOn etc. And you want to ignore these dynamic properties during the response validation. In such scenario, Default Validator can be used.

e.g. Let us suppose, the API returns the following response:

Status Code: 200

```
<?xml version="1.0" encoding="UTF-8"?>
<contact _id="536493015f56452a03000010">
   <aboutMe>My name can be used as a placeholder name and I don't have any identity.</aboutMe>
   <country>India</country>
   <createdOn>2014-05-03T06:56:01.134Z</createdOn>
   <designation>Chief Technical Officer</designation>
   <email>john.doe@example.com</email>
   <facebookId>fake.john.doe</facebookId>
   <githubId>fake.john.doe</githubId>
   <name>John Doe</name>
   <organization>Example.com</organization>
   <twitterId>fake.john.doe</twitterId>
</contact>
```

Here in the above response, you want to ignore the dynamically generated _id and createdOn field.
For such scenarios, simply use the special variable **__STAR_VAR__** for values, which you want to ignore.
Now, the expected body should look like this:



Scenario 3: My API returns a very large response and I am interested in validating only a small part of my API response.

If you want to validate only a small part of your API response and want to ignore rest of the properties then you can use special variable as xml tag **<__STAR_VAR__/>.**

Let us suppose, the API returns the following response:

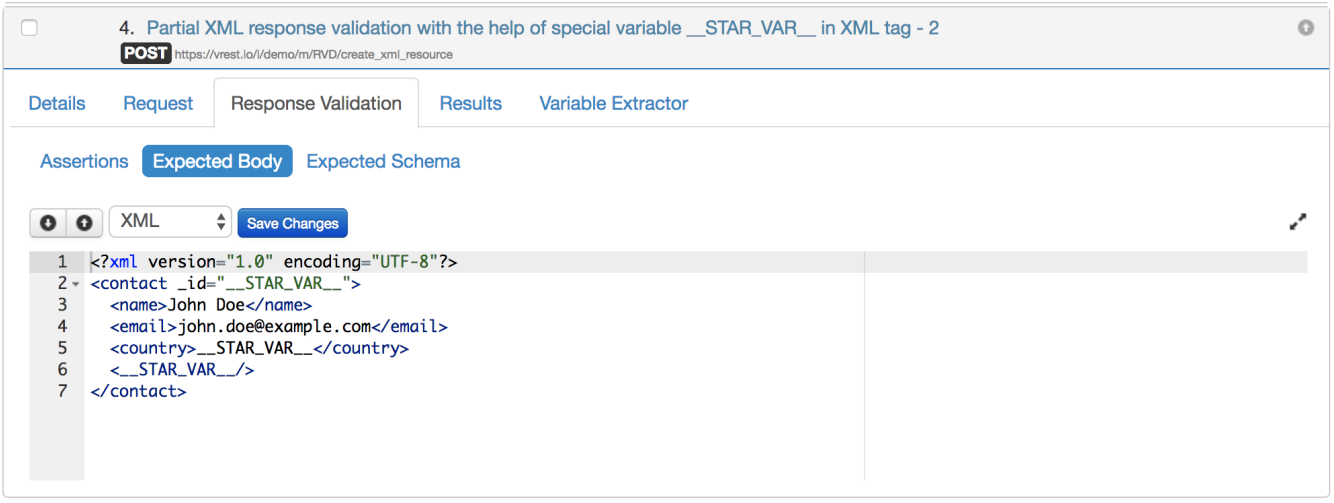
Status Code: 200

```
<?xml version="1.0" encoding="UTF-8"?>
<contact _id="536493015f56452a03000010">
   <aboutMe>My name can be used as a placeholder name and I don't have any identity.</aboutMe>
   <country>India</country>
   <createdOn>2014-05-03T06:56:01.134Z</createdOn>
   <designation>Chief Technical Officer</designation>
   <email>john.doe@example.com</email>
   <facebookId>fake.john.doe</facebookId>
   <githubId>fake.john.doe</githubId>
   <name>John Doe</name>
   <organization>Example.com</organization>
   <twitterId>fake.john.doe</twitterId>
</contact>
```

And in the above response, you only want to validate name and email values.

For such scenarios, simply use the special variable in xml tag **<__STAR_VAR__/>** to ignore rest of the keys and values. Now, the expected body should look like this:

Further, let us suppose, you want to validate only the existence of data in country tag in your response, not the value of country tag, then you can mix this scenario with scenario 2 and write your expected body like this:



Scenario 4: My API returns some response in which some part of the response can be obtained from the responses of previous test cases.

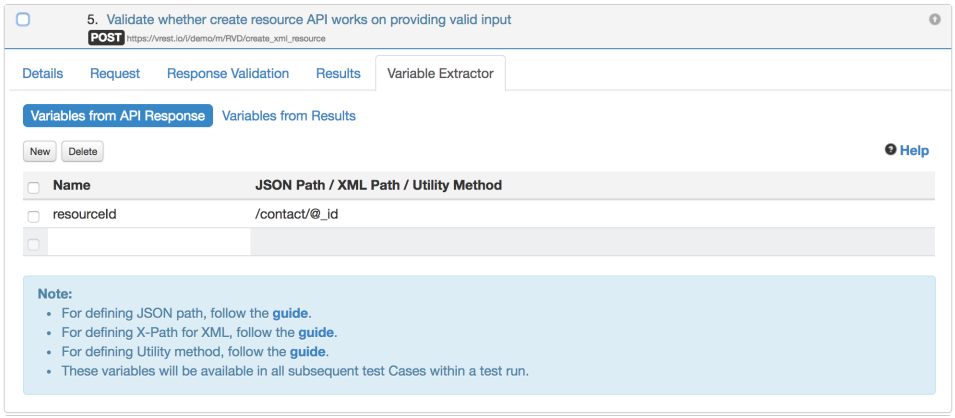
Let us take an example in which one test case creates a resource on server and second test case updates that newly created resource.

- Suppose you have an API which creates resource on server and returns the following XML response:

Status Code: 200

```
<?xml version="1.0" encoding="UTF-8"?>
<contact _id="536493015f56452a03000010">
   <aboutMe>My name can be used as a placeholder name and I don't have any identity.</aboutMe>
   <country>India</country>
   <createdOn>2014-05-03T06:56:01.134Z</createdOn>
   <designation>Chief Technical Officer</designation>
   <email>john.doe@example.com</email>
   <facebookId>fake.john.doe</facebookId>
   <githubId>fake.john.doe</githubId>
   <name>John Doe</name>
   <organization>Example.com</organization>
   <twitterId>fake.john.doe</twitterId>
</contact>
```

- Now, you can save the _id of newly created resource into variable say "resourceId". You can extract the variable in the following way:

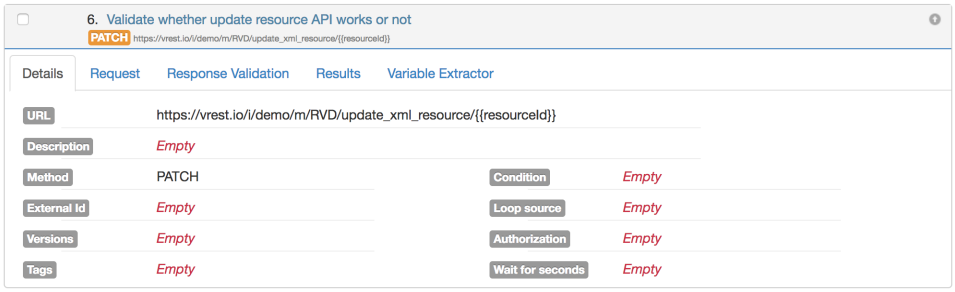


Few points regarding writing **Path** in the above table:

- Each individual property value can be extracted via XML Path (XPath) expression e.g. `_id`
- For more information, read XML Path syntax

- Now you can use these extracted variables in subsequent requests. Note that once a variable is defined, it can be used in all subsequent requests within that test run only. If you want to override this variable, simply re-define the variable in any request.

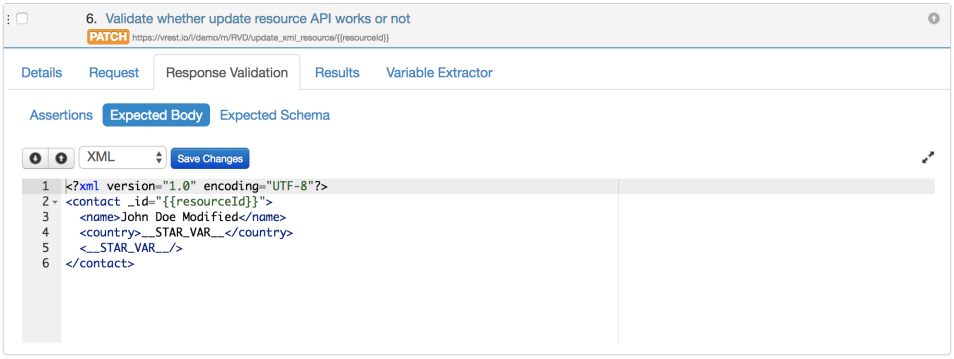
  Now, suppose you have an API which updates this newly created resource and it needs the ID of the resource to update. You can use the {{resourceId}} variable (extracted in previous step) in the URL as shown in the following figure:



- And let us suppose, the Update API returns the following response:

```
<?xml version="1.0" encoding="UTF-8"?>
<contact _id="536493015f56452a03000010">
   <name>John Doe Modified</name>
   <aboutMe>My name can be used as a placeholder name and I don't have any identity.</aboutMe>
   <country>India</country>
   <createdOn>2014-05-03T06:56:01.134Z</createdOn>
   <designation>Chief Technical Officer</designation>
   <email>john.doe@example.com</email>
   <facebookId>fake.john.doe</facebookId>
   <githubId>fake.john.doe</githubId>
   <organization>Example.com</organization>
   <twitterId>fake.john.doe</twitterId>
</contact>
```

Now, you can write our expected body like this:



Note: In the above test case, fields "_id" will be replaced from the values extracted from previous test case. So, we can use Default Validator in such scenarios.

> **Note**
>
> If you think, your scenario is not covered here then you can discuss your scenario with us by sending an email to "support@vrest.io".

Note:

- The above scenarios are also covered in our online demo available at Response Validation Demo
- Variables plays an important role in response validation. See also Environments / Variables