Testing RESTful Web Services

Testing RESTful Web Services is supported via the *REST Client* bundled plugin, which is by default enabled. If not, activate it in the Plugins page of the Settings dialog box.

There are two main use cases when you need to compose and run requests to a RESTful Web service:

- When you have developed and deployed a RESTful Web service and want to make sure it works as expected: that it is accessible in compliance with the specification and that it responds correctly.
- When you are developing an application that addresses a RESTful Web service. In this case it is helpful to investigate the access to the service and the required input data before you start the development. During the development, you may also call the Web service from outside your application. This may help locate errors when your application results in unexpected output while no logical errors are detected in your code and you suspect that the bottleneck is the interaction with the Web service.

Testing a RESTful Web service includes the following checks:

- That URL addresses are constituted correctly based on the service deployment end-point and the method annotations.
- That the generated server requests call the corresponding methods.
- That the methods return acceptable data.

IntelliJ IDEA enables you to run these checks from the REST Client tool window by composing and submitting requests to the local server, viewing and analyzing server responses.

If necessary, configure the Proxy settings on the HTTP Proxy page of the Settings dialog box.

On this page:

To compose and submit a test request to a Web service method

1. If you are going to test your own Web service, make sure it is deployed and running.
2. Choose **Tools | Test RESTful Web Service**. The REST Client dedicated tool window opens.
3. Select the test request method from the **HTTP method** drop-down list. The available options are:
   - GET
   - POST
   - PUT
   - PATCH
   - DELETE
   - HEAD
   - OPTIONS

4. Provide the data to calculate the URL address of the target method:
    1. In the **Host/port** text box, type the URL address of the host where the Web service is deployed.
    2. From the **Path** drop-down list, choose the **@path** annotation that corresponds to the method the call to which you need to test.

    > IntelliJ IDEA supports integration between the contents of this list and Web service Java source code. Any changes to the `>@Path` annotations are reflected in the contents of the **Path to Resource** list.
    >
    > To synchronize the contents of the **Path** drop-down list with the **@Path** annotations, click the **Update resource paths from code** button .

5. In the **Header data** pane, specify the technical data included in the request header. These data are passed through header fields and define the format of the input parameters (accept field), the response format (content-type field), the caching mechanism (cache-control field), etc.

    To add a field to the list, click **Add** +, then specify the field name in the **Name** text box and the field value in the **Value** drop-down list.

    The set of fields and their values should comply with the Web service API. In other words, the specified input format should be exactly the one expected by the Web service as well as the expected response format should be exactly the one that the service returns.

    For `accept`, `content-type`, and some other fields IntelliJ IDEA provides a list of suggested values. Choose the relevant format type from the **Value** drop-down list.

6. Create a set of parameters to be passed to the target method and specify their values. Depending on the chosen request method, you can create a list of parameters in two ways:
    - For `GET` requests, specify the parameters to be passed as a query string inside the URL. Use the **Request Parameters** pane. By default, the pane shows an empty list with one line.
        - To add a parameter, click **Add** +, then specify the name of the parameter in the **Name** text box and the value of the parameter in the **Value** drop-down list.
        - To delete a parameter from the list, select it and click **Remove** —.
        - To suppress sending the specified query string parameters and disable the controls in the **Request Parameters** pane, press the **Don't send anything** toggle button .
    - To have the parameters passed to the target method inside a request message body, use the **Request Body** pane or have them inserted in the request from a local file.
        - To specify the parameters explicitly, choose the **Text** option and type the parameters and values in the text box.
        - To have the parameters inserted from a text file, choose the **File contents** option and specify the file location in the **File to send** field.
        - To have a binary file converted and sent in the request, choose the **File upload(multipart/form-data)** option and specify the file location in the **File to send** field.

    In either case, the set of parameters and their types should comply with the Web service API, in particular, they should be exactly the same as the input parameters of the target method.

7. To submit a request to the server, click the **Submit request** button.

   Note that the server may lack certificate, or be untrusted.

   > If a server is not trusted, IntelliJ IDEA shows a dialog box suggesting you to accept the server, or reject it. If you accept the server as trusted, IntelliJ IDEA writes its certificate to the trust store. On the next connect to the server, this dialog box will not be shown.

Viewing and analyzing responses from a web service

- To view the response to the server request, switch to the **Response** tab. The tab is opened automatically when a response is received. By default, the server response is shown in the format, specified in the request header through the content-type field.
- To have the response converted into another format and opened in a separate tab in the editor, use the **View as HTML**, **View as XTML**, or **View as JSON** buttons.
- To view the technical data provided in the header of a Web service response, switch to the **Response Headers** tab.

Working with cookies

Using the dedicated **Cookies** tab, you can create, save, edit, and remove cookies, both received through responses and created manually. The *name* and *value* of a cookie is automatically included in each request to the URL address that matches the *domain* and *path* specified for the cookie, provided that the *expiry date* has not been reached.

The tab shows a list of all currently available cookies that you received through responses or created manually. The cookies are shown in the order they were added to the list. When you click a cookie, its details become editable and are displayed in text boxes. See REST Client Tool Window for details.

- No specific steps are required to save a cookie received through a response, all the cookies received from servers are saved automatically. To edit a received cookie, click the row with the cookie in the list and update the details that are now shown in editable text boxes.
- To create a cookie manually, click + and specify the following:
  - The *name* and *value* of the cookie to be included in requests.
  - The *domain* and *path* the requests to which must be supplied with the *name*, *value*, and *expiry date* of the cookie.
- To remove a cookie from the list, select the row with the cookie and click —.

Configuring proxy settings

1. Click the **Configure HTTP Proxy** icon.
2. In the Proxy dialog that opens, specify the following:
   - Enter the Proxy host name and Proxy port number in the **Proxy host** and **Proxy port** text boxes respectively.
   - To enable authorization, check the **Use authorization** check box and type the User name and password in the relevant fields.

Reusing requests

During a IntelliJ IDEA session, IntelliJ IDEA keeps track or requests and you can run any previously executed request. You can also save the settings of a request in an XML file so they are available in another IntelliJ IDEA session. When necessary, you can retrieve the saved settings and run the request again.

- To rerun a request within a IntelliJ IDEA session:
  1. Click the **Replay Recent Requests** button
  2. From the **Recent Requests** pop-up list, select the relevant request. The fields are filled in with the settings of the selected request.
  3. Click the **Submit Request** button .
- To save the settings of a request so they can be retrieved in another IntelliJ IDEA session, click the **Export Request** button . In the dialog box that opens, specify the name of the file to save the settings in and its parent folder.
- To run a request saved during a previous IntelliJ IDEA session:
  1. Click the **Import Request** button .
  2. In the dialog box that opens, select the relevant XML file. The fields are filled in with the settings of the selected request.
  3. Click the **Submit Request** button .

See Also

Procedures:

- RESTful WebServices

Reference:

- REST Client Tool Window

Web Resources:

- Developer Community