Prior to reading this page, it is important to have an understanding of the API from a non-technical point of view. First read the REST Web Services API page before going through this one.

These pages exist to help a developer walk through creating a new ws. They might be more helpful to some devs than this page:

- Adding a Web Service Step by Step Guide for Core Developers
- Adding a Web Service Step by Step Guide for Module Developers

REST URL request information flow

When a user calls

```plaintext
GET /openmrs/ws/rest/v1/patient/939921-32-DA-32342
```

this is the order of actions performed:

1. The server sees the /ws/ as part of the url and so sends the request to Spring
2. There is a filter in the restws module named AuthorizationFilter that watches all /ws/rest/* urls. It is called and will authenticate the user (using BASIC or a sessionid cookie) and put those credentials into Context.getAuthenticatedUser(). The filter will also reject requests from non-allowed IP addresses
3. MainResourceController and MainSubResourceController handle all requests to /ws/rest/* urls and delegate to appropriate resources and subresources. MainResourceController knows the PatientResource1_8 class has been linked to /ws/rest/1/patient because of the @Resource annotation.
4. MainResourceController has the retrieve GET method, which reads resource and uuid from the url.

```java
@RequestMapping(value = "/{resource}/(uuid)", method = RequestMethod.GET)
@ResponseBody
public Object retrieve(@PathVariable("resource") String resource, @PathVariable("uuid") String uuid, HttpServletRequest request)
```

5. The retrieve in the controller sets up the RequestContext for that request and calls the retrieve method from the resource.
6. The DelegatingCrudResource parent class to PatientResource1_8 has the implementation of the "retrieve" method. This method calls the getByUniqueId method that is actually defined on the PatientResource1_8.
   - getByUniqueId can match on either uuid or a unique name, if that object has it. Patient does not have a unique name string we can match to, so it only does a lookup on uuid using the PatientService.getPatientByUuid method.
7. The Patient object returned from the resource is converted into a SimpleObject (which is just a map of properties to values) by the rest framework
   - BaseDelegatingResource.asRepresentation tries to get the representation format from the getRepresentationDescription method on the PatientResource1_8 first.
   - If there is no description defined there, the @RepHandler annotation is looked for on PatientResource1_8 or on its parent classes.
8. The representation description is used against the Patient object to create the map of key-value pairs that are returned to the user.
   - Most properties will be located automatically on the delegate (the Patient object)
   - Some properties can be defined on the resource (PatientResource1_8), such as "getDisplayString" in the "ref" rep or the "getAuditInfo" in the "full" representation.
9. Spring will automatically convert the SimpleObject Map to json (or xml) for the text returned to the user because of the "MappingJacksonHttpMessageConverter" defined in the spring webModuleApplicationContext.xml config file in the restws module.

Examples using Curl

```plaintext
```

```plaintext
```

```plaintext
echo '{"patient":"5ee9b94c-81f8-4c15-96ec-claa9c0b0d66"}' | curl -i -u admin:test -H "Content-type: application/json" -X POST -d @- http://localhost:8080/openmrs/ws/rest/v1/cohort/0bbc73bd-0986-4581-9d3a-87cedeal3251/members
```
curl -i -u admin:test http://localhost:8080/openmrs/ws/rest/v1/cohort/0bbc73bd-0986-4581-9d3a-87cdeea13251
/members

curl -i -u admin:test http://localhost:8080/openmrs/ws/rest/v1/cohort/0bbc73bd-0986-4581-9d3a-87cdeea13251

```bash
echo '{
  "patient":"5ee9b94c-81f8-4c15-96ec-c1aa9c0b0d66",
  "encounterDatetime":"2011-02-18",
  "location":"Unknown Location",
  "encounterType":"ADULTINITIAL",
  "provider":"5ee9b94c-81f8-4c15-96ec-c1aa9c0b0d66"
```

```bash
echo '{
  "patient":"5ee9b94c-81f8-4c15-96ec-c1aa9c0b0d66",
  "encounterDatetime":"2011-02-18",
  "location":"Unknown Location",
  "encounterType":"ADULTINITIAL",
  "provider":"5ee9b94c-81f8-4c15-96ec-c1aa9c0b0d66",
  "obs": [ {
    "concept":"WEIGHT",
    "value":"70"
  } ]
```

```bash
echo '{
  "encounterDatetime":"2011-02-19"
}' | curl -i -u admin:test -H "Content-type: application/json" -X POST -d @- http://localhost:8080/openmrs/ws/rest/v1/encounter/403a41ac-8f5e-45a0-976c-45303ab3aa4d
```
Note

When using curl with multiple request parameters, you might note occurrences where the application fails to identify anything other than the very first request parameter in the url. This is because request parameters are separated by the & (ampersand character) This character may signify that any commands which follow immediately behind should be executed simultaneously. The only way to avoid such a situation is to surround your entire URL with a double quote (" "). Doing so will resolve the above error.

Other topics?

Please suggestion topic headings for other devs to fill in...

- Postman Rest Client  https://chrome.google.com/webstore/detail/fdmmgilgnpjigdojpjoooidkmcomcm  This plugin Chrome, allows resolve easily query through webservice.rest