Account API

Contents

[1. API Objective: 1](#_Toc518952148)

[2. HTTPs method mapping and status code: 3](#_Toc518952149)

[3. Quick design using JAX Jersey framework in Eclipse for JAVA 5](#_Toc518952150)

[4. Test Data/payload and response 7](#_Toc518952151)

[5. Create documentation file in swagger Hub and share API 8](#_Toc518952152)

[6. Additional testing tool: 8](#_Toc518952153)

# API Objective:

The API would need to allow handling the below resources, the design will use RESTful API, The Account APIs web services use HTTP requests to specific URLs, passing URL parameters as arguments to the services. Generally, these services return data in the HTTP request as either JSON or XML for parsing and/or processing by your application.

API functions overview:

- creation of a new account

- retrieval of account information

- changes to an existing account

- deleting of an existing account

- creation of a new subscription

- retrieval of account information

- changes to an existing subscription

- deleting of an existing subscription

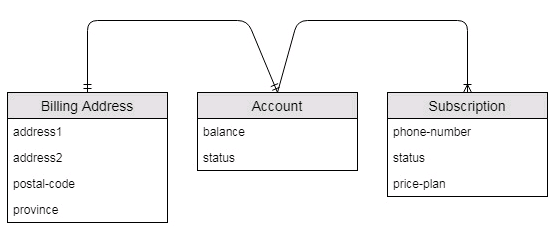
- creation of a billing address

- retrieval of a billing address

- changes to an existing billing address

- deleting of an existing billing address

**Entity relationships:**

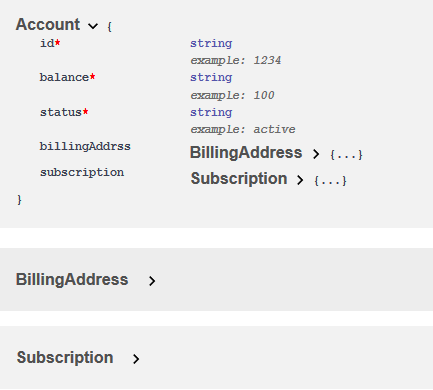


* An account must have exactly one billing address.
* An account may have zero or more subscriptions.
* Rules for CRUD: delete actions

Additional consideration

Authentication (OAuth 2.0/client Id ) and scope (roles)

* Domain Object: Account/BillingAddress/Subscription



Account:

required:

- id\*

- balance\*

- address\*

properties:

balance:

type: string

example: 100

status:

type: string

example: active

address:

$ref: '#/account/address'

subscription

$ref: '#/account/subscription’

\*required fields

Address:

type: object

required:

- addressline1

- post-code

- province

properties:

addressline1:

type: string

example: one street east

addressline2:

type: string

example: two street east

post-code

type: string

example: a1a2b2

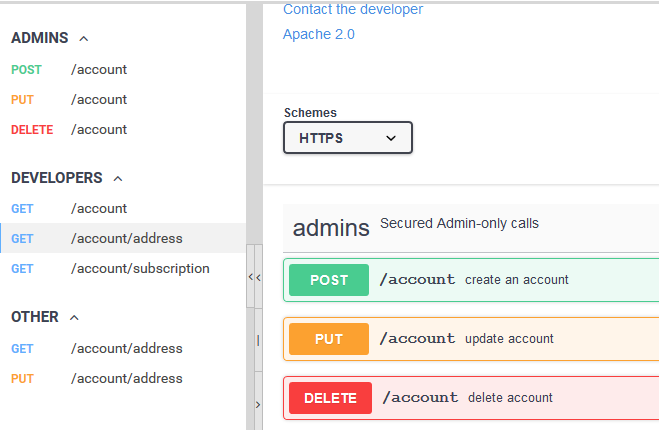
province

type: string

example: Ontario

* Scope and Roles:

Admins/Developers/other



# HTTPs method mapping and status code:

HTTPS is required for all API web service requests containing user data, or developer identifiers. Requests made over HTTP that include sensitive data will be rejected.

endpoints mapping:

200, 201, and 204 for successful responses, and 4XX codes for errors.

* **Account:**

POST / account HTTP/1.1 201: description: item created

Created account

GET / account/{accountId } HTTP/1.1 200 OK description: search results matching criteria

GET / account Search all or individual by id (404 not found)

PUT / account/{accountId} update account

DELETE / account/{accountId} delete account (role)

HTTP/1.1 401: description: Unauthorized

/\*need DELETE address and subscription, logic and rules here, administrator to delete, transaction handling\*/

* **BillingAddress**

POST / account/{accountId}/ address

GET / account/{accountId}/ address

PUT / account/{accountId}/ address

DELETE / account/{accountId}/ address /\* DELETE: not allowed if account exist\*/

* **Subscription**

GET / account/{accountId}/subscriptions

POST / account/{accountId}/ subscription

PUT / account/{accountId}/ {subscriptionId} /subscription

DELETE / account/{accountId}/ {subscriptionId }

GET / account? accountId =1234

HTTP/1.1 200 OK

{ Account {id: 1234, status “Active”, balance:100,

address:

{ id: 6789, address1: "street one", address: "street two", postcode: "a1a2a2" province: "ON" },

subscriptions: [ {subscriptionId :1,phoneNumber:4162541234568,status:”Active”,pricePlan:”Family”},

{ subscriptionId :2,phoneNumber:4161234567,status:”Active”,pricePlan:”Corp”}

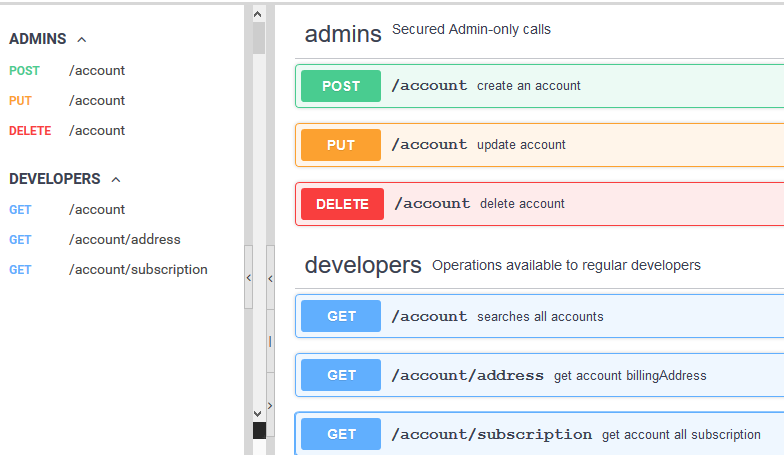
]

}

* GET account information example results using HTTPS (not http)

https://app.swaggerhub.com/apis/BlueJay4/JayAccount/1.0.0#/developers/Account





# Quick project using JAX Jersey framework in Eclipse for JAVA

THIS is Optional, API can be implemented with any language, just quick demo

package com.telus.test;

import javax.ws.rs.GET;

import javax.ws.rs.Path;

import javax.ws.rs.Produces;

import javax.ws.rs.core.MediaType;

@Path("/account")

public class Account {

@GET

@Produces(MediaType.TEXT\_XML)

public String getAccount(){

// TODO refer to testing mock data

String resource ="<? xml version='1.0'><account></account>";

return resource;

}

@GET

@Produces(MediaType.APPLICATION\_JSON)

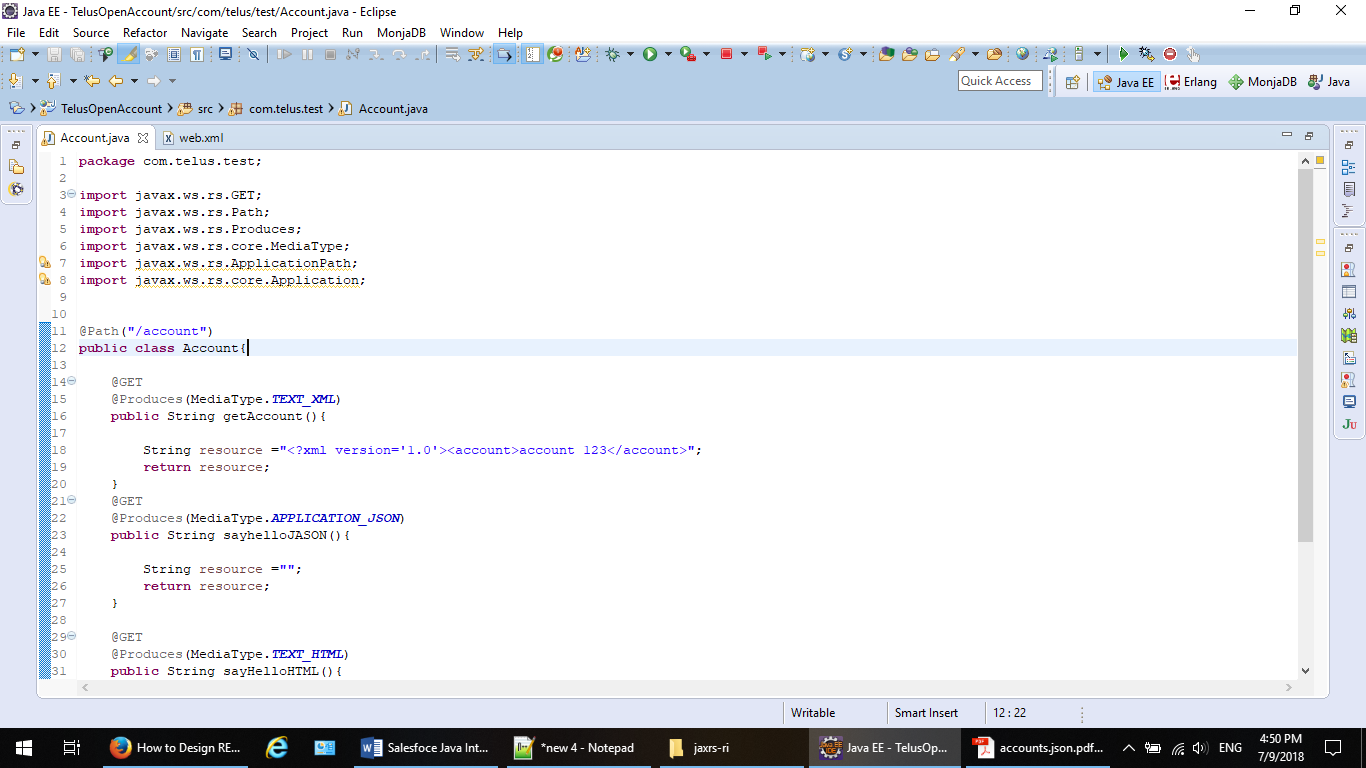
public String getAccountJSON(){

//TODO refer test mock data

String resource ="";

return resource;

}



Sample: TOMCAT server

web.xml configuration for URL pattern and in servlet mapping, java package and project info.

web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd" id="WebApp\_ID" version="3.1">

<display-name>TelusOpenAccount</display-name>

<servlet>

<servlet-name>JavaAPI</servlet-name>

<servlet-class>org.glassfish.jersey.servlet.ServletContainer.class</servlet-class>

<init-param>

<param-name>jersey.config.server.provider.packages</param-name>

<param-value> com.telus.test</param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>JavaAPI</servlet-name>

<url-pattern>/account/\*</url-pattern>

</servlet-mapping>

</web-app>

# Test Data/payload and response

payloads and responses data in API tool

* JSON format

{ Account {id: 1234, status “Active”, balance:100,

address:

{ id: 6789, address1: "street one", address: "street two", postcode: "a1a2a2" province: "ON" },

subscriptions: [{subscriptionId :1,phoneNumber:4162541234568,status:”Active”,pricePlan:”Family”},

{ subscriptionId :2,phoneNumber:4161234567,status:”Active”,pricePlan:”Corp”}

],

Account {id: 4321, status “Active”, balance:100,

address:

{ id: 6789, address1: "street one", address: "street two", postcode: "a1a2a2" province: "ON" },

subscriptions: [ {subscriptionId :1,phoneNumber:4162541234568,status:”Active”,pricePlan:”Family”},

{ subscriptionId :2,phoneNumber:4161234567,status:”Active”,pricePlan:”Corp”}

]}}

* XML format

<?xml version="1.0" encoding="utf-8" ?>

<accounts>

< account id="1234">

<status>Deactive</ status >

<balance>100</balance>

<address>

</address>

<subscriptions>

<subscription id =”1”>

<phone></phone>

<subscription>

<subscription id =”2”>

<phone></phone>

<subscription>

</subscriptions>

<link rel="self" href="/account/1234" />

<link rel="subscriptions" href="/account/1234/subscriptions">

</ account>

< account id="7312">

< status >Active</ status >

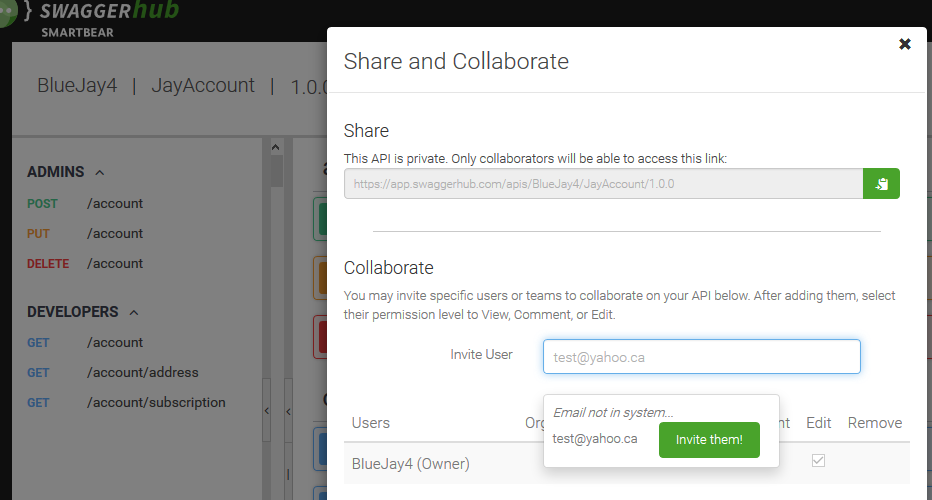
< balance >1955</ balance >

</ account >

</ accounts>

# Create API documentation in swagger Hub and share API

Share and collaborate in SWAGGER hub



# Additional testing tool:

