* **API versioning**
* **Environment Modes**
* **Criteria API**

**API versioning:**

If we change any API on production then it might be break for existing client. Like in our model class now I want to add more fields.

There are 4 ways to do the API versioning.

* **Send API version using URI Path parameter:** Create a new model class with updated field and put the version from @GetMapping(value = "/api/**v1**/stock1") to @GetMapping(value = "/api/**v2**/stock2").The older version will also there for old client.
* **Send API version using Request Parameter:** Pass the version like this

@GetMapping(value = "/api/stock", params="version=2")

[**http://localhost:8080/api/stock?version=2**](http://localhost:8080/api/stock?version=2)

* **Send API version using Header Parameter:** we can change the version without change the URL. Just pass the version in header.

@GetMapping(value = "/api/stock", headers="version=v2")

* Send API version using Content Negotiation: Use this for content Negotiation and pass the “application/stock.v2+json” as Accept in header
* @GetMapping(value = "/api/stock", produces="application/stock.v2+json") //v2

**Environment Modes:** For any application we should create different properties file like

**application.yml :** this one is the default property file.

**application-dev.yml:** we should create this for development purpose. Here, after “-“ the word will be consider as application name and the “App” word will be add as a postfix. So the application name is **DevApp**.

**application-prod.yml:** Same we can use this for production application. And the application name will be **ProdApp**.

**If we want to use prod or dev so in that case we have to use the below property in main application property file and it will get all the properties from specified file(dev/prod).**

spring:

profiles:

active:

- dev

Note: If we will put any wrong name then default application property will work.

And to get the properties value we can use the “@value”.

@RestController

@RequestMapping

**public** **class** VersionController {

@Value("${spring.application.name}")

**private** String appName;

@GetMapping(value = "/")

**public** String getStocks() {

**return** appName;

}

}

Create a class to get the app name.

@Component

**public** **class** AppConfig {

@Value("${spring.application.name}")

**private** String appName;

**public** String getAppConfig() {

**return** appName;

}

}

Just make it Autowired in controller class and call the getAppConfig method.

How to run the particular code for different profile? Like if profile is prod then create the real database object or if profile is dev then work on local db.

-Using @Profile

@Profile(value = "dev")

**public** **void** executeAnyCode()

{

System.***out***.println("ApiVersioningApplication.enclosing\_method()");

}