RestAssured Cheat Sheet

Ways to set multiple headers in Request:

```
1. Using multiple header() calls

      given()
      .header("Content-Type", "application/json").
      .header("Accept", "application/json")
      .when().


2. Using Map<String, String>

Map<String, String> map = new HashMap<>();

map.put("Content-Type", "application/json");
map.put("Accept", "application/json");

given().headers(map).when().


3. Using Headers class

Header h1 = new Header("Content-Type", "application/json");
Header h2 = new Header("x-request-tracking-id", "2aebe4ctrtaaappa");

Headers headers = new Headers(h1, h2);

given().headers(headers).when().
```

Ways to get header from Response:

```
Response response = given().when().get("https://api.example.com/data");

// Single header
String contentType = response.getHeader("Content-Type");

// All headers
Headers allHeaders = response.getHeaders();
for(Header h : allHeaders) {
    System.out.println(h.getName() + " : " + h.getValue());
}
```

Ways to set multiple cookies in Request:

```
1. Using multiple cookie() calls

given()
    .cookie("session_id", "12345")
    .cookie("user", "varun")
    .cookie("role", "admin")
    .when().

2. Using cookies(Map<String, String>)

Map<String, String> map = new HashMap<>();
map.put("session_id", "12345");
map.put("user", "varun");
map.put("role", "admin");
```

```
given().cookies(map).when().

3. Using Cookies class

Cookie c1 = new Cookie.Builder("session_id", "12345").build();
Cookie c2 = new Cookie.Builder("user", "varun").build();
Cookie c3 = new Cookie.Builder("role", "admin").build();

Cookies cookies = new Cookies(c1, c2, c3);

given().cookies(cookies).when().
```

Why Cookie uses a Builder ??

```
- HTTP cookie can have MANY attributes besides name=value:
  session_id=12345; Domain=example.com; Path=/; Secure; HttpOnly; Max-Age=3600
- Needs flexibility to configure optional properties.
- RestAssured uses Builder pattern for fluent setup:

  Cookie c = new Cookie.Builder("session_id", "12345")
                .setSecured(true)
                .setHttpOnly(true)
                .setDomain("example.com")
                .setPath("/")
                .build();

■ Shortcut: If we only need simple name=value cookies, then there is no need for Cookie class:
   given().cookie("session_id", "12345");
```

Ways to get Cookie(s) from RestAssured Response:

```
Response response = given().when().get("https://api.example.com/data");

// 1■■ Get a single cookie by name
String sessionId = response.getCookie("session_id");

// 2■■ Get all cookies as Map<String, String>
Map<String, String> allCookies = response.getCookies();
for (Map.Entry<String, String> cookie : allCookies.entrySet()) {
    System.out.println(cookie.getKey() + " : " + cookie.getValue());
}

// 3■■ Get detailed cookie object
Cookie detailedCookie = response.getDetailedCookie("session_id");
System.out.println("Detailed Cookie: " + detailedCookie);

// 4■■ Get all detailed cookies
Cookies detailedCookies = response.getDetailedCookies();
System.out.println("All Detailed Cookies: " + detailedCookies);
```

RestAssured example showing both query parameters and path parameters:

```
// URL: https://api.example.com/users/101?active=true&country=India
Response combinedResponse = given()
                            .pathParam("userId", 101)
                            .queryParam("active", true)
                            .queryParam("country", "India")
                            .when().get("https://api.example.com/users/{userId}");

// Multiple Query Params via Map
// URL: https://api.example.com/users?role=admin&active=true&dept=QA
```

```
Map<String, Object> params = new HashMap<>();
params.put("role", "admin");
params.put("active", true);
params.put("dept", "QA");

Response resp = given()
                  .params(params)
                  .when()
                  .get("https://api.example.com/users");
```

Extract Response fields using JSONPath in RestAssured:

```
Response response = given().when().get("https://api.example.com/user/101");

int id = response.jsonPath().getInt("id");
String name = response.jsonPath().getString("name");
String city = response.jsonPath().get("address.city").toString();
List<String> skills = response.jsonPath().getList("skills");
```