ReSTFul API

Orkun Krand

Old Dominion University

**ABSTRACT**

In this paper, we talk about ReSTFul APIs, their purpose and benefits.

**Keywords**

Rest, Representative State Transfer, API, Application Program Interface, Web, Server, Client, Stateless

**INTRODUCTION**

ReST stands for Representational State Transfer. It is stateless which means no information is stored on the client side. Rest uses HTTP requests to interact with the server. This may seem like a primitive way of interaction but that is somewhat the point. It is designed to be as simple and primitive as possible to avoid overhead that comes with pretty web pages.

API stands for Application Program Interface. Many big websites such as Google and Twitter use APIs so people can access their data or services in their own applications. For example, Twitter API will allow a user to post a tweet from another website such as Amazon.

**RESTFUL API**

The purpose of a Restful API is to be able to control and/or monitor the server without a fancy representation, using HTTP commands like GET, PUT, POST, DELETE. It can act as admin tools, if the endpoints are only accessible to members of the software/IT. Since calling an API doesn’t require processor-heavy functions like rendering UI, it is a fast and effective way to retrieve data from an application. Endpoints take a familiar form (and can even be documented in a familiar way with tools like Swagger), so they are useful for abstracting away complicated architecture from a programmer new or external to the project. With a RestFul API admin page, an admin doesn’t need to know SQL to interact with the records on the server. Because there are no sessions on a Restful API, security of information is provided via tokens which is a password that will only be given to admins.

If I did more research about RestFul APIs before I started implementing Highside, I could’ve built my whole website API-First. I would first describe the inputs and outputs of my system by defining API functions. I would then create API calls for all of the backend transactions. It would particularly have made creating the admin tools page easier, as the API itself could be considered an administrative tool. I would be able to avoid duplicating so many lines of SQL in PHP and I could’ve just made API calls to the database in my PHP code. This approach, which is called API-First development, would’ve made the admin tools which I intended to make for a semester project much easier and faster to implement. Another benefit of using the API in my client side would be to make the code easier to read and understand rather than using functions and queries that only I know the mechanics of. The backend information would have been completely abstracted from the user.

There are many ways to create an interface for a ReSTFul API one of which is to use a pretty looking third party UI builder such as Swagger or one can create an HTML interface to interact with the API. There are also add-ons such as Poster for Chrome that can do HTTP requests on a website which your API would catch and execute.

**CONCLUSION**

RestFul APIs are used by many companies big and small to maintain server information without requiring to know database languages. It is a very useful tool to have implemented in one’s website. It makes implementing the user interface much simpler. My only wish is that I had more time to be able to actually implement this in my website rather than the very small amount I was able to implement in the time I dedicated to this project.

**REFERENCES**

* <http://swagger.io/>
* <http://rest.elkstein.org/>
* <http://coreymaynard.com/blog/creating-a-restful-api-with-php/>
* http://blog.ijasoneverett.com/2013/02/rest-api-a-simple-php-tutorial/