HTTP Methods Worksheet-KEY

Created By Ryan Boyer (github, linkedin)
2022/07/06

Instructions

1. **What are the two most common HTTP Methods?**
   Get, Post
2. **What are the next three most common (total of five with question 1) HTTP Methods?**
   Put, Delete, Patch
3. **For each of the 5 major HTTP Methods, fill in the table:**

Method	Question	Answer
Get	What does it do?	*Retrieves data from a server*
	How do you pass data?	*You don't really pass data, just query parameters to specify what data to retrieve.*
	Typical CRUD Companion?	*Read*
	Example Use Case	*Get a list of orders belonging to a user*
	Example API Call	*GET https://www.myserver.com/orders?customerId=1234*
Post	What does it do?	*Sends data to a server; multiple requests of same values result in multiple actions on the recipient (i.e. multiple orders placed)*
	How do you pass data?	*- Supply a Content-Type in your header* *- Supply a request body with data in the same format as supplied by your header* *- Query parameters typically aren't used*

	Typical CRUD Companion?	*Create*
	Example Use Case	*Upload a post to a blog*
	Example API Call	*POST https://www.myserver.com/blog/add-post* *Content-Type: application/json* *Body: {* *"blog_id": 1234,* *“title”: “first post!”,* *"blog_content": "this is my first blog post"* *}*
Put	What does it do?	*Sends data to a server. Multiple requests of same values are idempotent (i.e. only a single order is placed)*
	How do you pass data?	*- Supply a Content-Type in your header* *- Supply a request body with data in the same format as supplied by your header* *- Query parameters may be used*
	Typical CRUD Companion?	*Update/Replace*
	Example Use Case	*Upload a post to a blog, replacing an existing one if it exists at the given path*
	Example API Call	*PUT https://www.myserver.com/blog/add-post?postId=1* *Content-Type: application/json* *Body: {* *"blog_id": 1234,* *“title”: “first post!”,* *"blog_content": "this is my first blog post"* *}*
Delete	What does it do?	*Asks the server to delete a resource*
	How do you pass data?	*You don’t really pass data, just query parameters to specify what data to retrieve.*
	Typical CRUD Companion?	*Delete*

	Example Use Case	*Delete a blog post*
	Example API Call	*DELETE https://www.myserver.com/blog/delete-post?id=1*
Patch	What does it do?	*Partially modify a resource on a server*
	How do you pass data?	*- Supply a Content-Type in your header* *- Supply a request body with data in the same format as supplied by your header* *- Query parameters may be used*
	Typical CRUD Companion?	*Update/Modify*
	Example Use Case	*Change the title of a blog post, but not the body*
	Example API Call	*PATCH https://www.myserver.com/blog/add-post?postId=1* *Content-Type: application/json* *Body: {* *"title": "OH YEAH BABY - first post!",* *}*

4. **What's the difference between a *get* request and a *head* request?**
   A head request response includes headers, but no body. A get request includes the headers AND the body. A head request can tell you if something exists, but not show it to you.
5. **What's the difference between a *post* request and a *put* request?**
   A put request is idempotent while a post request is not. This means that sending multiple of the same post request will create multiple resources on the server, while multiple of the same put requests should only create one resource.
6. **Does the type of request <u>enforce</u> its behavior?**
   No, the request types are a way for developers to quickly understand and specify what an API does; it's up to the developers on the backend to implement actions correctly in response to the request type, parameters, and payload.
7. **Does every API support all http methods?**
   No, the developers have to enable each of them.
8. **What are the remaining request types we haven't mentioned?**
   They are:

- *Head - like a get request, but doesn't return a body*
- *Connect - establishes a tunnel to the server*
- *Options - describes methods for communicating with an API. (i.e supports, Get, Head, Post, etc).*
- *Trace - Used mainly for debugging, performs a message loop-back test*