Request Plugin Documentation

Introduction..... 1
Important Features..... 1
Using Plugin..... 2
GET Example..... 3
POST Example..... 5
Get Array Example..... 7

Introduction

Welcome to the Request Plugin documentation. This plugin is designed to simplify the process of making HTTP requests and handling responses in Unreal Engine 5. It provides a straightforward way to interact with web services and APIs, whether you're fetching data, sending data, or working with arrays of structured data.

This documentation will guide you through the key features and usage of the Request Plugin, helping you understand how to interact with your API and integrate it into your Unreal Engine 5 projects.

Important Features

The Request Plugin supports various data types, including integers, floats, booleans, strings, text, arrays, and enums. It also handles inner struct properties that contain primitive properties. You can take full advantage of these features to tailor the plugin to your specific API needs.

By following this documentation, you'll be able to harness the power of the Request Plugin in Unreal Engine 5 for seamless API interaction.

Feel free to refer to specific sections of this documentation as needed for your project requirements. We're confident that this plugin will streamline your development process and help you create powerful and efficient Unreal Engine 5 projects.

Using Plugin

When you need to retrieve data from an API, the Request Plugin makes it easy. Here are the steps to follow:

- Define a struct that matches the API response structure.
- Prepare the request options, specifying the URL, method (GET), and any required headers.
- Send the request, and the response data will be automatically mapped to your struct.

Sending data to an API is as straightforward as fetching it. The steps are similar:

- Prepare the data structure that you want to send to the server.
- Set up the request options, including the URL, method (POST), and any necessary headers.
- Trigger the request, and the plugin will serialize your data into JSON for transmission.
- Handle the response and extract any relevant information.

When dealing with arrays of data, the Request Plugin simplifies the process. Here's what you need to do:

- Configure your data structure to match the API response, including arrays of structs.
- Make the request, specifying the URL and method.
- After the request is complete, you can work with the array data as needed.

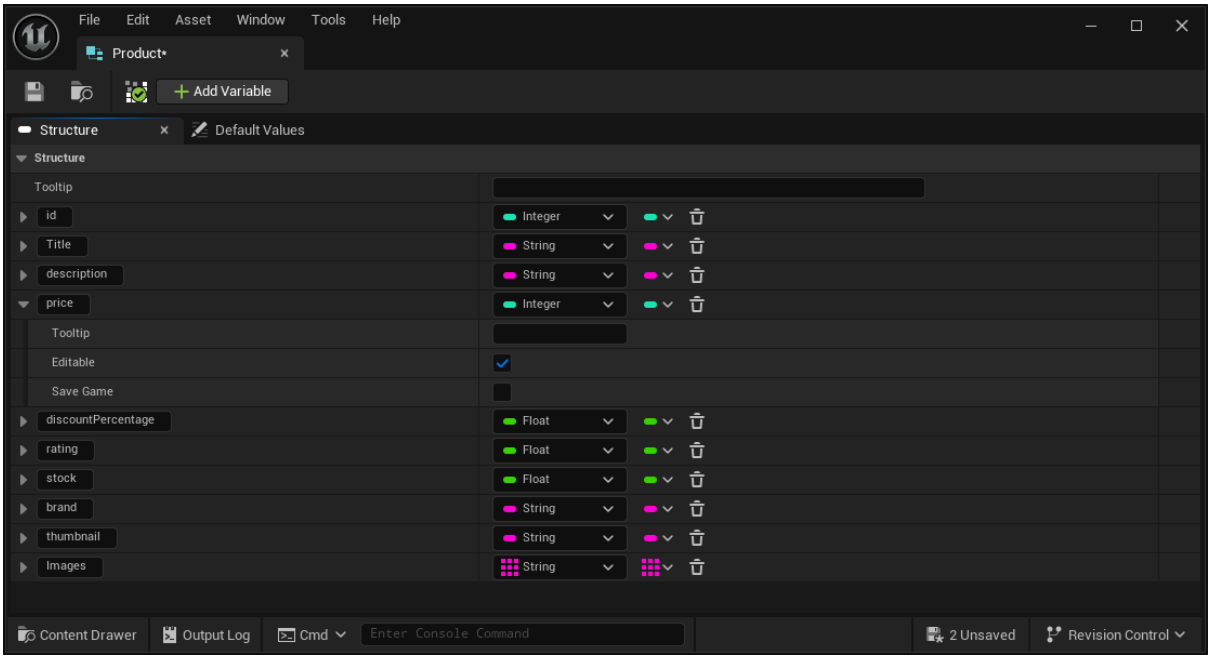
Create A Structure that matches your API for instance:

GET Example

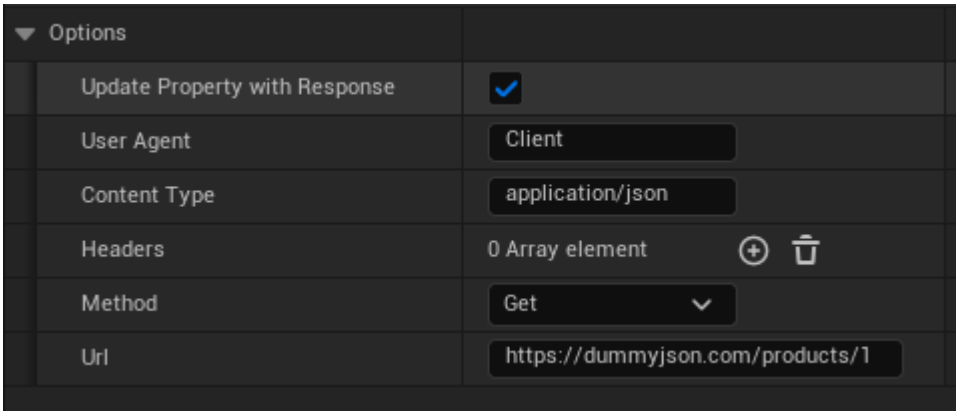
Product json from :https://dummyjson.com/products/1


```
{
    "brand": "Apple",
    "category": "smartphones",
    "description": "An apple mobile which is nothing like apple",
    "discountPercentage": 12.96,
    "id": 1,
    "images": [
        "https://i.dummyjson.com/data/products/1/1.jpg",
        "https://i.dummyjson.com/data/products/1/2.jpg",
        "https://i.dummyjson.com/data/products/1/3.jpg",
        "https://i.dummyjson.com/data/products/1/4.jpg",
        "https://i.dummyjson.com/data/products/1/thumbnail.jpg"
    ],
    "price": 549,
    "rating": 4.69,
    "stock": 94,
    "thumbnail":
"https://i.dummyjson.com/data/products/1/thumbnail.jpg",
    "title": "iPhone 9"
}
```


Has the structure you should define a struct that matches its structure. So matching structure defined as shown in the image at bellow



Prepare an options structure and set its User Agent & Content type respective to your API services. Also do not forget to set url & method as https://dummyjson.com/products/1 & GET

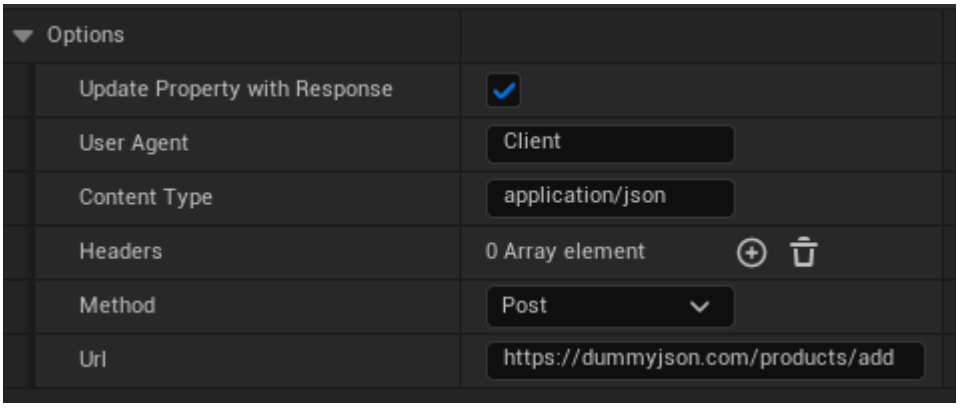


Fire it up to the server with any preferred event trigger or in your logic. Structure Bounded Request function provides a latent/Async action which is only triggered response received from server. So In Completed exec you can print or use provided data in structure. Product Structure's values filled up from response automatically.

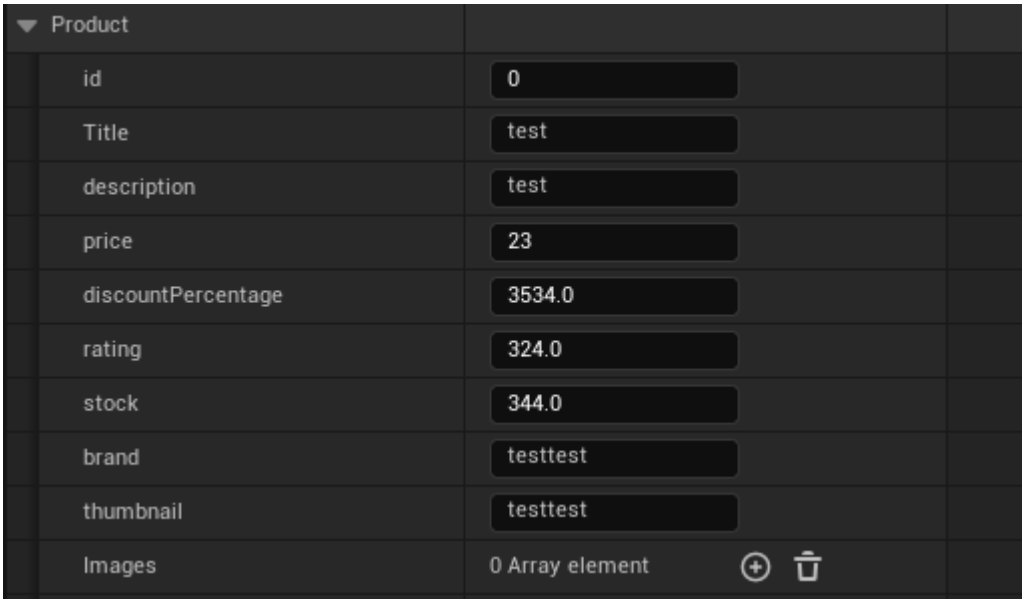
Also you can define extra properties in Product structure; it fills values that only respond json matched values.

POST Example

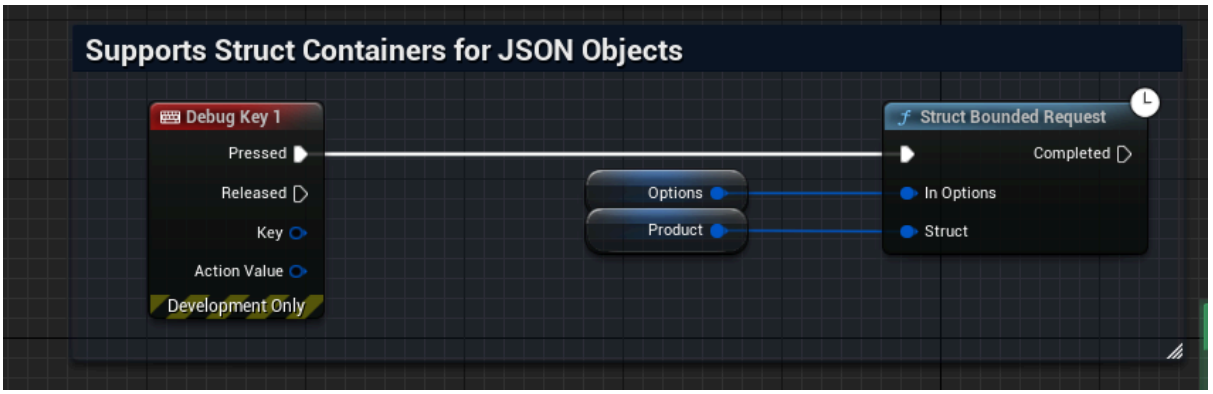
On posting some JSON values to the server it's also straightforward as all the steps are the same on blueprint. You just need to set up the structure values as you need. The plugin automatically serializes them to a json when construction of the request.
For instance: https://dummyjson.com/products/add endpoint expecting a product json from us.
Prepare the options structure as shown bellow.



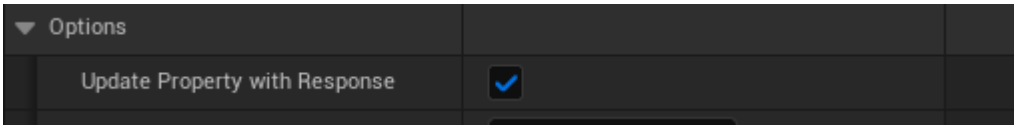
Prepare the product structure to send to the server. In our structure properties a value does not need to be here “ID”. In a properly setted up API endpoint this does not cause any issues.



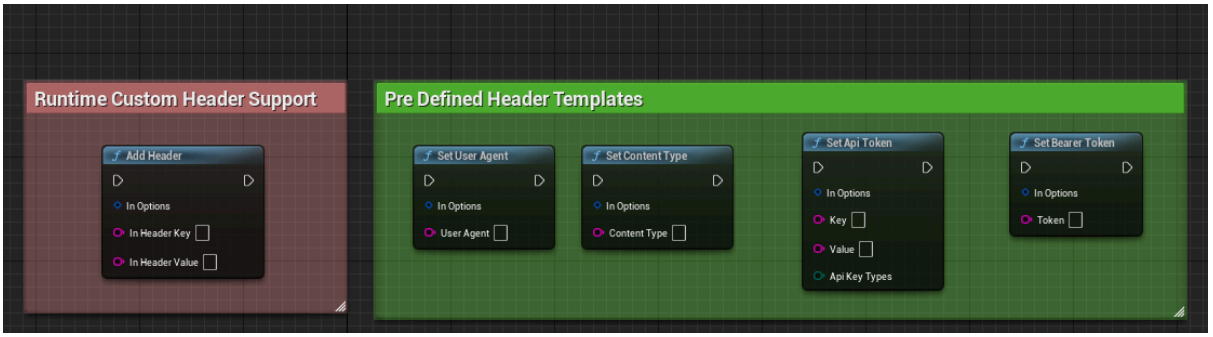
Trigger the request with a trigger.



In post method some well designed API's responses a json that contains id or any other auto created properties. So the plugin also detects this kind of property and sets our Structures matched property values with returning values of response. So in this case products/add endpoint sends us an id for the added product.



In options if you checked up the "Update Property with Response" response received json values automatically setted in matched properties.



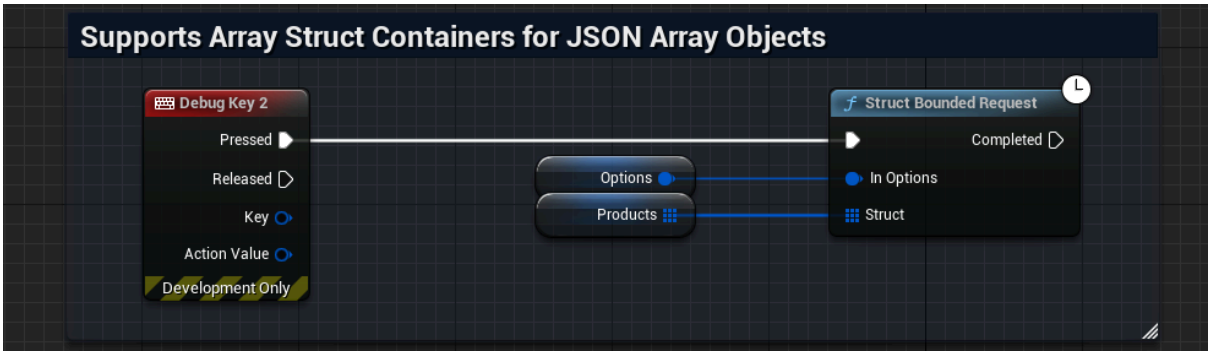
You could also define the always used headers and also you can add the headers in runtime, the plugin also supports setting an api token & bearer token

Get Array Example

The provided url :https://dummyjson.com/products
Its returns a Json object named Products and inside of the products object contains Product Array
Normally we need to define a structure that contains an array named products and typed as Product Struct. But it's inconvenient and also unnecessary. So with this plugin you also set the Struct container with an array of structs(TArray<Product>) But this variable has to be named the same as the json object that contains an array inside of it.

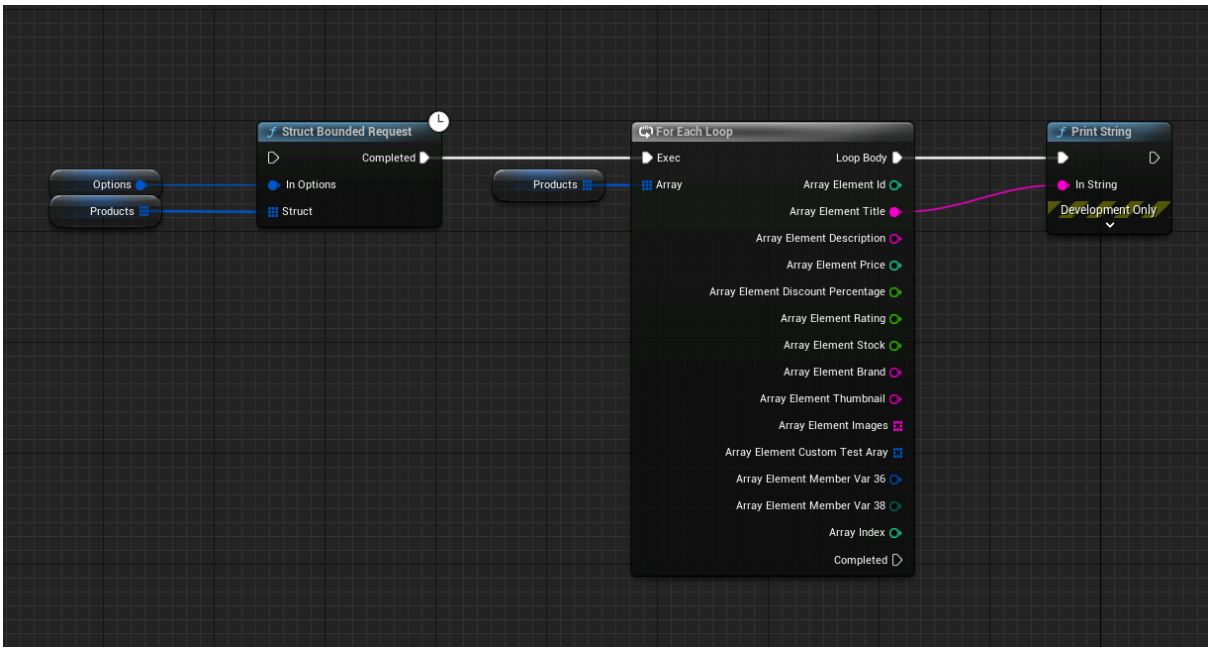

```
    "limit": 30,
    "products": [
        {
            "brand": "Apple",
            "category": "smartphones",
            "description": "An apple mobile which is nothing like
apple",
            "discountPercentage": 12.96,
            "id": 1,
            "images": [
                "https://i.dummyjson.com/data/products/1/1.jpg",
                "https://i.dummyjson.com/data/products/1/2.jpg",
                "https://i.dummyjson.com/data/products/1/3.jpg",
                "https://i.dummyjson.com/data/products/1/4.jpg",
                "https://i.dummyjson.com/data/products/1/thumbnail.jpg"
            ],
            "price": 549,
            "rating": 4.69,
            "stock": 94,
            "thumbnail":
"https://i.dummyjson.com/data/products/1/thumbnail.jpg",
            "title": "iPhone 9"
        },
      …
      …
      …
      ]}
```


So in this example The json object named as “products” so we need to be ensure our variable has the same name



Send the request when completed action is triggered. You have Product arrays you could use whenever you need.

For example print the titles on screen.



Important features

You could also define an enum for a json string value so for example when preparing our Products structure if we defined “Brand” as Enum instead of String. Plugin automatically matches given Json with user defined enum value and sets this value on provided structure.

Plugin supports:

- Integer Properties
- Float Properties
- Bool Properties
- Name Properties
- String Properties
- Text Properties
- Array Properties
- Enum Properties
- Inner Struct Properties that contains previous Primitive Properties

All of the Functions