Thank you for purchasing this asset. I hope you enjoy using it.

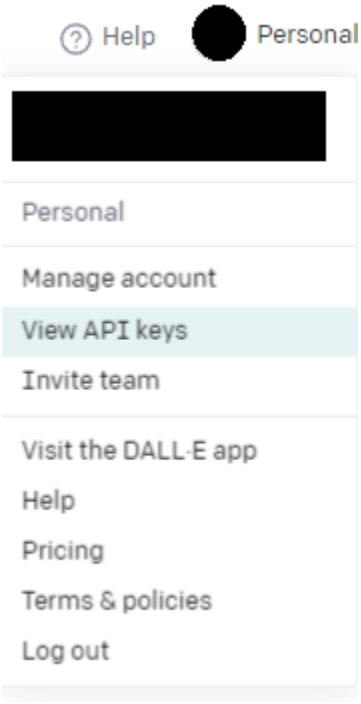
Contact my here, for bug reports, suggestions, or feedback.

Don't forget to leave a review if you enjoy using Text generation toolkit, this helps me a lot

Table of Content

How to set up Open AI

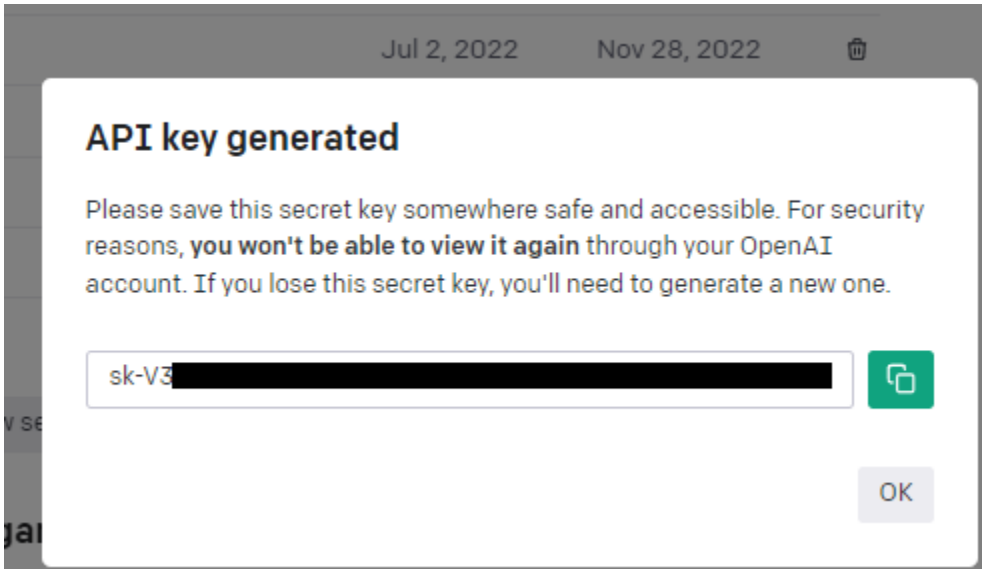
1. Create an OpenAI account by clicking on the "login" or “sign up” button on the top right of the page. As of writing this open AI offers 3 months of credit to test out their services.



2. Once you have an account, and are logged in, click on the "personal" button in the top right corner of the webpage.

3. From the dropdown menu, select "View API keys".

4. To create a new API key, click on "create new secret key. Note: You will only be able to see the secret key once.



5. Copy the API key that is generated.

6. Now open your unity project with the AI Text and Voice generation toolkit installed.

7. Find the scriptable object named "**OpenAIKey**" located under "Plugins/TextAndVoiceGenerationToolkit/OpenAI/Editor"

8. Past your API key that was generated in the Open AI Key slot in the scriptable object.

How to use Open AI In Editor

This is generalized so it can work with the following Open AI Scripts "**Call Chat-GPT**", "**Call GPT-3**", "**Chat-GPT AI**", or "**GPT-3 AI**". if you want to learn how to use the "**Reworder**" or the "**Get Sentiment**" click the there names.

1. Create a new GameObject.

2. Click "**Add Component**". Remove everything from the search bar. Then click "**Text & Voice Generation Toolkit**", then click "**OpenAI**", then click "**Call Chat-GPT**", "**Call GPT-3**", "**Chat-GPT AI**", or "**GPT-3 AI**" IMPORTANT: the "**GPT-3 AI**", and the "**Call GPT-3**" are going to be taken down on January 4, 2023.

3. Next, tell the AI what you want it to be in the "**What The Ai Is**" input box.

4. Ask the AI a question by typing the question in the "**Question**" box.

5. Then if you have any more instructions you can put them in the "**Post Prompt**".

6. If you would like the response to be shorter you can lower the "**Max Tokens**". Each token is roughly 4 characters. And each token will cost you different amounts depending on the model you're using. More Info HERE.

7. If you are using "**Chat-GPT AI**" or "**GPT-3 AI**", and if you would like the AI to have memory of the conversation then make sure the "**Use Memory**" checkbox is checked. Then click reset AI at the top of the script.

8. If you are using "**Call Chat-GPT**" or "**Call GPT-3**" you can change the "**Number Of Responses**" variable to change how many responses it will give you.

9. For more info on what each variable does you can go HERE.

10. Click "**Generate Response**" when you are ready to get the response form the AI.

How to use Call Chat-GPT In Editor

This script was made with the intention of using this to have Chat-GPT help you with brainstorming. It can be used for other things like NPC, but I would recommend using Chat-GPT AI for that.

1. Follow these instructions.

How to use Chat-GPT AI In Editor

This script is meant to be used in games. It can still be used for helping you brainstorm your game. To use Chat-GPT for in editor brian storming I would recommend using Call Chat-GPT.

1. Follow these instructions.

How to use GPT-3 AI In Editor

IMPORTANT: This model will be taken down January 4, 2023.
This script was initially built to be used for in-game purposes. It can still be used for helping you brainstorm your game, but I would recommend you using something like Call GPT-3 for that.

1. Follow these instructions.

2. You can give the AI a name in the "**Bot Name**" variable. This is really helpful for the "**Ada**", and "**Babbage**" models.

3. If you are using the "**Curie**" or "**Davinci**" models you can turn on the memory optimizer which will summarize the conversation so it does not cost you too much.

How to use Call GPT-3 In Editor

IMPORTANT: This model will be taken down January 4, 2023.
This script was built to have GPT-3 help you with brainstorming. While it is possible to have it be used in game for something like an NPC, I would recommend you using GPT-3 AI for that.

1. Follow these instructions.

How to use Reworder In Editor

This script was made to be used for both In Editor brainstorming, and for in game use. This script will reword something to make it sound like whatever you want it to.

1. Create a new GameObject
2. Add the "**Reworder**" Script to it.
3. Tell the AI what you want it to sound like in the "**Make Speech Sound Like**" input box.
4. Tell the AI what you want it to reword in the "**Reword The Following**" input box
5. If you would like the AI to keep specific words then put those words that you want it to keep into the "**Words To Keep**" input box. Make sure to separate each word with a coma.
6. If you want it to give you multiple variations then you can check the "**Give Multiple Response**" check box, then change the number of responses. NOTE: The AI sometimes likes to give the same response multiple times.
7. For more info on what each of the variables do go HERE.

How to use Get Sentiment In Editor

This script was built to check to see how aggressive the player is. -2 is very aggressive, and 2 is not aggressive at all.

1. Create a new Gameobject

2. Put what you want the AI to check into the "**Question**" input box.

3. Click Generate Response To get the sentiment.

How to use Open AI In-Game

This is generalized. so you should not need to change much to make it work with any one of the following scripts "**Call Chat-GPT**", "**Call GPT-3**", "**Chat-GPT AI**", or "**GPT-3 AI**". if you want to learn how to use the "**Reworder**" or the "**Get Sentiment**" click the there names.
IMPORTANT: the "**GPT-3 AI**", and the "**Call GPT-3**" are going to be taken down on January 4, 2023.

1. I will be using the "**Chat-GPT AI**" script. Follow along with any of the script labeled above, but first see what you need to change by clicking on the name.

2. Follow these instructions.

3. Create a new C# Script. This script will be responsible for calling the Open AI script whenever you would like it to. Attach it to the new GameObject, and open the script up. NOTE: Visual Studios Code sometimes has problems detecting the "**OpenAI.DLL**" file.

4. Get rid of the update function. Because you will not need it. Also it is not a good idea to call Open AI every frame.

5. Add the following Namespace at the top of your script.
```
using TheAshBot.Assets.TextAndVoiceGenerationToolkit.OpenAI.Core;
using TheAshBot.Assets.TextAndVoiceGenerationToolkit.OpenAI.MonoBehaviours;
```

6. In the start function you will need to Get the Open AI component.
```
ChatGPT chatGPT = GetComponent<ChatGPT>();
```

7. You can change the variable through code just like you can in the inspector.
```
chatGPT.maxTokens = 50;
```

8. Now you need to make the GPT call.

```
await chatGPT.MakeGTPCall("Hello!");
```

The "**await**" keyword makes the function stop and wait for the call to finish. If you do not have the "**await**" it will freeze the game. The string in the parenthesis is the question sent to the AI.

9. To be able to use the "**await**" keyword you have to put the keyword "**async**" right before the void.

```
private async void Start()
```

10. Now you need to get the returned value from the function. The function will return a custom class that has all the info that Open AI sends back.

```
OpenAIJsons.ChatResponse chatResponse = await chatGPT.MakeGTPCall("Hello!");
```

11. Now you will need to get the string response from all the data that Open AI sends. This might be different if you are not using the "**Chat-GPT AI**" script. Look at the link for your Open AI script ^^^

```
chatResponse.choices[0].message.content;
```

12. Now log the response as a string.

```
Debug.Log(chatResponse.choices[0].message.content);
```

13. Something that this plugin has that other plugins do not is the ability to get part of the response from Open AI while Open AI is sending more of the response. To use this functionality put this line of code in the start function before you Make a GPT Call.

```
chatGPT.OnPartResponseRecived += ChatGPT_OnPartResponseRecived;
OpenAIJsons.ChatResponse chatResponse = await chatGPT.MakeGTPCall("Hello!");
```

And add another function.

```
private void ChatGPT_OnPartResponseRecived()
{
    // This function will be called whenever more info comes in from Open AI
}
```

14. In the end your script should look something like this.

```
private async void Start()
{
    ChatGPT chatGPT = GetComponent<ChatGPT>();

    chatGPT.maxTokens = 50;

    chatGPT.OnPartResponseRecived += ChatGPT_OnPartResponseRecived;
    OpenAIJsons.ChatResponse chatResponse = await chatGPT.MakeGTPCall("Hello!");

    Debug.Log(chatResponse.choices[0].message.content);
}

private void ChatGPT_OnPartResponseRecived()
{
    // This function will be called whenever more info comes in from Open AI
}
```

How to use GPT-3 AI In-Game

IMPORTANT: This model will be taken down January 4, 2023.
The GPT-3 AI script is built for in-game use. Whereas Call GPT-3 was built for brainstorming.

1. Follow these instructions. But change the following.

2. Add this to the top of your script, where the namespaces are.
```
using GPT3AI = TheAshBot.Assets.TextAndVoiceGenerationToolkit.OpenAI.MonoBehaviours.GPT3AI;
```

3. Change all references of "**ChatGPT**" to "**GPT3AI**"

4. Change all references of "**OpenAIJsons.ChatResponse**" to "**OpenAIJsons.Response**"

5. To get the response as a string use the following code.
```
response.choices[0].text
```

How to use Chat-GPT AI In-Game

The GPT-3 AI script is built for in-game use. Whereas Call GPT-3 was built for brainstorming.

1. Follow these instructions.

How to use Call Chat-GPT In-Game

The GPT-3 AI script is built for in-game use. Whereas Call GPT-3 was built for brainstorming.

1. Follow these instructions, but Change the following.

2. Add this to the top of your script, where the namespaces are.

```
using CallChatGPT = TheAshBot.Assets.TextAndVoiceGenerationToolkit.OpenAI.MonoBehaviours.CallChatGPT;
```

3. Change all references of "**ChatGPT**" to "**CallChatGPT**"

How to use Call GPT-3 In-Game

IMPORTANT: This model will be taken down January 4, 2023.
The GPT-3 AI script is built for in-game use. Whereas Call GPT-3 was built for brainstorming.

1. Follow these instructions, but change the following.

2. Change all references of "**ChatGPT**" to "**CallGPT3**"

3. Change all references of "**OpenAIJsons.ChatResponse**" to "**OpenAIJsons.Response**"

4. To get the response as a string use the following code.

```
response.choices[0].text
```

How to use Open AI Core

This does NOT go into as much detail as the rest of the documentation. This Also is not very secure.

How to use Chat-GPT AI Core

How to use Call Chat-GPT Core

How to use GPT-3 AI Core

How to use Call GPT-3 Core

How to use Chat-GPT AI Core

This is the core of the Chat-GPT AI script. I would recommend you learn how to use the Chat-GPT AI script instead of this. The Chat-GPT AI script already has key security built in.

1. Create a new GameObject.

2. Make a new script and attach it to the GameObject. Then open it up. NOTE: Visual Studios Code sometimes has problems detecting the "**OpenAI.DLL**" file.

3. Get rid of the "**Update**" function.

4. Add this namespace

```
using TheAshBot.Assets.TextAndVoiceGenerationToolkit.OpenAI.Core;
```

5. In the start function you will need to create a new GPT3AI variable.

```
ChatGPTAI chatGPTAI = new ChatGPTAI();
```

6. Now you will need to put the following parameters into the parenthesis.

    a. "**useMemory**" is a bool. If true, the AI will have memory of the conversation.

    b. "**moderation**" is an enum. If set to true then an AI will check to make sure it is appropriate. Open AI requires this for all public applications.

    ```
    OpenAIEnums.Moderation.Moderation
    ```

    c. "**user**" is a string. It can help OpenAI to monitor and detect abuse.

    ```
    ChatGPTAI chatGPTAI = new ChatGPTAI(true, OpenAIEnums.Moderation.Moderation, "The Ash Bot");
    ```

7. Now input your Open AI API key. IMPORTANT: this is not secure. You can use Azure Key Vault to make it more secure!

```
chatGPTAI.OpenAiKey = "<YOUR OPEN AI KEY>";
```

8. Now you need to make the call to Open AI.

```
await chatGPTAI.CallOpenAI();
```

9. Because of the "**await**" you will now need to make the Start function an"**async**" function

```
private async void Start()
```

10. Now you will need the following parameters in the Open AI Function Call.
    a. "**model**" is an enum with different models.

    ```
    OpenAIEnums.ChatModels.gpt_35_turbo
    ```

b. “**whatTheAiIs**” is a string that tells the AI what you want it to be.

c. “**question**” is a string. It is the question that the AI will answer.

d. “**onPartOfResponseReceived**” is a delegate that will hold a function that will be called when Open AI sends part of the response back. Parameters.

   i. “**wholeResponse**” is a string with the whole response so far.

   ii. “**newPartOfResponse**” is a custom class that has all the info that Open AI sends with the new part of the response.

```
private void ChatGPTAI_OnNewPartOfResponseRecived(string wholeResponse,
    OpenAIJsons.ChatStreamResponse newPartOfResponse)
{
    Debug.Log("NewPartOfResponse: " + newPartOfResponse.choices[0].delta.content);
}
```

11. Now the function call should look like this.

```
await chatGPTAI.CallOpenAI(OpenAIEnums.ChatModels.gpt_35_turbo,
    "You are NPC", "Hello!", ChatGPTAI_OnNewPartOfResponseRecived);
```

12. Now you need to get the returned value from the function call.

```
OpenAIJsons.ChatResponse response =
```

13. Now you will need to get, and log the string response.

```
Debug.Log(response.choices[0].message.content);
```

14. Now the final function should look like this.

```
private async void Start()
{
    ChatGPTAI chatGPTAI = new ChatGPTAI(true, OpenAIEnums.Moderation.Moderation, "The Ash Bot");

    chatGPTAI.OpenAiKey = "<YOUR OPEN AI KEY>";

    OpenAIJsons.ChatResponse response = await chatGPTAI.CallOpenAI(OpenAIEnums.ChatModels.gpt_35_turbo,
        "You are NPC", "Hello!", ChatGPTAI_OnNewPartOfResponseRecived);

    Debug.Log(response.choices[0].message.content);
}

private void ChatGPTAI_OnNewPartOfResponseRecived(string wholeResponse,
    OpenAIJsons.ChatStreamResponse newPartOfResponse)
{
    Debug.Log("NewPartOfResponse: " + newPartOfResponse.choices[0].delta.content);
}
```

How to use Call Chat-GPT Core

1. Follow these Instructions, but change the following
2. Do not make the "**chatGPTAI**" variable.
3. Replace "**chatGPTAI**" with "**CallChatGPT**".
4. Add the "**moderation**", and "**user**" right after the prompt

```
await CallChatGPT.CallOpenAI(OpenAIEnums.ChatModels.gpt_35_turbo, "You are NPC", "Hello!",
    OpenAIEnums.Moderation.Moderation, "The Ash Bot", ChatGPTAI_OnNewPartOfResponseRecived);
```

5. Now the final function should look like this.

```
private async void Start()
{
      CallChatGPT.OpenAiKey = "<YOUR OPEN AI KEY>";

   OpenAIJsons.ChatResponse response = await CallChatGPT.CallOpenAI(OpenAIEnums.ChatModels.gpt_35_turbo,
      "You are NPC", "Hello!", OpenAIEnums.Moderation.Moderation, "The Ash Bot", ChatGPTAI_OnNewPartOfResponseRecived);

   Debug.Log(response.choices[0].message.content);
}

private void ChatGPTAI_OnNewPartOfResponseRecived(string wholeResponse,
   OpenAIJsons.ChatStreamResponse newPartOfResponse)
{
   Debug.Log("NewPartOfResponse: " + newPartOfResponse.choices[0].delta.content);
}
```

How to use GPT-3 AI Core

IMPORTANT: This model will be taken down January 4, 2023.
This is the core of the GPT-3 AI script. I would recommend you learn how to use the GPT-3 AI script instead of this. The GPT-3 AI script already has key security

built in.

1. Follow these Instructions, but change the following.

2. Replace “**ChatGPTAI**” with “**GPT3AI**”.

3. Replace “**chatGPTAI**” with “**gpt3AI**”.

4. When declaring the “**gpt3AI**” variable, add a string for the “**botName**” before “**useMemory**”, and add a bool for “**useMemoryOptimizer**”.

```
GPT3AI gpt3AI = new GPT3AI("AI", true, true, OpenAIEnums.Moderation.Moderation, "The Ash Bot");
```

5. Replace “**OpenAIJsons.ChatResponse**” with “**OpenAIJsons.Response**”.

6. Replace “**OpenAIEnums.ChatModels**” with “**OpenAIEnums.Models**”.

7. Replace “**response.choices[0].message.content**” with “**response.choices[0].text**”.

8. Replace “**OpenAIJsons.ChatStreamResponse**” with “**OpenAIJsons.StreamResponse**”.

9. Replace “**newPartOfResponse.choices[0].delta.content**” with “**newPartOfResponse.choices[0].text**”.

10. Now the final function should look like this.

```
private async void Start()
{
    GPT3AI gpt3AI = new GPT3AI("AI", true, true, OpenAIEnums.Moderation.Moderation, "The Ash Bot");

    gpt3AI.OpenAiKey = "<YOUR OPEN AI KEY>";

    OpenAIJsons.Response response = await gpt3AI.CallOpenAI(OpenAIEnums.Models.text_davinci_003,
        "You are NPC", "Hello!", ChatGPTAI_OnNewPartOfResponseRecived);

    Debug.Log(response.choices[0].text);
}

private void ChatGPTAI_OnNewPartOfResponseRecived(string wholeResponse,
    OpenAIJsons.StreamResponse newPartOfResponse)
{
    Debug.Log("NewPartOfResponse: " + newPartOfResponse.choices[0].text);
}
```

How to use Call GPT-3 Core

IMPORTANT: This model will be taken down January 4, 2023.

This is the core of the Call GPT-3 script. I would recommend you learn how to use the Call GPT-3 script instead of this. The Call GPT-3 already has key security built in.

1. Follow these Instructions, but change the following
2. Do not make the "**gpt3AI**" variable.
3. Replace "**gpt3AI**" with "**CallGPT3**".
4. Add the "**moderation**", and "**user**" right after the prompt

```
await CallGPT3.CallOpenAI(OpenAIEnums.Models.text_davinci_003, "You are NPC", "Hello!",
    OpenAIEnums.Moderation.Moderation, "The Ash Bot", ChatGPTAI_OnNewPartOfResponseRecived);
```

5. Now the final function should look like this.

```
private async void Start()
{
    CallGPT3.OpenAiKey = "<YOUR OPEN AI KEY>";

    OpenAIJsons.Response response = await CallGPT3.CallOpenAI(OpenAIEnums.Models.text_davinci_003, "You are NPC",
        "Hello!", OpenAIEnums.Moderation.Moderation, "The Ash Bot", ChatGPTAI_OnNewPartOfResponseRecived);

    Debug.Log(response.choices[0].text);
}

private void ChatGPTAI_OnNewPartOfResponseRecived(string wholeResponse,
    OpenAIJsons.StreamResponse newPartOfResponse)
{
    Debug.Log("NewPartOfResponse: " + newPartOfResponse.choices[0].text);
}
```

What Each Variable does

MaxTokens: is the max number of tokens that the AI will use in its response. More info about pricing.

Temperature: is how direct/factual and creative the AI is. Temperature is a value between 0 - 2. 0 is very factual and to the point. 2 is very creative.

TopP: is a value between 0, and 1. 0 repeats answers. 1 provides different answers.

FrequencyPenalty: is a value between -2, and 2. -2 is more likely to repeat words. 2 is less likely to repeat words.

PresencePenalty: is a value between -2, and 2. -2 is more likely to repeat topics. 2 is less likely to repeat topics.

User: is a unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse.

LogWebCallContent: if this is true then the json message that is sent to open AI will be logged onto the console.

Model: determines how good the response is and how much it costs per token. More info about the different Models.

WhatTheAiIs: tells the AI what you want it to be.

Question: is the question for the AI.

PostPrompt: is the final command to the AI.

BotName: is the name of the AI. This is really helpful for Ada, and Babbage.

UseMemory: will make the AI have memory of the whole conversation.

UseMemoryOptimizer: will shorten the conversation so it is not as expensive to call. This will result in slightly worse answers, on everything but Davinci.

Moderation: if set to moderation then an AI will check to see if it is appropriate. IMPORTANT: Open AI can take you down if there is anything inappropriate in a public application.

TimeTillNextCall: is the minimum amount of time that will pass before you can call open AI again. This makes sure you do not accidentally call it every frame.

OpenAiKey: is the key that open AI uses to make sure you are calling it, and not someone else. If you do not know how to get one, go HERE.

DefaultResponse: is the default response that the AI will give if you get an error when making the call to OpenAI, or moderation determines that it is not appropriate.

CanSend: is used to prevent you from calling open AI every frame.

Conversation: is the whole conversation the user has had with the ai as a single string.

ConversationList: is the whole conversation the user has had with the ai broken us into a list of strings.

OptimizedConversation: is the summarized conversation the user has had with the ai as a single string.

OptimizedConversationList: is the summarized conversation the user has had with the ai broken us into a list of strings.

Response: is a custom class that holds all the info of the response that the Complations-AI gives.

ModerationResponse: is a custom class that holds the info of the response that the Moderation-AI gives.

ChatResponse: is a custom class that holds all the info of the response that the Chat-AI gives.

Models: is an enum with all the AI's for Complations endpoint. To learn more please go to the documentation.

ChatModels: is an enum with all the AI's for Chat endpoint. To learn more please go to the documentation

NumberOfResponses: is the number of responses that the AI will give you. Note: sometimes the Ai might give you the same response.

UseSentiment: if true then the AI will automatically figure out how aggressive what the user sent is.

Open AI Jsons

This has all the classes that have the info sent back from Open AI.

This class contains the following classes.

1. "**Response**" is the response given back when you call Open AI with the GPT-3 scripts. Here is a sample response from open AI. NOTE: The class in the plugin does not contain the "**object**" variable.


```
{
  "warning": "This model version is deprecated. Migrate before January 4, 2024 to avoid disruption of service. Learn more https://platform.openai.com/docs/deprecations",
  "id": "cmpl-7qkIxenqGGqcknFhbg8d4JZXCIuD9",
  "object": "text_completion",
  "created": 1692805407,
  "model": "text-davinci-003",
  "choices": [
    {
      "text": "Hello!",
      "index": 0,
      "logprobs": null,
      "finish_reason": "stop"
    }
  ],
  "usage": {
    "prompt_tokens": 4,
    "completion_tokens": 4,
    "total_tokens": 8
  }
}
```

2. “**ModerationResponse**” is the response given back from when you call the moderation model. Here is a sample response from open AI.NOTE: The class in the plugin does not contain the “**results**” variable.


```
{
 "id": "modr-XXXXX",
 "model": "text-moderation-005",
 "results": [
  {
   "flagged": true,
   "categories": {
    "sexual": false,
    "hate": false,
    "harassment": false,
    "self-harm": false,
    "sexual/minors": false,
    "hate/threatening": false,
    "violence/graphic": false,
    "self-harm/intent": false,
    "self-harm/instructions": false,
    "harassment/threatening": true,
    "violence": true
   },
   "category_scores": {
    "sexual": 1.2282071e-06,
    "hate": 0.010696256,
    "harassment": 0.29842457,
    "self-harm": 1.5236925e-08,
    "sexual/minors": 5.7246268e-08,
    "hate/threatening": 0.0060676364,
    "violence/graphic": 4.435014e-06,
    "self-harm/intent": 8.098441e-10,
    "self-harm/instructions": 2.8498655e-11,
    "harassment/threatening": 0.63055265,
    "violence": 0.99011886
   }
  }
 ]
}
```

3. “**ChatResponse**” is the response given back when you call Open AI with the Chat-GPT scripts. Here is a sample response from open AI.NOTE: The class in the plugin does not contain the “**object**” variable.


```
{
  "id": "chatcmpl-123",
  "object": "chat.completion",
  "created": 1677652288,
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "\n\nHello there, how may I assist you today?"
      },
      "finish_reason": "stop"
    }
  ],
  "usage": {
    "prompt_tokens": 9,
    "completion_tokens": 12,
    "total_tokens": 21
  }
}
```

4. “**ChatStreamResponse**” is the new part of the response Open AI Sends when you call the Chat-GPT scripts. Here is a sample response from open AI. NOTE: The class in the plugin does not contain the “**object**” variables.


```
{
 "data": {
  "id": "chatcmpl-7hfKNmjGzvd6Is6sRv97TegYoHna2",
  "object": "chat.completion.chunk",
  "created": 1690641323,
  "model": "gpt-3.5-turbo-0613",
  "choices": [
   {
    "index": 0,
    "delta": {
     "role": "assistant",
     "content": ""
    },
    "finish_reason": null
   }
  ]
 },
 "data": {
  "id": "chatcmpl-7hfKNmjGzvd6Is6sRv97TegYoHna2",
  "object": "chat.completion.chunk",
  "created": 1690641323,
  "model": "gpt-3.5-turbo-0613",
  "choices": [
   {
    "index": 0,
    "delta": { "content": "Hello" },
    "finish_reason": null
   }
  ]
 },
 "data": {
  "id": "chatcmpl-7hfKNmjGzvd6Is6sRv97TegYoHna2",
  "object": "chat.completion.chunk",
  "created": 1690641323,
  "model": "gpt-3.5-turbo-0613",
  "choices": [
   {
    "index": 0,
    "delta": { "content": "!" },
    "finish_reason": null
   }
  ]
 },
 "data": [ "DONE" ]
}
```

5. “**StreamResponse**” is the new part of the response Open AI Sends when you call the GPT-3 scripts. Here is a sample response from open AI. NOTE: The class in the plugin does not contain the “**object**” variables.


```
{
 "data": {
  "warning": "This model version is deprecated. Migrate before January 4, 2024 to avoid disruption of service. Learn more https://platform.openai.com/docs/deprecations",
  "id": "cmpl-7vRyA4GLBZ3svpe3UVQbXOdq1LTKo",
  "object": "text_completion",
  "created": 1693926566,
  "choices": [
   {
    "text": "Hello",
    "index": 0,
    "logprobs": null,
    "finish_reason": null
   }
  ],
  "model": "text-davinci-003"
 },
 "data": {
  "warning": "This model version is deprecated. Migrate before January 4, 2024 to avoid disruption of service. Learn more https://platform.openai.com/docs/deprecations",
  "id": "cmpl-7vRyA4GLBZ3svpe3UVQbXOdq1LTKo",
  "object": "text_completion",
  "created": 1693926566,
  "choices": [
   {
    "text": "!",
    "index": 0,
    "logprobs": null,
    "finish_reason": null
   }
  ],
  "model": "text-davinci-003"
 },
 "data": {
  "warning": "This model version is deprecated. Migrate before January 4, 2024 to avoid disruption of service. Learn more https://platform.openai.com/docs/deprecations",
  "id": "cmpl-7vRyA4GLBZ3svpe3UVQbXOdq1LTKo",
  "object": "text_completion",
  "created": 1693926566,
  "choices": [
   {
    "text": "",
    "index": 0,
    "logprobs": null,
    "finish_reason": "stop"
   }
  ],
  "model": "text-davinci-003"
 },
 "data": [ "DONE" ]
}
```