



# Innovar OpenAI Plugin

# Introduction

Welcome to the OpenAI plugin documentation for Mirth Connect. This plugin is designed to use OpenAI ChatGPT models to provide real-time assistance to JavaScript code development on Mirth Connect. ChatGPT can help to clarify concepts, suggest solutions, provide explanations, and offer guidance on resolving bugs and errors in the code. ChatGPT can also generate the HL7 messages such as ADT, ORU, ORM for testing purposes on Mirth Connect.

## Why OpenAI?

OpenAI ChatGPT can help healthcare professionals quickly identify and troubleshoot issues in their JavaScript code, allowing them to address problems faster and more effectively. Ensuring that healthcare software applications are running smoothly and free of bugs is crucial for providing high-quality patient care. Using ChatGPT for debugging can help healthcare professionals deliver reliable and efficient services to their patients.

## Key Features

**JavaScript code debugger:** Innovar OpenAI plugin can help healthcare software engineers quickly identify and troubleshoot issues in their JavaScript code with powerful AI models. The models can understand complex code structures and provide suggestions for debugging, saving time and effort in the development process.

**Message generator:** Innovar OpenAI plugin allows Mirth Connect engine to connect to ChatGPT and quickly generate standardized HL7 messages based on user input, saving time and effort for healthcare professionals who need specific messages to test their interface. This can streamline communication processes and improve overall efficiency in healthcare delivery.

## Getting Started

Before you dive into the documentation, make sure to review the installation prerequisites and check compatibility with your existing Mirth Connect setup. The subsequent sections will guide you through the configuration, usage of the OpenAI plugin to improve the development experience of the interfaces on Mirth Connect.

Let's get started!

# Installation

If you are using Mirth Connect packaged by Innovar Healthcare from the AWS Marketplace, the extension should be pre-installed on “Advanced with SSL” and “Advanced with SSL Autoscaling” versions starting with version 4.5 and greater.

If, for some reason, you need to reinstall or update the plugin, you can do this from the Mirth Connect Administrator application. While logged into Mirth Connect Administrator, click on “Extensions”, then at the bottom of the screen, click “Browse”. A new window will pop up to allow you to browse for your plugin zip file on your local machine. Browse to the appropriate zip file and click “Open”. Once back on the Extensions screen, your file path should be filled in. Click “Install” to upload the file. Once completed, you will need to restart the Mirth Connect Service on the remote server.

## OpenAI Account/API Setup

### Create your OpenAI api key

If you do not already have a key, please create/login [OpenAI](#) to create your api key.

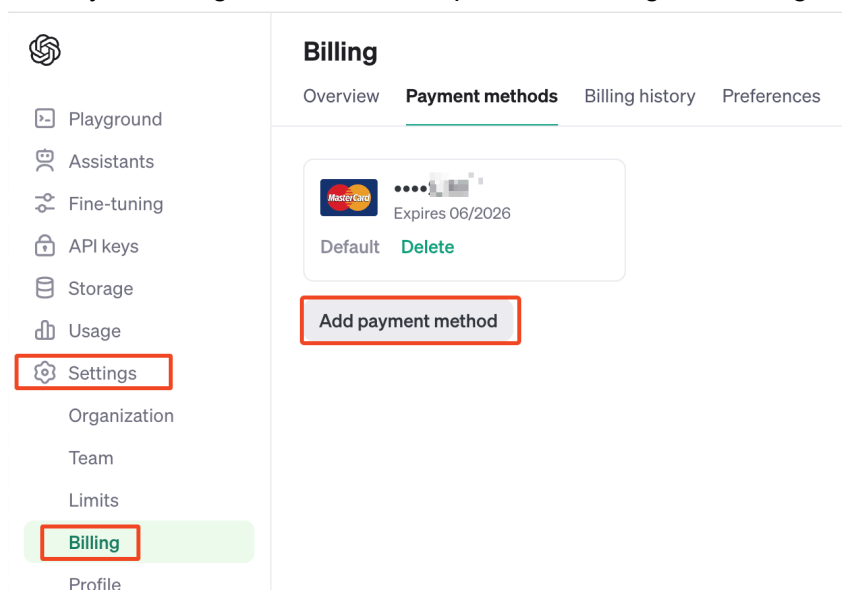
1. In the OpenAI console, got to “API keys” > “Create new secret key”

NAME	SECRET KEY	TRACKING	CREATED
Innovar openai key	[REDACTED]	Enabled	Mar 29, 2024

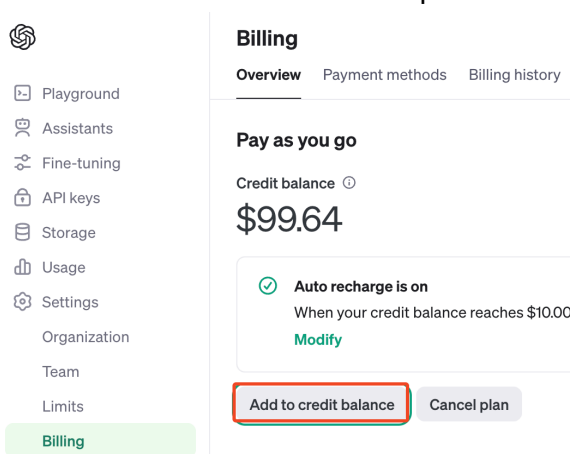
2. Give a name for your key and click “create secret key”.

Please make sure you save your api key at a safe location.

3. Add your billing information on OpenAI. Please go to Settings > Billing > Add payment method



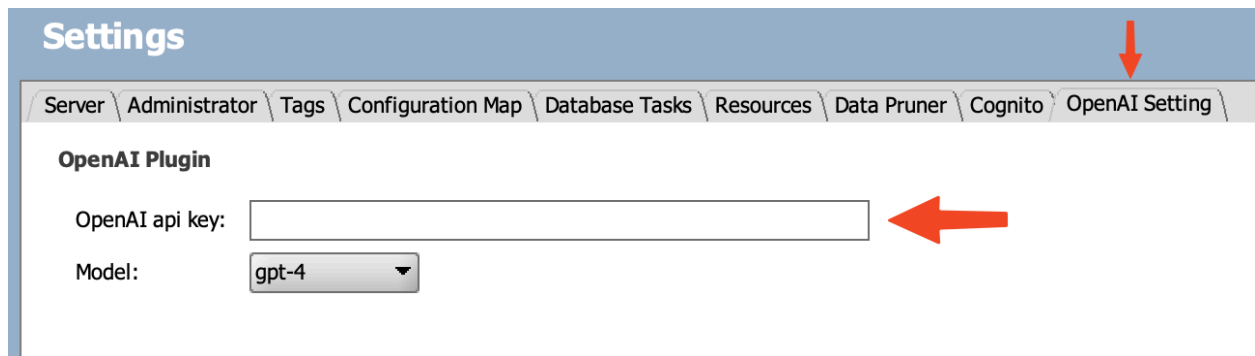
4. Click "Add to credit balance" to purchase the credits.



Now your OpenAI api key is ready!

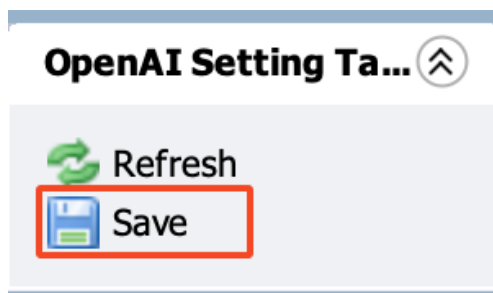
# Plugin Setup

In the Mirth Connect Administrator window, click on Settings. You will see a new tab for “OpenAI Setting” and enter your OpenAI api key.



The screenshot shows the 'Settings' window in the Mirth Connect Administrator. The 'OpenAI Setting' tab is selected, indicated by a red arrow pointing to it. Below the tab, the 'OpenAI Plugin' section contains a text input field for the 'OpenAI api key' and a dropdown menu for the 'Model' set to 'gpt-4'. A red arrow points to the 'OpenAI api key' input field.

And click “Save” on the left.

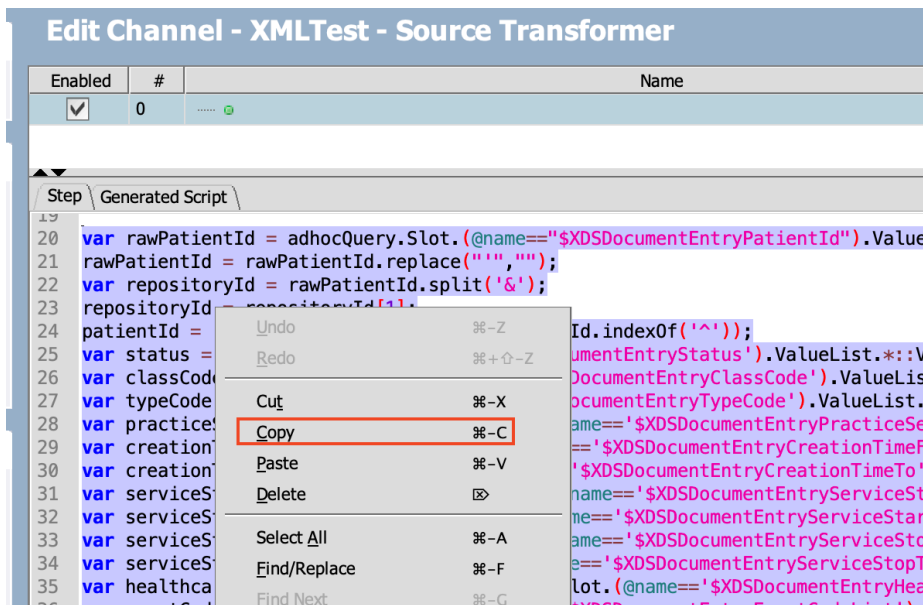


The screenshot shows a close-up of the 'OpenAI Setting Ta...' window. It features a 'Refresh' button with a circular arrow icon and a 'Save' button with a floppy disk icon. The 'Save' button is highlighted with a red rectangular border.

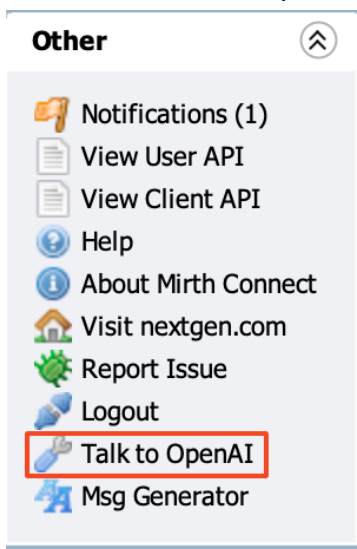
# Plugin Usage

## Javascript code debugger

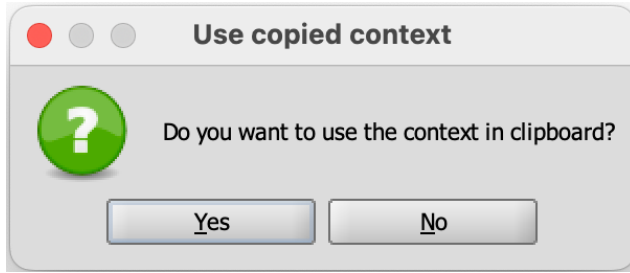
1. Highlight and copy the code you want reviewed to your system clipboard.



2. Click the "Talk to OpenAI" option on the Other task panel.

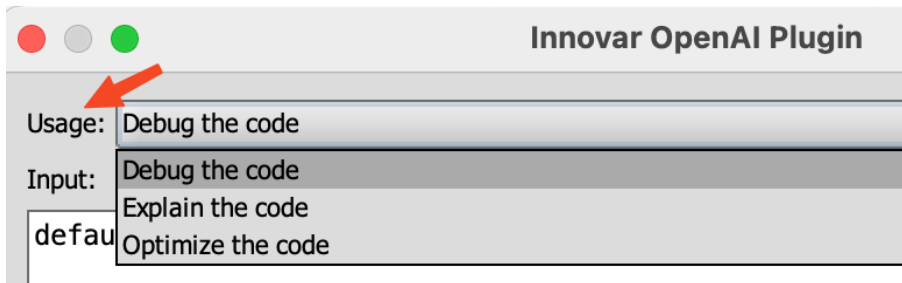


3. You can copy a section of JavaScript code in transformer step, code template or Global scripts



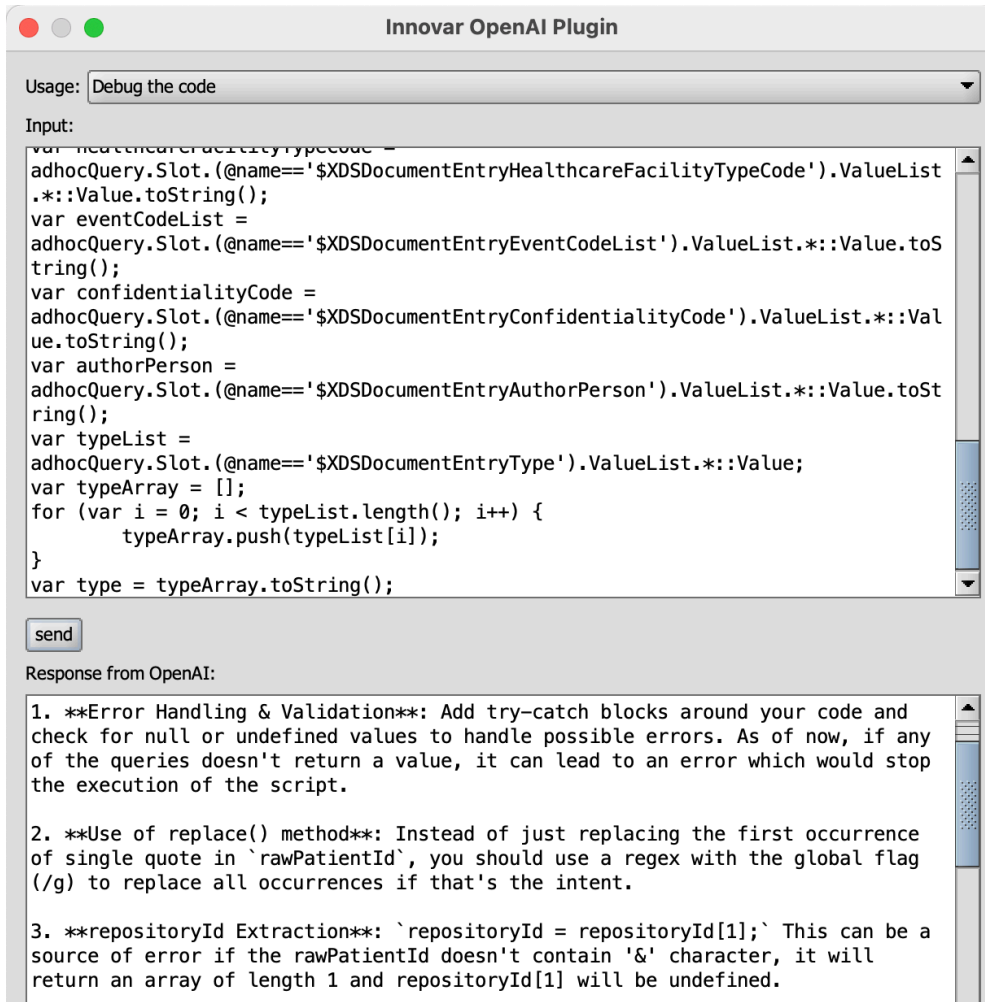
4. Click "send"

5. There are different usages in the Dialog



- Debug the code: requests OpenAI to review the code for errors and help correct it
- Explain the code: requests OpenAI to explain what the copied code block is doing in user-friendly terms. It may also add comments to the code.
- Optimize the code: requests OpenAI to review the code to see if it can be optimized, cleaned up, or otherwise made better/more efficient

## 6. It might take a few moments to get the response from OpenAI.



The screenshot shows a window titled "Innovar OpenAI Plugin". At the top, there is a "Usage:" dropdown menu set to "Debug the code". Below this is an "Input:" section containing a code editor with the following JavaScript code:

```
var healthcareFacilityTypeCode =
adhocQuery.Slot.@name=='$XDSDocumentEntryHealthcareFacilityTypeCode'.ValueList
.::*Value.toString();
var eventCodeList =
adhocQuery.Slot.@name=='$XDSDocumentEntryEventCodeList'.ValueList.::*Value.toS
tring();
var confidentialityCode =
adhocQuery.Slot.@name=='$XDSDocumentEntryConfidentialityCode'.ValueList.::*Val
ue.toString();
var authorPerson =
adhocQuery.Slot.@name=='$XDSDocumentEntryAuthorPerson'.ValueList.::*Value.toSt
ring();
var typeList =
adhocQuery.Slot.@name=='$XDSDocumentEntryType'.ValueList.::*Value;
var typeArray = [];
for (var i = 0; i < typeList.length(); i++) {
    typeArray.push(typeList[i]);
}
var type = typeArray.toString();
```

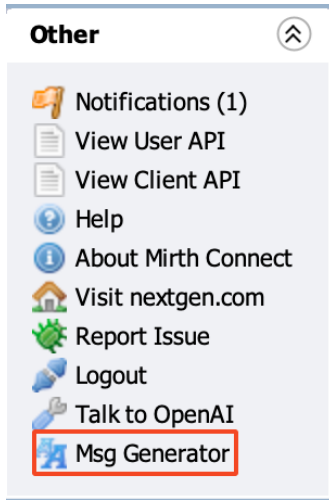
Below the code editor is a "send" button. Underneath that is a "Response from OpenAI:" section containing the following text:

- Error Handling & Validation**: Add try-catch blocks around your code and check for null or undefined values to handle possible errors. As of now, if any of the queries doesn't return a value, it can lead to an error which would stop the execution of the script.
- Use of replace() method**: Instead of just replacing the first occurrence of single quote in `rawPatientId`, you should use a regex with the global flag (/g) to replace all occurrences if that's the intent.
- repositoryId Extraction**: `repositoryId = repositoryId[1];` This can be a source of error if the rawPatientId doesn't contain '&' character, it will return an array of length 1 and repositoryId[1] will be undefined.

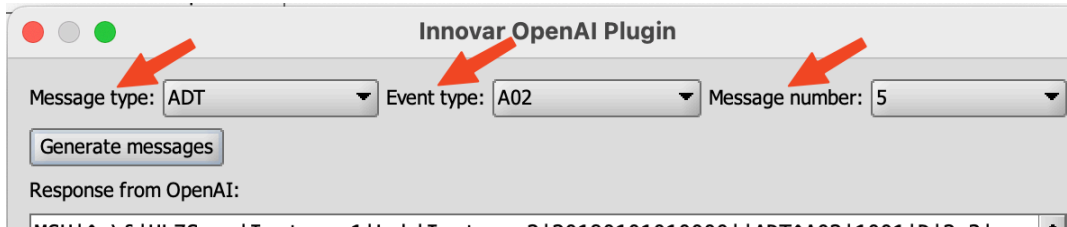


# Message Generator

1. Click the “Msg Generator” option on the Other task panel.



2. Please select the message type, event type, and message number



3. Click “Generate messages”. It might take a few moments to get the response from OpenAI. Here is an example response from OpenAI

