April 28, 2017

Adding an FAQ chatbot to your WordPress site using DialogFlow (API.AI)

One of the nice things about chatbots is that you can have them answer the Frequently Asked Questions (FAQs) from your website visitors. This has many benefits, and I will write more about them in a later post. For now, I will suppose you are already interested in adding an FAQ chatbot into your WordPress site.

> I have created a tool which will generate an FAQ chatbot ZIP file from a CSV file in a single click. You can read about it here.

I have written extensively on this blog about embedding an API.AI agent into your WordPress blog.

Step by step guide

Creating an FAQ Chatbot

Embedding an API.AI chatbot into your site

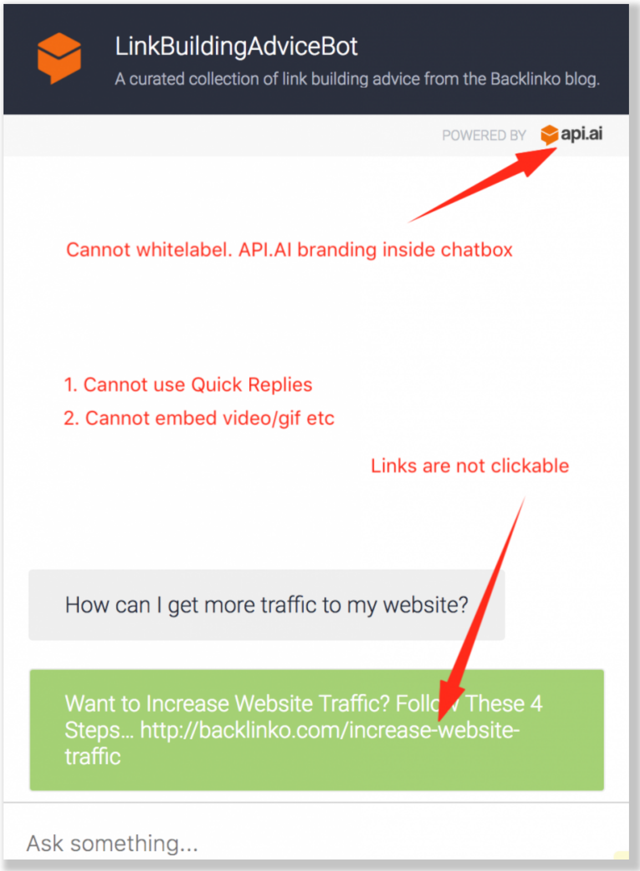
In addition, I have also created some example chatbots and embedded them into this site to show what is possible.

Botfolio

Current limitations

There is, however, a problem with the current implementation. Right now, the simplest and most straight-forward way to embed a chatbot into your website is to use the "web agent" integration already provided by API.AI.

There are many challenges associated with using the web agent provided by API.AI. And one of these is particularly challenging for an FAQ bot: no clickable hyperlinks!


LinkBuildingAdviceBot

A curated collection of link building advice from the Backlinko blog.

Cannot whitelabel. API.AI branding inside chatbox

1. Cannot use Quick Replies
2. Cannot embed video/gif etc

Links are not clickable

How can I get more traffic to my website?

Want to Increase Website Traffic? Follow These 4 Steps... http://backlinko.com/increase-website-traffic

When you use the built-in web agent which comes with API.AI, you face the following challenges:

1. The hyperlinks in the text are not clickable
2. You cannot whitelabel and use your own branding - you will have the "Powered by API.AI" link at the top
3. The web agent itself is more limited in what it can output, which means some of the features you could have on the other platforms are missing.
    1. No support for Quick Replies

2. No support for Audio/Video embeds
3. Rich messages are not supported
4. And certainly, you can forget about adding custom HTML

The solution

I have come up with perhaps the obvious workaround - it is possible to use the REST API that API.AI exposes to create the same experience, just that you can now have a much more customizable chatbot. The good news is that this is not very challenging if you are willing/able to write some code.

If you are non-technical, you should skip to the end of this post to the section titled "The New Chatbot". There is a way to download the code so you can ask your programmer to add it to your site.

Technical? Continue reading.

Now, to be able to use the REST API, you need to be able to use a server side programming language. Since WordPress already runs on the PHP server-side programming language, you have the most important piece of the puzzle already sorted out.

Plugin vs iFrame

Right now, you embed the built-in API.AI chatbot as an iFrame. We could use a similar approach and create a PHP page and use that as the base for our customizable chatbot. (that is, this would be the page which will go into the iFrame)

The other approach is to go the standard WordPress way and create a plugin.

For now, I am choosing the iFrame option for the following reasons:

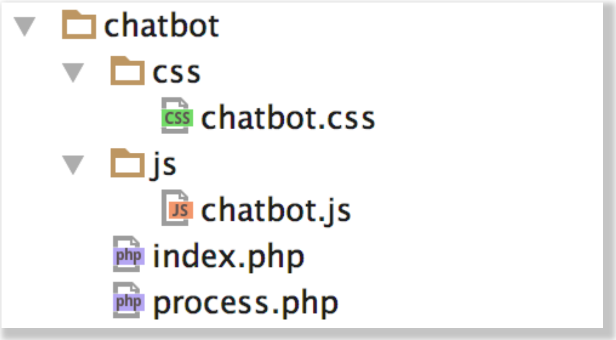
1. It is a simpler option. Creating a plugin is quite a bit of work, and I only have a small amount of experience in PHP programming.
2. The look and feel of the chatbot page is a lot more customizable because you don't have to worry about JavaScript issues

3. Separating the chatbot out as an embeddable iFrame also means you don't actually have any concerns about your plugin code interfering with any other code on the website, which seems to be a big concern in the WordPress ecosystem
4. You don't have to worry about site version updates and such and you can focus on simply improving the chatbot and its experience.
5. The page which will go into the iFrame will reside completely on your WordPress site, so there shouldn't be (hopefully) any security concerns

Now, I am not exactly a WordPress veteran. So if there are some out there who are reading this, I welcome your comments on the cons of this approach. I am sure there are some, but I am going with these options for now because the choice seems reasonable.

Folder structure

First you should create a chatbot folder, and all the chatbot code will be going into this folder.



```
▼ chatbot
  ▼ css
      chatbot.css
  ▼ js
      chatbot.js
    index.php
    process.php
```

There will be the standard css and js folders too, for the CSS styles and the JavaScript code respectively.

There will be two PHP files.

index.php will be the default page for the folder as usual, and there will also be in addition a **process.php** file. Since we need to use AJAX for our chatbot, the index.php file will post the user's message to the process.php file via AJAX, which will be interacting with the REST API, gathering the response, and sending it back to index.php so that it can be displayed within the chat box.

Code

index.php file

```
<html>
<head>
    <script src="https://code.jquery.com/jquery-1.9.1.js"></script>
    <script src="js/chatbot.js"></script>
    <link rel="stylesheet" href="css/chatbot.css" type="text/css">
</head>
<body>
<div class="topstrip" id="topstrip">Powered by <a href="https://www.miningbusinessdata.
<div class="topbar" id="chat-text">
</div>
<form>
    <span style="width:100%;">
        <input class="inputbox"
               placeholder="Write something and press Enter..." id="message" name="date
    </span>
    <input name="submit" type="hidden" value="Submit">
</form>
</body>
</html>
```

process.php file

```
<?php
if(isset($_POST['submit'])){
    $query = $_POST['message'];

    $postData = array('query' => array($query), 'lang' => 'en', 'sessionId' => 12345678
    $jsonData = json_encode($postData);
    $ch = curl_init('https://api.api.ai/v1/query?v=20170428');
    curl_setopt($ch, CURLOPT_POST, 1);
    curl_setopt($ch, CURLOPT_POSTFIELDS, $jsonData);
    curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type: application/json', 'Autho
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    $result = curl_exec($ch);

    echo $result;
    curl_close($ch);

}
?>
```

chatbot.css

```
.topstrip{
    width:99%;
    overflow-y: scroll;
    position:fixed;
    top:0px;
    height:50px;
    background-color: black;
    color:white;
    text-align: right;
    font-size: 16px;
    font-family: Georgia;
}
```

```
.topbar{
    display:block;
    width:100%;
    overflow-y: scroll;
    position:fixed;
    top:50px;
    bottom:65px;
}
.rounded-div-bot{
    display:block;
    width:80%;
    border-radius: 5px;
    border:solid 1px black;
    background-color: lightgreen;
    float:right;
    margin: 10px;
    padding:5px;
    min-height: 1em;
}
.rounded-div{
    display:block;
    width:80%;
    border-radius: 5px;
    margin: 10px;
    padding:5px;
    border:solid 1px black;
    background-color: lightgray;
    float:left;
}
.inputbox{
    outline:none;
    width:98%;
    height:40px;
    position:fixed;
    bottom:25px;
}
```

chatbot.js

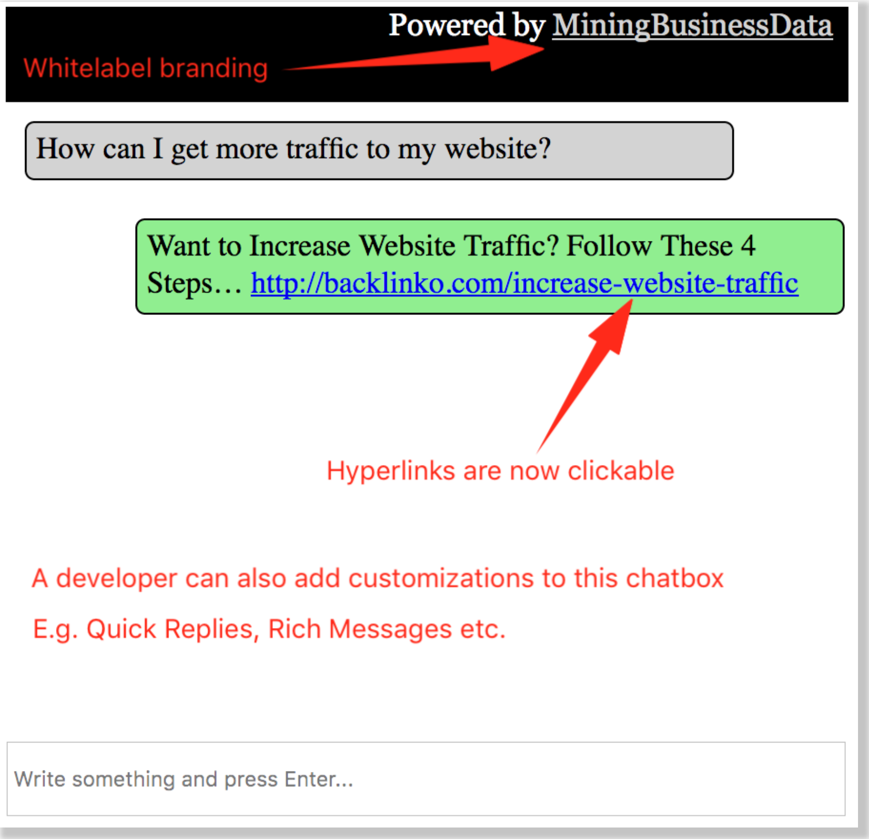
```
/**
 * Created by aravind on 4/28/17.
 */
$(function () {

    $('form').on('submit', function (e) {
        var query = $("#message").val();
        showUserText();
        e.preventDefault();

        $.ajax({
            type: 'post',
            url: 'process.php',
            //data: { userID : userID }
            data: {submit:true, message:query},
            success: function (response) {
                var obj = JSON.parse(response);
                var answerdiv = jQuery('<div/>', {
                    html: obj.result.fulfillment.speech.linkify()+'&nbsp;',
                    'class': "rounded-div-bot",
                    tabindex:1
                });
                $("#chat-text").append(answerdiv);
                $(answerdiv).focus();

                $("#message").focus();
            }
        });

    });

});

function showUserText(){
    var div = jQuery('<div/>', {
        text: $("#message").val(),
        'class': "rounded-div",
        tabindex:1
    });
    $("#chat-text" ).append(div);
    $("#message").val('');
}

if(!String.linkify) {
    String.prototype.linkify = function() {

        // http://, https://, ftp://
        var urlPattern = /\b(?:https?|ftp):\/\/[a-z0-9-+&@#\/%?=~_|!:,.;]*[a-z0-9-+&@#\

        // www. sans http:// or https://
        var pseudoUrlPattern = /(^|[^\/])(www\.[\S]+(\b|$))/gim;

        // Email addresses
        var emailAddressPattern = /[\w.]+@[a-zA-Z_-]+?(?:\.[a-zA-Z]{2,6})+/gim;

        return this
            .replace(urlPattern, '<a target="_blank" href="$&">$&</a>')
            .replace(pseudoUrlPattern, '$1<a target="_blank" href="http://$2">$2</a>')
            .replace(emailAddressPattern, '<a href="mailto:$&">$&</a>');
    };
}
```

Please note: the code above is experimental and is intended to get you started. If you choose to use it, you do so with the full understanding that I am not responsible if it causes any issues on your WordPress site. Also, a quick glance at process.php will immediately tell you that the sessionID is hard coded (which is wrong), and version (following v=?) should be based on the current date etc. Please email me or talk to your WordPress developer before running this code in production.

Once you create this folder and the requisite files, you can simply use the URL to the PHP file (or just the URL to the folder will be sufficient in a typical WordPress installation) and combine it with the system of embedding I already explained in my post on embedding an API.AI web agent.

The new chatbot

This is what the custom chatbox will look like:



Powered by MiningBusinessData

How can I get more traffic to my website?

Want to Increase Website Traffic? Follow These 4 Steps… http://backlinko.com/increase-website-traffic

Write something and press Enter...

Whitelabel branding
Hyperlinks are now clickable
A developer can also add customizations to this chatbox
E.g. Quick Replies, Rich Messages etc.

You can see that the somewhat OK design of the default API.AI chatbot has now been replaced by my really awesome MiningBusinessData branding. (OK, I am joking). But, **the hyperlinks are now active**! And since this UI is custom generated, you can add more code and customize the UI to your heart's content. Also, you now have the foundation to bypass the limitations of the default API.AI chatbox and you can also have quick replies, rich messages and such (after making the suitable tweaks, of course). And in theory, the CSS file has all the styles, so the chatbox colors are also "skinnable" to reflect your WordPress site's existing design.

You can see this bot in action on the chatbot demo page.

You can get the agent ZIP file as well as the code sample from my MBD Resources page (chapter 14)