**AI Chat Bot in Python using AIML**

AIML stands for Artificial Intelligence Markup Language, but it is just simple XML. AIML is a form of XML that defines rules for matching patterns and its responses. We will create the AIML files first and then use Python to give it some life.

**Create Standard Startup File**

It is standard to create a startup file called **std-startup.xml** as the main entry point for loading AIML files. In this case we will create a basic file that matches one pattern and takes one action. We want to match the pattern **load aiml b**, and have it load our aiml brain in response. We will create the basic\_chat.aiml.

**<aiml** **version**="1.0.1" **encoding**="UTF-8"**>**  
    <!-- std-startup.xml -->  
  
    <!-- Category is an atomic AIML unit -->  
    **<category>**  
  
        <!-- Pattern to match in user input -->  
        <!-- If user enters "LOAD AIML B" -->  
        **<pattern>**LOAD AIML B**</pattern>**  
  
        <!-- Template is the response to the pattern -->  
        <!-- This learn an aiml file -->  
        **<template>**  
            **<learn>**basic\_chat.aiml**</learn>**  
            <!-- You can add more aiml files here -->  
            <!--<learn>more\_aiml.aiml</learn>-->  
        **</template>**  
          
    **</category>**  
  
**</aiml>**

**Creating an AIML File**

Above we created the AIML file that only handles one pattern, **load aiml b**. When we enter that command to the bot, it will try to load **basic\_chat.aiml**. It won't work unless we actually create it. Here is what we can put inside **basic\_chat.aiml**. We will match two basic patterns and respond.

**<aiml** **version**="1.0.1" **encoding**="UTF-8"**>**  
<!-- basic\_chat.aiml -->  
<!—Content inside pattern is our message and content inside template is bot response -->  
    **<category>**  
        **<pattern>**HELLO**</pattern>**  
        **<template>**  
            Well, hello!  
        **</template>**  
    **</category>**  
      
    **<category>**  
        **<pattern>**WHAT ARE YOU**</pattern>**  
        **<template>**  
            I'm a bot, silly!  
        **</template>**  
    **</category>**  
      
**</aiml>**

**Simplest Python Program**

This is the simplest program we can start with. It creates the aiml object, learns the startup file, and then loads the rest of the aiml files. After that, it is ready to chat, and we enter an infinite loop that will continue to prompt the user for a message. We will need to enter a pattern the bot recognizes. The patterns recognized depend on what AIML files you loaded.

We create the startup file as a separate entity so that we can add more aiml files to the bot later without having to modify any of the programs source code. We can just add more files to learn in the startup xml file.

**import** aiml  
  
# Create the kernel and learn AIML files

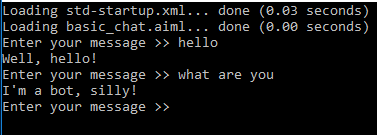
Creating aiml object named kernel  
kernel = aiml.Kernel()

# Making kernel object learn from “std-startup.xml” file  
  
kernel.learn("std-startup.xml")

# On purpose giving “load aiml b” to bot so that it learns from basic\_chat.aiml  
  
kernel.respond("load aiml b")  
  
# Press CTRL-C to break this loop  
**while** **True**:

Bot will respond to our messages  
  
    **print** kernel.respond(raw\_input("Enter your message >> "))

**After running this python code the output we get is:**



**How will it work in our execution?**

**Startup File**

It will be the same file, the main concept is to activate the learning process by pre entering the learn trigger input.(LOAD AIML B in this case )

<aiml version="1.0.1" encoding="UTF-8">

<!-- std-startup.xml -->

<!-- Category is an atomic AIML unit -->

<category>

<!—This is our learn trigger. Look at above mentioned python code, we had purposely given kernel “LOAD AIML B” so that it will respond to it. Its respond to text given is in this startup.xml file. -->

<pattern>LOAD AIML B</pattern>

<!-- Template is the response to the pattern -->

<!-- This learn an aiml file -->

<template>

<learn>basic\_chat.aiml</learn>

<!-- You can add more aiml files here -->

<!--<learn>more\_aiml.aiml</learn>-->

</template>

</category>

</aiml>

**Basic Chat file has to be made according to the no of patterns we want.**

**Below is the sample of the basic chat which we might want!**

<?xml version = "1.0" encoding = "UTF-8"?>

<aiml version="1.0.1" encoding="UTF-8">

<!-- basic\_chat.aiml -->

<!—This is our basic chat file -->

<category>

<pattern>HELLO</pattern>

<template>

Well, hello!

</template>

</category>

<category>

<pattern>WHAT ARE YOU</pattern>

<template>

I'm a bot, silly!

</template>

</category>

<category>

<pattern>GRAPH</pattern>

<template>

Select the graph you want to run

1.G1

2.G2

</template>

</category>

**<!—We can increase the no of choice and their respective outputs->**

<category>

<pattern>1</pattern>

<template>

G1 script to be run

</template>

</category>

<category>

<pattern>2</pattern>

<template>

G2 script to be run

</template>

</category>

</aiml>

**Our Python code will look something like this**

import aiml

**# Create the kernel and learn AIML files**

kernel = aiml.Kernel()

kernel.learn("std-startup.xml")

kernel.respond("LOAD AIML B")

**# Press CTRL-C to break this loop**

while True:

message=raw\_input("Enter your message for bot >> ")

if message == "quit":

exit()

else:

bot=kernel.respond(message)

print(bot)

if bot=="G1 script to be run":

print("g1")

elif bot=="G2 script to be run":

print("g2")

else:

print(bot)

**How to run unix script using python??**

**import os,sys**

**os.system(". ./scriptname.ksh")**

**python command to run unix script scriptname.ksh**

**By running a single python controlled script we can run ab initio graphs by taking user inputs.**