**基本格式**

<aiml>

<category>

<pattern>MY NAME IS \*</pattern>

<template> Hello <star/></template>

</category>

……

<category>

<pattern>WHAT IS YOUR NAME</pattern>

<template>My name is alice </template>

</category>

</aiml>

hello

Hello

说明：

1. 每个<category></category>为一个单元

2. 通配符： \_ 和 \* 且 \_ 优先级高于 \* 通配符需要与字符串分开，不能连在一起。

且必须匹配，

如，当<pattern>SAY \*</pattern> 时无法匹配到 say

3. 模板匹配不区分大小写。但是<pattern>的内容必须大写。

4. pattern忽略的标点符号为 ？. ! 将其作为句子分隔符

如：当用户输入为My name is 123！what’s your name?

AIML解释器会寻找

<pattern>MY NAME IS \*</pattern>

以及<pattern> WHAT IS YOUR NAME</pattern>

并且 用户输入的引号不能改变这一点。中文也会忽略标点符号，pattern中文符号同样省略，例如我要买房子，可以推荐一下吗？匹配模式

<pattern> 我 要 买 房 子 可 以 推 荐 一 下 吗 <pattern>，标点省略

5. 标签不能用大写比如<PATTERN>无效

6. <pattern>MATCH</pattern> MATCH不能有符号.

用户输入what’s your name？被自动展开并匹配WHAT IS YOUR NAME

进一步测试，let’s 却不能被识别成let us

1. <pattern> 有多个匹配时，匹配最后一个，前面的不会被匹配到：
2. <pattern> \*</pattern>,<pattern>\* \* 话</pattern>,第二个pattern不会被匹配到

<pattern>我 去 \* \* \* \* \* 大 概 得 花 多 少 钱 呢</pattern>

不会被匹配到，只需要一个\*

**说明：**

笔记中的例子存在大小写问题，无法正常运行，只是作为参考，具体测试的用例查看代码文件。

**标签：**

[1.<star>](#_<star>标签，用于匹配<pattern>标签中的通配符_*)

[2.<srai>](#_<srai>标签_有几个用途：_缩减符号，统一回复，合并同义词，针对关)

[3.<random> <li>](#_<random>标签，为pattern做出随机的response。与<)

[5.<set> <get>](#_<set>,<get>标签，设置和获取变量。变量可预设或在对话中获取)

[6.<that>](#_<that>标签，用于依据上下文的做出的响应)

[7.<topic>](#_<topic>标签，用于设置topic,限定部分category在某个)

[8.<think>](#_<think>标签，用于在设置变量时不显示)

[9.<condition>](#_<condition>标签，用于依据存储变量选择相应响应)

**补充：**

**来源： <http://www.alicebot.org/documentation/aiml101.html>**

**<http://www.alicebot.org/documentation/ptags.html>**

**服务器中aiml文件设置aiml版本为1.0官网相关文档已删，现有1.0.1版本文档，通过阅读官网提供文档补充如下：**

**对先前标签的补充：**

[<that>](M,N" \l "_<that_index=)

[<condition>](#_<condition>标签补充。)

**补充额外标签**

[<input index=”M,N”/>](M,N" \l "_<input_index=)

[<thatstar index=”N”/>](N" \l "_<thatstar_index=)

[<topicstar index=”N”/>](N" \l "_<topicstar_index=)

[<bot name=”XXX”/>](XXX" \l "_<bot_name=)

**获取信息**

[<date/>](#_<date/>标签，用于返回当前时间。没输出格式限制)

[<id/>](#_<id/>标签，用于返回ID，默认localhost)

[<size/>](#_<size/>标签，为当前加载的category数量)

[<version/>](#_<version/>标签，为当前AIML解释器版本)

**格式化输出**

[<uppercase>](#_<uppercase>标签，用于将内容转为大写)

[<lowercase>](#_<lowercase>标签，用于将内容转为小写,与<uppercase)

[<formal>](#_<formal>标签，用于将内容的每个单词首字母大写。)

[<sentence>](#_<sentence>标签，用于将内容的每个每句话首字母大写。)

**内容转变**

[<person>](#_<person>标签，用于对调第一人称代词和第二人称代词)

[<person2>](#_<person2>标签，用于对调第一人称代词和第三人称代词)

[<gender>](#_<gender>标签，用于对调男与女的代词，如he与she)

**学习相关**

[<gossip>](#_<gossip>标签，用于存储内容)

[<learn>](#_<learn>标签，用于命令aiml解释器加载其指定的资源。)

**进一步功能**

[<system>](#_<system>标签，用于将内容传到本地系统命令行中运行)

[<javascript>](#_<javascript>标签，用于将内容传到本地系统的javaScri)

**缩写**

[<sr/>](#_<sr/>标签，<srai><star/></srai>的缩写，无法添)

[<person2/>](#_<person2/>标签，<person2><star/></pers)

[<person/>](#_<person/>标签，<person><star/></person)

[<gender/>](#_<gender/>标签，<gender><star/></gender)

### 标签

##### <star>标签，用于匹配<pattern>标签中的通配符 \* 与 \_

**例：**

<category>

<pattern>My name is \*</pattern>

<template>

Hello <star/>

</template>

</category>

>> input: My name is WZ

>> response: Hello WZ

**补充说明：**

<star index = ”1”/> 也可直接简写为<star/>

\_ 比 \* 优先级高。甚至 在 \_ and \* 和 \* and \_ and \_ 同时能够匹配时，AIML解释器会选择 \_ and \*

##### <srai>标签 有几个用途： 缩减符号，统一回复，合并同义词，针对关键词回复

**例：**

<category>

<pattern>My name is \*</pattern>

<template>

Hello <star/>

</template>

</category>

<category>

<pattern>I’m \*</pattern>

<template>

<srai> My name is <star/></srai>

</template>

</category>

<category>

<pattern>\* I’m \*</pattern>

<template>

<srai> I am <star index =”2”/></srai>

</template>

</category>

>> input: I’m WZ

>> response: Hello WZ

**补充说明：**

该例子是以前一个例子为基础

如果 I’m WZ和 My name is WZ 相应都为 Hello WZ

在<star>的例子的基础上只需上述category即可

也就是说，可以将之前的pattern 作为我们这次的template

这里有意义的地方在于不需创建多个pattern、

1.对于一些相似对话（如hello \*和hello 都能统一回复 hello），

用 \* 过滤非关键词，然后统一回复。

2.对于同一个句话不同说法（如本例），

用srai调用之前的pattern作为template即可

3.对于出现某个关键词。用针对关键词的pattern进行回答

（如：用 \* 以及 \_ 过滤其他词语并匹配到“关键词”

则template返回之前pattern关键词）

1. pattern 匹配模式不应该是包含关系，包含关系会导致查找aiml循环，例如：

<pattern> 寄 \* 到 \* 的 重 量 为 \* 的 \* 快 递 要 多 少 钱 </pattern>

<pattern> 寄 \* 到 \* 的 重 \* 的 \* 快 件 要 多 少 钱 </pattern>

上面两句话是一样的。

这里说所说的多种用途，是根据<srai>的功能，所提出的用途

此外，srai的返回结果可以嵌套其他的内容.<srai>VALUE</srai>中的VALUE不区分大小写。

目前测试可以嵌套5层至少，上限没测

##### <random>标签，为pattern做出随机的response。与<li>标签一起使用

**例：**

<category>

<pattern>I’m \*</pattern>

<template>

<random>

<li> Hi <star/></li>

<li> Hello <star/></li>

<li> I’m Bob</li>

</random>

</template>

</category>

>> input: I’m Alice

>> response: I’m Bob

>> input: I’m Alice

>> response: Hi Alice

##### <set>,<get>标签，设置和获取变量。变量可预设或在对话中获取

**例：**

<aiml>

<category>

<pattern>I’m \*</pattern>

<template>

Hello <set name=”user”><star/></set>

</template>

</category>

<category>

<pattern>Bye</pattern>

<template>

Bye <get name=”user”/>

</template>

</category>

</aiml>

>> input: I’m Alice

>> response: Hello Alice

……

>> input: Bye

>> response: Bye Alice

##### <that>标签，用于依据上下文的做出的响应。返回上一句机器人回答

**例：**

<aiml>

<category>

<pattern>Hello</pattern>

<template>

Hi, are you happy today?

</template>

</category>

<category>

<pattern>Yes</pattern>

<that> HI ARE YOU HAPPY TODAY</that>

<template>

Tell me some good news

</template>

</category>

<category>

<pattern>No</pattern>

<that> HI ARE YOU HAPPY TODAY</that>

<template>

Let me tell you a joke

</template>

</category>

</aiml>

>> input: I’m Alice

>> response: Hi, are you happy today?

>> input: Yes

>> response: Tell me some good news

>> input: I’m Alice

>> response: Hi, are you happy today?

>> input: No

>> response: Let me tell you a joke

**补充说明：**

that内的字符全部大写，并且类似don’t这样的缩写应展开成 DO NOT（具体实现过程需要看源码）

That匹配中文：标点符号不会被识别，全部替换为空格。例如：

<category>  
 <pattern> 星 座 </pattern>  
 <template>  
 黄道十二宫，个个有玄机。请问你要查的是？  
 <think><set name = "topic">星座</set></think>  
 </template>  
</category>

<category>  
 <pattern> \* </pattern>  
 <that>黄 道 十 二 宫 个 个 有 玄 机 请 问 你 要 查 的 是 </that>  
 <template>  
 为小主奉上<star/>的概况哦#API  
 </template>  
</category>

##### <topic>标签，用于设置topic,限定部分category在某个topic中才有效

**例：**

<aiml>

<category>

<pattern>Hello Bob</pattern>

<template>

Hi, Do you like movies?

</template>

</category>

<category>

<pattern>Yes</pattern>

<that> Do you like movies? </that>

<template>

Let’s discuss <set name = ”topic” >movies</set>

</template>

</category>

<topic name = “MOVIES”>

<category>

<pattern>Tell me some things</pattern>

<template>

Could you tell me your favorite movie?

</template>

</category>

</topic>

</aiml>

>> input: Hello Bob

>> response: Hi, Do you like movies?

>> input: Yes

>> response: Let’s discuss movies

>> input: Tell me some things

>> response: Could you tell me your favorite movie?

**补充说明：**

**topic的赋值为set标签提供。**<set name = ”topic” >**这里的topic必须小写。**

set设置的其他边量是无法像topic一样划category有效范围

注意，<topic name = “VALUE”> 中的VALUE要大写。当设置topic的value存在引号或其他标点，<topic name = “VALUE”>中的VALUE的相应位置补上空格

**Topic是中文，**<set name = ”topic” >这里的topic正常写，<topic name = “中文”>，中文按照空格隔开。例如<set name = “topic”>星座</set>，<topic name = “星 座”>.

##### <think>标签，用于在设置变量时不显示

**例：**

<aiml>

<category>

<pattern>Hello Bob</pattern>

<template>

Hi, Do you like movies?

</template>

</category>

<category>

<pattern>Yes</pattern>

<that> Do you like movies? </that>

<template>

Let’s discuss it<think><set name =” topic” >movies</set></think>

</template>

</category>

<topic name = “movies”>

<category>

<pattern>Tell me some things</pattern>

<template>

Could you tell me your favorite movie?

</template>

</category>

</topic>

</aiml>

>> input: Hello Bob

>> response: Hi, Do you like movies?

>> input: Yes

>> response: Let’s discuss it

>> input: Tell me some things

>> response: Could you tell me your favorite movie?

**补充说明：**

在之前所说的<set>设置变量时，在设置同时会将<set>与</set>间内容输出

但在<set>外包裹一层<think>后，可以设置其不显示。

目的就为使回答更为拟人化吧

强调。如果是在think外面包裹一层set。则set会将变量赋值为空

##### <condition>标签，用于依据存储变量选择相应响应

**例：**

<aiml>

<category>

<pattern>Hello, I’m Alice</pattern>

<template>

Hi, are you happy today?

</template>

</category>

<category>

<pattern>Yes</pattern>

<that> Are you happy today? </that>

<template>

<think><set name =” feeling” >happy</set></think>

Tell me some good news

</template>

</category>

<category>

<pattern>No</pattern>

<that> Are you happy today? </that>

<template>

<think><set name =” feeling” >sad</set></think>

Oh, it’s so sad

</template>

</category>

<category>

<pattern>Are you happy? </pattern>

<template>

<condition name=”feeling” value =”happy”>

Yes, I’m so happy today!

</condition>

<condition name=”feeling” value= “sad”>

No.

</condition>

</template>

</category>

</aiml>

>> input: Hello, I’m Alice

>> response: Hi, are you happy today?

>> input: Yes

>> response: Tell me some good news

……

>> input: Are you happy?

>> response: Yes, I’m so happy today!

**补充说明：**

依据变量的值来选择相应的回答。利用这个选择符合当前环境下的对话，

个人与<topic>相比，两者侧重点不一样.

<topic>使得对话围绕某一话题，并设置针对性对话。

<condition>更加灵活地选择回复

### 对先前标签内容的补充

##### <that index=”M,N”/>标签补充，用于依据存储变量选择相应响应

**说明：**

that 用法与上述一样，只是指出是距离当前第M 个对话的，N是第N个机器人回应（?）

**额外：**

<that index=”M”/>即可得到距离当前第M个回答

##### <condition>标签补充。

**补充：**

<condition> 标签存在可选属性 name与value。

先前condition说明中将两个都添加上，此外还有<condition name=”X”>与<condition>两种情况。

**其中<condition name=”X”>用法如下：**

<condition name=”X”>

<li value=”Y”> response\_0</li>

……

<li> response\_n </li>

</condition>

**说明：**

至少包含一个li。其中<li>只能存在一个，意味着默认值。

**而<condition>用法如下：**

<condition>

<li name=”” value=”Y”>response\_0</li>

……

<li> response\_n</li>

</condition>

**说明：**

至少包含一个li。其中<li>只能存在一个，意味着默认值。

判断顺寻按书写顺序，匹配后结束判断。

**额外说明：**

**condition中 name 与 value 是严格区分大小的**

##### <input index=” N”/>标签，用于返回之前的某句话

##### <thatstar index=”N”/>标签，<that>标签里面的 \* 内容

##### <topicstar index=”N”/>标签，用于获取topic中的\*

指的是<topic name =”value”>中的value的 \*

##### <bot name=”XXX”/>标签，用于设置bot。只能用于加载过程中，运行过程中无法修改。 不会用

##### <date/>标签，用于返回当前时间。没输出格式限制

格式化输出无效？

##### <id/>标签，用于返回ID，默认localhost，测试时候为\_global

##### <size/>标签，为当前加载的category数量

##### <version/>标签，为当前AIML解释器版本

##### <uppercase>标签，用于将内容转为大写

**说明：**

<uppercase>value</uppercase> 将 value转为大写

##### <lowercase>标签，用于将内容转为小写,与<uppercase>对应

##### <formal>标签，用于将内容的每个单词首字母大写。

特别的，如果第一个单词前面紧跟着某个字符（如引号+-），则变成将该字符变成大写形式(如果存在)，该单词其他字符变为小写，

##### <sentence>标签，用于将内容的第一个单词的formal化

只能对第一个单词变成首字母大写形式，剩下的全部保留原样。

特别的，如果第一个单词前面紧跟着某个字符（如引号），则变成将该字符变成大写形式(如果存在)，该单词其他字符变为小写。

##### <person>标签，用于对调第一人称代词和第三人称代词

she和he都会调换成I对应的，而I只会调换成he对应的。

##### <person2>标签，用于对调第一人称代词和第二人称代词

##### <gender>标签，用于对调男与女的代词，如he与she

##### <gossip>标签，用于存储内容

**说明：**

<gossip>value</gossip>将value存入文件中。。

没法用

##### <learn>标签，用于命令aiml解释器加载其指定的资源。

##### <system>标签，用于将内容传到本地系统命令行中运行

##### <javascript>标签，用于将内容传到本地系统的javaScript解释器。

##### <sr/>标签，<srai><star/></srai>的缩写，无法添加内容

##### <person2/>标签，<person2><star/></person2>的缩写

##### <person/>标签，<person><star/></person>的缩写

##### <gender/>标签，<gender><star/></gender>的缩写