1. Data structure

* description: a list including all of result.
* data type:

effecitvePatterns =

{

"name":

{

value: "",

pattern:

[

{

"tags": ("div", [("class", "webtop-g")]),

“is in”: False

}

{

"tags": ("h2", [("class", "h")]),

“is in”: False

}

]

},

"word class":

{

value: "",

pattern:

[

{

"tags": ("div", [("class", "webtop-g")]),

“is in”: False

}

{

"tags": ("span", [("class", "pos")]),

“is in”: False

}

]

},

"phonetic symbol":

{

value: "",

pattern:

[

{

"tags": ("div", [("class", "pron-gs ei-g")]),

“is in”: False

}

{

"tags": ("span", [("class", "pron-g"), ("geo", n\_am)] ),

“is in”: False

}

]

},

"definition":

{

value: "",

pattern:

[

{

"tags": ("span|li", [("class", "sn-g")]),

“is in”: False

}

]

}

}

ignoredPatterns =

[

[

{

“tags”: ("span", [("class", "collapse")]),

“is in”: False

}

],

[

{

“tags”: ("span", [("class", "idm-gs")]),

“is in”: False

}

],

[

{

“tags”: ("div", [("class", "sound audio\_play\_button pron-us icon-audio")]),

“is in”: False

}

]

]

1. 判断2个tag是否相等, bool CompareTag(left, right).
2. 判断data 是否处于igored tags pattern,

Def IsInIgnoredPattern():

for pattern in ignoredPatterns:

if pattern[len(pattern – 1)] != 0:

return True

return False

1. 判断是否处于某个pattern之中：name， word class，phonetic symbol，definition

Def GetTagsPatternName(name):

return effecitvePatterns[name] [len(effecitvePatterns[i]) - 1][2] != 0

1. self.stack = [ (("span", [("class", "collapse")]), [(“ignoredPatterns”, 0, 0), ('effecitvePatterns', "name", 0)]) ]
2. handle\_starttag

stackProperties = []

for i in range(0, len(ignoredPatterns)):

pattern = ignoredPatterns[i]

for j in pattern:

topTag = pattern[j]

if CompareTag (topTag [0], (tag, attrs)) and topTag [1] == 0:

topTag [1] = 1

stackProperties.append((ignoredPatterns, i, j))

break

for pattern in effecitvePatterns:

for i in pattern:

topTag = pattern[i]

if CompareTag (topTag [0], (tag, attrs)) and topTag [1] == 0:

topTag [1] = 1

stackProperties.append((effecitvePatterns, pattern, i))

break

self. stack.append((tag, attrs), stackProperties)

1. handle\_endtag
2. handle\_data

if IsInIgnoredPattern():

return

for I in ["name", "word class" …]:

if GetTagsPatternName(i):

effecitvePatterns [i][value] += data

break

1. \_\_init\_\_(self, patterns, ignorePatterns)

为了方便做单元测试， effecitvePatterns 和 ignoredPatterns 的值不能写死， 而是通过类的构造函数输入的。

1. Css 样式处理机制

cssBlockTags = \

[

("span", [("class", "sn-gs")]),

("span", [("class", "sn-g")]),

("span", [("class", "x-gs")]),

("span", [("class", "x-g")]),

("span", [("class", "xr-gs")]),

]

1. GetStatck(self)

用于单元测试的函数。

1. GetEffectivePatternsState (self)

用于单元测试的函数。

1. GetIgnoredPatternsState()

用于单元测试的函数。