






MÔ TẢ

Các Class trong chương trình:

 Event.py
 EventCollection.py
 Flow.py
 Monitor.py
 Testcase.py

Monitor: đọc và lưu các thông tin trong file quy tắc (annotations.xlsx), có các method để xác định, trích xuất ra tên luồng, tên sự kiện, nhận biết annotation từ 1 chuỗi ký tự đầu vào. Lớp này cũng có các method trả về ý nghĩa, html tương ứng của các annotation.

Flows: tương ứng với 1 luồng sự kiện

Gồm các thuộc tính: tên luồng, danh sách sự kiện (events)


Event: tương ứng với 1 sự kiện

Gồm các thuộc tính: tên sự kiện (số thứ tự), danh sách các annotation nó chứa, danh sách các sự kiện liên tiếp nó

EventsCollection: giống như một tập hợp có thứ tự (ordered set) của các sự kiện. Lưu 1 tập các events mà không chứa các event trùng nhau

Testcase: tương ứng với 1 testcase của 1 flow.

Có các thuộc tính như: index, input element, input label, output state, output label, output element

 services.py

Services.py : các hàm chính của chương trình. Gồm:

1. Hàm đọc file đặc tả

Input: file đặc tả (usecase) định dạng txt

Output: danh sách flows và danh sách events

Code	Mã giả
<pre>def read_usecases_txt(usecases_file_path): flows = [] events_collection = EventCollection() lines = [] with open(usecases_file_path, mode="r", encoding="utf-8") as reader: lines = reader.readlines() for line in lines: if (monitor.is_flow_name(line)): flows.append(Flow(line)) elif (monitor.is_event_name(line)): e = Event(monitor.extract_event_name(line)) e.set_annotations(monitor.extract_annotations(line)) events_collection.add_event(e) flows[-1].add_event(events_collection.get_event(e.name)) return {'flows': flows, 'events': events_collection}</pre>	<p>B1: Khai báo: flows // danh sách các luồng có trong file đặc tả events_collection // tập hợp các sự kiện có trong file đặc tả</p> <p>B2: Đọc các dòng trong file usecase</p> <p>B3: Với mỗi dòng (line), kiểm tra: Nếu dòng đó là tên luồng: - Tạo 1 luồng với tên luồng = line - Thêm luồng đó vào flows Còn nếu dòng đó là sự kiện: - Tạo 1 sự kiện với tên và các annotation của nó được extract từ line - Thêm sự kiện đó vào events_collection // vì events_collection là thể hiện của lớp EventCollection: là 1 ordered set các events → các sự kiện chỉ được thêm vào nếu nó chưa tồn tại trong set - Tìm lại sự kiện (event) từ events_collection và thêm vào luồng đang xét // luồng đang xét: luồng mới được tạo = luồng ở cuối danh sách flows // lý do tìm lại event từ events_collection thay vì thêm trực tiếp vào luồng đang xét: để đảm bảo các event xuất hiện trong các flow đều nhất quán, không tồn tại 2 event có cùng tên nhưng có annotation khác nhau</p> <p>B4: trả về danh sách các luồng và danh sách các sự kiện có trong file đặc tả</p>

2.Hàm đọc file luồng sự kiện

Input: file luồng sự kiện (flows) định dạng txt

Output: danh sách flows và danh sách events

Code	Mã giả
<pre>def read_flows_txt(flows_file_path): flows = [] events_collection = EventCollection() collected = False lines = [] with open(flows_file_path, mode="r", encoding="utf-8") as reader: lines = reader.readlines() for line in lines: words = line.split() if (len(words) > 0 and monitor.is_event_name(line)): flow = Flow("Flow " + str(lines.index(line))) flows.append(flow) for word in words: if (monitor.is_event_name(word)): e = Event(word) if (events_collection.contain(e)): collected = True else: collected = False events_collection.add_event(e) flow.add_event(events_collection.get_event(e.name)) elif ((not collected) and (monitor.is_annotation "[" + word + "]")): flow.events[-1].add_annotation(word) return {'flows':flows, 'events':events_collection}</pre>	<p>B1: Khai báo: flows // danh sách các luồng có trong file luồng sự kiện events_collection // tập hợp các sự kiện (event) có trong file collected = false // nếu event đang xét đã xuất hiện trong tập hợp các sự kiện (events_collection)</p> <p>B2: Đọc các dòng trong file luồng sự kiện</p> <p>B3: Với mỗi dòng (line): Chia dòng đang xét thành các từ (words) Nếu dòng có số từ > 0 // dòng không rỗng và bắt đầu bằng tên 1 sự kiện:</p> <ul style="list-style-type: none">- Tạo 1 luồng có tên = số thứ tự của dòng đang xét- Thêm luồng đó vào flows- Thực hiện phân tích các từ (B4) <p>Nếu không → xét dòng tiếp theo (B3)</p> <p>B4: Với mỗi từ (word): Nếu word là tên sự kiện:</p> <ul style="list-style-type: none">- Tạo 1 sự kiện có tên = word- Kiểm tra xem sự kiện đó đã tồn tại trong events_collection chưa:<ul style="list-style-type: none">+ Nếu có → gán collected = true+ Nếu không → gán collected = false và thêm sự kiện // sự kiện này rỗng, chưa có annotation vào events_collection <p>- Tìm lại sự kiện (event) từ events_collection và thêm vào luồng đang xét</p> <p>Nếu word là annotation và biến collected = false thì thêm annotation này vào sự kiện đang xét</p>

	<p>// chỉ thêm annotation với sự kiện chưa tồn tại trong events_collection, nghĩa là những sự kiện chưa từng được xét</p> <p>// tránh trường hợp khi 1 sự kiện xuất hiện lặp lại → chương trình lại xét lại các annotation phía sau nó</p> <p>B5: trả về danh sách các luồng và danh sách các sự kiện có trong file luồng sự kiện</p>
--	---

3.Hàm đọc file sự kiện

Input: file sự kiện (events) định dạng txt

Output: danh sách events

Code	Mã giả
<pre>def read_events_txt(events_file_path): events_collection = EventCollection() lines = [] with open(events_file_path, mode="r", encoding="utf-8") as reader: lines = reader.readlines() for line in lines: words = line.split() if (len(words) > 0 and monitor.is_event_name(line)): event = Event(monitor.extract_event_name(line)) events_collection.add_event(event) event = events_collection.get_event(event.name) for word in range(1, len(words) - 1): if (monitor.is_event_name(word)): events_collection.add_event(Event(word)) e = events_collection.get_event(word) event.add_next_event(e) elif (monitor.is_annotation "[" + word + "]"): event.add_annotation(word) return {'events':events_collection}</pre>	<p>B1: Khai báo events_collection // tập hợp các sự kiện (event) có trong file</p> <p>B2: Đọc các dòng trong file sự kiện</p> <p>B3: Với mỗi dòng (line): Chia dòng đang xét thành các từ (words) Nếu dòng có số từ > 0 // dòng không rỗng và bắt đầu bằng tên 1 sự kiện:</p> <ul style="list-style-type: none">- Tạo 1 sự kiện với tên được extract từ line- Thêm sự kiện đó vào events_collection- Thực hiện phân tích các từ còn lại (B4) <p>Nếu không → xét dòng tiếp theo (B3)</p> <p>B4: Với mỗi từ (word): Nếu word là tên sự kiện:</p> <ul style="list-style-type: none">- Tạo 1 sự kiện có tên = word- Thêm sự kiện này vào events_collection- Thêm sự kiện này vào tập hợp sự kiện tiếp theo của sự kiện đang xét ở B3 <p>Còn nếu word là annotation:</p> <ul style="list-style-type: none">- Thêm annotation vào sự kiện đang xét ở B3 <p>B5: Trả về danh sách các sự kiện có trong file</p>

4.Hàm ghi luồng sự kiện

Input: danh sách các luồng

Output: xuất file luồng sự kiện định dạng txt

Code	Mã giả
<pre>def flows_to_txt(flows, txt_file_path): with open(txt_file_path, mode="w", encoding="utf-8") as writer: for flow in flows: writer.write(str(flow)) writer.write("\n")</pre>	B1: Mở file txt muốn viết B2: Với mỗi luồng (flow) trong danh sách luồng: Ghi thông tin luồng vào file

5.Hàm ghi sự kiện

Input: danh sách các sự kiện

Output: xuất file sự kiện dưới định dạng txt

Code	Mã giả
<pre>def events_to_txt(events_collection, txt_file_path): with open(txt_file_path, mode="w", encoding="utf-8") as writer: for e in events_collection.events: writer.write(str(e)) writer.write("\n")</pre>	B1: Mở file txt muốn viết B2: Với mỗi sự kiện (event) trong tập hợp sự kiện (events_collection): Ghi thông tin sự kiện vào file

6. Hàm sinh testcase từ 1 luồng sự kiện

Input: 1 luồng sự kiện (flow)

Output: 1 testcase

Code	Mã giả																																																		
<pre>def flow_to_testcase(index, flow): t = Testcase(index) # testcase input input_annotations = flow.get_input_annotations() i = 0 while (i < len(input_annotations)): # Nếu annotation là input element if (monitor.is_input_element(input_annotations[i])): input_element = monitor.get_element_meaning(input_annotations[i]) # laasy meaning # Nếu annotation tiếp theo là label if (monitor.is_label(input_annotations[i + 1])): t.add_input(input_element=input_element, input_label=input_annotations[i + 1].strip("#")) i = i + 1 else: t.add_input(input_element, "") # Nếu annotation là label elif (monitor.is_label(input_annotations[i])): t.add_input("", input_annotations[i].strip("#")) i = i + 1</pre>	<p>B1: Khai báo: t = Testcase (index)</p> <p>----- sinh testcase input -----</p> <p>B2: Trích xuất danh sách các input annotations từ flow</p> <pre>input_annotations = flow.get_input_annotations()</pre> <p>B3: Với mỗi input annotation, kiểm tra: Nếu annot là input element: // input element là annotation được định nghĩa trong sheet elements và không có 'output' trong meaning</p> <table><tr><th></th><th>A</th><th>B</th><th>C</th><th>D</th></tr><tr><td>1</td><td>annotation</td><td>meaning</td><td>html-open</td><td>html-close</td></tr><tr><td>2</td><td>#ao</td><td>alert box output</td><td></td><td></td></tr><tr><td>3</td><td>#bi</td><td>button onclick input</td><td><button type="button"></td><td></button></td></tr><tr><td>4</td><td>#fi</td><td>file input</td><td><input type="file">
</br></td><td></td></tr><tr><td>5</td><td>#li</td><td>link onclick input</td><td></td><td></td></tr><tr><td>6</td><td>#si</td><td>selection input</td><td><select> <option>A</option> <option>B</option> </select></td><td></td></tr><tr><td>7</td><td>#ti</td><td>text input</td><td><input type="text">
</br></td><td></td></tr><tr><td>8</td><td>#to</td><td>text output</td><td><p></td><td></p></td></tr><tr><td>9</td><td></td><td></td><td></td><td></td></tr></table> <p>elements states flows events annotations </p> <ul style="list-style-type: none">- Tìm ý nghĩa của annot (element)- Nếu annot tiếp theo là label → thêm 1 cặp <element, label> vào testcase input → bỏ qua bước duyệt annot tiếp theo- Nếu annot tiếp theo không phải label → thêm 1 cặp <element, ' ' > vào testcase input <p>Nếu annot là label:</p> <ul style="list-style-type: none">- Thêm 1 cặp < ' ', label> vào testcase input		A	B	C	D	1	annotation	meaning	html-open	html-close	2	#ao	alert box output			3	#bi	button onclick input	<button type="button">	</button>	4	#fi	file input	<input type="file"> </br>		5	#li	link onclick input			6	#si	selection input	<select> <option>A</option> <option>B</option> </select>		7	#ti	text input	<input type="text"> </br>		8	#to	text output	<p>	</p>	9				
	A	B	C	D																																															
1	annotation	meaning	html-open	html-close																																															
2	#ao	alert box output																																																	
3	#bi	button onclick input	<button type="button">	</button>																																															
4	#fi	file input	<input type="file"> </br>																																																
5	#li	link onclick input																																																	
6	#si	selection input	<select> <option>A</option> <option>B</option> </select>																																																
7	#ti	text input	<input type="text"> </br>																																																
8	#to	text output	<p>	</p>																																															
9																																																			

```

# testcase output
output_annotations = flow.get_output_annotations()
i = 0
while (i < len(output_annotations)):
    # Nếu annotation là state
    if (monitor.is_state(output_annotations[i])):
        t.add_output_state(output_annotations[i].strip("#"))
    # Nếu annotation là output element
    elif (monitor.is_output_element(output_annotations[i])):
        t.add_output_element(monitor.get_element_meaning(output_annotations[i]))
    # Nếu annotation là label
    else:
        t.add_output_label(output_annotations[i].strip("#"))

    i = i + 1
return t.dictionary()

```

----- sinh testcase output -----

B4: Trích xuất danh sách các output annotations từ flow

```
output_annotations = flow.get_output_annotations()
```

B5: Với mỗi output annotation, kiểm tra:
Nếu annot là state → Thêm annot vào testcase output state

// state là annotation được định nghĩa trong sheet state

	A	B	C
1	annotation		
2	#empty		
3	#invalid		
4	#valid		
5			

elements states

Nếu annot là output element → Thêm meaning của annot vào testcase output element

// output element là annotation được định nghĩa trong sheet elements và không có 'input' trong meaning

	A	B	C	D
1	annotation	meaning	html-open	html-close
2	#ao	alert box output		
3	#bo	button onclick input	<button type="button">	</button>
4	#fi	file input	<input type="file"> </br>	
5	#li	link onclick input		
6	#si	selection input	<select> <option>A</option> <option>B</option> </select>	
7	#ti	text input	<input type="text"> </br>	
8	#to	text output	<p>	</p>
9				

elements states flows events annotations

Nếu annot là label → Thêm annot vào testcase output label

B6: trả về testcase

7. Hàm sinh file testcase từ nhiều luồng sự kiện

Input: danh sách các luồng sự kiện

Output: xuất file testcase dưới định dạng excel

Code	Mã giả
<pre>def flows_to_excel(flows, excel_file_path): testcases = { 'STT': [], 'input element': [], 'input label': [], 'input data': [], 'expected output element': [], 'expected output label': [], 'expected output state': [] } for i in range(len(flows)): t = flow_to_testcase(i + 1, flows[i]) testcases['STT'].extend(t['STT']) testcases['input element'].extend(t['input element']) testcases['input label'].extend(t['input label']) testcases['input data'].extend(t['input data']) testcases['expected output label'].extend(t['expected output label']) testcases['expected output element'].extend(t['expected output element']) testcases['expected output state'].extend(t['expected output state']) data = DataFrame(testcases) with ExcelWriter(excel_file_path) as writer: data.to_excel(writer)</pre>	<p>B1: Khai báo các trường thông tin trong bảng testcase:</p> <p>STT Input element Input label Input data Expected output element Expected output label Expected output state</p> <p>B2: Với mỗi flow trong danh sách các flows đã cho:</p> <ul style="list-style-type: none">- Sinh testcase tương ứng- Thêm các thông tin trong testcase được sinh vào các trường đã khai báo <p>B3: Xuất file testcase định dạng excel</p>

8. Hàm sinh html từ 1 annotation

Input: 1 cặp <label, annotation>

Output: string biểu diễn 1 đối tượng html tương ứng

Code	Mã giả:
<pre>def to_html(label, element_annot): html_label = "" html_element = "" if (element_annot): open_html = monitor.get_element_o_html(element_annot) close_html = monitor.get_element_c_html(element_annot) if (len(close_html)): return '\t' + open_html + label + close_html + '\n' else: html_element = open_html if (label): html_label = "<label>" + label + "</label>" return '\t' + html_label + '\n' + '\t' + html_element + '\n'</pre>	<p>B1: Khai báo: Html_label: html của phần label Html_element: html của phần annotation</p> <p>B2: Xét annotation: Nếu annot có thẻ html đóng và mở:</p> <ul style="list-style-type: none">- Đặt label nằm giữa 2 thẻ này- Trả về chuỗi html có dạng: <open-html-tag> label <close-html-tag> <p>Nếu annot chỉ có thẻ mở</p> <ul style="list-style-type: none">- Tạo thẻ label tiếng html_label = "<label>" + label + "</label>"- Trả về chuỗi html có dạng: <open-html-tag> <label> label </label>

9. Hàm sinh file html từ 1 luồng sự kiện

Input: 1 luồng sự kiện (flow)

Output: file html chứa template UI tương ứng

Code	Mã giả
<pre>def flow_to_html(flow, html_file_path): html = '' testcase = flow_to_testcase(0, flow) for i in range(len(testcase['input element'])): html = html + to_html(testcase['input label'][i], testcase['input element'][i]) lines = [] with open(HTML_FILE_PATH, mode="r", encoding="utf-8") as reader: lines = reader.readlines() # Write the file out again with open(html_file_path, mode="w", encoding="utf-8") as writer: for line in lines: line = line.replace(HTML_REPLACE_TAG, html) writer.write(line)</pre>	<p>B1: sinh testcase từ luồng đã cho</p> <p>B2: chuyển từng cặp <input label, input element> trong testcase thành html tương ứng</p> <p>B3: đọc nội dung file html-template</p> <p>B4: điền thông tin html ở B2 vào vị trí HTML_REPLACE_TAG trong nội dung file đã đọc ở B3</p> <p>B5: ghi nội dung trên ra file html</p>