MÔ TẢ

Các Class trong chương trình:

Graphical user interface, application

Description automatically generated

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 liền 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** : 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 |
|  | 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  B2: Đọc các dòng trong file usecase  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  B4: trả về danh sách các luồng và danh sách các sự kiện có trong file đặc tả |

## 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 |
|  | 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)  B2: Đọc các dòng trong file luồng sự kiện  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:   * Tạo 1 luồng có tên = số thứ tự của dòng đang xét * Thêm luồng đó vào flows * Thực hiện phân tích các từ (B4)   Nếu không 🡪 xét dòng tiếp theo (B3)  B4: Với mỗi từ (word):  Nếu word là tên sự kiện:   * Tạo 1 sự kiện có tên = word * Kiểm tra xem sự kiện đó đã tồn tại trong events\_collection chưa: * Nếu có 🡪 gán collected = true * Nếu không 🡪 gán collected = false và thêm sự kiện // sự kiện này rỗng, chưa có annotation vào events\_collection * Tìm lại sự kiện (event) từ events\_collection và thêm vào luồng đang xét   Nếu word là annotation và biến collected = false thì thêm annotation này vào sự kiện đang xét  // chỉ thêm annotation với sự kiện chưa tồn tại trong events\_collection, nghĩa là những sự kiện chưa từng được xét  // tránh trường hợp khi 1 sự kiện xuất hiện lặp lại 🡪 chương trình lại xét lại các annotation phía sau nó  B5: trả về danh sách các luồng và danh sách các sự kiện có trong file luồng sự kiện |

## 3.Hàm đọc file sự kiện

Input: file sự kiện ( events ) định dạng txt

Output: danh sách events

|  |  |
| --- | --- |
| Code | Mã giả |
|  | B1: Khai báo  events\_collection // tập hợp các sự kiện (event) có trong file  B2: Đọc các dòng trong file sự kiện  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:   * 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)   Nếu không 🡪 xét dòng tiếp theo (B3)  B4: Với mỗi từ (word):  Nếu word là tên sự kiện:   * 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   Còn nếu word là annotation:   * Thêm annotation vào sự kiện đang xét ở B3   B5: Trả về danh sách các sự kiện có trong file |

## 4.Hàm ghi luồng sự kiện

Input: danh sách các luồng

Output: xuất file luồng sự kiện định dạng txt

|  |  |
| --- | --- |
| Code | Mã giả |
|  | 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 |
|  | 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 |
|  | B1: Khai báo: t = Testcase (index)  --------------- sinh testcase input ----------------  B2: Trích xuất danh sách các input annotations từ flow    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   * 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   Nếu annot là label:   * Thêm 1 cặp < ’ ‘, label> vào testcase input   --------------- sinh testcase output ----------------  B4: Trích xuất danh sách các output annotations từ flow    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    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  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 |
|  | B1: Khai báo các trường thông tin trong bảng testcase:  STT  Input element  Input label  Input data  Expected output element  Expected output label  Expected output state  B2: Với mỗi flow trong danh sách các flows đã cho:   * Sinh testcase tương ứng * Thêm các thông tin trong testcase được sinh vào các trường đã khai báo   B3: Xuất file testcase định dạng excel |

## 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: |
|  | B1: Khai báo:  Html\_label: html của phần label  Html\_element: html của phần annotation  B2: Xét annotation:  Nếu annot có thẻ html đóng và mở:   * Đặt label nằm giữa 2 thẻ này * Trả về chuỗi html có dạng:   <open-html-tag> label <close-html-tag>  Nếu annot chỉ có thẻ mở   * Tạo thẻ label tiêng      * 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 |
|  | B1: sinh testcase từ luồng đã cho  B2: chuyển từng cặp  <input label, input element>  trong testcase thành html tương ứng  B3: đọc nội dung file html-template  B4: điền thông tin html ở B2 vào vị trí HTML\_REPLACE\_TAG trong nội dung file đã đọc ở B3  B5: ghi nội dung trên ra file html |