Storage Format

key (string)	value (string)
""	{"req1":"{}", "req2":"{"area1":"{}"}" "req3":"{"areaA":"{}", "areaB":"{}"}", "req4":"{"area_1": "{"course_1":"not satisfied"}" }" "req5":"{"area_A":"{"course_A": "satisfied", "course_B":"not satisfied", "course_C":"in progress"}", "area_B":"{"Course_A":"in progress", "Course_B":"satisfied", "course_C": "not satisfied"}"}" }

Why Use Map?

just [key, value],
myMap.keys(),
or myMap.values()

- readable iteration cno Object.keys (localStorage...)
- Can get size of data through .size property if needed (returns {'0','1','2',...})
- readable setting key value pairs
  (myMap.set('key','value') vs
  (myObj.key='value'))

getMapOfWebData()

{"req1" ⇒ "{}",
"req2" ⇒ "{"area1": "{}"}"
"req3" ⇒ "{"areaA": "{}",
"areaB": "{}"},
"req4" ⇒ "{"area_1":
"{"course_1": "not satisfied"}"
}"
"req5" ⇒ "{"area_A": "{"course_A":
"satisfied", "course_B": "not satisfied",
"course_C": "in progress"}",
"area_B": "{"Course_A: "in progress",
"Course_B": "satisfied", "course_C":
"not satisfied"}"}
}

key: String [requirement]
value: String [requirement data]
- areas
- courses
- status

getMapOfRequirementData('req5')

{"area_A" => "{"course_A": "satisfied",
"course_B": "not satisfied",
"course_C": "in progress"
}",

"area_B" => "{"course_A: "in progress",
"course_B": "satisfied",
"course_C": "not satisfied"
}"
}

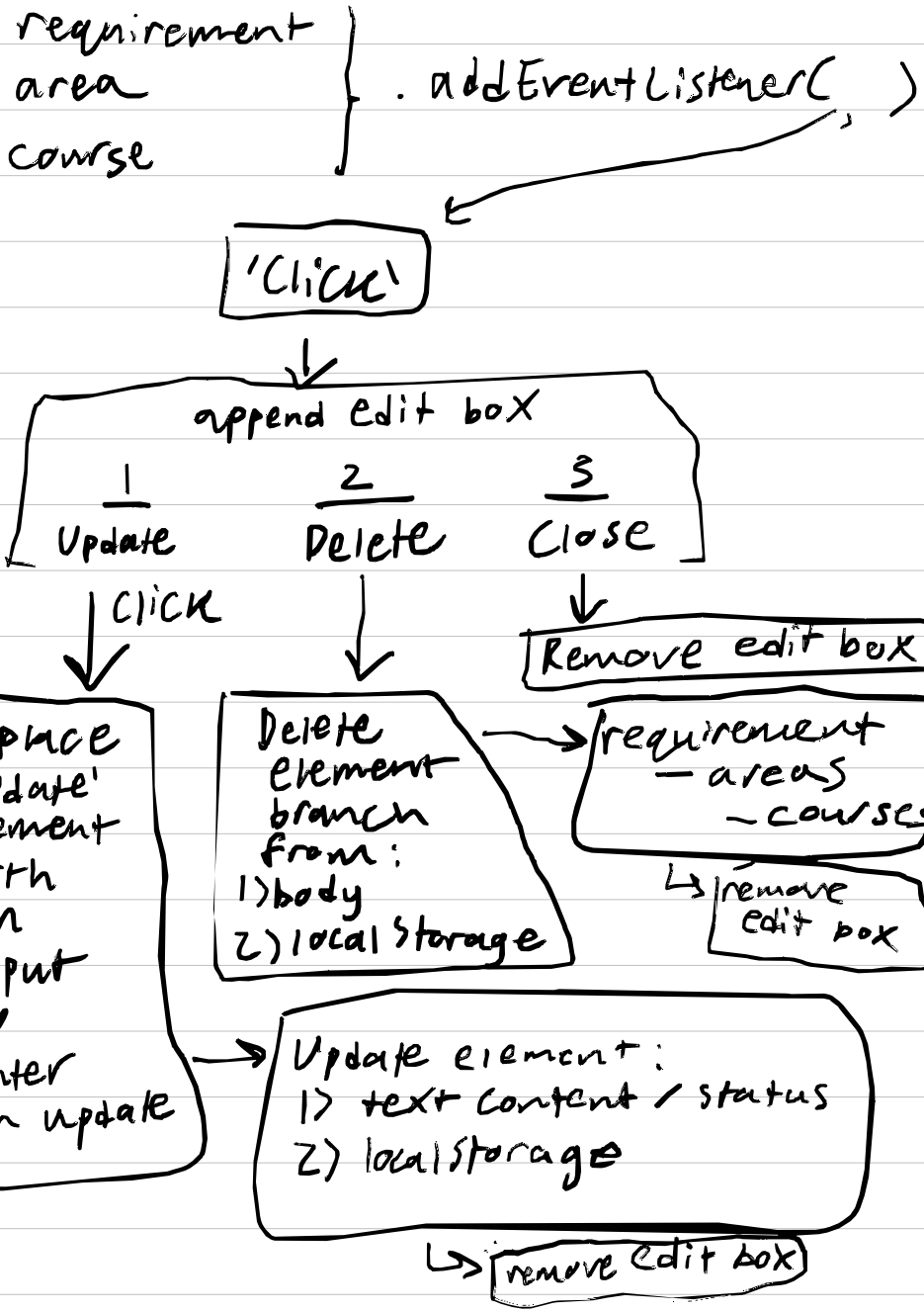
key: String [area]
value: String [area data]
- courses
- status

getMapOfAreaData('req_5', 'course_B')

{"course_A" => "in progress",
"course_B" => "satisfied",
"course_C" => "not satisfied"
}

key: string [course]
value: string [course data]
- status

Editing Elements



Deleting From localStorage

key (string)	value (string)
""	{"req1": {}, "req2": {"area1": "{}"} "req3": {"areaA": "{}", "areaB": "{}"}, "req4": {"area_1": {"course_1": "not satisfied"} } "req5": {"area_A": {"course_A": "satisfied", "course_B": "not satisfied", "course_C": "in progress"}, "area_B": {"Course_A": "in progress", "Course_B": "satisfied", "Course_C": "not satisfied"} } }

Deleting From localStorage cont.

Deleting requirements:

1) json = getJSONOfWebData() → {...}

2) delete json[deleted_requirement_string]

Deleting areas:

1) (see req)

2) delete json[requirement_string]
[deleted_area_string]

Deleting courses:

1) (see req)

2) delete json[requirement_string]
[area_string]
[deleted_course_string]

Updating localStorage

Updating requirements:

1) json = getJSONOfWebData() → {...}
2) updated_json = {}
3) for(const [req-str, data] of Object.entries(json))
   {
     if(req_str !== update_str){
        updated_json[req_str] = data;
     }
     else if (req_str === update_str){
        updated_json[update_str] = data;
     }
   }
4) stringified_update_json
   = JSON.stringify(updated_json);
5) localStorage.setItem('',
   stringified_updated_json);

Updating localStorage cont.

Updating Areas:

1) web_json = getJSONOfWebData() → {...}
2) requirement_json = web_json[requirement_str]
3) update_req_json = {}
4) for(const [area_str, data] of
   Object.entries(requirement_json)){
   if(area_str !== update_area_str){
   update_req_json[area_str] = data;
   }
   else if(area_str === update_area_str)
   {
   update_req_json[update_area_str]
   = data;
   }
   }
5) web_json[requirement_str] = update_req_json;
6) stringified_update_json
   = JSON.stringify(web_json)
7) localStorage.setItem(``,
   stringified_updated_json);

Updating localStorage cont.

Updating courses:

1) web_json = getJSONOfWebData() → {...}
2) req_json = web_json[req_str];
3) area_json = req_json[area_str];
4) update_area_json = {};
5) for(const [course_str, data] of
   Object.entries(area_json)){
      if(course_str !== update_course_str){
         update_area_json[course_str] = data;
      }
      else if(course_str===update_course_str)
      {
         update_area_json[update_course_str]
          = data;
      }
   }
6) req_json[area_str] = update_area_json;
7) web_json[req_str] = req_json;
8) stringified_update_json
     = JSON.stringify(web_json)
9) localStorage.setItem('',
     stringified_updated_json);