# 4주차 과제1 (todolist) 기능 명세

|  |  |  |
| --- | --- | --- |
| 기능 번호 | 세부 기능 설명 | 추가기능여부 |
| FR1 | 일정을 추가할 수 있어야 합니다. | N |
| FR2 | 일정을 삭제할 수 있어야 합니다. | N |
| FR3 | 일정 완료 표시를 할 수 있습니다. | Y |
| FR4 | 일정 생성 시 생성된 일정에 임의의 스타일을 배정할 수 있습니다. | Y |

# 구현

## 구현 사항

|  |  |  |
| --- | --- | --- |
| 기능 번호 | 완료여부  (O,X) | 세부사항 |
| FR1 | O | 하단에 일정을 입력하고 등록 버튼을 누르면 임의의 색상 메모지에 일정이 적혀 게시판에 등록됩니다. |
| FR2 | O | 메모지의 고정 핀을 클릭하면 일정을 삭제할 수 있습니다. |
| FR3 | O | 메모지의 내용을 클릭하면 내용에 선을 그어 완료여부를 표시할 수 있습니다. |
| FR4 | O | 메모지 추가 시 임의의 스타일 적용됩니다. |

## 웹 화면 구성

제목 : To-Do-List 전체 화면 (기본)

|  |  |
| --- | --- |
| 화면 | |
| 배지 3 단색으로 채워진배지 1 단색으로 채워진배지 단색으로 채워진 | |
| 설명 | ① 일정이 등록되는 게시판입니다. 입력된 일정은 화면과 같이 메모지에 작성되어 게시판에 등록됩니다. 일정은 최대 8개까지 입력됩니다.  ② 일정을 입력할 수 있는 곳입니다.  ③ 입력한 일정은 등록할 수 있는 버튼입니다. |

제목 : To-Do-List 전체 화면 (일정 등록)

|  |  |
| --- | --- |
| 화면 | |
| 배지 3 단색으로 채워진배지 단색으로 채워진배지 1 단색으로 채워진 | |
| 설명 | ① 해야 하는 일정을 입력합니다.  ② 일정을 등록하기 위해 등록 버튼을 클릭합니다.  ③ 화면과 같이 일정이 메모지와 함께 등록됩니다, 이때 메모지는 임의의 메모지와 메모지 핀 색상을 갖게 됩니다. |

제목 : To-Do-List 일부 화면 (일정 삭제)

|  |  |
| --- | --- |
| 화면 | |
| 배지 단색으로 채워진배지 1 단색으로 채워진 | |
| 설명 | ① 일정을 제거하기 위해 메모지의 핀을 클릭합니다.  ② 화면과 같이 게시판에서 메모지가 제거됩니다. |

제목 : To-Do-List 일부 화면 (일정 완료 표시)

|  |  |
| --- | --- |
| 화면 | |
| 배지 1 단색으로 채워진배지 단색으로 채워진텍스트, 스크린샷, 직사각형, 디자인이(가) 표시된 사진  자동 생성된 설명 | |
| 설명 | ① 일정 완료 표시를 위해 메모지의 내용을 클릭합니다.  ② 화면과 같이 메모지의 내용에 선으로 줄을 긋습니다. |

## 구현 내용

### HTML

|  |  |
| --- | --- |
| 파일명 | 구현 내용 |
| index.html | To-do-List 화면을 구성하고 css와 js 파일을 적용합니다. |

### CSS

|  |  |
| --- | --- |
| 파일명 | 구현 내용 |
| base/\_reset.scss | 브라우저의 기본 스타일을 초기화합니다. |
| base/\_global.scss | 공통적으로 적용되는 스타일을 적용합니다. |
| abstracts/\_mixins.scss | 호출해서 사용할 mixin 스타일들을 정의합니다. |
| abstracts/\_variables.scss | 자주 사용하는 색상을 변수로 정의합니다.   * $white, $black, $background: 기본 색상 정의 * $bright-wood, $dark-wood: 나무 색상 정의 * $bright-line, $dark-line: 게시판 줄 색상 정의 * $paper+n : 메모지 색상을 정의 * $pin+n: 메모지 핀의 색상을 정의 |
| pages/\_page.scss | 화면을 구성하는 세부 스타일을 정의합니다.   * line-container: 게시판 걸이에 대한 스타일 * nail, line: 게시판 줄, 못에 대한 스타일 * board-container: 게시판 전체 스타일 * memo-container: 메모판 목록 스타일 (grid) * memo: 메모에 대한 스타일 * memo-head, “-pin, “-body: 메모의 윗부분, 메모 핀, 메모지 내용에 대한 스타일 * memo—clear: 일정 완료에 대한 스타일 * memo-type+n: 메모별 색상에 대한 스타일 * input-container: 입력란에 대한 스타일 |
| main.scss | \_reset.scss, \_global.scss, \_page.scss 파일을 통합적으로 가져옵니다. |
| main.css | scss 파일을 css로 전환하여 화면에 스타일을 적용합니다. |

### JavaScript: 변수, 함수, 클래스/속성

파일 명 : main.js

|  |  |
| --- | --- |
| 함수명 | 구현 내용 |
| main() | 함수 인자 설명: 없음  리턴값: 없음  동작 설명  **최초 메모를 등록하고 전체적인 이벤트를 등록합니다.**  createMemo(), addMemo()를 통해 최초 메모를 생성하고 게시판에 등록합니다. 이후 addEvents()를 통해 이벤트를 등록해줍니다. |

파일 명 : event/addEvent.js

|  |  |
| --- | --- |
| 함수명 | 구현 내용 |
| addEvents() | 함수 인자 설명: 없음  리턴값: 없음  동작 설명  **등록 버튼 이벤트와 최초 메모지 핀에 대한 클릭 이벤트를 등록합니다.**  등록 버튼인 submitBtn에 클릭 이벤트로 submitHandler()를 등록합니다.  메모지 핀인 pinDiv에 클릭 이벤트로 pinClickHandler()를 등록합니다. |
| addPinEvent(pinDiv) | 함수 인자 설명  pinDiv: 클릭된 메모지 핀  리턴값: 없음  동작 설명  **메모지 핀에 대한 클릭 이벤트를 등록합니다.**  메모지 핀 pinDiv에 클릭 이벤트로 pinClickHandler()를 등록합니다. (후에 생성될 메모지 핀들에 이벤트를를 등록하기 위함임) |
| addMemoEvent(memo) | 함수 인자 설명  memo: 클릭된 메모지  리턴값: 없음  동작 설명  **메모지에 대한 클릭 이벤트를 등록합니다.**  메모지 memo에 클릭 이벤트로 memoClickHandler()를 등록합니다. (후에 생성될 메모들에 이벤트를를 등록하기 위함임) |

파일 명 : event/eventHandler.js

|  |  |
| --- | --- |
| 함수명 | 구현 내용 |
| submitHandler() | 함수 인자 설명: 없음  리턴값: 없음  동작 설명  **게시판에 메모지 등록이 가능한 상태인지 확인한 후, 입력된 내용을 메모지에 담아 게시판에 추가합니다.**  checkMemoList()를 통해 먼저 게시판에 메모지를 등록할 수 있는지 확인합니다. 게시판에 메모지가 8개 등록되어 있다면 더 이상 메모지를 추가할 수 없습니다.  등록이 가능한 상태라면, 입력된 내용을 가져와 createMemo()를 통해 메모를 생성한 뒤 게시판에 메모지를 addMemo()를 통해 추가해 줍니다.  메모지를 등록한 후 입력 칸에 내용을 비웁니다. |
| pinClickHandler(target) | 함수 인자 설명  target: 클릭된 메모지 핀  리턴값: 없음  동작 설명  **메모지 게시판에서 클릭 된 메모지 핀의 메모지를 제거합니다.**  target의 부모의 부모 요소를 가져와 (메모지 요소) 해당 요소가 존재하면 removeMemo()를 통해 제거합니다. |
| memoClickHandler(target) | 함수 인자 설명  target: 클릭된 메모지 내용  리턴값: 없음  동작 설명  **메모지 내용을 클릭하면 내용에 줄을 긋습니다. 한 번 더 클릭 시 줄이 제거됩니다.**  target에 “memo—clear” 클래스가 있는지 확인합니다. 있다면 제거하고 없다면 추가합니다. |

파일 명 : utils/memo.js

|  |  |
| --- | --- |
| 함수명 | 구현 내용 |
| checkMemoList() | 함수 인자 설명: 없음  리턴값: true / false  동작 설명  **메모 게시판에 메모지가 8개 인지 확인합니다.**  메모지 게시판 요소를 가져와 자식 요소가 8개 인지 확인합니다. 8개이면 false를 아니면 true를 반환합니다. |
| createMemo(content) | 함수 인자 설명  content: 메모지 내용에 입력될 내용  리턴값  memoDiv: 생성된 메모지  동작 설명  **메모지 요소를 생성하고 속성 및 내용을 추가하여 게시판에 추가할 메모지를 구성합니다.**  메모지 전체, 메모지 윗부분, 메모지 핀, 메모지 몸통, 메모지 내용부분을 요소로 생성하고 각각 요소에 맞는 클래스를 추가합니다.  내용부분에는 readonly 속성을 추가해 내용을 수정할 수 없게 하고, 입력된 내용 content를 추가합니다.  생성된 메모지의 제거, 완료 기능을 할당하기 위해 addPinEvent(), addMemoEvent()로 이벤트를 추가합니다.  메모지의 형태로 각 요소를 구성해준 다음 메모지의 색상을 decorateMemo()를 통해 지정합니다. 메모지가 게시판에 위치할 장소도 setGridPosition()을 통해 지정합니다.  완성된 메모지 요소를 반환합니다. |
| decorateMemo(memoDiv) | 함수 인자 설명  memoDiv: 생성한 메모지 요소  리턴값: 없음  동작 설명  **메모지의 임의의 스타일을 클래스로 추가합니다.**  1~6까지의 숫자를 난수로 생성합니다.  생성된 난수를 번호로 삼아 “memo-type”+n 클래스를 memoDiv에 추가합니다. |
| addMemo(memoEl) | 함수 인자 설명  memoEl: 생성된 메모지 요소  리턴값: 없음  동작 설명  **메모 게시판에 메모를 추가합니다.**  생성이 완료된 메모 요소 memoEl이 존재한다면 해당 요소를 메모 게시판 요소에 추가합니다. |
| removeMemo(memoEl) | 함수 인자 설명  memoEl: 전달된 메모지 요소  리턴값: 없음  동작 설명  **클릭된 메모지를 메모 게시판에서 제거합니다.**  전달받은 메모 요소 memoEl이 존재한다면 해당 요소를 메모 게시판 요소에서 제거합니다. |

파일 명 : utils/grid.js

|  |  |
| --- | --- |
| 함수명 | 구현 내용 |
| setGridPosition(div) | 함수 인자 설명  div: 배치될 메모지 요소  리턴값: 없음  동작 설명  **게시판에 추가될 메모지 요소의 위치를 지정합니다.**  checkGridArray()를 통해 현재 게시판의 grid 영역에서 메모지들이 등록되어 있는 공간을 확인합니다.  다음으로 현재 비어져 있는 첫 번째 공간을 찾아 새 메모지 요소 div의 gridColumn, gridRow 속성으로 지정합니다. |
| checkGridArray() | 함수 인자 설명: 없음  리턴값  checkArray: 현재 메모지들이 배치되어 있는 그리드 공간의 상태  동작 설명  **메모지 게시판에 메모지들이 어디에 등록되어 있는지를 확인하고 확인된 값을 반환합니다.**  0으로 꽉 찬 2x4 배열 checkArray를 생성하고 게시판에 등록되어 있는 메모지 요소들을 다 가져옵니다.  가져온 메모지 요소들의 gridColumn, gridRow를 확인해 배치가 되어 있는 곳은 checkArray에서 1로 표기합니다.  모든 메모지 요소를 확인한 뒤 checkArray를 반환합니다. |

# 테스트 결과

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | 테스트 제목 | 테스트 절차 | 관련 FR | 테스트 결과(O,X) |
| 1 | 메모지 추가 테스트 | 1. 내용 입력 칸에 내용을 입력한다.  2. 등록 버튼을 누른다.  3. 임의의 색상이 매정 된 메모지가 게시판에 등록이 되는지 확인한다. | FR1,  FR4 | O |
| 2 | 메모지 삭제 테스트 | 1. 메모지의 핀을 클릭한다  2. 메모지가 게시판에서 삭제된다.  3. 메모지가 삭제된 자리는 그대로 비어있는다. | FR2 | O |
| 3 | 메모지 완료 테스트 | 1. 메모지의 내용을 클릭한다.  2. 메모지의 내용에 줄이 그어지는지 확인한다.  3. 메모지의 내용을 한번 더 클릭한다.  4. 메모지의 내용에 줄이 사라지는지 확인한다. | FR3 | O |
| 4 | 메모지 꽉 채운 뒤 추가 테스트 | 1. 메모지를 8개까지 채운다.  2. 입력 칸에 내용을 입력한 뒤 등록 버튼을 누른다.  3. 메모지가 추가가 안되는지 확인한다. | FR1 | O |
| 5 | 메모지 꽉 채운 뒤 메모지 몇 개 삭제 후 추가 테스트 | 1. 메모지를 8개까지 채운다.  2. 임의의 메모지를 삭제 한다.  3. 메모지를 추가할 때 빈 공간 중 가장 앞 공간부터 채워지는지 확인한다. | FR1,  FR2 | O |

# 집중 피드백 받고 싶은 사항

|  |  |
| --- | --- |
| 번호 | 내용 |
| 1 | 처음에는 메모지가 제거될 때마다 남은 메모지들이 앞으로 한 칸씩 전진 되길래, 각 위치에 고정시키기 위해 각 메모지 별로 grid 속성 column과 row를 지정했습니다. 대신 이 방법을 쓰면 매번 새로운 메모지를 빈 곳에 추가할 때마다, grid 영역에서 비어 있는 공간을 알아내야 했기에 각 메모지들의 grid 속성 column과 row를 배열에 담아 확인해야 했습니다. 뭔가 비효율 적인 방법은 인 것 같아 더 좋은 방법이 있는지 여쭤봅니다. |