휴먼 컴퓨터 인터페이스

#3. 구현 완성도 개선

2018.05.27.

컴퓨터소프트웨어학과
2014726070
3학년 이상진

개요

오픈소스 라이브러리

- math.js / jquery
- jquery-ui-1.10.0.min (슬라이드 바)

기존 설계와의 변경사항

- **그래프 그리기 기능 추가**
  - 계산 기록상에 그래프로 그릴 수 있는 함수가 자동으로 그래프 목록에 등록
  - 그래프 모드에서 그래프 목록 확인 가능
  - 계산 기록을 삭제하면 그래프 목록도 초기화
  - 그래프 모드에서 그래프 목록을 선택하면 해당 함수의 그래프를 그릴 수 있음
  - 그래프 스케일 조정 가능
- 세부적 개선
  - 버튼 및 컨테이너 색 변경
  - 폰트, 글자색 변경, 오타 수정

그래프 그리기 구현 방식

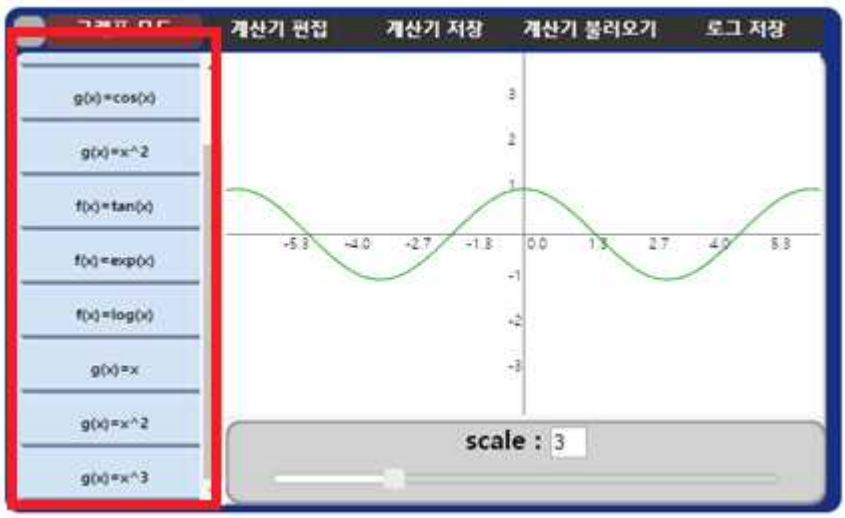
1. 그래프 함수 저장

- 계산 기록상에 그래프로 그릴 수 있는 함수가 자동으로 그래프 목록에 등록

```
if($(this).text() == 'EV')
{
    try
    {
        textVal = displayValue;
        displayValue = parser.eval(displayValue).toString();
        var tokens = displayValue.split(' ');
        if(tokens[0] == 'function')
        {
            graphBuf.push(textVal);
            displayValue = tokens[0];
        }
```

- 미리 선언해 둔 배열 형 변수에 함수 등록

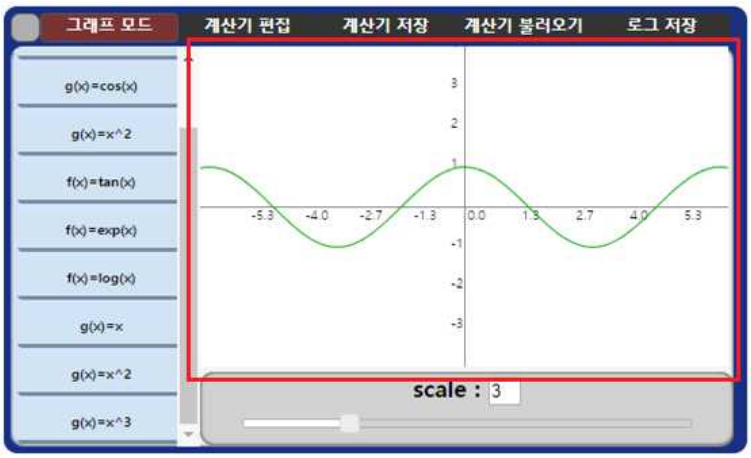
```
function load_graphlist(value, index, ar){
    var gl = document.createElement('gl');
    gl.style.width = "130px";
    gl.style.font = "bold 10px sans-serif";
    gl.style.boxShadow = "3px 3px 0 #789";

    gl.classList.add("graphlistVal");

    gl.innerText = value;
    gl.id = "gl" + index;

    document.getElementById('graphlist').appendChild(gl);
```



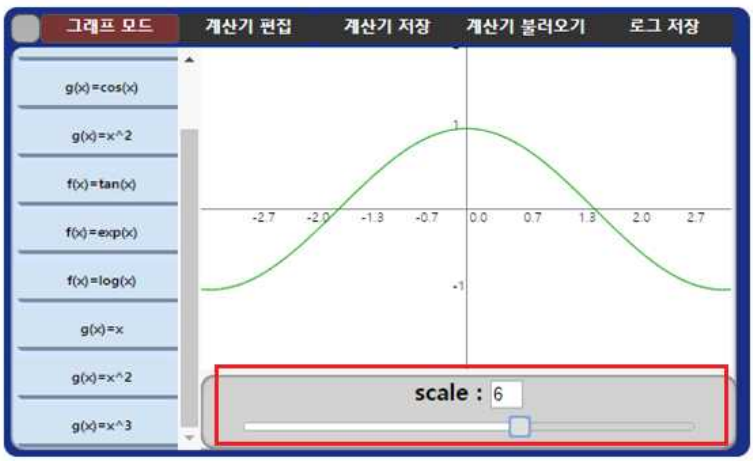
- 그래프 모드에서 등록된 그래프 목록을 확인 가능
- 계산기록 삭제 시 그래프 목록도 삭제

2. 그래프 그리기 / scale 조정

```
$('gl').click(function(e){
    var index = $(this).attr('id').split('gl');
    if(checkedGL != index[1]*1)
    {
        for(var i=0; i<graphBuf.length;i++)
            document.getElementById('gl' + i).style.background = "#CDE";
        $(this).css('background-color',"#AAA");
    }
    checkedGL = index[1]*1;
    draw($(this).text(),$("#data-value").val()*10);
});
```



- 계산 목록을 클릭하면 등록된 함수와 지정된 scale로 canvas에 그래프를 그림



- 하단의 슬라이드 바를 통해 scale 조정 가능 (총 9단계)

```
$(function() {
$("#slider-bar").slider({
    range:'max',
    value:5,
    min: 1,
    max: 9,
    step: 1,
    slide: function( event, ui ) {
        $("#data-value").val(ui.value );
        if(checkedGL != -1)
            draw(graphBuf[checkedGL],$("#data-value").val()*10);
    }});
$("#data-value").val($("#slider-bar").slider("value"));
});
```

시연 영상

https://youtu.be/HHn8-OTuWqs

고찰

구현 측면에서 성공적인 부분과 실패한 부분

성공적인 부분 : 그래프 그리기 / 그래프 목록

- 이번 구현의 가장 중심적인 목표로
  어떻게 하면 사용자에게 편리하게 그래프를 그릴 수 있을까 고민했고
  일부러 그래프를 그리기 위해 함수를 등록하지 않아도
  지금까지의 계산기록에서 그래프가 필요하다면 그릴 수 있게 구현하였다.

실패한 부분 : 기존 계산기 개선 - 연산자의 이름과 표시를 다르게 하는 것

- 저번 구현 때 뽑았던 개선사상으로 이번에 시도해보았지만
  구현의 기본이 연산자 이름을 텍스트로 그대로 사용 하는 것 이여서
  너무 많은 부분을 바꿔야 하는 문제가 생겨 구현에 실패했다.

사용성 측면에서 긍정적인 측면과 부정적인 측면

긍정적 측면

- 사용자가 그래프를 그리기 위해 새로 함수를 등록할 필요 없다.
- 계산 기록의 모든 함수를 그릴 수 있으므로, 기록에 따른 변화 등을 바로바로 확인 가능하다.
- scale을 사용자가 조정할 수 있다.

부정적 측면

- scale이 9단계로 나뉘어 조정할 수 있지만, 정의역과 치역을 사용자 임의로 설정할 수는 없다.
- 그래프를 겹쳐서 그릴 수 없다.