#객체 이벤트 등록p245~

객체에 접근할 때 클래스 속성값을 이용하는 방식과 아이디값을 이용하는 방식의 각각 장단점

클래스 속성값으로 접근하는 방식

장점:

동일한 클래스를 가진 여러 개의 객체를 선택할 수 있습니다.

스타일링을 클래스로 지정하는 경우 유용합니다.

단점:

동일한 클래스를 가진 여러 개의 객체를 선택하기 때문에 선택되는 객체의 수가 예측하기 어렵습니다.

클래스를 이용하여 접근하는 경우, 선택된 객체들의 상태나 속성이 모두 동일하다는 보장이 없습니다.

아이디값으로 접근하는 방식

장점:

아이디는 문서 내에서 유일한 값을 가져야 하기 때문에 선택되는 객체가 명확합니다.

아이디값을 이용하여 객체에 접근하는 것이 더 빠릅니다.

단점:

하나의 아이디값만 가질 수 있기 때문에, 하나의 객체에만 적용됩니다.

스타일링을 아이디값으로 지정하는 경우, 해당 객체의 스타일링을 변경하고자 할 때 유용하지만, 다른 객체와의 공통적인 스타일링을 지정하는 것은 어렵습니다.

따라서 선택자를 결정할 때, 객체의 특성과 상황에 맞게 선택해야 합니다. 일반적으로는 클래스 속성값으로 접근하는 방식이 더 유용하며, 아이디값은 유일한 객체를 지정할 때 사용하는 것이 좋습니다.

이벤트 등록방법

1.단독 이벤트 등록 : $(“이벤트대상”).이벤트 등록 메서드(function(){자바스크립트코드;})

2.그룹 이벤트 등록 : on()메소드를 사용하여 이벤트를 등록

$(“이벤트대상”).on({“이벤트 종류1 이벤트종류2 이벤트종류3” : function(){ 자바스크립트코드;}

, ”이벤트종류4” :function(){자바스크립트코드;}});

-On 메소드 이용하는이유 : On()메소드를 이용하면 선택한 요소에 이벤트를 등록한 이후에도 새롭게 생성(없는데 코드상으로 추가된경우)된요소에 이벤트를 적용할수있다

-$(“.클래스명 a”).addClass(추가 클래스명 b): 기존의 클래스명 a -> 추가된 클래스명 a b

->선택자로 선택할시 $(“.a.b”)해주면 a b 인 클래스명을 선택한다

-코드상으로 html조작하는방법

Var btn2 = $(“<p><button class=\btn\_2\”>버튼2</button></p>”);

$(“선택할요소”).append(btn2); 선택한부분의 하위에 추가시켜준다

-$(“선택요소”).on(“이벤트”, fucnction(e){ e.preventDefault();}) : preventDefault()메소드를 사용하여 해당 선택요소의 사전에 연결된 이벤트동작을 작동하지않게한다



P254 $(document).ready와 $(window).load의차이

$(function(){

            $(".btn1").click(function(){

                if($(".btn1").parent().next().css("color") == "rgb(255, 0, 0)"){

                    $(".btn1").parent().next().css({"color" : "rgb(0, 0, 0)"});

                }else if($(".btn1").parent().next().css("color")=="rgb(0, 0, 0)"){

                    $(".btn1").parent().next().css({"color" : "#f00"})

                }

                //css("color")는 값을 rgb()값으로 가져오기때문에 비교하기위해서는 "rgb()"로비교해주어야 한다

            });

            console.log($(".btn1").parent().next().css("color"));

            $(".btn2").on("click",function(){

                $(".btn1").parent().next().css({"color":"red"})

                $(".btn2").parent().next().css({"color":"red"})

            })

        })

-

<head>

    <title>Document</title>

    <script src="js/jquery.js"></script>

    <script>

        $(function(){

            var beforemenu = null;

            $(".gnb li>button>a").on("mouseover focus", function(){

                if(beforemenu)beforemenu.css({background : "none"});

                //$(".gnb >button> a").css("background","none");

                $(this).css("background-color","#0ff");

                console.log($(this).text());

                beforemenu = $(this);

            });

            $(".gnb>button>a").on("mouseleave",function(){

                if(beforemenu)beforemenu.css({background : "none"});

                //$(this).css("background","none");

            });

        })

    </script>

</head>

<body>

    <ul class="gnb">

        <li>

            <button><a href="#">menu1 </a></button>

        </li>

        <li>

            <button><a href="#">menu2 </a></button>

        </li>

        <li>

            <button><a href="#">menu3 </a></button>

        </li>

        <li>

            <button><a href="#">menu4 </a></button>

        </li>

    </ul>

</body>

-

<head>

    <title>Document</title>

    <script src="js/jquery.js"></script>

    <script>

        $(function () {

            $(".btn1").on({"mouseover focus": function () {

                //focus는 키보드접근성을 고려할때 가팅 사용하거나 input태그의 값을 유효성검사할때 틀리다면 focus시켜도 해당 위치로 이동시켜준다

                $(".btn1").parent().next().css({ "color": "#f00" });}

                ,"mouseout blur" : function(){$(".btn1").parent().next().css({ "color": "#000" });}

            });

            var btn3 = '<p><button class="btn3">버튼3</button></p>'

            $("#wrap").children("#1").next().after(btn3);// wrap아이디의 children해당 아이디태그의 종료태그후 삽입 하지만 p태그안의 p태그는 개별로 인식된다는점

            //$(".btn1").parent().after(btn3);//append는 종료태그전에 prepend는 시작태그바로이후에 after는 선택자이후에 삽입해준다

            $(".btn3").on({"mouseover focus": function(){

                $(".btn1").parent().next().css("color","#0f0");

                $(".btn2").parent().next().css("color","#0f0");

            },"mouseout " : function(){

                $(".btn1").parent().next().css("color","#000");

                $(".btn2").parent().next().css("color","#000");

            }

        })

        })

    </script>

</head>

<body>

    <div id="wrap">

        <p id="1">

            <button class="btn1">버튼1</button>

        </p>

        <p>내용1</p>

        <p>

            <button class="btn2">버튼2</button>

        </p>

        <p>내용2</p>

    </div>

</body>