

# JQUERY를 이용한 웹프로그램 구현

발행일 : 2023.01.01  
변경일 : 2023.01.01  
저자 : 박수민  
이메일 : [foxman12@hanmail.net](mailto:foxman12@hanmail.net)  
발행처 : 휴먼교육

# 목차

---

<b>01. JQuery</b>	<b>3</b>
> 01. jquery 란?	3
> 02. jquery CDN?	3
> 03. jquery 문법	4
> 06. 다양한 선택자	7
> 07. DOM 요소 읽고 쓰기	12
> 07. css관련 메소드 사용법	14
> 07. 다양한 이벤트	16
> 08. jquery 효과	21
> 09. html조작	24
> 10. dom조작	27
> 11. jquery 프로젝트1	35
> 12. jquery 프로젝트2	39
> 13. jquery 프로젝트3	42
> 14. jquery 프로젝트4	45
> 15. jquery 프로젝트5	48
> 16. ajax	53
동네예보 > 시간별 예보	63

# 01. JQuery

---

## > 01. jquery 란?

---

javascript라이브러리로 HTML/DOM 조작, CSS 조작, HTML 이벤트 방법, 효과 및 애니메이션, ajax 등을 쉽게 구현하는 용도로 사용한다.

---

## > 02. jquery CDN?

---

jQuery CDN은 Content Delivery Network의 약어로, jQuery 라이브러리를 제공하는 서비스입니다.

jQuery를 사용하고 싶다면 다음 처럼 상단에 jquery cdn를 기술해 주면 된다.

```
<head>  
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js">  
</script>  
</head>
```

---

## > 03. jquery 문법

---

jquery 문법은 다음과 같다. `$()`안에 css선택자를 사용하여 원하는 엘리먼트를 찾은 다음 `.`를 찍어 어떻게 처리할 것인지 처리할 메소드를 기술한다. 찾아서 숨기고, 찾아서 보여주고 찾아서 어떻게 하고 하는것이 jquery의 핵심 기능이다.

```
$(“선택자”).실행();
```

`$`는 jquery함수 이름이다.

다음은 버튼을 누르면 `div`가 토글되는 예제이다.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>jQuery 토글 예제</title>
    <!-- jQuery 라이브러리를 불러옵니다. -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        $(document).ready(function(){
            // 버튼 클릭 이벤트 처리
            $("#toggleButton").click(function(){
                // ID가 "myDiv"인 div 요소의 가시성을 토글합니다.
                $("#myDiv").toggle();
            });
        });
    </script>
    <style>
        /* div의 초기 상태는 숨겨진 상태입니다. */
        #myDiv {
            display: none;
            width: 200px;
            height: 200px;
            background-color: lightblue;
            text-align: center;
            line-height: 200px;
        }
    </style>
</head>
<body>
    <!-- 버튼을 추가하여 div 요소를 토글합니다. -->
    <button id="toggleButton">토글</button>
    <!-- 토글할 div 요소 -->
    <div id="myDiv">이 div는 토글됩니다.</div>
```

```
</body>
</html>
```

다음은 다양한 선택자를 사용한 예제이다. `.hide()` 찾은 엘리먼트를 숨기라는 이야기이다.

`$(this).hide()` - 현재 요소를 숨김

`$("p").hide()` - 모든 `<p>` 요소를 숨김

`$(".test").hide()` - `class="test"`인 모든 요소를 숨김

`$("#test").hide()` - `id="test"`인 요소를 숨김

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>jQuery 예제</title>
    <!-- jQuery 라이브러리를 불러옵니다. -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        $(document).ready(function(){//window.onload 비슷 하다고 생각하면 됨
            // 현재 요소를 숨기는 버튼 클릭 이벤트 처리
            $("#hideThisButton").click(function(){
                $(this).hide();
            });
            // 모든 <p> 요소를 숨기는 버튼 클릭 이벤트 처리
            $("#hidePButton").click(function(){
                $("p").hide();
            });
            // class="test"를 가진 모든 요소를 숨기는 버튼 클릭 이벤트 처리
            $("#hideClassButton").click(function(){
                $(".test").hide();
            });
            // id="test"를 가진 요소를 숨기는 버튼 클릭 이벤트 처리
            $("#hideIdButton").click(function(){
                $("#test").hide();
            });
        });
    </script>
    <style>
        /* div의 초기 상태는 숨겨진 상태입니다. */
        #test {
```

```

        display: none;
        width: 200px;
        height: 200px;
        background-color: lightblue;
        text-align: center;
        line-height: 200px;
    }
</style>
</head>
<body>
<!-- 버튼을 추가하여 각각의 기능을 실행합니다. -->
<button id="hideThisButton">$(this).hide()</button>
<button id="hidePButton">$("p").hide()</button>
<button id="hideClassButton">$(".test").hide()</button>
<button id="hideIdButton">"#test").hide()</button>
<!-- 토글할 div 요소와 class="test"를 가진 요소 -->
<div id="test">이 요소는 숨겨집니다.</div>
<p>이것은 문단입니다.</p>
<p class="test">이것도 문단입니다. (class="test" 포함)</p>
<p class="test">이것도 문단입니다. (class="test" 포함)</p>
</body>
</html>

```

window.onload와 비슷한 의미의 이벤트를 다음과 같이 다양한 방법으로 기술 할 수 있다.

```

$(document).ready(function(){
    // jQuery methods go here...
});

$(function(){
    // jQuery methods go here...
});

```

---

## > 06. 다양한 선택자

---

`$(“ul li:first”)`: <ul>의 자손중 <li> 요소를 찾아 찾은 첫번째 li를 선택합니다.

`$(“ul li:first-child”)`: <ul>의 자손중 <li> 요소를 찾아 찾은 li가 첫번째 자식일 경우에만 선택합니다.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
    <title>jQuery 예제</title>
    <style>
        /* 스타일링을 위한 간단한 CSS */
        ul {
            border: 1px solid #ccc;
            padding: 10px;
            margin: 10px;
        }
        li {
            margin: 5px;
            padding: 5px;
            border: 1px solid #ddd;
        }
    </style>
</head>
<body>
    <!-- 첫 번째 예제: $("ul li:first") -->
    <h2>첫 번째 예제: $("ul li:first")</h2>
    <ul>
        <p></p>
        <li>첫 번째 항목1</li>
        <li>두 번째 항목1</li>
        <li>
            <ul>
                <li>중첩된 첫 번째 항목1</li>
                <li>중첩된 두 번째 항목1</li>
            </ul>
        </li>
    </ul>
    <!-- 두 번째 예제: $("ul li:first-child") -->
    <h2>두 번째 예제: $("ul li:first-child")</h2>
    <ul>
        <li>첫 번째 항목2</li>
```

```

<li>두 번째 항목2</li>
<li>
    <ul>
        <li>중첩된 첫 번째 항목2</li>
        <li>중첩된 두 번째 항목2</li>
    </ul>
</li>
</ul>
<script>
    // jQuery를 사용하여 선택자를 적용하고 결과를 콘솔에 출력
    console.log($(".ul li:first").text());
    console.log($(".ul li:first-child").text());
//번째 항목1
// 중첩된 첫 번째 항목1첫 번째 항목2중첩된 첫 번째 항목2
</script>
</body>
</html>

```

`$("")`: 웹 페이지의 모든 HTML 요소를 선택합니다.

`$(this)`: 선택된 현재 HTML 요소를 선택합니다.

`$("p.intro")`: class 속성이 "intro"인 모든 `<p>` 요소를 선택합니다.

`$("p:first")`: 첫 번째 `<p>` 요소를 선택합니다.

`$("[href]")`: href 속성을 가진 모든 요소를 선택합니다.

`$("a[target='_blank']")`: target 속성 값이 "\_blank"인 모든 `<a>` 요소를 선택합니다.

`$("a[target!='_blank']")`: target 속성 값이 "\_blank"가 아닌 모든 `<a>` 요소를 선택합니다.

`.css()` 스타일 시트를 변경하거나 읽어 오라는 의미이다..

`.text()` 텍스트를 읽어오거나 변경하라는 의미이다.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>jQuery 선택자 예제</title>
    <!-- jQuery 라이브러리를 불러옵니다. -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        $(document).ready(function(){
            // 모든 요소를 선택하고 스타일을 변경합니다.
            $("*").css("border", "2px solid red");
            // 버튼 클릭 시 현재 버튼의 텍스트를 변경합니다.
            $("button").click(function(){
                $(this).text("클릭됨");
            });
        });
    </script>

```

```

        // class가 "intro"인 모든 <p> 요소를 선택하고 스타일을 변경합니다.
        $("p.intro").css("background-color", "yellow");
        // 첫 번째 <p> 요소를 선택하고 스타일을 변경합니다.
        $("p:first").css("font-weight", "bold");
    });
</script>
<style>
    /* 예제를 이해하기 쉽게 몇 가지 스타일을 추가합니다. */
    p { padding: 10px; margin: 10px; }
</style>
</head>
<body>
    <!-- 웹 페이지의 모든 요소에 빨간 테두리가 적용됩니다. -->
    <h1>제목</h1>
    <p>내용</p>
    <a href="#">링크</a>
    <!-- 버튼을 클릭하면 해당 버튼의 텍스트가 "클릭됨"으로 변경됩니다. -->
    <button>클릭</button>
    <button>클릭</button>
    <!-- class가 "intro"인 모든 <p> 요소와 첫 번째 <p> 요소에 스타일이 적용됩니다.
-->
    <p class="intro">첫 번째 "intro" 클래스를 가진 단락</p>
    <p class="intro">두 번째 "intro" 클래스를 가진 단락</p>
    <p>다른 단락</p>
</body>
</html>

```

3.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>다양한 jQuery 선택자 예제</title>
    <!-- jQuery 라이브러리를 불러옵니다. -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script>
    $(document).ready(function(){
        // 첫 번째 순서 목록(<ul>) 안의 첫 번째 목록 항목(<li>)을 선택
        $("ul:first li:first").css("color", "blue");
        // 모든 순서 목록(<ul>) 안의 첫 번째 목록 항목(<li>)을 선택
        $("ul li:first-child").css("font-weight", "bold");
        // href 속성을 가진 모든 요소를 선택하고 스타일을 변경합니다.
        $("[href]").css("text-decoration", "underline");
        // target 속성 값이 "_blank"인 모든 <a> 요소를 선택
        $("a[target='_blank']").css("color", "green");
    });

```

```

        // target 속성 값이 "_blank"가 아닌 모든 <a> 요소를 선택
        $("a[target!='_blank']").css("color", "red");
    });
</script>
</head>
<body>
    <ul>
        <li>첫 번째 목록 항목</li>
        <li>두 번째 목록 항목</li>
    </ul>
    <ul>
        <li>다른 목록 항목</li>
        <li>또 다른 목록 항목</li>
    </ul>
    <!-- href 속성을 가진 링크 요소 -->
    <a href="#">링크 1</a>
    <a href="#">링크 2</a>
    <!-- target 속성 값이 "_blank"인 링크 요소 -->
    <a href="#" target="_blank">새 창 열림</a>
    <a href="#" target="_blank">새 창 열림</a>
    <!-- target 속성 값이 "_blank"가 아닌 링크 요소 -->
    <a href="#">현재 창 열림</a>
    <a href="#">현재 창 열림</a>
</body>
</html>

```

`$(":button")`: 모든 `<button>` 요소와 `type` 속성이 "button"인 `<input>` 요소를 선택합니다.

`$("tr:even")`: 짝수 번째 테이블 행(`<tr>`) 요소를 선택합니다.

`$("tr:odd")`: 홀수 번째 테이블 행(`<tr>`) 요소를 선택합니다.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>다양한 jQuery 선택자 예제</title>
    <!-- jQuery 라이브러리를 불러옵니다. -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        $(document).ready(function(){
            // 모든 버튼 요소와 type이 "button"인 입력 필드를 선택
            $(":button").css("background-color", "blue");
            // 짝수 번째 테이블 행을 선택하고 스타일을 변경합니다.
            $("tr:even").css("background-color", "lightgray");
            // 홀수 번째 테이블 행을 선택하고 스타일을 변경합니다.
            $("tr:odd").css("background-color", "lightblue");
        });
    </script>

```

```
    });
  </script>
</head>
<body>
  <!-- 버튼 요소 -->
  <button>버튼 1</button>
  <button>버튼 2</button>
  <!-- type이 "button"인 입력 필드 -->
  <input type="button" value="입력 필드 1">
  <input type="button" value="입력 필드 2">
  <!-- 테이블 요소 -->
  <table>
    <tr>
      <td>테이블 셀 1-1</td>
      <td>테이블 셀 1-2</td>
    </tr>
    <tr>
      <td>테이블 셀 2-1</td>
      <td>테이블 셀 2-2</td>
    </tr>
  </table>
</body>
</html>
```

---

## > 07. DOM 요소 읽고 쓰기

---

`text()`: 이 메소드는 선택한 요소의 텍스트 내용을 가져오거나 설정합니다.

```
텍스트 가져오기: $("선택자").text();
var.textContent = $("p").text();
console.log(textContent);
텍스트 설정하기: $("선택자").text("새로운 텍스트");
$("p").text("이것은 새로운 텍스트입니다.");
```

`val()`: 이 메소드는 폼 요소(예: `input`, `select`, `textarea`)의 사용자 입력 값을 가져오거나 설정합니다.

```
값 가져오기: $("폼요소선택자").val();
var inputValue = $("#myInput").val();
console.log(inputValue);
값 설정하기: $("폼요소선택자").val("새로운 값");
$("#myInput").val("새로운 입력값");
```

`html()`: 이 메소드는 선택한 요소의 HTML 내용을 가져오거나 설정합니다.

```
HTML 내용 가져오기: $("선택자").html();
var.htmlContent = $("div").html();
console.log(htmlContent);
HTML 내용 설정하기: $("선택자").html("<p>새로운 HTML 내용</p>");
$("div").html("<p>이것은 새로운 HTML 내용입니다.</p>");

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>jQuery 메소드 예제</title>
  <script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
</head>
<body>

  <p id="textContent">이것은 초기 텍스트입니다.</p>
  <input type="text" id="myInput" value="기존 값">
  <div id="htmlContent"><p>이것은 초기 HTML 내용</p></div>

<script>
  // text() 메소드 예제
  var.textContent = $("#textContent").text();
  console.log("현재 텍스트 내용: " + textContent);
```

```
// text() 메소드를 사용하여 텍스트 설정
$("#textContent").text("새로운 텍스트 내용");

// val() 메소드 예제
var inputValue = $("#myInput").val();
console.log("현재 입력 값: " + inputValue);

// val() 메소드를 사용하여 입력 값 설정
$("#myInput").val("새로운 입력 값");

// html() 메소드 예제
var htmlContent = $("#htmlContent").html();
console.log("현재 HTML 내용: " + htmlContent);

// html() 메소드를 사용하여 HTML 내용 설정
$("#htmlContent").html("<p>새로운 HTML 내용</p>");
</script>

</body>
</html>
```

---

## > 07. css관련 메소드 사용법

---

jQuery의 css 메소드는 HTML 요소의 CSS 스타일을 동적으로 변경하는 데 사용됩니다.

문법:

`$(선택자).css(속성, 값);`

선택자: CSS 속성을 적용할 HTML 요소를 선택합니다.

속성: 변경하려는 CSS 속성의 이름을 문자열로 제공합니다.

값: 해당 속성에 적용할 값을 문자열로 제공합니다.

여러 속성 동시에 변경: json을 이용해서 한번에 여러개의 속성을 변경할 수 있다.

```
$(선택자).css({
    속성1: 값1,
    속성2: 값2,
    // 추가 속성들...
});
```

CSS 클래스 추가/제거: <style>태그에 기술된 클래스 관련 css를 추가, 삭제 가능하다.

`$(선택자).addClass("클래스이름"); // 클래스 추가`

`$(선택자).removeClass("클래스이름"); // 클래스 제거`

`$(선택자).toggleClass("클래스이름"); // 토글 (추가되어 있으면 제거, 없으면 추가)`

CSS 속성 읽기: css의 매개변수로 속성명을 넣으면 해당 속성의 값이 리턴된다.

```
var 값 = $(선택자).css("속성");
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>더 자세한 jQuery CSS 메소드 예제</title>
    <script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
    <style>
        .highlight {
            color: red;
            font-weight: bold;
        }
    </style>
</head>
<body>
    <p id="myParagraph">이 문장의 스타일을 변경합니다.</p>
    <script>
        $(document).ready(function () {
            // 특정 속성을 변경
    
```

```
$("#myParagraph").css("color", "blue");
// 여러 속성을 변경 (객체 형태로 전달)
$("#myParagraph").css({
    "font-size": "18px",
    "text-decoration": "underline"
});
// 클래스 추가
$("#myParagraph").addClass("highlight");
// 클래스 제거
// $("#myParagraph").removeClass("highlight");

// 클래스 토글 (추가되어 있으면 제거, 없으면 추가)
// $("#myParagraph").toggleClass("highlight");

// 특정 속성 읽기
var colorValue = $("#myParagraph").css("color");
console.log("현재 글자색: " + colorValue);
});

</script>

</body>
</html>
```

---

## > 07. 다양한 이벤트

---

jquery 선택자를 이용해서 이벤트 적용 엘리먼트를 찾은 다음 이벤트 메소드를 이용해서 이벤트 핸들러를 등록할 수 있다. 이벤트 함수 이름으로 해당 이벤트 이름을 기술 하면 된다. 이벤트 이름이 click이면 `$("#clickMe").click(function(){ })` 함수 이름도 클릭이다.

Mouse Events (마우스 이벤트):

`click`: 요소를 마우스로 클릭했을 때 발생하는 이벤트

`dblclick`: 요소를 빠르게 더블 클릭했을 때 발생하는 이벤트

`mouseenter`: 마우스 커서가 요소 안으로 이동했을 때 발생하는 이벤트

`mouseleave`: 마우스 커서가 요소에서 빠져나갈 때 발생하는 이벤트

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Mouse Events 예제</title>
    <!-- jQuery 라이브러리를 불러옵니다. -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        $(document).ready(function(){
            // 클릭 이벤트 처리
            $("#clickMe").click(function(){
                $(this).css("background-color", "yellow");
            });
            // 더블 클릭 이벤트 처리
            $("#dblClickMe").dblclick(function(){
                $(this).css("background-color", "green");
            });
            // 마우스 진입 이벤트 처리
            $("#enterMe").mouseenter(function(){
                $(this).text("마우스 진입!");
            });
            // 마우스 빠져나감 이벤트 처리
            $("#leaveMe").mouseleave(function(){
                $(this).text("마우스 빠져나감!");
            });
        });
    </script>
    <style>
        .event-box {
            width: 150px;
            height: 150px;
            margin: 10px;
        }
    </style>
```

```

        padding: 10px;
        background-color: lightgray;
        border: 2px solid darkgray;
        text-align: center;
        cursor: pointer;
    }

```

</style>

</head>

<body>

<div id="clickMe" class="event-box">

클릭하세요

</div>

<div id="dblClickMe" class="event-box">

더블 클릭하세요

</div>

<div id="enterMe" class="event-box">

마우스를 올려보세요

</div>

<div id="leaveMe" class="event-box">

여기로 마우스를 가져가세요

</div>

</body>

</html>

#### Keyboard Events (키보드 이벤트):

**keypress:** 키보드 키를 눌렀을 때 발생하는 이벤트입니다. 일반적으로 문자 입력을 처리하는 데 사용됩니다.

**keydown:** 키보드 키를 누르고 있는 동안 계속해서 발생하는 이벤트입니다. 긴 키를 누르고 있을 때 처리를 수행하는 데 사용됩니다.

**keyup:** 키보드 키를 놓았을 때 발생하는 이벤트입니다. 키를 놓을 때 처리를 수행하는 데 사용됩니다.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Keyboard Events 예제</title>
    <!-- jQuery 라이브러리를 불러옵니다. -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        $(document).ready(function(){
            // 키 누름 이벤트 처리
            $("#keyPressExample").keypress(function(event){
                // 누른 키의 코드를 가져와서 화면에 표시
                var keyCode = event.key;
                $("#keyPressResult").text("키 코드: " + keyCode);
            });
        });
    </script>

```

```

    // 키 다운 이벤트 처리
    $("#keyDownExample").keydown(function(event){
        // 누른 키의 코드를 가져와서 화면에 표시
        var keyCode = event.key;
        $("#keyDownResult").text("키 코드: " + keyCode);
    });
    // 키 놓음 이벤트 처리
    $("#keyUpExample").keyup(function(event){
        // 놓은 키의 코드를 가져와서 화면에 표시
        var keyCode = event.key;
        $("#keyUpResult").text("키 코드: " + keyCode);
    });
});
</script>
</head>
<body>
<input id="keyPressExample" type="text" placeholder="여기 입력">
<p id="keyPressResult">키 코드: </p>
<input id="keyDownExample" type="text" placeholder="여기 입력">
<p id="keyDownResult">키 코드: </p>
<input id="keyUpExample" type="text" placeholder="여기 입력">
<p id="keyUpResult">키 코드: </p>
</body>
</html>

```

### Form Events (폼 이벤트):

**submit:** 폼이 제출되었을 때 발생하는 이벤트입니다. 주로 폼 검증 및 서버로 데이터 제출에 사용됩니다.

**change:** 입력 필드 값이 변경되었을 때 발생하는 이벤트입니다. 주로 입력 값의 변경을 감지하고 처리하기 위해 사용됩니다.

**event.preventDefault():** 기본적으로 제공되는 이벤트를 막아 실행되지 않게 한다. 기본적으로 제공되는 이벤트에는 a태그의 페이지이동, form태그의 submit 등이 있다.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Form Events 예제</title>
    <!-- jQuery 라이브러리를 불러옵니다. -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script>
    $(document).ready(function(){
        // 폼 제출 이벤트 처리
        $("#myForm").submit(function(event){
            event.preventDefault(); // 폼 제출 기본 동작을 중지
        });
    });
</script>

```

```

        alert("폼이 제출되었습니다!");
    });
    // 입력 필드 값 변경 이벤트 처리
    $("#nameInput").change(function(){
        var newValue = $(this).val();
        $("#changeResult").text("변경된 값: " + newValue);
    });
});
</script>
</head>
<body>
<form id="myForm">
    <label for="nameInput">이름:</label>
    <input type="text" id="nameInput">
    <input type="submit" value="제출">
</form>
<p id="changeResult"></p>
</body>
</html>

```

#### Document/Window Events (문서 및 창 이벤트):

**load:** 문서 또는 웹 페이지가 모두 로드된 후 추가 작업을 수행하는 데 사용됩니다.

**resize:** 창 크기가 변경됐을 때 발생하는 이벤트입니다.

**focus:** 주로 입력 필드나 버튼 등의 요소가 선택 되었을 때 사용됩니다.

**blur:** 요소가 포커스를 잃었을 때 발생하는 이벤트입니다.

**unload:** 문서가 언로드(닫힘)될 때 발생하는 이벤트입니다. 현재 브라우저에서 지원해주지 않는다.

이벤트 이름 함수로 이벤트를 실행하는 것보다는 **on** 함수를 사용하여 이벤트 등록을 한다.

ex) 다음은 윈도우 객체에 load이벤트를 추가하는 예제이다.

```
.on(이벤트, 이벤트 핸들러)    $(window).on("load", function(){
```

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document/Window Events 예제</title>
    <!-- jQuery 라이브러리를 불러옵니다. -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        $(document).ready(function(){
            // 문서 로드 이벤트 처리
            $(window).on("load", function(){
                alert("문서 및 모든 컨텐츠가 로드되었습니다.");
            });
            // 창 크기 변경 이벤트 처리

```

```

$(window).on("resize", function(){
    var newWidth = $(window).width();
    var newHeight = $(window).height();
    $("#resizeResult").text("w"+newWidth+" h "+ newHeight);
});
// 포커스 이벤트 처리
$("#focusInput").focus(function(){
    $(this).css("background-color", "lightblue");
});
// 포커스 잃음 이벤트 처리
$("#blurInput").blur(function(){
    $(this).css("background-color", "white");
});
});

</script>
</head>
<body>
<p id="resizeResult">창 크기: </p>
<input id="focusInput" type="text" placeholder="포커스 받기">
<input id="blurInput" type="text" placeholder="포커스 잃음">
</body>
</html>

```

---

## > 08. jquery 효과

---

jquery를 이용해서 특정 태그를 움직이게 할수 있다.

다음 코드에서 사용한 메소드 설명이다.

`$("#fadeInButton").click(function() { ... });`: 이 코드는 "Fade In" 버튼을 클릭했을 때 콜백 함수에서는 `$("#element").fadeIn(1000);`으로 선택한 요소(#element)를 1초 동안 서서히 나타나게 하는 fadeIn 애니메이션을 실행합니다.

`$("#fadeOutButton").click(function() { ... });`: 이 코드는 "Fade Out" 버튼을 클릭했을 때 콜백 함수에서는 `$("#element").fadeOut(1000);`으로 선택한 요소(#element)를 1초 동안 서서히 사라지게 하는 fadeOut 애니메이션을 실행합니다.

`$("#slideInButton").click(function() { ... });`: 이 코드는 "Slide In" 버튼을 클릭했을 때 콜백 함수에서는 `$("#element").slideDown(1000);`으로 선택한 요소(#element)를 1초 동안 슬라이딩으로 나타나게 하는 slideDown 애니메이션을 실행합니다.

`$("#slideOutButton").click(function() { ... });`: 이 코드는 "Slide Out" 버튼을 클릭했을 때 콜백 함수에서는 `$("#element").slideUp(1000);`으로 선택한 요소(#element)를 1초 동안 슬라이딩으로 사라지게 하는 slideUp 애니메이션을 실행합니다.

`$("#toggleButton").click(function() { ... });`: 이 코드는 "Toggle" 버튼을 클릭했을 때 콜백 함수에서는 `$("#element").fadeToggle(1000);`으로 선택한 요소(#element)를 1초 동안 서서히 나타나거나 사라지게 하는 fadeToggle 애니메이션을 실행합니다. 이것은 토클 형태의 애니메이션이 된다.

`$("#customAnimateButton").click(function() { ... });`: 이 코드는 "Custom Animate" 버튼을 클릭했을 때 콜백 함수에서는 `$("#element").animate({...}, 1000);`으로 선택한 요소(#element)에 대한 사용자 지정 애니메이션을 실행합니다. 첫번째 매개변수는 변경정보를 json으로 넘겨주고, 두번째 매개변수로는 변경 시간을 설정 한다. 이 예제에서는 너비, 투명도, 좌우 여백을 변경하도록 설정되어 있으며, 1초 동안 실행됩니다.

```
<!DOCTYPE html>
<html>
<head>
  <title>jQuery Effects</title>
  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
  <style>
    #element {
      width: 100px;
      height: 100px;
      background-color: blue;
      margin-top: 20px;
```

```

        }
    </style>
<script>
    $(document).ready(function() {
        $("#fadeInButton").click(function() {
            $("#element").fadeIn(1000);
        });
        $("#fadeOutButton").click(function() {
            $("#element").fadeOut(1000);
        });
        $("#slideInButton").click(function() {
            $("#element").slideDown(1000);
        });
        $("#slideOutButton").click(function() {
            $("#element").slideUp(1000);
        });
        $("#toggleButton").click(function() {
            $("#element").fadeToggle(1000);
        });
        $("#customAnimateButton").click(function() {
            $("#element").animate({
                width: "200px",
                opacity: 0.5,
                marginLeft: "50px"
            }, 1000);
        });
    });
</script>
</head>
<body>
    <div id="element"></div>
    <button id="fadeInButton">Fade In</button>
    <button id="fadeOutButton">Fade Out</button>
    <button id="slideInButton">Slide In</button>
    <button id="slideOutButton">Slide Out</button>
    <button id="toggleButton">Toggle</button>
    <button id="customAnimateButton">Custom Animate</button>
</body>
</html>

```

다음 코드는 버튼을 클릭해서 사각형의 위치를 변화시키는 예제이다. 완성해서 결과를 확인해보자.

```
<!DOCTYPE html>
```

```

<html>
<head>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></s
cript>
</head>
<body>
<button>Start Animation</button>
<div style="background:#98bf21; height: 100px; width: 100px; position:
absolute;"></div>
<script>
var count = 0;
$(document).ready(function(){
    $("button").click(function(){
        switch(count){
            case 0:
                $("div").animate({left: '250px', opacity: '0.5'});
                count++;
                break;
            case 1:
                $("div").animate({top: '330px', opacity: '1'})
                count++;
                break;
            case 2:
                $("div").animate({left: '20px', opacity: '0.5'});
                count++;
                break;
            case 3:
                $("div").animate({top: '80px', opacity: '1'});
                count = 0;
                break;
        }
    });
});
</script>

</body>
</html>

```

---

## > 09. html조작

---

```

<!DOCTYPE html>
<html>
<head>
    <title>attr() 메서드 예제</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>
</head>
<body>
    <h1>attr() 메서드 예제</h1>
    

    <button id="changeImage">이미지 변경</button>

    <script>
        $(document).ready(function() {
            // 이미지 변경
            $("#changeImage").click(function() {
                $("#myImage").attr("src", "새로운이미지.jpg");
                $("#myImage").attr("alt", "새로운 이미지 설명");
            });
        });
    </script>
</body>
</html>

```

json을 이용해서 한번에 여러개의 속성을 변경할 수 있다. 다음 예제는 여러 개의 태그를 찾아서 json으로 여러개의 속성 값을 변경하는 예제이다.

```

<!DOCTYPE html>
<html>
<head>
    <title>여러 객체의 속성 변경 예제</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>
</head>
<body>
    <h1>여러 객체의 속성 변경 예제</h1>
    
    
    

    <button id="changeAttributes">속성 변경</button>

```

```

<script>
  $(document).ready(function() {
    // 속성 변경
    $("#changeAttributes").click(function() {
      // class가 "image"인 모든 이미지 요소의 src와 alt 속성 변경
      $(".image").attr({
        "src": "newimage.jpg",
        "alt": "새 이미지"
      });
    });
  });
</script>
</body>
</html>

```

다음 예제는 `$("#myLink").attr("href", function(i, origValue) { })` 함수의 설명을 다루는 예제이다. 이 함수는 현재 요소의 인덱스 i와 현재 href 속성 값인 origValue를 인수로 받아 리턴된 값을 새로운 href의 값으로 한다.

`$("#myLink")`: 이 부분은 jQuery의 선택자를 사용하여 문서에서 id가 "myLink"인 요소를 선택합니다. 즉, 이 코드는 id가 "myLink"인 하이퍼링크 요소를 선택합니다.

`.attr("href", function(i, origValue) {...})`: attr() 메서드를 호출하여 선택한 요소의 href 속성을 변경합니다. 이 메서드는 속성 이름과 콜백 함수를 인수로 사용합니다.

"href": 속성 이름을 지정합니다. 여기서는 href 속성을 변경하려고 합니다.

`function(i, origValue) {...}`: 콜백 함수를 정의합니다. 이 함수는 현재 요소의 인덱스 i와 현재 href 속성 값인 origValue를 인수로 받습니다.

i: 요소의 인덱스입니다. 이 콜백 함수가 여러 요소에 대해 실행될 때 각 요소의 인덱스를 나타냅니다. 이 예제에서는 단일 요소를 수정하므로 인덱스는 사용되지 않습니다.

origValue: 현재 요소의 href 속성 값입니다.

`return origValue + "/newpath";`: 이 콜백 함수는 현재 href 속성 값을 수정한 후 그 값을 반환합니다. 현재 href 속성 값인 origValue 뒤에 "/newpath"를 추가하고 있습니다.

```

<!DOCTYPE html>
<html>
<head>
  <title>attr() 메서드 예제</title>
  <script
  src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>

```

```

<script>
    $(document).ready(function() {
        // 링크 수정
        $("#modifyLink").click(function() {
            $("#myLink").attr("href", function(i, origValue) {
                return origValue + "/newpath";
            });
        });
    });
</script>
</head>
<body>
    <h1>attr() 메서드 예제</h1>
    <a id="myLink" href="https://www.example.com" title="예제 링크">이 링크를
    눌러보세요</a>
    <button id="modifyLink">링크 수정</button>
</body>
</html>

```

다음은 css관련 메소드이다.

.css(): HTML 요소의 CSS 속성을 가져오거나 설정합니다.

css("속성명"): 해당 속성의 값을 가져옴.

css("속성명", "값"): 해당 속성의 값을 설정.

// 속성 가져오기

var fontSize = \$("p").css("font-size");

// 속성 설정

\$(“p”).css(“color”, “red”);

.addClass(): 선택한 요소에 하나 이상의 CSS 클래스를 추가합니다.

\$(“div”).addClass(“highlight”);

.removeClass(): 선택한 요소에서 하나 이상의 CSS 클래스를 제거합니다.

\$(“div”).removeClass(“highlight”);

.toggleClass(): 선택한 요소에서 CSS 클래스를 추가하거나 제거합니다.

\$(“button”).click(function(){

\$("p").toggleClass("highlight");

});

```

<!DOCTYPE html>
<html>
<head>
    <title>jQuery CSS 및 클래스 메소드 예제</title>
    <script src=
    "https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>
    <style>

```

```

        .highlight { background-color: yellow; }
    </style>
</head>
<body>
<h1>jQuery CSS 및 클래스 메소드 예제</h1>
<p>이 문단의 스타일을 변경해보세요.</p>
<button id="changeStyle">스타일 변경</button>

<script>
$(document).ready(function() {
    // CSS 속성 가져오기
    var fontSize = $("p").css("font-size");
    console.log("폰트 크기: " + fontSize);
    // CSS 속성 설정
    $("p").css("color", "red");
    // 클래스 추가
    $("#changeStyle").click(function() {
        $("p").addClass("highlight");
    });
    // 클래스 제거
    $("#changeStyle").dblclick(function() {
        $("p").removeClass("highlight");
    });
    // 클래스 토글
    $("p").hover(function() {
        $(this).toggleClass("highlight");
    });
});
</script>
</body>
</html>

```

## > 10. dom조작

`remove()`와 `empty()` 메서드는 jQuery를 사용하여 HTML 요소를 변경하거나 제거하는 데 사용되는 메소드입니다. 각각의 메서드에 대한 설명을 제공하겠습니다.

// 선택한 요소와 하위 요소를 모두 제거 #myElement도 제거 된다.

```
$("#myElement").remove();
```

// 선택한 요소 내의 모든 자식 요소를 제거 #myElement은 제거 되지 않는다.

```

$( "#myElement" ).empty();

<!DOCTYPE html>
<html>
<head>
    <title>remove() vs. empty() 메서드 이해 예제</title>
    <style>
        #container {
            border: 1px solid #000;
            padding: 10px;
        }
    </style>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js">
</script>
    <script>
        $(document).ready(function() {
            // remove() 메서드 사용
            $("#removeButton").click(function() {
                var container = $("#container");
                container.remove();
                $("#result").text("remove() 메서드 사용: 컨테이너와 하위
요소 모두 삭제됨.");
            });
            // empty() 메서드 사용
            $("#emptyButton").click(function() {
                var container = $("#container");
                container.empty();
                $("#result").text("empty() 메서드 사용: 컨테이너 내의 하위
요소만 삭제됨.");
            });
        });
    </script>
</head>
<body>
    <h1>remove() vs. empty() 메서드 이해 예제</h1>
    <div id="container">
        <p>이것은 삭제될 문단입니다.</p>
        <button id="removeButton">remove() 사용</button>
        <button id="emptyButton">empty() 사용</button>
    </div>
    <div id="result"></div>

```

```
</body>
</html>
```

HTML 만드는 태그

```
var txt1 = "<p>텍스트.</p>";           // HTML로 요소 생성
var txt2 = $("<p></p>").text("텍스트."); // jQuery로 요소 생성
var txt3 = document.createElement("p");   // DOM을 사용하여 요소 생성
txt3.innerHTML = "텍스트.";             // 요소에 내용 추가
$("body").append(txt1, txt2, txt3);      // 새로운 요소들을 문서에 추가
```

append(), prepend(), after(), before()

append(): 이 메서드는 선택한 요소(부모 요소) 내에 다른 요소(자식 요소)를 하위로 추가합니다. 추가된 요소는 부모 요소 내의 마지막 자식 요소로 삽입됩니다.

```
var newElement = $("<p>New paragraph.</p>");// HTML 요소 생성
// 부모 요소에 자식 요소 추가
$("#container").append(newElement);
```

이 예제에서 "New paragraph."를 내용으로 가지는 새로운 <p> 요소가 #container 내의 마지막 자식 요소로 추가됩니다.

prepend(): 이 메서드는 선택한 요소(부모 요소) 내에 다른 요소(자식 요소)를 상위로 추가합니다. 추가된 요소는 부모 요소 내의 첫 번째 자식 요소로 삽입됩니다.

```
var newElement = $("<p>New paragraph.</p>");
// 부모 요소에 자식 요소 상단에 추가
$("#container").prepend(newElement);
```

이 예제에서 "New paragraph."를 내용으로 가지는 새로운 <p> 요소가 #container 내의 첫 번째 자식 요소로 추가됩니다.

after(): 이 메서드는 선택한 요소와 동일한 계층에 있는 다른 요소(형제 요소)를 선택한 요소 뒤에 추가합니다.

```

var newElement = $("<p>New paragraph.</p>");
// 선택한 요소 뒤에 형제 요소로 추가
$("#container p:first").after(newElement);

```

이 예제에서 "New paragraph."를 내용으로 가지는 새로운 <p> 요소가 #container 내의 첫 번째 <p> 요소 뒤에 추가됩니다.

**before()**: 선택한 요소와 동일한 계층에 있는 다른 요소(형제 요소)를 선택한 요소 앞에 추가합니다.

```

var newElement = $("<p>New paragraph.</p>");
// 선택한 요소 앞에 형제 요소로 추가
$("#container p:first").before(newElement);

```

이 예제에서 "New paragraph."를 내용으로 가지는 새로운 <p> 요소가 #container 내의 첫 번째 <p> 요소 앞에 추가됩니다.

```

<!DOCTYPE html>
<html>
<head>
    <title>append(), prepend(), after(), before() 예제</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js">
</script>
</head>
<body>
    <h1>append(), prepend(), after(), before() 예제</h1>

    <div id="container">
        <p>This is a paragraph.</p>
    </div>
    <button id="appendButton">append()</button>
    <button id="prependButton">prepend()</button>
    <button id="afterButton">after()</button>
    <button id="beforeButton">before()</button>
    <script>
        $(document).ready(function() {
            // append() 메서드 사용
            $("#appendButton").click(function() {
                var newElement = "<p>Appended paragraph.</p>";
                $("#container").append(newElement);
            });
        });
    </script>

```

```

    });
    // prepend() 메서드 사용
    $("#prependButton").click(function() {
        var newElement = "<p>Prepended paragraph.</p>";
        $("#container").prepend(newElement);
    });
    // after() 메서드 사용
    $("#afterButton").click(function() {
        var newElement = "<p>Paragraph inserted after.</p>";
        $("#container p").after(newElement);
    });
    // before() 메서드 사용
    $("#beforeButton").click(function() {
        var newElement = "<p>Paragraph inserted before.</p>";
        $("#container p").before(newElement);
    });
})
</script>
</body>
</html>

```

`children()` 메서드는 선택한 요소의 직계 자식 요소를 선택합니다. 이것은 요소의 바로 아래 단계의 자식들만 선택합니다.

`find()` 메서드는 선택한 요소 내의 모든 하위 요소를 선택합니다. 이것은 직계 자식 뿐만 아니라 모든 자손 요소를 선택합니다.

간단히 말하면, `children()`는 한 단계 아래의 자식을 선택하고, `find()`는 모든 하위 요소를 선택합니다.

```

<!DOCTYPE html>
<html>
<head>
    <title>children() vs. find() 메서드 예제</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>
</head>
<body>
    <h1>children() vs. find() 메서드 예제</h1>
    <div id="parent">
        <p>직계 자식 1</p>
        <p>직계 자식 2</p>
        <div>
            <p>자식의 자식 1</p>

```

```

        <p>자식의 자식 2</p>
    </div>
</div>
<button id="childrenButton">children()</button>
<button id="findButton">find()</button>
<div id="result"></div>
<script>
$(document).ready(function() {
    // children() 메서드 사용
    $("#childrenButton").click(function() {
        var directChildren = $("#parent").children("p");
        var result = "children() 메서드 결과: " + directChildren.length
+ "개의 직계 자식 <p> 요소";
        $("#result").text(result);
    });

    // find() 메서드 사용
    $("#findButton").click(function() {
        var descendants = $("#parent").find("p");
        var result = "find() 메서드 결과: " + descendants.length + "개의
모든 자손 <p> 요소";
        $("#result").text(result);
    });
});
</script>
</body>
</html>

```

**siblings()** next() nextAll() nextUntil() prev() prevAll() prevUntil()

**siblings()**: 선택한 요소의 형제 요소를 모두 선택합니다. 선택한 요소와 동일한 부모를 공유하는 다른 형제 요소를 모두 선택합니다.

**next()**: 선택한 요소의 다음 형제 요소를 선택합니다. 현재 요소의 다음에 위치한 형제 요소를 선택합니다.

**nextAll()**: 선택한 요소의 다음에 있는 모든 형제 요소를 선택합니다. 현재 요소 다음에 위치한 모든 형제 요소를 선택합니다.

**nextUntil()**: 선택한 요소의 다음에 있는 형제 요소 중에서 지정한 요소까지의 모든 형제 요소를 선택합니다. 현재 요소 다음에 위치한 형제 요소 중에서 특정 요소까지의 모든 형제 요소를 선택합니다.

**prev()**: 선택한 요소의 이전 형제 요소를 선택합니다. 현재 요소의 바로 앞에 위치한 형제 요소를 선택합니다.

**prevAll()**: 선택한 요소의 이전에 있는 모든 형제 요소를 선택합니다. 현재 요소 앞에 위치한 모든 형제 요소를 선택합니다.

`prevUntil()`: 선택한 요소의 이전에 있는 형제 요소 중에서 지정한 요소까지의 모든 형제 요소를 선택합니다. 현재 요소 앞에 위치한 형제 요소 중에서 특정 요소까지의 모든 형제 요소를 선택합니다.

```
<!DOCTYPE html>
<html>
<head>
    <title>Sibling 관련 메서드 예제</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>
</head>
<body>
    <h1>Sibling 관련 메서드 예제</h1>
    <ul>
        <li>사과</li><li>바나나</li>
        <li>체리</li><li>딸기</li><li>포도</li>
    </ul>
    <button id="siblingsButton">siblings()</button>
    <button id="nextButton">next()</button>
    <button id="nextAllButton">nextAll()</button>
    <button id="nextUntilButton">nextUntil()</button>
    <button id="prevButton">prev()</button>
    <button id="prevAllButton">prevAll()</button>
    <button id="prevUntilButton">prevUntil()</button>
    <div id="result"></div>
    <script>
        $(document).ready(function() {
            $("#siblingsButton").click(function() {
                var siblings = $("li:nth-child(3)").siblings();
                $("#result").text("siblings(): " + siblings.length + " 개의 형제 요소 선택");
            });
            $("#nextButton").click(function() {
                var nextElement = $("li:nth-child(3)").next().text();
                $("#result").text("next(): " + nextElement);
            });
            $("#nextAllButton").click(function() {
                var nextAllElements = $("li:nth-child(3)").nextAll().text();
                $("#result").text("nextAll(): " + nextAllElements);
            });
            $("#nextUntilButton").click(function() {
                var nextUntilElements =
                    $("li:nth-child(3)").nextUntil(":contains('딸기')").text();
                $("#result").text("nextUntil(): " + nextUntilElements);
            });
            $("#prevButton").click(function() {
```

```

        var prevElement = $("li:nth-child(3)").prev().text();
        $("#result").text("prev(): " + prevElement);
    });
    $("#prevAllButton").click(function() {
        var prevAllElements = $("li:nth-child(3)").prevAll().text();
        $("#result").text("prevAll(): " + prevAllElements);
    });
    $("#prevUntilButton").click(function() {
        var prevUntilElements =
$("li:nth-child(3)").prevUntil(":contains('사과')").text();
        $("#result").text("prevUntil(): " + prevUntilElements);
    });

```

</script>

</body>

</html>

The `first()`, `last()`, `eq()`, `filter()`, `not()`

`first()`: 일치하는 요소 중 첫 번째 요소를 선택합니다. 주로 첫 번째 요소만을 선택하는 데 사용됩니다.

`last()`: 일치하는 요소 중 마지막 요소를 선택합니다. 주로 마지막 요소를 선택하는 데 사용됩니다.

`eq()`: 일치하는 요소 중 특정 인덱스의 요소를 선택합니다. 인덱스(0부터 시작)를 제공하여 특정 요소를 선택하는 데 사용됩니다.

`filter()`: 일치하는 요소 집합을 특정 조건이나 선택자와 일치하는 요소로 필터링합니다. 필터링 조건이나 함수를 제공하여 어떤 요소를 포함시킬지 결정하는 데 사용됩니다.

`not()`: 일치하는 요소 집합을 특정 조건이나 선택자와 일치하지 않는 요소로 줄입니다. 지정된 필터 조건이나 선택자와 일치하지 않는 모든 요소를 선택하는 데 사용됩니다.

```

<!DOCTYPE html>
<html>
<head>
    <title>jQuery Selection Methods</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>
</head>
<body>
    <ul>
        <li>항목 1</li><li>항목 2</li>
        <li>항목 3</li><li>항목 4</li><li>항목 5</li>
    </ul>

```

```

<script>
$(document).ready(function() {
    // first() 메서드: 첫 번째 요소 선택
    var firstItem = $("li").first().text();
    console.log("first(): " + firstItem);
    // last() 메서드: 마지막 요소 선택
    var lastItem = $("li").last().text();
    console.log("last(): " + lastItem);
    // eq() 메서드: 특정 인덱스의 요소 선택 (0부터 시작)
    var thirdItem = $("li").eq(2).text();
    console.log("eq(): " + thirdItem);
    // filter() 메서드: 특정 조건에 맞는 요소 선택
    var filteredItems = $("li").filter(function() {
        return $(this).text().includes("2");
    }).text();
    console.log("filter(): " + filteredItems);
    // not() 메서드: 특정 조건에 맞지 않는 요소 선택
    var notItems = $("li").not(":contains('2')").text();
    console.log("not(): " + notItems);
});
</script>
</body>
</html>

```

## > 11. jquery 프로젝트1



제공된 코드는 HTML과 JavaScript를 사용하여 간단한 입력 양식 및 데이터 유효성 검사를 처리하는 웹 페이지를 만드는 예제입니다. 이 페이지는 사용자로부터 이름과 비밀번호를 입력받고, 유효성 검사를 통과한 경우에만 데이터를 전송합니다.

여기에서 주요 기능을 설명합니다:

<form> 요소 내에 입력 필드 (텍스트 필드 및 패스워드 필드)와 제출 버튼이 있습니다.

입력 필드에는 name 및 password라는 id 속성이 있어 JavaScript에서 해당 요소를 식별할 수 있습니다.

JavaScript 코드는 jQuery를 사용하여 페이지가 로드될 때 실행되며, 폼 제출 이벤트를 가로채고 유효성 검사를 수행합니다.

`$("#my_form").submit(function(e) { ... })` 코드는 폼이 제출될 때 실행되는 함수를 정의합니다.

`e.preventDefault();` 코드는 기본 이벤트를 막아 폼의 제출을 방지합니다.

사용자가 이름과 비밀번호를 입력한 후 "제출" 버튼을 클릭하면, 입력값이 비어 있지 않은지 확인하고, 둘 다 입력된 경우 alert() 함수를 사용하여 데이터를 경고 상자에 표시하고, 그렇지 않으면 "데이터 입력하세요"라는 경고를 표시합니다.

참고로, 이 예제에서는 데이터를 실제 서버로 보내거나 저장하지 않고, 사용자에게 경고창으로만 데이터를 보여줍니다. 만약 실제로 데이터를 서버로 전송하려면, 폼을 서버 측 스크립트 (예: PHP, Python, Node.js)를 사용하여 처리해야 합니다.

```
<html>
<head>
<style>
input[type=text],input[type=password]{
    height:24px;
    line-height:24px;
    text-indent:5px;
    font-family:"맑은 고딕";
}
input[type=submit] {
    margin-top: 15px;
    padding: 5px 15px;
    font-family: "맑은 고딕";
    background: #666;
    color: #fff;
    border: none;
    cursor: pointer;
    border-radius: 5px; /* ie9이상 사용가능*/
    -webkit-border-radius: 5px; /*사파리/ 크롬*/
}
</style>
<script src="jquery/jquery-3.2.1.js"></script>
<script>

$(function(){
    $("#my_form").submit(function(e){
        //e.preventDefault();
        // 입력 양식의 value를 가져옵니다.
        var name=$("#name").val();
        var password=$("#password").val();
        var str=name+" : "+password;
    })
})
```

```

    if(name!=""&&password!=""){//둘다 입력하였을 경우
        //데이터 전송
        alert(name+":"+password);
    }else{
        //전송을 막기
        //기본이벤트 막기
        alert("데이터 입력하세요");
        e.preventDefault();
    }
});
});

</script>

</head>
<body>
<form id="my_form">
    <input type="text" name="name" id="name"/><br>
    <input type="password" name="password" id="password"/><br>
    <input type="submit" value="제출"/><br>
</form>

</body>
</html>

```



이 HTML 및 JavaScript 코드는 "전체 선택" 기능을 가진 체크 박스와 해당 하위 항목 체크 박스 목록을 포함하는 간단한 웹 페이지를 생성합니다.

여기에서 코드의 주요 부분을 설명합니다:

<input type="checkbox" id="all\_check"/>: "전체 선택" 체크 박스를 나타내는 요소입니다. 이 체크 박스를 선택하면 모든 하위 체크 박스가 선택됩니다.

<div id="check\_item">: 여러 하위 체크 박스와 해당 라벨을 감싸는 div 요소입니다.

jQuery 라이브러리를 사용하여 "전체 선택" 체크 박스의 변경 이벤트를 처리합니다. 코드는 `$(function(){ ... })`를 사용하여 페이지가 로드될 때 실행됩니다.

`$("#all_check").change(function(){ ... })`: "전체 선택" 체크 박스의 상태가 변경될 때 실행되는 함수를 정의합니다.

`if(this.checked) { ... }`: "전체 선택" 체크 박스가 선택된 경우, 모든 하위 체크 박스를 선택하도록 하는 조건문입니다. `this.checked`는 "전체 선택" 체크 박스의 선택 상태를 나타냅니다.

`$("#check_item").children("input").prop("checked", true);`: 모든 하위 체크 박스를 선택 상태로 변경하는 코드입니다. `$("#check_item")`는 "check\_item" ID를 가진 div 요소를 선택하고, `.children("input")`는 해당 div 안에 있는 모든 input 요소를 선택합니다. 그런 다음 `.prop("checked", true)`를 사용하여 이러한 하위 체크 박스의 "checked" 속성을 true로 설정하여 선택 상태로 만듭니다.

반대로, "전체 선택" 체크 박스가 선택되지 않은 경우 모든 하위 체크 박스를 선택 해제합니다.

이 코드는 사용자가 "전체 선택" 체크 박스를 선택하면 모든 하위 체크 박스를 선택 또는 해제할 수 있는 유용한 기능을 제공합니다.

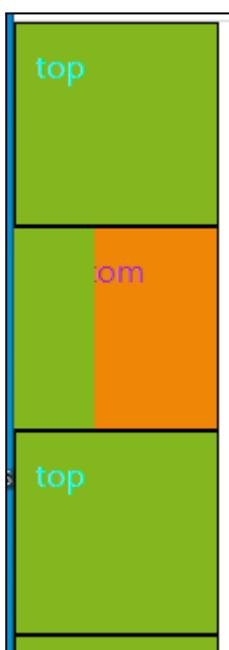
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script
        src="//ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js"></script>
    <script>
        $(function(){          //$(document).ready(function(){
            $("#allCheck").on("change",function(){
                //this //javascript dom
                //$(this) //jquery dom
                //$(this).is(':checked');
                if(this.checked){
                    // alert('checked');
                    //checked속성이 값을 가지고 있지 않아서 attr를 사용하면 문제가
                    //발생해서 prop를 사용함
                    //attr prop 차이 웹에서 확인
                    $("div").children("input").prop("checked",true);
                    // $("div").children("input").attr("checked",true);
                }else{
                    //alert('not checked');
                    $("div").children("input").prop("checked",false);
                    // $("div").children("input").attr("checked",true);
                }
            });
        })
    </script>
</head>
<body>
    <input type="checkbox" id="allCheck"/>allCheck
```

```
<div>
  <input type="checkbox"/>A
  <input type="checkbox"/>B
  <input type="checkbox"/>C
</div>
</body>
</html>
```

---

## > 12. jquery 프로젝트2

---



이 HTML 및 JavaScript 코드는 마우스를 요소 위에 가져가면 요소 내에 있는 상위 요소 `top`를 슬라이드하여 뒤에 있는 `button`을 보여주는 효과를 구현한 예제입니다.

여기에서 코드의 주요 부분을 설명합니다:

`<div id="container">`: 모든 리스트 아이템을 감싸는 컨테이너입니다.

`<ul>` 및 `<li>` 요소: 목록 항목을 나타내는 요소입니다. 각 항목은 "bottom"이라는 텍스트와 `.top` 클래스를 가진 하위 요소를 포함합니다.

.top 클래스: 이 클래스는 하위 요소를 나타내며, 각 항목 위에 표시되고 해당 항목 내에서 슬라이딩 효과를 만듭니다.

jQuery 라이브러리를 사용하여 페이지가 로드될 때 실행되는 함수를 정의합니다. 코드는 `$(function(){ ... })`를 사용하여 페이지가 로드될 때 실행됩니다.

`$("#container>ul>li").hover(...)` 코드는 모든 리스트 항목 (`<li>`)에 대한 hover 이벤트 핸들러를 정의합니다. 이 코드는 마우스를 항목 위로 가져갔을 때 (마우스 오버) 및 떠났을 때 (마우스 아웃) 두 가지 함수를 정의합니다.

`$(this).find(".top").stop().animate({ left: "-100px" })`: 마우스를 항목 위로 가져갈 때, 해당 항목 내에 있는 .top 클래스 요소를 찾아서 왼쪽으로 슬라이드하는 애니메이션을 실행합니다. 왼쪽으로 슬라이드 하면 div를 벗어나 화면에 보이지 않는다.

`$(this).find(".top").stop().animate({ left: "0px" })`: 마우스가 항목에서 떠날 때, .top 클래스 요소를 원래 위치로 되돌리는 애니메이션을 실행합니다.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        *{
            margin:0px; padding:0px;
        }
        #container ul li{
            border: 1px solid black;
            position: relative;
            padding:10px;
            width:80px; height:80px;
            background: #ef8605; color:rgb(173, 36, 200);
        }
        .top{
            position: absolute;
            left: 0px; top:0px;
            padding:10px;
            width:80px; height:80px;
            background: #82b81d; color:aqua;
        }
    </style>
    <script src=
    "https://ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js"></script>
    <script>
        //$(document).ready(function(){
        $(function(){


```

```

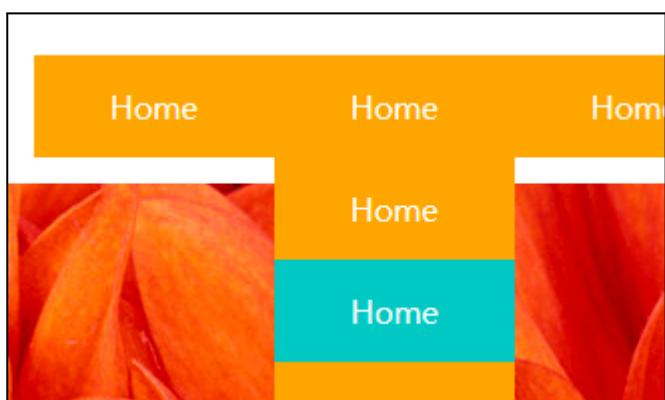
        $("#container>ul>li").hover(
            function(){
                $(this).find(".top").stop().animate({
                    left:"-100px"
                })
            },function(){
                $(this).find(".top").stop().animate({
                    left:"0px"
                })
            }
        )
    </script>
</head>
<body>
    <div id="container">
        <ul>
            <li>
                bottom
                <div class="top">top</div>
            </li>
            <li>
                bottom
                <div class="top">top</div>
            </li>
        </ul>
    </div>
</body>
</html>

```

---

## > 13. jquery 프로젝트3

---



다른 페이지로 이동 가능한 상단 메뉴이다.  
마우스를 A태그 위에 올렸을 때 배경색이  
변경되고, 마우스를 떼면 원래 배경색으로  
돌아갑니다.

이 HTML, CSS 및 JavaScript 코드는  
다음과 같은 특징을 가진 웹 페이지를  
만듭니다:

수평 메뉴 생성:

<ul> 요소 안에 있는 <li> 요소를 사용하여

수평 메뉴를 생성합니다.

각 메뉴 항목은 가로로 나란히 배치되며 (float: left), 너비는 120px로 설정됩니다.

마우스 호버 효과:

사용자가 메뉴 항목 위로 마우스를 올리면, 해당 항목의 하위 메뉴가 아래로 슬라이드되어 표시됩니다. 이 동작은 JavaScript를 사용하여 구현되며 hover 이벤트를 활용합니다.

`$(this).find("div>div").stop().slideDown();`와  
`$(this).find("div>div").stop().slideUp();` 코드로 하위 메뉴를 슬라이드 인 및 아웃합니다.

배경색 변경:

메뉴 항목 또는 하위 메뉴 항목에 마우스를 올리면 배경색이 변경됩니다. 배경색 변경은 사용자 경험을 향상시키고 메뉴 항목에 대한 시각적 피드백을 제공합니다.

`data-bg` 속성에 지정된 색상을 가져와 배경색을 동적으로 변경합니다.

메뉴 항목에 대한 배경색 변경: `$(this).stop().css("background-color", overColor);`  
마우스를 이탈하면 원래의 배경색으로 복원: `$(this).stop().css("background-color", "#ffa500");`

이미지 표시:

메뉴 위에 위치한 `div` 요소 아래에 이미지가 표시됩니다. 이미지는 메뉴 위에 겹쳐져 있으며 `z-index`를 사용하여 위치가 조정됩니다.

이 코드를 사용하여 사용자 친화적인 메뉴 인터페이스를 만들 수 있으며, 메뉴 항목에 대한 시각적 효과를 포함시켜 사용자에게 더 나은 사용자 경험을 제공할 수 있습니다.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <!-- jQuery CDN -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <style>
        li{
            float:left;
            list-style: none;
            width: 120px;
        }
        li a{
            display: block;
            padding: 15px 0px;
            width: 120px;
            text-align: center;
            text-decoration: none;
            background-color: orange;
            color: white;
        }
        li>div>div{
            display: none;
        }
        ul{
            display: inline-block;
        }
    </style>
```

```

<script>
    $(function(){
        $("li").hover(
            function(){
                $(this).find("div>div").stop().slideDown();
            },
            function(){
                $(this).find("div>div").stop().slideUp();
            }
        )
        $("li a").hover(
            function(){
                var overColor = $(this).attr("data-bg");
                $(this).stop().css("background-color", overColor);
            },
            function(){
                $(this).stop().css("background-color", "#ffa500");
            }
        )
    })
</script>
<title>Document</title>
</head>
<body>
    <div style="position:relative;width:1000px;height:80px;z-index:2;">
        <ul>
            <li>
                <div>
                    <a href="#" data-bg="#00c8c3">Home</a>
                    <div><a href="#" data-bg="#00c8c3">Home</a><a href="#" data-bg="#00c8c3">Home</a></div>
                </div>
            </li>
            <li><div><a href="#" data-bg="#00c8ff">Home</a><div><a href="#" data-bg="#00c8c3">Home</a><a href="#" data-bg="#00c8c3">Home</a><a href="#" data-bg="#00c8c3">Home</a></div></div></li>
                <li><div><a href="#" data-bg="#ffc8c3">Home</a><div><a href="#" data-bg="#00c8c3">Home</a><a href="#" data-bg="#00c8c3">Home</a></div></div></li>
                    <li><div><a href="#" data-bg="#00fc3">Home</a><div><a href="#" data-bg="#00c8c3">Home</a><div><a href="#" data-bg="#00c8c3">Home</a><a href="#" data-bg="#00c8c3">Home</a></div></div></li>
                        <li><div><a href="#" data-bg="#00fffc3">Home</a><div><a href="#" data-bg="#00c8c3">Home</a><a href="#" data-bg="#00c8c3">Home</a></div></div></li>
                            <li><div><a href="#" data-bg="#00fffc3">Home</a><div><a href="#" data-bg="#00c8c3">Home</a><a href="#" data-bg="#00c8c3">Home</a></div></div></li>

```

```
</ul>
</div>
<div style="position:relative;z-index:1;">
  
</div>
</body>
</html>
```

---

## > 14. jquery 프로젝트4

---



<ul> 요소 안에 있는 <li> 요소를 사용하여 수직 메뉴를 생성합니다. 각 메뉴 항목은 하위 메뉴를 포함합니다. 메뉴 항목은 확장 가능한 섹션으로 제공됩니다.

**마우스 호버 효과:**

사용자가 메뉴 항목에 마우스를 올리면 하위 메뉴가 나타나며, 배경색이 변경됩니다.

메뉴 항목의 배경색은 초록색으로 설정되며 호버 시 진한 초록색으로 변경됩니다.

하위 메뉴 항목은 보라색으로 배경색이 설정되며 호버 시 투명한 보라색으로 변경됩니다.

**메뉴 동작:**

메뉴 항목을 클릭하면 해당 섹션이 확장됩니다. 클릭한 메뉴의

하위 메뉴가 표시됩니다.

사용자가 메뉴를 빠져나갈 때, 2초 후에 현재 선택한 메뉴의 하위 메뉴가 다시 표시됩니다. 이것은

`setTimeout` 함수를 사용하여 구현되었습니다.

`$(".sub").hide();`은 모든 하위 메뉴 항목을 숨깁니다.

`$("#menu>li>div").on("mouseenter", function() { ... });`은 메뉴 항목에 마우스를 올리면 이벤트를 설정합니다. 호버 시 다른 하위 메뉴를 숨기고 해당 메뉴 항목의 하위 메뉴를 나타나게 합니다.

`($("#menu>li>div").on("click", function() { ... });`은 메뉴 항목을 클릭할 때 해당 메뉴 항목을 `selectDiv`로 설정합니다.

`($("#menu").on("mouseleave", function() { ... });`은 사용자가 메뉴를 빠져나갈 때, 2초 후에 현재 선택한 메뉴의 하위 메뉴를 다시 표시하는 기능을 구현합니다. 이것은 `setTimeout` 함수를 사용하여 구현되며, `selectDiv`로 선택된 메뉴를 추적하여 해당 메뉴의 하위 메뉴를 나타냅니다.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Document</title>
<style>
#menu>li>div{
    margin-bottom: 2px;
    height: 35px;
    line-height: 35px;
    background-color: green;
    cursor: pointer;
}
#menu, #menu ul{
    margin: 0px;
    padding: 0px;
    list-style: none;
}
ul#menu{
    width: 200px;
    text-indent: 35px;
}
ul.sub li{
    background: rgb(242, 10, 234);
    line-height: 35px;
    margin-bottom: 2px;
}
ul.sub li:hover{
    background: rgba(4, 116, 118, 0.354);
}
```

```

    ul.sub li a{
        display: block;
        text-decoration: none;
        color: #fff;
    }

```

</style>

```

<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
```

```

<script>
    var selectDiv=$("#menu>li>div").eq(0);
    $(function(){
        $(".sub").hide();
        $("#menu>li>div").on("mouseenter",function(){
            $(".sub").stop().slideUp(300);
            $(this).next().stop().slideDown(300);
        });
        $("#menu>li>div").on("click",function(){
            selectDiv=$(this);
        })
        $("#menu").on("mouseleave",function(){
            setTimeout(function(){
                $(".sub").stop().slideUp(300);
                selectDiv.next().stop().slideDown(300);
            },2000);
        })
    })
    //title를 클릭하면 사용자가 메뉴를 사용하다가 메뉴를 빠져나가면 클릭한 메뉴를
    //3초후에 보여주기.
</script>

```

</head>

<body>

```

<ul id="menu">
    <li>
        <div>title</div>
        <ul class="sub">
            <li><a href="#">sub1</a></li>
            <li><a href="#">sub2</a></li>
            <li><a href="#">sub3</a></li>
        </ul>
    </li>
    <li>
        <div>title</div>
        <ul class="sub">
            <li><a href="#">sub1</a></li>
            <li><a href="#">sub2</a></li>

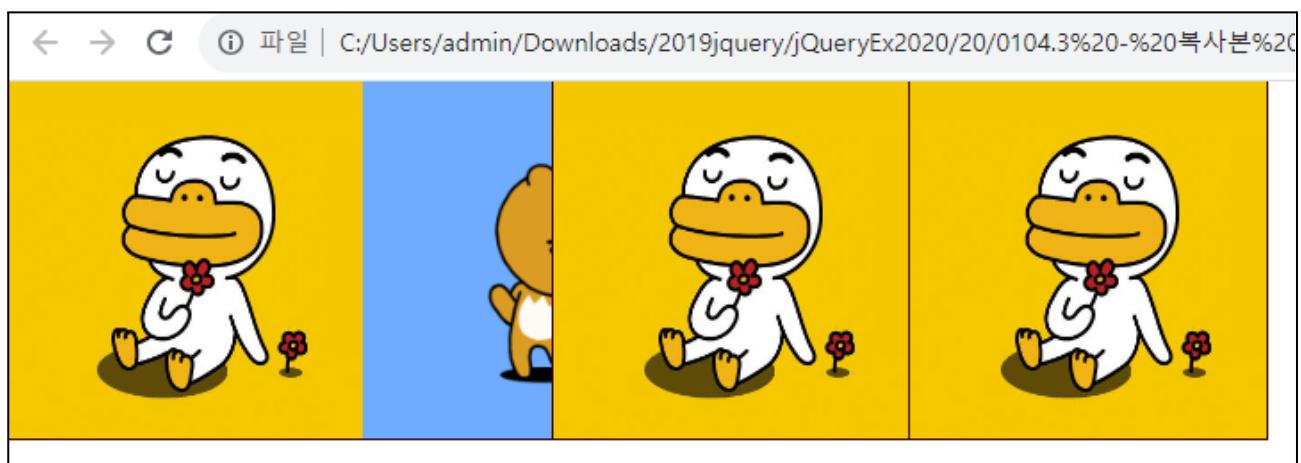
```

```
</ul>
</li>
<li>
  <div>title</div>
  <ul class="sub">
    <li><a href="#">sub1</a></li>
    <li><a href="#">sub2</a></li>
    <li><a href="#">sub3</a></li>
    <li><a href="#">sub4</a></li>
  </ul> </li> </ul></body></html>
```

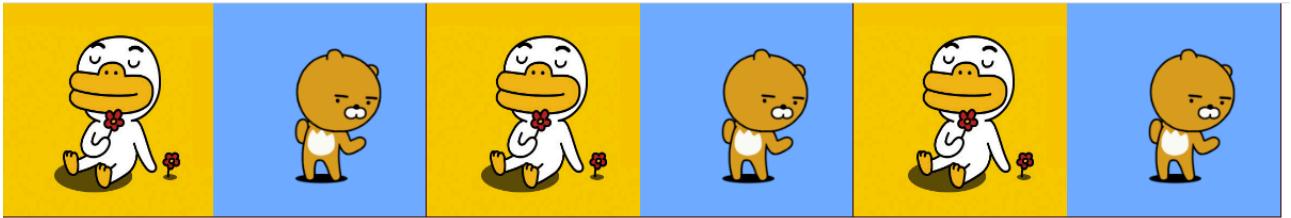
---

## > 15. jquery 프로젝트5

---



상위 이미지 처럼 3개의 오리 중 하나를 선택하면 이미지와 짹이되는 이미지가 선택된 이미지 옆에 짹이미지가 나타난다.



최초 `div01`의 크기를 400으로 하고 오리 이미지와 짹이미지를 각각 200으로 만들면 모든 이미지가 다 보인다. 만약 `div01`를 200으로 고정하고 영역을 벗어난 이미지는 안보이게 한다면 오리 이미지만 보일 것이다. 이때 만약 오리이미지가 있는 `div01`에 마우스가 올라가면 `div01`를 400으로 만들면 해당 오리의 짹만 보일 것이다. 이런 방식을 이용해서 6개의 이미지중 오리와 선택된 오리의 짹만 보이게 하는 예제를 만들어 보았다.

이 HTML 및 jQuery 코드는 마우스 호버 효과를 사용하여 `div01` 클래스를 가진 요소의 너비를 확장 및 축소하는 기능을 구현합니다.

여기에 대한 코드 설명:

**스타일 부분:**

모든 요소의 여백 (마진 및 패딩)을 0으로 설정합니다.

**.div01 클래스 스타일:**

외곽선은 1px 두께의 어두운 빨간색으로 설정됩니다.

왼쪽으로 플로팅되어 있으며, 너비와 높이는 각각 200px로 설정됩니다.

배경색은 밝은 빨간색으로 지정됩니다.

`overflow: hidden`은 내부 콘텐츠가 요소 영역을 벗어날 때 가려지도록 합니다.

**.div01 클래스를 가진 요소의 호버 이벤트:**

`.hover()` 함수는 마우스 호버(마우스를 올리거나 내림) 이벤트를 처리합니다.

`function(){}`의 첫 번째 함수는 마우스를 요소 위로 올릴 때 실행됩니다.

`$(this)`는 현재 호버된 `.div01` 요소를 나타냅니다.

`stop().animate()`를 사용하여 너비를 400px로 확장하는 애니메이션을 실행합니다. 이 때 애니메이션의 지속 시간은 300밀리초로 설정됩니다.

`function(){}`의 두 번째 함수는 마우스를 요소 밖으로 옮길 때 실행됩니다.

다시 `stop().animate()`를 사용하여 너비를 200px로 축소하는 애니메이션을 실행합니다.

**<body> 섹션:**

`.div01` 클래스를 가진 요소로 구성된 `div` 컨테이너를 생성합니다.

각 `.div01` 요소는 확장 및 축소될 수 있습니다. 각 요소에는 이미지 (`img`)와 배경이 있는 내부 `div`가 포함되어 있습니다.

이 코드는 마우스 호버시 `div01` 클래스를 가진 요소의 너비를 확장하고 마우스를 요소에서 떼면 다시 축소하는 간단한 효과를 제공합니다.

이 HTML 및 jQuery 코드는 마우스 호버 효과를 사용하여 `div01` 및 `div02` 클래스를 가진 요소의 너비를 확장하고 축소하는 기능을 구현합니다.

아래는 코드의 설명:

`<script src="https://code.jquery.com/jquery-3.2.1.js"></script>`:

jQuery 라이브러리를 페이지에 추가합니다.

<script> 섹션:

startMenu 변수는 초기 메뉴 아이디로 "box1"을 설정합니다.

intervalControl 함수:

intervalControl 함수는 주기적으로 메뉴의 너비를 변경하는 역할을 합니다.

\$("#background>div").each(function(){});를 사용하여 모든 메뉴 요소를 순회합니다.

startMenu과 메뉴의 아이디를 비교하여 같으면 너비를 400px로 확장하고 다르면 너비를 200px로 축소하는 애니메이션을 적용합니다.

\$("#background").mouseleave(function(){}):

배경을 마우스로 떠날 때, intervalControl 함수가 2초 뒤에 호출되도록 설정되었습니다.

.div02 클래스를 가진 요소의 호버 이벤트:

.hover() 함수를 사용하여 마우스 호버(마우스를 올리거나 내림) 이벤트를 처리합니다.

호버 시, .div01 클래스를 가진 모든 요소의 너비는 200px로 축소되며, 호버된 .div02 요소의 부모인 .div01 요소는 너비가 400px로 확장됩니다.

<body> 섹션:

.div01 및 .div02 클래스를 가진 요소로 구성된 div 컨테이너인 "background"를 생성합니다.

각 .div01 요소는 400px로 확장될 수 있고, 그 안에 있는 .div02 요소는 그 부모 .div01의 너비에 영향을 줍니다.

이미지와 데이터가 포함된 요소들로 구성되며, 마우스 호버 시 너비가 확장하고 축소합니다.

```
<html>
  <head>
    <script
      src="https://code.jquery.com/jquery-3.2.1.js"></script>

  <style>
    *{
      margin: 0;      padding:0;
    }
    #background{
    }
    .div01{
      outline: 1px solid rgb(58, 4, 4);
      float: left;
      width:200px;      height:200px;
      padding:0px;
      background-color: rgb(255, 103, 103);
      overflow: hidden;
    }
    .div02{
      outline: 1px solid rgb(58, 4, 4);
      display: inline-block;
      width:200px;      height:200px;
    }
  </style>

```

```

padding:0px;
background-color: rgb(216, 59, 59);
vertical-align: top;
}
</style>
<script>
var startMenu="box1";
function intervalControl(){
    $("background>div").each(function(){
        if(startMenu==$this).attr("id")){
            //같으면 400
            $(this).animate({width:"400"},300);
        }else{
            //다르면 200
            $(this).animate({width:"200"},300);
        }
    })
}
$(document).ready(function(){
    $("#background").mouseleave(function(){
        // setTimeout(intervalControl,5000);
    })
    // intervalControl()
    // $("#" + startMenu).animate({width: "300"},300);
    // $(".div02").hover(function(){
    //     $(this).attr("id")
    //     startMenu=$(this).parent().parent().attr("id");
    //     //$(".box").animate({width: "100"},300);
    //     //$(this).parent().animate({width: "300"},300);
    // })
    $(".div02").hover(
        function(){
            $(".div01").animate({width:"200"},300);
            $(this).parent().parent().animate({width:"400"},300);
        },
        function(){
            // $(this).parent().parent().animate({width: "200"},300);
        }
    )
})
</script>
<body>
<div id="background">
    <div class="div01" id="box1"><!--yellow div-->
        <div style="width:400; background:black;" >
            <div class="div02"><!--blue div-->

```

```

<span></span><!--메뉴의 데일리-->
</div>
</div>
</div>
<div class="div01" id="box2">
    <div style="width:400; background:black;" >
        <div class="div02"><!--blue div-->

            <span></span><!--메뉴의 데일리-->
                </div>
                </div>
            </div>
            <div class="div01" id="box3">
                <div style="width:400; background:black;" >
                    <div class="div02"><!--blue div-->
                        <span></span><!--메뉴의 데일리-->
                            </div>
                            </div>
                        </div>
                    </div>
                </div>
            </body>
</html>

```

## > 16. ajax

Ajax는 "Asynchronous JavaScript and XML"의 약자로, 웹 페이지에서 비동기적으로 서버와

통신하는 기술입니다. Ajax를 사용하면 페이지를 새로 고치지 않고도 데이터를 로드하거나 서버로 데이터를 전송할 수 있습니다. 이로써 웹 페이지는 보다 동적이고 사용자 친화적인 경험을 제공할 수 있게 됩니다.

사용방법은 다음과 같이 jquery cdn를 추가 한다.

```
<!-- jQuery Framework -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
</script>
ajax 메서드를 사용한다.
$.ajax({
    url:"ajax/text/01.txt", //데이터를 얻어올 주소
    type:"get",           //전송방식
    data:"",             //querystring{key1:'value1',key2:'value2'} or id=p&pw=ab
    dataType:"text",      // 받아올 데이터 종료 html은 text, json, xml
    timeout:5000,         // 대기시간
    cache:false,          // 캐시 사용 유무
    success:function(data){ //성공한 경우 data에 01.txt 문자열을 읽어옴
        $("#result").html(data);
    },
    error:function(xhr,textStatus,errorThrown){ //실패할 경우 실행
        // 읽으려는 대상 주소가 잘못된 경우에 사용됩니다.
        $("div").html("<div>" + textStatus + " (HTTP-" + xhr.status + " / "
+ errorThrown + "</div>"); //error 404 not found
    }
})
```

01Text.html를 실행해서 버튼을 클릭 하면 01.text의 내용이 화면에 나온다.

← → ⌂ ① localhost:8081/07ajax/01Text.html

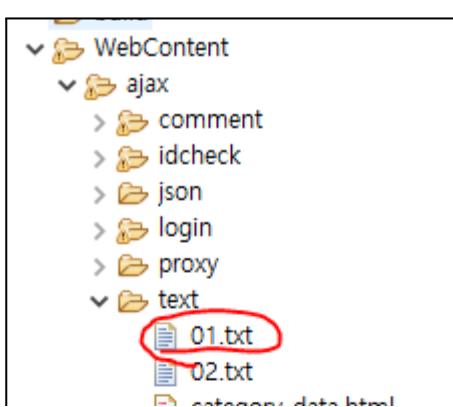
## \$ajax()함수를 사용한 텍스트 읽기

txt파일 가져오기

## \$ajax()함수를 사용한 텍스트 읽기

txt파일 가져오기

jQuery ajax 테스트 hello헬로~~



```
//01.text 내용
<br>
jQuery ajax 테스트 hello헬로~~
```

```

//01Text.html
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
<!-- jQuery Framework -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>
<!-- ajax 정의 -->
<script>
    $(function(){
        // 버튼의 클릭 이벤트
        $("#mybtn").click(function(){
            $.ajax({
                url:"ajax/text/01.txt",
                type:"get",
                data:"",
                dataType:"text",
                timeout:5000,
                cache:false,
                success:function(data){
                    $("#result").html(data);
                },
                error:function(xhr,textStatus,errorThrown){
                    $("div").html("<div>" + textStatus + " (HTTP-" + xhr.status + " / " +
                    errorThrown + ")</div>");
                    //error 404 not found
                }
            })
        });
    });
</script>
</head>
<body>
<h1>$.ajax()함수를 사용한 텍스트 읽기</h1>
<input type="button" value="txt파일 가져오기" id="mybtn"/>
<div class="console" id="result"></div>
</body>
</html>

```

02Text.html를 실행해서 버튼을 누르면 02.txt에 기술된 html를 읽어와서 버튼 클릭과 함께 화면에 그대로 html를 적용해 주는 예제이다.

# \$.ajax()함수를 사용한 텍스트 읽기

txt파일 가져오기

Ajax를 통해서 다른 파일을 읽어옵니다.

다른 내용은 HTML태그로 구성될 수 도 있습니다.



03Text3.html은 ajax로 데이터를 담아서 서버쪽에 전송해서 전송 받은 데이터를 가공하여 결과를 읽어 오는 예제이다.

```
$.ajax({  
    url:"ajax/text/textdata.jsp",  
    type:"get",  
    data:"keyword= park" + "&pass= abc123",  
    dataType:"text",  
    timeout:30000,  
    cache:false,
```

전송한 쿼리스트링을 가공하여 새로운 결과를 문자열로 받는다.

```
<%@ page language="java" contentType="text/text; charset=UTF-8"  
pageEncoding="UTF-8"%>  
<%@ page trimDirectiveWhitespaces="true" %> //공백을 없애기 위해 꼭 추가해야 한다.  
<%  
// 페이지 지시자에서 contentType을 "text/text"로 설정한 JSP파일은 출력 결과물이 텍스트  
파일로 인식된다.  
// 출력 결과물이 텍스트문서라는 점 외에는 JSP가 갖는 웹 프로그래밍적 부분이 그대로  
유지된다.
```

```
/** (1) 파라미터를 받는 것이 가능하다. */  
String keyword = request.getParameter("keyword");  
String keyword2 = request.getParameter("pass");  
  
/** (2) 이 부분에 Java의 모든 문법적 처리나 DATABASE의 연동 처리가 가능하다. */
```

```
/** (3) 결과를 출력한다.
```

여기는 받은 데이터를 그냥 찍지만 가공해서 찍을 수 있다.\* /

```
out.print(keyword);  
out.print(keyword2);  
%>
```

//04category.html  
동적 메뉴를 제공하는 프로그램이다. load 메서드는 ajax를 개선한 메소드로 load에 의해서  
불러온 데이터를 #category1에 넣는다. 콜백함수는 작업이 완료되고 실행되는 함수이다.

ajax/text/category-data.html #category1-1 은 해당 html파일에서 #category1-1 부분만 읽어 온다.

```
$( "#category1" ).load("ajax/text/category-data.html #category1-1",function(){
    // 로딩이 완료되면 드롭다운을 감싸는 태그 요소를 화면에 표시함
    $(this).show();
});
```

//05xml.html

다음 xml를 읽어와서 dom를 구성하는 예제이다.

```
<?xml version="1.0" encoding="utf-8" ?>
<school>
    <subject>
        <title>Javascript+jQuery+Ajax</title>
        <time>매주 월/수/금 오후 7시30분~10시20분</time>
        <teacher>주영아</teacher>
    </subject>
    <subject>
        <title>HTML5+CSS3 기반의 반응형 웹</title>
        <time>매주 화/목 오후 7시30분~10시20분</time>
        <teacher>주영아</teacher>
    </subject>
    <subject>
        <title>Java 입문에서 활용까지</title>
        <time>매주 화/목 오후 7시30분~10시20분</time>
        <teacher>이광호</teacher>
    </subject>
</school>
```

//06-idcheck4.html

xml.ajax(idcheck/idcheck\_ok.xml)를 이용해서 아이디 중복 체크하는 예제이다.

화면 간신없이 아이디 중복 체크를 할 수 있다.

//07-login8.html

xml.ajax(login/login\_ok.xml)를 이용해서 로그인 체크를 하는 예제이다.

화면 간신없이 로그인 체크를 할 수 있다.

//09weather.html WeatherServlet.java

ajax로 WeatherServlet를 요청해서 WeatherServlet.java 서블릿에서 원하는 xml를 얻어 날씨 정보를 얻어 온다.

```
$.ajax({
    url : "WeatherServlet", /*com.human.ex 안에 WeatherServlet 요청*/
    type : "get",
    data : {"cityname" : cityname},
```

다음 주소를 브라우저에 넣으면 xml 형태로 날씨관련 정보가 제공된다.

<http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp?stnId=108>

url : "WeatherServlet", 부분에

<http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp?stnId=108> 를 넣으면 바로 데이터를 받아서 처리할 수 있을 거 같지만 실행해 보면 크로스 도메인 때문에 읽어 올 수 없다.

크로스 도메인이란? ajax는 같은 서버에서만 가져다가 호출 할 수 있는데 기상청 서버는 다른 서버에 위치해서 읽어 올 수 없다. 이런 문제를 해결하기 위해서 카피 서버인

WeatherServlet를 만들어서 같은 서버에 날씨를 제공하는 서버를 만들었다.

weatherServlet은 ajax 요청이 아니므로 다른 서버에 있는 데이터를 읽어 올수 있고 읽어온 내용을 그대로 요청자에게 제공해주는 카피 서버이다.

<http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp?stnId=108> 에서 제공해주는 xml과 같은 내용을 가진다.

두서버의 내용은 같으나 weatherServlet은 같은 서버에 위치하여 ajax로 접근할 수 있고 다른 하나는 다른 서버에 위치 하므로 ajax로 접근 할 수 없다.

//10-google-news.html

구글 기사 관련 api를 사용하는 방법이다.

//11-json(1).html

04.json 데이터를 읽어와서 처리하는 예제이다. dataType: "json",

<!DOCTYPE html>

<html lang="ko">

  <head>

    <meta charset="utf-8" />

  <title>json(1)</title>

  <!-- jQuery Framework 참조하기 -->

  <script type="text/javascript" src="js/jquery-1.11.0.min.js"></script>

<!-- 사용자 스크립트 블록 -->

<script type="text/javascript">

  \$(function() {

    \$("#mybtn").click(function() {

      \$.ajax({

        url: "ajax/json/04.json",

        type: "get",

        dataType: "json",

        cache: false,

        timeout: 30000,

        success: function(json) {

          // 파라미터로 전달되는 객체는 JSON 자체이다.

          var title = json.school.subject.title;

          var time = json.school.subject.time;

```

var teacher = json.school.subject.teacher;

    /** 요소의 동적 생성 및 데이터 출력 구현 */
    var ul = $("<div>");
    var li1 = $("<p>").html(title);
    var li2 = $("<p>").html(time);
    var li3 = $("<p>").html(teacher);
    $("#result").append(ul.append(li1).append(li2).append(li3));
},
error: function(xhr, textStatus, errorThrown) {
}
});
});
});

```

</script>

</head>

<body>

<h1 class="title">\$.ajax() 함수를 사용한 JSON데이터 읽기 (1)</h1>

<div class="exec">

<input type="button" value="JSON데이터 가져오기" id="mybtn" />

</div>

<div class="console" id="result"></div>

</body>

</html>

//12-json(2).html  
json데이터를 가져와서 처리하는 예제이다.

### 국가에서 제공하는 데이터 확인하기

The screenshot shows the homepage of the National Data Portal (data.go.kr). The search bar at the top contains the query "한국환경공단\_에어코리아\_대기오염정보". Below the search bar, there are dropdown menus for "검색조건" (Search Conditions), "분류체계" (Classification System), "서비스유형" (Service Type), and "확장자" (File Extension). To the right of the search bar, there is a button labeled "3. 날씨" (Weather) with a dropdown arrow. The page has a blue header with navigation links for "로그아웃" (Logout), "마이페이지" (My Page), "사이트맵" (Site Map), and "ENGLISH". The main content area displays search results, with the first result being highlighted in red.

[www.data.go.kr](http://www.data.go.kr) 사이트에 들어가 원하는 공공 데이터를 상위처럼 로그인 한 다음 검색한다.

전체(9건) | 파일데이터(2건) | **오픈 API(7건)** | 표준데이터셋0개(0건)

정확도순 ▾ | 10개씩 ▾ | 정렬

### 오픈 API (7건)

**한국환경공단\_에어코리아\_대기오염정보**

각 측정소별 대기오염정보를 조회하기 위한 서비스로 기간별, 시도별 대기오염 정보와 통합대기환경지수 나쁨 이상 측정소 내역, 대기질(미세먼지/오존) 예보 통보 내역 등을 조회할 수...

제공기관 한국환경공단 수정일 2023-11-07 조회수 141797 활용신청 25481 키워드 미세먼지,코로나,오존

**한국환경공단\_에어코리아\_대기오염통계 현황**

대기오염 통계 정보를 조회하기 위한 서비스로 각 측정소별 농도 정보와 기간별 통계수치 정보를 조회할 수 있다.

제공기관 한국환경공단 수정일 2023-11-07 조회수 42499 활용신청 5299 키워드 미세먼지,코로나,오존

검색 결과에서 오픈 API를 선택하고 하위 목록에서 원하는 항목을 선택 한다.

### 오픈API 상세

**한국환경공단\_에어코리아\_대기오염정보**

각 측정소별 대기오염정보를 조회하기 위한 서비스로 기간별, 시도별 대기오염 정보와 통합대기환경지수 나쁨 이상 측정소 내역, 대기질(미세먼지/오존) 예보 통보 내역 등을 조회할 수 있다.

※ 운영계정으로 사용하고자 할 경우 '한국환경공단 에어코리아 OpenAPI 기술문서' 내 신청 가이드 참고

60 | 24 | 관심

활용신청

활용 신청을 클릭하여 사용신청을 한다.

박수민님, 반갑습니다.

회원정보 수정 >

파일데이터	API신청	관심데이터						
0건	2건	0건						
개인 API인증키								
<table border="1"> <thead> <tr> <th>구분</th> <th>발급일자</th> <th>인증키</th> </tr> </thead> <tbody> <tr> <td>일반</td> <td>2023-11-10</td> <td>0xbH6Jl1tTqh/IC4yGcvnQCWTGWPvDTPz4qFSOQ... <b>인증키 복사(Encoding)</b> <b>인증키 복사(Decoding)</b></td> </tr> </tbody> </table>			구분	발급일자	인증키	일반	2023-11-10	0xbH6Jl1tTqh/IC4yGcvnQCWTGWPvDTPz4qFSOQ... <b>인증키 복사(Encoding)</b> <b>인증키 복사(Decoding)</b>
구분	발급일자	인증키						
일반	2023-11-10	0xbH6Jl1tTqh/IC4yGcvnQCWTGWPvDTPz4qFSOQ... <b>인증키 복사(Encoding)</b> <b>인증키 복사(Decoding)</b>						
더보기 >								

마이페이지에서 인증키 복사 부분을 클릭해서 인증키를 얻는다.

<input type="checkbox"/> 환경기상	<input type="checkbox"/> 한국환경공단
<b>활용신청 [승인] 한국환경공단_에어코리아_대기오염정보</b>	
계정 개발	신청일 2023-11-10 만료예정일 2025-11-10
<input type="checkbox"/> 문화관광 <input type="checkbox"/> 충청북도 괴산군	
<b>활용신청 [승인] 충청북도 괴산군_낚시터관리대장목록</b>	
계정 개발	신청일 2023-11-10 만료예정일 2025-11-10

api를 신청률 클릭 한다음 상단처럼 대고오염정보 목록을 선택 한다.



<https://www.data.go.kr/iim/api/selectAPIAccontView.do>  
확인 버튼을 누른다.

NO	상세기능	설명	일일 트래픽	미리보기
1	대기질 예보통보 조회	통보코드와 통보시간으로 예보정보와 발생 원인 정보를 조회하는 대기질(미세먼지/오존) 예보통보 조회	500	<input type="button" value="확인"/>

**요청변수(Request Parameter)**

항목명	샘플데이터	설명
serviceKey	0txbH6JvI1tTqh%2FIC4yG	공공데이터포털에서 받은 인증키
returnType	json	xml 또는 json
numOfRows	100 <small>샘플데이터</small>	한 페이지 결과 수(조회 날짜로 검색 시 사용 안함)
pageNo	1	페이지번호(조회 날짜로 검색 시 사용 안함)
searchDate		통보시간 검색(조회 날짜 입력이 없을 경우 한달동안 예보통보 발령 날짜의 리스트 정 보를 확인)
InformCode	PM10 <small>샘플데이터</small>	통보코드검색(PM10, PM25, O3)

searchDate를 지우고 serviceKey에 아까 복사한 주소를 넣은 다음 미리 보기 선택하면 검색 가능한 날짜가 나온다.

날짜하나를 복사해서 다시 searchDate에 넣어 보자.

해당 날짜의 요청 데이터를 확인할 수 있다.

<https://apis.data.go.kr/B552584/ArpltnInforInquireSvc/getMinuDustFrcstDspth?serviceKey=0tXbH6JvI1tTqh%2FIC4yGcvnQCWTGWPvDTPz4qFS0Qo7ICrF5dqRWBhFy5fscHKz%2FJmA67k17lHX0%2FHYweDcYA%3D%3D&returnType=json&numOfRows=100&pageNo=1&informCode=PM>

미리 보기 버튼을 누르면 상위와 같은 주소로 브라우저에 입력되어 있다. 복사해서 원하는 데이터를 가져오는 용도로 프로그램에서 사용하면 된다.

자세한 사용방법은 사이트에서 잘 확인해보면 기술문서와 같은 사용 문서를 찾을 수 있다. 기술 문서를 잘 읽고 api를 사용해 보자.

검색하면 도서, 뉴스터, 버스 알림이 같은 다양한 api가 있다. 필요한 api를 검색해서

프로그램을 구현해 보자.

JSON 데이터는 미세먼지 정보 및 예보 데이터를 포함하고 있습니다. 주요 키의 의미는 다음과 같습니다:

- **informCode**: 정보 코드 (예: PM10, PM25), 미세먼지 종류를 나타냅니다.
- **informCause**: 미세먼지 발생 원인에 대한 설명.
- **informOverall**: 전반적인 미세먼지 상태 요약.
- **informGrade**: 지역별 미세먼지 등급.
- **informData**: 예보 데이터 날짜.
- **dateTime**: 정보 발표 시각.
- **imageUrl1 ~ imageUrl6**: 예보 이미지를 나타내는 URL (각 시간대별 상태).

## 주요 정보 추출

예를 들어 특정 날짜의 요약 및 등급 정보를 보고 싶다면 다음과 같이 확인할 수 있습니다:

- **날짜별 요약 정보**
  - 2024-12-25: "전 권역이 '좋음'~'보통'으로 예상됩니다."
  - 2024-12-26: "수도권·강원영서·세종·충북은 '나쁨', 그 밖의 권역은 '보통'으로 예상됩니다."
  - 2024-12-27: "전 권역이 '좋음'으로 예상됩니다."
- **서울 지역 등급**
  - 2024-12-25: 보통
  - 2024-12-26: 보통
  - 2024-12-27: 좋음

## 시각화

이미지 URL을 활용하여 시간별 미세먼지 상태를 시각화할 수 있습니다. 예를 들어, **imageUrl1**은 2024-12-25 21시의 예보 이미지를 나타냅니다.

## 데이터를 활용한 기능 제안

1. **날짜별 데이터 요약 출력**: 각 날짜에 대해 **informOverall**, **informGrade**를 정리.
2. **지역별 데이터 필터링**: 특정 지역(예: **서울**)의 등급을 날짜별로 정리.
3. **이미지 시각화 링크 제공**: 시간대별 이미지 링크를 UI에 표시.

추가적으로 JSON을 다루는 데 도움이 필요하면 알려주세요!

대체서비스: 기상청 API허브 (<https://apihub.kma.go.kr>) > 예특보 > 단·중기예보

1. 서비스 명세

### 1.1 단기예보 조회서비스

가. API 서비스 개요

API 서비스 정보	API 명(영문)	VillageFcstInfoService_2.0
	API 명(국문)	단기예보 조회서비스[2.0]
	설명	조단기집합, 조단기예보, 단기예보, 예보배경 정보; 조회하는 서비스입니다. 조단기집합정보는 예보 구역에 대한 대체 AWS 관리값을, 조단기예보는 예보시작부터 6시간 미내의 예보[1], 단기예보는 예보기간과 구역을 시공간적으로 세밀한 예보[2] 제공합니다.
API 서비스 보안적용 기술 수준	서비스 인증/암호화	<input checked="" type="checkbox"/> 암호화 (GPG/NPKI) <input type="checkbox"/> Basic (ID/PW) <input checked="" type="checkbox"/> HTTPS
	메시지 랜턴 암호화	<input checked="" type="checkbox"/> 전자서명 <input checked="" type="checkbox"/> 암호화 <input checked="" type="checkbox"/> 암호화
	전송 랜턴 암호화	<input checked="" type="checkbox"/> SSL <input checked="" type="checkbox"/> 암호화 (RPC-Encoded, Document Literal, Document Literal Wrapped)
	인터넷메시 표준	<input checked="" type="checkbox"/> SOAP 1.2 <input checked="" type="checkbox"/> REST (GET) <input checked="" type="checkbox"/> RSS 1.0 <input checked="" type="checkbox"/> RSS 2.0 <input checked="" type="checkbox"/> Atom 1.0 <input checked="" type="checkbox"/> 기타
	교환 데이터 표준 (동록선택 가능)	<input checked="" type="checkbox"/> XML <input checked="" type="checkbox"/> JSON <input checked="" type="checkbox"/> MIME <input checked="" type="checkbox"/> MTOM
API 서비스 배포	서비스 URL	http://apis.data.go.kr/1360000/villageFcstInfoService_2.0
	서비스 명세 URL (WSDD 또는 WSDL)	N/A
	서비스 버전	1.0

나. 상세기능 목록

번호	API 명(국문)	상세기능명(영문)	상세기능명(국문)
1		getUltraFcstNctv	조단기 예보조회[1]
2		getUltraFcstv	조단기 예보조회[2]
3		getVilageFcstv	단기예보조회[1]
4		getFcstVersionv	예보버전조회[2]

다. 상세기능내역

1) [조단기설명] 상세기능概要

2) 상세기능 등록

상세기능 번호	1	상세기능 유형	조회 (목록)
상세기능명(영문)	조단기설명조회[1]		
상세기능 설명	설명정보를 조회하기 위해 필요할자, 발표시각, 예보자집 X 외로, 예보지점 Y 외로의 조작 조건으로 자료구조코드, 출처값, 발표필자, 발표시각, 예보자집 X 외로, 예보자집 Y 외로의 정보를 조회하는 기능[1]		
Call Back URL	http://apis.data.go.kr/1360000/villageFcstInfoService_2.0/getUltraFcstNctv		
최대 메시지 사이즈	(1764) byte		
평균 대기 시간	[100] ms	조단 대기 노력작성	[30] tps

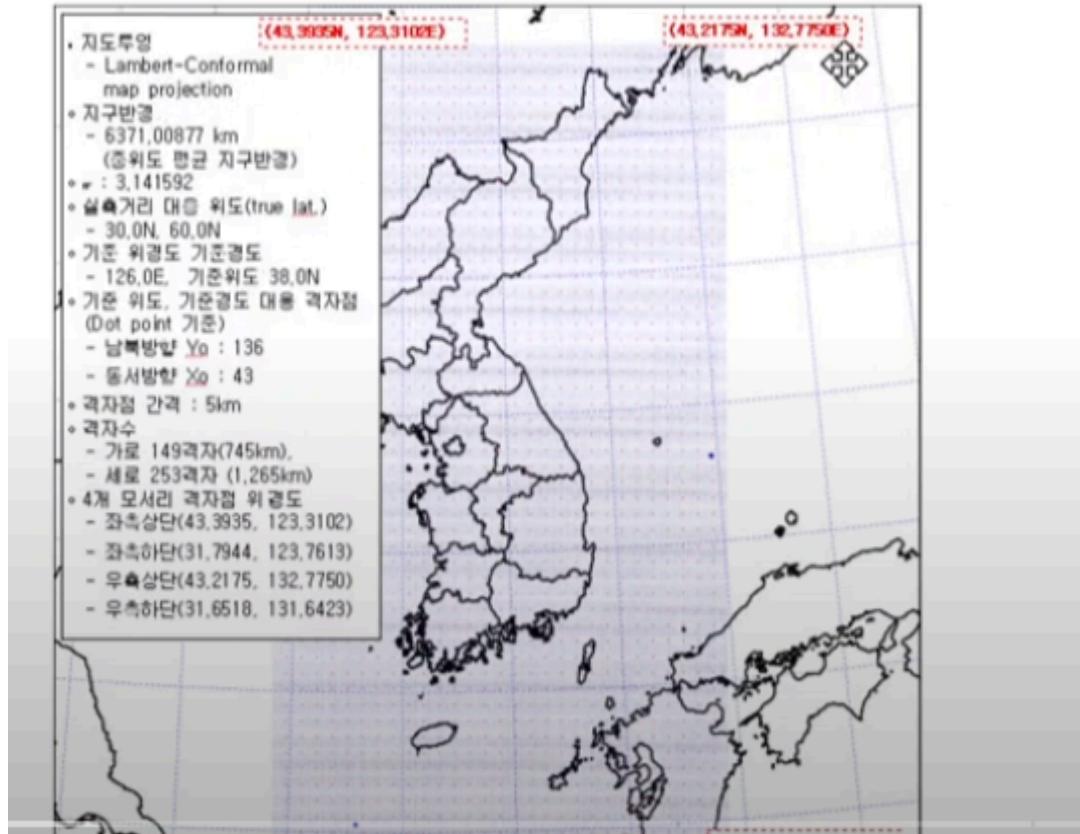
b) 요청 메시지 양세

항목명(영문)	항목명(국문)	항목크기	항목구조	샘플데이터	항목설명
serviceKeyv	인증키	100	1	인증키	간접데이터호출에서 (URL Encode)
numOfrowsv	한 페이지 결과 수	40	1	10	한 페이지 결과 수 Default: 10
pageNov	페이지 번호	40	1	1	페이지 번호 Default: 1

$nx$ ,  $xy$  관련 정보를 문서에서 확인해 보자.

#### 참고] 예보 영역 :

동네예보를 위해 설정한 영역은 Fig. 1과 같이 한반도와 서해 5도를 포함한 우리나라와 인근 해역으로 남쪽으로는 이어도, 동쪽으로는 독도, 서쪽 끝으로는 백령도 까지 포함하고 있다. 이 영역은 5km×5km의 격자 간격으로 총 37,697개의 격자를 포함한다(동서 149개 ×남북 253개).



격자 정보 상위 처럼 그리드를 만들어서 작은 박스의 위치를 숫자로 얻어내는 좌표 엑셀에서 검색을 통해서 좌표를 얻을 수 있다.

구글 맵에서 원하는 위치에 마우스 오른쪽 클릭을 하면 격자 정보를 얻을 수 있다.

위도 경도 정보로 격자정보 얻는 사이트

<https://fronteer.kr/service/kmaxy>

기상청 날씨 API

jQuery 135 [ weather API #1/5 ] 기상청 날씨 API 신청하기(매우 짧습니다^^)

https://www.data.go.kr/

**기상청\_단기예보 ((구)\_동네예보) 조회서비스**

**날씨 예보 테스트**

The screenshot shows a web application interface for applying for a weather API. It includes sections for service details, service configuration, and a test result. The test result shows a successful response from the 'National Weather Forecasting Service' with a status code of 200.

설명	내용
설명	기상청_단기예보 ((구)_동네예보) 조회서비스
설명	날씨 예보 테스트
설명	날씨 예보 테스트

로그인해서 들어가면 해당 요청을 테스트할수 있는 사이트에 접근할 수 있다.

\* 향후 포털에서 더 명확한 정보를 제공하기 위해 노력하겠습니다.

일반 인증키 (Encoding)	0tXbH6JvI1tTqh%2FIC4yGcvnQCWTGWPvDTPz4qFSOQo7lCrF5dqRBhFy5fscHKz%2FjmA67kl7lHX0%2FHweDcYA%3D%3D
일반 인증키 (Decoding)	0tXbH6JvI1tTqh/lC4yGcvnQCWTGWPvDTPz4qFSOQo7lCrF5dqRBhFy5fscHKz/JmA67kl7lHX0/HYweDcYA==

### 활용신청 상세기능정보

NO	상세기능	설명	일일 트래픽	미리보기
1	초단기실황조회	실황정보를 조회하기 위해 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 조건으로 자료구분코드, 실황값, 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 정보를 조회하는 기능	10000	<button>확인</button>

### 요청변수(Request Parameter)

[닫기](#)

항목명	샘플데이터	설명
ServiceKey	0tXbH6JvI1tTqh%2FIC4yG	공공데이터포털에서 받은 인증키
pageNo	1 <a href="#">샘플데이터</a>	페이지번호
numOfRows	1000	한 페이지 결과 수
dataType	XML	요청자료형식(XML/JSON) Default: XML
base_date	20210628	'21년 6월 28일 발표
base_time	0600	06시 발표(정시단위)
nx	55	예보지점의 X 좌표값
ny	127	예보지점의 Y 좌표값

[미리보기](#)

제목을 클릭해서 원하는 위치 검색 가능

구분	행정구역 코드	단계	단계	단계	경자 X	경자 Y	경도(시)	경도(도)	위도(시)	위도(도)
175.kor	4157055000	경기도	김포시	정부5	55	127	126	43	37	36
3795										
3796										
3797										
3798										
3799										
3800										
3801										
3802										
3803										
3804										
3805										
3806										
3807										

getJSON

```
integrity="sha256-pvPw+upLPUjgMXY0G+800xUf+/
Im1MZjXxxg0cBQBXU="
crossorigin="anonymous"
></script>
<script>
$.getJSON(
  "https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/
  getUltraSrtNcst?
  serviceKey=dx%2BmKKHuGEyuQWSpZxKjyx4x3fBhfVDIPE00nZE0FOZ5yk
  vTy%2B8X8i8B%2B1h1ukFka8qe27r7pN%2B5maYk541lgQ%3D%3D&
  pageNo=1&numOfRows=1000&dataType=JSON&base_date=20230118&
  base_time=0600&nx=60&ny=127",
  function (data) {
    console.log(data);
    console.log(data.response.body.items.item[3].obsrValue);
  }
);
</script>
</body>
</html>
```

Live reload enabled. index.html:56  
Default levels ▾ No Issues  
Live reload enabled. index.html:24  
Object {  
 response:  
 body:  
 dataType: "JSON"  
 items:  
 item: Array(8)  
 0: {baseDate: "20230118", baseTime: "0600", category: "T1H", nx: 60, ny: 127, obsrValue: "-4.3"}  
 1: {baseDate: "20230118", baseTime: "0600", category: "T1H", nx: 60, ny: 127, obsrValue: "-4.3"}  
 2: {baseDate: "20230118", baseTime: "0600", category: "T1H", nx: 60, ny: 127, obsrValue: "-4.3"}  
 3: {baseDate: "20230118", baseTime: "0600", category: "T1H", nx: 60, ny: 127, obsrValue: "-4.3"}  
 4: {baseDate: "20230118", baseTime: "0600", category: "T1H", nx: 60, ny: 127, obsrValue: "-4.3"}  
 5: {baseDate: "20230118", baseTime: "0600", category: "T1H", nx: 60, ny: 127, obsrValue: "-4.3"}  
 6: {baseDate: "20230118", baseTime: "0600", category: "T1H", nx: 60, ny: 127, obsrValue: "-4.3"}  
 7: {baseDate: "20230118", baseTime: "0600", category: "T1H", nx: 60, ny: 127, obsrValue: "-4.3"}  
 length: 8  
 [[Prototype]]: Object  
 }  
 4: {baseDate: "20230118", baseTime: "0600", category: "T1H", nx: 60, ny: 127, obsrValue: "-4.3"}  
 }  
 [[Prototype]]: Object  
 numOfRows: 1000  
 pageNo: 1  
 totalCount: 8  
 [[Prototype]]: Object  
 resultCode: "00"

The screenshot shows a browser window with two panes. The left pane displays the source code of 'index\_ajax.html'. The right pane shows the results of the API call, which includes the date and time of the request.

```
<!-- index_ajax.html -->
<!-- index_ajax.html > html > body > script > success > content
16   src="https://code.jquery.com/jquery-3.6.3.min.js"
17   integrity="sha256-pvPw+upLPUjgMXY0G+800xUF+/Im1MZjXxxg0cBQBXU="
18   crossorigin="anonymous"
19 ></script>
20 <script>
21   $.ajax({
22     url: "https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtNcst?
23     serviceKey=dX%2BmKKHuGEyuQWSpZxKjyx4x3f8hfVDIPE0nZE0FOZ5ykTy%2B8X8i8B%2BlhiukF
24     a8qe27r7pN%2B5maYk541lgQ%3D%3D&pageNo=1&numOfRows=1000&dataType=JSON&
25     base_date=20230118&base_time=0600&nx=60&ny=127",
26     success: function (result) {
27       console.log(result);
28       let item = result.response.body.items.item[3];
29       let content =
30         item.baseDate +
31         "," +
32         item.baseTime +
33         "," +
34         item.obsrValue +
35         "입니다";
36         $(".result").text(content);
37     },
38   });
39 </script>
40 </body>
41 </html>
```

기상청 날씨 API  
종로 날씨  
20230118.0600-4.3입니다

## 단기 예보 확인하기

조회하는 기능	
단기예보조회	단기예보 정보(조회하기 위해 날짜입자, 날짜시각, 예보지점 X좌표, 예보지점 Y 좌표의 조회 조건으로 날짜일자, 날짜시각, 자료구분문자, 예보 값, 예보일자, 예보시각, 예보지점 X 좌표, 예보지점 Y 좌표의 정보) 조회하는 기능
하인	10000

### 요청변수(Request Parameter)

521

항목명	생성데이터	설명
ServiceKey	dX%2BmKKHuGEyuQWSf	공공데이터포털에서 받은 인증키
pageNo	1	페이지번호
numOfRows	1000	한 페이지 결과 수
dataType	json	요청자료형식(XML/JSON) Default: XML
base_date	20230118	'21년 6월 28일 발표
base_time	0500	05시 발표
nx	60	예보지점의 X 좌표값
ny	127	예보지점의 Y 좌표값

미리보기

4 예보비정조회	단기예보정보조회서비스 각각의 오퍼레이션(초단 기실행 조단기예보 단기예보)▷의 수경회 예보	10000	<input type="button" value="하이"/>
----------	--	-------	-----------------------------------

11개씩 배열에 다을 시간 데이터가 들어 있다. 06시 07시

DevTools - 127.0.0.1:5500/index\_forecast.html

Elements Console Network Performance Memory Sources Application Security Lighthouse > |

top Filter Default levels ▾ No Issues

Live reload enabled.

index forecast.html:56  
index forecast.html:24

▼ Object ▾  
  ▼ response:  
    ▼ body:  
      dataType: "JSON"  
      ▼ items: Array(809)  
        ▼ [0 ... 99]  
          ▶ 0: {baseDate: '20230118', baseTime: '0500', category: 'TMP', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 1: {baseDate: '20230118', baseTime: '0500', category: 'UUU', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 2: {baseDate: '20230118', baseTime: '0500', category: 'VVV', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 3: {baseDate: '20230118', baseTime: '0500', category: 'VEC', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 4: {baseDate: '20230118', baseTime: '0500', category: 'WSD', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 5: {baseDate: '20230118', baseTime: '0500', category: 'SKY', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 6: {baseDate: '20230118', baseTime: '0500', category: 'PTY', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 7: {baseDate: '20230118', baseTime: '0500', category: 'POP', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 8: {baseDate: '20230118', baseTime: '0500', category: 'WAV', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 9: {baseDate: '20230118', baseTime: '0500', category: 'PCP', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 10: {baseDate: '20230118', baseTime: '0500', category: 'REH', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 11: {baseDate: '20230118', baseTime: '0500', category: 'SNO', fcstDate: '20230118', fcstTime: '0600', ...}  
          ▶ 12: {baseDate: '20230118', baseTime: '0500', category: 'TMP', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 13: {baseDate: '20230118', baseTime: '0500', category: 'UUU', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 14: {baseDate: '20230118', baseTime: '0500', category: 'VVV', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 15: {baseDate: '20230118', baseTime: '0500', category: 'VEC', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 16: {baseDate: '20230118', baseTime: '0500', category: 'WSD', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 17: {baseDate: '20230118', baseTime: '0500', category: 'SKY', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 18: {baseDate: '20230118', baseTime: '0500', category: 'PTY', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 19: {baseDate: '20230118', baseTime: '0500', category: 'POP', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 20: {baseDate: '20230118', baseTime: '0500', category: 'WAV', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 21: {baseDate: '20230118', baseTime: '0500', category: 'PCP', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 22: {baseDate: '20230118', baseTime: '0500', category: 'REH', fcstDate: '20230118', fcstTime: '0700', ...}  
          ▶ 23: {baseDate: '20230118', baseTime: '0500', category: 'SNO', fcstDate: '20230118', fcstTime: '0700', ...}

The screenshot shows a browser window with developer tools open. The address bar displays the URL: `https://apis.data.go.kr/1360000/VillageFcstInfoService_2/getVilageFcst?`. The status bar at the bottom of the browser window also shows this URL. The developer tools sidebar on the right has a red circle drawn around it, highlighting the status bar area. A red arrow points from the status bar in the browser to the URL in the developer tools sidebar.

```
<!-- 기상청 날씨 API -->
<!-- https://apis.data.go.kr/1360000/VillageFcstInfoService_2/getVilageFcst? -->
<!-- serviceKey=dX%2BmKKHuGEyuQlWSpZxKjyx4x3fBhfVDIPE00nZE0F0Z5ykvTy%2B8X8i8B%2Blh1ukFka8qe27r7pN%2B5maYk541lg%3D%3D&pageNo=1&numOfRows=1000&dataType=json&base_date=${initDate}&base_time=0500&nx=60&ny=127 -->
success: function (result) {
  console.log(result);
}
```

시간별 단기 예보를 만들어 보자.

The screenshot shows a browser developer tools window with the 'Console' tab selected. The code being run is:

```

ex_forecast.html > html > body > script > weather > success
pageNo=1&numOfRows=1000&dataType=json&base_date=$
{initDate}&base_time=0500&nx=60&ny=127` ,
success: function (result) {
  console.log(result);
  let items = result.response.body.items.item;
  /*
  let filteredItems = [];
  for (let i = 0; i < items.length; i++) {
    if (items[i].category == "TMP") {
      filteredItems.push(items[i]);
    }
  }
  */
  let filteredItems = items.filter((item) => {
    return item.category == "TMP";
  });
  console.log(filteredItems);
},
} //날씨조회 테이블 생성 함수
weather(initDate);

```

The output in the console shows the filtered items:

```

(67) [{}]

```

## 온도만 출력 방법

The screenshot shows a browser developer tools window with the 'Console' tab selected. The code being run is:

```

Selection View Go ... ← → ⌂ 기상청_날씨 [Administ...]
x_forecast.html > html > body > script > makeTable

```

```

function makeTable(src) {
  let tableHTML = "";
  src.forEach((item) => {
    tableHTML += `
      <tr>
        <td>${item.fcstDate}</td>
        <td>${item.fcstTime}</td>
        <td>${item.fcstValue}</td>
      </tr>`;
  });
  $("table tbody").html(tableHTML);
}
</script>
</body>
</html>

```

The output in the console shows the filtered items:

```

(67) [{}]

```

## 카카오 api 북 영화

카카오 맵 사용하기  
<https://www.youtube.com/watch?v=T23PHQbeZ6E>

2024-07-29 오후 03:30 조회

json.response.header.resultMsg:  
NORMAL\_SERVICE

baseDate :20240729  
baseTime :1530  
category :T1H  
fcstDate :20240729  
fcstTime :1600  
온도(fcstValue):28  
nx :55  
ny :127

fcstDate	fcstTime	fcstValue
20240729	1600	28
20240729	1700	28
20240729	1800	28
20240729	1900	27
20240729	2000	26
20240729	2100	26

```
<!--          :          -->
<%           ="java"           ="text/html; charset=UTF-8"           ="UTF-8"%>
<!DOCTYPE      >
<    >
<    >
<           ="UTF-8">
<    >           </    >

<!--          -->
<
  ="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></        >

<    >
/* //현재 날씨

//https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtNcst?serviceKey=
0tXbH6JvI1tTqh%2FIC4yGcvnQCWTGWPvDTPz4qFS0Qo7ICrF5dqRWBhFy5fscHKz%2FJmA67k17lHX0%2FH
eDcYA%3D%3D&pageNo=1&numOfRows=1000&data_Type=JSON&base_date=20240726&base_time=0600&nx=
55&ny=127

//https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getVilageFcst?serviceKey=59
DlkKuPMMfiNa2kZi8d0fwyWyk6luQkC5Zim5MJJjGavqqvv05a8WGJnSYdQq0Wj7Fnw5a3tZCi96mEbnlohQ%3D
```

```

%3D&pageNo=1&numOfRows=1000&dataType=JSON&base_date=20240726&base_time=1700&nx=55&ny=12
7 */
$ function
    $ ".btn" click function
        preventDefault
        //alert($("#weatherTime").val()); //2024-07-09T18:08
        var = new $("#weatherTime" val

            var = getFullYear //2024
            var = '0' + getMonth + 1 //1월 0
            var = substr -2 //항상 뒤에서 부터 2자리로 만든다.
            var = '0' + getDate
            var = substr -2

            var = $ "#weatherTime" val substr -5 2
                +
$ "#weatherTime" val substr -2
                //1getUltraSrtFcst
                var =
"59D1kKuPMMfiNa2kZi8d0fwyWyk6luQkC5Zim5MJJjGavqqvv05a8WGJnSYdQq0Wj7FnW5a3tZCi96mEbnlohQ
%3D%3D"
                var = "1"
                var = "100"
                var = "JSON"
                var = + +
                var =
                var = "55"
                var = "127"

                var =
"https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtFcst?"
                +
                "serviceKey="+
                "&pageNo="+
                "&numOfRows="+
                "&dataType="+
                "&base_date="+
                "&base_time="+
                "&nx=" +
                "&ny=" +

                alert + ":" +
                log
//https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtFcst?serviceKey=
59D1kKuPMMfiNa2kZi8d0fwyWyk6luQkC5Zim5MJJjGavqqvv05a8WGJnSYdQq0Wj7FnW5a3tZCi96mEbnlohQ%
3D%3D&pageNo=1&numOfRows=100&dataType=JSON&base_date=20240729&base_time=1502&nx=55&ny=1
27
//var
myUrl="https://apis.data.go.kr/1360000/VilageFcstInfoService_2.0/getUltraSrtFcst?serviceKey=59D1kKuPMMfiNa2kZi8d0fwyWyk6luQkC5Zim5MJJjGavqqvv05a8WGJnSYdQq0Wj7FnW5a3tZCi96mEbnlohQ%3D%3D&pageNo=1&numOfRows=100&dataType=JSON&base_date=20240729&base_time=1400&nx=55&ny=127";

```

```

ajax
:           //"/ajax/json/2getUltraSrtFcst.json",
: "get"
:   : "JSON"
: false
:   : function
alert "success"
var           =
if           == "NO_DATA"

$ ".headerMsg"  text
else if           == "NORMAL_SERVICE"
$ ".headerMsg"  text "NORMAL_SERVICE"

var           = 24
var           ="<br>
+= " <br>baseDate :+
+ "<br>baseTime
:"+
+ "<br>category
:"+
+ "<br>fcstDate
:"+
+ "<br>fcstTime
:"+
+ "<br>
온도(fcstValue):"+
+ "<br>nx :"+
+ "<br>ny :"+

$ ".result"  html

var           =
var           ="<table border=1>" 24+
+= "<tr>
+= "<th>fcstDate</th>
+= "<th>fcstTime</th>
+= "<th>fcstValue</th>
+= "</tr>
for var  =0  <6  ++
+= "<tr>

+= "<td>"+ 24+
+"</td>" 24+
+= "<td>"+ 24+
+"</td>" 24+
+"</td>"

+= "</tr>"
```

```
+="</table>"  
$ ".table"  html  
  
log  
  
: function  
alert "fail"  
$ ".error"  html  
    "<div>" +           + " (HTTP- " +  
+ " / " +  
")</div>"  
  
  
  
</>  
  
</>  
<>  
<>  
  <>  
    <>      ="datetime-local"      ="date"      ="weatherTime">  
    <>      ="submit"      ="btn" class="btn">  </>  
</>  
<>  
  : <>      class="headerMsg"></>  
<>      class="result"></>  
<>      class="error"></>  
<>      class="table"></>  
  
</>  
 </html>
```