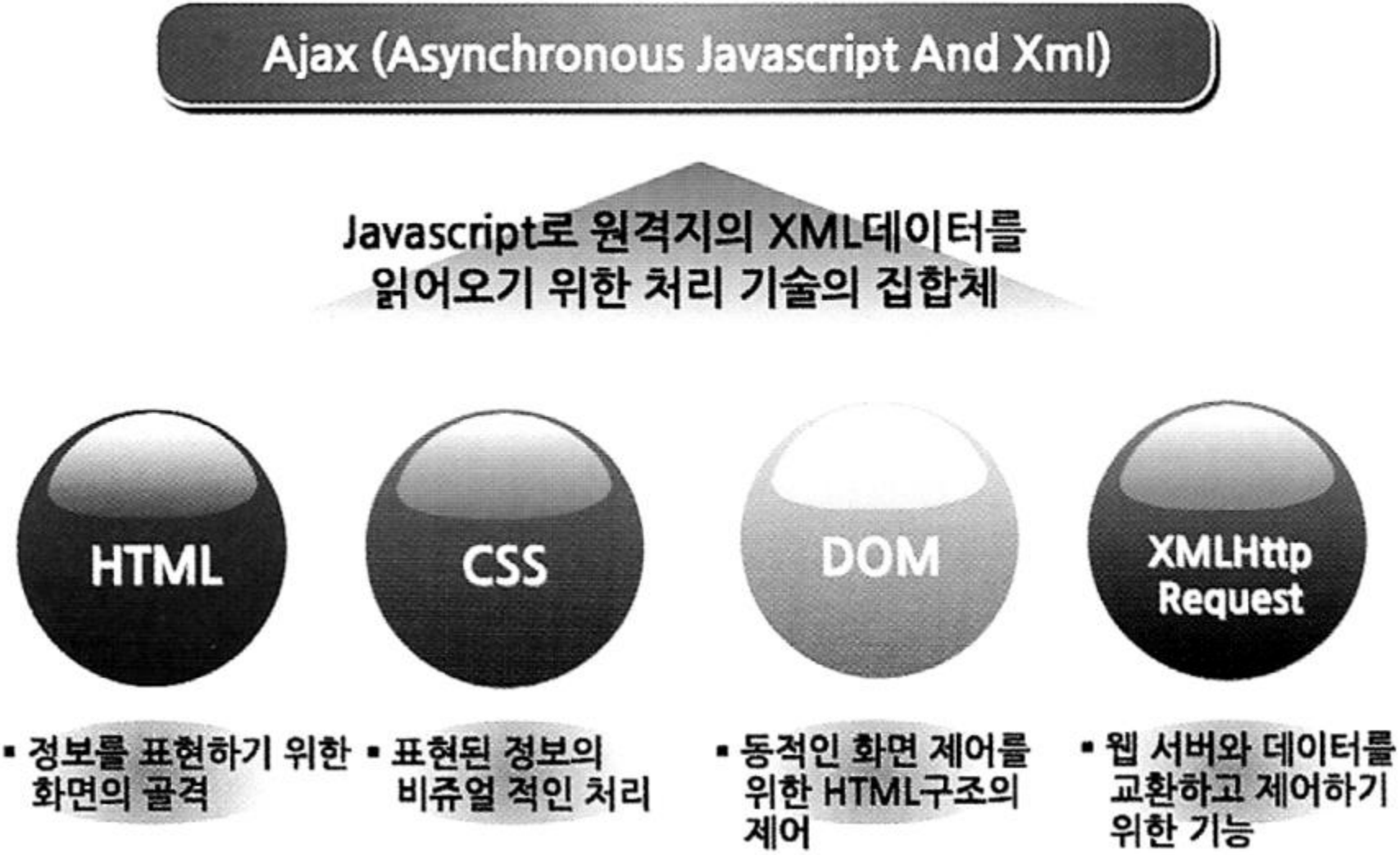
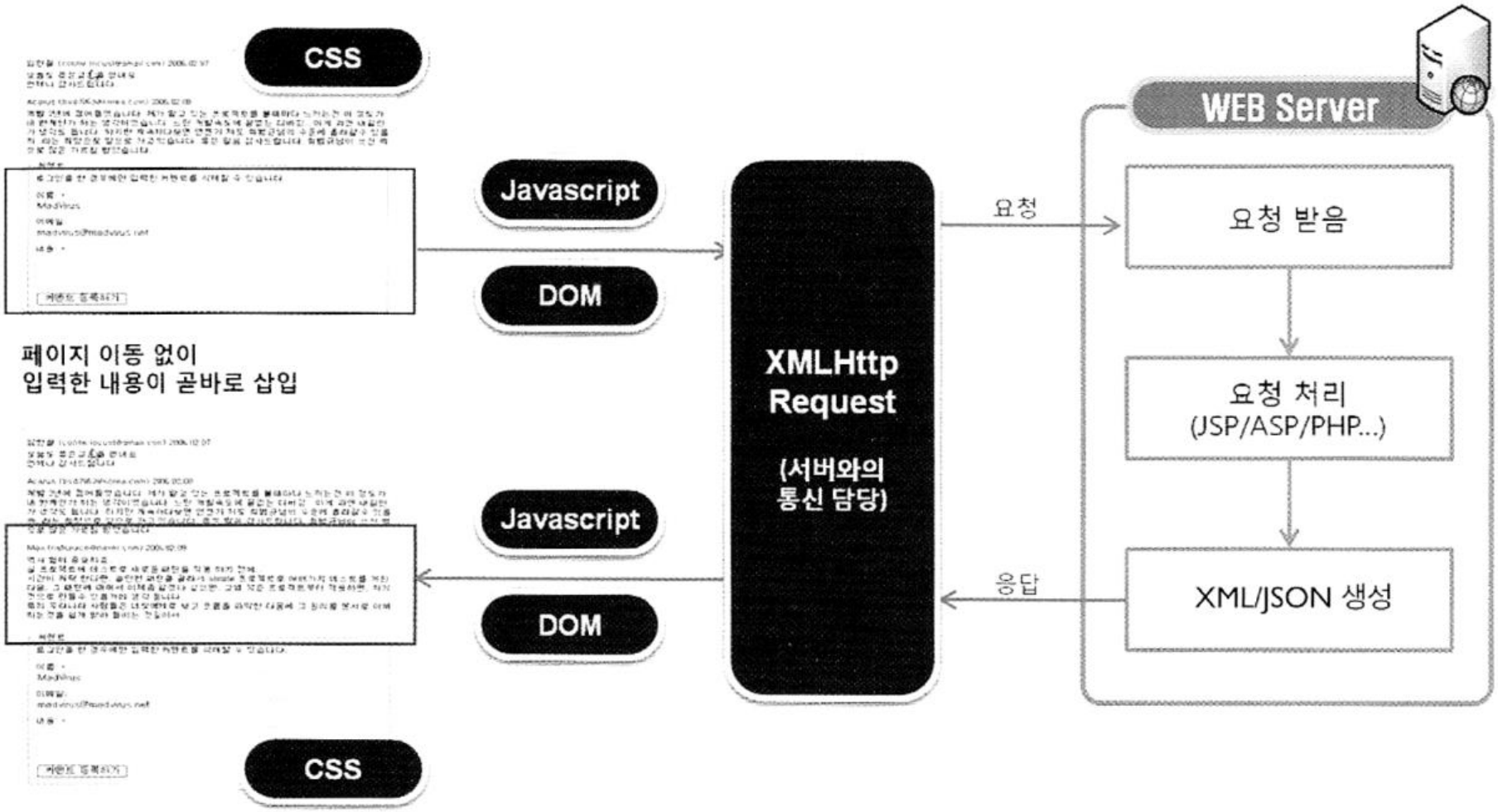
Ajax

❖ Ajax(Asynchronous Javascript And Xml)



❖ Ajax 방식의 데이터 처리

❖ Ajax 방식의 데이터 처리



❖ $.ajax()

- jQuery의 ajax 처리 함수 함수

```
$.ajax({
    "url": "접속할 페이지 주소",
    "type": "get/post",
    "data": "파라미터 문자열 key=value&key=value",
    "dataType": "text/xml/json/jsonp",
    "timeout": 밀리세컨드단위 제한시간,
    "cache": 이전 요청에 대한 캐쉬 저장 여부 (true=사용함, false=사용안함),
    "success": function(data) {
        // 통신이 성공했을 때 실행되는 함수.
    },
    "error": function(xhr, textStatus, errorThrown) {
        // 통신에 실패했을 때 실행되는 함수.
    }
});
```

❖ 콜백 함수

- success(data)
  - 웹 서버에서 전달하는 결과값을 정상적으로 읽어 온 경우 실행
  - 파라미터로 서버에서 전송한 값 전달
  - 요청시 dataType에 text, xml, json, jsonp 중 어떤 것을 지정했는지에 따라 변경

- error(xhr, statusText, errorThrown)
  - 웹 서버에러 시 실행(400 대 에러, 500 대 에러)
  - xhr : XMLHttpRequest 객체
  - statusText : "ERROR"
  - errorThrown : 에러 상세 문자열

Ajax
- 텍스트 파일 로딩 -

텍스트 파일 로딩

❖ 응답 파일 준비

- text/01.txt (UTF-8 인코딩)

```
jQuery AJAX 테스트
```

텍스트 파일 로딩

❖ 텍스트 파일을 위한 $.ajax( )

- dataType을 text로 설정

```
$.ajax({
    url : "읽어올 파일의 URL",
    type : "get",
    dataType : "text",
    timeout : 30000,
    cache : false,
    success : function(data) { … },
    error : function(xhr, textStatus, errorThrown) { … }
});
```

텍스트 파일 로딩

❖ ajax_text.html

```
<body>
   <h1>$.ajax() 함수로 읽은 텍스트</h1>

   <div>
      <input type="button" value="읽기">
   </div>

   <div id="result">

   </div>
</body>
```

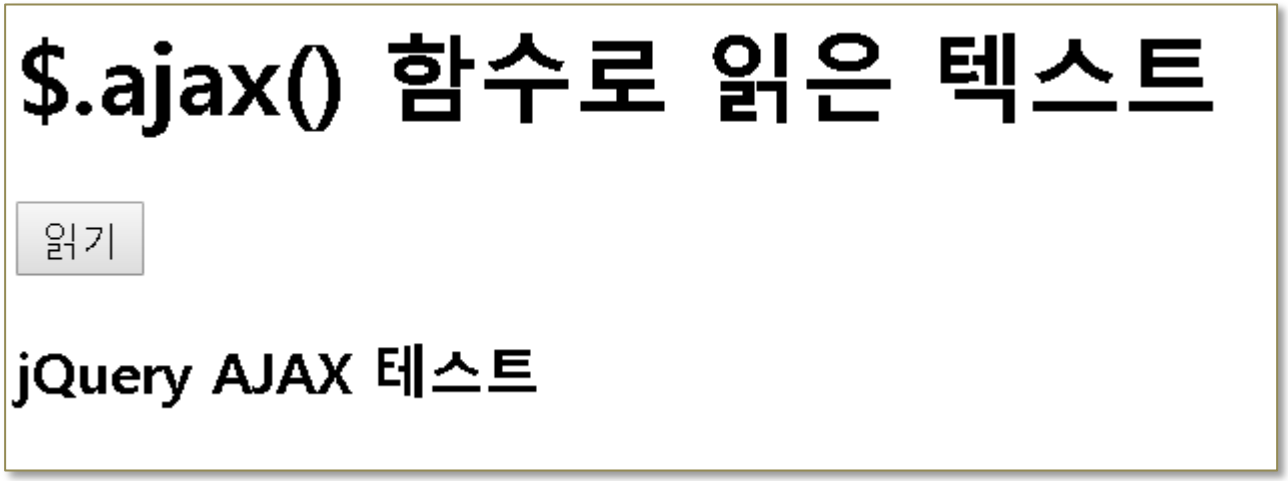
텍스트 파일 로딩

❖ ajax_text.html

```
<script>
$(function(){
   $('input:button').click(function(){
      $.ajax({
         url : 'text/01.txt',
         type : 'get',
         dataType : 'text',
         cache : false,
         success : function(data) {
            $('#result').html(data);
         },
         error : function(xhr, statusText, errorThrown) {
            $('#result').html(`${statusText} - ${xhr.status} /
$errorThrown`);
         }
      })
   });

});
</script>
```

❖ 결과



텍스트 파일 로딩

❖ 응답 파일 준비

- text/01.txt (UTF-8 인코딩)

```
<h2>jQuery AJAX 테스트</h2>
<p> HTML 엘리먼트</p>
```

텍스트 파일 로딩

❖ load( ) 함수

```
$("선택자").load(url [, function(data) {
   // 읽기 성공 후 실행될 함수
});
```

- url에 readme.html #my
  - readme.html 내 #id 태그만 읽기

텍스트 파일 로딩

❖ text/category_data.html (depth 1)

```
<select name="category1" id="category1">
   <option>---- 선택하세요 ----</option>
   <option value="의류" data-target="#category2-1">의류</option>
   <option value="디지털/가전" data-target="#category2-2">디지털/가전
</option>
</select>
```

텍스트 파일 로딩

❖ text/category_data.html (depth 2)

```
<select name="category2" id="category2-1">
   <option>---- 선택하세요 ----</option>
   <option value="여성의류" data-target="#category2-1-1">여성의류
</option>
   <option value="남성의류" data-target="#category2-1-2">남성의류
</option>
</select>

<select name="category2" id="category2-2">
   <option>---- 선택하세요 ----</option>
   <option value="가전제품" data-target="#category2-2-1">가전제품
</option>
   <option value="컴퓨터" data-target="#category2-2-2">컴퓨터
</option>
</select>
```

텍스트 파일 로딩

❖ text/category_data.html (depth 3)

```
<select name="category3" id="category2-1-1">
   <option>---- 선택하세요 ----</option>
   <option value="원피스">원피스</option>
   <option value="투피스">투피스</option>
</select>
<select name="category3" id="category2-1-2">
   <option>---- 선택하세요 ----</option>
   <option value="조끼">조끼</option>
   <option value="셔츠">셔츠</option>
</select>

<select name="category3" id="category2-2-1">
   <option>---- 선택하세요 ----</option>
   <option value="TV">TV</option>
   <option value="냉장고">냉장고</option>
</select>
<select name="category3" id="category2-2-2">
   <option>---- 선택하세요 ----</option>
   <option value="노트북">노트북</option>
   <option value="데스크탑">데스크탑</option>
</select>
```

텍스트 파일 로딩

❖ category.html

```
<style>
   form span { display : none }
</style>

<body>
   <h1 class="title">동적 드롭다운</h1>

   <div>
      <form>
         <span id="category1_container"></span>
         <span id="category2_container"></span>
         <span id="category3_container"></span>
      </form>
   </div>

   <div class="console"></div>
</body>
```

텍스트 파일 로딩

❖ category.html

```
<script>
$(function(){
   var base_url = 'text/category_data.html ';
   $('#category1_container')
      .load(base_url + '#category1', function(){
         $(this).show();
      });
});
</script>
```

텍스트 파일 로딩

❖ category.html

```
<script>
$(function(){
   var base_url = 'text/category_data.html ';
   $('#category1_container')
      .load(base_url + '#category1', function(){
         $(this).show();
      });

   $('form').on('change','#category1_container>select', function(){

   });

   $('form').on('change','#category2_container>select', function(){

   });

   $('form').on('change','#category3_container>select', function(){

   });
});
</script>
```

텍스트 파일 로딩

❖ category.html

```
$('form').on('change','#category1_container>select', function(){

   $('[name=category2]').empty().hide();
   $('[name=category3]').empty().hide();

   var target = $(this).find('option:selected').data('target');
   $('#category2_container').load(base_url+target, function() {
      $(this).show();
   });
});
```

텍스트 파일 로딩

❖ category.html

```
$('form').on('change','#category2_container>select', function(){
   $('[name=category3]').empty().hide();

   var target = $(this).find('option:selected').data('target');
   $('#category3_container').load(base_url+target, function() {
      $(this).show();

   });
});
```

텍스트 파일 로딩

❖ category.html

```
$('form').on('change','#category3_container>select', function(){

   // 최종 선택항목이 있는 경우
   if($(this).find('option:selected').index() > 0) {
      var val1 = $('#category1_container option:selected').val();
      var val2 = $('#category2_container option:selected').val();
      var val3 = $('#category3_container option:selected').val();

      $('#console').text(`${val1} > ${val2} > ${val3}`);
   }
});
```

Ajax
- 웹 프로그래밍 연동-

❖ Context-Type의 설정

Content-Type	설명
text/text	일반 텍스트 파일
text/xml	XML 형식
application/json	JSON 형식
application/javascript	Javascript 소스코드. JSONP일 경우 사용한다.

- `contentType` 속성에 설정

❖ 웹 파라미터 설정

- data 속성에 설정

```
data: "이름1=값1&이름2=값2&…이름n&값n"
```

  - $.param(객체) → 프로퍼티들을 파라미터 문자열로 리턴

- data 속성에 객체 지정하면 자동 변환

```
data: {
  이름1: 값1, 이름2: 값2, … 이름n: 값n
}
```

❖ $.get( ) / $.post( )

```
$.ajax({
    "url" : "접속할 페이지 주소",
    "type" : "get or post",
    "dataType" : "xml",
    "data" : "파라미터 문자열 key=value&key=value",
    "success" : function(data) {
        // 통신이 성공했을 때 실행되는 함수.
    }
});
```

```
$.get("url", {파라미터 JSON}, function(data) {
    // 통신이 성공했을 때 실행되는 함수.
}, ["xml"]);
```

← **생략시 jQuery가 스스로 판단**

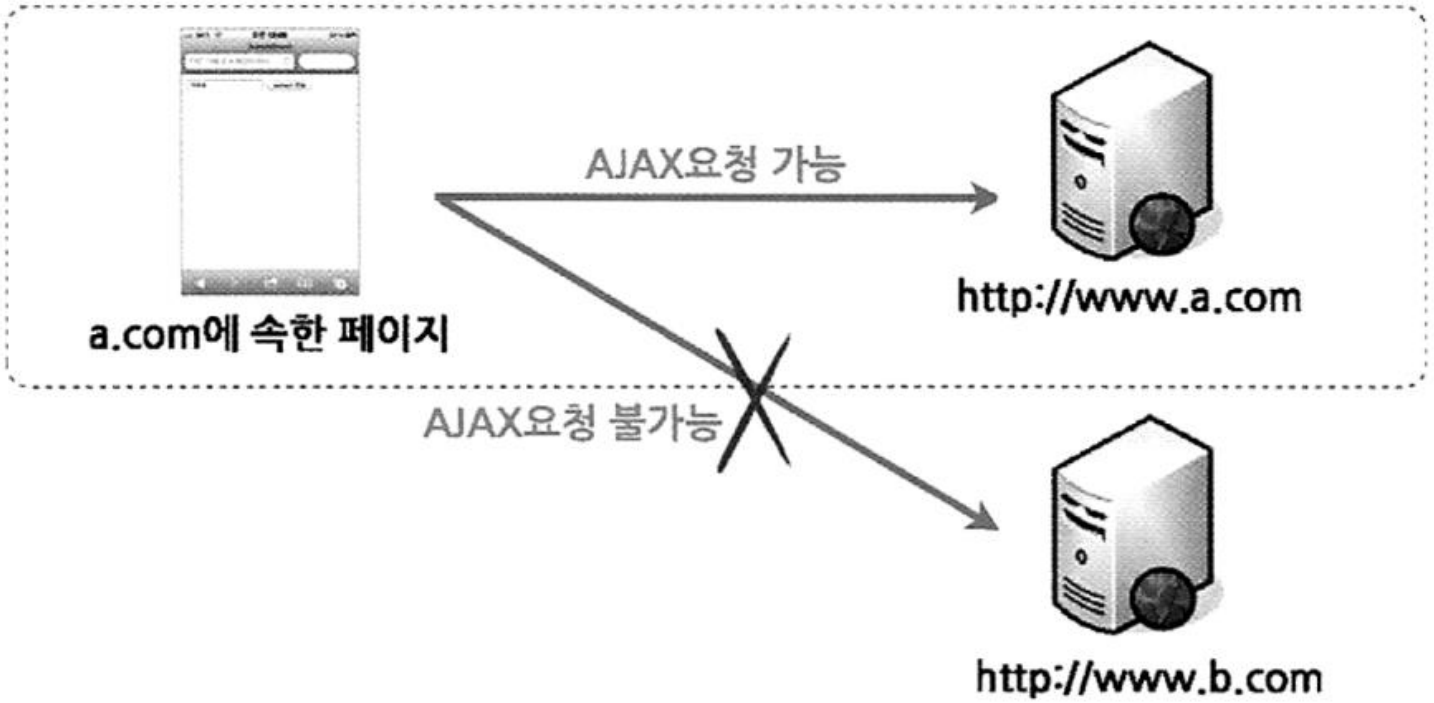
❖ $.getJSON( )

```
$.ajax({
    "url": "접속할 페이지 주소”,
    "dataType": "jsonp",
    "data": "파라미터 문자열 key=value&key=value",
    "success": function(data) {
        // 통신이 성공했을 때 실행되는 함수.
    }
});
```

```
$.getJSON(“접속할 페이지 주소”, {파라미터 JSON}, function(data) {
    // 통신이 성공했을 때 실행되는 함수.
});
```

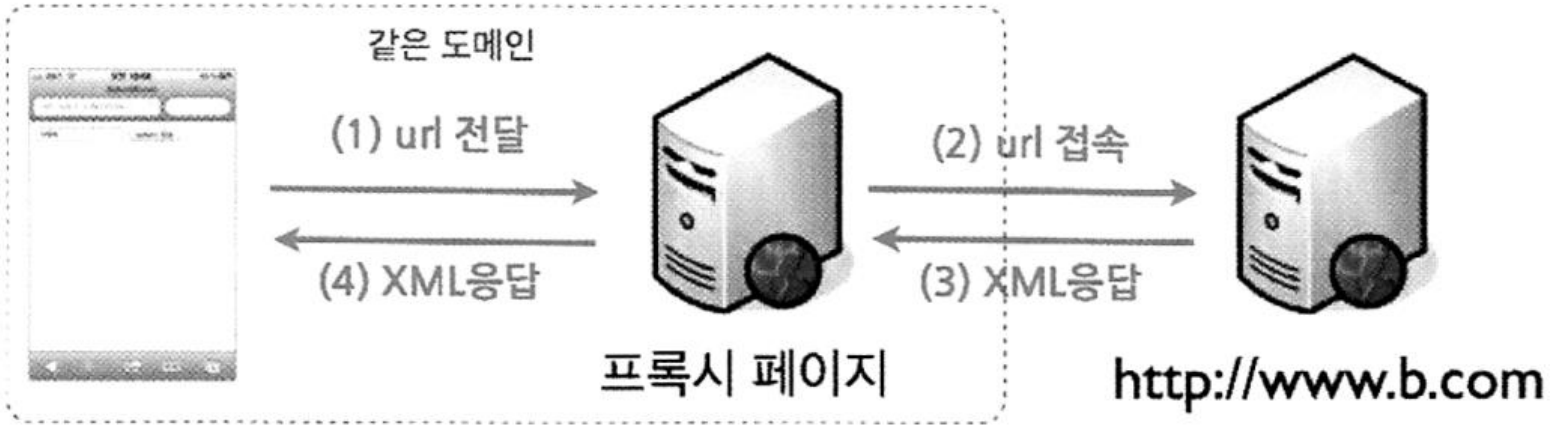
❖ 크로스 도메인 제약

- ㅇ 동일 출처 정책(브라우저의 기능)
  - 웹 브라우저에서는 보안상의 이유로 동일 서버에 대한 Ajax 호출만 지원



❖ 크로스 도메인 제약

- ㅇ 프록시의 사용



❖ JSONP(JSON with Padding)

- url에 callback=?이 지정된 경우 자동으로 JSONP 처리

```
http://itpaper.co.kr/myjsonp.jsp?callback=?
```

```
http://abc.com/script.php? callback=랜덤한값 &name=user1&age=20&...
```

이 함수를 가리키는 이름이다.

```
$.getJSON("url", {파라미터 JSON}, function(data) {
    // 통신이 성공했을 때 실행되는 함수.
});
```