

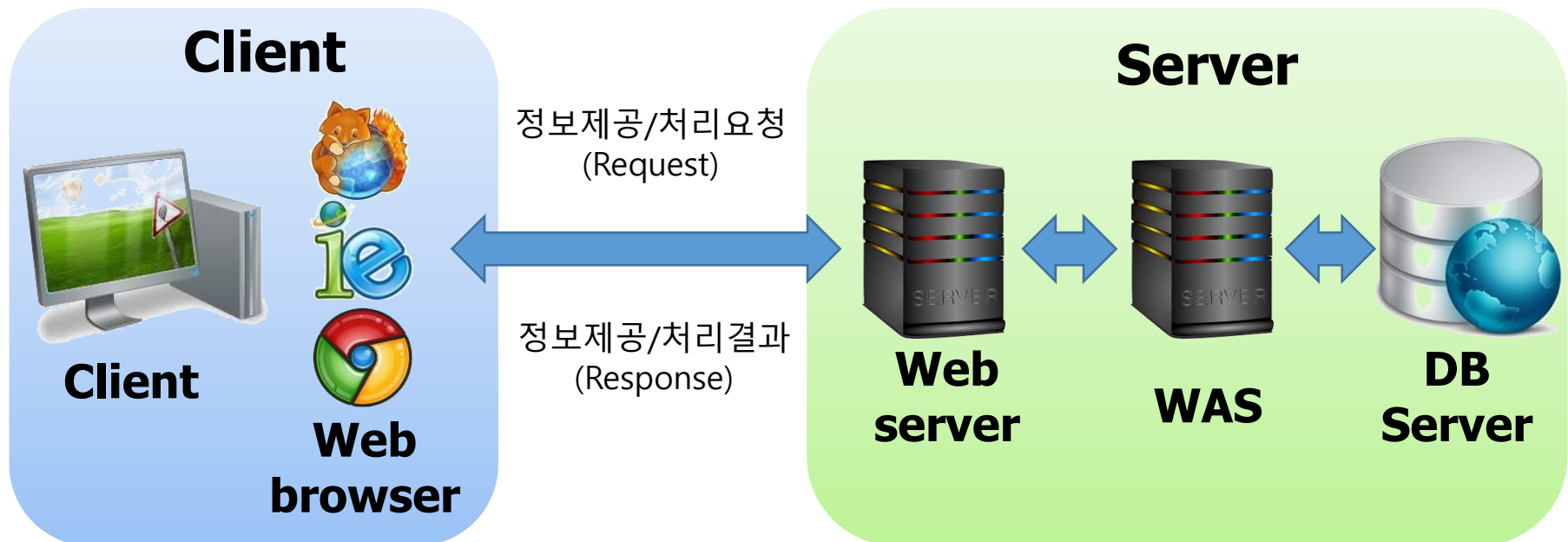
JavaScript

개요

▶ 클라이언트-서버 구조

프로토콜

- HTTP: 하이퍼텍스트 전송 규약(웹 통신)
- FTP : 대량의 파일을 전송/수신하기 위한 파일 전송 규약
- TELNET : 원격지에서 서버 컴퓨터를 원격 제어하기 위한 규약



▶ 클라이언트 주요 언어

- HTML : 하이퍼텍스트를 구현하기 위한 뼈대가 되는 핵심적인 기술
마크업 언어
- CSS : HTML은 뼈대라면 CSS는 꾸미기 위한 옷의 기능
- 자바스크립트(JavaScript) : 기능, 로컬의 브라우저에서 실행되는
인터프리터 방식의 프로그래밍 언어
- jQuery : 존 레식(John Resig) 창안, JavaScript 기반 라이브러리 중 하나
자바스크립트의 코드가 길어지면 사용하기 복잡한 단점 개선

▶ 서버 주요 언어

- JSP : 운영체제의 구매를 받지 않으며
실행 톰캣 컨테이너 위에서 자바 기반의 언어 사용
- ASP : 윈도우 기반의 IIS 서버에서만 작동하고 MS-SQL DBMS와 연동 됨
- PHP : 리눅스 기반에 아파치 서버에서 동작, 기존 제로보드나 그누보드와
같은 사이트 빌더에서 기본적으로 사용되는 언어
- node.js : 노드제이에스 또는 노드라고 하는 자바스크립트 라이브러리로
소켓을 이용하여 쉽게 실시간 서버를 구축 가능하도록 함

▶ JavaScript

자바스크립트(JavaScript)는 웹 브라우저에서 많이 사용하는
인터프리트 방식의 객체지향 프로그래밍 언어로
ECMA(European Computer Manufacturers Association) 스크립트 표준을
따르는 대표적인 웹 기술

작성 및 실행

▶ 브라우저 개발자 도구

브라우저 별로 개발자 도구(Developer Tools)가 있는데
크롬이나 IE브라우저의 경우 F12키를 눌러 실행함
브라우저 창에서 원하는 코드를 확인하고 싶으면 원하는 위치에서 우 클릭
후 해당 메뉴를 선택함(크롬 : 검사 / IE : 요소 검사)
자바스크립트 소스코드 중 console.log()는 브라우저에서 출력하는 것이
아니라 개발자 도구의 console패널에서 출력하고 스크립트 구문을 디버깅
할 때 자주 사용함

▶ 자바스크립트 선언

HTML에서 제공하는 `<script>` `</script>` 태그를 사용하며
자바스크립트 작성 영역을 설정하고 그 사이에 자바스크립트 코드 작성
type속성은 브라우저 호환성을 위해 사용되나 default값으로 생략 가능

```
<script type="text/javascript">
```

자바스크립트 내용

```
</script>
```

* language속성과 charset속성이 있었지만 language속성은 폐기되고
charset속성은 meta태그가 적용되기 때문에 사용할 필요 없음

▶ 자바스크립트 위치

`<script>` `</script>`는 `<head>`, `<body>` 안 어느 영역에나 작성 가능
특히 `<html>` 태그 영역 밖에서 작성도 가능하지만 웹 표준과 웹 접근성을
고려해 `<head>`나 `<body>`안에 작성함

`<script>`자바스크립트 내용`</script>`

`<html>`

`<head>` `<script>`자바스크립트 내용`</script>` `</head>`

`<body>` `<script>`자바스크립트 내용`</script>` `</body>`

`</html>`

`<script>`자바스크립트 내용`</script>`

▶ 자바스크립트 작성 방식

- inline 방식 : 자바스크립트 양이 한 두 줄 정도로 소량일 때 사용
태그에 이벤트 핸들러 속성을 이용하여 직접 실행 코드 작성
- internal 방식 : 가장 일반적인 방식으로 html파일 내
<head>나 <body>안에 자바스크립트 소스 작성
- external 방식 : 자바스크립트 양이 많을 경우 자바스크립트 코드 부분을
외부 파일로 저장하여 작성
<script src="경로">태그를 이용해 내용 삽입 후 사용

▶ 자바스크립트 작성 방식

✓ inline 방식

html 태그의 on이벤트 속성을 이용하여 내장 메소드를 호출하거나 개발자가 선언한 사용자 정의 함수를 호출할 때 사용

```
<태그 명 on이벤트 = "함수명();">
```

✓ internal 방식

자바스크립트 코드 작성, 함수 단위로 소스코드를 작성하고 html태그에서 이벤트 핸들러를 통해 함수를 실행시키는 방식

```
<script> 실행할 소스코드 작성 </script>
```

▶ 자바스크립트 작성 방식

✓ external 방식

외부에 별도의 자바스크립트 소스파일(*.js)을 작성하고
html에서 <script src="경로.js">태그를 이용하여 해당 파일을 불러와
사용하는 방식으로 여러 개 html파일에서 공통적으로 사용하는
기능일 경우 많이 씀

```
<script src="경로"> </script>
```

▶ 자바스크립트 실행 방식

인터프리터 방식은 웹 브라우저에 내장되어 있는 자바스크립트 파서가 소스코드를 한 줄씩 읽고 해석함

전체를 해석해 놓은 컴파일 언어와는 차이가 있음

자바스크립트 실행은 작성된 html문서를 브라우저에서 읽으면 바로 실행을 할 수 있음

* 자바스크립트를 지원하지 않는 브라우저를 대비해 출력문구를 <noscript>에 작성

<noscript>

 지원하지 않을 경우 출력문구

</noscript>

데이터 입·출력

▶ 데이터 출력

코드	설명
document.write(내용);	브라우저 화면 상의 페이지에 값 출력
window.alert(내용);	내용을 메시지 창에 출력 * window객체는 모두 적용되는 것으로 생략 가능
innerHTML = 내용;	태그 엘리먼트의 내용을 변경하여 출력
console.log(내용);	개발자 도구 화면의 콘솔에 출력

▶ 데이터 입력

자바스크립트 내장객체인 window 객체가 제공하는
confirm(), prompt() 메소드를 사용하여 입력 받을 수도 있고
HTML태그에 접근해 대상의 값을 읽거나
HTML form태그의 input 입력 양식을 통해 값을 입력 받을 수도 있음

▶ 데이터 입력

✓ **window.confirm()**

어떤 질문에 대해 "예/아니오"의 결과를 얻을 때 사용
대화 창에 메시지와 확인, 취소 버튼 표시 * 리턴 값 : 확인(true), 취소(false)

```
var 변수 = [window.]confirm("질문내용");
```

✓ **window.prompt()**

텍스트 필드와 확인/취소버튼이 있는 대화 창 출력 * 리턴 값 : 입력한 내용

```
var 변수 = [window.]prompt("메시지");
```

HTML태그 접근

▶ 메소드

메소드	설명
<code>getElementById("아이디명")</code>	태그의 id 속성 값을 이용해 태그 엘리먼트 객체의 정보를 가져옴
<code>getElementsByName("이름")</code>	태그의 name속성 값을 이용해 태그 엘리먼트의 객체 정보를 배열에 담아 가져옴 같은 이름의 태그가 여러 개 존재할 수 있기 때문에 기본적으로 배열로 리턴
<code>getElementsByTagName("태그명")</code>	태그 명을 이용하여 해당 태그들의 객체 정보를 배열에 담아 가져옴

▶ document.getElementById()

HTML태그의 id속성 값은 페이지에서 유일한 식별자 역할을 하도록 권장

* 리턴 값 : 단일 값(id는 중복 허용 안 함)

```
var 변수 = document.getElementById("아이디명");
```

* 변수는 객체를 의미하는 레퍼런스 변수

▶ document.getElementsByName()

HTML태그의 name속성 값으로 객체 정보를 가져올 때 사용

* 리턴 값 : 배열(name은 중복 가능)

```
var 변수 = document.getEleemtsByName("이름");
```

* 변수는 배열이 됨

▶ document.getElementsByTagName()

HTML태그의 태그 이름을 이용해 태그들을 한꺼번에 가져와 순서대로 반환

* 리턴 값 : 배열(태그 중복 가능)

```
var 변수 = document.getElementsByTagName("태그 명");
```

* 변수는 배열이 됨