웹 앱 개발을 위한 Javascript 기초

8. 자바스크립트 API 활용 및 그림판 프로그램 만들기

한국기술교육대학교
온라인평생교육원

학습내용

1. API 활용
2. 그림판 프로그램

학습목표

1. API 활용에 대해 설명할 수 있다.
2. 자바스크립트 응용 프로그램으로 그림판 프로그램을 만들 수 있다.

1. API 활용

클라이언트 측 고유 기술 요소

* 웹브라우저에서 동작하는 자바스크립트를 클라이언트 측 자바스크립트라 함
* 클라이언트 측 자바스크립트 구성
  - ECMAScript가 규정한 코어 언어와 웹 브라우저의 API(Application Program Interface)
* 웹 브라우저의 주요 API
  - Window인터페이스
    - 자바스크립트로 브라우저 또는 창을 조작하는 기능 제공
  - DOM
    - 자바스크립트로 HTML 문서의 요소를 제어하는 기능 제공
  - XMLHttpRequest
    - 서버와 비동기로 통신하는 기능 제공
* 웹 서버에서 동작하는 자바스크립트
* 웹 서버를 구현하는데 Perl, PHP, Python, Ruby 등의 프로그래밍 언어를 사용
* 웹 브라우저의 주요 API
  - Node.js
    - 구글이 개발한 자바스크립트 실행 환경
  - Rhino
    - 오픈 소스로 개발되어 현재는 모질라가 관리하고 있는 자바스크립트 실행 환경
  - Aptana Jaxer
    - 압타나 사가 개발하고 현재는 오픈소스로 개발되고 있는 자바스크립트 실행 환경

1. API 활용

주요 API

API	설명
Drag and Drop	HTML 요소 혹은 파일을 끌어서(드래그) 다른 HTML 요소에 놓을 때(드롭할 때) 데이터를 전달하는 기능을 제공
Blob	이진 데이터를 다루는 기능을 제공
File	프로그램 여러 개를 멀티스레드로 병렬 처리하는 기능을 제공
Web Workers	대용량이며 저장 기간에 제한이 없는 데이터를 로컬에 저장하는 기능을 제공
Indexed Database	로컬에 키값 타입의 관계형 데이터 베이스 기능을 제공
WebSockets	서버와의 양방향 통신 기능을 제공
Geolocation	GPS 등의 위치 정보를 다루는 기능을 제공
Canvas	2차원 3차원 그래픽스 기능을 제공

드래그 앤 드롭 API

* HTML 요소나 로컬 파일을 마우스로 끌어서 옮길 수 있으며 다른 요소에 드롭할 수 있음
* 이때 드래그한 요소 또는 파일의 데이터는 드롭 타킷 요소에 전달
* HTML 요소를 드래그 할 수 있게 만들기
  - 〈div draggable="true"〉 드래그 할 수 있습니다.〈/div〉
  - 이 속성을 지정하지 않거나 auto로 지정하면 해당 HTML요소의 기본값을 사용
  - href 속성을 지정한 a 요소와 src 속성을 지정한 img 요소는 기본적으로 드래그 할 수 있도록 만들어져 있음

1. API 활용

드래그 앤 드롭 이벤트

API	설명
dragstrart	드래그를 시작할 때 발생
drag	드래그를 하는 동안 발생
dragend	드래그가 끝났을 때 발생
dragenter	마우스 포인터가 드롭 요소의 경계선 안쪽으로 들어갈 때 발생
dragover	마우스 포인터가 드롭 요소의 경계선 안쪽에 있을 때 발생
dragleave	마우스 포인터가 드롭 요소의 경계선 바깥으로 나왔을 때 발생
drop	요소에 드롭할 때 발생

드래그 앤 드롭 API

* 모든 드래그 앤 드롭 이벤트는 dataTransfer 프로퍼티를 가짐
* dataTransfer 프로퍼티 값은 Datatransfer 객체이며, 이 객체로 드래그 타깃 요소가 드롭 타깃 요소에 데이터를 전달 할 수 있음
* DataTransfer 객체의 프로퍼티와 메소드

1. API 활용

드래그 앤 드롭 이벤트

프로퍼티 이름 / 메소드 이름	설명
type	setData 메소드로 설정한 데이터 타입 목록
files	드래그한 파일 객체 목록
effectAllowed	드래그 타킷 요소가 허용하는 작업의 유형(“none”, “copy”, “copyLink”, “copyMove” 등)
dropEffect	드롭 타깃 요소에 표시하는 효과(“none”, “copy”, “move”, “link”)
setData(format,data)	드래그 타깃 요소의 데이터 타입을 특정 데이터 타입으로 설정
getData(format)	드롭 타깃 요소에서 데이터를 특정 타입(format)으로 가져옴
clearData(format)	Format타입으로 저장된 데이터를 삭제, format으로 지정하지 않으면 모든 데이터 삭제
setDragImage (element x, y)	드래그 이미지(드래그중 표시되는 이미지)를 설정. Element는 img요소, x는 이미지 수평 오프셋, y는 이미지 수직 오프셋
addElement(element)	드래그 타깃의 HTML요소를 드롭 타깃에 추가. 이 메소드를 호출 하지 않아도 드롭하는 요소 자체가 드래그 타깃의 HTML요소가 됨

1. API 활용

데이터 전달하기(드래그 타깃 요소에서 드롭 타깃 요소에 데이터 전달)

* 드래그 타깃 요소의 dragstart이벤트 처리기 안에서 dataTransfer 프로퍼티의 setData메소드에 데이터 타입을 지정한 데이터를 추가
  e.dataTransfer.setData("text/plain", value);
* 드롭 타깃 요소의 dragover 이벤트 처리기 안에서 브라우저의 기본 동작을 취소
  드래그 타깃 요소가 드롭 타깃 요소 위에 올라가면 브라우저의 기본 동작인 drop 이벤트가 취소되기 때문 e.preventDefault();
* 드롭 타깃 요소의 drop 이벤트 처리기 안에서 dataTransfer 프로퍼티의 getData 메소드를 사용해서 데이터를 지정한 데이터 타입으로 가져옴
* var value = e.dataTransfer.getData("text/plain");
  데이터는 데이터 타입(format)별로 하나만 전달할 수 있음
* setData메소드로 같은 데이터 타입의 데이터를 두 번 설정하면 이전에 설정한 데이터를 덮어씀

1. API 활용

드래그 앤 드롭 예제

이것을 드래그하세요.

이것을 드롭하세요.

이것을 드래그하세요.

이것을 드래그하세요.

이것을 드롭하세요.

드롭하는 영역을 사용자에게 알림

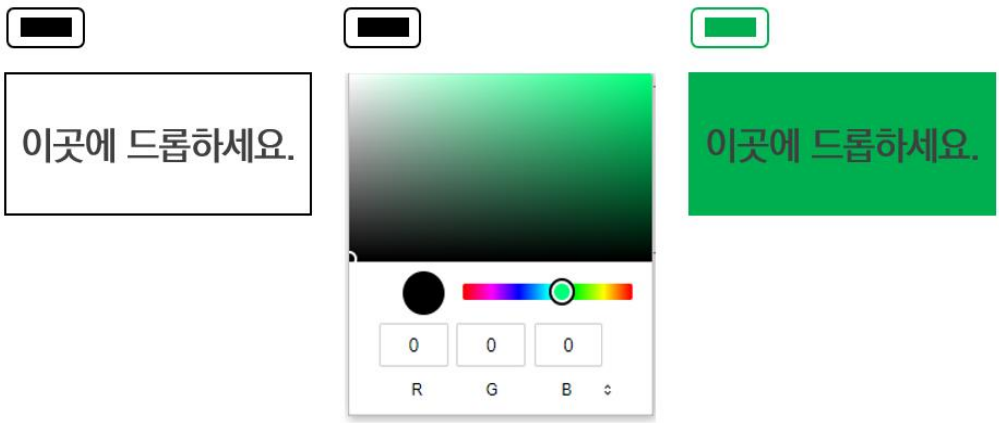
```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>드롭 하는 영역을 사용자에게 알리기</title>
    <script>
        window.onload = function() {
            var dragbox = document.getElementById("dragbox");
            var dropbox = document.getElementById("dropbox");

            dropbox.addEventListener("dragenter", function(e) {
                e.target.style.borderColor = "red";
            }, false);
            dropbox.addEventListener("dragleave", function(e) {
                e.target.style.borderColor = "gray";
            }, false);
            dropbox.addEventListener("drop", function(e) {
                e.target.style.borderColor = "gray";
            }, false);
        };
    </script>
    <style>
        #dragbox { width: 150px; border: 10px solid blue; }
        #dropbox { width: 150px; padding: 50px; border: 10px solid blue; }
    </style>
<body>
    <div id="dragbox" draggable="true">이것을 드래그하세요</div>
    <div id="dropbox">이곳에 드롭하세요</div>
</body>
</head>
```

1. API 활용

드래그 앤 드롭 예제

* 드래그 앤 드롭으로 배경색 설정하기



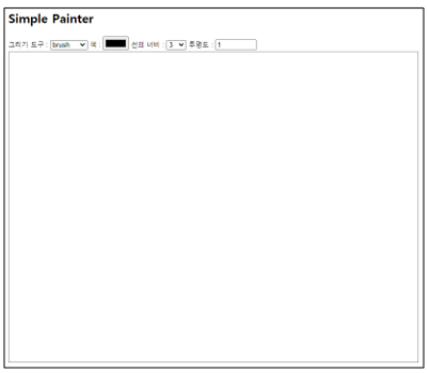
* 드래그 앤 드롭으로 배경색 설정하기

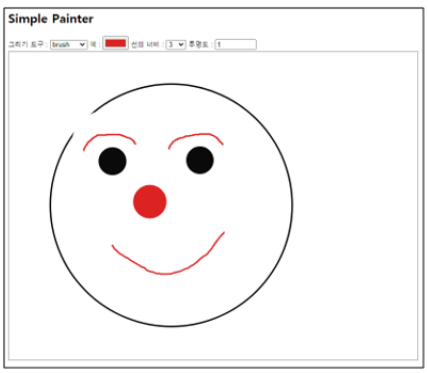
```html
<!DOCTYPE html>
<html lang="ko">
<head>
<meta charset="UTF-8">
<title>드래그 앤 드롭 예제</title>
<script>
    window.onload = function() {
        var color = document.getElementById("color");
        var dropbox = document.getElementById("dropbox");
        // 드래그를 시작할 때, 색상의 값을 dataTransfer 객체의 데이터로 설정한다
        color.ondragstart = function(e) {
            e.dataTransfer.setData("text/plain", e.target.value);
        };
        // 드래그 타깃 요소 위에 마우스 포인터가 올라가면, 브라우저의 기본 동작을 취소한다 (필수)
        dropbox.ondragover = function(e) {
            e.preventDefault();
        };
        // 요소를 드롭하면, datTransfer의 데이터로 보더 박스의 배경색을 설정한다
        dropbox.ondrop = function(e) {
            e.preventDefault (); // 브라우저의 기본 동작을 취소한다 (선택사항)
            e.target.style.backgroundColor = e.dataTransfer.getData("text/plain");
        };
    };
</script>
<style>
    #color { margin-bottom: 10px; }
    #dropbox { width: 150px; padding: 50px; border: 1px solid gray; }
</style>
</head>
<body>
    <input type="color" id="color" draggable="true">
    <div id="dropbox">이곳에 드롭하세요</div>
</body>
</html>
```

2. 그림판 프로그램

그림판 프로그램 실행 화면

* painter.html
* elt.js
* painter.js





그림판 프로그램 분석

* painter.html

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>Simple Painter</title>
    <script src="./elt.js"></script>
    <script src="./painter.js"></script>
    <script>
        window.onload = function() {
            createPainter(document.body, 800, 600);
        };
    </script>
</head>
<body>
</body>
</html>
```

2. 그림판 프로그램

그림판 프로그램 분석

elt.js

```
/*-------------------------------------------------------------------*
 * 함수 이름 : elt
 * 주어진 이름(name)과 속성(attributes), 자식 노드를 포함하는 엘리먼트를 만들어서 반환하는 함수
 *-------------------------------------------------------------------*/
function elt(name, attributes) {
    var node = document.createElement(name);
    if( attributes ) {
        for(var attr in attributes) {
            if(attributes.hasOwnProperty(attr)) {
                node.setAttribute(attr,attributes[attr]);
            }
        }
    }
    for(var i=2; i<arguments.length; i++) {
        var child = arguments[i];
        if( typeof child == "string" ) {
            child = document.createTextNode(child);
        }
        node.appendChild(child);
    }
    return node;
}
```

painter.js

```
/*-------------------------------------------------------------------------*
 * 화면을 구성하는 요소를 생성하고,  요소에 이벤트 리스너를 등록한다
 *-------------------------------------------------------------------------*/
function createPainter(parent, width, height) {
    // 타이틀
    var title = elt("h2", null, "Simple Painter");
    // canvas 요소와 랜더링 컨텍스트를 가져온다
    var [canvas,ctx] = createCanvas(width, height);
    // 도구 막대 : controls 객체의 프로퍼티를 순회하면서 등록한다
    var toolbar = elt("div", null);
    for(var name in controls) {
        toolbar.appendChild(controls[name](ctx));
    }
    toolbar.style.fontSize = "small";
    toolbar.style.marginBottom = "3px";
    // toolbar 요소와 canvas 요소를, 지정한 요소(parent)의 자식 요소로 삽입한다
    parent.appendChild(elt("div", null, title, toolbar, canvas));
}
function createCanvas(canvasWidth,canvasHeight) {
    var canvas = elt("canvas", { width: canvasWidth, height: canvasHeight });
    var ctx = canvas.getContext("2d");
    canvas.style.border = "1px solid gray";
    canvas.style.cursor = "pointer";
    // 그리기 도구를 mousedown 이벤트의 이벤트 리스너로 등록한다
    canvas.addEventListener("mousedown", function(e) {
        // Firefox 대책 : 색상을 선택하면, change 이벤트를 강제로 발생시킨다
        var event = document.createEvent("HTMLEvents");
        event.initEvent("change", false, true);
        colorInput.dispatchEvent(event);
        // 선택한 그리기 도구를 초기화
        paintTools[paintTool](e,ctx);
    }, false);
    return [canvas,ctx];
}
```

2. 그림판 프로그램

그림판 프로그램 분석

painter.js

```
35  /*-------------------------------------------------------------------------*
36   * 유틸리티
37   *-------------------------------------------------------------------------*/
38  // * element의 왼쪽 위 모서리에서 마우스의 상대 위치를 가져온다
39  function relativePosition(event, element) {
40      var rect = element.getBoundingClientRect();
41      return { x: Math.floor(event.clientX - rect.left),
42               y: Math.floor(event.clientY - rect.top ) };
43  }
```

그림판 프로그램 분석

painter.js

```
44  /*-------------------------------------------------------------------------*
45   * 그리기 도구
46   * paintTools 메서드는 사용할 수 있는 그리기 도구의 모음입니다.
47   * 각 그리기 도구는, 그림을 그리기 위한 각종 설정과 이벤트 리스너의 등록을 담당합니다.
48   * 각 메서드는 controls.painter를 통해 자동으로 도구 선택 메뉴에 추가됩니다.
49   * 메뉴에서 선택한 도구는 변수 paintTool에 저장되며, 그림을 그릴 때 사용합니다.
50   * 그리기 도구를 추가하려면, paintTools 메서드에 새로운 그리기 도구를 추가하십시오.
51   *-------------------------------------------------------------------------*/
52  var paintTool; // 선택된 그리기 도구 (controls.painter로 선택)
53  var paintTools = Object.create(null); // 그리기 도구 객체
54  // * brush : 브러시 도구
55  paintTools.brush = function(e, ctx) {
56      ctx.lineCap   = "round";
57      ctx.lineJoin  = "round";
58      // Canvas 화면을 img에 저장한다
59      var img = ctx.getImageData(0, 0, ctx.canvas.width, ctx.canvas.height);
60      // canvas 요소에 대한, 마우스 포인터의 상대 위치를 구한다
61      var p = relativePosition(e, ctx.canvas);
62      // 경로를 정의한다
63      ctx.beginPath();
64      ctx.moveTo(p.x,p.y);
65      // 드래그 이벤트 리스너를 등록한다
66      setDragListeners(ctx, img, function(q) {
67          ctx.lineTo(q.x, q.y); // 경로를 추가한다
68          ctx.stroke(); // 경로를 그린다
69      });
70  };
```

2. 그림판 프로그램

그림판 프로그램 분석

painter.js

```
71  // * line : 선 그리기 도구
72  paintTools.line = function(e, ctx) {
73      // ❶ 그림을 그리기 위한 초기화 처리 : 선의 끝부분을 둥글게 만든다
74      ctx.lineCap = "round";
75      // ❷ 그림을 그리기 전에, Canvas에 담긴 그림을 img에 저장한다
76      var img = ctx.getImageData(0, 0, ctx.canvas.width, ctx.canvas.height);
77      // ❸ canvas 요소에 대한, 마우스 포인터의 상대 위치를 구한다
78      var p = relativePosition(e, ctx.canvas);
79      // ❹ 마우스를 드래그할 때의 이벤트 리스너를 등록한다
80      setDragListeners(ctx, img, function(q) {
81          ctx.beginPath();
82          ctx.moveTo(p.x,p.y); ctx.lineTo(q.x,q.y);
83          ctx.stroke();
84      });
85  };
```

그림판 프로그램 분석

painter.js

```
86  // * circle : 원 그리기 도구
87  paintTools.circle = function(e, ctx) {
88      var img = ctx.getImageData(0, 0, ctx.canvas.width, ctx.canvas.height);
89      var p = relativePosition(e,ctx.canvas);
90      setDragListeners(ctx, img, function(q) {
91          var dx = q.x - p.x;
92          var dy = q.y - p.y;
93          var r = Math.sqrt(dx*dx+dy*dy);
94          ctx.beginPath();
95          ctx.arc(p.x, p.y, r, 0, 2*Math.PI, false);
96          ctx.stroke();
97      });
98  };
```

2. 그림판 프로그램

그림판 프로그램 분석

painter.js

```
 99  // * circleFill : 채워진 원 그리기 도구
100  paintTools.circleFill = function(e, ctx) {
101      var img = ctx.getImageData(0, 0, ctx.canvas.width, ctx.canvas.height);
102      var p = relativePosition(e,ctx.canvas);
103      setDragListeners(ctx, img, function(q) {
104          var dx = q.x - p.x;
105          var dy = q.y - p.y;
106          var r = Math.sqrt(dx*dx+dy*dy);
107          ctx.beginPath();
108          ctx.arc(p.x, p.y, r, 0, 2*Math.PI,false);
109          ctx.fill();
110      });
111  };
```

그림판 프로그램 분석

painter.js

```
112  /*-------------------------------------------------------------------------*
113   * 그리기 도구의 유틸리티
114   *-------------------------------------------------------------------------*/
115  // * 마우스를 드래그할 때의 이벤트 리스너를 등록한다
116  function setDragListeners(ctx,img,draw) {
117      // mousemove 이벤트 리스너를 등록한다
118      var mousemoveEventListener = function(e) {
119          // 저장한 이미지를 읽어들인다
120          ctx.putImageData(img, 0, 0);
121          // 지정한 그리기 함수 draw로 마우스 위치까지 그린다
122          draw(relativePosition(e, ctx.canvas));
123      };
124      document.addEventListener("mousemove", mousemoveEventListener, false);
125      // mouseup 이벤트 리스너를 등록한다
126      document.addEventListener("mouseup", function(e) {
127          // 저장된 이미지로 되돌린다
128          ctx.putImageData(img, 0, 0);
129          // 지정한 그리기 함수 draw로 마우스 위치까지 그린다
130          draw(relativePosition(e, ctx.canvas));
131          // mousemove, mouseup 이벤트 리스너를 제거한다
132          document.removeEventListener("mousemove", mousemoveEventListener, false);
133          document.removeEventListener("mouseup", arguments.callee, false);
134      },false);
135  }
```

2. 그림판 프로그램

그림판 프로그램 분석

painter.js

```
/*-------------------------------------------------------------------------------*
 * 컨트롤러
 * 각종 설정을 변경하는 제어 도구 목록을 정의합니다.
 * 각 컨트롤러는 controls 객체의 메서드로 등록되어 있습니다.
 * 각 메서드는 필요한 HTML 요소를 생성해서 반환하며, 이벤트 리스너를 등록합니다.
 * 각 메서드는, createPainter를 통해 자동으로 도구 막대에 추가됩니다.
 * 새로운 컨트롤을 추가하려면, controls 객체에 새로운 메서드를 추가하십시오.
 *-------------------------------------------------------------------------------*/
var controls = Object.create(null); // 컨트롤러 객체
var colorInput; // Firefox의 change 이벤트 대책. input[type="color"] 객체를 저장한다
// * 그리기 도구 선택
controls.painter = function(ctx) {
    var DEFAULT_TOOL = 0;
    var select = elt("select", null);
    var label = elt("label", null, "그리기 도구 : ", select);
    for(var name in paintTools) {
        select.appendChild(elt("option", {value: name}, name));
    }
    select.selectedIndex = DEFAULT_TOOL;
    paintTool = select.children[DEFAULT_TOOL].value;
    select.addEventListener("change", function(e) {
        paintTool = this.children[this.selectedIndex].value;
    },false);
    return label;
};
```

그림판 프로그램 분석

painter.js

```
// * 색상 선택 (선과 채우기를 모두 설정함 → 필요하다면, 두 개의 기능을 분리해서 별도의 컨트롤로 만드십시오)
controls.color = function(ctx) {
    var input = colorInput = elt("input", {type: "color"});
    var label = elt("label", null, " 색 : ", input);
    input.addEventListener("change", function (e) {// 참고 : Firefox에서는 change 이벤트가 발생하지 않습니다.
        ctx.strokeStyle = this.value;
        ctx.fillStyle = this.value;
    },false);
    return label;
};
```

2. 그림판 프로그램

그림판 프로그램 분석

painter.js

```
171  // * 선의 너비 선택
172  controls.brushsize = function(ctx) {
173      var size = [1,2,3,4,5,6,8,10,12,14,16,20,24,28];
174      var select = elt("select", null);
175      for(var i=0; i<size.length; i++) {
176          select.appendChild(elt("option",{value:size[i].toString()},size[i].toString()));
177      }
178      select.selectedIndex = 2;
179      ctx.lineWidth = size[select.selectedIndex];
180      var label = elt("label",null," 선의 너비 : ",select);
181      select.addEventListener("change", function(e) {
182          ctx.lineWidth = this.value;
183      },false);
184      return label;
185  };
186  // * 투명도 선택
187  controls.alpha = function(ctx) {
188      var input = elt("input", {type:"number",min:"0", max:"1",step:"0.05",value:"1"});
189      var label = elt("label", null, " 투명도 : ", input);
190      input.addEventListener("change", function(e) {
191          ctx.globalAlpha = this.value;
192      },false);
193      return label;
194  };
```

정리하기

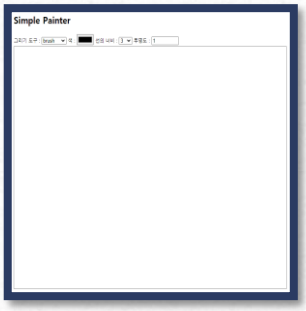
1. API 활용

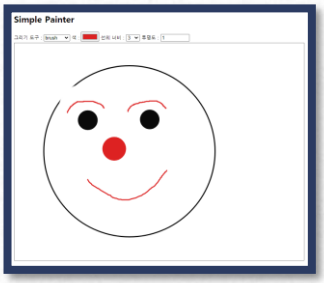
- 클라이언트와 서버측 고유 기술 : 웹브라우저에서 동작하는 자바스크립트를 클라이언트 측 자바스크립트라 함
- 웹 브라우저의 주요 API : Window인터페이스, DOM, XMLHttpRequest
- 주요 API : Drag and Drop, Blob, File, Web Workers, Web Storage, Indexed Database, WebSockets, Geolocation, Canvas
- Drag and Drop API : HTML 요소나 로컬 파일을 마우스로 끌어서 옮길 수 있으며 다른 요소에 드롭할 수 있음, 이때 드래그한 요소 또는 파일의 데이터는 드롭 타킷 요소에 전달
- 데이터 전달하기 : 모든 드래그 앤 드롭 이벤트는 dataTransfer 프로퍼티를 가짐, dataTransfer 프로퍼티 값은 Datatransfer 객체이며, 이 객체로 드래그 타깃 요소가 드롭 타깃 요소에 데이터를 전달 할 수 있음

정리하기

2. 그림판 프로그램

- 그림판 프로그램 실행 화면
- painter.html
- elt.js
- painter.js





- 프로그램 분석