JavaScript Game “Find Equal”

Author : Rhio.kim
Date : 2008.10.03
Blog : blog.ecmas4.com
Mail : Rhio.kim@gmail.com
Version : 0.9.9 / Update : 2008.10.03

Find Equal 게임이란?

간단히 같은 이미지 찾기를 JavaScript 로 개발한 게임입니다.

모티브

외국 사람들도 JavaScript, SVG, Canvas 를 이용해서 게임을 만드는데 국내 개발자도 알고리즘과 아이디어로 게임을 만들 수 있다는 것을 보여주고 싶어설까?

한가지 본연의 동기부여를 추가한다면 Ajax/Rich Application 을 개발할 때 HTML 구조, CSS, JavaScript 의 DOM 과의 작동에 대한 잘된 설계와 UX(사용자 경험)가 잘 혼합된다면 어플리케이션의 성능 향상과 실 사용자(end user)에게 분명 유용한 서비스 혹은 어플리케이션이 된다는 것을 보여주고 싶어서입니다.

왜 그런지는 다음 설명들을 통해서 살펴 보겠습니다.

요구

1. 게임용 이미지 리소스는 요소별로 따로 관리하지 않고 게임에 필요한 리소스를 하나의 파일로 관리한다.
2. HTML, CSS, JavaScript 의 완벽한 구분을 꾀한 Unobtrusive JavaScript 를 지향한다.
3. DOM, CSS 제어를 위해 Prototype.js Framework 를 사용한다.

구현 방식

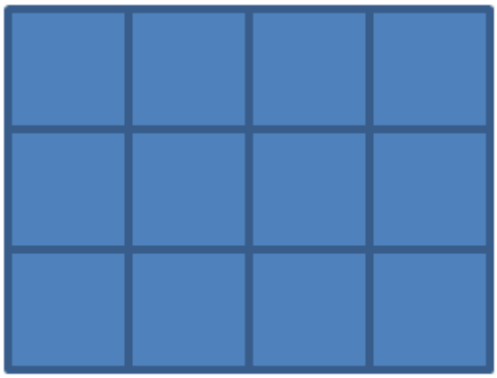
요구사항

A. 게임을 구동하고 사용자가 사용함에 필요한 기본 리소스는 초기 HTML 과 함께 로딩한다.
B. 초급자, 중급자, 고급자별 Game Panel 의 요소의 개수는 변할 수 있다.
C. 다양한 이미지로 화면이 리프래쉬 되거나 유저에 의해 reset 버튼을 이용할 경우에 다양한 이미지를 제공하여 식상하지 않도록 한다.
D. 틀린 그림 찾기 게임의 룰을 만족시킨다.

첫번째 요구사항은 매우 간단합니다. 하지만 왜 이렇게 하는 이유는 알아야겠네요.

****

이렇게 html 페이지에 위와 같이하게 되면 이미지 리소스는 화면상에 노출되지 않은 채 로딩됩니다. 이는 유저에 의해서 게임을 할 때 이미지 리소스를 가져오거나 잘못된 CSS Background-Image 속성을 사용하게 되면 동일 한 이미지를 캐시를 활용하지 못할 수가 있기 때문입니다. 그래서 미리 HTML 에서 로딩을 하게 되면 캐시를 활용할 수 있어 게임 동작 시 성능을 향상할 수 있습니다.



Game Panel

두번째 요구사항을 만족하기 위해서는 위의 Game Panel 을 Table 구조나 Div 구조로 개발되어야 합니다. 하지만 틀린 그림 특성상 Table 구조도 적합하기 때문에 Table 구조를 선택하였고 동적으로 n*n(cel * row) 의 table 을 생성할 수 있도록 하였습니다.

자 지금부터 설명할 부분이 이 게임을 구현하는데 있어서 필수적인 요소가 되겠습니다. 위에서 언급한 개발 시 요구사항 부분에서 언급한 내용처럼 이미 게임의 동작을 위한 Image Resource 는 미리 제작되어 있습니다. 이것은 위의 Game Panel 의 TD 에 들어갈 요소들로 요소별 하나의 이미지가 아닌 이 게임(틀린 그림 찾기)에서 사용될 수 있는 모든 요소 이미지를 하나의 이미지로 만들었습니다. 다음 그림과 같아집니다.

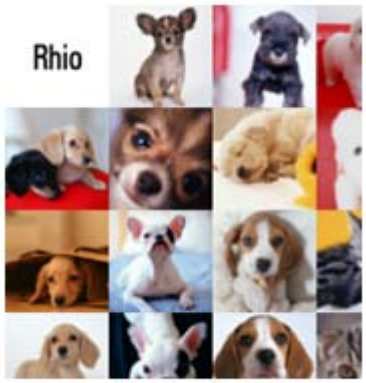


Image Resource

흐름과 틀리지만 예를 들어 위의 Game Panel의 Table을 동적으로 만들 때 TD에 blank 이미지를 설정해 놓고 클릭할 때 해당 blank 이미지에 JavaScript를 이용해 src 속성에 이미지의 위치를 지정하는 방식도 있습니다. 하지만 그렇게 하면 다양한 요구사항을 받아들이기가 어려워지고 원격지의 이미지를 HTTP를 통해 가져오는 동안 문제가 발생하는 경우 표시를 못하게 되거나 다양한 문제점이 게임 진행중에 발생할 수 있습니다.

그래서 위의 이미지 리소스와 같이 하나의 이미지 파일에 다양한 요소를 지정해 놓았습니다. 적어도 게임이 진행중에는 오류로 인한 문제는 최소화 해야 하기 때문입니다. 만약 초기 로딩 시 이미지를 가져오지 못한 경우에는 체크해서 사용자에게 알려주어 게임 진행에 대한 상황을 사용자에게 전달 할 수 있습니다.

그러면 이 하나의 이미지를 어떻게 Game Panel 에 표시할 것인가가 문제입니다. 큰 문제는 아닙니다. 누구든지 아는 부분일 것입니다.

이미 로딩된 Image Resource 를 Game Panel 의 각 요소(TD)에 Style 제어를 통해서 표시할 수 있습니다.

각 TD 에 **style.backgroundPosition = '0px 0px';** 와 같은 방식으로 부여를 하면 됩니다.

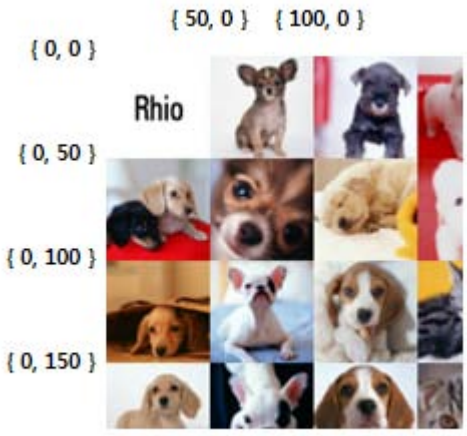


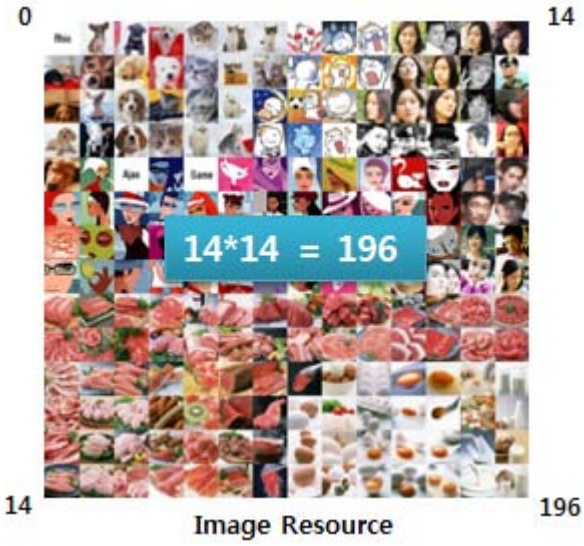
Image Resource

위의 좌표처럼 TD 에 backgroundImage 는 Image Resource 로 설정해주고 backgroundPosition 을 제어하면 하나의 이미지 리소스로 다양한 위치를 각 요소에 설정할 수 있습니다.

마지막 요구사항은 각 요소에 표시될 이미지가 순차적이라면 틀린 그림 찾기의 게임 룰을 만족할 수 없기 때문에 이미지 리소스에 있는 이미지 요소들을 랜덤으로 Game Panel 에 설정하는 부분만 처리하면 유저가 게임을 할 수 있는 기본적인 설정이 갖춰집니다.

랜덤 설정은 알고리즘이니 조금만 고민하면 위의 조건들을 만족하는 렌덤 배열을 생성할 수 있을 것입니다.

랜덤 배열 범위 = **(ImageResource 가로요소 개수 * ImageResource 세로요소 개수) – 1;**



랜덤 배열의 범위 즉 [0, 1 ,..., 195] 를 random 알고리즘을 이용하여 뒤섞어 줍니다.

```
_random: function(arr){
        var i = arr.length;
        if (i == 0)
                return false;

        while (--i) {
                var j = Math.floor(Math.random() * (i + 1));

                var ti = arr[i];
```

```
            var tj = arr[j];
            arr[i] = tj;
            arr[j] = ti;
        }

        return arr;

    }
```

섞여진 배열(return arr)은 Image Resource 의 index 를 가리킵니다. 즉 TD 의 backgroundPositon 을 제어할 때 바로 이 섞여진 배열을 통해서 지정하게 됩니다. 하지만 틀린 그림 찾기는 위의 섞여진 배열로만은 이용할 수 없습니다.

Arr = [28, 13, 99, 2, 84, 50, 38];
Arr = [28, 13, 99, 2, 84, 50, 38, 28, 13, 99, 2, 84, 50, 38];
Arr = [84, 28, 38, 13, 2, 13, 84, 50, 2, 38, 28, 99, 99, 50];

위의 과정을 거치면 정말로 틀린 그림을 찾기의 배열 요소가 갖춰집니다. 이 배열을 이용해 Game Panel 의 각 구성 요소에 Image 를 지정할 때 Image Resource 의 이미지 index 와 매칭하여 Position 을 TD 의 CSS Style 로 부여합니다.

요약

여기까지는 게임을 만들면서 생겼던 주요 이슈에 대한 정리입니다. 대단한 건 아니지만 여러 가지 이유는 숨어있습니다.

요약해보자면 HTML 에 미리 <img> 를 이용하여 Image Resource 를 로딩한 이유 또한 이미지를 하나의 파일에 넣어둔 이유, CSS Style 제어를 통해서 이미지를 렌더링 한 이유 좀더 찾아보면 몇 가지가 더 있겠네요.

구현의 초점은 게임을 즐기는 유저에게 게임을 즐기는 동안에는 최대한의 성능과 오류가 발생할 요인을 최대한 줄이는데 있습니다. 간단한 게임을 가지고 복잡하게 설명한 건 아닌지.

웹 서비스도 마찬가지 입니다. Ajax/Rich Application 개발 시에 간과해서는 안될 상황들이 매우 많습니다. 아시다시피 크로스 브라우징부터 검색에 노출에 대한 부분, HTTP 에 대한 이해, 다국어 지원 여부, Embeding 에 대한 부분... 나열하자면 매우 많겠네요.

그만큼 단순 요구사항에 대한 구현이 아닌 웹 서비스에서는 다양한 환경과 사용자 경험에 대한 바탕으로 설계가 일차적으로 되고(머리 속의 설계도 좋을 것 같습니다.) 개발이 되어진다면 좋지 않을까요?