

효과

jQuery를 사용하면 화면에 멋진 시각 효과를 만들 수 있습니다. jQuery는 간단한 효과를 만들 수 있는 저수준의 애니메이션 메서드와 사용자가 직접 애니메이션을 만들 수 있는 메서드를 제공합니다. 또한 애니메이션을 제어하는 기능도 제공합니다. 17장에서는 간단한 효과와 관련된 메서드, 사용자가 정의하는 애니메이션 메서드, 애니메이션을 제어하는 기능을 살펴보고 jQuery 플러그인도 몇 가지 알아봅니다.

17장의 코드는 굉장히 쉽지만, 실행 결과를 직접 확인하는 게 중요합니다. 책이라는 매체의 특성상 움직이는 결과를 보여드릴 수 없어서 죄송합니다. 반드시 모든 코드를 직접 실행하고 결과를 확인해보세요.

17.1 기본 시각 효과

jQuery는 기본적으로 세 가지의 효과를 표 17-1의 9개의 메서드로 제공합니다.

표 17-1 기본 시각 효과 메서드

메서드 이름	설명
show()	문서 객체를 크게 하며 보여줍니다.
hide()	문서 객체를 작게 하며 사라지게 합니다.
toggle()	show() 메서드와 hide() 메서드를 번갈아 실행합니다.

메서드 이름	설명
slideDown()	문서 객체를 슬라이드 효과와 함께 보여줍니다.
slideUp()	문서 객체를 슬라이드 효과와 함께 사라지게 합니다.
slideToggle()	slideDown() 메서드와 slideUp() 메서드를 번갈아 실행합니다.
fadeIn()	문서 객체를 선명하게 보여줍니다.
fadeOut()	문서 객체를 흐리게 사라지게 합니다.
fadeToggle()	fadeIn() 메서드와 fadeOut() 메서드를 번갈아 실행합니다.

표 17-1의 메서드는 다음 네 가지 형태로 사용합니다.

- 1 \$(selector).method();
- 2 \$(selector).method(speed);
- 3 \$(selector).method(speed, callback);
- 4 \$(selector).method(speed, easing, callback);

각 매개변수는 다음과 같은 의미가 있습니다.

- **speed**

- 효과를 진행할 속도를 지정합니다.
- 밀리 초 단위의 숫자 또는 문자열 slow, normal, fast를 입력합니다.

- **callback**

- 효과를 모두 완료하고 실행할 함수를 지정합니다.

- **easing**

- 애니메이션의 easing 형태를 지정합니다.
- 별도의 플러그인을 사용하지 않으면 문자열 linear와 swing만 입력 가능합니다.

예제를 만들며 사용해봅시다. body 태그를 코드 17-1처럼 구성합니다. button 태그를 누르면 button 태그 바로 다음에 있는 div 태그를 사라지게도 하고 생기게도 합니다.

코드 17-1 body 태그 구성

```
<body>
  <button>Toggle Show</button>
  <div class="page">
    <h1>Lorem ipsum dolor sit amet</h1>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. </p>
  </div>
</body>
```

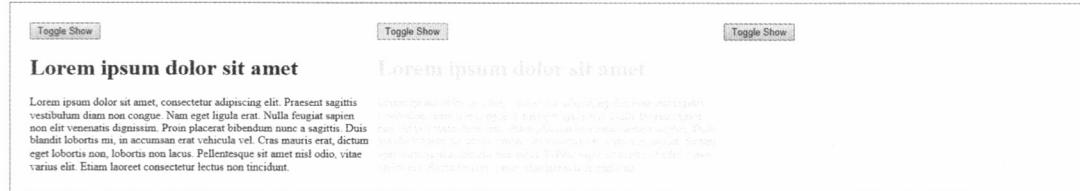
코드 17-2처럼 button 태그에 클릭 이벤트를 연결합니다. button 태그가 클릭되었을 때 .page 태그에 toggle() 메서드를 사용합니다.

코드 17-2 toggle() 효과 메서드

```
<script>
    // 이벤트를 연결합니다.
    $(document).ready(function () {
        // 이벤트를 연결합니다.
        $('button').click(function () {
            // 간단한 효과를 적용합니다.
            $('.page').toggle('slow');
        });
    });
</script>
```

toggle() 메서드는 show() 메서드와 hide() 메서드를 번갈아 실행하는 메서드입니다. 코드를 실행하고 OPEN & CLOSE를 누르면 그림 17-1처럼 div 태그가 사라졌다가 나타납니다.

그림 17-1 toggle() 효과 메서드

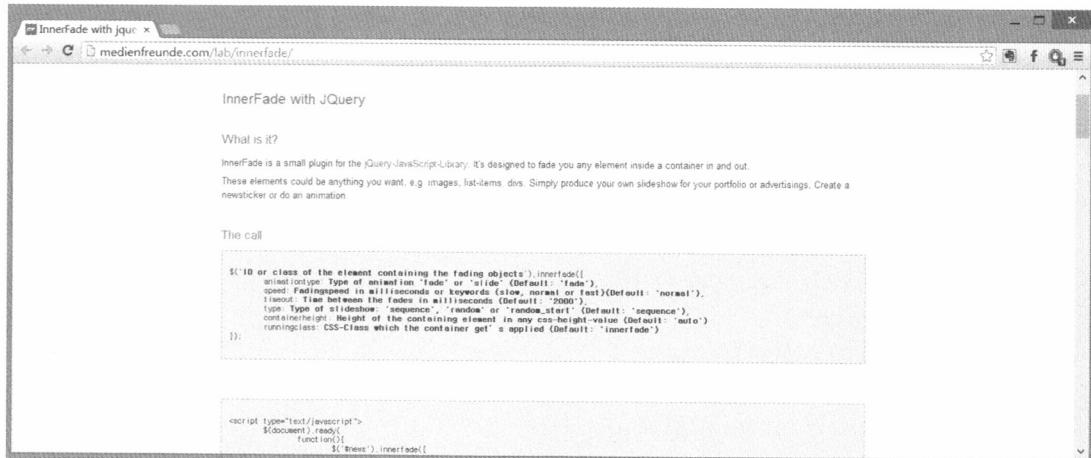


toggle() 메서드 이외에도 slideToggle() 메서드, fadeToggle() 메서드로도 사라졌다 나타나는 모습만 다르지 같은 기능을 수행합니다. 직접 사용해보세요. 이름 그대로 저수준의 시각적 효과이므로 쉽게 사용할 수 있을 것입니다.

17.2 innerfade 플러그인

이 절에서는 jQuery 효과와 관련된 innerfade 플러그인을 알아봅니다. jQuery innerfade 플러그인을 사용하면 간단하게 화면 전환 효과를 만들 수 있습니다. innerfade 플러그인은 그림 17-2의 <http://medienfreunde.com/lab/innerfade/>에서 내려받을 수 있습니다.

그림 17-2 http://medienfreunde.com/lab/innerfade/



innerfade 플러그인을 내려받은 후 압축 파일 내부의 jquery.innerfade.js 파일로 그림 17-3처럼 폴더를 구성합니다. 이미지는 윈도의 기본 내장 사진 샘플을 사용하겠습니다.

그림 17-3 폴더 구성



innerfade 플러그인을 사용하려면 코드 17-3처럼 head 태그를 구성해야 합니다. 모든 jQuery 플러그인은 jQuery.js 파일이 플러그인 파일보다 먼저 추가돼야 합니다.

코드 17-3 head 태그 구성

```
<head>
<style>
</style>
<script src="http://code.jquery.com/jquery-1.10.2.js"></script>
```

```
<script src="jquery.innerfade.js"></script>
<script>
$(document).ready(function () {
    });
</script>
</head>
```

body 태그는 코드 17-4처럼 구성합니다. innerfade 플러그인은 ul 태그에 적용하는 플러그인입니다. 내부의 li를 자동으로 화면 전환합니다. li 태그 내부에 img 태그를 넣으면 이미지가 자동으로 전환되겠죠?

코드 17-4 body 태그 구성

```
<body>
    <ul id="inner-fade">
        <li></li>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
    </ul>
</body>
```

코드 17-5처럼 style 태그를 구성했습니다.

코드 17-5 style 태그 구성

```
<style>
    * { margin: 0px; padding: 0px; }
    ul { list-style:none; }
    img { width: 500px; height: 350px; }
</style>
```

innerfade 플러그인의 가장 핵심적인 메서드는 innerfade() 메서드입니다. 코드 17-6처럼 ul 태그를 대상으로 innerfade() 메서드를 적용합니다. innerfade() 메서드의 첫 번째 매개변수에는 객체를 입력하는데요. 모든 jQuery 플러그인은 이렇게 입력해도 되고 안해도 되는 객체를 옵션 객체라고 부릅니다.

코드 17-6 innerfade() 메서드

```
<script>
$(document).ready(function () {
    // innerfade 플러그인을 적용합니다.
    $('#inner-fade').innerfade({
        animationtype: 'fade',
        speed: 750,
        timeout: 2000,
        type: 'random',
        containerheight: '350px'
    });
});
</script>
```

innerfade() 메서드는 표 17-2의 옵션 객체의 속성을 갖습니다.

표 17-2 innerfade() 메서드의 옵션

옵션 이름	설명	들어갈 수 있는 값
animationtype	내용물의 변경 효과	'fade', 'slide'
speed	내용물의 변경 속도	'slow', 'normal', 'fast', 숫자
timeout	변경 효과가 적용되는 속도	숫자
type	내용물의 변경 방식	'sequence', 'random', 'random_start'
containerheight	내용물의 높이	'auto', 숫자

코드를 실행하면 그림 17-4처럼 2초에 한 번 이미지가 변경됩니다.

그림 17-4 innerfade 플러그인



li 태그 내에 이미지가 아니라 어떠한 것이 있어도 됩니다. 여러 가지로 활용해보세요.

17.3 사용자 정의 효과

저수준의 효과만 사용해도 사용자에게 충분히 많은 시각적 효과를 보여줄 수 있습니다. 하지만, 조금 더 개발자가 원하는 방향으로 효과를 만들고 싶을 때는 표 17-3의 메서드를 사용합니다.

표 17-3 사용자 정의 효과 메서드

메서드 이름	설명
animate()	사용자 지정 효과를 생성합니다.

animate() 메서드는 다음 네 가지 형태로 사용합니다.

- 1 \$(selector).animate(object);
- 2 \$(selector).animate(object, speed);
- 3 \$(selector).animate(object, speed, easing);
- 4 \$(selector).animate(object, speed, easing, callback);

animate() 메서드는 첫 번째 매개변수에 객체가 들어간다는 것을 제외하면 저수준의 시각 효과 메서드와 같은 매개변수입니다. animate() 메서드의 첫 번째 매개변수인 객체에 입력할 수 있는 속성은 다음과 같습니다. 현재 형태에서 지정한 속성의 형태로 애니메이션이 동작합니다.

- opacity
- height
- top
- width
- left
- margin
- right
- padding
- bottom

사각형을 좌우로 움직이는 간단한 예제를 만들며 익혀봅시다. body 태그를 코드 17-7처럼 구성합니다.

코드 17-7 body 태그 구성

```
<body>
  <div></div><div></div>
  <div></div><div></div>
  <div></div><div></div>
</body>
```

style 태그를 작성합니다. style 태그는 이 절에서 가장 중요한 부분입니다. 위치 속성(left 속성과 top 속성)을 변경하는 animate() 메서드를 사용하려면 position 스타일 속성이 absolute이거나 relative여야 합니다.

코드 17-8 style 태그 구성

```
<style>
div {
  width: 50px; height: 50px;
  background-color: orange;
  position: relative;
}
</style>
```

NOTE 위치 속성과 관련된 내용은 18장에서 더 자세히 살펴봅니다.

이제 animate() 메서드를 사용해봅시다. 코드 17-9처럼 animate() 메서드의 첫 번째 매개 변수에 변경하고자 하는 속성을 입력했습니다. width 속성이나 height 속성 모두 쉽게 응용할 수 있겠죠?

코드 17-9 animate() 메서드

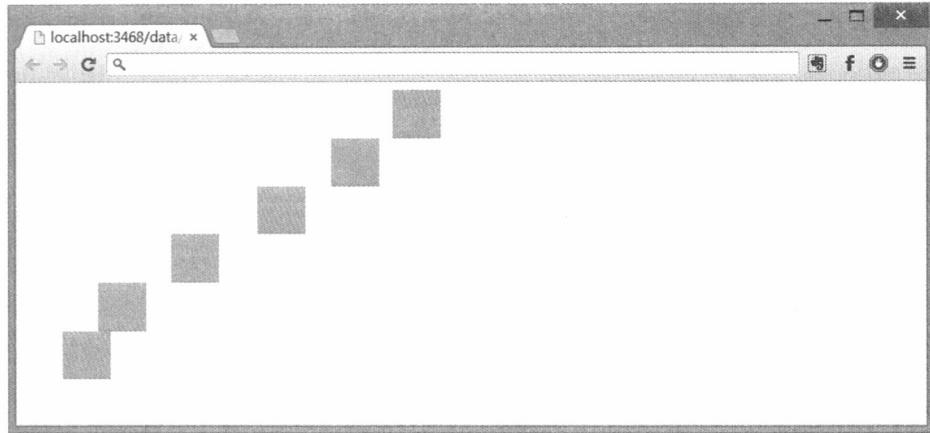
```
<script>
$(document).ready(function () {
  $('div').hover(function () {
```

```
$(this).animate({
    left: 500
}, 'slow');
}, function () {
    $(this).animate({
        left: 0
    }, 'slow');
});
});
```

```
</script>
```

코드를 실행하고 각각의 사각형 위에 마우스를 올려보세요. 그림 17-5처럼 각각의 사각형이 좌우로 이동하는 모습을 살펴볼 수 있습니다.

그림 17-5 이동하는 사각형



이 절에서 살펴본 left 스타일 속성 이외의 것도 직접 예제를 작성해보세요. animate() 메서드는 정말 많이 사용하는 메서드이므로 기본적인 사용 방법을 꼭 기억해주세요.

17.4 상대적 애니메이션

현재 상태에서 상대적으로 애니메이션을 적용하려면 어떻게 해야 할까요? animate() 메서드는 스타일 속성을 동적으로 변경해 애니메이션 효과를 내는 것이므로, 현재 스타일 속성을 알아내서 추가 값을 부여해 상대적으로 애니메이션을 적용할 수 있습니다.

우선, 이 방법으로 간단한 애니메이션을 만들어봅시다. `style` 태그를 다음과 같이 간단하게 구성합니다. `width` 스타일 속성과 `height` 스타일 속성을 변화시키는 예제를 작성할 것이므로 `position` 스타일 속성을 지정하지 않았습니다.

코드 17-10 `style` 태그 구성

```
<style>
  div {
    width: 50px; height: 50px;
    background-color: orange;
  }
</style>
```

`body` 태그에는 간단하게 코드 17-11처럼 `div` 태그를 하나만 넣습니다.

코드 17-11 `body` 태그 구성

```
<body>
  <div></div>
</body>
```

상대적으로 속성을 변화시키려면 현재 속성을 알아내야 합니다. 코드 17-12처럼 `css()` 메서드로 이벤트 발생 객체의 `width` 속성과 `height` 속성을 추출합니다.

코드 17-12 문서 객체의 스타일 속성 추출

```
<script>
$(document).ready(function () {
  $('div').click(function () {
    // 변수를 선언합니다.
    var width = $(this).css('width');
    var height = $(this).css('height');
  });
});
</script>
```

하지만, 현재 변수 width와 height는 숫자가 아니라 '50px' 형태의 문자열입니다. 상대적으로 크기를 추가하려면 숫자로 바꾼 뒤에 더해야 합니다. 코드 17-13처럼 parseInt() 함수를 사용하면 문자열 "50px"을 숫자 50으로 바꿀 수 있습니다. 현재 크기를 기준으로 50을 추가하므로 점점 커지는 애니메이션을 만들 수 있습니다.

코드 17-13 현재 상태를 기준으로 한 상대적 애니메이션(1)

```
$(‘div’).click(function () {
    // 변수를 선언합니다.
    var width = $(this).css(‘width’);
    var height = $(this).css(‘height’);

    // animate() 메서드를 사용합니다.
    $(this).animate({
        width: parseInt(width) + 50,
        height: parseInt(height) + 50
    }, ‘slow’);
});
```

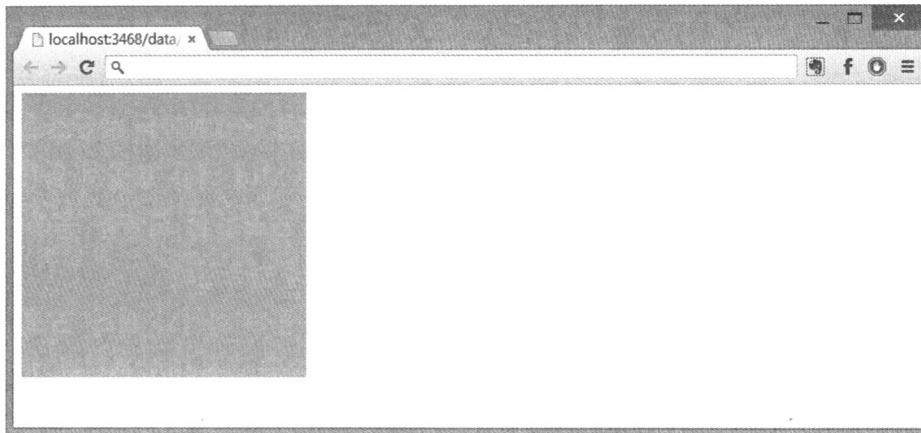
이렇게 현재 상태를 기준으로 속성을 상대적으로 변화시키는 형태의 애니메이션은 굉장히 많이 사용합니다. jQuery는 사용자의 편의를 위하여 animate() 메서드에 정말 유용한 형태의 방법을 제공합니다. 코드 17-14처럼 += 연산자나 -= 연산자를 사용하면 상대적으로 속성이 변합니다.

코드 17-14 현재 상태를 기준으로 한 상대적 애니메이션(2)

```
$(‘div’).click(function () {
    // animate() 메서드를 사용합니다.
    $(this).animate({
        width: ‘+=50’,
        height: ‘+=50’
    }, ‘slow’);
});
```

코드를 실행하고 사각형을 클릭하면 계속 사각형이 커집니다.

그림 17-6 누르면 점점 커지는 사각형



간단한 내용이지만 중요한 내용이므로 꼭 기억해주세요. 다음 절의 내용으로 넘어가기 전에 사각형을 마우스로 빠르게 여러 번 클릭해보세요. 마우스 클릭을 멈춘 후에도 계속해서 클릭 이벤트가 누적돼 사각형이 커집니다.

17.5 애니메이션 큐

앞 절의 마지막에서 사각형을 여러 번 클릭한 독자라면 알 수 있지만 jQuery의 효과 메서드는 계속 누적됩니다. 누적된 효과 명령은 큐 Queue에 누적됩니다. 큐는 먼저 들어간 것이 먼저 나오는 공간입니다. jQuery의 효과 메서드를 사용하면 명령이 차례로 큐에 들어가고, 들어간 순서대로 실행됩니다.

큐를 모르는 독자는 다음 코드를 실행해 큐에 명령이 하나씩 차례로 들어간다는 것이 무슨 의미인지 살펴봅시다. 코드 17-15는 animate() 메서드를 체이닝으로 연결했습니다. 애니메이션이 한꺼번에 실행될지 차례대로 실행될지 예측해보세요.

코드 17-15 HTML 페이지 구성

```
<!DOCTYPE html>
<html>
<head>
<style>
```

```


코드를 실행하면 동시에 애니메이션이 실행되는 것이 아니라 차례대로 실행됩니다. 각 애니메이션을 2초 동안 작동하게 했으므로 6초 동안 애니메이션이 작동합니다. 이렇게 그림 17-7처럼 애니메이션 큐에 animate() 메서드의 내용이 차례로 하나씩 들어가고 나오면서 실행하므로 애니메이션이 순서대로 동작합니다.



그림 17-7 애니메이션 큐



CHAPTER 17 효과 487



IMAG0129.JPG


```

마치 놀이공원에 가서 놀이기구를 타는 것처럼 먼저 들어간 명령을 차례대로 실행합니다. 큐에 있는 내용을 제거하고 싶으면 표 17-4의 메서드를 사용합니다.

표 17-4 애니메이션 큐 관리 메서드

메서드 이름	설명
clearQueue()	큐의 내용을 제거합니다.

코드 17-16은 애니메이션이 동작하고 3초 후에 clearQueue() 메서드를 사용하는 예제입니다. 4초 후에 clearQueue() 메서드를 실행하게 했다면 결과를 더 쉽게 예측할 수 있겠지만, 일단 코드 17-16의 결과를 예측해봅시다.

코드 17-16 clearQueue() 메서드

```
<script>
$(document).ready(function () {
    // animate() 메서드를 사용합니다.
    $('div').animate({ left: '+=60' }, 2000);
    $('div').animate({ width: '+=60' }, 2000);
    $('div').animate({ height: '+=60' }, 2000);

    // 3초 후에 함수를 실행합니다.
    setTimeout(function () {
        // 애니메이션 큐를 제거합니다.
        $('div').clearQueue();

        // 효과 메서드를 사용합니다.
        $('div').hide();
    }, 3000);
});
</script>
```

대부분의 독자가 left 스타일 속성 애니메이션이 실행된 후에 width 스타일 속성 애니메이션이 실행되면서 애니메이션이 모두 종료되고, 큐를 비운 뒤에 hide() 메서드를 실행할 것이라고 예측했을 것입니다. 얼핏 보면 그렇게 실행되는 것처럼 보입니다.

hide() 메서드를 제거하고 다시 한 번 실행해보세요. width 스타일 속성 애니메이션은 모두 끝까지 진행합니다. clearQueue() 메서드가 큐를 비우므로 이후에 추가하는 효과가 실행되지 않지만, 이전에 실행되던 애니메이션을 정지하는 기능은 없다는 사실을 기억해주세요.

17.6 애니메이션 정지

clearQueue() 메서드에는 애니메이션 정지 기능이 없습니다. 애니메이션을 정지시키려면 표 17-5의 메서드를 사용합니다.

표 17-5 애니메이션 정지 메서드

메서드 이름	설명
stop()	효과 및 애니메이션을 정지합니다.

stop() 메서드는 다음 세 가지 형태로 사용합니다.

- 1 \$(selector).stop();
- 2 \$(selector).stop(clearQueue);
- 3 \$(selector).stop(clearQueue, goToEnd);

매개변수 clearQueue와 goToEnd는 불을 입력하며, 입력하지 않으면 자동으로 false를 입력한 것으로 간주합니다. 첫 번째 매개변수 clearQueue를 true로 설정하면 clearQueue() 메서드를 실행하는 것과 같은 효과를 냅니다. 두 번째 매개변수 goToEnd를 true로 설정하면 제자리에서 멈추는 것이 아니라 지정한 최종 형태에서 멈춥니다.

간단하게 stop() 메서드가 나올 수 있는 모든 형태의 조건을 사용해봅시다. style 태그를 코드 17-17처럼 구성해주세요.

코드 17-17 style 태그 구성

```
<style>
  div {
    width: 100px; height: 100px;
    background-color: orange;
    position: relative;
  }
</style>
```

body 태그는 코드 17-18처럼 버튼 4개를 놓아줍니다. 각각 이름이 의미하는 형태의 메서드를 eval() 함수로 실행할 것이므로 대소문자에 유의해주세요.

코드 17-18 body 태그 구성

```
<body>
  <button>stop()</button>
  <button>stop(true)</button>
  <button>stop(false, true)</button>
  <button>stop(true, true)</button>
  <div></div>
</body>
```

script 태그는 코드 17-19처럼 구성합니다. div 태그는 2초마다 좌우로 한 번씩 이동하게 설정하고, button 태그를 클릭하면 각 메서드를 실행하게 만들었습니다.

코드 17-19 stop() 메서드

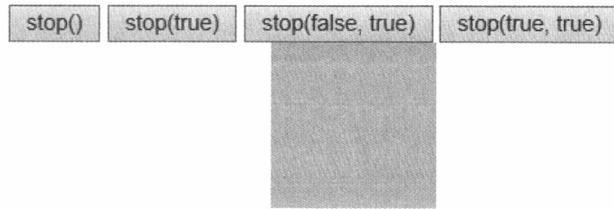
```
<script>
$(document).ready(function () {
  // 이벤트를 연결합니다.
  $('button').click(function () {
    // 변수를 선언합니다.
    var html = $(this).html();
    var evalText = "$('div')." + html;

    // 메서드를 실행합니다.
    eval(evalText);
  });

  // 애니메이션을 시작합니다.
  setInterval(function () {
    $('div').animate({
      left: '500'
    }, 1000).animate({
      left: '0'
    }, 1000);
  }, 2000);
});
</script>
```

각 메서드가 어떤 차이를 보이는지 설명하는 것보다 직접 보는 것이 빠릅니다. 코드를 실행해 메서드가 어떠한 차이를 보이는지 확인해보세요. 사각형이 오른쪽으로 움직일 때 버튼을 누르면 차이를 더 확실히 알 수 있습니다.

그림 17-8 실행 결과



사각형이 오른쪽으로 움직일 때 어떤 식으로 실행하는지 정리하면 다음과 같습니다.

- 1 **stop()** 오른쪽으로 이동하는 것을 멈추고 제자리에서 바로 왼쪽으로 다시 이동합니다.
- 2 **stop(true)** 오른쪽으로 이동하는 것을 멈추고, 다음 setInterval() 함수가 실행될 때까지 대기합니다.
- 3 **stop(false, true)** 버튼을 누르는 순간 사각형이 오른쪽 끝으로 이동하고, 바로 왼쪽으로 이동합니다.
- 4 **stop(true, true)** 버튼을 누르는 순간 사각형이 오른쪽 끝으로 이동하고, 다음 setInterval() 함수가 실행될 때까지 대기합니다.

17.7 애니메이션 지연

지금까지 사용한 모든 효과 메서드는 메서드를 실행하자마자 바로 시작했습니다. jQuery의 효과 관련 메서드를 특정한 시간 후에 실행하고 싶을 때는 표 17-6의 메서드를 사용합니다.

표 17-6 애니메이션 지연 메서드

메서드 이름	설명
delay()	큐에 있는 명령을 잠시 중지합니다.

delay() 메서드의 매개변수에 정지하고자 하는 시간을 밀리 초 단위로 입력합니다. 코드 17-20처럼 예제를 만들면 사각형이 순서대로 오른쪽으로 이동하는 효과를 볼 수 있습니다.

코드 17-20 HTML 페이지 구성

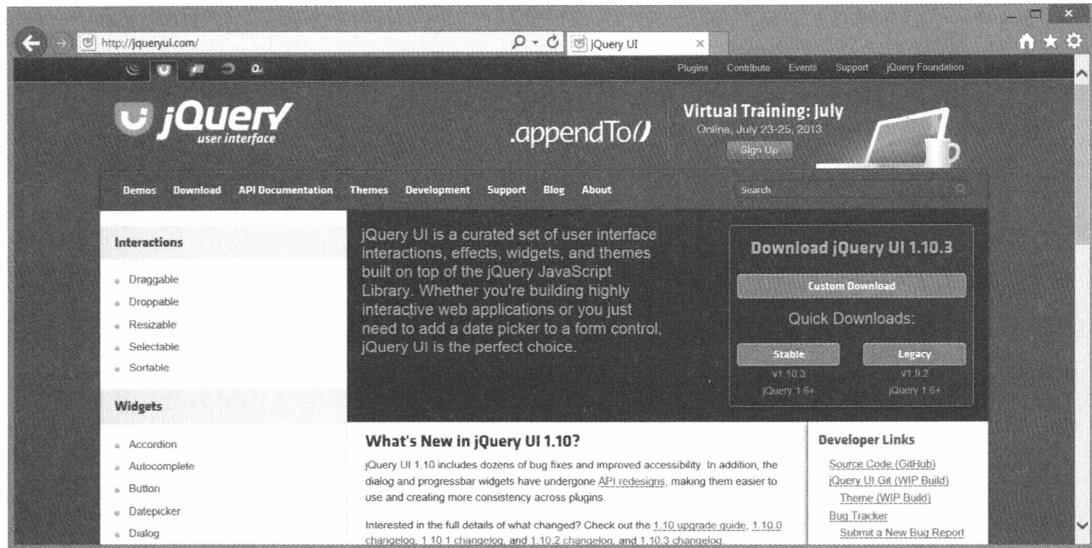
```
<!DOCTYPE html>
<html>
<head>
<style>
```

```
div {
    width: 100px; height: 100px;
    background-color: orange;
    position: relative;
}
</style>
<script src="http://code.jquery.com/jquery-1.10.2.js"></script>
<script>
$(document).ready(function () {
    $('div').each(function (index) {
        // (index * 500)초 후 animate() 메서드를 실행합니다.
        $(this).delay(index * 500).animate({
            left: 500
        }, 'slow');
    });
});
</script>
</head>
<body>
<div></div><div></div>
<div></div><div></div>
<div></div><div></div>
</body>
</html>
```

17.8 jQuery UI Effect 플러그인

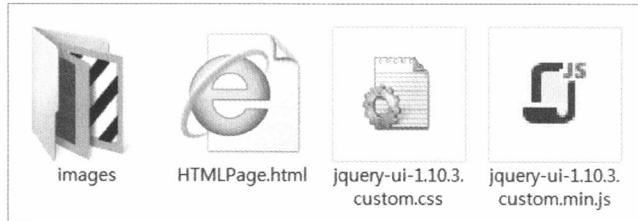
jQuery UI 플러그인은 jQuery에서 공식적으로 지정한 플러그인입니다. 2008년 초까지 많이 사용한 유명한 jQuery 플러그인을 모아 정리한 것인데요. 자세한 내용은 4부에서 다룹니다. jQuery UI 플러그인의 공식 사이트 <http://jqueryui.com/>에서 jQuery UI 플러그인을 내려 받습니다.

그림 17-9 jQuery UI Effect 플러그인 공식 사이트



내려받을 때 체크된 것이 굉장히 많은데요. 이 책의 전체에 걸쳐 많이 사용되므로 모두 체크된 상태에서 내려받으세요. 내려받은 후에 압축 파일에 들어 있는 development–bundle 폴더를 제외한 나머지 폴더에 있는 파일로 그림 17-10처럼 구성합니다.

그림 17-10 폴더 구성



코드 17-21 head 태그를 구성합니다. 사실 지금부터 배울 jQuery UI 플러그인의 Effect 부분을 사용할 때는 CSS 파일을 추가할 필요가 없습니다.

코드 17-21 head 태그 구성

```
<head>
    <link rel="stylesheet" href="ui-lightness/jquery-ui-1.10.3.custom.css" />
    <style>

    </style>
```

```
<script src="http://code.jquery.com/jquery-1.10.2.js"></script>
<script src="jquery-ui-1.10.3.custom.min.js"></script>
<script>
$(document).ready(function () {

});

</script>
</head>
```

jQuery UI Effect 플러그인은 2008년 초에 정말 많이 사용된 Effect 관련 플러그인을 모아 정리한 것입니다. jQuery UI Effect 플러그인에는 다음 기능이 있습니다.

- 색상 변환 애니메이션
- addClass(), removeClass(), switchClass() 메서드에 애니메이션 기능 추가
- 고수준의 시각적 효과
- 추가적인 easing 형태

고수준의 시각적 효과는 거의 사용하지 않는 내용이므로 이 책에서 다루지 않습니다. 이어지는 절에서 나머지 내용을 차례대로 살펴봅시다.

17.8.1 색상 변환 애니메이션

색상 변환 애니메이션과 함께 addClass() 메서드와 removeClass() 메서드의 애니메이션 기능을 함께 알아보겠습니다. addClass() 메서드와 removeClass() 메서드를 사용하려면 당연히 코드 17-22처럼 style 태그 안에 클래스를 정의해야겠죠?

코드 17-22 style 태그 구성

```
<style>
.reverse {
    color: white;
    background-color: black;
}
</style>
```

body 태그에 클래스를 적용할 태그를 만듭니다. 코드 17-23은 div 태그를 세 개 놓아주었습니다. div 태그가 공간을 차지하려면 내용물이 있어야 하므로 간단하게 h1 태그와 p 태그를 입력했습니다.

코드 17-23 body 태그 구성

```
<body>
  <div>
    <h1>Lorem ipsum</h1>
    <p>Lorem ipsum dolor sit amet</p>
  </div>
  <div>
    <h1>Lorem ipsum</h1>
    <p>Lorem ipsum dolor sit amet</p>
  </div>
  <div>
    <h1>Lorem ipsum</h1>
    <p>Lorem ipsum dolor sit amet</p>
  </div>
</body>
```

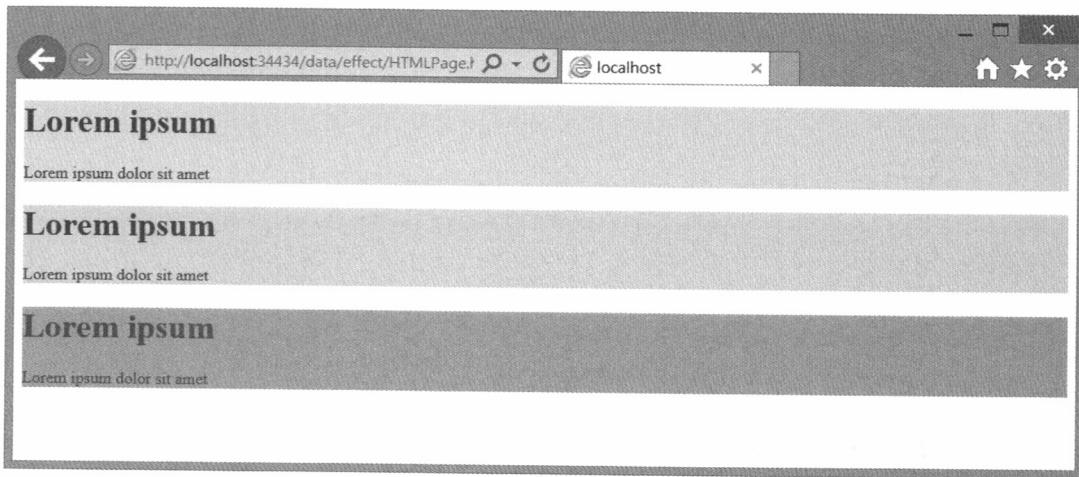
jQuery UI Effect 플러그인으로 addClass() 메서드와 removeClass() 메서드에 애니메이션을 부여하는 작업은 굉장히 간단합니다. 코드 17-24처럼 각 메서드의 두 번째 매개변수에 애니메이션 적용 속도를 입력합니다.

코드 17-24 script 태그 구성

```
<script>
$(document).ready(function () {
  $('div').hover(function () {
    $(this).addClass('reverse', 1000);
  }, function () {
    $(this).removeClass('reverse', 1000);
  });
});
</script>
```

코드를 실행하고 마우스로 각각의 태그 위에 마우스 커서를 올려보세요. 그림 17-11처럼 색상 변화 효과가 적용됩니다.

그림 17-11 색상 변환 애니메이션



이번 예제에서는 `clearQueue()` 메서드나 `stop()` 메서드를 사용하지 않았으므로 약간 실행 결과가 지저분해 보일 수도 있습니다. 이전에 배운 `clearQueue()` 메서드와 `stop()` 메서드로 깔끔하게 수정해보세요.

17.8.2 easing 플러그인

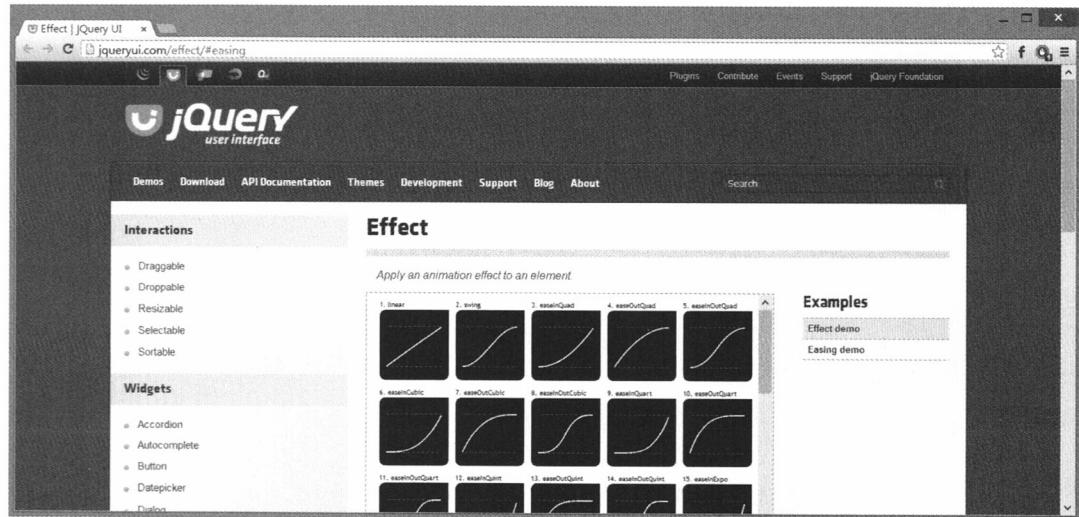
기존의 jQuery를 사용할 때 효과 관련 메서드의 `easing` 속성에 문자열 `swing`이나 `linear`밖에 입력할 수 없었습니다. jQuery UI Effect 플러그인을 사용하면 `easing` 속성에 훨씬 다양한 형태의 문자열을 넣을 수 있습니다. jQuery UI Effect 플러그인이 제공하는 `easing` 속성은 표 17-7과 같습니다.

표 17-7 다양한 종류의 easing 속성

linear	swing	easeInQuad	easeOutQuad
easeInOutQuad	easeInCubic	easeOutCubic	easeInOutCubic
easeInQuart	easeOutQuart	easeInOutQuart	easeInQuint
easeOutQuint	easeInOutQuint	easeInSine	easeOutSine
easeInOutSine	easeInExpo	easeOutExpo	easeInOutExpo
easeInCirc	easeOutCirc	easeInOutCirc	easeInElastic
easeOutElastic	easeInOutElastic	easeInBack	easeOutBack
easeInOutBack	easeInBounce	easeOutBounce	easeInOutBounce

jQuery UI 공식 홈페이지의 Demos 탭에 들어가 jQuery UI Effect 플러그인을 확인해보세요. 그림 17-12처럼 각각의 easing 속성이 어떠한 형태를 의미하는지 알 수 있습니다.

그림 17-12 <http://jqueryui.com/effect/#easing>



간단하게 모든 easing 속성이 어떠한 방식으로 작동하는지 알아볼 수 있는 예제를 작성합시다.

body 태그를 코드 17-25처럼 구성합니다. 생략된 부분에 표 17-7의 모든 속성을 option 태그에 넣어서 적어주세요. 총 32개의 option 태그를 입력하면 됩니다. 이어서 button 태그와 움직이는 사각형이 되어줄 div 태그를 만듭니다.

코드 17-25 body 태그 구성

```
<body>
  <select>
    <option>linear</option>
    <option>swing</option>
    <option>easeInQuad</option>
    <!-- 생략 -->
    <option>easeInBounce</option>
    <option>easeOutBounce</option>
    <option>easeInOutBounce</option>
  </select>
  <button>MOVE</button>
  <div></div>
</body>
```

코드 17-26처럼 style 태그를 구성합니다.

코드 17-26 style 태그 구성

```
<style>
div {
    background-color:orange;
    width: 150px; height: 150px;
    position: relative;
}
</style>
```

코드 17-27처럼 선택한 option 문서 객체의 내부 문자열을 가져와 animate() 메서드의 세 번째 매개변수에 입력합니다. 선택한 option 문서 객체를 선택하는 선택자는 자주 사용하므로 기억해두세요.

코드 17-27 easing 속성을 사용한 animate() 메서드

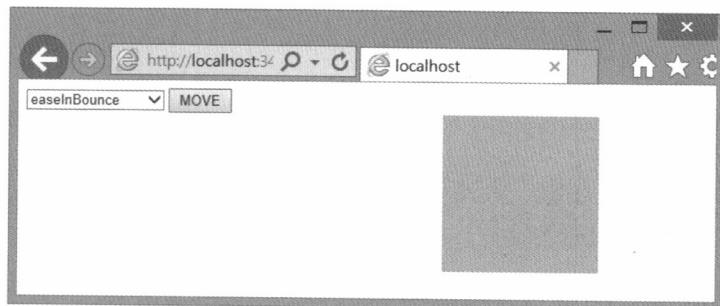
```
<script>
$(document).ready(function () {
    // 이벤트를 연결합니다.
    $('button').click(function () {
        // 변수를 선언합니다.
        var easing = $('select > option:selected').html();

        // animate() 메서드를 사용합니다.
        $('div').animate({
            left: '400'
        }, 2000, easing).animate({
            left: '0'
        }, 1000, easing);
    });
});
</script>
```

코드를 실행하면 그림 17-13처럼 구성돼 있고, 시험해보려는 easing 속성을 선택해 테스트할 수 있습니다. 이번 예제는 정말 글로 설명할 수 없습니다. 직접 실행하고 어떤 방식으로 동작하는지 확인해보세요. 사각형이 이전보다 훨씬 재미있게 움직일 것입니다.



그림 17-13 easing 플러그인



이렇게 jQuery의 효과와 관련된 대부분의 내용을 살펴보았습니다. 전체적으로 어려운 부분은 없지만 플러그인 때문에 약간 어렵게 느끼는 독자가 있을 것입니다.

“플러그인과 관련된 내용은 모두 외워야 하나요?”

플러그인과 관련된 내용은 필요하면 찾아서 쓰면 됩니다. 또한 플러그인과 관련된 내용은 해당 플러그인의 공식 사이트에서 설명서를 읽으면 사용하면 됩니다.

17장에서 소개한 플러그인 이외에도 효과, 크기, 회전과 관련된 플러그인이 많으니 살펴보세요.