Web Worker(웹 작업자)

Internet Explorer 10 및 JavaScript 를 사용하는 Windows 스토어 앱은 웹 작업자를 새롭게 지원합니다.
웹 작업자 API 는 백그라운드에서 스크립트를 실행하는 방법을 정의합니다.
웹 작업자는 W3C(World Wide Web 컨소시엄)의 웹 작업자 사양에 지정되어 있습니다.

과거에는 브라우저가 단일 스레드 방식이어서, 응용 프로그램의 모든 스크립트가 단일 UI 스레드에서 함께 실행되었습니다.
DOM(문서 개체 모델) 이벤트 및 setTimeout API 를 사용하여 여러 작업이 동시에 일어나는 것 같은 효과를 줄 수 있지만, 많은 계산이 필요한 작업이 하나만 있어도 브라우저 작동이 중단됩니다.

웹 작업자 API 를 사용하면 웹 응용 프로그램 작성자가 기본 페이지와 함께 실행되는 백그라운드 스크립트를 생성할 수 있습니다.
장기 실행 작업에 사용할 여러 스레드를 동시에 생성할 수 있습니다.
새 작업자 개체에는 .js 파일이 필요하며, 이 파일은 서버에 대한 비동기 요청을 통해 포함됩니다.

JavaScript :
```
var myWorker = new Worker('worker.js');
```

작업자 스레드와의 모든 통신은 메시지를 통해 관리됩니다.
호스트 작업자와 작업자 스크립트 모두 postMessage 를 사용하여 메시지를 보내고, onmessage 이벤트를 사용하여 응답을 수신 대기할 수 있습니다.
메시지 내용은 이벤트의 data 속성으로 전송됩니다.

다음 예제에서는 작업자 스레드를 만들고 메시지를 수신 대기합니다.
JavaScript :
```
var hello = new Worker('hello.js');
hello.onmessage = function(e) {
  alert(e.data);
};
```

작업자 스레드는 표시할 메시지를 보냅니다.
JavaScript:
```
postMessage('Hello world!');
```

웹 작업자와의 양방향 통신

양방향 통신을 설정하기 위해 기본 페이지와 작업자 스레드 모두 **onmessage** 이벤트를 수신 대기합니다.

다음 예제에서 작업자 스레드는 지정된 지연 후에 메시지를 반환합니다.
먼저 스크립트가 작업자 스레드를 만듭니다.

JavaScript:

```
var echo = new Worker('echo.js');
echo.onmessage = function(e) {
  alert(e.data);
}
```

메시지 텍스트와 시간 제한 값은 양식에서 지정됩니다.
사용자가 제출 단추를 클릭하면 스크립트에서 JavaScript 개체 리터럴에 두 가지 정보를 포함하여 작업자에게 전달합니다.
페이지가 새 HTTP 요청의 양식 값을 제출할 수 없도록 이벤트 처리기에서 이벤트 개체에 대해 **preventDefault** 도 호출합니다.
DOM 개체에 대한 참조를 작업자 스레드로 보낼 수는 없습니다.
웹 작업자는 액세스할 수 있는 데이터가 제한됩니다.
개체, 문자열 값 등의 JavaScript primitive 만 허용됩니다.

HTML:

```
<script>
window.onload = function() {
  var echoForm = document.getElementById('echoForm');
  echoForm.addEventListener('submit', function(e) {
    echo.postMessage({
      message : e.target.message.value,
      timeout : e.target.timeout.value
    });
    e.preventDefault();
  }, false);
  }
</script>
<form id="echoForm">
  <p>Echo the following message after a delay.</p>
  <input type="text" name="message" value="Input message here."/><br/>
  <input type="number" name="timeout" max="10" value="2"/> seconds.<br/>
  <button type="submit">Send Message</button>
</form>
```

마지막으로, 작업자가 메시지를 수신 대기하고 지정된 시간 제한 간격 후에 메시지를 반환합니다.
JavaScript:

```
onmessage = function(e)
{
  setTimeout(function()
  {
    postMessage(e.data.message);
  },
  e.data.timeout * 1000);
}
```

Internet Explorer 10 및 JavaScript 를 사용하는 Windows 스토어 앱에서 웹 작업자 API 는 다음 메서드를 지원합니다.

메서드	설명
void close();	작업자 스레드를 종료합니다.
void importScripts(**inDOMString**... *urls*);	쉼표로 구분된 추가 JavaScript 파일 목록을 가져옵니다.
void postMessage(모든 데이터);	작업자 스레드와 메시지를 주고 받습니다.

Internet Explorer 10 및 JavaScript 를 사용하는 Windows 스토어 앱은 다음 웹 작업자 API 특성을 지원합니다.

특성	유형	설명
location	WorkerLocation	**protocol**, **host**, **port**, **hostname**,**pathname**, **search** 및 **hash** 구성 요소를 포함하여 절대 URL 을 나타냅니다.
navigator	WorkerNavigator	사용자 에이전트 클라이언트의 ID 와 **onLine** 상태를 나타냅니다.
self	WorkerGlobalScope	WorkerLocation 및 WorkerNavigator 개체를 포함하는 작업자 범위입니다.

Internet Explorer 10 및 JavaScript 를 사용하는 Windows 스토어 앱은 다음 웹 작업자 API 이벤트를 지원합니다.

이벤트	설명
onerror	런타임 오류가 발생했습니다.
onmessage	메시지 데이터가 수신되었습니다.

WindowTimers

웹 작업자 API 는 업데이트된 HTML5 *WindowTimers* 기능도 지원합니다.

메서드	설명
void clearInterval(**in** **long** *handle*);	핸들에서 식별된 시간 제한을 취소합니다.
void clearTimeout(**in** **long** *handle*);	핸들에서 식별된 시간 제한을 취소합니다.
long setInterval(**in** *any* *handler*, **in** **optional** *any* *timeout*, **in** *any*... *args*);	지정한 시간(밀리초) 후에 시간 제한이 반복해서 실행되도록 예약합니다. 이제 추가 인수를 처리기에 직접 전달할 수 있습니다. 처리기가 **DOMString** 인 경우 JavaScript 로 컴파일됩니다. 시간 제한에 대한 핸들을 반환합니다.clearInterval 로 지웁니다.
long setTimeout(**in** *any* *handler*, 모든 선택적 시간 제한, 모든 인수);	지정한 시간(밀리초) 후에 시간 제한이 실행되도록 예약합니다. 이제 추가 인수를 처리기에 직접 전달할 수 있습니다. 처리기가 **DOMString** 인 경우 JavaScript 로 컴파일됩니다. 시간 제한에 대한 핸들을

	반환합니다.clearTimeout 로 지웁니다.

IE10 Platform Preview Build 4 의 웹 작업자 업데이트

Internet Explorer 10 Platform Preview Build 4 에서는 프로세스당 웹 작업자 스레드 25 개의 제한을 적용합니다. 스크립트에서 추가 작업자를 만들 수 있지만 동시에 활성화되는 작업자 수는 25 개뿐입니다.
최대 스레드 수에 도달한 경우 작업자를 만들 때 예외가 발생하지 않습니다.
호출이 항상 성공하며, 처리기를 추가하고 메시지를 게시할 수 있습니다. 그러나 기존 25 개 스레드 중 하나를 사용할 수 있을 때까지 작업자가 시작되지 않습니다.

JavaScript:

```
// Coding pattern before IE10 Platform Preview Build 4
try {
    var worker = new Worker(url);
} catch(ex) {
    // IE might throw...?
}

// After IE10 Platform Preview Build 4
var worker = new Worker(url);
// Continue with confidence...
```

API 참조

Web Workers

샘플 및 자습서

웹 작업자 샘플

HTML5 를 사용하여 Mandelbrot 집합을 탐색하는 방법

Internet Explorer 테스트 드라이브 데모

Mandelbrot 탐색기

웹 작업자 분수

test262 의 웹 작업자 테스트 도구

IEBlog 게시물

IE10 에서 웹 작업자 디버그

IE10 의 웹 작업자: 백그라운드 JavaScript 를 통해 웹앱 속도 향상

사양

웹 작업자

관련 항목

웹 작업자를 사용한 HTML5 스레딩 및 IndexedDB 를 사용한 데이터 저장소

HTML5 웹 작업자 소개: JavaScript 다중 스레딩 방법