**Using SharedWebWorker:**

var myWorker = new SharedWorker(*aURL*, *name*);

aURL : path of shared-worker.js file. It must obey the same-origin policy.

name:

* Optional argument
* specifies an existing SharedWorkerGlobalScope.name — if this is specified then that [SharedWorkerGlobalScope](https://developer.mozilla.org/en-US/docs/Web/API/SharedWorkerGlobalScope) will be used as the scope for this shared worker.

**Note:**

* The object returned by the SharedWorker () constructor holds a reference to the port on its port attribute. For e.g :

worker.port.postMessage('some message');

**Connecting to multiple pages**:

* The script uses the onconnect event listener to listen for multiple connections.

**Direct Channel**:

* When the worker receives a “msg” message from one viewer naming another viewer, it sets up a direct connection between the two, so that the two viewers can communicate directly without the worker having proxy all the messages.

**Pros:**

* With multicore CPUs , to obtain better performance is to split computationally expensive tasks amongst multiple workers( shared workers).

**Main.js**

if(SharedWorker) {

var shared\_worker = new SharedWorker("src/js/worker.js");

shared\_worker.port.addEventListener("message", function(evt){

console.log("main thread : " + evt.data);

}, false);

shared\_worker.port.addEventListener('error', function(e){

throw new Error(' Error: could not open SharedWorker', e);

}, false);

shared\_worker.port.start();

shared\_worker.port.postMessage("Shared worker");

} else {

console.log("not supported " );

}

**worker.js**

self.addEventListener("connect", function(evt){

var port = evt.ports[0];

port.addEventListener("message",function(evt){

port.postMessage(" message from shared worker");

},false);

port.start();

},false);

EXAMPLE 2 -

In this case, the second page is merely in an iframe on the first page, but the same principle would apply to an entirely separate page in a separate top-level browsing context.

INDEX.HTML

<!doctype html>

<html lang="en" ng-app="myApp">

<head>

  <meta charset="UTF-8">

  <title>Shared Web worker </title>

<link rel="stylesheet" type="text/css" href="css/main.css">

</head>

<body>

<pre id="log">Log:</pre>

<!-- <iframe src="src/html/inner.html"></iframe> -->

<script type="text/javascript" src="src/js/main.js"></script>

</body>

</html>

INNER.HTML

<!doctype html>

<html lang="en" ng-app="myApp">

<head>

  <meta charset="UTF-8">

  <title>Shared Web worker </title>

</head>

<body>

<pre id="log"> Inner Log:</pre>

<script type="text/javascript" src="../js/main.js"></script>

</body>

</html>

MAIN.JS

var worker = new SharedWorker('/src/js/sharedWorker.js');

worker.port.addEventListener("message", function(e) {

var log = document.getElementById('log');

log.textContent += '\n' + e.data;

});

worker.port.start();

SHAREDWORKER.JS

var count = 0;

self.addEventListener("connect", function(e){

count += 1;

var port = e.ports[0];

port.postMessage('Hello World! You are connection #' + count);

self.addEventListener("message",function(event){

port.postMessage('pong');

});

},false);