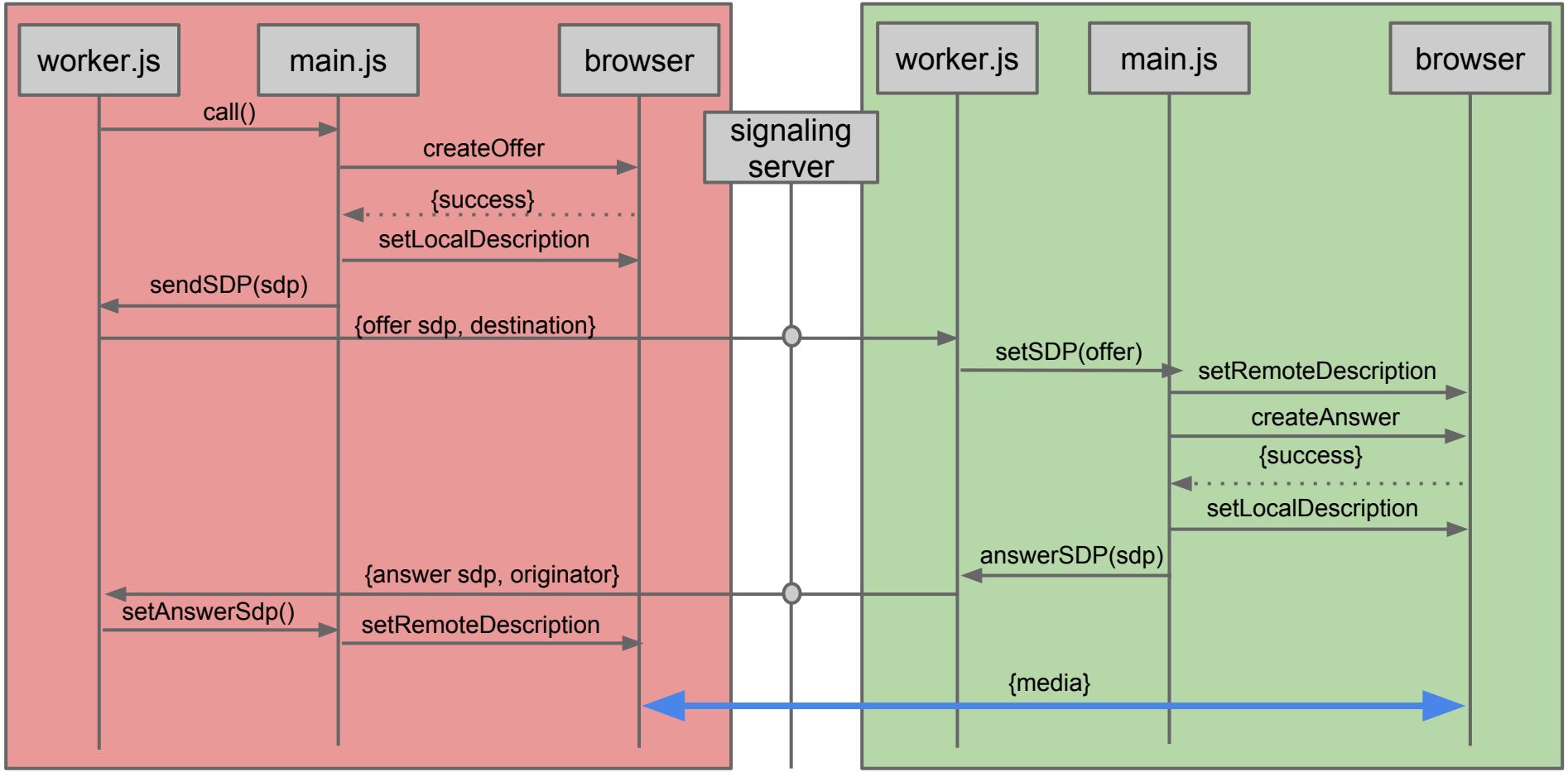
Workflow


worker.js
main.js
browser
call()
createOffer
{success}
setLocalDescription
sendSDP(sdp)
{offer sdp, destination}
signaling server
worker.js
main.js
browser
setSDP(offer)
setRemoteDescription
createAnswer
{success}
setLocalDescription
answerSDP(sdp)
{answer sdp, originator}
setAnswerSdp()
setRemoteDescription
{media}

Elements

Element	Description
worker.js	WebWorker. It is an operating system thread with limited access to the browser resources, not allowed to interact neither with the webRTC API nor the DOM
main.js	Javascript code to run in the main execution thread. It has the ability to interact with the webRTC API, so it will play as a “proxy” for the webWorkers
browser	Refers to the javascript APIs exposed by the browser
signaling server	Intermediate element needed to transport the SDP in the negotiation process.

Interfaces. Communication between elements

Element	Description

worker.js - main.js	Object serialization. WebWorker API provides a mechanism to interchange objects with the main thread. We will send JSON objects like this:
	
	`{`
	`  method: <some method>,`
	`  data: <data for this method>`
	`}`
	
	This way, it is possible to emulate a RPC mechanism

worker.js - worker.js	This communication is accomplished through the signaling server. Both worker.js know the address of this server, so it can be used to relay messages between them
main.js - browser	Javascript API
browser-browser	WebRTC has a mechanism to find out the route between the browsers involved in the call. This mechanism performs the ICE gathering process, which will end up with an available channel connecting the two endpoints.