Class: Debugger

> *An alternate transport for Chrome's remote debugging protocol.*

Process: Main
This class is not exported from the `'electron'` module. It is only available as a return value of other methods in the Electron API.

Chrome Developer Tools has a special binding available at JavaScript runtime that allows interacting with pages and instrumenting them.

```
const { BrowserWindow } = require('electron')
const win = new BrowserWindow()

try {
  win.webContents.debugger.attach('1.1')
} catch (err) {
  console.log('Debugger attach failed : ', err)
}

win.webContents.debugger.on('detach', (event, reason) => {
  console.log('Debugger detached due to : ', reason)
})

win.webContents.debugger.on('message', (event, method, params) => {
  if (method === 'Network.requestWillBeSent') {
    if (params.request.url === 'https://www.github.com') {
      win.webContents.debugger.detach()
    }
  }
})

win.webContents.debugger.sendCommand('Network.enable')
```

Instance Events

Event: 'detach'

Returns:

- `event` Event
- `reason` string - Reason for detaching debugger.

Emitted when the debugging session is terminated. This happens either when `webContents` is closed or devtools is invoked for the attached `webContents`.

Event: 'message'

Returns:

- `event` Event
- `method` string - Method name.
- `params` any - Event parameters defined by the 'parameters' attribute in the remote debugging protocol.

- `sessionId` string - Unique identifier of attached debugging session, will match the value sent from `debugger.sendCommand`.

Emitted whenever the debugging target issues an instrumentation event.

Instance Methods

`debugger.attach([protocolVersion])`

- `protocolVersion` string (optional) - Requested debugging protocol version.

Attaches the debugger to the `webContents`.

`debugger.isAttached()`

Returns `boolean` - Whether a debugger is attached to the `webContents`.

`debugger.detach()`

Detaches the debugger from the `webContents`.

`debugger.sendCommand(method[, commandParams, sessionId])`

- `method` string - Method name, should be one of the methods defined by the remote debugging protocol.
- `commandParams` any (optional) - JSON object with request parameters.
- `sessionId` string (optional) - send command to the target with associated debugging session id. The initial value can be obtained by sending Target.attachToTarget message.

Returns `Promise<any>` - A promise that resolves with the response defined by the 'returns' attribute of the command description in the remote debugging protocol or is rejected indicating the failure of the command.

Send given command to the debugging target.