Class: ProxyAgent

Extends: `undici.Dispatcher`

A Proxy Agent class that implements the Agent API. It allows the connection through proxy in a simple way.

new ProxyAgent([options])

Arguments:

- **options** `ProxyAgentOptions` (required) - It extends the `Agent` options.

Returns: `ProxyAgent`

Parameter: `ProxyAgentOptions`

Extends: `AgentOptions`

- **uri** `string` (required) - It can be passed either by a string or a object containing `uri` as string.

Examples:

```
import { ProxyAgent } from 'undici'

const proxyAgent = new ProxyAgent('my.proxy.server')
// or
const proxyAgent = new ProxyAgent({ uri: 'my.proxy.server' })
```

Example - Basic ProxyAgent instantiation This will instantiate the ProxyAgent. It will not do anything until registered as the agent to use with requests.

```
import { ProxyAgent } from 'undici'

const proxyAgent = new ProxyAgent('my.proxy.server')
```

Example - Basic Proxy Request with global agent dispatcher

```
import { setGlobalDispatcher, request, ProxyAgent } from 'undici'

const proxyAgent = new ProxyAgent('my.proxy.server')
setGlobalDispatcher(proxyAgent)

const { statusCode, body } = await request('http://localhost:3000/foo')

console.log('response received', statusCode) // response received 200

for await (const data of body) {
```

```
  console.log('data', data.toString('utf8')) // data foo
}
```

Example - Basic Proxy Request with local agent dispatcher

```
import { ProxyAgent, request } from 'undici'

const proxyAgent = new ProxyAgent('my.proxy.server')

const {
  statusCode,
  body
} = await request('http://localhost:3000/foo', { dispatcher: proxyAgent })

console.log('response received', statusCode) // response received 200

for await (const data of body) {
  console.log('data', data.toString('utf8')) // data foo
}
```

ProxyAgent.close()

Closes the proxy agent and waits for registered pools and clients to also close before resolving.

Returns: `Promise<void>`

Example - clean up after tests are complete

```
import { ProxyAgent, setGlobalDispatcher } from 'undici'

const proxyAgent = new ProxyAgent('my.proxy.server')
setGlobalDispatcher(proxyAgent)

await proxyAgent.close()
```

ProxyAgent.dispatch(options, handlers)

Implements `Agent.dispatch(options, handlers)`.

ProxyAgent.request(options[, callback])

See `Dispatcher.request(options [, callback])`.