

Coordinates:

```
<dependency>
  <groupId>org.apache.meecrowave</groupId>
  <artifactId>meecrowave-proxy</artifactId>
  <version>${meecrowave.version}</version>
  </dependency>
```

Simple proxy module using Meecrowave as backbone. It can be extended using CDI programming model and JAX-RS client.

Configuration

Name	Description
proxy-async-timeout	Asynchronous execution timeout.
proxy-configuration	The route file.
proxy-mapping	Where to bind the proxy (url pattern).
proxy-multipart	Is multipart explicit.
proxy-multipart-maxfilesizethreshold	Max file size threshold for multipart requests.
proxy-multipart-location	The multipart temporary folder.
proxy-multipart-maxfilesize	Max file size for multipart requests.
proxy-multipart-maxrequestsize	Max request size for multipart requests.
proxy-skip	Should default setup be ignored



you can use that servlet in a plain Servlet container (adding JAX-RS+JSON-B client). An integration example can be found in org.apache.meecrowave.proxy.servlet.meecrowave.ProxyServletSetup#accept.

Configuration File

Each route defines an execution context which means:

- 1. A way to match the incoming request (by method + prefix for now),
- 2. A way to forward the incoming request (which target server is called),
- 3. A way to execute the request isolated in a dedicated thread (how many threads are allocated to the route, which timeout to use, ...).

The routes file follows the following shape:

```
{
  "defaultRoute": { // optional
   // ... anything a route can get, it is used as default for plain "routes"
 },
  "routes": [
   {
      "id": "get-simple",
      "requestConfiguration": {
        "method": "GET",
        "prefix": "/prefix-to-match",
        "addedHeaders" : { "Authorization": "Value", ... },
        "skippedHeaders" : [ "Content-Length", ... ],
        "skippedCookies" : [ "Cookie", ... ],
      },
      "responseConfiguration": {
        "target": "http://....",
        "skippedHeaders" : [ "Content-Length", ... ],
        "skippedCookies" : [ "Cookie", ... ],
      },
      "clientConfiguration": {
        "executor": {
            "core": 8,
            "max": 512,
            "keepAlive": 60000,
            "shutdownTimeout": 1
        },
        "timeouts": {
            "connect": 30000,
            "read": 30000,
            "execution": 60000
        },
        "sslConfiguration": {
            "acceptAnyCertificate": false,
            "keystoreLocation": "...",
            "keystoreType": "...",
            "keystorePassword": "...",
            "truststoreType": "...",
            "verifiedHostnames": ["..."]
       }
      "extensions": { // optional, used for custom extensions and let the user enrich
the route configuration
      }
   },
   // ...
 "extensions": { // optional
 }
}
```



Extend

The default implementation uses CDIProxyServlet which triggers multiple events to let you extend the proxy implementation:

- 1. BeforeRequest and AfterResponse which are sent around the proxying,
- 2. OnRequest and OnResponse which enables you to replace the way the request is mapped to the proxied server and the way the response of the proxied server is mapped to the client.

Since meecrowave-proxy is a simple meecrowave module you can embed it and customize it as any CDI application.