



MIDDLEWARE

jwt

Kind: Function

VERSION
5

UPDATED
12 August 2026

DOCUMENT
reference-subsystem-src-middleware/function-jwt

CONTENTS

1	Signature	2	4	Usage	3
2	Parameters	2	5	AI Coding Instructions	3
3	Diagram	2			

Kind: Function

Source: `src/middleware/jwt/jwt.ts`

Part of: [Middleware](#)

JWT Auth Middleware for Hono.

`jwt` creates Hono middleware that reads a JWT from the request, verifies it with the configured options, and stores the decoded payload in the request context. Routes protected by this middleware can read the payload from `c.get('jwtPayload')`.

1 Signature

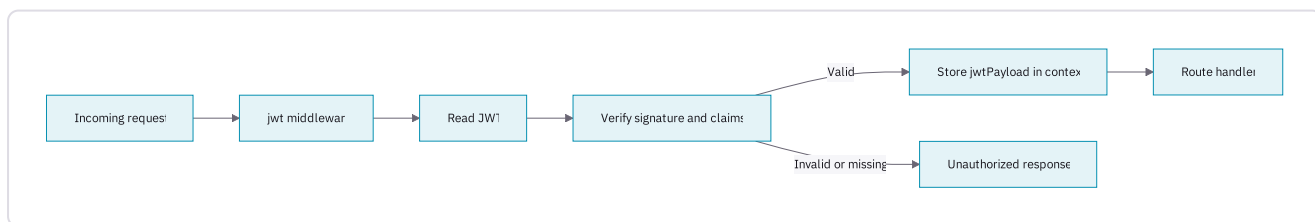
```
function jwt(options: { secret: SignatureKey cookie?: | string | { key: string; secret?: string | BufferSource; prefixOptions?: CookiePrefixOptions } alg: SignatureAlgorithm headerName?: string realm?: string verification?: VerifyOptions }): MiddlewareHandler
```

2 Parameters

Name	Type
<code>options</code>	<code>{ secret: SignatureKey cookie?:</code>

Returns: `MiddlewareHandler`

3 Diagram



4 Usage

```
import { Hono } from 'hono'
import { jwt } from 'hono/jwt'

const app = new Hono()

app.use(
  '/api/*',
  jwt({
    secret: 'my-secret-key',
  })
)

app.get('/api/profile', (c) => {
  const payload = c.get('jwtPayload')

  return c.json({
    userId: payload.sub,
  })
})

export default app
```

5 AI Coding Instructions

- Register `jwt` before route handlers that read `c.get('jwtPayload')`.
- Pass the same secret or public-key configuration used when signing tokens.
- Scope the middleware to protected route paths instead of applying it to public endpoints.
- Read claims from `jwtPayload` in handlers and validate application-specific fields before granting access.
- Return tokens through the expected authorization header format when calling protected routes.