except

August 18, 2026

except

Kind: Function

Source: `src/middleware/combine/index.ts`

Part of: Middleware

Create a composed middleware that runs all middleware except when the condition is met.

`except` creates a middleware handler that evaluates a condition before running the middleware passed to it. When the condition matches, it skips the composed middleware and continues to the next handler; otherwise, it runs the composed middleware.

Signature

```
function except(condition: string | Condition | (string | Condition)[], middleware: MiddlewareH
```

Parameters

Name	Type
`condition`	`string
`middleware`	`MiddlewareHandler[]`

Returns: `MiddlewareHandler`

Diagram

```
graph LR
  Request[Request] --> Condition{Condition matches?}
  Condition -->|Yes| Next[Next handler]
```

```
  Condition -->|No| Compose[Compose middleware]
  Compose --> Middleware[Middleware handlers]
  Middleware --> Next
```

Usage

```ts
import { Hono } from 'hono'
import { except } from 'hono/combine'
import { bearerAuth } from 'hono/bearer-auth'

const app = new Hono()

app.use(
  '/api/*',
  except(
    (c) => c.req.path === '/api/health',
    bearerAuth({ token: 'secret-token' })
  )
)

app.get('/api/health', (c) => c.text('ok'))

app.get('/api/profile', (c) => {
  return c.json({ id: 'user' })
})
```

AI Coding Instructions

- Return `true` from the condition when the middleware should be skipped.
- Keep condition checks limited to request data available on the context.
- Pass middleware handlers after the condition; they run only when the condition does not match.
- Ensure skipped requests can be handled by a later route or middleware handler.