RegExpRouter

August 18, 2026

RegExpRouter

Kind: Class

Source: `src/router/reg-exp-router/router.ts`

Part of: Router

`RegExpRouter` stores method and path registrations, then builds regular-expression matchers for route lookup. It is a router implementation used by the application to map incoming request paths to registered handlers.

Implements: `Router`

Methods

Method	Signature	Returns
`#insertPath`	`#insertPath(method: string, path: string)`	`void`
`add`	`add(method: string, path: string, handler: T)`	`void`
`buildAllMatchers`	`buildAllMatchers()`	`MatcherMap<T>`
`#buildMatcher`	`#buildMatcher(method: string)`	`Matcher<T>`

Properties

Property	Type
`name`	`string`
`#middleware`	`Record<string, Record<string, HandlerWithMetadata<T>[]>>`
`#routes`	`Record<string, Record<string, HandlerWithMetadata<T>[]>>`
`#tries`	`Record<string, Trie>`
`match`	`typeof match<Router<T>, T>`

Where it refuses work

- `RegExpRouter` stops the work with `Error` when `!middleware || !routes`.

When something fails

- `RegExpRouter` handles failure in 1 place: it lets it reach the caller in all 1.

Diagram

```
graph LR
  App[Application] --> Add[add()]
  Add --> Insert["#insertPath()"]
  Insert --> Routes[Route definitions]
  Routes --> BuildAll["buildAllMatchers()"]
  BuildAll --> Build["#buildMatcher()"]
  Build --> Matchers["MatcherMap<T>"]
```

Usage

```
import { Hono } from 'hono'
import { RegExpRouter } from 'hono/router/reg-exp-router'

const router = new RegExpRouter()
const app = new Hono({ router })

app.get('/users/:id', (c) => {
  return c.json({
    id: c.req.param('id'),
  })
})

app.post('/users', async (c) => {
  const body = await c.req.json()
  return c.json(body)
})

export default app
```

AI Coding Instructions

- Register routes through `add()` or the application routing API; do not call private methods such as `#insertPath()` directly.

- Keep route method and path patterns consistent, since matcher generation groups routes by HTTP method.
- Update `#buildMatcher()` when changing how path parameters or route patterns are translated into regular expressions.
- Preserve the `MatcherMap<T>` contract when modifying `buildAllMatchers()`, because request matching depends on its generated matchers.

Relationships

- IMPORTS → `MESSAGE_MATCHER_IS_ALREADY_BUILT`
- IMPORTS → `METHOD_NAME_ALL`
- IMPORTS → `UnsupportedPathError`
- IMPORTS → `checkOptionalParameter`
- IMPORTS → `match`
- IMPORTS → `emptyParam`
- IMPORTS → `PATH_ERROR`
- IMPORTS → `Trie`

Used by

1 reference from 1 file. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (1)

- `PreparedRegExpRouter` — `src/router/reg-exp-router/prepared-router.ts`:9