This is a tree code from https://github.com/julienschmidt/httprouter with an important fixes/improvements made inside Gin web framework.

See: * https://github.com/julienschmidt/httprouter/pull/329 * https://github.com/gin-gonic/gin/issues/2786

See also https://github.com/julienschmidt/httprouter/issues/235 – that's the reason why we can't use a custom branch patched with fixes.

Original LICENSE and copyright left unchanged here.

How channel pattern matching works

As stated in httprouter readme about route matching behavior:

> Only explicit matches: With other routers, like http.ServeMux, a requested URL path could match multiple patterns. Therefore they have some awkward pattern priority rules, like longest match or first registered, first matched. By design of this router, a request can only match exactly one or no route. As a result, there are also no unintended matches

This copy of the httprouter follows this semantics. But it also contains several improvements where original httprouter was too restrictive.

One possible way is to start with `tree_test.go` file for routing examples.

What's possible:

- Possible to define explicit patterns without any parameters: `stream/metrics/cpu`, `stream/metrics/mem`, `stream/metrics/cpu/spikes`
- Possible to have named parameter to match multiple channels: `stream/metrics/:metric`, `stream/metrics/:metric/spikes`
- Possible to have `stream/metrics/:metric` and `stream/metrics/cpu` defined together so a version without named parameter is preferred when matching
- Possible to use catch-all parameters like `stream/metrics/*rest` which will match multiple segments

What's impossible:

- Having 2 patterns like this: `stream/:scope/cpu` and `stream/metrics/:metric` since it will cause conflict in the underlying tree
- Having pattern like `stream/metrics/:metric` while you already have `stream/*rest` – again this is a conflict. So use catch-all parameters only in cases where you know for sure there won't be any sub-patterns in the future for which you need to override the behavior.