-Stream

HTTP

Content Menu

- < v2.4
- >= v2.4

< v2.4

v2.4beta

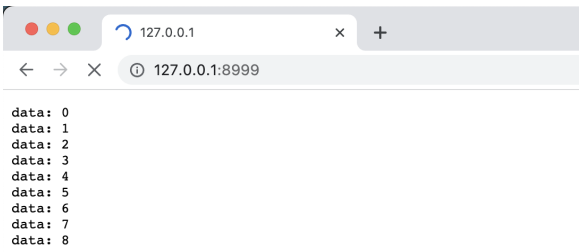
```
package main

import (
        "fmt"
        "net/http"
        "time"

        "github.com/gogf/gf/v2/frame/g"
        "github.com/gogf/gf/v2/net/ghttp"
)

func main() {
        s := g.Server()
        s.BindHandler("/", func(r *ghttp.Request) {
                rw := r.Response.RawWriter()
                flusher, ok := rw.(http.Flusher)
                if !ok {
                        http.Error(rw, "Streaming unsupported!", http.
StatusInternalServerError)
                        return
                }
                r.Response.Header().Set("Content-Type", "text/event-
stream")
                r.Response.Header().Set("Cache-Control", "no-cache")
                r.Response.Header().Set("Connection", "keep-alive")

                for i := 0; i < 100; i++ {
                        _, err := fmt.Fprintf(rw, "data: %d\n", i)
                        if err != nil {
                                panic(err)
                        }
                        flusher.Flush()
                        time.Sleep(time.Millisecond * 200)
                }
        })
        s.SetPort(8999)
        s.Run()
}
```

>= v2.4

v2.4beta

```
package main

import (
        "time"

        "github.com/gogf/gf/v2/frame/g"
        "github.com/gogf/gf/v2/net/ghttp"
)

func main() {
        s := g.Server()
        s.BindHandler("/", func(r *ghttp.Request) {
                r.Response.Header().Set("Content-Type", "text/event-
stream")
                r.Response.Header().Set("Cache-Control", "no-cache")
                r.Response.Header().Set("Connection", "keep-alive")

                for i := 0; i < 100; i++ {
                        r.Response.Writefln("data: %d", i)
                        r.Response.Flush()
                        time.Sleep(time.Millisecond * 200)
                }
        })
        s.SetPort(8999)
        s.Run()
}
```

http://127.0.0.1:8999/



- `Requestg.RequestFromCtx(ctx)`
- `r.ExitAll()`

Server-Sent Events SSE