**NetworkedMultiplayerENet**

1. **Mô tả**

Một triển khai PacketPeer sẽ được chuyển tới SceneTree.network\_peer sau khi được khởi tạo dưới dạng máy khách hoặc máy chủ. Các sự kiện sau đó có thể được xử lý bằng cách kết nối với các tín hiệu SceneTree .

Mục đích của ENet là cung cấp một lớp giao tiếp mạng tương đối mỏng, đơn giản và mạnh mẽ trên UDP (User Datagram Protocol).

ENet chỉ sử dụng UDP, không sử dụng TCP. Khi chuyển tiếp cổng máy chủ để làm cho máy chủ có thể truy cập được trên Internet công cộng, người dùng chỉ cần chuyển tiếp cổng máy chủ trong UDP. Có thể sử dụng lớp UPNP để cố gắng chuyển tiếp cổng máy chủ tự động khi khởi động máy chủ.

1. **Thuộc tính**

Graphical user interface, table

Description automatically generated

1. **Methods**

Graphical user interface, text, application, email

Description automatically generated

1. **Bảng kê**

COMPRESS\_NONE = 0 --- Không nén. Điều này sử dụng nhiều băng thông nhất, nhưng có mặt trái là yêu cầu ít tài nguyên CPU nhất. Tùy chọn này cũng có thể được sử dụng để gỡ lỗi mạng bằng các công cụ như Wireshark dễ dàng hơn.

COMPRESS\_RANGE\_CODER = 1 --- Mã hóa phạm vi tích hợp của ENet. Hoạt động tốt trên các gói nhỏ, nhưng không phải là thuật toán hiệu quả nhất trên các gói lớn hơn 4 KB.

COMPRESS\_FASTLZ = 2 --- Nén FastLZ . Tùy chọn này sử dụng ít tài nguyên CPU hơn so với COMPRESS\_ZLIB , với chi phí sử dụng nhiều băng thông hơn.

COMPRESS\_ZLIB = 3 --- Nén Zlib . Tùy chọn này sử dụng ít băng thông hơn so với COMPRESS\_FASTLZ , với chi phí sử dụng nhiều tài nguyên CPU hơn. Lưu ý rằng thuật toán này không hiệu quả đối với các gói nhỏ hơn 4 KB. Do đó, bạn nên sử dụng các thuật toán nén khác trong hầu hết các trường hợp.

COMPRESS\_ZSTD = 4 --- Nén tiêu chuẩn .

1. **Property Descriptions**

**Bool always\_ordered**

|  |  |
| --- | --- |
| Default | False |
| Setter | Set-always-ordered(value) |
| Getter | Is\_always\_ordered() |

Thực thi các gói có thứ tự khi sử dụng NetworkedMultiplayerPeer.TRANSFER\_MODE\_UNRELIABLE (do đó hoạt động tương tự như NetworkedMultiplayerPeer.TRANSFER\_MODE\_UNRELIABLE\_ORDERED ). Đây là cách duy nhất để sử dụng đặt hàng với hệ thống RPC.

**int channel\_count**

|  |  |
| --- | --- |
| Default | 3 |
| Setter | set\_channel\_count(value) |
| Getter | get\_channel\_count() |

Số lượng kênh sẽ được sử dụng bởi ENet. Các kênh được sử dụng để phân tách các loại dữ liệu khác nhau. Ví dụ, trong chế độ đáng tin cậy hoặc được đặt hàng, thứ tự phân phối gói được đảm bảo trên cơ sở mỗi kênh. Điều này được thực hiện để chống lại độ trễ và giảm các hạn chế đặt hàng đối với các gói. Trạng thái phân phối của một gói trong một kênh sẽ không làm ngưng trệ việc phân phối các gói khác trong một kênh khác.

**compression\_mode**

|  |  |
| --- | --- |
| Default | 1 |
| Setter | set\_compression\_mode(value) |
| Getter | get\_compression\_mode() |

Phương pháp nén được sử dụng cho các gói mạng. Chúng có sự đánh đổi khác nhau giữa tốc độ nén và băng thông, người dùng có thể cần phải kiểm tra xem cái nào hoạt động tốt nhất cho trường hợp sử dụng của mình nếu sử dụng tính năng nén.

Lưu ý: Thiết kế mạng của hầu hết các trò chơi liên quan đến việc gửi nhiều gói nhỏ thường xuyên (mỗi gói nhỏ hơn 4 KB). Vì vậy nên giữ thuật toán nén mặc định vì nó hoạt động tốt nhất trên các gói nhỏ này.

Lưu ý: chế độ nén phải được đặt thành cùng một giá trị trên cả máy chủ và tất cả các máy khách của nó. Máy khách sẽ không kết nối được nếu bộ nén\_mạch trên máy khách khác với bộ chế độ trên máy chủ.

**String dtls\_hostname**

|  |  |
| --- | --- |
| Default | “ “ |
| Setter | set\_dtls\_hostname(value) |
| Getter | get\_dtls\_hostname() |

Khi được đặt thành một chuỗi trống, addresstham số được truyền cho create\_client sẽ được sử dụng thay thế.

**Bool dtls\_verify**

|  |  |
| --- | --- |
| Default | true |
| Setter | set\_dtls\_verify\_enabled(value) |
| Getter | is\_dtls\_verify\_enabled() |

Bật hoặc tắt xác minh chứng chỉ khi use\_dtls true .

**bool server\_relay**

|  |  |
| --- | --- |
| Default | true |
| Setter | set\_server\_relay\_enabled(value |
| Getter | is\_server\_relay\_enabled() |

Bật hoặc tắt tính năng máy chủ thông báo cho khách hàng về kết nối / ngắt kết nối của các người chơi khác và chuyển tiếp các thông báo giữa chúng. Khi có tùy chọn này false, các máy khách sẽ không được tự động thông báo về các người chơi khác và sẽ không thể gửi cho họ các gói thông qua máy chủ.

**int transfer\_channel**

|  |  |
| --- | --- |
| Default | -1 |
| Setter | set\_transfer\_channel(value) |
| Getter | get\_transfer\_channel() |

Đặt kênh mặc định được sử dụng để truyền dữ liệu. Theo mặc định, giá trị này -1có nghĩa là ENet sẽ chỉ sử dụng 2 kênh: một cho các gói đáng tin cậy và một cho các gói không đáng tin cậy. Kênh đã 0 được bảo lưu và không thể sử dụng được. Đặt thành viên này thành bất kỳ giá trị nào giữa 0 và channel\_count (bị loại trừ) sẽ buộc ENet phải sử dụng kênh đó để gửi dữ liệu.**bool use\_dtls**

|  |  |
| --- | --- |
| Default | false |
| Setter | Set\_dtls\_enabled(value) |
| Getter | is\_dtls\_enabled() |

Khi được bật, máy khách hoặc máy chủ được tạo bởi máy ngang hàng này, sẽ sử dụng PacketPeerDTLS thay vì các ổ cắm UDP thô để giao tiếp với máy ngang hàng từ xa. Điều này sẽ làm cho giao tiếp được mã hóa bằng DTLS với chi phí sử dụng tài nguyên cao hơn và kích thước gói có khả năng lớn hơn.

1. **Method Descriptions**

* void close\_connection ( int wait\_usec=100 )

Đóng kết nối, bỏ qua nếu hiện không có kết nối nào được thiết lập. Nếu đây là một máy chủ, nó sẽ cố gắng thông báo cho tất cả các máy khách trước khi buộc ngắt kết nối chúng. Nếu đây là một máy khách, nó chỉ cần đóng kết nối với máy chủ.

* Error create\_client ( String address, int port, int in\_bandwidth=0, int out\_bandwidth=0, int client\_port=0 )

Tạo máy khách kết nối với máy chủ addressbằng cách sử dụng được chỉ định port. Địa chỉ đã cho cần phải là tên miền đủ điều kiện (ví dụ "www.example.com") hoặc địa chỉ IP ở định dạng IPv4 hoặc IPv6 (ví dụ "192.168.1.1"). Đây là port mà cổng mà máy chủ đang nghe. Các tham số in\_bandwidth và out\_bandwidth có thể được sử dụng để giới hạn băng thông vào và ra ở số byte cho trước trong một giây. Giá trị mặc định là 0 có nghĩa là băng thông không giới hạn.

* Error create\_server ( int port, int max\_clients=32, int in\_bandwidth=0, int out\_bandwidth=0)

Tạo máy chủ ghi các kết nối qua port. Port cần phải là cổng khả dụng, chưa sử dụng trong khoảng từ 0 đến 65535. Các cổng dưới 1024 là đặc quyền và có thể yêu cầu quyền cao hơn tùy thuộc vào nền tảng. Để thay đổi giao diện mà máy chủ ghi,ta sử dụng set\_bind\_ip . IP mặc định là ký tự đại diện "\*", sẽ ghi trên tất cả các giao diện có sẵn. max\_clients là số lượng người chơi tối đa được phép cùng một lúc, bất kỳ số nào lên đến 4095 đều có thể được sử dụng, mặc dù số lượng người chơi đồng thời có thể đạt được có thể thấp hơn nhiều và tùy thuộc vào ứng dụng.

* void disconnect\_peer ( int id, bool now=false )

Ngắt kết nối mạng ngang hàng đã cho. Nếu "now" được đặt thành true, kết nối sẽ bị đóng ngay lập tức mà không gửi thư đã xếp hàng đợi.

* int get\_last\_packet\_channel ( ) const

Trả về kênh của gói cuối cùng được tìm nạp qua PacketPeer.get\_packet.

* int get\_packet\_channel ( ) const

Trả về kênh của gói tiếp theo sẽ được truy xuất qua PacketPeer.get\_packet .

* String get\_peer\_address ( int id ) const

Trả về địa chỉ IP của ứng dụng ngang hàng đã cho.

* int get\_peer\_port ( int id ) const

Trả về cổng từ xa của ứng dụng ngang hàng đã cho.

* void set\_bind\_ip ( String ip )

IP được sử dụng khi tạo máy chủ. Điều này được đặt thành ký tự đại diện "\*"theo mặc định, liên kết với tất cả các giao diện có sẵn. IP đã cho cần phải ở định dạng địa chỉ IPv4 hoặc IPv6, ví dụ "192.168.1.1":.

* void set\_dtls\_key ( khóa CryptoKey )

Định cấu hình CryptoKey để sử dụng khi use\_dtlstrue. Gọi set\_dtls\_certificate để thiết lập X509Certificate

* void set\_peer\_timeout ( int id, int timeout\_limit, int timeout\_min, int timeout\_max

Đặt thông số thời gian chờ cho một ứng dụng ngang hàng. Các tham số thời gian chờ kiểm soát cách thức và thời điểm một máy ngang hàng sẽ hết thời gian chờ không xác nhận được lưu lượng truy cập đáng tin cậy. Giá trị thời gian chờ được biểu thị bằng mili giây.

Hệ timeout\_limit số nhân với một giá trị dựa trên thời gian trung bình của chuyến đi, sẽ xác định giới hạn thời gian chờ cho một gói tin đáng tin cậy. Khi đạt đến giới hạn đó, thời gian chờ sẽ được nhân đôi và ngang hàng sẽ bị ngắt kết nối nếu đã đạt đến giới hạn đó timeout\_min. Timeout\_max tham số xác định thời gian chờ cố định mà bất kỳ gói nào phải được xác nhận nếu không ngang hàng sẽ bị loại bỏ.