Link : “ http://docs2x.smartfoxserver.com/AdvancedTopics/advanced-mmo-api ”

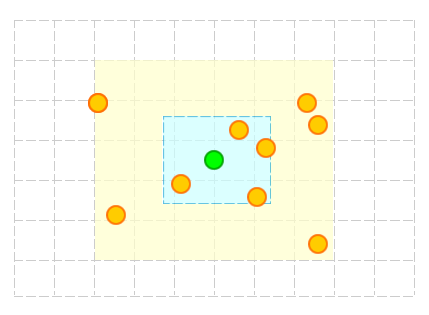
**» Sử dụng nâng cao API MMORoom và MMO**

Bài viết này yêu cầu bạn đã quen thuộc với các tính năng của MMORoom được giới thiệu trong **SmartFoxServer 2X phiên bản 2.8.** Ở đây chúng tôi mở rộng một số kỹ thuật nâng cao có thể được sử dụng từ phía máy chủ. Nếu bạn không quen thuộc với MMORooms và API MMO, vui lòng đảm bảo **[đọc Tổng quan về MMORoom tại đây](http://docs2x.smartfoxserver.com/AdvancedTopics/mmo-rooms)** .

**» Lĩnh vực quan tâm tùy chỉnh (AoI)**

Một khía cạnh nâng cao của API MMO phía máy chủ là khả năng sử dụng AoI tùy chỉnh nhỏ hơn hoặc bằng với AoI được định cấu hình trong MMORoom. Tính năng này cho phép hạn chế khu vực bản đồ trong đó một sự kiện cụ thể được phát sóng.

Chúng ta hãy xem sơ đồ bản đồ 2D đơn giản này:



Vòng tròn màu xanh lá cây là người chơi của chúng tôi và khu vực màu cam đại diện cho AoI, vì vậy tất cả các sự kiện do người chơi tạo ra sẽ được truyền đến các khách hàng khác nằm trong phạm vi gần đó (vòng tròn màu cam). Từ Mã mở rộng, chúng tôi cũng có thể phát các sự kiện và thông báo ảnh hưởng đến một khu vực nhỏ hơn: trong trường hợp này, hình chữ nhật màu lục lam biểu thị một AoI nhỏ hơn có thể được xác định trong thời gian chạy.

**" Mục đích sử dụng**

Các ứng dụng của AoI tùy chỉnh có thể khác nhau, chẳng hạn như chỉ gửi tin nhắn công khai cho Người dùng ở trong phạm vi ngắn hơn hoặc kích hoạt các sự kiện ở các khu vực trong AoI chính, ví dụ: một vụ nổ sẽ chỉ ảnh hưởng đến những người chơi gần nguồn của AoI hơn. biến cố.

**SFSMMOApi** phía máy chủ (trong gói *com.smartfoxserver.v2.api* ) hiển thị một số phương thức hữu ích để thực hiện việc này:

* **sendPublicMessage** (Phòng, Người gửi người dùng, Thông báo chuỗi, ISFSObject params, Vec3D aoi)
* **sendObjectMessage** (Phòng, Người gửi người dùng, ISFSObject tin nhắn, Vec3D aoi)

Thay vì lấy danh sách người nhận làm "API thông thường", ở đây chúng tôi phải cung cấp Vec3D đại diện cho AoI xung quanh người gửi. Như chúng tôi đã đề cập, AoI tùy chỉnh **phải <=** AoI mặc định.

**H: Tại sao chúng tôi không thể xác định AoI tùy chỉnh ở bất kỳ kích thước nào, ví dụ: lớn hơn AoI mặc định của Phòng**

MMORoom sử dụng cấu trúc dữ liệu nội bộ được tối ưu hóa cao để duy trì tất cả các hoạt động tra cứu rất nhanh và hiệu quả, ngay cả khi có hàng nghìn Người dùng và MMOItem tương tác cùng lúc. Để thực hiện điều này, cấu trúc dữ liệu sử dụng AoI mặc định làm đơn vị lưu trữ chính và chỉ có thể cho phép tìm kiếm trong các đơn vị đó.

Nếu cần chọn một nhóm Người dùng phức tạp hơn dựa trên nhiều khu vực của MMORoom, bạn vẫn có thể thực hiện thủ công, chúng tôi sẽ trình bày chính xác cách thực hiện trong một vài đoạn tiếp theo.

**» Mở rộng lớp MMOItem**

Chúng ta đã thấy trong bài viết Tổng quan về MMORoom về cách **MMOItems** có thể được sử dụng để đại diện cho tất cả các loại đối tượng tương tác: trình kích hoạt, phần thưởng, đường đạn, v.v... Bây giờ chúng ta sẽ trình bày một ví dụ thực tế bằng cách xác định hai thực thể cụ thể mà chúng ta sẽ sử dụng trong bài viết của mình. trò chơi giả định: công tắc và cửa ra vào.

Chúng tôi muốn MMORoom của mình có một số lối đi bí mật có thể mở được từ xa thông qua một số công tắc nằm trên bản đồ. Khi công tắc được kích hoạt, một cánh cửa bí mật gần đó sẽ mở ra và chúng tôi sẽ kích hoạt sự kiện này cho khách hàng để hiển thị hoạt ảnh về việc mở (hoặc đóng) cửa.

Để mô tả các đối tượng này ở phía máy chủ, chúng tôi sẽ mở rộng lớp MMOItem và thêm các bit thông tin bổ sung được yêu cầu:

Java

[**?**](http://docs2x.smartfoxserver.com/AdvancedTopics/advanced-mmo-api)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | **public** **class** SwitchItem **extends** MMOItem  {  **private** **boolean** state;    **public** DoorItem(**boolean** state)      {  **super**();  **this**.state = state;      }    **public** **boolean** getState()      {  **return** state;      }  } |

[**?**](http://docs2x.smartfoxserver.com/AdvancedTopics/advanced-mmo-api)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | **public** **class** DoorItem **extends** MMOItem  {  **private** Vec3D location;  **private** SwitchItem relatedSwitch;    **public** DoorItem(Vec3D location, SwitchItem relatedSwitch)      {  **super**();  **this**.location = location;  **this**.relatedSwitch = relatedSwitch;      }    **public** Vec3D getLocation()      {  **return** location;      }    **public** **boolean** isOpen()      {  **return** relatedSwitch.getState();      }  } |

**SwitchItem** của chúng tôi đã thêm một boolean bổ sung biểu thị trạng thái của nó, trong khi **DoorItem** có Vec3D biểu thị vị trí của nó trên bản đồ và tham chiếu đến SwitchItem liên quan mở/đóng lối đi.

**» Tin nhắn mở rộng**

Bây giờ chúng tôi đã xác định hai mục trò chơi, hãy xem cách chúng tôi có thể kích hoạt một sự kiện từ Mã mở rộng chỉ cho những người dùng nằm trong AoI của cửa.

Java

[**?**](http://docs2x.smartfoxserver.com/AdvancedTopics/advanced-mmo-api)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | **public** **void** handleSecretDoorStateChange(DoorItem secretPassage)  {      SFSObject msg = **new** SFSObject();        // we add the Id of the passage to be opened/closed      msg.putInt("itemId", secretPassage.getId());        // we add the status of the secret passage      msg.putBool("open", secretPassage.isOpen());        // Obtain the list of Users within the item's AOI      List<User> recipients = getParentRoom().getProximityList(secretPassage.getLocation());        // Send event to selected clients      send("handleDoor", msg, recipients);  } |

"Mẹo" ở đây là sử dụng các phương thức của MMORoom để lấy thứ chúng ta cần. Trong trường hợp này, chúng tôi đang yêu cầu một danh sách Người dùng nằm trong AoI mặc định được cung cấp các tọa độ nhất định. Đối tượng MMORoom cung cấp một số phương thức hữu ích cho các tác vụ này:

* **public List<User> getProximityList(User target)**   
  lấy danh sách Người dùng trong AoI mặc định xung quanh khách hàng mục tiêu được cung cấp
* **public List<User> getProximityList(User target, Vec3D aoi)**   
  lấy danh sách Người dùng trong AOI tùy chỉnh xung quanh khách hàng mục tiêu được cung cấp
* **public List<User> getProximityList(Vec3D position)**   
  lấy danh sách Người dùng trong AoI mặc định xung quanh tọa độ đích được cung cấp
* **public List<User> getProximityList(Vec3D position, Vec3D aoi)**   
  lấy danh sách Người dùng trong AoI tùy chỉnh xung quanh tọa độ đích được cung cấp

Bây giờ, giả sử trò chơi của chúng ta cũng hỗ trợ cửa bẫy sàn. Người chơi phát hiện ra công tắc ẩn dẫn đến cửa bẫy có thể đợi cho đến khi một hoặc nhiều Người dùng không cẩn thận bước vào đúng vị trí, sau đó kích hoạt cửa để bắt họ.

Trong trường hợp này, sự kiện chỉ nên được truyền tới Người dùng ở khu vực nhỏ hơn của Phòng: chỉ những người có vị trí nằm trong chu vi cửa bẫy. Chúng ta có thể hoàn thành nhiệm vụ này bằng cách sử dụng AoI tùy chỉnh và gọi phương thức cuối cùng trong danh sách trước đó.

**» Thủ thuật nâng cao**

Như chúng tôi đã đề cập, chúng tôi có thể thực hiện tìm kiếm bên trong Phòng bằng cách sử dụng AoI tùy chỉnh miễn là kích thước của nó nhỏ hơn hoặc bằng AoI mặc định. Điều này sẽ bao gồm 95% các trường hợp, nhưng còn một số trường hợp còn lại thì sao?

Vì lợi ích của ví dụ này, chúng tôi sẽ có một người chơi tạo ra một sự kiện mà nhiều Người dùng hơn sẽ nhận được bên ngoài AoI mặc định. Giả sử anh hùng RPG của chúng ta vừa thi triển một câu thần chú động đất, hiệu ứng của nó phải lan truyền đến một khu vực rộng lớn hơn nhiều so với AoI mặc định.

Để hoàn thành tác vụ này, chúng tôi chỉ cần thực hiện nhiều tra cứu Người dùng ở các khu vực lân cận (AoI), tổng hợp các kết quả trong một Danh sách duy nhất và cuối cùng sử dụng nó với phương thức Tiện ích mở rộng **send(...) thông thường** . Nói cách khác, chúng ta có thể sử dụng **AoI mặc định làm đơn vị chia nhỏ** bản đồ ảo và thực hiện nhiều tìm kiếm bằng cách di chuyển tọa độ mục tiêu theo một hoặc nhiều đơn vị.

Theo mặc định, MMORoom sẽ chỉ tìm kiếm trong giới hạn 1 đơn vị, nhưng từ mã máy chủ, chúng tôi có thể tự do tổng hợp nhiều tìm kiếm và thậm chí chọn những người chơi cụ thể từ danh sách người dùng đầy đủ của MMORoom, do đó cho phép tính linh hoạt cao để lọc người nhận của một đơn vị cụ thể tin nhắn hoặc sự kiện máy chủ.