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

**» API trò chơi**

**SmartFoxServer 2X** cung cấp một bộ API máy khách và máy chủ mới được thiết kế đặc biệt để tạo và quản lý trò chơi, bao gồm các trò chơi công khai và riêng tư, lời mời tham gia trò chơi, ghép Người dùng và Phòng, v.v. **API trò chơi** mới dựa trên ba khối xây dựng cơ bản:

* [**Biểu thức đối sánh**](http://docs2x.smartfoxserver.com/AdvancedTopics/game-api#matchExp) : chúng cho phép tạo tiêu chí tìm kiếm theo cách rất tự nhiên, sau đó có thể được chuyển đến MatchingEngine để thực hiện bất kỳ loại truy vấn nào trên Phòng và Người dùng
* [**Lời mời**](http://docs2x.smartfoxserver.com/AdvancedTopics/game-api#invApi) : đây là hệ thống Lời mời chung cho phép bạn quản lý nhiều lời mời tham gia trò chơi riêng tư, nhưng cũng có thể được sử dụng trong nhiều hoạt động khác
* [**Lớp SFSGame**](http://docs2x.smartfoxserver.com/AdvancedTopics/game-api#sfsGame) : SFSGame là loại Phòng mới được thiết kế đặc biệt để quản lý trò chơi và cung cấp các dịch vụ độc đáo để quản lý trò chơi nâng cao

Ba thành phần này rất linh hoạt và có thể được sử dụng riêng lẻ cho một nhiệm vụ cụ thể hoặc được phối hợp với nhau để tạo ra các hành vi máy chủ nâng cao, như bạn sẽ tìm hiểu sau vài giây.

**" Tổng quan**

Lớp SmartFoxServer hiển thị một phương thức getAPIManager ( ), từ đó cho phép bạn truy cập tất cả các tập hợp con API khác nhau trong khung SFS2X. APIManager **cung** cấp:

* **getSFSApi ( )** : trả về một tham chiếu đến API SFS2X cơ bản
* **getBuddyApi ( )** : trả về một tham chiếu đến tập hợp con API Danh sách bạn bè
* **getGameApi ( )** : trả về một tham chiếu đến tập hợp con API của trò chơi

Bây giờ chúng ta sẽ tập trung vào **Game API** .

**» Biểu thức đối sánh**

Biểu thức khớp được xây dựng giống như điều kiện "nếu" trong bất kỳ ngôn ngữ lập trình phổ biến nào: chúng hoạt động giống như các truy vấn trong cơ sở dữ liệu. Đây là một ví dụ phía máy chủ để giúp bạn bắt đầu, nhưng điều này cũng tương tự ở phía máy khách:

Java

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

|  |  |
| --- | --- |
| 1 | MatchExpression exp = **new** MatchExpression('rank', NumberMatch.GREATER\_THAN, 5)  .and('country', StringMatch.EQUALS, 'Italy'); |

Các biểu thức được tạo thành từ ba yếu tố:

* tên biến
* Toán tử đối sánh
* Giá trị

Ngoài ra , bất kỳ số lượng biểu thức nào cũng có thể được liên kết với nhau bằng toán tử logic AND/OR, giống như trong mã thông thường. Trong ví dụ trên, chúng tôi đã tạo một biểu thức sẽ kiểm tra giá trị **xếp hạng** > 5 và giá trị **quốc gia** == "Ý".

Nhưng ... tập hợp các điều kiện này được áp dụng cho đâu? Thông thường, các biểu thức được sử dụng để khớp **Người dùng** (thông qua Biến người dùng của họ) và **Phòng** (thông qua Biến phòng của họ). **API SFS2X** tiêu chuẩn cung cấp hai phương thức hữu ích được gọi là **findUsers ( )** và **findRooms ()** trong đó bạn có thể thực thi bất kỳ Biểu thức đối sánh nào và lấy tập hợp Người dùng/Phòng đã lọc.

Một ví dụ nhỏ sẽ làm rõ điều này hơn:

Java

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

|  |  |
| --- | --- |
| 1 | List<User> matchingUsers = sfsApi.findUsers(zone.getUserList(), exp, 50); |

Đối số đầu tiên cung cấp danh sách các đối tượng Người dùng để tìm kiếm, trong trường hợp này là tất cả Người dùng bên trong Vùng hiện tại. Tham số thứ hai là Biểu thức khớp mà chúng ta vừa tạo ở đầu phần này và số cuối cùng (50) là giới hạn tùy chọn cho số lượng phần tử được trả về. Trong trường hợp này, chúng tôi muốn tìm kiếm không quá 50 phần tử phù hợp (nếu chúng tôi chuyển 0, tìm kiếm sẽ trả về tất cả Người dùng phù hợp).

Các tùy chọn tìm kiếm không chỉ giới hạn ở tên Biến người dùng/Phòng. Trên thực tế , công cụ Kết hợp cung cấp thêm hai lớp RoomProperties và UserProperties , nơi bạn có thể truy cập nhiều thuộc tính cụ thể của các lớp Phòng và Người dùng.

Đây là một ví dụ về việc so khớp các thuộc tính và Biến phòng cụ thể:

Java

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

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | // Prepare match expression  MatchExpression exp = **new** MatchExpression(RoomProperties.IS\_GAME, BoolMatch.EQUALS, **true**)                                       .and(RoomProperties.HAS\_FREE\_PLAYER\_SLOTS,  BoolMatch.EQUALS, **true**)                                       .and("isGameStarted", BoolMatch.EQUALS, **false**);    // Search Rooms  List<Rooms> joinableRooms = sfsApi.findRooms(zone.getRoomListFromGroup("chess"), exp, 0); |

Đoạn mã trên sẽ khớp với tất cả các Phòng trò chơi có ít nhất một vị trí người chơi miễn phí và trong đó biến *isStarted* được đặt thành false. Ngoài ra , việc tìm kiếm sẽ chỉ được thực hiện trong Nhóm Phòng có tên là "cờ vua".

**" Các tính năng tiên tiến**

Biểu thức so khớp cung cấp các khả năng nâng cao để tìm kiếm thông qua các cấu trúc dữ liệu lồng nhau, chẳng hạn như **SFSObject** và **SFSArray** . Điều này được thực hiện thông qua một biểu thức **cú pháp dấu chấm rất đơn giản** . Đây là một ví dụ về cách nó hoạt động:

Java

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

|  |  |
| --- | --- |
| 1 | MatchExpression exp = **new** MatchExpression("europe.italy.capital", StringMatch.EQUALS, "Rome"); |

The above example goes down deep into an SFSObject called *europe*, taking the *italy* object (another SFSObject) and finally reading its String field *capital* and matching it with another String. Here is one more examples using SFSObject and SFSArray:

Java

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

|  |  |
| --- | --- |
| 1 | MatchExpression exp = **new** MatchExpression("europe.italy.majorCities.3.name",  StringMatch.EQUALS, "Milan"); |

Từ đối tượng *italy* chúng ta có được một *majorCities* SFSArray và chúng tôi lấy mục thứ ba trong đó ( biểu thức **.3** có nghĩa là 'hãy đưa cho tôi phần tử tại chỉ mục == 3'). Mục này lại là một SFSObject có thuộc tính *tên* cuối cùng chúng ta so sánh với một Chuỗi.

Sức mạnh của Match Expression không kết thúc ở đây. Bạn có thể chạy nhiều lượt khớp nếu bạn cần thực hiện các tìm kiếm phức tạp. Ví dụ: bạn có thể chạy đối sánh đầu tiên và nhận danh sách các Phòng đã lọc, sau đó sử dụng danh sách đó để áp dụng một biểu thức khác nhằm tinh chỉnh thêm tìm kiếm của mình, v.v.   
Ngoài ra, bạn sẽ tìm hiểu thêm cách sử dụng thú vị của Biểu thức đối sánh kết hợp với lớp SFSGame sau trong chính bài viết này.

Để biết thêm chi tiết về Biểu thức khớp, chúng tôi khuyên bạn nên tham khảo javadoc , trong gói **com.smartfoxserver .v2.entities.match** .

**» Lời mời**

Hệ thống lời mời có trong SmartFoxServer 2X cung cấp một khung chung để gửi lời mời tới một hoặc nhiều người dùng được kết nối hơn và dễ dàng quản lý phản hồi của họ cũng như thời hạn của lời mời. Lời mời có thể được sử dụng để thách thức người chơi trong trò chơi, mời bạn bè đến chỗ của người dùng, xin quyền cho các tác vụ cụ thể như thêm người dùng được mời vào Danh sách bạn bè, v.v.

Tạo lời mời rất đơn giản, vì cần có bốn tham số cơ bản:

* **mời** : Người dùng bắt đầu lời mời
* **mời** : Người dùng/Người chơi/ Bạn thân được mời
* **Thời gian hết hạn** : số giây được phép để người được mời trả lời
* **Tham số tùy chỉnh** : một SFSObject chung bao gồm các tham số lời mời cụ thể (tin nhắn, hình ảnh, chi tiết trò chơi, v.v. )

Người dùng được mời sẽ nhận được lời mời và có thể trả lời trong khoảng thời gian quy định. Một mã CHẤP NHẬN hoặc TỪ CHỐI đơn giản là tất cả những gì cần thiết để trả lời cùng với một SFSObject tùy chọn với các tham số phản hồi, nếu cần. Trong trường hợp không có phản hồi nào được gửi đến máy chủ trong số giây dự kiến, lời mời sẽ được coi là "từ chối".

Java

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

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42 | **public** **class** GameInvitation **extends** SFSExtension  {  **private** ISFSGameApi gameAPI;        @Override  **public** **void** init()      {          gameAPI = SmartFoxServer.getInstance().getAPIManager().getGameApi();      }    **public** **void** sendInvitation()      {          // Prepare Invitation object          User inviter = getParentZone().getUserByName("Fozzie");          User invitee = getParentZone().getUserByName("Gonzo");            // We set the timeout for a reply to 50 seconds          Invitation invitation = **new** SFSInvitation(inviter, invitee, 50);            // Send the invitation          gameAPI.sendInvitation(invitation, **new** InvitationCallback()          {              @Override  **public** **void** onRefused(Invitation invObj, ISFSObject params)              {                  // Handle the refused invitation...              }                @Override  **public** **void** onExpired(Invitation invObj)              {                  // Handle the expired invitation...              }                @Override  **public** **void** onAccepted(Invitation invObj, ISFSObject params)              {                  // Handle the accepted invitation...              }          });      }  } |
| 1  2  3  4  5  6  7  số 8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42 | **công cộng** **lớp** Lời mời trò chơi **mở rộng** Tiện ích mở rộng SFFS  {  **riêng** ISFSGameApi trò chơiAPI ;        @Ghi đè  **công cộng** **khoảng trống** ban đầu ( )      {          gameAPI = SmartFoxServer.getInstance ( ). getAPIManager (). getGameApi ();      }    **công cộng** **khoảng trống** gửi lời mời ( )      {          // Chuẩn bị đối tượng Lời mời          Người mời người dùng = getParentZone ( ). getUserByName ( "Fozzie" );          Người dùng được mời = getParentZone ( ). getUserByName ( "Gonzo" );            // Chúng tôi đặt thời gian chờ trả lời là 50 giây          Lời mời mời = **mới** SFSInvitation ( người mời, người được mời, 50 );            // Gửi lời mời          gameAPI.sendInvitation (lời mời, **mới** Lời mờiGọi lại ( )          {              @Ghi đè  **công cộng** **khoảng trống** onRefused ( Lời mời invObj , thông số ISFSObject )              {                  // Xử lý lời mời bị từ chối...              }                @Ghi đè  **công cộng** **khoảng trống** onExpired ( Lời mời invObj )              {                  // Xử lý lời mời đã hết hạn...              }                @Ghi đè  **công cộng** **khoảng trống** onAccepted ( Lời mời invObj , thông số ISFSObject )              {                  // Xử lý lời mời được chấp nhận...              }          });      }  } |

Để biết thêm chi tiết về Lời mời, hãy kiểm tra javadoc , cụ thể là lớp **SFSGameAPI và gói com.smartfoxserver .v2.entities.invitation** . Thông tin bổ sung cũng có sẵn trong tài liệu API phía máy khách .

**» SFSTrò chơi**

Lớp **SFSGame** mở rộng các khả năng bình thường của **Phòng** , thêm khả năng đặt Trò chơi ở chế độ công khai hoặc riêng tư và cung cấp danh sách những người được mời mà hệ thống sẽ mời trong trò chơi. Ngoài ra hệ thống sẽ có thể mời thêm người nếu số lượng người chơi không đủ để bắt đầu trò chơi.

Mỗi trò chơi có thể được định cấu hình để khớp với các loại Người dùng cụ thể bằng cách cung cấp **Biểu thức khớp** . Biểu thức chứa các tiêu chí được kiểm tra đối với từng Người dùng muốn tham gia trò chơi và cung cấp ý nghĩa để lọc người chơi.

Hãy xem một ví dụ: người dùng Kermit có hai Biến người dùng sau được đặt:

* Xếp hạng: 10
* Điểm tốt nhất : 2500

Anh ấy muốn chơi, vì vậy anh ấy chọn một SFSGame công khai và cố gắng tham gia. Rất tiếc , biểu thức SFSGame được đặt như sau: **(Xếp hạng > 10) HOẶC ( Điểm tốt nhất > 3000)** .   
Mọi nỗ lực tham gia Trò chơi sẽ bị từ chối vì người chơi không phù hợp với tiêu chí của SFSGame .

Tạo Room kiểu SFSGame tương tự như tạo Room bình thường. Về phía máy khách, bạn sẽ tìm thấy lớp CreateRoomRequest và lớp CreateSFSGameRequest , cả hai đều lấy một đối tượng *cài đặt* : cụ thể RoomSettings và SFSGameSettings .

Sau đây là tổng quan nhanh về các thông số bổ sung mà SFSGame có thể sử dụng, so với Phòng *thông thường* :

* **isGamePublic** : một trò chơi công khai có thể được tham gia bởi bất kỳ người chơi nào có biến khớp với Biểu thức so khớp của trình phát SFSGame . Nếu không có biểu thức nào được sử dụng, bất kỳ Người dùng nào cũng có thể tham gia trò chơi. Thay vào đó, các trò chơi riêng tư dựa trên lời mời do người tạo SFSGame cung cấp (xem thuộc tính *được mờiNgười chơi* bên dưới) nên chúng thường không cần chỉ định Biểu thức so khớp.
* **minPlayersToStartGame** : số lượng người chơi tối thiểu để bắt đầu trò chơi. Nếu trò chơi đang chạy và số lượng người chơi vượt quá giới hạn này thì trò chơi sẽ bị dừng (xem thuộc tính notifyGameStartedViaRoomVariable bên dưới).
* **được mờiNgười chơi** : (chỉ dành cho trò chơi riêng tư) danh sách người chơi được mời trong SFSGame . Mỗi người chơi sẽ nhận được một sự kiện lời mời và sẽ có thể trả lời trong một khoảng thời gian nhất định (xem thuộc tính của *lời mờiExpiryTime* bên dưới).
* **searchableRooms** : (chỉ các trò chơi riêng tư) danh sách các Phòng nơi API trò chơi có thể tìm kiếm thêm người chơi để mời. API sẽ tìm kiếm nhiều người chơi hơn nếu số người được mời nhỏ hơn giá trị được đặt trong thuộc tính *minPlayersToStartGame* . Bằng cách này, bạn có thể thêm bạn bè của mình vào trò chơi và để hệ thống tìm thêm người chơi để bắt đầu trò chơi. Cơ chế này sẽ chỉ hoạt động khi trò chơi được bắt đầu lần đầu tiên, không phải lúc nào số lượng người dùng xuống dưới giá trị tối thiểu.   
  Nếu cần tìm kiếm và mời thêm người chơi mỗi khi trận đấu dừng bạn có thể dễ dàng thực hiện thông qua phương thức SFSApi.findUsers ().
* **leftLastJoinedRoom** : tự động xóa người chơi khỏi Phòng trước đó của họ sau khi tham gia thành công trong trường hợp họ nhận được lời mời và chấp nhận.
* **playerMatchExpression** : một biểu thức để khớp với những người chơi sẵn sàng chơi trò chơi; theo mặc định , không có biểu thức nào được sử dụng.
* **spectatorMatchExpression** : biểu thức khớp với khán giả sẵn sàng xem trận đấu; theo mặc định , không có biểu thức nào được sử dụng.
* **lời mờiExpiryTime** : khoảng thời gian cho phép người chơi được mời chấp nhận/từ chối.
* **lời mờiParams** : thông số tùy chỉnh tùy chọn sẽ được gửi cùng với lời mời. Chúng có thể cung cấp thông tin chi tiết về người mời, trò chơi, thông báo lời mời, v.v.
* **notifyGameStartedViaRoomVariable** : tự động cập nhật Biến phòng dành riêng để báo hiệu rằng trò chơi đã bắt đầu/dừng. Biến Phòng sử dụng **cài** đặt chung để phát ra bên ngoài Phòng. Điều này có thể được sử dụng ở phía máy khách để hiển thị trạng thái trò chơi trong danh sách các trò chơi có sẵn.   
  Biến phòng dành riêng được tìm thấy trong lớp *com.smartfoxserver .v2.entities.variables.ReserveedRoomVariables* .

Cuối cùng , điều quan trọng cần lưu ý là lớp **SFSGame mở rộng SFSRoom** và có thể được coi như bất kỳ Phòng nào khác trong hệ thống. Các đối tượng **SFSGame** có thể được sử dụng và chuyển qua bất kỳ phương thức hoặc chức năng nào hoạt động với giao diện **Phòng cơ sở** (điều này hợp lệ cho cả phía máy chủ và máy khách).

**» Tham gia game nhanh**

Một tính năng khác được cung cấp bởi **Game API** (cả phía máy khách và máy chủ) là yêu cầu *QuickJoinGame* . Bằng cách cung cấp Biểu thức khớp và danh sách Phòng (thuộc loại SFSGame ) hoặc Nhóm phòng, hệ thống có thể tìm kiếm Phòng phù hợp và dịch chuyển tức thời người chơi trong hành động trò chơi.

Như thường lệ, một ví dụ sẽ làm rõ khái niệm:

C# JS AS3

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

Mã máy khách ở trên cố gắng nhanh chóng tham gia cùng người dùng trong bất kỳ trò chơi nào từ nhóm phòng *poker* hoặc *blackJack trong đó biến* **xếp hạng** trò chơi lớn hơn 3 và nhỏ hơn 8. Ở hậu trường, hệ thống cũng sẽ đảm bảo rằng các Phòng thuộc loại SFSGame và rằng có ít nhất một vị trí người chơi có sẵn. Nếu không tìm thấy Phòng phù hợp với các tiêu chí này, lỗi tham gia sẽ được kích hoạt lại cho người dùng.

**QUAN TRỌNG**   
Tính năng **QuickJoinGame** chỉ hoạt động với các Phòng thuộc loại **SFSGame** , hỗ trợ Biểu thức khớp; tất cả các loại Phòng khác sẽ bị bỏ qua.