**EVENT LISTENERS TRONG PLUGINS**

* Tạo sự kiện và bắt sự kiện khi tương tác về thứ gì đó hay xảy ra một tình huống nó sẽ bắt sự kiện.
* Tương tự ở bài trước, tạo một project như thường và có 2 phương thức onEnable() và onDisable()
* Có 2 trường hợp có thể tạo sự kiện cho plugins:
  + Tại trong class chính (class mà có phương thức onEnable())
  + Ở một class khác và class main và phải tạo cho class main nhận diện được hoạt động của sự kiện.
* Đối với trường hợp tạo sự kiện trong class chính:

|  |
| --- |
| Public final class RunPlugin extends JavaPlugin implements Listener {  @Override  Public void onEnable() {  This.getServer().getPluginManager().registerEvent([listener], [plugin]);  }  @EventHandler  Public void onPlayerJoin(PlayerJoinEvent event) {  //Code sẽ thực thi khi mà bắt được sự kiện  }  } |

* Đối với ví dụ trên nó sẽ tạo một sự kiện là “PlayerJoinEvent”, đây là sự kiện khi mà người chơi join vào server nó sẽ bắt.
* Khi tạo sự kiện xong cho dù là cái phương thức đó ở class nào di chăng nữa thì ngay cả phần onEnable() (nơi mà source code sẽ bắt đầu hoạt động) đều phải bắt sự kiện (gọi để nhận diện sự kiện) thông qua phương thức registerEvent().
* Đối với phương thức registerEvent(), truyền bên trong 2 phương thức:
  + [listener]: là một class được cho là dùng để bắt sự kiện (vì ví dụ trên class RunPlugin là class để bắt sự kiện do class này implements Listener nên trong trường hợp này là truyền vào là **this**)
  + [plugin]: là class để khởi động chạy plugins hay nói cách khác giống như class có chứa hàm main y như lập trình Java bình thường.
* Ở phương thức onPlayerJoin() truyền bên trong là một class dùng để chứa phương thức sự kiện đó còn phương thức onPlayerJoin() là kiểu sự kiện để khi xảy ra tình huống gì nó sẽ bắt sự kiện.
* Đối với trường hợp tạo sự kiện ở bên ngoài class chính:
  + Tương tự như tạo sự kiện ở trong class chính, tiến hành implements Listener và tạo phương thức sự kiện.
  + Thông thường phương thức theo cú pháp:

|  |
| --- |
| @EventHandler  Public void [kiểu sự kiện] ([class chứa kiểu sự kiện] e){  //Code bên trong hoạt động khi bắt được sự kiện.  } |

* + Quay lại class chính, đăng ký sự kiện theo phương thức registerEvent()

|  |
| --- |
| This.getServer().getPluginManager().registerEvent([class chứa sự kiện], [class chứ plugin để khởi động]); |

* Ở một số trường hợp chúng ta có thể tiến hành hủy sự kiện, chẳng hạn bắt sự kiện khi người dung phá một blocks nào đó trong thế giới nếu chúng ta hủy sự kiện coi như là hủy luôn trường hợp người dung không thể phá được block coi như người chơi sẽ không có khả năng phá được block.
* Ví dụ, đây là hủy sự kiện cho việc người chơi lấy kéo cưa cắt lông cừu:

|  |
| --- |
| Public class ShearSheepListener implements Listener {  @EventHandler  Public onShearSheep(PlayerShearEntityEvent e) {  Player player = e.getPlayer();  Entity entity = e.getEntity();    If(entity.getType() == Entity.Sheep) {  e.setCancelled(true);  Player.sendMessage(“This is a sheep. You can’t do that.”);  } else {  Player.sendMessage(“This is not a sheep.”);  }  }  } |

* Và tiến hành nhận diện sự kiện thông qua phương thức registerEvent(new ShearSheepListener(), this) ở trong class chính.