

# Command Registry

This page is unfinished and only provides basic information

## Example:

The following code has not been completely tested on 1.18 and not at all on 1.16

```
onEvent("command.registry", event => { //command registry event
  const { commands: Commands, arguments: Arguments } = event;
  event.register( //register a new command
    Commands.literal("myCommand") //the command is called myCommand
    [1].requires(src => src.hasPermission(2)) //2 is op. This line is optional, but you can also instead of just one value,
    wrap it in {}s and use return to write a more complex requirement checks
    [2].then(Commands.argument('arg1', Arguments.STRING.create(event))) //takes argument string called arg1. You
    can have as many (or none) as you want.
    [3].then(Commands.argument('arg2', Arguments.FLOAT.create(event))) //takes argument float called arg2. The oth
    type you can use can be found with ProbeJS
    [4].executes(ctx => { //run the command
      [5][6]const arg1 = Arguments.STRING.getResult(ctx, "arg1"); //get recipe
      [7][8]const arg2 = Arguments.FLOAT.getResult(ctx, "arg2"); //get the value
      //your code goes here
      [9][10]if(arg1 == "example")
        [11]return 0 //return 0 means command did not work
        let level = ctx.source.level.asKJS()
        let position = ctx.source.position
        //hurt entities in a around a area of where the command was run
        let i = 0
        level.getEntitiesWithin(AABB.of(position.x()-2,position.y()-2,position.z()-
        2,position.x()+2,position.y()+2,position.z()+2)).forEach(entity => {
          [12]if (entity.living) {
            entity.attack(arg2)
            i++
            if (entity.type == "minecraft:player") entity.tell(arg1) //tell players that got hurt the message
            that is arg1
          }
        })
      }
    })
  }
```

```
        }  
    })  
    return i // always return something  
    })  
    // every then requires a ')' so dont forget them  
    //but requires does not  
    )  
})
```

Revision #4

Created 11 January 2023 22:57:14 by Q6

Updated 12 January 2023 04:42:04 by Q6