MCLuaBukkit v0.63 API Reference

Don't forget to check out the official Lua 5.2 documentation: http://www.lua.org/manual/5.2/

1. Globals

chat(message)
Send message to all players.

info(message)
Send an info message to the console.

warning(message)
Send a warning message to the console.

schedule(function, delay)
Schedules a function to be run after a specified number of ticks. 1 tick = 1/20 second.

out = **chatCode**(in)
Formats a string to use Minecraft color codes by replacing number symbols (#) with section signs (§).

stop()
Saves world data and stops the server.

banList = **getBans**()
Get a table of banned player usernames.

IPBanList = **getIPBans**()
Get a table of banned IP addresses.

whitelist = **getWhitelists**()
Get a table of whitelisted player usernames.

opList = **getOps**()
Get a table of opped player usernames.

worlds = **getWorlds**()
Get a table of world objects.

players = **getPlayers**()
Get a table of online player objects.

world = **getWorld**(name)
Get a world object.

player = **getPlayer**(username)
Get an online player object.

mode = **getDefaultGamemode**()
Gets the default gamemode for the server.

rad = **getSpawnRadius**()
Gets the spawn protection radius.

setDefaultGamemode(mode)
Sets the default gamemode for the server.

setSpawnRadius(rad)
Sets server's spawn protection radius.

kick(username, reason)
Kick an online player for the specified reason.

ban(username)
Ban a player.

unban(username)
Pardon a player.

banIP(username)
Ban an IP address.

unbanIP(username)
Unban an IP Address.

whitelist(username)
Adds a player to the whitelist.

unwhitelist(username)
Removes a player from the whitelist.

op(username)
Ops a player.

deop(username)

Deops a player.

world = **loadWorld**(name, envID, type, genStructs, seed)
Loads (or creates) a world.
Environment ID Values: -1 = Nether, 0 = Overworld, 1 = End
Type Values: FLAT, LARGE_BIOMES, NORMAL

config = **loadConfig**(name)
Returns a config object for the named config file. A file extension is automatically appended.

2. Events

events.onPlayerJoin
An event object for when a player joins.
Listener args: player, message
Listener optional returns: message

events.onPlayerLeave
An event object for when a player leaves.
Listener args: player, message
Listener optional returns: message

events.onPlayerChat
An event object for when a player says a chat message. In arg and return format, %1$s will be replaced with the player name and %2$s will be replaced with the message. DO NOT insert the player name or message manually into the format, as this may cause the string to be improperly formatted. Also, listeners should not call most world related functions, as this event is asynchronous.
Listener args: player, message, format
Listener optional returns: cancel, message, format

events.onPlayerDeath
An event object for when a player dies. Last 3 args may be nil.
Listener args: player, message, cause, damager, playerDamager
Cause values: See http://jd.bukkit.org/dev/apidocs/org/bukkit/event/entity/EntityDamageEvent.DamageCause.html
Listener optional returns: message, noDrops

events.onPlayerMove
An event object for when a player moves.
Listener args: player, toX, toY, toZ, toYaw, toPitch, fromX, fromY, fromZ, fromYaw, FromPitch
Listener optional returns: cancel, toX, toY, toZ, toYaw, toPitch, fromX, fromY, fromZ,

fromYaw, FromPitch

events.onPlayerDamage
An event object for when a player takes damage. Last 2 args may be nil.
Listener args: player, damage, cause, damager, playerDamager
Cause values: See http://jd.bukkit.org/dev/apidocs/org/bukkit/event/entity/EntityDamageEvent.DamageCause.html
Listener optional returns: cancel, damage

events.onPlayerRespawn
An event object for when a player respawns.
Listener args: player, world, x, y, z, yaw, pitch
Listener optional returns: world, x, y, z, yaw, pitch

events.onPlayerInteract
An event object for when a player interacts with blocks or the air. Block and item args may be nil.
Listener args: player, action, itemID, itemAmount, itemData, blockID, blockData, blockX, blockY, blockZ
Action values: LEFT_CLICK_AIR, LEFT_CLICK_BLOCK, RIGHT_CLICK_AIR, RIGHT_CLICK_BLOCK, PHYSICAL
Listener optional returns: cancel

events.onChunkPopulate
An event object for when a chunk gets populated. Multiply chunk coordinates by 16 to get block coordinates.
Listener args: world, chunkX, chunkZ

events.onCreatureSpawn
An event object for when a creature spawns.
Listener args: type, reason, world, x, y, z
Type values: See http://jd.bukkit.org/dev/apidocs/org/bukkit/entity/EntityType.html
Reason values: See http://jd.bukkit.org/dev/apidocs/org/bukkit/event/entity/CreatureSpawnEvent.SpawnReason.html
Listener optional returns: cancel

3. Event Object

event:add(listener)
Adds a listener function to this event.

event:remove(listener)
Removes a listener function from this event.

listeners = **event:getAll**()
Gets a table of all listeners functions for this event.

event:clear()
Remove all listeners functions from this event.

4. World Object

players = **world:getPlayers**()
Gets a table of all online players in this world.

id, meta = **world:getBlock**(x, y, z)
Gets the ID and metadata of a block in this world.

pvp = **world:getPVP**()
Gets if pvp is enabled.

diff = **world:getDifficulty**()
Gets the world difficulty ID.

x, y, z = **world:getSpawn**()
Gets the world's default spawn location.

time = **world:getTime**()
Gets the current time for this world.

name = **world:getName**()
Gets the name of this world.

type = **world:getType**()
Gets the type of this world.
Type Values: FLAT, LARGE_BIOMES, NORMAL

env = **world:getEnvironment**()
Gets the environment of this world.
Environment Values: -1 = Nether, 0 = Overworld, 1 = End

seed = **world:getSeed**()
Gets the seed for this world.

genStructs = **world:getGenerateStructures**()
Gets if this world will generate structures.

world:setBlock(x, y, z, [id], [meta], [update])

Sets the ID and metadata of a block in this world.

world:setPVP(pvp)
Sets if PVP is enabled.

world:setDifficulty(diff)
Sets the world difficulty ID.

world:setSpawn(x, y, z)
Sets the world's default spawn location.

world:setTime(time)
Sets the current time for this world.

world:setBiome(x, z, biome)
Sets the biome at the given x and z coordinates.
Biome Values: See http://jd.bukkit.org/dev/apidocs/org/bukkit/block/Biome.html

world:setSpawnFlags(monsters, animals)
Sets the mob spawning flags for this world. A value of true allows the type of mob to spawn.

world:chat(message)
Sends a chat message to all players in this world.

world:explode(x, y, z, power, [doFire], [doBlockDamage])
Creates an explosion in this world.

world:lightning(x, y, z, [doDamage])
Strikes lightning in this world.

world:unload()
Saves and unloads this world.

world:spawn(x, y, z, entityType...)
Spawns the listed entities at the given coordinates. The first entity will be riding the second, the second riding the third, etc...
Entity Type values: See http://jd.bukkit.org/dev/apidocs/org/bukkit/entity/EntityType.html

5. Player Object

player:hide(player)
Hides a player from this player.

player:show(player)
Shows a player to this player.

player:chat(message)
Sends a chat message to this player.

player:shootProjectile(type)
Shoots an egg in the direction this player is facing.
Type values: ARROW, WITHER_SKULL, FIREBALL, SMALL_FIREBALL, SNOWBALL, EGG

player:giveExperience(amount)
Gives the specified amount of experience to the player.

failed = **player:giveItem**(id, amount, [damage])
Gives the player the specified item(s). Returns the amount of items that could not fit in the player's inventory.

failed = **player:takeItem**(id, amount, [damage])
Takes specified item(s) away from the player. Returns the amount of items that could not be removed.

air = **player:getAir**()
Gets the player's air ticks.

flight = **player:getAllowFlight**()
Gets if the player is allowed to fly.

exhaustion = **player:getExhaustion**()
Gets the player's exhaustion.

sat = **player:getSaturation**()
Gets the player's saturation.

food = **player:getFood**()
Gets the player's food.

exp = **player:getExperience**()
Gets the player's experience, from 0 to 1.

level = **player:getLevel**()
Gets the player's level.

mode = **player:getGamemode**()
Gets the player's gamemode.

hp = **player:getHealth**()
Gets the player's health.

maxHp = **player:getMaxHealth**()
Gets the player's maximum health.

world, x, y, z, yaw, pitch = **player:getLocation**()
Gets the player's current location, including coordinates and head angle.

ticks = **player:getFireTicks**()
Gets the ticks until the player is no longer on fire.

name = **player:getChatName**()
Gets the name used when the player says something in chat.

name = **player:getListName**()
Gets the name used in the player list.

name = **player:getName**()
Gets the player's username.

x, y, z = **player:getTargetBlock**(maxDist)
Gets the coordinates of the block the player is looking at with the specified maximum distance.

world, x, y, z = **player:getSpawn**()
Gets the player's spawn location. If world is nil, then it doesn't exist.

world, x, y, z = **player:getCompassTarget**()
Gets the player's compass target. If world is nil, then it doesn't exist.

pickup = **player:getPickupItems**()
Gets if the player can pick items up.

ignore = **player:getSleepingIgnored**()
Gets if this player is not required to sleep in order to set the time to day.

slot = **player:getHeldSlot**()
Returns the position of the item this player is holding in the hotbar. Returned value will be 1-9, inclusive.

ID, amount, data = **player:getArmorItem**(index)
Returns the data for the item at the specified location in the armor slots, or nil is it is empty. The first slot is 1.

ID, amount, data = **player:getInventoryItem**(index)
Returns the data for the item at the specified location in the inventory slots, or nil is it is empty. The first slot is 1.

player:setAir(air)
Sets the player's air ticks.

player:setAllowFlight(flight)
Sets if the player is allowed to fly.

player:setExhaustion(exhaustion)
Sets the player's exhaustion.

player:setSaturation(sat)
Sets the player's saturation.

player:setFood(food)
Sets the player's food.

player:getExperience(exp)
Sets the player's experience, from 0 to 1.

player:setLevel(level)
Sets the player's level.

player:setGamemode(mode)
Sets the player's gamemode.

player:setHealth(hp)
Sets the player's health.

player:setMaxHealth(maxHp)
Sets the player's maximum health.

player:setLocation([world], [x], [y], [z], [yaw], [pitch])
Sets the player's current location, including coordinates and head angle.

player:setFireTicks(ticks)
Sets the ticks until the player is no longer on fire.

player:setChatName(name)
Sets the name used when the player says something in chat.

player:setListName(name)
Sets the name used in the player list.

player:setSpawn(world, x, y, z)
Sets the player's spawn location. If world is nil, then clears it.

player:setCompassTarget(world, x, y, z)
Sets the player's compass target. If world is nil, then clears it.

player:setPickupItems(pickup)
Sets if the player can pick items up.

player:setSleepingIgnored(ignore)
Sets if this player is not required to sleep in order to set the time to day.

player:setHeldSlot(slot)
Sets the position of the item this player is holding in the hotbar. Value must be 1-9, inclusive.

player:setArmorItem(index, ID, amount, [data])
Sets the data for the item at the specified location in the armor slots. The first slot is 1.

player:setInventoryItem(index, ID, amount, [data])
Sets the data for the item at the specified location in the inventory slots. The first slot is 1.

5. Config Object

keys = **config:getKeys**()
Returns a table of keys in this config.

value = **config:get**(key)
Returns the value associated with the given key in this config.

config:set(key, value)
Sets the value associated with the given key in this config.

config:save()
Saves this config to its file.