screeps > api

Back to docs

Global Objects

Game

The main global game object containing all the game play information.

Game.constructionSitesobject<string, ConstructionSite>

A hash containing all your construction sites with their id as hash keys.

Game.cpuobject

An object containing information about your CPU usage with the following properties:

parameter	type	description
`limit`	number	Your assigned CPU limit for the current shard.
`tickLimit`	number	An amount of available CPU time at the current game tick. Usually it is higher than `Game.cpu.limit`. Learn more
`bucket`	number	An amount of unused CPU accumulated in your bucket.
`shardLimits`	object <string,number>	An object with limits for each shard with shard names as keys. You can use `setShardLimits` method to re-assign them.

Game.creepsobject<string, Creep>

```
for(const i in Game.creeps) {
    Game.creeps[i].moveTo(flag);
}
```

A hash containing all your creeps with creep names as hash keys.

Game.flagsobject<string, Flag>

```
creep.moveTo(Game.flags.Flag1);
```

A hash containing all your flags with flag names as hash keys.

Game.gclobject

Your Global Control Level, an object with the following properties :

parameter	type	description
`level`	number	The current level.
`progress`	number	The current progress to the next level.
`progressTotal`	number	The progress required to reach the next level.

Game.mapobject

A global object representing world map. See the documentation below.

Game.marketobject

A global object representing the in-game market. See the documentation below.

Game.resourcesobject

An object with your global resources that are bound to the account, like subscription tokens. Each object key is a resource constant, values are resources amounts.

Game.roomsobject<string, Room>

A hash containing all the rooms available to you with room names as hash keys. A room is visible if you have a creep or an owned structure in it.

Game.shardobject

An object describing the world shard where your script is currently being executed in.

parameter	type	description
`name`	string	The name of the shard.
`type`	string	Currently always equals to `normal`.
`ptr`	boolean	Whether this shard belongs to the PTR.

Game.spawnsobject<string, StructureSpawn>

```
for(const i in Game.spawns) {
    Game.spawns[i].createCreep(body);
}
```

A hash containing all your spawns with spawn names as hash keys.

Game.structuresobject<string, Structure>

A hash containing all your structures with structure id as hash keys.

Game.timenumber

```
console.log(Game.time);
```

System game tick counter. It is automatically incremented on every tick. Learn more

Game.cpu.getHeapStatistics()

```
let heap = Game.cpu.getHeapStatistics();
console.log(`Used ${heap.total_heap_size} / ${heap.heap_size_limit}`);
```

*This method is only available when **Virtual machine** is set to **Isolated** in your account runtime settings.*

Use this method to get heap statistics for your virtual machine. The return value is almost identical to the Node.js function `v8.getHeapStatistics()`. This function returns one additional property: `externally_allocated_size` which is the total amount of currently allocated memory which is not included in the v8 heap but counts against this isolate's memory limit. `ArrayBuffer` instances over a certain size are externally allocated and will be counted here.

RETURN VALUE

Returns an objects with heap statistics in the following format:

```
{
  "total_heap_size": 29085696,
  "total_heap_size_executable": 3670016,
  "total_physical_size": 26447928,
  "total_available_size": 319649520,
  "used_heap_size": 17493824,
  "heap_size_limit": 343932928,
  "malloced_memory": 8192,
  "peak_malloced_memory": 1060096,
  "does_zap_garbage": 0,
  "externally_allocated_size": 38430000
}
```

Game.cpu.getUsed()

```
if(Game.cpu.getUsed() > Game.cpu.tickLimit / 2) {
    console.log("Used half of CPU already!");
}
```

```
for(const name in Game.creeps) {
    const startCpu = Game.cpu.getUsed();

    // creep logic goes here

    const elapsed = Game.cpu.getUsed() - startCpu;
    console.log('Creep '+name+' has used '+elapsed+' CPU time');
}
```

Get amount of CPU time used from the beginning of the current game tick. Always returns 0 in the Simulation mode.

RETURN VALUE

Returns currently used CPU time as a float number.

Game.cpu.setShardLimits(limits)

```
Game.cpu.setShardLimits({shard0: 20, shard1: 10});
```

Allocate CPU limits to different shards. Total amount of CPU should remain equal to `Game.cpu.shardLimits`. This method can be used only once per 12 hours.

parameter	type	description
`limits`	object<string, number>	An object with CPU values for each shard in the same format as `Game.cpu.shardLimits`.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_BUSY`	-4	12-hours cooldown period is not over yet.
`ERR_INVALID_ARGS`	-10	The argument is not a valid shard limits object.

Game.getObjectById(id)

```
creep.memory.sourceId = creep.pos.findClosestByRange(FIND_SOURCES).id;
const source = Game.getObjectById(creep.memory.sourceId);
```

Get an object with the specified unique ID. It may be a game object of any type. Only objects from the rooms which are visible to you can be accessed.

parameter	type	description
`id`	string	The unique identificator.

RETURN VALUE

Returns an object instance or null if it cannot be found.

Game.notify(message, [groupInterval])

```
if(creep.hits < creep.memory.lastHits) {
    Game.notify('Creep '+creep+' has been attacked at '+creep.pos+'!');
}
creep.memory.lastHits = creep.hits;
```

```
if(Game.spawns['Spawn1'].energy == 0) {
    Game.notify(
        'Spawn1 is out of energy',
        180  // group these notifications for 3 hours
    );
}
```

Send a custom message at your profile email. This way, you can set up notifications to yourself on any occasion within the game. You can schedule up to 20 notifications during one game tick. Not available in the Simulation Room.

parameter	type	description
`message`	string	Custom text which will be sent in the message. Maximum length is 1000 characters.
`groupInterval`	number	If set to 0 (default), the notification will be scheduled immediately. Otherwise, it will be grouped with other notifications and mailed out later using the specified time in minutes.

Game.map

A global object representing world map. Use it to navigate between rooms.

`Game.map.describeExits(roomName)`

```
const exits = Game.map.describeExits('W8N3');
```

List all exits available from the room with the given name.

parameter	type	description
`roomName`	string	The room name.

RETURN VALUE

The exits information in the following format, or null if the room not found.

```
{
    "1": "W8N4",    // TOP
    "3": "W7N3",    // RIGHT
    "5": "W8N2",    // BOTTOM
    "7": "W9N3"     // LEFT
}
```

Game.map.findExit(fromRoom, toRoom, [opts])

```
if(creep.room != anotherRoomName) {
    const exitDir = Game.map.findExit(creep.room, anotherRoomName);
    const exit = creep.pos.findClosestByRange(exitDir);
    creep.moveTo(exit);
}
else {
    // go to some place in another room
}
```

```
creep.moveTo(new RoomPosition(25, 25, anotherRoomName));
```

Find the exit direction from the given room en route to another room.

parameter	type	description
`fromRoom`	string, Room	Start room name or room object.
`toRoom`	string, Room	Finish room name or room object.
`opts` *optional*	object	An object with the pathfinding options. See `findRoute`.

RETURN VALUE

The room direction constant, one of the following:

- `FIND_EXIT_TOP`
- `FIND_EXIT_RIGHT`
- `FIND_EXIT_BOTTOM`
- `FIND_EXIT_LEFT`

Or one of the following error codes:

constant	value	description
`ERR_NO_PATH`	-2	Path can not be found.
`ERR_INVALID_ARGS`	-10	The location is incorrect.

Game.map.findRoute(fromRoom, toRoom, [opts])

```
const route = Game.map.findRoute(creep.room, anotherRoomName);
if(route.length > 0) {
    console.log('Now heading to room '+route[0].room);
    const exit = creep.pos.findClosestByRange(route[0].exit);
    creep.moveTo(exit);
}
```

```
const route = Game.map.findRoute(creep.room, anotherRoomName, {
    routeCallback(roomName, fromRoomName) {
        if(roomName == 'W10S10') {    // avoid this room
            return Infinity;
        }
        return 1;
    }});
```

```
let from = new RoomPosition(25, 25, 'E1N1');
let to = new RoomPosition(25, 25, 'E4N1');

// Use `findRoute` to calculate a high-level plan for this path,
// prioritizing highways and owned rooms
let allowedRooms = { [ from.roomName ]: true };
Game.map.findRoute(from.roomName, to.roomName, {
    routeCallback(roomName) {
        let parsed = /^[WE]([0-9]+)[NS]([0-9]+)$/.exec(roomName);
        let isHighway = (parsed[1] % 10 === 0) ||
                        (parsed[2] % 10 === 0);
        let isMyRoom = Game.rooms[roomName] &&
            Game.rooms[roomName].controller &&
            Game.rooms[roomName].controller.my;
        if (isHighway || isMyRoom) {
            return 1;
        } else {
            return 2.5;
        }
    }
}).forEach(function(info) {
    allowedRooms[info.room] = true;
});

// Invoke PathFinder, allowing access only to rooms from `findRoute`
let ret = PathFinder.search(from, to, {
    roomCallback(roomName) {
        if (allowedRooms[roomName] === undefined) {
            return false;
        }
    }
});

console.log(ret.path);
```

Find route from the given room to another room.

parameter	type	description
`fromRoom`	string, Room	Start room name or room object.
`toRoom`	string, Room	Finish room name or room object.

parameter	type	description
`opts` *optional*	object	An object with the following options: • routeCallback function This callback accepts two arguments: `function(roomName, fromRoomName)`. It can be used to calculate the cost of entering that room. You can use this to do things like prioritize your own rooms, or avoid some rooms. You can return a floating point cost or `Infinity` to block the room.

RETURN VALUE

The route array in the following format:

```
[
    { exit: FIND_EXIT_RIGHT, room: 'arena21' },
    { exit: FIND_EXIT_BOTTOM, room: 'arena22' },
    ...
]
```

Or one of the following error codes:

constant	value	description
`ERR_NO_PATH`	-2	Path can not be found.

Game.map.getRoomLinearDistance(roomName1, roomName2, [continuous])

```
Game.map.getRoomLinearDistance('W1N1', 'W4N2'); // 3
Game.map.getRoomLinearDistance('E65S55','W65S55', false) // 131
Game.map.getRoomLinearDistance('E65S55','W65S55', true) // 11
```

Get the linear distance (in rooms) between two rooms. You can use this function to estimate the energy cost of sending resources through terminals, or using observers and nukes.

parameter	type	description
`roomName1`	string	The name of the first room.
`roomName2`	string	The name of the second room.
`continuous` *optional*	boolean	Whether to treat the world map continuous on borders. Set to true if you want to calculate the trade or terminal send cost. Default is false.

RETURN VALUE

A number of rooms between the given two rooms.

Game.map.getTerrainAt(x, y, roomName) (pos)

```
console.log(Game.map.getTerrainAt(25,20,'W10N10'));
```

```
console.log(Game.map.getTerrainAt(new RoomPosition(25,20,'W10N10'));
```

Get terrain type at the specified room position. This method works for any room in the world even if you have no access to it.

parameter	type	description
`x`	number	X position in the room.
`y`	number	Y position in the room.
`roomName`	string	The room name.
`pos`	RoomPosition	The position object.

RETURN VALUE

One of the following string values:

- `plain`
- `swamp`
- `wall`

Game.map.getWorldSize()

Returns the world size as a number of rooms between world corners. For example, for a world with rooms from W50N50 to E50S50 this method will return 102.

Game.map.isRoomAvailable(roomName)

```
if(Game.map.isRoomAvailable(room.name)) {
    creep.moveTo(room.getPositionAt(25,25));
}
```

Check if the room is available to move into.

parameter	type	description

parameter	type	description
`roomName`	string	The room name.

RETURN VALUE

A boolean value.

Game.market

A global object representing the in-game market. You can use this object to track resource transactions to/from your terminals, and your buy/sell orders.

Learn more about the market system from this article.

Game.market.creditsnumber

Your current credits balance.

Game.market.incomingTransactionsarray

```
[{
    transactionId : "56dec546a180ce641dd65960",
    time : 10390687,
    sender : {username: "Sender"},
    recipient : {username: "Me"},
    resourceType : "U",
    amount : 100,
    from : "W0N0",
    to : "W10N10",
    description : "trade contract #1",
    order: {        // optional
        id : "55c34a6b5be41a0a6e80c68b",
        type : "sell",
        price : 2.95
    }
}]
```

An array of the last 100 incoming transactions to your terminals with the following format:

Game.market.outgoingTransactionsarray

```
[{
    transactionId : "56dec546a180ce641dd65960",
    time : 10390687,
    sender : {username: "Me"},
    recipient : {username: "Recipient"},
    resourceType : "U",
    amount : 100,
    from : "W0N0",
    to : "W10N10",
    description : "trade contract #1",
    order: {        // optional
        id : "55c34a6b5be41a0a6e80c68b",
        type : "sell",
        price : 2.95
    }
}]
```

An array of the last 100 outgoing transactions from your terminals with the following format:

Game.market.ordersobject

```
{
    "55c34a6b5be41a0a6e80c68b": {
        id : "55c34a6b5be41a0a6e80c68b",
        created : 13131117,
        active: true,
        type : "sell"
        resourceType : "OH",
        roomName : "W1N1",
        amount : 15821,
        remainingAmount : 30000,
        totalAmount : 50000,
        price : 2.95
    },
    "55c34a6b52411a0a6e80693a": {
        id : "55c34a6b52411a0a6e80693a",
        created : 13134122,
        active: true,
        type : "buy"
        resourceType : "energy",
        roomName : "W1N1",
        amount : 94000,
        remainingAmount : 94000,
        totalAmount : 94000
        price : 0.45
    },
    "55c34a6b5be41a0a6e80c123": {
        id : "55c34a6b5be41a0a6e80c123",
        created : 13105123,
        active: false,
        type : "sell"
        resourceType : "token",
        amount : 0,
        remainingAmount : 10,
        totalAmount : 10,
        price : 50000
    }
}
```

An object with your active and inactive buy/sell orders on the market.

Game.market.calcTransactionCost(amount, roomName1, roomName2)

```
const cost = Game.market.calcTransactionCost(1000, 'W0N0', 'W10N5');
// -> 284 energy units
```

Estimate the energy transaction cost of `StructureTerminal.send` and `Game.market.deal` methods. The formula:

```
Math.ceil( amount * ( 1 - Math.exp(-distanceBetweenRooms/30) ) )
```

parameter	type	description
`amount`	number	Amount of resources to be sent.
`roomName1`	string	The name of the first room.
`roomName2`	string	The name of the second room.

RETURN VALUE

The amount of energy required to perform the transaction.

Game.market.cancelOrder(orderId)

```
for(const id in Game.market.orders) {
    Game.market.cancelOrder(id);
}
```

Cancel a previously created order. The 5% fee is not returned.

parameter	type	description
`orderId`	string	The order ID as provided in `Game.market.orders`.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_INVALID_ARGS`	-10	The order ID is not valid.

Game.market.changeOrderPrice(orderId, newPrice)

```
Game.market.changeOrderPrice('57bec1bf77f4d17c4c011960', 9.95);
```

Change the price of an existing order. If `newPrice` is greater than old price, you will be charged `(newPrice-oldPrice)*remainingAmount*0.05` credits.

parameter	type	description
`orderId`	string	The order ID as provided in `Game.market.orders`.
`newPrice`	number	The new order price.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of the room's terminal or there is no terminal.
`ERR_NOT_ENOUGH_RESOURCES`	-6	You don't have enough credits to pay a fee.
`ERR_INVALID_ARGS`	-10	The arguments provided are invalid.

Game.market.createOrder(type, resourceType, price, totalAmount, [roomName])

```
Game.market.createOrder(ORDER_SELL, RESOURCE_GHODIUM, 9.95, 10000, "W1N1");
```

Create a market order in your terminal. You will be charged `price*amount*0.05` credits when the order is placed. The maximum orders count is 50 per player. You can create an order at any time with any amount, it will be automatically activated and deactivated depending on the resource/credits availability.

parameter	type	description
`type`	string	The order type, either `ORDER_SELL` or `ORDER_BUY`.

parameter	type	description
`resourceType`	string	Either one of the `RESOURCE_*` constants or `SUBSCRIPTION_TOKEN`. If your Terminal doesn't have the specified resource, the order will be temporary inactive.
`price`	number	The price for one resource unit in credits. Can be a decimal number.
`totalAmount`	number	The amount of resources to be traded in total.
`roomName` *optional*	string	The room where your order will be created. You must have your own Terminal structure in this room, otherwise the created order will be temporary inactive. This argument is not used when `resourceType` equals to `SUBSCRIPTION_TOKEN`.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of the room's terminal or there is no terminal.
`ERR_NOT_ENOUGH_RESOURCES`	-6	You don't have enough credits to pay a fee.
`ERR_FULL`	-8	You cannot create more than 50 orders.
`ERR_INVALID_ARGS`	-10	The arguments provided are invalid.

Game.market.deal(orderId, amount, [yourRoomName])

```
Game.market.deal('57cd2b12cda69a004ae223a3', 1000, "W1N1");
```

```
const amountToBuy = 2000, maxTransferEnergyCost = 500;
const orders = Game.market.getAllOrders({type: ORDER_SELL, resourceType: RESOURCE_GHODIUM});

for(let i=0; i<orders.length; i++) {
    const transferEnergyCost = Game.market.calcTransactionCost(
        amountToBuy, 'W1N1', orders[i].roomName);

    if(transferEnergyCost < maxTransferEnergyCost) {
        Game.market.deal(orders[i].id, amountToBuy, "W1N1");
        break;
    }
}
```

Execute a trade deal from your Terminal in `yourRoomName` to another player's Terminal using the specified buy/sell order. Your Terminal will be charged energy units of transfer cost regardless of the order resource type. You can use `Game.market.calcTransactionCost` method to estimate it. When multiple players try to execute the same deal, the one with the shortest distance takes precedence. You cannot execute more than 10 deals during one tick.

parameter	type	description
`orderId`	string	The order ID as provided in `Game.market.getAllOrders`.
`amount`	number	The amount of resources to transfer.
`yourRoomName` *optional*	string	The name of your room which has to contain an active Terminal with enough amount of energy. This argument is not used when the order resource type equals to `SUBSCRIPTION_TOKEN`.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You don't have a terminal in the target room.
`ERR_NOT_ENOUGH_RESOURCES`	-6	You don't have enough credits or resource units.
`ERR_FULL`	-8	You cannot execute more than 10 deals during one tick.
`ERR_INVALID_ARGS`	-10	The arguments provided are invalid.
`ERR_TIRED`	-11	The target terminal is still cooling down.

Game.market.extendOrder(orderId, addAmount)

```
Game.market.extendOrder('57bec1bf77f4d17c4c011960', 10000);
```

Add more capacity to an existing order. It will affect `remainingAmount` and `totalAmount` properties. You will be charged `price*addAmount*0.05` credits.

parameter	type	description
`orderId`	string	The order ID as provided in `Game.market.orders`.
`addAmount`	number	How much capacity to add. Cannot be a negative value.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_ENOUGH_RESOURCES`	-6	You don't have enough credits to pay a fee.
`ERR_INVALID_ARGS`	-10	The arguments provided are invalid.

Game.market.getAllOrders([filter])

```
Game.market.getAllOrders();
```

```
Game.market.getAllOrders({type: ORDER_SELL, resourceType: RESOURCE_GHODIUM});
```

```
const targetRoom = "W1N1";
Game.market.getAllOrders(order => order.resourceType == RESOURCE_GHODIUM &&
    order.type == ORDER_SELL &&
    Game.market.calcTransactionCost(1000, targetRoom, order.roomName) < 500);
```

```
// Output:

[{
    id : "55c34a6b5be41a0a6e80c68b",
    created : 13131117,
    type : "sell"
    resourceType : "OH",
    roomName : "W1N1",
    amount : 15821,
    remainingAmount : 30000,
    price : 2.95
}, {
    id : "55c34a6b52411a0a6e80693a",
    created : 13134122,
    type : "buy"
    resourceType : "energy",
    roomName : "W1N1",
    amount : 94000,
    remainingAmount : 94000,
    price : 0.45
}, {
    id : "55c34a6b5be41a0a6e80c123",
    created : 13105123,
    type : "sell"
    resourceType : "token",
    amount : 3,
    remainingAmount : 10,
    price : 50000
}]
```

Get other players' orders currently active on the market.

parameter	type	description
`filter` *optional*	object, function	An object or function that will filter the resulting list using the `lodash.filter` method.

RETURN VALUE

An orders array in the following form:

property	description
`id`	The unique order ID.
`created`	The order creation time in game ticks.
`type`	Either `ORDER_SELL` or `ORDER_BUY`.
`resourceType`	Either one of the `RESOURCE_*` constants or `SUBSCRIPTION_TOKEN`.
`roomName`	The room where this order is placed.
`amount`	Currently available amount to trade.
`remainingAmount`	How many resources are left to trade via this order.
`price`	The current price per unit.

`Game.market.getOrderById(id)`

```
const order = Game.market.getOrderById('55c34a6b5be41a0a6e80c123');
```

Retrieve info for specific market order.

parameter	type	description
`id`	string	The order ID.

RETURN VALUE

An object with the order info. See `getAllOrders` for properties explanation.

Memory

A global plain object which can contain arbitrary data. You can access it both using the API and the Memory UI in the game editor. Learn how to work with memory from this article.

PathFinder

Contains powerful methods for pathfinding in the game world. This module is written in fast native C++ code and supports custom navigation costs and paths which span multiple rooms.

PathFinder.search(origin, goal, [opts])

```
let creep = Game.creeps.John;

let goals = _.map(creep.room.find(FIND_SOURCES), function(source) {
  // We can't actually walk on sources-- set `range` to 1
  // so we path next to it.
  return { pos: source.pos, range: 1 };
});

let ret = PathFinder.search(
  creep.pos, goals,
  {
    // We need to set the defaults costs higher so that we
    // can set the road cost lower in `roomCallback`
    plainCost: 2,
    swampCost: 10,

    roomCallback: function(roomName) {

      let room = Game.rooms[roomName];
      // In this example `room` will always exist, but since
      // PathFinder supports searches which span multiple rooms
      // you should be careful!
      if (!room) return;
      let costs = new PathFinder.CostMatrix;

      room.find(FIND_STRUCTURES).forEach(function(struct) {
        if (struct.structureType === STRUCTURE_ROAD) {
          // Favor roads over plain tiles
          costs.set(struct.pos.x, struct.pos.y, 1);
        } else if (struct.structureType !== STRUCTURE_CONTAINER &&
                   (struct.structureType !== STRUCTURE_RAMPART ||
                    !struct.my)) {
          // Can't walk through non-walkable buildings
          costs.set(struct.pos.x, struct.pos.y, 0xff);
        }
      });

      // Avoid creeps in the room
      room.find(FIND_CREEPS).forEach(function(creep) {
        costs.set(creep.pos.x, creep.pos.y, 0xff);
      });

      return costs;
    },
  }
);

let pos = ret.path[0];
creep.move(creep.pos.getDirectionTo(pos));
```

Find an optimal path between `origin` and `goal`.

parameter	type	description
`origin`	RoomPosition	The start position.
`goal`	object	A goal or an array of goals. If more than one goal is supplied then the cheapest path found out of all the goals will be returned. A goal is either a RoomPosition or an object as defined below. ***Important:*** Please note that if your goal is not walkable (for instance, a source) then you should set `range` to at least 1 or else you will waste many CPU cycles searching for a target that you can't walk on. • pos `RoomPosition` The target. • range number Range to `pos` before goal is considered reached. The default is 0.
`opts` *optional*	object	An object containing additional pathfinding flags. • roomCallback function Request from the pathfinder to generate a `CostMatrix` for a certain room. The callback accepts one argument, `roomName`. This callback will only be called once per room per search. If you are running multiple pathfinding operations in a single room and in a single tick you may consider caching your CostMatrix to speed up your code. Please read the CostMatrix documentation below for more information on CostMatrix. If you return `false` from the callback the requested room will not be searched, and it won't count against `maxRooms` • plainCost number Cost for walking on plain positions. The default is 1. • swampCost number Cost for walking on swamp positions. The default is 5. • flee boolean Instead of searching for a path *to* the goals this will search for a path *away* from the goals. The cheapest path that is out of `range` of every goal will be returned. The default is false. • maxOps number The maximum allowed pathfinding operations. You can limit CPU time used for the search based on ratio 1 op ~ 0.001 CPU. The default value is 2000. • maxRooms number The maximum allowed rooms to search. The default (and maximum) is 16. • maxCost number The maximum allowed cost of the path returned. If at any point the pathfinder detects that it is impossible to find a path with a cost less than or equal to `maxCost` it will immediately halt the search. The default is Infinity. • heuristicWeight number Weight to apply to the heuristic in the A* formula `F = G + weight * H`. Use this option only if you understand the underlying A* algorithm mechanics! The default value is 1.2.

RETURN VALUE

An object containing the following properties:

property	description
`path`	An array of RoomPosition objects.
`ops`	Total number of operations performed before this path was calculated.
`cost`	The total cost of the path as derived from `plainCost`, `swampCost` and any given CostMatrix instances.
`incomplete`	If the pathfinder fails to find a complete path, this will be true. Note that `path` will still be populated with a partial path which represents the closest path it could find given the search parameters.

PathFinder.use(isEnabled)

This method is deprecated and will be removed soon.

```
PathFinder.use(true);
Game.creeps.John.moveTo(Game.spawns['Spawn1']);
```

Specify whether to use this new experimental pathfinder in game objects methods. This method should be invoked every tick. It affects the following methods behavior: `Room.findPath`, `RoomPosition.findPathTo`, `RoomPosition.findClosestByPath`, `Creep.moveTo`.

parameter	type	description
`isEnabled`	boolean	Whether to activate the new pathfinder or deactivate. The default is `true`.

RawMemory

RawMemory object allows to implement your own memory stringifier instead of built-in serializer based on `JSON.stringify`. It also allows to request up to 10 MB of additional memory using asynchronous memory segments feature.

You can also access memory segments of other players using methods below.

RawMemory.segmentsobject

```
RawMemory.setActiveSegments([0,3]);
// on the next tick
console.log(RawMemory.segments[0]);
console.log(RawMemory.segments[3]);
RawMemory.segments[3] = '{"foo": "bar", "counter": 15}';
```

An object with asynchronous memory segments available on this tick. Each object key is the segment ID with data in string values. Use `setActiveSegments` to fetch segments on the next tick. Segments data is saved automatically in the end of the tick. The maximum size per segment is 100

KB.

RawMemory.foreignSegmentobject

```
RawMemory.setActiveForeignSegment('player');
// on the next tick
console.log(RawMemory.foreignSegment);
// --> {"username": "player", "id": 40, "data": "Hello!"}
```

An object with a memory segment of another player available on this tick. Use `setActiveForeignSegment` to fetch segments on the next tick. The object consists of the following properties:

parameter	type	description
`username`	string	Another player's name.
`id`	number	The ID of the requested memory segment.
`data`	string	The segment contents.

RawMemory.interShardSegmentstring

```
RawMemory.interShardSegment = JSON.stringify({
    creeps: {
        Bob: {role: 'claimer'}
    }
});

// on another shard
var interShardData = JSON.parse(RawMemory.interShardSegment);
if(interShardData.creeps[creep.name]) {
    creep.memory = interShardData[creep.name];
    delete interShardData.creeps[creep.name];
}
RawMemory.interShardSegment = JSON.stringify(interShardData);
```

A string with a shared memory segment available on every world shard. Maximum string length is 100 KB.

Warning: this segment is not safe for concurrent usage! All shards have shared access to the same instance of data. When the segment contents is changed by two shards simultaneously, you may lose some data, since the segment string value is written all at once atomically. You must implement your own system to determine when each shard is allowed to rewrite the inter-shard memory, e.g. based on mutual exclusions.

RawMemory.get()

```
const myMemory = JSON.parse(RawMemory.get());
```

Get a raw string representation of the `Memory` object.

RETURN VALUE

Returns a string value.

RawMemory.set(value)

```
RawMemory.set(JSON.stringify(myMemory));
```

Set new `Memory` value.

parameter	type	description
`value`	string	New memory value as a string.

RawMemory.setActiveSegments(ids)

```
RawMemory.setActiveSegments([0,3]);
```

Request memory segments using the list of their IDs. Memory segments will become available on the next tick in `segments` object.

parameter	type	description
`ids`	array	An array of segment IDs. Each ID should be a number from 0 to 99. Maximum 10 segments can be active at the same time. Subsequent calls of `setActiveSegments` override previous ones.

RawMemory.setActiveForeignSegment(username, [id])

```
RawMemory.setActiveForeignSegment('player');
```

```
RawMemory.setActiveForeignSegment('player', 10);
```

```
RawMemory.setActiveForeignSegment(null);
```

Request a memory segment of another user. The segment should be marked by its owner as public using `setPublicSegments`. The segment data will become available on the next tick in `foreignSegment` object. You can only have access to one foreign segment at the same time.

parameter	type	description
`username`	string | null	The name of another user. Pass `null` to clear the foreign segment.
`id` *optional*	number	The ID of the requested segment from 0 to 99. If undefined, the user's default public segment is requested as set by `setDefaultPublicSegment`.

RawMemory.setDefaultPublicSegment(id)

```
RawMemory.setPublicSegments([5,20,21]);
RawMemory.setDefaultPublicSegment(5);
```

```
RawMemory.setDefaultPublicSegment(null);
```

Set the specified segment as your default public segment. It will be returned if no `id` parameter is passed to `setActiveForeignSegment` by another user.

parameter	type	description
`id`	number | null	The ID of the memory segment from 0 to 99. Pass `null` to remove your default public segment.

RawMemory.setPublicSegments(ids)

```
RawMemory.setPublicSegments([5,3]);
```

```
RawMemory.setPublicSegments([]);
```

Set specified segments as public. Other users will be able to request access to them using `setActiveForeignSegment`.

parameter	type	description
`ids`	array	An array of segment IDs. Each ID should be a number from 0 to 99. Subsequent calls of `setPublicSegments` override previous ones.

Constants

All the following constant names are available in the global scope:

```
{
    OK: 0,
    ERR_NOT_OWNER: -1,
    ERR_NO_PATH: -2,
    ERR_NAME_EXISTS: -3,
    ERR_BUSY: -4,
    ERR_NOT_FOUND: -5,
    ERR_NOT_ENOUGH_ENERGY: -6,
    ERR_NOT_ENOUGH_RESOURCES: -6,
    ERR_INVALID_TARGET: -7,
    ERR_FULL: -8,
    ERR_NOT_IN_RANGE: -9,
    ERR_INVALID_ARGS: -10,
    ERR_TIRED: -11,
    ERR_NO_BODYPART: -12,
    ERR_NOT_ENOUGH_EXTENSIONS: -6,
    ERR_RCL_NOT_ENOUGH: -14,
    ERR_GCL_NOT_ENOUGH: -15,

    FIND_EXIT_TOP: 1,
    FIND_EXIT_RIGHT: 3,
    FIND_EXIT_BOTTOM: 5,
    FIND_EXIT_LEFT: 7,
    FIND_EXIT: 10,
    FIND_CREEPS: 101,
    FIND_MY_CREEPS: 102,
    FIND_HOSTILE_CREEPS: 103,
    FIND_SOURCES_ACTIVE: 104,
    FIND_SOURCES: 105,
    FIND_DROPPED_ENERGY: -106,
    FIND_DROPPED_RESOURCES: 106,
    FIND_STRUCTURES: 107,
    FIND_MY_STRUCTURES: 108,
    FIND_HOSTILE_STRUCTURES: 109,
    FIND_FLAGS: 110,
    FIND_CONSTRUCTION_SITES: 111,
    FIND_MY_SPAWNS: 112,
    FIND_HOSTILE_SPAWNS: 113,
    FIND_MY_CONSTRUCTION_SITES: 114,
    FIND_HOSTILE_CONSTRUCTION_SITES: 115,
    FIND_MINERALS: 116,
    FIND_NUKES: 117,
    FIND_TOMBSTONES: 118,

    TOP: 1,
    TOP_RIGHT: 2,
    RIGHT: 3,
    BOTTOM_RIGHT: 4,
    BOTTOM: 5,
    BOTTOM_LEFT: 6,
    LEFT: 7,
    TOP_LEFT: 8,

    COLOR_RED: 1,
    COLOR_PURPLE: 2,
    COLOR_BLUE: 3,
    COLOR_CYAN: 4,
    COLOR_GREEN: 5,
    COLOR_YELLOW: 6,
    COLOR_ORANGE: 7,
    COLOR_BROWN: 8,
    COLOR_GREY: 9,
    COLOR_WHITE: 10,

    LOOK_CREEPS: "creep",
    LOOK_ENERGY: "energy",
    LOOK_RESOURCES: "resource",
    LOOK_SOURCES: "source",
    LOOK_MINERALS: "mineral",
    LOOK_STRUCTURES: "structure",
    LOOK_FLAGS: "flag",
    LOOK_CONSTRUCTION_SITES: "constructionSite",
    LOOK_NUKES: "nuke",
    LOOK_TERRAIN: "terrain",
    LOOK_TOMBSTONES: "tombstone",

    OBSTACLE_OBJECT_TYPES: ["spawn", "creep", "wall", "source", "constructedWall", "extension", "link", "storage", "tower", "observer", "powerSpawn", "powerBank", "lab", "terminal

    MOVE: "move",
    WORK: "work",
    CARRY: "carry",
    ATTACK: "attack",
    RANGED_ATTACK: "ranged_attack",
    TOUGH: "tough",
    HEAL: "heal",
    CLAIM: "claim",

    BODYPART_COST: {
        "move": 50,
        "work": 100,
        "attack": 80,
        "carry": 50,
        "heal": 250,
        "ranged_attack": 150,
        "tough": 10,
        "claim": 600
    },

    // WORLD_WIDTH and WORLD_HEIGHT constants are deprecated, please use Game.map.getWorldSize() instead
    WORLD_WIDTH: 202,
    WORLD_HEIGHT: 202,

    CREEP_LIFE_TIME: 1500,
    CREEP_CLAIM_LIFE_TIME: 600,
    CREEP_CORPSE_RATE: 0.2,

    CARRY_CAPACITY: 50,
    HARVEST_POWER: 2,
    HARVEST_MINERAL_POWER: 1,
    REPAIR_POWER: 100,
    DISMANTLE_POWER: 50,
    BUILD_POWER: 5,
    ATTACK_POWER: 30,
    UPGRADE_CONTROLLER_POWER: 1,
    RANGED_ATTACK_POWER: 10,
    HEAL_POWER: 12,
    RANGED_HEAL_POWER: 4,
    REPAIR_COST: 0.01,
    DISMANTLE_COST: 0.005,

    RAMPART_DECAY_AMOUNT: 300,
    RAMPART_DECAY_TIME: 100,
    RAMPART_HITS: 1,
    RAMPART_HITS_MAX: {2: 300000, 3: 1000000, 4: 3000000, 5: 10000000, 6: 30000000, 7: 100000000, 8: 300000000},

    ENERGY_REGEN_TIME: 300,
    ENERGY_DECAY: 1000,

    SPAWN_HITS: 5000,
    SPAWN_ENERGY_START: 300,
    SPAWN_ENERGY_CAPACITY: 300,
    CREEP_SPAWN_TIME: 3,
    SPAWN_RENEW_RATIO: 1.2,

    SOURCE_ENERGY_CAPACITY: 3000,
    SOURCE_ENERGY_NEUTRAL_CAPACITY: 1500,
    SOURCE_ENERGY_KEEPER_CAPACITY: 4000,

    WALL_HITS: 1,
    WALL_HITS_MAX: 300000000,

    EXTENSION_HITS: 1000,
    EXTENSION_ENERGY_CAPACITY: {0: 50, 1: 50, 2: 50, 3: 50, 4: 50, 5: 50, 6: 50, 7: 100, 8: 200},
    ROAD_HITS: 5000,
    ROAD_WEAROUT: 1,
    ROAD_DECAY_AMOUNT: 100,
    ROAD_DECAY_TIME: 1000,

    LINK_HITS: 1000,
    LINK_HITS_MAX: 1000,
    LINK_CAPACITY: 800,
    LINK_COOLDOWN: 1,
    LINK_LOSS_RATIO: 0.03,

    STORAGE_CAPACITY: 1000000,
    STORAGE_HITS: 10000,

    STRUCTURE_SPAWN: "spawn",
    STRUCTURE_EXTENSION: "extension",
    STRUCTURE_ROAD: "road",
    STRUCTURE_WALL: "constructedWall",
    STRUCTURE_RAMPART: "rampart",
    STRUCTURE_KEEPER_LAIR: "keeperLair",
    STRUCTURE_PORTAL: "portal",
    STRUCTURE_CONTROLLER: "controller",
    STRUCTURE_LINK: "link",
    STRUCTURE_STORAGE: "storage",
    STRUCTURE_TOWER: "tower",
    STRUCTURE_OBSERVER: "observer",
    STRUCTURE_POWER_BANK: "powerBank",
    STRUCTURE_POWER_SPAWN: "powerSpawn",
    STRUCTURE_EXTRACTOR: "extractor",
    STRUCTURE_LAB: "lab",
    STRUCTURE_TERMINAL: "terminal",
    STRUCTURE_CONTAINER: "container",
    STRUCTURE_NUKER: "nuker",

    CONSTRUCTION_COST: {
        "spawn": 15000,
        "extension": 3000,
        "road": 300,
        "constructedWall": 1,
        "rampart": 1,
        "link": 5000,
        "storage": 30000,
        "tower": 5000,
        "observer": 8000,
        "powerSpawn": 100000,
        "extractor": 5000,
        "lab": 50000,
        "terminal": 100000,
        "container": 5000,
        "nuker": 100000
    },
    CONSTRUCTION_COST_ROAD_SWAMP_RATIO: 5,

    CONTROLLER_LEVELS: {1: 200, 2: 45000, 3: 135000, 4: 405000, 5: 1215000, 6: 3645000, 7: 10935000},
    CONTROLLER_STRUCTURES: {
        "spawn": {0: 0, 1: 1, 2: 1, 3: 1, 4: 1, 5: 1, 6: 1, 7: 2, 8: 3},
        "extension": {0: 0, 1: 0, 2: 5, 3: 10, 4: 20, 5: 30, 6: 40, 7: 50, 8: 60},
        "link": {1: 0, 2: 0, 3: 0, 4: 0, 5: 2, 6: 3, 7: 4, 8: 6},
        "road": {0: 2500, 1: 2500, 2: 2500, 3: 2500, 4: 2500, 5: 2500, 6: 2500, 7: 2500, 8: 2500},
        "constructedWall": {1: 0, 2: 2500, 3: 2500, 4: 2500, 5: 2500, 6: 2500, 7: 2500, 8: 2500},
        "rampart": {1: 0, 2: 2500, 3: 2500, 4: 2500, 5: 2500, 6: 2500, 7: 2500, 8: 2500},
        "storage": {1: 0, 2: 0, 3: 0, 4: 1, 5: 1, 6: 1, 7: 1, 8: 1},
        "tower": {1: 0, 2: 0, 3: 1, 4: 1, 5: 2, 6: 2, 7: 3, 8: 6},
        "observer": {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1},
        "powerSpawn": {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1},
        "extractor": {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 1, 7: 1, 8: 1},
        "terminal": {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 1, 7: 1, 8: 1},
        "lab": {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 3, 7: 6, 8: 10},
        "container": {0: 5, 1: 5, 2: 5, 3: 5, 4: 5, 5: 5, 6: 5, 7: 5, 8: 5},
        "nuker": {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1}
    },
    CONTROLLER_DOWNGRADE: {1: 20000, 2: 5000, 3: 10000, 4: 20000, 5: 40000, 6: 60000, 7: 100000, 8: 150000},
    CONTROLLER_DOWNGRADE_RESTORE: 100,
    CONTROLLER_DOWNGRADE_SAFEMODE_THRESHOLD: 5000,
    CONTROLLER_CLAIM_DOWNGRADE: 300,
    CONTROLLER_RESERVE: 1,
    CONTROLLER_RESERVE_MAX: 5000,
    CONTROLLER_MAX_UPGRADE_PER_TICK: 15,
    CONTROLLER_ATTACK_BLOCKED_UPGRADE: 1000,
    CONTROLLER_NUKE_BLOCKED_UPGRADE: 200,

    SAFE_MODE_DURATION: 20000,
    SAFE_MODE_COOLDOWN: 50000,
    SAFE_MODE_COST: 1000,

    TOWER_HITS: 3000,
    TOWER_CAPACITY: 1000,
    TOWER_ENERGY_COST: 10,
    TOWER_POWER_ATTACK: 600,
    TOWER_POWER_HEAL: 400,
    TOWER_POWER_REPAIR: 800,
    TOWER_OPTIMAL_RANGE: 5,
    TOWER_FALLOFF_RANGE: 20,
    TOWER_FALLOFF: 0.75,

    OBSERVER_HITS: 500,
    OBSERVER_RANGE: 10,

    POWER_BANK_HITS: 2000000,
    POWER_BANK_CAPACITY_MAX: 5000,
    POWER_BANK_CAPACITY_MIN: 500,
    POWER_BANK_CAPACITY_CRIT: 0.3,
    POWER_BANK_DECAY: 5000,
    POWER_BANK_HIT_BACK: 0.5,

    POWER_SPAWN_HITS: 5000,
    POWER_SPAWN_ENERGY_CAPACITY: 5000,
    POWER_SPAWN_POWER_CAPACITY: 100,
    POWER_SPAWN_ENERGY_RATIO: 50,

    EXTRACTOR_HITS: 500,
    EXTRACTOR_COOLDOWN: 5,

    LAB_HITS: 500,
    LAB_MINERAL_CAPACITY: 3000,
    LAB_ENERGY_CAPACITY: 2000,
    LAB_BOOST_ENERGY: 20,
    LAB_BOOST_MINERAL: 30,
    LAB_COOLDOWN: 10,
    LAB_REACTION_AMOUNT: 5,

    GCL_POW: 2.4,
    GCL_MULTIPLY: 1000000,
    GCL_NOVICE: 3,

    MODE_SIMULATION: null,
    MODE_WORLD: null,

    TERRAIN_MASK_WALL: 1,
    TERRAIN_MASK_SWAMP: 2,
    TERRAIN_MASK_LAVA: 4,

    MAX_CONSTRUCTION_SITES: 100,
    MAX_CREEP_SIZE: 50,

    MINERAL_REGEN_TIME: 50000,
    MINERAL_MIN_AMOUNT: {
        "H": 35000,
        "O": 35000,
        "L": 35000,
        "K": 35000,
        "Z": 35000,
        "U": 35000,
        "X": 35000
    },
    MINERAL_RANDOM_FACTOR: 2,

    MINERAL_DENSITY: {
        1: 15000,
        2: 35000,
        3: 70000,
        4: 100000
    },
    MINERAL_DENSITY_PROBABILITY  : {
        1: 0.1,
        2: 0.5,
        3: 0.9,
        4: 1.0
    },
    MINERAL_DENSITY_CHANGE: 0.05,

    DENSITY_LOW: 1,
    DENSITY_MODERATE: 2,
    DENSITY_HIGH: 3,
    DENSITY_ULTRA: 4,

    TERMINAL_CAPACITY: 300000,
    TERMINAL_HITS: 3000,
    TERMINAL_SEND_COST: 0.1,
    TERMINAL_MIN_SEND: 100,
    TERMINAL_COOLDOWN: 10,

    CONTAINER_HITS: 250000,
    CONTAINER_CAPACITY: 2000,
    CONTAINER_DECAY: 5000,
    CONTAINER_DECAY_TIME: 100,
    CONTAINER_DECAY_TIME_OWNED: 500,

    NUKER_HITS: 1000,
    NUKER_COOLDOWN: 100000,
    NUKER_ENERGY_CAPACITY: 300000,
    NUKER_GHODIUM_CAPACITY: 5000,
    NUKE_LAND_TIME: 50000,
    NUKE_RANGE: 10,
    NUKE_DAMAGE: {
        0: 10000000,
        2: 5000000
    },

    TOMBSTONE_DECAY_PER_PART: 5,

    PORTAL_DECAY: 30000,

    ORDER_SELL: "sell",
    ORDER_BUY: "buy",

    MARKET_FEE: 0.05,

    FLAGS_LIMIT: 10000,

    SUBSCRIPTION_TOKEN: "token",

    RESOURCE_ENERGY: "energy",
    RESOURCE_POWER: "power",

    RESOURCE_HYDROGEN: "H",
    RESOURCE_OXYGEN: "O",
    RESOURCE_UTRIUM: "U",
    RESOURCE_LEMERGIUM: "L",
    RESOURCE_KEANIUM: "K",
    RESOURCE_ZYNTHIUM: "Z",
    RESOURCE_CATALYST: "X",
    RESOURCE_GHODIUM: "G",

    RESOURCE_HYDROXIDE: "OH",
    RESOURCE_ZYNTHIUM_KEANITE: "ZK",
    RESOURCE_UTRIUM_LEMERGITE: "UL",

    RESOURCE_UTRIUM_HYDRIDE: "UH",
    RESOURCE_UTRIUM_OXIDE: "UO",
    RESOURCE_KEANIUM_HYDRIDE: "KH",
    RESOURCE_KEANIUM_OXIDE: "KO",
    RESOURCE_LEMERGIUM_HYDRIDE: "LH",
    RESOURCE_LEMERGIUM_OXIDE: "LO",
    RESOURCE_ZYNTHIUM_HYDRIDE: "ZH",
    RESOURCE_ZYNTHIUM_OXIDE: "ZO",
    RESOURCE_GHODIUM_HYDRIDE: "GH",
    RESOURCE_GHODIUM_OXIDE: "GO",

    RESOURCE_UTRIUM_ACID: "UH2O",
    RESOURCE_UTRIUM_ALKALIDE: "UHO2",
    RESOURCE_KEANIUM_ACID: "KH2O",
    RESOURCE_KEANIUM_ALKALIDE: "KHO2",
    RESOURCE_LEMERGIUM_ACID: "LH2O",
    RESOURCE_LEMERGIUM_ALKALIDE: "LHO2",
    RESOURCE_ZYNTHIUM_ACID: "ZH2O",
    RESOURCE_ZYNTHIUM_ALKALIDE: "ZHO2",
    RESOURCE_GHODIUM_ACID: "GH2O",
    RESOURCE_GHODIUM_ALKALIDE: "GHO2",

    RESOURCE_CATALYZED_UTRIUM_ACID: "XUH2O",
    RESOURCE_CATALYZED_UTRIUM_ALKALIDE: "XUHO2",
    RESOURCE_CATALYZED_KEANIUM_ACID: "XKH2O",
    RESOURCE_CATALYZED_KEANIUM_ALKALIDE: "XKHO2",
    RESOURCE_CATALYZED_LEMERGIUM_ACID: "XLH2O",
    RESOURCE_CATALYZED_LEMERGIUM_ALKALIDE: "XLHO2",
    RESOURCE_CATALYZED_ZYNTHIUM_ACID: "XZH2O",
    RESOURCE_CATALYZED_ZYNTHIUM_ALKALIDE: "XZHO2",
    RESOURCE_CATALYZED_GHODIUM_ACID: "XGH2O",
    RESOURCE_CATALYZED_GHODIUM_ALKALIDE: "XGHO2",

    REACTIONS: {
        H: {
            O: "OH",
            L: "LH",
            K: "KH",
            U: "UH",
            Z: "ZH",
            G: "GH"
        },
        O: {
            H: "OH",
            L: "LO",
            K: "KO",
            U: "UO",
            Z: "ZO",
            G: "GO"
        },
        Z: {
            K: "ZK",
            H: "ZH",
            O: "ZO"
        },
        L: {
            U: "UL",
            H: "LH",
            O: "LO"
        },
        K: {
            Z: "ZK",
            H: "KH",
            O: "KO"
        },
        G: {
            H: "GH",
            O: "GO"
        },
    U: {
        L: "UL",
        H: "UH",
        O: "UO"
    },
    OH: {
        UH: "UH2O",
        UO: "UHO2",
        ZH: "ZH2O",
        ZO: "ZHO2",
        KH: "KH2O",
        KO: "KHO2",
        LH: "LH2O",
        LO: "LHO2",
        GH: "GH2O",
        GO: "GHO2"
    },
    X: {
        UH2O: "XUH2O",
        UHO2: "XUHO2",
        LH2O: "XLH2O",
        LHO2: "XLHO2",
        KH2O: "XKH2O",
        KHO2: "XKHO2",
        ZH2O: "XZH2O",
        ZHO2: "XZHO2",
        GH2O: "XGH2O",
        GHO2: "XGHO2"
    },
    ZK: {
        UL: "G"
    },
    UL: {
        ZK: "G"
    },
    LH: {
        OH: "LH2O"
    },
    ZH: {
        OH: "ZH2O"
    },
    GH: {
        OH: "GH2O"
    },
    KH: {
        OH: "KH2O"
    },
    UH: {
        OH: "UH2O"
    },
    LO: {
        OH: "LHO2"
    },
    ZO: {
        OH: "ZHO2"
    },
    KO: {
        OH: "KHO2"
    },
    UO: {
        OH: "UHO2"
    },
    GO: {
        OH: "GHO2"
    },
    LH2O: {
        X: "XLH2O"
    },
    KH2O: {
        X: "XKH2O"
    },
    ZH2O: {
            X: "XZH2O"
        },
        UH2O: {
            X: "XUH2O"
        },
        GH2O: {
            X: "XGH2O"
        },
        LHO2: {
            X: "XLHO2"
        },
        UHO2: {
            X: "XUHO2"
        },
        KHO2: {
            X: "XKHO2"
        },
        ZHO2: {
            X: "XZHO2"
        },
        GHO2: {
            X: "XGHO2"
        }
    },

    BOOSTS: {
        work: {
            UO: {
                harvest: 3
            },
            UHO2: {
                harvest: 5
            },
            XUHO2: {
                harvest: 7
            },
            LH: {
                build: 1.5,
                repair: 1.5
            },
            LH2O: {
                build: 1.8,
                repair: 1.8
            },
            XLH2O: {
                build: 2,
                repair: 2
            },
            ZH: {
                dismantle: 2
            },
            ZH2O: {
                dismantle: 3
            },
            XZH2O: {
                dismantle: 4
            },
            GH: {
                upgradeController: 1.5
            },
            GH2O: {
                upgradeController: 1.8
            },
            XGH2O: {
                upgradeController: 2
            }
        },
        attack: {
            UH: {
                attack: 2
            },
            UH2O: {
                attack: 3
            },
            XUH2O: {
                attack: 4
            }
        },
        ranged_attack: {
            KO: {
                rangedAttack: 2,
                rangedMassAttack: 2
            },
            KHO2: {
                rangedAttack: 3,
                rangedMassAttack: 3
            },
            XKHO2: {
                rangedAttack: 4,
                rangedMassAttack: 4
            }
        },
        heal: {
            LO: {
                heal: 2,
                rangedHeal: 2
            },
            LHO2: {
                heal: 3,
                rangedHeal: 3
            },
            XLHO2: {
                heal: 4,
                rangedHeal: 4
            }
        },
        carry: {
            KH: {
                capacity: 2
            },
            KH2O: {
                capacity: 3
            },
            XKH2O: {
                capacity: 4
            }
        },
        move: {
            ZO: {
                fatigue: 2
            },
            ZHO2: {
                fatigue: 3
            },
            XZHO2: {
                fatigue: 4
            }
        },
        tough: {
            GO: {
                damage: .7
            },
            GHO2: {
                damage: .5
            },
            XGHO2: {
                damage: .3
            }
        }
    },

    REACTION_TIME: {
        OH: 20,
        ZK: 5,
        UL: 5,
        G: 5,
        UH: 10,
        UH2O: 5,
        XUH2O: 60,
        UO: 10,
        UHO2: 5,
        XUHO2: 60,
        KH: 10,
        KH2O: 5,
        XKH2O: 60,
        KO: 10,
        KHO2: 5,
        XKHO2: 60,
        LH: 15,
        LH2O: 10,
        XLH2O: 65,
        LO: 10,
        LHO2: 5,
        XLHO2: 60,
        ZH: 20,
        ZH2O: 40,
        XZH2O: 160,
        ZO: 10,
        ZHO2: 5,
        XZHO2: 60,
        GH: 10,
        GH2O: 15,
        XGH2O: 80,
        GO: 10,
        GHO2: 15,
        XGHO2: 90,
    },

    PORTAL_UNSTABLE: 10*24*3600*1000,
    PORTAL_MIN_TIMEOUT: 12*24*3600*1000,
    PORTAL_MAX_TIMEOUT: 22*24*3600*1000,

    POWER_BANK_RESPAWN_TIME: 50000,

    INVADERS_ENERGY_GOAL: 100000,

    SYSTEM_USERNAME: 'Screeps',

    SIGN_NOVICE_AREA: 'A new Novice Area is being planned somewhere in this sector. Please make sure all important rooms are reserved.',
    SIGN_RESPAWN_AREA: 'A new Respawn Area is being planned somewhere in this sector. Please make sure all important rooms are reserved.'
};


BODYPARTS_ALL = [
    MOVE,
    WORK,
    CARRY,
    ATTACK,
    RANGED_ATTACK,
    TOUGH,
    HEAL,
    CLAIM
];

RESOURCES_ALL = [
    RESOURCE_ENERGY,
    RESOURCE_POWER,

    RESOURCE_HYDROGEN,
    RESOURCE_OXYGEN,
    RESOURCE_UTRIUM,
    RESOURCE_KEANIUM,
    RESOURCE_LEMERGIUM,
    RESOURCE_ZYNTHIUM,
    RESOURCE_CATALYST,
    RESOURCE_GHODIUM,
    RESOURCE_HYDROXIDE,
    RESOURCE_ZYNTHIUM_KEANITE,
    RESOURCE_UTRIUM_LEMERGITE,

    RESOURCE_UTRIUM_HYDRIDE,
    RESOURCE_UTRIUM_OXIDE,
    RESOURCE_KEANIUM_HYDRIDE,
    RESOURCE_KEANIUM_OXIDE,
    RESOURCE_LEMERGIUM_HYDRIDE,
    RESOURCE_LEMERGIUM_OXIDE,
    RESOURCE_ZYNTHIUM_HYDRIDE,
    RESOURCE_ZYNTHIUM_OXIDE,
    RESOURCE_GHODIUM_HYDRIDE,
    RESOURCE_GHODIUM_OXIDE,

    RESOURCE_UTRIUM_ACID,
    RESOURCE_UTRIUM_ALKALIDE,
    RESOURCE_KEANIUM_ACID,
    RESOURCE_KEANIUM_ALKALIDE,
    RESOURCE_LEMERGIUM_ACID,
    RESOURCE_LEMERGIUM_ALKALIDE,
    RESOURCE_ZYNTHIUM_ACID,
    RESOURCE_ZYNTHIUM_ALKALIDE,
    RESOURCE_GHODIUM_ACID,
    RESOURCE_GHODIUM_ALKALIDE,

    RESOURCE_CATALYZED_UTRIUM_ACID,
    RESOURCE_CATALYZED_UTRIUM_ALKALIDE,
    RESOURCE_CATALYZED_KEANIUM_ACID,
    RESOURCE_CATALYZED_KEANIUM_ALKALIDE,
    RESOURCE_CATALYZED_LEMERGIUM_ACID,
    RESOURCE_CATALYZED_LEMERGIUM_ALKALIDE,
    RESOURCE_CATALYZED_ZYNTHIUM_ACID,
    RESOURCE_CATALYZED_ZYNTHIUM_ALKALIDE,
    RESOURCE_CATALYZED_GHODIUM_ACID,
    RESOURCE_CATALYZED_GHODIUM_ALKALIDE
];

COLORS_ALL = [
    COLOR_RED,
    COLOR_PURPLE,
    COLOR_BLUE,
    COLOR_CYAN,
    COLOR_GREEN,
    COLOR_YELLOW,
    COLOR_ORANGE,
    COLOR_BROWN,
    COLOR_GREY,
    COLOR_WHITE
];
```

Prototypes

ConstructionSite

A site of a structure which is currently under construction. A construction site can be created using the 'Construct' button at the left of the game field or the `Room.createConstructionSite` method.

To build a structure on the construction site, give a worker creep some amount of energy and perform `Creep.build` action.

You can remove enemy construction sites by moving a creep on it.

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

myboolean

Whether this is your own construction site.

ownerobject

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

progressnumber

The current construction progress.

`progressTotal`number

The total construction progress needed for the structure to be built.

`structureType`string

One of the `STRUCTURE_*` constants.

`remove()`

Remove the construction site.

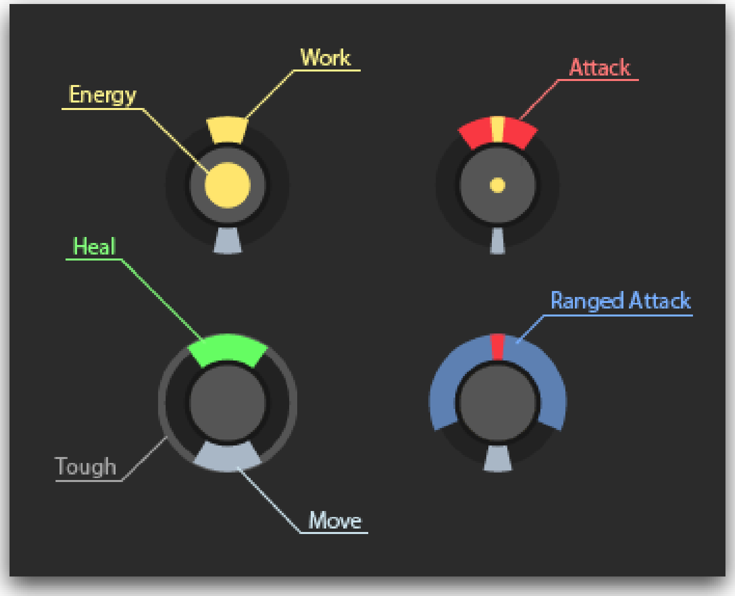
RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this construction site.

Creep

Creeps are your units. Creeps can move, harvest energy, construct structures, attack another creeps, and perform other actions. Each creep consists of up to 50 body parts with the following possible types:



Body part	Build cost	Effect per one body part
`MOVE`	50	Decreases fatigue by 2 points per tick.
`WORK`	100	Harvests 2 energy units from a source per tick. Harvests 1 mineral unit from a deposit per tick. Builds a structure for 5 energy units per tick. Repairs a structure for 100 hits per tick consuming 1 energy unit per tick. Dismantles a structure for 50 hits per tick returning 0.25 energy unit per tick. Upgrades a controller for 1 energy unit per tick.
`CARRY`	50	Can contain up to 50 resource units.
`ATTACK`	80	Attacks another creep/structure with 30 hits per tick in a short-ranged attack.
`RANGED_ATTACK`	150	Attacks another single creep/structure with 10 hits per tick in a long-range attack up to 3 squares long. Attacks all hostile creeps/structures within 3 squares range with 1-4-10 hits (depending on the range).
`HEAL`	250	Heals self or another creep restoring 12 hits per tick in short range or 4 hits per tick at a distance.
`CLAIM`	600	Claims a neutral room controller. Reserves a neutral room controller for 1 tick per body part. Attacks a hostile room controller downgrading its timer by 300 ticks per body parts. Attacks a neutral room controller reservation timer by 1 tick per body parts. A creep with this body part will have a reduced life time of 600 ticks and cannot be renewed.
`TOUGH`	10	No effect, just additional hit points to the creep's body. Can be boosted to resist damage.

Inherited from `RoomObject`

`posRoomPosition`

An object representing the position of this object in the room.

Inherited from `RoomObject`

`roomRoom`

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

bodyarray

An array describing the creep’s body. Each element contains the following properties:

parameter	type	description
`boost`	string | undefined	If the body part is boosted, this property specifies the mineral type which is used for boosting. One of the `RESOURCE_*` constants. Learn more
`type`	string	One of the body part types constants.
`hits`	number	The remaining amount of hit points of this body part.

carryobject

```
const total = _.sum(creep.carry);
```

An object with the creep's cargo contents. Each object key is one of the `RESOURCE_*` constants, values are resources amounts. `RESOURCE_ENERGY` is always defined and equals to 0 when empty, other resources are undefined when empty. You can use `lodash.sum` to get the total amount of contents:

carryCapacitynumber

The total amount of resources the creep can carry.

fatiguenumber

The movement fatigue indicator. If it is greater than zero, the creep cannot move.

hitsnumber

The current amount of hit points of the creep.

hitsMaxnumber

The maximum amount of hit points of the creep.

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

memoryany

```
creep.memory.task = 'building';
```

A shorthand to `Memory.creeps[creep.name]`. You can use it for quick access the creep's specific memory data object. Learn more about memory

myboolean

Whether it is your creep or foe.

namestring

Creep's name. You can choose the name while creating a new creep, and it cannot be changed later. This name is a hash key to access the creep via the Game.creeps object.

ownerobject

An object with the creep's owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

sayingstring

The text message that the creep was saying at the last tick.

spawningboolean

Whether this creep is still being spawned.

ticksToLivenumber

The remaining amount of game ticks after which the creep will die.

attack(target)

```
const target = creep.pos.findClosestByRange(FIND_HOSTILE_CREEPS);
if(target) {
    if(creep.attack(target) == ERR_NOT_IN_RANGE) {
        creep.moveTo(target);
    }
}
```

Attack another creep or structure in a short-ranged attack. Requires the `ATTACK` body part. If the target is inside a rampart, then the rampart is attacked instead. The target has to be at adjacent square to the creep. If the target is a creep with `ATTACK` body parts and is not inside a rampart, it will automatically hit back at the attacker.

parameter	type	description
`target`	Creep, Structure	The target object to be attacked.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_TARGET`	-7	The target is not a valid attackable object.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.

constant	value	description
`ERR_NO_BODYPART`	-12	There are no `ATTACK` body parts in this creep’s body.

`attackController(target)`

```
if(creep.room.controller && !creep.room.controller.my) {
    if(creep.attackController(creep.room.controller) == ERR_NOT_IN_RANGE) {
        creep.moveTo(creep.room.controller);
    }
}
```

Decreases the controller's downgrade timer by 300 ticks per every `CLAIM` body part, or reservation timer by 1 tick per every `CLAIM` body part. If the controller under attack is owned, it cannot be upgraded or attacked again for the next 1,000 ticks. The target has to be at adjacent square to the creep.

parameter	type	description
`target`	Structure	The target controller object.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_TARGET`	-7	The target is not a valid owned or reserved controller object.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_TIRED`	-11	You have to wait until the next attack is possible.
`ERR_NO_BODYPART`	-12	There are not enough `CLAIM` body parts in this creep’s body.

build(target)

```
const target = creep.pos.findClosestByRange(FIND_CONSTRUCTION_SITES);
if(target) {
    if(creep.build(target) == ERR_NOT_IN_RANGE) {
        creep.moveTo(target);
    }
}
```

Build a structure at the target construction site using carried energy. Requires `WORK` and `CARRY` body parts. The target has to be within 3 squares range of the creep.

parameter	type	description
`target`	ConstructionSite	The target construction site to be built.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The creep does not have any carried energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid construction site object or the structure cannot be built here (probably because of a creep at the same square).
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_NO_BODYPART`	-12	There are no `WORK` body parts in this creep’s body.
`ERR_RCL_NOT_ENOUGH`	-14	Room Controller Level insufficient. Learn more

cancelOrder(methodName)

```
creep.move(LEFT);
creep.cancelOrder('move');
//The creep will not move in this game tick
```

Cancel the order given during the current game tick.

parameter	type	description
`methodName`	string	The name of a creep's method to be cancelled.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been cancelled successfully.
`ERR_NOT_FOUND`	-5	The order with the specified name is not found.

claimController(target)

```
if(creep.room.controller) {
    if(creep.claimController(creep.room.controller) == ERR_NOT_IN_RANGE) {
        creep.moveTo(creep.room.controller);
    }
}
```

Claims a neutral controller under your control. Requires the `CLAIM` body part. The target has to be at adjacent square to the creep. You need to have the corresponding Global Control Level in order to claim a new room. If you don't have enough GCL, consider reserving this room instead. Learn more

parameter	type	description
`target`	StructureController	The target controller object.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.

constant	value	description
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_TARGET`	-7	The target is not a valid neutral controller object.
`ERR_FULL`	-8	You cannot claim more than 3 rooms in the Novice Area.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_NO_BODYPART`	-12	There are no `CLAIM` body parts in this creep's body.
`ERR_GCL_NOT_ENOUGH`	-15	Your Global Control Level is not enough.

dismantle(target)

```
const target = creep.pos.findClosestByRange(FIND_STRUCTURES,
    {filter: {structureType: STRUCTURE_WALL}});
if(target) {
    if(creep.dismantle(target) == ERR_NOT_IN_RANGE) {
        creep.moveTo(target);
    }
}
```

Dismantles any (even hostile) structure returning 50% of the energy spent on its repair. Requires the `WORK` body part. If the creep has an empty `CARRY` body part, the energy is put into it; otherwise it is dropped on the ground. The target has to be at adjacent square to the creep.

parameter	type	description
`target`	Structure	The target structure.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.

constant	value	description
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_TARGET`	-7	The target is not a valid creep object.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_NO_BODYPART`	-12	There are no `WORK` body parts in this creep’s body.

drop(resourceType, [amount])

```
creep.drop(RESOURCE_ENERGY);
```

```
// drop all resources
for(const resourceType in creep.carry) {
    creep.drop(resourceType);
}
```

Drop this resource on the ground.

parameter	type	description
`resourceType`	string	One of the `RESOURCE_*` constants.
`amount` *optional*	number	The amount of resource units to be dropped. If omitted, all the available carried amount is used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.

constant	value	description
ERR_NOT_ENOUGH_RESOURCES	-6	The creep does not have the given amount of energy.

generateSafeMode(controller)

```
if(creep.generateSafeMode(creep.room.controller) == ERR_NOT_IN_RANGE) {
    creep.moveTo(creep.room.controller);
}
```

Add one more available safe mode activation to a room controller. The creep has to be at adjacent square to the target room controller and have 1000 ghodium resource.

parameter	type	description
target	StructureController	The target room controller.

RETURN VALUE

One of the following codes:

constant	value	description
OK	0	The operation has been scheduled successfully.
ERR_NOT_OWNER	-1	You are not the owner of this creep.
ERR_BUSY	-4	The creep is still being spawned.
ERR_NOT_ENOUGH_RESOURCES	-6	The creep does not have enough ghodium.
ERR_INVALID_TARGET	-7	The target is not a valid controller object.
ERR_NOT_IN_RANGE	-9	The target is too far away.

getActiveBodyparts(type)

```
const target = creep.pos.findClosestByRange(FIND_HOSTILE_CREEPS, {
    filter: function(object) {
        return object.getActiveBodyparts(ATTACK) == 0;
    }
});
if(target) {
    creep.moveTo(target);
}
```

Get the quantity of live body parts of the given type. Fully damaged parts do not count.

parameter	type	description
`type`	string	A body part type, one of the following body part constants: • `MOVE` • `WORK` • `CARRY` • `ATTACK` • `RANGED_ATTACK` • `HEAL` • `TOUGH`

RETURN VALUE

A number representing the quantity of body parts.

`harvest(target)`

```
const target = creep.pos.findClosestByRange(FIND_SOURCES_ACTIVE);
if(target) {
    if(creep.harvest(target) == ERR_NOT_IN_RANGE) {
        creep.moveTo(target);
    }
}
```

Harvest energy from the source or minerals from the mineral deposit. Requires the `WORK` body part. If the creep has an empty `CARRY` body part, the harvested resource is put into it; otherwise it is dropped on the ground. The target has to be at an adjacent square to the creep.

parameter	type	description
`target`	Source, Mineral	The object to be harvested.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.

constant	value	description
`ERR_NOT_OWNER`	-1	You are not the owner of this creep, or the room controller is owned or reserved by another player.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_NOT_FOUND`	-5	Extractor not found. You must build an extractor structure to harvest minerals. Learn more
`ERR_NOT_ENOUGH_RESOURCES`	-6	The target source does not contain any harvestable energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid source object.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_NO_BODYPART`	-12	There are no `WORK` body parts in this creep’s body.

heal(target)

```
const target = creep.pos.findClosestByRange(FIND_MY_CREEPS, {
    filter: function(object) {
        return object.hits < object.hitsMax;
    }
});
if(target) {
    if(creep.heal(target) == ERR_NOT_IN_RANGE) {
        creep.moveTo(target);
    }
}
```

Heal self or another creep. It will restore the target creep’s damaged body parts function and increase the hits counter. Requires the `HEAL` body part. The target has to be at adjacent square to the creep.

parameter	type	description
`target`	Creep	The target creep object.

RETURN VALUE

One of the following codes:

constant	value	description

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_TARGET`	-7	The target is not a valid creep object.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_NO_BODYPART`	-12	There are no `HEAL` body parts in this creep’s body.

move(direction)

```
creep.move(RIGHT);
```

```
const path = creep.pos.findPathTo(Game.flags.Flag1);
if(path.length > 0) {
    creep.move(path[0].direction);
}
```

Move the creep one square in the specified direction. Requires the `MOVE` body part.

parameter	type	description
`direction`	number	One of the following constants: • `TOP` • `TOP_RIGHT` • `RIGHT` • `BOTTOM_RIGHT` • `BOTTOM` • `BOTTOM_LEFT` • `LEFT` • `TOP_LEFT`

RETURN VALUE

One of the following codes:

constant	value	description

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_ARGS`	-10	The provided direction is incorrect.
`ERR_TIRED`	-11	The fatigue indicator of the creep is non-zero.
`ERR_NO_BODYPART`	-12	There are no MOVE body parts in this creep’s body.

moveByPath(path)

```
const path = spawn.room.findPath(spawn, source);
creep.moveByPath(path);
```

```
if(!creep.memory.path) {
    creep.memory.path = creep.pos.findPathTo(target);
}
creep.moveByPath(creep.memory.path);
```

Move the creep using the specified predefined path. Requires the `MOVE` body part.

parameter	type	description
`path`	array|string	A path value as returned from `Room.findPath`, `RoomPosition.findPathTo`, or `PathFinder.search` methods. Both array form and serialized string form are accepted.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.

constant	value	description
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_NOT_FOUND`	-5	The specified path doesn't match the creep's location.
`ERR_INVALID_ARGS`	-10	`path` is not a valid path array.
`ERR_TIRED`	-11	The fatigue indicator of the creep is non-zero.
`ERR_NO_BODYPART`	-12	There are no `MOVE` body parts in this creep's body.

moveTo(x, y, [opts])
(target, [opts])

```
creep.moveTo(10, 20);
```

```
creep.moveTo(Game.flags.Flag1);
```

```
creep.moveTo(new RoomPosition(25, 20, 'W10N5'));
```

```
creep.moveTo(pos, {reusePath: 50});
```

```
// Execute moves by cached paths at first
for(const name in Game.creeps) {
    Game.creeps[name].moveTo(target, {noPathFinding: true});
}

// Perform pathfinding only if we have enough CPU
if(Game.cpu.tickLimit - Game.cpu.getUsed() > 20) {
    for(const name in Game.creeps) {
        Game.creeps[name].moveTo(target);
    }
}
```

Find the optimal path to the target within the same room and move to it. A shorthand to consequent calls of pos.findPathTo() and move() methods. If the target is in another room, then the corresponding exit will be used as a target. Requires the `MOVE` body part.

parameter	type	description
`x`	number	X position of the target in the same room.

parameter	type	description
`y`	number	Y position of the target in the same room.
`target`	object	Can be a RoomPosition object or any object containing RoomPosition. The position doesn't have to be in the same room with the creep.
`opts` *optional*	object	An object containing additional options: • reusePath number This option enables reusing the path found along multiple game ticks. It allows to save CPU time, but can result in a slightly slower creep reaction behavior. The path is stored into the creep's memory to the `_move` property. The `reusePath` value defines the amount of ticks which the path should be reused for. The default value is 5. Increase the amount to save more CPU, decrease to make the movement more consistent. Set to 0 if you want to disable path reusing. • serializeMemory boolean If `reusePath` is enabled and this option is set to true, the path will be stored in memory in the short serialized form using `Room.serializePath`. The default value is true. • noPathFinding boolean If this option is set to true, `moveTo` method will return `ERR_NOT_FOUND` if there is no memorized path to reuse. This can significantly save CPU time in some cases. The default value is false. • visualizePathStyle object Draw a line along the creep’s path using `RoomVisual.poly`. You can provide either an empty object or custom style parameters. The default style is equivalent to: `{ fill: 'transparent', stroke: '#fff', lineStyle: 'dashed', strokeWidth: .15, opacity: .1 }` • Any options supported by `Room.findPath` method.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_NO_PATH`	-2	No path to the target could be found.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_TARGET`	-7	The target provided is invalid.

constant	value	description
`ERR_TIRED`	-11	The fatigue indicator of the creep is non-zero.
`ERR_NO_BODYPART`	-12	There are no MOVE body parts in this creep’s body.

notifyWhenAttacked(enabled)

```
if(creep.memory.role == 'scout') {
    creep.notifyWhenAttacked(false);
}
else {
    creep.notifyWhenAttacked(true);
}
```

Toggle auto notification when the creep is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

pickup(target)

```
const target = creep.pos.findClosestByRange(FIND_DROPPED_ENERGY);
if(target) {
    if(creep.pickup(target) == ERR_NOT_IN_RANGE) {
        creep.moveTo(target);
    }
}
```

Pick up an item (a dropped piece of energy). Requires the `CARRY` body part. The target has to be at adjacent square to the creep or at the same square.

parameter	type	description
`target`	Resource	The target object to be picked up.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_TARGET`	-7	The target is not a valid object to pick up.
`ERR_FULL`	-8	The creep cannot receive any more energy.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.

rangedAttack(target)

```
const targets = creep.pos.findInRange(FIND_HOSTILE_CREEPS, 3);
if(targets.length > 0) {
    creep.rangedAttack(targets[0]);
}
```

A ranged attack against another creep or structure. Requires the `RANGED_ATTACK` body part. If the target is inside a rampart, the rampart is attacked instead. The target has to be within 3 squares range of the creep.

parameter	type	description
`target`	Creep, Structure	The target object to be attacked.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_TARGET`	-7	The target is not a valid attackable object.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_NO_BODYPART`	-12	There are no `RANGED_ATTACK` body parts in this creep's body.

rangedHeal(target)

```
const target = creep.pos.findClosestByRange(FIND_MY_CREEPS, {
    filter: function(object) {
        return object.hits < object.hitsMax;
    }
});
if(target) {
    creep.moveTo(target);
    if(creep.pos.isNearTo(target)) {
        creep.heal(target);
    }
    else {
        creep.rangedHeal(target);
    }
}
```

Heal another creep at a distance. It will restore the target creep's damaged body parts function and increase the hits counter. Requires the `HEAL` body part. The target has to be within 3 squares range of the creep.

parameter	type	description
`target`	Creep	The target creep object.

RETURN VALUE

One of the following codes:

constant	value	description

constant	value	description
OK	0	The operation has been scheduled successfully.
ERR_NOT_OWNER	-1	You are not the owner of this creep.
ERR_BUSY	-4	The creep is still being spawned.
ERR_INVALID_TARGET	-7	The target is not a valid creep object.
ERR_NOT_IN_RANGE	-9	The target is too far away.
ERR_NO_BODYPART	-12	There are no HEAL body parts in this creep’s body.

rangedMassAttack()

```
const targets = creep.pos.findInRange(FIND_HOSTILE_CREEPS, 3);
if(targets.length > 0) {
    creep.rangedMassAttack();
}
```

A ranged attack against all hostile creeps or structures within 3 squares range. Requires the RANGED_ATTACK body part. The attack power depends on the range to each target. Friendly units are not affected.

RETURN VALUE

One of the following codes:

constant	value	description
OK	0	The operation has been scheduled successfully.
ERR_NOT_OWNER	-1	You are not the owner of this creep.
ERR_BUSY	-4	The creep is still being spawned.
ERR_NO_BODYPART	-12	There are no RANGED_ATTACK body parts in this creep’s body.

repair(target)

```
const targets = creep.room.find(FIND_STRUCTURES, {
    filter: object => object.hits < object.hitsMax
});

targets.sort((a,b) => a.hits - b.hits);

if(targets.length > 0) {
    if(creep.repair(targets[0]) == ERR_NOT_IN_RANGE) {
        creep.moveTo(targets[0]);
    }
}
```

Repair a damaged structure using carried energy. Requires the `WORK` and `CARRY` body parts. The target has to be within 3 squares range of the creep.

parameter	type	description
`target`	Structure	The target structure to be repaired.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The creep does not carry any energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid creep object.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_NO_BODYPART`	-12	There are no `WORK` body parts in this creep’s body.

reserveController(target)

```
if(creep.room.controller) {
    if(creep.reserveController(creep.room.controller) == ERR_NOT_IN_RANGE) {
        creep.moveTo(creep.room.controller);
    }
}
```

Temporarily block a neutral controller from claiming by other players and restore energy sources to their full capacity. Each tick, this command increases the counter of the period during which the controller is unavailable by 1 tick per each `CLAIM` body part. The maximum reservation period to maintain is 5,000 ticks. The target has to be at adjacent square to the creep.

parameter	type	description
`target`	StructureController	The target controller object to be reserved.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_TARGET`	-7	The target is not a valid neutral controller object.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_NO_BODYPART`	-12	There are no `CLAIM` body parts in this creep’s body.

say(message, [public])

```
const hostiles = creep.pos.findInRange(FIND_HOSTILE_CREEPS, 10);
if(hostiles.length > 0) {
    creep.say('OMG!😨');
    creep.moveTo(Game.spawns['Spawn1']);
}
else {
    doWork(creep);
}
```

Display a visual speech balloon above the creep with the specified message. The message will be available for one tick. You can read the last message using the `saying` property. Any valid Unicode characters are allowed, including emoji.

parameter	type	description
`message`	string	The message to be displayed. Maximum length is 10 characters.
`public` *optional*	boolean	Set to true to allow other players to see this message. Default is false.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.

signController(target, text)

```
if(creep.room.controller) {
    if(creep.signController(creep.room.controller, "I'm going to claim this room in a few days. I warned ya!") == ERR_NOT_IN_RANGE) {
        creep.moveTo(creep.room.controller);
    }
}
```

Sign a controller with an arbitrary text visible to all players. This text will appear in the room UI, in the world map, and can be accessed via the API. You can sign unowned and hostile controllers. The target has to be at adjacent square to the creep. Pass an empty string to remove the sign.

parameter	type	description
`target`	StructureController	The target controller object to be signed.
`text`	string	The sign text. The string is cut off after 100 characters.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.

constant	value	description
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_INVALID_TARGET`	-7	The target is not a valid controller object.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.

suicide()

Kill the creep immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.

transfer(target, resourceType, [amount])

```
if(creep.transfer(storage, RESOURCE_ENERGY) == ERR_NOT_IN_RANGE) {
    creep.moveTo(storage);
}
```

```
// transfer all resources
for(const resourceType in creep.carry) {
    creep.transfer(storage, resourceType);
}
```

Transfer resource from the creep to another object. The target has to be at adjacent square to the creep.

parameter	type	description
`target`	Creep, Structure	The target object.

parameter	type	description
`resourceType`	string	One of the `RESOURCE_*` constants.
`amount` *optional*	number	The amount of resources to be transferred. If omitted, all the available carried amount is used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The creep does not have the given amount of resources.
`ERR_INVALID_TARGET`	-7	The target is not a valid object which can contain the specified resource.
`ERR_FULL`	-8	The target cannot receive any more resources.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_INVALID_ARGS`	-10	The resources amount is incorrect.

upgradeController(target)

```
if(creep.room.controller) {
    if(creep.upgradeController(creep.room.controller) == ERR_NOT_IN_RANGE) {
        creep.moveTo(creep.room.controller);
    }
}
```

Upgrade your controller to the next level using carried energy. Upgrading controllers raises your Global Control Level in parallel. Requires `WORK` and `CARRY` body parts. The target has to be within 3 squares range of the creep.

A fully upgraded level 8 controller can't be upgraded over 15 energy units per tick regardless of creeps abilities. The cumulative effect of all the creeps performing `upgradeController` in the current tick is taken into account. This limit can be increased by using ghodium mineral boost.

Upgrading the controller raises its `ticksToDowngrade` timer by 100. The timer must be full in order for controller to be levelled up.

parameter	type	description
`target`	StructureController	The target controller object to be upgraded.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep or the target controller.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The creep does not have any carried energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid controller object.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_NO_BODYPART`	-12	There are no `WORK` body parts in this creep’s body.

withdraw(target, resourceType, [amount])

```
if(creep.withdraw(storage, RESOURCE_ENERGY) == ERR_NOT_IN_RANGE) {
    creep.moveTo(storage);
}
```

Withdraw resources from a structure. The target has to be at adjacent square to the creep. Multiple creeps can withdraw from the same structure in the same tick. Your creeps can withdraw resources from hostile structures as well, in case if there is no hostile rampart on top of it.

parameter	type	description
`target`	Structure	The target object.

parameter	type	description
`resourceType`	string	One of the `RESOURCE_*` constants.
`amount` *optional*	number	The amount of resources to be transferred. If omitted, all the available amount is used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this creep, or there is a hostile rampart on top of the target.
`ERR_BUSY`	-4	The creep is still being spawned.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The target does not have the given amount of resources.
`ERR_INVALID_TARGET`	-7	The target is not a valid object which can contain the specified resource.
`ERR_FULL`	-8	The creep's carry is full.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_INVALID_ARGS`	-10	The resource amount or type is incorrect.

Flag

A flag. Flags can be used to mark particular spots in a room. Flags are visible to their owners only. You cannot have more than 10,000 flags.

Inherited from `RoomObject`

`posRoomPosition`

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

colornumber

Flag primary color. One of the `COLOR_*` constants.

memoryany

A shorthand to `Memory.flags[flag.name]`. You can use it for quick access the flag's specific memory data object.

namestring

Flag’s name. You can choose the name while creating a new flag, and it cannot be changed later. This name is a hash key to access the flag via the Game.flags object. The maximum name length is 60 charactes.

secondaryColornumber

Flag secondary color. One of the `COLOR_*` constants.

remove()

Remove the flag.

RETURN VALUE

Always returns OK .

setColor(color, [secondaryColor])

```
Game.flags.Flag1.setColor(COLOR_GREEN, COLOR_WHITE);
```

Set new color of the flag.

parameter	type	description
`color`	string	Primary color of the flag. One of the `COLOR_*` constants.
`secondaryColor` *optional*	string	Secondary color of the flag. One of the `COLOR_*` constants.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_INVALID_ARGS`	-10	`color` is not a valid color constant.

setPosition(x,y)
(pos)

```
Game.flags.Flag1.setPosition(10,20);
```

```
Game.flags.Flag1.setPosition( new RoomPosition(10, 20, 'W3S5') );
```

Set new position of the flag.

parameter	type	description
`x`	number	The X position in the room.
`y`	number	The Y position in the room.
`pos`	object	Can be a RoomPosition object or any object containing RoomPosition.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.

constant	value	description
`ERR_INVALID_TARGET`	-7	The target provided is invalid.

Mineral

A mineral deposit. Can be harvested by creeps with a `WORK` body part using the extractor structure. Learn more about minerals from this article.

Regeneration amount	`DENSITY_LOW`: 15,000 `DENSITY_MODERATE`: 35,000 `DENSITY_HIGH`: 70,000 `DENSITY_ULTRA`: 100,000
Regeneration time	50,000 ticks
Density change probability	`DENSITY_LOW`: 100% chance `DENSITY_MODERATE`: 5% chance `DENSITY_HIGH`: 5% chance `DENSITY_ULTRA`: 100% chance

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

densitynumber

The density that this mineral deposit will be refilled to once `ticksToRegeneration` reaches 0. This is one of the `DENSITY_*` constants.

mineralAmountnumber

The remaining amount of resources.

mineralTypestring

The resource type, one of the `RESOURCE_*` constants.

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

ticksToRegenerationnumber

The remaining time after which the deposit will be refilled.

Nuke

A nuke landing position. This object cannot be removed or modified. You can find incoming nukes in the room using the `FIND_NUKES` constant.

Landing time	50,000 ticks
Effect	All creeps, construction sites and dropped resources in the room are removed immediately, even inside ramparts. Damage to structures: • 10,000,000 hits at the landing position; • 5,000,000 hits to all structures in 5x5 area. Note that you can stack multiple nukes from different rooms at the same target position to increase damage. If the room is in safe mode, then the safe mode is cancelled immediately, and the safe mode cooldown is reset to 0. The room controller is hit by triggering `upgradeBlocked` period, which means it is unavailable to activate safe mode again within the next 200 ticks.

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

launchRoomNamestring

The name of the room where this nuke has been launched from.

timeToLandnumber

The remaining landing time.

OwnedStructure

The base prototype for a structure that has an owner. Such structures can be found using `FIND_MY_STRUCTURES` and `FIND_HOSTILE_STRUCTURES` constants.

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

`hitsMax`number

The total amount of hit points of the structure.

Inherited from `Structure`

`id`string

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

`structureType`string

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

`destroy()`

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

`isActive()`

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

`myboolean`

Whether this is your own structure.

`ownerobject`

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

PathFinder.CostMatrix

Container for custom navigation cost data. By default `PathFinder` will only consider terrain data (plain, swamp, wall) — if you need to route around obstacles such as buildings or creeps you must put them into a `CostMatrix`. Generally you will create your `CostMatrix` from within `roomCallback`. If a non-0 value is found in a room's CostMatrix then that value will be used instead of the default terrain cost. You should avoid using large values in your CostMatrix and terrain cost flags. For example, running `PathFinder.search` with `{ plainCost: 1, swampCost: 5 }` is faster than running it with `{plainCost: 2, swampCost: 10 }` even though your paths will be the same.

constructor()

```
let costs = new PathFinder.CostMatrix;
```

Creates a new CostMatrix containing 0's for all positions.

set(x, y, cost)

```
let costs = new PathFinder.CostMatrix;
let pos = Game.spawns['Spawn1'].pos;
costs.set(pos.x, pos.y, 255); // Can't walk over a building
```

Set the cost of a position in this CostMatrix.

parameter	type	description
`x`	number	X position in the room.
`y`	number	Y position in the room.
`cost`	number	Cost of this position. Must be a whole number. A cost of 0 will use the terrain cost for that tile. A cost greater than or equal to 255 will be treated as unwalkable.

get(x, y)

Get the cost of a position in this CostMatrix.

parameter	type	description
`x`	number	X position in the room.
`y`	number	Y position in the room.

clone()

Copy this CostMatrix into a new CostMatrix with the same data.

RETURN VALUE

A new CostMatrix instance.

serialize()

```
let costs = new PathFinder.CostMatrix;
Memory.savedMatrix = costs.serialize();
```

Returns a compact representation of this CostMatrix which can be stored via `JSON.stringify`.

RETURN VALUE

An array of numbers. There's not much you can do with the numbers besides store them for later.

PathFinder.CostMatrix.deserialize(val)

```
let costs = PathFinder.CostMatrix.deserialize(Memory.savedMatrix)
```

Static method which deserializes a new CostMatrix using the return value of `serialize`.

parameter	type	description
`val`	object	Whatever `serialize` returned

RETURN VALUE

Returns new `CostMatrix` instance.

Resource

A dropped piece of resource. It will decay after a while if not picked up. Dropped resource pile decays for `ceil(amount/1000)` units per tick.

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

amountnumber

The amount of resource units containing.

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

resourceTypestring

One of the `RESOURCE_*` constants.

Room

An object representing the room in which your units and structures are in. It can be used to look around, find paths, etc. Every `RoomObject` in the room contains its linked `Room` instance in the `room` property.

controllerStructureController

The Controller structure of this room, if present, otherwise undefined.

energyAvailablenumber

Total amount of energy available in all spawns and extensions in the room.

energyCapacityAvailablenumber

Total amount of `energyCapacity` of all spawns and extensions in the room.

memoryany

```
room.memory.stage = 2;
```

A shorthand to `Memory.rooms[room.name]`. You can use it for quick access the room’s specific memory data object. Learn more about memory

namestring

The name of the room.

storageStructureStorage

The Storage structure of this room, if present, otherwise undefined.

terminalStructureTerminal

The Terminal structure of this room, if present, otherwise undefined.

visualRoomVisual

A RoomVisual object for this room. You can use this object to draw simple shapes (lines, circles, text labels) in the room.

Room.serializePath(path)

```
const path = spawn.pos.findPathTo(source);
Memory.path = Room.serializePath(path);
creep.moveByPath(Memory.path);
```

Serialize a path array into a short string representation, which is suitable to store in memory.

parameter	type	description
`path`	array	A path array retrieved from `Room.findPath`.

RETURN VALUE

A serialized string form of the given path.

Room.deserializePath(path)

```
const path = Room.deserializePath(Memory.path);
creep.moveByPath(path);
```

Deserialize a short string path representation into an array form.

parameter	type	description
`path`	string	A serialized path string.

RETURN VALUE

A path array.

createConstructionSite(x, y, structureType, [name]) (pos, structureType, [name])

```
Game.rooms.sim.createConstructionSite(10, 15, STRUCTURE_ROAD);
```

```
Game.rooms.sim.createConstructionSite(10, 15, STRUCTURE_SPAWN,
    'MySpawn2');
```

Create new ConstructionSite at the specified location.

parameter	type	description
`x`	number	The X position.
`y`	number	The Y position.
`pos`	object	Can be a RoomPosition object or any object containing RoomPosition.
`structureType`	string	One of the `STRUCTURE_*` constants.
`name` *optional*	string	The name of the structure, for structures that support it (currently only spawns).

RETURN VALUE

One of the following codes:

constant	value	description

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_INVALID_TARGET`	-7	The structure cannot be placed at the specified location.
`ERR_FULL`	-8	You have too many construction sites. The maximum number of construction sites per player is 100.
`ERR_INVALID_ARGS`	-10	The location is incorrect.
`ERR_RCL_NOT_ENOUGH`	-14	Room Controller Level insufficient. Learn more

createFlag(x, y, [name], [color], [secondaryColor])
(pos, [name], [color], [secondaryColor])

```
Game.rooms.sim.createFlag(5, 12, 'Flag1');
```

Create new Flag at the specified location.

parameter	type	description
`x`	number	The X position.
`y`	number	The Y position.
`pos`	object	Can be a RoomPosition object or any object containing RoomPosition.
`name` *optional*	string	The name of a new flag. It should be unique, i.e. the `Game.flags` object should not contain another flag with the same name (hash key). If not defined, a random name will be generated. The maximum length is 60 characters.
`color` *optional*	string	The color of a new flag. Should be one of the `COLOR_*` constants. The default value is `COLOR_WHITE`.
`secondaryColor` *optional*	string	The secondary color of a new flag. Should be one of the `COLOR_*` constants. The default value is equal to `color`.

RETURN VALUE

The name of a new flag, or one of the following error codes:

constant	value	description
`ERR_NAME_EXISTS`	-3	There is a flag with the same name already.
`ERR_INVALID_ARGS`	-10	The location or the color constant is incorrect.

find(type, [opts])

```
const targets = creep.room.find(FIND_DROPPED_RESOURCES);
if(targets.length) {
    creep.moveTo(targets[0]);
    creep.pickup(targets[0]);
}
```

```
const extensions = Game.spawns['Spawn1'].room.find(FIND_MY_STRUCTURES, {
    filter: { structureType: STRUCTURE_EXTENSION }
});
console.log('Spawn has '+extensions.length+' extensions available');
```

```
const targets = creep.room.find(FIND_HOSTILE_CREEPS, {
    filter: function(object) {
        return object.getActiveBodyparts(ATTACK) == 0;
    }
});
```

Find all objects of the specified type in the room.

parameter	type	description
`type`	number	One of the `FIND_*` constants.
`opts` *optional*	object	An object with additional options: • filter object, function, string The result list will be filtered using the Lodash.filter method.

RETURN VALUE

An array with the objects found.

findExitTo(room)

```
const exitDir = creep.room.findExitTo(anotherCreep.room);
const exit = creep.pos.findClosestByRange(exitDir);
creep.moveTo(exit);
```

```
// or simply:
creep.moveTo(anotherCreep);
creep.moveTo(new RoomPosition(25,25, anotherCreep.pos.roomName));
```

Find the exit direction en route to another room. Please note that this method is not required for inter-room movement, you can simply pass the target in another room into `Creep.moveTo` method.

parameter	type	description
`room`	string, Room	Another room name or room object.

RETURN VALUE

The room direction constant, one of the following:

- `FIND_EXIT_TOP`
- `FIND_EXIT_RIGHT`
- `FIND_EXIT_BOTTOM`
- `FIND_EXIT_LEFT`

Or one of the following error codes:

constant	value	description
`ERR_NO_PATH`	-2	Path can not be found.
`ERR_INVALID_ARGS`	-10	The location is incorrect.

findPath(fromPos, toPos, [opts])

```
const path = creep.room.findPath(creep.pos, targetPos);
creep.move(path[0].direction);
```

```
PathFinder.use(true);
const path = creep.room.findPath(creep.pos, targetPos, {
    costCallback: function(roomName, costMatrix) {
        if(roomName == 'W1N5') {
            // set anotherCreep's location as walkable
            costMatrix.set(anotherCreep.pos.x, anotherCreep.pos.y, 0);
            // set flag location as an obstacle
            costMatrix.set(flag.pos.x, flag.pos.y, 255);
            // increase cost for (25,20) location to 50
            costMatrix.set(25, 20, 50);
        }
    }
});
```

```
let path = creep.room.findPath(creep.pos, targetPos, {maxOps: 200});
if( !path.length || !targetPos.isEqualTo(path[path.length - 1]) ) {
    path = creep.room.findPath(creep.pos, targetPos, {
        maxOps: 1000, ignoreDestructibleStructures: true
    });
}
if( path.length ) {
    creep.move(path[0].direction);
}
```

Find an optimal path inside the room between fromPos and toPos using Jump Point Search algorithm.

parameter	type	description
`fromPos`	RoomPosition	The start position.
`toPos`	RoomPosition	The end position.

parameter	type	description
`opts` *optional*	object	An object containing additonal pathfinding flags: - ignoreCreeps boolean Treat squares with creeps as walkable. Can be useful with too many moving creeps around or in some other cases. The default value is false. - ignoreDestructibleStructures boolean Treat squares with destructible structures (constructed walls, ramparts, spawns, extensions) as walkable. The default value is false. - ignoreRoads boolean Ignore road structures. Enabling this option can speed up the search. The default value is false. This is only used when the new `PathFinder` is enabled. - costCallback function(string, CostMatrix) You can use this callback to modify a `CostMatrix` for any room during the search. The callback accepts two arguments, `roomName` and `costMatrix`. Use the `costMatrix` instance to make changes to the positions costs. If you return a new matrix from this callback, it will be used instead of the built-in cached one. This option is only used when the new `PathFinder` is enabled. - ignore array An array of the room's objects or RoomPosition objects which should be treated as walkable tiles during the search. This option cannot be used when the new `PathFinder` is enabled (use `costCallback` option instead). - avoid array An array of the room's objects or RoomPosition objects which should be treated as obstacles during the search. This option cannot be used when the new `PathFinder` is enabled (use `costCallback` option instead). - maxOps number The maximum limit of possible pathfinding operations. You can limit CPU time used for the search based on ratio 1 op ~ 0.001 CPU. The default value is 2000. - heuristicWeight number Weight to apply to the heuristic in the A *formula* `F = G + weight` H. Use this option only if you understand the underlying A* algorithm mechanics! The default value is 1.2. - serialize boolean If true, the result path will be serialized using `Room.serializePath`. The default is false. - maxRooms number The maximum allowed rooms to search. The default (and maximum) is 16. This is only used when the new `PathFinder` is enabled. - range number Find a path to a position in specified linear range of target. The default is 0. - plainCost number Cost for walking on plain positions. The default is 1. - swampCost number Cost for walking on swamp positions. The default is 5.

RETURN VALUE

An array with path steps in the following format:

```
[
    { x: 10, y: 5, dx: 1,  dy: 0, direction: RIGHT },
    { x: 10, y: 6, dx: 0,  dy: 1, direction: BOTTOM },
    { x: 9,  y: 7, dx: -1, dy: 1, direction: BOTTOM_LEFT },
    ...
]
```

getPositionAt(x, y)

```
const pos = Game.rooms.sim.getPositionAt(5,12);
const source = pos.findClosestByRange(FIND_SOURCES_ACTIVE);
```

Creates a RoomPosition object at the specified location.

parameter	type	description
x	number	The X position.
y	number	The Y position.

RETURN VALUE

A RoomPosition object or null if it cannot be obtained.

lookAt(x, y)
(target)

```
const look = creep.room.lookAt(target);
look.forEach(function(lookObject) {
    if(lookObject.type == LOOK_CREEPS &&
       lookObject[LOOK_CREEPS].getActiveBodyparts(ATTACK) == 0) {
        creep.moveTo(lookObject.creep);
    }
});
```

Get the list of objects at the specified room position.

parameter	type	description
x	number	X position in the room.
y	number	Y position in the room.
target	object	Can be a RoomPosition object or any object containing RoomPosition.

RETURN VALUE

An array with objects at the specified position in the following format:

```
[
    { type: 'creep', creep: {...} },
    { type: 'structure', structure: {...} },
    ...
    { type: 'terrain', terrain: 'swamp' }
]
```

lookAtArea(top, left, bottom, right, [asArray])

```
const look = creep.room.lookAtArea(10,5,11,7);
```

Get the list of objects at the specified room area.

parameter	type	description
`top`	number	The top Y boundary of the area.
`left`	number	The left X boundary of the area.
`bottom`	number	The bottom Y boundary of the area.
`right`	number	The right X boundary of the area.
`asArray` *optional*	boolean	Set to true if you want to get the result as a plain array.

RETURN VALUE

If `asArray` is set to false or undefined, the method returns an object with all the objects in the specified area in the following format:

```
// 10,5,11,7

{
    10: {
        5: [{ type: 'creep', creep: {...} },
            { type: 'terrain', terrain: 'swamp' }],
        6: [{ type: 'terrain', terrain: 'swamp' }],
        7: [{ type: 'terrain', terrain: 'swamp' }]
    },
    11: {
        5: [{ type: 'terrain', terrain: 'plain' }],
        6: [{ type: 'structure', structure: {...} },
            { type: 'terrain', terrain: 'swamp' }],
        7: [{ type: 'terrain', terrain: 'wall' }]
    }
}
```

If `asArray` is set to true, the method returns an array in the following format:

```
[
    {x: 5, y: 10, type: 'creep', creep: {...}},
    {x: 5, y: 10, type: 'terrain', terrain: 'swamp'},
    {x: 6, y: 10, type: 'terrain', terrain: 'swamp'},
    {x: 7, y: 10, type: 'terrain', terrain: 'swamp'},
    {x: 5, y: 11, type: 'terrain', terrain: 'plain'},
    {x: 6, y: 11, type: 'structure', structure: {...}},
    {x: 6, y: 11, type: 'terrain', terrain: 'swamp'},
    {x: 7, y: 11, type: 'terrain', terrain: 'wall'}
]
```

lookForAt(type, x, y)
(type, target)

```
const found = creep.room.lookForAt(LOOK_CREEPS, target);
if(found.length && found[0].getActiveBodyparts(ATTACK) == 0) {
    creep.moveTo(found[0]);
}
```

Get an object with the given type at the specified room position.

parameter	type	description
`type`	string	One of the `LOOK_*` constants.
`x`	number	X position in the room.
`y`	number	Y position in the room.
`target`	object	Can be a RoomPosition object or any object containing RoomPosition.

RETURN VALUE

An array of objects of the given type at the specified position if found.

lookForAtArea(type, top, left, bottom, right, [asArray])

```
const look = creep.room.lookForAtArea(LOOK_STRUCTURES,10,5,11,7);
```

Get the list of objects with the given type at the specified room area.

parameter	type	description

parameter	type	description
`type`	string	One of the `LOOK_*` constants.
`top`	number	The top Y boundary of the area.
`left`	number	The left X boundary of the area.
`bottom`	number	The bottom Y boundary of the area.
`right`	number	The right X boundary of the area.
`asArray` *optional*	boolean	Set to true if you want to get the result as a plain array.

RETURN VALUE

If `asArray` is set to false or undefined, the method returns an object with all the objects of the given type in the specified area in the following format:

```
// 10,5,11,7

{
    10: {
        5: [{...}],
        6: undefined,
        7: undefined
    },
    11: {
        5: undefined,
        6: [{...}, {...}],
        7: undefined
    }
}
```

If `asArray` is set to true, the method returns an array in the following format:

```
[
    {x: 5, y: 10, structure: {...}},
    {x: 6, y: 11, structure: {...}},
    {x: 6, y: 11, structure: {...}}
]
```

RoomObject

Any object with a position in a room. Almost all game objects prototypes are derived from `RoomObject`.

posRoomPosition

An object representing the position of this object in the room.

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

RoomPosition

An object representing the specified position in the room. Every `RoomObject` in the room contains `RoomPosition` as the `pos` property. The position object of a custom location can be obtained using the `Room.getPositionAt` method or using the constructor.

constructor(x, y, roomName)

```
const pos = new RoomPosition(10, 25, 'sim');
```

You can create new `RoomPosition` object using its constructor.

parameter	type	description
`x`	number	X position in the room.
`y`	number	Y position in the room.
`roomName`	string	The room name.

roomNamestring

The name of the room.

xnumber

X position in the room.

ynumber

Y position in the room.

createConstructionSite(structureType, [name])

```
Game.flags['Flag1'].pos.createConstructionSite(STRUCTURE_ROAD);
```

```
Game.flags['Flag1'].pos.createConstructionSite(10, 15, STRUCTURE_SPAWN,
    'MySpawn2');
```

Create new ConstructionSite at the specified location.

parameter	type	description
`structureType`	string	One of the `STRUCTURE_*` constants.
`name` *optional*	string	The name of the structure, for structures that support it (currently only spawns).

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_INVALID_TARGET`	-7	The structure cannot be placed at the specified location.
`ERR_FULL`	-8	You have too many construction sites. The maximum number of construction sites per player is 100.
`ERR_INVALID_ARGS`	-10	The location is incorrect.
`ERR_RCL_NOT_ENOUGH`	-14	Room Controller Level insufficient. Learn more

createFlag([name], [color], [secondaryColor])

```
creep.pos.createFlag('Flag1');
```

Create new Flag at the specified location.

parameter	type	description
`name` *optional*	string	The name of a new flag. It should be unique, i.e. the `Game.flags` object should not contain another flag with the same name (hash key). If not defined, a random name will be generated.
`color` *optional*	string	The color of a new flag. Should be one of the `COLOR_*` constants. The default value is `COLOR_WHITE`.
`secondaryColor` *optional*	string	The secondary color of a new flag. Should be one of the `COLOR_*` constants. The default value is equal to `color`.

RETURN VALUE

The name of a new flag, or one of the following error codes:

constant	value	description
`ERR_NAME_EXISTS`	-3	There is a flag with the same name already.
`ERR_INVALID_ARGS`	-10	The location or the color constant is incorrect.

findClosestByPath(type, [opts])
(objects, [opts])

```
const target = creep.pos.findClosestByPath(FIND_MY_SPAWNS);
creep.moveTo(target);
```

```
const target = creep.pos.findClosestByPath(FIND_MY_SPAWNS, {maxOps: 500});
creep.moveTo(target);
```

```
const target = creep.pos.findClosestByPath(FIND_HOSTILE_CREEPS, {
    filter: function(object) {
        return object.getActiveBodyparts(ATTACK) == 0;
    }
});
```

```
const target = creep.pos.findClosestByPath(FIND_HOSTILE_CREEPS, {
    filter: { owner: { username: 'Invader' } }
});
```

```
const targets = [
    Game.creeps.John,
    Game.creeps.Mike,
    room.getPositionAt(10,10)
];
const closest = creep.pos.findClosestByPath(targets);
```

Find an object with the shortest path from the given position. Uses Jump Point Search algorithm and Dijkstra's algorithm.

parameter	type	description
`type`	number	See Room.find.
`objects`	array	An array of room's objects or RoomPosition objects that the search should be executed against.
`opts` *optional*	object	An object containing pathfinding options (see Room.findPath), or one of the following: • filter object, function, string Only the objects which pass the filter using the Lodash.filter method will be used. • algorithm string One of the following constants: ◦ `astar` is faster when there are relatively few possible targets; ◦ `dijkstra` is faster when there are a lot of possible targets or when the closest target is nearby. The default value is determined automatically using heuristics.

RETURN VALUE

The closest object if found, null otherwise.

findClosestByRange(type, [opts]) (objects, [opts])

```
const target = creep.pos.findClosestByRange(FIND_MY_SPAWNS);
creep.moveTo(target);
```

```
const target = creep.pos.findClosestByRange(FIND_HOSTILE_CREEPS, {
    filter: function(object) {
        return object.getActiveBodyparts(ATTACK) == 0;
    }
});
```

```
const target = creep.pos.findClosestByRange(FIND_HOSTILE_CREEPS, {
    filter: { owner: { username: 'Invader' } }
});
```

```
const targets = [
    Game.creeps.John,
    Game.creeps.Mike,
    room.getPositionAt(10,10)
];
const closest = creep.pos.findClosestByRange(targets);
```

Find an object with the shortest linear distance from the given position.

parameter	type	description
`type`	number	See Room.find.
`objects`	array	An array of room's objects or RoomPosition objects that the search should be executed against.
`opts` *optional*	object	An object containing one of the following options: • filter object, function, string Only the objects which pass the filter using the Lodash.filter method will be used.

RETURN VALUE

The closest object if found, null otherwise.

findInRange(type, range, [opts]) (objects, range, [opts])

```
const targets = creep.pos.findInRange(FIND_HOSTILE_CREEPS, 3);
if(targets.length > 0) {
    creep.rangedAttack(targets[0]);
}
```

```
const targets = [
    Game.creeps.John,
    Game.creeps.Mike,
    room.getPositionAt(10,10)
];
const inRangeTargets = creep.pos.findInRange(targets, 3);
```

Find all objects in the specified linear range.

parameter	type	description
`type`	number	See Room.find.
`objects`	array	An array of room's objects or RoomPosition objects that the search should be executed against.
`range`	number	The range distance.

parameter	type	description
opts *optional*	object	See Room.find.

RETURN VALUE

An array with the objects found.

findPathTo(x, y, [opts])
(target, [opts])

```
const path = creep.pos.findPathTo(target);
creep.move(path[0].direction);
```

```
let path = creep.pos.findPathTo(target, {maxOps: 200});
if( !path.length || !target.equalsTo(path[path.length - 1]) ) {
    path = creep.pos.findPathTo(target,
        {maxOps: 1000, ignoreDestructibleStructures: true});
}
if( path.length ) {
    creep.move(path[0].direction);
}
```

Find an optimal path to the specified position using Jump Point Search algorithm. This method is a shorthand for Room.findPath. If the target is in another room, then the corresponding exit will be used as a target.

parameter	type	description
x	number	X position in the room.
y	number	Y position in the room.
target	object	Can be a RoomPosition object or any object containing RoomPosition.
opts *optional*	object	An object containing pathfinding options flags (see Room.findPath for more details).

RETURN VALUE

An array with path steps in the following format:

```
[
    { x: 10, y: 5, dx: 1,  dy: 0, direction: RIGHT },
    { x: 10, y: 6, dx: 0,  dy: 1, direction: BOTTOM },
    { x: 9,  y: 7, dx: -1, dy: 1, direction: BOTTOM_LEFT },
    ...
]
```

getDirectionTo(x,y)
(target)

```
const direction = creep.pos.getDirectionTo(target);
creep.move(direction);
```

Get linear direction to the specified position.

parameter	type	description
`x`	number	X position in the room.
`y`	number	Y position in the room.
`target`	object	Can be a RoomPosition object or any object containing RoomPosition.

RETURN VALUE

A number representing one of the direction constants.

getRangeTo(x,y)
(target)

```
const range = creep.pos.getRangeTo(target);
if(range <= 3) {
    creep.rangedAttack(target);
}
```

Get linear range to the specified position.

parameter	type	description
`x`	number	X position in the room.
`y`	number	Y position in the room.
`target`	object	Can be a RoomPosition object or any object containing RoomPosition.

RETURN VALUE

A number of squares to the given position.

inRangeTo(x, y, range) (target, range)

```
if(creep.pos.inRangeTo(target, 3)) {
    creep.rangedAttack(target);
}
```

Check whether this position is in the given range of another position.

parameter	type	description
`x`	number	X position in the same room.
`y`	number	Y position in the same room.
`target`	RoomPosition	The target position.
`range`	number	The range distance.

RETURN VALUE

A boolean value.

isEqualTo(x,y) (target)

```
if(creep.pos.isEqualTo(10,25)) {
    creep.move(RIGHT);
}
```

```
if(creep.pos.isEqualTo(Game.flags.Flag1)) {
    creep.move(RIGHT);
}
```

Check whether this position is the same as the specified position.

parameter	type	description
`x`	number	X position in the room.
`y`	number	Y position in the room.

parameter	type	description
`target`	object	Can be a RoomPosition object or any object containing RoomPosition.

RETURN VALUE

A boolean value.

`isNearTo(x,y)`
`(target)`

```
if(creep.pos.isNearTo(target)) {
    creep.transferEnergy(target);
}
```

Check whether this position is on the adjacent square to the specified position. The same as `inRangeTo(target, 1)`.

parameter	type	description
`x`	number	X position in the room.
`y`	number	Y position in the room.
`target`	object	Can be a RoomPosition object or any object containing RoomPosition.

RETURN VALUE

A boolean value.

`look()`

```
const look = Game.flags.Flag1.pos.look();
look.forEach(function(lookObject) {
    if(lookObject.type == LOOK_CREEPS &&
       lookObject[LOOK_CREEPS].getActiveBodyparts(ATTACK) == 0) {
        creep.moveTo(lookObject.creep);
    }
});
```

Get the list of objects at the specified room position.

RETURN VALUE

An array with objects at the specified position in the following format:

```
[
    { type: 'creep', creep: {...} },
    { type: 'structure', structure: {...} },
    ...
    { type: 'terrain', terrain: 'swamp' }
]
```

`lookFor(type)`

```
const found = Game.flags.Flag1.pos.lookFor(LOOK_CREEPS);
if(found.length && found[0].getActiveBodyparts(ATTACK) == 0) {
    creep.moveTo(found[0]);
}
```

Get an object with the given type at the specified room position.

parameter	type	description
`type`	string	One of the `LOOK_*` constants.

RETURN VALUE

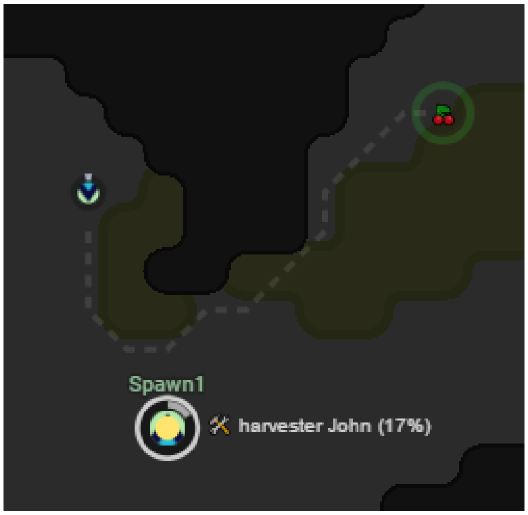
An array of objects of the given type at the specified position if found.

RoomVisual

Room visuals provide a way to show various visual debug info in game rooms. You can use the `RoomVisual` object to draw simple shapes that are visible only to you. Every existing `Room` object already contains the `visual` property, but you also can create new `RoomVisual` objects for any room (even without visibility) using the constructor.

Room visuals are not stored in the database, their only purpose is to display something in your browser. All drawings will persist for one tick and will disappear if not updated. All `RoomVisual` API calls have no added CPU cost (their cost is natural and mostly related to simple `JSON.serialize` calls). However, there is a usage limit: you cannot post more than 500 KB of serialized data per one room (see `getSize` method).

All draw coordinates are measured in game coordinates and centered to tile centers, i.e. (10,10) will point to the center of the creep at `x:10; y:10` position. Fractional coordinates are allowed.



`constructor([roomName])`

```
Game.rooms['W10N10'].visual.circle(10,20).line(0,0,10,20);
// the same as:
new RoomVisual('W10N10').circle(10,20).line(0,0,10,20);
```

```
// this text will be displayed in all rooms
new RoomVisual().text('Some text', 1, 1, {align: 'left'});
```

You can directly create new `RoomVisual` object in any room, even if it's invisible to your script.

parameter	type	description
`roomName` *optional*	string	The room name. If undefined, visuals will be posted to all rooms simultaneously.

roomNamestring

The name of the room.

line(x1, y1, x2, y2, [style])
(pos1, pos2, [style])

```
new RoomVisual('W1N1').line(10,15, 20,20);
```

```
creep.room.visual.line(creep.pos, target.pos,
    {color: 'red', style: 'dashed'});
```

Draw a line.

parameter	type	description
`x1`	number	The start X coordinate.
`y1`	number	The start Y coordinate.
`x2`	number	The finish X coordinate.
`y2`	number	The finish Y coordinate.
`pos1`	RoomPosition	The start position object.
`pos2`	RoomPosition	The finish position object.

parameter	type	description
`style` *optional*	object	An object with the following properties: • width number Line width, default is 0.1. • color string Line color in any web format, default is `#ffffff` (white). • opacity number Opacity value, default is 0.5. • lineStyle string Either `undefined` (solid line), `dashed`, or `dotted`. Default is undefined.

RETURN VALUE

The `RoomVisual` object itself, so that you can chain calls.

circle(x, y, [style]) (pos, [style])

```
new RoomVisual('W1N1').circle(10,15);
```

```
creep.room.visual.circle(creep.pos,
    {fill: 'transparent', radius: 0.55, stroke: 'red'});
```

Draw a circle.

parameter	type	description
`x`	number	The X coordinate of the center.
`y`	number	The Y coordinate of the center.
`pos`	RoomPosition	The position object of the center.

parameter	type	description
`style` *optional*	object	An object with the following properties: - radius number Circle radius, default is 0.15. - fill string Fill color in any web format, default is `#ffffff` (white). - opacity number Opacity value, default is 0.5. - stroke string Stroke color in any web format, default is undefined (no stroke). - strokeWidth number Stroke line width, default is 0.1. - lineStyle string Either `undefined` (solid line), `dashed`, or `dotted`. Default is undefined.

RETURN VALUE

The `RoomVisual` object itself, so that you can chain calls.

`rect(x, y, width, hheight, [style])`
`(topLeftPos, width, height, [style])`

```
// 9x9 area from (2,2) to (10,10)
new RoomVisual('W1N1').rect(1.5, 1.5, 9, 9);
```

```
// a rectangle border on creep
creep.room.visual.rect(creep.pos.x - 0.6, creep.pos.y - 0.6,
    1.2, 1.2,
    {fill: 'transparent', stroke: '#f00'});
```

Draw a rectangle.

parameter	type	description
`x`	number	The X coordinate of the top-left corner.
`y`	number	The Y coordinate of the top-left corner.
`topLeftPos`	RoomPosition	The position object of the top-left corner.
`width`	number	The width of the rectangle.

parameter	type	description
`height`	number	The height of the rectangle.
`style` *optional*	object	An object with the following properties: • fill string Fill color in any web format, default is `#ffffff` (white). • opacity number Opacity value, default is 0.5. • stroke string Stroke color in any web format, default is undefined (no stroke). • strokeWidth number Stroke line width, default is 0.1. • lineStyle string Either `undefined` (solid line), `dashed`, or `dotted`. Default is undefined.

RETURN VALUE

The `RoomVisual` object itself, so that you can chain calls.

poly(points, [style])

```
const points = [];
points.push(creep1.pos);
points.push([10,15]);
points.push(new RoomPosition(20,21,'W1N1'));
new RoomVisual('W1N1').poly(points, {fill: 'aqua'});
```

```
// visualize the path
const path = Game.rooms['W1N1'].findPath(from, to);
new RoomVisual('W1N1').poly(path, {stroke: '#fff', strokeWidth: .15,
    opacity: .2, lineStyle: 'dashed'});
```

Draw a polyline.

parameter	type	description
`points`	array	An array of points. Every item should be either an array with 2 numbers (i.e. `[10,15]`), or a `RoomPosition` object.

parameter	type	description
`style` *optional*	object	An object with the following properties: - fill string Fill color in any web format, default is `undefined` (no fill). - opacity number Opacity value, default is 0.5. - stroke string Stroke color in any web format, default is `#ffffff` (white). - strokeWidth number Stroke line width, default is 0.1. - lineStyle string Either `undefined` (solid line), `dashed`, or `dotted`. Default is undefined.

RETURN VALUE

The `RoomVisual` object itself, so that you can chain calls.

text(text, x, y, [style])
(text, pos, [style])

```
new RoomVisual('W1N1').text("Target💥", 10, 15, {color: 'green', font: 0.8});
```

Draw a text label. You can use any valid Unicode characters, including emoji.

parameter	type	description
`text`	string	The text message.
`x`	number	The X coordinate of the label baseline point.
`y`	number	The Y coordinate of the label baseline point.
`pos`	RoomPosition	The position object of the label baseline.

parameter	type	description
`style` *optional*	object	An object with the following properties: • color string Font color in any web format, default is `#ffffff` (white). • font number, string Either a number or a string in one of the following forms: ◦ `0.7` - relative size in game coordinates ◦ `20px` - absolute size in pixels ◦ `0.7 serif` ◦ `bold italic 1.5 Times New Roman` • stroke string Stroke color in any web format, default is undefined (no stroke). • strokeWidth number Stroke width, default is 0.15. • backgroundColor string Background color in any web format, default is undefined (no background). When background is enabled, text vertical align is set to middle (default is baseline). • backgroundPadding number Background rectangle padding, default is 0.3. • align string Text align, either `center`, `left`, or `right`. Default is `center`. • opacity number Opacity value, default is 1.0.

RETURN VALUE

The `RoomVisual` object itself, so that you can chain calls.

clear()

```
new RoomVisual('W1N1').clear();
```

Remove all visuals from the room.

RETURN VALUE

The `RoomVisual` object itself, so that you can chain calls.

getSize()

```
if(creep.room.visual.getSize() >= 512000) {
    // cannot add more visuals in this tick
}
```

Get the stored size of all visuals added in the room in the current tick. It must not exceed 512,000 (500 KB).

RETURN VALUE

The size of the visuals in bytes.

Source

An energy source object. Can be harvested by creeps with a `WORK` body part.

Energy amount	4000 in center rooms 3000 in an owned or reserved room 1500 in an unreserved room
Energy regeneration	Every 300 game ticks

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

energynumber

The remaining amount of energy.

energyCapacitynumber

The total amount of energy in the source.

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

ticksToRegenerationnumber

The remaining time after which the source will be refilled.

Structure

The base prototype object of all structures.

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

hitsnumber

The current amount of hit points of the structure.

hitsMaxnumber

The total amount of hit points of the structure.

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

structureTypestring

One of the `STRUCTURE_*` constants.

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

notifyWhenAttacked(enabled)

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

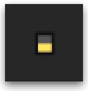
RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.

constant	value	description
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

StructureContainer



A small container that can be used to store resources. This is a walkable structure. All dropped resources automatically goes to the container at the same tile.

Controller level	Any (including neutral rooms)
Available per room	5
Capacity	2,000
Cost	5,000
Hits	250,000
Decay	Loses 5,000 hits every 500 ticks in an owned room, and every 100 ticks in an unowned room.

Inherited from `RoomObject`

`posRoomPosition`

An object representing the position of this object in the room.

Inherited from `RoomObject`

`roomRoom`

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

`hitsnumber`

The current amount of hit points of the structure.

Inherited from `Structure`

`hitsMaxnumber`

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

notifyWhenAttacked(enabled)

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

storeobject

```
const containersWithEnergy = room.find(FIND_STRUCTURES, {
    filter: (i) => i.structureType == STRUCTURE_CONTAINER &&
                   i.store[RESOURCE_ENERGY] > 0
});
```

```
const total = _.sum(container.store);
```

An object with the structure contents. Each object key is one of the `RESOURCE_*` constants, values are resources amounts. `RESOURCE_ENERGY` is always defined and equals to 0 when empty, other resources are undefined when empty. You can use `lodash.sum` to get the total amount of contents.

storeCapacitynumber

The total amount of resources the structure can contain.

ticksToDecaynumber

The amount of game ticks when this container will lose some hit points.

transfer(target, resourceType, [amount])

This method is deprecated and will be removed soon. Please use `Creep.withdraw` instead.

```
const containers = creep.pos.findInRange(FIND_STRUCTURES, 1,
      {filter: {structureType: STRUCTURE_CONTAINER}});
containers[0].transfer(creep, RESOURCE_ENERGY);
```

Transfer resource from this structure to a creep. The target has to be at adjacent square.

parameter	type	description
`target`	Creep	The target object.
`resourceType`	string	One of the `RESOURCE_*` constants.
`amount` *optional*	number	The amount of resources to be transferred. If omitted, all the available amount is used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The structure does not have the given amount of energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid object which can contain energy.
`ERR_FULL`	-8	The target cannot receive any more energy.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_INVALID_ARGS`	-10	The energy amount is incorrect.

StructureController

Claim this structure to take control over the room. The controller structure cannot be damaged or destroyed.

It can be addressed by `Room.controller` property.

Level	Upgrade to next level	Downgrade timer
1	200 energy	20,000 ticks
2	45,000 energy	5,000 ticks
3	135,000 energy	10,000 ticks
4	405,000 energy	20,000 ticks
5	1,215,000 energy	40,000 ticks
6	3,645,000 energy	60,000 ticks
7	10,935,000 energy	100,000 ticks
8	—	150,000 ticks

SAFE MODE

Effect	Blocks `attack`, `rangedAttack`, `rangedMassAttack`, `dismantle`, `heal`, `rangedHeal`, `attackController` and `withdraw` methods of all hostile creeps in the room. Only one room can be in safe mode at the same time. When safe mode is active, all hostile creeps become visually transparent and passable - your creeps can move through them freely (but not vice versa).
Duration	20,000 ticks
Cooldown	50,000 ticks (no cooldown in Novice Areas)
Generation	• Each new controller level generates one available activation • Can be generated by creeps using 1,000 ghodium • All available activations are reset if the controller is downgraded

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

notifyWhenAttacked(enabled)

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

myboolean

Whether this is your own structure.

Inherited from `OwnedStructure`

ownerobject

An object with the structure’s owner info containing the following properties:

parameter	type	description

parameter	type	description
`username`	string	The name of the owner user.

levelnumber

Current controller level, from 0 to 8.

progressnumber

The current progress of upgrading the controller to the next level.

progressTotalnumber

The progress needed to reach the next level.

reservationobject

An object with the controller reservation info if present:

parameter	type	description
`username`	string	The name of a player who reserved this controller.
`ticksToEnd`	number	The amount of game ticks when the reservation will end.

safeModenumber

How many ticks of safe mode remaining, or undefined.

safeModeAvailablenumber

Safe mode activations available to use.

safeModeCooldownnumber

During this period in ticks new safe mode activations will be blocked, undefined if cooldown is inactive.

signobject

An object with the controller sign info if present:

parameter	type	description
`username`	string	The name of a player who signed this controller.
`text`	string	The sign text.
`time`	number	The sign time in game ticks.
`datetime`	Date	The sign real date.

ticksToDowngradenumber

The amount of game ticks when this controller will lose one level. This timer can be reset by using `Creep.upgradeController`.

upgradeBlockednumber

The amount of game ticks while this controller cannot be upgraded due to attack. Safe mode is also unavailable during this period.

activateSafeMode()

```
room.controller.activateSafeMode();
```

Activate safe mode if available.

RETURN VALUE

One of the following codes:

constant	value	description

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this controller.
`ERR_BUSY`	-4	There is another room in safe mode already.
`ERR_NOT_ENOUGH_RESOURCES`	-6	There is no safe mode activations available.
`ERR_TIRED`	-11	The previous safe mode is still cooling down, or the controller is `upgradeBlocked`, or the controller is downgraded for 5000 ticks or more.

`unclaim()`

```
room.controller.unclaim();
```

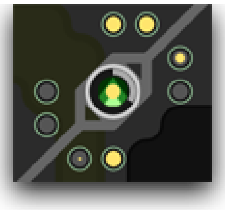
Make your claimed controller neutral again.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this controller.

StructureExtension



Contains energy which can be spent on spawning bigger creeps. Extensions can be placed anywhere in the room, any spawns will be able to use them regardless of distance.

Controller level	
1	—

2	5 extensions (50 capacity)
3	10 extensions (50 capacity)
4	20 extensions (50 capacity)
5	30 extensions (50 capacity)
6	40 extensions (50 capacity)
7	50 extensions (100 capacity)
8	60 extensions (200 capacity)
Cost	3,000
Hits	1,000

Inherited from `RoomObject`

`posRoomPosition`

An object representing the position of this object in the room.

Inherited from `RoomObject`

`roomRoom`

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

`hitsnumber`

The current amount of hit points of the structure.

Inherited from `Structure`

`hitsMaxnumber`

The total amount of hit points of the structure.

Inherited from `Structure`

`idstring`

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

notifyWhenAttacked(enabled)

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

`myboolean`

Whether this is your own structure.

Inherited from `OwnedStructure`

`ownerobject`

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

`energynumber`

The amount of energy containing in the extension.

`energyCapacitynumber`

The total amount of energy the extension can contain.

`transferEnergy(target, [amount])`

This method is deprecated and will be removed soon. Please use `Creep.withdraw` instead.

Transfer the energy from the extension to a creep. You can transfer resources to your creeps from hostile structures as well.

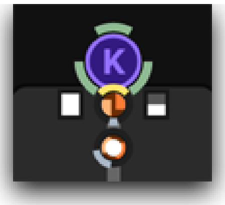
parameter	type	description
`target`	Creep	The creep object which energy should be transferred to.
`amount` *optional*	number	The amount of energy to be transferred. If omitted, all the remaining amount of energy will be used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of the target creep, or there is a hostile rampart on top of the structure.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The extension contains less energy than the given amount.
`ERR_INVALID_TARGET`	-7	The specified target object is not a creep.
`ERR_FULL`	-8	The target creep can not carry the given amount of energy.
`ERR_NOT_IN_RANGE`	-9	The target creep is too far away.

StructureExtractor



Allows to harvest a mineral deposit. Learn more about minerals from this article.

Controller level	6
Cost	5,000
Hits	500
Cooldown	5 ticks on each `harvest` action

Inherited from `RoomObject`

`posRoomPosition`

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

`isActive()`

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

myboolean

Whether this is your own structure.

Inherited from `OwnedStructure`

ownerobject

An object with the structure’s owner info containing the following properties:

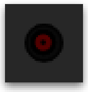
parameter	type	description
`username`	string	The name of the owner user.

cooldownnumber

The amount of game ticks until the next harvest action is possible.

StructureKeeperLair

Non-player structure. Spawns NPC Source Keepers that guards energy sources and minerals in some rooms. This structure cannot be destroyed.



Spawning time	300

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

notifyWhenAttacked(enabled)

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

myboolean

Whether this is your own structure.

Inherited from `OwnedStructure`

ownerobject

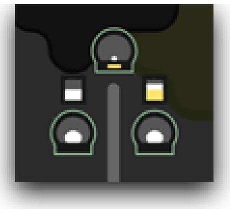
An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

`ticksToSpawnnumber`

Time to spawning of the next Source Keeper.

StructureLab



Produces mineral compounds from base minerals and boosts creeps. Learn more about minerals from this article.

Controller level	
1-5	—
6	3 labs
7	6 labs
8	10 labs
Cost	50,000
Hits	500
Capacity	3000 mineral units, 2000 energy units
Produce	5 mineral compound units per reaction
Reaction cooldown	10 ticks
Distance to input labs	2 squares
Boost cost	30 mineral units, 20 energy units per body part

Inherited from `RoomObject`

`posRoomPosition`

An object representing the position of this object in the room.

Inherited from `RoomObject`

`roomRoom`

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.

constant	value	description
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

`isActive()`

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

`myboolean`

Whether this is your own structure.

Inherited from `OwnedStructure`

ownerobject

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

cooldownnumber

The amount of game ticks the lab has to wait until the next reaction is possible.

energynumber

The amount of energy containing in the lab. Energy is used for boosting creeps.

energyCapacitynumber

The total amount of energy the lab can contain.

mineralAmountnumber

The amount of mineral resources containing in the lab.

mineralTypestring

The type of minerals containing in the lab. Labs can contain only one mineral type at the same time.

mineralCapacitynumber

The total amount of minerals the lab can contain.

`boostCreep(creep, [bodyPartsCount])`

Boosts creep body parts using the containing mineral compound. The creep has to be at adjacent square to the lab.

parameter	type	description
`creep`	Creep	The target creep.
`bodyPartsCount` *optional*	number	The number of body parts of the corresponding type to be boosted. Body parts are always counted left-to-right for `TOUGH`, and right-to-left for other types. If undefined, all the eligible body parts are boosted.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this lab.
`ERR_NOT_FOUND`	-5	The mineral containing in the lab cannot boost any of the creep's body parts.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The lab does not have enough energy or minerals.
`ERR_INVALID_TARGET`	-7	The targets is not valid creep object.
`ERR_NOT_IN_RANGE`	-9	The targets are too far away.

`runReaction(lab1, lab2)`

Produce mineral compounds using reagents from two other labs. The same input labs can be used by many output labs.

parameter	type	description
`lab1`	StructureLab (lab)	The first source lab.
`lab2`	StructureLab (lab)	The second source lab.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this lab.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The source lab do not have enough resources.
`ERR_INVALID_TARGET`	-7	The targets are not valid lab objects.
`ERR_FULL`	-8	The target cannot receive any more energy.
`ERR_NOT_IN_RANGE`	-9	The targets are too far away.
`ERR_INVALID_ARGS`	-10	The reaction cannot be run using this resources.
`ERR_TIRED`	-11	The lab is still cooling down.

`transfer(target, resourceType, [amount])`

This method is deprecated and will be removed soon. Please use `Creep.withdraw` instead.

Transfer resource from this structure to a creep. The target has to be at adjacent square. You can transfer resources to your creeps from hostile structures as well.

parameter	type	description
`target`	Creep	The target object.
`resourceType`	string	One of the `RESOURCE_*` constants.
`amount` *optional*	number	The amount of resources to be transferred. If omitted, all the available amount is used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of the target creep, or there is a hostile rampart on top of the structure.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The creep does not have the given amount of resources.
`ERR_INVALID_TARGET`	-7	The target is not a valid Creep object.
`ERR_FULL`	-8	The target cannot receive any more energy.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_INVALID_ARGS`	-10	The amount or resource type is incorrect.

StructureLink

Remotely transfers energy to another Link in the same room.

Controller level	
1-4	—
5	2 links
6	3 links
7	4 links
8	6 links
Cost	5,000
Hits	1,000
Capacity	800
Cooldown time	1 tick per tile of the linear distance to the target
Energy loss	3%

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

`isActive()`

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

myboolean

Whether this is your own structure.

Inherited from `OwnedStructure`

ownerobject

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

cooldownnumber

The amount of game ticks the link has to wait until the next transfer is possible.

energynumber

The amount of energy containing in the link.

energyCapacitynumber

The total amount of energy the link can contain.

transferEnergy(target, [amount])

```
const linkFrom = Game.rooms['W1N1'].lookForAt('structure', 10, 25)[0];

const linkTo = linkFrom.pos.findInRange(FIND_MY_STRUCTURES, 2,
    {filter: {structureType: STRUCTURE_LINK}})[0];

linkFrom.transferEnergy(linkTo);
```

Remotely transfer energy to another link at any location in the same room.

parameter	type	description

parameter	type	description
`target`	StructureLink	The target object.
`amount` *optional*	number	The amount of energy to be transferred. If omitted, all the available energy is used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this link.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The structure does not have the given amount of energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid StructureLink object.
`ERR_FULL`	-8	The target cannot receive any more energy.
`ERR_INVALID_ARGS`	-10	The energy amount is incorrect.
`ERR_TIRED`	-11	The link is still cooling down.
`ERR_RCL_NOT_ENOUGH`	-14	Room Controller Level insufficient to use this link.

StructureNuker

Launches a nuke to another room dealing huge damage to the landing area. Each launch has a cooldown and requires energy and ghodium resources. Launching creates a Nuke object at the target room position which is visible to any player until it is landed. Incoming nuke cannot be moved or cancelled. Nukes cannot be launched from or to novice rooms. Resources placed into a StructureNuker cannot be withdrawn.

Controller level	
1-7	—
8	1 nuke

Cost	100,000
Hits	1,000
Range	10 rooms
Launch cost	300,000 energy 5,000 ghodium
Launch cooldown	100,000 ticks
Landing time	50,000 ticks
Effect	All creeps, construction sites and dropped resources in the room are removed immediately, even inside ramparts. Damage to structures: • 10,000,000 hits at the landing position; • 5,000,000 hits to all structures in 5x5 area. Note that you can stack multiple nukes from different rooms at the same target position to increase damage. If the room is in safe mode, then the safe mode is cancelled immediately, and the safe mode cooldown is reset to 0. The room controller is hit by triggering `upgradeBlocked` period, which means it is unavailable to activate safe mode again within the next 200 ticks.

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

notifyWhenAttacked(enabled)

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

myboolean

Whether this is your own structure.

Inherited from `OwnedStructure`

ownerobject

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

energynumber

The amount of energy containing in this structure.

energyCapacitynumber

The total amount of energy this structure can contain.

ghodiumnumber

The amount of ghodium containing in this structure.

ghodiumCapacitynumber

The total amount of ghodium this structure can contain.

cooldownnumber

The amount of game ticks until the next launch is possible.

launchNuke(pos)

```
nuker.launchNuke(new RoomPosition(20,30, 'W1N1'));
```

Launch a nuke to the specified position.

parameter	type	description
`pos`	RoomPosition	The target room position.

RETURN VALUE

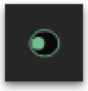
One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The structure does not have enough energy and/or ghodium.

constant	value	description
`ERR_INVALID_TARGET`	-7	The target is not a valid RoomPosition.
`ERR_NOT_IN_RANGE`	-9	The target room is out of range.
`ERR_TIRED`	-11	This structure is still cooling down.
`ERR_RCL_NOT_ENOUGH`	-14	Room Controller Level insufficient to use this structure.

StructureObserver

Provides visibility into a distant room from your script.



Controller level	
1-7	—
8	1 observer
Cost	8,000
Hits	500
Range	10 rooms

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

`myboolean`

Whether this is your own structure.

Inherited from `OwnedStructure`

`ownerobject`

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

`observeRoom(roomName)`

Provide visibility into a distant room from your script. The target room object will be available on the next tick.

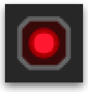
parameter	type	description
`roomName`	string	The name of the target room.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_IN_RANGE`	-9	Room `roomName` is not in observing range.
`ERR_INVALID_ARGS`	-10	`roomName` argument is not a valid room name value.
`ERR_RCL_NOT_ENOUGH`	-14	Room Controller Level insufficient to use this structure.

StructurePowerBank



Non-player structure. Contains power resource which can be obtained by destroying the structure. Hits the attacker creep back on each attack. Learn more about power from this article.

Hits	2,000,000
Return damage	50%
Capacity	500 — 10,000
Decay	5,000 ticks

Inherited from `RoomObject`

`pos` `RoomPosition`

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.

constant	value	description
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

notifyWhenAttacked(enabled)

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

powernumber

The amount of power containing.

ticksToDecaynumber

The amount of game ticks when this structure will disappear.

StructurePowerSpawn

Processes power into your account, and spawns power creeps with special unique powers (in development). Learn more about power from this article.

Controller level	
1-7	—
8	1 power spawn
Cost	100,000
Hits	5,000
Capacity	5,000 energy units, 100 power units
Processing cost	50 energy units per 1 power unit
Processing speed	1 power unit per tick

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

`myboolean`

Whether this is your own structure.

Inherited from `OwnedStructure`

`ownerobject`

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

energynumber

The amount of energy containing in this structure.

energyCapacitynumber

The total amount of energy this structure can contain.

powernumber

The amount of power containing in this structure.

powerCapacitynumber

The total amount of power this structure can contain.

createPowerCreep(name)

Create a power creep. *This method is under development.*

parameter	type	description
`roomName`	string	The name of the power creep.

processPower()

Register power resource units into your account. Registered power allows to develop power creeps skills.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The structure does not have enough energy or power resource units.

constant	value	description
`ERR_RCL_NOT_ENOUGH`	-14	Room Controller Level insufficient to use this structure.

transferEnergy(target, [amount])

This method is deprecated and will be removed soon. Please use `Creep.withdraw` instead.

Transfer the energy from this structure to a creep. You can transfer resources to your creeps from hostile structures as well.

parameter	type	description
`target`	Creep	The creep object which energy should be transferred to.
`amount` *optional*	number	The amount of energy to be transferred. If omitted, all the remaining amount of energy will be used.

RETURN VALUE

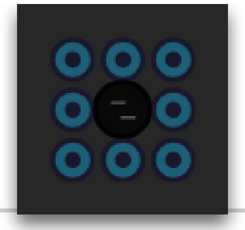
One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of the target creep, or there is a hostile rampart on top of the structure.
`ERR_NOT_ENOUGH_RESOURCES`	-6	This structure less energy than the given amount.
`ERR_INVALID_TARGET`	-7	The specified target object is not a creep.
`ERR_FULL`	-8	The target creep can not carry the given amount of energy.
`ERR_NOT_IN_RANGE`	-9	The target creep is too far away.

StructurePortal

A non-player structure. Instantly teleports your creeps to a distant room acting as a room exit tile. Portals appear randomly in the central room of each sector.

Stable time	10 days
Decay time	30,000 ticks



Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

`destroy()`

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

`isActive()`

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.

constant	value	description
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

`destinationRoomPosition | object`

If this is an **inter-room** portal, then this property contains a `RoomPosition` object leading to the point in the destination room.

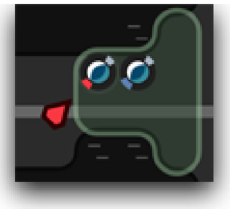
If this is an **inter-shard** portal, then this property contains an object with `shard` and `room` string properties. Exact coordinates are undetermined, the creep will appear at any free spot in the destination room.

`ticksToDecaynumber`

The amount of game ticks when the portal disappears, or undefined when the portal is stable.

StructureRampart

Blocks movement of hostile creeps, and defends your creeps and structures on the same tile. Can be used as a controllable gate.



Controller level	
1	—
2	300,000 max hits
3	1,000,000 max hits
4	3,000,000 max hits
5	10,000,000 max hits
6	30,000,000 max hits
7	100,000,000 max hits
8	300,000,000 max hits
Cost	1
Hits when constructed	1
Decay	Loses 300 hits every 100 ticks

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

`isActive()`

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

myboolean

Whether this is your own structure.

Inherited from `OwnedStructure`

ownerobject

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

isPublicboolean

If false (default), only your creeps can step on the same square. If true, any hostile creeps can pass through.

ticksToDecaynumber

The amount of game ticks when this rampart will lose some hit points.

setPublic(isPublic)

Make this rampart public to allow other players' creeps to pass through.

parameter	type	description
`isPublic`	boolean	Whether this rampart should be public or non-public.

RETURN VALUE

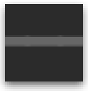
One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.

constant	value	description
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.

StructureRoad

Decreases movement cost to 1. Using roads allows creating creeps with less `MOVE` body parts.



Controller level	Any (including neutral rooms)
Cost	• 300 at plain land • 1,500 at swamp
Hits	• 5,000 at plain land • 25,000 at swamp
Decay	• Loses 100 hits every 1,000 ticks at plain land • Loses 500 hits every 1,000 ticks at swamp Note: every creep step decreases the decay timer for 1 tick per each creep body part

Inherited from `RoomObject`

`posRoomPosition`

An object representing the position of this object in the room.

Inherited from `RoomObject`

`roomRoom`

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

`hitsnumber`

The current amount of hit points of the structure.

Inherited from `Structure`

`hitsMax`number

The total amount of hit points of the structure.

Inherited from `Structure`

`id`string

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

`structureType`string

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

`destroy()`

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

`isActive()`

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

notifyWhenAttacked(enabled)

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

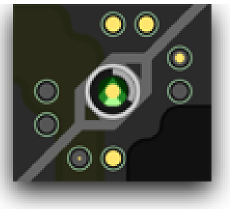
One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

ticksToDecaynumber

The amount of game ticks when this road will lose some hit points.

StructureSpawn



Spawn is your colony center. This structure can create, renew, and recycle creeps. All your spawns are accessible through `Game.spawns` hash list. Spawns auto-regenerate a little amount of energy each tick, so that you can easily recover even if all your creeps died.

Controller level	
1-6	1 spawn
7	2 spawns
8	3 spawns
Cost	15,000

Hits	5,000
Capacity	300
Spawn time	3 ticks per each body part
Energy auto-regeneration	1 energy unit per tick while energy available in the room (in all spawns and extensions) is less than 300

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

`destroy()`

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

`isActive()`

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.

constant	value	description
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

myboolean

Whether this is your own structure.

Inherited from `OwnedStructure`

ownerobject

An object with the structure's owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

energynumber

The amount of energy containing in the spawn.

energyCapacitynumber

The total amount of energy the spawn can contain

memoryany

```
spawn.memory.queue = [];
```

A shorthand to `Memory.spawns[spawn.name]`. You can use it for quick access the spawn's specific memory data object. Learn more about memory

namestring

Spawn’s name. You choose the name upon creating a new spawn, and it cannot be changed later. This name is a hash key to access the spawn via the Game.spawns object.

spawningStructureSpawn.Spawning

If the spawn is in process of spawning a new creep, this object will contain a `StructureSpawn.Spawning` object, or null otherwise.

canCreateCreep(body, [name])

This method is deprecated and will be removed soon. Please use `StructureSpawn.spawnCreep` with `dryRun` flag instead.

```
if(spawn.canCreateCreep(body, name) == OK) {
    spawn.createCreep(body, name);
}
```

Check if a creep can be created.

parameter	type	description
`body`	array<string>	An array describing the new creep’s body. Should contain 1 to 50 elements with one of these constants: • `WORK` • `MOVE` • `CARRY` • `ATTACK` • `RANGED_ATTACK` • `HEAL` • `TOUGH` • `CLAIM`
`name` *optional*	string	The name of a new creep. It should be unique creep name, i.e. the `Game.creeps` object should not contain another creep with the same name (hash key). If not defined, a random name will be generated.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	A creep with the given body and name can be created.
`ERR_NOT_OWNER`	-1	You are not the owner of this spawn.

constant	value	description
`ERR_NAME_EXISTS`	-3	There is a creep with the same name already.
`ERR_BUSY`	-4	The spawn is already in process of spawning another creep.
`ERR_NOT_ENOUGH_ENERGY`	-6	The spawn and its extensions contain not enough energy to create a creep with the given body.
`ERR_INVALID_ARGS`	-10	Body is not properly described.
`ERR_RCL_NOT_ENOUGH`	-14	Your Room Controller level is insufficient to use this spawn.

createCreep(body, [name], [memory])

This method is deprecated and will be removed soon. Please use `StructureSpawn.spawnCreep` instead.

```
Game.spawns['Spawn1'].createCreep([WORK, CARRY, MOVE], 'Worker1');
```

```
Game.spawns['Spawn1'].createCreep([WORK, CARRY, MOVE], null,
    {role: 'harvester'});
```

```
const result = Game.spawns['Spawn1'].createCreep([WORK, CARRY, MOVE]);

if(_.isString(result)) {
    console.log('The name is: '+result);
}
else {
    console.log('Spawn error: '+result);
}
```

Start the creep spawning process. The required energy amount can be withdrawn from all spawns and extensions in the room.

parameter	type	description
`body`	array<string>	An array describing the new creep’s body. Should contain 1 to 50 elements with one of these constants: • `WORK` • `MOVE` • `CARRY` • `ATTACK` • `RANGED_ATTACK` • `HEAL` • `TOUGH` • `CLAIM`

parameter	type	description
`name` *optional*	string	The name of a new creep. It should be unique creep name, i.e. the `Game.creeps` object should not contain another creep with the same name (hash key). If not defined, a random name will be generated.
`memory` *optional*	any	The memory of a new creep. If provided, it will be immediately stored into `Memory.creeps[name]`.

RETURN VALUE

The name of a new creep or one of these error codes:

constant	value	description
`ERR_NOT_OWNER`	-1	You are not the owner of this spawn.
`ERR_NAME_EXISTS`	-3	There is a creep with the same name already.
`ERR_BUSY`	-4	The spawn is already in process of spawning another creep.
`ERR_NOT_ENOUGH_ENERGY`	-6	The spawn and its extensions contain not enough energy to create a creep with the given body.
`ERR_INVALID_ARGS`	-10	Body is not properly described.
`ERR_RCL_NOT_ENOUGH`	-14	Your Room Controller level is insufficient to use this spawn.

spawnCreep(body, name, [opts])

```
Game.spawns['Spawn1'].spawnCreep([WORK, CARRY, MOVE], 'Worker1');
```

```
Game.spawns['Spawn1'].spawnCreep([WORK, CARRY, MOVE], 'Worker1', {
    memory: {role: 'harvester'}
});
```

```
Game.spawns['Spawn1'].spawnCreep([WORK, CARRY, MOVE], 'Worker1', {
    energyStructures: [
        Game.spawns['Spawn1'],
        Game.getObjectById('anExtensionId')
    ]
});
```

```
var testIfCanSpawn = Game.spawns['Spawn1'].spawnCreep([WORK, CARRY, MOVE],
    'Worker1', { dryRun: true });
```

Start the creep spawning process. The required energy amount can be withdrawn from all spawns and extensions in the room.

parameter	type	description
`body`	array<string>	An array describing the new creep’s body. Should contain 1 to 50 elements with one of these constants: • `WORK` • `MOVE` • `CARRY` • `ATTACK` • `RANGED_ATTACK` • `HEAL` • `TOUGH` • `CLAIM`
`name`	string	The name of a new creep. It must be a unique creep name, i.e. the `Game.creeps` object should not contain another creep with the same name (hash key).
`opts` *optional*	object	An object with additional options for the spawning process. • memory any Memory of the new creep. If provided, it will be immediately stored into `Memory.creeps[name]`. • energyStructures array Array of spawns/extensions from which to draw energy for the spawning process. Structures will be used according to the array order. • dryRun boolean If `dryRun` is true, the operation will only check if it is possible to create a creep. • directions array Set desired directions where the creep should move when spawned. An array with the direction constants: ○ `TOP` ○ `TOP_RIGHT` ○ `RIGHT` ○ `BOTTOM_RIGHT` ○ `BOTTOM` ○ `BOTTOM_LEFT` ○ `LEFT` ○ `TOP_LEFT`

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this spawn.

constant	value	description
`ERR_NAME_EXISTS`	-3	There is a creep with the same name already.
`ERR_BUSY`	-4	The spawn is already in process of spawning another creep.
`ERR_NOT_ENOUGH_ENERGY`	-6	The spawn and its extensions contain not enough energy to create a creep with the given body.
`ERR_INVALID_ARGS`	-10	Body is not properly described or name was not provided.
`ERR_RCL_NOT_ENOUGH`	-14	Your Room Controller level is insufficient to use this spawn.

recycleCreep(target)

Kill the creep and drop up to 100% of resources spent on its spawning and boosting depending on remaining life time. The target should be at adjacent square.

parameter	type	description
`target`	Creep	The target creep object.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this spawn or the target creep.
`ERR_INVALID_TARGET`	-7	The specified target object is not a creep.
`ERR_NOT_IN_RANGE`	-9	The target creep is too far away.

renewCreep(target)

Increase the remaining time to live of the target creep. The target should be at adjacent square. The spawn should not be busy with the spawning process. Each execution increases the creep's timer by amount of ticks according to this formula:

```
floor(600/body_size)
```

Energy required for each execution is determined using this formula:

```
ceil(creep_cost/2.5/body_size)
```

Renewing a creep removes all of its boosts.

parameter	type	description
`target`	Creep	The target creep object.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of the spawn, or the creep.
`ERR_BUSY`	-4	The spawn is spawning another creep.
`ERR_NOT_ENOUGH_ENERGY`	-6	The spawn does not have enough energy.
`ERR_INVALID_TARGET`	-7	The specified target object is not a creep.
`ERR_FULL`	-8	The target creep's time to live timer is full.
`ERR_NOT_IN_RANGE`	-9	The target creep is too far away.

transferEnergy(target, [amount])

This method is deprecated and will be removed soon. Please use `Creep.withdraw` instead.

Transfer the energy from the spawn to a creep.

parameter	type	description

parameter	type	description
`target`	Creep	The creep object which energy should be transferred to.
`amount` *optional*	number	The amount of energy to be transferred. If omitted, all the remaining amount of energy will be used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this spawn.
`ERR_NOT_ENOUGH_ENERGY`	-6	The spawn contains less energy than the given amount.
`ERR_INVALID_TARGET`	-7	The specified target object is not a creep.
`ERR_FULL`	-8	The target creep can not carry the given amount of energy.
`ERR_NOT_IN_RANGE`	-9	The target creep is too far away.

StructureSpawn.Spawning

Details of the creep being spawned currently that can be addressed by the `StructureSpawn.spawning` property.

directionsarray

An array with the spawn directions, see `StructureSpawn.Spawning.setDirections`.

namestring

The name of a new creep.

needTimenumber

Time needed in total to complete the spawning.

remainingTimenumber

Remaining time to go.

spawnStructureSpawn

A link to the spawn.

cancel()

```
Game.spawns['Spawn1'].spawning.cancel();
```

Cancel spawning immediately. Energy spent on spawning is not returned.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this spawn.

setDirections(directions)

```
Game.spawns['Spawn1'].spawning.setDirections([RIGHT, TOP_RIGHT]);
```

Set desired directions where the creep should move when spawned.

parameter	type	description

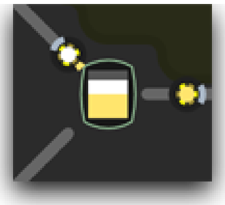
parameter	type	description
`directions`	array<number>	An array with the direction constants: - `TOP` - `TOP_RIGHT` - `RIGHT` - `BOTTOM_RIGHT` - `BOTTOM` - `BOTTOM_LEFT` - `LEFT` - `TOP_LEFT`

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this spawn.
`ERR_INVALID_ARGS`	-10	The directions is array is invalid.

StructureStorage



A structure that can store huge amount of resource units. Only one structure per room is allowed that can be addressed by `Room.storage` property.

Controller level	
1-3	—
4-8	1 storage
Cost	30,000
Hits	10,000
Capacity	1,000,000

Inherited from `RoomObject`

`posRoomPosition`

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
OK	0	The operation has been scheduled successfully.

constant	value	description
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

`isActive()`

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

`myboolean`

Whether this is your own structure.

Inherited from `OwnedStructure`

ownerobject

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

storeobject

```
if(creep.room.storage.store[RESOURCE_ENERGY] == 0) {
    // replenish the storage!
}
```

```
if( !(RESOURCE_UTRIUM in creep.room.storage.store) ) {
    // need more utrium!
}
```

```
const total = _.sum(Game.rooms['W1N1'].storage.store);
```

An object with the storage contents. Each object key is one of the `RESOURCE_*` constants, values are resources amounts. `RESOURCE_ENERGY` is always defined and equals to 0 when empty, other resources are undefined when empty. You can use `lodash.sum` to get the total amount of contents.

storeCapacitynumber

The total amount of resources the storage can contain.

transfer(target, resourceType, [amount])

This method is deprecated and will be removed soon. Please use `Creep.withdraw` instead.

```
storage.transfer(creep, RESOURCE_ENERGY);
```

Transfer resource from this storage to a creep. The target has to be at adjacent square. You can transfer resources to your creeps from hostile structures as well.

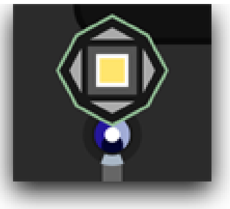
parameter	type	description
`target`	Creep	The target object.
`resourceType`	string	One of the `RESOURCE_*` constants.
`amount` *optional*	number	The amount of resources to be transferred. If omitted, all the available amount is used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of the target creep, or there is a hostile rampart on top of the structure.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The structure does not have the given amount of energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid object which can contain energy.
`ERR_FULL`	-8	The target cannot receive any more energy.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_INVALID_ARGS`	-10	The energy amount is incorrect.

StructureTerminal



Sends any resources to a Terminal in another room. The destination Terminal can belong to any player. Each transaction requires additional energy (regardless of the transfer resource type) that can be calculated using `Game.market.calcTransactionCost` method. For example, sending 1000 mineral units from W0N0 to W10N5 will consume 742 energy units. You can track your incoming and outgoing transactions using the `Game.market` object. Only one Terminal per room is allowed that can be addressed by `Room.terminal` property.

Terminals are used in the Market system.

Controller level

1-5	—
6-8	1 terminal
Cost	100,000
Hits	3,000
Capacity	300,000
Cooldown on send	10 ticks

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

notifyWhenAttacked(enabled)

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

myboolean

Whether this is your own structure.

Inherited from `OwnedStructure`

ownerobject

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

cooldownnumber

The remaining amount of ticks while this terminal cannot be used to make `StructureTerminal.send` or `Game.market.deal` calls.

storeobject

```
if( !(RESOURCE_UTRIUM in Game.rooms['W1N1'].terminal.store) ) {
    // need more utrium!
}
```

```
const total = _.sum(Game.rooms['W1N1'].terminal.store);
```

An object with the storage contents. Each object key is one of the `RESOURCE_*` constants, values are resources amounts. `RESOURCE_ENERGY` is always defined and equals to 0 when empty, other resources are undefined when empty. You can use `lodash.sum` to get the total amount of contents.

storeCapacitynumber

The total amount of resources the storage can contain.

send(resourceType, amount, destination, [description])

```
Game.rooms['W1N1'].terminal.send(RESOURCE_UTRIUM, 100, 'W2N3',
    'trade contract #1');
```

Sends resource to a Terminal in another room with the specified name.

parameter	type	description
`resourceType`	string	One of the `RESOURCE_*` constants.
`amount`	number	The amount of resources to be sent. The minimum amount is 100.
`destination`	string	The name of the target room. You don't have to gain visibility in this room.
`description` *optional*	string	The description of the transaction. It is visible to the recipient. The maximum length is 100 characters.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The structure does not have the required amount of resources.
`ERR_INVALID_ARGS`	-10	The arguments provided are incorrect.
`ERR_TIRED`	-11	The terminal is still cooling down.

transfer(target, resourceType, [amount])

This method is deprecated and will be removed soon. Please use `Creep.withdraw` instead.

```
Game.rooms['W1N1'].terminal.transfer(creep, RESOURCE_ENERGY);
```

Transfer resource from this terminal to a creep. The target has to be at adjacent square. You can transfer resources to your creeps from hostile structures as well.

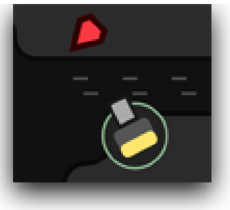
parameter	type	description
`target`	Creep	The target object.
`resourceType`	string	One of the `RESOURCE_*` constants.
`amount` *optional*	number	The amount of resources to be transferred. If omitted, all the available amount is used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of the target creep, or there is a hostile rampart on top of the structure.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The structure does not have the given amount of energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid object which can contain energy.
`ERR_FULL`	-8	The target cannot receive any more energy.
`ERR_NOT_IN_RANGE`	-9	The target is too far away.
`ERR_INVALID_ARGS`	-10	The energy amount is incorrect.

StructureTower



Remotely attacks or heals creeps, or repairs structures. Can be targeted to any object in the room. However, its effectiveness linearly depends on the distance. Each action consumes energy.

Controller level	
1-2	—
3-4	1 tower
5-6	2 towers
7	3 towers
8	6 towers
Cost	5,000
Hits	3,000
Capacity	1,000
Energy per action	10
Attack effectiveness	600 hits at range ≤5 to 150 hits at range ≥20
Heal effectiveness	400 hits at range ≤5 to 100 hits at range ≥20
Repair effectiveness	800 hits at range ≤5 to 200 hits at range ≥20

Inherited from `RoomObject`

`posRoomPosition`

An object representing the position of this object in the room.

Inherited from `RoomObject`

`roomRoom`

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

`hitsnumber`

The current amount of hit points of the structure.

Inherited from `Structure`

`hitsMaxnumber`

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

isActive()

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Inherited from `OwnedStructure`

`myboolean`

Whether this is your own structure.

Inherited from `OwnedStructure`

`ownerobject`

An object with the structure’s owner info containing the following properties:

parameter	type	description
`username`	string	The name of the owner user.

`energynumber`

The amount of energy containing in this structure.

energyCapacitynumber

The total amount of energy this structure can contain.

attack(target)

Remotely attack any creep in the room.

parameter	type	description
`target`	Creep	The target creep.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The tower does not have enough energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid attackable object.
`ERR_RCL_NOT_ENOUGH`	-14	Room Controller Level insufficient to use this structure.

heal(target)

Remotely heal any creep in the room.

parameter	type	description
`target`	Creep	The target creep.

RETURN VALUE

One of the following codes:

constant	value	description

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The tower does not have enough energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid creep object.
`ERR_RCL_NOT_ENOUGH`	-14	Room Controller Level insufficient to use this structure.

`repair(target)`

Remotely repair any structure in the room.

parameter	type	description
`target`	Structure	The target structure.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The tower does not have enough energy.
`ERR_INVALID_TARGET`	-7	The target is not a valid repairable object.
`ERR_RCL_NOT_ENOUGH`	-14	Room Controller Level insufficient to use this structure.

`transferEnergy(target, [amount])`

This method is deprecated and will be removed soon. Please use `Creep.withdraw` instead.

Transfer energy from the structure to a creep. The target has to be at adjacent square.

parameter	type	description
`target`	Creep	The creep object which energy should be transferred to.
`amount` *optional*	number	The amount of energy to be transferred. If omitted, all the remaining amount of energy will be used.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of the target creep, or there is a hostile rampart on top of the structure.
`ERR_NOT_ENOUGH_RESOURCES`	-6	The structure contains less energy than the given amount.
`ERR_INVALID_TARGET`	-7	The specified target object is not a creep.
`ERR_FULL`	-8	The target creep can not carry the given amount of energy.
`ERR_NOT_IN_RANGE`	-9	The target creep is too far away.

StructureWall

Blocks movement of all creeps. Players can build destructible walls in controlled rooms. Some rooms also contain indestructible walls separating novice and respawn areas from the rest of the world or dividing novice / respawn areas into smaller sections. Indestructible walls have no `hits` property.

Controller level	2
Cost	1
Hits when constructed	1
Max hits	300,000,000

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

Inherited from `Structure`

hitsnumber

The current amount of hit points of the structure.

Inherited from `Structure`

hitsMaxnumber

The total amount of hit points of the structure.

Inherited from `Structure`

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

Inherited from `Structure`

structureTypestring

One of the `STRUCTURE_*` constants.

Inherited from `Structure`

destroy()

Destroy this structure immediately.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_BUSY`	-4	Hostile creeps are in the room.

Inherited from `Structure`

`isActive()`

Check whether this structure can be used. If room controller level is insufficient, then this method will return false, and the structure will be highlighted with red in the game.

RETURN VALUE

A boolean value.

Inherited from `Structure`

`notifyWhenAttacked(enabled)`

Toggle auto notification when the structure is under attack. The notification will be sent to your account email. Turned on by default.

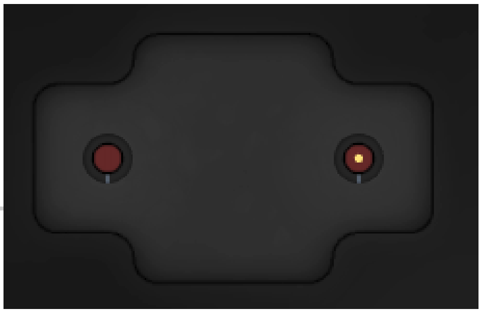
parameter	type	description
`enabled`	boolean	Whether to enable notification or disable.

RETURN VALUE

One of the following codes:

constant	value	description
`OK`	0	The operation has been scheduled successfully.
`ERR_NOT_OWNER`	-1	You are not the owner of this structure.
`ERR_INVALID_ARGS`	-10	`enable` argument is not a boolean value.

Tombstone



A remnant of dead creeps. This is a walkable object.

Decay	5 ticks per body part of the deceased creep

Inherited from `RoomObject`

posRoomPosition

An object representing the position of this object in the room.

Inherited from `RoomObject`

roomRoom

The link to the Room object. May be undefined in case if an object is a flag or a construction site and is placed in a room that is not visible to you.

creepCreep

```
room.find(FIND_TOMBSTONES).forEach(tombstone => {
    if(tombstone.creep.my) {
        console.log(`My creep died with ID=${tombstone.creep.id} ` +
             `and role=${Memory.creeps[tombstone.creep.name].role}`);
    }
});
```

An object containing the deceased creep.

deathTimenumber

Time of death.

idstring

A unique object identificator. You can use `Game.getObjectById` method to retrieve an object instance by its `id`.

storeobject

An object with the tombstone contents. Each object key is one of the `RESOURCE_*` constants, values are resources amounts. `RESOURCE_ENERGY` is always defined and equals to 0 when empty, other resources are undefined when empty. You can use `lodash.sum` to get the total amount of contents.

ticksToDecaynumber

The amount of game ticks before this tombstone decays.