I. THE COMPETITION

THE CONTEST

MechMania is a 24-hour Artificial Intelligence programming competition that takes place during the annual ACM Reflections | Projections conference.

The kickoff for the event will be held at 9:00 pm in room 1404 of the Siebel Center for Computer Science on Friday, October 11th, 2013. The competition will begin at 10:00 pm on Friday, and will end at 10:00 pm on Saturday.

A game and server will be provided to each team. The challenge is to design an Artificial Intelligence client that can play the game per the specifications, and win in a challenge bracket against all other teams. The MechMania Staff will handle the bracket and scoring; your focus should be on your AI.

II. THE MISSION

The year is 2043, and the overpopulation of Earth, depleted natural resources, and introduction of cheap, fast space travel have forced the people of Earth to leave and seek new planets, full of resources and landmasses to squander and overpopulate.

Fortunately, thanks to the unlimited possibilities offered by space, relevance dawns anew, along with incomprehensible mass destruction.

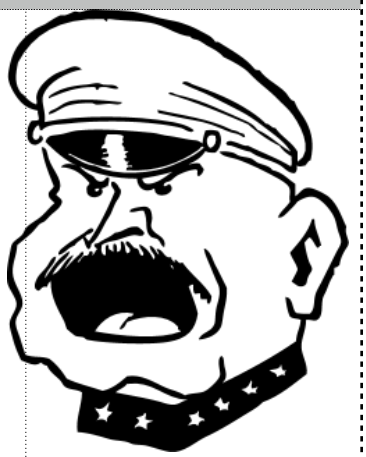
Due to a clerical error, a war has broken out on the small water covered planet of [[REDACTED]]. Mass orbital bombardment was considered too expensive; instead, the allied powers have decided on the equally costly tactic of lifting in prefabricated naval units. You've been selected to ensure victory via any means possible.

MESSAGE FROM THE ADMIRAL

Commander!

Our intelligence reports that an enemy fleet has been deployed to the attached coordinates. Deploy your fleet at once.

You must hold our position on [[REDACTED]]. We're counting on you!



You immediately gather your fellow officers and head out to sea. Seraph Technologies, Incorporated has recently released the MS-8 -- the latest and greatest in naval technologies -- and your army has been lucky enough to get its hands on one. This 500-foot long, uranium fission powered monstrosity has an onboard Seraph NR-3 configuration and control facility, and can automatically reconfigure your naval units on demand using only resources found on the ocean floor. In only a few hours, you can convert any of the generic vessels brought with you into either the Seraph DS-4 Destroyer or PR-2 Pilot Ship. You have a total compliment of nineteen ships to configure before the battle begins.

Unfortunately, due to budget cuts and fear of the "Helvetica Scenario," this transformation capacity only works in hard vacuum -- be ready with a correct configuration once you hit atmosphere. It is important to know that if and when this ship sinks, all others in your fleet will spontaneously explode. We're not sure why this happens, but don't worry; the MS-8 can receive 6 direct missile hits before sinking.

SERAPH TECHNOLOGIES, INC.

AUTOMATIC PRODUCTION NETWORK ONLINE

SERAPH PR-2 PILOT SHIP

HEALTH: 2 MISSILE SHOTS

The PR-2 Pilot Ship is a 200-foot long support ship that uses high-powered magnets and high-resolution radar to scan the bottom of the ocean for usable resources, and retrieve those resources for use by your army. The resource collection process is completely automatic, and those collected will be automatically beamed to your main ship via SeraphNet satellites.

SERAPH DS-4 DESTROYER UNIT

HEALTH: 4 MISSILE SHOTS

The DS-4 Destroyer is a 400-foot long offensive vessel that is capable of firing heavy-duty rockets that are capable of taking down enemy ships in only a few hits. Featuring cutting edge satellite technology, your naval division can remotely fire missiles that are 100% accurate and launched within seconds of ordering the strike.

Because intelligence on the enemy is critical, you have decided to execute military operations, and then pause to perform reconnaissance on the battlefield. Unfortunately, this leaves the perfect opportunity for the enemy to retaliate. Knowledge of the enemy's battle plan is critical to any success on the planet [[REDACTED]].

TOP SECRET

NAVAL STRATEGY HANDBOOK

FIRE	You are able to give any of your destroyer ships the order to fire on a coordinate. Launching a rocket costs 50 resources. After you pause your orders for reconnaissance, friendly satellites will inform you if an enemy vessel was hit by your missile.
SCAN	Each pilot ship will automatically locate and retrieve a constant amount of resources whenever you are in command. This is done without receiving any orders from the commander, and will repeat once per turn. Your main ship will also collect resources.

Due to the massive energy requirements to perform any of these functions in [[REDACTED]]'s unique atmosphere, ships will go into a temporary shutdown mode after completing a given action to rebuild their energy reserves. As resource collection is completely automatic, it is not affected by this power drain.

Since all of the ships you are using are based off of brand new technology, there are some experimental features equipped that can be activated during battle and give your team an edge. These abilities can only be activated once per turn, and cost additional resources.

SERAPH TECHNOLOGIES RESEARCH DIVISION

EXPERIMENTAL TACTICAL OPERATIONS

MOVE	Old earth sea ships moved through water by use of a propeller, traveling in only straight lines. Seraph has been working on something better - the ability to instantly teleport to any valid landing position on the battlefield, for the low cost of only 50 resources times the length of your ship. A side effect of this is that your ships cannot move otherwise.	Resources: 50 * Length All ship types
BURST SHOT	For the cost of 250 Resources, one of your Destroyer Ships can release a series of nine missiles in rapid-fire. However, due to the speed of this launch, these missiles are automatically aimed at a 3x3 grid area; the coordinate on which you are firing will be the center of this grid. The explosion also creates an immense amount of heat--so much heat that your observational satellites won't be able to tell whether or not the missile has struck an enemy unit.	Resources: 250 Destroyer Ships and Main Ship
SONAR	For a cost of 110 Resources, your Pilot Boats can refocus their resource scanners on the battlefield and scan a 5x5 area for enemy presence. Your ship's main computer will then present to you how far away the closest part of an enemy unit is from the center of the 5x5 grid area. The enemy, however, will be able to detect the strong magnetic field created by the operation, and will be notified if one of their ships was detected by your radar.	Resources: 110 Pilot Boats Only

III. THINGS TO KNOW

THE CONTEST

MechMania is a 24-hour Artificial Intelligence programming competition that takes place during the annual ACM Reflections | Projections conference.

The kickoff for the event will be held at 9:00 pm in room 1404 of the Siebel Center for Computer Science on Friday, October 11th, 2013. The competition will begin at 10:00 pm on Friday, and will end at 10:00 pm on Saturday.

A game and server will be provided to each team. The challenge is to design an Artificial Intelligence client that can play the game per the specifications, and win in a challenge bracket against all other teams. The MechMania Staff will handle the bracket and scoring; your focus should be on your AI.

COMPETITION GUIDELINES

To ensure fairness for each participant and team competing in MechMania XIX, MechMania Staff asks that you adhere to the following guidelines during the competition. Failure to adhere to these guidelines will result in the disqualification of your entire team from the competition. MechMania staff always have discretion.

- **Please be respectful to other participants/teams. Keep the competition friendly!**
- **If you encounter a bug in the code provided to you, you must report it to the MechMania Staff immediately. Exploiting a bug in the MechMania server is cheating.**
- **You may not use a library that was designed or created specifically for MechMania. External libraries that were not written for MechMania are allowed.**
- **Your team must write the majority of code that runs your AI Client.**
- **Please do not bring food into the MechMania lab, and keep the lab area clean.**
- **Your AI Client must be committed to the provided repository before the competition closes. Code committed after the deadline will not be run.**

IRC CHATROOM/CHANNEL

The IRC Channel will be the official means of communication during the competition. Within the channel, you can talk with other teams and staff members. There are also a number of commands that may be used to request assistance from the MechMania Staff.

!help <team number> - request assistance from MechMania Staff for your team.

!lastannouncement - view the latest announcement from the MechMania Staff.

If you have an IRC client installed on your computer, you may connect to "irc.freenode.net" and join channel ##mm19.

If you do not have an IRC client installed, you may use the web client at http://acm.uiuc.edu/mm-irc

COMPETITION LOCATION

The MechMania Kickoff and Showdown will be held in room 1404 of the Thomas M. Siebel Center for Computer Science. A map of the Siebel Center can be found in the Reflections | Projections program.

Two workspaces are provided for MechMania participants. Room 0218 of Siebel Center is a computer lab that is available for teams that do not have laptops. Room 0216 of Siebel Center has open workspace for teams that prefer to work on their own laptops.

You may work on MechMania anywhere, even outside of the space provided for MechMania. However, MechMania Staff will only be able to provide support to teams that are in 0216 and 0218 Siebel Center.

NETWORK ACCESS

Competitors that are not students/staff at the University of Illinois should use the "IllinoisNet" wireless SSID, with the credentials located on the back of their name badge. The "UIPublicWifi" SSID may block ports that are necessary or useful for MechMania from crossing the firewall; the firewall on the IllinoisNet network is less strict than all other available networks.

COMPUTER LAB SPACE

The computer lab equipment that is available for use during the MechMania competition is property of the University of Illinois and should be treated with respect. Please clean up your workspace after use, and keep food outside of the lab.

If you are using your own laptop, you may log into a MechMania computer with "ssh <username>@mechmania.acm.uiuc.edu."

Please note that all storage is local to the MechMania computers (including remote workstations) - storage is not synchronized between workstations. If you wish to share files or code, it is recommended you use your team's Git repository to do this.

FOOD AND MEALS

Meals are provided as part of the MechMania registration fee. Your MechMania name badge will allow you to join meals. Snacks and meals will also be brought down to the MechMania competition area. Please keep the lab space clean, and refrain from bringing food into the lab.

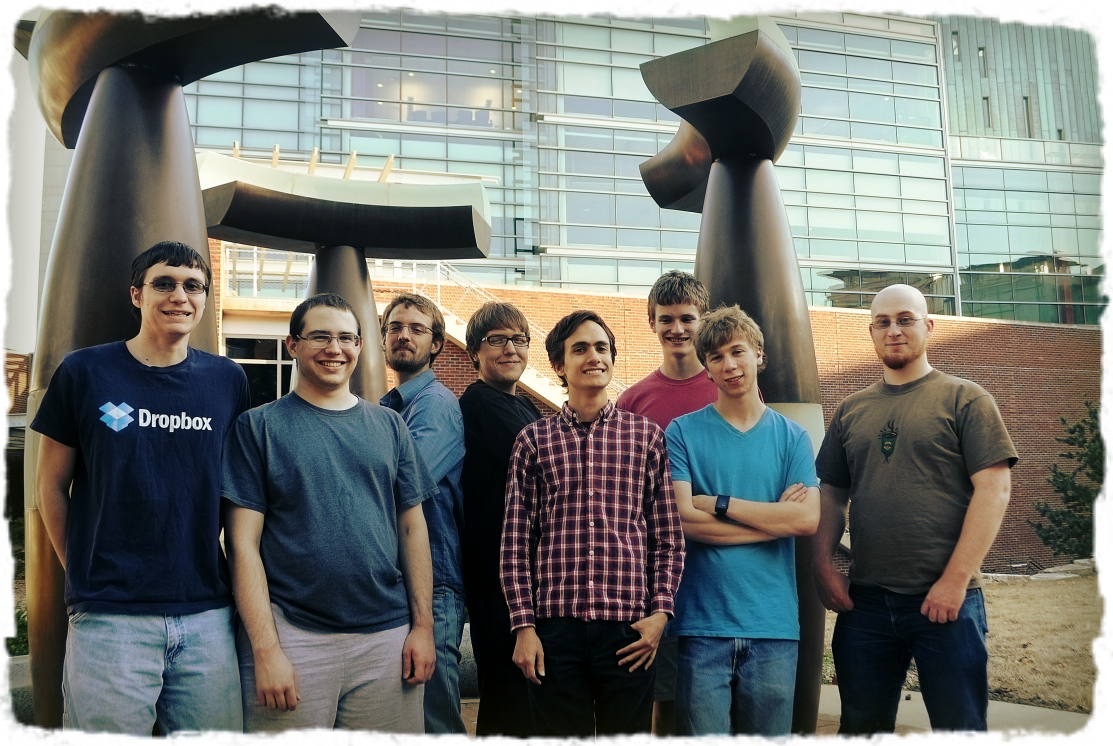
GETTING HELP

If you need assistance from the MechMania Staff, please use the "!help <team #> command within the ##mm19 IRC Channel. This allows MechMania Staff to assist teams in the order that they have requested assistance, and ensure that all teams receive equal and fair treatment from Staff. Please display the provided table tent with your team number on it in your team's work area so that the MechMania Staff can easily find and assist you.

GIT REPOSITORIES

Each team will be provided with a Git repository. The login credentials for your team's repository are provided on the inside cover page of this booklet. At the close of the competition, MechMania Staff will run the most recent commit. Your AI Client will be run on the Lab machines; therefore, it is to your benefit to test your code on a lab machine before the deadline.

MECHMANIA XIX STAFF CREDITS



(from left to right)
Nick Jeffrey, Ace Nassri, Philip Hann, Alex Hendrix, Sam Laane, Evan Schiewe, Calvin Shirley, Josh Ginsburg

Not pictured: Drew Cross, Dylan Nugent, Eric Parsons

IV. OVERVIEW/CODE

GAME OVERVIEW

In this game, you are the commander of a fleet of up to 19 ships, This game is fought in a 1 vs 1, turn-based style on a 100x100 grid. In the beginning of the game, both players place their ships, then they take turns, giving each of their ships an action during a turn. You can win this game in two ways:

1. **Ship the enemy's main ship**
2. **The turn limit has been reached (in which case, there is a tiebreaker)**

SHIP OVERVIEW

Main Ship - Each player must have one, and only one, Main Ship. This ship takes up a 5x1 area, and has 60 health. It generates 30 resources each turn. It is able to fire for 50 resources, and move for 250 resources. If this ship is sunk, the player is defeated. Besides the main ship, the rest of the fleet (up to 18 other ships) can be made up of any combination of the two other types of ships, defined below.

Destroyer Ship - This ship takes up a 4x1 area, and has 40 health. It doesn't generate any resources. It is able to fire for 50 resources, burst shot for 250 resources, and move for 200 resources.

Pilot Ship - This ship takes up a 2x1 area, and has 20 health. This ship cannot attack the other player. It generates 50 resources per turn, can move for 100 resources, and use its sonar ability for 110 resources.

ABILITY OVERVIEW

Each ship has a special set of abilities (defined above) that they can use. There are two types of abilities: normal abilities and special abilities. The special abilities are burst shot, sonar, and move. The normal abilities are fire and resource generation. Only one ship can use a special ability per turn. For example, if a player moves his or her main ship, they cannot move any other ship or have any destroyer ship use burst shot, or have any other pilot ship use sonar. The normal abilities are firing and resource generation; any ship that has the ability to fire can fire during a turn unless the ship is already using a special ability. For example, the main ship cannot move and fire on the same turn, however he/she can move their main ship and fire with all of their destroyers on one turn.

This is all assuming the player has enough resources to do so. Resource generation happens automatically every turn with each ship that has the ability to do so. So, for example, if the player decides to use the sonar ability with his pilot ship, the pilot ship also generates 50 resources that turn. The only way resource generation will not happen is if the ship has sunken.

Fire - both the main ship and destroyer ship can shoot a single missile at the other board. When a player fires at the other board, he will receive a response telling him whether or not he has hit a ship. This ability fails if the player doesn't have enough resources, or if they player gives invalid coordinates. The opposing player will be able to see where you fired. If a shot hits another ship, it does 10 damage to that ship.

Resource Generation - Both the main ship and pilot ship will generate resources as long as they're alive. Resource generation happens automatically every turn. For example, if the player has a main ship and 5 pilot ships, he or she will automatically generate 5 * 50 + 30 = 280 resources per turn.

Moving - Any ship can move. Moving is a special ability. When moving the ship, it can travel to any coordinate on the grid, and switch orientation. This move fails if the player doesn't have enough resources, another ship has already used a special ability during the turn, or the ship intersects with another ship at the destination. The opponent is not notified if a player moves a ship.

Burst Shot - only the destroyer can use burst shot. Burst shot is a special ability. When using burst shot, the player sends a 3x3 grid of missiles, where the specified coordinate is the center of that 3x3 grid. After firing, the player will not be able to see which missiles hit or missed. The opponents will only be able to see the missiles that hit their ship.

Sonar - Only the pilot ship can use sonar. Sonar is a special ability. When using sonar, the player sends a "ping" to the enemy board. This ping encompasses a 5x5 grid on the opponent's board, where the specified coordinate is the center of the 5x5 grid. When the player sends a ping, he also receives a response. This response contains a list of the radial distance from the center of the ping (maximum 2) where the closest part of all pinged ships exist. For example, if part of a pinged ship is in the center of the grid, it returns a distance of 0. The opponent is notified if any of their ships is pinged, but they don't receive the radial distance from the center of the ping.

BOARD OVERVIEW

The board is a 100x100 grid. The coordinate system uses 0-based indexing. The x-axis increases from left to right; the y-axis increases from top to bottom. Therefore, (0,0) is the top left, and (99,99) is the bottom right. When a ship is placed on the board, it is placed with reference to the top-most part (if the ship is placed vertically), or the left-most part (if the ship is placed horizontally).

PLAYING THE GAME

The server and players communicate via JSON and a socket connection that is maintained throughout the entire game.

To start the game, both players send the server a Join Request. If the both Join Requests are valid, the server starts the game. The server first sends s Turn Notification to player 1. Player 1 processes a turn, and sends a Turn Request back to the server. It the server processes the Turn Request, and sends out a Turn Response to Player 1. The server then simultaneously sends out a Turn Notification to Player 2. Player 2 sends back a Turn Request. The server processes that Turn Request and sends a Turn Response to Player 2, simultaneously sends a Turn Notification to Player 1, and the cycle continues. If a player takes too long to send their turn, the server skips that player's turn by sending an Interrupt, and sends a Turn Notification to the other player. This cycle continues until the game ends.

IV. API REFERENCE

Player JSON Object Format

The player sends two types of JSON objects: an initial Join Request, and Turn Requests.

Join Request

This object is only sent once when joining the game. It is the first thing you send to the server. It consists of the following fields:

- playerName : (String) The name of the player to join
- mainShip : (Object) The coordinates/orientation of the main ship
- ships : (Object Array) An array of the up to 18 other ships to be placed on the board

mainShip

The mainShip is formatted in the following way:

- xCoord : (Integer) The x-coordinate to place your main ship
- yCoord : (Integer) The y-coordinate to place your main ship
- orientation : (Character) The orientation of the ship, could be 1 of the 2 below

    H : Horizontal

    V : Vertical

Ship

This is the format of each object in the ships array:

- type : (Character) The type of the ship to be placed, can be 1 of the 2 below (Note: This field cannot be M because the player cannot have more than 1 Main Ship, the Main Ship is defined in the mainShip field above)

    P : Pilot Ship

    D : Destroyer Ship
- xCoord : (Integer) The x-coordinate of the ship to be placed
- yCoord : (Integer) The y-coordinate of the ship to be placed
- orientation : (Character) The orientation of the ship to be placed, can be 1 of the 2 below

    H : Horizontal

    V: Vertical

(Turn Request is on next page for readability)

Turn Request

This object is sent once every turn after a Turn Notification is sent from the server. It contains the following fields:

- playerToken : (String) The token that the player was given during the first server response for identification purposes
- shipActions : (Object Array) An array of actions to carry out this turn

Ship Action

This is the format of each object in the shipActions array:

- ID : (Integer) The ID of the ship to carry out the action
- actionID : (String) The ID of the action to carry out, could be 1 of the 6 below

    F : Fire

    MH : Move, ending up in a horizontal orientation

    MV : Move, ending up in a vertical orientation

    B : Burst Shot (Destroyer ship only)

    S : Sonar (Pilot ship only)

    N : Nothing (This is just to be explicit, not every ship needs a shipAction every turn)
- actionX : (Integer) The x-coordinate of where the action will take place
- actionY : (Integer) The y-coordinate of where the action will take place
- actionExtra : (Integer) This isn't used, but could be in the future, so make this field zero for now

Player Join Request

```
{
	"playerName" : "Roger",
	"mainShip" : {
		"xCoord" : 5,
		"yCoord" : 5,
		"orientation" : "H"
	} ,
	"ships" : [
		{
			"type" : "D",
			"xCoord" : 10,
			"yCoord" : 10,
			"orientation" : "H"
		},
		{
			"type" : "P",
			"xCoord" : 20,
			"yCoord" : 20,
			"orientation" : "V"
		},
		{
			"type" : "D",
			"xCoord" : 30,
			"yCoord" : 30,
			"orientation" : "V"
		},
		{
			"type" : "P",
			"xCoord" : 40,
			"yCoord" : 40,
			"orientation" : "H"
		}
	]
}
```

Player Turn Request

```
{
      “playerToken” : “qwerty123”,
      “shipActions” : [
            {
                  “ID” : 0,
                  “actionID” : “F”,
                  “actionX” : 20,
                  “actionY” : 13,
                  “actionExtra” : 0
            },
           {
                  “ID” : 1,
                  “actionID” : “BS”,
                  “actionX” : 20,
                  “actionY” : 13,
                  “actionExtra” : 0
            },
            {
                  “ID” : 2,
                  “actionID” : “N”,
                  “actionX” : 20,
                  “actionY” : 13,
                  “actionExtra” : 0
            },
            {
                  “ID” : 3,
                  “actionID” : “F”,
                  “actionX” : 5,
                  “actionY” : 4,
                  “actionExtra” : 0
            },
            {
                  “ID” : 4,
                  “actionID” : “N”,
                  “actionX” : 32,
                  “actionY” : 17,
                  “actionExtra” : 0
            }
      ]
}
```

Server Turn Notification


```
{
	"error" : [],
	"responseCode" : 100,
	"playerName" : "Alex",
	"playerToken" : "qwerty123",
	"ships" : [
		{
			"health" : 50,
			"ID" : 0,
			"type" : "M",
			"xCoord" : 5,
			"yCoord" : 5,
			"orientation" : "H"
		},
		{
			"health" : 30,
			"ID" : 1,
			"type" : "D",
			"xCoord" : 10,
			"yCoord" : 10,
			"orientation" : "H"
		},
		{
			"health" : 20,
			"ID" : 2,
			"type" : "P",
			"xCoord" : 20,
			"yCoord" : 20,
			"orientation" : "V"
		},
		{
			"health" : 30,
			"ID" : 3,
			"type" : "D",
			"xCoord" : 30,
			"yCoord" : 30,
			"orientation" : "V"
		},
	],
	"shipActionResults" : [],
	"hitReport" : [],
	"pingReport" : [
		{
			"shipID" : 3
			"distance" : -1
		}
	]

}
```

Server Turn Response


```
{
      “error” : [“Ship 4 did not have enough resources”]
      “responseCode” : 201
      “playerName” : “Roger”,
      “playerToken” : “querty123”
      “resources” : 50,
      “ships” : [
            {
                  “health” : 50,
                  “ID” : 0,
                  “type” : “M”,
                  “xCoord” : 5,
                  “yCoord” : 5,
                  “orientation” : “H”
            },
            {
                  “health” : 30,
                  “ID” : 1,
                  “type” : “D”,
                  “xCoord” : 10,
                  “yCoord” : 10,
                  “orientation” : “H”
            },
            {
                  “health” : 20,
                  “ID” : 2,
                  “type” : “P”,
                  “xCoord” : 20,
                  “yCoord” : 20,
                  “orientation” : “V”
            },
            {
                  “health” : 30,
                  “ID” : 3,
                  “type” : “D”,
                  “xCoord” : 30,
                  “yCoord” : 30,
                  “orientation” : “V”
            },
      ],
      “shipActionResults” : [
            {
                  “ID” : 0,
                  “result” : “S”
            },
            {
                  “ID” : 1,
                  “result” : “S”
            },
            {
                  “ID” : 2,
                  “result” : “S”
            },
            {
```

```
                "ID" : 3,
                "result" : "R"
        }
    ],


    "hitReport" : [
        {
                "xCoord" : 0,
                "yCoord" : 0,
                "hit" : false
        },
        {
                "xCoord" : 10,
                "yCoord" : 10,
                "hit" : true
        }
    ] ,


    "pingReport" : [
        {
                "shipID" : 3
                "distance" : 2
        }
    ]
}
```

Server Interrupt/Error

```
{
      “error” : [“You have gone over time. You must be a teapot.”]
      “responseCode” : 418
      “playerName” : “Alex”
      “playerToken” : “qwerty123”
            “ships” : [
            {
                  “health” : 50,
                  “ID” : 0,
                  “type” : “M”,
                  “xCoord” : 5,
                  “yCoord” : 5,
                  “orientation” : “H”
            },
            {
                  “health” : 30,
                  “ID” : 1,
                  “type” : “D”,
                  “xCoord” : 10,
                  “yCoord” : 10,
                  “orientation” : “H”
            },
            {
                  “health” : 20,
                  “ID” : 2,
                  “type” : “P”,
                  “xCoord” : 20,
                  “yCoord” : 20,
                  “orientation” : “V”
            },
            {
                  “health” : 30,
                  “ID” : 3,
                  “type” : “D”,
                  “xCoord” : 30,
                  “yCoord” : 30,
                  “orientation” : “V”
            },
      ],

      “shipActionResults” : []
      “hitReport” : []
      “pingReport” : []

}
```