Accessing Data

An `HTTP GET` request can be sent to the server in order to easily retrieve data from the server. The response will always be in `text/json`. There are two ways to retrieve data, shown as follows.

Using Query Strings

A request can be sent to `~/read`, with the following parameters specified. + `team` or `competition`, set equal to the desired team or competition. Alternatively, *all* can be requested, returning the entire set of teams or competitions, respectively. + If no parameters are specified in the query, the entire data set will be returned.

Example:

```
~/read?team=5752
```

will return

```
{
  "teamName": "BevBotics",
  "teamNumber": 5752,
  "teamHome": "Beverly, MA",
  "teamColour": "#E67E22",
  "teamRecord": {
    "win": 10,
    "loss": 17,
    "tie": 1
  },
  "scout": {
    "game": [
      "Consistent with the bottom goal.."
    ],
    "pit": {
      "MainPoint": {
        "del": "Gears"
      },
      "ClimbRope": "Yes",
      "SomethingElse": "4x8 (Standard)",
      "hidden": true
    }
  },
  "hidden": true
}
```

Using URL

Any request to `~/read2` will not accept query string parameters. Instead, requests should be formatted like so: `~/read2/[type]/[spec]`. Required parameters are naturally:

- `type`, either *comp* or *team*. Each doing exactly what you would expect.
- `spec`, the specific element to target. If `team` is the specified type, then a *team number* is expected. If `comp` is specified, a *competition name* is expected.

Adding New Data and Modifying Existing Data

> As of current, only scouting data can be modified. *competition* and *team* data **cannot** be modified.

An `HTTP GET` or `POST` request to `~/write` can be made to write to the database. If you do this in any way, then the structure for the specified team will be created if it does not exist (Currently immutatable elements included).

A strung query of `JSON` is expected. The required parameters are

- `sTeamNumber`, the selected team number. Quite simply, this specifies which team to target.
- `chosenOptions`, an *object* in the format

```
{
  "optionName": "optionValue",
  ...
}
```

with as many options as you wish specified. + To write to the GameScout dataset, an external `game` parameter must be set to `true`.

Example: A team has been scouted and the data for `GearAuto` needs to be changed to `true` for team `5752`.

```
~/write?q={"chosenOptions":{"GearAuto": true}, "sTeamNumber": 5752}
```

This will modify the correct values.

Mark Values as Hidden

Complete removal is essentially pointless. There should not be any situation where data is deleted. It can simply be overwritten as specified previously. If, for whatever, reason, this needs to be done, a dataset and all of it's children can be marked as hidden.

An `HTTP DELETE` request must be sent to `~/rm`. The expected format is `~/rm/[type]/[param1]/...`. Up to four additional parameters can be specified, though only one is required.

- `Type` must be either *comp* or *team*.
- `param1` must be either the *team number* or *competition name*
- Params 2-4 must be following children elements.

Example:

```
~/rm/team/5752/scout/pit/GearAuto
```

will target our team's pit scout element *GearAuto*.