Gradebook JSON API

Introduction

JSON objects are loaded via a standard HTTP or HTTPS connection. The Gradebook JSON API only supports authenticated HTTPS. This helps ensure you only get your scores and your data is securely transmitted. The URL to load has two parts. The first part the is base URL. It provides the full path to the actual JSON program. Use the base URL specified in the related lab writeup. The second portion provides the parameters to the JSON program. Those parameters, and how you interpret the results, change depending on which objects you are loading. This document describes the second part of the URL.

Assuming the gradebook URL is https://www.gradebook.com/json and you are attempting to load the list of available sections (record=sections), the complete URL would be https://www.gradebook.com/json?record=sections .

Since the JSON objects are transported within a standard HTTPS connection, you can use a web browser to see the raw JSON. This will help your design and debugging process. Simply type the complete URL into the web browser, enter your CSL credentials, and away you go.

Sections

The section records provide a list of available sections. It includes identifiers that you will need later when accessing more specific score data. Normally there will be one entry in the section list for each course you have taken with me. To retrieve the section list use the parameters:

?record=sections

The result will contain a single object with the name "`sections`". Sections contains an unordered array of objects. Each object describes a single section. The following is an example of the sections list:

```
{
 "sections" : [
     {
          "id" : 10,
          "polynum" : 7324,
          "term" : 2114,
          "termname" : "Spring 2011",
          "dept" : "CPE",
          "course" : "458",
        "title" : "Current Topics in Computer Systems",
        "first_day" : 1301299201,
        "last_day" : 1307865601
    }
]
}
```

Most of the field names are self-explanatory. The days are expressed in seconds since Unix epoch , which is the POSIX standard. Note there may be more that one section in the section array.

Enrollments

The second step is loading the enrollments for the section you are interested in. There is no option to load all enrollments for all sections, so you must provide a way to select a section before loading enrollments. The parameters for loading enrollment data is:

?record=enrollments&term=<TERM>&course=<COURSE>

Where <TERM> is replaced with the numeric "term" value from the section object and <COURSE> is replaced with the numeric "course" value from the section object.

The result will contain a single object with the name "`enrollments`". Enrollments contains an unordered array of enrollment objects. Each object describes a single enrollment. The following is an example of the enrollments list:

```
{
"enrollments" : [
    {
        "id": 318,
        "role": 1,
        "dropped": 0,
        "age": 4,
        "admin_failure": 0,
        "ferpa": 1,
        "major": "CSC",
        "email_level": 1,
        "emplid": 111111111,
        "first_name" : "John",
        "middle_name" : "M.",
        "last_name" : "Bellardo",
        "bb_id" : "00000000000000000000000000000001",
        "username" : "bellardo",
        "csc_username" : "bellardo"
```

```
     }
]
}
```

Again, most of the fields are self explanatory. Age is not your actual, calendar age. It is your university age. 1 indicates you are a freshman, and so on.

Userscores

The userscores record contains the bulk of the data, and is the third step in retrieving your scores. One your App's user has selected a term and enrollment, you can load the userscores object from the following URL:

?record=userscores&term=<TERM>&course=<COURSE>&user=<USER>

<TERM> and <COURSE> are as described in the enrollments section. <USER> is the campus username of the person you wish to view scores for. This comes from the "username" field in the enrollments structure. Note that students only have permission to view their own scores, so substituting someone else's username here won't work. Some non-student people enrolled in the course (e.g., the instructor) can view the scores of any enrolled student.

The result contains a single JSON object whose sole member is an array named "userscores". Each object in the userscores array describes a single assignment. Not all fields are present in all assignments, depending on your enrollment type (student, TA, instructor) and assignment permissions. An example is:

```
{
"userscores" : [
     {
          "id" : 254,
          "max_points" : 10,
          "name" : "Project Proposal",
          "abbreviated_name" : "M1",
          "extra_credit_allowed" : 0,
          "email_notification" : 1,
          "sort_order" : 499,
          "compute_func" : 0,
          "display_type" : 1,
          "due_date" : 1302361198,
          "permissions":          {
               "id": -1,
               "visible": 1,
               "view_files": 1,
               "view_stats": 1,
                "view_histogram": 1,
                "view_properties": 1,
                "view_computed_by": 1
            },
            "scores" : [
                {
                    "counts" : 1,
                    "id" : 1,
                    "score" : 10,
                    "display_score" : "CR",
                    "post_date" : 1302397316,
                    "submitted_work" : {
                        "id" : 1,
                        "mimetype" : "application/pdf",
                        "file_extension" : ".pdf",
                        "url" : "/cgi-bin/..."
                    },
                    "feedback" : {
                        "id" : 2,
                        "mimetype" : "text/plain",
                        "file_extension" : ".txt",
                        "url" : "/cgi-bin/..."
                    }
                }
            ]
        }
]
}
```

The fields in the assignments structure are used as follows:

1. id - the database id for the assignment.
2. max_points - the maximum advertised points available for the assignment.
3. extra_credit_allowed - a boolean value (0 == NO) indicating if extra credit (e.g., scores > max_points) are allowed on the assignment.
4. name - the official, long display name for the assignment.
5. abbreviated_name - a much shorter assignment name, useful when displaying many assignments in a small space. It's mapping to the "name" may not be immediately obvious to anyone except the instructor.
6. email_notification - a boolean value indicating if notification emails (on score posts or changes -- you've seen these) are sent for the assignment.

7. sort_order - a numeric value indicating the instructor's chosen display order. Lower values are displayed first.
8. compute_func - an enumerated type indicating how the score is computed. Most will be the value 0 -- highest score from multiple attempts. Examples of other options are weighted sums and curves.
9. display_type - indicates how the score is to be displayed. It may be the actual value (e.g., 23), CR/NC, or a letter. In general you can ignore this field because the score records already have a translated "display_score" field.
10. due_date - the time the assignment is / was due. If not specified it will have a numeric value of 0.
11. permissions - Your permissions for the given assignment. These are typically used to customize the user interface depending on permissions. For example, including an "edit" button makes no sense if you can't actually change the assignment's properties. The database backend would reject the changes anyway. Sometimes permissions that aren't granted will be omitted from the object and sometimes they will be 0. The current available permissions are
    - i. id - the permission's id
    - ii. person - the person (emplid) or group (-role) the permission applies to
    - iii. visible - can see the assignment exists and the scores
    - iv. view_files - can view any files associated with the scores
    - v. view_stats - can view the summary statistics for an assignment (mean, etc).
    - vi. view_histogram - can view a histogram (summarized score performance of all users) for the assignment
    - vii. view_permissions - can see the full permission set for the assignment
    - viii. create - can create new scores
    - ix. change = can change and existing score
    - x. view_computed_by - can view how each score is computed
12. student_permissions - assignment permissions that apply to the students enrolled in the course. This may be omitted if the person requesting the score data is actually a student.
13. scores - an array of score object. Each object contains a single score for the assignment as follows:
    - i. counts - a boolean value indicating if the score actually counts. If there is only 1 score for an assignment it will always count.
    - ii. id - the score id for non-computed scores. If the score is computed it's id will be -1.
    - iii. score - the numeric score value
    - iv. display_score - a string containing the score that should be displayed to the user. It may exactly match score, or it may contain letters (e.g., CR/NC).

v. post_date - the date the score was posted to the gradebook database.

vi. There are two optional, additional fields that may be present. These are "feedback" and "student_work". "feedback" is for any assessment information the instructor provides to the student via the database. "student_work" is for student assignment submissions that are stored in the database. If present, each of these is a "file" object containing the following fields:

   i. id - the database id of the file

   ii. mimetype - the mime type of the file (e.g., pdf, png, etc). Search online for a more detailed description of mime types and possible values.

   iii. file_extension - the extension that corresponds to the mime type. When determining the type of the file use the mime type. The file extensions are heavily overloaded.

   iv. url - the url where the file is located. This excludes the server name. You must add the server name yourself for this to be useful.

Assignment Statistics

Assuming you have appropriate permissions, you can retrieve summary statistics for an assignment with the asnstats request:

```
?record=asnstats&term=<TERM>&course=<COURSE>&id=<ASNID>
```

<TERM> and <COURSE> have the same meaning as they do in the enrollments record. <ASNID> is the numeric assignment ID whose stats you are interested in. The returned record is self-explanatory:

```
{
    "asnstats" : {
        "id" : 254,
        "points" : 10,
        "unique_students" : 58,
        "min_score" : 10,
        "mean_score" : 10.000000,
        "median_score" : 10,
        "max_score" : 10,
        "std_dev" : 0.000000,
        "attempts" : 58
    }
}
```