Period Countdown Web Specification

Period-Countdown Software Development

Revision 0.3

November 23, 2022

Table of Contents

[Changelog 4](#_Toc115531890)

[Dependencies 5](#_Toc115531891)

[Python Web Server 5](#_Toc115531892)

[Java Transport and Period-Countdown 5](#_Toc115531893)

[Setting Up the Web Server (Development) 6](#_Toc115531894)

[Starting the Web Server 6](#_Toc115531895)

[Building for the Web 6](#_Toc115531896)

[General Command Information 7](#_Toc115531897)

[Default Payload Headers 7](#_Toc115531898)

[Input Payload 7](#_Toc115531899)

[Output Payload 7](#_Toc115531900)

[URL Tree 7](#_Toc115531901)

[Opcode Quick Reference 7](#_Toc115531902)

[Adding New Commands 9](#_Toc115531903)

[Python Web Server 9](#_Toc115531904)

[Java Transport 9](#_Toc115531905)

[Get Resources 10](#_Toc115531906)

[Get Available Schools 10](#_Toc115531907)

[Used By 10](#_Toc115531908)

[Transport Opcode 10](#_Toc115531909)

[Transport Input Payload 10](#_Toc115531910)

[Transport Output Payload 10](#_Toc115531911)

[Get Next Up List 10](#_Toc115531912)

[Used By 10](#_Toc115531913)

[Transport Opcode 10](#_Toc115531914)

[Transport Input Payload 10](#_Toc115531915)

[Transport Output Payload 11](#_Toc115531916)

[Get Period Numbers 11](#_Toc115531917)

[Used By 11](#_Toc115531918)

[Transport Opcode 11](#_Toc115531919)

[Transport Input Payload 11](#_Toc115531920)

[Transport Output Payload 11](#_Toc115531921)

[Get Time Remaining 11](#_Toc115531922)

[Used By 11](#_Toc115531923)

[Transport Opcode 12](#_Toc115531924)

[Transport Input Payload 12](#_Toc115531925)

[Transport Output Payload 12](#_Toc115531926)

[Get User Period 12](#_Toc115531927)

[Used By 12](#_Toc115531928)

[Transport Opcode 12](#_Toc115531929)

[Transport Input Payload 12](#_Toc115531930)

[Transport Output Payload 12](#_Toc115531931)

[Set Resources 13](#_Toc115531932)

[Login User 13](#_Toc115531933)

[Used By 13](#_Toc115531934)

[Transport Opcode 13](#_Toc115531935)

[Transport Input Payload 13](#_Toc115531936)

[Transport Output Payload 13](#_Toc115531937)

[None 13](#_Toc115531938)

[Set Current School 13](#_Toc115531939)

[Used By 13](#_Toc115531940)

[Transport Opcode 13](#_Toc115531941)

[Transport Input Payload 13](#_Toc115531942)

[Transport Output Payload 13](#_Toc115531943)

[Set Next Up 14](#_Toc115531944)

[Used By 14](#_Toc115531945)

[Transport Opcode 14](#_Toc115531946)

[Transport Input Payload 14](#_Toc115531947)

[Transport Output Payload 14](#_Toc115531948)

[Set User Period 14](#_Toc115531949)

[Used By 14](#_Toc115531950)

[Transport Opcode 14](#_Toc115531951)

[Transport Input Payload 14](#_Toc115531952)

[Transport Output Payload 14](#_Toc115531953)

# Changelog

|  |  |
| --- | --- |
| **Version** | **Commentary** |
| 0.1 | Initial draft of specification |
| 0.2 | * Minor change to the output payload of GetTimeRemaining * Minor corrections for starting and building the web server |
| 0.3 | * Removed deprecated command LoginUser * Minor corrections |

Note: comments <MARK: in this format> are points that should be addressed as quickly as possible.

# Dependencies

## Python Web Server

Python 3.9 or above - <https://www.python.org/downloads/>

pip 3 or above - built-in with most Python installations

Flask - pip3 install flask - <https://flask.palletsprojects.com/en/2.2.x/>

## Java Transport and Period-Countdown

Java 17 or above - <https://www.oracle.com/java/technologies/downloads/>

Google GSON 2.2.2 or above - packaged with PeriodCountdown-web.jar

# Setting Up the Web Server (Development)

## Starting the Web Server

Copy the release/PeriodCountdown-X.X.X-web folder to the host machine. cd to the keys folder and run ./gen\_keys.sh [OPTIONS]. While none of the command line options are needed, they should be set according to the registrant information for the domain; the password and certificate duration should also be changed from the defaults for security reasons.

With keys generated, run the following to host the Java transport:

java -jar path/to/PeriodCountdown-web.jar [OPTIONS]

The available options for the transport are:

|  |  |
| --- | --- |
| **Option** | **Commentary** |
| --transport-ip <IP ADDR> | Starts the transport with IP "IP\_ADDR". If this IP cannot be reached, a fatal error will be thrown and the transport will exit |
| --transport-port <PORT> | Binds the transport to the port "PORT". If this port cannot be bound to, a fatal error will be thrown and the transport will exit |

Finally, once the transport has started, start the Python web server with the preferred framework (Apache2, Gunicorn, etc.)

## Building for the Web

To create a new web build of Period Countdown, run:

./compile.sh -t web

./jar.sh -t web

./build -t web

Follow the instructions above for starting the web server.

# General Command Information

## Default Payload Headers

This information is included with every input and output payload sent between the Python web server and the Java transport. It includes identifying information for the command being processed and the user being served.

### Input Payload

{

"Opcode": "<OPCODE>",

"UserID": "<USER IDENTIFIER>",

"InputPayload": {

<INPUT PAYLOAD>

}

}

### Output Payload

Note that if the ReturnCode field is not 0, the OutputPayload may not contain the expected data.

{

"Opcode": "<OPCODE>",

"UserID": "<USER IDENTIFIER>",

"ReturnCode": "<RETURN CODE>",

"OutputPayload": {

<OUTPUT PAYLOAD>

}

}

## URL Tree

https://<domain>/

settings/

login/

callback/

logout/

## Opcode Quick Reference

* GetAvailableSchools
* GetNextUpList
* GetPeriodNumbers
* GetTimeRemaining
* GetUserPeriod
* SetCurrentSchool
* SetNextUp
* SetUserPeriod

# Adding New Commands

## Python Web Server

1. Add an opcode variable and any new error variables to the commands.py file
2. Create a new method in commands.py with the following format:

def NAME\_HERE(send\_socket: SocketType, user\_id: str, /\* parameres here \*/) -> str:

if (not isinstance(send\_socket, SocketType) or

not isinstance(user\_id, str) or

/\* parameter checks here \*/):

return return\_err("NAME\_HERE called with invalid args", "", "", ERR\_INVALID)

message: str = construct\_msg(NAME\_HERE, user\_id, {/\* parameters here \*/})

return handle\_msg(send\_socket, message)

## Java Transport

1. Add an opcode variable and any new error variables to the Command class
2. Create a new file in the web.transport.commands package that extends the Command class. Use the following template:

package web.transport.commands;

import com.google.gson.Gson;

import com.google.gson.JsonSyntaxException;

import com.google.gson.annotations.SerializedName;

public class NAME\_HERE extends Command {

public class InputPayload {

/\* parameters here \*/

}

public class OutputPayload {

/\* parameters here \*/

}

@SerializedName("InputPayload")

public InputPayload inputPayload;

@SerializedName("OutputPayload")

public OutputPayload outputPayload;

@Override

public String process(String request, SchoolAPI schoolAPI, UserAPI userAPI) throws JsonSyntaxException {

Gson gson = new Gson();

NAME\_HERE command = gson.fromJson(request,NAME\_HERE.class);

String opcode = command.opcode;

String userID = command.userID;

/\* process here \*/

NAME\_HERE response = new NAME\_HERE();

response.outputPayload = response.new OutputPayload();

response.opcode = opcode;

response.userID = userID;

response.returnCode = Command.SUCCESS;

/\* set parameters here \*/

return gson.toJson(response);

}

}

1. Write the process(String, SchoolAPI, UserAPI): String routine for the new command
2. Add the command parse case to the switch statement in SSLServer.handleRequest()

# Get Resources

## Get Available Schools

Requests a list of supported schools, including the one currently chosen by the user. If the user is not logged in, CurrentSchool will be the MVHS school file (as defined by the User.json template file).

### Used By

GET @ https://<domain>/settings/schoolinformation/

### Transport Opcode

GetAvailableSchools

### Transport Input Payload

None

### Transport Output Payload

"OutputPayload": {

"CurrentSchool": "<CURRENT SCHOOL FILE>",

"AvailableSchools": [

"<SCHOOL FILE>",

"<SCHOOL FILE>",

...

]

}

## Get Next Up List

Requests a list of strings describing the upcoming classes in the day. If the user is logged out, this data is taken from the generic User.json template file and MVHS school file as needed to construct a UserAPI and SchoolAPI object. If the user is logged in and has the Next Up feature disabled, the returned array is empty (length 0).

### Used By

GET @ https://<domain>/

GET @ https://<domain>/settings/nextup/

### Transport Opcode

GetNextUpList

### Transport Input Payload

None

### Transport Output Payload

"OutputPayload": {

"NextUp": "<NEXT UP VERBOSITY>",

"NextUpList": [

"<NEXT CLASS>",

"<NEXT CLASS>",

...

]

}

## Get Period Numbers

Requests an array of valid period numbers. These can be used in the Type field of the GetUserPeriod command. They will always be integers. If the user is logged out, they will not be able to access the settings/ URL tree.

### Used By

GET @ https://<domain>/settings/classinformation/

### Transport Opcode

GetPeriodNumbers

### Transport Input Payload

None

### Transport Output Payload

"OutputPayload": {

"PeriodNumbers": [

"<NUM>",

"<NUM>",

...

]

}

## Get Time Remaining

Requests the time remaining until the start of the next counted period. If the user is logged out, time will be calculated based on the User.json template file and MVHS school file as needed to construct a UserAPI and SchoolAPI object.

### Used By

GET @ https://<domain>/

### Transport Opcode

GetTimeRemaining

### Transport Input Payload

None

### Transport Output Payload

"OutputPayload": {

"End": "<END TIME STAMP>"

"TimeRemaining": "<TIME REMAINING>"

}

## Get User Period

Requests an user period. If no user period exists, Name will be "Summer", Status will be "Free", and the other fields will be blank. If the user is not logged in, Name will be the Name of the school period for MVHS, Status will be the Type of the school period for MVHS, and the other fields will be blank.

### Used By

GET @ https://<domain>/

GET @ https://<domain>/settings/classinformation/

### Transport Opcode

GetUserPeriod

### Transport Input Payload

None, to get the current user period, or

"InputPayload": {

"Type": "<PERIOD NUMBER>"

}

### Transport Output Payload

"OutputPayload": {

"Name": "<NAME>",

"Status": "<STATUS>",

"Teacher": "<TEACHER>",

"Room": "<ROOM>"

}

# Set Resources

## Set Current School

Sets the user's chosen school file. If the user is not logged in, they will not be able to access the settings/ URL tree.

### Used By

PUT @ https://<domain>/settings/schoolinformation/

### Transport Opcode

SetCurrentSchool

### Transport Input Payload

"InputPayload": {

"CurrentSchool": "<CURRENT SCHOOL FILE>"

}

### Transport Output Payload

None

## Set Next Up

Sets the next up feature's verbosity. If the user is not logged in, they will not be able to access the settings/ URL tree.

### Used By

PUT @ https://<domain>/settings/nextup

### Transport Opcode

SetNextUp

### Transport Input Payload

"InputPayload": {

"NextUp": "<VERBOSITY>"

}

### Transport Output Payload

None

## Set User Period

Sets information for a user period. If the user is not logged in, they will not be able to access the settings/ URL tree.

### Used By

PUT @ https://<domain>/settings/classinformation/

### Transport Opcode

SetUserPeriod

### Transport Input Payload

"InputPayload": {

"Type": "<PERIOD NUMBER>",

"Name": "<NAME>",

"Teacher": "<TEACHER>",

"Room": "<ROOM>"

}

### Transport Output Payload

None