The Replay Protocol for DerbyNet

The replay kiosk provides video replay capability for DerbyNet. It communicates with the DerbyNet web server using the communication protocol described in this document.

Messages Sent To The Server

The replay kiosk polls the server by sending HTTP POST requests to server, normally several times per second. If the server doesn't get a replay message for two seconds, it will report to the user that the replay connection has been lost.

The body of the replay-to-server POST request is in `application/x-www-form-urlencoded` format, and comprises three parameters:

- **`action`**: This always has the value **`replay-message`** to identify the message as part of the replay protocol.
- **`status`**: This has an integer value summarizing the status of the replay application:
  - 0: ready/idle
  - 1: recording
  - 2: playing back
  - -1: connecting
  - -2: No video source
  - -3: No audio source
  - -4: Recording error
- **`finished-replay`**: This has value 1 for the very first request after replay finishes playing back a video; otherwise 0.

Responses From The Server

If the server recognizes a replay message, it responds with an HTTP response whose body is a simple XML document.

The response's XML document will include zero or more `<replay-message>` elements. Each of these elements has a text body which is a command to the replay application. The possible replay commands are:

•"HELLO" is the server's response to a first connection, confirming that a connection has been successfully established.

- Example: **`<replay-message>HELLO</replay-message>`**

•"TEST *skipback showings rate*" runs the replay test or demo clip. *skipback* tells how many milliseconds from the end of the clip should be shown. This is repeated *showings* times, and playback is at *rate* percent of normal speed (e.g., 100 is normal speed playback).

- Example: **`<replay-message>TEST 3500 2 75</replay-message>`**

•"START *video_name_root*" starts a recording video (video_name_root is a suggested file name stem).

- Example: **`<replay-message>START Bears_Round1_Heat02</replay-message>`**

•"RACE_STARTS *skipback showings rate*" sets a deadline for playing the next replay. If a REPLAY message is not received within approximately *skipback* milliseconds, replay begins as if it had been.

•"REPLAY *skipback showings rate*" stops the recording if one is in progress, and plays back the last *skipback* milliseconds of that recording. This is repeated *showings* times, and playback is at *rate* percent of normal speed (e.g., 100 is normal speed playback).

- Example: **`<replay-message>REPLAY 3000 2 100</replay-message>`**

•"CANCEL" cancels (abandons) the current recording

- Example: **`<replay-message>CANCEL</replay-message>`**

It is NOT necessary for the replay application to track cookies sent from the server.

Example Initial Message Exchange

The following is an example initial message exchange between replay and the server.

Replay sends an initial introductory message:

```
POST /derbynet/action.php HTTP/1.1
Host: localhost
Content-Type: application/x-www-form-urlencoded
Connection: keep-alive
Accept: */*
User-Agent: MacDerbyReplay/1 CFNetwork/893.13.1 Darwin/17.3.0 (x86_64)
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Content-Length: 49

action=replay-message&status=-1&finished-replay=0
```

(Note the single blank line separating the headers from the body.) The response from the server is:

```
HTTP/1.1 200 OK
Date: Tue, 09 Jan 2018 23:01:17 GMT
Server: Apache/2.4.28 (Unix) LibreSSL/2.2.7 PHP/7.1.7
X-Powered-By: PHP/7.1.7
Set-Cookie: PHPSESSID=0e39d9b53b7e4a56ee88ed88d780f50f; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 206
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/xml; charset=utf-8

<?xml version="1.0" encoding="UTF-8"?>
```

```
<action-response action="replay-message" status="-1" finished-replay="0">
<replay-message>HELLO</replay-message>
<success/>
</action-response>
```

Note the single `<replay-message>` element carrying a “HELLO” message.