Recording file format

for Playbook “Unobtrusive Data Collection”

Each recording of a Playbook session (where a “session” occurs from the time the site is visited until the window is closed or refreshed) is stored in a file named something like 2015-07-30T180032\_923Z\_8147.log. (The name is not significant except that it is designed to be unique across all missions and starts with a timestamp approximately equal to the file’s creation date and to the timestamp embedded in the file contents.) It consists of a large number of comma-separated value entries, one per line, with a few fields per line.

Opening in a spreadsheet program (most will be able to view it, as long as they tolerate a different number of cells in each row) is one way of getting a sense of the content of a file. Looking at the raw text is another, but to keep from being overwhelmed by the long lines starting with DOM (whose purpose we’ll get to later), a good practice would be to either truncate the lines to a reasonable length using the Unix command cut -c1-80, or filter out all DOM changes with the Unix command grep -v ^DOM.

After are a few informational header rows, the rest of the file consists of two broad types of rows: rows that recording something that happened, interspersed with timestamp rows. If a sequence of several things happen immediately after one another (less than 10 milliseconds apart) there will be no timestamps between those rows. There are two types of rows that recording something that happened: something changing on the screen (Document Object Model changes), and everything else, including user actions such as touch events.

# Header

The first few lines contain some non-recurring information.

recordingId=unique id, same as filename

clientAddress=numbers and colons

An internal address and port that could probably be traced to a real machine by correlating it with NGINX logs but is otherwise not very useful.

This may be repeated later in the file if the browser lost and reestablished its connection with the server.

t=absolute time

Format is ISO-8601. Largely self-explanatory; note that “Z” = “Zulu” = UTC = GMT. The timestamp lines after this, described later, are all incremental changes to this.

browser,[HTTP user agent info](http://www.useragentstring.com)

describing browser type and OS

DOM,initial,very long escaped-HTML string

A snapshot of was displayed on the screen at the time the session started. The Document Object Model lines after this, described later, are all incremental changes to this. The third field is the HTML for the entire document, with a few characters, notably double-quote, escaped (using the same coding used in URLs) to avoid being mistaken for multiple fields.

window,innerHeight and window,innerHeight

Window size. Technically these are not header info but recur occasionally through the file: whenever the window is resized or the iPad rotated, and sometimes for no reason.

# Timestamps

Timestamps which show how much time has passed since the previous timestamp. Those rows have no comma; they consist of a lowercase t and plus sign (+) followed by a whole number of milliseconds. For example, a line like

t+60011

might appear when Playbook is idle except that the clock in the sidebar (hidden or not) is ticking once every minute, approximately. A series of touch events caused by a quick swipe might have timestamps more like t+15. Frames recording an animation will also have small timestamps between them.

The absolute time at any point in the recording should, in principle, be derivable by adding all preceding t+ lines to the t= line near the top.

# Recorded information

The non-timestamp lines after the initial header lines break down into two broad categories:

* 1. Document Object Model changes.

Captures what was displayed on the screen, sort of like a video.

* 1. Everything else.

Includes user actions, as well as states such as scroll position, zoom level, and window size, and high-level Playbook states such as current view and day.

## Document Object Model changes

The [DOM](http://www.w3.org/DOM/) is the HTML that the browser renders into what is presented to the user. Playbook, like any JavaScript-based client-side application, can change the Document Object Model and the changes are immediately reflected on the screen. Together with the scroll position, the DOM determines what the user sees in the window. For example, changing the style attribute of a rectangle can change its color or move it to a new position on the screen, and removing an element from the DOM makes it disappear from the screen. These low-level changes are recorded in lines starting with DOM, (or, viewed as a spreadsheet, rows whose first field is DOM). They are all relative to the initial document contents in the initial line.

Essentially, each row, or each run of rows uninterrupted by a timestamp, is like a video frame, but it captures incremental changes to the HTML markup rather than pixel-level changes. This information, along with scroll position and little else, is enough to visually replay what the user saw on the screen, as if it were a video.

### Specific DOM changes:

DOM,@=,attribute,element,value

DOM,@+,attribute,element

DOM,@-,attribute,element

Indicates that a certain attribute of a certain HTML element was given a new value or removed entirely. The attribute is identified by its name and the element by its nasa-recorder-id attribute.

DOM,El+

DOM,El-

DOM,AddElBack

Indicates that a certain HTML element was added or removed. If the element was removed and later added back, its contents are not restated in the recording.

DOM,Txt+

DOM,Txt-

DOM,Txt=

Indicates that a certain HTML element had its text content set, removed, or replaced.

## User actions and other types of events

mouse,over

mouse,out

mouse,move

touch,touchend

touch,touchmove

touch,touchstart

click,command

tap,command

hold,command

Records mouse and touch events.

drag,dragging

dragend,endDrag

Records dragging events such as self-scheduling.

open-hyperlink,element,URL

Records a link being clicked on.

txtsel,attribute,element,start,end

Records what part of an input field has the insertion cursor or highlighted selection.

key,element,key-code,insertion-point

Records typing (one key press).

text,element,text-contents

Records the text content of a text input field at the end of typing. This is included because it’s impossible to reconstruct from the key presses and selection alone, in general. There may have been auto corrections, predictive typing, and personal abbreviations defined in Settings, for example.

state,state,view,view-name

state,state,day,number

Records which part of Playbook is currently being displayed.

scroll,element,left,top

Records which part of Playbook is currently being displayed.

window,blur

window,focus

Records when the browser window or tab gains or loses focus, e.g. when buried.