Contents

[Scoping the work, relations. 2](#_Toc372985681)

[User flow 2](#_Toc372985682)

[Entering experiment information 2](#_Toc372985683)

[Defining session types 2](#_Toc372985684)

[Defining sessions 2](#_Toc372985685)

[Defining users 2](#_Toc372985686)

[Defining behavior codes 3](#_Toc372985687)

[Generating events 3](#_Toc372985688)

[Project structure (\*.badass files) 3](#_Toc372985689)

[Resultant CSV files 4](#_Toc372985690)

[Exporting a given session-type 4](#_Toc372985691)

[Exporting all session-types 5](#_Toc372985692)

[Sketches 5](#_Toc372985693)

[Project manager 5](#_Toc372985694)

[Working window 8](#_Toc372985695)

[Creating behavior codes (event types) 9](#_Toc372985696)

[Creating users 10](#_Toc372985697)

[Defining sessions 11](#_Toc372985698)

[Editing event parameters 11](#_Toc372985699)

[Navigation 12](#_Toc372985700)

[Time estimates 13](#_Toc372985701)

# Scoping the work, relations.

A single experiment can have many sessions. There can be different types of sessions.

A video or audio file corresponds to a single (session, session type) pair.

All results of an experiment are kept in a BADASS project. There is one-to-one relationship between experiment and BADASS project.

# User flow

In order to use the program, user:

1. Creates a BADASS project and enters information about experiment, its sessions, and users.
2. Defines experiment-specific types of behavior codes. These and only these codes can be used in the given BADASS project.
3. Works with a session:
   1. Loads or creates a particular session by selecting session name and its type
   2. Loads video(s)/audio files
   3. Adjusts start times of all the videos/audio files by setting start time markers
   4. Generates all events
   5. Saves the project
   6. Exports results to \*.CSV

# Entering experiment information

\*badass file contains the following general information about experiment:

1. Experiment name (string).

# Defining session types

Each session can be of one of predefined types. User defines possible session types before defining sessions. When user defines a session type, BADASS saves the following:

1. Name of session type (string). Uniquely identifies the type across BADASS project.

# Defining sessions

In order to generate events, user first has to create a list of sessions to work with. When user defines a session, BADASS saves the following:

1. Session type (predefined). One of predefined session types.
2. Session name (string). Not unique, but pair (Session name, session type) uniquely identifies session across BADASS project.
3. A set of video and audio files for this session. For each file, its name is kept in \*.badass file.
4. Information about adjustments of start times of all video and audio files.
5. Generated events (at later phase). Later, when user generates events, they are saved in the appropriate session.

# Defining users

User defines a list of experiment participants for a BADASS project. Any defined user can then be used in any (session, session type) pair. It’s not required, but possible that one participant takes part in several sessions. For every user, BADASS saves the following:

1. User name (string). Not unique.
2. User id (integer). Automatically generated integer id that uniquely identifies the user across BADASS project.
3. User color (integer). Integer that encodes user color.

# Defining behavior codes

User defines behavior codes specific to a particular experiment in every BADASS project. There are 2 types of behavior codes – primary and secondary.

When defining a primary code, user enters the following:

1. Name of the code (string). Examples are “badge created”, “window closed”.
2. Keyboard shortcut. The shortcut that can be used later to generate events of this type.
3. List of associated secondary codes that are available for this primary code. The list can be empty. There could be up to 5 secondary behavior codes.

When defining a secondary code, user enters the following:

1. Name of the code (string). Examples are “with left hand”, “with brain control”.
2. Keyboard shortcut. The shortcut that can be used later to set secondary code in the primary code.

# Generating events

When user generates an event, the following properties are recorded:

1. Event timestamp (integer). Number of seconds elapsed since the start of the timeline. In a video or audio, this can map to any play position, depending on start time marker set for the video/audio.
2. User (predefined). One of predefined users for the current (session, session type) pair. The user who generated this event.
3. Event type. One of predefined constants. User defines possible event types for each BADASS project.
4. Note (string). Arbitrary string attached to the event.
5. Information about which of the secondary codes are set.

Information about session and session type is not recorded at the event level as user cannot generate events for different (session, session type) pairs at the same time.

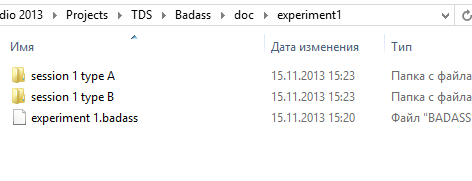
# Project structure (\*.badass files)

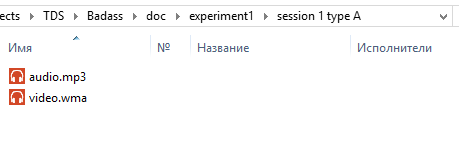
A single project is kept in a single folder that has \*.badass file.

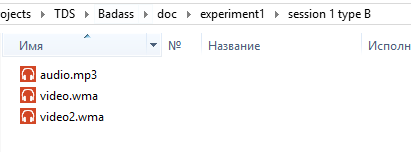
A \*badass file contains names of all video and audio files used in the project. All these files are required to reside in the same folder as the \*.badass file or in its subfolders. They should be copied there manually if necessary.

Also \*badass file contains all other information about sessions and generated events.

Here is an example of BADASS project layout, with media files kept in subfolders:







## Resultant CSV files

Some of the data available in \*.badass can be exported to CSV for external use. There are 2 types of export – export of full project and export of single session-type.

### Exporting a given session-type

A \*.csv file contains information for a single (session, session type) pair. When exporting, BADASS will name the file following the pattern “<session name>.<session type>.csv”. Here is the format:

Event1Timestamp, Event1UserId, Event1UserName, Event1Type, Event1Note, Event1Subcode1, Event1Subcode2, Event1ubcode3, Event1Subcode4, Event1Subcode5

Event2Timestamp, Event2UserId, Event2UserName, Event2Type, Event2Note, Event2Subcode1, Event2Subcode2, Event2ubcode3, Event2Subcode4, Event2Subcode5

…

EventNTimestamp, EventNUserId, EventNUserName, EventNType, EventNNote, EventNSubcode1, EventNSubcode2, EventNSubcode3, EventNSubcode4, EventNSubcode5

Where:

1. Event[i]Timestamp (integer) is the number of seconds elapsed since the start of the timeline by the time the event happened
2. Event[i]UserId (integer) is automatically generated id of the user, unique across BADASS project
3. Event[i]UserName (string) is non-unique name of the user
4. Event[i]Type (string) type of event, unique across BADASS project
5. Event[i]Note (string) arbitrary note about the event
6. Event[i]Subcode[j] is null if event[i] doesn’t define subcode[j] (less than j subcodes are defined for event[i]), true if subcode[j] is set for event[i], false if subcode[j] is not set for event[i].

### Exporting all session-types

When exporting a full project, \*.csv file is named by default identically to the project, but can be renamed to anything. Here is the format of resulting CSV file:

<SessionName[i]><SessionType[j]>, NumEvents[i,j]

Event1Timestamp, Event1UserId, Event1UserName, Event1Type, Event1Note, Event1Subcode1, Event1Subcode2, Event1ubcode3, Event1Subcode4, Event1Subcode5

Event2Timestamp, Event2UserId, Event2UserName, Event2Type, Event2Note, Event2Subcode1, Event2Subcode2, Event2ubcode3, Event2Subcode4, Event2Subcode5

…

EventNTimestamp, EventNUserId, EventNUserName, EventNType, EventNNote, EventNSubcode1, EventNSubcode2, EventNSubcode3, EventNSubcode4, EventNSubcode5

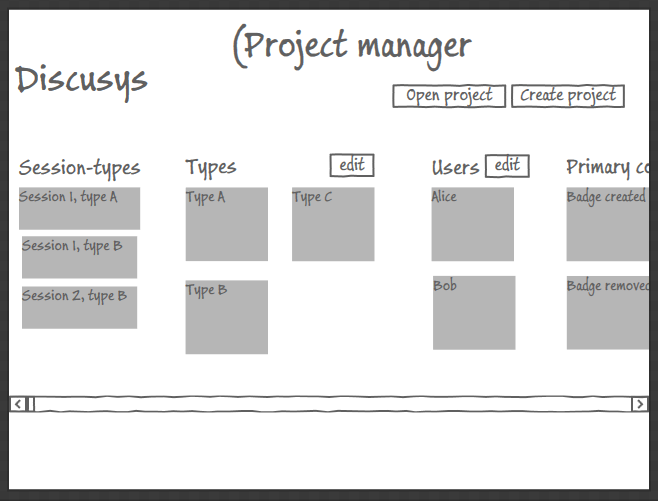
Where:

* NumEvents[i,j] is the number of events generated in <SessionName[i]><SessionType[j]>. Exactly NumEvents[i,j] lines like EventNTimestamp, EventNUserId, EventNUserName, EventNType, EventNNote, EventNSubcode1, EventNSubcode2, EventNSubcode3, EventNSubcode4, EventNSubcode5 follow.
* After that, another block for another session-type starts. On its first line it has a header of the same format <SessionName[i]><SessionType[j]>, NumEvents[i,j] and event lines follow.
* So CSV file enumerates all session-types in the project.

# Sketches

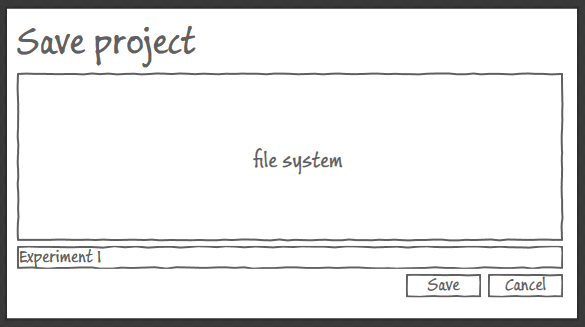
## Project manager

Start window is Project manager. Project manager presents information related to a single project.



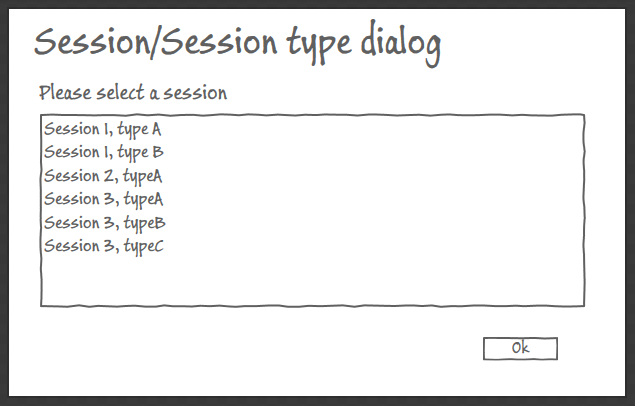
* **Session-types.** By clicking any session-type, we open working window with timeline with that session-type.
* **Types**. Shows types of session. [Edit] button opens Session manager.
* **Users.** List of users that can be used in all session of current project. Edit button opens User manager.
* **Primary codes.** Shows primary behavior codes. Edit button opens Behavior code manager.

“Create project” button opens up Save dialog that accepts experiment (and project) name:



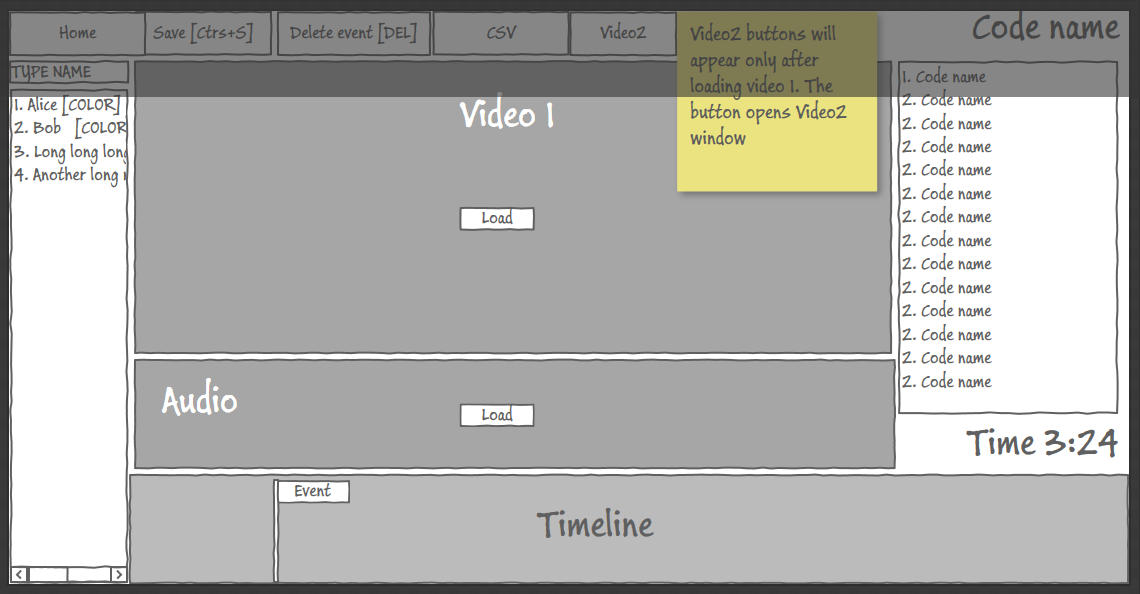
And after that the project manager will show the new empty project.

“Open a session” button opens another dialog:



Here user can select (session, session type) pair and after clicking Ok button working window will be opened with the selected session/session type loaded:

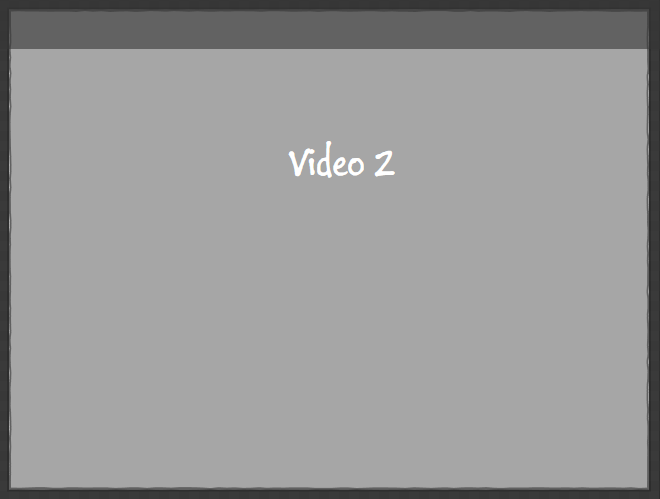
## Working window



* Home button – returns to Project manager window.
* Save button – saves all changes in the currently open BADASS project.
* Delete event – deletes currently selected timeline event, if any.
* CSV – exports events from the current (session, session type) to a csv.

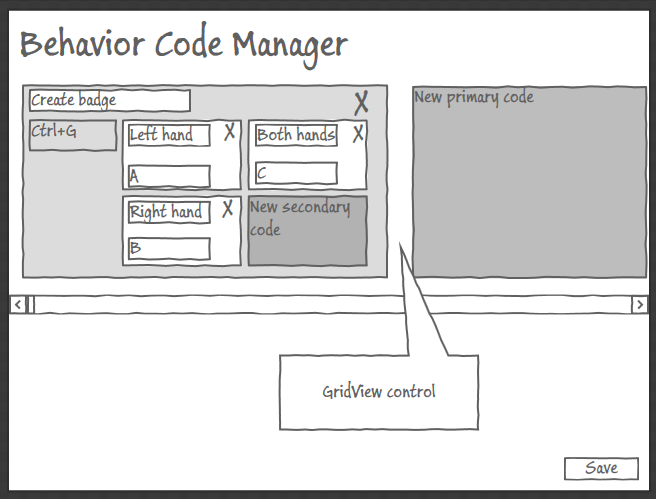
Start markers will be set in this window too. User loads video or audio and sets start time marker at any position.

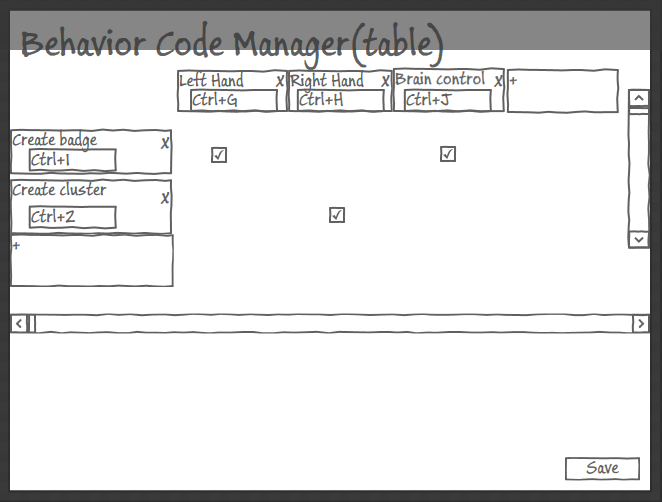
### Video 2 window



The window only show 2nd video.

## Creating behavior codes (event types)

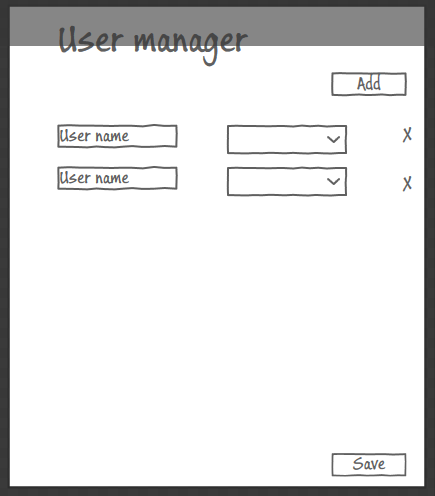
old version



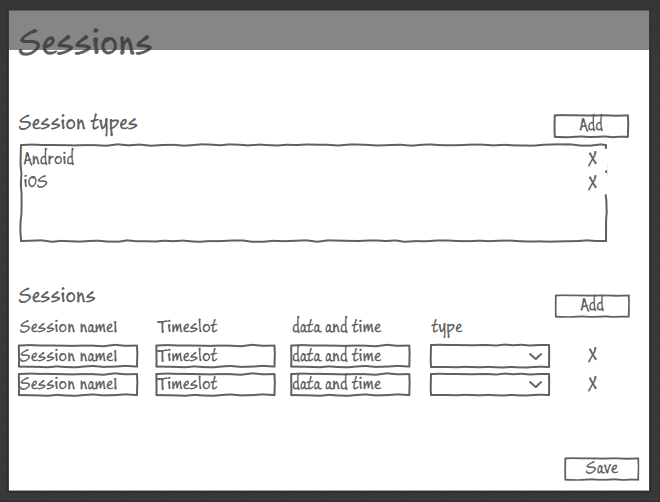
Lets edit primary and secondary behavior codes. Each primary code takes 1 row in the grid. Each secondary code takes 1 column in the grid.

* **New primary code**. “+” button (bottom row). Works like button and adds a new primary code to the collection.
* **New secondary code**. “+” button (rightmost column). Adds a new column for a secondary code.
* Any code, primary or secondary, can be renamed by focusing name textbox and entering a name.
* **X buttons** delete codes.
* **Checkboxes are set to make particular secondary codes available for primary codes.**
* Focusing a shortcut textbox and pressing a keyboard shortcut captures the shortcut and shows it in the textbox
* Save button saves any changes in codes in the current BADASS project.

## Creating users



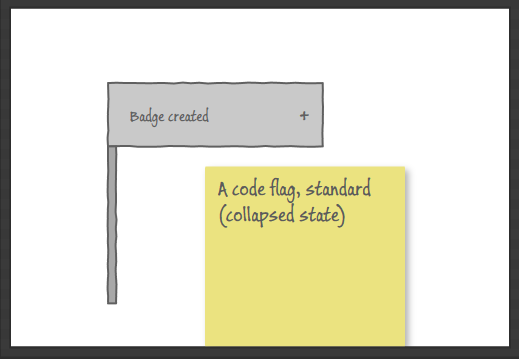
* Add button – creates a new user.
* X button – removes user
* Save – saves all changes in user list, of any. User list is saved in the current project.

Defining sessions

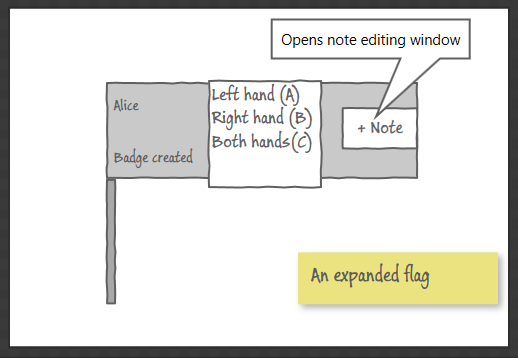
* Top list defines session types. Names of session types can be edited.
* Bottom list of sessions. “Type” column is combobox showing selection of the predefined session types.
* Save button saves any changes in session types and sessions.
* When removing a non-empty session (at least 1 generated event), there is warning. If such a session is removed, all its generated events are also removed.
* When removing a session type that’s used in at least one session, there is error. First all sessions using the type need to be removed or switch to other types before the type can be removed.

## Editing event parameters

Each event is represented as a flag on timeline. There are two states of flags, collapsed and expanded. By default, all flags are collapsed. A collapsed flag shows only name of its primary behavior code:



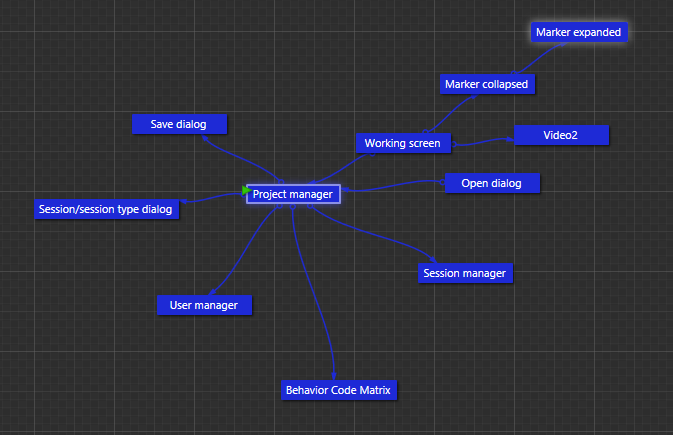
Whenever user hovers the mouse over a collapsed flag, or clicks dropdown button, the flag is expanded:



In expanded state, the user can change secondary codes or add note/edit note.

## Navigation

Here is a graph showing all windows and navigation between them:



## Time estimates

Here is my cautious time estimate:

* Object model for projects (Project, session, session types, events, users) and persistence of project information – 10 hours
* CSV export of single session-type – 7 hours
* CSV export of whole project (all session-types) – 7 hours
* Project manager UI and view models – 12 hours
* Session-type dialog and view models – 2 hours
* Adjustment of start times of all videos and audio – 10 hours
* Working window (with timeline) UI and models – 15 hours
* Behavior code manager UI and models – 12 hours
* User manager UI and models – 3 hours
* Session manager UI and models – 8 hours
* Event flag UI – 8 hours
* Testing – 15 hours
* Total 109 hours.