TilBuci

Movie Format Specification

TilBuci movies are saved as a collection of JSON-formatted files along with all required media, everything placed inside a *.movie* folder with a defined structure. This document describes this structure and all expected files inside of it.

Lucas Junqueira, July 11th 2024

1. The *.movie* folder

All TilBuci movies are saved inside a folder with the *.movie* “extension” like *tilbuci.movie*. While the editor is being used to create content, it manipulates JSON-formatted text files inside this folder to describe your creation, and also saves all the required media files inside of it.

The *.movie* folder has a fixed structure, as follows (files in blue, folders in orange).

- collection
- media
  - audio
  - font
  - html
  - picture
  - spritemap
  - video
- scene
- movie.json
- strings.json

While all media files are stored inside the *media* folder and its subfolders, the *collection* one holds descriptions of these media. The *scene* folder contains the published versions of all movie scenes.

2. movie.json

At the root of the *.movie* folder there mus always be *movie.json* file. This is the main movie file as it holds all settings for the loaded movie. It is a standard JSON file like the example below.

```
{
    "version": 1,
    "id": "tilbuci",
    "author": "Lucas Junqueira <lucas@var.art.br>",
    "copyright": "Lucas Junqueira",
    "copyleft": "Lucas Junqueira",
    "title": "TilBuci",
    "description": "Time to discover the Buci story!",
    "tags": ["til", "buci"],
    "favicon": "social\/favicon.png",
    "image": "social\/shareimage.png",
    "start": "c2dd3343eb2b1ef50071785fe0deae0c",
    "acstart": "{ \"ac\": \"run\", \"param\": [ \"clear all variables\" ] }",
    "screen": {
        "big": 1920,
        "small": 1080,
        "type": "both",
        "bgcolor": "0x000000"
    },
    "time": 1,
    "origin": "alpha",
    "animation": "cubic.out",
    "fonts": [{
            "name": "roboto",
            "file": "roboto.woff2"
        }
    ],
    "style": ".title {\n\tfont-family: \"Averia Serif GWF\";\n\tcolor:
#FFEF00;\n\tfont-weight: bold;\n\tfont-size: 60;\n\tmargin-bott0m:
20px;\n}\n\n.item {\n\tfont-family: \"Averia Serif GWF\";\n\tcolor:
#FFFFFF;\n\tfont-weight: bold;\n\tfont-size: 40;\n\tmargin-bottom:
10px;\n\tmargin-top: 10px;\n}\n\n.text {\n\tfont-family: \"Averia Serif
GWF\";\n\tcolor: #FFFFFF;\n\tfont-size: 30;\n\tmargin-bottom: 5px;\n}\n\na
{\n\tfont-family: \"Averia Serif GWF\";\n\tcolor: #FFEF00;\n\ttext-decoration:
underline;\n}",
    "actions": [{
            "name": "clear all variables",
            "ac": "[\n\t{ \"ac\": \"int.clearall\", \"param\": [ ] }, \n\t{
\"ac\": \"bool.clearall\", \"param\": [ ]  }, \n\t{ \"ac\":
\"float.clearall\", \"param\": [ ]  }, \n\t{ \"ac\": \"string.clearall\",
\"param\": [ ]  }\n]"
        }
    ],
    "theme": "{\"themeColor\":\"0x000000\"}",
    "texts": {
        "Dog name": "Buci"
    },
    "numbers": [],
    "flags": [],
    "created": "2024-06-24 17:21:40",
    "updated": "2024-07-07 12:44:27",
    "plugins": {
        "original.debug": {
            "active": true,
            "config": []
        },
        "original.share": {
            "active": true,
            "config": []
        }
    }
}
```

These arethe required elements for this file:

version

The current file versiom (currently 1).

id

The movie ID. It is expected to be the same as the .movie folder name.

author

Name of the movie author.

copyright

Copyright disclaimers.

copyleft

Copyleft disclaimers.

title

The movie title (also usef for content sharing).

description

A small text about the movie (also used for sharing).

tags

Tags related to the movie (also used for sharing).

favicon

Icon image for browser tabs. The picture file must be saved at the *media/picture* folder.

image

Image used for content sharing on social media. The picture file must be saved at the *media/picture* folder.

key

A secret key the visitor must provide to access the content. It must be a blank string (no access limitations) or an encrypted one that will be checked on movie open.

start

Initial scene ID (just the ID, without the *.json* extension – the scene file must be saved at the *scene* folder).

acstart

Actions to run on movie load (JSON-formatted text of a standard TilBuci action).

screen

The screen element holds information about te movie display. It contains 4 sub elements:

- big: bigger size of the movie display (points)
- small: smaller size of the movie display (points)
- type: screen orientation type (*horizontal*, *vertical* or *both*)
- bgcolor: movie background color in hex 0x###### format

time

The time among keyframes on animation, in seconds (accepts float values).

origin

Display images appear/disappear mode (currently available ones are *alpha* – images are shown with an opacity effect – and *center* – images arw shown with an animation from the screen center.

animation

Animation method used on the movie. Available options can be found here:
https://github.com/openfl/actuate/tree/master/src/motion/easing

fonts

Fonts used only on the movie that are not availabe to the TilBuci system. This element is an array of objects with two properties: the font name (*name*) and the font file (*file*). The file must be saved inside the *media/font* folder.

style

A CSS-formatted text with styles for HTML texts.

actions

An array with the movie action snippets. Each snippet is described by an object with two properties: *name*, the snippet name, and *ac*, the actions as a JOSN-formatted string.

theme

Adjusts for TilBuci UI colors for the movie as a JSON-formatted string.

texts

A list of the movie global texts and their values.

numbers

A list of the movie global numbers and their values.

flags

A list of the movie global flags and their values.

created

Date/time of the movie creation.

updated

Date/time of the movie properties last update.

plugins

Information about movie plugin usage as a group of objects named after the plugin ID, with two parameters: *active*, true/false – whether the plugin is used on the movie -, and *config*, an array with plugin configuration for the movie.

3. strings.json

This file is intended to hold large amount of texts that will be used on your content to simplify its handling. It is a standard JSON file with group names as the first level of parameters (*simple* and *complex* at the example below). Inside these groups pairs os string names/values can be set ti be used at runtime. Here is an example:

```
{
    "simple": {
        "thank": "thanx"
    },
      "complex": {
        "thank": "thank you a lot"
    }
}
```

4. media

The *media* folder is used to store all media files used by the movie. It holds several folders according to the media type for organization. Sub folders may be created inside them to simplify the content creation and media location.

audio

For audio medlia like .mp3 files.

font

Additional font files must be stored here.

html

HTML files used for text must be placed here.

picture

The place for image files like png and jpg.

spritemap

The folder for png files used on spritemap animations.

video

Save all video files here.

5. collection

TilBuci movies organize their media files inside *collections*. Everytime a media is placed on a movie, it is incorporated as an *asset* of a *collection*. Creators may not even notice that by using several one-file collections, but more sophisticated movies can use the various collection properties, like transitions, to enhance the experience.

The *collection* folder holds several JSON files. The file names indicate the collection ID. Avoid the usage of special chars and spaces on these Ids. The TilBuci editor always use a MD5 encoded string to avoid unsafe chars.

Here is an example of a collection file holding two assets.

```
{
    "id": "2eac758ad6ae2feac8c4a080f1bf779c",
    "movie": "tilbuci",
    "name": "slideshow",
    "transition": "right",
    "time": 0.75,
    "assets": {
        "e08fe7a0e9": {
            "order": 0,
            "name": "slide1",
            "type": "picture",
            "time": 3,
            "action": "next",
            "frames": 1,
            "frtime": 100,
            "file": {
                "@1": "slideshow\/slide1.jpg",
                "@2": "slideshow\/slide1.jpg",
                "@3": "slideshow\/slide1.jpg",
                "@4": "slideshow\/slide1.jpg",
                "@5": "slideshow\/slide1.jpg"
            }
        },
        "0fe4f8b595": {
            "order": 1,
            "name": "slide2",
            "type": "picture",
            "time": 3,
            "action": "next",
            "frames": 1,
            "frtime": 100,
            "file": {
                "@1": "slideshow\/slide2.jpg",
                "@2": "slideshow\/slide2.jpg",
                "@3": "slideshow\/slide2.jpg",
                "@4": "slideshow\/slide2.jpg",
                "@5": "slideshow\/slide2.jpg"
            }
        }
    }
}
```

The requires parameters on collection files are:

id

The collection ID (must be the same of the file name).

movie

The movie ID.

name

The collection name.

transition

When used on an instance with changing assets, like a slideshow, this sets the animation used. Possible values are *right*, *left*, *up*, *down*, *alpha* and *none*.

time

The change asset transition time in secons (accept float values).

assets

A group of objects describing the collection assets.

5.1. Asset descriptions

All collection assets are described by several properties. At the collection file, every asset object is named after the asset ID (can be any text, but the TilBuci editor keeps them as truncated MD5 encoded strings). Here are the required properties:

order

The asset order in the collections, starting at 0 and continuiosly incremented.

name

The asset name.

type

Asset type. Possible ones are *audio*, *html*, *paragraph*, *picture*, *shape*, *spritemap* and *video*.

time

Asset total time for all types except audio and video (they use the actual media total time).

action

Action to take when the asset reaches the end of its time. Possible values are *loop*, *next*, *previous*, *stop*. Besides these simple values, a complex TilBuci interaction action description can be used as well.

frames

Number of frames for spritemap assets.

frtime

Frame time for spritemap assets in miliseconds.

file

The path to the media file stored at the media sub folder related to the asset type. This property is, in fact, a group of 5 elements: @1 to @5. The numbers are related to the real display size according to the movie design one. If the display size is lower then 1.5 times the design one, the @1 media file is used. From 1.5 to 2.5, the @2 is used. @3 is used from 2.5 to 3.5, @4 from 3.5 to 4.5 and @5 for all above. This enables a simple multiple resolution support for the movie.

Assets of paragraph type don’t use files. The actual paragraph texts are stores at the @1 to @5 file elements. Assets of shape type also don’t use files. Instead, the @1 to @5 elements hold a JSON-formatted string with the shape description, with these fields:

- type: the shape type – allowed values are *circle*, *triangle*, *isoscelestriangle*, *righttriangle*, *square*, *pentagon*, *hexagon*, *heptagon*, *octagon*, *eneagon* and *decagon*
- color: shape background color in 0x###### hex format
- alpha: shape opacity form 0.0 to 1.0
- border: border line size (0 for none)

- bdcolor: border color in 0x###### hex format
- bdalpha: border opacity from 0.0 to 1.0
- rotation: shape rotation from 0 to 359

6. scene

Every TilBuci movie is made of several scenes. Their descriptions are saved as JSON files inside the scene folder. The file must be named after tshe scene ID. Avoid special chars and spaces on these ones. The TilBuci editor always use MD5 encoded strings as scene IDs.

A sample scene file is shown below:

```
{
    "title": "Page 14",
    "id": "8c24a9b1d1012e1777b0265fc1c4c34f",
    "movie": "tilbuci",
    "about": "",
    "image": "",
    "navigation": {
        "up": "2427ce6be1c257af12f9c43c5efb2bc4",
        "down": "",
        "left": "d2ff8d65f685b9f0268b962baaf70354",
        "right": "",
        "nin": "",
        "nout": ""
    },
    "collections": ["f83be46442163608f70314115770f718",
"4aa344057a12a0d7e5635ebfe3fb564f" ],
    "loop": 0,
    "acstart": "[\n\t{ \"ac\": \"string.set\", \"param\": [ \"pageid\",
\"$_SCENEID\" ] }, \n\t{ \"ac\": \"string.set\", \"param\": [ \"pagetitle\",
\"$_SCENETITLE\" ] }\n]",
    "ackeyframes": ["", "", "", ""],
    "keyframes": [
    ]
}
```

These are the required properties:

title

The scene name (alos used for sharing).

id

The scene ID (must be the same used on file name).

movie

The movie ID.

about

A scene description (also used for sharing).

image

An image for social media sharing (mus be saves at the media/picture folder).

navigation

Scene IDs for content navigation. This propoerty is an object with 6 elements: *up*, *down*, *left*, *right*, *nin* and *nout*. Each of them may receive a scene ID to be loaded after a navigation command.

collections

A list of the collection IDs used on the scene.

loop

The keyframe to loop to when the scene reaches its end (0 to return to the initial keyframe).

acstart

Actions to run when the scene starts,

ackeyframes

An array of strings containing the actions to run at the end of each scene keyframe.

keyframes

An array of keyframe descriptions with elements in the order they shoul be played on the scene.

6.1. keyframe descriptions and instances

Each scene keyframe is described by an object with a collection of *instances*. Check out a keyframe description example below:

```
{
    "background": {
        "collection": "f83be46442163608f70314115770f718",
        "asset": "cf02971b47",
        "action": "",
        "play": true,
        "horizontal": {
            "order": 0,
            "x": 0,
            "y": 0,
            "alpha": 1,
            "width": 1920,
            "height": 1080,
            "rotation": 0,
            "visible": true,
            "color": "0xFFFFFF",
            "colorAlpha": 0,
            "volume": 1,
            "pan": 0,
            "blur": "",
            "dropshadow": "",
            "textFont": "sans",
            "textSize": 20,
            "textColor": "0xFFFFFF",
            "textBold": false,
            "textItalic": false,
            "textLeading": 10,
            "textSpacing": 0,
            "textBackground": "",
            "textAlign": "left"
        },
        "vertical": {
            "order": 0,
            "x": 0,
            "y": 0,
            "alpha": 1,
            "width": 1080,
            "height": 1920,
            "rotation": 0,
            "visible": true,
            "color": "0xFFFFFF",
            "colorAlpha": 0,
            "volume": 1,
            "pan": 0,
            "blur": "",
            "dropshadow": "",
            "textFont": "sans",
            "textSize": 20,
            "textColor": "0xFFFFFF",
            "textBold": false,
            "textItalic": false,
            "textLeading": 10,
            "textSpacing": 0,
            "textBackground": "",
            "textAlign": "left"
        }
    },
    "buci": {
        "collection": "4aa344057a12a0d7e5635ebfe3fb564f",
        "asset": "25a4eed6ba",
        "action": "",
        "play": true,
        "horizontal": {
            "order": 1,
            "x": 287,
            "y": 50,
            "alpha": 1,
            "width": 1346,
            "height": 790,
            "rotation": 0,
            "visible": true,
            "color": "0xFFFFFF",
            "colorAlpha": 0,
            "volume": 1,
            "pan": 0,
            "blur": "",
            "dropshadow": "",
            "textFont": "sans",
            "textSize": 20,
            "textColor": "0xFFFFFF",
            "textBold": false,
            "textItalic": false,
            "textLeading": 10,
            "textSpacing": 0,
            "textBackground": "",
            "textAlign": "left"
        },
        "vertical": {
            "order": 1,
            "x": -35,
            "y": 278,
            "alpha": 1,
            "width": 1150,
            "height": 929,
            "rotation": 0,
            "visible": true,
            "color": "0xFFFFFF",
            "colorAlpha": 0,
            "volume": 1,
            "pan": 0,
            "blur": "",
            "dropshadow": "",
            "textFont": "sans",
            "textSize": 20,
            "textColor": "0xFFFFFF",
            "textBold": false,
            "textItalic": false,
            "textLeading": 10,
            "textSpacing": 0,
            "textBackground": "",
            "textAlign": "left"
        }
    }
}
```

This keyframe is made of two instances, background and buci, each one with their own descriptors. A single scene can cotain from one to many of these keyframe objects. Each keyframe object may contain many instances. The instance objects has their own requires properties, as follows.

collection

The collection ID to load the assets from.

asset

The asset ID to load (actions may change this at runtime).

action

The action to run when the instance is clicked/touched while the movie is playing.

play

Whether the instance media starts playing just after load.

horizontal and vertical

These properties describe the instance display setting for both horizontal and vertical display. Each of them receive an object with the following elements:

- order: instance order in display list – must start at 0 and be sequential up to the last instance
- x: x position
- y: y position
- alpha: instance opacitu from 0.0 to 1.0
- width: width size
- height: height size
- rotation: instance rotation from 0 to 359 (experimental)
- visible: true/false whether the instance should be displayed
- color: overlay color value as a 0x###### hex string
- colorAlpha: color overlay opacity from 0.0 to 1.0
- volume: sound volume form 0.0 to 1.0
- pan: sound pan from -1.0 to 1.0 (not yet active)
- blur: blur effect description as an array of values
- dropshadow: dropshadow effect description as an array of values
- textFont: text face name to use
- textSize: font size in ponts

- textColor: font color on 0x###### hex string format
- textBold: text diplayed as bold?
- textItalic: text displayed as italic?
- textLeading: text line leading in pixels
- textSpacing: inner letter spacing (not yet active)
- textBackgroud: block text backgrond color in hex 0x###### format – blank string for no background
- textAlign: paragraph alignment