# bodymovin
After Effects plugin for exporting animations to svg/canvas/html + js or natively on Android and iOS through Lottie (https://medium.com/airbnb-engineering/introducing-lottie-4ff4a0afac0e)

Installation instructions:

- Extract content and search for the .zxp file from '/build/extension'
- Use the ZXP installer (http://aescripts.com/learn/zxp-installer/) from aescripts.com.

V 5.8.1

- FEATURE: added more methods available for the lottie worker player

V 5.8.0

- FEATURE: lottie player as a worker support
- FEATURE: added content-visibility prop support
- FEATURE: added header check for json type (improves initial parsing specially on large files)
- FEATURE: added lottie.useWebWorker method to support loading file on a separate web worker (different than the first feature)
- FEATURE: added frame rate to comp on export (not supported by payers yet)
- FIX: random seed expression fix

V 5.7.14

- FEATURE: added support for Smoothness property on animated texts
- FEATURE: added support for animated text path properties
- FEATURE: added support for exporting source names as ids
- FEATURE: added clear cache button
- FIX: fixed importer with last keyframe

V 5.7.13

- FEATURE: added support for emoji modifiers and sequences
- FEATURE: added support for metadata export on json files

V 5.7.12

- FEATURE: added support for masked strokes line cap and line join
- FIX: callbacks list executing beyond index list
- FIX: missing TextExpressionSelectorPropFactor
- FEATURE: exporter improvements

V 5.7.11

- FIX: another exporter fix

V 5.7.10

- FIX: exporter fix

V 5.7.9

- FEATURE: added support for prefixig ids
- FEATURE: security updates
- FIX: fix image sequencing preloading
- FEATURE: added support for gradient fill expressions

V 5.7.8

- FIX: disable screen reader visibility of created font-nodes (thanks @)
- FEATURE: added FootageElement that supports json data layer types with expressions

V 5.7.7

- FIX: xhr open order
- FEATURE: added markers support
- FIX: repeaters with reduced copies
- FIX: unintentional stroke clipping on shapes with large stroke-width (thanks Manan Jadhav)
- FIX: ie11 append missing
- FIX: repeater calling trim twice

V 5.7.6

- FIX: es6 support on export
- IMPROVEMENT: added linting rules
- FIX: seedrandom and main fix
- FIX: image export fix

V 5.7.5

- FIX: Enable HImageElement to use preloaded images thanks @dbettini
- FIX: lottie-light effects breaking on destroy
- FIX: loaded_images event fired in safari
- FIX: original asset names for audios

V 5.7.4

- FIX: `addEventListener` method returns function except `void`
- EXPRESSIONS: added support for toWorldVec, fromWorldVec and getValueAtTime for transforms
- EXPRESSIONS: fixed propertyGroup expression
- FIX: added svg effects placeholder
- FIX: added check for wrapper on destroy canvas
- EXPORTER: missing layer styles
- EXPORTER: export only work area

V 5.7.3

- EXPRESSIONS: Added more expressions support

V 5.7.2

- FIX: Trusted Types compliance by removing calls to .innerHTML
- FIX: make callback parameter of removeEventListener optional
- FEATURE: Audio Support

V 5.7.1

- REPORT: Improved animation report
- FIX: Expressions separate dimensions
- FIX: propertyGroup for expressions (Duik bones are supported)
- FEATURE: supported Pucked and Bloat

V 5.7.0

- FEATURE: Extension: reports for not supported features
- FEATURE: Extension: baking keyframes for unsupported expressions
- FEATURE: Extension: improved preview and added Skottie preview

Lottie + Bodymovin

Lottie is the native engine that Airbnb's awesome team built. It uses Bodymovin as the animation exporter and is the ideal complement for getting animations to play natively everywhere.
Follow these links to get each player:

- Android's player (https://github.com/airbnb/lottie-android)
- iOS's player (https://github.com/airbnb/lottie-ios)
- React Native's wrapper (https://github.com/airbnb/lottie-react-native)

Lottie and AVD

Some animations can be exported for Android using the AVD format.
It can fit for some cases where you'll gain a performance improvement.
But Lottie brings much more features, a level of animation control and dynamic loading that couldn't be achieved with avd.
Here's a link (http://airbnb.io/lottie/lottie-avd.html) with a full comparison of both technologies.

After installing

- Go to Edit > Preferences > General > and check on "Allow Scripts to Write Files and Access Network"

HTML player installation

```
# with npm
npm install lottie-web

# with bower
bower install bodymovin
```

Or you can use the script file from here:
https://cdnjs.com/libraries/bodymovin
Or get it directly from the AE plugin clicking on Get Player

Demo

See a basic implementation here. (https://codepen.io/airnan/project/editor/ZeNONO/) <br/>

Examples

See examples on codepen. (http://codepen.io/collection/nVYWZR/) <br/>

How it works

Here's (https://www.youtube.com/watch?v=5XMUJdjI0L8) a video tutorial explaining how to export a basic animation and load it in an html page <br />

After Effects

- Open your AE project and select the bodymovin extension on Window > Extensions > bodymovin
- A Panel will open with a Compositions tab listing all of your Project Compositions.

- Select the composition you want to export.
- Select a Destination Folder.
- Click Render
- look for the exported json file (if you had images or AI layers on your animation, there will be an images folder with the exported files)

HTML

- get the lottie.js file from the build/player/ folder for the latest build
- include the .js file on your html (remember to gzip it for production)

```
<script src="js/lottie.js" type="text/javascript"></script>
```

You can call lottie.loadAnimation() to start an animation.
It takes an object as a unique param with:

- animationData: an Object with the exported animation data.
- path: the relative path to the animation object. (animationData and path are mutually exclusive)
- loop: true / false / number
- autoplay: true / false it will start playing as soon as it is ready
- name: animation name for future reference
- renderer: 'svg' / 'canvas' / 'html' to set the renderer
- container: the dom element on which to render the animation

It returns the animation instance you can control with play, pause, setSpeed, etc.

```
lottie.loadAnimation({
  container: element, // the dom element that will contain the animation
  renderer: 'svg',
  loop: true,
  autoplay: true,
  path: 'data.json' // the path to the animation json
});
```

Composition Settings:

Check this wiki page for an explanation for each setting.
https://github.com/airbnb/lottie-web/wiki/Composition-Settings

Usage

Animation instances have these main methods:

play

stop

pause

setLocationHref(href)

- `href`: usually pass as `location.href`. Its useful when you experience mask issue in safari where your url does not have `#` symbol.

setSpeed(speed)

- `speed`: 1 is normal speed.

goToAndStop(value, isFrame)

- `value`: numeric value.
- `isFrame`: defines if first argument is a time based value or a frame based (default false).

goToAndPlay(value, isFrame)

- `value`: numeric value.
- `isFrame`: defines if first argument is a time based value or a frame based (default false).

setDirection(direction)

- `direction`: 1 is forward, -1 is reverse.

playSegments(segments, forceFlag)

- `segments`: array. Can contain 2 numeric values that will be used as first and last frame of the animation. Or can contain a sequence of arrays each with 2 numeric values.
- `forceFlag`: boolean. If set to false, it will wait until the current segment is complete. If true, it will update values immediately.

setSubframe(useSubFrames)

- `useSubFrames`: If false, it will respect the original AE fps. If true, it will update on every requestAnimationFrame with intermediate values. Default is true.

destroy()

getDuration(inFrames)

- `inFrames`: If true, returns duration in frames, if false, in seconds.

Aditional methods:

- updateTextDocumentData -- updates a text layer's data
  More Info (https://github.com/airbnb/lottie-web/wiki/TextLayer.updateDocumentData)

Lottie has several global methods that will affect all animations:

lottie.play() -- with 1 optional parameter **name** to target a specific animation <br/>
lottie.stop() -- with 1 optional parameter **name** to target a specific animation <br/>
lottie.goToAndStop(value, isFrame, name) -- Moves an animation with the specified name playback to the defined time. If name is omitted, moves all animation instances.<br />
lottie.setSpeed() -- first argument speed (1 is normal speed) -- with 1 optional parameter **name** to target a specific animation <br/>
lottie.setDirection() -- first argument direction (1 is normal direction.) -- with 1 optional parameter **name** to target a specific animation <br/>
lottie.searchAnimations() -- looks for elements with class "lottie" or "bodymovin" <br/>
lottie.loadAnimation() -- Explained above. returns an animation instance to control individually. <br/>
lottie.destroy(name) -- Destroys an animation with the specified name. If name is omitted, destroys all animation instances. The DOM element will be emptied.<br />
lottie.registerAnimation() -- you can register an element directly with registerAnimation. It must have the "data-animation-path" attribute pointing at the data.json url<br />
lottie.getRegisteredAnimations() -- returns all animations instances<br />
lottie.setQuality() -- default 'high', set 'high','medium','low', or a number > 1 to improve player performance. In some animations as low as 2 won't show any difference.<br />
lottie.setLocationHref() -- Sets the relative location from where svg elements with ids are referenced. It's useful when you experience mask issues in Safari.<br />
lottie.freeze() -- Freezes all playing animations or animations that will be loaded<br />
lottie.unfreeze() -- Unfreezes all animations<br />
lottie.inBrowser() -- true if the library is being run in a browser<br />
lottie.resize() -- Resizes all animation instances<br />

Events

- onComplete
- onLoopComplete
- onEnterFrame
- onSegmentStart

you can also use addEventListener with the following events:

- complete
- loopComplete
- enterFrame
- segmentStart
- config_ready (when initial config is done)
- data_ready (when all parts of the animation have been loaded)
- loaded_images (when all image loads have either succeeded or errored)
- DOMLoaded (when elements have been added to the DOM)
- destroy

Other loading options

- if you want to use an existing canvas to draw, you can pass an extra object: 'renderer' with the following configuration:

```
lottie.loadAnimation({
  container: element, // the dom element
  renderer: 'svg',
  loop: true,
  autoplay: true,
  animationData: animationData, // the animation data
  rendererSettings: {
    context: canvasContext, // the canvas context
    scaleMode: 'noScale',
    clearCanvas: false,
    progressiveLoad: false, // Boolean, only svg renderer, loads dom elements when needed. Might speed up initialization for large number of elements.
    hideOnTransparent: true, //Boolean, only svg renderer, hides elements when opacity reaches 0 (defaults to true)
    className: 'some-css-class-name'
  }
});
```

Doing this you will have to handle the canvas clearing after each frame
<br/>
Another way to load animations is adding specific attributes to a dom element.
You have to include a div and set it's class to lottie.
If you do it before page load, it will automatically search for all tags with the class "lottie".
Or you can call lottie.searchAnimations() after page load and it will search all elements with the class "lottie".
<br/>

- add the data.json to a folder relative to the html
- create a div that will contain the animation.
  <br/>
  Required
  <br/>
  . a class called "lottie"
  . a "data-animation-path" attribute with relative path to the data.json
  <br/>

Optional
<br/>
. a "data-anim-loop" attribute
. a "data-name" attribute to specify a name to target play controls specifically
<br/>
Example
<br/>

```
<div style="width:1067px;height:600px" class="lottie" data-animation-path="animation/" data-anim-loop="true" data-name="ninja"></div>
```

<br/>

Preview

You can preview or take an svg snapshot of the animation to use as poster. After you render your animation, you can take a snapshot of any frame in the animation and save it to your disk. I recommend to pass the svg through an svg optimizer like https://jakearchibald.github.io/svgomg/ and play around with their settings.<br/>

Recommendations

Files

If you have any images or AI layers that you haven't converted to shapes (I recommend that you convert them, so they get exported as vectors, right click each layer and do: "Create shapes from Vector Layers"), they will be saved to an images folder relative to the destination json folder.
Beware not to overwrite an exiting folder on that same location.

Performance

This is real time rendering. Although it is pretty optimized, it always helps if you keep your AE project to what is necessary<br/>
More optimizations are on their way, but try not to use huge shapes in AE only to mask a small part of it.<br/>
Too many nodes will also affect performance.

Help

If you have any animations that don't work or want me to export them, don't hesitate to write. <br/>
I'm really interested in seeing what kind of problems the plugin has. <br/>
my email is **hernantorrisi@gmail.com**

AE Feature Support

- The script supports precomps, shapes, solids, images, null objects, texts
- It supports masks and inverted masks. Maybe other modes will come but it has a huge performance hit.
- It supports time remapping
- The script supports shapes, rectangles, ellipses and stars.
- Expressions. Check the wiki page for more info. (https://github.com/airbnb/lottie-web/wiki/Expressions)
- Not supported: image sequences, videos and audio are not supported
- **No negative layer stretching**! No idea why, but stretching a layer messes with all the data.

Development

`npm install` or `bower install` first
`npm start`

Notes

- If you want to modify the parser or the player, there are some gulp commands that can simplify the task
- look at the great animations exported on codepen See examples on codepen. (http://codepen.io/collection/nVYWZR/)
- gzipping the animation jsons and the player have a huge reduction on the filesize. I recommend doing it if you use it for a project.

Issues

- For missing mask in Safari browser, please anim.setLocationHref(locationHref) before animation is generated. It usually caused by usage of base tag in html. (see above for description of setLocationHref)