Community Experience Distilled

Learning Adobe Edge Animate

Create engaging motion and rich interactivity with Adobe Edge Animate

Joseph Labrecque

[PACKT] PUBLISHING

Learning Adobe Edge Animate

Create engaging motion and rich interactivity
with Adobe Edge Animate

Joseph Labrecque

BIRMINGHAM - MUMBAI

Learning Adobe Edge Animate

First published: October 2012

Production Reference: 1171012

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-84969-242-7

www.packtpub.com

Cover Image by Duraid Fatouhi (duraidfatouhi@yahoo.com)

Credits

Author
Joseph Labrecque

Reviewers
Sajith Amma
Deborah Gray
Darrell Heath

Acquisition Editors
Wilson D'souza
Mary Nadar

Lead Technical Editors
Hithesh Uchil
Unnati Shah

Technical Editors
Prasanna Joglekar
Manmeet Singh Vasir

Project Coordinator
Joel Goveya

Proofreader
Aaron Nash

Indexer
Rekha Nair

Graphics
Aditi Gajjar

Production Coordinator
Melwyn D'sa

Cover Work
Melwyn D'sa

About the Author

Joseph Labrecque is primarily employed by the University of Denver as Senior Interactive Software Engineer specializing in the Adobe Flash Platform, where he produces innovative academic toolsets for both traditional desktop environments and emerging mobile spaces. Alongside this principal role, he often serves as adjunct faculty communicating upon a variety of Flash Platform solutions and general web design and development subjects.

In addition to his accomplishments in higher education, Joseph is the Proprietor of Fractured Vision Media, LLC, a digital media production company, technical consultancy, and distribution vehicle for his creative works. He is founder and sole abiding member of the dark ambient recording project *An Early Morning Letter, Displaced,* whose releases have received international award nominations and underground acclaim.

Joseph has contributed to a number of respected community publications as an article writer and video tutorialist. He is also the author of *Flash Development for Android Cookbook*, *Packt Publishing* (2011), *What's New in Adobe AIR 3*, *O'Reilly Media* (2011), *What's New in Flash Player 11*, *O'Reilly Media* (2011), *Adobe Edge Quickstart Guide*, Packt Publishing (2012) and co-author of *Mobile Development with Flash Professional CS5.5 and Flash Builder 4.5: Learn by Video*, *Adobe Press* (2011). He also serves as author on a number of video training publications through video2brain, Adobe Press, and Peachpit Press.

He regularly speaks at user group meetings and industry conferences such as Adobe MAX, FITC, D2W, 360|Flex, and a variety of other educational and technical conferences. In 2010, he received an Adobe Impact Award in recognition of his outstanding contribution to the education community. He has served as an Adobe Education Leader since 2008 and is also an Adobe Community Professional.

Visit him on the Web at `http://josephlabrecque.com/`.

Thanks to my family, friends, and benefactors for your continued support.

Superabundant thanks to Leslie, Paige, and Lily!

About the Reviewers

Sajith Amma has been working as a Technology Consultant for various UK and Indian companies since 2005. He is a hardcore PHP Programmer and a passionate Web Developer working on cutting-edge technologies, especially in the mobile platform. He has received his Masters degree in Business Information Systems from the University of East London. He uses his technical blog (`sajithmr.me`) to share his findings and interesting articles related to programming.

Sajith is the co-founder of MobMe Wireless Solutions, an Indian company, which is listed in the top 100 as one of the best innovation companies in India. He has worked as a Technology Consultant for companies such as Vodafone UK, Orange UK, and T-Mobile UK. He is currently working as a Solution Architect for a UK-based company, Muzicall Ltd.

I would like to thank Muzicall Ltd. for giving me the opportunity to work with HTML5 and Adobe Edge. I would also like to thank Joseph Labrecque for giving me the opportunity to review this book.

Deborah Gray has been in the design business for 20 years, having started in the 1980s in the newspaper business. For the last seven years, she has run a successful Sonoma County design agency, Deborah Gray Design (`http://deborahgraydesign.com/`). She has designed and developed hundreds of websites and print campaigns for clients in the San Francisco Bay Area and worldwide.

Deborah specializes in frontend development and WordPress. Her early experience as a Print Designer created a passion for design and typography that is clean, simple, and withstands the test of time.

Darrell Heath has an education in Applied Information Technology. He is an experienced Visual Artist, and for the past nine years has been a freelance Web Developer through various local IT consultants. Since 2005, he has been a tutorialist for the National Association of Photoshop Professionals (NAPP), Layers Magazine, and Planet Photoshop.

Darrell has also provided a technical review for *Adobe Edge: The Missing Manual*, *Chris Grover*, *O'Reilly Media*. He is an active member in a number of Adobe community forums, and is a co-moderator for NAPP's private membership board.

Visit him on the Web at `http://www.heathrowe.com`.

www.PacktPub.com

Support files, eBooks, discount offers and more

You might want to visit `www.PacktPub.com` for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at `www.PacktPub.com` and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at `service@packtpub.com` for more details.

At `www.PacktPub.com`, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

`http://PacktLib.PacktPub.com`

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

Why Subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print and bookmark content
- On demand and accessible via web browser

Free Access for Packt account holders

If you have an account with Packt at `www.PacktPub.com`, you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

Table of Contents

Preface

Adobe Edge Animate is an all new tool from Adobe which seeks to enable the authoring of motion and interactive experiences through HTML5, CSS3, and JavaScript in a manner consistent with Creative Suite applications. Edge Animate is able to create such experiences at this time, due to advancements in browser technology and the need for a consistent, cross-platform solution which is able to function across desktop and mobile operating systems.

Why do we need Adobe Edge Animate?

Some may ask for an explanation: why do we need Edge Animate when we have tools such as Flash Professional which also create animation and interactive content for the Web? There are a number of reasons for this, which we will now attempt to illustrate.

Adobe Flash Player restrictions

Traditionally, those of us designing animated or highly interactive content for the Web have been able to rely on Adobe Flash Player to display this content without issue across Windows, Mac, and Linux. In fact, Adobe and many independent entities still reports that Flash Player is installed on 96 to 99 percent of desktop machines. There are problems though, as we must now account for mobile operating systems which place restrictions upon the Flash Player, or even outright ban it entirely. The most problematic of these platforms is Apple iOS.

It is worth noting that the Google Android, Windows 8 (desktop and mobile), BlackBerry 10, and BlackBerry Tablet OS mobile operating systems all have robust Flash Player 11 support. However, Adobe has halted any further development for the mobile Flash Player after version 11.1 as of the publication of this book. Others do have the option of licensing Flash Player and integrating it into their systems, as RIM continues to do for their QNX-based systems such as BB10 and PlayBook.

Since Flash content is restricted from running within the mobile iOS Safari browser, designers have been searching for alternative ways of delivering experiences to these devices.

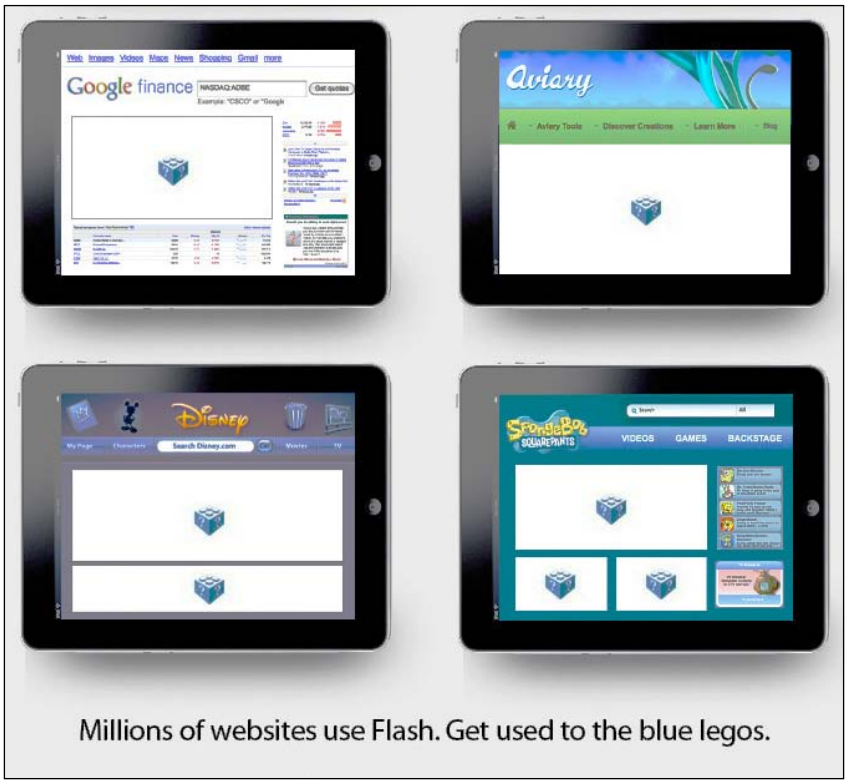


Image courtesy of Lee Brimelow

Though Apple iOS has banned Flash Player in the browser, Flash content can be distributed through the Apple App Store in the form of compiled applications which target this platform. Similarly, other mobile operating systems such as Google Android, Windows 8, RIM BlackBerry 10, and Tablet OS also include full support for Flash-based projects through Adobe AIR.

HTML technology maturity

For much of its history, HTML has provided a way for web designers to creatively markup content for rendering within a browser. With the draft HTML5 specification currently under development, this role has been expanded in some ways which attempt to move beyond simple textual markup and into the rich media space.

Three tags often cited as examples of this include the following:

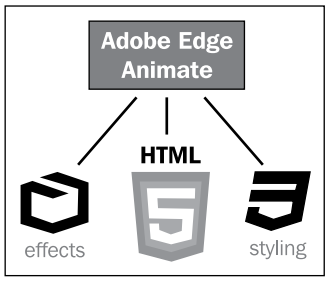
- `<video>`: For simple video playback in HTML
- `<audio>`: For simple audio playback in HTML
- `<canvas>`: For programmatically rendering bitmap visuals in HTML through JavaScript APIs

Along with the core HTML specification in development are related specifications such as CSS3 and a variety of additional specifications meant to extend the core technologies of the Web. We have also seen great increases in the speed of JavaScript engines over the past couple of years, enabling greater use of the basic scripting language for the Web. Add a number of frameworks (such as the popular jQuery [`http://jquery.com/`] framework) to this environment and we have quite the revolution in core web technologies!

Shifting roles

Adobe Flash Player has always served as an extension to core web technologies such as HTML, CSS, and JavaScript—enabling experiences within the browser that were just not possible using these technologies on their own. With the recent expansions we've already detailed, some of the capabilities of Flash Player have now been made possible in other technologies.

Primary among these is the ability to create motion and animated objects employing core web technologies using tools such as Adobe Edge Animate.



Although it is now possible to create website intros, rich ads, and other motion content using a tool such as Edge, designers should be careful not to replicate the nuisances of the past. The Web doesn't need more "Skip Intro" landing pages.

While HTML and related technologies have adopted some of what designers used the Flash Player for years ago, it is important to consider that the Flash Platform has also grown quite a bit over the last few years. The role of Flash has shifted from enabling motion and rich interactivity on the Web to providing rich video experiences, enterprise applications, advanced web modules, and console-quality games with Flash Player 11. The Flash Platform itself has expanded from the browser and onto desktop and mobile operating systems using Adobe AIR: smartphones, tablets, and even television units and automobile dashboards have benefitted from this shift in technology.

Interested in what lies in the future for the Flash runtimes? Adobe has published a white paper that outlines the roadmap for the next two years and sets the foundation for technological advances for the next decade.

Read the Adobe roadmap for the Flash runtimes at `http://www.adobe.com/go/flashplayer_roadmap/`.

While web browser technology (as seen in Chrome, Firefox, Safari, Opera, and Internet Explorer 10) is evolving to provide web professionals with more choices in what technology is used to create content for the browser, Flash Player still holds a strong place in this environment and the two sets of technologies will work together to expand the Web, just as they have done for the past 15+ years.

Mobile deployment

Perhaps the single largest driving factor in the rapid evolution of core web technologies over the past two years has been the prevalence of advanced browsers on mobile devices. Due to the fact that mobile computing is still so new, users are not coming into this environment with old technology. This enables browser makers and device manufacturers to bundle web browsers with these systems that take full advantage of HTML5, CSS3, and advanced JavaScript rendering engines.

Most mobile browsers are based upon the open source WebKit [`http://www.webkit.org/`] rendering engine. Couple this with the fact that prominent desktop browsers such as Google Chrome and Apple Safari also use WebKit for their rendering engines and we have a widely adopted baseline to lean upon when developing experiences using newer technologies.

Note that WebKit is the rendering engine for the actual Edge Animate application environment, offering a true WYSIWYG experience during composition authoring. WebKit is also used in integrated runtimes such as Adobe AIR, furthering the reach of this popular HTML rendering codebase.

What can Adobe Edge Animate be used for?

Generally, Edge Animate can be used to create many of the same types of animations and interactions that we would have expected Flash Player to handle on the Web in the mid to late 1990s.

This includes the movement of visual objects across the stage and basic mouse interactions.

Web animation

Edge Animate uses an all-new timeline for producing motion which borrows a lot from other applications such as Adobe After Effects. Through the use of keyframes along the timeline, designers have very fine-grained control over many object properties and can easily enable easing algorithms, which provide an additional flair to animated content. Edge Animate offers a unique approach to creating and accessing page level elements, their properties, and animating them on the timeline.

Interactive content

Edge Animate is not just about making things move. The Edge Animate Runtime also includes a robust API to enable interactivity through mouse, touch, and time-based actions. These interactive commands can be applied to individual, visible objects upon the stage, or used along the timeline in the form of triggers. Interactivity can modify aspects of the stage timeline, modify the properties of other objects within an Edge Animate project, or even invoke calls to content outside of the project.

Is Adobe Edge Animate for me?

While this book will often make reference to other applications such as Flash Professional or After Effects, you will not need to have prior experiences with these applications to get the most out of Edge Animate. So long as you understand the basics of HTML, CSS, and JavaScript—and have a desire to learn a worthwhile motion and interaction tool which targets these standards, then you should be all set!

Let's get started!

We have now taken a look at some ways in which the Web landscape is changing, specifically when talking about the roles of the primary technologies used to create motion and interactive design in the browser. The content produced by Edge Animate would only have been possible using Flash Player in years past. HTML, CSS, and JavaScript have advanced to the point that this sort of content can now be produced using core web technologies. At the same time, Adobe Flash Player and the wider Flash Platform have expanded beyond these roles. We've also had a high level view of Adobe Edge Animate and some of the content types which are enabled through use of this authoring tool.

Throughout this book, we'll be taking a complete look at the Animate application interface, demonstrate how to create and import project assets, and use those assets in the creation of compositions which feature advanced motion and interactivity using web standards.

What this book covers

Chapter 1, *Introducing Adobe Edge Animate*, provides a comprehensive overview of the entire Edge Animate application interface. This overview includes a look at the panels, tools, menus, and other application elements we will need to familiarize ourselves with when using Animate.

Chapter 2, *Drawing and Adjusting Composition Elements*, delves into the drawing tools contained within the Animate application to allow the creation of simple rectangular elements and assorted other objects.

Chapter 3, *Selecting and Transforming Elements*, provides a look at the Selection and Transform tools, their uses, and unique attributes. We also have a good overview of the Properties panel and its use across element types.

Chapter 4, *Using Text and Web Fonts*, demonstrates the creation of text elements within an Edge Animate project and provides detailed examples of using web fonts for even more expressive textual content.

Chapter 5, *Importing External Assets*, will show how to import and use an abundance of external assets within our Edge Animate compositions.

Chapter 6, *Creating Motion Through the Timeline*, demonstrates how simple it is to build a composition which involves a number of animated elements and presents a unique toolset for dealing with motion on the Web.

Chapter 7, *Interactivity with Actions, Triggers, and Labels*, will expand upon the motion-based topics of the previous chapter through the addition of interactive elements within an Edge Animate project. We'll also have a look at some of the more complex uses of the Adobe Edge Animate Runtime APIs.

Chapter 8, *Making Use of Symbols, Nested Elements, and Grouping*, provides a deep analysis of the powerful Symbol architecture within Edge Animate and demonstrates a variety of uses for Symbol instances. We also take a look at nested elements and provide some example projects.

Chapter 9, *Advanced Animation Techniques*, delves into the world of clipping, sprite sheets, and image sequences in extending much of the core motion functionality through the use of external assets and additional techniques.

Chapter 10, *Publishing Edge Animate Compositions*, examines the many options available to us when preparing and publishing an Edge Animate composition for the Web or other supported targets.

Chapter 11, *Further Explorations with Adobe Edge Animate*, contains many techniques which are either too general in nature, or are too expansive to fit within any of the other chapters.

What you need for this book

To use this book effectively, you will need to acquire Adobe Edge Animate from Adobe. Edge Animate is available with a subscription to the Creative Cloud service.

Adobe Edge Animate can be acquired from `http://html.adobe.com/edge/animate`.

Who this book is for

This book is for anyone who wants to get started using Adobe Edge Animate to create engaging, interactive content for the Web. It isn't necessary that you have any prior knowledge of website or motion design.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text are shown as follows: "These elements will default to a `<div>` HTML element, but can be changed to employ the following HTML elements instead."

A block of code is set as follows:

```
(function(symbolName) {
Symbol.bindElementAction(compId, symbolName, "${_fvm001}",
"mouseover", function(sym, e) {
// Change an Element's contents.
//  (sym.$("name") resolves an Edge element name to a DOM
//  element that can be used with jQuery)
sym.$("Info").html("August (2000)");
});
```

When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:

```
Symbol.bindElementAction(compId, symbolName, "${_Rectangle}",
"mousedown", function(sym, e) {
sym.playReverse();
// insert code for mousedown here
});
//Edge binding end
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "The first option is to simply click on **Create New** on the welcome screen".

Warnings or important notes appear in a box like this.

Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to `feedback@packtpub.com`, and mention the book title through the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at `http://www.packtpub.com`. If you purchased this book elsewhere, you can visit `http://www.packtpub.com/support` and register to have the files e-mailed directly to you.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/support`, selecting your book, clicking on the errata submission form link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded to our website, or added to any list of existing errata, under the Errata section of that title.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at `questions@packtpub.com` if you are having a problem with any aspect of the book, and we will do our best to address it.

1
Introducing Adobe Edge Animate

This chapter will delve into Adobe Edge Animate, concentrating on the history of the Edge Animate project, looking at the technologies behind Edge Animate, comparing Edge Animate with Flash Professional (as the two applications share many similarities), providing a full overview of many Edge Animate application interface features, and finally taking a brief look at the Edge Animate welcome screen and how to create a new project.

Adobe Edge Animate boasts a modern, designer-friendly user interface that should be somewhat familiar to long-time users of the Adobe Creative Suite applications. We will run through each aspect of the interface including the following options:

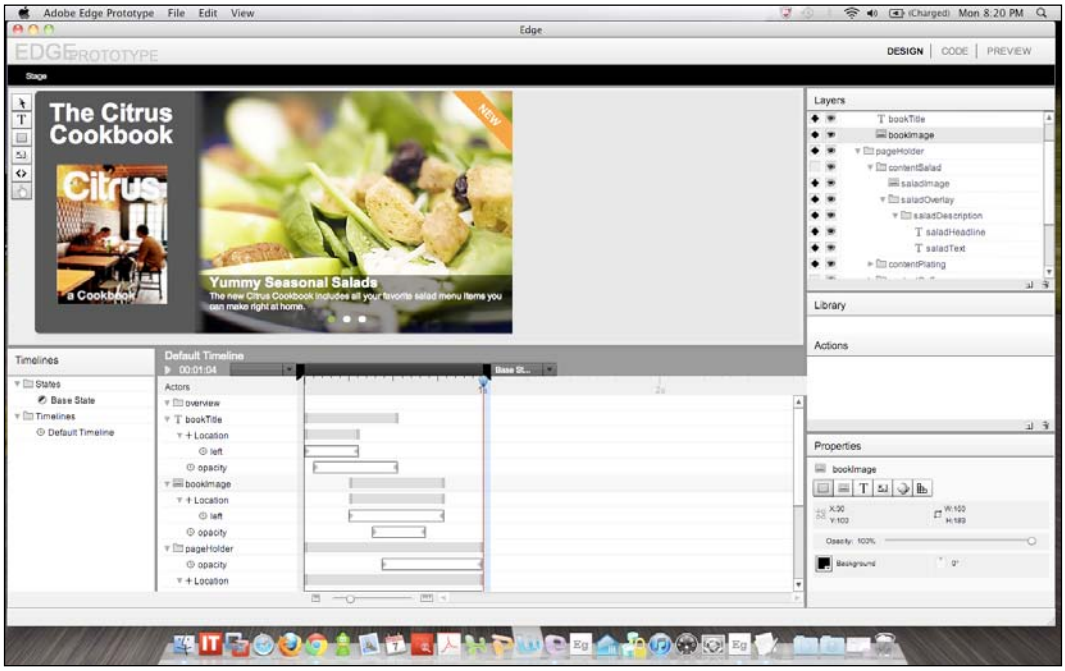
- Interface features
- Application menus
- The toolbar
- Stage
- Timeline
- Edge Animate panels

After processing the information presented here, we should have a clear understanding of the interface as a whole and also the usefulness of its individual aspects.

The history of Adobe Edge Animate

During the Adobe MAX 2010 conference in Los Angeles, California, Adobe engineers got on stage in front of over 5000 attendees to present a software prototype built in Adobe AIR. This software allowed a user to adjust the properties of imported assets in a way very similar to the workflow of Flash Professional, but instead of outputting to SWF to target the Flash Player, the Adobe Edge Prototype actually output content to HTML, CSS, and JavaScript for playback in a web browser, without the need for any additional plugins.

Adobe AIR is a solution for creating desktop and mobile applications built on Flash Platform technology. Many Adobe products are built using AIR, including the new touch applications for use on tablets and Adobe Muse. Visit `http://www.adobe.com/products/air.html`.



While the Edge Prototype certainly appeared very different from what we know today as **Adobe Edge Animate**, MAX attendees went wild over the prospects of such a tool. This was the first glimpse of what would eventually become the product we know today as Adobe Edge Animate. Since that time, Adobe has released periodic

updates to the Adobe Edge Preview releases on Adobe Labs, with the intent to gather user feedback early and often in order to make the product conform to user expectations and become a useful addition to the Creative Suite.

With Adobe's long history of motion and interactivity in products such as Director, After Effects, and Flash Professional, Edge Animate has an excellent lineage behind it, and while creating content like this which targets HTML is quite new, the tools and techniques for authoring this material comes to us along a well-tread path.

The inner workings of Edge Animate

Adobe Edge Animate relies heavily on three related technologies: HTML5, CSS3, and JavaScript. The default doctype for Edge Animate created projects is HTML5; all the 2D transforms, translate(), rotate(),scale(), and skew() for example, are rendered as CSS3 (for modern browsers). There are also specific JavaScript libraries that play an essential role in making all of this work together. These include jQuery and the Adobe Edge Animate Runtime.

In order for Edge Animate content to work successfully, all of these components must be in their correct place and there are certain files which should not be edited once generated by the application. The Edge Animate application itself also requires a `.an` file type to be present in order to author and edit a project.

Any `.html` file can also be opened up within Edge Animate and be worked upon. A `.an` file and associated imports will be created upon save and publication.

HTML, CSS, and JavaScript

Edge Animate primarily targets HTML for display, supported by both CSS and JavaScript. Why? Well, the fact of the matter is these technologies have finally become capable of handling rich motion and interactive content and as these are the core technologies of the Web, it makes sense to use them whenever we can.

Let's take a quick look at these three specifications in light of their primary function on the Web and relation to one another.

HTML

Hyper Text Markup Language (**HTML**) is the core of the Web. With the HTML5 specification (still in draft), we not only have an organic evolution of the language through additional semantic tags, but also a new set of APIs that can allow elements within the documents to be greatly influenced through JavaScript.

CSS

Cascading Style Sheets (**CSS**) determine to a great extent how a website is visually structured and designed. With the CSS3 specification (still in draft), designers can still use these specifications in all modern browsers to influence the way certain elements behave.

JavaScript

The **JavaScript** language is a superset of **ECMAScript** (**ECMA-262**) **Edition 3**, formalized by ECMA International, a worldwide standards body. The latest version of the language is JavaScript 1.8.5 but the real improvements in recent years have come from the browser manufacturers themselves, as they seek to improve JavaScript execution through the development of faster JavaScript engines.

So when we look into an HTML document produced by Edge Animate, we see the following code:

```
<div id="Stage" class="EDGE-1632861112">
 </div>
```

This is the stage symbol element within which all other elements are injected upon runtime, through the use of JavaScript libraries.

This may be the only HTML element you will ever see produced by Edge Animate. Everything else is handled via JSON objects and specialized JavaScript includes(features). There is an option to render other elements as static HTML, but that is optional.

How jQuery is used in Edge Animate

It is no exaggeration to state that jQuery is the most popular JavaScript framework in use today. Many similar JavaScript frameworks arose in 2007 with the emergence of **Asynchronous JavaScript and XML** (**AJAX**) and more dynamic HTML data transfer methods. At one point, there were over 250 of these frameworks, but with the passing of time, only a handful remain in active development.

As stated on the project website,

> *jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development.*

In a nutshell, jQuery aims to make using JavaScript more accessible to non-programmers or those who are not familiar with the language, make it more consistent across browsers, and more powerful in its simplicity. Documentation for jQuery can be found online at `http://docs.jquery.com/`.

Adobe Edge Animate leverages jQuery and builds upon it within the Adobe Edge Animate Runtime and also makes use of the jQuery easing library when dealing with motion. When opening any HTML document generated by Edge Animate, we can see these includes in the head of our published document through the library preloader:

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=Edge"/>
  <title>AnimateProject</title>
<!--Adobe Edge Runtime-->
    <script type="text/javascript" charset="utf-8"
src="AnimateProject_edgePreload.js"></script>
    <style>
        .edgeLoad-EDGE-1159339764 { visibility:hidden; }
    </style>
<!--Adobe Edge Runtime End-->

</head>
<body style="margin:0;padding:0;">
    <div id="Stage" class="EDGE-1159339764">
    </div>
</body>
</html>
```

Other Adobe products, such as Adobe Dreamweaver, also make heavy use of jQuery. In fact, Adobe actively contributes back to the jQuery and jQuery Mobile libraries.

JSON

JavaScript Object Notation (**JSON**) is a data-interchange format used to exchange data from one system to another. Over the past few years, it has been adopted by a variety of languages and systems for both data transmission and storage. In some ways, it is very similar to XML. Unlike XML, JSON is not a markup language but rather stores data in objects and structures represented in name/value pairs.

Edge Animate uses JSON to store element definitions and attributes with a project. For example, the following JSON fragment represents a rectangle on the Stage:

```
content: {
  dom: [
    {
    id:'Rectangle',
    type:'rect',
    rect:['25px','40px','211px','147px','auto','auto'],
    fill:["rgba(192,192,192,1)"],
    stroke:[0,"rgba(0,0,0,1)","none"]
    }],
  symbolInstances: [
  ]
  }
```

To learn more about JSON, visit `http://www.json.org/`.

The Adobe Edge Animate Runtime

The set of JavaScript libraries used in an Edge Animate project is collectively referred to as the **Adobe Edge Animate Runtime**. Normally, when we think of a runtime, we are talking about a piece of software like Adobe Flash Player, the **Adobe Integrated Runtime** (**AIR**), or the **Java Runtime Environment**. These are all self-contained pieces of software which enable the playback of applications and other content that targets these specific runtimes. The Adobe Edge Animate Runtime is very different in that it is a set of files that supports the content defined through the Adobe Edge Animate application, but even these libraries rely upon another piece of software for them to run properly: the web browser.

If you look within an HTML file produced by Edge Animate, you will see a JavaScript **include** that handles the runtime libraries included within the head of that document, as shown in the following code:

```
<!--Adobe Edge Runtime-->
    <script type="text/javascript" charset="utf-8"
src="AnimateProject_edgePreload.js"></script>
    <style>
        .edgeLoad-EDGE-1159339764 { visibility:hidden; }
    </style>
<!--Adobe Edge Runtime End-->
```