



Aurelia

tutorialspoint
SIMPLY EASY LEARNING

www.tutorialspoint.com



<https://www.facebook.com/tutorialspointindia>



<https://twitter.com/tutorialspoint>

About the Tutorial

Aurelia is a modern, open source UI framework for web and mobile app development. It allows you to write clean, modular JavaScript. The framework follows simple conventions and is focused on web standards.

Audience

This tutorial is designed for developers who didn't have a chance to work with this framework before. The tutorial contains simple, easily understandable examples. These examples can be used as a reference for future projects.

Prerequisites

Since Aurelia is promoting pure **JavaScript**, you will need to know the language prior to learning the framework. We will use EcmaScript2016 syntax. You will also need to have previous experience working with **HTML**.

Copyright & Disclaimer

© Copyright 2017 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at contact@tutorialspoint.com

Table of Contents

About the Tutorial.....	i
Audience	i
Prerequisites	i
Copyright & Disclaimer.....	i
Table of Contents	ii
1. AURELIA – OVERVIEW	1
2. AURELIA – ENVIRONMENT SETUP	3
3. AURELIA – FIRST APPLICATION	7
Folder Structure	7
Source Files	7
4. AURELIA – COMPONENTS.....	10
Simple Component.....	10
Component Functions	11
5. AURELIA – COMPONENT LIFECYCLE.....	13
6. AURELIA – CUSTOM ELEMENTS.....	15
7. AURELIA – DEPENDENCY INJECTIONS.....	18
8. AURELIA – CONFIGURATION.....	20
9. AURELIA – PLUGINS.....	22
Standard Plugins	22
Official Plugins	22
Installing Plugins	23

10. AURELIA – DATA BINDING	25
Simple Binding	25
Two-Way Binding	26
11. AURELIA – BINDING BEHAVIOR.....	29
Throttle	29
Debounce.....	30
oneTime	30
12. AURELIA – CONVERTERS.....	32
Convert Date	32
Convert Currency	34
13. AURELIA – EVENTS.....	36
Event Delegate	36
Event Trigger	37
14. AURELIA – EVENT AGGREGATOR	38
15. AURELIA – FORMS	41
Text Input.....	41
Checkbox.....	42
Radio Buttons	43
16. AURELIA – HTTP.....	46
17. AURELIA – REFS	51
18. AURELIA – ROUTING.....	54
19. AURELIA – HISTORY	58
20. AURELIA – ANIMATIONS.....	62

21. AURELIA – DIALOG	66
22. AURELIA – LOCALIZATION.....	71
23. AURELIA – TOOLS	77
24. AURELIA – BUNDLING.....	80
25. AURELIA – DEBUGGING	84
26. AURELIA – COMMUNITY.....	86
27. AURELIA – BEST PRACTICES	88
Starting a New Project	88
Folder Structure	89
Web Standards	89
EcmaScript 6	89

1. AURELIA—OVERVIEW

The best definition of the framework can be found in Aurelia official docs –

Well, it's actually simple. Aurelia is just JavaScript. However, it's not yesterday's JavaScript, but the JavaScript of tomorrow. By using modern tooling we've been able to write Aurelia from the ground up in ECMAScript 2016. This means we have native modules, classes, decorators and more at our disposal...and you have them too.

Not only is Aurelia written in modern and future JavaScript, but it also takes a modern approach to architecture. In the past, frameworks have been monolithic beasts. Not Aurelia though. It's built as a series of collaborating libraries. Taken together, they form a powerful and robust framework for building Single Page Apps (SPAs). However, Aurelia's libraries can often be used individually, in traditional web sites or even on the server-side through technologies such as NodeJS.

Aurelia — Features

- **Components** – Components are building blocks of Aurelia framework. It is composed of HTML view and JavaScript view-model pairs.
- **Web Standards** – This is one of the cleanest modern frameworks, completely focused on web standards without unnecessary abstractions.
- **Extensible** – The framework offers an easy way to integrate with the other needed tools.
- **Commercial Support** – Aurelia offers commercial and enterprise support. It is an official product of Durandal Inc.
- **License** – Aurelia is open sourced and licensed under MIT license.

Aurelia — Advantages

- Aurelia is very clean. If you follow the frameworks conventions, you can focus on your app without the framework getting in your way.
- It is also easily extensible. You can add or remove any tools that the framework offers and you can also add any other tools that aren't part of the framework.
- Aurelia is very easy to work with. It is directed towards developers' experience. It saves you lots of time.
- The framework itself is directed towards web standards so you will always stay up to date with modern concepts.
- Aurelia doesn't have the largest community out there, but it is very agile, knowledgeable and willing to help within short notice.

Limitations

There are no major limitations. The Framework is powerful and easy to work with.

2. AURELIA—ENVIRONMENT SETUP

In this chapter, you will learn how to get started with Aurelia framework. Before you do that, you will need **NodeJS** installed on your system.

Sr. No.	Software	Description
1	NodeJS and NPM	NodeJS is the platform needed for Aurelia development. Checkout our NodeJS Environment Setup .

Step 1 - Download Aurelia Package

Before we download Aurelia package, let's create a folder on desktop where our app will be placed.

```
C:\Users\username\Desktop>mkdir aureliaApp
```

Now we can download the package from official [Aurelia website](#).

Aurelia supports **ES2016** and **TypeScript**. We will use **ES2016**. Extract the downloaded files inside the **aureliaApp** folder that we created above.

Step 2 - Install the Web Server

First, we need to install the web server from **command prompt** window.

```
C:\Users\username\Desktop\AureliaApp>npm install http-server -g
```

Step 3 - Start the Web Server

To start the web server, we need to run the following code in **command prompt**.

```
C:\Users\username\Desktop\AureliaApp>http-server -o -c-1
```

We can see our first Aurelia app in the browser.



3. AURELIA—FIRST APPLICATION

In this chapter, we will explain Aurelia starting app created in our last chapter. We will also guide you through the folder structure, so you can grasp the core concepts behind Aurelia framework.

Folder Structure

- **package.json** represents documentation about **npm** packages installed. It also shows the version of those packages and provides an easy way to add, delete, change version or automatically install all packages when the app needs to be shared between developers.
- **index.html** is the default page of the app like in most of the HTML based apps. It is a place where scripts and stylesheets are loaded.
- **config.js** is Aurelia loader configuration file. You will not spend much time working with this file.
- **jspm_packages** is the directory for the **SystemJS** loaded modules.
- **styles** is the default styling directory. You can always change the place where you keep your styling files.
- **src** folder is a place where you will spend most of your development time. It keeps **HTML** and **js** files.

Source Files

As we already stated, the **src** directory is the place where your app logic will be held. If you look at the default app you can see that **app.js** and **app.html** are very simple.

Aurelia allows us to use JavaScript core language for class definitions. Following default example shows EC6 class.

app.js

```
export class App {  
  message = 'Welcome to Aurelia!';  
}
```

The **message** property is bound to the HTML template using **\${message}** syntax. This syntax represents one-way binding converted into string and showed inside the template view.

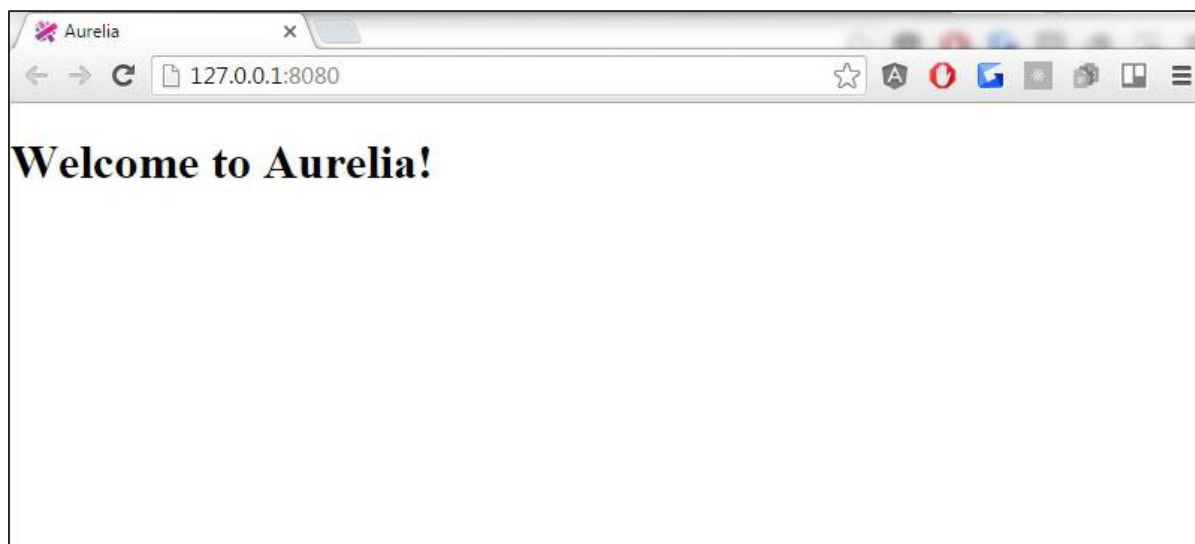
app.html

```
<template>
  <h1>${message}</h1>
</template>
```

As we already discussed in the last chapter, we can start the server by running the following command in the **command prompt** window.

```
C:\Users\username\Desktop\AureliaApp>http-server -o -c-1
```

Application will be rendered on the screen.



4. AURELIA—COMPONENTS

Components are the main building blocks of Aurelia framework. In this chapter, you will learn how to create simple components.

Simple Component

As already discussed in the previous chapter, each component contains **view-model** which is written in **JavaScript**, and **view** written in **HTML**. You can see the following **view-model** definition. It is an **ES6** example but you can also use **TypeScript**.

app.js

```
export class MyComponent {  
  header = "This is Header";  
  content = "This is content";  
}
```

We can bind our values to the view as shown in the following example. **\${header}** syntax will bind the defined **header** value from **MyComponent**. The same concept is applied for **content**.

app.html

```
<template>  
  <h1>${header}</h1>  
  <p>${content}</p>  
</template>
```

The above code will produce the following output.

End of ebook preview
If you liked what you saw...
Buy it from our store @ **<https://store.tutorialspoint.com>**