Федеральное государственное образовательное бюджетное учреждение высшего образования

«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ

РОССИЙСКОЙ ФЕДЕРАЦИИ»

**Колледж информатики и программирования**

Доклад

на тему библиотека Aurelia

Выполнил:

студент группы 4ИСИП-619

Мандриков М.C.

Проверил:

преподаватель Сибирев И. В.

Москва 2023

Aurelia, выпущенная в 2016 году, представляет собой простой, ненавязчивый и мощный JS-фреймворк с открытым исходным кодом для создания адаптивных мобильных, настольных и браузерных приложений.

Особенности / преимущества:

Aurelia предназначена для выполнения высокопроизводительных и эффективных пакетных обновлений DOM.

Обеспечивает стабильную и масштабируемую производительность даже при сложном пользовательском интерфейсе.

Обширная экосистема с государственным управлением, проверкой и интернационализацией.

Включает реактивную привязку и автоматически синхронизирует ваше состояние с высокой производительностью.

Более простое модульное тестирование.

Беспрецедентная расширяемость для создания пользовательских элементов, добавления атрибутов, управления генерацией шаблонов и т. Д.

Использует расширенную маршрутизацию на стороне клиента, композицию пользовательского интерфейса и прогрессивные улучшения.

Он нацелен на согласование веб-спецификаций с соглашением, а не с конфигурацией, и требует меньшего вмешательства в фреймворк

Почему Aurelia лучше?

Легко учиться

Основанный на стандартах, ненавязчивый стиль Aurelia делает его единственной платформой, позволяющей создавать компоненты с использованием ванильного JavaScript или TypeScript. Если вы владеете современными JS и HTML, вам будет немного больше, чтобы научиться создавать даже самые сложные приложения

Высокая производительность

В основе Aurelia лежит высокопроизводительная реактивная система, способная пакетировать обновления DOM таким образом, что другие фреймворки и их виртуальные домены оказываются в пыли. Испытайте стабильную, масштабируемую производительность, независимо от того, насколько сложным является ваш пользовательский интерфейс

Обширная экосистема

Управление состоянием, интернационализация и проверка - все официальные плагины от основной команды. CLI, плагин VS Code и отладчик Chrome - дополнительные инструменты для улучшения разработки. Aurelia - это не просто фреймворк, а мощная платформа и экосистема, призванные обеспечить ваш быстрый успех.

Реактивное связывание

Aurelia обеспечивает мощную реактивную привязку к любому объекту. Используя адаптивные методы, Aurelia выбирает наиболее эффективный способ наблюдения за каждым свойством в вашей модели и автоматически синхронизирует ваш пользовательский интерфейс и ваше состояние с лучшей в своем классе производительностью.

Простое тестирование

Сочетая современные модули JS с ненавязчивым подходом, Aurelia делает модульное тестирование таким же простым, как тестирование ванильного JS. Нужно написать интеграционные тесты? Мощный контейнер для внедрения зависимостей и библиотека тестирования делают это быстрым и простым. Вы получаете выгоду от удобных в обслуживании и долговечных приложений

Непревзойденная расширяемость

Ничто в отрасли не может сравниться с расширяемостью Aurelia. Вы можете создавать пользовательские элементы, добавлять пользовательские атрибуты к существующим элементам, управлять генерацией шаблонов, настраивать синтаксис шаблонов, создавать новые типы реактивной привязки, расширять DI и практически все, что только можно придумать.

Простое кодирование с использованием соглашений

Соглашения позволяют вам сосредоточиться на вашем коде и вашей бизнес-логике. Aurelia выделяется как единственный фреймворк на сегодняшний день, который помещает ваш код в центр вашего приложения, а не в фреймворк. Не нравятся условности? Подключите свои собственные или вообще откажитесь от них. Ты все контролируешь.

Маршрутизация, композиция и прогрессивное улучшение

Используйте усовершенствованный клиентский маршрутизатор Aurelia с подключаемым конвейером и дочерними маршрутизаторами. Вам не нужен маршрутизатор, но вам нужна динамическая композиция пользовательского интерфейса, управляемая данными? Мы тоже так делаем. У вас есть устаревшее приложение? Прогрессивное усовершенствование Aurelia и сборка скриптовых тегов также упрощают постепенное внедрение.

Установка Aurelia

Aurelia на данный момент находится в статусе Beta. Пока ее можно установить только через jspm. Для установки Aurelia выполним в PowerShell команду:

jspm install aurelia-framework

И следом еще одну команду.

jspm install aurelia-bootstrapper

Если сообщения на ваших мониторах соответствует двум последним скриншотов с моего монитора, то можно приступать к программированию на Aurelia.

Приложение на Aurelia

Первым делом создадим два файла app.html и app.js. Aurelia при старте ищет именно эти два файла (как view и viewModel) связывает их и открывает. В настройках, естественно, можно изменить это поведение.

Итак, в папке wwwroot я создал *app.html* и перенес некоторое количество html-кода из файла default.html:

<template>

<div>

<h1>Welcome to Aurelia</h1>

<p>This is a simple paragraph of plain text.</p>

</div>

</template>

Далее файл *app.js*. Пусть пока он выглядит "пустоватым":

export class App

{

}

Теперь вся магия будет реализована в файле *default.html*. После добавления некоторых скриптов и html-атрибутов он стал выглядить так:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title>Aurelia Demo</title>

</head>

<body aurelia-app>

<script src="jspm\_packages/system.js"></script>

<script src="config.js"></script>

<script>

System.import('aurelia-bootstrapper')

</script>

</body>

</html>

Запустим проект...

Ничего не изменилось с момента последнего запуска, но мы с вами знаем, что теперь этот контент "рисует" уже Aurelia. Давайте добавим немного интерактивности в наш тестовый пример. Вот так я изменил app.js:

export class App

{

constructor() {

this.heading = 'Welcome to Aurelia! Again!';

this.FirstName = 'John';

this.LastName = 'Doe';

this.buttonName = 'Say hello';

}

get fullName() {

return `${this.FirstName} ${this.LastName}`;

}

submit() {

alert(`Hi, ${this.fullName}`);

}

}

А вот так изенился app.html:

<template>

<div>

<h1>Welcome to Aurelia</h1>

<p>This is a simple paragraph of plain text.</p>

<h1>${heading}</h1>

<form role="form" submit.delegate="submit()">

<input value.bind="FirstName" placeholder="firstname"/><br/>

<input value.bind="LastName" placeholder="lastname"/><br/>

<p>${fullName}</p>

<button type="submit">${buttonName}</button>

</form>

</div>

</template>