



链滴

关于微前端实现原理与 ngx-planet(四) 服务端渲染

作者: [someone61489](#)

原文链接: <https://ld246.com/article/1614224127325>

来源网站: 链滴

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

1.为什么要服务端渲染

因为公司后端服务在k8s上，是分布式的微服务，之前端全部打包部署在了物理机器(虚拟机)nginx上，果通过helm做应用商店的话，nginx前端这部分无法处理，包括灰度部署，CI等，全部都只能做到接级别的处理，并不能连带静态资源文件一起处理，所以基于分布式的前端整改迫在眉睫。

偶然发现了ngx-planet，整篇文章基于前几篇文章，可以看之前的几篇文章。

2.如何基于ngx-planet进行服务端渲染

2.0 有路由前缀的情况下服务端渲染ngx-planet如何改进

```
start(): void {
  this.subscription = this.router.events
    .pipe(
      filter((event) => {
        console.log(event)
        return event instanceof NavigationEnd;
      }),
      map((event: NavigationEnd) => {
        console.log(event)
        return event.urlAfterRedirects || event.url;
      }),
      startWith(location.pathname.split("/star-universe")[1]),
      distinctUntilChanged()
    )
    .subscribe((url: string) => {
      console.log(url)
      this.starApplicationLoader.reroute({ url: url });
    });
}
```

在 `start` 函数中监听路由阶段，其中的 `startWith` 过滤路由要把 `location.pathname` 修改好，要将前去除掉，至于如何动态去除不写死，仁者见仁智者见智。

2.1 首先，准备打包好的静态文件

将 `build` 命令修改，注意 `--deploy-url` 的意思是，打包完成后，静态资源路径是什么

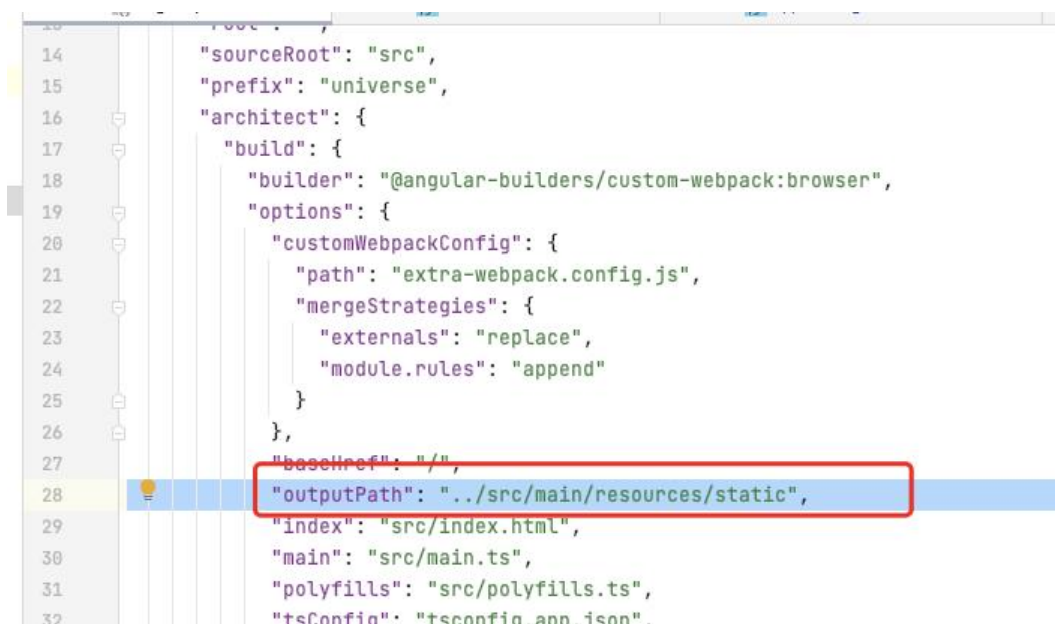
```
"build": "ng build --prod --deploy-url=/static/star-universe/ --base-href=/star-universe",
```



打包后的index.html如图，静态资源路径变成了 `/static/star-universe` 前缀，这里和 `base-href` 不能突，否则在渲染时有死循环问题

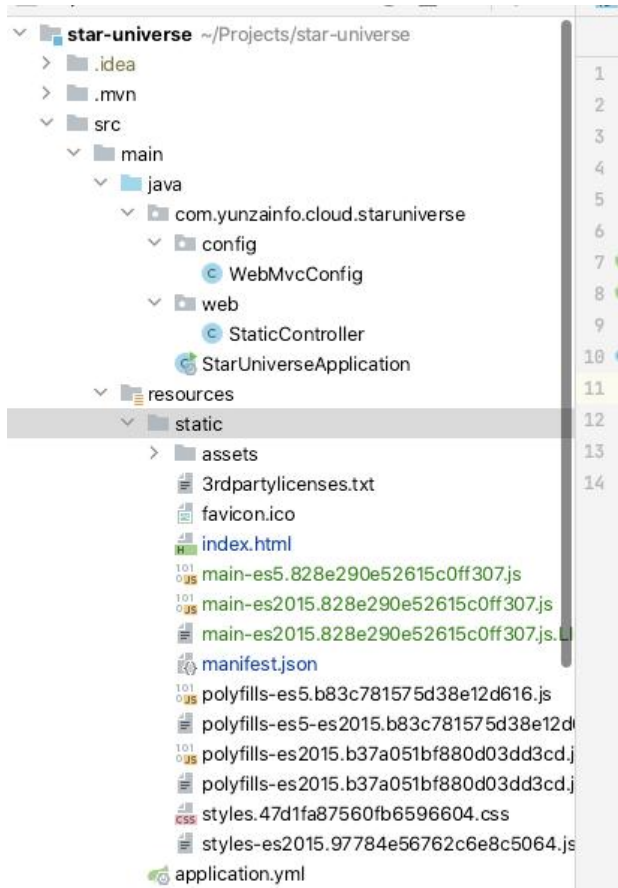
其次 `base-href` 就是项目访问路径前缀

然后修改 `angular.json` 中的 `build` 放到你的静态资源目录，这样在执行 `npm run build` 后静态资源自放入后台项目中的静态目录中



2.2 准备Portal后台项目

我的项目基于SpringBoot自动生成，项目结构如下



以下是用到的maven配置文件

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/
MLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xs
/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.4.3</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.yunzainfo.cloud</groupId>
    <artifactId>star-universe</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>star-universe</name>
    <description>Demo project for Spring Boot</description>
    <properties>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
```

```

<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter</artifactId>
</dependency>

<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-thymeleaf</artifactId>
</dependency>

<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>
</dependencies>

<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
    </plugin>
  </plugins>
</build>

</project>

```

一个Controller负责重定向路由到 [index.html](#)

```

package com.yunzainfo.cloud.staruniverse.web;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.ModelAndView;

import javax.servlet.http.HttpServletRequest;

@Controller
public class StaticController {

    @RequestMapping(value = {"/star-universe", "/star-universe/**"})
    public String index(HttpServletRequest request) {
        System.out.println(request.getRequestURI());
        return "index";
    }

}

```

portal的静态资源处理，通过实现 [WebMvcConfigurer](#)的 [addResourceHandlers](#)函数,注意函数传数的静态资源路径谓 [/static/star-universe/**](#)要和 [--deploy-url](#)对应起来，保证 [index.html](#)中的静态资源被正确加载

```

package com.yunzainfo.cloud.staruniverse.config;

import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
public class WebMvcConfig implements WebMvcConfigurer {
    @Override
    public void addResourceHandlers(ResourceHandlerRegistry registry) {
        registry.addResourceHandler("/static/star-universe/**").addResourceLocations("classpath:/static/");
    }
}

```

然后是portal的applicaiton.yml，给一个端口吧，这里我选择的是3000,和proxy.config中的相同，于静态资源路径为什么是 `/static`也是为了和注册应用时的路径保持一致

```

server:
  port: 3000

spring:
  thymeleaf:
    cache: false
    prefix: classpath:/static/

```

关于 `app.component`中的应用注册,`resourcePathPrefix`和 `manifest`全都加入应用路径完成远程注册

```

[
  {
    "name": "star-dust",
    "hostParent": "#wormhole",
    "hostClass": "star-dust",
    "routerPathPrefix": "/star-dust",
    "stylePrefix": "star-dust",
    "resourcePathPrefix": "http://localhost:3001/static/star-dust/",
    "loadSerial": true,
    "preload": false,
    "switchMode": "coexist",
    "scripts": [
      "main.js"
    ],
    "styles": [
      "styles.css"
    ],
    "manifest": "http://localhost:3001/static/star-dust/manifest.json"
  }
]

```

2.3 准备子项目前台项目

同样修改 `build`和 `angular.json` 然后打包

2.4 准备子项目后台项目

这里唯一和portal项目不同的时，要加入跨域访问允许，因为portal加载js/css的时候是远程加载的。

```
package com.yunzainfo.cloud.stardust.config;
```

```
import org.springframework.context.annotation.Configuration;  
import org.springframework.web.servlet.config.annotation.*;
```

```
@Configuration
```

```
public class WebMvcConfig implements WebMvcConfigurer {
```

```
    @Override  
    public void addResourceHandlers(ResourceHandlerRegistry registry) {  
        registry.addResourceHandler("/static/star-dust/**").addResourceLocations("classpath:/static/");  
    }
```

```
    @Override  
    public void addCorsMappings(CorsRegistry registry) {  
        registry.addMapping("/**")  
            .allowCredentials(true)  
            .allowedOrigins("http://localhost:3000", "http://localhost:3001", "http://localhost:3002", "http://localhost:3003")  
            .allowedMethods("POST", "GET", "PUT", "OPTIONS", "DELETE")  
            .maxAge(3600)  
            .allowCredentials(true);  
    }  
}
```

其余和portal一致，只是路径不一样

3.看效果

localhost:3000/star-universe/star-dust/one2

应用 应用与服务

subnav 1

option1

option2

option3

option4

subnav 2

subnav 3

Home / List / App

dust
uranus
1111
star-dust index works!

Elements Console Sources Network Performance Memory

Filter

Has blocked cookies Blocked Requests

1000 ms 2000 ms 3000 ms 4000 ms 5000 ms

Name	Status	Type	Initiator	Size	Time	Waterfall
one2	200	document	Other	5.6 kB	37 ms	
styles.47d1fa87560fb659660...	200	stylesheet	one2	237 B	181 ms	
polyfills-es2015.b37a051bf8...	200	script	one2	237 B	8 ms	
main-es2015.828e290e5261...	200	script	one2	237 B	124 ms	
token	200	xhr	polyfills-es20...	809 B	1.02 s	
ng-validate.js	200	script	content-script...	127 kB	7 ms	
user	200	xhr	polyfills-es20...	2.5 MB	2.24 s	
info	200	xhr	polyfills-es20...	1.4 kB	240 ms	
all	200	xhr	polyfills-es20...	147 kB	242 ms	
applications.json	200	xhr	polyfills-es20...	237 B	43 ms	
v2	200	xhr	polyfills-es20...	65.5 kB	454 ms	
2c948401775d18b2017761e...	200	png	main-es2015...	7.4 kB	55 ms	
menu-fold.svg	200	xhr	polyfills-es20...	237 B	6 ms	
user.svg	200	xhr	polyfills-es20...	237 B	10 ms	
laptop.svg	200	xhr	polyfills-es20...	237 B	9 ms	
notification.svg	200	xhr	polyfills-es20...	237 B	5 ms	
manifest.json?t=1614223865...	200	Other	Other	0 B	79 ms	
manifest.json?t=1614223865...	200	xhr	polyfills-es20...	641 B	32 ms	
main-es2015.c9fcb4d96827...	200	script	main-es2015...	237 B	32 ms	
styles.5391090c25d5c7d29b...	200	stylesheet	main-es2015...	237 B	242 ms	

Name	Headers	Preview	Response	Initiator	Timing
one2					
styles.47d1fa87560fb6596604.css					
polyfills-es2015.b37a051bf880d03dd3cd.js					
main-es2015.828e290e52615c0ff307.js					
token					
ng-validate.js					
user					
info					
all					
applications.json					
v2					
2c948401775d18b2017761e94852007d					
menu-fold.svg					
user.svg					
laptop.svg					
notification.svg					
manifest.json?t=1614223865541					
manifest.json?t=1614223865541					
main-es2015.c9fcb4d96827333d526a.js					
styles.5391090c25d5c7d29b23.css					

General

Request URL: http://localhost:3001/static/star-dust/manifest.json?t=1614223865541

Request Method: GET

Status Code: 200

Remote Address: [::1]:3001

Referrer Policy: strict-origin-when-cross-origin

Response Headers view source

Accept-Ranges: bytes

Access-Control-Allow-Credentials: true

Access-Control-Allow-Origin: http://localhost:3000

Connection: keep-alive

Content-Length: 237

Content-Type: application/json

Date: Thu, 25 Feb 2021 03:31:05 GMT

Keep-Alive: timeout=60

Last-Modified: Thu, 25 Feb 2021 03:05:58 GMT

Vary: Origin

Vary: Access-Control-Request-Method

Vary: Access-Control-Request-Headers

Request Headers view source

Accept: application/json, text/plain, */*

Accept-Encoding: gzip, deflate, br

Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7

这样，静态资源全都被远程请求过来了

4.这种远程技术可以运用到什么地方

首先组件共享的好处不说了，结合了后端微服务的 **真正**的前端微服务而不是依托于 **nginx**静态部署的端微服务

可以优化开发流程，可以做灰度部署了，可以完善CI了等等，好处也有，坏处也有。有利有弊，看各需要吧。