

实验6

1. 使用 gulp 编写资源压缩程序

在项目工程目录下面执行下面的一些代码

首先初始化整个项目

```
yarn init
```

初始化完成后项目目录中会生成出**package.json**这个文件

接下来对gulp进行安装

```
yarn global add gulp-cli
```

```
yarn add gulp
```

以及一些gulp插件的安装

```
//这是一些自己使用到的插件
```

```
yarn add gulp-minify-css gulp-uglify gulp-rename gulp-concat gulp-htmlmin gulp-imagemin
```

之后安装官网教程说的那样子在项目**根目录**来编写**gulpfile.js**文件。这里就展示个大概吧。

```
mobileWeb > 实验6 > jyt > gulpfile.js > gulp.task('minifyCss') callback
1  var gulp = require('gulp'),
2      minifyCss = require('gulp-minify-css'),
3      uglify = require('gulp-uglify'),
4      rename = require('gulp-rename'),
5      concat = require('gulp-concat')
6      htmlmin = require('gulp-htmlmin');
7
8  gulp.task('minifyCss',function(){
9      return gulp.src('sys/css/*.css')
10         .pipe(rename({suffix:'.min'}))
11         .pipe(minifyCss())
12         .pipe(concat('style.css'))
13         .pipe(gulp.dest('dist/css'));
14  });
15
16  gulp.task('minifyJs',function(){
17      return gulp.src('sys/js/*.js')
18         .pipe(uglify())
19         .pipe(gulp.dest('dist/js'));
20  });
21
22  gulp.task('minfyJsrender',function(){
23      return gulp.src('lib/jsrender/*.js')
24         .pipe(uglify())
25         .pipe(gulp.dest('dist/lib/jsrender'))
26  })
```

编写完成后执行一下,下面的指令(可选)

```
gulp --tasks
```

看看自己大半天到底忙了个啥东西出来。

```
[22:01:08] Tasks for ~\Desktop\web\mobileWeb\实验6\jyt\gulpfile.js
[22:01:08] |— minifyCss
[22:01:08] |— minifyJs
[22:01:08] |— minfyJsrender
[22:01:08] |— minnifyjQueryUi
[22:01:08] |— minnifyjQueryJs
[22:01:08] |— minfyHtml
```

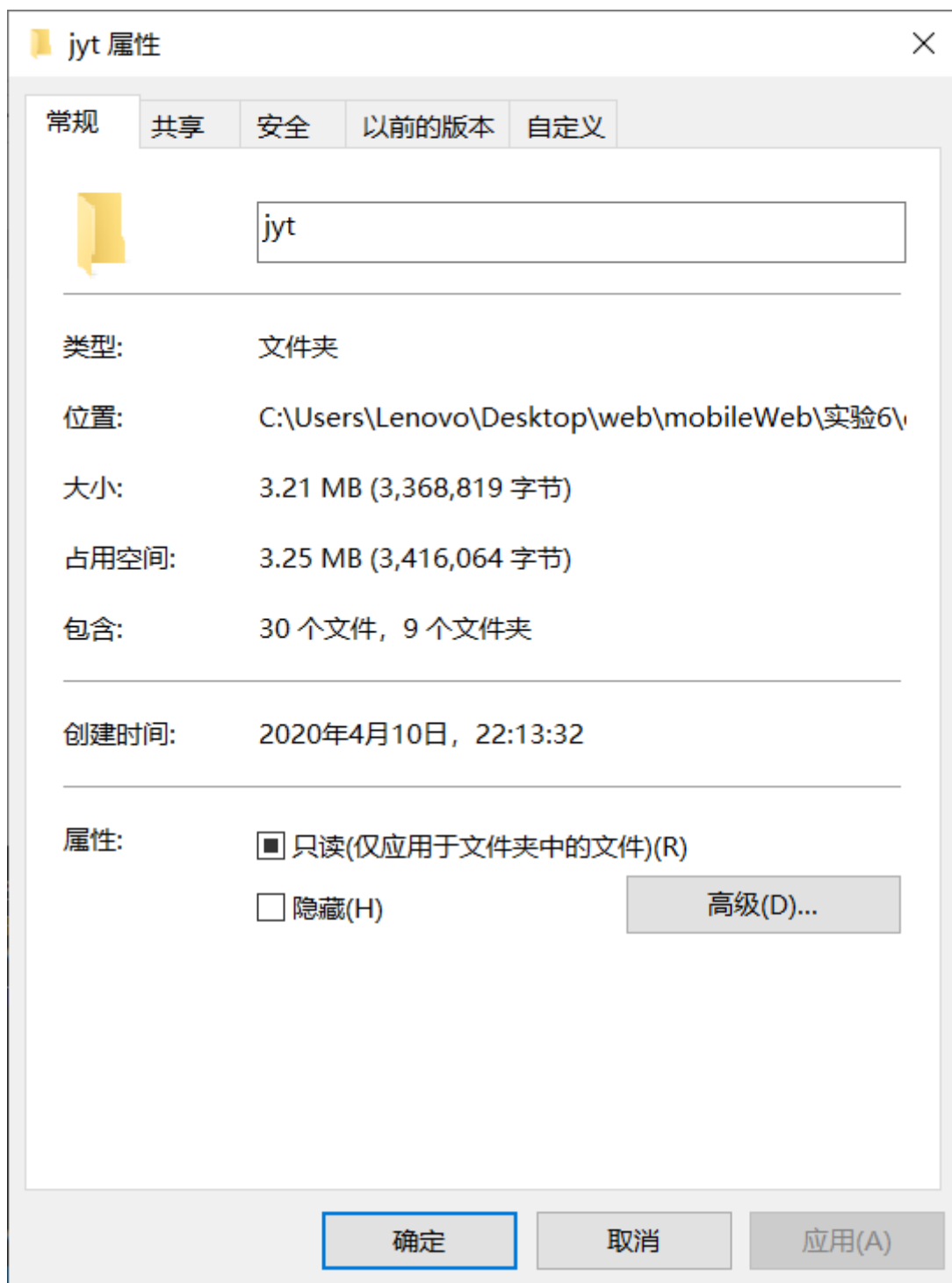
写好了后面执行一下就好了

```
gulp minifyCss
gulp minifyJs
gulp minfyJsrender
...
/**
 * 后面的都是一些重复性的东西了
 */
```

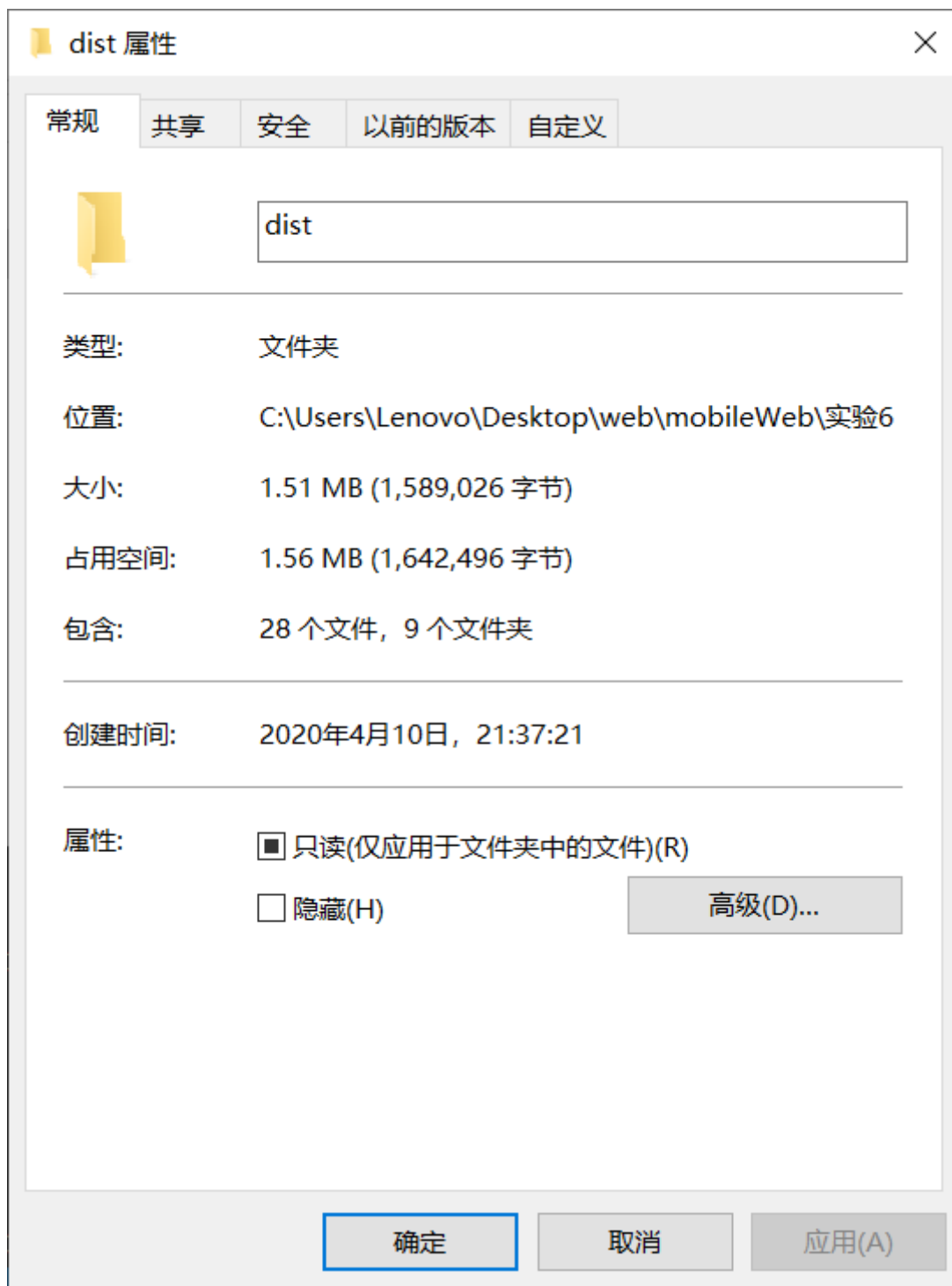
对于图片压缩并没有选择用gulp进行压缩,而是依旧选择了那个可爱的小熊猫去压缩

home_info_1.jpg	26.2 KB	Finished	4.5 KB	download	-83%
home_info_2.jpg	27.8 KB	Finished	5.2 KB	download	-81%
home_info_3.jpg	24.4 KB	Finished	3.9 KB	download	-84%
home_info_4.jpg	27.9 KB	Finished	5.6 KB	download	-80%
home_info_5.jpg	35.1 KB	Finished	8.7 KB	download	-75%
logo.png	7.2 KB	Finished	1.7 KB	download	-77%
slider1.jpg	691.7 KB	Finished	326.0 KB	download	-53%
slider2.jpg	663.4 KB	Finished	305.3 KB	download	-54%
slider3.jpg	422.5 KB	Finished	179.7 KB	download	-57%
slider4.jpg	710.3 KB	Finished	340.0 KB	download	-52%

下面来看看效果如何吧,下面是压缩前的整个文件,可以看到有**3.21M**的大小




把一些未被压缩的文件原封不动的放进来,下面看看压缩后的



WOW可以看到小了一半

2. 替换标题和菜单的字体

在网上随便找了一个中文字体把它下载了下来

 chinese.ttf	1995/9/15 9:47	TrueType 字体文件	1,304 KB
---	----------------	---------------	----------

使用nodejs字体转换程序进行转换,安装一下



```
[4/4] Building fresh packages...
success Installed "ttf2eot@2.0.0" with binaries:
- ttf2eot
success Installed "ttf2woff@2.0.1" with binaries:
- ttf2woff
success Installed "ttf2svg@1.2.0" with binaries:
- ttf2svg
success Installed "svg2ttf@4.3.0" with binaries:
- svg2ttf
```

然后使用其将字体文件进行转换,看到社区里面说一些中文字体转换后可能会有问题
我这个这么老而且low的字体应该没事吧(但愿没事)

执行程序

```
PS C:\Users\Lenovo\Desktop\web\mobileWeb\实验6\jyt> ttf2woff chinese.ttf chinese.woff
```

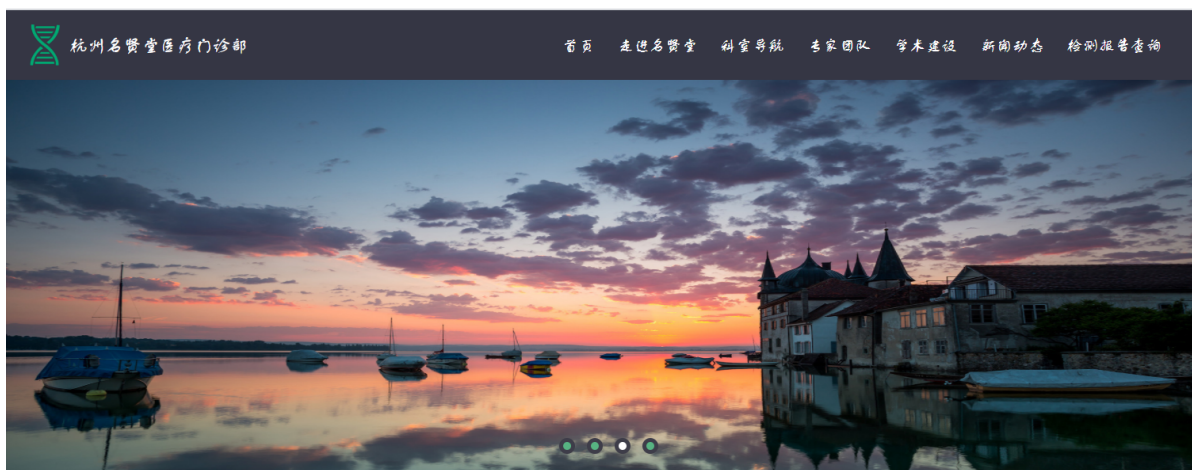
生成成功

 chinese.ttf	1995/9/15 9:47	TrueType 字体文件	1,304 KB
 chinese.woff	2020/4/10 23:06	WOFF 文件	851 KB

修改css

```
@font-face{
  font-family:'newChinese';
  font-style: normal;
  font-weight: 400;
  src: url('.././chinese.woff') format('woff');
}
```











看下效果



3.找些有意思的东西

Unicode base64编码器我认为很有意思,因为平时后端在进行存储时经常会用到base64的相关转换。文件图片等等都可以转换为base64编码进行相关存储

me.

-  - doughnut
-  - cookie
-  - popcorn
-  - ice cream
-  - cake
-  - cupcake
-  - pie
-  - chocolate
-  - candy
-  - lollipop

2DzfaQAgIBMAIABkAG8AdQBnAGgAbgB1AHQA
Ctg832oAICATACAAYwBvAG8AawBpAGUActg8
338AICATACAACABvAHAAYwBvAHIAbgAK2Dzf
ZgAgIBMAIABpAGMAZQAgAGMAcglAGEAbQAK
2DzfggAgIBMAIABjAGEAawBlAArYPt3BACAg
EwAgAGMAdQBwAGMAYQBrAGUActg+3WcAICAT
ACAACABpAGUActg832sAICATACAAYwBoAG8A
YwBvAGwAYQB0AGUActg832wAICATACAAYwBh
AG4AZAB5AArYPN9tACAgEwAgAGwAbwBsAGwA
aQBwAG8AcA==