## 章节介绍

我们使用 TypeScript 可以进行多文件的合并，不过这不推荐。但毕竟是在讲 typescript 所以这块内容也是要讲的。
使用 webpack 进行 TypeScript 模块打包是推荐的，后盾人 (opens new window)已经录制了 webpack 与 typescript 课程

## #命名空间

TypeScript 像其他编程语言一样，存在命名空间（namespace）的概念。
当我们定义以下同名变量时是不允许的，这种情况可以通过命名空间解决

```
let name: string = '后盾人'
let name: string = 'houdunren.com'

```

下面是使用命名空间将变量隔离

- 数据需要使用 export 导出才可以使用
- 子命名空间也需要 export 后才可以使用

```
namespace User {
    export let name: string = '后盾人'
}
namespace Member {
    let name: string = 'houdunren.com'
}

console.log(User.name);

console.log(Member.name); //报错，因为没有使用 export 将变量导出

```

命名空间支持嵌套

```
namespace User {
    export let name: string = '后盾人'
    export namespace Member {
```

```
        export let name: string = 'houdunren.com'
    }
}

console.log(User.name);
console.log(User.Member.name); //houdunren.com 获取子命名空间中的数据

```

## #单独编译

下面将每个 ts 文件单独进行编译，然后在 html 文件中依次引入

首先创建 tsconfig.js

```
tsc --init

```

然后执行文件监测

```
tsc -w

```

下面我们创建`user.ts`模块文件

```
namespace User {
    export let name: string = '后盾人'
}
namespace Member {
    export let name: string = 'houdunren.com'
}

```

在 index.ts 文件中定义业务内容，即使用 User.ts 中的数据 User.name

```
console.log(User.name);

```

然后执行编译

```
tsc -w

```

创建 hd.html 文件引入这两个 JS 文件，注意要将 namespace.js 文件先引入

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>后盾人</title>
        <script src="user.js" defer></script>
        <script src="index.js" defer></script>
    </head>
    <body></body>
</html>

```

## #合并打包

上面的多文件处理方式非常不方便，如果有多个文件要写多个 script 标签来引入。
下面我们来介绍通过命令将多个文件进行打包

```
tsc --outFile ./dist/app.js user.ts index.ts

```

然后在 hd.html 文件引入打包好的文件

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>后盾人</title>
        <script src="dist/app.js" defer></script>
    </head>
    <body></body>
</html>

```

## #reference

如果存在多个文件都像上面一样在命令行填写，显然是很麻烦了。我们可以使用以下方法优化

在 index.ts 中使用 `reference` 引入依赖的文件

```
/// <reference path="user.ts"/>
console.log(User.name);

```

然后使用下面的命令编译到一个文件中

```
tsc --outFile ./dist/app.js index.ts

```

然后访问 hd.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>后盾人</title>
        <script src="dist/app.js" defer></script>
    </head>
    <body></body>
</html>

```

## #amd

使用 amd 模块打包较上面的打包方式方便些，因为我们可以在代码中使用 js 模块的 export/import 语法

> 有关于 JS 的模块已经在后盾人网站 (opens new window)发布请登录学习。

首先创建 ts 配置文件 tsconfig.js

```
tsc --init

```

然后修改配置项

```
...
"module": "amd"
...

```

然后创建 user.ts，并且只导出 User 类

```
export namespace User {
    export let name: string = '后盾人'
}
export namespace Member {
    export let name: string = 'houdunren.com'
}

```

在 index.ts 文件中 import 导入 ts 编译后的 User.js 模块

```
import { User } from './user.js'

```

然后执行命令进行编译

```
tsc --outFile ./dist/app.js

```

然后修改 hd.html 文件内容

- 因为是 amd 模块所以需要使用 require.js 处理

> 有关 require.js 的工作原理，可以访问后盾人网站上的 js 模块章节内容，向军大叔已经有讲

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <script src="
https://cdn.bootcdn.net/ajax/libs/require.js/2.3.6/require.min.js
    <script src="
  </head>
```

```
  <body>
    <script src="dist/app.js"></script>
    <script>
      require(['App'])
      require(['User'], User => {
        console.log(User.title)
      })
    </script>
  </body>
</html>
```