



链滴

angular 组件实现模糊查询并且实现保存删除功能

作者: [wangwenbo](#)

原文链接: <https://ld246.com/article/1547634826835>

来源网站: [链滴](#)

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

Chips Autocomplete组件:

(这是一个弹窗的案例)

1.首先是html代码:

```
<mat-form-field class="example-chip-list">
  <mat-chip-list #chipList>
    <mat-chip
      *ngFor="let user of selectedUsers"
      [selectable]="selectable"
      [removable]="removable"
      (removed)="removePeople(user)"
      [compareWith]="compareObjects"
      style="cursor: pointer;">
      {{user.firstName}}
      <mat-icon matChipRemove *ngIf="removable">cancel</mat-icon>
    </mat-chip>
    <input matInput
      placeholder="添加人员"
      #personInput
      [formControl]="peopleControl"
      [matAutocomplete]="auto"
      [matChipInputFor]="chipList"
      [disabled]="operation=='show'">
  </mat-chip-list>
  <mat-autocomplete #auto="matAutocomplete">
    <mat-option *ngFor="let user of users" [value]="user" (click)="selectedPeople(user)">
      {{user.firstName}}
    </mat-option>
  </mat-autocomplete>
</mat-form-field>
```

2.ts文件里面的内容

```
// 人员控制器
peopleControl: FormControl = new FormControl();
// 模糊查询人员
users: User[] = [];
user: User = new User();
// 选择的人员信息
selectedUsers: User[] = [];
// 组件自带的变量声明
visible = true;
selectable = true;
removable = true;
account: Account;

@ViewChild('personInput') personInput: ElementRef<any>;
```

3. (1) ts文件的constructor()

以及一些要用的service的声明,

```
private principal: Principal,
```

(2) constructor的{}内容

```
// 获取用户信息
principal.identity().then(account => {
  this.account = account;
});
// 判断条目里的user是否存在，存在执行下面的代码
if (this.Tasks.users) {
  this.selectedUsers = this.Tasks.users;
}
// 监听这个control下的input输入框（判断输入框内容是否变化，然后根据值请求后台，后台返回
// 的数据）
this.peopleControl.valueChanges.subscribe(
  name => {
    // 判断name是否存在，并且是不是字符串类型
    if (name && typeof name === 'string' && name.trim().length > 0) {
      // 当name符合条件时，执行搜索人员方法
      this.searchPeople(name.trim());
    }
  }
);
```

4.ngOnInit

```
// 当用户执行的操作是新增的时候，请求当前用户的信息，在选择框中默认显示选中当前用户
if (this.operation === 'add') {
  this.usersService.findById(this.account.id).subscribe(
    (res) => {
      this.user = res.body;
      this.selectedUsers.push(this.user);
    });
}
```

5.相关请求方法（ts文件）：

```
// 删除选中的用户
removePeople(person): void {
  const index = this.selectedUsers.findIndex(p => p.id === person.id);
  if (index !== -1) {
    person.id = undefined;
    this.selectedUsers.splice(index, 1);
  }
}
```

```
//选中的用户
selectedPeople(person): void {
  const index = this.selectedUsers.findIndex(p => p.id === person.id);
  if (index === -1) {
    this.selectedUsers.push(person);
  }
  this.personInput.nativeElement.value = "";
}
```

```
// 根据名称模糊搜索人员
private searchPeople(name: string) {
```

```

this.users = [];
this.isSaving = true;
this.userService.queryByName({
  name: name
}).subscribe(res => {
  this.users = res.body;
  this.isSaving = false;
}, error1 => {
  console.error(error1);
  this.isSaving = false;
});
}

// 比较打开页面后，显示这个对象已经被选中的信息
compareObjects(o1: any, o2: any) {
  if (o1 && o2) {
    return o1.name === o2.name && o1.id === o2.id;
  }
  return false;
}

// 最后再执行保存方法
save(){}

```

6.在父组件中要引入相应的module

```

imports: [
  MatChipsModule,
],
providers: [
  {
    provide: MAT_CHIPS_DEFAULT_OPTIONS,
    useValue: {
      separatorKeyCodes: [ENTER, COMMA]
    }
  }
]

```