# 一、FormGroupName

|  |  |
| --- | --- |
| valid: boolean | null | ***只读***  报告控件是否有效。如果控件的当前值没有验证错误，则认为该控件是有效的。如果控件不存在，则返回null |
| invalid: boolean | null | ***只读***  报告控件是否无效，即输入值中存在错误。如果控件不存在，则返回null。 |
| pending: boolean | null | ***只读***  报告控件是否挂起，这意味着发生异步验证并且输入值还没有错误。如果控件不存在，则返回null。 |
| disabled: boolean | null | ***只读***  报告控件是否被禁用，这意味着该控件在UI中被禁用，并且不受验证检查的约束，也不受祖先控件的聚合值的排斥。如果控件不存在，则返回null。 |
| enabled: boolean | null | ***只读***  报告是否启用该控件，这意味着该控件包含在有效性或值的祖先计算中。如果控件不存在，则返回null。 |
| errors: [ValidationErrors](https://angular.cn/api/forms/ValidationErrors) | null | ***只读***  报告控件的验证错误。如果控件不存在，则返回null。 |
| pristine: boolean | null | ***只读***  报告控件是否原始，这意味着用户尚未更改UI中的值。如果控件不存在，则返回null。 |
| dirty: boolean | null | ***只读***  报告控件是否脏，这意味着用户更改了UI中的值。如果控件不存在，则返回null。 |
| touched: boolean | null | ***只读***  报告控件是否被触摸，这意味着用户在控件上触发了一个模糊事件。如果控件不存在，则返回null。 |
| status: string | null | ***只读***  报告控件的验证状态。可能的值包括:“VALID”、“INVALID”、“DISABLED”和“PENDING”。如果控件不存在，则返回null。 |
| untouched: boolean | null | ***只读***  报告控件是否未被触及，这意味着用户尚未在其上触发模糊事件。如果控件不存在，则返回null。 |
| statusChanges: Observable<any> | null | ***只读***  返回一个多播观察，该观察在为控件计算验证状态时发出验证状态。如果控件不存在，则返回null。 |
| valueChanges: Observable<any> | null | ***只读***  返回控件值更改的多播观察，每当控件的值在UI或编程中更改时发出该观察。如果控件不存在，则返回null。 |
| path: string[] | null | ***只读***  返回一个数组，该数组表示从顶级窗体到此控件的路径。每个索引都是该级别上控件的字符串名称。 |

# 二、**formGroup的校验触发**

# / 需要在app.module.ts 中添加

# import { errorStateMatcher, showOnDirtyErrorStateMatcher } from '@angular/material'

# providers: [

# { provide: errorStateMatcher, useClass: showOnDirtyErrorStateMatcher }

# ],

<mati -error>仅在FormControl无效时显示，但您已经对formgroup进行了验证。在这种情况下，你需要使用ErrorStateMatcher  
在你的情况下，它看起来是这样的:

导出类MyErrorStateMatcher实现ErrorStateMatcher {

isErrorState（control：FormControl | null，form：FormGroupDirective | NgForm | null）：boolean {

const invalidCtrl = !!（control&& control.invalid）;

const invalidParent = !!（control&& control.parent&& control.parent.invalid）;

return（invalidCtrl || invalidParent）;

}

}

另外值得一提的是，不建议有两个绑定，即 formControl 和 ngModel 。删除 ngModel 并改为使用表单控件。如果您稍后收到开始日期和结束日期，则可以使用 patchValue （只需设置一些值以形成）或 setValue （将所有值设置为表格）

在组件中标记errorstatematcher：

matcher = new MyErrorStateMatcher（）;

至于您的自定义验证器，您无需解析日期，只需检查结束日期小于开始日期：

checkDates（group：FormGroup）{

if（ group.controls.endDate.value< group.controls.startDate.value）{

return {endDateLessThanStartDate：true}

}

返回null;

}

然后在模板中标记错误状态匹配器：

< mat-form-field>

< input matInput [matDatepicker] =picker2type =textformControlName =endDate[errorStateMatcher] =matcher>

< mat-datepicker-toggle matSuffix [for] =picker2>< / mat-datepicker-toggle>

< mat-datepicker＃picker2>< / mat-datepicker>

< mat-error \* ngIf =myForm.hasError（'endDateLessThanStartDate'）>结束日期不能早于开始日期< / mat-error>

< / mat-form-field>

# 二、

# 二、

# 二、