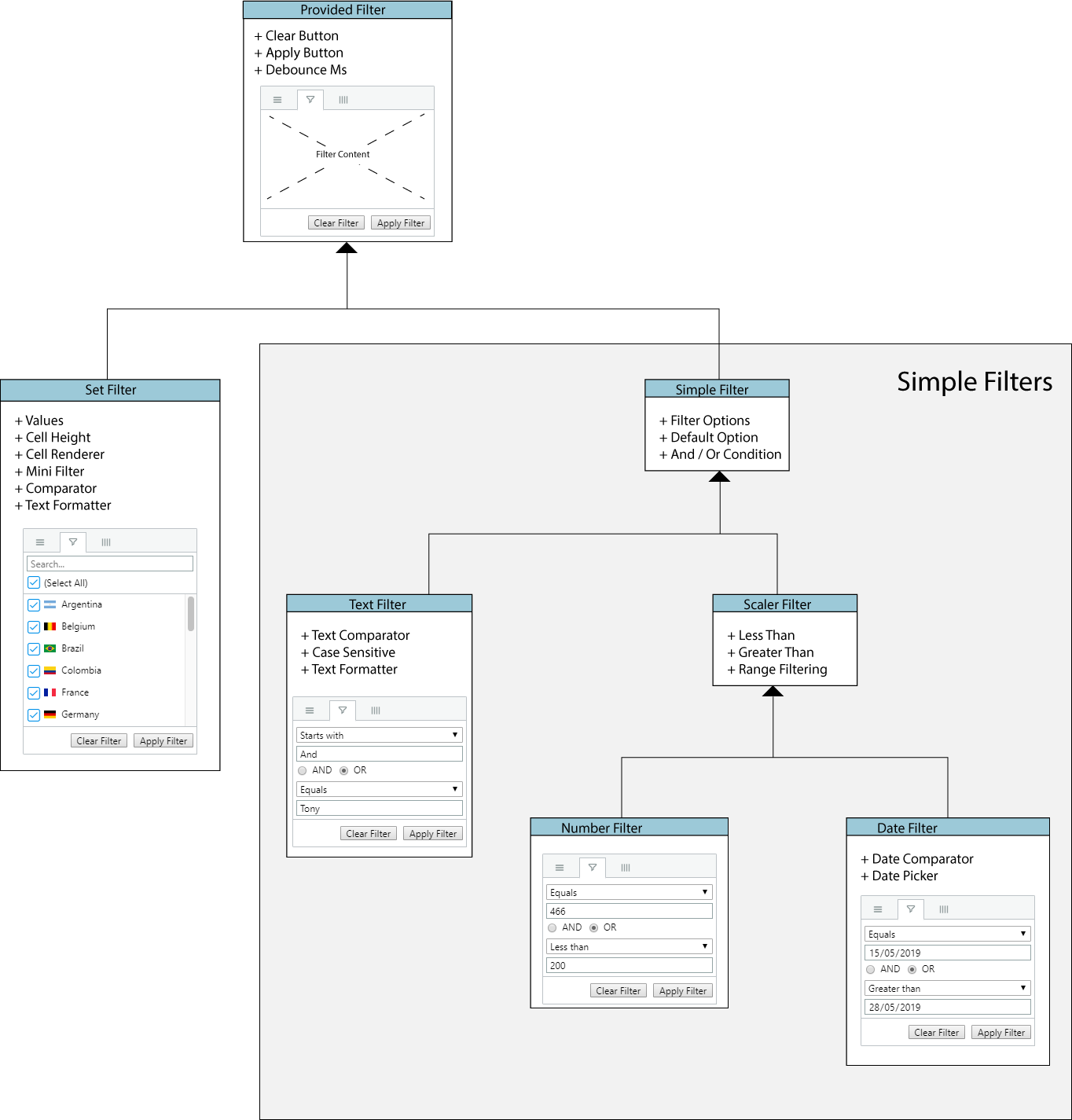
提供的过滤器的结构  
下图概述了过滤器的结构。每个框表示一个过滤器类型，其中列出了函数。例如，所有提供的过滤器都有应用和清除按钮逻辑。只有日期筛选器有日期比较器或日期选择器



简单的过滤器模型

当在过滤器上保存或恢复状态时，使用过滤器模型。过滤器模型表示过滤器的状态。如，下面的代码首先获取，然后设置运动员列的过滤器模型:

// get filter instance 获得过滤器实例

var filterInstance = gridOptions.api.getFilterInstance('athlete');

// get filter model

var model = filterInstance.getModel();

// OR set filter model and update

filterInstance.setModel({

type:'endsWith',

filter:'thing'

});

// tell the grid to refresh rows based on the filter. the filter does not

// do this automatically in case multiple filters are going to get set.

gridOptions.api.onFilterChanged()

本节解释每个简单过滤器的过滤器模型。每个filter类型后面的接口如下:

1 - text // text filter uses this filter model

interface TextFilterModel {

// always 'text' for text filter

filterType: string;

// one of the filter options eg 'equals'

type: string;

// the text value associated with the filter.

// it's optional as custom filters may not

// have a text value

filter?: string;

}

2 - number // number filter uses this filter model

interface NumberFilterModel {

// always 'number' for number filter

filterType: string;

// one of the filter options eg 'equals'

type: string;

// the number value(s) associated with the filter.

// custom filters can have no values (hence both are optional).

// range filter has two values (from and to).

filter?: number;

filterTo?: number;

}

3 - date // date filter uses this filter model

interface DateFilterModel {

// always 'date' for date filter

filterType: string;

// one of the filter options eg 'equals'

type: string;

// the date value(s) associated with the filter.

// the type is string and format is always YYYY-MM-DD eg 2019-05-24

// custom filters can have no values (hence both are optional).

// range filter has two values (from and to).

dateFrom: string;

dateTo: string;

}

过滤器模型实例的例子如下

// number filter with one condition, with equals type

var numberLessThan35 = {

filterType: 'number',

type: 'lessThan',

filter: 35

};

// number filter with one condition, with inRange type

var numberBetween35And40 = {

filterType: 'number',

type: 'inRange',

filter: 35,

filterTo: 40

};

自定义过滤器选项

对于具有定制过滤要求的应用程序，还可以向数字、文本和日期过滤器添加新的自定义过滤选项。例如，“Not Equal (with Nulls)”筛选器选项可以与内置的“Not Equal”选项一起使用。  
 自定义过滤器选项通过filterParams提供给网格。filterOptions必须符合以下接口:

export interface IFilterOptionDef {

displayKey: string;

displayName: string;

test: (filterValue: any, cellValue: any) => boolean;

hideFilterInput?: boolean;

}

displayKey应该包含一个不与内置过滤器键冲突的唯一键值。还应该提供默认的displayName，但是可以使用localeTextFunc用特定于地区的值代替。  
自定义过滤器逻辑是通过测试函数实现的，该函数从网格中接收用户输入的filterValue和cellValue，并返回true或false。  
还可以通过启用可选属性hideFilterInput来隐藏筛选器输入字段。  
自定义FilterOptionDef可以与内置的过滤器选项string keys a一起提供

{

field: "age",

filter: 'agNumberColumnFilter',

filterParams: {

filterOptions: [

'lessThan',

{

displayKey: 'lessThanWithNulls',

displayName: 'Less Than with Nulls',

test: function(filterValue, cellValue) {

return cellValue == null || cellValue < filterValue;

}

},

'greaterThan',

{

displayKey: 'greaterThanWithNulls',

displayName: 'Greater Than with Nulls',

test: function(filterValue, cellValue) {

return cellValue == null || cellValue > filterValue;

}

}

]

}

}

下面的例子演示了几个自定义过滤器选项:

代码如下：https://www.ag-grid.com/javascript-grid-filter-provided-simple/

“Athlete”列包含两个自定义筛选选项，以“A”开头，以“B”开头。这两个选项都不接受文本筛选器输入。  
 “Age”列包含两个自定义筛选选项偶数、奇数和空白。它还使用了'empty'过滤器和suppressAndOrCondition=true。  
 “Date”列包含一个自定义的equalsWithNulls过滤器。请注意，对于内置的日期筛选器选项，例如equals，仍然需要一个定制的比较器。  
 “Country”列包含一个定制的notEqualNoNulls fi