目录

readme

特性

基于 [bootstrap]和[Light-Year-Admin-Template]的后台模板， 封装了大部分常用组件。

扩展依赖

> tpext https://gitee.com/tpext/tpext

> tpextbuilder https://gitee.com/tpext/tpextbuilder

> tpextmanager https://gitee.com/tpext/tpextmanager

> lightyearadmin https://gitee.com/tpext/lightyearadmin

> tpextmyadmin https://gitee.com/tpext/tpextmyadmin

.gitignore

/.idea /.vscode /.vscode /.obsidian _config.yml

安装

框架安装

安装

框架有两个版本

- `5.0` 基于thinkphp 5.1 (推荐(文档基于此版本))
- `6.0` 基于thinkphp 6.0 (beta)

框架需要使用 composer和git

安装 `composer`

https://pkg.phpcomposer.com/#how-to-install-composer/

安装 `git`

https://git-scm.com/

使用 **`composer`** 全新安装

> 安装 thinkphp(`5.0` 或 `6.0` ，根据您的需要，选择其中一个版本)，

[5.0]分支对应 `tpextmyadmin` 的[1.0]分支，依次执行以下命令， `myadmin` 为新项目目录，可自行调整

```
composer create-project topthink/think=5.1.* myadmin

cd myadmin

composer require ichynul/tpextmyadmin:^1.*
```

或

[6.0]分支对应 `tpextmyadmin` 的[3.0]分支，依次执行以下命令， `myadmin6` 为新项目目录，可自行调整

```
composer create-project topthink/think=6.0.* myadmin6

cd myadmin6

composer require ichynul/tpextmyadmin:^3.*
```

> 安装完毕，此安装版是最小模式，只包含基本的后台功能，建议开发新项目时使用此方式。

使用 **git** 安装演示站

> 拉取 `5.0` 分支代码，依次执行以下命令， `myadmin` 为新项目目录，可自行调整

```
git clone -b 5.0 https://github.com/hi-tpext/myadmin.git myadmin

cd myadmin

composer update
```

或 拉取 `6.0` 分支代码，依次执行以下命令， `myadmin6` 为新项目目录，可自行调整

```
git clone -b 6.0 https://github.com/hi-tpext/myadmin.git myadmin6

cd myadmin6

composer update
```

> 相关演示代码在 `application/admin/` 或 `app/admin/` 中，数据库脚本由 `[myadmindata]` 扩展提供，请下载安装。 安装完毕，此安装版是最和演示站同步的，如果你想自己搭建演示站可用此方式。 注意：此方式的仓库是不带 `composer` 依赖 `vendor` 目录和 `thinkphp` 目录的，请务必运行 `composer update` 安装所有依赖后再访问网站。

后台配置

配置

- `apache/nginx` 重写规则，略(自行百度) *重要
- 没配置 `重写规则` 的话后续的url中加上 `index.php`
- 如： `http://localhost:8081/index.php/admin`

扩展安装

扩展

不出意外的话会自动跳转到扩展安装页面，现有后台相关功能都是基于扩展的方式完成的。

初始扩展有5个

> tpext.core => 核心模块提供扩展的基本支持 tpext.builder => ui生成功能支持 tpext.manager => 扩展管理[安装/卸载/刷新资源/配置]
> lightyear.admin => 后台静态资源包 tpext.myadmin => 后台逻辑，管理员、权限、日志等后台基础功能

依次安装它们，最后安装完tpextmyadmin后，再次打http://youhost/admin，正式进入后台。

后台默认账号

> admin tpextadmin

功能开发

常规方式

常规方式

你可以按常规方式，在thinkphp【admin】模块里面开发

比如创建文件 `/application/admin/controller/Member.php` :

```
<?php

namespace app\admin\controller;

use app\common\logic\MemberLogic;
use app\common\model;
use think\Controller;
use tpext\builder\traits\HasBuilder;

/**
 * Undocumented class
 * @title 会员管理
 */
class Member extends Controller
{
    use HasBuilder;

    /**
     * Undocumented variable
     *
     * @var model\Member
     */
    protected $dataModel;

    protected function initialize()
    {
        $this->dataModel = new model\Member;
        $this->pageTitle = '会员管理';
        $this->enableField = 'status';
        $this->pagesize = 8;

        //作为下拉选择数据源  相关设置
        $this->selectTextField = '{id}#{nickname}({mobile})';//显示
        $this->selectFields = 'id,nickname,mobile';// ->field('id,nickname,mobile') 字段，配合[显示],优化查询性能
        $this->selectSearch = 'username|nickname|mobile'; //关键字  like  查询字段  ->where('username|nickname|mobile', 'like', $kwd);
    }

    protected function filterWhere()
    {
        $searchData = request()->post();

        $where = [];

        if (!empty($searchData['id'])) {
            $where[] = ['id', 'eq', $searchData['id']];
        }

        if (!empty($searchData['username'])) {
            $where[] = ['username', 'like', '%' . $searchData['username'] . '%'];
        }
        if (!empty($searchData['nickname'])) {
            $where[] = ['nickname', 'like', '%' . $searchData['nickname'] . '%'];
        }
        if (!empty($searchData['mobile'])) {
            $where[] = ['mobile', 'like', '%' . $searchData['mobile'] . '%'];
        }
        if (isset($searchData['status']) && $searchData['status'] != '') {
```

```
            $where[] = ['status', 'eq', $searchData['status']];
        }
        if (isset($searchData['level']) && $searchData['level'] != '') {
            $where[] = ['level', 'eq', $searchData['level']];
        }
        if (!empty($searchData['province'])) {
            $where[] = ['province', 'eq', $searchData['province']];
            if (!empty($searchData['city'])) {
                $where[] = ['city', 'eq', $searchData['city']];
                if (!empty($searchData['area'])) {
                    $where[] = ['area', 'eq', $searchData['area']];
                }
            }
        }

        return $where;
    }

    /**
     * 构建搜索
     *
     * @return void
     */
    protected function builSearch()
    {
        $search = $this->search;

        $search->text('id', '会员id')->maxlength(20);
        $search->text('username', '账号')->maxlength(20);
        $search->text('nickname', '昵称')->maxlength(20);
        $search->text('mobile', '手机号')->maxlength(20);

        $search->select('level', '等级')->optionsData(model\MemberLevel::order('level')->select(), 'name', 'level')->afterOptions([0 => '普通会员']);
        $search->select('status', '状态')->options([0 => '禁用', 1 => '正常']);
        $search->select('province', '省份')->dataUrl(url('api/areacity/province'), 'ext_name')->withNext(
            $search->select('city', '城市')->dataUrl(url('api/areacity/city'), 'ext_name')->withNext(
                $search->select('area', '地区')->dataUrl(url('api/areacity/area'), 'ext_name')
            )
        );
    }

    /**
     * 构建表格
     *
     * @return void
     */
    protected function buildTable(&$data = [])
    {
        $table = $this->table;

        $table->show('id', 'ID');
        $table->image('avatar', '头像')->thumbSize(50, 50)->default('/static/images/touxiang.png');
        $table->show('username', '账号');
        $table->text('nickname', '昵称')->autoPost()->getWrapper()->addStyle('width:130px');
        $table->show('mobile', '手机号')->getWrapper()->addStyle('width:100px');
        $table->match('gender', '性别')->options([1 => '男', 2 => '女', 0 => '未知'])->getWrapper()->addStyle('width:50px');
        $table->show('age', '性别');
        $table->show('level_name', '等级');
        $table->show('money', model\MemberAccount::$types['money']);
        $table->show('points', model\MemberAccount::$types['points']);
        $table->show('pca', '省市区');
        $table->switchBtn('status', '状态')->default(1)->autoPost()->getWrapper()->addStyle('width:60px');
        $table->show('last_login_time', '最近登录')->getWrapper()->addStyle('width:150px');
        $table->show('create_time', '注册时间')->getWrapper()->addStyle('width:150px');

        $table->sortable('id,sort,money,points,commission,re_comm,shares,last_login_time');

        $table->getToolbar()
            ->btnAdd()
            ->btnEnableAndDisable('启用', '禁用')
            ->btnRefresh();
```

```
        $table->getActionbar()
            ->btnEdit()
            ->btnView()
            ->btnLink('account', url('/admin/memberaccount/add', ['member_id' => '__data.pk__']), '', 'btn-success', 'mdi-square-inc-cash');
    }

    /**
     * 构建表单
     *
     * @param boolean $isEdit
     * @param array $data
     */
    protected function builForm($isEdit, &$data = [])
    {
        $form = $this->form;

        $form->tab('基本信息');
        $form->image('avatar', '头像')->thumbSize(50, 50);
        $form->text('username', '账号')->required()->maxlength(20);
        $form->text('nickname', '昵称')->required()->maxlength(20);
        $form->text('mobile', '手机号')->maxlength(11);
        $form->text('email', '邮件')->maxlength(60);
        $form->radio('gender', '性别')->options([0 => '未知', 1 => '男', 2 => '女'])->default(0);
        $form->number('age', '年龄')->max(100)->min(1)->default(18);

        if ($isEdit) {
            $form->show('points', model\MemberAccount::$types['points']);
            $form->show('money', model\MemberAccount::$types['money']);
        }

        $form->tab('其他信息');
        $form->fields('省/市/区');
        $form->select('province', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/province'), 'ext_name')->withNext(
            $form->select('city', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/city'), 'ext_name')->withNext(
                $form->select('area', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/area'), 'ext_name')
            )
        );
        $form->fieldsEnd();

        $form->textarea('remark', '备注')->maxlength(255);
        $form->switchBtn('status', '状态')->default(1);
        $form->image('erweima_img', '二维码')->thumbSize(50, 50);

        $levels = model\MemberLevel::order('level')->field('name,level')->select();

        if ($isEdit) {
            $form->match('level', '等级')->optionsData($levels, 'name', 'level');

            $form->show('last_login_ip', '最近登录IP')->default('-');

            if (session('admin_id') == 1) {
                $form->show('openid', 'openid')->default('-');
            }

            $form->show('last_login_time', '最近登录时间');
            $form->show('create_time', '注册时间');
            $form->show('update_time', '修改时间');
        }
    }

    /**
     * 保存数据
     *
     * @param integer $id
     * @return void
     */
    private function save($id = 0)
    {
        $data = request()->only([
            'avatar',
```

```
            'username',
            'nickname',
            'mobile',
            'email',
            'gender',
            'age',
            'level',
            'province',
            'city',
            'area',
            'status',
            'remark',
            'erweima_img',
        ], 'post');

        $result = $this->validate($data, [
            'username|账号' => 'require',
            'nickname|昵称' => 'require',
            'mobile|手机号' => 'mobile',
            'level|等级' => 'number',
            'age|年龄' => 'number',
        ]);

        if (true !== $result) {

            $this->error($result);
        }

        if ($data['mobile'] && $exist = $this->dataModel->where(['mobile' => $data['mobile']])->find()) {
            if ($id) {
                if ($exist['id'] != $id) {
                    $this->error('手机号未能修改，已被占用');
                }
            } else {
                $this->error('手机号已被占用');
            }
        }

        return $this->doSave($data, $id);
    }
}
```

不使用 `HasBuilder` 的话，就是常规开发，按tp5.1来。

扩展开发

扩展开发

> 可参考：tpext.myadmin[https://gitee.com/tpext/tpextmyadmin] 提供了后台基础逻辑，管理员、权限、日志等后台基础功能。

你可以开发自己的扩展类增加功能

```
<?php

namespace com\youadmin\common;

use tpext\common\Module as baseModule;

class Module extends baseModule
{
    protected $version = '1.0.1';

    protected $name = 'youname.module';

    protected $title = '我的框架';

    protected $description = '我的框架提供了xxxx功能';

    //扩展根目录，如果是以composer方式扩展，程序代码放在src目录，则此处为/../../，如果是以extend方式开发，则为/../
    protected $root = __DIR__ . '/../../';

    protected $assets = 'assets';  //如果有静态资源，否则不用写

    protected $modules = [
        'admin' => ['index' ,'article' ,'product'], //admin 模块，有哪些控制器
        'home' => ['index' ,'article' ,'product'],  //home 模块，有哪些控制器
        'test' => ['test1' ,'test2' ,'test3'],  //test 模块，有哪些控制器
        // 更多
    ];

    /**
     * 后台菜单
     *
     * @var array
     */
    protected $menus = [
        [
            'title' => '菜单',
            'sort' => 1,
            'url' => '#',
            'icon' => 'mdi mdi-coffee',
            'children' => [
                [
                    'title' => '子菜单1',
                    'sort' => 1,
                    'url' => '/admin/extdemo/index1',
                    'icon' => 'mdi mdi-collage',
                ],
                [
                    'title' => '子菜单2',
                    'sort' => 2,
                    'url' => '/admin/extdemo/index2',
                    'icon' => 'mdi mdi-collage',
                ],
            ],
        ]
    ];
}
```

> 扩展开发支持多模块，不只是 `admin` 。

关于模块开发的更详细细节，后面会有更介绍。

扩展开发

目录结构

目录结构

这里拿 `tpextmyadmin` 做介绍

```
── assets                  (资源，不需则留空，发布时复制到`/public/assets/`)
   ├── css
   ├── images
   └── js
── data                    (安装或卸载脚本，不需要安装则留空)
   ├── install.sql
   └── uninstall.sql
── src                     (源码目录)
   ├── admin                    (admin模块)
   │   ├── controller
   │   │   ├── Admin.php
   │   │   ├── Index.php
   │   │   ├── Menu.php
   │   │   ├── Operationlog.php
   │   │   ├── Permission.php
   │   │   └── Role.php
   │   ├── model
   │   │   ├── AdminMenu.php
   │   │   ├── AdminOperationLog.php
   │   │   ├── AdminPermission.php
   │   │   ├── AdminRole.php
   │   │   ├── AdminRoleMenu.php
   │   │   ├── AdminRolePermission.php
   │   │   └── AdminUser.php
   │   └── view                 (视图模板)
   │       ├──
   │       └── ...
   └── common
   │   ├── behavior
   │   │   ├── Auth.php
   │   │   ├── Log.php
   │   │   ├── Menu.php
   │   │   └── Setup.php
   │   ├── MinifyTool.php
   │   └── Module.php           (模块定义)
   ├── common.php
   ├── config.php               (扩展自定义配置)
   └── helper.php                       (扩展加载)
── composer.json
── LICENSE.txt
── README.md
```

主要有 `admin`，`common` 两个模块，实际中可以按需要增加其他模块，如前台 `index` 模块。

扩展定义

扩展定义

/src/common/Module.php

```
namespace com\youadmin\common;

use tpext\common\Module as baseModule;

class Module extends baseModule
{
    protected $version = '1.0.1';

    protected $name = 'youname.module';//

    protected $title = '我的框架';

    protected $description = '我的框架提供了xxxx功能';

    //扩展根目录，如果是以composer方式扩展，程序代码放在src目录，则此处为/../../，如果是以extend方式开发，则为/../
    protected $root = __DIR__ . '/../../';

    protected $assets = 'assets';  //如果有静态资源，否则不用写

    //模块定义
    protected $modules = [
        'admin' => ['index' ,'article' ,'product'], //admin 模块，有哪些控制器
        'home' => ['index' ,'article' ,'product'],  //home 模块，有哪些控制器
        'test' => ['test1' ,'test2' ,'test3'],  //test 模块，有哪些控制器
        // 更多
    ];

    //后台菜单
    protected $menus =
     [
         [
            'title' => 'test',
            'url' => '#',
            'icon' => 'mdi mdi-account-card-details',
            'children' => [
                [
                    'title' => 'test1',
                    'url' => '/admin/group/index',
                    'icon' => 'mdi mdi-account-network',
                ],
                [
                    'title' => 'test2',
                    'url' => '/admin/role/index',
                    'icon' => 'mdi mdi-account-multiple',
                ],
                //...
            ],
         ],
         //...
      ];

    //安装 ，无特别需求，没必要重写
    public function install()
    {
        if (parent::install()) {
            //sql执行成功,你的逻辑
            return true;
        }
        //sql安装sql执行失败,你的逻辑
        return false;
    }

   //卸载，无特别需求，没必要重写
```

```
    public function uninstall()
    {
        if (parent::uninstall()) {
            //sql执行成功,你的逻辑
            return true;
        }

        //sql安装sql执行失败,你的逻辑
        return false;
    }
}
```

composer.json

配置范例

```
{
    "name": "youname/modulename",
    "description": "thinkphp extension",
    "type": "library",
    "keywords": [
        "thinkphp"
    ],
    "homepage": "http://your-web-site.com/",
    "license": "Apache-2.0",
    "authors": [
        {
            "name": "youname",
            "email": "youname@email.com"
        }
    ],
    "require": {
        "topthink/framework": "5.1.*",
        "ichynul/tpext": "^1.0.1",
        "ichynul/tpextbuilder": "^1.0.1"
    },
    "autoload": {
        "psr-4": {
            "tpext\\modulename\\": "src/"
        },
        "files": [
            "src/helper.php"
        ]
    }
}
```

说明

> autoload>psr-4:定义命名空间[`"tpext\\modulename\\": "src/"`]

> autoload>files:定义helper[`src/helper.php`]

扩展注册

扩展注册

[composer] 方式的扩展包

通过 `helper.php` 助手注册

```
<?php

use tpext\common\ExtLoader;

$classMap = [
    'tpext\\myadmin\\common\\Module'
];

//注册扩展，重要，不然你的扩展无法自动启动
ExtLoader::addClassMap($classMap);
```

[extend] 方式的扩展包

tpextmanager 版本在[1.6.7/3.1.9]以后的会动自扫描[extend]目录并处理，以下内容可忽略

通过监听`tpext_find_extensions`事件，然后在事件处理中使用添加;

tp5.1

在`application/tags.php`中添加监听：

```
<?php

use tpext\common\ExtLoader;

// 应用行为扩展定义文件
return [
    // 应用初始化
    'app_init'     => [],
    // 应用开始
    'app_begin'    => [],
    // 模块初始化
    'module_init'  => [],
    // 操作开始执行
    'action_begin' => [],
    // 视图内容过滤
    'view_filter'  => [],
    // 日志写入
    'log_write'    => [],
    // 应用结束
    'app_end'      => [],
    // 扩展加载
    'tpext_find_extensions' => [
        'app\\common\\behavior\\Extensions',
    ],
];
```

处理程序`application/common/behavior/Extensions.php`：

```
<?php

namespace app\common\behavior;

use tpext\common\ExtLoader;

class Extensions
{
    public function run()
    {
        /*也可在 【扩展管理－tpext综合管理－配置】中添加 */
        $classMap = [
            'extdemo\\common\\Module',
            'myadmindata\\common\\Module',
            'hplusadmin\\common\\Resource',
        ];

        ExtLoader::addClassMap($classMap);
    }
}
```

tp6.0

在`app/event.php`中添加监听：

```
<?php
// 事件定义文件
return [
    'bind'      => [
    ],

    'listen'    => [
```

```
        'AppInit'  => [],
        'HttpRun'  => [],
        'HttpEnd'  => [],
        'LogLevel' => [],
        'LogWrite' => [],
        'tpext_find_extensions' => [
            'app\\event\\Extensions',
        ],
    ],

    'subscribe' => [
    ],
];
```

处理程序 `app/event/Extensions.php` :

```
<?php
declare (strict_types = 1);

namespace app\event;

use tpext\common\ExtLoader;

class Extensions
{
    public function handle()
    {
        /*也可在 【扩展管理 – tpext综合管理 – 配置】中添加 */
        $classMap = [
            'extdemo\\common\\Module',
            //其他自定义扩展
            'myadmindata\\common\\Module',
            'hplusadmin\\common\\Resource'
        ];
        ExtLoader::addClassMap($classMap);
    }
}
```

其他方式

也可在 [tpext.manager]【扩展管理 – tpext综合管理 – 配置】中添加

tpextbuilder-UI生成

构建器-Builder

基本布局

简介

Builder提供全局组件(form/tabel/layer/tree/cotent)的创建方法

主要方法

```
//获取一个全局实例（单列模式）

//$title：页面标题,$desc：页面描述
getInstance($title = '', $desc = '');

//获取一个row
row();

//获取一个column
column($size = 12);

//获取一个表单
form($size = 12);

//获取一个表格
table($size = 12);

//获取一个工具栏
toolbar($size = 12);

//默认获取一个ZTree树
tree($size = 12);

//获取一个ZTree树
zTree($size = 12);

//获取一个JSTree树
jsTree($size = 12);

//获取一自定义内容
content($size = 12);

//获取一tab内容
tab($size = 12);

//获取一Layer实例
layer();
```

布局

页面结构： 行(row) -> 列(column) -> 组件(form/tabel/layer/tree/cotent)

```
$builder = Builder::getInstance();
```

```
$form = $builder->form(12);
//相当于：
$form = $builder->column(12)->form();
//相当于：
$form = $builder->row()->column(12)->form();
```

demo 1

```
$builder = Builder::getInstance();
$form = $builder->form(6);
$table = $builder->table(6);
```

-col-md-6-	-col-md-6-
form	table

demo 2

```
$builder = Builder::getInstance();
$form = $builder->content()->display('hello');
$form = $builder->form(6);
$table = $builder->table(6);
```

----------col-md-12----------
cntent

-col-md-6-	-col-md-6-
form	table

demo 3

```
$builder = Builder::getInstance();
$form =  $builder->form(6);
$table = $builder->row()->table(6);
```

-col-md-6-	-col-md-6-
form	
table	

demo 4

```
$builder = Builder::getInstance();
$column1 =  $builder->column(6);
$column2 = $builder->column(6);

$form = $column1->row()->column(6)->form();

$table = $column2->row()->column(6)->row()->table();
```

`$column->row()` 与 `$row->column(n)` 理论上可以不停嵌套，但实际中很少用到。

显示效果复杂，不做演示。

js、css

```
//用户添加自己的js/css 不要直接使用[$builder->AddJs()/$builder->addCss()]，这两个方法一般用于添加displayer等组件的资源，全局需要，会被minify压缩为

//在当前页面引入一个js文件
//如：
$builder->customJs('/static/admin/js/my.js');

//在当前页面引入一个css文件
//如：
$builder->customCss('/static/admin/css/my.css');

//在当前页面添加一段js脚本
//如：
$script = <<<EOT
alert('666');
EOT;
$builder->addScript($script);

//在当前页面添加一段style样式
//如：
$builder->addStyleSheet('
.row
{
    mrgin:0;
}
');

//在当前页面显示一个顶部通知消息，(仅GET请求时)
//如：
$builder->notify('您未完善个人信息');

//关闭当前layer弹出层
layer()->close($success = true, $msg = '操作成功');

//关闭当前layer弹出层，并刷新上一级页面列表
layer()->closeRefresh($success = true, $msg = '操作成功');

//关闭当前layer弹出层，并刷让上一级页面跳转到一个新的url
layer()->closeGo($success = true, $msg = '操作成功', $url);
```

请返回builder

builder

大多数情况，请返回 `$builder->render()` 或 `$builder` .

```
$builder = $this->builder('出错了');
//错误，返回的是一个`content`，只渲染了部分内容，js、css等资源都没有。不算绝对错误，但可能与想象不符。
return $builder->content()->display('<p>' . '未能读取文件:' . $fileurl . '</p>');
```

```
$builder = $this->builder('出错了');
$builder->content()->display('<p>' . '未能读取文件:' . $fileurl . '</p>');
//正确
//return $builder->render();
return $builder;
```

特殊：`layer系列`

```
return $this->builder()->layer()->closeRefresh(1, '保存成功');
return $this->builder()->layer()->closeRefresh(0, '保存失败');
return $this->builder()->layer()->close(0, '数据不存在');
```

表单-Form

form表单

支持的组件

```
/**
 * Methods.
 *
 * Field          field($name, $label = '', $colSize = 12)
 * Text           text($name, $label = '', $colSize = 12)
 * Checkbox       checkbox($name, $label = '', $colSize = 12)
 * Radio          radio($name, $label = '', $colSize = 12)
 * Button         button($type, $label = '', $colSize = 12)
 * Select         select($name, $label = '', $colSize = 12)
 * MultipleSelect multipleSelect($name, $label = '', $colSize = 12)
 * Textarea       textarea($name, $label = '', $colSize = 12)
 * Hidden         hidden($name)
 * Color          color($name, $label = '', $colSize = 12)
 * RangeSlider    rangeSlider($name, $label = '', $colSize = 12)
 * File           file($name, $label = '', $colSize = 12)
 * Image          image($name, $label = '', $colSize = 12)
 * Date           date($name, $label = '', $colSize = 12)
 * Datetime       datetime($name, $label = '', $colSize = 12)
 * Time           time($name, $label = '', $colSize = 12)
 * Year           year($name, $label = '', $colSize = 12)
 * Month          month($name, $label = '', $colSize = 12)
 * DateRange      dateRange($name, $label = '', $colSize = 12)
 * DateTimeRange  datetimeRange($name, $label = '', $colSize = 12)
 * TimeRange      timeRange($name, $label = '', $colSize = 12)
 * Number         number($name, $label = '', $colSize = 12)
 * SwitchBtn      switchBtn($name, $label = '', $colSize = 12)
 * Rate           rate($name, $label = '', $colSize = 12)
 * Divider        divider($text, $label = '', $colSize = 12)
 * Password       password($name, $label = '', $colSize = 12)
 * Decimal        decimal($name, $label = '', $colSize = 12)
 * Html           html($html, $label = '', $colSize = 12)
 * Raw            raw($name, $label = '', $colSize = 12)
 * Show           show($name, $label = '', $colSize = 12)
 * Tags           tags($name, $label = '', $colSize = 12)
 * Icon           icon($name, $label = '', $colSize = 12)
 * MultipleImage  multipleImage($name, $label = '', $colSize = 12)
 * MultipleFile   multipleFile($name, $label = '', $colSize = 12)
 * WangEditor     wangEditor($name, $label = '', $colSize = 12)
 * Tinymce        tinymce($name, $label = '', $colSize = 12)
 * UEditor        ueditor($name, $label = '', $colSize = 12)
 * WangEditor     editor($name, $label = '', $colSize = 12)
 * CKEditor       ckeditor($name, $label = '', $colSize = 12)
 * MDEditor       mdeditor($name, $label = '', $colSize = 12)
 * MDReader       mdreader($name, $label = '', $colSize = 12)
 * Match          match($name, $label = '', $colSize = 12)
 * Matches        matches($name, $label = '', $colSize = 12)
 * Fields         fields($name, $label = '', $colSize = 12)
 * Map            map($name, $label = '', $colSize = 12)
 * Items          items($name, $label = '', $colSize = 12)
 */
```

特殊组件

`fields`, `items`, `tab`, `step`

field参数说明

`$name` 字段名称 必填 ,如 `goods_name` 或 `category.name` (关联模型)

`$label` 显示label ，不填则取name值

`$cloSize` col-md-大小，默认:12

form常用方法

```
//设置字段元素的默认大小，后面创建的元素就不必一个一个去设置大小了。
$form->defaultDisplayerSize(2, 6);

//设置字段元素默认`col-md`大小
$form->defaultDisplayerColSize(12);

//只读，表单里面所有元素只读
$form->readonly(true);

//是否使用ajax提交，默认使用，一般按默即可
$form->ajax(true);

//设置表单的id，同一个页面有多个表单时可设置不同的id，避免js冲突，一般按默即可
$form->formId('form2');

//设置表单提交的url/action
$form->action(url('myaction');

//设置【提交】【重置】按钮大小
$form->butonsSizeClass('btn-xs');

//禁用自动生成底部【提交】【重置】按钮
$form->bottomButtons(false);

//手动设置按钮
$form->btnSubmit('提  交');
$form->btnReset('重  置');
$form->btnBack('返  回');
```

分组布局

建议使用 `fields` 分组，基于 `clo-size` 的分组布局不易控制，高度不同会互相影响。

假如要分成左右两列

1.基于 `co-sizel` 分:

```
$form->defaultDisplayerColSize(6);//设置默认col大小col-md-6，自动分为左右两列。

$form->text('nickname', '姓名');
$form->text('mobile', '手机号')->maxlength(11);
$form->select('department_id', '部门')->options($departList);
$form->image('avatar', '照片');//图片高度与其他的不同，容易影响布局
$form->text('school', '学校');
$form->select('nation', '民族')->options($nationList);
```

2.使用 `fields` :

```
//第一组：
$form->fields('g1', '', 6)->showLabel(false)->size(0, 12);
$form->text('nickname', '姓名');
$form->text('mobile', '手机号')->maxlength(11);
$form->select('department_id', '部门')->options($departList);
$form->fieldsEnd();

//第二组：
$form->fields('g2', '', 6)->showLabel(false)->size(0, 12);
$form->image('avatar', '照片');
$form->text('school', '学校');
$form->select('nation', '民族')->options($nationList);
$form->fieldsEnd();
```

[推荐]3.使用 `left/right()` ，方法2的升级版

```
$form->left(6)->with(function () use ($form) {
 $form->text('nickname', '姓名');
 $form->text('mobile', '手机号')->maxlength(11);
 $form->select('department_id', '部门')->options($departList);
});

$form->right(6)->with(
    $form->image('logo', '封面图')->required()->mediumSize(),
    $form->text('share_commission', '分销佣金')->default(0),
    $form->text('market_price', '市场价', 4),
    $form->text('cost_price', '成本价', 4)
);
//注意两种方式with里面的不同
```

表格-Table

table 表格

支持的组件

```
/**
 * Methods.
 *
 * Field          field($name, $label = '', $colSize = 12)
 * Text           text($name, $label = '', $colSize = 12)
 * Checkbox       checkbox($name, $label = '', $colSize = 12)
 * Radio          radio($name, $label = '', $colSize = 12)
 * Select         select($name, $label = '', $colSize = 12)
 * MultipleSelect multipleSelect($name, $label = '', $colSize = 12)
 * Textarea       textarea($name, $label = '', $colSize = 12)
 * Color          color($name, $label = '', $colSize = 12)
 * RangeSlider    rangeSlider($name, $label = '', $colSize = 12)
 * File           file($name, $label = '', $colSize = 12)
 * Image          image($name, $label = '', $colSize = 12)
 * Date           date($name, $label = '', $colSize = 12)
 * Datetime       datetime($name, $label = '', $colSize = 12)
 * Time           time($name, $label = '', $colSize = 12)
 * Year           year($name, $label = '', $colSize = 12)
 * Month          month($name, $label = '', $colSize = 12)
 * TimeRange      timeRange($name, $label = '', $colSize = 12)
 * Number         number($name, $label = '', $colSize = 12)
 * SwitchBtn      switchBtn($name, $label = '', $colSize = 12)
 * Decimal        decimal($name, $label = '', $colSize = 12)
 * Html           html($html, $label = '', $colSize = 12)
 * Raw            raw($name, $label = '', $colSize = 12)
 * Show           show($name, $label = '', $colSize = 12)
 * Tags           tags($name, $label = '', $colSize = 12)
 * Icon           icon($name, $label = '', $colSize = 12)
 * MultipleImage  multipleImage($name, $label = '', $colSize = 12)
 * MultipleFile   multipleFile($name, $label = '', $colSize = 12)
 * Match          match($name, $label = '', $colSize = 12)
 * Matches        matches($name, $label = '', $colSize = 12)
 * Fields         fields($name, $label = '', $colSize = 12)
 *
 */
```

field参数说明

`$name` 字段名称 必填

`$label` 显示label ，不填则取name值

`$cloSize` col-md-大小，暂无实际用处

- 理论上支持全部 `form` 组件，但一般来说，使用 `show`，`field`，`text`，`Checkbox`，`Radio`，`Select`，`Textarea` 等基本够用了。
- `show` 和 `field` 纯显示， `text`，`Checkbox` 等表单元素支持在表格行内修改并失去焦点自动提交【配合autoPost】。

基本使用

```
    $table->show('id', 'ID');
    $table->show('username', '登录帐号');
    $table->text('name', '姓名')->autoPost()->getWrapper()->addStyle('max-width:80px');
    $table->show('role_name', '角色');
    $table->show('email', '电子邮箱')->default('无');
    $table->show('phone', '手机号')->default('无');
    $table->show('errors', '登录失败');
    //多字段组合使用`fields`
    $table->fields('times', '添加/更新时间')->with(
```

```
        $table->show('create_time', '添加时间'),
        $table->show('update_time', '修改时间')
    )->getWrapper()->addStyle('width:180px');
```

常用

- `show()` 显示
- `raw()` 显示带html的内容

支持部分的form组件行内编辑

`text` , `textarea` , `radio` , `select` , `checkbox`

行内编辑续配合 `autoPost($url)` 方法使用， `url` 参数不传则默认请求到同一个控制器的[`autoPost`]action

Toolbar工具栏

> 默认自动生成 [添加 / 批量删除 / 刷新] 即

```
btnAdd() / btnDelete() / btnRefresh()
```

完整实列

```
$table->getToolbar()
    ->btnAdd()
    ->btnEnable()
    ->btnDisable()
    ->btnDelete()
    ->btnExport()
    ->btnExports(['xlxs'=>'xlsx','xls'=>'xls'])
    ->btnRefresh();
```

使用 **dropdown actions**

```
$table->getToolbar()
    ->btnAdd()
    ->btnActions(
        [
            'enable' => ['url' => url('enable', ['state' => 1]) , 'label' => '启用'],
            'disable' => ['url' => url('enable', ['state' => 0]), 'label' => '禁用'],
            'delete' => '删除',
        ]
    )
    ->btnExport()
    ->btnExports(['xlxs'=>'xlsx','xls'=>'xls'])
    ->btnRefresh();
```

禁用工具栏

```
$table->useToolbar(false);
```

手动设置

```

//添加
btnAdd($url = '', $label = '添加', $class = 'btn-primary', $icon = 'mdi-plus', $attr = '');

//批量删除
```

```
btnDelete($postUrl = '', $label = '删除', $class = 'btn-danger', $icon = 'mdi-delete', $confirm = true, $attr = '');

//刷新
btnRefresh($label = '', $class = 'btn-cyan', $icon = 'mdi-refresh', $attr = 'title="刷新"');

//启用
btnEnable($postUrl = '', $label = '启用', $class = 'btn-success', $icon = 'mdi-check', $confirm = true, $attr = '');

//禁用
btnDisable($postUrl = '', $label = '禁用', $class = 'btn-warning', $icon = 'mdi-block-helper', $confirm = true, $attr = '');

//导入
btnImport($afterSuccessUrl, $acceptedExts = "rar,zip,doc,docx,xls,xlsx,ppt,pptx,pdf", $fileSize = '20', $label = '导入', $class = 'btn-pink', $icon =

//导出（默认，点击按钮直接请求后台）
btnExport($postUrl = '', $label = '导出', $class = 'btn-pink', $icon = 'mdi-export', $attr = 'title="导出"')

//导出（可选，点击弹出菜单，选择导出类型）
btnExports($items, $postUrl = '', $label = '导出', $class = 'btn-secondary', $icon = 'mdi-export', $attr = 'title="导出"')
```

> 其他，如果上面的不够用，你可以添加自定义按钮

```
//添加一个链接，打开$ulr
btnLink($url, $label = '', $class = 'btn-secondary', $icon = 'mdi-checkbox-marked-outline', $attr = '')

//添加一个批量操作，自动附带多选框选中的id发送post请求到`$url`，`$confirm` 批量操作前是否显示确认提示框。
btnPostChecked($url, $label = '', $class = 'btn-secondary', $icon = 'mdi-checkbox-marked-outline', $attr = '', $confirm = true)
//已选中多个id参数获取
//$ids = input('post.ids');

//添加一个批量打开，自动附带多选框选中的id发送get请求到`$url`。一般用于批量分配，移动等功能。
btnOpenChecked($url, $label = '', $class = 'btn-secondary', $icon = 'mdi-checkbox-marked-outline', $attr = '')

//已选中多个id参数获取
//$ids = input('get.ids');
```

Actionbar动作栏

> 默认自动生成 [编辑 / 删除] 即

```
btnEdit() / btnDelete()
```

- 基本使用

```
    $table->getActionbar()
          ->btnEdit()
          ->btnEnable()
          ->br()//换行
          ->btnDisable()
          ->btnDelete()
          ->mapClass([
             'delete' => ['hidden' => '__h_del__'],
             'enable' => ['hidden' => '__h_en__'],
             'disable' => ['hidden' => '__h_dis__'],
          ]);
```

使用 **dropdown actions**

```
$table->getActionbar()
   ->btnEdit()
   ->btnActions(
      [
         'enable' => ['url' => url('enable', ['state' => 1]), 'label' => '启用'],
         'disable' => ['url' => url('enable', ['state' => 0]), 'label' => '禁用'],
         'delete' => '删除',
         'view' => ['url' => url('view', ['id' => '__dat.pk__']), 'label' => '查看','confirm' => '2'],
      ]
   )
   ->mapClass([
      'delete' => ['hidden' => '__h_del__'],
      'enable' => ['hidden' => '__h_en__'],
      'disable' => ['hidden' => '__h_dis__'],
   ]);
```

> 禁用

```
$table->useActionbar(false);
```

> 手动设置

```
//编辑
btnEdit($url = '', $label = '', $class = 'btn-primary', $icon = 'mdi-lead-pencil', $attr = 'title="编辑"')

//删除
btnDelete($postUrl = '', $label = '', $class = 'btn-danger', $icon = 'mdi-delete', $confirm = true, $attr = 'title="删除"')

//查看
btnView($url = '', $label = '', $class = 'btn-primary', $icon = 'mdi-lead-pencil', $attr = 'title="查看"')

//禁用
btnDisable($postUrl = '', $label = '', $class = 'btn-warning', $icon = 'mdi-block-helper', $confirm = true, $attr = 'title="禁用"')

//启用
btnEnable($postUrl = '', $label = '', $class = 'btn-success', $icon = 'mdi-check', $confirm = true, $attr = 'title="启用"')
```

> 其他，如果上面的不够用，你可以自己添加自定义按钮

```

//添加一个链接，打开$ulr
```

```
btnLink($name = '', $url, $label = '', $class = 'btn-secondary', $icon = '', $attr = '')
```

- `url('demo', ['id' => '__data.pk__']);`
- 相当于 `url('demo', ['id'=>'__data.id__']);` ，因为`pk`代表当前的主键。
- 其他参数：`__data.字段名__`

  如：`url('demo', ['id' => '__data.id__', 'type' => '__data.type__', 'status' => 1]);`

```
//添加一个操作，自动附带当前列id参数post到`$postUrl`，`$confirm` 操作前是否显示确认提示框。
btnPostRowid($name = '', $postUrl, $label = '', $class = 'btn-secondary', $icon = 'mdi-checkbox-marked-outline', $attr = '', $confirm = true)
```

> 控制`action`的显示禁用

```
$table->getActionbar()
    ->btnEdit()
    ->btnEnableAndDisable()
    ->btnView()
    ->btnDelete()
    ->btnPostRowid('clear_errors', url('clearErrors'), '', 'btn-info', 'mdi-backup-restore', 'title="重置登录失败次数"')
    ->mapClass([
        'delete' => ['hidden' => '__h_del__'],//当这行数据的`__h_del__`字段值为真(1或true)时，`enable`这个按钮加上`hidden`的class
        'enable' => ['hidden' => '__h_en__'],
        'disable' => ['hidden' => '__h_dis__'],
        //也可以像下面这样,就不用去循环数组设置。
        //'enable' => ['hidden' => [1, 'enable']],//,当这行数据的`enable`字段值为`1`时，`enable`这个按钮加上`hidden`的class
        //'disable' => ['hidden' => [[0,2], 'enable']],//当这行数据的`enable`字段值为`0`或`2`时，`disable`这个按钮加上`hidden`的class
        //'disable' => ['hidden' => [2, 'enable', '>']],//当这行数据的`enable`字段值大于`2`时，`disable`这个按钮加上`hidden`的class
        //[2, 'enable', '>'] 中 `>` 为比较逻辑
        // 更多逻辑：in_array|not_in_array|eq|gt|lt|egt|elt|strstr|not_strstr
        // eq|gt|lt|egt|elt等价于 =|>|<|>=|<=
        // !in_array 等价于not_in_array,!strstr等价于not_strstr

        // 'enable' => ['hidden' => function ($data) { //闭包，返回真则加上对应的
        //     return $data['enable'] == 1;
        // }],
    ]);
//循环数组去设置
foreach ($data as &$d) {
    $d['__h_del__'] = $d['id'] == 1;
    $d['__h_en__'] = $d['enable'] == 1;
    $d['__h_dis__'] = $d['enable'] != 1 || $d['id'] == 1;
}
unset($d);
```

- `delete|enable|disable` 按钮名称，如果是自定义[btnLink/btnPostRowid]则为传入的`$name`.
- '`hidden`' => '`__h_del__`'，当这一条记录的`__h_del__`值为真时，这个action会加上`hidden`这个class
- 同理，可以加上`disabled`

```
'enable' => ['disabled' => '__dis_en__'],
```

ActionBtn的**label**支持使用变量

如`评论({comments_count})` => `评论(12)`

模型定义一个获取器：

```
class ShopGoods extends Model
{
    public function getCommentCountAttr($value, $data)
    {
        return ShopGoodsComment::where(['goods_id' => $data['id']])->count();
    }
```

```
}
```

```
$table->getActionbar()
    ->btnEdit()
    ->btnLink('comments', url('/admin/shopgoodscomment/index', ['goods_id' => '__data.pk__']), '评论{comments_count}', 'btn-warning', 'mdi-com
```

addTop / addBottom ,顶部或底部内容

```
$table->addTop()->content()->fetch('demo', ['name' => '小明']);

$table->addBottom()->content()->display('我的名字叫{name}', ['name' => '小明']);
```

搜索-Search

其实就是`Form`，把某些不适合使用的组件隐藏了

```
/**
 * Methods.
 *
 * Text          text($name, $label = '', $colSize = 2, $filter = '')
 * Checkbox      checkbox($name, $label = '', $colSize = 2, $filter = '')
 * Radio         radio($name, $label = '', $colSize = 2, $filter = '')
 * Button        button($type, $label = '', $colSize = 2, $filter = '')
 * Select        select($name, $label = '', $colSize = 2, $filter = '')
 * MultipleSelect multipleSelect($name, $label = '', $colSize = 2, $filter = '')
 * Textarea      textarea($name, $label = '', $colSize = 2, $filter = '')
 * Hidden        hidden($name)
 * Color         color($name, $label = '', $colSize = 2, $filter = '')
 * RangeSlider   rangeSlider($name, $label = '', $colSize = 2, $filter = '')
 * Date          date($name, $label = '', $colSize = 2, $filter = '')
 * Datetime      datetime($name, $label = '', $colSize = 2, $filter = '')
 * Time          time($name, $label = '', $colSize = 2, $filter = '')
 * Year          year($name, $label = '', $colSize = 2, $filter = '')
 * Month         month($name, $label = '', $colSize = 2, $filter = '')
 * DateRange     dateRange($name, $label = '', $colSize = 2, $filter = '')
 * DateTimeRange  datetimeRange($name, $label = '', $colSize = 2, $filter = '')
 * TimeRange     timeRange($name, $label = '', $colSize = 2, $filter = '')
 * Number        number($name, $label = '', $colSize = 2, $filter = '')
 * SwitchBtn     switchBtn($name, $label = '', $colSize = 2, $filter = '')
 * Rate          rate($name, $label = '', $colSize = 2, $filter = '')
 * Divider       divider($text, $label = '', $colSize = 2, $filter = '')
 * Decimal       decimal($name, $label = '', $colSize = 2, $filter = '')
 * Tags          tags($name, $label = '', $colSize = 2, $filter = '')
 * Icon          icon($name, $label = '', $colSize = 2, $filter = '')
 * Fields        fields($name, $label = '', $colSize = 12, $filter = '')
 */
```

field参数说明

> $name 字段名称 必填

> $label 显示label ，不填则取name值

> $cloSize col-md-大小，默认:2

> $filter 搜索条件，默认 'eq'

```
###### search常用方法
```php
//设置字段元素的默认大小，后面创建的元素就不必一个一个去设置大小了。
$search->defaultDisplayerSize(4, 8);

//设置字段元素默认`col-md`大小
$search->defaultDisplayerColSize(2);
```

`$search`相当于一个`$form`，是`$table`的一部分。

protected function filterWhere() { //根据提交数据返回搜索条件，此方法可以不手动重写，会自动生成搜索条件，没怎么测试过，所以还是推荐手写。 }

```
protected function buildSearch()
{
    //$search = $table->getSearch();//获取一个搜索

    $search = $this->search;

    //页面顶部快速切换：tabLink。

    $search->tabLink('is_onsasle')->options([1 => '已上架', 2 => '未上架']);
```
```

```
    $search->hidden('is_onsasle');//用一个隐藏字段接收切换的值，字段的名称要和上面tabLink的一样。

    //$search->select('is_onsasle', '上架')->options([1 => '已上架', 2 => '未上架']);//或者用一个select或radio也行。

    //其他
    $search->text('kwd', '名称/spu', 3)->maxlength(20);

    $search->select('category_id', '分类', 3)->dataUrl(url('/admin/shopcategory/selectPage'), 'name');

    $search->select('brand_id', '品牌', 3)->dataUrl(url('/admin/shopbrand/selectPage'));
}
```

addTop / addBottom ,顶部或底部内容

```
$search->addTop()->content()->fetch('demo');

$search->addBottom()->content()->display('{name}', ['name' => 'jim']);
```

组件-Displayers

Field

Field

所有 **displayer** 的基类

直接输出value(支持html)，一般不直接使用。

```
{$value|raw}
```

`form` 中使用：无 `label`，不支持 `col-md-n` 大小控制。

主要通用方法

方法	说明	备注
`class($val)`	设置field的class	
`labelClass($val)`	设置label的class	
`attr($val)`	设置field的属性	
`addClass($val)`	添加field的class	
`addAttr($val)`	添加field的属性	
`addStyle($val)`	添加field的style	
`labelAttr($val)`	添加label的属性	
`size($label, $elemetm)`	设置大小(label,field)	默认: 2, 8
`help($text)`	添加帮助信息	
`readonly($val =true)`	只读	
`disabled($val =true)`	禁用	
`required($val =true)`	必填	主要是前端js验证，不涉及后端
`showLabel($val =true)`	是否显示label	
`default($val)`	默认值	
`value($val)`	设置值	
`to($tpl)`	简单的转换	
`mapClass()`	样式匹配	
`mapClassGroup($GroupArr)`	样式组匹配	
`rendering($callback)`	渲染前的回调	

> to 支持模板变量：{val} 或 {其他字段名}

如

```
$table->show('name','姓名')->to('{val}#{mobile}');//{val}代表当前字段`name`值，{mobile}为这条记录中的`mobile`字段值。

$table->raw('link','链接')->to('<a href="{val}">{val}</a>');//渲染html要用`raw`或`field`
```

也可使用闭包：

```
$table->show('name', '姓名')->to(function ($value, $data) {
    return $value .'#'. $data['mobile'];//$value为当前字段`name`的值，若要使用其他字段，就使用`$data['field']`
});
```

> mapClass

`mapClass($values, $class, $field = '', $logic = 'in_array')` 样式匹配

`mapClass(function closure(){...}, $class)` 样式匹配，使用闭包

```
$table->match('status','状态')->mapClass(function ($value, $data) {
    return $value == 1;//$value为当前字段`status`的值，若要使用其他字段，就使用`$data['field']`
}, 'success');
```

> mapClassGroup

`mapClassGroup([[$values1, $class1, $field1 = '', $logic1 = 'in_array'], [$values2, $class2, $field2 = '', $logic2 = 'in_array']]])` 批量样式匹配

`mapClassGroup(['class1' => [$values1, $field1, $logic1], 'class2'=> [$values2, $field2, $logic2], ... ])` 批量样式匹配， `class` 作为数组键

`mapClassGroup(['class1' => function closure1(){...}, 'class2'=> function closure2(){...}, ... ])` 批量样式匹配，使用闭包

```
$table->match('status','状态')->mapClassGroup([
    'success' => function ($value, $data) {
        return $value == 1;//$value为当前字段`status`的值，若要使用其他字段，就使用`$data['field']`
    },
    'danger' => function ($value, $data) {
        return $value == 0;
    }
]);
```

如

```
$table->match('open', '状态')->options(['0' => '关闭', '1' => '开启'])->mapClass(1, 'hidden');

$table->match('pay_status', '支付状态')
    ->options(['0' => '未支付', '1' => '已支付', '2' =>'已关闭'])
    ->mapClassGroup([[1, 'success'], [2, 'danger']]);
```

css 样式：

```
span.the-field.default {
    color: #8b95a5;
}
span.the-field.primary {
    color: #33cabb;
}
span.the-field.success {
    color: #72b754;
}
span.the-field.info {
    color: #48b0f7;
}
span.the-field.warning {
    color: #faa64b;
}
span.the-field.danger {
    color: #f96868;
}
span.the-field.dark {
    color: #465161;
}
span.the-field.secondary {
    color: #e4e7ea;
}
span.the-field.purple {
    color: #926dde;
}
```

```
span.the-field.pink {
    color: #f96197;
}
span.the-field.cyan {
    color: #57c7d4;
}
span.the-field.yellow {
    color: #fcc525;
}
```

> rendering

渲染前的回调，非常具有灵活性

```
//实现类似于mapClass的效果,如果表格的某一列数据的姓名是[小明]那么把手机号替换成****，并且加上[danger]样式。
$table->show('mobile', '手机')->rendering(function ($field) {
    if ($field->data['name'] == '小明') {
        $field->addClass('danger');
        $field->data['mobile'] = '****';
    }
});

$table->show('mobile', '手机')->mapClass('小明', 'danger', 'name');
```

Button

按钮

一般不需要手动调用。

用于`form`或`search`中的[提交]、[重置]等按钮

HasOptions [trait]

HasOptions

`HasOptions` trait 为 `Checkbox` `Radio` `Select` `MultipleSelect` `DualListbox` `Match` `Matches` 共有。

```
//选项,传入数组  如 [1=>'男',  2=>'女'];
public function options($options){}

//键值一样的选项,传入数组  如 ['男', '女']
public function texts($texts){}

//传入查询结果集  textField  为表中可作为显示文本的字段，idField  为表中可作为key的字段
public function optionsData($optionsData, $textField = '', $idField = 'id'){}

/*以下为3个方法为增加/合并选项操作，在上面3个方法设置了选项以后再使用*/

//在现有选项【前面】加入选项
public function beforOptions($options){}

//在现有选项【后面】加入选项
public function afterOptions($options){}

//与现有选项合并，会重排数组键
public function mergeOptions($options){}
```

optionsData使用说明

数据库表: tp_gender_type ,模型 \app\common\model\GenderType;

id	name	key
1	男	m
2	女	f
3	未知	n

```
use \app\common\model\GenderType;
```

```
//指定text/key字段
$form->select('gender','性别')->optionsData(GenderType::select(), 'name', 'key');
```

```
<select name="gender">
<option vlaue="m">男</option>
<option vlaue="f">女</option>
<option vlaue="n">未知</option>
</select>
```

//指定text，主键id作为key:

```
$form->select('gender','性别')->optionsData(GenderType::select(), 'name');
```

```
<select name="gender">
<option vlaue="1">男</option>
<option vlaue="2">女</option>
<option vlaue="3">未知</option>
</select>
```

注意：`optionsData` 的驱动表尽量结果集较少的表，对于数据较多的表应使用其他替代方法。

如表单中：

```
$form->select('type_id','类型')->optionsData(Type::select(), 'name', 'id');//Type表数据较少，可以使用
//$form->select('member_id','会员')->optionsData(Member::select(), 'nicknname', 'id');//不推荐，会员数量很多，全部查询出来不划算
$form->select('member_id','会员')->dataUrl(url('/admin/member/selectPage')); //推荐，使用ajax加载
```

如表格中：

```
$table->match('type_id','类型')->optionsData(Type::select(), 'name', 'id');//Type表数据较少，可以使用
//$table->match('member_id','会员')->optionsData(Member::select(), 'nicknname', 'id');//不推荐
$table->show('member.nickname','会员'); //推荐，关联模型
```

Checkbox

多选框

主要方法

```
//多个选项框是否在同一行
public function inline($val = true)

//【全选】按钮文字，若传入空字符串，则不显示此按钮
public function checkallBtn($val = '全选')

//以块状显示，一种美化
public function blockStyle($val = true)
```

`HasOptions` trait 为 `Checkbox` `Radio` `Select` `DualListbox` `MultipleSelect` `Match` `Matches` 共有。

颜色主题

```
$form->checkbox('attr', '属性')
    ->adddClass('checkbox-warning')//颜色主题
    ->blockStyle()//显示为块状
    ->checkallBtn()//显示[全选]按钮
    ->options(['is_recommend' => '推荐', 'is_hot' => '热门', 'is_top' => '置顶']);
```

`checkbox-warning`，其中[warning]为颜色主题，所有可选主题如下：

- primary（主色）
- success（成功）
- secondary（灰色）
- info（一般信息）
- warning（警告）
- danger（警告）
- dark（黑色）
- purple（紫色）
- pink（粉红色）
- cyan（青色）
- yellow（黄色）
- brown（棕色）

CKEditor

CKEditor 富文本编辑器

未内置资源，若要使用，请单独安装

```
composer require ichynul/builder-ckeditor
```

Color

颜色选择器

主要方法

```
//设置颜色代码格式，可选格式为下面几个方法的名称。
public function format($val){}

//设置颜色代码格式:rgb
public function rgb(){}

//设置颜色代码格式:rgba
public function rgba(){}

//设置颜色代码格式:hsl
public function hsl(){}

//设置颜色代码格式:hsla
public function hsla(){}

//设置颜色代码格式:hex
public function hex(){}
```

Date

日期选择

主要方法

```
//修改格式，默认为YYYY-MM-DD
public function format($val){}

//时间戳格式化，若value值为时间戳数字格式
public function timespan($val = 'Y-m-d'){}
```

只选择到月份：

```
$form->date('date', '日期')->format('YYYY-MM');
```

选择到月份，号数固定为1号：

```
$form->date('date', '日期')->format('YYYY-MM-01');
```

DateRange

日期区间选择

主要方法

```
//修改格式，默认为yyyy-mm-dd
public function format($val){}

//时间戳格式化，若value值为时间戳数字格式
public function timespan($val = 'Y-m-d'){}

//设置分隔符
public function separator($val = ','){}
```

DateTime

日期时间选择

主要方法

```
//修改格式，默认为YYYY-MM-DD HH:mm:ss
public function format($val){}

//时间戳格式化，若value值为时间戳数字格式
public function timespan($val = 'Y-m-d H:i:s'){}
```

DateTimeRange

日期时间区间选择

Divider

分割线

主要方法

```
$form->divider('分隔线');
```

效果类似：

----------分割线----------

DualListbox

左右穿梭多选

`HasOptions` trait 为 `Checkbox` `Radio` `Select` `DualListbox` `MultipleSelect` `Match` `Matches` 共有。

https://github.com/istvan-ujjmeszaros/bootstrap-duallistbox

Fields

Fields

多字段组合

多个组件组合为一个，多用于排版布局。

主要方法：

```
//with 包含多个其他组件
public function with(...$fields){}

//设置　内部组件的值(数组)　同: fill,overWrite=ture
public function value($val){}

//设置　内部组件的值(数组)，overWrite是否覆盖.
public function fill($data = [], $overWrite = false){}
```

with用法

有4种写法:

-
  - 使用with可变参数(...fields);

```
$form->fields('', '基本信息', 7)->with(
   $form->text('name', '名称')->required()->maxlength(55),
   $form->text('spu', 'spu码')->maxlength(100)
   //其他组件以,分割
);
```

-
  - 使用with包含数组中的fields;

```
$form->fields('', '基本信息', 7)->with([
   $form->text('name', '名称')->required()->maxlength(55),
   $form->text('spu', 'spu码')->maxlength(100)
   //其他组件以,分割
 ]
);
```

-
  - [版本>=1.9.0010]使用with匿名方法;

```
$form->fields('', '基本信息', 7)->with(
   function() use($from){
      $form->text('name', '名称')->required()->maxlength(55);
      $form->text('spu', 'spu码')->maxlength(100;
      //其他组件以
   }
);
//或者
$form->fields('', '基本信息', 7)->with(
   function(\tpext\builder\common\Form $from){
      $form->text('name', '名称')->required()->maxlength(55);
      $form->text('spu', 'spu码')->maxlength(100;
      //其他组件以
   }
);
```

-
  - 使用[fieldsEnd]收尾;

```
$form->fields('', '基本信息', 7);
//写包含的组件
$form->text('name', '名称')->required()->maxlength(55);
$form->text('spu', 'spu码')->maxlength(100);
//其他组件

$form->fieldsEnd();//结束
```

主要功能

-
  - : 页面布局，左边 `col-md-7` + 右边 `col-md-5` ,把页码整体做一个基本的划分。

```
$form->fields('left', '', 7)->size(0, 12)->showLabel(false);
$form->text('name', '名称')->required()->maxlength(55);
$form->text('spu', 'spu码')->maxlength(100);
$form->tags('keyword', '关键字')->maxlength(255);
$form->textarea('description', '摘要')->maxlength(255);
$form->wangEditor('content', '产品详情')->required();
$form->fieldsEnd();

$form->fields('right', '', 5)->size(0, 12)->showLabel(false);
$form->image('logo', '封面图')->required()->mediumSize();
$form->text('market_sale', '市场价', 4);
$form->text('cost_price', '成本价', 4);
$form->number('sort', '排序')->default(0);
$form->number('weight', '重量')->default(1000)->help('单位:克');
$form->fieldsEnd();
```

也可使用3个针对[fields]个语法糖： `left` , `middle` , `right` 。

`$form->left(string $size[, Closure $call]);` `$form->middle(string $size[, Closure $call]);` `$form->right(string $size[, Closure $call]);`

如：把表单分为左右两部分，每边各占一半[col-md-6]:

```
$form->left(6, function () use ($form) {
    $form->text('name', '名称')->required()->maxlength(55);
    $form->select('category_id', '分类')->required()->dataUrl(url('/admin/shopcategory/selectPage'));
    $form->select('brand_id', '品牌')->dataUrl(url('/admin/shopbrand/selectPage'));
    $form->select('admin_group_id', '商家')->dataUrl(url('/admin/group/selectPage'));
});

// 同理，如果不想使用use($form)，可以在方法传参并声明类型
// $form->left(6, function (\tpext\builder\common\Form $from) {
//     $form->text('name', '名称')->required()->maxlength(55);
//     $form->select('category_id', '分类')->required()->dataUrl(url('/admin/shopcategory/selectPage'));
//     $form->select('brand_id', '品牌')->dataUrl(url('/admin/shopbrand/selectPage'));
//     $form->select('admin_group_id', '商家')->dataUrl(url('/admin/group/selectPage'));
// });

//右边，这里演示另外一种形式，如果第二个方法不传入匿名方法，可以继续调用with方法再传入字段，with方法的用法和fields的with方法一致。
$form->right(6)->with(
    $form->image('logo', '封面图')->required()->mediumSize(),
    $form->text('share_commission', '分销佣金')->default(0),
    $form->text('market_price', '市场价', 4),
    $form->text('cost_price', '成本价', 4)
);
```

-
  - : 把一些相关的字段组合到一起.

```
$form->fields('省/市/区');
$form->select('province', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/province'), 'ext_name')->withNext(
    $form->select('city', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/city'), 'ext_name')->withNext(
        $form->select('area', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/area'), 'ext_name'))
);
```

```
$form->fieldsEnd();
```

- 功能3：`table`中使用，把相关字段合并到同一列，避免字段过多时表格显示不便。

```
$table->fields('consignee', '收货人/电话')->with(
    $table->show('consignee', '收货人'),
    $table->show('mobile', '电话')->default('--')
);

$table->show('pay_money', '支付金额');

$table->fields('pay_status', '支付状态/时间')->with(
    $table->match('pay_status', '支付状态')->options(OrderModel::$pay_status_types),
    $table->show('pay_time', '支付时间')->default('--')
);
```

显示：

收货人/电话	支付金额	支付状态/时间
小明 13612345678	100.00	已支付 2020-08-29 12:31:24

ps：`table`中使用 `fields`组合多个字段时，表头只显示`fields`的label，但里面的各个字段还是需要单独设置其label，因为导出数据时，字段是分开的。

File

文件上传

继承：`MultipleFile`

Hidden

隐藏字段

```
$form->hidden('cid')->value(1);
```

Html

输出Html

```
{$value|raw}
```

和 `raw` 类似，但多了两个方法，支持渲染文本或渲染模板

主要方法

```
//渲染一个模板文件，和thinkphp框架类似`Controller`,模板存放位置也遵循
public function fetch($template = '', $vars = []){}

//渲染一个字符串(模板)
public function display($content = '', $vars = []){}
```

实例

```
$form->html('tree', 'tree');//填充数据里面有`tree`这个字段

$form->html('tree', 'tree')->vale($treeHtml);//直接显示一段html

$form->html('tree', 'tree')->fetch('tree',  ['data' => $data]);//渲染模板

$form->html('tree', 'tree')->display('<p>{$name}</p>',  ['name' => '小明']);//渲染带变量的字符串
```

Icon

iconfont字符图标选择器

Image

图片文件上传

继承：File

Items

Items

数据条目

一般用于一对多的数据录入。

主要方法：

```
//包含多个组件
public function with(...$fields){}

//设置内容组件的值(数组),同fill,overWrite=true
public function value($val){}

//设置内容组件的值(数组),overWrite是否覆盖。数据结构实例:  [1 => ['id'=>1, 'name' => '小明'], 2 => ['id'=>2, 'name' => '小红']]
public function fill($data = [], $overWrite = false){}

//条目是否可[删除]
public function cnaDelete($val){}

//条目是否可[添加]
public function canAdd($val){}

//条目只读，不可[删除]或[添加]
public function canNotAddOrDelete(){}

//把数据库结果集转换格式，然后调用fill
//[['id'=>5, 'name' => '小刚'],['id'=>6, 'name' => '小芳']]
//   转换为
//[5 => ['id'=>5, 'name' => '小刚'], 6 => ['id'=>6, 'name' => '小芳']]
public function dataWithId($data, $idField = 'id', $overWrite = false)
```

with用法

有4种写法:

-
  - 使用with可变参数(fields);

```
$form->items('', '产品属性')->dataWithId($attrList)->with(
    $form->show('action_note', '操作备注'),
    $form->show('status_desc', '描述')
    //其他组件以,分割
);
```

-
  - 使用with包含数组中的fields;

```
$form->items('', '产品属性')->dataWithId($attrList)->with([
    $form->show('action_note', '操作备注'),
    $form->show('status_desc', '描述'),
    //其他组件以,分割
 ]
);
```

-
  - [版本>=1.9.0010]使用with匿名方法(function(){});

```
$form->items('', '产品属性')->dataWithId($attrList)->with(
    function() use($from){
        $form->show('action_note', '操作备注');
        $form->show('status_desc', '描述');
```

```
        //其他组件以
    }
);
//或者
$form->items('', '产品属性')->dataWithId($attrList)->with(
    function(\tpext\builder\common\Form $from){
        $form->show('action_note', '操作备注');
        $form->show('status_desc', '描述');
        //其他组件以
    }
);
```

-
    - 使用[fieldsEnd]收尾;

```
$form->items('', '产品属性')->dataWithId($attrList);
//写包含的组件
$form->show('action_note', '操作备注');
$form->show('status_desc', '描述');
//其他组件

$form->itemsEnd();//结束
```

主要功能

-
    - 数据录入

```
$attrList = $isEdit ? ShopGoodsAttr::where(['goods_id' => $data['id']])->order('sort')->select() : [];

$form->items('attr_list', '产品属性')->dataWithId($attrList)->with(
    $form->text('name', '名称')->placeholder('属性名称，如生产日期')->maxlength(55)->required()->getWrapper()->addStyle('width:200px;'),
    $form->text('sort', '排序')->placeholder('规格名称，如颜色')->default(1)->required()->getWrapper()->addStyle('width:80px;'),
    $form->text('value', '属性值')->required()->getWrapper()->addStyle('min-width:70%;')
)->help('【属性】不影响价格，仅展示');
```

-
    - 数据展示，针对数据量比较少的情况，比如在订单详情页显示订单日志，日志数量一般不会太多，把相关日志一次性显示为列表。

```
$logList = model\ShopOrderAction::where(['order_id' => $data['id']])->order('id desc')->select();

$form->items('log_list', '操作日志')->dataWithId($logList)->with(
    $form->show('action_note', '操作备注'),
    $form->show('status_desc', '描述'),
    $form->match('order_status', '订单状态')->options(OrderModel::$order_status_types),
    $form->match('pay_status', '支付状态')->options(OrderModel::$pay_status_types),
    $form->match('shipping_status', '物流状态')->options(OrderModel::$shipping_status_types),
    $form->show('create_time', '时间')
)->canNotAddOrDelete();
```

也可使用针对[items]个语法糖： `logs` 。

```
$form->logs('log_list', $logList, function () use ($form) {
    $form->show('action_note', '操作备注');
    $form->show('status_desc', '描述');
    $form->match('order_status', '订单状态')->options(OrderModel::$order_status_types);
    $form->match('pay_status', '支付状态')->options(OrderModel::$pay_status_types);
    $form->match('shipping_status', '物流状态')->options(OrderModel::$shipping_status_types);
    $form->show('create_time', '时间2');
});
```

Load

远程加载(单个值)

相当于只读且带ajax的`select`。

数据库表: tp_hobby_type

id	name	hoby
1	唱歌	sing
2	跳舞	dance
3	爬山	climb
4	游泳	swim

假控制器`/admin/hobytype`实现了下拉选择数据方法`selectPage`，load可以复用此接口

```
$form->load('hobbies','爱好')->dataUrl(url('/admin/hobytype/selectPage'), 'name')->value(1);
//ajax加载并输出 ：唱歌
```

一般情况下，用户无需主动使用，在form的view[只读]模式中，带ajax的select自动替换为load

Loads

远程加载(多个值)

相当于只读且带ajax的 `multipleSelect` 。

数据库表: tp_hobby_type

id	name	hoby
1	唱歌	sing
2	跳舞	dance
3	爬山	climb
4	游泳	swim

假控制器 `/admin/hobytype` 实现了下拉选择数据方法 `selectPage` ，loads可以复用此接口

```
$form->loads('hobbies','爱好')->dataUrl(url('/admin/hobytype/selectPage'), 'name')->value('1,3,4');
//ajax加载并输出 ：唱歌、爬山、游泳
```

一般情况下，用户无需主动使用，在form的view[只读]模式中，带ajax的multipleSelect自动替换为loads

Map

地图

支持高德、百度、腾讯等 使用前需要配置地图key

主要方法

```
//使用高德，默认
public function amap(){}

//使用百度
public function baidu(){}

//使用谷歌，国内不可用
public function google(){}

//使用腾讯
public function tcent(){}

//使用yandex
public function yandex(){}
```

Match

匹配

`HasOptions` trait 为 `Checkbox` `Radio` `Select` `DualListbox` `MultipleSelect` `Match` `Matches` 共有。

相当于只读的 `radio`。

```
$table->match('type', '类型')->options([
    1 => '<label class="label label-success">增加</label>',
    2 => '<label class="label label-danger">支出</label>'
])->value(1);

//输出 ：增加
```

```
$table->match('type', '类型')->options([
    1 => '男',
    2 => '女',
    3 => '未知'
])->value(2);

//输出 ：女
```

```
use \app\common\model\GenderType;
```

数据库表: tp_gender_type ,模型 \app\common\model\GenderType;

id	name	key
1	男	m
2	女	f
3	未知	n

```
//指定text字段
$table->match('gender','性别')->optionsData(GenderType::select(), 'name')->value(3);//默认主键`id`作为key
//输出 ：未知
```

```
//指定text/key字段
$table->match('gender','性别')->optionsData(GenderType::select(), 'name', 'key')->value('m');
//输出 ：男
```

Matches

多个值匹配

`HasOptions` trait 为 `Checkbox` `Radio` `Select` `DualListbox` `MultipleSelect` `Match` `Matches` 共有。

相当于只读的 `checkbox` 。

```
$table->matches('hobbies', '爱好')->options([
    1 => '唱歌',
    2 => '跳舞',
    3 => '爬山',
    4 => '游泳'
])->value('2,4');

//输出 ：跳舞、游泳
```

数据库表: tp_hobby_type ,模型 \app\common\model\HobbyType;

id	name	hoby
1	唱歌	sing
2	跳舞	dance
3	爬山	climb
4	游泳	swim

```
use \app\common\model\HobbyType;
```

```
//指定text字段
$table->matches('hobbies','爱好')->optionsData(HobbyType::select(), 'name')->value('1,3,4');//默认主键`id`作为key
//输出 ：唱歌、爬山、游泳
```

```
//指定text/key字段
$table->matches('hobbies','爱好')->optionsData(HobbyType::select(), 'name', 'hoby')->value('dance,climb');
//输出 ：跳舞、爬山
```

MDEditor

markdown 编辑器

未内置资源，若要使用，请单独安装

```
composer require ichynul/builder-mdeditor
```

MDReader

markdown 显示

未内置资源，若要使用，请单独安装

```
composer require ichynul/builder-mdeditor
```

Month

月份选择

主要方法

```
//修改格式，默认为MM
public function format($val){}

//时间戳格式化，若value值为时间戳数字格式
public function timespan($val = 'm'){}
```

MultipleFile

多个文件上传

主要方法

```
//设置是否可以上传
public function canUpload($val){}

//是否显示文件路径输入框，可以直接手动修改
public function showInput($val){}

//是否显示[选择]按钮从已上传的文件中选择
public function showChooseBtn($val){}

//文件总数量限制
public function limit($val){}

//缩略图小尺寸[50 x 50]
public function smallSize(){}

//缩略图中尺寸[120 x 120]
public function mediumSize(){}

//缩略图大尺寸[240 x 240]
public function bigSize(){}

//缩略图巨大尺寸[480 x 480]
public function largeSize(){}

//自定缩略图中尺寸[w x h]
public function thumbSize($w, $h){}

//参数设置
public function jsOptions($options){}
```

MultipleImage

多图上传

继承: MultipleFile

MultipleSelect

下拉多选

继承: Select

HasOptions trait 为 Checkbox Radio Select DualListbox MultipleSelect Match Matches 共有。

Number

数字输入框

带 `+` `-`号

主要方法

```
//最小限制
public function min($val){}

//最大限制
public function max($val){}
```

Password

密码输入框

Radio

单选框

主要方法

```
//多个选项框是否在同一行
public function inline($val = true)

//以块状显示，一种美化
public function blockStyle($val = true)
```

`HasOptions` trait 为 `Checkbox` `Radio` `Select` `DualListbox` `MultipleSelect` `Match` `Matches` 共有。

颜色主题

```
$form->radio('on_sale', '上架')
    ->adddClass('radio-warning')//颜色主题
    ->blockStyle()//显示为块状
    ->options([1 => '已上架', 0 => '未上架'])->default(0);
```

`radio-warning`，其中[warning]为颜色主题，所有可选主题如下：

- primary（主色）
- success（成功）
- secondary（灰色）
- info（一般信息）
- warning（警告）
- danger（警告）
- dark（黑色）
- purple（紫色）
- pink（粉红色）
- cyan（青色）
- yellow（黄色）
- brown（棕色）

RangeSlider

滑块

Rate

百分比

Raw

原始输出

内容可带html

```
{$value|raw}
```

相当于带label的 `field`

Select

下拉选择

`HasOptions` trait 为 `Checkbox` `Radio` `Select` `DualListbox` `MultipleSelect` `Match` `Matches` 共有。

主要方法

```
//是否使用增强的select2，默认为使用
public function select2($use){}

//ajax加载url
public function dataUrl($url, $textField = '', $idField = '', $delay = 250, $loadmore = true){}

//占位提示
public function placeholder($val){}

//js 参数设置
public function jsOptions($options){}

//联动，当此select的选值改变时，nextSelect会重新加载，nextSelect必须设置了ajax加载url
public function withNext($nextSelect){}
```

HasOptions trait 为[Checkbox][Radio][Select][MultipleSelect][Match][Matches]共有。

关于联动

```
$search->select('province', '省份')->dataUrl(url('api/areacity/province'), 'ext_name')->withNext(
    $search->select('city', '城市')->dataUrl(url('api/areacity/city'), 'ext_name')->withNext(
        $search->select('area', '地区')->dataUrl(url('api/areacity/area'), 'ext_name')
    )
);
//省份变化了，会以选中的省份值作为参数去请求`api/areacity/city`把下面的城市列出来，
//城市变化也类似
```

相当于：

```
$province = $search->select('province', '省份')->dataUrl(url('api/areacity/province'), 'ext_name');
$city = $search->select('city', '城市')->dataUrl(url('api/areacity/city'), 'ext_name')->withPrev($area);
$area = $search->select('area', '地区')->dataUrl(url('api/areacity/area'), 'ext_name')->withPrev($city);
```

但一般不这么用，除非这几个字段被其他字段分开了，位置上没有连在一起，但仍然想保持联动效果。

配合fields使用

最好用 `fields` 包围起来，使多个联动字段组合起来，共享一个左边的 `label` 然后平分右边 `fields` ，这样布局更容易。

如：

```
$form->fields('省/市/区');
$form->select('province', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/province'), 'ext_name')->withNext(
    $form->select('city', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/city'), 'ext_name')->withNext(
        $form->select('area', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/area'), 'ext_name')
    )
);
$form->fieldsEnd();

//或：

$form->fields('省/市/区')->with(
    $form->select('province', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/province'), 'ext_name')->withNext(
        $form->select('city', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/city'), 'ext_name')->withNext(
            $form->select('area', ' ', 4)->size(0, 12)->showLabel(false)->dataUrl(url('api/areacity/area'), 'ext_name')
```

```
        )
    );
);
```

ajax 数据源

`tpext\builder\traits\actions\HasIndex` 已内置了以当前控制器模型 `$dataModel` 为基础的[selectPage]

如有控制器：`\app\admin\controller\Member`

```
<?php

namespace app\admin\controller;

use app\common\model;
use think\Controller;
use tpext\builder\traits\HasBuilder;

/**
 * Undocumented class
 * @title 会员管理
 */
class Member extends Controller
{
    use HasBuilder;

    /**
     * Undocumented variable
     *
     * @var model\Member
     */
    protected $dataModel;

    protected function initialize()
    {
        $this->dataModel = new model\Member;
        $this->pageTitle = '会员管理';
        $this->enableField = 'status';
        $this->pagesize = 8;

        $this->selectTextField = '{id}#{nickname}({mobile})';//设置下拉显示格式
        $this->selectSearch = 'username|nickname|mobile';//设置搜索字段
    }
}
```

那么其他页面可以：`$search->select('member_id', '会员')->dataUrl(url('/admin/member/selectPage'));`

显示：

```
<select name="member_id">
<option vlaue="1001">1001#小明(13312345678)</option>
<option vlaue="1002">1002#小红(13312345677)</option>
<option vlaue="1003">1003#小刚(13312345676)</option>
<!--更多-->
</select>
```

其他情况可以自己写 `action` 实现

实例：

```
/**
* Undocumented function
*
* @title 下拉选择用户
* @return mixed
*/
public function selectPageUser()
```

```
{
    $q = input('q');
    $page = input('page/d');
    $selected = input('selected');

    if ($selected) {
        $list = $this->dataModel->where('id', 'in', $selected)->order($sortOrder)->select();
    } else {
        $q = input('q', '');
        $page = $page < 1 ? 1 : $page;
        $pagesize = 20;

        $where = [];

        if ($q) {
            $where[] = ['nickname|mobile|username', 'like', '%' . $q . '%'];
        }

        $list = $this->dataModel->where($where)->limit(($page - 1) * $pagesize, $pagesize)->select();
    }

    $data = [];

    foreach ($list as $li) {
        $data[] = [
            'id' => $li['id'],
            'text' => $li['id_name'],
        ];
    }

    return json(
        [
            'data' => $data,
            'has_more' => count($data) == $pagesize,
        ]
    );
}
```

ajax附带其他字段值

`->withParams($parasm)`;

如：

```
$form->radio('type', '人员类型')->options([1 => '男', 2 => '女']);
$form->select('member_id', '人员')->dataUrl(url('/admin/member/selectPage2'))->withParams('type');
//每次ajax请求[/admin/member/selectPage2]页面时，都会附带一个参数如[&type=1]，在selectPage2方法中就可以根据此参数，只返回男性或女性的姓名。
//$parasm参数可为多个，用`,`分割，如：->withParams('type,age');
```

Show

显示内容

跟 `Raw` 差不多，区别是 `Raw` 可以显示html, `show` 会转义html。

主要方法

```
//对于较长的内容，切割一部分显示，点击[...]按钮显示全部
public function cut($len = 0, $more = '...'){}

//内容默认是被一个`div`包围的,inline模式就没有外面的`div`。此用法在fields包含多个`show`时有用
public function inline($val){}
```

`form` 中使用 `field` 和实用 `show` 效果差不多。

`table` 中使用 `fields` 里面使用 `show` 和 `field` 有略微差别。 `show` 有[div]包裹着，每个字段占一行，是块级元素， `field` 无[div]包裹，多个字段合并占一行。

```
$table->fields('consignee', '收货人/电话')->with(
    $table->show('consignee', '收货人'),
    $table->show('mobile', '电话')->default('--')
);
```

显示：

收货人/电话
小明 13612345678

```
$table->fields('consignee', '收货人/电话')->with(
    $table->field('consignee', '收货人'),
    $table->field('mobile', '电话')->default('--')
);
```

显示：

收货人/电话
小明13612345678

新版本中为 `show` 新加了个方法 `inline()` ，table中使用 `inline` 的 `show` 和 `field` 表现一样,要说区别，就是使用 `field` 可少敲几次键盘。

SwitchBtn

切换按钮

主要方法

```
//设置开关状态对应的值，默认为[1=>'开',0=>'关']，例如可以设置为['on'=>'开','off'=>'关']
public function pair($on = 1, $off = 0){}
```

Tags

标签

Text

输入框

主要方法

```
//设置最大长度提示
public function maxlength($val = 0){}

//占位提示
public function placeholder($val){}

//前面加入html
public function befor($html){}

//后面加入html
public function after($html){}

//前面加入字符或字体
public function beforSymbol($text){}

//后面加入字符或字体
public function afterSymbol($text){}
```

Textarea

文本域

主要方法

```
//设置最大长度提示
public function maxlength($val = 0){}

//设置rows
public function rows($val = 3){}

//占位提示
public function placeholder($val){}
```

Time

时间选择

主要方法

```
//修改格式，默认为HH:mm:ss
public function format($val){}

//时间戳格式化，若value值为时间戳数字格式
public function timespan($val = 'H:i:s'){}
```

TimeRange

时间区间选择

Tinymce

Tinymce富文本编辑器

未内置资源，若要使用，请单独安装

```
composer require ichynul/builder-tinymce
```

UEditor

百度UEditor富文本编辑器

未内置资源，若要使用，请单独安装

```
composer require ichynul/builder-ueditor
```

WangEditor

WangEditor富文本编辑器

已内置 wangEditor 资源

wangEditor (也是默认编辑器：调用 `$form->editor()` 时默认使用它)

Year

年份选择

主要方法

```
//修改格式，默认为YYYY
public function format($val){}

//时间戳格式化，若value值为时间戳数字格式
public function timespan($val = 'Y'){}
```

Tab & Step

Tab & Step

Tab和Step是`form`内置的用于分割组件的方法。

tab实例：

-
  - 平铺直叙式，多次调用`$form->tab($label)`,每次调用后开启一个新的`tab`，然后跟随其他[fields]

```
$form->tab('基本信息');//tab 1

$form->image('avatar', '头像')->thumbSize(50, 50);
$form->text('username', '账号')->required()->maxlength(20);
$form->text('nickname', '昵称')->required()->maxlength(20);
$form->text('mobile', '手机号')->maxlength(11);
$form->text('email', '邮件')->maxlength(60);
//
$form->tab('其他信息');//tab 2

$form->textarea('remark', '备注')->maxlength(255);
$form->switchBtn('status', '状态')->default(1);
```

-
  - 使用`with`，此方式把内部的字段使用[with]方法包围，层次更加分明

```
$form->tab('基本信息')->with(
    $form->image('avatar', '头像')->thumbSize(50, 50),
    $form->text('username', '账号')->required()->maxlength(20),
    $form->text('nickname', '昵称')->required()->maxlength(20),
    $form->text('mobile', '手机号')->maxlength(11),
    $form->text('email', '邮件')->maxlength(60)
);//tab 1

//注意with参数与上面的不同之处
$form->tab('其他信息')->with(function(\tpext\builder\common\Form $from){
    $form->textarea('remark', '备注')->maxlength(255);
    $form->switchBtn('status', '状态')->default(1);
});//tab 2
```

step实例：

不常用也不实用，用法跟`tab`差不多

when表单联动

表单联动

[form]和[search]中可用

单选: radio / select

```
// 单选，radio / select
$form->radio('test1', '测试1')->options(['1' => '选项1', '2' => '选项2', '3' => '选项3'])->default(1)
   ->when(
      1, //选中值为1时
      $form->text('test_1_a', 'test_1_a')->required()
      //... 更多字段
   )->when(
      [2, 3],//选中值为[2 或 3]时，多个情况时传入参数为数组，数组各元素之间为[或]的关系
      $form->text('test_1_b', 'test_1_b')->required(),
      $form->textarea('test_1_c', 'test_1_c')->required()
      //... 更多字段
   );
```

多选：checkbox / multipleSelect / dualListbox

```
//
$form->checkbox('test2', '测试2')->options(['1' => '选项1', '2' => '选项2', '3' => '选项3', '4' => '选项4'])->default(1)
   ->when(
      1,//只选中一个值，且这个值为1时
      $form->text('test_2_a', 'test_2_a')->required()
      //... 更多字段
   )->when(
      [2, 3],//只选中一个值，且这个值为[2 或 3]时，多个情况时传入参数为数组，数组各元素之间为[或]的关系
      $form->text('test_2_b', 'test_2_b')->required(),
      $form->textarea('test_2_c', 'test_2_c')->required()
      //... 更多字段
   )->when(
      [4, '3+4', '2+1+4'],//(只选中一个值，且这个值为4时) 或 (同时选中3,4两个值) 或 (同时选中1,2,4三个值)。
      //数组各元素之间为[或]的关系，单个元素用+号连接多个值表示同时选中（值之间不分先后顺序[2+1+4]和[1+2+4]和[4+1+2]等情况等效）
      $form->radio('test_2_d', 'test_2_d')->options([1 => '1', 2 => '2']),
      $form->textarea('test_2_e', 'test_2_e')->required()
      //... 更多字段
   );
```

单个元素用+号连接多个值

单个元素内字符串都可以用`+`相连，不一定是数字，取决于根据选项值的类型：

```
//
$form->checkbox('test2', '测试2')->options(['type1' => '选项1', 'type2' => '选项2', 'type3' => '选项3', 'type4' => '选项4'])->default(1)
   ->when(
      'type1',
      $form->text('test_2_a', 'test_2_a')->required()
      //...
   )->when(
      ['type2', 'type3'],
      $form->text('test_2_b', 'test_2_b')->required()
      //...
   )->when(
      ['type4', 'type3 + type4', 'type2 + type1 + type4'],
      $form->radio('test_2_d', 'test_2_d')->options([1 => '1', 2 => '2'])
      //...
   );
```

when的使用

基本格式 `when($cases, ...$toggleFields)`;

第二参数 `$toggleFields` 的3种使用方式：

-
  - 作为可变参数：

`when($cases, $field1, $field2, $field3, $fieldN, );`

-
  - 作为数组元素[作为1的折衷方案，可变参数最后跟随一个,号，在低版本php中会报错，放在数组中可避免]

`with($cases, [$field1, $field2, $field3, $fieldN, ]);`

-
  - 作为匿名方法中的语句

`with($cases, function($form){$field1;$field2;$field3;$fieldN;});`

```
$form->radio('test1', '测试1')->options(['1' => '选项1', '2' => '选项2', '3' => '选项3', '4' => '选项4'])->default(1)
    ->when(
        1,
        $form->text('test_1_a', 'test_1_a')->required(),
        $form->textarea('test_1_b', 'test_1_b'),
        //... 更多字段
    )->when(
        2,
        [
            $form->text('test_1_c', 'test_1_c')->required(),
            $form->textarea('test_1_d', 'test_1_d'),
            //... 更多字段
        ]

    )->when(
        [3, 4],
        function(\tpext\builder\common\Form $_form) use ($form){
            //$_form 和 $form 是同一个东西，实际中使用其中一种方式即可。
            $_form->text('test_1_e', 'test_1_e')->required();
            $form->textarea('test_1_f', 'test_1_f');
            //... 更多字段
        }

    );
```

若第二参数 `$toggleFields` 不传，则可再调用 `with(...$toggleFields)` 方法。

```
$form->radio('test1', '测试1')->options(['1' => '选项1', '2' => '选项2', '3' => '选项3', '4' => '选项4'])->default(1)
    ->when(1)->with(
        $form->text('test_1_a', 'test_1_a')->required(),
        $form->textarea('test_1_b', 'test_1_b'),
        //... 更多字段
    )
    ->when(2)->with(
      [
          $form->text('test_1_c', 'test_1_c')->required(),
          $form->textarea('test_1_d', 'test_1_d'),
          //... 更多字段
      ]
    )
    ->when([3, 4])->with(function(\tpext\builder\common\Form $_form) use ($form){
          //$_form 和 $form 是同一个东西，实际中使用其中一种方式即可。
          $_form->text('test_1_e', 'test_1_e')->required();
          $form->textarea('test_1_f', 'test_1_f');
          //... 更多字段
    });
```

注意，第二个参数 **`$toggleFields`** 的传入时机

要么 `when` 的时候传入，要么 `when` 的时候不传，然后再调用 `with` 方法传入。不要两种方式同时使用，如下面的用法是错误的:

```
$form->radio('test1', '测试1')->options(['1' => '选项1', '2' => '选项2', '3' => '选项3', '4' => '选项4'])->default(1)
   ->when(
      1,
      $form->text('test_1_a', 'test_1_a')->required(),
      //... 更多字段
   )->with(
     [
         $form->text('test_1_b', 'test_1_b'),
         $form->textarea('test_1_c', 'test_1_c'),
         //... 更多字段
      ]
   );
```

拓展用法，利用 **with** 实现分散布局

```
$text1 = $form->text('test_1_a', 'test_1_a')->required();
//
$radio1 = $form->radio('test1', '测试1')->options(['1' => '选项1', '2' => '选项2'])->default(1);
//
$text2 = $form->text('test_1_b', 'test_1_b')->required();
$text3 = $form->text('test_1_c', 'test_1_c')->required();
//
$radio1->when(1)->with($text1, $text2);
$radio1->when(2)->with($text3);
//text1,text2,text3在文档中的位置相对于radio1有前有后是分散开的，如果在when中传入，那位置是受限的，使用`with`则更灵活。
```

切换fields

```
$form->radio('test1', '测试1')->options(['1' => '选项1', '2' => '选项2'])->default(1)
   ->when(1)->with(
      $form->left(12)->with(
         //fields
      )
   )
   ->when(2)->with(
      $form->left(12)->with(
         //fields
      )
   );
```

表单提交

- 切换后被隐藏的元素，表单提交时是被忽略的，后台获取不到对应的字段
- 可以在不同的 `case` 里面重复同一个字段

```
$form->radio('test1', '测试1')->options(['1' => '选项1', '2' => '选项2', '3' => '选项3'])
   ->when(1)->with(
      $form->text('test_1', 'test_1')->required()->help('case-1 的test_1'),
      $form->text('test_2', 'test_2')->required(),
   )
   ->when(2)->with(
      $form->text('test_1', 'test_1')->required()->help('case-2 的test_1'),//不同case的字段重复是允许的，只有其中一个会提交
      $form->text('test_3', 'test_3')->required(),
   )
   ->when(3)->with(
      $form->text('test_4', 'test_4')->required(),
   );
```

- 切换后隐藏的字段js验证暂时取消，重新切换回来后重新生效

with方法的使用

With

`with` 是 `fields`，`items`，`tab`，`step` 中的通用方法。

`fields` 和 `items` 如果不使用 `with` 方法，需要配合 `fieldsEnd` 、 `itemsEnd` 结尾。

使用 `with` 可省略结尾方法(`fieldsEnd`、 `itemsEnd`)，使代码层次更分明。

`tab`，`step` 不需要显式调用 `end` 方法，但也支持 `with` 方法。

有3种使用方式：

-
  - 作为可变参数：

`with($field1, $field2, $field3, $fieldN, );`

-
  - 作为数组元素[作为1的折衷方案，可变参数最后跟随一个,号，在低版本php中会报错，放在数组中可避免]

`with([$field1, $field2, $field3, $fieldN, ]);`

-
  - 作为匿名方法中的语句

`with(function($form){$field1;$field2;$field3;$fieldN;});`

以下代码可正常运行，其中仅使用了 `tab` 、 `fields` ， `step` 、 `items` 也类似。

```
$form->tab('tab1')->with(
    function () use ($form) {
        $form->fields('left1', '', 6)->size(0, 12)->showLabel(false)->with(
            function () use ($form) {
                $form->text('name', '名称')->required()->maxlength(55);
                $form->tags('keyword', '关键字');
                $form->textarea('description', '摘要')->maxlength(255);
            }
        );

        $form->fields('right1', '', 6)->size(0, 12)->showLabel(false)->with(
            $form->text('name', '名称')->required()->maxlength(55),
            $form->tags('keyword', '关键字'),
            $form->textarea('description', '摘要')->maxlength(255),//注意最后这个,号在低版本php中会报错，删除或者把fields放在[]中作为一个数组
        );
    }
);

$form->tab('tab2')->with(
    //如果不想使用use($from)，那可以在匿名方法传入参数.
    //并声明类型:\tpext\builder\common\Form(方便代码编辑器提示).
    //最好是在php文件头部引入: use tpext\builder\common\Form;
    //然后可简写为 : function (Form $_form){/* code */}
    //以下代码中 $_form 和 $form 是同一个东西，实际中使用其中一种方式即可。
    function (\tpext\builder\common\Form $_form) use ($form) {
        //left2 start
        $_form->fields('left2', '', 7)->showLabel(false)->size(0, 12);
        $_form->number('stock', '库存')->default(99);
        $form->number('click', '点击量')->default(1);
        $form->number('sales_sum', '销量')->default(0);
        $form->fieldsEnd();
        //left2 end
        //
        $form->fields('right2', '', 5)->size(0, 12)->showLabel(false)->with(
            [
                $_form->text('market_price', '市场价'),
                $form->text('cost_price', '成本价'),
            ]
        );
```

With

```
    }
);
```

页面布局-表头

表头

1. 在表格上面使用 `$builder->content()->fetch('header');` 渲染自定义视图.

实列

```
//准备模板里面需要的数据
$data = ['in' => [2,5,6,6], 'out' => [2,6,8,7,8]];

//渲染表头，模板文件:/admin/view/funding/header.html
$builder->content()->fetch('header',['data' => $data]);

//表格数据
$table = $builder->table();
$table->show('id', 'ID');
$table->show('money', '金额(元)');
//略...
```

2. 在表格上面使用 `$table->addTop()->content()->fetch('header');` 渲染自定义视图.

实列

```
//表格数据
$table = $builder->table();
$table->show('id', 'ID');
$table->show('money', '金额(元)');

$data = ['in' => [2,5,6,6], 'out' => [2,6,8,7,8]];
$table->addTop()->content()->fetch('header',['data' => $data]);
//略...
```

区别

`$table->addTop()->content()->fetch('header',['data' => $data])` ;会随表格ajax刷新

页面布局-左侧树形导航

左侧树形导航,主要有３种方式

1. 最原始的，使用`column`分割左右

```
class User extends Controller
{
   public function index()
   {
      $builder = $this->builder($this->pageTitle, $this->indexText);
      //****
      $left = $builder->column(1);
      $right = $builder->olumn(11);

      $treeData = $this->getTree();//以某种方式获取的数据

      //渲染一个自定义模板，在其中完成树形结构
      // /admin/view/user/tree.html
      $left->content()->fetch('tree',['treeData' => $treeData]);

      /*或者通过拼接html的形式 */
      //$html ='<ul>';
      //foreach($treeData as $d)
      //{
      //   $html .= "<li>{$d['name']}</li>";
      //}
      //$html .='</ul>';
      //$left->content()->display($html);
      //display/fetch 和 think框架的`Controller`类似，只能使用其中一种，`fetch`渲染模板文件，`display`直接输出html

      $table = $right->table();
      //略
      return $builder->render();
  }
}
```

2. 使用ZTree

```
use tpext\builder\traits\HasBuilder;

class User extends Controller
{

  public function index()
   {

      $builder = $this->builder($this->pageTitle, $this->indexText);

      $tree = $builder->tree('left');
      $tree->fill($this->categModel->all(), 'title');// categModel 中  `parent_id`为上级id字段

      $tree->trigger('.row-category_id');//被点击时，触发元素

      $builder->addStyleSheet('
      .col-md-left
      {
         width:13%;
         float:left;
      }
      .col-md-right
      {
         width:87%;
         float:right;
      }
```

```
        ');

        $table = $builder->table('right');//灵活运用，正常情况下此处参数是数字，但传字符串也行。
        //***************

        return $builder->render();
    }
}
```

3． 使用`HasBuilder`时可使用封装好的，是对方法 2 的进一步封装。

```
namespace app\admin\controller;

use tpext\builder\traits\HasBuilder;
use think\Controller;

/**
 * Undocumented class
 * @title 产品管理
 */
class Shopgoods extends Controller
{
    protected function initialize()
    {
            $this->dataModel = new GoodsModel;//商品模型，其中 `category_id`字段关联的到分类模型
            $this->categoryModel = new ShopCategory;//商品分类模型

        // 其他初始化...

        //左侧树
        $this->treeType = 'ztree'; //js插件类型，ztree或jstree
        $this->treeModel = $this->categoryModel;//分类模型
        $this->treeTextField = 'name';//分类模型中的分类名称字段
        $this->treeKey = 'category_id';//关联的键   localKey
    }
}
```

HasBuilder(封装页面)

- `HasBuilder`封装了增、删、查、改等所有动作
- 也可按需分别引入`tpext\builder\traits\actions\`命名空间下的相应动作。

```
use tpext\builder\traits\HasBuilder;
use tpext\builder\traits\actions;

class Admin extends Controller
{
    //方式1.完整引入，引入全部[增删查改]等动作。
    //use HasBuilder;

    //方式2.按需引入
    //基础
    use actions\HasBase;
    //按需加载，避免暴露不必要的action

    //列表
    use actions\HasIndex;
    //添加/修改
    use actions\HasAdd;
    use actions\HasEdit;

    //查看
    use actions\HasView;

    //字段编辑
    use actions\HasAutopost;
    //禁用/启用
    use actions\HasEnable;
    //删除
    use actions\HasDelete;

     /**
     * 数据筛选条件，配合buildSearch实用，若不需要搜索，这两个方法不用重写
     *
     * @return void
     */
    protected function filterWhere()
    {
        $searchData = request()->get();

        $where = [];
        //构建搜索条件
        if (!empty($searchData['username'])) {
            $where[] = ['username', 'like', '%' . $searchData['username'] . '%'];
        }
        return $where;
    }

     /**
     * 构建搜索
     *
     * @return void
     */
    protected function buildSearch()
    {
        $search = $this->search;
        $search->text('username', '账号', 3)->maxlength(20);
        //构建搜索表单
    }

    /**
     * Undocumented function
     *
     * @param array $data
     * @return void
     */
```

With

```
    protected function buildTable(&$data = [])
    {
        $table = $this->table;

        $table->show('id', 'ID');
        $table->show('username', '登录帐号');
        //构建表格
        //略

        $table->getToolbar()
            ->btnAdd()
            ->btnDelete();

         $table->getActionbar()
            ->btnEdit()
            ->btnDelete();
     }

    /**
     * 构建表单
     *
     * @param boolean $isEdit
     * @param array $data
     */
    protected function buildForm($isEdit, &$data = [])
    {
        $form = $this->form;
        $form->text('username', '登录帐号')->required()->beforSymbol('<i class="mdi mdi-account-key"></i>');
        //构建表单
        //略
    }

   /**
     * 保存数据
     *
     * @param integer $id
     * @return void
     */
    private function save($id = 0)
    {
        $data = request()->only([
          'username',
        ], 'post');

        // 数据验证等，略
        if ($id) {
            $data['update_time'] = date('Y-m-d H:i:s');
            $res = $this->dataModel->where(['id' => $id])->update($data);
        } else {
            $res = $this->dataModel->create($data);
        }
    }
}
```

selectPage(下拉数据)

默认每一个有列表页`index`的控制器都对应一个`selectPage`，`url('/admin/{controller}/selectPage')`；

```
class Member extends Controller
{
    protected function initialize()
    {
        //...其他初始化

        $this->selectTextField = '{id}#{nickname}({mobile})';
        $this->selectSearch = 'nickname|mobile';

        $this->selectIdField ='id';
        $this->selectFields ='id,nickname,mobile';
        $this->selectOrder = 'nickname';
        $this->selectScope = [['enable', 'eq' ,1]];
    }
}
```

说明

1. `selectTextField` 格式化显示的文本，单个字段直接用字段名，多字段名称用大括号包围; `{fieldname}`。

如上面的格式表示：会员id#昵称(手机号)：

```
<select>
<option value="10001">10001#小明(13312345670)</option>
<option value="10003">10002#小红(13312345671)</option>
<option value="10003">10003#小刚(13312345672)</option>
<select>
```

2. `selectSearch` 查询字段 用户在下拉框中输入字符串查询，ajax请求后台接口，接口中返回 `昵称` 或 `手机号包含` 这个关键字的数据。 核心代码：

```
$kwd = input('q');
$data = $this->dataModel->where('nickname|mobile','like',"%$kwd%")->select();
```

3. `selectIdField` 键，控制的是 `<option value="10001">10001#小明(13312345670)</option>` 中的 `value` 对应到哪个字段。如果键不是 `id`，那就需要设置。
4. `selectFields` 优化查询效果，比如上面的查询，只需要 `id`、`nickname`、`mobile` 三个字段。默认是 `*` 全部字段，如果追求性能，可以设置查询字段只这三个 。
5. `selectOrder` 顾名思义，排序方式。
6. `selectScope` 默认条件，比如上面的例子，只显示已启用的用户，未启用的就不显示出来让选择。

其他说明

`selectTextField`、`selectIdField` 是默认情况，如果使用 `select` 的 `dataUrl` 方法时没设置 `textField`、`idField` 两个参数，就按默认配置的。

默认情况：

```
$select->select('member_id', '会员')->dataUrl(url('/admin/member/selectPage'));
```

```
<select>
<option value="10001">10001#小明(13312345670)</option>
<option value="10003">10002#小红(13312345671)</option>
<option value="10003">10003#小刚(13312345672)</option>
<select>
```

如果指定了`textField`、`idField`，就可以覆盖：

```
$select->select('mobile', '会员手机')->dataUrl(url('/admin/member/selectPage'),'{id}#{nickname}','mobile');
```

```
<select>
<option value="13312345670">10001#小明</option>
<option value="13312345671">10002#小红</option>
<option value="13312345672">10003#小刚</option>
<select>
```

`textField`是单个字段，大括号`{}`省略 `idField` 未指定，用默认

```
$select->select('mobile', '会员手机')->dataUrl(url('/admin/member/selectPage'),'nickname');
```

```
<select>
<option value="10001">小明</option>
<option value="10003">小红</option>
<option value="10003">小刚</option>
<select>
```

模型关联

//用户模型：Member

```
class Member extends Model
{
    /** **/
    public function level()
    {
        return $this->belongsTo(MelberLevel::class, 'level_id', 'id');//MelberLevel是用户等级的模型类
    }
}
```

//控制器

```
class Member extends Controller
{
    protected function initialize()
    {
        //...其他初始化

        $this->selectTextField = '{id}#{nickname}({level.name})';
        $this->selectSearch = 'nickname|mobile';

        $this->selectIdField ='id';
        $this->selectFields ='id,nickname,level_id';//   `level_id`这个字段是必须的
        $this->selectOrder = 'nickname';
        $this->selectScope = [['enable', 'eq' ,1]];

        $this->selectWith= ['level'];//关联模型
    }
}
```

```
$select->select('member_id', '会员')->dataUrl(url('/admin/member/selectPage'));
```

```
<select>
<option value="10001">10001#小明(一年级)</option>
<option value="10002">10002#小刚(二年级)</option>
<option value="10003">10003#小红(三年级)</option>
<select>
```

模型关联

此特性在 `1.8.91` 及以后版本中提供

tp5 关联模型文档：https://www.kancloud.cn/manual/thinkphp5_1/354057

假如有商品表和商品分类表，每个商品都归属于一个分类。

```
class ShopCategory extends Model
{
    protected $autoWriteTimestamp = 'datetime';
}
```

```
class ShopGoods extends Model
{
    protected $autoWriteTimestamp = 'datetime';

    //一对一关联
    public function category()
    {
        //两种都行
        //return $this->hasOne('ShopCategory', 'id', 'category_id');
            return $this->belongsTo('ShopCategory', 'category_id', 'id');
    }

    //获取器
    public function getCategoryNameAttr($value, $data)
    {
        $category = ShopCategory::get($data['category_id']);
        return $category ? $category['name'] : '--';
    }
}
```

```
class Shopgoods extends Controller
{

    protected function initialize()
    {
        //... 其他初始化代码

        //配合方式二此使用，处设置列表页需要加载的关联模型
        $this->indexWith = ['category'];
    }

    protected function buildTable(&$data = [])
    {
        $table = $this->table;

        //...其他字段

        //方式一：获取器，用在列表中查询次数过多
        //$table->show('category_name', '分类');

        //方式二：关联（推荐）
        $table->show('category.name', '分类');
        //...其他字段
    }

    protected function buildForm($isEdit, &$data = [])
    {
        $form = $this->form;

        //...其他字段

        //方式一：获取器
        //$table->show('category_name', '分类');
```

```
        //方式二：关联
        $form->show('category.name', '分类');
        //...其他字段

        //由于是单条数据，性能上来说二者等效
    }
}
```

字段名称中包含`.`，一般代表关联模型的字段。如果作为输入，则作为一个数组。

```
protected function buildForm($isEdit, &$data = [])
{
    $form->text('name', '姓名');        //<input type="text" name="name">
    $form->text('age', '年龄');         //<input type="text" name="age">
    //...其他字段
    $form->text('ext.data1', '扩展1');  //<input type="text" name="ext[data1]">
    $form->text('ext.data2', '扩展2');  //<input type="text" name="ext[data2]">
    $form->text('ext.data3', '扩展3');  //<input type="text" name="ext[data3]">
}

private function save($id = 0)
{
   $data = request()->only([
       'name',
       'age'
       //...
   ], 'post');
   //保存主数据，略

   $ext = input('post.ext/a');//接收扩展数据，并转换为数组

   //保存扩展数据数据，略
}
```

导出

导出功能由`table`组件提供，无需做额外的处理。
默认只提供 csv 格式导出。
要导出 xls/xlsx 格式需安装excel插件：
`phpoffice/phpexcel`
`composer require phpoffice/phpexcel`

或`phpoffice/phpspreadsheet`（推荐）

`composer require phpoffice/phpspreadsheet`

```
//导出（默认，点击按钮直接请求后台）
$table->getToolbar()->btnExport('', '导出', 'btn-pink','mdi-export', 'title="导出"');

//自定义支持的导出类型
$items = [
    'xls' => 'xls文件',
    'xlsx' => 'xlsx文件',
    'json' => 'json文件',
];
//导出（可选，点击弹出菜单，选择导出类型）
$table->getToolbar()->btnExports($items, '', '导出', 'btn-secondary',  'mdi-export', 'title="导出"');
```

除默认的[csv/xls/xlsx]三种格式外，其他的要自己实现到处文件的逻辑。如不实现，任然导出为默认的csv格式。

请重写控制器中的`exportTo`方法.

```
    /**
     * Undocumented function
     *
     * @param string $fileType 其他类型的导出
     * @return mixed
     */
    protected function exportTo($data, $displayers, $__file_type__)
    {
        //$logic = new Export;
        //return $logic->toCsv($this->pageTitle, $data, $displayers);

        //$data 数据
        //$displayers 字段
        //$__file_type__ 点击的文件类型
        // 可参考 Export 表格导出的逻辑，怎么处理数据，怎么保存文件，怎么返回数据等。
    }
```

//默认所有页面都会有导出按钮，如果不需要导出，使用以下代码禁用： $table->useExport(false);

字段限制

`exportOnly` 只导出这些字段

`exportExcept` 除这些之外的字段

二者使用其一即可。

或者在构建表格的时候根据全局变量`$this->isExporting`判断当前是否为导出：

```
/**
 * 构建表格
 *
 * @return void
 */
protected function buildTable(&$data = [])
{
    $table = $this->table;

    if (!$this->isExporting) {//导出时不显示
```

```
        $table->show('order_id', '订单id');
        $table->show('order_sn', '订单sn');
    }

    $table->show('consignee', '收件人姓名');
    $table->show('mobile', '收件人手机/电话');
    $table->show('province_text', '收件人省');
    $table->show('city_text', '收件人市');
    $table->show('area_text', '收件人区');
    //....

    if ($this->isExporting) {//如果是导出，添加一些字段
        $table->show('send_name', '寄件人姓名')->default('张三');
        $table->show('send_phone', '寄件人手机/电话')->default('13333333333');
        $table->show('send_p', '寄件人省')->default('云南省');
        $table->show('send_c', '寄件人市')->default('昆明市');
        $table->show('send_a', '寄件人县/区')->default('五华区');
        $table->show('send_address', '寄件人详细地址')->default('xxx区中1234号'); //寄件人详细地址
    }
    //....
}
```

```
class Cmsbanner extends Controller
{
    protected function initialize()
    {
        //...
        $this->exportOnly = ['name', 'description'];//只导出[标题]和[简介]

        //$this->exportExcept = ['create_time', 'update_time'];//导出除[添加时间、更新时间]以外所有字段
    }
}
```

导入

```
$table->getToolbar()
    ->btnImport(url('import'), 'xls,xlsx', ['800px', '550px'], 20, '导入发货单')//方式1：默认的上传页面，文件上传成功后跳转到同控制器的`import()`
    ->btnLink(url('upexcel'), '导入', 'btn-pink', 'mdi-cloud-upload', 'data-layer-size="800px,550px"')//（推荐）方式2：自定义一个弹出页面`upexcel`上传
    ->html('<a class="label label-info" target="_blank" href="/template/发货订单模板.xls">发货单模板下载</a>');


pubic function import()
{
   //方式1，只有一个文件路径参数
   $fileurl = input('fileurl');
   if (is_file(app()->getRootPath() . 'public' . $fileurl)) {
      // 导入逻辑...
      return $this->builder()->layer()->closeRefresh(1, '导入成功：' . $fileurl);
   }

   $builder = $this->builder('出错了');
   $builder->content()->display('<p>' . '未能读取文件:' . $fileurl . '</p>');
   return $builder->render();
}

protected function import2()
{
   //方式2，可以有更多参数
   $fileurl = input('fileurl');
   $date = input('date ');
   $is_overwrite = input('is_overwrite ');
   if (is_file(app()->getRootPath() . 'public' . $fileurl)) {
      // 导入逻辑...
      return $this->builder()->layer()->closeRefresh(1, '导入成功：' . $fileurl);
   }

   $builder = $this->builder('出错了');
   $builder->content()->display('<p>' . '未能读取文件:' . $fileurl . '</p>');
   return $builder->render();
}

public function upexcel()
{
   if (request()->isPost()) {//上传后提交
      return $this->import2();
   }

   $builder = $this->builder();
   $form = $builder->form();
   $form->raw('template', '模板')->value('<a class="label label-info" target="_blank" href="/template/发货订单模板.xls">发货单模板下载</a>');
   $form->file('fileurl', '上传表格文件xlx/xlsx')->jsOptions(['fileSingleSizeLimit' => 20 * 1024 * 1024, 'ext' => ['xlx', 'xlsx']])->showChooseBtn(false
   $form->date('date','导入月份')->default(date('Y-m-01'));
   $form->checkbox('is_overwrite','是否覆盖')->default(1);

   $form->ajax(false);

   $form->btnSubmit('开始导入', '1 col-xs-2', 'btn-danger btn-loading');
   $form->html('', ' ', '4 col-xs-2')->showLabel(false);
   $form->btnLayerClose();

   return $builder->render();
}
```

tpextmyadmin-后台权限

开发主题包

开发主题包

以下以[H+admin]主题为例

```
<?php

namespace hplusadmin\common;

use tpext\common\Resource as baseResource;
use tpext\myadmin\common\Module as adminModule;

class Resource extends baseResource
{
    protected $version = '1.0.1';

    protected $name = 'hplus.admin';

    protected $title = 'H+admin后台主框架模板样式';

    protected $description = '提供[H+admin]相关样式资源，主要改变主框架[/admin/index/index]的样式';

    protected $root = __DIR__ . '/../';

    protected $assets = 'assets';//样式目录，你的主题需要的其他样式脚本

    // loaded方法会在扩展被发现的情况下执行[无论是否启用]
    //一般用于向其他模块提供一些额外的选项，但是否使用这些选项，控制权在于其他模块(一般通过配置项)。
    public function loaded()
    {
        $indexView = $this->getRoot() . implode(DIRECTORY_SEPARATOR, ['admin', 'view', 'index', 'index.html']);///admin/view/index/index.html
        $loginView = $this->getRoot() . implode(DIRECTORY_SEPARATOR, ['admin', 'view', 'index', 'login.html']);

        adminModule::getInstance()->addIndexView($indexView, 'H+后台模板');//添加[/admin/index/index]模板
        adminModule::getInstance()->addLoginView($loginView, 'H+后台模板');//添加[/admin/index/login]模板
        //添加后不会自动生效，需要在[tpext.myadmin]后台扩展的配置页面选择保存后生效。
    }
}
```

此外，如果不需要替换html模板，仅是修改页面样式，以通过 `tpext\myadmin\common\MinifyTool` 的几个静态方法对其内置的资源进行删除、替换操作。

```
<?php

use tpext\myadmin\common\MinifyTool;

public function loaded()
{
    MinifyTool::removeCss('/assets/lightyearadmin/css/style.min.css');//删除
    MinifyTool::replaceJs('/assets/lightyearadmin/js/main.min.js', '/assets/mymodule/js/main.min.js');//替换
}
```

隐藏登陆页面

#隐藏登陆页面

插件配置中设置[隐藏登录页面]为 是 。 此时[未登录]状态下直接访问 http://yourhost/admin/* 任何页面都显示404或空白。

修改文件： application\index\controller\Index.php （tp框架自带，没有则新建）

添加：

```
public function adminpage313123213()
{
    session('login_session_key', '666');//随便设置个字符串都可以，不一定是666
    return redirect(url('/admin/index/login'));
}
```

访问 http://yourhost/index/index/adminpage313123213 ，页面会跳转到 http://yourhost/admin/index/login ，就可以正常登陆。

可以定期修改 adminpage313123213 名称。

这里只是个思路，换个模块名，或换个控制器名也都可以。

控制器和方法的标注规范

控制器和方法的标注规范

控制器和方法都通过`@title desc`的注解标注，生成权限列表时就可以读取出来作为备注。

一些常用方法已经内置了说明，不需要额外标注：

```
$actionNames = [
            'index' => '列表',
            'list' => '列表',
            'add' => '添加',
            'create' => '新建',
            'edit' => '修改',
            'view' => '查看',
            'update' => '更新',
            'delete' => '删除',
            'enable' => '启用',
            'disable' => '禁用',
            'status' => '状态',
            'install' => '安装',
            'uninstall' => '卸载',
            'login' => '登录',
            'logout' => '注销',
            'dashbord' => '仪表盘',
            'upload' => '上传',
            'download' => '下载',
            'autopost' => '字段编辑',
            'import' => '导入',
            'export' => '导出',
            'welcom' => '欢迎',
            'selectpage' => '下拉选择',
        ];
```

完整实例：

```
<?php

namespace tpext\myadmin\admin\controller;

use think\Controller;

/**
 * @title 管理员管理
 */
class Admin extends Controller
{
    /**
     * @title 清空错误次数
     * @return mixed
     */
    public function clearErrors()
    {

    }
}
```

自定义左上角LOGO

自定义左上角LOGO

默认为一张图片：

```
<img src="/assets/lightyearadmin/images/logo.png" alt="Admin logo" title="Tpext后台管理系统">
```

也可以是纯html:

```
<h4 style="color:#fff;height:50px;line-height:50px;margin:0;">Tpext后台管理系统</h4>
```

在后台首页【右上角】添加按钮

在后台首页【右上角】添加按钮

-
  - 新建 `application/common/behavior/RightLinks.php` :

```
<?php

namespace app\common\behavior;

class RightLinks
{
    public function run()
    {
            //这里只有一个链接，如果有多个的话用多个`<li></li>`标签
        echo '<li><a title="打开手机版" class="link" href="/mobile/" target="_blank"><i class="mdi mdi-cellphone-iphone "></i></a></li>';
    }
}
```

-
  - 在 `application\tags.php` 中监听事件 `topbar_right_links` :

```
// 应用行为扩展定义文件
<?php

return [
    // 应用初始化
    'app_dispatch' => [
        'app\\common\\behavior\\Config', // 注册配置行为
    ],
    // 应用开始
    'app_begin' => [],
    // 模块初始化
    'module_init' => [],
    // 操作开始执行
    'action_begin' => [],
    // 视图内容过滤
    'view_filter' => [],
    // 日志写入
    'log_write' => [],
    // 应用结束
    'app_end' => [],
    // 扩展加载
    'tpext_find_extensions' => [
        'app\\common\\behavior\\Extensions',
    ],
    // 'topbar_left_links' =>[
    //     'app\common\behavior\LeftLinks'
    // ],
    'topbar_right_links' => [
        'app\common\behavior\RightLinks'
    ]
];
```

-
  - 添加左上角也类似，监听事件 `topbar_left_links` 。

```
<?php

namespace app\common\behavior;

class LeftLinks
{
    public function run()
```

```
    {
            //这里只有一个链接，如果有多个的话用多个`<a></a>`标签，外围不需要`<li></li>`了
        echo '<a style="margin-left:10px;" title="打开手机版" class="link" href="/mobile/" target="_blank"><i class="mdi mdi-cellphone-iphone "><
    }
}
```

在后台首页【右上角】添加按钮