# 应用主体

应用主体是管理 Yii 应用系统整体结构和生命周期的对象。

应用主体在 [入口脚本](https://www.yiichina.com/doc/guide/2.0/structure-entry-scripts) 中创建并能通过表达式 \Yii::$app 全局范围内访问。

Yii有两种应用主体: [网页应用主体](https://www.yiichina.com/doc/api/2.0/yii-web-application) 和 [控制台应用主体](https://www.yiichina.com/doc/api/2.0/yii-console-application)， 如名称所示，前者主要处理网页请求，后者处理控制台请求。

# 应用主体配置

当 [入口脚本](https://www.yiichina.com/doc/guide/2.0/structure-entry-scripts) 创建了一个应用主体， 它会加载一个 [配置](https://www.yiichina.com/doc/guide/2.0/concept-configurations) 文件并传给应用主体。



一般在config/web.php

 应用主体配置文件标明如何设置应用对象初始属性。

# 应用主体属性

## 必要属性

在一个应用中，至少要配置2个属性: [id](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$id-detail) 和 [basePath](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$basePath-detail)。

### [id](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$id-detail)

[id](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$id-detail) 属性用来区分其他应用的唯一标识ID。主要给程序使用。 为了方便协作，最好使用数字作为应用主体ID， 但不强制要求为数字。

### [basePath](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$basePath-detail)

[basePath](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$basePath-detail) 指定该应用的根目录。 根目录包含应用系统所有受保护的源代码。 在根目录下可以看到对应 MVC 设计模式的models， views，controllers 等子目录。

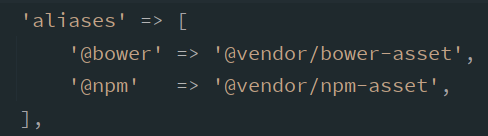
[basePath](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$basePath-detail) 属性经常用于派生一些其他重要路径（如 runtime 路径）， 因此，系统预定义 @app 代表这个路径。 派生路径可以通过这个别名组成（如@app/runtime代表runtime的路径）。

## 重要属性

本小节所描述的属性通常需要设置， 因为不同的应用属性不同。

### [aliases](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$aliases-detail)

该属性允许你用一个数组定义多个 [别名](https://www.yiichina.com/doc/guide/2.0/concept-aliases)。 数组的key为别名名称，值为对应的路径。 例如：



使用这个属性来定义别名， 代替 [Yii::setAlias()](https://www.yiichina.com/doc/api/2.0/yii-baseyii" \l "setAlias()-detail) 方法来设置。

### [bootstrap](https://www.yiichina.com/doc/api/2.0/yii-base-application" \l "bootstrap()-detail)

用数组指定启动阶段 [bootstrapping process](https://www.yiichina.com/doc/api/2.0/yii-base-application" \l "bootstrap()-detail) 需要运行的组件。 比如，如果你希望一个 [模块](https://www.yiichina.com/doc/guide/2.0/structure-modules) 自定义 [URL 规则](https://www.yiichina.com/doc/guide/2.0/runtime-url-handling)， 你可以将模块ID加入到bootstrap数组中。

属性中的每个组件需要指定以下一项:

* 应用 [组件](https://www.yiichina.com/doc/guide/2.0/structure-applications" \l "components) ID.
* [模块](https://www.yiichina.com/doc/guide/2.0/structure-applications" \l "modules) ID.
* 类名.
* 配置数组.
* 创建并返回一个组件的无名称函数.

例如：



### [components](https://www.yiichina.com/doc/api/2.0/yii-di-servicelocator" \l "$components-detail)

这是最重要的属性，它允许你注册多个在其他地方使用的 [应用组件](https://www.yiichina.com/doc/guide/2.0/" \l "structure-application-components)。 例如



在应用中可以任意注册组件，并可以通过表达式 \Yii::$app->ComponentID 全局访问。

第一次使用以上表达式时候会创建应用组件实例， 后续再访问会返回此实例，无需再次创建。

一个应用组件只会在第一次访问时实例化， 如果处理请求过程没有访问的话就不实例化。 有时你想在每个请求处理过程都实例化某个组件即便它不会被访问， 可以将该组件ID加入到应用主体的 [bootstrap](https://www.yiichina.com/doc/api/2.0/yii-base-application" \l "bootstrap()-detail) 属性中。

应用组件可以是任意对象。

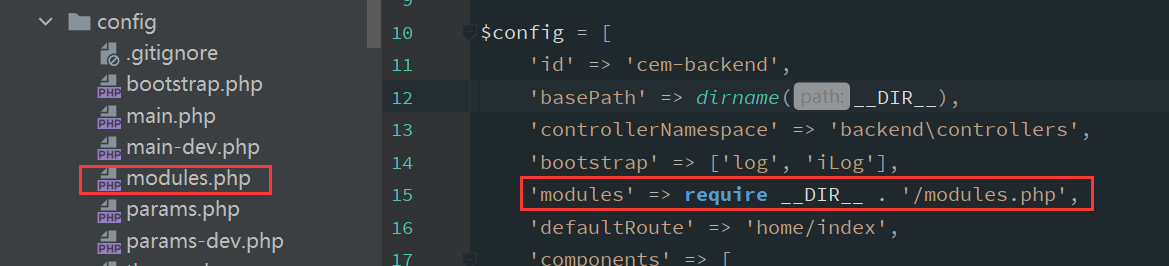
### [modules](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$modules-detail)

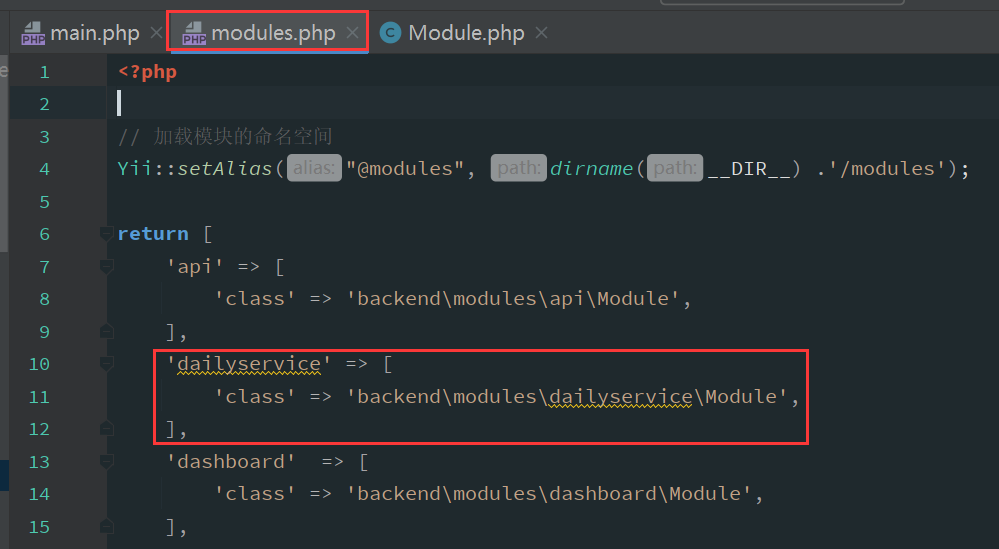
该属性指定应用所包含的 [模块](https://www.yiichina.com/doc/guide/2.0/structure-modules)。

该属性使用数组包含多个模块类 [配置](https://www.yiichina.com/doc/guide/2.0/concept-configurations)，数组的键为模块ID， 例：



或者把模块配置单独搞一个配置文件







模块在大型项目中常备使用，这些项目的特性可分组， 每个组包含一些强相关的特性， 每个特性组可以做成一个模块由特定的开发人员和开发组来开发和维护。

在特性组上，使用模块也是重用代码的好方式， 一些常用特性，如用户管理，评论管理，可以开发成模块， 这样在相关项目中非常容易被重用。

可以把模块当成一个小的应用来对待。

### [params](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$params-detail)

该属性为一个数组，指定可以全局访问的参数， 代替程序中硬编码的数字和字符， 应用中的参数定义到一个单独的文件并随时可以访问是一个好习惯。 例如用参数定义缩略图的长宽如下：

[

'params' => [

'thumbnail.size' => [128, 128],

],

]

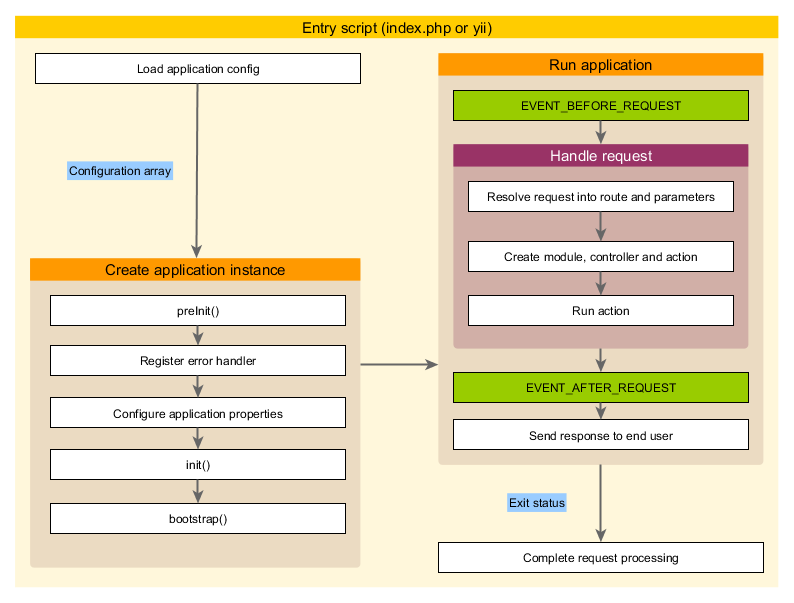
然后简单的使用如下代码即可获取到你需要的长宽参数：

$size = \Yii::$app->params['thumbnail.size'];$width = \Yii::$app->params['thumbnail.size'][0];

以后想修改缩略图长宽， 只需要修改该参数而不需要相关的代码。

这里是配置整个应用级的常量。模块级的常量可以在模块目录下单独配置。

# 应用主体生命周期



当运行 [入口脚本](https://www.yiichina.com/doc/guide/2.0/structure-entry-scripts) 处理请求时， 应用主体会经历以下生命周期:

1. 入口脚本加载应用主体配置数组。
2. 入口脚本创建一个应用主体实例：
   1. 调用 [preInit()](https://www.yiichina.com/doc/api/2.0/yii-base-application" \l "preInit()-detail) 配置几个高级别应用主体属性， 比如 [basePath](https://www.yiichina.com/doc/api/2.0/yii-base-module" \l "$basePath-detail)。
   2. 注册 [error handler](https://www.yiichina.com/doc/api/2.0/yii-base-application" \l "$errorHandler-detail) 错误处理方法。
   3. 配置应用主体属性。
   4. 调用 [init()](https://www.yiichina.com/doc/api/2.0/yii-base-application" \l "init()-detail) 初始化，该函数会调用 [bootstrap()](https://www.yiichina.com/doc/api/2.0/yii-base-application" \l "bootstrap()-detail) 运行引导启动组件。
3. 入口脚本调用 [yii\base\Application::run()](https://www.yiichina.com/doc/api/2.0/yii-base-application" \l "run()-detail) 运行应用主体:
   1. 触发 [EVENT\_BEFORE\_REQUEST](https://www.yiichina.com/doc/api/2.0/yii-base-application" \l "EVENT_BEFORE_REQUEST-detail) 事件。
   2. 处理请求：解析请求 [路由](https://www.yiichina.com/doc/guide/2.0/runtime-routing) 和相关参数； 创建路由指定的模块、控制器和动作对应的类，并运行动作。
   3. 触发 [EVENT\_AFTER\_REQUEST](https://www.yiichina.com/doc/api/2.0/yii-base-application" \l "EVENT_AFTER_REQUEST-detail) 事件。
   4. 发送响应到终端用户。
4. 入口脚本接收应用主体传来的退出状态并完成请求的处理。