Other languages: 繁體中文

Controller - Introduction

The controller is very important in web application development. This is where we will define our URLs, which HTTP methods are allowed, which filters will be applied and how requests will be processed and responded to. The drogon framework has helped us to handle the network transmission, Http protocol analysis and so on. We only need to pay attention to the logic of the controller; each controller object can have one or more processing functions (generally called handlers), and the interface of the function is generally defined as follows:

```
Void handlerName(const HttpRequestPtr &req,
                  std::function<void (const HttpResponsePtr &)> &&callback,
                 ...);
```

Where `req` is the object of the Http request (wrapped by the smart pointer), the `callback` is the callback function object that the framework passes to the controller, and the controller generates the response object (also wrapped by the smart pointer) and then passes the object to the drogon through the callback. Then the framework will send the response content to the browser for you. The last part `...` is a list of parameters. The drogon maps the parameters in the Http request to the corresponding parameter parameters according to the mapping rules. This is very convenient for application development.

Obviously, this is an asynchronous interface, one can call the callback after completing the time-consuming operation at other threads;

Drogon have three types controllers, HttpSimpleController, HttpController, and WebSocketController. When you use them, the corresponding class template needs to be inherited. For example, a custom class "MyClass" declaration of HttpSimpleController is as follows:

```
class MyClass:public drogon::HttpSimpleController<MyClass>
{
public:
    //TestController(){}
    virtual void asyncHandleHttpRequest(const HttpRequestPtr &req,
                                         std::function<void (const
HttpResponsePtr &)> &&callback) override;

    PATH_LIST_BEGIN
    PATH_ADD("/json");
    PATH_LIST_END
};
```

Controller life cycle

A controller registered to a drogon framework will have at most only one instance and will not be destroyed during the entire application run, so users can declare and use member variables in the controller class. Note that when the handler of the controller is called, it is in a multi-threaded environment (when the number of IO threads of the framework is configured to be greater than 1), if you need to access non-temporary variables, please do the concurrent protection work.

Next: HttpSimpleController