Other languages: 繁體中文

References Request

The HttpRequest type pointer commonly named `req` in the examples in this documentation represents the data contained in a request received or sent by drogon, below are the some methods by which you can interact with this object:

- `isOnSecureConnection()`

  Summary

  Function that returns if the request was made on https.

  Inputs

  None.

  Returns

  bool type.

- `getMethod()`

  Summary

  Function that returns the request method. Useful to differentiate the request method if a single handle allows more than one type.

  Inputs

  None.

  Returns

  `HttpMethod` request method object.

  Examples

```
#include "mycontroller.h"

using namespace drogon;

void mycontroller::anyhandle(const HttpRequestPtr &req,
std::function<void (const HttpResponsePtr &)> &&callback) {
    if (req->getMethod() == HttpMethod::Get) {
      // do something
```

```
    } else if (req->getMethod() == HttpMethod::Post) {
      // do other something
    }
}
```

- `getParameter(const std::string &key)`

Summary

Function that returns the value of a parameter based on an identifier. The behavior changes based on the Get or Post request type.

Inputs

string type identifier of param.

Returns

param content on `string` format.

Examples

On Get type:

```
#include "mycontroller.h"
#include <string>

using namespace drogon;

void mycontroller::anyhandle(const HttpRequestPtr &req,
std::function<void (const HttpResponsePtr &)> &&callback) {
    // https://mysite.com/an-path/?id=5
    std::string id = req->getParameter("id");
    // or
    long id = std::strtol(req->getParameter("id"));
}
```

Or On Post type:

```
#include "mycontroller.h"
#include <string>

using namespace drogon;

void mycontroller::loginHandle(const HttpRequestPtr &req,
std::function<void (const HttpResponsePtr &)> &&callback) {
    // request contain a Form Login
    std::string email = req->getParameter("email");
```

```
    std::string password = req->getParameter("password");
}
```

- `getPath()`

  Similar

  path()

  Summary

  Function that returns the request path. Useful if you use an ADD_METHOD_VIA_REGEX or other type of dynamic URL in the controller.

  Inputs

  None.

  Returns

  string representing the request path.

  Examples

```
#include "mycontroller.h"
#include <string>

using namespace drogon;

void mycontroller::anyhandle(const HttpRequestPtr &req,
std::function<void (const HttpResponsePtr &)> &&callback) {
    // https://mysite.com/an-path/?id=5
    std::string url = req->getPath();

    // url = /an-path/
}
```

- `getBody()`

  Similar

  body()

  Summary

  Function that returns the request body content (if any).

Inputs

None.

Returns

String representing the request body (if any).

- getHeader(std::string key)

Summary

Function that returns a request header based on an identifier.

Inputs

String header identifier.

Returns

The content of the header in string format.

Examples

```
#include "mycontroller.h"
#include <string>

using namespace drogon;

void mycontroller::anyhandle(const HttpRequestPtr &req,
std::function<void (const HttpResponsePtr &)> &&callback) {
    if (req->getHeader("Host") != "mysite.com") {
      // return http 403
    }
}
```

- headers()

Summary

Function that returns all headers of a request.

Inputs

None.

Returns

An unordered_map containing the headers.

Examples

```
#include "mycontroller.h"
#include <unordered_map>
#include <string>

using namespace drogon;

void mycontroller::anyhandle(const HttpRequestPtr &req,
std::function<void (const HttpResponsePtr &)> &&callback) {
    for (const std::pair<const std::string, const std::string> &header
: req->headers()) {
      auto header_key = header.first;
      auto header_value = header.second;
    }
}
```

- `getCookie()`

Summary

Function that returns request cookie based on an identifier.

Inputs

None.

Returns

Value of cookie on string format.

- `cookies()`

Summary

Function that returns all cookies of a request.

Inputs

None.

Returns

An unordered_map containing the cookies.

Examples

```
#include "mycontroller.h"
#include <unordered_map>
#include <string>

using namespace drogon;

void mycontroller::anyhandle(const HttpRequestPtr &req,
std::function<void (const HttpResponsePtr &)> &&callback) {
    for (const std::pair<const std::string, const std::string> &header
: req->cookies()) {
      auto cookie_key = header.first;
      auto cookie_value = header.second;
    }
}
```

- `getJsonObject()`

Summary

Function that converts the body value of a request into a Json object (normally POST requests).

Inputs

None.

Returns

A Json object.

Examples

```
#include "mycontroller.h"

using namespace drogon;

void mycontroller::anyhandle(const HttpRequestPtr &req,
std::function<void (const HttpResponsePtr &)> &&callback) {
    // body = {"email": "test@gmail.com"}
    auto jsonData = *req->getJsonObject();

    std::string email = jsonData["email"].asString();
}
```

Useful Things

From here are not methods of the Http Request object, but some useful things you can do to process the requests you will receive

Parsing File Request

```
#include "mycontroller.h"

using namespace drogon;

void mycontroller::postfile(const HttpRequestPtr &req, std::function<void
(const HttpResponsePtr &)> &&callback) {
    // Only Post Requests (File Form)

    MultiPartParser file;
    file.parse(req);

    if (file.getFiles().empty()) {
      // Not Files Found
    }

    // Get First file and save then
    const HttpFile archive = file.getFiles()[0];
    archive.saveAs("/tmp/" + archive.getFileName());
  }
```

For more information about parsing file: File Handler

Next: File Handler