檔案處理器

原文：ENG-09-1-File-Handler.md

檔案解析是指將 multipart-data POST 請求中的檔案（或多個檔案）透過 `MultiPartParser` 物件解析為 `HttpFile` 物件，以下為相關說明：

`MultiPartParser` 物件

MultiPartParser 物件說明

用於解析並暫存請求中的檔案。

- `parse(const std::shared_ptr<HttpRequest> &req)`

  parse() 說明

接收請求物件參數，讀取並識別檔案（如有），並轉存至 MultiPartParser 物件。

parse() 範例

```
  #include "mycontroller.h"

  using namespace drogon;

  void mycontroller::postfile(const HttpRequestPtr &req, std::function<void
(const HttpResponsePtr &)> &&callback) {
      // 僅限 Post 請求（檔案表單）

      MultiPartParser fileParser;
      fileParser.parse(req);
  }
```

- `getFiles()`

  說明

  必須在 `parse()` 之後呼叫，回傳請求中的檔案，型態為 `std::vector<HttpFile>`。

  範例

```
   #include "mycontroller.h"

   using namespace drogon;

   void mycontroller::postfile(const HttpRequestPtr &req,
 std::function<void (const HttpResponsePtr &)> &&callback) {
```

```
        // 僅限 Post 請求（檔案表單）

        MultiPartParser fileParser;
        fileParser.parse(req);

        // 檢查是否有檔案
        if (fileParser.getFiles().empty()) {
            // 未找到檔案
        }

        size_t num_of_files = fileParser.getFiles().size();
    }
```

- `getParameters()`

  getParameters() 說明

必須在 `parse()` 之後呼叫，回傳 MultiPartData 表單中的其他欄位。

getParameters() 回傳

`std::unordered_map<std::string, std::string>`（key, value）

getParameters() 範例

```
  #include "mycontroller.h"

  using namespace drogon;

  void mycontroller::postfile(const HttpRequestPtr &req, std::function<void
(const HttpResponsePtr &)> &&callback) {
      // 僅限 Post 請求（檔案表單）

      MultiPartParser fileParser;
      fileParser.parse(req);

      if (!fileParser.getFiles().empty()) {
          for (const auto &header : fileParser.getParameters()){
              header.first // 表單欄位名稱
              header.second // 欄位值
          }
      }
  }
```

- `getParameter<T>(const std::string &key)`

  getParameter() 說明

必須在 `parse()` 之後呼叫，取得指定 key 的單一欄位值。

getParameter() 輸入

欲取得的型別（自動轉型）、參數 key。

getParameter() 回傳

指定 key 的參數內容（型別自動轉換），若不存在則回傳 `T` 型別預設值。

getParameter() 範例

```
  #include "mycontroller.h"

  using namespace drogon;

  void mycontroller::postfile(const HttpRequestPtr &req, std::function<void
(const HttpResponsePtr &)> &&callback) {
      // 僅限 Post 請求（檔案表單）

      MultiPartParser fileParser;
      fileParser.parse(req);

      std::string email = fileParser.getParameter<std::string>
("email_form");

      // string 預設值為 ""
      if (email.empty()) {
          // email_form 未找到
      }
  }
```

`HttpFile` 物件

HttpFile 物件說明

代表記憶體中的檔案，由 `MultiPartParser` 使用。

- `getFileName()`

  getFileName() 說明

取得收到檔案的原始檔名。

getFileName() 回傳

std::string。

getFileName() 範例

```
  #include "mycontroller.h"

  using namespace drogon;

  void mycontroller::postfile(const HttpRequestPtr &req, std::function<void
(const HttpResponsePtr &)> &&callback) {
      // 僅限 Post 請求（檔案表單）

      MultiPartParser fileParser;
      fileParser.parse(req);

      std::string filename = fileParser.getFiles()[0].getFileName();
  }
```

- `fileLength()`

  fileLength() 說明

取得檔案大小。

fileLength() 回傳

`size_t`

fileLength() 範例

```
  #include "mycontroller.h"

  using namespace drogon;

  void mycontroller::postfile(const HttpRequestPtr &req, std::function<void
(const HttpResponsePtr &)> &&callback) {
      // 僅限 Post 請求（檔案表單）

      MultiPartParser fileParser;
      fileParser.parse(req);

      size_t filesize = fileParser.getFiles()[0].fileLength();
  }
```

- `getFileExtension()`

  getFileExtension() 說明

取得檔案副檔名。

getFileExtension() 回傳

std::string

getFileExtension() 範例

```
  #include "mycontroller.h"

  using namespace drogon;

  void mycontroller::postfile(const HttpRequestPtr &req, std::function<void
(const HttpResponsePtr &)> &&callback) {
      // 僅限 Post 請求（檔案表單）

      MultiPartParser fileParser;
      fileParser.parse(req);

      std::string file_extension = fileParser.getFiles()
[0].getFileExtension();
  }
```

- `getMd5()`

  getMd5() 說明

取得檔案 MD5 雜湊值，可用於檢查完整性。

getMd5() 回傳

std::string

- `save()`

  save() 說明

儲存檔案至檔案系統。儲存目錄為 config.json（或等效設定）中的 `UploadPath`。完整路徑如下：

```
drogon::app().getUploadPath()+"/"+this->getFileName()
```

或簡化為：UploadPath/檔名

- `save(const std::string &path)`

  save(path) 說明

傳入 path 參數時，使用指定路徑儲存檔案而非 UploadPath。

save(path) 範例

```
  #include "mycontroller.h"

  using namespace drogon;

  void mycontroller::postfile(const HttpRequestPtr &req, std::function<void
(const HttpResponsePtr &)> &&callback) {
      // 僅限 Post 請求（檔案表單）

      MultiPartParser fileParser;
      fileParser.parse(req);

      // 相對路徑
      fileParser.getFiles()[0].save("./"); // 儲存至伺服器同目錄，檔名不變

      // 絕對路徑
      fileParser.getFiles()[0].save("/home/user/downloads/"); // 儲存至指定目
錄，檔名不變
  }
```

- `saveAs(const std::string &path)`

  saveAs(path) 說明

以新檔名儲存檔案（忽略原始檔名）。

saveAs(path) 範例

```
  #include "mycontroller.h"

  using namespace drogon;

  void mycontroller::postfile(const HttpRequestPtr &req, std::function<void
(const HttpResponsePtr &)> &&callback) {
      // 僅限 Post 請求（檔案表單）

      MultiPartParser fileParser;
      fileParser.parse(req);

      // 相對路徑
      fileParser.getFiles()[0].saveAs("./image.png"); // 儲存至伺服器同目錄，
檔名自訂
      /* 僅為範例，實務勿直接覆蓋檔案格式，應先檢查是否真為 png */

      // 絕對路徑
      fileParser.getFiles()[0].save("/home/user/downloads/anyname." +
fileParser.getFiles()[0].getFileExtension());
  }
```

下一步: 插件