**Documentation Zia**

Le but du projet Zia est de créer un serveur HTTP. Ce serveur sera capable de servir des documents HTTP typiques et des requêtes de pages, ainsi que l'exécution de CGI et plus encore. Le serveur doit être écrit en C++, avec un support pour les modules interopérables.

Classes et fonctions de notre Zia

class ConfigManager

ConfigManager();

ConfigManager(std::string path)

~ConfigManager()

std::vector<std::string> split(std::string s, std::string delimiter)

void Init()

void Reload()

void Print()

std::list<std::pair<std::string, std::string>> GetList()

void start()

std::string getIn(std::string search)

std::filesystem::file\_time\_type getDate()

void checkRightConfig()

bool isAlive()

class Module

Module(std::string name, std::string path)

~Module()

std::string getName()

std::string getPath()

std::string executeCmd(std::string cmd)

class ModuleManager

ModuleManager(std::string list\_module, std::string module\_path, std::filesystem::file\_time\_type date)

~ModuleManager()

int InitListModule(std::string list\_module, std::string module\_path)

int addModule(std::string name, std::string path)

int removeModule(std::string name)

std::list<Module \*> getModuleList()

std::vector<std::string> split(std::string s, std::string delimiter)

void Print()

int Count()

int CheckIfModif(std::filesystem::file\_time\_type check, std::string list\_module, std::string module\_path)

int Reload(std::string list\_module, std::string module\_path)

class Execute

Execute()

~Execute()

std::string run(const std::string &cmd, const std::unordered\_map<std::string, std::string> &test\_env)

void openfd()

void startCmd()

void exeCmd()

void getResult()

void closeHost()

void waitCmd() const

char \*\*generateEnv(const std::unordered\_map<std::string, std::string> &envMap) const

std::vector<std::string> split(std::string s, std::string delimiter)

std::string readFile(std::string path)

class ASocket

~ASocket();

virtual void CreateSock()

virtual void DeleteSock(int cock)

class Client

Client(int sock)

~Client()

void setPhp(Module \*php)

void SendHttp(std::string msg)

std::string Receive()

void ParseRequest()

void GetMethod()

void PostMethod()

std::vector<std::string> split(std::string s, std::string delimiter)

class IClient

virtual ~IClient()

virtual void SendHttp(std::string msg)

virtual std::string Receive()

virtual void ParseRequest()

virtual void GetMethod()

virtual void PostMethod()

class Server

erver()

Server(int port)

~Server()

void setPort(int port)

void startServer()

void acceptCli()

void setModule(std::list<Module \*> module\_list)

void start\_thread(int tmp\_cli)

Module \*getPhp()

class Socket

Socket()

~Socket()

void CreateSock()

void DeleteSock(int cock)

class IThread

virtual ~IThread()

virtual void lockThread()

virtual void unlockThread()

virtual bool isAlive()

virtual void waitEnd()

virtual void run()

class Zia

Zia()

~Zia()

int Core()

void start\_thread()