[Apache 的hook 一览](http://blog.csdn.net/wind_cludy/article/details/6680973)

看了一个日本人总结的Apache的钩子函数一览，我觉得对于Apache模块开发的童鞋挺有用的，就搬过来了。

本章的重要内容：

关于hook的调用顺序

钩子函数的调用顺序是：「设定的初期化」「进程的初期化」「连接」「请求」

关于RUN\_ALL，RUN\_FIRST

RUN\_FIRST  类型的hook 是：被调用的 hook函数的返回值为 OK 或者是）时，后面的hook是不被执行的。

RUN\_ALL 类型的hook 是：被调用的 hook函数的返回值不为）时，已经加载的hook将被全部执行。

关于hook 登録函数 ap\_hook\_xxx() 的参数

ap\_hook\_xxx() 的参数全部相同．具体的请参照下面．

**Index**

|  |  |
| --- | --- |
| **設定初期化** | |
| RUN\_ALL | [ap\_hook\_pre\_config()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_pre_config) |
| RUN\_ALL | [ap\_hook\_open\_logs()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_open_logs) |
| RUN\_ALL | [ap\_hook\_post\_config()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_post_config) |
| VOID | [ap\_hook\_optional\_fn\_retrieve()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_optional_fn_retrieve) |
| **进程初期化** | |
| RUN\_ALL | [ap\_hook\_pre\_mpm()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_pre_mpm) |
| VOID | [ap\_hook\_child\_init()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_child_init) |
| **连接** | |
| RUN\_FIRST | [ap\_hook\_create\_connection()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_create_connection) |
| RUN\_ALL | [ap\_hook\_pre\_connection()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_pre_connection) |
| RUN\_FIRST | [ap\_hook\_process\_connection()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_process_connection) |
| **请求（Request）** | |
| RUN\_ALL | [ap\_hook\_create\_request()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_create_request) |
| RUN\_ALL | [ap\_hook\_post\_read\_request()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_post_read_request) |
| RUN\_FIRST | [ap\_hook\_quick\_handler()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_quick_handler) |
| RUN\_FIRST | [ap\_hook\_translate\_name()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_translate_name) |
| RUN\_FIRST | [ap\_hook\_map\_to\_storage()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_map_to_storage) |
| RUN\_ALL | [ap\_hook\_header\_parser()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_header_parser) |
| RUN\_ALL | [ap\_hook\_access\_checker()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_access_checker) |
| RUN\_FIRST | [ap\_hook\_check\_user\_id()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_check_user_id) |
| RUN\_FIRST | [ap\_hook\_auth\_checker()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_auth_checker) |
| RUN\_FIRST | [ap\_hook\_type\_checker()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_type_checker) |
| RUN\_ALL | [ap\_hook\_fixups()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_fixups) |
| VOID | [ap\_hook\_insert\_filter()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_insert_filter) |
| RUN\_FIRST | [ap\_hook\_handler()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_handler) |
| RUN\_ALL | [ap\_hook\_log\_transaction()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_log_transaction) |
| **出错 ERRor** | |
| VOID | [ap\_hook\_insert\_error\_filter()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_insert_error_filter) |
| VOID | [ap\_hook\_error\_log()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_error_log) |
| **必要時** | |
| RUN\_FIRST | [ap\_hook\_default\_port()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_default_port) |
| RUN\_FIRST | [ap\_hook\_http\_method()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_http_method) |
| **随时被调用?** | |
| RUN\_ALL | [ap\_hook\_fatal\_exception()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_fatal_exception) |
| RUN\_ALL | [ap\_hook\_get\_mgmt\_items()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_get_mgmt_items) |
| RUN\_FIRST | [ap\_hook\_uexec\_identity()](http://dsas.blog.klab.org/archives/50626863.html#ap_hook_uexec_identity) |

**説明**

|  |  |
| --- | --- |
| **ap\_hook\_pre\_config()** | RUN\_ALL |
| Run the pre\_config function for each module | |
| |  |  | | --- | --- | | int x\_pre\_config(apr\_pool\_t \*pconf,  apr\_pool\_t \*plog,  apr\_pool\_t \*ptemp) | | | *return* | OK or DECLINED on success anything else is a error | | pconf | The config pool | | plog | The logging streams pool | | ptemp | The temporary pool | | |

|  |  |
| --- | --- |
| **ap\_hook\_open\_logs()** | RUN\_ALL |
| Run the open\_logs functions for each module | |
| |  |  | | --- | --- | | int x\_open\_logs(apr\_pool\_t \*pconf,  apr\_pool\_t \*plog,  apr\_pool\_t \*ptemp,  server\_rec \*s) | | | *return* | OK or DECLINED on success anything else is a error | | pconf | The config pool | | plog | The logging streams pool | | ptemp | The temporary pool | | s | The list of server\_recs | | |

|  |  |
| --- | --- |
| **ap\_hook\_post\_config()** | RUN\_ALL |
| Run the post\_config function for each module | |
| |  |  | | --- | --- | | int x\_post\_config(apr\_pool\_t \*pconf,  apr\_pool\_t \*plog,  apr\_pool\_t \*ptemp,  server\_rec \*s) | | | *return* | OK or DECLINED on success anything else is a error | | pconf | The config pool | | plog | The logging streams pool | | ptemp | The temporary pool | | s | The list of server\_recs | | |

|  |
| --- |
| **ap\_hook\_optional\_fn\_retrieve()** |
| Retrieve the optional functions for each module. This is run immediately before the server starts. Optional functions should be registered during the hook registration phase. | |
| |  | | --- | | void x\_optional\_fn\_retrieve(void) | | |

|  |  |
| --- | --- |
| **ap\_hook\_pre\_mpm()** | RUN\_ALL |
| Hook for post scoreboard creation, pre mpm. | |
| |  |  | | --- | --- | | int x\_pre\_mpm(apr\_pool\_t \*p,  ap\_scoreboard\_e sb\_type) | | | *return* | OK or DECLINE on success; anything else is a error | | p | Apache pool to allocate from. | | sb\_type |  | | |

|  |
| --- |
| **ap\_hook\_child\_init()** |
| Run the child\_init functions for each module | |
| |  |  | | --- | --- | | void x\_child\_init(apr\_pool\_t \*pchild,  server\_rec \*s) | | | pchild | The child pool | | s | The list of server\_recs in this server | | |

|  |  |
| --- | --- |
| **ap\_hook\_create\_connection()** | RUN\_FIRST |
| create\_connection is a RUN\_FIRST hook which allows modules to create connections. In general, you should not install filters with the create\_connection hook. If you require vhost configuration information to make filter installation decisions, you must use the pre\_connection or install\_network\_transport hook. This hook should close the connection if it encounters a fatal error condition. | |
| |  |  | | --- | --- | | conn\_rec \* x\_create\_connection(apr\_pool\_t \*p,  server\_rec \*server,  apr\_socket\_t \*csd,  long conn\_id,  void \*sbh,  apr\_bucket\_alloc\_t \*alloc) | | | *return* | An allocated connection record or NULL. | | server |  | | p | The pool from which to allocate the connection record | | csd | The socket that has been accepted | | conn\_id | A unique identifier for this connection. The ID only needs to be unique at that time, not forever. | | sbh | A handle to scoreboard information for this connection. | | alloc |  | | |

|  |  |
| --- | --- |
| **ap\_hook\_pre\_connection()** | RUN\_ALL |
| This hook gives protocol modules an opportunity to set everything up before calling the protocol handler. All pre-connection hooks are run until one returns something other than ok or decline [Apache の付属文書](http://httpd.apache.org/docs/2.0/developer/modules.html)によれば，accept()直後に呼ばれる | |
| |  |  | | --- | --- | | int x\_pre\_connection(conn\_rec \*c,  void \*csd) | | | *return* | OK or DECLINED | | c | The connection on which the request has been received. | | csd | The mechanism on which this connection is to be read. Most times this will be a socket, but it is up to the module that accepts the request to determine the exact type. | | |

|  |  |
| --- | --- |
| **ap\_hook\_process\_connection()** | RUN\_FIRST |
| This hook implements different protocols. After a connection has been established, the protocol module must read and serve the request. This function does that for each protocol module. The first protocol module to handle the request is the last module run. | |
| |  |  | | --- | --- | | int x\_process\_connection(conn\_rec \*c) | | | *return* | OK or DECLINED | | c | The connection on which the request has been received. | | |

|  |  |
| --- | --- |
| **ap\_hook\_create\_request()** | RUN\_ALL |
| Gives modules a chance to create their request\_config entry when the request is created. | |
| |  |  | | --- | --- | | int x\_create\_request(request\_rec \*r) | | | *return* | OK or DECLINED | | r | The current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_post\_read\_request()** | RUN\_ALL |
| This hook allows modules to affect the request immediately after the request has been read, and before any other phases have been processes. This allows modules to make decisions based upon the input header fields | |
| |  |  | | --- | --- | | int x\_post\_read\_request(request\_rec \*r) | | | *return* | OK or DECLINED | | r | The current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_quick\_handler()** | RUN\_FIRST |
| Run the quick handler functions for each module. The quick\_handler is run before any other requests hooks are called (location\_walk, directory\_walk, access checking, et. al.). This hook was added to provide a quick way to serve content from a URI keyed cache. [Apache の付属文書](http://httpd.apache.org/docs/2.0/developer/modules.html)によれば，キャッシュモジュールが使用している． | |
| |  |  | | --- | --- | | int x\_quick\_handler(request\_rec \*r,  int lookup\_uri) | | | r | The request\_rec | | lookup\_uri | Controls whether the caller actually wants content or not. lookup is set when the quick\_handler is called out of ap\_sub\_req\_lookup\_uri() | | |

|  |  |
| --- | --- |
| **ap\_hook\_translate\_name()** | RUN\_FIRST |
| This hook allow modules an opportunity to translate the URI into an actual filename. If no modules do anything special, the server's default rules will be followed. | |
| |  |  | | --- | --- | | int x\_translate\_name(request\_rec \*r) | | | *return* | OK, DECLINED, or HTTP\_... | | r | The current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_map\_to\_storage()** | RUN\_FIRST |
| This hook allow modules to set the per\_dir\_config based on their own context (such as sections) and responds to contextless requests such as TRACE that need no security or filesystem mapping. based on the filesystem. | |
| |  |  | | --- | --- | | int x\_map\_to\_storage(request\_rec \*r) | | | *return* | DONE (or HTTP\_) if this contextless request was just fulfilled (such as TRACE), OK if this is not a file, and DECLINED if this is a file. The core map\_to\_storage (HOOK\_RUN\_REALLY\_LAST) will directory\_walk and file\_walk the r->filename. | | r | The current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_header\_parser()** | RUN\_ALL |
| Run the header parser functions for each module | |
| |  |  | | --- | --- | | int x\_header\_parser(request\_rec \*r) | | | *return* | OK or DECLINED | | r | The current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_access\_checker()** | RUN\_ALL |
| This hook is used to apply additional access control to this resource. It runs \*before\* a user is authenticated, so this hook is really to apply additional restrictions independent of a user. It also runs independent of 'Require' directive usage. | |
| |  |  | | --- | --- | | int x\_access\_checker(request\_rec \*r) | | | *return* | OK, DECLINED, or HTTP\_... | | r | the current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_check\_user\_id()** | RUN\_FIRST |
| This hook is used to analyze the request headers, authenticate the user, and set the user information in the request record (r->user and r->ap\_auth\_type). This hook is only run when Apache determines that authentication/authorization is required for this resource (as determined by the 'Require' directive). It runs after the access\_checker hook, and before the auth\_checker hook. | |
| |  |  | | --- | --- | | int x\_check\_user\_id(request\_rec \*r) | | | *return* | OK, DECLINED, or HTTP\_... | | r | The current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_auth\_checker()** | RUN\_FIRST |
| This hook is used to check to see if the resource being requested is available for the authenticated user (r->user and r->ap\_auth\_type). It runs after the access\_checker and check\_user\_id hooks. Note that it will \*only\* be called if Apache determines that access control has been applied to this resource (through a 'Require' directive). | |
| |  |  | | --- | --- | | int x\_auth\_checker(request\_rec \*r) | | | *return* | OK, DECLINED, or HTTP\_... | | r | the current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_type\_checker()** | RUN\_FIRST |
| This routine is called to determine and/or set the various document type information bits, like Content-type (via r->content\_type), language, et cetera. | |
| |  |  | | --- | --- | | int x\_type\_checker(request\_rec \*r) | | | *return* | OK, DECLINED, or HTTP\_... | | r | the current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_fixups()** | RUN\_ALL |
| Allows modules to perform module-specific fixing of header fields. This is invoked just before any content-handler | |
| |  |  | | --- | --- | | int x\_fixups(request\_rec \*r) | | | *return* | OK, DECLINED, or HTTP\_... | | r | The current request | | |

|  |
| --- |
| **ap\_hook\_insert\_filter()** |
| This hook allows modules to insert filters for the current request | |
| |  |  | | --- | --- | | void x\_insert\_filter(request\_rec \*r) | | | r | the current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_handler()** | RUN\_FIRST |
| Run the handler functions for each module **non-wildcard handlers should HOOK\_MIDDLE, wildcard HOOK\_LAST** | |
| |  |  | | --- | --- | | int x\_handler(request\_rec \*r) | | | r | The request\_rec | | |

|  |  |
| --- | --- |
| **ap\_hook\_log\_transaction()** | RUN\_ALL |
| This hook allows modules to perform any module-specific logging activities over and above the normal server things. | |
| |  |  | | --- | --- | | int x\_log\_transaction(request\_rec \*r) | | | *return* | OK, DECLINED, or HTTP\_... | | r | The current request | | |

|  |
| --- |
| **ap\_hook\_insert\_error\_filter()** |
| This hook allows modules to insert filters for the current error response | |
| |  |  | | --- | --- | | void x\_insert\_error\_filter(request\_rec \*r) | | | r | the current request | | |

|  |
| --- |
| **ap\_hook\_error\_log()** |
|  | |
| |  | | --- | | void x\_error\_log(const char \*file,  int line,  int level,  apr\_status\_t status,  const server\_rec \*s,  const request\_rec \*r,  apr\_pool\_t \*pool,  const char \*errstr) | | |

|  |  |
| --- | --- |
| **ap\_hook\_default\_port()** | RUN\_FIRST |
| Return the default port from the current request | |
| |  |  | | --- | --- | | apr\_port\_t x\_default\_port(const request\_rec \*r) | | | *return* | The current port | | r | The current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_http\_method()** | RUN\_FIRST |
| This hook allows modules to retrieve the http method from a request. This allows Apache modules to easily extend the methods that Apache understands ここで言う"method" は scheme の間違いのようです．実際，Apache 2.2 では ap\_hook\_http\_scheme() に変更されています． | |
| |  |  | | --- | --- | | const char \* x\_http\_method(const request\_rec \*r) | | | *return* | The http method from the request | | r | The current request | | |

|  |  |
| --- | --- |
| **ap\_hook\_fatal\_exception()** | RUN\_ALL |
|  | |
| |  | | --- | | int x\_fatal\_exception(ap\_exception\_info\_t \*ei) | | |

|  |  |
| --- | --- |
| **ap\_hook\_get\_mgmt\_items()** | RUN\_ALL |
| This hook provdes a way for modules to provide metrics/statistics about their operational status. | |
| |  |  | | --- | --- | | int x\_get\_mgmt\_items(apr\_pool\_t \*p,  const char \* val,  apr\_hash\_t \*ht) | | | p | A pool to use to create entries in the hash table | | val | The name of the parameter(s) that is wanted. This is tree-structured would be in the form ('\*' is all the tree, 'module.\*' all of the module , 'module.foo.\*', or 'module.foo.bar' ) | | ht | The hash table to store the results. Keys are item names, and the values point to ap\_mgmt\_item\_t structures. | | |

|  |  |
| --- | --- |
| **ap\_hook\_get\_suexec\_identity()** | RUN\_FIRST |
|  | |
| |  | | --- | | ap\_unix\_identity\_t \* x\_get\_suexec\_identity(const request\_rec \*r) | | |