

HTTPOcalypse

master-httpocalypse-juc #78857

**Ask your questions
any time!**

Root issues

- **Multiple rich scopes**
 - base (controllers)
 - web (RPCs)
 - portal/website (web pages)
- **Knowledge spread across multiple teams**
 - MetastORM
 - JS Framework
 - Website

Symptoms

- **Conflicting APIs**
 - type="http" vs type="json"
 - Base ir.http vs Website ir.http
- **Technical dept**
 - Unclear ORM initialization
 - Many ir.http _dispatch() overrides
 - Glorified request attributes
- **Bad error reporting**
 - <https://www.odoo.com/r/llkS>

Consequences

- **Websocket anyone?**
- **Lang in URL madness**
- **RPC over RPC over HTTP**
- **Payment webhooks anyone?**

HTTP Apocalypse

Implementation

call → serve → dispatch → endpoint

- **Application.__call__(environ, start_response)**
 - WSGI entrypoint
 - Create and expose Request()
 - Determine if static or nodb or db
 - Error reporting

__call__ → serve → dispatch → endpoint

- Request._serve_static()
 - Match a route /<module>/static/<path>
 - Stream the file from the file system

__call__ → serve → **dispatch** → **endpoint**

- **Request._serve_nodb()**
 - When database cookie is missing
 - Actually not a cookie by itself, the info is concatenated to the session-id
 - Match @route(auth=None)
 - Doesn't connect to a DB/doesn't init the ORM
 - Doesn't use ir.http
 - Continue with a type-specialized dispatcher

__call__ → serve → dispatch → endpoint

- Request. serve_db() & serve_ir_http()
 - When database cookie is set
 - Open a cursor to the database, setup the registry
 - Load the session from the database
 - Model ir.session {sid, data}
 - Data as JSON: {context, debug, login, uid, session_token}
 - Setup the environment using session uid and ctx
 - Call `retrying(_serve_ir_http)`

__call__ → serve → dispatch → endpoint

- Request._serve_db() & serve ir http()
 - Delegate to ir.http:
 - _match()
 - _authenticate()
 - _pre_dispatch()
 - Continue with a type-specialized dispatcher

__call__ → **serve** → **dispatch** → **endpoint**

- **HttpDispatcher.dispatch()**
 - Load Request.params from:
 - the URL
 - the query-string
 - the html form
 - the html files
 - Verify CSRF

__call__ → **serve** → **dispatch** → **endpoint**

- **JsonDispatcher.dispatch()**

- Lobotomized JSON-RPC2 implementation
 - Ignore the “jsonrpc” and “method” keys
 - Only support params by name
 - Response/errors are well supported
- Load Request.params from
 - The JSON “params” object in the body
- Should be named type=“jsonrpc”
- Impossible to send non-jsonrpc json data :(

__call__ → serve → dispatch → endpoint

- The @route decorated method

ir.http base

- **Changed `_dispatch()` method**
 - Still wrap the call to the endpoint
 - No longer the entrypoint of `ir.http`
 - Responsibility moved to `http.py`:
`Request._serve_ir_http`
- **`_postprocess_args` → `_pre_dispatch`**

ir.http web

- A lot of small changes related to the various logins (/web/login, totp, oauth)
- `session.authenticate()` does no longer change `request.env.cr`, the request's cr stays open in the original database

ir.http http_routing/portal/website

Showing 16 changed files with 457 additions and 468 deletions.

458 addons/http_routing/models/ir_http.py

[Load diff](#)

Large diffs are not rendered by default.



ir.http http_routing/portal/website

- **Complete refactor**
 - No more `_dispatch()` override black magic
 - Lang in URL moved to `_match()`
- **`_match()` vs `_serve_fallback()`**
 - Website pages are not registered in the router
 - `_authenticate()` / `_pre_dispatch()` ?
- **`_frontend_pre_dispatch()`**
 - For both `_serve_fallback`, `@route(website=True)` and `_handle_error`

ir.http http_routing/portal/website

- **Read the commit message!**
 - It explains the functional challenges of the three modules
 - It explains the previous implementation
 - It shows how that previous implem. clashes with the HTTPocalypse
 - It details the proposed changes
- **We are quite proud of the result**

HTTP Apocalypse

API

ir.http match

- `_match(path) → (rule, args)`
 - Use the werkzeug router to match an endpoint given a path.
- *When to override ?*
 - *You need to modify the request path before you match a backend route and you cannot redirect (3xx response) the user on that URL.*

ir.http authenticate

- `_authenticate(endpoint)`
 - Verify the user is logged in (`auth==='user'`)
 - Grant the public user in case the user is not logged in (`auth='public'`)
- *When to override ?*
 - *You don't override it, you provide a new `_auth_method_x()` authentication method to use with `@route(auth='x')`*

ir.http Pre Dispatch

- `_pre_dispatch(rule, args)`
 - Called when a backend endpoint matched
 - Used to prepare the system to handle the request
- *When to override ?*
 - *You need to save some options from the query-string in the session, in the context or in the response cookies before serving a backend endpoint.*

ir.http Frontend Pre Dispatch

- `_frontend_pre_dispatch()`
 - Called when you serve a website page or a `@route(website=True)` backend endpoint
 - Used to prepare the system to handle the request
- *When to override ?*
 - *You need to save some options from the query-string in the session, in the context or in the response cookies before serving a website page.*

ir.http Serve Fallback

- `_serve_fallback()`
 - Called when no endpoint matched
 - Used to serve content unreachable by the router
- *When to override ?*
 - *This is discouraged, the user is not authenticated yet*
 - *You need to deliver a new kind of content, content that cannot be served nor via the static files, nor via a backend endpoint nor via website but you need Odoo.*

Thank you!