HttpCache

Content Cache

eZ Platform uses Symfony HttpCache to manage content "view" cache with an expiration model. In addition it is extended *(using FOSHttpCache)* to add several advanced features. For content coming from the CMS the following is taken advantage of out of the box:

- To be able to always keep cache up to date, cache is made "content-aware" to allow updates to content to trigger *cache invalidation.*
  - Uses a custom `X-Location-Id` header, which both Symfony and Varnish Proxy are able to invalidate cache on *(for details see cache purge document.)*
- To be able to also cache requests by logged-in users, cache is made "Context-Aware."
  - Uses a custom vary header `X-User-Hash` to allow pages to var by user rights *(so not per unique user, that is better served by browser cache.)*

Cache and Expiration Configuration

ezplatform.yml

```
ezpublish:
    system:
        my_siteaccess:
            content:
                view_cache: true      # Activates HttpCache for content
                ttl_cache: true       # Activates expiration based HttpCache for
content (very fast)
                default_ttl: 60       # Number of seconds an Http response is valid in
cache (if ttl_cache is true)
```

Making your controller response content-aware

Sometimes you need your controller's cache to be invalidated at the same time as specific content changes (i.e. ESI sub-requests with `render` twig helper, for a menu for instance). To be able to do that, you just need to add **`X-Location-Id`** header to the response object:

```
use Symfony\Component\HttpFoundation\Response;

// Inside a controller action
// "Connects" the response to location #123 and sets a max age (TTL) of 1 hour.
$response = new Response();
$response->headers->set('X-Location-Id', 123);
$response->setSharedMaxAge(3600);
```

Making your controller response context-aware

If the content you're rendering depends on a user's permissions, then you should make the response Context-aware:

```
use Symfony\Component\HttpFoundation\Response;

// Inside a controller action
// Tells proxy configured to support this header to take the rights of a user (user
hash) into account for the cache
$response = new Response();
$response->setVary('X-User-Hash');
```