Moodle Caching

Two aspects to caching

- **Caching backends** (storage) – e.g. MongoDB, APC user cache (APCu), and Redis. These are provided by *cache*, more specifically *cache store*, plugins.
- **Moodle Universal Cache** (MUC) - using the Caching API – used in code.

Types Of Caching

These are defined in ***/cache/classes/store.php***:

- **Request** cache – `MODE_REQUEST` – alive for the duration of the request and cleared at the end of every request
- **Session** cache – `MODE_SESSION` – user-specific and alive for the duration of the user's session
- **Application** cache – `MODE_APPLICATION` – a shared cache that all users can access, making common data accessible

It is also possible to create an **adhoc** cache though this is not recommended.

Cache API

1. Define the cache in ***db/caches.php*** file and requires language strings to be defined.
2. Get the cache object with `cache::make`
3. Set a cache with the `set()` method e.g. `$cache→set($key, $value)`. You can also use `set_many()` method to set multiple cache objects.
4. Retrieve the cache object with the `get()` method e.g. `$cache→get($key)`. You can also use `get_many()` to retrieve multiple objects.
5. Delete when no longer required with the `delete()` method e.g. `$cache->delete($key)`.

```
$definitions = [
    'area' => [    // The name of the cache area is the key.
        'mode' => cache_store::MODE_APPLICATION,
        'simplekeys' => false,  //  Only a-z A-Z 0-9 _
        'simpledata' => false, // Data is scalar or array of scalar.
        'requireidentifiers' => ['ident1', 'ident2'],
        'requiredataguarantee' => false,
        'requiremultipleidentifiers' => false,
        'requirelockingread' => false,
        'requirelockingwrite' => false,
        'maxsize' => null,
        'overrideclass' => null,
        'overrideclassfile' => null,
        'datasource' => null,
        'datasourcefile' => null,
        'staticacceleration' => false,
        'staticaccelerationsize' => null,
        'ttl' => 0, // Time to live - not recommended - create an event driven invalidation system
        'mappingsonly' => false,
        'invalidationevents' => ['event1', 'event2'],
        'canuselocalstore' => false,
        'sharingoptions' => cache_definition::SHARING_DEFAULT,
        'defaultsharing' => cache_definition::SHARING_DEFAULT,
    ],
];
```

- For more information about using the API, please see the ‘Cache API’ page https://docs.moodle.org/dev/Cache_API
- For general background information about caching, see The Moodle Universal Cache page at https://docs.moodle.org/dev/The_Moodle_Universal_Cache_(MUC)
- Also, see the ‘Caching’ page https://docs.moodle.org/311/en/Caching for a general background.