Session

Sessions are handled by the Symfony2 framework, specifically API and underlying session handlers provided by HTTP Fundation component[1][2], this is further enhanced in eZ Publish with support for SiteAccess aware session cookie configuration.

Use of Memcached (or experimentally using PDO) as session handler is a requirement in Cluster setup, for details see below, for an overview of clustering feature see Clustering.

Session handlers

In Symfony, session handler is configured using `framework.session.handler_id.` Symfony can be configured to use custom handlers[2], or just fallback to what is configured in PHP by setting it to null (~).

Default configuration

Prior to 5.4 / 2014.11

Before 5.4 eZ Publish uses default Framework Bundle configuration, which on Symfony 2.3 implies Symfony's *NativeFileSessionHandler*[3] (`session.handler.native_file` service). This handler forces PHP's builtin "files" session save handler, and specifically configures it to use `session.save_path` set to `ezpublish/sessions` by default (`ezpublish/cache/<env>/sessions` before 5.3).

Default config.yml session configuration in 5.3 / 2014.03

```
framework:
    session:
        save_path: "%kernel.root_dir%/sessions"
        # The session name defined here will be overridden by the one defined in your
ezpublish.yml, for your SiteAccess.
        # Default session name is "eZSESSID{siteaccess_hash}" (unique session name per
SiteAccess).
        # See ezpublish.yml.example for an example on how to configure this.
```

Session Garbage collection on Debian & Ubuntu
Debian based Linux distros disables session.gc_probability by default and uses cronjob instead to clear sessions files. As we use custom save_path for sessions here that would normally be a problem, however default Symfony configuration makes sure to re enable this in `framework.session.gc_probability` so with default Symfony and PHP settings it should garbage collecting sessions files roughly every 100th time on average (1% probability by default).

As of 5.4 / 2014.11

Uses same default configuration as recent versions of Symfony standard distribution, this makes sure you can configure sessions purely in php by default, and allows Debian/Ubuntu session file cleanup cronjob to work as intended.

Default config.yml session configuration as of 5.4 / 2014.11

```
framework:
    session:
        # handler_id set to null will use default session handler from php.ini
        handler_id:  ~
```

Recommendations for production setup

Single server setup

For single server, default handler should be preferred.

Cluster setup

For Cluster setup we need to configure Sessions to use a backend that is shared between web servers, and supports locking. Only options out of the box supporting this in Symfony is native PHP memcached session save handler provided by php-memcached extension, and Symfony session handler for PDO (database).

Storing sessions in Memcached using php-memcached

For setting up eZ Publish using this memcached you'll need to configure the session save handler settings in php.ini as documented here, optionally tweak php-memcached session settings, and use default configuration as of eZ Publish 5.4 / 2014.11 documented above.

Alternative storing sessions in database using PDO

While not currently our recommendation from performance perspective, for setups where Database is preferred for storing Sessions, you may use Symfony's PdoSessionHandler.
Below is an configuration example for eZ Publish, but please refer to documented in Symfony Cookbook documentation for full documentation.

```
framework:
    session:
        # ...
        handler_id: session.handler.pdo

parameters:
    pdo.db_options:
        db_table:    session
        db_id_col:   session_id
        db_data_col: session_value
        db_time_col: session_time

services:
    pdo:
        class: PDO
        arguments:
            dsn:      "mysql:dbname=<mysql_database>"
            user:     <mysql_user>
            password: <mysql_password>

    session.handler.pdo:
        class:
Symfony\Component\HttpFoundation\Session\Storage\Handler\PdoSessionHandler
        arguments: ["@pdo", "%pdo.db_options%"]
```

Further Symfony references

1. Cookbook Session recipes (symfony.com)
2. HTTP Fundation Component documentation (symfony.com)
3. Source code of NativeFileSessionHandler (github.com),
4. Cookbook Configuration recipe for setting-up PdoSessionHandler (symfony.com), aka `session.handler.pdo` service