Legacy configuration injection

For better integration between 5.x (symfony based) kernel and legacy (4.x) kernel, injection is used to inject settings, session and current siteaccess from 5.x into legacy using an event: kernel.event_subscriber

Injected settings

Several settings are injected to avoid having to duplicate settings across new and old kernel, this is done in eZ\Bundle\EzPublishLegacyBundle\LegacyMapper\Configuration.
For mappings below, yml settings prefixed with ezpublish are system settings, while the rest are settings you can define pr siteaccess.

Database settings

The settings for "Server", "Port", "User", "Password", "Database" and "DatabaseImplementation" is set based on the current settings in your yml files, either from the explicit settings defined below, or the "dsn"

Mapping:

yml	site.ini [DatabaseSettings]
server	Server
port	Port
user	User
password	Password
database_name	Database
type	DatabaseImplementation

Storage settings

The settings for "VarDir" and "StorageDir" is set based on current settings in your yml files as shown below.

Mapping:

yml	site.ini [FileSettings]
var_dir	VarDir
storage_dir	StorageDir

Image settings

Some of the settings for image sub system is set based on your yml files as shown below.

Mapping:

yml	image.ini

ezpublish.image.temporary_dir	[FileSettings]\TemporaryDir
ezpublish.image.published_images_dir	[FileSettings]\PublishedImages
ezpublish.image.versioned_images_dir	[FileSettings]\VersionedImages
image_variations	[FileSettings]\AliasList
ezpublish.image.imagemagick.enabled	[ImageMagick]\IsEnabled
ezpublish.image.imagemagick.executable_path	[ImageMagick]\ExecutablePath
ezpublish.image.imagemagick.executable	[ImageMagick]\Executable
imagemagick.pre_parameters	[ImageMagick]\PreParameters
imagemagick.post_parameters	[ImageMagick]\PostParameters
ezpublish.image.imagemagick.filters	[ImageMagick]\Filters

Using ImageMagick filters

The following block shows a valid ImageMagick filters usage example for `ezpublish.yml`:

ImageMagick filters usage example

```
ezpublish:
    imagemagick:
        filters:
            geometry/scale: "-geometry {1}x{2}"
```

Since ImageMagick filters usage changed from eZ Publish 4.x versions you can find the list of filters existing by default to use eZ Publish 5.x:

ImageMagick filters list for yml
`sharpen: "-sharpen 0.5"`
`geometry/scale: "-geometry {1}x{2}"`
`geometry/scalewidth: "-geometry {1}"`
`geometry/scaleheight: "-geometry x{1}"`
`geometry/scaledownonly: "-geometry {1}x{2}>"`
`geometry/scalewidthdownonly: "-geometry {1}>"`
`geometry/scaleheightdownonly: "-geometry x{1}>"`
`geometry/scaleexact: "-geometry {1}x{2}!"`
`geometry/scalepercent: "-geometry {1}x{2}"`
`geometry/crop: "-crop {1}x{2}+{3}+{4}"`
`filter/noise: "-noise {1}"`
`filter/swirl: "-swirl {1}"`
`colorspace/gray: "-colorspace GRAY"`
`colorspace/transparent: "-colorspace Transparent"`
`colorspace: "-colorspace {1}"`
`border: "-border {1}x{2}"`

`border/color: "-bordercolor rgb({1},{2},{3})"`
`border/width: "-borderwidth {1}"`
`flatten: "-flatten"`
`resize: "-resize {1}"`

For more details on setting ImageMagick filters on image.ini please refer to the [imagemagick] / filters documentation.

Extending injected settings

It's possible to add your own kernel event subscriber and also inject your own settings by following how it is done in LegacyMapper\Configuration, and then at the end merge it with existing injected settings like so:

injected-settings

```
$event->getParameters()->set(
            "injected-settings",
            $settings + (array)$event->getParameters()->get( "injected-settings" )
        );
```

Injected Behavior

In addition to injected settings, some injection of behavior is also performed.

eZFormToken (CSRF) integration

This feature is only available as of eZ Publish 5.1 (2013.01)

If your config.yml setting have *framework.csrf_protection.enabled* set to *true*, then both *kernel.secret* and *framework.csrf_protection.field_name* will be sent to ezxFormToken class so csrf protection in legacy uses the same token and form field name.

By making sure all your Symfony forms uses the provided csrf protection, forms with *intention=legacy* can be setup to send data to legacy kernel.

Please note that `framework.csrf_protection.field_name` shouldn't be changed as that would prevent eZFormToken from working with most AJAX custom code.

ezpEvent (Cache clear) integration

A listener is setup for both content/cache and content/cache/all to make sure Symfony (Internal proxy or Varnish with custom vcl) HTTP cache is cleared when cache is cleared in eZ Publish admin interface.