Content and Location view providers

View\Manager & View\Provider

The role of the `(eZ\Publish\Core\MVC\Symfony\)View\Manager` is to select the right template for displaying a given content or location. It aggregates objects called *content and location view providers* which respectively implement `eZ\Publish\Core\MVC\Symfony\View\Provider\Content` and `eZ\Publish\Core\MVC\Symfony\View\Provider\Location` interfaces.

Each time a content is to be displayed through the `Content\ViewController`, the `View\Manager` iterates over the registered content or location `View\Provider` objects and calls `getView()`.

Provided View\Provider implementations

Name	Usage
`View provider configuration`	Based on application configuration. Formerly known as *Template override system*.
`eZ\Publish\Core\MVC\Legacy\View\Provider\Content` `eZ\Publish\Core\MVC\Legacy\View\Provider\Location`	Forwards view selection to the legacy kernel by running the old content/view module. Pagelayout used is the one configured in `ezpublish_legacy.<scope>.view_default_layout`. For more details about the `<scope>` please refer to the scope configuration documentation.

Custom View\Provider

Difference between `View\Provider\Location` and `View\Provider\Content`

- A `View\Provider\Location` only deals with `Location` objects and implements `eZ\Publish\Core\MVC\Symfony\View\Provider\Location` interface.
- A `View\Provider\Content` only deals with `ContentInfo` objects and implements `eZ\Publish\Core\MVC\Symfony\View\Provider\Content` interface.

When to develop a custom `View\Provider\(Location|Content)`

- You want a custom template selection based on a very specific state of your application
- You depend on external resources for view selection
- You want to override the default one (based on configuration) for some reason

`View\Provider` objects need to be properly registered in the service container with the **`ezpublish.location_view_provider`** or **`ezpublish.content_view_provider`** service tag.

```
parameters:
    acme.location_view_provider.class: Acme\DemoBundle\Content\MyLocationViewProvider

services:
    acme.location_view_provider:
        class: %ezdemo.location_view_provider.class%
        tags:
            - {name: ezpublish.location_view_provider, priority: 30}
```

Tag attribute name	Usage
priority	An integer giving the priority to the `View\Provider\(Content|Location)` in the `View\Manager`. The priority range is **from -255 to 255**

Example

Custom View\Provider\Location

```
<?php

namespace Acme\DemoBundle\Content;

use eZ\Publish\Core\MVC\Symfony\View\ContentView;
use eZ\Publish\Core\MVC\Symfony\View\Provider\Location as LocationViewProvider;
use eZ\Publish\API\Repository\Values\Content\Location;

class MyLocationViewProvider implements LocationViewProvider
{
    /**
     * Returns a ContentView object corresponding to $location, or void if not
applicable
     *
     * @param \eZ\Publish\API\Repository\Values\Content\Location $location
     * @param string $viewType
     * @return \eZ\Publish\Core\MVC\Symfony\View\ContentView|null
     */
    public function getView( Location $location, $viewType )
    {
        // Let's check location Id
        switch ( $location->id )
        {
            // Special template for home page, passing "foo" variable to the template
            case 2:
                return new ContentView( "AcmeDemoBundle:$viewType:home.html.twig",
array( 'foo' => 'bar' ) );
        }

        // ContentType identifier (formerly "class identifier")
        switch ( $contentInfo->contentType->identifier )
        {
            // For view full, it will load AcmeDemoBundle:full:small_folder.html.twig
            case 'folder':
                return new ContentView(
"AcmeDemoBundle:$viewType:small_folder.html.twig" );
        }
    }
}
```