Legacy kernel event

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

This page describe how you can do that.

Adding a kernel.event_subscriber

A legacy kernel event subscriber is added by tagging your event subscriber service.

Bellow is an example of yml configuration involved:

kernel.event_subscriber

```
my_app.legacy_mapper.something:
        class: %my_app.legacy_mapper.something.class%
        arguments: [@service_container]
        tags:
            - { name: kernel.event_subscriber }
```

The class refered to as %my_app.legacy_mapper.something.class% can look like this:

my_app.legacy_mapper.something.class

```
namespace MyApp\LegacyMapper;
use eZ\Publish\Core\MVC\Legacy\LegacyEvents;
use eZ\Publish\Core\MVC\Legacy\Event\PreBuildKernelWebHandlerEvent;
use Symfony\Component\EventDispatcher\EventSubscriberInterface;
use Symfony\Component\DependencyInjection\ContainerInterface;
/**
 * Maps something into Legacy kernel
 */
class Something implements EventSubscriberInterface
{
    /**
     * @var \Symfony\Component\DependencyInjection\ContainerInterface
     */
    private $container;

   /**
    * @param \Symfony\Component\DependencyInjection\ContainerInterface $container
    */
    public function __construct( ContainerInterface $container )
    {
        $this->container = $container;
    }
    public static function getSubscribedEvents()
    {
        return array(
            LegacyEvents::PRE_BUILD_LEGACY_KERNEL_WEB => array(
'onBuildKernelWebHandler', 128 )
        );
    }
    /**
     * Maps matched siteaccess to the legacy parameters
     *
     * @param \eZ\Publish\Core\MVC\Legacy\Event\PreBuildKernelWebHandlerEvent $event
     *
     * @return void
     */
    public function onBuildKernelWebHandler( PreBuildKernelWebHandlerEvent $event )
    {
        // Do something, see eZ\Bundle\EzPublishLegacyBundle\LegacyMapper\* for
examples
        // Example for injecting some settings:

        $settings = array(
            'site.ini/Block/Setting' => $this->container->getParameter( 'some.setting'
),
            'site.ini/Block/Setting2' => $this->configResolver->getParameter(
'some.setting2' )
        );

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