

# Developer Operator for Kubernetes Applications

---

Developer Operator

*xrow GmbH*

*Copyright xrow GmbH*

## Table of contents

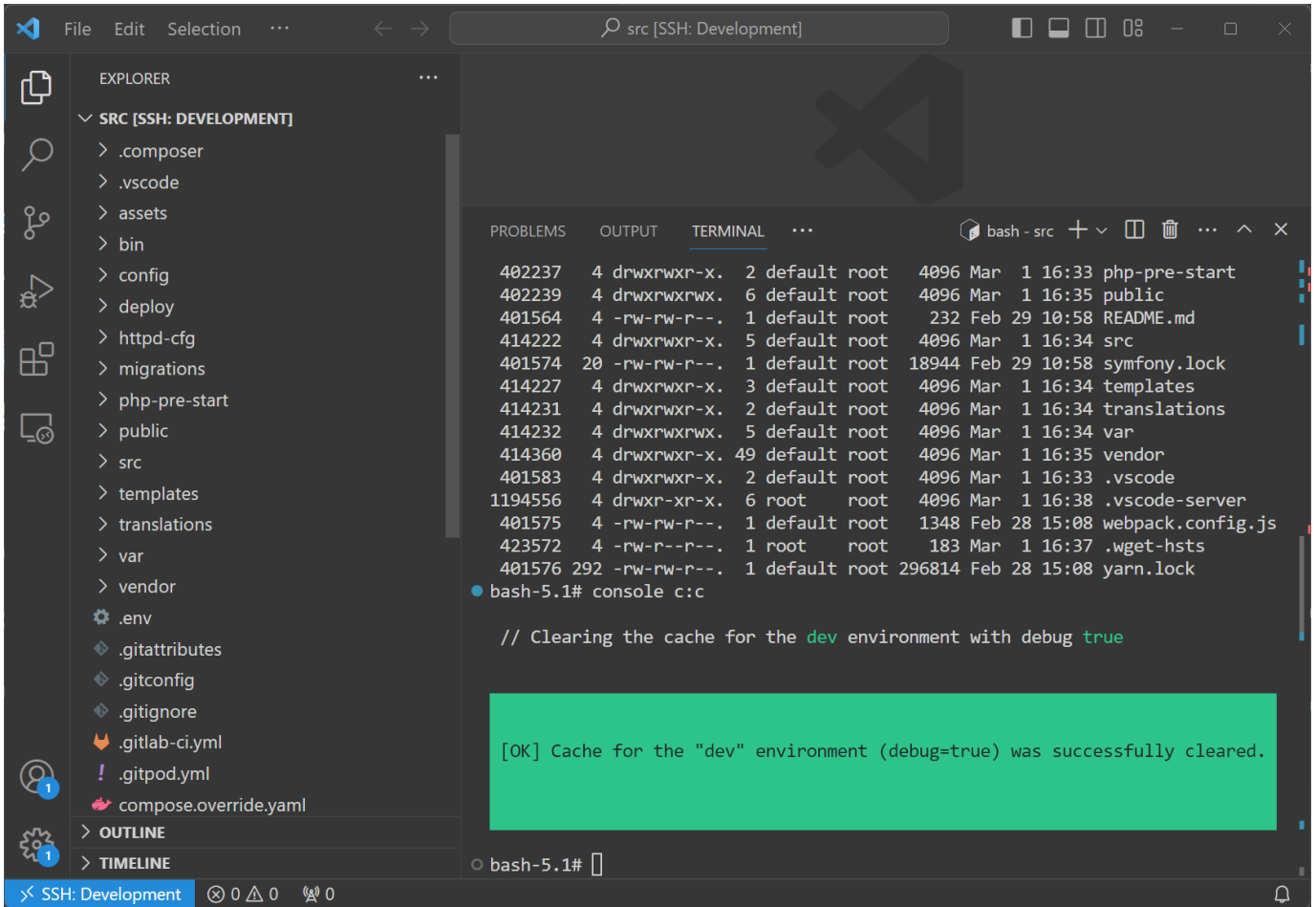
---

1. Developer Operator	3
1.1 Developer Operator	3
1.2 Installation	3
1.3 The Developer Workspace	5
1.4 Configuration settings	7
1.5 Day two development	8
1.6 Recreate all dev pods	8
1.7 Working with ...	9
1.8 Troubleshooting	16
1.9 Guide for Administrators	18

# 1. Developer Operator

## 1.1 Developer Operator

Improves the developer experience for applications running in Kubernetes.



[PDF Download](#)

## 1.2 Installation

### 1.2.1 Helm chart installation

The installation step needs to be done only once per cluster. You may skip this step if you have a cluster already enabled.

Install stable:

```
helm upgrade --install developer-crd oci://registry.gitlab.com/xrow-public/repository/charts/developer-crd --version 2.0.3 -n kube-system
```

```
helm upgrade --install developer oci://registry.gitlab.com/xrow-public/repository/charts/developer-operator --version 2.0.3 -n kube-system
```

### Install latest:

```
helm upgrade --install developer-crd oci://registry.gitlab.com/xrow-public/repository/charts/developer-crd --version 0.0.0+1a86d5f -n kube-system
```

```
helm upgrade --install developer oci://registry.gitlab.com/xrow-public/repository/charts/developer-operator --version 0.0.0+1a86d5f -n kube-system
```

The latest unstable build is version 0.0.0+1a86d5f from 2024-10-29 10:41:52+01:00 by Andy Caiger

## 1.2.2 Helm chart uninstall

---

```
kubectl delete developer --all-namespaces --all
helm uninstall developer -n kube-system
helm uninstall developer-crd -n kube-system
```

## 1.2.3 Kubernetes Server Setup

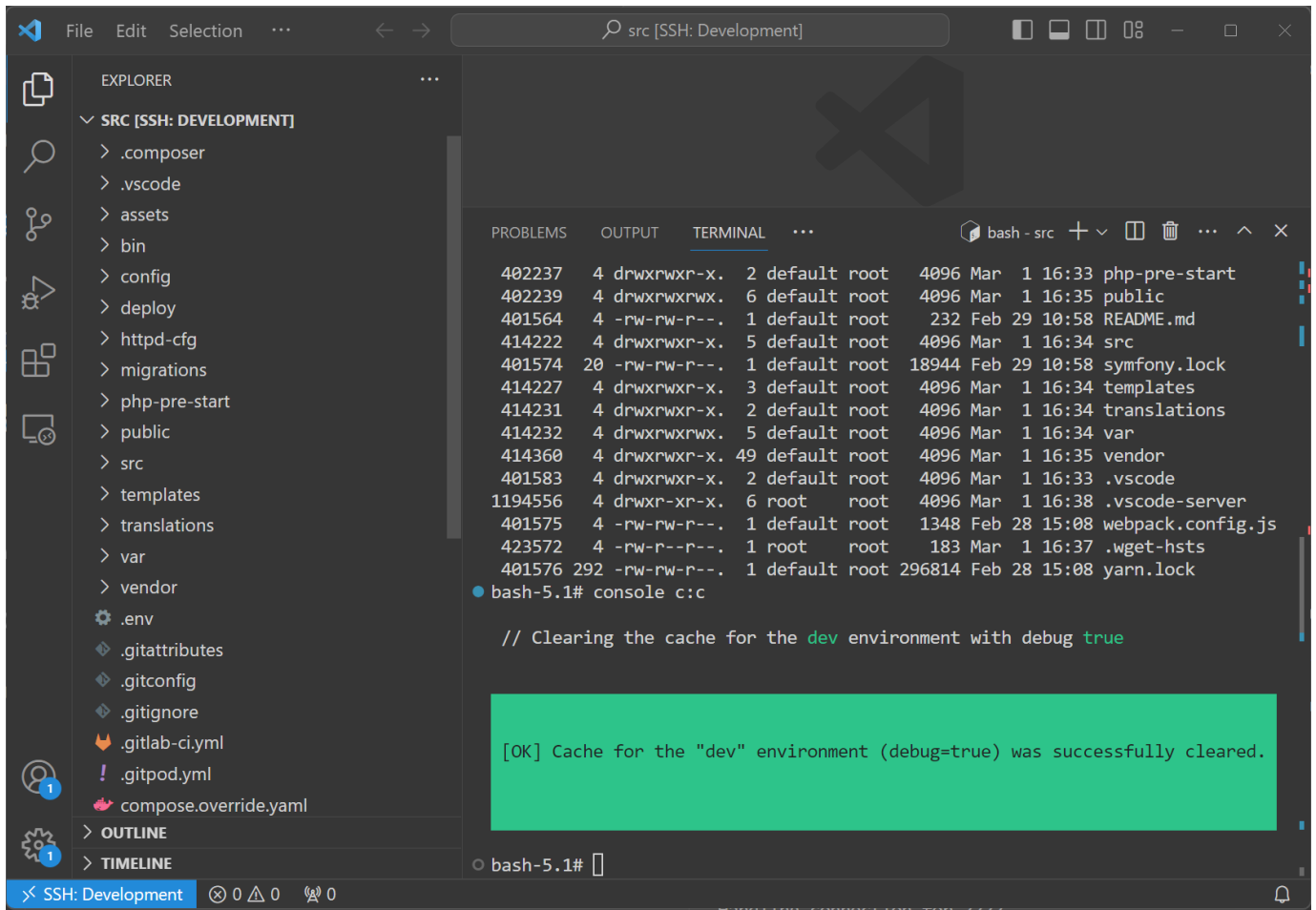
---

Skip these steps if you have done it already

- Edit `/etc/sysctl.conf` and add the line `fs.inotify.max_user_watches=524288`
- run `sudo sysctl -p`

🕒 28. Oktober 2024

## 1.3 The Developer Workspace



### 1.3.1 Prerequisites

- A running application installation that will provide a deployment as a workspace template.

### 1.3.2 Workspace Setup

Take the sample YAML configuration and modify it:

- Look up the deployment name of your application and add it to use CRD as "deployment" under `spec`
- Enter your SSH public key, email, name and the desired container image
- Save your customized YAML configuration somewhere. Keep it for future reference.
- Open the `Kubernetes` console and add the `CRD`

#### Sample YAML to create a developer workspace

```
apiVersion: xrow.com/v2
kind: Developer
metadata:
  name: bjoern-dieding
  namespace: ibexa-test
spec:
  deployment: test-ibexa-frontend
  key: "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOXGvbBWWfPKoMEU8zKMaKGLocGLVMAq4K24UZMyzjmc"
  email: "bjoern@xrow.de"
  name: "Björn Dieding"
  env:
```

```
- name: APP_ENV
  value: dev
- name: APP_DEBUG
  value: "1"
```

### 1.3.3 Port Forward to your local maschine

Just the ssh and the webserver ports are really important.

```
kubectl config use-context 08.xrow.net
kubectl config set-context --current --namespace=project-test
kubectl port-forward service/bjoern-dieding 2222:2222 &
kubectl port-forward service/bjoern-dieding 8080:8080 &
kubectl port-forward service/bjoern-dieding 8443:8443 &
kubectl port-forward service/$(kubectl get service | grep solr | awk '{print $1}') 8983:8983 &
kubectl port-forward service/$(kubectl get service | grep mysql | awk '{print $1}') 3306:3306 &
```

Alternatively you can forward with a tool like [kubefwd](#)

### 1.3.4 Connect to the Pod / development environment via an SSH Agent

#### Linux: Connect via SSH

```
[root@host ~]# ssh root@127.0.0.1 -p 2222 -A
Last login: Tue Feb 18 00:37:33 2020 from 10.128.0.1
Hi Björn Dieding <bjoern@xrow.de>, you can start coding
-bash-4.2$ ssh -T git@gitlab.com
Warning: Permanently added 'gitlab.com,35.231.145.151' (ECDSA) to the list of known hosts.
Welcome to GitLab, @xrow!
```

#### Connect via VS CODE

- Use the vscode kubernetes extension
- Select the desired SVC or POD and select forward all

### 1.3.5 Start to code

- For best experience make sure a .git folder is present in the workdir.
- Modify any file and commit to git.
- Start an additional dev server if needed. In the example it is a node dev server.

```
yarn install
$(yarn bin)/encore dev-server --host 0.0.0.0 --port 9000 --config-name ibexa --allowed-hosts 192.168.222.2 --public http://192.168.222.2:9000/
```

 28. Oktober 2024

## 1.4 Configuration settings

---

### 1.4.1 Common settings

Configuration	Default	Description	Sample
deployment	-	Name of the deployment to use	-
env	{}	Env Variables	-
image.repository	registry.gitlab.com/xrow-shared/ezplatform/dev	image to use	-
image.tag	latest	Use hash or default	-
image.pullPolicy	Always	-	-
imageCredentials.registry	-	Not required since the service account has access	registry.gitlab.com
imageCredentials.username	-	Not required since the service account has access	gitlab+deploy-token-183447
imageCredentials.password	-	Not required since the service account has access	ABCDEF1234
storage.class	-	set a custom storage class	-

### 1.4.2 PHP Symfony

See <https://gitlab.com/xrow-public/docker-php-ezplatform>

```
apiVersion: xrow.com/v2
kind: Developer
....
env:
- name: APP_ENV
  value: dev
- name: APP_DEBUG
  value: "1"
...
```

### 1.4.3 Node

See <https://gitlab.com/xrow-public/s2i-nodejs-container>

```
apiVersion: xrow.com/v2
kind: Developer
....
env:
- name: NODE_ENV
  value: development
- name: DEV_MODE
  value: 'true'
...
```

## 1.5 Day two development

---

### 1.5.1 Setting an ENV variable

---

You can set an ENV variable for different purposes in different ways. By default, helm, kubernetes and ConfigMap variables for services are injected by default.

View them by typing `env`.

If you want to set a static env variable, set them via `.s2i/environment`. It might be better though to set a static symfony parameter. If you want to set variables that differ per stage, set them in the helm chart or via `ezplatform / developer operator`.

```
env:  
- name: APP_ENV  
  value: "dev"  
- name: APP_DEBUG  
  value: "1"
```

## 1.6 Recreate all dev pods

---

To force the regeneration of all pods you can follow this logic.

```
oc get Developer --all-namespaces  
oc get Developer --all-namespaces -o yaml > export.yaml  
oc get developer --no-headers=true --all-namespaces | sed -r 's/(\S+)\s+(\S+).*/oc --namespace \1 delete developer \2/e'  
oc apply -f export.yaml
```

🕒 28. Oktober 2024



## 1.7 Working with ...

---

### 1.7.1 Working with VS Code

---

This guide helps you connect your recommended IDE [VS Code](#).

#### Setup the IDE [VS Code](#)

- Install the IDE [VS Code](#)
- Install the [kubernetes extension](#) in [VS Code](#)
- Install the [remote SSH extension](#) in [VS Code](#)
- Acquire a [kubeconfig](#) from an administrator from a service account with sufficient rights to connect to the api.
- Place the [kubeconfig](#) in the standard kubernetes directory `~/.kube/config`
- Use the kubernetes extension, select the proper [kubeconfig](#) and connect to kubernetes via the extension. Errors fetching namespaces might be desired from the administrator.
- Create a [Developer CRD](#) with your public key credentials
- Navigate to your developer pod created by the [Developer CRD](#) under workloads and right click to forward all ports
- Connect via remote SSH with this recommended settings:

`~/.ssh/ssh_config`

```
Host "Development Pod"
  HostName 127.0.0.1
  User root
  Port 2222
  ForwardAgent yes
  IdentityFile ~/.ssh/id_ed25519
  StrictHostKeyChecking no
  UserKnownHostsFile /dev/null
  # ProxyJump vagrant
```

- If you are connecting over a vs code remote shell, you can use the `ProxyJump` setting from above.
- If you are connected, open the projects root folder
- **You are now ready to start coding.** You can switch to the master branch and create for example a feature branch.
- You can access your webserver via `http://127.0.0.1:8080` or `https://127.0.0.1:8443`

In case you use vs code via a remote host you can make `kubectl port-forward` listen to all interfaces via alias in your profile (e.g. `/etc/profile.d/vscode.sh`).

`/etc/profile.d/vscode.sh`

```
# Make vs code listen too all interfaces
kubectl() {
  if [ "$1" = "port-forward" ]
  then
    vars=${@:2}
    echo "Alias kubectl port-forward --address 0.0.0.0 $vars"
    command kubectl port-forward --address 0.0.0.0 $vars
  else
    command kubectl $@
  fi;
}
```

If you don't use a ForwardAgent SSH agent you can add the key to development pod

1. Add a new connection
2. Place your key in `~/.ssh/id_ed25519` or create a new key with `ssh-keygen -t ed25519`
3. `chmod 600 ~/.ssh/id_ed25519`

### Guide to create a kubeconfig from the default service account

This is not fully working or incomplete. We keep this part for future reference.

```
#!/bin/bash
mkdir -p ~/.kube
chmod 600 ~/.kube

# Point to the internal API server hostname
APISERVER=https://kubernetes.default.svc
# Path to ServiceAccount token
SERVICEACCOUNT=/var/run/secrets/kubernetes.io/serviceaccount
# Read this Pod's namespace
NAMESPACE=$(cat ${SERVICEACCOUNT}/namespace)
# Read the ServiceAccount bearer token
TOKEN=$(cat ${SERVICEACCOUNT}/token)
# Reference the internal certificate authority (CA)
CACERT=${SERVICEACCOUNT}/ca.crt
# Explore the API with TOKEN
curl --cacert ${CACERT} --header "Authorization: Bearer ${TOKEN}" -X GET ${APISERVER}/api

TOKEN=$(cat /var/run/secrets/kubernetes.io/serviceaccount/token)
NAMESPACE=$(cat /var/run/secrets/kubernetes.io/serviceaccount/namespace)
cat << EOF > ~/.kube/config
apiVersion: v1
clusters:
- cluster:
  server: https://kubernetes.default.svc:8443
  name: local
contexts:
- context:
  cluster: local
  namespace: $NAMESPACE
  user: namespace-token-user
  name: default
current-context: default
kind: Config
preferences: {}
users:
- name: namespace-token-user
  user:
    token: $TOKEN
EOF
chmod 600 ~/.kube/config
```

🕒 28. Oktober 2024

## 1.7.2 How to use XDEBUG

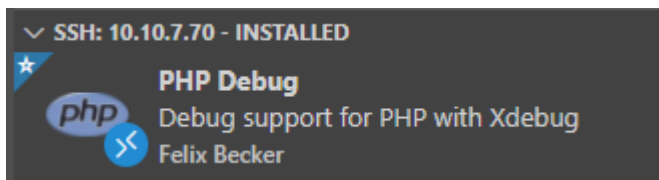
### Modify the Developer CRD

See the [list of all available modes](#) .

```
apiVersion: xrow.com/v2
kind: Developer
....
env:
  - name: XDEBUG_MODE
    value: debug
...
```

### Setup VSCODE

- start Visual Studio Code
- go to **Extensions** in the toolbar on the left
- search for **PHP Debug** by Felix Becker
- click on **Install** on a remote host



Before actually using Xdebug a few things need to be prepared:

- In VS Code open the file `.vscode/launch.json` in your project's root directory and edit it's content.

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "Listen for Xdebug",
      "type": "php",
      "request": "launch",
      "port": 9003,
      "stopOnEntry": false
    }
  ]
}
```

- Decide change `stopOnEntry` to **true** if needed.

### Run a debugging session

Keep in mind that you always need to trigger a function if you want to debug it.

1. either by reloading the frontend
2. or by executing it via terminal if it can't be triggered by refreshing the frontend

#### STARTING A DEBUGGING SESSION

- open a file in the IDE which should be debugged
- create one or multiple breakpoints by clicking in the beginning of the line which should be investigated
- go to tab **Run** and click on **Start Debugging**
- If necessary select "XDebug" in the new toolbar **Listen for Xdebug**
- If you use **the browser** open a URL like `http://192.168.222.2:8080/dashboard` to start debugging
- If you use **the CLI** run a command like `console ibexa:migrations:migrate` to start debugging

After this the following scenarios are possible:

1. the tool gives a result without doing anything additional
2. the tool gives a result only after refreshing the dev-pod's frontend
3. the tool gives a result not until the affected function was called via terminal
4. if the breakpoint was reached the results are listed in the menu on the left side
5. the first section **VARIABLES** shows the used variables during the debug call
6. the second section **CALL STACK** shows which classes and functions were used
7. the third section **BREAKPOINTS** allows to filter events

✓ **VARIABLES** 📄  
 ✓ **Locals**  
 > \$arguments: array(4)  
 > \$controller: array(2)  
 > \$event: Symfony\Component\HttpKernel\Event\ControllerArgumentsEvent  
 \$msg: uninitialized  
 > \$request: Symfony\Component\HttpFoundation\Request  
 > \$response: Symfony\Component\HttpFoundation\Response  
 \$type: 2  
 > \$this: **Symfony\Component\HttpKernel\HttpKernel**  
 > Superglobals  
 > User defined constants

---

> **WATCH**

---

✓ **CALL STACK** PAUSED ON STEP  
 Symfony\Component\HttpKernel\HttpKernel->handleRaw      HttpKernel.php 160:1  
 Symfony\Component\HttpKernel\HttpKernel->handle          HttpKernel.php 79:1  
 Symfony\Component\HttpKernel\HttpCache\SubRequestHandler::handle    SubRequestHa...  
 Symfony\Component\HttpKernel\Fragment\InlineFragmentRenderer->render    ../http-ker...  
 eZ\Bundle\EzPublishCoreBundle\Fragment\InlineFragmentRenderer->render    ../EzPubl...  
 Symfony\Component\HttpKernel\DependencyInjection\LazyLoadingFragmentHandler->rende  
 Symfony\Component\HttpKernel\DependencyInjection\LazyLoadingFragmentHandler->rende  
 Symfony\Bridge\Twig\Extension\HttpKernelRuntime->renderFragment    HttpKernelRunti...  
 \_\_TwigTemplate\_eb0c1c3575ba9552b1d8535c60976fed5d91ff3fd6d235c9c076eb8294388234->b  
 \_\_TwigTemplate\_eb0c1c3575ba9552b1d8535c60976fed5d91ff3fd6d235c9c076eb8294388234->c  
 \_\_TwigTemplate\_eb0c1c3575ba9552b1d8535c60976fed5d91ff3fd6d235c9c076eb8294388234->d  
 \_\_TwigTemplate\_eb0c1c3575ba9552b1d8535c60976fed5d91ff3fd6d235c9c076eb8294388234->d

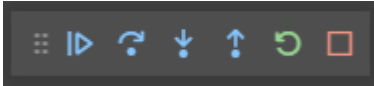
---

✓ **BREAKPOINTS**  
 Notices  
 Warnings  
 Errors  
 Exceptions  
 Everything  
 EventController.php    src/Controller      60  
 MenuController.php    src/Hannover/FrontendBundle/Controller      119  
 MenuController.php    src/Hannover/FrontendBundle/Controller      121  
 MenuController.php    src/Hannover/FrontendBundle/Controller      124

### The debugging toolbar

After Xdebug reached a breakpoint different options become available. These options can be found in the Xdebug-Toolbar that appears in VS Code after starting the debug process.

- **Continue:** navigate to the next breakpoint
- **Step Over:** jump to the next function/variable/if-statement etc. (breakpoints won't be considered)
- **Step In:** go inside a statement and jump to the related function that is used
- **Step Out:** go outside of a statement and jump either to the previous breakpoint or to the overlaying function
- **Restart:** start the debug process from the beginning (first breakpoint)
- **Stop:** end the debug process and close the toolbar



🕒 28. Oktober 2024

## 1.7.3 How to use Blackfire

### Modify the Developer CRD

Copy your [credentials](#).

```
apiVersion: xrow.com/v2
kind: Developer
....
env:
- name: BLACKFIRE_CLIENT_ID
  value: XXX
- name: BLACKFIRE_CLIENT_TOKEN
  value: XXX
- name: BLACKFIRE_SERVER_ID
  value: XXX
- name: BLACKFIRE_SERVER_TOKEN
  value: XXX
....
```

### Run some requests

You need to copy the cookie from your browser into the env frist.

```
COOKIE="eZSESSID21232f297a57a5a743894a0e4a801fc3=6a496ea21b98c357ff1e7c0cc0082a57"
```

```
blackfire client:curl \
-H "x-siteaccess: admin" \
-H "Cookie: ${COOKIE}" \
http://127.0.0.1:8080/dashboard
```

```
COOKIE="eZSESSID21232f297a57a5a743894a0e4a801fc3=6a496ea21b98c357ff1e7c0cc0082a57"
```

```
blackfire client:curl \
-H "x-siteaccess: admin" \
-H "Cookie: ${COOKIE}" \
-H "accept: application/vnd.ez.api.View+json" \
-H "content-type: application/vnd.ez.api.ViewInput+json;version=1.1" \
--data-raw '{"ViewInput":{"identifier":"navigation","public":false,"LocationQuery":{"FacetBuilders":{},"Filter":
{"ContentTypeIdentifierCriterion":"page","SubtreeCriterion":"/1/2/","VisibilityCriterion":false,"SortClauses":{"LocationDepth":"ascending"},"limit":
300,"offset":0}}}' \
"http://127.0.0.1:8080/api/ezp/v2/views" \
--compressed
```

Visit the [Blackfire dashboard](#).

🕒 28. Oktober 2024

## 1.8 Troubleshooting

---

### 1.8.1 Image Pull Backoff

---

This could have the following reasons:

- You are notified that your images repositories credentials are incorrect or the image doesn't exist.
- Your image is very big.
- You use persistent storage for your code. Which slows down the initial startup.

### 1.8.2 Image tag latest gives 500

---

- Try to find an older version of that image that works.

### 1.8.3 Your pod returns a 500 errors / Using an image with code errors

---

Things to consider:

- latest image is broken
- try a different image
- your code might be broken, try to fix the code
- the data in the database might be awkward and might cause the application to fail
- binary data might be missing that causes an exception in the code

### 1.8.4 Your POD IP is not reachable

---

- You might be running two developer CRCs that use the same external IP

### 1.8.5 The result you see in the browser is not the result you expect

---

- You have been using your OS hosts file to address the POD, meanwhile you changed the IP of the POD and your hosts file, but browser doesn't notice. Restart the browser.

### 1.8.6 Can't login via ssh

---

Error: child\_set\_env: too many env vars

<https://github.com/br1/obfuscated-openssh/blob/master/session.c#L972>

This can be fixed by removing the amount of pods in a namespace.



## 1.8.7 Dev the system is slow in general

Here are hint on how fast a System should be

Scenario	Result
APP_ENV=prod, XDEBUG_MODE=, BLACKFIRE-*=	Request will take 1 sec it has no overhead
APP_ENV=dev, APP_DEV=1, XDEBUG_MODE=, BLACKFIRE-*=	Request will take 3 sec it has overhead of the webdeveloper toolbar
APP_ENV=dev, APP_DEV=1, XDEBUG_MODE=develop, BLACKFIRE-*=	Request will take 6, if the IDE is connected. <b>It will take 60+ sec if the IDE is not listening to the request, because XDEBUG can't reach the IDE (timeout). See /tmp/xdebug.log</b>
APP_ENV=prod, APP_DEV=, XDEBUG_MODE=, BLACKFIRE-*=XXX	Request will take 4 sec, has overhead of the Blackfire running in the background

## 1.8.8 If you use the env APP\_DEV set to dev the system is slow

- Check the logs. There might be to many unseen errors.

## 1.8.9 If you use the env XDEBUG\_MODE set to dev the system is slow

- Check /tmp/xdebug.log for connection errors, this mean you have not startet the remote debugging
- If you do not need to debug pph code remove XDEBUG\_MODE from env

## 1.8.10 You see an out of memory error

- You might have var\_dump in the code.

🕒 28. Oktober 2024

## 1.9 Guide for Administrators

---

### 1.9.1 Grant Access via a auth provider for Kubernetes

---

- [Grant a Gitlab User Access](#)

#### Setup a new Application Token

#### Add the gitlab auth provider

```

oauthConfig:
  assetPublicURL: https://openshift.XXX.xrow.net:8443/console/
  grantConfig:
    method: auto
  identityProviders:
  - challenge: true
    login: true
    mappingMethod: lookup
    name: gitlab
    provider:
      apiVersion: v1
      clientID: XXX
      clientSecret: XXX
      kind: GitLabIdentityProvider
      legacy: false
      url: https://gitlab.com/

```

### 1.9.2 Map an identity

---

#### Create a cluster admin user

```

oc create identity gitlab:44745
oc create useridentitymapping gitlab:44745 admin

```

#### Create a namespace admin

```

oc create identity gitlab:44745
oc create user service@xrow.de --full-name="xrow GmbH"
oc policy add-role-to-user admin service@xrow.de -n project-name-production
oc create useridentitymapping gitlab:44745 service@xrow.de

```

#### Create a namespace developer:

Uses the special role developer-operator-developer.

```

oc create identity gitlab:44745
oc create user service@xrow.de --full-name="xrow GmbH"
oc policy add-role-to-user developer-operator-developer service@xrow.de -n project-name-production
oc create useridentitymapping gitlab:44745 service@xrow.de

```

#### Revoke access

```

oc delete useridentitymapping gitlab:44745

```

 28. Oktober 2024