Kumori PaaS - Getting Started

Kumori Systems v1.0.0, June 2018

Table of Contents

This document provides a quick demonstration of how to try Kumori PaaS platform and deploy your first **Hello-World** application in less than 5 minutes.

Kumori PaaS lets you create and deploy scalable services, managing the complexity of their life cycle, letting you focus on their functionality.

Building complex services is simple when following our service model. Our tools let you get started quickly.

1. A simple example

Before working with Kumori PaaS you need to sign up (if you have not done so already).

After signing up you will be able to interact with the platform and manage your services through its Dashboard.

We also offer a command-line interface tool, that you will be using in what follows.

1.1. Prerequisites

You will need docker installed in your machine.

As stated in the Docker installation guide, to use Docker as a non-root user, you should add your user to the "docker" group with:
`sudo usermod -aG docker $USER`

You also need nodejs 8.9 or later (with npm) to install and run the tools.

Then you can install Kumori CLI as follows:

```
sudo npm install -g @kumori/cli
```

1.2. Simple express server

Create a directory for your project, and go into it:

```
mkdir simpleweb
cd simpleweb
```

Initialize the project workspace:

```
kumori init
```

Kumori's naming scheme for services and components uses a domain name belonging to the user to avoid name collisions. You should now configure that domain name with the CLI for your

project:

```
kumori set domain <your_domain>
```

Now we are ready to start building our service. Our simple service will have just one component (the web server). We offer a set of templates which you can use to quickly implement simple service patterns. We make use of one of them:

```
kumori component add -t hello-world <name>
```

This creates a component skeleton within the `components/<your_domain>/<name>` directory, which we will leave untouched for this example. Note that the internal name this component will receive is `eslap://<your_domain>/component/<name>/0_0_1` (further details can be found in the Building Services for the Kumori PaaS guide).

We now need to build the component:

```
kumori component build <name>
```

A component must be inserted within a service application as one of its roles in order to be deployable.

We create a service application making use of another template:

```
kumori service add -t hello-world <name>
```

Note that, currently, we should use the same **<name>** for the service application, as we used for the component. Future versions of the tool will provide more flexibility. Also note that the internal name of the created service app is `eslap://<your_domain>/service/<name>/0_0_1`.

We are ready to prepare the deployment. Run the following command:

```
kumori deployment add <deployment_nickname> <name>
```

Where `<deployment_nickname>` is an arbitrary name you give your deployment, and `<name>` is the name you gave your service application.

Before we can actually deploy, we need to configure our workspace with a valid API access token. This token can be obtained from Kumori Dashboard - Settings page (three-dot menu at the top right).

With the token in our hands we use:

```
kumori stamp update -t <token> baco
kumori stamp use baco
```

baco is the current production version of Kumori PaaS.

And now we can deploy like so:

```
kumori deployment deploy <deployment_nickname>
```

Which will return us an URL to browse the service we just deployed.

To learn more, continue reading the Quick Start guide you can find in this same repo.