# Contributing to Che

- [Other Che repositories](#other-che-repositories)

- [Devfile to contribute](#devfile-to-contribute)

- [Contribute to ...](#contribute-to-...)

- [Dashboard](#dashboard)

- [Che Server a.k.a WS master](#che-server-a.k.a-ws-master)

## Other Che repositories

Che is composed of multiple sub projects. For each projects we provide a \*CONTRIBUTE.md\* file describing how to setup the development environment to start your contribution. Most of the time, we encourage you to use Che to contribute to Che.

<!-- begin repository list -->

Repository | Component | Description | Devfile | Documentation

--- | --- | --- | --- | ---

[che](https://github.com/eclipse/che) | | (this repository) the main project repository | [devfile](https://github.com/eclipse/che/blob/master/devfile.yaml) | [doc](https://github.com/eclipse/che/blob/master/CONTRIBUTING.md#che-server-aka-ws-master)

---| [dockerfiles](https://github.com/eclipse/che/tree/master/dockerfiles) | source code, dockerfiles to build our main docker images. Note that Che-theia related dockerfiles are located in che-theia repo. | | |

---| [wsmaster](https://github.com/eclipse/che/tree/master/wsmaster) | orchestrates the Che workspaces with devfiles on Kubernetes | | |

---| [tests](https://github.com/eclipse/che/tree/master/tests) | source code of our integration tests. | | |

[che-theia](https://github.com/eclipse/che-theia) | | Theia IDE integrated in Che. | [devfile](https://github.com/eclipse/che-theia/blob/master/devfiles/che-theia-all.devfile.yaml) | [doc](https://github.com/eclipse/che-theia/blob/master/CONTRIBUTING.md)

---| [generator](https://github.com/eclipse/che-theia/tree/master/generator) | `che:theia init` CLI to prepare and build che-theia | | |

[chectl](https://github.com/che-incubator/chectl) | | The CLI to install Che, create and start workspaces and devfiles | [devfile](https://github.com/che-incubator/chectl/blob/master/devfile.yaml) | [doc](https://github.com/che-incubator/chectl/blob/master/CONTRIBUTING.md)

[dashboard](https://github.com/eclipse/che-dashboard) | | UI to manage workspaces, devfiles, etc. | [devfile](https://github.com/eclipse/che-dashboard/blob/master/devfile.yaml) | [doc](https://github.com/eclipse/che/blob/master/CONTRIBUTING.md#dashboard)

[devfile-registry](https://github.com/eclipse/che-devfile-registry) | | The default set of devfiles that would be made available on the Che dashboard stacks. | |

[docs](https://github.com/eclipse/che-docs) | | Eclipse Che documentation https://www.eclipse.org/che/docs/ source code. | [devfile](https://github.com/eclipse/che-docs/blob/master/devfile.yaml) | [doc](https://github.com/eclipse/che/blob/master/CONTRIBUTING.md#dashboard)

[machine-exec](https://github.com/eclipse/che-machine-exec) | | Interface to execute tasks and terminals on other containers within a workspace. | [devfile](https://github.com/eclipse/che-machine-exec/blob/master/devfile.yaml) | [doc](https://github.com/eclipse/che-machine-exec/blob/master/CONTRIBUTING.md)

[operator](https://github.com/eclipse/che-operator) | | Che operator to deploy, update and manage K8S/OpenShift resources of Che. | [devfile](https://github.com/eclipse/che-operator/blob/master/devfile.yaml) |

[plugin-broker](https://github.com/eclipse/che-plugin-broker) | | The workspace microservice that is in charge of analyzing, preparing and installing the workspace components defined in a Devfile. | [devfile](https://github.com/eclipse/che-plugin-broker/blob/master/devfile.yaml) | [doc](https://github.com/eclipse/che-plugin-broker/blob/master/CONTRIBUTING.md)

[plugin-registry](https://github.com/eclipse/che-plugin-registry) | | The default set of Che plugins (vscode extension + containers) or editors that could be installed on any Che workspaces. | |

[website](https://github.com/eclipse/che-website) | | https://eclipse.org/che website source code. | [devfile](https://github.com/eclipse/che-website/blob/master/devfile.yaml) |

[workspace-client](https://github.com/eclipse/che-workspace-client) | | JS library to interact with a che-server. | |

[workspace-loader](https://github.com/eclipse/che-workspace-loader) | | UI to load workspaces within dashboard | [devfile](https://github.com/eclipse/che-workspace-loader/blob/master/devfile.yaml) | [doc](https://github.com/eclipse/che-workspace-loader/blob/master/CONTRIBUTING.md)

[che-sidecar-bazel](https://github.com/che-dockerfiles/che-sidecar-bazel) | | Eclipse Che Sidecar container for Bazel tooling | |

[che-sidecar-protobuf](https://github.com/che-dockerfiles/che-sidecar-protobuf) | | Eclipse Che Sidecar container for Protobuf tooling | |

[che-sidecar-tekton](https://github.com/che-dockerfiles/che-sidecar-tekton) | | Eclipse Che Sidecar container for Tekton tooling | |

[che-sidecar-kubernetes-tooling](https://github.com/che-dockerfiles/che-sidecar-kubernetes-tooling) | | Eclipse Che Sidecar container for Kubernetes tooling | |

[che-sidecar-openshift-connector](https://github.com/che-dockerfiles/che-sidecar-openshift-connector) | | Eclipse Che Sidecar container for OpenShift connector tooling | |

[che-container-tools](https://github.com/che-dockerfiles/che-container-tools) | | Base image used for sidecars that service container tooling plugins | |

[che-sidecar-python](https://github.com/che-dockerfiles/che-sidecar-python) | | Eclipse Che Sidecar container for python tooling | |

[che-custom-nodejs-deasync](https://github.com/che-dockerfiles/che-custom-nodejs-deasync) | | Provides a custom nodejs binary embedding deasync node-gyp module as builtin module | |

[che-sidecar-go](https://github.com/che-dockerfiles/che-sidecar-go) | | Eclipse Che sidecar container for go | |

[che-sidecar-sonarlint](https://github.com/che-dockerfiles/che-sidecar-sonarlint) | | Eclipse Che sidecar container for sonalint extension | |

[che-sidecar-dotnet](https://github.com/che-dockerfiles/che-sidecar-dotnet) | | Eclipse Che sidecar container for dotnet | |

[che-sidecar-shellcheck](https://github.com/che-dockerfiles/che-sidecar-shellcheck) | | Eclipse Che sidecar container for shellcheck | |

[che-sidecar-camelk](https://github.com/che-dockerfiles/che-sidecar-camelk) | | Eclipse Che sidecar container for camelk | |

[che-sidecar-vale](https://github.com/che-dockerfiles/che-sidecar-vale) | | Eclipse Che sidecar container for vale | |

[che-cert-manager-ca-cert-generator-image](https://github.com/che-dockerfiles/che-cert-manager-ca-cert-generator-image) | | CA cert generation job image used by chectl | |

[che-buildkit-base](https://github.com/che-dockerfiles/che-buildkit-base) | | Eclipse Che Sidecar container for buildkit tooling | |

[che-sidecar-scala](https://github.com/che-dockerfiles/che-sidecar-scala) | | Eclipse Che Sidecar container for scala tooling | |

[che-buildah-base](https://github.com/che-dockerfiles/che-buildah-base) | | Use this image to build docker images using buildah | |

[che-docker-registry-image-copier](https://github.com/che-dockerfiles/che-docker-registry-image-copier) | | copy images between public and private docker registry inside k8s cluster | |

[che-php-base](https://github.com/che-dockerfiles/che-php-base) | | Base image to be used for the PHP devfile | |

[che-tls-secret-creator](https://github.com/che-dockerfiles/che-tls-secret-creator) | | This images generates TLS certificates | |

[build-action](https://github.com/che-dockerfiles/build-action) | | Custom GitHub Action for building sidecar Dockerfiles | |

[che-sidecar-podman](https://github.com/che-dockerfiles/che-sidecar-podman) | | Eclipse Che Sidecar container for podman tooling | |

[che-sidecar-clang](https://github.com/che-dockerfiles/che-sidecar-clang) | | Eclipse Che Sidecar container for clang tooling | |

[che-sidecar-php](https://github.com/che-dockerfiles/che-sidecar-php) | | Eclipse Che Sidecar container for php tooling | |

[che-sidecar-java](https://github.com/che-dockerfiles/che-sidecar-java) | | Eclipse Che Sidecar container for java tooling | |

[che-sidecar-dependency-analytics](https://github.com/che-dockerfiles/che-sidecar-dependency-analytics) | | Eclipse Che Sidecar container for dependency analytics tooling | |

[che-sidecar-node](https://github.com/che-dockerfiles/che-sidecar-node) | | Eclipse Che Sidecar container for node tooling | |

[che-dashboard-next](https://github.com/che-incubator/che-dashboard-next) | | New dashboard for Eclipse CHE | |

[che-theia-openshift-auth](https://github.com/che-incubator/che-theia-openshift-auth) | | OpenShift authentication plugin | |

[configbump](https://github.com/che-incubator/configbump) | | Simple Kubernetes controller that is able to quickly synchronize a set of config maps | |

[workspace-data-sync](https://github.com/che-incubator/workspace-data-sync) | | Provides the ability to increase I/O performance for a developer workspaces | |

[che-workspace-telemetry-client](https://github.com/che-incubator/che-workspace-telemetry-client) | | abstract telemetry API and a Typescript implementation of the API. | |

[kubernetes-image-puller](https://github.com/che-incubator/kubernetes-image-puller) | | ensures that all nodes in the cluster have those images cached | |

<!-- end repository list -->

## Devfile to contribute

We are trying to provide a devfile for each areas where you could contribute. Each devfile could be run on any Che instances to setup a \*ready-to-code\* developer environment. Beware that each of them may need a certain amount of memory.

Devfile could be launched through a factory or [chectl](https://github.com/che-incubator/chectl) cli.

```bash

$ chectl workspace:start -f devfiles/che-theia-all.devfile.yaml

```

or

```bash

$ chectl workspace:start -f https://raw.githubusercontent.com/eclipse/che-theia/master/devfiles/che-theia-all.devfile.yaml

```

or `https://<CheInstance>/f?url=https://raw.githubusercontent.com/eclipse/che-theia/master/devfiles/che-theia-all.devfile.yaml`

## Contribute to ...

Let's cover the developer flow for theses projects:

### Dashboard

Dashboard source code is located in [https://github.com/eclipse/che-dashboard](https://github.com/eclipse/che-dashboard) repository.

It is an AngularJS application. Here is the developer workflow if you want to contribute to it:

#### Devfile for dashboard development

The devfile: [https://github.com/eclipse/che-dashboard/blob/master/devfile.yaml](https://github.com/eclipse/che-dashboard/blob/master/devfile.yaml)

In this section, we show how to setup a Che environment to work on the Che dashboard, and how to use it.

For the whole workflows, we will need a workspace with such containers:

- Dashboard Dev container (a.k.a dash-dev): Dashdev is a all in one container for running commands such as build, test or start the dashboard server.

All containers have `/projects` folder mounted, which is shared among them.

Developer workflow:

1. Start the workspace with the devfile, it is cloning Che repo.

2. Build

3. Code ...

4. Run unit test

5. Start dashboard server and preview

#### Step 1: Start the workspace with the devfile, it is cloning Che repo.

In this section we are going to start a new workspace to work on che-theia. The new workspace will have few projects cloned: `theia` and `che-theia`. It will also setup the containers and commands in the `My workspace` view. We will use these commands in the next steps.

The devfile could be started using `chectl`:

```bash

$ chectl workspace:start -f https://raw.githubusercontent.com/eclipse/che-dashboard/master/devfile.yaml

```

#### Step 2: Build

In this section we are going to build the dashboard project.

You can use the Che command `dashboard\_build` (command pallette > Run task > or containers view)

Basically, this command will run

```bash

# [dash-dev]

$ yarn

```

#### Step 3: Code ...

#### Step 4: Run unit test (optional)

In this step, we will run the Dashboard unit tests:

You can use the Che command `dashboard\_test` (command pallette > Run task > or containers view)

Basically, this command will run

```bash

# [dash-dev]

$ yarn test

```

#### Step 5: Start dashboard server and preview

In this step, we will run the dashboard server and see the live reloadable preview.

You can use the Che command `dashboard\_dev\_server` (command pallette > Run task > or containers view)

```bash

# [dashboard\_dev\_server]

$ node\_modules/.bin/gulp serve --server=<che\_api\_url>

```

### Che server a.k.a WS master

There is a [devfile](https://github.com/eclipse/che/blob/master/devfile.yaml) for development of Che server in Che.

To build Che one may run a predefined build task from the devfile.

Starting Che master requires some manual steps.

Open a terminal in runtime container (`che-server-runtime`) and perform:

- First, set `CHE\_HOME` environment variable with absolute path to parent folder of Che master's Tomcat.

It might look like `/projects/che/assembly/assembly-main/target/eclipse-che-\*-SNAPSHOT/eclipse-che-\*-SNAPSHOT`.

- Then set `CHE\_HOST` with the endpoint of new Che master.

If using the [devfile](devfile.yaml) the endpoint is `che-dev` and already set.

- After, set `CHE\_INFRASTRUCTURE\_ACTIVE` according to your environment.

For example: `openshift` (note, use `kubernetes` and `openshift` insted of `minikube` and `minishift` correspondingly).

- Run `/entrypoint.sh`.

After this, new Che master should be accesible from the `che-dev` endpoint.

To reach Swagger use url from `che-dev` endpoint with `/swagger` suffix.

To start a workspace from Che server under development some additional configuration of the cluster is needed.

One should add rights for the service account to be able to perform all needed for Che server actions.

Example for Openshift (in case of Kubernetes replace `oc` with `kubectl`):

```bash

cat << EOF | oc apply -f -

apiVersion: rbac.authorization.k8s.io/v1

kind: RoleBinding

metadata:

labels:

app: che

component: che

name: che-workspace-admin

namespace: che

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole

name: admin

subjects:

- kind: ServiceAccount

name: che-workspace

namespace: che

EOF

```

Also `CHE\_API\_INTERNAL`, `CHE\_API\_EXTERNAL` and `CHE\_API` should be set in runner container and point to new Che server API.

If one uses provided devfile, they are already set to: `http://che-dev:8080/api`, which should be changed in case of https protocol.