Using Launch with CLI

Launch is a deployment platform that enables you to host your Contentstack-powered Jamstack website instantly. Launch hosts websites managed by any CMS (headless or traditional).

You can create a new project in Launch and link it with your GitHub repository. Launch picks that code and deploys your website instantly. Launch also allows you to create a project by directly uploading a project folder or a .zip file.

This step-by-step guide lets you perform different operations in Launch using the CLI.

Prerequisites

Note: This prerequisite is applicable only if you are importing a project via GitHub.

- Contentstack account
- CLI installed and configured (version 1.6.0 and above)
- CLI authenticated
- Access to Launch for your organization
- GitHub account where your website code is hosted

Steps for execution

1. Open a terminal and fire the launch command:
    - Run the below command if you are in the current working directory.

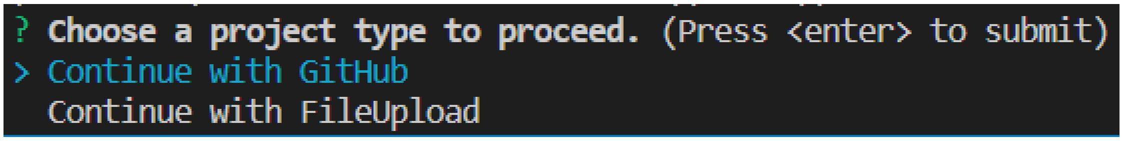
      ```
      $ csdx launch
      ```

    - If you are in a different directory, provide the path of the current working directory.

      ```
      $ csdx launch –data-dir
      ```

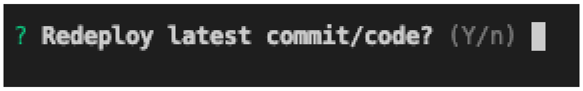
2. Note: Launch automatically identifies Git projects.

3. If you are creating a project by uploading a file, or if the GitHub is not already connected, you will be prompted to choose a project type to proceed. Select Continue with GitHub or Continue with FileUpload to proceed.



```
? Choose a project type to proceed. (Press <enter> to submit)
> Continue with GitHub
  Continue with FileUpload
```

4. Select the organization where you want to create your project.
5. In this step, select a branch if you decided to create your project using GitHub.
6. Enter a name for your project.
7. Enter a name for your Environment or click Enter to proceed with the Default environment.
8. By default, Launch identifies the framework. Select a Framework Preset if you want to change the default framework, and then click Enter.
9. In the Build Command section, perform one of the following:
    - Click Enter if you want to proceed with the default Build command npm run build.
    - Enter a new build command or update the existing command if you want to customize the command.
10. Enter the output path in the Output Directory section or click Enter to proceed with the default output path.
11. In the next step, follow the instructions provided on screen to perform one of the following:
    - Import variables from a stack
    - Manually add custom variables to the list
    - Import variables from the local env file
12. A prompt, Would you like to add more variables.? appears.
    - Enter Y to add another custom variable.
    - Enter n to proceed with the project deployment.
13. You have successfully deployed a Launch project using the CLI.
14. To redeploy/re-initialize an existing project, run the launch command with the existing project’s directory path.
    You will get a prompt as displayed in the screenshot below.



```
? Redeploy latest commit/code? (Y/n) █
```

    - Click Y to redeploy the project.
      If you are redeploying using the File Upload method, then you will get a prompt Redeploy with last file upload?.
        1. Click y if you want to redeploy with the last file upload.
        2. Click N to redeploy by uploading a new file.
    - Note: By default, Launch assumes to redeploy a new file.
    - Click N to terminate the process.

Usage

```
$ csdx launch
```

Options

- -d, --data-dir=data-dir: Current working directory.
- -c, --config=config: Path to the local config file.
- --type=type: Optional type of adapters. Options [GitHub or FileUpload].
- -e, --environment=environment: [optional] Environment name for launch project.
- -n, --name=name: [optional] Name of project.
- --branch=branch: [optional] GitHub Branch name.
- --build-command=build-command: [optional] Build Command.
- --framework=: [optional] Type of framework. Options [GATSBY|NEXTJS|OTHER]
- --org=org: [optional] Provide the organization's UID to create a new project or deployment.
- --out-dir=: [optional] Output Directory.
- --init: [Optional, Hidden] Reinitialize the project if it is an existing Launch project.

Examples

- To create a Launch project by providing the config file path and file type:

  ```
  $ csdx launch --config --type
  ```

- To create a Launch project by providing the directory path and file type:

  ```
  $ csdx launch --data-dir --type
  ```

Note: When there are multiple branches identified in the configuration, you will be prompted to Choose a branch to proceed with the Launch operations. A single config file can have n number of configurations based on the branch.

Logs

You can fetch the previous deployment logs and the latest server logs for Launch projects in CLI using the launch:logs command.

Steps for execution

```
$ csdx launch:logs
```

1. Open a terminal and fire the launch:logs command:
2. Select the organization which has your project.
3. Select your project for which you want to fetch the logs.
4. Select the required environment.
5. You have successfully fetched the logs for your selected project.
6.

Usage

```
$ csdx launch:logs
```

Options

- -d, --data-dir=data-dir: Current working directory.
- -c, --config=config: Path to the local config file.
- -e, --environment=environment: Environment name or UID.
- --deployment=deployment: Deployment number or uid.
- --type=type: Optional type of flags to show logs. By default, these are server logs. Options [d - deployment logs, s - server logs]

Note: If you do not pass a deployment ID, by default launch fetches the latest deployment logs.

Examples

- To fetch Launch project logs based on the environment number and type of flags:

  ```
  $ csdx launch:logs -e "environment number or uid" --type "types of flags"
  ```

- To fetch Launch project logs based on the environment number and deployment number:

  ```
  $ csdx launch:logs -e "environment number or uid" --deployment "deployment number or uid"
  ```

Functions

You can test your Launch project Cloud Functions locally using the launch:functions command in CLI.

Usage

```
$ csdx launch:functions
```

Options

- -d, --data-dir=data-dir: Current working directory.
- -c, --config=config: Path to the local config file.
- -p, --port=port: Port number.

Examples

- To test your Launch project Cloud Function locally:

  ```
  $ csdx launch:functions
  ```

- To test your Launch project Cloud Function locally in a specific port:

  ```
  $ csdx launch:functions -p "port number"
  ```

Deployments

You can display the list of deployments for an environment for Launch projects using the launch:deployments command in CLI.

Steps for execution

```
$ csdx launch:deployments
```

1. Open a terminal and fire the launch:deployments command:
2. Select the organization which has your Launch project.
3. Select your project.
4. Select the environment for which you want to list the deployments.

You have successfully listed the deployments for the selected environment.

Usage

```
$ csdx launch:deployments
```

Options

- -d, --data-dir=data-dir: Current working directory.
- -c, --config=config: Path to the local '.cs-launch.json' file.
- -e, --environment=environment: Environment name or UID.
- -–org=org: [Optional] Provide the organization UID.
- -–project=project: [Optional] Provide the project UID.

Examples

```
$ csdx launch:deployments -d "current working directory"
```

```
$ csdx launch:deployments -e "environment number or uid"
```

- To list the deployments in your current working directory
- To list the deployments for a given environment

Environments

You can display the list of environments that are available for a particular project using the launch:environments command in CLI.

Steps for execution

```
$ csdx launch:environments
```

1. Open a terminal and fire the launch:environments command:
2. Select the organization which has your project.
3. Select your project.

You have successfully listed the environments for the selected project.

Usage

```
$ csdx launch:environments
```

Options

- -d, --data-dir=data-dir: Current working directory.
- -c, --config=config: Path to the local '.cs-launch.json' file.
- -–org=org: [Optional] Provide the organization UID.
- -–project=project: [Optional] Provide the project UID.

Examples

```
$ csdx launch:environments -d "current working directory"
```

```
$ csdx launch:environments --org= --project=
```

- To list the environments in your current working directory
- To list the environments for a specific project under a given organization

Open

You can open the live site for an environment of your Launch projects using the launch:open command in CLI.

Steps for execution

```
$ csdx launch:open
```

1. Open a terminal and fire the launch:open command:
2. Select the organization which has your project.
3. Select your project.
4. Select the environment for which you want to view the live site.

You have successfully opened the live site for the selected environment.

Usage

```
$ csdx launch:open
```

Options

- -d, --data-dir=data-dir: Current working directory.
- -c, --config=config: Path to the local '.cs-launch.json' file.
- -e, --environment=environment: Environment name or UID.
- -–org=org: [Optional] Provide the organization UID.
- -–project=project: [Optional] Provide the project UID.

Examples

```
$ csdx launch:open --environment=environment --data-dir
```

```
$ csdx launch:open --environment=environment --config
```

- To open the website for an environment by passing your current working directory
- To open the website for an environment for a given configuration