DTI Guide: Provisioning

Provisioning deploys code (in a package) onto a tenant and tag on the C3 Cluster. At a high level, a package is defined as a specific set of files and directories defining the Types, configuration, runtimes, and initial data for the deployment. Once a package is provisioned to a tenant/tag, users can access documentation, fetch Types, evaluate metrics, and perform more advanced tasks such as training machine learning models or running batch jobs.

To provision code to a tenant/tag you first need to understand what exactly a package is and how it is structured. Generally speaking, packages follow a specific directory structure (e.g., all .c3typ files are located in the `src` directory). C3 AI official documentation on this matter is quite extensive:

- **C3 package structure**
- **Provisioning**

Provisioning Tutorial Video

Your browser does not support the HTML5 video element

Prerequisites

Please see the **DTI Readiness Checklist** to ensure you are ready to begin provisioning. You must be able to access a C3 Cluster to complete any of the steps on this page.

Please contact help@c3dti.ai if you can't access your c3 tenant/tag.

COVID-19 Data Lake Provisioning

To deploy the COVID-19 Data Lake to your tenant/tag, please clone the base COVID-19 package located **here**. Then, following either of the methods below, provision the '`baseCovidDatalake`' package from the `dtiTraining` subdirectory.

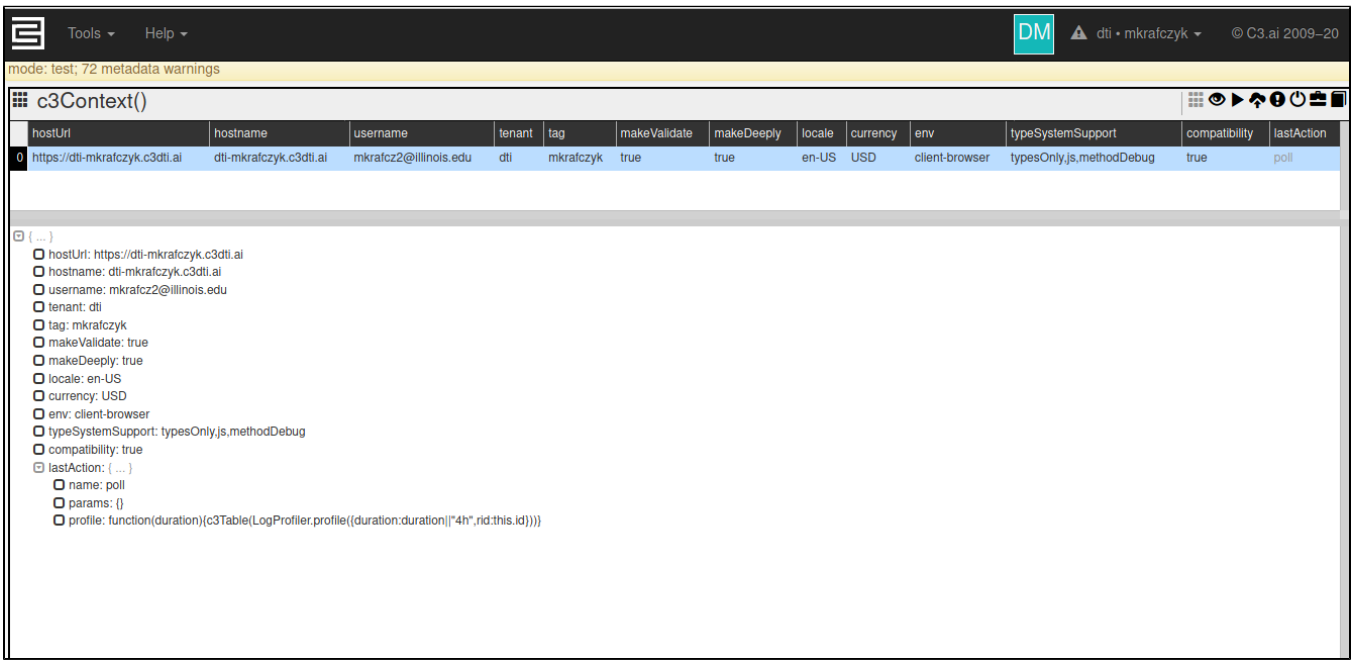
Provisioning Methods

There are two methods to provision a package:

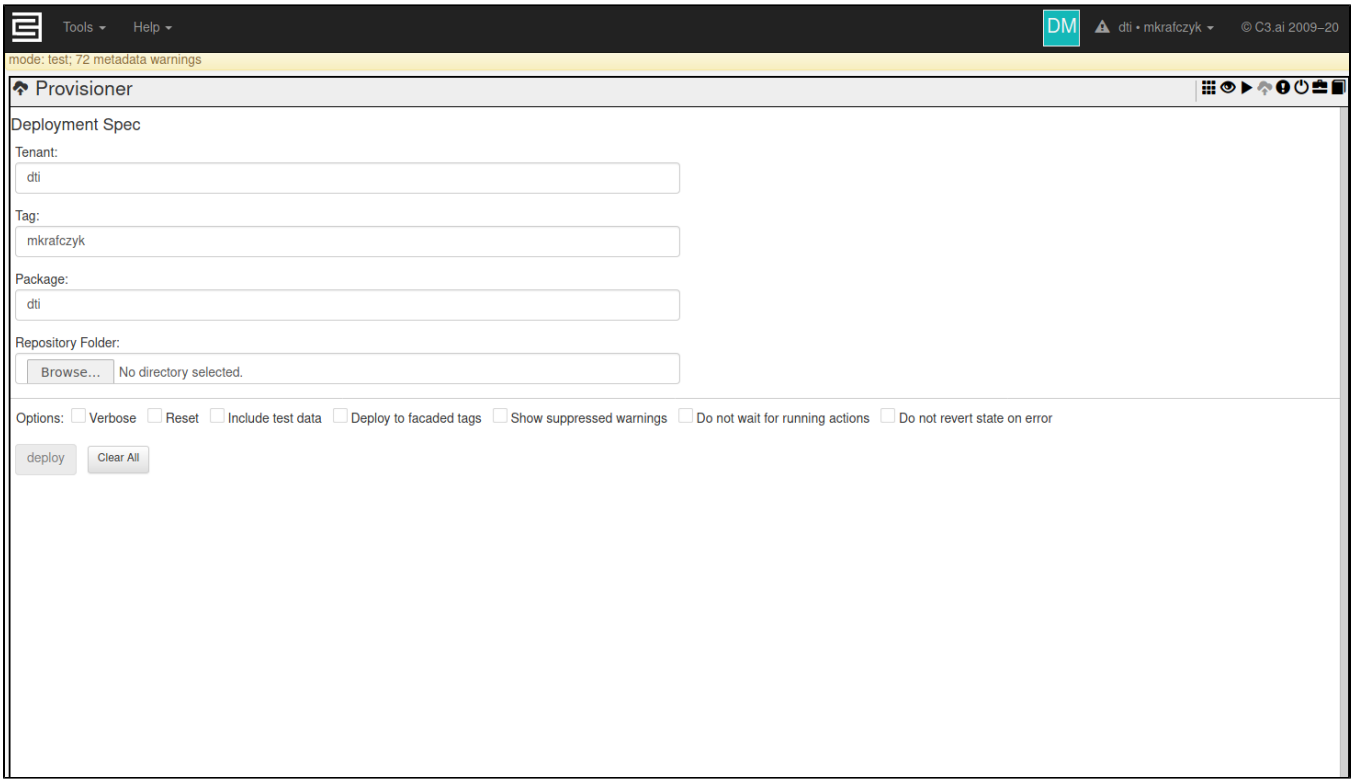
1. Via a web-based provisioner in a tenant/tag's static console.
2. Via C3 AI command line interface (CLI), which enables developers to provision code without a web-browser.

Web Provisioner

After accessing your tenant/tag you should see a page like the following:



In the upper right hand corner, you will see a button featuring a cloud with an upward pointing arrow ( ). Clicking on it opens the provisioner tool:



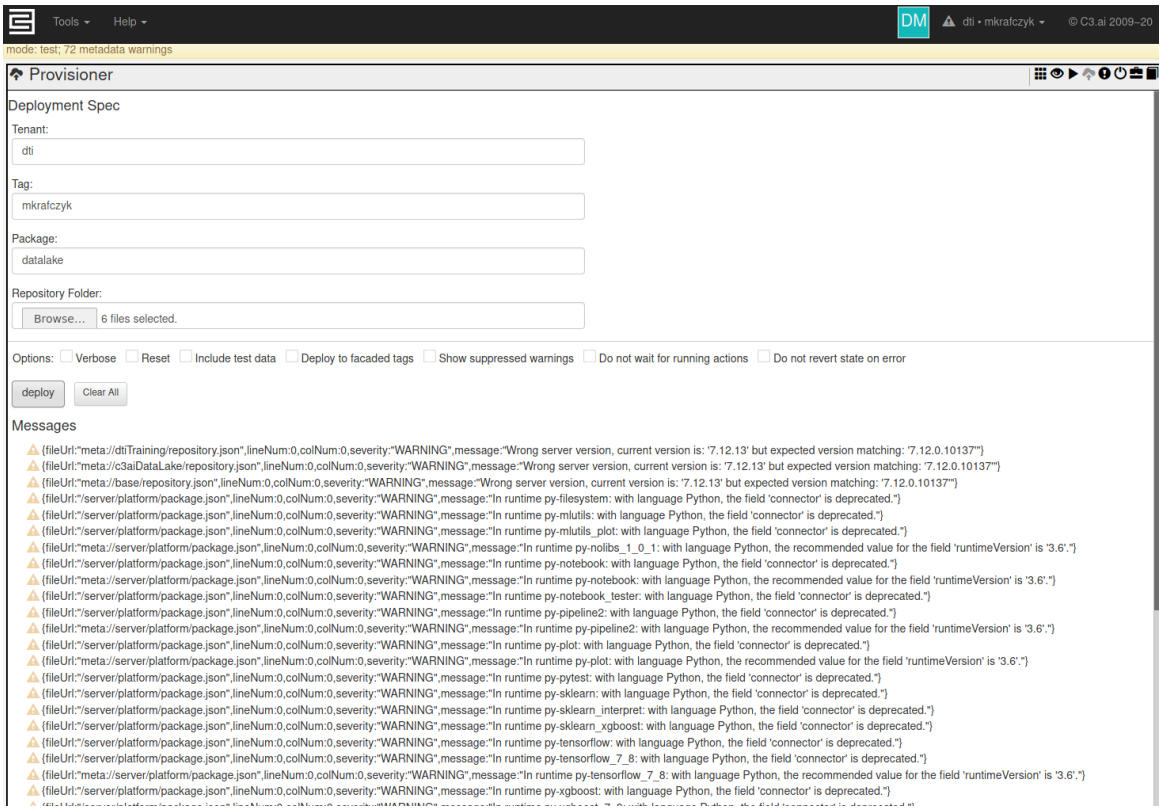
Using this tool is very simple:

1. Enter the name of your tenant in the textbox 'Tenant'.
2. Enter the name of your tag in the textbox 'Tag'.
3. Enter the name of your package in the textbox 'Package'.
4. Click the 'Browse...' button and select the top-level directory of your package (where the repository.json file is located).
5. Click the 'Deploy' button.
6. Wait! **Do not refresh your browser tab!** Once provisioning is complete, your tab will respond again. Provisioning blocks the execution on that tab.

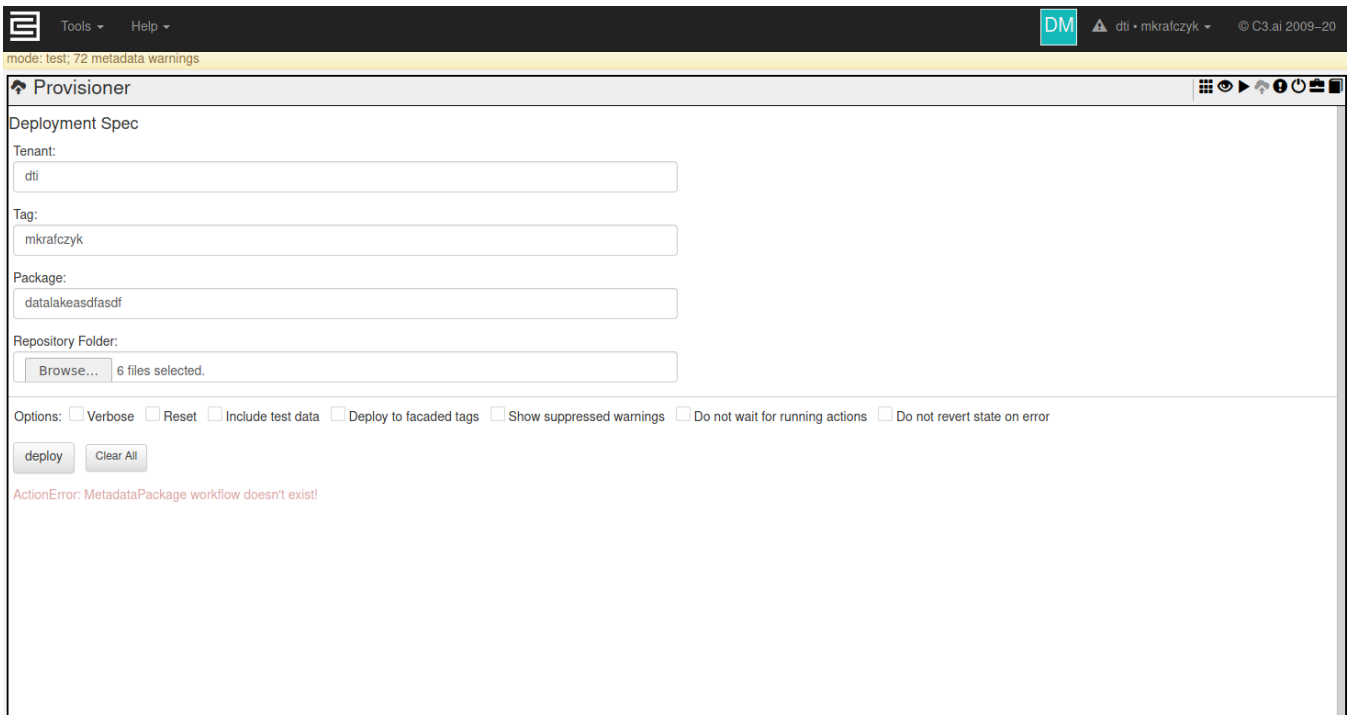
As an example, for the DTI training cluster, you will be assigned a tag which will be related to your or your group's name. For example, professorX's group is assigned the tag 'professorX' on the tenant 'dti', so the 'Tenant' field should say 'dti', the 'Tag' field should say 'professorX'.

Once provisioning completes or fails, your tab should become responsive again, and there should be some messages below the 'Deploy' button. If successful, this will be a list of WARNINGS which can be safely ignored. We will mention here any warnings you need to be concerned about. If provisioning failed, there should be a list of ERROR messages which should give some clue what the problem was.

Here is an example of the provisioner page after a successful provisioning:



And here is the provisioner after a failure:



Command Line Provisioner

See Installing the C3 command line tool.

Before executing this command, navigate to the top-level directory of your package source code. Then, run a form of the following command:

```
c3 prov tag -t <tenant>:<tag> -c <package> -e <vanity_url> -T <auth_token> -a <package_directory> [-E]
```

Replace all <values> above with the following:

- **tenant**: The name of your tenant (e.g., `dti`).
- **tag**: The name of your tag (e.g., `bertsimas`).
- **package**: The name of your package (the '`name`' field in your '`package.json`' file).
- **vanity_url**: The vanity url of your tenant/tag (e.g., `https://dti-berstimas.c3dti.ai`).
- **package_directory**: The path to the directory containing the package you want to deploy. This directory should contain a '`repository.json`' file.
  The '`-a <package_directory>`' option can be left out, however if it is, the current directory is used.
- **auth_token**: A generated authentication token. See below for details.

To generate an authentication token for your package, execute one of the following commands:

```
// From static console
Authenticator.generateC3AuthToken()
```

```
# From a C3-connected Python Jupyter notebook
c3.Authenticator.generateC3AuthToken()
```

The '`-E`' argument is optional. If your package includes test data or Types, `-E` tells the C3 CLI to provision those Types and data too.

Here's an example of a message in the command line interface after a successful provision:

```
[dti/mkrafczyk] Created 2.56Kb ZIP of one package in 0.0s
[dti/mkrafczyk] Server Warnings:
{fileUrl:"meta://dtiTraining/repository.json",lineNum:0,colNum:0,severity:"WARNING",message:"Wrong server
version, current version is: '7.12.13' but expected version matching: '7.12.0.10137'"}
{fileUrl:"meta://c3aiDataLake/repository.json",lineNum:0,colNum:0,severity:"WARNING",message:"Wrong server
version, current version is: '7.12.13' but expected version matching: '7.12.0.10137'"}
....
....
{fileUrl:"meta://server/uiFramework/src/changeLog-uiFramework.c3doc",lineNum:3,colNum:0,severity:"WARNING",
message:"Unknown documentation category 'Change Log'.",listenerType:"DocumentationTopicListener"}
{fileUrl:"meta://server/webdriverProtocol/src/changeLog-webdriverProtocol.c3doc",lineNum:3,colNum:0,severity:"
WARNING",message:"Unknown documentation category 'Change Log'.",listenerType:"DocumentationTopicListener"}
[dti/mkrafczyk] Provisioning completed with some warnings (2020-08-19T17:05:13.116-05:00)
Time taken: 1m 34.145s
Finished
```

After Provisioning

- If you're using the JavaScript static console to interact with your package, refresh your browser tab or execute the command '`c3ImportAll()`'. Once completed, all code in your tenant/tag (e.g., Types, metrics, seed data) should be updated to match your new package.
- If you're using the C3 AI Jupyter service you may need to restart the Jupyter interface.
- If you're connected to C3 AI Suite through a remove python or Jupyter session, you will need to re-run the cell which creates the `` `c3` `` object.

Troubleshooting

Provisioning can fail for many reasons. We list here some common errors along and how to fix them. **If provisioning fails, C3 AI Suite will return your tenant/tag to its last working state.**

If your problem isn't listed here, please contact us at help@c3dti.ai so we can help you out. If your problem is common enough, it'll be added to the list here.

ActionError: MetadataPackage workflow doesn't exist

This means the provisioner can't find the package you specified. Ensure the package name you give the provisioner matches the '`name`' field of your '`package.json`' file.