



Content testing and deployment done the right way

By Shy Ruparel



Content model changes done the right way part 2.

By Shy Ruparel

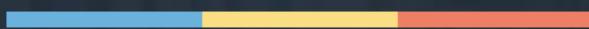


Shy Ruparel

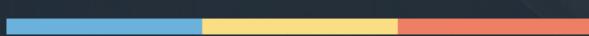
Developer Evangelist

Twitter: [@ShyRuparel](https://twitter.com/ShyRuparel)

Building Software



Nobody gets it right the first time





When you can't have
perfection, the next best thing
is change.

What could go wrong?



In a past meetup we talked about the
Contentful Migration CLI 

TL;DR

What you can do

- Create content type
- Delete content type
- Edit content type
- Create/edit/delete fields
- Change field ID

All in Javascript

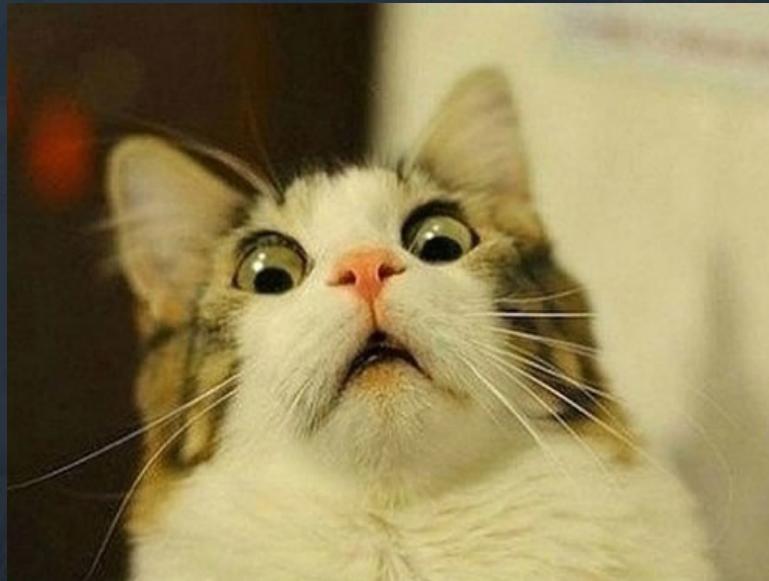
```
contentful-migration
  --space-id $YOUR_SPACE_ID
  --access-token $CONTENTFUL_MANAGEMENT_ACCESS_TOKEN
migration-demo.js
```

Advantages

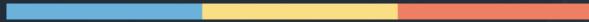
- Repeatable
- Dry-runs
- Can be kept in version control
- Sanity checks
- Use CI to apply.

That's nice

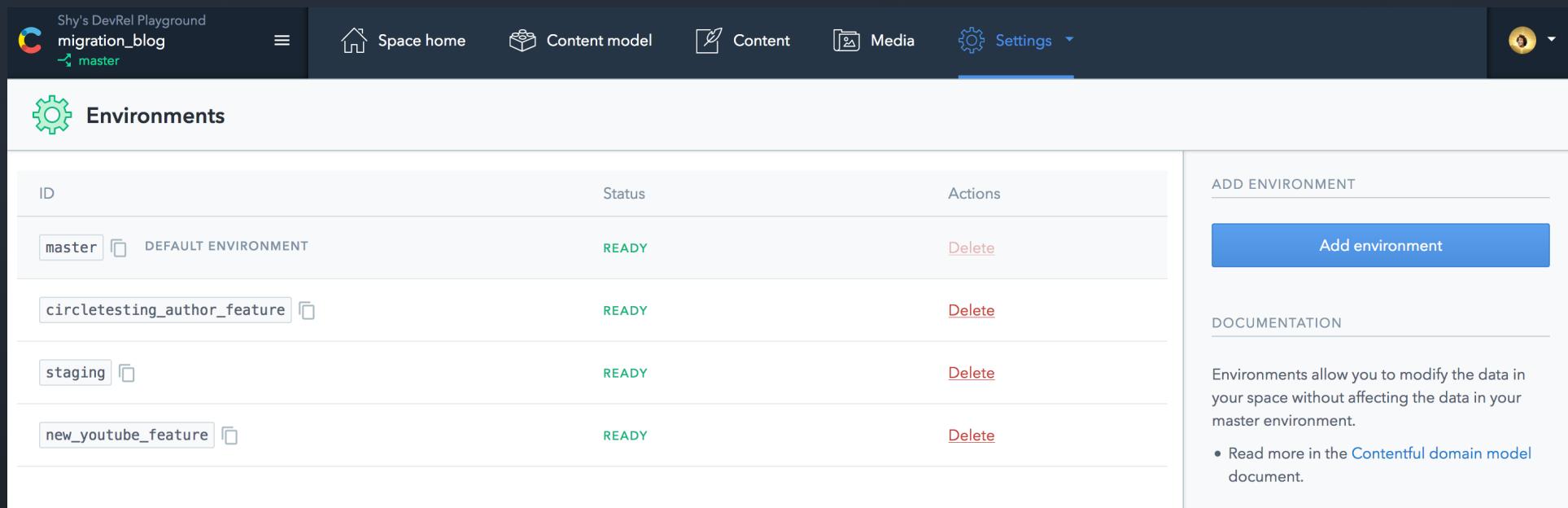
But what if I want to discard, test or preview a migration?



Space Environments



Space environments allow you to create multiple versions of the space, and change them in isolation.



The screenshot shows the Contentful Space environments interface. At the top, there's a navigation bar with the space name "Shy's DevRel Playground" and a branch "migration_blog" (master). The navigation bar includes links for "Space home", "Content model", "Content", "Media", and "Settings". A user profile icon is also present.

The main area is titled "Environments". It displays a table with four rows:

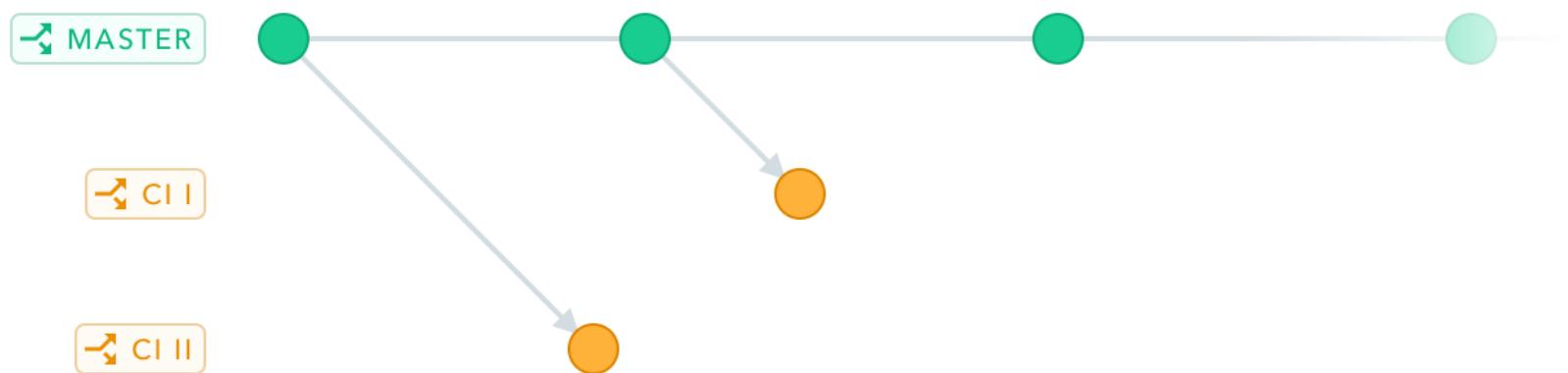
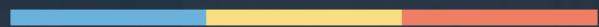
ID	Status	Actions
master	READY	Delete
circletesting_author_feature	READY	Delete
staging	READY	Delete
new_youtube_feature	READY	Delete

To the right of the table, there's a sidebar with sections for "ADD ENVIRONMENT" (containing a blue "Add environment" button) and "DOCUMENTATION" (containing text about environments and a link to the "Contentful domain model" document).

Common uses for space environments

- Local development
- Staging / QA
- Continuous integration

Let's talk about Continuous Integration



A simple Continuous Integration pipeline:

- Build
- Test
- Deploy

Integrating Migrations into CI Pipeline:

- Build
 - *Spin up a copy of the master space*
 - *Migrate copy*
- Test
- Deploy
 - *Migrate master space*



circleci

Build & Test

The screenshot shows a build interface for a project named "Shy". The top navigation bar includes "Updates", "Support", and user profile icons. The main area displays a "Test Summary" tab, a "Queue (00:01)" status, and tabs for "Artifacts", "Configuration", "Build Timing", and "Build Parameters". A "Set Up Test Summary" button is visible. Below this, a green bar indicates a successful build step. The log section, titled "TEST", lists the following steps with their execution times:

Step	Description	Time
0	(00:45)	
Spin up Environment		00:18
Checkout code		00:00
Restoring Cache		00:01
install dependencies		00:14
Saving Cache		00:00
Preparing environment for testing		00:06
run tests		00:04

Deploy

Shy

Builds » Shy » migration-env-demo » master » 65 (deploy) 2.0

Rerun job with SSH

Test Summary Queue (00:02) Artifacts Configuration Build Timing Build Parameters

Show containers: All (1) Successful (1) Failed (0)

0 (00:56) TEST

Spin up Environment 00:26

Checkout code 00:00

Restoring Cache 00:02

install dependencies 00:06

Saving Cache 00:00

Updating Master Environment 00:02

Deploy Master to Heroku 00:17

Set Up Test Summary

WORKFLOWS INSIGHTS ADD PROJECTS TEAM SETTINGS

This screenshot shows a CI/CD pipeline interface for a project named 'Shy' under the 'migration-env-demo' environment. The build number is 65, and it is a deployment (deploy). The status is 2.0, and there is an option to 'Rerun job with SSH'. The pipeline summary shows a green bar for the build duration (00:56) and a blue bar for the test duration (00:02). Below the summary, a list of steps is shown with their execution times: Spin up Environment (00:26), Checkout code (00:00), Restoring Cache (00:02), install dependencies (00:06), Saving Cache (00:00), Updating Master Environment (00:02), and Deploy Master to Heroku (00:17). On the left, a sidebar provides navigation links for Builds, Workflows, Insights, Add Projects, Team, and Settings. The top right features user profile icons for 'Updates' and 'Support'.

CircleCI config.yml - Ready Migrations

```
- run:  
  name: Preparing environment for testing  
  command: |  
    . venv/bin/activate  
    python migration_prep.py "circletesting_${CIRCLE_BRANCH}"  
    node_modules/contentful-migration/bin/contentful-migration  
      --access-token $MANAGEMENT_API_KEY --space-id ${SPACE_ID}  
      --environment-id "circletesting_${CIRCLE_BRANCH}"  
      --yes  
    migration-tests/migration_helper.js
```

migration_prep.py

```
import contentful_management
import os, sys
from dotenv import load_dotenv

SPACE_ID = os.getenv("SPACE_ID")
MANGEMENT_API_KEY = os.getenv("MANGEMENT_API_KEY")
TESTING_ENV = sys.argv[1]

client = contentful_management.Client(MANGEMENT_API_KEY)
environment = client.environments(SPACE_ID)
    .create(TESTING_ENV, {"name": TESTING_ENV})
```

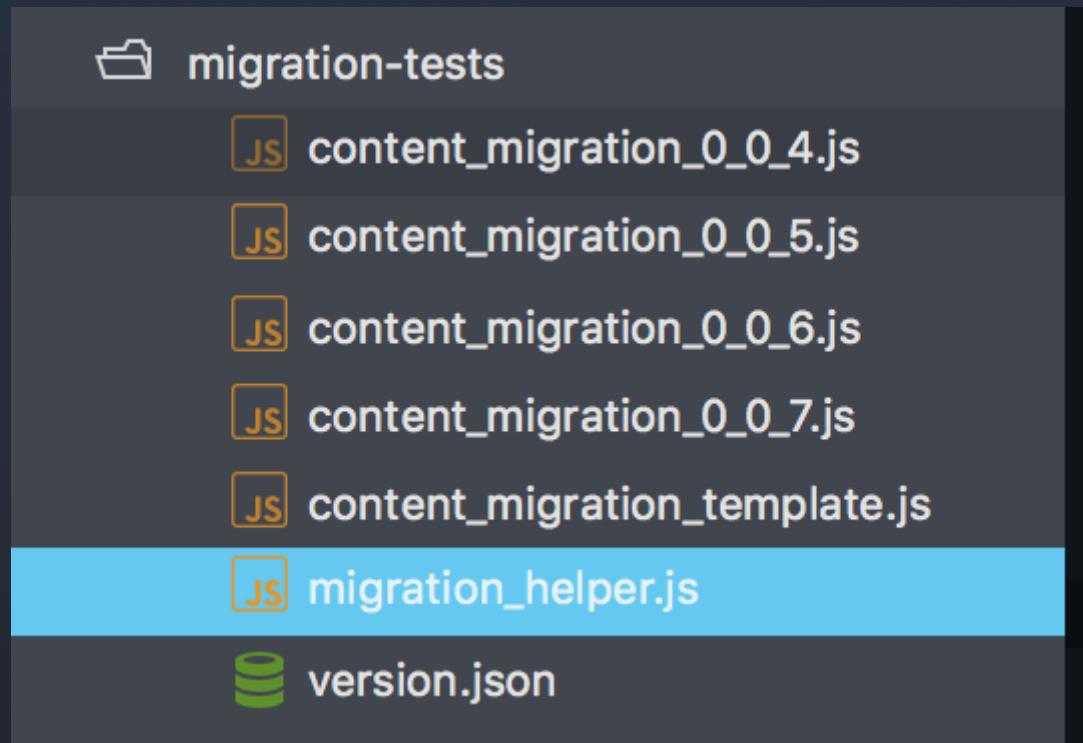
migration_helper.js

```
module.exports = function(migration) {
    var parsedJSON = require("./version.json");
    const expectedVersion = parsedJSON.expectedVersion;
    const upgradeVersion = parsedJSON.upgradeVersion;

    migrationString =
        "./content_migration_" +
        upgradeVersion.split(".").join("_") +
        ".js";
    var migrationCode = require(migrationString);

    if (migrationCode.validateVersion(expectedVersion, upgradever
        migrationCode.runMigration(migration);
    } else {
        console.log("Migration Version Mismatch");
        process.exit(1);
    }
}
```

Migration Files



content_migration_0_0_7.js pt 1

```
const migration_expected_version = "0.0.6";
const migration_upgrade_version = "0.0.7";

function validateVersion(expected_version, upgrade_version) {
    if (upgrade_version == migration_upgrade_version) {
        if (expected_version == migration_expected_version) {
            return true;
        }
    }
    return false;
}
```

content_migration_0_0_7.js pt 2

```
function runMigration(migration) {
  const post = migration.editContentType("post");
  post
    .createField("author")
    .name("author")
    .type("Symbol")
    .required(false);
  return;
}

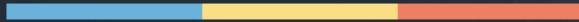
module.exports = {
  runMigration,
  validateVersion
};
```

CircleCI config.yml - Run Tests

```
- run:  
    name: run tests  
    command: |  
        . venv/bin/activate  
        pytest --environment-id="circletesting_{$CIRCLE_BRANCH}"
```

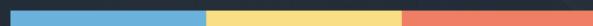
Demo

Integrate evolving your content
model into your delivery pipeline



Check out the example:

github.com/contentful-labs/continuous-delivery-environments-example





Shy Ruparel

Twitter: [@ShyRuparel](https://twitter.com/ShyRuparel)

Email: shy@contentful.com