



# Enter Hyperspace with Golden Paths

A Hitchhiker's Guide to  
Developer Productivity

Josephine Pfeiffer  
Consultant

Janina Sutter  
DevX Specialist

FOREWORD

USEFUL INFO

GOLDEN PATHS

DEMO

WHAT

DON'T PANIC

WHY

HOW

WHERE

WHEN

WHO



## USEFUL INFO

## GOLDEN PATHS

### WHAT IS IT ?



How to organize code  
in a microservices  
architecture

How to handle  
application errors

Which tools to use  
for XYZ

You name it...

**Golden Paths** are a prescribed **practice** that guides Developers and Engineers through the most efficient and effective way to accomplish common tasks.

**Software Templates** is pre-defined **code /configurations** that allow an implementation of Golden Paths in an automated manner.

## USEFUL INFO

## SOFTWARE TEMPLATES

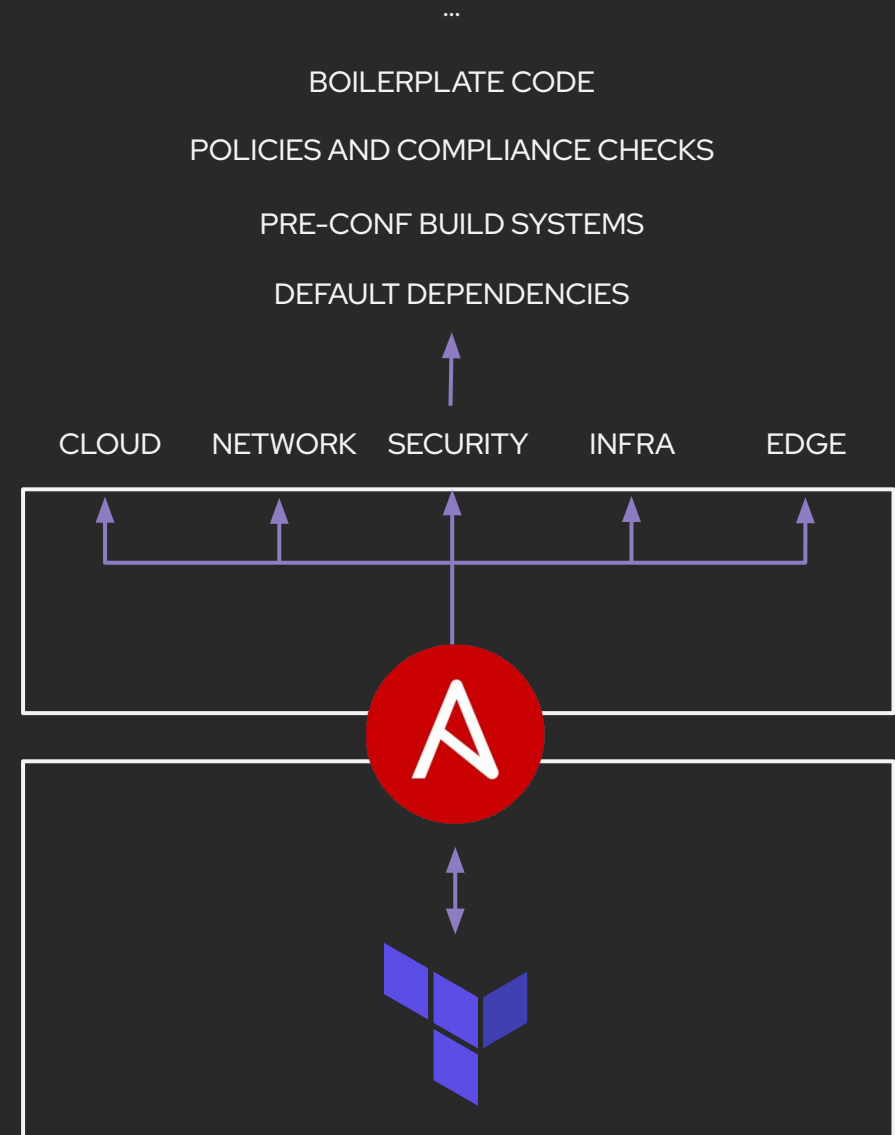
### WHAT IS IT ?

Automatic  
provisioning of  
integrated database

Automatic creation of  
compliant git project

Automatic creation of  
CI/CD pipeline with  
built-in security

You name it...



USEFUL INFO

~~GOLDEN PATHS~~

WHY DO IT ?

"???"

"Finally got my  
Bitbucket!"

"DB...lalala"

"Aarrgghh  
security!"

"Where's that  
confluence page  
again...?!"

"Namespace,  
here I come..."

"Approve!"

USEFUL INFO

~~GOLDEN PATHS~~

WHY DO IT ?

### Infinite Improbability Loop

Time lost on repetitive tasks

### Cognitive Overload

Added infra and security load

### Configuration Drift

Repetition fatigue introduces errors



USEFUL INFO

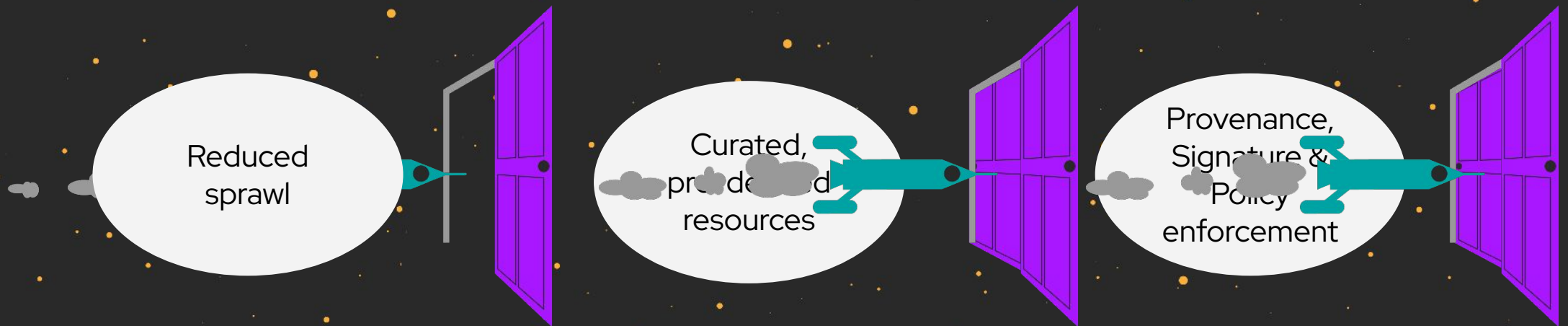
GOLDEN PATHS

WHY DO IT ?

Reduced  
sprawl

Curated,  
provided  
resources

Provenance,  
Signature &  
Policy  
enforcement





USEFUL INFO

GOLDEN PATHS

HOW TO ?



```
input:
  url: https://${{
    parameters.githubHost }}/${{
    parameters.githubOrganization }}/${{
    parameters.repositoryName }}

- id: catalogTemplate
  name: Create catalog-info.yaml
  file
  action: fetch:local
  input:
    url: ../../../../skeletons/
    catalog-info/
    values:
      componentNames:
        - componentNames
      parameters.componentNames:
        - parameters.componentNames
      parameters.componentType:
        - parameters.componentType

- id: publishPR
  name: Open PR with catalog-
    info.yaml
  action: publish:github:pull-
    request
  input:
    repoUrl: ${{
      parameters.githubHost }}/${{
      parameters.githubOrganization }}/${{
      parameters.repositoryName }}
    branchName: add-catalog-info
    title: add catalog-info.yaml
    description: |
      This pull request adds a
      Backstage entity metadata file to
      this repository.

  After this pull request is
  merged, the component will become
  available.

  For more information, read an
  overview of the Backstage software
  catalog at (https://backstage.io/docs/
  features/software-catalog/).

- id: register
  name: Register component
  action: catalog:register
  input:
    repoContentsUrl: https://${{
      parameters.githubHost }}/${{
      parameters.githubOrganization }}/${{
      parameters.repositoryName }}/blob/${{
      publishPR.output.targetBranchName }}
    catalogInfoPath: /catalog-
      info.yaml
    optional: true
  spec:
    title: 'Open PR #${{
      publishPR.output.pullRequestNumber }}'
    url: ${{
      steps.publishPR.output.remoteUrl }}
```

```
default: owner
componentLifecycle:
  title: Lifecycle
  type: string
  description: 'The lifecycle
    state of the component. Well-known and
    common values: experimental,
    production, deprecated.'
  default: unknown

steps:
  - id: fetchRepository
    name: Fetch repository
    action: fetch:plain
    input:
      url: https://${{
        parameters.githubHost }}/${{
        parameters.githubOrganization }}/${{
        parameters.repositoryName }}

  - id: catalogTemplate
    name: Create catalog-info.yaml
    action: fetch:local
    input:
      url: ../../../../skeletons/
      values:
        componentNames: ${{
          parameters.componentNames }}
        orgName: ${{
          parameters.githubOrganization }}
        sourceControl: github.com
        repoName: ${{
          parameters.repositoryName }}
        componentLifecycle: ${{
          parameters.componentLifecycle }}
        applicationType: ${{
          parameters.componentType }}
        owner: ${{
          parameters.componentOwner }}

  - id: publishPR
    name: Open PR with catalog-
      info.yaml
    action: publish:github:pull-
      request
    input:
      repoUrl: ${{
        parameters.githubHost }}/${{
        parameters.githubOrganization }}/${{
        parameters.repositoryName }}
      branchName: add-catalog-info
      title: add catalog-info.yaml
      description: |
        This pull request adds a
        Backstage entity metadata file to
        this repository.

    After this pull request is
    merged, the component will become
    available.

  For more information, read an
  overview of the Backstage software
  catalog at (https://backstage.io/docs/
  features/software-catalog/).

- id: register
  name: Register component
  action: catalog:register
  input:
    repoContentsUrl: https://${{
      parameters.githubHost }}/${{
      parameters.githubOrganization }}/${{
      parameters.repositoryName }}/blob/${{
      steps.publishPR.output.targetBranchName }}
```

```
apiVersion: scaffolder.backstage.io/
  v1beta3
kind: Template
metadata:
  name: register-component
  title: Register existing component to
  Software Catalog
  description: registers existing
  component to the Software Catalog
  tags:
    - short
    - catalog
    - register
spec:
  owner: janus
  system: janus
  type: service
  parameters:
    - title: Provide information about
      the GitHub repository
    required:
      - githubHost
      - githubOrganization
      - repositoryName
    properties:
      githubHost:
        title: GitHub hostname
        type: string
        description: Use github.com
        for GitHub Free, Pro, & Team or specify
        a hostname of your GitHub Enterprise
        instance.
      default: github.com
      githubOrganization:
        title: GitHub Organization
        type: string
        description: Repository name
      repositoryName:
        title: Repository name
        type: string
    - title: Provide information about
      the new component
    required:
      - componentOwner
      - componentType
      - componentLifecycle
    properties:
      componentNames:
        title: Component Name
        type: string
        description: Name of the
        created component. If leaved empty the
        name of the repository will be used.
      componentOwner:
        title: Owner
        description: Select an owner
        from the list or enter a reference to a
        Group or a User
      type:
        title: Type
        type: string
        description: 'The type of
        component. Well-known and common
        values: service, website, library.'
      default: other
```





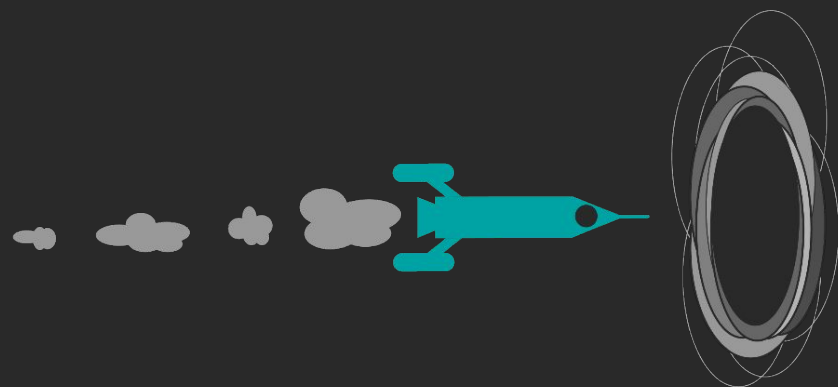
DEMO

</>

USEFUL INFO

GOLDEN PATHS

PITFALLS



### **The Galaxy Moves**

Templates need lifecycling

### **Culture Endures**

Templates are a user product

USEFUL INFO

CHANGE

RESISTANCE



*"You can't lay in front of the bulldozers forever."*

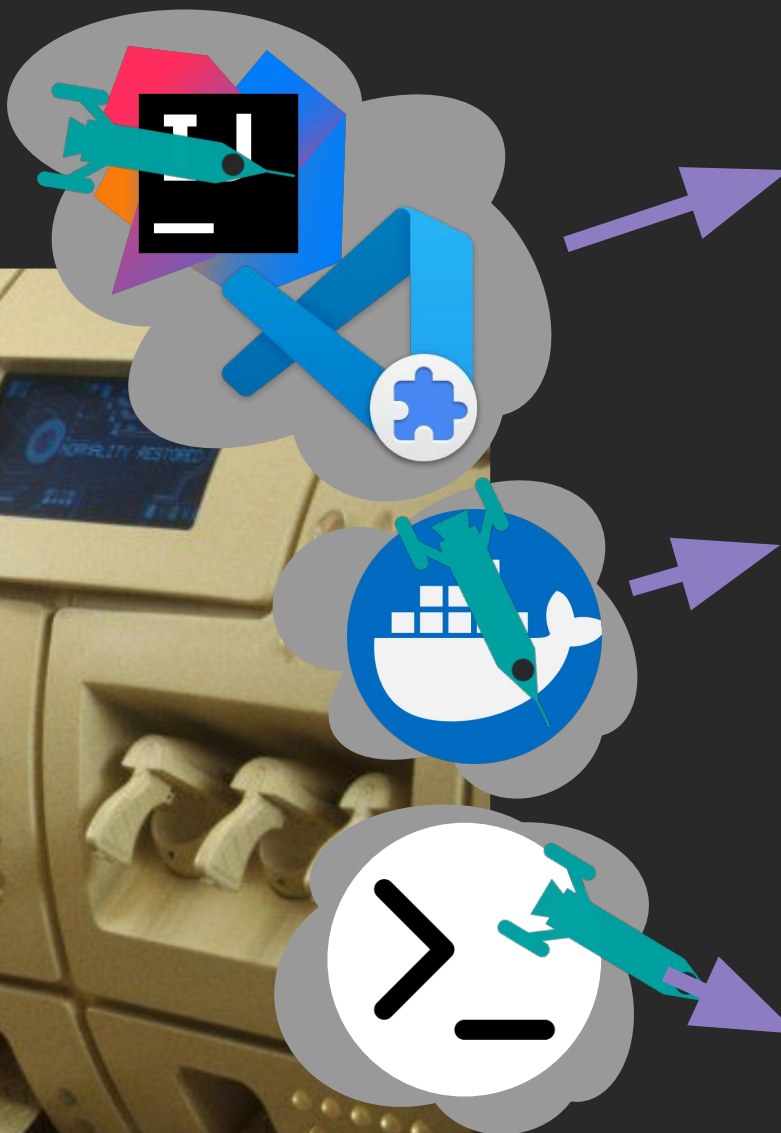
*"Oh we'll see who rusts first!"*



## USEFUL INFO

## GOLDEN PATHS

## HOW TO ?



```
schemaVersion: 2.3.0
metadata:
  generateName: terraform-dev
  version: 1.0.0
  description: A devfile for Terraform development and testing.
  projectType: Infrastructure-as-Code
  language: HCL
  provider: Open Source
  tags: ['Terraform', 'IaC']
  displayName: Terraform Development Environment
  icon: https://www.terraform.io/assets/images/og-image-8b3e4f7d.png
  attributes:
    .vscode/extensions.json: |
      {
        "recommendations": [
          "hashicorp.terraform"
        ]
      }
    .che/che-theia-plugins.yaml: |
      - id: hashicorp/terraform/latest

components:
- name: terraform-tools
  container:
    image: quay.io/pfeifferj/tfdev:latest
    memoryLimit: 512Mi
    cpuLimit: "1"
    mountSources: true
    volumeMounts:
      - name: terraform-data
        path: /workspace

- name: terraform-data
  volume:
    size: 1Gi
    ephemeral: true

commands:
- id: terraform-init
  exec:
    component: terraform-tools
    commandLine: "terraform init"
    workingDir: /workspace
    group:
      kind: build
      isDefault: true

- id: terraform-plan
  exec:
    component: terraform-tools
    commandLine: "terraform plan"
    workingDir: /workspace
    group:
      kind: build
      isDefault: false

- id: terraform-apply
  exec:
    component: terraform-tools
    commandLine: "terraform apply -auto-approve"
    workingDir: /workspace
    group:
      kind: build
      isDefault: false

- id: terraform-lint
  exec:
    component: terraform-tools
    commandLine: "tflint"
    workingDir: /workspace
    group:
      kind: build
      isDefault: false

- id: generate-docs
```

DEMO

</>





USEFUL INFO

GOLDEN PATHS

HOW TO ?

Data driven

User centric

WHO ?

Product Team

+ InnerSourcing

WHERE ?

Place of birth = IDE

Natural habitat = IDP / Repo

WHEN ?



USEFUL INFO

GOLDEN PATHS

EASY

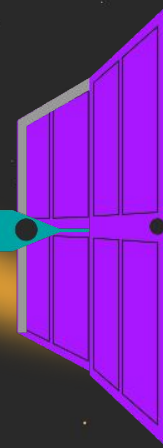
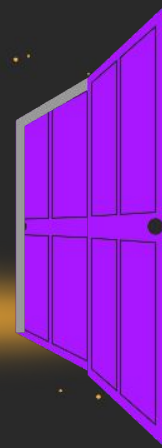
SECURE

FAST

Beware of

LCM

CULTURE



DON'T PANIC

QUESTIONS ?

42