Understanding Buildpacks

A Deep Dive Into Their Functionality

Patrick Baumgartner
42talents GmbH, Zürich, Switzerland

@patbaumgartner
patrick.baumgartner@42talents.com

Abstract

Understanding Buildpacks: A Deep Dive Into Their Functionality

Cloud Native Buildpacks make it easier for developers to create container images, offering a seamless experience with built-in features such as rebasing, reproducibility, and support for multiple entry points. In this talk, we begin by covering the basics of Buildpacks, looking at their architecture and the benefits they bring compared to traditional methods of building container images.

We'll then explore the lifecycle component, which is the central binary of Cloud Native Buildpacks. It shows how it manages the conversion of source code into OCI images and oversees the various stages of the build process, including detection, installation, and export.

We will also discuss the Paketo Buildpacks project, emphasizing its flexibility and community focus. This section will show how Paketo improves the developer experience by offering a curated selection of buildpacks that make configuration easier and enhance performance. Additionally, we'll share real-world examples of how organizations are using Buildpacks to streamline their CI/CD pipelines and speed up their deployment cycles. To wrap up, we'll go over some best practices for integrating Buildpacks into your workflows, providing tips on maximizing their features for better performance and security.

Understanding Buildpacks:

A Deep Dive Into Their Functionality

Patrick Baumgartner
42talents GmbH, Zürich, Switzerland

@patbaumgartner
patrick.baumgartner@42talents.com

Who am I?

Patrick Baumgartner

Technical Agile Coach @ **42talents**

My focus is on the **development of software solutions *with* humans**.

Coaching, Architecture, Development, Reviews, and Training.

Lecturer @ **Zurich University of Applied Sciences ZHAW**

Co-Organizer of **Voxxed Days Zurich**, **JUG Switzerland**, Java Champion, Oracle ACE Pro Java ...

@patbaumgartner

Agenda

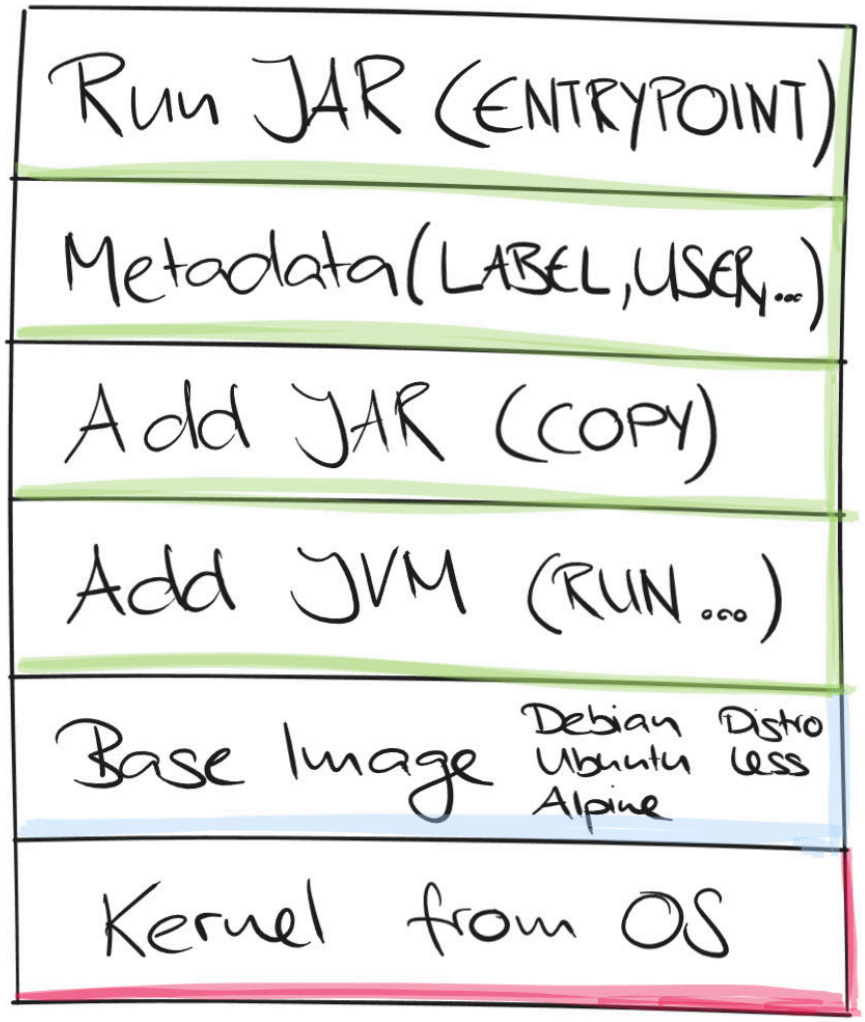
Agenda

1. What Are Containers, Really?
2. What Are Cloud Native Buildpacks?
3. Comparing Buildpacks to Dockerfiles
4. Buildpacks CLI and CA Certificate Integration
5. Software Supply Chain: SBOM and Observability
6. Advanced Java Buildpack Features
7. Customizing and Reusing Buildpacks
8. Paketo Buildpacks Overview
9. Integrating Buildpacks into CI/CD Pipelines

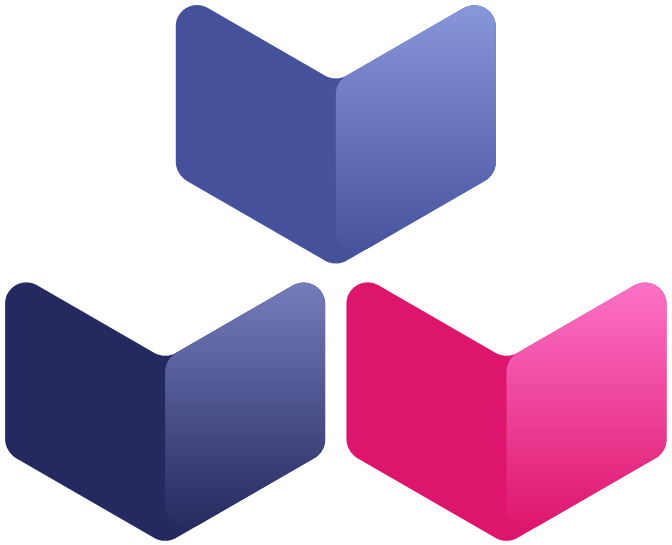
What Are Containers, Really?

What Are Containers, Really?

- Isolated processes using Linux primitives: `chroot`, `namespaces`, and `cgroups`
- Provide filesystem, resource, and process isolation
- Lightweight, fast to start, and portable
- Run on any host with a container runtime like Docker or containerd



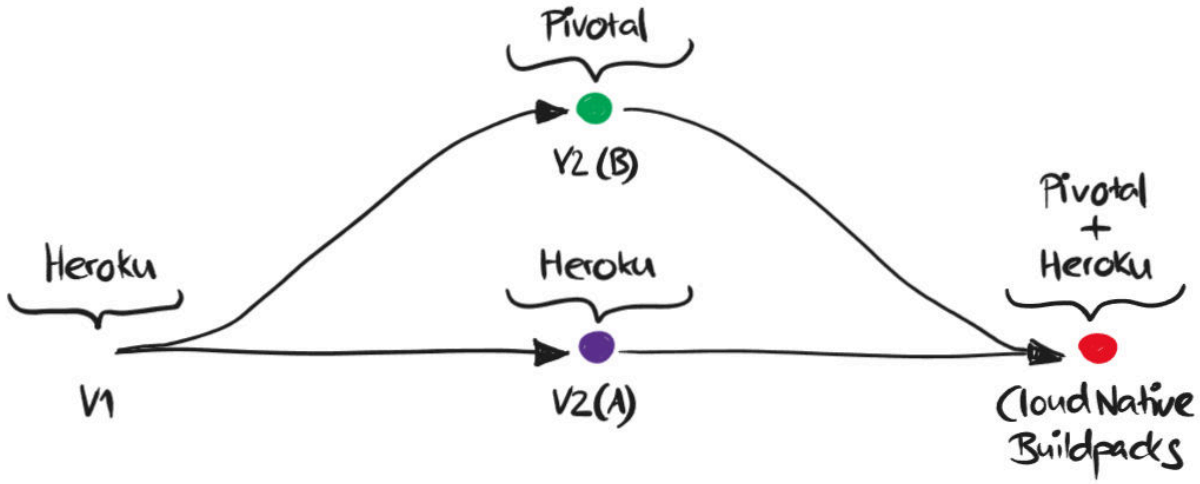
What Are Cloud Native Buildpacks?



Buildpacks.io

What Are Cloud Native Buildpacks?

- Convert source code into OCI-compliant images.
- Automatically detect app type and required dependencies.
- No Dockerfile needed.
- Originated at Heroku, now under CNCF.
- Co-developed by Pivotal/VMware and Heroku.



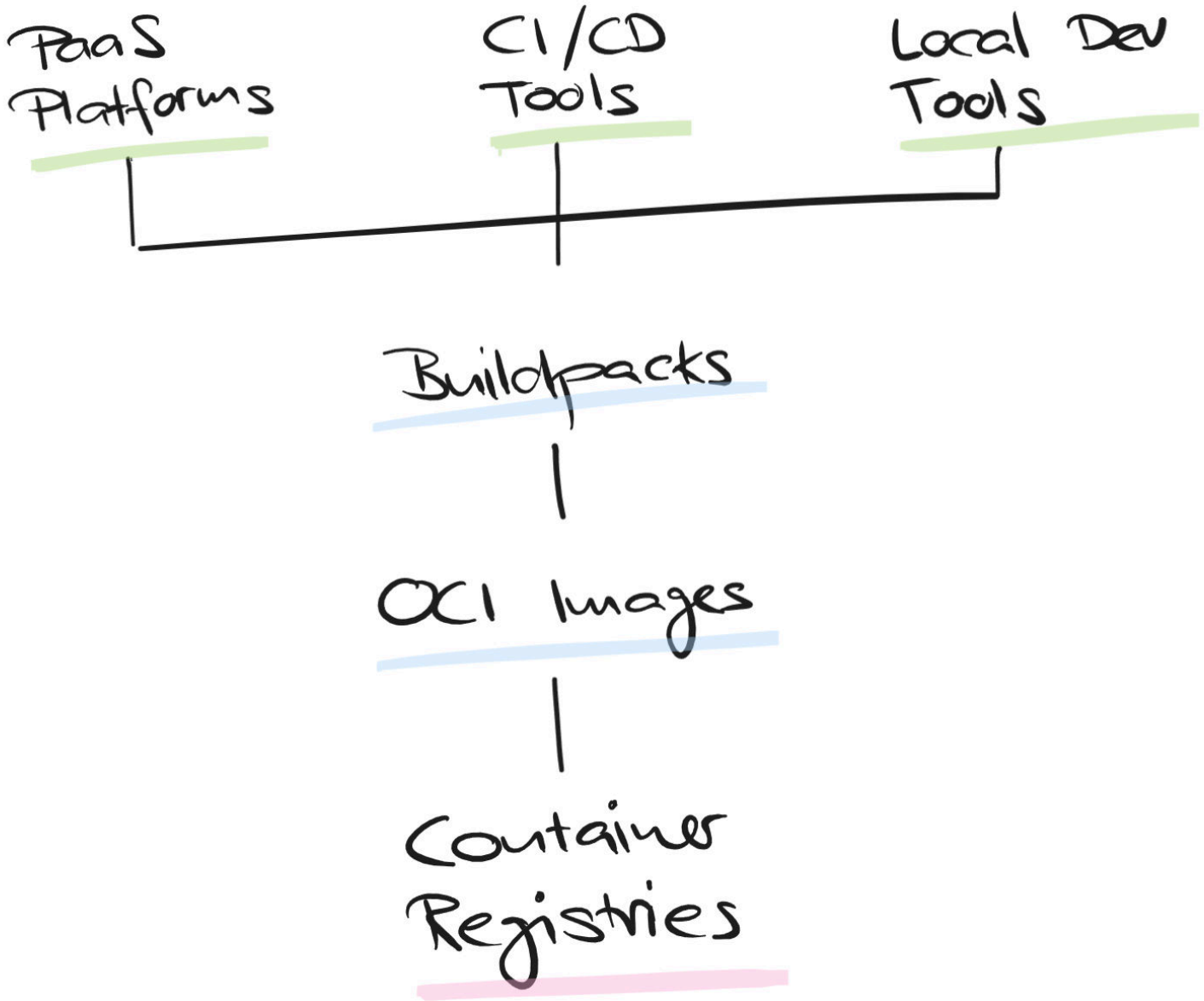
paketo

buildpacks

Your app,
in your favorite language,
ready to run in the cloud

Why Cloud Native Buildpacks Are Relevant Today

- Simplifies building container images without Dockerfiles
- Ensures consistent builds across teams and environments
- Enhances security via curated and updatable buildpacks
- Accelerates CI/CD with reproducible builds
- Integrates smoothly into modern DevOps workflows


PaaS
Platforms
CI/CD
Tools
Local Dev
Tools
Buildpacks
OCI Images
Container
Registries

Widespread Ecosystem Adoption

- **PaaS Platforms:** Heroku, Fly.io, Cloud Foundry, Google Cloud Run, DigitalOcean, Azure Container Apps, Epinio, etc.
- **CI/CD Tools:** GitHub Actions, CircleCI, GitLab, Jenkins, Tekton, etc.
- **Local Dev Tools:** Hashicorp Waypoint, Spring Boot, Quarkus, Pack CLI, Maven/Gradle, kpack, etc.
- **Container Registries:** Docker Hub, GCR, ECR, etc. (show output as OCI image)

> From source code to production-ready container image, standardized and Dockerfile-free. Buildpacks automate and unify application delivery.

Cloud Native Buildpacks

475 followers https://buildpacks.io

Pinned

pack Public
CLI for building apps using Cloud Native Buildpacks
Go 2.7k 303

spec Public
Specification for Cloud Native Buildpacks
Shell 259 71

lifecycle Public
Reference implementation of the Cloud Native Buildpacks lifecycle
Go 194 115

rfcs Public
RFCs for Cloud Native Buildpacks
Shell 58 74

community Public
Community content for the Cloud Native Buildpacks (CNB) project
47 24

samples Public
Samples for Cloud Native Buildpacks
Shell 208 151

Why Choose Buildpacks Instead of Dockerfiles?

The Limitations of Dockerfiles

```
FROM openjdk

COPY target/*.jar /app/app-runner.jar

WORKDIR /app

EXPOSE 8080

ENTRYPOINT [ "java","-jar" ]

CMD ["app-runner.jar"]
```

```
java -jar app-runner.jar
```

See: https://snyk.io/blog/10-best-practices-to-containerize-nodejs-web-applications-with-docker/

The Buildpacks CLI: Simplicity in Action

```
pack builder suggest
```

Google Cloud Buildpacks:

```
pack build my-app --builder gcr.io/buildpacks/builder
```

Paketo Buildpacks:

```
pack build my-app --builder paketobuildpacks/builder-jammy-base # Ubuntu 20.04 LTS

# Alternatively

pack build my-app --builder paketobuildpacks/builder-noble-java-tiny # Ubuntu 22.04 LTS
```

```
Calculating JVM memory based on 25843068K available memory
For more information on this calculation, see https://paketo.io/docs/reference/java-reference/#memory-calculator
Calculated JVM Memory Configuration: -XX:MaxDirectMemorySize=10M -Xmx25251143K -XX:MaxMetaspaceSize=79924K -XX:ReservedCodeCacheSize=240M
  -Xss1M (Total Memory: 25843068K, Thread Count: 250, Loaded Class Count: 11697, Headroom: 0%)
Enabling Java Native Memory Tracking
Adding 146 container CA certificates to JVM truststore
Spring Cloud Bindings Enabled
Picked up JAVA_TOOL_OPTIONS: -Djava.security.properties=/layers/paketo-buildpacks_bellsoft-liberica/java-security-properties/java-security.properties
  -XX:+ExitOnOutOfMemoryError -XX:MaxDirectMemorySize=10M -Xmx25251143K -XX:MaxMetaspaceSize=79924K -XX:ReservedCodeCacheSize=240M -Xss1M -XX:+UnlockDiagnosticVMOptions
  -XX:NativeMemoryTracking=summary -XX:+PrintNMTStatistics -Dorg.springframework.cloud.bindings.boot.enable=true


  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v3.4.5)

2025-05-03T12:07:31.047Z  INFO 1 --- [demo] [           main] com.example.demo.DemoApplication        : Starting DemoApplication v0.0.1-SNAPSHOT using Java 21.0.7 with PID 1
  (/workspace/BOOT-INF/classes started by cnb in /workspace)
2025-05-03T12:07:31.049Z  INFO 1 --- [demo] [           main] com.example.demo.DemoApplication        : No active profile set, falling back to 1 default profile: "default"
2025-05-03T12:07:31.735Z  INFO 1 --- [demo] [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port 8080 (http)
2025-05-03T12:07:31.748Z  INFO 1 --- [demo] [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2025-05-03T12:07:31.748Z  INFO 1 --- [demo] [           main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10.1.40]
2025-05-03T12:07:31.776Z  INFO 1 --- [demo] [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2025-05-03T12:07:31.778Z  INFO 1 --- [demo] [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 690 ms
2025-05-03T12:07:32.066Z  INFO 1 --- [demo] [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) with context path '/'
2025-05-03T12:07:32.078Z  INFO 1 --- [demo] [           main] com.example.demo.DemoApplication        : Started DemoApplication in 1.379 seconds (process running for 1.638)
^C2025-05-03T12:07:39.856Z  INFO 1 --- [demo] [ionShutdownHook] o.s.b.w.e.tomcat.GracefulShutdown       : Commencing graceful shutdown. Waiting for active requests to complete
2025-05-03T12:07:39.860Z  INFO 1 --- [demo] [tomcat-shutdown] o.s.b.w.e.tomcat.GracefulShutdown       : Graceful shutdown complete

Native Memory Tracking:

Total: reserved=26897799346, committed=260511922
       malloc: 49469618 #118067
       mmap:   reserved=26848329728, committed=211042304

-                 Java Heap (reserved=25870467072, committed=117440512)
                            (mmap: reserved=25870467072, committed=117440512, peak=486539264)
```

Adding CA Certificates with Dockerfiles

```
FROM openjdk

COPY custom-ca.crt /app
RUN keytool -importcert -file /app/custom-ca.crt
COPY target/*.jar /app/app-runner.jar

WORKDIR /app

EXPOSE 8080

ENTRYPOINT [ "java","-jar" ]

CMD ["app-runner.jar" ]
```

Adding CA Certificates with Buildpacks

```
mkdir -p bindings/ca-certificates/certs
echo "ca-certificates" > bindings/ca-certificates/type
cp my-cert.crt bindings/ca-certificates/my-cert.crt
```

```
bindings/
└── ca-certificates/
    ├── type          # file contains: ca-certificates
    └── my-cert.crt
```

```
pack build my-app \
  --builder paketobuildpacks/builder-jammy-base \
  --volume "$(pwd)/bindings:/platform/bindings"
```

```
BellSoft Liberica JDK 21.0.7: Contributing to layer
  Downloading from https://github.com/bell-sw/Liberica/releases/download/21.0.7+9/bellsoft-jdk21.0.7+9-linux-amd64.tar.gz
  Verifying checksum
  Expanding to /layers/paketo-buildpacks_bellsoft-liberica/jdk
  Adding 147 container CA certificates to JVM truststore
```

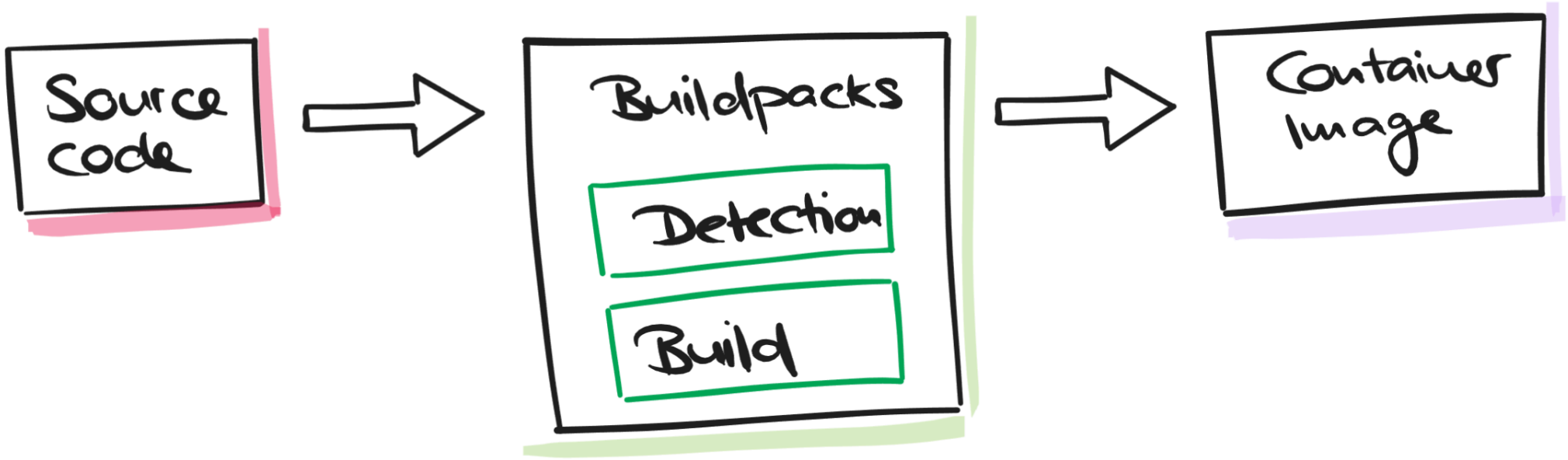
> Imagine if building your Go, Node.js, and Python apps with buildpacks were that simple.

The Basics of Cloud Native Buildpacks

Buildpacks Architecture

- **BuildImage and RunImage**: Base OS (e.g. Ubuntu) -> Stack (deprecated)
- **Buildpacks:** Language/runtime logic (Node.js, Java, etc.)
- **Lifecycle:** Orchestrates the build
- **Builder:** All of the above in one image

How Buildpacks Work



- **buildpack.toml:** Provides metadata for buildpacks
- **Detect:** Identify the app type and required dependencies
- **Build:** Install dependencies and create the image

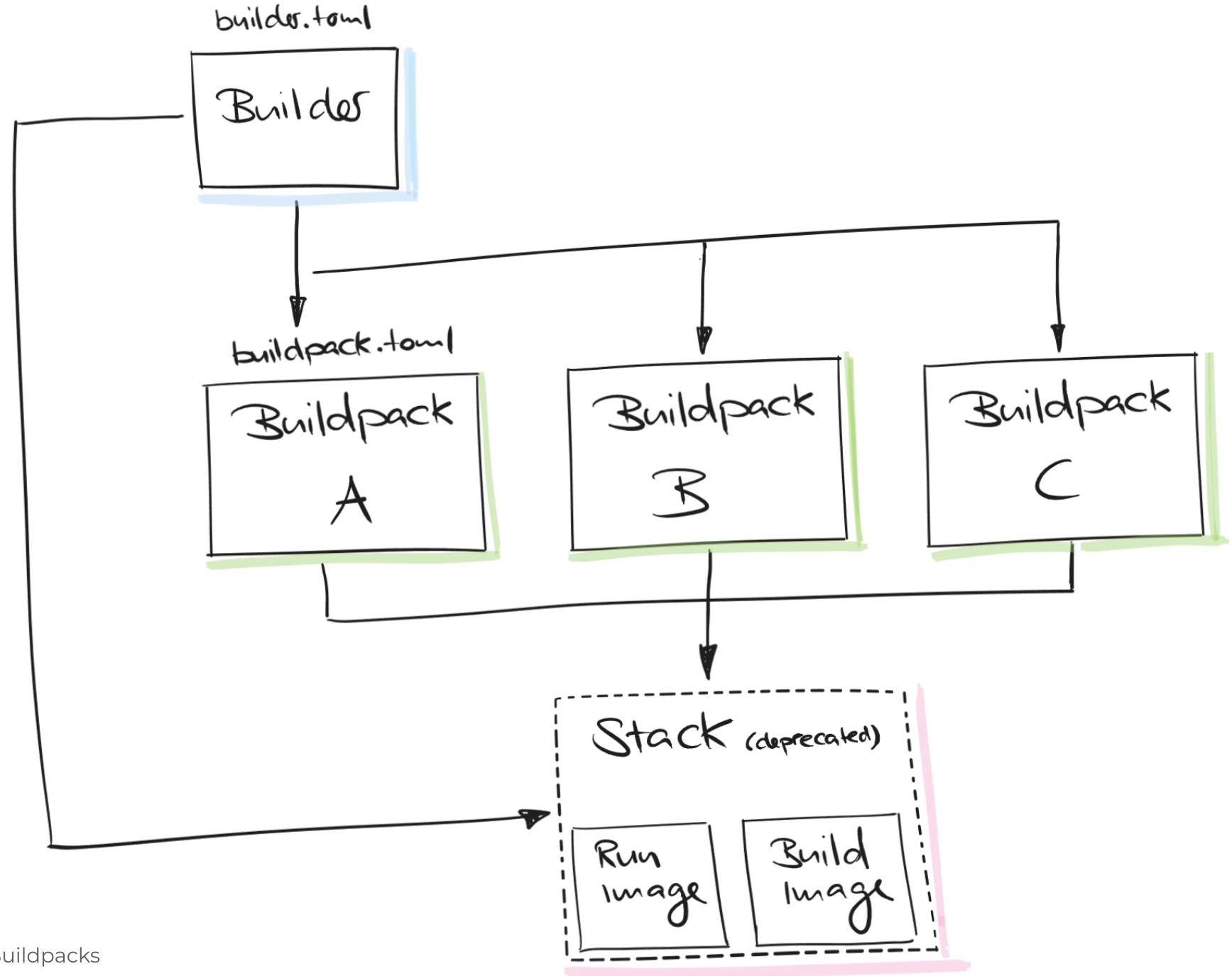
Anatomy of a Buildpack

- `bin/detect`
  - Runs against source to determine if the buildpack is applicable
  - Example:
    - A Java CNB might look for `.java` files or a `.jar`
    - A Node.js CNB might look for `package.json`
- `bin/build`
  - Downloads build-time and run-time dependencies
  - Example:
    - A Java CNB might download a JDK or JRE
    - Compiles sources (if needed)
    - Sets launch command and environment variables

Lifecycle Phases (Create)

The buildpack lifecycle is split into phases. Each plays a critical role in image creation.

- **Analyze:** Provides metadata from previous builds to buildpacks.
- **Detect:** Selects compatible buildpacks and creates a build plan.
- **Restore:** Retrieves cached data from earlier builds.
- **Build:** Rebuilds only the layers that need updating.
- **Export:** Replaces outdated layers with the newly built ones.


builder.toml
Builder
buildpack.toml
Buildpack A
Buildpack B
Buildpack C
Stack (deprecated)
Run Image
Build Image

Customizing and Reusing Buildpacks

- **Why customize?**
  - Select a specific JVM vendor or version (e.g. Adoptium, BellSoft)
  - Add CA certificates for internal services
  - Configure Maven or Gradle to use internal repositories or proxies
  - Extend functionality with custom environment variables or bindings
- **Keep your image updated:**
  - Rebase your image with the latest run image (no rebuild needed)
  - `pack rebase my-app`
- **Browse available buildpacks:** https://registry.buildpacks.io

paketo

buildpacks

Paketo Buildpacks Overview

- **Available in 3 variants:**
  - **Base** – Core buildpacks for major languages
  - **Tiny** – Minimal footprint, ideal for small apps or functions
  - **Full** – Includes all Paketo-supported buildpacks and tools
- **Broad language support:**
  - Java, Node.js, Python, Go, .NET, PHP, Ruby, Rust, and more
- **Rich feature set:**
  - Support for custom CA certificates
  - SBOM generation using Syft
  - OCI image annotations and additional labels
  - Arm64 and Red Hat UBI compatibility

Software Bill of Materials (SBOM) Support

```
pack sbom download my-app --output-dir ./my-app-sbom
```

```
cat ./my-app-sbom/layers/sbom/launch/paketo-buildpacks_executable-jar/sbom.cdx.json | \
  jq '.components[] | select(.name=="jackson-databind")'
```

```
{
  "bom-ref": "pkg:maven/com.fasterxml.jackson.core/jackson-databind@2.18.3?package-id=d6932672ba26a871",
  "cpe": "cpe:2.3:a:com.fasterxml.jackson.core.jackson-databind:jackson-databind:2.18.3:*:*:*:*:*:*:*",
  "externalReferences": [
    {
      "hashes": [
        {
          "alg": "SHA-1",
          "content": "537e3886263e3b3464385040453e92567fd509e2"
        }
      ],
      "type": "build-meta",
      "url": ""
...
```

Observability Integration with Datadog

- At build time, the Datadog agent is added via Paketo Java buildpack.
- A dedicated layer is created with `dd-java-agent.jar` , and `$JAVA_TOOL_OPTIONS` is auto-configured.

```
pack build my-app -e BP_DATADOG_ENABLED=true ...
```

- At runtime, observability is activated via environment variables

```
docker run my-app -e BPL_JMX_ENABLED=true \
 -e BPL_JMX_PORT=8000 \
 -e DD_ENV=prod \
 -e DD_SITE=<DATADOG_SITE> \
 -e DD_API_KEY=<DATADOG_API_KEY>
```

Advanced Features in Java Buildpacks

- Set JVM version using `BP_JVM_VERSION`
- Easy Java vendor selection using buildpack order

```
<buildpacks>
  <buildpack>gcr.io/paketo-buildpacks/adoptium</buildpack>
  <buildpack>urn:cnb:builder:paketo-buildpacks/java</buildpack>
</buildpacks>
```

- Optimizations for Spring Boot (AOT, CDS, Cloud Bindings) and memory management (memory calculator)
- Jlink and JMX support
- Native image support with GraalVM


Demo

Challenges & Solutions

Patching the Base Image

- Base images do not contain fonts 😔

```
FROM paketobuildpacks/run-jammy-base:latest

USER root

## Idea from: https://blog.adoptopenjdk.net/2021/01/prerequisites-for-font-support-in-adoptopenjdk/
RUN apt-get update \
  && apt-get install -y --no-install-recommends \
  wget \
  fontconfig \
  && rm -rf /var/lib/apt/lists/*

# Install Noto Emoji Manually
RUN mkdir -p /usr/share/fonts/truetype/noto \
  && cd /usr/share/fonts/truetype/noto \
  && wget https://github.com/googlefonts/noto-emoji/raw/9a5261d871451f9b5183c93483cbd68ed916b1e9/fonts/NotoEmoji-Regular.ttf \
  && fc-cache -fv

USER cnb
```

When to Create a Custom Buildpack

- Custom JDK Version & JVM Tuning
- Application Configuration Injection
- Metrics/Monitoring Integration
- License Compliance
- Logging Configuration

```
pack create-buildpack \
  --path ./my-custom-buildpack \
  --buildpack my-custom-buildpack \
  --config ./buildpack.toml
```

```
pack build my-app \
  --buildpack my-custom-buildpack \
  --builder paketobuildpacks/builder-jammy-base
```

Custom Builder

- Using a custom builder for a custom base-image and run-image (stack)
- Allows a (pre-)selection of buildpacks to be used

```
<plugin>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-maven-plugin</artifactId>
  <configuration>
    <image>
      <builder>patbaumgartner/buildpack-builder:noble</builder>
    </image>
  </configuration>
</plugin>
```

Custom Java Buildpacks

```
<plugin>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-maven-plugin</artifactId>
  <configuration>
    <image>
      <buildpacks>
        <!-- <buildpack>gcr.io/paketo-buildpacks/adoptium:latest</buildpack> -->
        <!-- <buildpack>gcr.io/paketo-buildpacks/alibaba-dragonwell:latest</buildpack> -->
        <!-- <buildpack>gcr.io/paketo-buildpacks/amazon-corretto:latest</buildpack> -->
        <buildpack>gcr.io/paketo-buildpacks/azul-zulu:latest</buildpack>
        <!-- <buildpack>gcr.io/paketo-buildpacks/bellsoft-liberica:latest</buildpack> -->
        <!-- <buildpack>gcr.io/paketo-buildpacks/eclipse-openj9:latest</buildpack> -->
        <!-- <buildpack>gcr.io/paketo-buildpacks/graalvm:latest</buildpack> -->
        <!-- <buildpack>gcr.io/paketo-buildpacks/sap-machine:latest</buildpack> -->
        <!-- <buildpack>gcr.io/paketo-buildpacks/oracle:latest</buildpack> -->
        <!-- Used to inherit all the other buildpacks -->
        <buildpack>gcr.io/paketo-buildpacks/java:latest</buildpack>
      </buildpacks>
    </image>
  </configuration>
</plugin>
```

JLink

```
<plugin>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-maven-plugin</artifactId>
  <configuration>
    <image>
      <buildpacks>
        <buildpack>gcr.io/paketo-buildpacks/java:latest</buildpack>
      </buildpacks>
      <env>
        <BP_JVM_JLINK_ENABLED>true</BP_JVM_JLINK_ENABLED>
        <BP_JVM_JLINK_ARGS>--add-modules
          java.base,java.compiler,java.desktop,java.instrument,java.net.http,java.prefs,java.rmi,
          java.scripting,java.security.jgss,java.sql.rowset,jdk.jfr,jdk.management,jdk.net,jdk.unsupported
          --compress=zip-6 --no-header-files --no-man-pages --strip-debug</BP_JVM_JLINK_ARGS>
      </env>
    </image>
  </configuration>
</plugin>
```

AOT and App CDS

```
<plugin>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-maven-plugin</artifactId>
  <configuration>
    <image>
      <buildpacks>
        <buildpack>gcr.io/paketo-buildpacks/java:latest</buildpack>
      </buildpacks>
      <env>
        <BP_JVM_CDS_ENABLED>true</BP_JVM_CDS_ENABLED>
        <BP_SPRING_AOT_ENABLED>true</BP_SPRING_AOT_ENABLED>
      </env>
    </image>
  </configuration>
  <executions>
    <execution>
      <id>process-aot</id>
      <goals>
        <goal>process-aot</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

Integrating Buildpacks into CI/CD Pipelines

- **Choose the Right Buildpacks**
  - Use only officially maintained and security-audited buildpacks.
- **Automate Security Scans**
  - Integrate static and dynamic security scans immediately after the build process.
- **Optimize Caching and Layer Reuse**
  - Enable buildpack-specific caching to avoid redundant steps.
- **Ensure Versioning and Traceability**
  - Explicitly document and version the buildpacks used in your CI configuration.
- **Integrate Observability and Metrics**
  - Continuously monitor build durations, error rates, and artifact sizes.

Summary and Key Takeaways

Why use Cloud Native Buildpacks?

- No Dockerfile needed
- Focus on providing value to the end users
- Composable (Node.js, Bash, Java, Python, ...)
- Build any app with the same set of buildpacks
- Only rebuild layers that have changed
- Rebasing without rebuilding (put your app on a new base layer)


Q&A


I ❤ Feedback 🙏

Understanding Buildpacks:

A Deep Dive Into Their Functionality

Patrick Baumgartner
42talents GmbH, Zürich, Switzerland

@patbaumgartner
patrick.baumgartner@42talents.com

bitly

	Cloud Native Buildpacks	Dockerfile	source-to-image (s2i)	Jib	ko
Advanced Caching	Yes	No	Yes	No	Yes
Auto-detection	Yes	No	Yes	Yes	Yes
Bill-of-Materials	Yes	No	No	No	Yes
Modular / Pluggable	Yes	No	No	N/A †	N/A †
Multi-language	Yes	Yes	Yes	No	No
Multi-process	Yes	No	No	No	No
Minimal app image	Yes	Yes ♦	Yes ‡	Yes	Yes
Rebasing	Yes	No	No	No	No
Reproducibility	Yes	No	No	Yes	Yes
Reusability	Yes	No	Yes	N/A †	N/A †

Resources

Official Documentation and Tools

- Cloud Native Buildpacks (CNCF)
- Paketo Buildpacks
- Buildpacks Registry
- Spring Boot Buildpacks Integration
- Pack CLI Docs
- GraalVM Native Image Support
- Syft – SBOM Generation
- Snyk Docker Best Practices
- Datadog Java Integration